手册页第 5 部分：标准、环境和宏

文件号码 E37322
2012 年 10 月

121209@25097

目录

前言

无论是初次使用 SunOS 操作系统的用户还是熟悉该操作系统的用户，均可通过联机手册页获取与系统及其功能有关的信息。手册页用于简要回答关于用途/用法的问题。手册页通常是参考手册的组成部分，并不用作教程。

概述

以下内容包含对手册页各部分及其所引用信息的简要说明：

- 第 1 部分按照字母顺序介绍了操作系统中提供的各种命令。
- 第 1M 部分按照字母顺序介绍了主要用于系统维护和管理的各种命令。
- 第 2 部分介绍了所有的系统调用。其中的大多数调用可能返回一个或多个错误。如果返回一个异常值，则表明有错误情况。
- 第 3 部分介绍了各种库中包含的函数，这些函数不属于第 2 部分介绍的那些直接调用 UNIX 系统原语 (primitive) 的函数。
- 第 4 部分简要介绍了各种文件的格式，并在适当之处给出了文件格式的 C 结构声明。
- 第 5 部分包含其他文档，如字符集表。
- 第 7 部分介绍了涉及特定硬件外围设备和设备驱动程序的各种特殊文件。还介绍了 STREAMS 软件驱动程序、模块和 STREAMS 通用的一组系统调用。
- 第 9E 部分介绍了 DDI（Device Driver Interface，设备驱动程序接口）/DKI（Driver/Kernel Interface，驱动程序/内核接口）、仅 DDI 和仅 DKI 入口点例程，开发者可以将这些例程包含在设备驱动程序中。
- 第 9F 部分介绍了可供设备驱动程序使用的内核函数。
- 第 9S 部分介绍了驱动程序用来在驱动程序和内核之间共享信息的数据结构。

下面是手册页的通用格式。每个手册的手册页部分通常遵循该顺序，但只包括需要的标题。例如，如果未报告任何已知问题，则不包括“已知问题”部分。有关每一部分的更多详细信息，请参见 `intro` 页；有关手册页的更多一般信息，请参见 man(1)。

名称　　本部分提供了记录的命令或函数的名称，后跟其用途的简要说明。

用法概要

本部分说明了命令或函数的语法。如果命令或文件不存在于标准路径中，则显示其全路径名。除非要求使用不同的参数顺序，否则选项和参数均按字母顺序排列，首先是单个字母的参数，接下来是带有参数的选项。

本部分使用以下特殊字符：

[ ] 方括号。括在方括号中的选项或参数是可选的。如果没有方括号，则必须指定参数。

. . . 省略号。可以为该符号前面的参数提供多个值，或者可以多次指定该参数，例如“`filename`...”。

| 分隔符。一次只能指定一个由该字符分隔的参数。

{ } 大括号。括在大括号内的选项和/或参数是相互依赖的，因此必须将大括号中的所有内容视为一个单元。

协议

本部分仅出现在第 3R 子部分，用于指示协议说明文件。

描述

本部分定义了服务的功能和行为。因此，它简明地介绍了命令执行哪些操作。它不讨论“选项”或引用“示例”。在“用法”下介绍了交互式命令、子命令、请求、宏和函数。

IOCtl

本部分仅出现在第 7 部分的手册页中。只有为 `ioctl`(2) 系统调用提供了适当参数的设备类才被称为 `ioctl`，并生成自己的标题。特定设备的 `ioctl` 调用按字母顺序显示在该特定设备的手册页中。`ioctl` 调用用于特殊类别的设备。所有这些调用都以 io 结尾，例如 `mtio`(7I)。

选项

本部分列出了各命令选项及每个选项用途的简明摘要。逐个列出各个选项，并以它们在“用法概要”部分显示的顺序排列。在选项下讨论各个选项可能的参数，还提供缺省值（如果适用）。

操作数

本部分列出了命令操作数，并介绍它们对命令操作的影响。

输出

本部分介绍了命令所生成的输出（标准输出、标准错误或输出文件）。

返回值

如果手册页记录返回值的函数，则本部分列出这些值并介绍返回这些值应满足的条件。如果函数只能返回常量值（例如 0 或 –1），则将在标记的段落中列出这些值。否则，会有单个段落介绍每个函数的返回值。声明为 void 的函数不返回值，因此不会在“返回值”中讨论这些函数。

错误

对于故障，大多数函数将指出它们出现故障的原因的错误代码置于全局变量 `errno` 中。本部分按字母顺序列出了函数可以生成的所有错误代码，并介绍了导致每个错误的条件。如果多个条件可以导致同一错误，则在错误代码下以单独的段落介绍每个条件。

用法

本部分列出了需要详细说明的特殊规则、功能和命令。此处列出的子部分用于说明内置功能：

命令

修饰符
变量
表达式
输入语法

示例 本部分提供了用法的示例，或者如何使用命令或函数的示例。会尽可能显示包括命令行输入和计算机响应的完整示例。只要给出了示例，就会显示 `example%` 提示，如果用户必须为超级用户，则提示显示为 `example#`。示例后面跟有说明、变量替换规则或返回值。大部分示例说明了“用法概要”、“说明”、“选项”和“用法”部分的概念。

环境变量 本部分列出了命令或函数影响的所有环境变量，其后附加了关于影响的简要说明。

退出状态 本部分列出了命令返回到调用程序或 shell 中的值以及导致返回这些值的条件。通常，返回零表示成功完成，返回非零值表示各种错误条件。

文件 本部分列出了手册页引用的所有文件名称、相关文件以及命令创建或所需的文件。每个文件名称后面都具有描述性摘要或说明。

属性 本部分通过定义属性类型及其相应的值列出了命令、实用程序和设备驱动程序的特征。有关更多信息，请参见 `attributes(5)`。

另请参见 本部分列出了对其他手册页、内部文档和外部出版物的引用。

诊断 本部分列出了诊断消息以及导致错误的条件的简要说明。

警告 本部分列出了有关特殊条件的警告，这些条件可能会严重影响您的工作状况。此部分不是诊断列表。

附注 本部分列出了不属于页面任何部分的其他信息。它采用对用户旁白提示的形式，包含用户特别关注的要点。此处不包含关键信息。

已知问题 本部分介绍了已知问题，并尽可能给出解决方法。

参考文档

简介

引用名

Intro – introduction to miscellany

描述

Among the topics presented in this section are:

Standards — The POSIX (IEEE) Standards and the X/Open Specifications are described on the `standards` page.

Environments — The user environment (`environ`), the subset of the user environment that depends on language and cultural conventions (`locale`), the large file compilation environment (`lfcompile`), and the transitional compilation environment (`lfcompile64`) are described.

Macros — The macros to format Reference Manual pages (`man` and `mansun`) as well as other text format macros (`me`, `mm`, and `ms`) are described.

Characters — Tables of character sets (`ascii`, `charmap`, `eqnchar`, and `iconv`), file format notation (`formats`), file name pattern matching (`fnmatch`), and regular expressions (`regex` and `regexp`) are presented.

Acknowledgments

Oracle America, Inc. gratefully acknowledges The Open Group for permission to reproduce portions of its copyrighted documentation. Original documentation from The Open Group can be obtained online at `http://www.opengroup.org/bookstore/`.

The Institute of Electrical and Electronics Engineers and The Open Group, have given us permission to reprint portions of their documentation.

In the following statement, the phrase “this text" refers to portions of the system documentation.

Portions of this text are reprinted and reproduced in electronic form in the SunOS Reference Manual, from IEEE Std 1003.1, 2004 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright (C) 2001-2004 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between these versions and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at `http://www.opengroup.org/unix/online.html`.

This notice shall appear on any product containing this material.

参考文档

Standards, Environments, and Macros

引用名

acl – 访问控制列表

描述

访问控制列表 (Access control list, ACL) 是自主访问控制机制，可授予和拒绝对文件和目录的访问权限。Solaris 发行版本中支持两种不同的 ACL 模型：POSIX 样式的 ACL 和 NFSv4 ACL。

UFS 文件系统支持较早的 POSIX 样式的模型。该模型基于已撤销的从未进行标准化的 ACL POSIX 规范。POSIX 委员会后来撤销了该模型。

另一个模型基于 NFSv4 工作组的标准，是 Internet 工程任务组 (Internet Engineering Task Force, IETF) 批准的标准。ZFS 文件系统使用 NFSv4 模型，提供比 POSIX 样式的模型更丰富的语义和更精细的权限功能。

POSIX 样式的 ACL

POSIX 样式的 ACL 为 Solaris 发行版中的基本 UNIX 文件权限提供备用安全机制。其目的是进一步限制对文件和目录的访问或扩展特定用户的权限。ACL 可用于更改标准所有者、组和文件模式的其他类位的权限。ACL 可为其他用户和组授予对文件的访问权限。目录还可以有一种称为**缺省** ACL 的特殊 ACL，用于定义将由目录后代继承的 ACL 条目。POSIX 样式的 ACL 有一个称为**掩码**的 ACL 条目。该掩码定义可向其他用户和组条目授予的最大权限。每当创建文件或通过 chmod(1) 或 chmod(2) 更改其模式时，就会重新计算掩码。该掩码将重新计算为传递到 chmod(2) 的模式中定义的组权限。

POSIX 样式的 ACL 模型使用传统 UNIX 权限的标准 rwx 模型。

ACL 的表示方式如下：

acl_entry[,*acl_entry*]...

每个 *acl_entry* 都包含一个 ACL 条目。ACL 条目由两个或三个冒号分隔的 (:) 字段表示。

user:[*uid*]:*perms*	如果 *uid* 为空，则它表示文件所有者。
group:[*gid*]:*perms*	如果 *gid* 为空，则它表示所属组。
other:*perms*	表示文件的其他类。
mask:*perms*	定义要分发的**最大**权限。

例如，要为用户授予 joe 读写权限，则 ACL 条目应指定为：

```
user:joe:rw-
```

NFSv4 ACL

NFSv4 ACL 总体上基于 Windows NT ACL 模型。NFSv4 ACL 提供比 POSIX 样式的 ACL 更丰富的 ACL 模型。

NFSv4 和 POSIX 样式的 ACL 之间的主要区别如下：

- NFSv4 ACL 提供比 rwx 模型更精细的权限。
- NFSv4 ACL 允许同时有 ALLOW 和 DENY 条目。

- NFSv4 ACL 提供一组丰富的继承语义。POSIX ACL 也具有继承性，但使用 NFSv4 模型，您可以控制以下继承功能：
  - 继承同时级联到文件和目录还是仅级联到文件和目录之一。
  - 对于目录，您可以指示将继承应用于目录本身，应用于一个级别的子目录或级联到目录的所有子目录。
- NFSv4 ACL 提供用于挂钩到系统的审计迹的机制。目前，Solaris 不支持此机制。
- NFSv4 ACL 使管理员能够指定检查 ACL 条目的顺序。使用 POSIX 样式的 ACL 时，文件系统会按照明确定义且访问控制严格的检查顺序对 ACL 条目进行重新排序。

POSIX 样式的 ACL 语义可通过 NFSv4 ACL 来实现。但是，只有一些 NFSv4 ACL 可以转换为等效的 POSIX 样式的 ACL。

可采用三种不同 chmod ACL 格式指定权限：详细、紧凑或位置。详细格式使用字词指示用正斜杠 (/) 字符分隔权限。紧凑格式使用权限字母标识权限，位置格式使用权限字母标识权限，或使用连字符 (-) 标识没有任何权限。

详细模式的权限以及其括号中的缩写形式（紧凑和位置模式）如下所述：

read_data (r)	读取文件数据的权限
list_directory (r)	列出目录内容的权限。
write_data (w)	在文件的偏移范围中的任意位置修改文件数据的权限。这包括增加文件大小或写入任意偏移的能力。
add_file (w)	向目录中添加新文件的权限。
append_data (p)	修改文件数据的能力，但仅从 EOF 开始。当前不支持此权限。
add_subdirectory (p)	在目录中创建子目录的权限。
read_xattr (R)	读取文件的扩展属性或在扩展的属性目录中执行查找的能力。
write_xattr (W)	创建扩展属性或向扩展属性目录进行写入的能力。
execute (x)	执行文件的权限。
read_attributes (a)	读取文件的基本属性（非 ACL）的能力。基本属性将视为 stat 级别属性。允许此访问掩码位意味着实体可以执行 ls(1) 和 stat(2)。
write_attributes (A)	将与文件或目录关联的时间更改为任意值的权限。
delete (d)	删除文件的权限。
delete_child (D)	删除目录中的文件的权限。

read_acl (c)	读取 ACL 的权限。
write_acl (C)	写入 ACL 的权限，或执行 chmod(1) 或 setfacl(1) 的能力。
write_owner (o)	更改所有者的权限，或执行 chown(1) 或 chgrp(1) 的能力。
synchronize (s)	在具有同步读写的服务器上本地访问文件的权限。当前不支持此权限。

NFSv4 支持以下继承标志：

file_inherit (f)	继承到目录中所有新创建的文件。
dir_inherit (d)	继承到目录中所有新创建的目录。
inherit_only (i)	放置到目录中，但不应用于目录自身，只应用于新创建的文件和目录。此标志需要 file_inherit 和/或 dir_inherit 以指示要继承的内容。
no_propagate (n)	放置在目录中并指示只能继承一个树级别的 ACL 条目。此标志需要 file_inherit 和/或 dir_inherit 以指示要继承的内容。
successful_access (S))	指示在访问成功时是否应该启动报警或审计记录。与审计/报警 ACE 类型一起使用。
failed_access (F)	指示在访问失败时是否应该启动报警或审计记录。与审计/报警 ACE 类型一起使用。
inherited (I)	继承了 ACE。
-	未授予权限。

使用以下语法表示 NFSv4 ACL：

acl_entry[,*acl_entry*]...

```
owner@:<perms>[:inheritance flags]:<allow|deny>
group@:<perms>[:inheritance flags]:<allow|deny>
everyone@:<perms>[:inheritance flags]:<allow|deny>
user:<username>[:inheritance flags]:<allow|deny>
group:<groupname>[:inheritance flags]:<allow|deny>
```

owner@	文件所有者
group@	组所有者
user	特定用户的权限
group	特定组的权限

权限和继承标志由 / 字符分隔。

ACL 规范示例：

```
user:fred:read_data/write_data/read_attributes:file_inherit:allow
owner@:read_data:allow,group@:read_data:allow,user:tom:read_data:deny
```

使用紧凑 ACL 格式时，将通过使用指示权限的 14 个唯一字母来指定权限。

使用位置 ACL 格式时，将权限指定为类似于 `ls -V` 格式的位置参数。连字符 (-) 指示在该位置未授予任何权限，可以忽略它，只需指定必需的字母。

将按照位置表示法中指定的顺序列出上面的字母。

使用这些字母，您可以按以下等效方式指定权限。

```
user:fred:rw------R------:file_inherit:allow
```

也可以删除 - 使权限紧挨在一起。

```
user:fred:rwR:file_inherit:allow
```

还可按更紧凑的方式指定继承标志，如下所示：

```
user:fred:rwR:f:allow
user:fred:rwR:f------:allow
```

Shell 级别 Solaris API

Solaris 命令接口支持对 ACL 进行操作。以下 Solaris 实用程序适应两种 ACL 模型：

`chmod` — `chmod` 实用程序已得到增强，允许设置和删除 ACL。通过扩展 symbolic-mode 参数以支持 ACL 操作可实现此目的。有关详细信息，请参见 `chmod(1)`。

`compress` — 压缩文件时，与原始文件关联的任何 ACL 将与压缩文件一起保留。

`cp` — 缺省情况下，`cp` 将忽略 ACL，除非指定了 `-p` 选项。指定 `-p` 时，将保留所有者和组 ID、权限模式、修改和访问时间、ACL 和扩展属性（如果适用）。

`cpio` — 指定 `-P` 选项时将保留 ACL。

`find` — 指定 `-acl` 标志时查找包含 ACL 的文件。

`ls` — 缺省情况下，`ls` 不显示 ACL 信息。指定 `-v` 选项时，将显示文件的 ACL。

`mv` — 移动文件时，将随重命名的文件携带所有属性。跨文件系统边界移动文件时，将复制 ACL。如果不能复制 ACL 信息，移动将失败，并且不会删除源文件。

`pack` — 打包文件时，与原始文件关联的任何 ACL 将与打包文件一起保留。

`rcp` — `rcp` 已增强为支持复制。仅当远程主机支持 ACL 时，才保留文件的 ACL。

`tar` — 指定 `-p` 选项时将保留 ACL。

unpack 解压缩包含 ACL 的文件时，解压缩的文件会保留 ACL 信息。

应用程序级别 API

在程序级别访问文件系统 ACL 所需的主要接口是 acl_get() 和 acl_set() 函数。这些函数同时支持 POSIX 样式的 ACL 和 NFSv4 ACL。

检索文件的 ACL

```
int acl_get(const char *path, int flag, acl_t **aclp);
int facl_get(int fd, int flag, acl_t **aclp);
```

acl_get(3SEC) 和 facl_get(3SEC) 函数可在其名称由路径给定或由打开文件描述符 fd 引用的文件上检索 ACL。标志参数指定是否应检索普通 ACL。如果标志参数等于 ACL_NO_TRIVIAL，则将仅检索非普通 ACL。将在 aclp 参数中返回 ACL。

释放 ACL 结构

```
void acl_free(acl_t *aclp)s;
```

acl_free() 函数可释放为参数 aclp; 分配的内存。

在文件上设置 ACL

```
int acl_set(const char *path, acl_t *aclp);
int facl_set(int fd, acl_t *aclp);
```

acl_set(3SEC) 和 facl_get(3SEC) 函数用于在其名称由路径给定或由打开文件描述符 fd 引用的文件上设置 ACL。aclp 参数指定要设置的 ACL。在目标文件系统支持 NFSv4 ACL 时，acl_set(3SEC) 可将 POSIX 样式的 ACL 转换为 NFSv4 ACL。当尝试在 POSIX 样式的 ACL 支持的文件系统上设置 NFSv4 ACL 时，将不执行任何转换。

确定 ACL 的普通程度

```
int acl_trivial(const char *path);
```

acl_trivial() 函数用于确定文件是否具有普通 ACL。ACL 是否普通取决于 ACL 的类型。如果 POSIX 样式的 ACL 具有的条目数多于 MIN_ACL_ENTRIES，则它是非普通的。如果 NFSv4/ZFS 样式的 ACL 具有除 owner@、group@ 和 everyone@ 之外的条目，具有继承标志集，或未按照符合 POSIX 访问控制要求的方式进行排序，则它是非普通的。

从文件中删除所有 ACL

```
int acl_strip(const char *path, uid_t uid, gid_t gid, mode_t mode);
```

acl_strip() 函数可从文件中删除所有 ACL，并将其替换为基于传入的参数模式的普通 ACL。在替换 ACL 后，文件所有者和组将设置为 uid 和 gid 参数中指定的值。

在 ACL 与外部表示形式之间转换

```
int acl_fromtext(const char *path, acl_t **aclp);
char *acl_totext(acl_t *aclp, int flags);
```

acl_text() 函数可将 aclp 指向的内部 ACL 表示形式转换为外部表示形式。有关外部表示形式的详细信息，请参见“描述”部分。

acl_fromtext() 函数可将外部表示形式转换为内部表示形式。有关外部表示形式的详细信息，请参见“描述”部分。

示例

以下示例演示 API 如何用于对 ACL 执行基本操作。

示例 1 检索和设置某个 ACL

使用以下命令可检索某个 ACL 并在另一个文件上设置它：

```
error = acl_get("file", ACL_NO_TRIVIAL, &aclp);

if (error == 0 && aclp != NULL) {
error = acl_set("file2", aclp)
acl_free(aclp);
}
...
```

示例 2 检索和设置任何 ACL

使用以下命令可检索任何 ACL（包括普通 ACL），并在另一个文件上设置它：

```
error = acl_get("file3", 0, &aclp);
if (error == 0) {
error = acl_set("file4", aclp)
acl_free(aclp);
}
...
```

示例 3 确定文件是否具有普通 ACL

使用以下命令可确定文件是否具有普通 ACL：

```
istrivial = acl_trivial("file")

if (istrivial == 0)
printf("file %s has a trivial ACL\n", file);
else
printf("file %s has a NON-trivial ACL\n", file);
...
```

示例 4 从文件中删除所有 ACL

使用以下命令可从文件中删除所有 ACL，并设置新模式、所有者和组：

```
error = acl_strip("file", 10, 100, 0644);
...
```

另请参见

chgrp(1)、chmod(1)、chown(1)、cp(1)、cpio(1)、find(1)、ls(1)、mv(1)、tar(1)、setfacl(1)、

引用名

ad – Active Directory 用作命名系统信息库

描述

Solaris 客户机可以从 Active Directory (AD) 服务器获取命名信息。

Solaris 系统必须首先加入 AD 域，然后再将 ad 关键字添加到 nsswitch.conf(4) 文件的相应条目中。Solaris 系统通过使用 kclient(1M) 实用程序来加入 AD 域。AD 名称服务仅对 passwd 和 group 支持命名数据库。

Windows 用户无法登录。user_attr(4) 数据库不包含 Windows 用户的对应条目，并且 passwd(1) 命令不支持与 AD 之间同步用户口令。

Solaris AD 客户机使用自动搜索方法查找 AD 目录服务器，例如，域控制器和全局目录服务器。客户机还使用 LDAP v3 协议访问 AD 服务器中的命名信息。由于 AD 客户机与本机 AD 模式协同工作，因此无需修改 AD 服务器模式。Solaris AD 客户机使用 idmap(1M) 服务在 Windows 安全标识符 (security identifier, SID) 和 Solaris 用户标识符 (user identifier, UID) 及组标识符 (group identifier, GID) 之间映射。用户名和组名源自 AD 用户和组对象的 sAMAccountName 属性，这些名称使用对象所在的域进行标记。使用 @ 字符分隔域名和用户名或组名。

客户机使用 SASL/GSSAPI/KRB5 安全模型。kclient 实用程序用于将客户机加入到 AD。执行加入操作时，kclient 在客户机上配置 Kerberos v5。请参见 kclient(1M)。

文件

/etc/nsswitch.conf	用于名称服务转换的配置文件。
/etc/nsswitch.ad	使用 ad、dns 和文件配置的名称服务转换的样例配置文件。
/usr/lib/nss_ad.so.1	AD 的名称服务转换模块。

另请参见

passwd(1)、svcs(1)、idmap(1M)、idmapd(1M)、kclient(1M)、svcadm(1M)、svccfg(1M)、svccfg(1

引用名

ars – receive Solaris audit logs from a remote server

用法概要

```
/usr/sbin/auditd
```

描述

Audit Remote Server (ARS) is the counterpart of the `audit_remote(5)` plugin. Data sent by the plugin can be captured, processed, and stored by the server according to the its configuration.

ARS is delivered as a disabled Solaris audit component. It is necessary to configure ARS before it can be used to process a remote audit trail. ARS configuration is twofold:

- the underlying security mechanisms used for secure audit data transport has to be configured (see `audit_remote(5)`);
- the audit remote subsystem has to be configured.

To observe and configure the ARS, use the `auditconfig(1M)` `-setremote` and `-getremote` options. The configuration is divided between the configuration of *server* and *group*. The *server* configuration allows for changing common ARS parameters, while the *group* keyword allows configuration of connection groups, the sets of hosts sharing the same local storage parameters.

Server configuration attributes

listen_address
The address the server listens on. An empty *listen_address* attribute defaults to listen on all local addresses.

listen_port
The local listening port; 0 defaults to 16162, the port associated with the “solaris-audit” Internet service name. See `services(4)`.

login_grace_time
The server disconnects after login grace time (in seconds) if the connection has not been successfully established; 0 defaults to no limit.

max_startups
The number of concurrent unauthenticated connections to the server at which the server starts refusing new connections. The value might be specified in *begin*:*rate*:*full* format to allow random early drop mode, for example 10:30:60, meaning that ARS would refuse connection attempts with a probability of *rate*/100 (30% in our example) if there are currently 10 (from the *start* field) unauthenticated connections. The probability increases linearly and all connection attempts are refused if the number of unauthenticated connections reaches *full* (60 in our example).

Group configuration attributes

The *binfile_dir*, *binfile_fsize*, and *binfile_minfree* attributes follow the respective *p_** attributes defined in `audit_binfile(5)`. Brief descriptions follow.

binfile_dir
The directory for storing per host audit data.

binfile_fsize
The maximum size of each of the stored audit trail files; 0 defaults to no limit.

binfile_minfree
The minimum free space on file system with *binfile_dir* before the `audit_binfile` informs the administrator via `audit_warn`(1M); 0 defaults to no limit.

hosts
The hosts in the given connection group allowed to send audit data to server. A comma is a delimiter in case of multiple host entries. If *hosts* is empty, such connection group is called a wild card connection group. If a new connection cannot be classified to any other (non-wild card) connection group and there is an active wild card connection group configured, the new connection is classified to that connection group. Only one active wild card connection group can be configured.

For comprehensive configuration description and examples, see the section on Auditing in 《Oracle Solaris 11.1 管理：安全服务》.

示例

示例 1 Audit Remote Server configuration

The following example describes steps to configure audit remote server to listen on specific address. One wild card and one non-wild card connection group will be created. The non-wild card connection group configuration will address remote audit data from `tic.cz.example.com` and `tac.us.example.com`, the trail will be stored in `/var/audit/remote`.

```
# Print the current audit remote server configuration.
# Both server and connection groups (if any) is displayed.
auditconfig -getremote

# Set address the audit remote server will listen on.
auditconfig -setremote server "listen_address=192.168.0.1"

# Create two connection groups. Note that by default the
# connection group is created with no hosts specified
# (wild card connection group).
auditconfig -setremote group create clockhouse
auditconfig -setremote group create sink

# Add hosts to the connection group (convert the wild card
# connection group no non-wild card one). Set the storage
# directory and activate the connection group.
auditconfig -setremote group active clockhouse \
    "hosts=tic.cz.example.com,tac.us.example.com,
    binfile_dir=/var/audit/remote"

# Activate the wild card connection group.
auditconfig -setremote group active sink
```

示例 1 Audit Remote Server configuration （续）

```
# Verify the audit remote server configuration.
auditconfig -getremote

# Start or refresh the audit service.
audit -s
```

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	Committed

另请参见

audit(1M), auditconfig(1M), auditd(1M), audit_warn(1M), services(4), attributes(5), audit_binfile(5), smf(5)

See the section on Auditing in 《Oracle Solaris 11.1 管理：安全服务》.

附注

The audit service FMRI is svc:/system/auditd:default.

引用名

ascii – map of ASCII character set

用法概要

```
cat /usr/pub/ascii
```

描述

`/usr/pub/ascii` is a map of the ASCII character set, to be printed as needed. It contains octal and hexadecimal values for each character. While not included in that file, a chart of decimal values is also shown here.

```
                         Octal — Character

000 NUL  001 SOH  002 STX  003 ETX  004 EOT  005 ENQ  006 ACK  007 BEL
010 BS   011 HT   012 NL   013 VT   014 NP   015 CR   016 SO   017 SI
020 DLE  021 DC1  022 DC2  023 DC3  024 DC4  025 NAK  026 SYN  027 ETB
030 CAN  031 EM   032 SUB  033 ESC  034 FS   035 GS   036 RS   037 US
040 SP   041 !    042 "    043 #    044 $    045 %    046 &    047 ’
050 (    051 )    052 *    053 +    054 ,    055 –    056 .    057 /
060 0    061 1    062 2    063 3    064 4    065 5    066 6    067 7
070 8    071 9    072 :    073 ;    074 <    075 =    076 >    077 ?
100 @    101 A    102 B    103 C    104 D    105 E    106 F    107 G
110 H    111 I    112 J    113 K    114 L    115 M    116 N    117 O
120 P    121 Q    122 R    123 S    124 T    125 U    126 V    127 W
130 X    131 Y    132 Z    133 [    134 \    135 ]    136 ^    137 _
140 ‘    141 a    142 b    143 c    144 d    145 e    146 f    147 g
150 h    151 i    152 j    153 k    154 l    155 m    156 n    157 o
160 p    161 q    162 r    163 s    164 t    165 u    166 v    167 w
170 x    171 y    172 z    173 {    174 |    175 }    176 ~    177 DEL
```

```
                      Hexadecimal — Character

00 NUL   01 SOH   02 STX   03 ETX   04 EOT   05 ENQ   06 ACK   07 BEL
08 BS    09 HT    0A NL    0B VT    0C NP    0D CR    0E SO    0F SI
10 DLE   11 DC1   12 DC2   13 DC3   14 DC4   15 NAK   16 SYN   17 ETB
18 CAN   19 EM    1A SUB   1B ESC   1C FS    1D GS    1E RS    1F US
20 SP    21 !     22 "     23 #     24 $     25 %     26 &     27 ’
28 (     29 )     2A *     2B +     2C ,     2D –     2E .     2F /
30 0     31 1     32 2     33 3     34 4     35 5     36 6     37 7
38 8     39 9     3A :     3B ;     3C <     3D =     3E >     3F ?
40 @     41 A     42 B     43 C     44 D     45 E     46 F     47 G
48 H     49 I     4A J     4B K     4C L     4D M     4E N     4F O
50 P     51 Q     52 R     53 S     54 T     55 U     56 V     57 W
58 X     59 Y     5A Z     5B [     5C \     5D ]     5E ^     5F _
60 ‘     61 a     62 b     63 c     64 d     65 e     66 f     67 g
68 h     69 i     6A j     6B k     6C l     6D m     6E n     6F o
70 p     71 q     72 r     73 s     74 t     75 u     76 v     77 w
78 x     79 y     7A z     7B {     7C |     7D }     7E ~     7F DEL
```

```
                        Decimal — Character

  0 NUL    1 SOH    2 STX    3 ETX    4 EOT    5 ENQ    6 ACK    7 BEL
  8 BS     9 HT    10 NL    11 VT    12 NP    13 CR    14 SO    15 SI
```

```
 16 DLE  17 DC1   18 DC2   19 DC3   20 DC4   21 NAK   22 SYN   23 ETB
 24 CAN  25 EM    26 SUB   27 ESC   28 FS    29 GS    30 RS    31 US
 32 SP   33 !     34 "     35 #     36 $     37 %     38 &     39 ’
 40 (    41 )     42 *     43 +     44 ,     45 —     46 .     47 /
 48 0    49 1     50 2     51 3     52 4     53 5     54 6     55 7
 56 8    57 9     58 :     59 ;     60 <     61 =     62 >     63 ?
 64 @    65 A     66 B     67 C     68 D     69 E     70 F     71 G
 72 H    73 I     74 J     75 K     76 L     77 M     78 N     79 O
 80 P    81 Q     82 R     83 S     84 T     85 U     86 V     87 W
 88 X    89 Y     90 Z     91 [     92 \     93 ]     94 ^     95 _
 96 ‘    97 a     98 b     99 c    100 d    101 e    102 f    103 g
104 h   105 i    106 j    107 k    108 l    109 m    110 n    111 o
112 p   113 q    114 r    115 s    116 t    117 u    118 v    119 w
120 x   121 y    122 z    123 {    124 |    125 }    126 ~    127 DEL
```

文件

`/usr/pub/ascii` On-line chart of octal and hexadecimal values for the ASCII character set.

引用名

attributes, architecture, availability, CSI, stability, MT-Level, standard – 接口属性

描述

手册页的**属性**部分包含一个定义属性类型及其相应值的表。下面提供了属性表的一个示例。并非所有属性类型都适用于所有接口类型。

属性类型	属性值
体系结构	SPARC
可用性	system/kernel
CSI	Enabled（已启用）
接口稳定性	Committed（已确定）
MT 级别	Safe（安全）
标准	请参见 standards(5)。

体系结构

体系结构定义处理器或特定硬件。请参见 uname(1) 的 -p 选项。在某些情况下，体系结构可能表示所需适配器或外围设备。

可用性

这表示包含本手册页介绍的命令或组件的软件包。要使用此命令，必须安装指示的软件包。有关如何添加软件包的信息，请参见 pkg(1)。

代码集独立性 (Code Set Independence, CSI)

不依赖于任何代码集的属性的 OS 实用程序和库具有代码集独立性 (Code Set Independence, CSI)。它们具有启用 CSI 的属性。这区别于许多仅使用扩展 Unix 代码集 (Extended Unix Codeset, EUC) 等编码方法的命令和实用程序。扩展 Unix 代码集编码方法允许同时支持最多四个代码集并且通常用于表示亚洲字符集。

不过，出于实际原因，这种独立性不是绝对的。某些假设对当前 CSI 实现依然适用：

- 文件代码是 ASCII 的超集。
- 要支持多字节字符和以 null 结尾的 UNIX 文件名，任何多字节字符不能包含 NULL 和 /（斜杠）字符。
- 仅支持“无状态”文件代码编码。无状态编码可避免移位、锁定移位、指定、调用等，但单一移位未排除在外。
- 进程代码（wchar_t 值）取决于实现，并且可随时间、实现或语言环境而发生更改。
- 并非每个对象的名称都可由任意字符组成。下列对象的名称必须由 ASCII 字符组成：
  - 用户名、组名和口令
  - 系统名称
  - 打印机和特殊设备的名称
  - 终端名称 (/dev/tty*)
  - 进程 ID 号

- 消息队列、信号量和共享内存标签。
- 下列各项可由 ISO Latin-1 或 EUC 字符组成：
  - 文件名
  - 目录名称
  - 命令名称
  - Shell 变量和环境变量名称
  - 文件系统挂载点
  - NIS 键名和域名

- NFS 共享文件的名称应由 ASCII 字符组成。尽管文件和目录的名称及内容可由非 ASCII 代码集中的字符组成，但如果仅使用 ASCII 代码集，则允许在任何计算机上挂载 NFS，而不管是否已本地化。对于启用了 CSI 的命令和实用程序，均可以处理 2.6 版本中发布的单字节和多字节语言环境。对于要获取国际化服务完全支持的应用程序，必须应用动态绑定。只有 C 和 POSIX 语言环境才支持静态绑定程序。

接口稳定性

Oracle Solaris 通常使开发者提前就能够接触到新技术，这使得开发者能够尽早对这些技术进行评估。遗憾的是，新技术容易发生更改，并且标准化新技术往往会导致接口与以前的版本不兼容。

为了进行合理的风险评估，开发者必须了解接口在将来发行版中发生更改的可能性。为了帮助开发者进行上述评估，某些手册页中提供了命令、入口点和文件格式的接口稳定性信息。

由于 Oracle Solaris 将尽力确保在将来的次要发行版中继续使用这些接口，因此这些接口将更加稳定，可供几乎所有应用程序安全地使用。仅依赖于 "Committed"（已确定）接口的应用程序应在将来的次要发行版中（而不一定是在早期主要发行版中）继续可靠地正常运行。

欠稳定的接口可用于进行实验和设计原型，但使用时必须了解这些接口可能会发生不兼容的更改，甚至可能在将来的次要发行版中被删除或替换为其他接口。

Oracle Solaris 未记录的“接口”（例如，大多数内核数据结构以及系统头文件中的某些符号）可能是实现工件。此类内部接口不仅会发生不兼容的更改或被删除，我们还可能不会在发行说明中提及此类更改。

发行版级别

产品具有指定的发行版级别和名称，这些有助于进行兼容性介绍。每个发行版级别还可能包括适合较低级别的更改。

发行版	版本	含义
主	x.0	可能包含增加的主要功能；遵循可能不兼容的不同标准修订；可能会更改、删除或替换 "Committed"（已确定）接口（虽然这些情况不太可能发生）。产品初始发行版通常为 1.0。

发行版	版本	含义
次	x.y	与 x.0 或早期发行版 (y!=0) 相比，此发行版可能包括：增加的功能、对 "Committed"（已确定）接口所做的兼容更改、或者可能对 "Uncommitted"（未确定）或 "Volatile"（可变）接口所做的不兼容更改。
微	x.y.z	应为与上一个发行版 (z!=0) 兼容的接口，但可能修复了更多错误、改进了性能并且支持其他硬件。可能对 "Volatile"（可变）接口进行了不兼容的更改。

在接口稳定性上下文中，更新发行版（有时称为修补发行版）应视为等效于微发行版。

分类

下表概述了稳定性级别分类与发行版级别的关系。第一列中列出了稳定性级别。第二列中列出了不兼容的更改所对应的发行版级别，第三列中列出了其他注释。有关各分类的完整讨论，请参见下文的相应小节。

稳定性	发行版	注释
Committed（已确定）	主发行版 (x.0)	极少发生不兼容情况。
Uncommitted（未确定）	次发行版 (x.y)	经常发生不兼容情况。
Volatile（可变）	微发行版 (x.y.z)	经常发生不兼容情况。

除非另行说明，否则本手册页中介绍的接口稳定性级别分类适用于源代码接口和二进制接口。所有稳定性级别分类都是公共的，但 "Private"（专用）分类除外。除非明确说明，否则不会指定公共接口（即本手册页中记录的接口）的确切稳定性级别。未记录接口的稳定性级别缺省为 "Private"（专用）。

除了 Oracle Solaris 产品中包含的文档以外，其他现有文档不应解释为暗指 Oracle Solaris 产品所提供接口的任何稳定性级别。参考手册页是稳定性级别信息的唯一来源。

Committed（已确定）

"Committed"（已确定）接口的用途在于使第三方能够根据这些接口开发和发布应用程序，并确信这些应用程序能够在引入接口的产品发行版的所有后续发行版（属于同一主要发行版）中正常运行。即使在主要发行版中，不兼容的更改也应当极少出现，并且应具有正当理由。

作为行业标准定义和控制的接口通常视为 "Committed"（已确定）接口。在这种情况下，属性表中的“标准”条目或其他文档位置通常会说明监管机构和/或公共文档版本。

虽然不兼容的更改很少发生，但是如果相关缺陷极其严重（如本文档的“例外情况”部分中所述），在任何发行版中都可能会发生不兼容的更改；或者在次要发行版中，可能会通过“功能终止”过程来执行不兼容的更改。如果必须停止对 "Committed"（已确定）接口的支持，Oracle Solaris 将会尝试提供通知并将稳定性级别标记为 "Obsolete"（已过时）。

Uncommitted（未确定）

不对这些接口在不同次要发行版中的源代码或二进制兼容性进行任何承诺。甚至在次要发行版中可能会发生接口删除等重大的不兼容更改。"Uncommitted"（未确定）接口通常不适用于与发行版无关的产品。

对接口进行不兼容更改旨在对接口进行实质性的改进（包括考虑到易用性等因素）。一般情况下，"Uncommitted"（未确定）接口不太可能进行不兼容的更改，如果发生此类更改，这些更改将影响甚微，并且通常具有减轻风险计划。

"Uncommitted"（未确定）接口通常属于下列子类别之一：

1. 实验性或过渡性接口。这些接口通常旨在使外部开发者可以及早接触到一些新兴的、不断发展变化的技术，或者提供一个临时的解决问题的办法，有待将来再寻求更通用的解决方案。
2. 其规范由外部机构控制的接口，但 Oracle Solaris 希望在提供与外部规范同步的下一个次要发行版之前尽力与以前的发行版保持兼容。
3. 相较于稳定性而言，其目标用户更重视创新（或者易用性）的接口。此属性通常与较高层组件的管理接口相关联。

对于 "Uncommitted"（未确定）接口，Oracle Solaris 不会对不同次要发行版之间的源代码或二进制文件兼容性做出任何声明。根据这些接口开发的应用程序可能无法在将来的次要发行版中运行。

Volatile（可变）

"Volatile"（可变）接口可能出于任何原因而随时发生更改。

通过 "Volatile"（可变）接口稳定性级别，Oracle Solaris 产品可以快速跟上不断发展变化的规范。在许多情况下，与为接口提供额外的稳定性相比，"Volatile"（可变）接口能够更好地满足使用者的期望，因此人们更喜欢使用 "Volatile"（可变）接口。

此分类级别最常应用于由 Oracle Solaris 以外的机构控制的接口，但与重视接口兼容性的标准机构或免费/开源软件 (Free or Open Source Software, FOSS) 社区控制的规范不同的是，无法声明极少对接口规范进行的不兼容更改。此外，此接口还适用于由 FOSS 控制的软件，对于此类软件，大家认为在最短时间内了解社区动态比向我们的客户提供稳定性更为重要。

通常还可以在可靠组织或广泛认可的组织定义接口的过程中，将 "Volatile"（可变）分类级别应用于接口。这些级别通常称为标准草案。“IETF Internet 草案”就是一个广为人知的正在开发的规范的示例。

此外，实验性接口也可以是 "Volatile"（可变）接口。

我们不对任何两个发行版（包括修补程序）之间的 "Volatile"（可变）接口的源代码或二进制兼容性做出任何声明。包含这些接口的应用程序可能无法在将来的任何发行版中正常运行。

`Not-an-Interface`（不是接口）

有时会出现以下情况：推断存在的某个实体可能是一个接口，但实际上却并非是接口。常见示例包括：仅供人员使用的 CLI 的输出以及 GUI 的确切布局。

此分类是一个适合用于阐明确定可能存在此类混淆的术语。如果无法对实体应用此术语，也并不意味着该实体就是某种形式的接口。它仅表明未确定可能存在此类混淆。

`Private`（专用）

"Private"（专用）接口是由组件（或产品）提供的专用于该组件的接口。"Private"（专用）接口仍可对其他组件可见或可由其他组件访问。由于使用其他组件的专用接口存在巨大的稳定性风险，因此明确不支持这种使用方式。未由 Oracle Solaris 提供的组件不应使用 "Private"（专用）接口。

大多数 "Private"（专用）接口都未予以记录。我们很少记录 "Private"（专用）接口。记录 "Private"（专用）接口的原因包括（但不限于）：接口用途可能会在将来重新分类为某个公共稳定性级别分类，或者该接口会无规律地显现。

`Obsolete`（过时）

"Obsolete"（已过时）是可与上述分类级别一起显示的修饰符。"Obsolete"（已过时）修饰符表示接口“已弃用”并且/或者建议不要继续用于一般用途。通过应用 "Obsolete"（已过时）修饰符，现有接口可从某个其他状态（例如，"Committed"（已确定）或 "Uncommitted"（未确定））发生降级，以便建议客户先从该接口进行迁移，然后再删除（或以不兼容方式更改）该接口。

当前发行版支持 "Obsolete"（已过时）接口，但计划在将来的（次要）发行版中将其删除。停止接口支持之前，Oracle Solaris 将会首先尝试提供通知，然后再停止支持相应接口。使用 "Obsolete"（已过时）接口会生成警告消息。

例外情况

在极少情况下，为了维护 Oracle Solaris 和客户的最佳利益，需要违反接口稳定性承诺。下表包含接口提供者违反接口稳定性确定的常见已知原因，但不排除存在其他原因。

1. 存在安全漏洞（接口固有的漏洞）。
2. 发生数据损坏（接口固有的漏洞）。
3. 违反标准的情况（由一致性测试的解释或改进中的更改所揭示）。
4. 某个非由 Oracle Solaris 控制的接口规范已发生不兼容的更改，并且大多数接口使用者都希望提供更新的接口。
5. 对于客户来说，不进行不兼容更改是无法接受的。例如，在放弃 DOS 8.3 命名限制后，如果不对 pcfs 进行不兼容的更改，就会是一个这样的示例。

例外情况允许的不兼容更改将始终尽可能在“最主要的”发行版中提供。但是，漏洞带来的后果或合同相关规定要求往往会强制在修补程序中提供。

与早期接口分类机制的兼容性

在 Solaris 10 以及先前的发行版中，采用不同的接口分类机制。下表概述了新旧分类机制之间的映射。

旧机制	新机制	注释
Standard（标准）	Committed（已确定）	应显示属性表中标准属性类型的条目。
Stable（稳定）	Committed（已确定）	名称更改。
Evolving（发展中）	Uncommitted（未确定）	实际承诺相符。
Unstable（不稳定）	Uncommitted（未确定）	名称更改。
External（外部）	Volatile（可变）	名称更改，同时扩展了允许的使用情况。
Obsolete（过时）	(Obsolete)（过时）	以前为分类，现在为修饰符。

免费/开源软件的重要性日益提高，促使将名称从 "Stable"（稳定）/"Unstable"（不稳定）更改为 "Committed"（已确定）/"Uncommitted"（未确定）。"Stable"（稳定）一词与该术语在 FOSS 社区中的常见用途相冲突。

"Evolving"（发展中）的定义比较模糊，导致很难理解此术语。在迁移到新分类机制的过程中，以前的许多 "Evolving"（发展中）的接口都已升级为 "Committed"（已确定）。不过，在遇到术语 "Evolving"（发展中）时，应推断为 "Uncommitted"（未确定）。

MT 级别

库分为若干类别，这些类别定义了其支持多个线程的能力。包含属于多个或不同级别的函数的手册页在“附注”部分或“用法”部分中对此方面进行了介绍。

`Safe`（安全）

“安全”是可从多线程应用程序调用的代码的属性。调入安全接口或安全代码段的作用是：即使由多个线程调用，结果仍然有效。人们常常忽视的一点是：此安全接口或安全代码段的结果可产生影响所有线程的全局后果。例如，从一个线程打开或关闭文件的操作对进程中的所有线程都可见。多线程应用程序负责安全地使用这些接口，这与此接口是否安全有所不同。例如，关闭应用程序中其他线程仍在使用的文件的多线程应用程序未安全地使用 `close(2)` 接口。

`Unsafe`（非安全）

非安全库包含不受保护的全局和静态数据。除非应用程序安排每次仅在库中执行一个线程，否则使用此库会不安全。非安全库可能包含安全函数；不过，库包含的大

多数函数在调用时都是不安全的。某些非安全函数具有多线程安全的可重入函数。可重入函数由附加到函数名称的 _r 后缀指定。

MT-Safe（MT 安全）

多线程安全库是为多线程访问而充分准备的库。它通过锁定保护其全局和静态数据，并且可提供合理数目的并发性。可以安全使用的库并不能视为多线程安全。例如，使用监视器监视整个库可使库保持安全，但它不支持并发性，因此不能视为多线程安全。多线程安全库必须允许合理数目的并发性。（此定义的目的是精确定义安全库的含义。安全库的定义不会指定该库是否支持并发性。多线程安全定义明确指明该库是安全的，并且支持一定程度的并发性。这阐明了安全定义，它可以表示从单线程到任何并发度的多线程的所有内容。）

Async-Signal-Safe（异步信号安全）

"异步信号安全"表示可从信号处理程序安全地调用的特定库函数。执行异步信号安全函数的线程在被信号中断时，自身不会发生死锁。信号只会为获取锁定的多线程安全函数带来问题。

异步信号安全函数也具有多线程安全性。在异步信号安全函数中获取锁定时，将禁用信号。这些信号用于防止调用可能获取相同锁定的信号处理程序。

MT-Safe with Exceptions（MT 安全，但存在异常）

有关异常的说明，请参见这些页面的"附注"部分或"用法"部分。

Safe with Exceptions（安全，但存在异常）

有关异常的说明，请参见这些页面的"附注"部分或"用法"部分。

Fork-Safe（Fork 安全）

fork(2) 函数仅在子进程中复制调用线程。fork1(2) 函数的存在目的是为了与以前版本兼容，它与 fork() 同义。当调用 fork() 时，如果未在执行派生的其他线程保持锁定，该锁定仍将保持在子进程中，但是由于未复制所属线程，因此没有锁定所有者。调用尝试获取锁定的函数的子进程自身将发生死锁。

当调用 fork() 时，Fork 安全库安排仅让执行派生的线程保留该库的所有内部锁定。这通常是使用 pthread_atfork(3C) 实现的，该函数在初始化库时调用。

在极少情况下，如果进程需要在执行派生时复制其所有线程，forkall(2) 函数会提供此功能。调用 forkall() 时，不会执行 pthread_atfork() 操作。调用 forkall() 存在相应的危险。当某个线程调用 forkall() 时，如果进程中的某些其他线程正在执行 I/O 操作，这些线程将继续在父进程和子进程中执行相同的 I/O 操作，这可能会导致数据损坏。出于此原因以及其他竞争情况原因，不建议使用 forkall()。

在 Solaris 10 之前的所有 Solaris 发行版中，fork() 的行为取决于应用程序是否与 -lpthread 相链接（有关 POSIX 线程，请参见 standards(5)）。如果与 -lpthread 链接，fork() 的行为与 fork1() 相似，否则与 forkall() 相似。为了避免产生有关 fork() 行为的任何混淆，应用程序可以根据需要明确地调用 fork1() 或 forkall()。

Cancel-Safety（取消安全）

如果多线程应用程序使用 pthread_cancel(3C) 取消（即中止）线程，目标线程在中止时可能会保留某项资源，例如锁定或分配的内存。如果线程未安装有适当的取消清除处理程序来释放相应资源（请参见 pthread_cancel(3C)），该应用程序即为“取消不安全”，也就是说，从线程取消方面来说，该应用程序不安全。由于取消的线程未释放锁定，这种非安全性可能导致死锁或资源泄漏；例如，不会在取消线程时释放内存。使用 pthread_cancel(3C) 的所有应用程序都应确保它们在“取消安全”环境中运行。此外，如果库具有取消点并且获取锁定等资源或动态分配内存，也会导致与这些库关联的应用程序的取消不安全性。这为多线程程序中的库引入了另一个安全级别：取消安全。取消安全包含两个子类别：延迟取消安全以及异步取消安全。如果应用程序对于取消类型为 PTHREAD_CANCEL_DEFERRED 的线程为取消安全时，该应用程序被视为延迟取消安全。如果应用程序对于取消类型为 PTHREAD_CANCEL_ASYNCHRONOUS 的线程为取消安全时，该应用程序被视为异步取消安全。由于具有延迟取消类型的线程只能在正确定义的取消点取消，而具有异步取消类型的线程可在任意位置取消，因此延迟取消安全比异步取消安全更容易实现。缺省情况下，创建的所有线程都具有延迟取消类型，因此可能永远不需要担心异步取消安全。大多数应用程序和库都应当始终为异步取消不安全。根据定义，异步取消安全的应用程序同时也是延迟取消安全的。

标准

许多接口都作为行业标准进行定义和控制。在这种情况下，本部分中将说明监管机构和/或公共文档版本。

程序员在生成可移植应用程序时，应该遵照此应用程序应符合的标准或规范中提供的接口说明，而不能遵照基于公共标准的接口的手册页说明。当标准或规范允许采用替代实现选择时，手册页通常仅介绍由 Oracle Solaris 实现的替代项。手册页还介绍对 Oracle Solaris 提供的标准接口的基本定义的所有可兼容扩展。

对于文中引用的监管机构或文档，并不意味着我们将其认可为“标准”条目。监管机构可以是非常正式的组织（例如 ISO 或 ANSII）、较不正式但广泛接受的组织（例如 IETF）或非正式的独立贡献者（例如 FOSS（Free or Open Source Software，免费/开源软件）贡献者）。

另请参见

uname(1)、Intro(3)、standards(5)

pkg(1)

引用名

audit_binfile – 生成 Solaris 审计日志

用法概要

```
/usr/lib/security/audit_binfile.so
```

描述

用于 Solaris 审计的 audit_binfile 插件模块 /usr/lib/security/audit_binfile.so 可按照 auditconfig(1M) 中的配置将二进制审计数据写入文件；它是 Solaris 审计守护进程 auditd(1M) 的缺省插件。其输出由 audit.log(4) 进行描述。

如果 audit_binfile 插件通过 auditconfig 配置为活动状态，则由 auditd 装入该插件。使用 auditconfig -setplugin 选项可以更改所有与插件相关的配置参数。

对象属性

下列属性指定 audit_binfile 插件的配置：

p_dir

```
dir1[,dir2],.. [,dirn]
```

将在其中创建审计文件的目录列表。可以指定任何有效可写目录。

p_minfree

一个百分比，用于指示目标 p_dir 上所需的可用空间量。如果可用空间低于此阈值，则审计守护进程 auditd(1M) 将调用 shell 脚本 audit_warn(1M)。如果未指定阈值，则缺省百分比为 1%。

p_fsize

p_fsize 属性定义审计文件在自动关闭并打开一个新审计文件之前可以达到的最大大小。这与审计文件大小等于管理员指定的值时管理员发出 audit -n 命令等效。缺省大小为零 (0)，这允许文件一直增长而不受约束。指定的值必须大于 500KB 并小于 16 艾字节 (EB)。所使用的文件系统可能会进一步降低该限制。p_fsize 值的格式可以指定为以字节为单位的确切值，或者使用后缀 B, K, M, G, T, P, E, Z（分别代表字节、KB、MB、GB、TB、PB、EB 和 ZB）以可读的格式指定值。也接受 KB, MB, GB, TB, PB, EB 和 ZB 作为后缀。

示例

使用以下指令可装入 audit_binfile.so，指定用于写入审计日志的目录，并指定每个目录所需可用空间的百分比。

```
auditconfig -setplugin audit_binfile active \
    "p_dir=/var/audit/jedgar/eggplant,/var/audit/jedgar.aux/eggplant,
     /var/audit/global/eggplant;p_minfree=20;p_fsize=4.5GB"
```

属性

有关以下属性的说明，请参见 attributes(5)：

属性类型	属性值
MT 级别	MT-Safe（MT 安全）
接口稳定性	Committed（已确定）

另请参见

auditconfig(1M)、auditd(1M)、audit_warn(1M)、syslog.conf(4)、attributes(5)

《Oracle Solaris 11.1 管理：安全服务》

引用名

audit_flags – 审计预选标志

描述

审计标志指定将审计进程的哪些审计类。审计类在 audit_class(4) 文件中定义，并像 audit_event(4) 文件中定义的审计事件一样组合在一起。缺省的 Solaris 系统范围的审计标志使用 auditconfig(1M) 配置为审计服务的一部分。其他每用户或每角色审计标志可在 user_attr(4) 数据库中进行配置，或在通过 *audit_flags=always-audit-flags:never-audit-flags* 关键字授予用户的配置文件中进行配置。进程的审计标志称为预选掩码。预选掩码在进行登录和角色承担时设置，方法是将缺省的 Solaris 系统范围的审计标志和每用户审计标志（缺省标志 + *always-audit-flags*）- *never-audit-flags* 组合在一起。

审计标志指定为表示要审计的审计类名的字符串。每个标志标识一个审计类，用逗号 (`,`) 与字符串中的其他审计类隔开。以 - 开头的审计类名意味着只审计该类的失败尝试；不审计成功尝试。以 + 开头的审计类名称意味着只审计该类的成功尝试；不审计失败尝试。如果没有前缀，审计类名指示审计该类的成功和失败尝试。特殊字符串 “all” 指示将审计所有审计事件；`-all` 指示将审计所有失败尝试，`+all` 指示将审计所有成功尝试。前缀 `^`、`^-` 和 `^+` 会禁用之前在字符串中指定的标志（`^-` 和 `^+` 分别对应失败和成功尝试，`^` 对应两者）。这些前缀通常用于重置标志。特殊字符串 `no` 指示将不审计任何审计事件。

示例

示例 1 通过预选可审计成功和失败的 "lo"（登录/注销）、"am"（管理）审计事件以及除失败的 "fm"（文件属性修改）事件之外的所有失败的审计事件。

```
lo,am,-all,^-fm
```

示例 2 通过预选可审计成功和失败的 "lo"（登录/注销）、"as"（系统范围管理）和失败的 "fm"（文件属性修改）事件。

```
lo,as,-fm
```

另请参见

profiles(1)、auditconfig(1M)、auditd(1M)、usermod(1M)、audit_class(4)、audit_event(4)、pr

《Oracle Solaris 11.1 管理：安全服务》

引用名

audit_remote – 将 Solaris 审计日志发送到远程服务器

用法概要

```
/usr/lib/security/audit_remote.so
```

描述

用于 Solaris 审计的 audit_remote 插件模块 /usr/lib/security/audit_remote.so 可按照二进制审计记录 (audit.log(4)) 在auditconfig(1M) 中的配置方式将二进制审计记录发送到审计服务器。

如果 audit_remote 插件通过 auditconfig 配置为活动状态，则由 auditd(1M) 装入该插件。使用 auditconfig -setplugin 选项可以更改所有与插件相关的配置参数。

Solaris 审计服务守护进程的审计远程服务 ars(5) 可使用 auditconfig 进行配置以接收 audit_remote 发送的二进制审计记录。

对象属性

下列属性指定 audit_remote 插件的配置：

p_hosts

host1[:[*port1*][:*mech1*]][,*host2*[:[*port2*][:*mech2*]],... \
  hostn[:[*portn*][:*mechn*]]]

审计主机/服务器列表。审计记录将发送到第一台可用主机。如果在发送数据时主机无法访问或出现超时，则将尝试列表中的下一台主机。如果与所有主机的连接均失败，则将从头开始重新尝试该列表。

p_hosts 条目的 *host* 部分可采用 getipnodebyname(3SOCKET) 接受的任何形式。

p_hosts 条目的 *port* 部分是将联系以启动审计服务器连接的主机上的端口。如果未指定，则该端口号是分配给 solaris-audit 服务的端口号。请参见 getservbyname(3XNET)。

p_host 条目的 mech 部分是 GSS-API 机制名称 (mech(4))。如果未指定，则使用本地主机的缺省机制。建议的机制为 kerberos_v5。

p_retries
尝试连接到服务器并向其发送数据的次数。

缺省值为 3。

p_timeout
连接/发送数据超时的秒数。

缺省值为 5 秒。

qsize
要保留的未处理审计记录的最大数目。

缺省值是内核队列控制高界限的值。请参见 auditconfig(1M)。

如果设置为 0，则缺省值是内核队列控制高界限的值。请参见 auditconfig(1M)。

GSS 会话

audit_remote plugin 是一个 TCP 客户机，可使用 GSS-API (libgss(3LIB)) 对配置的审计服务器进行验证。发送的二进制 Solaris 审计记录将通过 gss_wrap(3GSS) 生成的每消息令牌形式实施完整性和保密性保护。

该插件启动与审计服务器 (*host:port:mech*) 的 TCP 连接并通过相应的安全机制 (mech(4)) 建立 GSS 安全上下文（使用 gss_init_sec_context(3GSS)）。

如果未指定端口，则将查找服务名称 solaris-audit 以获取 TCP 端口号。如果未指定机制，GSS_C_NO_OID 将用作 gss_init_sec_context(3GSS) 的 mech_type 参数，并使基础 GSS-API 使用本地缺省机制。

gss_init_sec_context(3GSS) 使用 GSS_C_NO_CREDENTIAL 作为启动器的凭证句柄和 audit@<host_fqdn> 格式的目标名称。服务器需要使用 gss_accept_sec_context(3GSS) 以完成上下文建立。

在建立安全上下文后，客户机（audit_remote 插件）调用 gss_wrap(3GSS) 以对传送的有效负荷（审计记录）实现保密性保护。服务器需要使用 gss_unwrap(3GSS) 才能展开收到的数据，并需要使用 gss_get_mic(3GSS) 获取稍后将作为消息检索确认发送回插件的 MIC（Message Integrity Code，消息完整性代码）。

例如，如果 kerberos_v5 机制在客户机上配置为 GSS_API 机制，且双方就使用此机制达成一致，则客户机必须有资格以非交互方式从 Kerberos KDC/TGS 获取 audit/<host_fqdn>@<REALM> 主体的会话密钥。同时，运行审计服务器应用程序的身份必须具有与存储在 keytab 文件 (krb5.conf(4)) 中的 audit/<host_fqdn>@<REALM> 主体关联的长期密钥才能解密会话密钥。

audit_remote 插件启动与 p_hosts 列表中的第一台服务器的连接。如果连接失败或审计记录发送在 p_timeout 秒内没有响应，则在 p_retries 次尝试后插件会尝试连接到下一台服务器。如果与最后一台服务器的连接失败，则插件会重新尝试连接到列表中的第一台主机。每次尝试连接到服务器未成功或发送超时（插件选项插件 audit_remote.so retry <count> <error>.<error> 是连接 <host:port> <the network error>），都将执行 audit_warn(1M)。EPROTO 网络错误指示客户机插件未获得成功的协议版本握手。

协议说明

所有协议消息的前面都带有说明后跟数据大小的 4 个八位字节。此大小采用网络字节顺序。

协议以版本协商开头，后跟 GSS-API 安全上下文令牌交换。出现错误时将关闭连接（可以选择发送任何输出令牌）。

版本协商以明文形式进行，插件将发送受支持版本的逗号 (,) 分隔列表的八位字节数组。当前版本号是字符 01。接收者需要使用其接受的版本（在当前情况下为字符 01）进行响应。不匹配将视为错误，并将关闭连接。

插件发送的版本八位字节数组和接收者接受的版本字符串联在一起，以构成 GSS 安全上下文建立的通道绑定的应用程序数据字段。

```
<plugin version characters> || <server accepted version characters>
"||" represents concatenation
```

后续令牌包含一个采用网络字节顺序的 64 位序列号以及一条审计记录 (audit.log(4))；客户机使用保密性保护。wrap (64 位序列号 || 审计记录)

服务器使用收到的 64 位序列号以及已展开的 64 位序列号和审计记录的 MIC 令牌确认收到，然后对任何数据丢失负责。客户端上的 MIC 验证确认可以释放审计记录且不保存其用于可能的重新传输。

```
64 bit sequence number || mic (64 bit sequence number || audit record)
```

安全的远程审计客户机/服务器通信流：

```
1) Client <--> Server - TCP handshake

2) Client <--> Server - protocol version negotiation:
   a) Client  --> Server - send data size - uint32_t value (2)
   b) Client  --> Server - send clear text message of the versions
                           supported comma separated, e.g.,
                           "01,02,03" for versions 1 and 2 and 3.
                           The only version supported at present is
                           "01"
   c) Client <--  Server - send data size - uint32_t value (2)
   d) Client <--  Server - send clear text version selected
                           ("01")
   :no version match; close connection; try next host

3) Security context initiation:
   a) Client - Construct channel bindings:
         initiator address type (GSS_C_AF_NULLADDR)
         acceptor address type (GSS_C_AF_NULLADDR)
         application data value (4 octets  "0101")
   b) Client  --> Server - send token (data) size - uint32_t value
   c) Client  --> Server - GSS-API per-context token
   d) Client <--  Server - send token (data) size
   e) Client <--  Server - GSS-API per-context token
      :repeat a-e until security context is initialized; if unsuccessful,
      close connection; try next host

4) Client - transmit thread, when audit record to be sent:
   a) Client  --> Server - send data size
   b) Client  --> Server - GSS-API per-message token
                  wrap (sequence number || audit record)
      :repeat a-b while less than max (qsize) outstanding records

 5) Client - receive thread:
    a) Client <--  Server - receive data size - uint32_t value
    b) Client <--  Server - receive sequence number - uint64_t value
    c) Client <--  Server - receive MIC
    d) Client             - MIC verification - OK
    e) Client             - remove particular audit record
```

```
                            pointed by the sequence number from the
                            retransmit buffer
  :repeat a-e, on error close connection; try next host;
   retransmit unacknowledged audit records

6) Server - receive thread:
    a) Client  --> Server - receive data size
    b) Client  --> Server - GSS-API receive, uwrap, store
                   per-message token

7) Server - transmit thread:
    a) Server - MIC generation - message integrity code
                    mic (sequence number || audit record)
    b) Client <--  Server - send data size
    c) Client < -- Server - send sequence number
    d) Client <--  Server - send MIC
```

示例

示例1 装入 audit_remote.so 并指定远程审计服务器

使用以下指令可装入 audit_remote.so 并指定要将审计记录发送到的远程审计服务器。kerberos_v5 安全机制被定义为在与服务器通信时使用。

```
auditconfig -setplugin audit_remote active \
    "p_timeout=90;p_retries=2;
     p_hosts=eggplant.eng.sun.com::kerberos_v5,
     purple.ebay.sun.com:4592:kerberos_v5"
```

示例2 使用缺省安全机制使用率的配置

以下示例说明了缺省安全机制使用率的配置。它还说明了如何在其中一台配置的服务器上使用缺省端口：

```
auditconfig -setplugin audit_remote active \
    "p_timeout=10;p_retries=2;
     p_hosts=jedger.eng.sun.com, jbadams.ebay.sun.com:4592"
```

示例3 内部插件队列大小设置

某些条件（例如，高峰或突发审计数据通信流量以及服务器和客户机之间线路通信速度缓慢）可能使 audit_remote 插件排队的未处理审计记录的数量达到配置的最大数量。以下示例说明了如何设置队列大小参数。

```
auditconfig -setplugin audit_remote "" 1000
```

属性

有关以下属性的说明，请参见 attributes(5)：

属性类型	属性值
MT 级别	MT-Safe（MT 安全）

属性类型	属性值
接口稳定性	请参见下文。

插件配置参数是 "Committed"（已确定）。客户机/服务器协议（版本 "01"）是 "Contracted Project Private"（合同项目专用）。有关审计记录格式和内容稳定性，请参见 audit.log(4)。

另请参见

auditd(1M)、auditconfig(1M)、audit_warn(1M)、getipnodebyname(3SOCKET)、getservbyna

《Oracle Solaris 11.1 管理：安全服务》

附注

audit_remote 通过 GSS-API (libgss(3LIB)) 向远程审计服务验证自己。使用 gss 实现机制（例如 Kerberos）提供的缺省 gss 凭证。

IANA 分配的 solaris-audit 服务端口为 16162。

引用名

audit_syslog – 提供从Solaris审计数据到系统日志消息的实时转换

用法概要

```
/usr/lib/security/audit_syslog.so
```

描述

用于Solaris审计的audit_syslog插件模块/usr/lib/security/audit_syslog.so提供Solaris审计数据到系统日志格式（文本）数据的实时转换，并按照syslog.conf(4)中的配置将数据发送到系统日志守护进程。该插件的路径通过auditconfig(1M)实用程序来指定。

如果通过auditconfig将plugin配置为活动状态，则会将消息写入到syslog。使用auditconfig -setplugin选项可以更改所有与插件相关的配置参数。将使用工具代码LOG_AUDIT（syslog.conf(4)中的audit）和严重性LOG_NOTICE生成系统日志消息。syslog审计消息包含从用于二进制审计日志的标记中选择的数据。（请参见audit.log(4)。）与所有syslog消息一样，syslog文件中的每一行都包括两部分，即syslog头和消息。

系统日志头包含生成消息的日期和时间，从中发送消息的主机名，指示消息是由审计守护进程生成的auditd，syslogd在内部使用的ID字段以及指示syslog工具和严重性值的audit.notice。syslog标以字符]结尾，即一个结束方括号和一个空格。

消息部分以头标记中的事件类型开头。仅当所有后续数据包含在原始审计记录中且1024字节的最大长度syslog行中有空间时，才会显示这些数据。在以下示例中，反斜杠(\)指示继续；实际syslog消息包含在一行中：

```
Oct 31 11:38:08 smothers auditd: [ID 917521 audit.notice] chdir(2) ok\
session 401 by joeuser as root:other from myultra obj /export/home
```

在上述示例中，chdir(2)是事件类型。此字段后是如下所述的其他数据。如果这些数据未包含在源审计记录中，将省略这些数据。

ok 或 failed　　来自return或exit标记。

session <#>　　<#>是subject标记中的会话ID。

by <*name*>　　<*name*>是subject标记中的审计ID。

as <*name*>:<*group*>　　<*name*>和<*group*>分别是subject标记中的是有效用户ID和有效组ID。

in <*zone name*>　　区域名称。仅在设置zonename审计策略时才会生成此字段。

from <*terminal*>　　<*terminal*>是subject标记中的文本计算机地址。

obj <*path*>　　<*path*>是path标记中的路径。为了使其适合行大小，可能会从左侧截断该路径。使用前导省略号(...)来指示截断。

proc_uid <*owner*>　　<*owner*>是进程所有者的有效用户ID。

proc_auid <*owner*>　　<*owner*>是进程所有者的审计ID。

以下是示例syslog消息：

```
Nov  4  8:27:07 smothers auditd: [ID 175219 audit.notice]
\system booted

Nov  4  9:28:17 smothers auditd: [ID 752191 audit.notice] \
login - rlogin ok session 401 by joeuser as joeuser:staff from myultra

Nov  4 10:29:27 smothers auditd: [ID 521917 audit.notice] \
access(2) ok session 255 by janeuser as janeuser:staff from  \
129.146.89.30 obj /etc/passwd
```

对象属性

p_flag 属性用于进一步过滤要发送到 syslog 守护进程的审计数据，该守护进程超出了通过 flags 和 naflags（请参见auditconfig(1M)）以及通过 user_attr(4) 的用户特定行指定的类范围。该参数是一个逗号分隔列表；每一项表示一个审计类（请参见 audit_class(4)）并使用 audit_flags(5) 中所述的语法进行指定。缺省设置（空 p_flags）表示将不生成任何审计记录。

示例

示例 1 plugin 行的一种用法

在下面所示的规范中，plugin（与设置 flags 和 naflags 结合使用）用于允许 lo 的类记录，但仅失败时才允许 am 的类记录。省略 fm 类记录会导致不输出任何 fm 类记录。pc 参数没有任何效果，因为您无法向通过 flags 和 naflags 以及由 user_attr(4) 定义的那些标志添加类。您只能删除它们。

```
auditconfig -setflags lo,am,fm
auditconfig -setnaflags lo
auditconfig -setplugin audit_syslog active "p_flags=lo,-am,pc"
```

示例 2 all 的用法

在下面所示的规范中，一种例外情况是，all 允许通过 flags 和 naflags（以及 user_attr(4)）定义的所有标志。该例外情况是 am 元类，它等同于 ss,as,ua，后者修改为可输出所有 ua 事件但仅限 ss 和 as 的失败事件。

```
auditconfig -setflags lo,am
auditconfig -setnaflags lo
auditconfig -setplugin audit_syslog active "p_flags=all,^+ss,^+as"
```

属性

有关以下属性的说明，请参见 attributes(5)：

属性类型	属性值
MT 级别	MT-Safe（MT 安全）
接口稳定性	请参见下文。

消息格式和消息内容是 "Uncommitted"（未确定）。配置参数是 "Committed"（已确定）。

另请参见

auditconfig(1M)、auditd(1M)、audit_class(4)、syslog.conf(4)、user_attr(4)、attributes(5)、

《Oracle Solaris 11.1 管理：安全服务》

附注

激活 audit_syslog plugin 要求 /etc/syslog.conf 配置为将工具 audit 和严重性 notice 或更高级别的 syslog 消息存储在适用于 Solaris 审计记录的文件中。syslog.conf 中此类行的一个示例为：

```
audit.notice                /var/audit/audit.log
```

syslog 中的消息将通过 UDP 发送到远程 syslog 服务器，这种方式并不能保证送达或确保按正确顺序收到消息。

如果使用为 plugin 行指定的参数导致未预选任何类，则将通过 syslog 警报以及 LOG_DAEMON 工具代码报告错误。

syslog 头中的时间字段由 syslog(3C) 生成，仅与二进制审计日志中给定的时间近似。通常，时间字段显示同一整秒，或者最多相差几秒。

引用名

brands – 非全局区域的备用操作环境

描述

标记区域 (Branded Zone, BrandZ) 框架扩展了 zones(5) 中介绍的 Solaris Zones 基础结构，在其中包含标记创建功能，这些标记提供包含非本机操作环境的非全局区域。

术语“标记”可以指各种操作环境。所有标记管理都通过扩展当前区域结构来执行。

每个区域都配有一个关联标记。标记类型用来确定安装和引导区域时执行的脚本。此外，区域的标记还可用来在应用程序启动时确定正确的应用程序类型。缺省标记由全局区域中安装的分发软件决定。

一个标记区域只支持一个非本地二进制命令标记，这意味着一个标记区域只提供一种操作环境。一旦为区域分配了标记，便不能更改或删除该标记。

BrandZ 通过以下方式来扩展区域工具：

- 标记是区域的一个属性，在创建区域时设置。
- zonecfg 工具（请参见 zonecfg(1M)）用于设置区域的标记类型并配置区域。
- zoneadm 工具（请参见 zoneadm(1M)）用于报告区域的标记类型并管理区域。

标记

对于 Oracle Solaris 11，可用标记包括：

- `solaris`

  请参见 solaris(5)。

- `solaris10`

  请参见 solaris10(5)。

- `labeled`

  实现 Trusted Extensions，请参见 txzonemgr(1M)。

设备支持

每个区域支持的设备都记录在该标记的手册页和其他文档中。区域基础结构检测任何尝试添加不支持设备的操作并向管理员发出警告。如果管理员不顾警告而选择添加不支持的设备，该设备可能无法如预期那样工作。该配置将会是未经过测试的，并且不受支持。

属性

有关以下属性的说明，请参见 attributes(5)：

属性类型	属性值
可用性	system/zones
接口稳定性	Committed（已确定）

另请参见

mdb(1)、zlogin(1)、zonename(1)、dtrace(1M)、in.rlogind(1M)、sshd(1M)、txzonemgr(1M)、

引用名

cancellation – overview of concepts related to POSIX thread cancellation

描述

FUNCTION	ACTION
`pthread_cancel()`	Cancels thread execution.
`pthread_setcancelstate()`	Sets the cancellation *state* of a thread.
`pthread_setcanceltype()`	Sets the cancellation *type* of a thread.
`pthread_testcancel()`	Creates a cancellation point in the calling thread.
`pthread_cleanup_push()`	Pushes a cleanup handler routine.
`pthread_cleanup_pop()`	Pops a cleanup handler routine.

Cancellation

Thread cancellation allows a thread to terminate the execution of any application thread in the process. Cancellation is useful when further operations of one or more threads are undesirable or unnecessary.

An example of a situation that could benefit from using cancellation is an asynchronously-generated cancel condition such as a user requesting to close or exit some running operation. Another example is the completion of a task undertaken by a number of threads, such as solving a maze. While many threads search for the solution, one of the threads might solve the puzzle while the others continue to operate. Since they are serving no purpose at that point, they should all be canceled.

Planning Steps

Planning and programming for most cancellations follow this pattern:

1. Identify which threads you want to cancel, and insert `pthread_cancel`(3C) statements.
2. Identify system-defined cancellation points where a thread that might be canceled could have changed system or program state that should be restored. See the `Cancellation Points` for a list.
3. When a thread changes the system or program state just before a cancellation point, and should restore that state before the thread is canceled, place a cleanup handler before the cancellation point with `pthread_cleanup_push`(3C). Wherever a thread restores the changed state, pop the cleanup handler from the cleanup stack with `pthread_cleanup_pop`(3C).
4. Know whether the threads you are canceling call into cancel-unsafe libraries, and disable cancellation with `pthread_setcancelstate`(3C) before the call into the library. See `Cancellation State` and `Cancel-Safe`.
5. To cancel a thread in a procedure that contains no cancellation points, insert your own cancellation points with `pthread_testcancel`(3C). This function creates cancellation points by testing for pending cancellations and performing those cancellations if they are found. Push and pop cleanup handlers around the cancellation point, if necessary (see Step 3, above).

Cancellation Points

The system defines certain points at which cancellation can occur (cancellation points), and you can create additional cancellation points in your application with pthread_testcancel().

The following cancellation points are defined by the system (system-defined cancellation points): creat(2), aio_suspend(3C), close(2), creat(2), getmsg(2), getpmsg(2), lockf(3C), mq_receive(3C), mq_send(3C), msgrcv(2), msgsnd(2), msync(3C), nanosleep(3C), open(2), pause(2), poll(2), pread(2), pthread_cond_timedwait(3C), pthread_cond_wait(3C), pthread_join(3C), pthread_testcancel(3C), putmsg(2), putpmsg(2), pwrite(2), read(2), readv(2), select(3C), sem_wait(3C), sigpause(3C), sigwaitinfo(3C), sigsuspend(2), sigtimedwait(3C), sigwait(2), sleep(3C), sync(2), system(3C), tcdrain(3C), usleep(3C), wait(3C), waitid(2), wait3(3C), waitpid(3C), write(2), writev(2), and fcntl(2), when specifying F_SETLKW as the command.

When cancellation is asynchronous, cancellation can occur at any time (before, during, or after the execution of the function defined as the cancellation point). When cancellation is deferred (the default case), cancellation occurs only within the scope of a function defined as a cancellation point (after the function is called and before the function returns). See Cancellation Type for more information about deferred and asynchronous cancellation.

Choosing where to place cancellation points and understanding how cancellation affects your program depend upon your understanding of both your application and of cancellation mechanics.

Typically, any call that might require a long wait should be a cancellation point. Operations need to check for pending cancellation requests when the operation is about to block indefinitely. This includes threads waiting in pthread_cond_wait() and pthread_cond_timedwait(), threads waiting for the termination of another thread in pthread_join(), and threads blocked on sigwait().

A mutex is explicitly not a cancellation point and should be held for only the minimal essential time.

Most of the dangers in performing cancellations deal with properly restoring invariants and freeing shared resources. For example, a carelessly canceled thread might leave a mutex in a locked state, leading to a deadlock. Or it might leave a region of memory allocated with no way to identify it and therefore no way to free it.

Cleanup Handlers

When a thread is canceled, it should release resources and clean up the state that is shared with other threads. So, whenever a thread that might be canceled changes the state of the system or of the program, be sure to push a cleanup handler with pthread_cleanup_push(3C) before the cancellation point.

When a thread is canceled, all the currently-stacked cleanup handlers are executed in last-in-first-out (LIFO) order. Each handler is run in the scope in which it was pushed. When the last cleanup handler returns, the thread-specific data destructor functions are called. Thread execution terminates when the last destructor function returns.

When, in the normal course of the program, an uncanceled thread restores state that it had previously changed, be sure to pop the cleanup handler (that you had set up where the change took place) using pthread_cleanup_pop(3C). That way, if the thread is canceled later, only currently-changed state will be restored by the handlers that are left in the stack.

The pthread_cleanup_push() and pthread_cleanup_pop() functions can be implemented as macros. The application must ensure that they appear as statements, and in pairs within the same lexical scope (that is, the pthread_cleanup_push() macro can be thought to expand to a token list whose first token is '{' with pthread_cleanup_pop() expanding to a token list whose last token is the corresponding '}').

The effect of the use of return, break, continue, and goto to prematurely leave a code block described by a pair of pthread_cleanup_push() and pthread_cleanup_pop() function calls is undefined.

Cancellation State

Most programmers will use only the default cancellation state of PTHREAD_CANCEL_ENABLE, but can choose to change the state by using pthread_setcancelstate(3C), which determines whether a thread is cancelable at all. With the default *state* of PTHREAD_CANCEL_ENABLE, cancellation is enabled and the thread is cancelable at points determined by its cancellation *type*. See Cancellation Type.

If the *state* is PTHREAD_CANCEL_DISABLE, cancellation is disabled, the thread is not cancelable at any point, and all cancellation requests to it are held pending.

You might want to disable cancellation before a call to a cancel-unsafe library, restoring the old cancel state when the call returns from the library. See Cancel-Safe for explanations of cancel safety.

Cancellation Type

A thread's cancellation type is set with pthread_setcanceltype(3C), and determines whether the thread can be canceled anywhere in its execution or only at cancellation points.

With the default *type* of PTHREAD_CANCEL_DEFERRED, the thread is cancelable only at cancellation points, and then only when cancellation is enabled.

If the *type* is PTHREAD_CANCEL_ASYNCHRONOUS, the thread is cancelable at any point in its execution (assuming, of course, that cancellation is enabled). Try to limit regions of asynchronous cancellation to sequences with no external dependencies that could result in dangling resources or unresolved state conditions. Using asynchronous cancellation is discouraged because of the danger involved in trying to guarantee correct cleanup handling at absolutely every point in the program.

Cancellation Type/State Table

Type	State	
	Enabled (Default)	Disabled

Cancellation Type/State Table

Deferred (Default)	Cancellation occurs when the target thread reaches a cancellation point and a cancel is pending. (Default)	All cancellation requests to the target thread are held pending.
Asynchronous	Receipt of a pthread_cancel() call causes immediate cancellation.	All cancellation requests to the target thread are held pending; as soon as cancellation is re-enabled, pending cancellations are executedimmediately.

Cancel-Safe

With the arrival of POSIX cancellation, the Cancel-Safe level has been added to the list of MT-Safety levels. See attributes(5). An application or library is Cancel-Safe whenever it has arranged for cleanup handlers to restore system or program state wherever cancellation can occur. The application or library is specifically Deferred-Cancel-Safe when it is Cancel-Safe for threads whose cancellation type is PTHREAD_CANCEL_DEFERRED. See Cancellation State. It is specifically Asynchronous-Cancel-Safe when it is Cancel-Safe for threads whose cancellation type is PTHREAD_CANCEL_ASYNCHRONOUS.

It is easier to arrange for deferred cancel safety, as this requires system and program state protection only around cancellation points. In general, expect that most applications and libraries are not Asynchronous-Cancel-Safe.

POSIX Threads Only

The cancellation functions described in this manual page are available for POSIX threads, only (the Solaris threads interfaces do not provide cancellation functions).

示例

示例 1 Cancellation example

The following short C++ example shows the pushing/popping of cancellation handlers, the disabling/enabling of cancellation, the use of pthread_testcancel(), and so on. The free_res() cancellation handler in this example is a dummy function that simply prints a message, but that would free resources in a real application. The function f2() is called from the main thread, and goes deep into its call stack by calling itself recursively.

Before f2() starts running, the newly created thread has probably posted a cancellation on the main thread since the main thread calls thr_yield() right after creating thread2. Because cancellation was initially disabled in the main thread, through a call to pthread_setcancelstate(), the call to f2() from main() continues and constructs X at each recursive call, even though the main thread has a pending cancellation.

When f2() is called for the fifty-first time (when "i == 50"), f2() enables cancellation by calling pthread_setcancelstate(). It then establishes a cancellation point for itself by calling pthread_testcancel(). (Because a cancellation is pending, a call to a cancellation point such as read(2) or write(2) would also cancel the caller here.)

示例 1 Cancellation example （续）

After the main() thread is canceled at the fifty-first iteration, all the cleanup handlers that were pushed are called in sequence; this is indicated by the calls to free_res() and the calls to the destructor for *X*. At each level, the C++ runtime calls the destructor for *X* and then the cancellation handler, free_res(). The print messages from free_res() and *X*'s destructor show the sequence of calls.

At the end, the main thread is joined by thread2. Because the main thread was canceled, its return status from pthread_join() is PTHREAD_CANCELED. After the status is printed, thread2 returns, killing the process (since it is the last thread in the process).

```
#include <pthread.h>
#include <sched.h>
extern "C" void thr_yield(void);

extern "C" void printf(...);

struct X {
        int x;
        X(int i){x = i; printf("X(%d) constructed.\n", i);}
        ~X(){ printf("X(%d) destroyed.\n", x);}
};

void
free_res(void *i)
{
        printf("Freeing ‘%d‘\n",i);
}

char* f2(int i)
{
        try {
        X dummy(i);
        pthread_cleanup_push(free_res, (void *)i);
        if (i == 50) {
               pthread_setcancelstate(PTHREAD_CANCEL_ENABLE, NULL);
            pthread_testcancel();
        }
        f2(i+1);
        pthread_cleanup_pop(0);
        }
        catch (int) {
        printf("Error: In handler.\n");
        }
        return "f2";
}
```

示例 1　Cancellation example　（续）

```
void *
thread2(void *tid)
{
        void *sts;

        printf("I am new thread :%d\n", pthread_self());

        pthread_cancel((pthread_t)tid);

        pthread_join((pthread_t)tid, &sts);

        printf("main thread cancelled due to %d\n", sts);

        return (sts);
}

main()
{
        pthread_setcancelstate(PTHREAD_CANCEL_DISABLE, NULL);
        pthread_create(NULL, NULL, thread2, (void *)pthread_self());
        thr_yield();
        printf("Returned from %s\n",f2(0));
}
```

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT-Level	MT-Safe

另请参见

read(2), sigwait(2), write(2), Intro(3), condition(5), pthread_cleanup_pop(3C), pthread_cleanup_push(3C), pthread_exit(3C), pthread_join(3C), pthread_setcancelstate(3C), pthread_setcanceltype(3C), pthread_testcancel(3C), setjmp(3C), attributes(5), standards(5)

引用名

charmap – character set description file

描述

A character set description file or *charmap* defines characteristics for a coded character set. Other information about the coded character set may also be in the file. Coded character set character values are defined using symbolic character names followed by character encoding values.

The character set description file provides:

- The capability to describe character set attributes (such as collation order or character classes) independent of character set encoding, and using only the characters in the portable character set. This makes it possible to create generic localedef(1) source files for all codesets that share the portable character set.
- Standardized symbolic names for all characters in the portable character set, making it possible to refer to any such character regardless of encoding.

Symbolic Names

Each symbolic name is included in the file and is mapped to a unique encoding value (except for those symbolic names that are shown with identical glyphs). If the control characters commonly associated with the symbolic names in the following table are supported by the implementation, the symbolic names and their corresponding encoding values are included in the file. Some of the encodings associated with the symbolic names in this table may be the same as characters in the portable character set table.

<ACK>	<DC2>	<ENQ>	<FS>	<IS4>	<SOH>
<BEL>	<DC3>	<EOT>	<GS>	<LF>	<STX>
<BS>	<DC4>	<ESC>	<HT>	<NAK>	<SUB>
<CAN>	<DEL>	<ETB>	<IS1>	<RS>	<SYN>
<CR>	<DLE>	<ETX>	<IS2>	<SI>	<US>
<DC1>	<EM>	<FF>	<IS3>	<SO>	<VT>

Declarations

The following declarations can precede the character definitions. Each must consist of the symbol shown in the following list, starting in column 1, including the surrounding brackets, followed by one or more blank characters, followed by the value to be assigned to the symbol.

<code_set_name> The name of the coded character set for which the character set description file is defined.

<mb_cur_max> The maximum number of bytes in a multi-byte character. This defaults to 1.

<mb_cur_min> An unsigned positive integer value that defines the minimum number of bytes in a character for the encoded character set.

<escape_char> The escape character used to indicate that the characters following will be interpreted in a special way, as defined later in this section. This defaults to backslash ('\'), which is the character glyph used in all the following text and examples, unless otherwise noted.

<comment_char> The character that when placed in column 1 of a charmap line, is used to indicate that the line is to be ignored. The default character is the number sign (#).

Format

The character set mapping definitions will be all the lines immediately following an identifier line containing the string `CHARMAP` starting in column 1, and preceding a trailer line containing the string `END CHARMAP` starting in column 1. Empty lines and lines containing a *<comment_char>* in the first column will be ignored. Each non-comment line of the character set mapping definition, that is, between the `CHARMAP` and `END CHARMAP` lines of the file), must be in either of two forms:

```
"%s %s %s\n",<symbolic-name>,<encoding>,<comments>
```

or

```
"%s...%s %s %s\n",<symbolic-name>,<symbolic-name>, <encoding>,\
          <comments>
```

In the first format, the line in the character set mapping definition defines a single symbolic name and a corresponding encoding. A character following an escape character is interpreted as itself; for example, the sequence ”<\\\>>“ represents the symbolic name “\>” enclosed between angle brackets.

In the second format, the line in the character set mapping definition defines a range of one or more symbolic names. In this form, the symbolic names must consist of zero or more non-numeric characters, followed by an integer formed by one or more decimal digits. The characters preceding the integer must be identical in the two symbolic names, and the integer formed by the digits in the second symbolic name must be equal to or greater than the integer formed by the digits in the first name. This is interpreted as a series of symbolic names formed from the common part and each of the integers between the first and the second integer, inclusive. As an example, `<j0101>...<j0104>` is interpreted as the symbolic names `<j0101>`, `<j0102>`, `<j0103>`, and `<j0104>`, in that order.

A character set mapping definition line must exist for all symbolic names and must define the coded character value that corresponds to the character glyph indicated in the table, or the coded character value that corresponds with the control character symbolic name. If the control characters commonly associated with the symbolic names are supported by the implementation, the symbolic name and the corresponding encoding value must be included in the file. Additional unique symbolic names may be included. A coded character value can be represented by more than one symbolic name.

The encoding part is expressed as one (for single-byte character values) or more concatenated decimal, octal or hexadecimal constants in the following formats:

"%cd%d",*<escape_char>*,*<decimal byte value>*

"%cx%x",*<escape_char>*,*<hexadecimal byte value>*

"%c%o",*<escape_char>*,*<octal byte value>*

Decimal Constants

Decimal constants must be represented by two or three decimal digits, preceded by the escape character and the lower-case letter d; for example, \d05, \d97, or \d143. Hexadecimal constants must be represented by two hexadecimal digits, preceded by the escape character and the lower-case letter x; for example, \x05, \x61, or \x8f. Octal constants must be represented by two or three octal digits, preceded by the escape character; for example, \05, \141, or \217. In a portable charmap file, each constant must represent an 8-bit byte. Implementations supporting other byte sizes may allow constants to represent values larger than those that can be represented in 8-bit bytes, and to allow additional digits in constants. When constants are concatenated for multi-byte character values, they must be of the same type, and interpreted in byte order from first to last with the least significant byte of the multi-byte character specified by the last constant.

Ranges of Symbolic Names

In lines defining ranges of symbolic names, the encoded value is the value for the first symbolic name in the range (the symbolic name preceding the ellipsis). Subsequent symbolic names defined by the range will have encoding values in increasing order. Bytes are treated as unsigned octets and carry is propagated between the bytes as necessary to represent the range. However, because this causes a null byte in the second or subsequent bytes of a character, such a declaration should not be specified. For example, the line

```
<j0101>...<j0104>     \d129\d254
```

is interpreted as:

```
<j0101>                 \d129\d254
<j0102>                 \d129\d255
<j0103>                 \d130\d00
<j0104>                 \d130\d01
```

The expanded declaration of the symbol <j0103> in the above example is an invalid specification, because it contains a null byte in the second byte of a character.

The comment is optional.

Width Specification

The following declarations can follow the character set mapping definitions (after the "END CHARMAP" statement). Each consists of the keyword shown in the following list, starting in column 1, followed by the value(s) to be associated to the keyword, as defined below.

WIDTH

A non-negative integer value defining the column width for the printable character in the coded character set mapping definitions. Coded character set character values are defined using symbolic character names followed by column width values. Defining a character with more than one WIDTH produces undefined results. The END WIDTH keyword is used to terminate the WIDTH definitions. Specifying the width of a non-printable character in a WIDTH declaration produces undefined results.

WIDTH_DEFAULT — A non-negative integer value defining the default column width for any printable character not listed by one of the WIDTH keywords. If no WIDTH_DEFAULT keyword is included in the charmap, the default character width is 1.

Example:

After the “END CHARMAP” statement, a syntax for a width definition would be:

```
WIDTH
<A>             1
<B>             1
<C>...<Z>       1
...
<fool>...<foon> 2
...
END WIDTH
```

In this example, the numerical code point values represented by the symbols <A> and <B> are assigned a width of 1. The code point values < C> to <Z> inclusive, that is, <C>, <D>, <E>, and so on, are also assigned a width of 1. Using <A>. . .<Z> would have required fewer lines, but the alternative was shown to demonstrate flexibility. The keyword WIDTH_DEFAULT could have been added as appropriate.

另请参见

locale(1), localedef(1), nl_langinfo(3C), extensions(5), locale(5)

引用名

condition – concepts related to condition variables

描述

Occasionally, a thread running within a mutex needs to wait for an event, in which case it blocks or sleeps. When a thread is waiting for another thread to communicate its disposition, it uses a condition variable in conjunction with a mutex. Although a mutex is exclusive and the code it protects is sharable (at certain moments), condition variables enable the synchronization of differing events that share a mutex, but not necessarily data. Several condition variables may be used by threads to signal each other when a task is complete, which then allows the next waiting thread to take ownership of the mutex.

A condition variable enables threads to atomically block and test the condition under the protection of a mutual exclusion lock (mutex) until the condition is satisfied. If the condition is false, a thread blocks on a condition variable and atomically releases the mutex that is waiting for the condition to change. If another thread changes the condition, it may wake up waiting threads by signaling the associated condition variable. The waiting threads, upon awakening, reacquire the mutex and re-evaluate the condition.

Initialize

Condition variables and mutexes should be global. Condition variables that are allocated in writable memory can synchronize threads among processes if they are shared by the cooperating processes (see `mmap(2)`) and are initialized for this purpose.

The scope of a condition variable is either intra-process or inter-process. This is dependent upon whether the argument is passed implicitly or explicitly to the initialization of that condition variable. A condition variable does not need to be explicitly initialized. A condition variable is initialized with all zeros, by default, and its scope is set to within the calling process. For inter-process synchronization, a condition variable must be initialized once, and only once, before use.

A condition variable must not be simultaneously initialized by multiple threads or re-initialized while in use by other threads.

Condition variables attributes may be set to the default or customized at initialization. POSIX threads even allow the default values to be customized. Establishing these attributes varies depending upon whether POSIX or Solaris threads are used. Similar to the distinctions between POSIX and Solaris thread creation, POSIX condition variables implement the default, intra-process, unless an attribute object is modified for inter-process prior to the initialization of the condition variable. Solaris condition variables also implement as the default, intra-process; however, they set this attribute according to the argument, *type*, passed to their initialization function.

Condition Wait

The condition wait interface allows a thread to wait for a condition and atomically release the associated mutex that it needs to hold to check the condition. The thread waits for another thread to make the condition true and that thread's resulting call to signal and wakeup the waiting thread.

Condition Signaling

A condition signal allows a thread to unblock the next thread waiting on the condition variable, whereas, a condition broadcast allows a thread to unblock all threads waiting on the condition variable.

Destroy

The condition destroy functions destroy any state, but not the space, associated with the condition variable.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT-Level	MT-Safe

另请参见

fork(2), mmap(2), setitimer(2), shmop(2), cond_broadcast(3C), cond_destroy(3C), cond_init(3C), cond_signal(3C), cond_timedwait(3C), cond_wait(3C), pthread_cond_broadcast(3C), pthread_cond_destroy(3C), pthread_cond_init(3C), pthread_cond_signal(3C), pthread_cond_timedwait(3C), pthread_cond_wait(3C), pthread_condattr_init(3C), signal(3C), attributes(5), mutex(5), standards(5)

附注

If more than one thread is blocked on a condition variable, the order in which threads are unblocked is determined by the scheduling policy.

USYNC_THREAD does not support multiple mapplings to the same logical synch object. If you need to mmap() a synch object to different locations within the same address space, then the synch object should be initialized as a shared object USYNC_PROCESS for Solaris, and PTHREAD_PROCESS_PRIVATE for POSIX.

引用名

crypt_bsdbf – password hashing module using Blowfish cryptographic algorithm

用法概要

```
/usr/lib/security/$ISA/crypt_bsdbf.so
```

描述

The crypt_bsdbf module is a one-way password hashing module for use with crypt(3C) that uses the Blowfish cryptographic algorithm. The algorithm identifier for crypt.conf(4) and policy.conf(4) is 2a.

The maximum password length for crypt_bsdbf is 72 characters.

The default number of rounds is log(4). This can be increased by updating /etc/security/crypt.conf, for example, for log(12) rounds:

```
2a crypt_bsdbf.so.1    12
```

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT-Level	Safe

另请参见

passwd(1), crypt(3C), crypt_genhash_impl(3C), crypt_gensalt(3C), crypt_gensalt_impl(3C), getpassphrase(3C), crypt.conf(4), passwd(4), policy.conf(4), attributes(5)

引用名

crypt_bsdmd5 – password hashing module using MD5 message hash algorithm

用法概要

```
/usr/lib/security/$ISA/crypt_bsdmd5.so
```

描述

The crypt_bsdmd5 module is a one-way password hashing module for use with crypt(3C) that uses the MD5 message hash algorithm. The algorithm identifier for crypt.conf(4) and policy.conf(4) is 1. The output is compatible with md5crypt on BSD and Linux systems.

The maximum password length for crypt_bsdmd5 is 255 characters.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT-Level	Safe

另请参见

passwd(1), crypt(3C), crypt_genhash_impl(3C), crypt_gensalt(3C), crypt_gensalt_impl(3C), getpassphrase(3C), crypt.conf(4), passwd(4), policy.conf(4), attributes(5)

引用名

crypt_sha256 – password hashing module using SHA–256 message hash algorithm

用法概要

```
/usr/lib/security/$ISA/crypt_sha256.so
```

描述

The crypt_sha256 module is a one-way password hashing module for use with crypt(3C) that uses the SHA–256 message hash algorithm. The algorithm identifier for crypt.conf(4) and policy.conf(4) is 5.

This module is designed to make it difficult to crack passwords that use brute force attacks based on high speed SHA–256 implementations that use code inlining, unrolled loops, and table lookup.

The maximum password length for crypt_sha256 is 255 characters.

The following options can be passed to the module by means of crypt.conf(4):

rounds=<*positive_number*>

Specifies the number of rounds of SHA-256 to use in generation of the salt; the default number of rounds is 5000. Negative values have no effect and are ignored. The minimum number of rounds cannot be below 1000.

The number of additional rounds is stored in the salt string returned by crypt_gensalt(3C). For example:

```
$5,rounds=6000$nlxmTTpz$
```

When crypt_gensalt(3C) is being used to generate a new salt, if the number of additional rounds configured in crypt.conf(4) is greater than that in the old salt, the value from crypt.conf(4) is used instead. This allows for migration to stronger (but more time-consuming) salts on password change.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT-Level	Safe

另请参见

passwd(1), crypt(3C), crypt_genhash_impl(3C), crypt_gensalt(3C), crypt_gensalt_impl(3C), getpassphrase(3C), crypt.conf(4), passwd(4), policy.conf(4), attributes(5)

引用名

crypt_sha512 – password hashing module using SHA–512 message hash algorithm

用法概要

```
/usr/lib/security/$ISA/crypt_sha512.so
```

描述

The crypt_sha512 module is a one-way password hashing module for use with crypt(3C) that uses the SHA–512 message hash algorithm. The algorithm identifier for crypt.conf(4) and policy.conf(4) is 6.

This module is designed to make it difficult to crack passwords that use brute force attacks based on high speed SHA–512 implementations that use code inlining, unrolled loops, and table lookup.

The maximum password length for crypt_sha512 is 255 characters.

The following options can be passed to the module by means of crypt.conf(4):

rounds=<*positive_number*>

Specifies the number of rounds of SHA-512 to use in generation of the salt; the default number of rounds is 5000. Negative values have no effect and are ignored. The minimum number of rounds cannot be below 1000.

The number of additional rounds is stored in the salt string returned by crypt_gensalt(3C). For example:

```
$6,rounds=6000$nlxmTTpz$
```

When crypt_gensalt(3C) is being used to generate a new salt, if the number of additional rounds configured in crypt.conf(4) is greater than that in the old salt, the value from crypt.conf(4) is used instead. This allows for migration to stronger (but more time-consuming) salts on password change.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT-Level	Safe

另请参见

passwd(1), crypt(3C), crypt_genhash_impl(3C), crypt_gensalt(3C), crypt_gensalt_impl(3C), getpassphrase(3C), crypt.conf(4), passwd(4), policy.conf(4), attributes(5)

引用名

crypt_sunmd5 – password hashing module using MD5 message hash algorithm

用法概要

```
/usr/lib/security/$ISA/crypt_sunmd5.so
```

描述

The crypt_sunmd5 module is a one-way password hashing module for use with crypt(3C) that uses the MD5 message hash algorithm. The algorithm identifier for crypt.conf(4) and policy.conf(4) is md5.

This module is designed to make it difficult to crack passwords that use brute force attacks based on high speed MD5 implementations that use code inlining, unrolled loops, and table lookup.

The maximum password length for crypt_sunmd5 is 255 characters.

The following options can be passed to the module by means of crypt.conf(4):

rounds=<*positive_number*>
: Specifies the number of additional rounds of MD5 to use in generation of the salt; the default number of rounds is 4096. Negative values have no effect and are ignored, that is, the number of rounds cannot be lowered below 4096.

  The number of additional rounds is stored in the salt string returned by crypt_gensalt(3C). For example:

  ```
  $md5,rounds=1000$nlxmTTpz$
  ```

  When crypt_gensalt(3C) is being used to generate a new salt, if the number of additional rounds configured in crypt.conf(4) is greater than that in the old salt, the value from crypt.conf(4) is used instead. This allows for migration to stronger (but more time-consuming) salts on password change.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT-Level	Safe

另请参见

passwd(1), crypt(3C), crypt_genhash_impl(3C), crypt_gensalt(3C), crypt_gensalt_impl(3C), getpassphrase(3C), crypt.conf(4), passwd(4), policy.conf(4), attributes(5)

引用名

crypt_unix – traditional UNIX crypt algorithm

描述

The crypt_unix algorithm is the traditional UNIX crypt algorithm. It is not considered sufficiently secure for current systems and is provided for backwards compatibility. The crypt_sunmd5(5), crypt_bsdmd5(5), or crypt_bsdbf(5) algorithm should be used instead.

The algorithm identifier for policy.conf(4) is __unix__. There is no entry in crypt.conf(4) for this algorithm.

The crypt_unix algorithm is internal to libc and provides the string encoding function used by crypt(3C) when the first character of the salt is not a "$".

This algorithm is based on a one-way encryption algorithm with variations intended (among other things) to frustrate use of hardware implementations of a key search. Only the first eight characters of the key passed to crypt() are used with this algorithm; the rest are silently ignored. The salt is a two-character string chosen from the set [a-zA-Z0-9./]. This string is used to perturb the hashing algorithm in one of 4096 different ways.

The maximum password length for crypt_unix is 8 characters.

用法

The return value of the crypt_unix algorithm might not be portable among standard-conforming systems. See standards(5).

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT-Level	Safe

另请参见

passwd(1), crypt(3C), crypt_genhash_impl(3C), crypt_gensalt(3C), crypt_gensalt_impl(3C), getpassphrase(3C), crypt.conf(4), passwd(4), policy.conf(4), attributes(5), crypt_bsdbf(5), crypt_bsdmd5(5), crypt_sunmd5(5), standards(5)

引用名

datasets – Oracle Solaris ZFS dataset organization

描述

Oracle Solaris is installed to a hierarchical set of ZFS datasets with well-defined names. The ZFS datasets, composed of both file systems and volumes, are stored on a bootable ZFS storage pool in the global zone. Snapshots are also ZFS datasets, but will not be discussed here.

The datasets are created automatically by one of the installation tools: the automated installer, the text installer, or the GUI installer. For non-global zones, the datasets are created by the zone brand during the zone installation and might differ from the hierarchy described here, which applies to the `solaris(5)` brand and the global zone.

rpool Dataset

`rpool` is the default name of the top-level dataset in the bootable ZFS storage pool and is mounted at `/rpool`. Bootable ZFS storage pools have certain restrictions over normal ZFS storage pools. These restrictions are discussed in the “Oracle Solaris Administration: ZFS File Systems” section of the Oracle Solaris 11 Information Library.

Administrators can create datasets anywhere beneath the top-level dataset, but those created within the '`ROOT/be-<name>`' dataset are treated specially by `beadm(1M)` and are discussed later.

ROOT Dataset

This is a container dataset which is not mounted by Solaris during operation. It collects sets of related datasets, each set forming a ZFS Boot Environment (BE). A BE is created on installation. Additional BEs are created by `pkg(1)` and can be modified, created, or destroyed by `beadm`.

In the global zone, the ROOT dataset must be stored within a bootable ZFS storage pool, called '`rpool`' by default. For non-global zones, the ROOT dataset is stored in a dataset with the leaf name '`rpool`' beneath the zonepath of each zone.

The list below shows the datasets that are created beneath the ROOT dataset. This list uses the default storage pool name.

`rpool/ROOT/solaris`

This is the default dataset that is created for the first boot environment installed on the system. It contains the root file system and is mounted on '/'.

The name of the leaf component of this dataset corresponds to the name of the BE that is associated with it. Any datasets created by the administrator as children of the '`ROOT/<be-name>`' dataset will be associated with that BE, and will be created, cloned, destroyed and have snapshots taken of them as part of the BE by `beadm`. New datasets created beneath '`ROOT/<be-name>`' should have the zfs property '`canmount=noauto`' set so that the datasets are mounted by `beadm` when appropriate.

All packaged content that is part of the image in a boot environment must fully reside within the datasets belonging to the boot environment.

`rpool/ROOT/solaris/var`

This is a dataset associated with each boot environment that contains the `/var` file system for the BE.

Dump Dataset

rpool/dump is a ZFS volume created during installation and is used as the default dump device for the system crash dump facility. This dataset is shared across all BEs. The crash dump facility can be managed with dumpadm(1M) and the dump device can be resized using zfs(1M). The rpool/dump dataset is not present on non-global zones. Other devices can be used instead of rpool/dump.

Swap Dataset

rpool/swap is a ZFS zvol created during installation and is used as the default swap area for the system. This dataset is shared across all BEs. Use swap(1M) to add or remove swap volumes. Use zfs to resize swap volumes. The rpool/swap dataset is not present on non-global zones. Other swap volumes can be used instead, as well as rpool/swap.

rpool/VARSHARE

rpool/VARSHARE is a dataset created on installation or upgrade, and is not part of any one BE. Instead, it is shared across all BEs and is mounted automatically by that BE during boot. It contains unpackaged content from /var. For example, audit information is stored in /var/audit, core files are stored in /var/cores, saved crash dumps are stored in /var/crash, and system mail is stored in /var/mail.

rpool/export/home

rpool/export/home is a dataset created on installation which can be used for user home directories. It is the default location for user home directories if the -d option is not passed to useradd(1M) to specify a different directory. If the user home directory does not already exist and the parent directory is the mount point of a ZFS dataset (for example, rpool/export/home), then a child of that dataset will be created and mounted at the specified location. These datasets are shared across all BEs.

另请参见

beadm(1M), dumpadm(1M), swap(1M), useradd(1M), zfs(1M), zoneadm(1M), zpool(1M), brands(5), filesystem(5), pkg(5), solaris(5)

引用名

device_clean – device clean programs

描述

Each allocatable device has a device clean program associated with it. Device clean programs are invoked by deallocate(1) to clean device states, registers, and any residual information in the device before the device is allocated to a user. Such cleaning is required by the object reuse policy.

Use list_devices(1) to obtain the names and types of allocatable devices as well as the cleaning program and the authorizations that are associated with each device.

On a system configured with Trusted Extensions, device clean programs are also invoked by allocate(1), in which case the program can optionally mount appropriate media for the caller.

The following device clean programs reside in `/etc/security/lib`.

`audio_clean`	audio devices
`st_clean`	tape devices
`sr_clean`	CD-ROM devices

On a system configured with Trusted Extensions, the following additional cleaning programs and wrappers are available.

`disk_clean`	CD-ROM and other removable media devices. This program mounts the device during the execution of `allocate`, if required.
`audio_clean_wrapper`	wrapper to make audio_clean work with CDE
`wdwwrapper`	wrapper to make other cleaning programs work with CDE
`wdwmsg`	CDE dialog boxes for cleaning programs

Administrators can create device clean programs for their sites. These programs must adhere to the syntax described below.

```
/etc/security/lib/device-clean-program [−i | −f | −s | −I] \
−m mode −u user-name −z zone-name −p zone-path device-name
```

where:

device-name The name of the device that is to be cleaned. Use `list_devices` to obtain the list of allocatable devices.

`-i` Invoke boot-time initialization.

`-f` Force cleanup by the administrator.

`-s` Invoke standard cleanup by the user.

`-I` Same as `-i`, with no error or warning.

The following options are supported only when the system is configured with Trusted Extensions.

-m *mode* Specify the mode in which the clean program is invoked. Valid values are allo- cate and deallocate. The default mode is allocate.

-u *user-name* Specify the name of user who executes the device clean program. The default user is the caller.

-z *zone-name* Specify the name of the zone in which the device is to be allocated or deallocated. The default zone is the global zone.

-p *zone-path* Establish the root path of the zone that is specified by *zone-name*. Default is "/".

退出状态

The following exit values are returned:

`0`
Successful completion.

`1`
An error. Caller can place device in error state.

`2`
A system error. Caller can place device in error state.

On a system configured with Trusted Extensions, the following additional exit values are returned:

`3`
Mounting of device failed. Caller shall not place device in error state.

`4`
Mounting of device succeeded.

文件

`/etc/security/lib/*` device clean programs

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	See below.

The Invocation is Uncommitted. The Output is Not-an-interface.

另请参见

allocate(1), deallocate(1), list_devices(1), attributes(5)

《Oracle Solaris 11.1 管理：安全服务》

引用名

dhcp – 动态主机配置协议

描述

通过动态主机配置协议 (Dynamic Host Configuration Protocol, DHCP)，可以在 TCP/IP 网络中的主机系统引导时为该网络自动配置这些系统。DHCP 使用客户机/服务器机制：服务器存储客户机的配置信息，并在客户机请求时提供该信息。该信息可以包含客户机的 IP 地址，以及有关客户机可使用的网络服务的信息。

本手册页提供 Oracle Solaris DHCP 实现的简短摘要。

Oracle Solaris DHCP 客户机

Oracle Solaris DHCP 客户机作为后台守护进程 dhcpagent(1M) 来实现。

对于 IPv4，如果 /etc 中至少存在一个 dhcp.*interface* 文件，此守护进程在引导期间将自动启动。仅具有相应 /etc/dhcp. *interface* 文件的接口才会在引导期间自动配置。

对于 IPv6，此守护进程将在 in.ndpd（基于 IPv6 路由通告消息）发出命令时自动启动。不需要任何 /etc/dhcp.*interface* 文件，但此类文件可用于将某个接口指定为“主接口”，条件是 IPv4 DHCP 也在使用中。

将通过使用 dhcpinfo(1) 命令从守护进程收到的信息中提取引导期间系统配置所需的网络参数。可通过更改 /etc/default/dhcpagent 文件中的可调参数来更改守护进程的缺省行为。守护进程由 ifconfig(1M)实用程序控制。使用 netstat(1M) 和 ifconfig(1M) 命令检查守护进程的状态。

Oracle Solaris DHCP 服务器

Oracle Solaris DHCP 服务器作为后台守护进程 in.dhcpd(1M) 来实现。此守护进程可以将网络配置信息传输到 BOOTP 或 DHCP 客户机。Oracle Solaris DHCP 服务可使用 dhcpmgr(1M) GUI 进行管理，或使用命令行实用程序 dhcpconfig(1M)、dhtadm(1M) 和 pntadm(1M) 进行管理。

Oracle Solaris DHCP 服务器已过时。改用 ISC DHCP 服务器。请参见 usr/share/man/man5/isc-dhcp.5

DHCP 配置表

Oracle Solaris DHCP 服务器将客户机配置信息存储在以下两种类型的表中：

dhcptab 表

包含宏和选项（也称为符号），用于构建要发送到每个 DHCP 客户机的配置信息包。DHCP 服务仅存在一个 dhcptab。可使用 dhtadm(1M) 命令或 dhcpmgr(1M) 图形实用程序查看和修改 dhcptab(4)。有关 dhcptab 记录的语法的更多信息，请参见 dhcptab(4)。有关 DHCP 选项和符号的更多信息，请参见 dhcp_inittab(4)。

DHCP 网络表

DHCP 网络表，其中包含客户机 ID 到 IP 地址的映射和与这些地址关联的参数。网络表以网络的 IP 地址命名，可以使用 pntadm 命令或 dhcpmgr 图形实用程序创建、查看和修改。有关网络表的更多信息，请参见 dhcp_network(4)。

另请参见

dhcpinfo(1)、dhcpagent(1M)、dhcpconfig(1M)、dhcpmgr(1M)、dhtadm(1M)、ifconfig(1M)、

usr/share/man/man5/isc-dhcp.5

《Oracle Solaris DHCP Service Developer's Guide》

由 Alexander, S. 和 R. Droms. 合著的《DHCP Options and BOOTP Vendor Extensions》，RFC 2132。Silicon Graphics, Inc. 与 Bucknell University 出版，1997 年 3 月。

由 Droms, R. 编著的《Interoperation Between DHCP and BOOTP》，RFC 1534。Bucknell University 出版。1993 年 10 月。

由 Droms, R. 编著的《Dynamic Host Configuration Protocol》，RFC 2131。Bucknell University 出版。1997 年 3 月。

由 Wimer, W. 编著的《Clarifications and Extensions for the Bootstrap Protocol》，RFC 1542。Carnegie Mellon University 出版。1993 年 10 月。

由 Lemon, T. 和 B. Sommerfeld. 合著的《Node-specific Client Identifiers for Dynamic Host Configuration Protocol Version Four (DHCPv4)》，RFC 4361。Nominum 和 Sun Microsystems 出版。2006 年 2 月。

由 Droms, R. 编著的《Dynamic Host Configuration Protocol for IPv6 (DHCPv6)》，RFC 3315。Cisco Systems 出版。2003 年 7 月。

引用名

dhcp_modules – data storage modules for the DHCP service

描述

This man page describes the characteristics of data storage modules (public modules) for use by the Solaris Dynamic Host Configuration Protocol (DHCP) service.

Public modules are the part of the DHCP service architecture that encapsulate the details of storing DHCP service data in a data storage service. Examples of data storage services are Oracle and ufs file systems.

Public modules are dynamic objects which can be shipped separately from the Solaris DHCP service. Once installed, a public module is visible to the DHCP service, and can be selected for use by the service through the DHCP service management interfaces (dhcpmgr(1M), dhcpconfig(1M), dhtadm(1M), and pntadm(1M)).

Public modules may be provided by Sun Microsystems, Inc or by third parties.

The Solaris DHCP service management architecture provides a mechanism for plugging in public module-specific administration functionality into the dhcpmgr(1M) and dhcpconfig(1M) utilities. This functionality is in the form of a Java Bean, which is provided by the public module vendor. This Java Bean collects public module-specific configuration from the user (you) and provides it to the Solaris DHCP service.

The Solaris DHCP service bundles three modules with the service, which are described below. There are three dhcpsvc.conf(4) DHCP service configuration parameters pertaining to public modules: RESOURCE, PATH, and RESOURCE_CONFIG. See dhcpsvc.conf(4) for more information about these parameters.

SUNWfiles

This module stores its data in ASCII files. Although the format is ASCII, hand-editing is discouraged. It is useful for DHCP service environments that support several hundred to a couple thousand of clients and lease times are a few hours or more.

This module's data may be shared between DHCP servers through the use of NFS.

SUNWbinfiles

This module stores its data in binary files. It is useful for DHCP service environments with many networks and many thousands of clients. This module provides an order of magnitude increase in performance and capacity over SUNWfiles.

This module's data cannot be shared between DHCP servers.

另请参见

crontab(1), dhcpconfig(1M), dhcpmgr(1M), dhtadm(1M), pntadm(1M), dhcpsvc.conf(4), dhcp(5)

《Oracle Solaris DHCP Service Developer’s Guide》

引用名

environ – 用户环境

描述

当进程开始执行时，exec 函数系列中的一个函数将激活一个字符串数组，该字符串数组被称为环境；请参见 exec(2)。根据约定，这些字符串的形式为 *variable=value*。例如，PATH=/sbin:/usr/sbin。这些环境变量提供了使程序环境相关信息可用于程序的方法。

可以通过 sh(1) 中的 export 命令和 *name=value* 参数或某一个 exec 函数将名称放入环境中。某些 shell 变量（例如，MAIL、PS1、PS2 和 IFS）经常被 .profile 文件导出，与这些变量发生冲突是不可取的；请参见 profile(4)。

以下环境变量可由应用程序使用，并且应在目标运行时环境中设置。

HOME
: 用户登录目录的名称，由 login(1) 通过口令文件设置；请参见 passwd(4)。

LANG
: 用于指定国际化信息的字符串，通过国际化信息，用户可以使用不同的国家约定。setlocale(3C) 函数检查 LANG 环境变量（当通过 "" 将它作为 locale 参数调用时）。如果特定类别的相应环境变量未设置或为 Null，LANG 将作为缺省语言环境使用。如果 LC_ALL 设置为有效的非空值，将使用其内容覆盖 LANG 和其他 LC_* 变量。例如，在以 setlocale(LC_CTYPE, "") 形式调用时，setlocale() 将首先查询 LC_CTYPE 环境变量来查看它是否已设置并且不为 Null。如果 LC_CTYPE 未设置或者为 Null，则 setlocale() 将检查 LANG 环境变量来查看它是否已设置并且不为 Null。如果 LANG 和 LC_CTYPE 均未设置或为 NULL，将使用缺省的 "C" 语言环境来设置 LC_CTYPE 类别。

大部分命令都会在执行任何其他处理之前调用 setlocale(LC_ALL, "")。这样，通过设置相应的环境变量，可以将命令与不同的国家约定一起使用。

以下环境变量对应于 setlocale(3C) 的每个类别：

LC_ALL
: 如果设置为有效的非空字符串值，将覆盖 LANG 以及所有其他 LC_* 变量的值。

LC_COLLATE
: 此类别指定所使用字符排序规则序列。此类别对应的信息存储在 localedef(1) 命令创建的数据库中。此环境变量影响 strcoll(3C) 和 strxfrm(3C)。

LC_CTYPE
: 此类别指定字符分类、字符转换以及多字节字符的宽度。当 LC_CTYPE 设置为有效值时，调用实用程序可以显示并处理包含下列字符的文本和文件名：该语言环境的有效字符；扩展 Unix 编码 (Extended Unix Code, EUC) 字符，其中任何单个字符的宽度可以为 1、2 或 3 个字节；以及 1、2 或 3 个列宽的 EUC 字符。缺省 "C" 语言环境对应于 7 位 ASCII 字符集；只有 ISO 8859-1 字符有效。此类别对应的信息存储在 localedef() 命令创建的数据库中。此环境变量由 ctype(3C)、mblen(3C) 以及许多其他命令（例如，cat(1)、ed(1)、ls(1) 和 vi(1)）使用。

LC_MESSAGES

此类别指定所使用消息数据库的语言。例如，应用程序的一个消息数据库可能包含法语消息，而另一个数据库可能包含德语消息。消息数据库由 mkmsgs(1) 命令创建。此环境变量由 exstr(1)、gettxt(1)、srchtxt(1)、gettxt(3C) 以及 gettext(3C) 使用。

LC_MONETARY

此类别指定用于特定语言环境的货币符号和分隔符。此类别对应的信息存储在 localedef(1) 命令创建的数据库中。此环境变量由 localeconv(3C) 使用。

LC_NUMERIC

此类别指定小数点分隔符和千分位分隔符。此类别对应的信息存储在 localedef() 命令创建的数据库中。在缺省的 C 语言环境中，使用 "." 作为小数点分隔符，不使用千分位分隔符。此环境变量由 localeconv(3C)、printf(3C) 以及 strtod(3C) 使用。

LC_TIME

此类别指定日期和时间格式。此类别对应的信息存储在 localedef() 指定的数据库中。在缺省的 C 语言环境中，使用美国的日期和时间格式。此环境变量由许多命令和函数使用。例如：at(1)、calendar(1)、date(1)、strftime(3C) 以及 getdate(3C)。

MSGVERB

控制在向 stderr 显示消息时 fmtmsg 选择哪些标准格式消息组件；请参见 fmtmsg(1) 和 fmtmsg(3C)。

NETPATH

冒号分隔的网络标识符列表。网络标识符是系统的网络选择组件使用的字符串，用于提供应用程序特定的缺省网络搜索路径。网络标识符必须由非 Null 字符构成且长度至少为 1。不指定最大长度。网络标识符通常由系统管理员选择。网络标识符还是任意 /etc/netconfig 文件条目中的第一个字段。因此，NETPATH 提供了指向 /etc/netconfig 文件的链接以及有关该网络条目中包含的网络的信息。/etc/netconfig 由系统管理员维护。getnetpath(3NSL) 中介绍的库例程可以访问 NETPATH 环境变量。

NLSPATH

包含一个模板序列，当 catopen(3C) 和 gettext(3C) 尝试定位消息目录时将使用这些模板。每个模板都由一个可选前缀、一个或多个替换字段、一个文件名以及一个可选后缀组成。例如：

```
NLSPATH="/system/nlslib/%N.cat"
```

定义 catopen() 应搜索目录 /system/nlslib 中的所有消息目录，其中目录名由传递到 catopen() 的 *name* 参数（即 %N）以及 .cat 后缀构成。

替换字段包含一个 % 符号，后面跟一个单字母关键字。当前定义了下列关键字：

%N

传递到 catopen() 的 *name* 参数的值。

%L

LANG 或 LC_MESSAGES 的值。

%l

LANG 或 LC_MESSAGES 中的语言元素。

%t

LANG 或 LC_MESSAGES 中的地区元素。

%c

LANG 或 LC_MESSAGES 中的代码集元素。

%%

单个 % 字符。

如果指定的值当前未定义，将替换为空字符串。分隔符 "_" 和 "." 不包含在 %t 和 %c 替换中。

NLSPATH 中定义的模板由冒号 (:) 分隔。前导冒号或两个相邻的冒号 (::) 相当于指定 %N。例如：

```
NLSPATH=":%N.cat:/nlslib/%L/%N.cat"
```

指示 catopen() 应搜索 *name*、*name*.cat 和 /nlslib/$LANG/*name*.cat 中的请求的消息目录。对于 gettext()，%N 将自动映射到 messages。

如果 NLSPATH 未设置或为 NULL，catopen() 和 gettext() 将调用 setlocale(3C)，后者将检查 LANG 和 LC_* 变量以定位消息目录。

Solaris 中对 %L 的扩展解释包括对已接受语言环境名称别名的支持，如 gettext(1)、gettext(3C)、catopen(3C)、setlocale(3C) 和 locale_alias(5) 中所述。

NLSPATH 通常在系统范围内设置（在 /etc/profile 中），这样可以使与消息目录相关的位置和命名约定对程序和用户都是透明的。

PATH

sh(1)、time(1)、nice(1)、nohup(1) 和其他实用程序在按不完整的路径名搜索文件时应用的目录前缀序列。前缀由冒号 (:) 分隔。login(1) 用于设置 PATH=/usr/bin。有关详细信息，请参见 sh(1)。

SEV_LEVEL

定义严重性级别，并且在标准格式错误消息中将字符串与严重性级别相关联并输出其字符串；请参见 addseverity(3C)、fmtmsg(1) 以及 fmtmsg(3C)。

TERM

要为其准备输出的终端的种类。此信息由可能利用该终端的特殊功能的命令（例如 vi(1)）使用。

TZ

时区信息。此环境变量的内容由函数 ctime(3C)、localtime(3C)、strftime(3C) 以及 mktime(3C) 用来覆盖缺省时区。TZ 值为以下两种格式（为清晰可见，插入了空格）之一：

```
:characters
```

或

```
std offset dst offset, rule
```

如果 TZ 属于第一种格式（即如果第一个字符为冒号 (:)）或者 TZ 不属于第二种格式，则 TZ 将指定时区数据库文件的路径（相对于 /usr/share/lib/zoneinfo/），如果存在前导冒号，将忽略它。

否则，如果 TZ 属于第二种格式，其展开形式如下：

stdoffset[*dst*[*offset*][,*start*[/*time*],*end*[/*time*]]]

std 和 *dst*

指示作为标准 (*std*) 时区或备用（*dst*，例如夏时制）时区指定的字符串，不少于三个字节且不超过 {TZNAME_MAX}。只有 *std* 是必需的；如果缺少 *dst*，则在此时区中不应用备用时间。其中每个字段都以两种格式之一出现，即括起或不括起：

- 在括起的格式中，第一个字符是小于号 ('<') 字符，最后一个字符是大于号 ('>') 字符。两个尖括号字符之间的所有字符都是当前语言环境的可移植字符集中的字母数字字符、加号 ('+') 字符或减号 ('-') 字符。这种情况下，*std* 和 *dst* 字段不包含尖括号字符。
- 在不括起的格式中，这些字段中的所有字符都是当前语言环境的可移植字符集中的字母字符。

如果任意字段的长度小于三个字节（缺少 *dst* 的情况除外）、大于 {TZNAME_MAX} 个字节或者包含指定字符之外的其他字符，则这些字段的解释是未明确指定的。

offset

指示本地时间与世界标准时间之间的差值。此时差的格式如下：

hh[:*mm*[:*ss*]]

分钟 (*mm*) 和秒 (*ss*) 是可选的。小时 (*hh*) 是必需的，可以是一位数。*offset*（放在 *std* 后面）是必需的。如果 *offset* 未出现在 *dst* 后面，将假定夏时制时间比标准时间早一小时。可以使用一位数或多位数。此值始终解释为十进制数字。小时必须介于 0 到 24 之间，如果出现分钟（和秒），分钟（和秒）必须介于 0 到 59 之间。如果值超出范围，可能会导致不可预测的行为。如果开头带有 -，表示时区在本初子午线的东侧。否则，时区位于本初子午线的西侧（通过开头的 "+" 符号表示，该符号是可选的）。

start/*time*, *end*/*time*

指示何时进行夏时制切换，其中，*start*/*time* 说明何时从标准时间更改为夏时制时间，而 *end*/*time* 说明何时从夏时制时间更改回标准时间。每个 *time* 字段都用本地时间说明发生更改的时间。

start 和 *end* 为以下格式之一：

`J`*n*

儒略日 *n* (1 ≤ *n* ≤ 365)。不计算闰日。也就是说，在所有年份中，2 月 28 日是第 59 日，3 月 1 日是第 60 日。无法表示特殊的 2 月 29 日。

n

从零开始计算的儒略日 (0 ≤ *n* ≤ 365)。计算闰日，而且可以表示 2 月 29 日。

`M`*m*.*n*.*d*

年度的第 *m* 月的第 *n* 周 (1 ≤ *n* ≤ 5, 1 ≤ *m* ≤ 12) 第 *d* 日 (0 ≤ *d* ≤ 6)，其中，第 5 周表示“第 *m* 月中的最后 *d* 日”，它可能出现在第四周或第五周。第 1 周是第 *d* 日后的第一周。第零日是周日。

实现特定的缺省值用于 *start* 和 *end*（如果未指定这些可选字段）。

time 的格式与 *offset* 相同，但前者不允许使用前导符号（"–" 或 "+"）。如果未指定 *time*，缺省值为 02:00:00。

另请参见

`cat(1)`、`date(1)`、`ed(1)`、`fmtmsg(1)`、`localedef(1)`、`login(1)`、`ls(1)`、`mkmsgs(1)`、`nice(1)`、`no`

引用名

eqnchar – special character definitions for eqn

用法概要

```
eqn  /usr/share/lib/pub/eqnchar filename |  troff options
```

```
neqn  /usr/share/lib/pub/eqnchar filename |  troff options
```

描述

The eqnchar command contains nroff(1) and troff(1) character definitions for constructing characters that are not available on the Graphic Systems typesetter. These definitions are primarily intended for use with eqn(1) and neqn(1). It contains definitions for the characters listed in the following table.

ciplus	⊕	| |	‖	*square*	□
citimes	⊗	*langle*	⟨	*circle*	○
wig	~	*rangle*	⟩	*blot*	□
-wig	≃	*hbar*	ℏ	*bullet*	•
>wig	≳	*ppd*	⊥	*prop*	∝
<wig	≲	<->	↔	*empty*	∅
=wig	≅	<=>	⇔	*member*	∈
star	*	|<	≮	*nomem*	∉
bigstar	✳	|>	≯	*cup*	∪
=dot	≐	*ang*	∠	*cap*	∩
orsign	∨	*rang*	∟	*incl*	⊑
andsign	∧	*3dot*	⋮	*subset*	⊂
=del	≜	*thf*	∴	*supset*	⊃
oppA	∀	*quarter*	¼	*!subset*	⊆
oppE	∃	*3quarter*	¾	*!supset*	⊇
angstrom	Å	*degree*	°		

文件

/usr/share/lib/pub/eqnchar

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	text/doctools

另请参见

eqn(1), nroff(1), troff(1), attributes(5)

引用名

extendedFILE – enable extended FILE facility usage

用法概要

```
$ ulimit -n N_file_descriptors
$ LD_PRELOAD_32=/usr/lib/extendedFILE.so.1 application [arg...]
```

描述

The extendedFILE.so.1 is not a library but an enabler of the extended FILE facility.

The extended FILE facility allows 32-bit processes to use any valid file descriptor with the standard I/O (see stdio(3C)) C library functions. Historically, 32-bit applications have been limited to using the first 256 numerical file descriptors for use with standard I/O streams. By using the extended FILE facility this limitation is lifted. Any valid file descriptor can be used with standard I/O. See the NOTES section of enable_extended_FILE_stdio(3C).

The extended FILE facility is enabled from the shell level before an application is launched. The file descriptor limit must also be raised. The syntax for raising the file descriptor limit is

```
$ ulimit -n max_file_descriptors
$ LD_PRELOAD_32=/usr/lib/extendedFILE.so.1 application [arg...]
```

where *max_file_descriptors* is the maximum number of file descriptors desired. See limit(1). The maximum value is the same as the maximum value for open(2).

环境变量

The following environment variables control the behavior of the extended FILE facility.

_STDIO_BADFD
: This variable takes an integer representing the lowest file descriptor, which will be made unallocatable. This action provides a protection mechanism so that applications that abuse interfaces do not experience silent data corruption. The value must be between 3 and 255 inclusive.

_STDIO_BADFD_SIGNAL
: This variable takes an integer or string representing any valid signal. See signal.h(3HEAD) for valid values or strings. This environment variable causes the specified signal to be sent to the application if certain exceptional cases are detected during the use of this facility. The default signal is SIGABRT.

示例

示例 1 Limit the number of file descriptors and FILE standard I/O structures.

The following example limits the number of file descriptors and FILE standard I/O structures to 1000.

```
$ ulimit -n 1000
$ LD_PRELOAD_32=/usr/lib/extendedFILE.so.1 application [arg...]
```

示例 2 Enable the extended FILE facility.

The following example enables the extended FILE facility. See enable_extended_FILE_stdio(3C) for more examples.

示例 2 Enable the extended FILE facility. （续）

```
$ ulimit -n 1000
$ _STDIO_BADFD=100 _STDIO_BADFD_SIGNAL=SIGABRT \
   LD_PRELOAD_32=/usr/lib/extendedFILE.so.1  \
   application [arg ...]
```

示例 3 Set up the extended FILE environment and start the application.

The following shell script first sets up the proper extended FILE environment and then starts the application:

```
#!/bin/sh
if [ $# = 0 ]; then
   echo "usage: $0 application [arguments...]"
   exit 1
fi
ulimit -n 1000
# _STDIO_BADFD=196; export _STDIO_BADFD
# _STDIO_BADFD_SIGNAL=SIGABRT; export _STDIO_BADFD_SIGNAL
LD_PRELOAD_32=/usr/lib/extendedFILE.so.1; export LD_PRELOAD_32
"$@"
```

文件

`/usr/lib/extendedFILE.so.1` enabling library

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/library (32–bit)
Interface Stability	Committed
MT-Level	Safe

另请参见

limit(1), open(2), enable_extended_FILE_stdio(3C), fdopen(3C), fopen(3C), popen(3C), signal.h(3HEAD), stdio(3C), attributes(5)

警告

The following displayed message

```
Application violated extended FILE safety mechanism.
Please read the man page for extendedFILE.
Aborting
```

is an indication that your application is modifying the internal file descriptor field of the FILE structure from standard I/O. Continued use of this extended FILE facility could harm your data. Do not use the extended FILE facility with your application.

引用名

extensions – localedef extensions description file

描述

A localedef extensions description file or *extensions* file defines various extensions for the `localedef(1)` command.

The localedef extensions description file provides:

- EUC code set width information via the `cswidth` keyword: `cswidth bc1 : sw1,bc2 : sw2,bc3 : sw3`where `bc1`, `bc2`, and `bc3` indicate the number of bytes (byte count) per character for EUC codesets `1`, `2`, and `3`, respectively. `sw1`, `sw2`, and `sw3` indicate screen width for EUC codesets `1`, `2`, and `3`, respectively.
- Other extensions which will be documented in a future release.

另请参见

`locale(1)`, `localedef(1)`, `environ(5)`, `locale(5)`

引用名

fedfs – overview of Federated Filesystem (FedFS)

描述

The NFSv4 Working Group of the Internet Engineering Task Force (IETF) has been working to standarize a set of protocols which together permit construction and maintenance of a federated filesystem, where many different file servers together share a single namespace. These protocols make it possible to create a multi-vendor global namespace.

Solaris currently has support for two of the three FedFS protocols:

DNS SRV records

The Solaris automounter (`automount`(1M)) supports the `-fedfs` map, which by default will mount `/nfs4/`*dns_domain* by looking up a DNS SRV record for the domain. This map is enabled by default in `/etc/auto_master`.

LDAP-based referrals

The Solaris NFS server will follow FedFS referrals by looking up location data in LDAP, guided by the schema specified by the FedFS specs.

示例

示例 1 Using the `-fedfs` Automounter Map

To mount a path like `/nfs4/cthon.org`, your NFS client should be set up to use DNS such that the domain's DNS server should answer a query for the `_nfs-domainroot._tcp` SRV record like this:

```
$ nslookup '-q=srv' _nfs-domainroot._tcp.cthon.org
Server:         1.2.3.4
Address:        1.2.3.4#53

_nfs-domainroot._tcp.cthon.org  service = 0 0 2049 root-a.cthon.org.
_nfs-domainroot._tcp.cthon.org  service = 1 1 2049 root-n.cthon.org.

$ cd /nfs4/cthon.org

$ df .
/nfs4/cthon.org    (root-a.cthon.org,root-b.cthon.org:/.domainroot/\
cthon.org):120379963 blocks 120379963 files
```

示例 2 Using `nsdbparams` to Set Up Communications with an NSDB

The following example sets up communications with an NSDB called `nsdb.cthon.org` and makes it the default NSDB:

```
# nsdbparams update -D cn=Manager,dc=cthon,dc=org -w cthon.org \
nsdb.cthon.org
# nsdbparams show nikon.us.example.com
nikon.us.example.com:389
     default bind DN: cn=Manager,dc=cthon,dc=org
     default bind PW: cthon.org
     default NCE: dc=cthon,dc=org
     sectype: FEDFS_SEC_NONE
```

示例 2　Using nsdbparams to Set Up Communications with an NSDB　（续）

```
# nsdbparams set nsdb.cthon.org
# nsdbparams get
default nsdb: nsdb.cthon.org
default port: 389
```

示例 3　Using the NSDB Tools

The following commands illustrate the use of the NSDB tools available in Oracle Solaris.

```
# nsdb-nces
Host: nsdb.cthon.org:389
  namingContext 'dc=cthon,dc=org' is a FedFS NCE, DIT starts at ''

# nsdb-list
SDB: nsdb.cthon.org:389, dc=cthon,dc=org
  FSN UUID: 7cc0bf04-5459-11e1-8083-80093d11d889
     FSL UUID: 7cc33c02-5459-11e1-8084-00093d11d889 = filer-a:/tmp
  FSN UUID: db48f160-5858-11e1-b459-80093d11d889
     FSL UUID: db4998c2-5858-11e1-b45a-00093d11d889 = filer-j:/tmp

# nsdb-resolve-fsn 7cc0bf04-5459-11e1-8083-80093d11d889
For FSN UUID 7cc0bf04-5459-11e1-8083-80093d11d889
  FSL UUID: 7cc33c02-5459-11e1-8084-00093d11d889
    Location: filer-a:/tmp
```

示例 4　Using nfsref to Create a Referral

The following sequence of commands illustrates the use of nfsref to create a referral.

```
# nsdb-list
NSDB: nsdb.cthon.org:389, dc=cthon,dc=org
  FSN UUID: 7cc0bf04-5459-11e1-8083-80093d11d889
    FSL UUID: 7cc33c02-5459-11e1-8084-00093d11d889 = filer-a:/tmp
  FSN UUID: db48f160-5858-11e1-b459-80093d11d889
    FSL UUID: db4998c2-5858-11e1-b45a-00093d11d889 = filer-j:/tmp

# nfsref -t nfs-fedfs add /root/tools filer-k:/tools filer-x:/tools
Enter password for cn=Manager,dc=cthon,dc=org:
Created reparse point /root/tools

# nsdb-list
NSDB: nsdb.cthon.org:389, dc=cthon,dc=org
  FSN UUID: 7cc0bf04-5459-11e1-8083-80093d11d889
    FSL UUID: 7cc33c02-5459-11e1-8084-00093d11d889 = filer-a:/tmp
  FSN UUID: db48f160-5858-11e1-b459-80093d11d889
    FSL UUID: db4998c2-5858-11e1-b45a-00093d11d889 = filer-j:/tmp
  FSN UUID: 004b2382-9663-11e1-8c79-80093d11d888
    FSL UUID: 004da2b0-9663-11e1-8c7a-00093d11d888 = filer-k:/tools
```

示例 4 Using nfsref to Create a Referral （续）

```
    FSL UUID: 004e9bac-9663-11e1-8c7b-00093d11d888 = filer-x:/tools
```

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/file-system/nfs

另请参见

automount(1M), nfsref(1M), nsdb-list(1M), nsdbparams(1M), attributes(5)

RFC 5716: Requirements for Federated File Systems

引用名

filesystem – file system organization

用法概要

/

描述

The file system is a hierarchical structure of descendent file systems and directories that are used to organize system-related components and binaries as well as non-system-related components and home directories in the Oracle Solaris OS. By default, the root file system is installed within a ZFS root pool and specifically, is a ZFS file system with separate directories of system-related components, such as `etc`, `usr`, and `var`, that must be available for the system to function correctly. After a system is installed, the root of the Solaris file system is mounted, which means files and directories are accessible.

All subdirectories of the root file system that are part of the Oracle Solaris OS, with the exception of `/var`, must be contained in the same file system as the root file system. In addition, all packaged software must reside in the root pool, with the exception of the swap and dump devices. A default swap device and dump device are created automatically as ZFS volumes in the root pool when a system is installed.

The following file system content descriptions make use of platform, platform-dependent, platform-independent, and platform-specific terms. Platform refers to a system's Instruction Set Architecture or processor type as returned by the `uname -i` command. Platform-dependent refers to a file that is installed on all platforms and whose contents vary depending on the platform. Like a platform-dependent file, a platform-independent file is installed on all platforms. However, the contents of the latter type remains the same on all platforms. An example of a platform-dependent file is compiled, executable program. An example of a platform-independent file is a standard configuration file, such as `/etc/hosts`. Unlike a platform-dependent or a platform- independent file, the platform-specific file is installed only on a subset of supported platforms. Most platform-specific files are gathered under `/platform` and `/usr/platform`.

Root File System

The root file system contains files and directories that are critical for system operation, such as the kernel, the device drivers, and the programs used to boot the system. These components are described below. The root (`/`) directory also contains mount point directories where local and remote file systems can be attached to the file system hierarchy.

`/`
: Root directory of the entire file system name space. This is a special file system that is mounted by the kernel at system boot time.

`/boot`
: Directory that contains files and executables that are needed for booting the system.

`/bin`
: Symbolic link to the `/usr/bin` directory that contains system executables and scripts.

`/dev`
: Directory that contains special device files. Typically, device files are built to match the kernel and hardware configuration of the system.

/devices
Mount point directory for the devfs file system that manages the device name space.

/etc
Directory that contains platform-dependent administrative and configuration files and databases that are not shared among systems. This directory defines the system's identity. An approved installation location for bundled Solaris software.

/export/home or /home
Directory or file system mount point for user home directories, which store user files. By default, the /home directory is an automounted file system.

/kernel
Directory of platform-dependent loadable kernel modules required as part of the boot process. It includes the generic part of the core kernel that is platform-independent, /kernel/genunix. See kernel(1M). An approved installation location for bundled Oracle Solaris software and for add-on system software.

/lib
Directory that contains core system libraries. Historically, this directory contained essential library components for system startup.

/media
Directory for accessing removable media that is automatically mounted.

/mnt
Default temporary mount point directory for file systems. This empty directory is used to temporarily mount a file system.

/net
Temporary mount point directory for file systems that are mounted by the automounter.

/opt
Directory for unbundled application packages.

/platform
Directory of platform-specific objects that need to reside in the root file system. It contains a series of directories, one per supported platform. The semantics of the series of directories is equivalent to / (root).

/proc
Mount point directory for the process file system.

/root
Home directory for the root user.

/rpool
Mount point directory for the ZFS boot-related components. By default, the root pool is named rpool during installation.

`/sbin`
Symbolic link to the `/usr/sbin` directory.

`/system`
Mount point directory for the contract (`CTFS`) and object (`OBJFS`) file systems.

`/tmp`
Directory that contains temporary files that are removed during a boot operation.

`/usr`
Directory that contains platform-dependent and platform-independent binaries and files. The `/usr/share` subdirectory contains platform-independent files. The rest of the `/usr` directory contains platform-dependent files.

`/usr/bin`
Directory that contains platform-dependent, user-invoked executables. These are commands that users expect to be run as part of their normal `$PATH`. An approved installation location for bundled Oracle Solaris software. The analogous location for unbundled system software or for applications is `/opt/packagename/bin`.

`/usr/lib`
Directory that augments the contents of `/lib` with additional system libraries, and other supporting files that are required by programs at runtime.

`/usr/java`
Directory that contains Java files and executables.

`/usr/sbin`
Contains essential executables used in the booting process and in manual system recovery. Historically, this directory was needed to recover the system before the `/usr` file system was mounted. In this Oracle Solaris release, `/usr` is a directory, not a separate file system and is available when the root file system is mounted.

`/var`
Directory or file system that contains varying files that are unique to a system but can grow to an arbitrary or variable size. An example is a log file. An approved installation location for bundled Oracle Solaris software.

`/var/share`
Directory that is shared across several boot environments, and is resident on a separate dataset beneath the `zpool` containing boot environments. Packages should not deliver content here, since package metadata in a given boot environment may not reflect the current content of `/var/share`.

Packages may deliver directories to `/var/.migrate`, which will be created automatically beneath `/var/share` during boot. See `pkg`(5) and the IPS Developer's Guide for a description of how to share data across boot environments.

By default, the following directories are shared:

```
/var/share/audit
/var/share/cores
/var/share/crash
/var/share/mail
```

Symlinks are delivered to `/var` to point to each shared directory.

If existing datasets attempt to mount on one of those symlinks (for example, users with an existing dataset that normally mounts on `/var/mail`), then those datasets will be mounted beneath `/var/share`, since filesystem mounts traverse symlinks. The dataset will still be accessible through the original mountpoint.

`/var/tmp`
Directory that contains files that vary in size or presence during normal system operations. The content of this directory is not removed during a boot operation. It is possible to change the default behavior for `/var/tmp` to clear all of the files except editor temporary files by setting the `clean_vartmp` property value of the `rmtmpfiles` service. This is done with the following commands:

```
# svccfg -s svc:/system/rmtmpfiles setprop\
            options/clean_vartmp = "true"
# svcadm refresh svc:/system/rmtmpfiles:default
```

The `solaris.smf.value.rmtmpfiles` authorization is required to modify this property.

另请参见

isainfo(1), svcs(1), uname(1), automount(1M), automountd(1M), boot(1M), init(1M), kernel(1M), mount(1M), svcadm(1M), svccfg(1M), zfs(1M), zpool(1M), mount(2), Intro(4), proc(4), , ctfs(7FS), devfs(7FS), objfs(7FS),

引用名

fmri – Fault Managed Resource Identifier

描述

Oracle Solaris subsystems such as the Service Management Facility (see smf(5)), the Fault Management Daemon (see fmd(1M), fmadm(1M)), and the Image Packaging System (see pkg(5)) identify each element they manage using a *fault managed resource identifier*, abbreviated as FMRI.

FMRIs are used to identify both hardware resources and software resources and abstractions. For hardware resources the FMRIs are usually an implementation detail, and when a component is diagnosed as faulty it is the FRU (field replaceable unit) location label which is used to identify it. If no FRU label is available, the FMRI is used. For software abstractions such as an SMF service or a pkg(5) software package the FMRI (often in string form) is the one and only means of naming the managed resource (abbreviations aside).

All FMRIs include an indication of the FMRI scheme adhered to, and the version of that FMRI scheme in use. Once the scheme and version are known, the remainder of the FMRI is interpreted as specified in the formal definition of that FMRI scheme version. Schemes exists for `svc`, `pkg`, `cpu`, `hc` (hardware component), `dev` (device), and a number of others.

The formal definition of a given version of a particular FMRI scheme is specified as an unordered series of FMRI members specified as (member-name, data-type, data-semantics) tuples. An instance of an FMRI is a series of (name, type, value) tuples wherein the name and type match the FMRI specification and the value provided is a value of the given type and derived and interpreted according to the semantics described in the formal FMRI definition for that scheme and version. In Oracle Solaris C library APIs such as libfmevent(3LIB), FMRIs are represented as `nvlists` using the libnvpair(3LIB) library.

It is often more convenient to represent a given FMRI instance as a string, such as in command line interaction or in command output. The specification of each FMRI scheme version includes rules for formatting an FMRI of that scheme and version as a string instead of an name-value pair list, and given an FMRI string those same rules can be applied in reverse to recover the more-formal FMRI form. In some cases the string form of an FMRI can include a reduced amount of information as compared to the formal form, but still provide enough information to identify the resource for administrative purposes. The string form of an FMRI usually does not include the FMRI version number, and so the version is understood to be that whose formal FMRI-to-string formatting rules the given FMRI string matches. The string form of an FMRI is not unlike a URI as outlined in RFC 2396 which we use as a syntactical model. In particular, we reserve the same set of characters `:`, `;`, `=`, `@`, `/`, `&`, `'`, `$` and `,` and require the same escaped encoding conventions should these characters appear in an FMRI member value that is rendered in the string.

FMRI Members

The formal definition of a given version of a particular FMRI scheme is specified as an unordered series of:

```
(member-name, data-type, data-semantics)
```

tuples, some of which are required and some of which are optional. The formal definition also specifies how to render the FMRI in string form.

`member-name`

This associates a name with the FMRI member, with characters drawn from the following alphabet:

```
A-Z
a-z
'_' (underscore)
'-' (hyphen)
'.' (period)
```

`data-type`

This specifies the data type for this member. Legal primitive datatypes are listed in the following table, and for all primitive types we also define an array of that type as a legal datatype. The primitive integer and double datatypes are defined as in C, and we add boolean value and string primitive datatypes:

Primitive Datatype	Description
boolean_value	Values can be 0 or 1
string	ASCII string
[u]int8	(Un)signed 8-bit integer
[u]int16	(Un)signed 16-bit integer
[u]int32	(Un)signed 32-bit integer
[u]int64	(Un)signed 64-bit integer
double	Double precision floating point

For each primitive datatype we form an aggregate datatype that is an ordered array of the primitive type:

Primitive Datatype	Description
boolean_value array	Array of boolean_value
string array	Array of ASCII strings
[u]int8 array	Array of (un)signed 8-bit integer
[u]int16 array	Array of (un)signed 16-bit integer
[u]int32 array	Array of (un)signed 32-bit integer
[u]int64 array	Array of (un)signed 64-bit integer
double array	Array of double

A further aggregate datatype is defined - the `nvlist` (name- value list). An `nvlist` is an unordered list of (`list-member-name, list-member-datatype`) in which `list-member-name` names the member using the same alphabet as for `member-name` above, and `list-member-datatype` is any primitive or aggregate datatype including `nvlist` itself. We term the members of an `nvlist`: `nvpairs` (name-value pairs).

A special type of `nvlist` is an `fmri` which is an `nvlist` that conforms to one of the FMRI scheme versions defined below.

Synthetic Datatype	Description
nvlist	Unordered list of nvpairs

```
 nvlist array          Array of nvlist
 fmri                  An nvlist that specifies an FMRI
                       scheme and version and includes other
                       members so as to be a valid FMRI
                       in that scheme version.
```

`data-semantics`

The data semantics for a given member in an FMRI definition specify how the values for that member should be interpreted, that is, their meaning. The data semantics can constrain the set of legal values that the member can assume to a subset of those representable by the member data type.

Stabilities

Some FMRI schemes, such as `svc` for SMF, are administrator-facing while others are more of a private implementation detail. Correspondingly, the definition of some FMRI schemes constitute a `Committed` interface, while the definition of others form a `Private` interface. See `attributes(5)`. As with all `Private` interfaces the definition of `Private` FMRI components is subject to possibly incompatible change at any time, and you should not consume such interfaces nor rely on the presence or assumed semantics of private FMRI members.

In defining an FMRI scheme below we list a series of (`member-name, data-type, data-semantics`), specify which members are required and which are optional, and the rules for rendering an FMRI in string form. In terms of stability assignments, there are two stability levels associated with each member:

- Stability of the (`member-name, data-type`) pair
- Stability of the corresponding data-semantics

The default in both cases is `Private`. When the stability of both aspects of a member are identical (the common case) we write the stability level just once. When they differ then we write the member name and data type stability first, followed by a comma and the data semantics stability level. Today the only stability levels employed in FMRIs are `Committed`, `Private`, and `Not An Interface` but others are allowed. The semantics of a `Committed` FMRI component do not change incompatibly for given scheme and version thereof.

FMRI Scheme and Version Identification

All FMRI definitions are required to include members for the scheme name and major version thereof. The name/datatype and data semantics stabilities are `Committed`, thus all FMRI definitions are required to begin as follows:

```
Member Name      Data Type    Stability
----------------------------------------
scheme           string       Committed
version          uint8        Committed
```

`scheme`

Names this FMRI scheme

`version`

Scheme major version

Together these select the set of rules (as below) by which the given FMRI should be interpreted. A small number of scheme names are defined to cope with identifying resources within differing subsystems, and each scheme is versioned starting at version number 0.

When using the informal string representation of an FMRI, the scheme is always written but the scheme version in use is usually not included. The version in use is understood to be the latest for which the given FMRI string matches the FMRI-to-string conversion rules.

FMRI Authority

A further required, although implicitly present in some cases, FMRI member is `authority`:

```
Member Name   Data Type  Stability
---------------------------------------
[authority]   nvlist     Committed, (Varies)
```

Every FMRI includes authority information, either explicitly with the `authority` `nvlist` if present in the FMRI or implicitly that of the local fault management domain if not present. Authority information has two flavors:

- High-level authority information, such as to identify the platform and hostname within which a particular resource that is the subject of an FMRI is located
- Leaf-level authority information, providing identity information for the specific resource that is the subject of the FMRI

For example, an FMRI identifying a disk can have high-level authority information detailing the platform (host name, product name, chassis serial) in which the disk resides, and leaf-level authority information detailing the individual disk part number, serial, and so forth.

It is an unfortunate fact that some FMRI schemes have evolved to include leaf-level authority information outside of the `authority` `nvlist` as standalone FMRI members, and written as part of the path component in string form.

In their string representation, both high-level and leaf-level, FMRI authority information is prefaced with a colon (:), for example:

```
hc://:chassis-serial=12345/<string>
```

“:chassis-serial=12345” is a string represented authority member.

There are two versions of the high level authority (version 0 and version 1). Following are the descriptions for each.

The leaf-level authority members are described in the scheme specific sections below.

Version 0

```
Member Name   Data Type   Stability
-----------------------------------------------------
[server-id]   string      Committed, Not an Interface
[chassis-id]  string      Committed, Not an Interface
```

[product-sn]	string	Committed, Not an Interface
[product-id]	string	Committed, Not an Interface
[domain-id]	string	Committed, Not an Interface
[host-id]	string	Committed, Not an Interface

[server-id]
The hostname (uname -n) string for the entity on which the fault manager is running. Not invariant.

[chassis-id]
This is a serial number identifying the chassis within which the resource resides. Some large systems can include multiple distinct chassis components.

[product-sn]
In some cases chassis-id alone does not fully identify the system, for example, for service entitlement keyed by a product serial number and where the product can comprise multiple chassis elements. Invariant.

[product-id]
The product name string, such as Sun-Fire-X4600. Invariant.

[domain-id]
Identifies a particular hardware domain in the case of hardware that supports this feature. Not invariant.

[host-id]
The hostid(1) string in Oracle Solaris); not generally used. Not necessarily invariant.

Version 1

Member Name	Data Type	Stability
[system-mfg]	string	Committed, Not an Interface
[system-name]	string	Committed, Not an Interface
[system-part]	string	Committed, Not an Interface
[system-serial]	string	Committed, Not an Interface
[sys-comp-mfg]	string	Committed, Not an Interface
[sys-comp-name]	string	Committed, Not an Interface
[sys-comp-part]	string	Committed, Not an Interface
[sys-comp-serial]	string	Committed, Not an Interface
[chassis-mfg]	string	Committed, Not an Interface
[chassis-name]	string	Committed, Not an Interface
[chassis-part]	string	Committed, Not an Interface
[chassis-serial]	string	Committed, Not an Interface
[chassis-alias]	string	Committed, Not an Interface
[server-name]	string	Committed, Not an Interface
[domain-name]	string	Committed, Not an Interface
[host-id]	string	Committed, Not an Interface

`[system-mfg]`
The manufacturer of the top-level product. In a complex product the system-* members contain the service entitlement data. In a simple standalone rack system the system-*, sys-comp-* and chassis-* data is duplicated. Invariant.

`[system-name]`
The model name of the top-level product. Invariant.

`[system-part]`
The part number of the top-level product. Invariant.

`[system-serial]`
The serial number of the top-level product. Invariant.

`[sys-comp-mfg]`
The manufacturer of the component system-level product. Invariant.

`[sys-comp-name]`
The model name of the component system-level product. Invariant.

`[sys-comp-part]`
The part number of the component system-level product. Invariant.

`[sys-comp-serial]`
The serial number of the component system-level product. Invariant.

`[chassis-mfg]`
The manufacturer of the chassis within which the resource resides. Some large systems can include multiple distinct chassis components. Invariant.

`[chassis-name]`
The model name of the chassis. Invariant.

`[chassis-part]`
The part number of the chassis. Invariant.

`[chassis-serial]`
The serial number of the chassis. Invariant.

In some cases a single set of *-name/*-part/*-serial alone does not fully identify the system, for example, for service entitlement keyed by the product serial number and where the product can comprise multiple chassis elements. Invariant.

`[chassis-alias]`
This is the Chassis Receptacle Occupant alias value assigned to this chassis via ‘fmadm add-alias‘. Not invariant.

`[server-name]`
The hostname (`uname -n`) string for the entity on which the fault manager is running. Not invariant.

[domain-name]
Identifies a particular hardware domain in the case of hardware that supports this feature. Not invariant.

[host-id]
The hostid(1) string in Oracle Solaris); not generally used. Not necessarily invariant.

Logical vs. Universal FMRI Schemes

A logical FMRI scheme defines FMRIs that can only meaningfully be interpreted within the fault management domain (typically an Oracle Solaris instance) in which they were generated. Identical FMRIs of a logical scheme that are native to distinct fault management domains do not necessarily identify the same actual resource. For example a cpu identified as cpu id 1 using the cpu scheme must be interpreted in the Oracle Solaris instance that generated the FMRI (that is, its native context), and many other Oracle Solaris instances also have a logical processor id number 1 but all such FMRIs do not identify the same actual processor. The native fault management domain is identified by its high-level authority information which is typically not captured in an FMRI instance but instead understood to be that of the local Oracle Solaris instance.

A universal FMRI scheme identifies resources in a universally unique manner, and two identical FMRIs in a universal scheme identify the same actual resource wherever they are interpreted. Such schemes are used when ambiguity must be avoided, such as in identifying hardware components that are faulted.

FMRI schemes do not include a member indicating whether the scheme is logical or universal.

FMRI Schemes and Versions

The following FMRI scheme versions are defined in the sections that follow:

Scheme	Version(s)	Universal?	Description
cpu	0, 1	No	Logical processor identification
dev	0	No	Device resources
fmd	0, 1	No	Fault Management Daemon modules
hc	0, 1	Yes	Hardware Components
mem	0	No	Memory modules and cache
mod	0	No	Kernel modules
pkg	0, 1	Yes	Software packages
svc	0	No	SMF services & service instances
sw	0	No	Software objects
zfs	0	No	zfs filesystem resources

Of these, the svc and pkg schemes are the most-likely to be encountered in day-to-day Oracle Solaris use, while the remainder are usually hidden behind command line interaction with the fault management subsystem when problems have been diagnosed.

SCHEME cpu VERSION 0

The cpu scheme is used to identify the logical Oracle Solaris processor resource that can be affected by a problem that has been diagnosed, and which can be offlined in response to that problem diagnosis. It does not necessarily identify the physical resource involved, and should not be used to identify parts for replacement.

The cpu scheme is a logical FMRI scheme; an FMRI in the cpu scheme is meaningful only within the fault management domain identified by the authority information.

```
Member Name   Data Type    Stability
------------------------------------
scheme        string       Committed, value "cpu"
version       uint8        Committed, value 0
cpuid         uint32       Committed
[serial]      uint64       Private
```

`cpuid`
The Oracle Solaris processor instance id of this logical execution unit, as listed by psradm(1M) on the Oracle Solaris instance identified by the authority information.

`[serial]`
The semantics of this Private FMRI member are not documented.

`String form:`

```
cpu:///cpuid=<cpuid>[/serial=<hex-serial>]
```

`<hex-serial>` is the *serial* member written in hexadecimal with no `0x` prefix.

SCHEME cpu VERSION 1

Version 1 of the cpu FMRI scheme has the same use as version 0, with some private FMRI member changes and additions.

```
Member Name   Data Type   Stability
-----------------------------------------------
scheme        string      Committed, value "cpu"
version       uint8       Committed, value 1
cpuid         uint32      Committed
[serial]      string      Private
[cpumask]     uint8       Private
[cpufru]      string      Private
[cacheindex]  uint32      Private
[cacheway]    uint32      Private
[cachebit]    uint16      Private
[cachetype]   uint8       Private

cpuid

       As per cpu scheme version 0

[serial]
```

[cpumask]
[cpufru]
[cacheindex]
[cacheway]
[cachebit]
[cachetype]

The semantics of these Private FMRI members are not documented.

String form:

```
cpu:///cpuid=<cpuid>[/serial=<serial>]
         [/cacheindex=.../cacheway=.../
         cachebit=.../cachetype=...]
```

Members cpumask and cpufru do not appear in the string form.

SCHEME dev VERSION 0

The dev scheme is used to identify devices. It is not an invariant scheme. A dev scheme FMRI should be interpreted only in the fault management domain identified within the authority information.

Member Name	Data Type	Stability
scheme	string	Committed, value "dev"
version	uint8	Committed, value 0
device-path	string	Committed
[devid]	string	Private
[target-port-l0id]	string	Private

device-path
A filesystem path within the Oracle Solaris /devices tree that identifies a device node, such as pci@0,0/pci108e,cb84@2,1. Beyond this interpretation that this member identifies a /devices path, the internal structure of such paths, that is, the structure and node naming of the Oracle Solaris /devices tree) is Not An Interface.

[devid]
[target-port-l0id]
The semantics of these Private FMRI members are not documented.

String form:

```
dev:///[:devid=<devid>]
    [:target-port-l0id=<target-port-l0id>]
    <device-path>
```

SCHEME fmd VERSION 0

This scheme is used to identify Oracle Solaris Fault Manager (fmd(1M)) modules.

High-level authority:

Member Name	Data Type	Stability
[server-id]	string	Committed, Not an Interface
[chassis-id]	string	Committed, Not an Interface
[product-sn]	string	Committed, Not an Interface
[product-id]	string	Committed, Not an Interface
[domain-id]	string	Committed, Not an Interface

FMRI does not include any leaf-level authority information within the authority nvlist, instead it is featured as top-level members of the FMRI.

Member Name	Data Type	Stability
scheme	string	Committed, value "fmd"
version	uint8	Committed, value 0
[authority]	nvlist	Committed, See above
mod-name	string	Committed, Not an Interface
mod-version	string	Committed, Not an Interface

`mod-name`
The `fmd` module name, an opaque string.

`mod-version`
The `fmd` module version, an opaque string.

String form:

```
fmd:///module/mod-name>
```

SCHEME fmd VERSION 1

This scheme is used to identify Oracle Solaris Fault Manager (fmd(1M)) modules.

High-level authority:

Member Name	Data Type	Stability
[system-mfg]	string	Committed, Not an Interface
[system-name]	string	Committed, Not an Interface
[system-part]	string	Committed, Not an Interface
[system-serial]	string	Committed, Not an Interface
[sys-comp-mfg]	string	Committed, Not an Interface
[sys-comp-name]	string	Committed, Not an Interface
[sys-comp-part]	string	Committed, Not an Interface
[sys-comp-serial]	string	Committed, Not an Interface
[server-name]	string	Committed, Not an Interface
[domain-name]	string	Committed, Not an Interface
[host-id]	string	Committed, Not an Interface

As with version 0 the FMRI does not contain leaf-level authority informaion within the authority nvlist. Instead it is featured as top-level members of the FMRI.

Member Name	Data Type	Stability
scheme	string	Committed, value "fmd"
version	uint8	Committed, value 1
[authority]	nvlist	Committed, See above
mod-name	string	Committed, Not an Interface
mod-version	string	Committed, Not an Interface

`mod-name`
The `fmd` module name, an opaque string.

`mod-version`
The `fmd` module version, an opaque string.

String form:

```
fmd:///module/<mod-name>
```

SCHEME legacy-hc VERSION 0

The hc scheme is used to identify hardware components. In most cases these FMRIs should remain as an internal implementation detail, and where a hardware component is diagnosed as faulty it is identified (for example, in `fmadm`(1M)) by its FRU label or location label.

There is only one version of the `hc` scheme in use (version 0) but we differentiate two minor versions, distinguished by the presence or absence of a member named component. An `hc` scheme FMRI that includes a component member is an instance of the obsoleted legacy `hc` scheme.

Obsolete legacy `hc` variant:

Member Name	Data Type	Stability
scheme	string	Committed, value "hc"
version	uint8	Committed, value 0
component	string	Committed

`component`
The FRU label for the resource.

String form for legacy `hc`:

```
hc:///component=<component>
```

SCHEME hc VERSION 0

The authority `nvlist` is the high-level authority information concerned with identifying the fault management domain in which a given FMRI was generated. Thus it includes members for the hostname, product name, chassis serial number and so on. Some of these members are invariant (such as platform serial number) and serve uniquely to identify some element, while others (such as hostname) are a soft identification which could change over time (albeit infrequently).

Member Name	Data Type	Stability
[server-id]	string	Committed, Not an Interface
[chassis-id]	string	Committed, Not an Interface
[product-sn]	string	Committed, Not an Interface
[product-id]	string	Committed, Not an Interface
[domain-id]	string	Committed, Not an Interface
[host-id]	string	Committed, Not an Interface

While all members are listed as optional, as many as make sense for a given FMRI should be included. If the authority `nvlist` is absent in an FMRI then the authority is understood to be the local fault management domain; if an `hc` scheme FMRI is transmitted outside of its native domain it should include explicit high-level authority information.

The `Not an Interface` stability for the data semantics of each means that the value can be treated as an opaque string only and any internal structure to the string is not an interface. For example a serial number can be matched using string compare, but the length of the serial number string and the meaning of individual characters therein is not an interface.

String form:

These authority members, those that are present in a given FMRI, can be written in string form as an unordered series of colon-separated *name*=*value* pairs. We'll define this as `authstr`, to be used in the specification of the string form for individual FMRI schemes below which import the default authority information.

```
authstr = server-id=<server-id>
                [:chassis-id=<chassis-id>]
                [:product-sn=<product-sn>]
                [:product-id=<product-id>]
                [:domain-id=<domain-id>]
                [:host-id=<host-id>]
                [:devid=<devid>]
```

There is no leaf-level authority/identity information included in the authority `nvlist` - instead such information (serial and part numbers) feature as top-level members of the FMRI.

Member Name	Data Type	Stability
scheme	string	Committed, value "hc"
version	uint8	Committed, value 0
[authority]	nvlist	Committed, See above
[serial]	string	Committed, Not-an-Interface
[devid]	string	Private
[part]	string	Committed, Not-an-Interface
[revision]	string	Committed, Not-an-Interface
[hc-root]	string	Private
hc-list-sz	uint32	Private
hc-list	nvlist array	Private

[facility]	nvlist	Private
[hc-specific]	nvlist	Private

`[serial]`

The serial number string of the hardware component identified in this FMRI. This is an opaque string whose internal structure is not an interface.

`[devid]`

The semantics of these Private FMRI members are not documented.

`[part]`

The part number string of the hardware component identified in this FMRI. This is an opaque string whose internal structure is not an interface.

`[revision]`

The revision of the part number, as a string. Again this is an opaque string.

`[hc-root]`

The semantics of this Private FMRI member are not documented.

`hc-list-sz`

The semantics of this Private FMRI member are not documented.

`hc-list`

The semantics of this Private FMRI member are not documented, however we enumerate the structure of the content (subject to incompatible change). Each member of the (ordered) `nvlist` array has the following structure:

Member Name	Data Type	Stability
hc-name	string	Private
hc-id	string	Private

`[facility]`

The semantics of this Private FMRI member are not documented, however we enumerate the structure of the content (subject to incompatible change). The facility `nvlist` has members as follows:

Member Name	Data Type	Stability
facility-type	string	Private
facility-name	string	Private

`[hc-specific]`

The semantics of this Private FMRI member are not documented.

`String form:`

```
authstr is defined above
hcauthstr = [:serial=<serial>]
                 [:devid=<devid>]
                 [:part=<part>]
```

```
                  [:revision=<revision>]

hc-path =<hc-list[0].hc-name>=<hc-list[0].hc-id>
               /<hc-list[1].hc-name>=<hc-list[1].hc-id>
               /<hc-list[2].hc-name>=<hc-list[2].hc-id>
               ...
hcspecstr = private string formed from hc-specific nvlist,
               of the form "/name=value"

facstr = <facility.facility-type>=<facility.facility-name>

hc://[authstr][hcauthstr]/[hc-root/]<hc-path>
                                     [<hcspecstr>]
                                     [?<facstr>]
```

The high-level authority information from the authority nvlist appears adjacent to the leaf-level authority information of serial and part number, all colon separated. The components of this authority string can appear in any order, with high-level and leaf authority elements interleaved. Parsing such a string is clumsy and relies on knowing in advance which components belong to which class of authority information.

SCHEME hc VERSION 1

Version 1 of the hc FMRI scheme has the same use as version 0, with some private FMRI member changes and additions.

Member Name	Data Type	Stability
[chassis-mfg]	string	Committed, Not an Interface
[chassis-name]	string	Committed, Not an Interface
[chassis-part]	string	Committed, Not an Interface
[chassis-serial]	string	Committed, Not an Interface
[chassis-alias]	string	Committed, Not an Interface

String form:

The same semantics apply to the hc FMRI scheme version 1 string form as version 0.

```
authstr = [:chassis-mfg=<chassis-mfg>]
                [:chassis-name=<chassis-name>]
                [:chassis-part=<chassis-part>]
                [:chassis-serial=<chassis-serial>]
                [:chassis-alias=<chassis-alias>]
```

As with hc scheme version 0 there is no leaf-level authority/identity information included in the authority nvlist with version 1 - instead such information (for example, serial/revision/part numbers) feature as top-level members of the FMRI.

Member Name	Data Type	Stability
scheme	string	Committed, value "hc"

```
version        uint8        Committed, value 1
[authority]    nvlist       Committed, See above
[fru-mfg]      string       Committed, Not-an-Interface
[fru-serial]   string       Committed, Not-an-Interface
[fru-part]     string       Committed, Not-an-Interface
[fru-revision] string       Committed, Not-an-Interface
[devid]        string       Private
[hc-root]      string       Private
hc-list-sz     uint32       Private
hc-list        nvlist array Private
[facility]     nvlist       Private
[hc-specific]  nvlist       Private
```

`[fru-mfg]`
The manufacturer string of the hardware component identified in this FMRI. This is an opaque string whose internal structure is not an interface.

`[fru-serial]`
The serial number string of the hardware component identified in this FMRI. This is an opaque string whose internal structure is not an interface.

`[fru-part]`
The part number string of the hardware component identified in this FMRI.This is an opaque string whose internal structure is not an interface.

`[fru-revision]`
The revision of the part number, as a string. Again this is an opaque string.

`[devid]`
The semantics of these Private FMRI members are not documented.

`[hc-root]`
The semantics of this Private FMRI member are not documented.

`hc-list-sz`
The semantics of this Private FMRI member are not documented.

`hc-list`
The semantics of this Private FMRI member are not documented, however we enumerate the structure of the content (subject to incompatible change). Each member of the (ordered) nvlist array has the following structure:

```
Member Name    Data Type   Stability
------------------------------------
hc-name        string      Private
hc-id          string      Private
```

`[facility]`
The semantics of this Private FMRI member are not documented, however we enumerate the structure of the content (subject to incompatible change). The facility nvlist has members as follows:

Member Name	Data Type	Stability
facility-type	string	Private
facility-name	string	Private

[hc-specific]
The semantics of this Private FMRI member are not documented.

String form:

```
authstr is defined above
hcauthstr = [:fru-mfg=<fru-mfg>]
                 [:fru-serial=<fru-serial>]
                 [:devid=<devid>]
                 [:fru-part=<fru-part>]
                 [:fru-revision=<fru-revision>]

hc-path =<hc-list[0].hc-name>=<hc-list[0].hc-id>
               /<hc-list[1].hc-name>=<hc-list[1].hc-id>
               /<hc-list[2].hc-name>=<hc-list[2].hc-id>
               ...
hcspecstr = private string formed from hc-specific nvlist,
               of the form "/name=value"

facstr = <facility.facility-type>=<facility.facility-name>

hc://[authstr][hcauthstr]/[hc-root/]<hc-path>
                                     [<hcspecstr>]
                                     [?<facstr>]
```

The high-level authority information from the authority nvlist appears adjacent to the leaf-level authority information of serial and part number, all colon separated. The components of this authority string can appear in any order, with high-level and leaf authority elements interleaved. Parsing such a string is clumsy and relies on knowing in advance which components belong to which class of authority information.

SCHEME mem VERSION 0

This scheme is used to identify memory modules and memory pages, although more recently these have been identified in the `hc` scheme. The `mem` scheme has also been used to identify cpu cache resources.

Member Name	Data Type	Stability
scheme	string	Committed, value "mem"
version	uint8	Committed, value 0
unum	string	Committed, Private
[serial]	string array	Private
[physaddr]	uint64	Private
[offset]	uint64	Private

`unum`
This string identifies the memory module, generally as labelled on the platform silkscreens or service label. The presence of this member of type string is a Committed interface, but the internal structure of the `unum` string is a `Private` interface. Therefore, you can use the opaque `unum` string to identify a memory module, but the interpretation of the components of a `unum` string is platform dependent and `Private`.

A `unum` can identify a group (or bank) of memory modules. These should instead have been identified by multiple FMRIs, one for each memory module, but this is an historical anomaly from early implementations.

`[serial]`
A string array of the serial number(s) of the memory module(s) named by the `unum` member. The internal structure of a serial number string is not an interface so you can use only the opaque string value in matching serial number and not apply any further interpretation.

`[physaddr]`
`[offset]`
The semantics of these Private FMRI members are not documented.

`String form:`

```
mem:///unum=<unum>
        [/offset=<hex-offset> | /physaddr=<hex-physaddr>]
```

SCHEME mod VERSION 0

This scheme is used to identify Oracle Solaris kernel modules.

```
Member Name    Data Type    Stability
--------------------------------------------------
scheme         string       Committed, value "fmd"
version        uint8        Committed, value 0
mod-name       string       Committed
[mod-desc]     string       Committed
[mod-pkg]      fmri         (fmri)
```

`mod-name`
The module name, as in per `modinfo`(1M).

`[mod-desc]`
The module description, again as per `modinfo`(1M).

`[mod-pkg]`
A `pkg` scheme FMRI of the package which delivers this module.

`String form:`

```
mod:///mod-name=<mod-name>
```

SCHEME pkg VERSION 0

This version of the pkg scheme corresponds to legacy SVR4 packaging as used with pkgadd(1M), pkginfo(1), pkgrm(1M), and related utilities. This scheme is universal if package name and package version conventions are adhered to.

```
Member Name   Data Type   Stability
-------------------------------------------------------------------
scheme        string      Committed, value "pkg"
version       int8        Committed, value 0
pkg-name      string      Committed
pkg-basedir   string      Committed
pkg-inst      string      Committed
pkg-version   string      Committed
```

pkg-name
The package name.

pkg-basedir
BASEDIR as per pkginfo(1)

pkg-inst
PKGINST as per pkginfo(1)

pkg-version
VERSION as per pkginfo(1)

String form:

```
pkg:///<pkg-name>/:version=<pkg-version>
```

SCHEME pkg VERSION 1

This version of the pkg scheme corresponds to the pkg(5) Image Packaging System. Such FMRIs are used in string form in the pkg(1) command line and its output, and the command line permits abbreviations of the FMRI string.

FMRIs in the pkg scheme version 1 are universal: the same FMRI interpreted in two distinct contexts (such as in distinct Oracle Solaris instances) identify the same actual package (or copies thereof).

The pkg(5) command line uses the string form of version 1 pkg FMRIs, and permits abbreviations of such strings.

```
Member Name   Data Type    Stability
-------------------------------------------------
scheme        string       Committed, value "pkg"
version       uint8        Committed, value 1
[authority]   nvlist       Committed
pkg-name      string       Committed
[pkg-version] nvlist       Committed
```

[authority]
The authority nvlist specifies the package publisher, per pkg publisher output:

Member Name	Data Type	Stability
publisher	string	Committed

publisher

The publisher of this package, such as Oracle Solaris.

pkg-name

The name (stem) of the package

[pkg-version]

Member Name	Data Type	Stability
release	string	Committed
[built-on]	string	Committed
[branch]	string	Committed
[timestamp]	string	Committed

release

The primary version of the component, written as a dot sequence string (an arbitrary-length, dot-separated series of non-zero-leading decimal digits).

built-on

The operating system version on which the component was built, again specified as a dot sequence.

branch

The vendor branch version, as a dot sequence.

timestamp

An ISO 8601 date string: *yyyymmdd*T*hhmmss*Z in which the T and Z are literals.

String form:

```
pkgauthstr = <authority.publisher>

pkg://[pkgauthstr]/<pkg-name>
    [@<release>[,<built-on>][-<branch>][:<timestamp>]
```

SCHEME svc VERSION 0

The svc scheme describes SMF (seesmf(5) services and service instances. In SMF command lines such as svcadm(1M), svccfg(1M), svcprop(1), svcs(1) and their output such FMRI are always written in their string form, and the SMF command lines permit abbreviations.

The svc scheme FMRI is a logical scheme, and, as such an FMRI should only be interpreted in the fault management domain (Oracle Solaris instance) in which it was generated.

Member Name	Data Type	Stability
scheme	string	Committed, value "svc"

```
version         uint8       Committed, value 0
svc-name        string      Committed
[svc-instance] string      Committed
[contract-id]  string      Committed
[svc-scope]    string      Committed
```

```
pkg://[pkgauthstr]/<pkg-name>
```

`svc-name`
The service name, such as `network/smtp`. See `smf(5)`.

`[svc-instance]`
Used when identifying an instance of a service. Often `default`, but can also be things like the `sendmail` instance of service `network/smtp`.

`[contract-id]`
This identifies the numeric (although written as a decimal string) contract id (see `smf(5)` and `contract(4)` used to manage a particular running instance of a service.

`[svc-scope]`
As per `smf(5)`, all service and instance objects are contained in a `scope` that represents a collection of configuration information. The configuration of the local Oracle Solaris instance is called the `localhost` scope, and is the only currently supported scope.

`String form:`

```
svc://[<svc-scope>]/<svc-name>
        [:<svc-instance>[@<contract-id>]]
```

SMF subsystem commands such as `svcadm(1M)` permit (or even require) abbreviations of the string form above. The indication of `svc` scheme at the start of the string is considered optional and implied by the SMF command line utility. When the `svc` scheme identifier is used, the authority component is also abbreviated to `svc:/<svc-name>` instead of the more-formal `svc:///<svc-name>`. For example, in SMF we write

```
svc:/network/smtp:sendmail
```

instead of the more-formal

```
svc:///network/smtp:sendmail
```

or even the more formal

```
svc://localhost/network/smtp:sendmail
```

Furthermore, SMF permits further abbreviation if it identifies a unique service or instance. For example the instance name `sendmail` is unique and one can use `svcs sendmail`, `svcadm restart sendmail`, and so forth.

Such abbreviations are a convention of the SMF subsystem and not part of the formal FMRI definition. When a `svc` scheme FMRI is received in `nvlist` form (for example, in a `libfmevent(3LIB)` subscription it appears in the more formal form when rendered as a string.

SCHEME sw VERSION 0

This FMRI scheme is used to identify software resources such as executables, library objects, and core files. This is a logical FMRI scheme.

Member Name	Data Type	Stability
scheme	string	Committed, value "sw"
version	uint8	Committed, value 0
object	nvlist	See belowPrivate
[site]	nvlist	See belowPrivate
[context]	nvlist	See belowPrivate

The `object`, `site`, and `context` members are all `Private` and so subject to incompatible change, but their content is documented below for informational purposes.

`object`

This required member identifies the software object. At this time only objects resident within the filesystem are catered for.

Member	Type	Description
path	string	Filesystem path to object
[root]	string	If present, real path to chroot root directory
[pkg]	fmri	"pkg" scheme fmri of package that delivers path

`site`

This optional member identifies a site within the object.

Member	Type	Description
[token]	string	Vendor and subsystem unique publisher token id
[module]	string	Source module information
[file]	string	Source filename (translation unit)
[func]	string	Source function
[line]	int64	Source file line number

`context`

This optional member communicates runtime information.

Member	Type	Description
[origin]	string	"userland" or "kernel"
[execname]	string	Executable name
[pid]	uint64	Process id
[thread-id]	uint64	Thread id

```
   [os-instance-uuid] string       Solaris instance UUID
   [zone]             string       Zone name, if not the global zone
   [ctid]             uint64       Contract id
   [stack]            string array Symbolic stack trace
```

String form:

```
   sw://[<authority>]/
          [:root=<object.root]
          :path=<object.path>
          [#<fragment-identifier>]
```

Slash characters / in the root and object path are not escaped.

<fragment-identifier> is one of:

```
:token=<site.token>
```

or

```
:file=<site.file>[:func=<site.func>][:line=<site.line>]
```

SCHEME zfs VERSION 0

```
Member Name   Data Type    Stability
-------------------------------------------------------------------
scheme        string       Committed, value "zfs"
version       uint8        Committed, value 0
[pool-name]   string       Committed
pool          uint64       Committed
[vdev]        uint64       Private
[vdev-name]   string       Private
```

[pool-name]
The pool name, as per `zpool list -o name`

pool
The pool GUID as per `zpool list -o guid`

[vdev]
The semantics of this `Private` FMRI member are not documented

[vdev-name]
The semantics of this `Private` FMRI member are not documented.

String form:

```
   hex-vdev = ’vdev’ in hexadecimal with no ’0x’ prefix.
   hex-pool = ’pool’ in hexadecimal with no ’0x’ prefix.
   zfs://[pool_name=<pool-name>/]pool=<hex-pool>
   zfs://[pool_name=<zfs://[pool_name=<hex-pool>
   [/vdev=<hex-vdev>][:vdev_name=<vdev-name>]
```

The <vdev-name> is written without escaping any / characters.

另请参见

hostid(1), fmadm(1M), fmd(1M), fmdump(1M), pkgadd(1M), pkginfo(1), pkgrm(1M), psradm(1M), svcadm(1M), svccfg(1M), svcprop(1), svcs(1), libfmevent(3LIB), libnvpair(3LIB), contract(4), attributes(5), smf(5)

pkg(5)

RFC 2396

引用名

fnmatch – 文件名模式匹配

描述

下面介绍的模式匹配表示法用于指定匹配 shell 中字符串的模式。以前，模式匹配表示法与正则表达式表示法相关，但与其稍有不同。出于此原因，对此模式匹配表示法规则的说明基于 regex(5) 手册页上所述的正则表达式表示法说明。

匹配单个字符的模式

以下**匹配单个字符的模式**匹配单个字符：**普通字符**、**特殊模式字符**和**模式方括号表达式**。模式方括号表达式也匹配单个整理元素。

普通字符是与其自身匹配的模式。普通字符可以是支持的字符集中的任何字符（NUL 除外），需要引号的那些特殊 shell 字符，以及以下三种特殊模式字符。匹配基于字符编码所用的位模式，而非字符的图形表示形式。如果任何字符（普通字符、shell 特殊字符或模式特殊字符）带引号，则该模式将匹配字符自身。Shell 特殊字符始终需要带引号。

当不带引号且位于方括号表达式之外时，以下三个字符在模式指定中具有特殊含义：

? 问号是与任意字符匹配的模式。

* 星号是与多个字符匹配的模式，如下面的**匹配多个字符的模式**中所述。

[左方括号表示模式方括号表达式的开始。

regex(5) 手册页上的基本正则表达式的方括号表达式说明也适用于模式方括号表达式，但正则表达式表示法中用于表示**不匹配列表**的抑扬符 (^) 被感叹号字符 (!) 取代。以不带引号的抑扬符开头的方括号表达式不会产生指定的结果。

限制在方括号表达式中使用抑扬符是为了支持某些实现，这些实现支持使用感叹号和抑扬符作为否定字符的模式匹配。可移植应用程序必须使用类似 [\^!] 的形式来匹配这两种字符。

如果在不执行 shell 引号删除的情况下使用模式匹配（例如使用 exec 函数之一调用 find 时在 find –name 主体的参数中使用模式匹配，或在fnmatch(3C) 函数的 *pattern* 参数中使用模式匹配），可通过在特殊字符前面加上反斜杠字符来进行转义，从而使其不具有特殊含义。该转义反斜杠将被丢弃。序列 \\ 表示一个文本反斜杠。引号对普通字符、shell 特殊字符和特殊模式字符的所有要求和效果也适用于此处的转义。

这里同时介绍引号和转义是因为模式匹配必须能在三种不同情况下起作用：

- 直接调用 shell，例如在路径名扩展或在 case 语句中。以下所有项都将匹配字符串或文件 abc：

abc	"abc"	a"b"c	a\bc	a[b]c
a["b"]c	a[\b]c	a["\b"]c	a?c	a*c

以下项则不然：

"a?c"	a*c	a\[b]c

- 不通过 shell 调用实用程序或函数，如 find(1) 和函数 fnmatch(3C) 中所述
- 通过 shell 命令行调用 find、cpio、tar 或 pax 等实用程序在这种情况下，在实用程序看到参数之前会执行 shell 引号删除。例如，在以下命令中：

  find /bin -name e\c[\h]o -print

  删除引号后，反斜杠将呈现给 find，后者会将其视为转义字符。两个反斜杠都在普通字符之前，因此 c 和 h 表示自身，可在许多以前的系统上找到 echo（包含在 /bin 中）。要查找包含 shell 特殊字符或模式字符（需要引号和转义二者）的文件名，例如：

  ```
  pax -r . . . "*a\ ( \?"
  ```

  将提取以 a(? 结尾的文件名。

必须在支持的应用程序中引用或转义 shell 特殊字符（有时称为元字符）。如果在不带这种保护的情况下使用，则会产生语法错误，或触发实现扩展。例如，KornShell 支持在模式中使用基于括号的一系列扩展，请参见 ksh(1)

匹配多个字符的模式

以下规则用于根据**匹配单个字符的模式**构造**匹配多个字符的模式**：

- 星号 (*) 是一种匹配任意字符串（包括 null 字符串）的模式。
- 串联**匹配单个字符的模式**是有效模式，该模式将匹配单个字符的串联或匹配每个串联模式所匹配的整理元素。
- 将一个或多个**匹配单个字符的模式**与一个或多个星号串联也是有效模式。在此类模式中，每个星号将匹配由零个或更多个字符组成的字符串，在匹配尽可能多的字符的同时，仍允许模式的其余部分与字符串匹配。

因为每个星号都有零个或更多个匹配项，所以模式 a*b 和 a**b 具有完全相同的作用。

示例：

a[bc]	匹配字符串 ab 和 ac。
a*d	匹配字符串 ad、abd 和 abcd，但不匹配字符串 abc。
a*d*	匹配字符串 ad、abcd、abcdef、aaaad 和 adddd。
*a*d	匹配字符串 ad、abcd、efabcd、aaaad 和 adddd。

用于文件名扩展的模式

将模式匹配表示法用于文件名扩展时，**匹配多个字符的模式**和**匹配单个字符的模式**中的上述规则遵循以下规则。

1. 路径名中的斜杠字符必须用模式中的一个或多个斜杠来明确匹配；它不能用星号或问号特殊字符或方括号表达式来匹配。模式中的斜杠在方括号表达式之前识别；因此，不能在用于文件名扩展的模式方括号表达式中包含斜杠。例如，模式 a[b/c]d 不会匹配诸如 abd 或 a/d 之类的路径名。它只匹配 a[b/c]d 字面上指定的路径名。
2. 如果文件名以句点 (.) 开头，则必须将句点用作模式的第一个字符或紧接在斜杠字符后的字符来明确匹配该句点。前导句点不会用以下项匹配：

   - 星号或问号特殊字符
   - 包含不匹配列表的方括号表达式，例如：

   ```
   [!a]
   ```

   范围表达式，例如：

   ```
   [%–0]
   ```

   或字符类表达式，例如：

   ```
   [[:punct:]]
   ```

   未指定诸如以下方括号表达式匹配列表中的显式句点：

   ```
   [.abc]
   ```

   是否可以匹配文件名中的前导句点。
3. 指定的模式将相应地与现有文件名和路径名匹配。包含模式字符的每个组成部分都要求对包含该组成部分的目录有读取权限。任何不包含模式字符的组成部分（最后一个组成部分除外）都要求有搜索权限。例如，以下给定模式：

   ```
   /foo/bar/x*/bam
   ```

   要求对目录 / 和 foo 具有搜索权限，对目录 bar 具有搜索和读取权限，对每个 x* 目录具有搜索权限。

   如果模式匹配任何现有文件名或路径名，则模式将替换为这些文件名和路径名，根据当前语言环境中有效的整理序列进行排序。如果模式包含无效方括号表达式或不匹配任何现有文件名或路径名，则模式字符串将保留不变。

另请参见

find(1)、ksh(1)、fnmatch(3C)、regex(5)

引用名

formats – file format notation

描述

Utility descriptions use a syntax to describe the data organization within files—stdin, stdout, stderr, input files, and output files—when that organization is not otherwise obvious. The syntax is similar to that used by the `printf(3C)` function. When used for stdin or input file descriptions, this syntax describes the format that could have been used to write the text to be read, not a format that could be used by the `scanf(3C)` function to read the input file.

Format

The description of an individual record is as follows:

```
"<format>", [<arg1>, <arg2>, . . ., <argn>]
```

The `format` is a character string that contains three types of objects defined below:

characters — Characters that are not *escape sequences* or *conversion specifications*, as described below, are copied to the output.

escape sequences — Represent non-graphic characters.

conversion specifications — Specifies the output format of each argument. (See below.)

The following characters have the following special meaning in the format string:

„ (An empty character position.) One or more blank characters.

/\ Exactly one space character.

The notation for spaces allows some flexibility for application output. Note that an empty character position in `format` represents one or more blank characters on the output (not *white space*, which can include newline characters). Therefore, another utility that reads that output as its input must be prepared to parse the data using `scanf(3C)`, `awk(1)`, and so forth. The character is used when exactly one space character is output.

Escape Sequences

The following table lists escape sequences and associated actions on display devices capable of the action.

Sequence	Character	Terminal Action
`\\`	backslash	None.
`\a`	alert	Attempts to alert the user through audible or visible notification.
`\b`	backspace	Moves the printing position to one column before the current position, unless the current position is the start of a line.
`\f`	form-feed	Moves the printing position to the initial printing position of the next logical page.

Sequence	Character	Terminal Action
`\n`	newline	Moves the printing position to the start of the next line.
`\r`	carriage-return	Moves the printing position to the start of the current line.
`\t`	tab	Moves the printing position to the next tab position on the current line. If there are no more tab positions left on the line, the behavior is undefined.
`\v`	vertical-tab	Moves the printing position to the start of the next vertical tab position. If there are no more vertical tab positions left on the page, the behavior is undefined.

Conversion Specifications

Each conversion specification is introduced by the percent-sign character (%). After the character %, the following appear in sequence:

flags — Zero or more *flags*, in any order, that modify the meaning of the conversion specification.

field width — An optional string of decimal digits to specify a minimum *field width*. For an output field, if the converted value has fewer bytes than the field width, it is padded on the left (or right, if the left-adjustment flag (–), described below, has been given to the field width).

precision — Gives the minimum number of digits to appear for the d, o, i, u, x or X conversions (the field is padded with leading zeros), the number of digits to appear after the radix character for the e and f conversions, the maximum number of significant digits for the g conversion; or the maximum number of bytes to be written from a string in s conversion. The precision takes the form of a period (.) followed by a decimal digit string; a null digit string is treated as zero.

conversion characters — A conversion character (see below) that indicates the type of conversion to be applied.

flags

The *flags* and their meanings are:

– The result of the conversion is left-justified within the field.

+ The result of a signed conversion always begins with a sign (+ or –).

<*space*> If the first character of a signed conversion is not a sign, a space character is prefixed to the result. This means that if the space character and + flags both appear, the space character flag is ignored.

# The value is to be converted to an alternative form. For c, d, i, u, and s conversions, the behaviour is undefined. For o conversion, it increases the

precision to force the first digit of the result to be a zero. For x or X conversion, a non-zero result has 0x or 0X prefixed to it, respectively. For e, E, f, g, and G conversions, the result always contains a radix character, even if no digits follow the radix character. For g and G conversions, trailing zeros are not removed from the result as they usually are.

0 For d, i, o, u, x, X, e, E, f, g, and G conversions, leading zeros (following any indication of sign or base) are used to pad to the field width; no space padding is performed. If the 0 and – flags both appear, the 0 flag is ignored. For d, i, o, u, x and X conversions, if a precision is specified, the 0 flag is ignored. For other conversions, the behaviour is undefined.

Conversion Characters

Each conversion character results in fetching zero or more arguments. The results are undefined if there are insufficient arguments for the format. If the format is exhausted while arguments remain, the excess arguments are ignored.

The *conversion characters* and their meanings are:

d,i,o,u,x,X The integer argument is written as signed decimal (d or i), unsigned octal (o), unsigned decimal (u), or unsigned hexadecimal notation (x and X). The d and i specifiers convert to signed decimal in the style `[−]`*dddd*. The x conversion uses the numbers and letters 0123456789abcdef and the X conversion uses the numbers and letters 0123456789ABCDEF. The *precision* component of the argument specifies the minimum number of digits to appear. If the value being converted can be represented in fewer digits than the specified minimum, it is expanded with leading zeros. The default precision is 1. The result of converting a zero value with a precision of 0 is no characters. If both the field width and precision are omitted, the implementation may precede, follow or precede and follow numeric arguments of types d, i and u with blank characters; arguments of type o (octal) may be preceded with leading zeros.

The treatment of integers and spaces is different from the printf(3C) function in that they can be surrounded with blank characters. This was done so that, given a format such as:

```
"%d\n",<foo>
```

the implementation could use a `printf()` call such as:

```
printf("%6d\n", foo);
```

and still conform. This notation is thus somewhat like `scanf()` in addition to `printf( )`.

f The floating point number argument is written in decimal notation in the style `[−]`*ddd.ddd*, where the number of digits after the radix character (shown here as a decimal point) is equal to the *precision* specification. The `LC_NUMERIC` locale category determines the radix character to use in this format. If the *precision* is omitted from the argument, six digits are written after the radix character; if the *precision* is explicitly 0, no radix character appears.

e,E

The floating point number argument is written in the style `[−]`*d.ddd*e±`dd` (the symbol ± indicates either a plus or minus sign), where there is one digit before the radix character (shown here as a decimal point) and the number of digits after it is equal to the precision. The `LC_NUMERIC` locale category determines the radix character to use in this format. When the precision is missing, six digits are written after the radix character; if the precision is 0, no radix character appears. The E conversion character produces a number with E instead of e introducing the exponent. The exponent always contains at least two digits. However, if the value to be written requires an exponent greater than two digits, additional exponent digits are written as necessary.

g,G

The floating point number argument is written in style f or e (or in style E in the case of a G conversion character), with the precision specifying the number of significant digits. The style used depends on the value converted: style g is used only if the exponent resulting from the conversion is less than −4 or greater than or equal to the precision. Trailing zeros are removed from the result. A radix character appears only if it is followed by a digit.

c

The integer argument is converted to an `unsigned char` and the resulting byte is written.

s

The argument is taken to be a string and bytes from the string are written until the end of the string or the number of bytes indicated by the *precision* specification of the argument is reached. If the precision is omitted from the argument, it is taken to be infinite, so all bytes up to the end of the string are written.

%

Write a % character; no argument is converted.

In no case does a non-existent or insufficient *field width* cause truncation of a field; if the result of a conversion is wider than the field width, the field is simply expanded to contain the conversion result. The term *field width* should not be confused with the term *precision* used in the description of %s.

One difference from the C function `printf()` is that the l and h conversion characters are not used. There is no differentiation between decimal values for type `int`, type `long`, or type `short`. The specifications %d or %i should be interpreted as an arbitrary length sequence of digits. Also, no distinction is made between single precision and double precision numbers (`float` or `double` in C). These are simply referred to as floating point numbers.

Many of the output descriptions use the term `line`, such as:

`"%s",` *<input line>*

Since the definition of `line` includes the trailing newline character already, there is no need to include a `\n` in the format; a double newline character would otherwise result.

示例

示例 1 To represent the output of a program that prints a date and time in the form Sunday, July 3, 10:02, where <*weekday*> and <*month*> are strings:

"%s,/\%s/\%d,/\%d:%.2d\n",<*weekday*>,<*month*>,<*day*>,<*hour*>,<*min*>

示例 2 To show pi written to 5 decimal places:

"pi/\=/\%.5f\n",<*value of pi*>

示例 3 To show an input file format consisting of five colon-separated fields:

"%s:%s:%s:%s:%s\n",<*arg1*>,<*arg2*>,<*arg3*>,<*arg4*>,<*arg5*>

另请参见

`awk`(1), `printf`(1), `printf`(3C), `scanf`(3C)

引用名

fsattr – extended file attributes

描述

Attributes are logically supported as files within the file system. The file system is therefore augmented with an orthogonal name space of file attributes. Any file (including attribute files) can have an arbitrarily deep attribute tree associated with it. Attribute values are accessed by file descriptors obtained through a special attribute interface. This logical view of "attributes as files" allows the leveraging of existing file system interface functionality to support the construction, deletion, and manipulation of attributes.

The special files "." and ". ." retain their accustomed semantics within the attribute hierarchy. The "." attribute file refers to the current directory and the ". ." attribute file refers to the parent directory. The unnamed directory at the head of each attribute tree is considered the "child" of the file it is associated with and the ". ." file refers to the associated file. For any non-directory file with attributes, the ". ." entry in the unnamed directory refers to a file that is not a directory.

Conceptually, the attribute model is fully general. Extended attributes can be any type of file (doors, links, directories, and so forth) and can even have their own attributes (fully recursive). As a result, the attributes associated with a file could be an arbitrarily deep directory hierarchy where each attribute could have an equally complex attribute tree associated with it. Not all implementations are able to, or want to, support the full model. Implementation are therefore permitted to reject operations that are not supported. For example, the implementation for the UFS file system allows only regular files as attributes (for example, no sub-directories) and rejects attempts to place attributes on attributes.

The following list details the operations that are rejected in the current implementation:

`link` Any attempt to create links between attribute and non-attribute space is rejected to prevent security-related or otherwise sensitive attributes from being exposed, and therefore manipulable, as regular files.

`rename` Any attempt to rename between attribute and non-attribute space is rejected to prevent an already linked file from being renamed and thereby circumventing the `link` restriction above.

`mkdir`
symlink
mknod Any attempt to create a "non-regular" file in attribute space is rejected to reduce the functionality, and therefore exposure and risk, of the initial implementation.

The entire available name space has been allocated to "general use" to bring the implementation in line with the NFSv4 draft standard [NFSv4]. That standard defines "named attributes" (equivalent to Solaris Extended Attributes) with no naming restrictions. All Sun applications making use of opaque extended attributes will use the prefix "SUNW".

Shell-level API

The command interface for extended attributes is the set of applications provided by Solaris for the manipulation of attributes from the command line. This interface consists of a set of existing utilities that have been extended to be "attribute-aware", plus the `runat` utility designed to "expose" the extended attribute space so that extended attributes can be manipulated as regular files.

The `-@` option enable utilities to manipulate extended attributes. As a rule, this option enables the utility to enter into attribute space when the utility is performing a recursive traversal of file system space. This is a fully recursive concept. If the underlying file system supports recursive attributes and directory structures, the `-@` option opens these spaces to the file tree-walking algorithms.

The following utilities accommodate extended attributes (see the individual manual pages for details):

`cp` By default, `cp` ignores attributes and copies only file data. This is intended to maintain the semantics implied by `cp` currently, where attributes (such as owner and mode) are not copied unless the `-p` option is specified. With the `-@` (or `-p`) option, `cp` attempts to copy all attributes along with the file data.

`cpio` The `-@` option informs `cpio` to archive attributes, but by default `cpio` ignores extended attributes. See `Extended Archive Formats` below for a description of the new archive records.

`du` File sizes computed include the space allocated for any extended attributes present.

`find` By default, `find` ignores attributes. The `-xattr` expression provides support for searches involving attribute space. It returns true if extended attributes are present on the current file.

`fsck` The `fsck` utility manages extended attribute data on the disk. A file system with extended attributes can be mounted on versions of Solaris that are not attribute-aware (versions prior to Solaris 9), but the attributes will not be accessible and `fsck` will strip them from the files and place them in `lost+found`. Once the attributes have been stripped the file system is completely stable on Solaris versions that are not attribute-aware, but would now be considered corrupted on attribute-aware versions of Solaris. The attribute-aware `fsck` utility should be run to stabilize the file system before using it in an attribute-aware environment.

`fsdb` This `fsdb` utility is able to find the inode for the "hidden" extended attribute directory.

`ls` The `ls -@` command displays an "@" following the mode information when extended attributes are present. More precisely, the output line for a given file contains an "@" character following the mode characters if the `pathconf(2)` variable `XATTR_EXISTS` is set to true. See the `pathconf()` section below. The `-@` option uses the same general output format as the `-l` option.

mv — When a file is moved, all attributes are carried along with the file rename. When a file is moved across a file system boundary, the copy command invoked is similar to the cp -p variant described above and extended attributes are "moved". If the extended file attributes cannot be replicated, the move operation fails and the source file is not removed.

pax — The -@ option informs pax to archive attributes, but by default pax ignores extended attributes. The pax(1) utility is a generic replacement for both tar(1) and cpio(1) and is able to produce either output format in its archive. See Extended Archive Formats below for a description of the new archive records.

tar — In the default case, tar does not attempt to place attributes in the archive. If the -@ option is specified, however, tar traverses into the attribute space of all files being placed in the archive and attempts to add the attributes to the archive. A new record type has been introduced for extended attribute entries in tar archive files (the same is true for pax and cpio archives) similar to the way ACLs records were defined. See Extended Archive Formats below for a description of the new archive records.

There is a class of utilities (chmod, chown, chgrp) that one might expect to be modified in a manner similar to those listed above. For example, one might expect that performing chmod on a file would not only affect the file itself but would also affect at least the extended attribute directory if not any existing extended attribute files. This is not the case. The model chosen for extended attributes implies that the attribute directory and the attributes themselves are all file objects in their own right, and can therefore have independent file status attributes associated with them (a given implementation cannot support this, for example, for intrinsic attributes). The relationship is left undefined and a fine-grained control mechanism (runat(1)) is provided to allow manipulation of extended attribute status attributes as necessary.

The runat utility has the following syntax:

```
runat filename [command]
```

The runat utility executes the supplied command in the context of the "attribute space" associated with the indicated file. If no command argument is supplied, a shell is invoked. See runat(1) for details.

Application-level API

The primary interface required to access extended attributes at the programmatic level is the openat(2) function. Once a file descriptor has been obtained for an attribute file by an openat() call, all normal file system semantics apply. There is no attempt to place special semantics on read(2), write(2), ftruncate(3C), or other functions when applied to attribute file descriptors relative to "normal" file descriptors.

The set of existing attributes can be browsed by calling openat() with "." as the file name and the O_XATTR flag set, resulting in a file descriptor for the attribute directory. The list of attributes is obtained by calls to getdents(2) on the returned file descriptor. If the target file did not previously have any attributes associated with it, an empty top-level attribute directory is created for the file and subsequent `getdents()` calls will return only "." and ". .". While the owner of the parent file owns the extended attribute directory, it is not charged against its quota if the directory is empty. Attribute files themselves, however, are charged against the user quota as any other regular file.

Additional system calls have been provided as convenience functions, including `faccessat(2)`, `fchownat(2)`, `fstatat(2)`, `futimesat(2)`, `renameat(2)`, `unlinkat(2)`. These new functions, along with `openat()`, provide a mechanism to access files relative to an arbitrary point in the file system, rather than only the current working directory. This mechanism is particularly useful in situations when a file descriptor is available with no path. The `openat()` function, in particular, can be used in many contexts where `chdir()` or `fchdir()` is currently required. See `chdir(2)`.

Open a file relative to a file descriptor

`int openat (int` *fd*`, const char *`*path*`, int` *oflag* `[, mode_t` *mode*`])`

The `openat(2)` function behaves exactly as `open(2)` except when given a relative path. Where `open()` resolves a relative path from the current working directory, `openat()` resolves the path based on the vnode indicated by the supplied file descriptor. When *oflag* is `O_XATTR`, `openat()` interprets the *path* argument as an extended attribute reference. The following code fragment uses `openat()` to examine the attributes of some already opened file:

```
dfd = openat(fd, ".", O_RDONLY|O_XATTR);
(void)getdents(dfd, buf, nbytes);
```

If `openat()` is passed the special value `AT_FDCWD` as its first (*fd*) argument, its behavior is identical to `open()` and the relative path arguments are interpreted relative to the current working directory. If the `O_XATTR` flag is provided to `openat()` or to `open()`, the supplied path is interpreted as a reference to an extended attribute on the current working directory.

Unlink a file relative to a directory file descriptor

`int unlinkat (int` *dirfd*`, const char *path`*flag*`, int flag`*flag*`)`

The `unlinkat(2)` function deletes an entry from a directory. The *path* argument indicates the name of the entry to remove. If *path* an absolute path, the *dirfd* argument is ignored. If it is a relative path, it is interpreted relative to the directory indicated by the *dirfd* argument. If *dirfd* does not refer to a valid directory, the function returns `ENOTDIR`. If the special value `AT_FDCWD` is specified for *dirfd*, a relative path argument is resolved relative to the current working directory. If the *flag* argument is 0, all other semantics of this function are equivalent to `unlink(2)`. If *flag* is set to `AT_REMOVEDIR`, all other semantics of this function are equivalent to `rmdir(2)`.

Rename a file relative to directories

`int renameat (int` *fromfd*`, const char *`*old*`, int` *tofd*`, const char *`*new*`)`

The renameat(2) function renames an entry in a directory, possibly moving the entry into a different directory. The *old* argument indicates the name of the entry to rename. If this argument is a relative path, it is interpreted relative to the directory indicated by the *fd* argument. If it is an absolute path, the *fromfd* argument is ignored. The *new* argument indicates the new name for the entry. If this argument is a relative path, it is interpreted relative to the directory indicated by the *tofd* argument. If it is an absolute path, the *tofd* argument is ignored.

In the relative path cases, if the directory file descriptor arguments do not refer to a valid directory, the function returns ENOTDIR. All other semantics of this function are equivalent to rename(2).

If a special value AT_FDCWD is specified for either the *fromfd* or *tofd* arguments, their associated path arguments (*old* and *new*) are interpreted relative to the current working directory if they are not specified as absolute paths. Any attempt to use renameat() to move a file that is not an extended attribute into an extended attribute directory (so that it becomes an extended attribute) will fail. The same is true for an attempt to move a file that is an extended attribute into a directory that is not an extended attribute directory.

Obtain information about a file

```
int fstatat (int fd, const char *path, struct stat* buf, int flag)
```

The fstatat(2) function obtains information about a file. If the *path* argument is relative, it is resolved relative to the *fd* argument file descriptor, otherwise the *fd* argument is ignored. If the *fd* argument is a special value AT_FDCWD the path is resolved relative to the current working directory. If the *path* argument is a null pointer, the function returns information about the file referenced by the *fd* argument. In all other relative path cases, if the *fd* argument does not refer to a valid directory, the function returns ENOTDIR. If AT_SYMLINK_NOFOLLOW is set in the *flag* argument, the function will not automatically traverse a symbolic link at the position of the path. If _AT_TRIGGER is set in the *flag* argument and the vnode is a trigger mount point, the mount is performed and the function returns the attributes of the root of the mounted filesystem. The fstatat() function is a multipurpose function that can be used in place of stat(), lstat(), or fstat(). See stat(2)

The function call stat(*path*, *buf*) is identical to fstatat(AT_FDCWD, *path*, *buf*, 0).

The function call lstat(*path*, *buf*) is identical to fstatat(AT_FDCWD, *path*, *buf*, AT_SYMLINK_NOFOLLOW)

The function call fstat(*fildes*, *buf*) is identical to fstatat(*fildes*, NULL, *buf*, 0).

Set owner and group ID

```
int fchownat (int fd, const char *path, uid_t owner, gid_t group, \
          int flag)
```

The fchownat(2) function sets the owner ID and group ID for a file. If the *path* argument is relative, it is resolved relative to the *fd* argument file descriptor, otherwise the *fd* argument is ignored. If the *fd* argument is a special value AT_FDCWD the path is resolved relative to the current working directory. If the path argument is a null pointer, the function sets the owner and group ID of the file referenced by the *fd* argument. In all other relative path cases, if the *fd* argument does not refer to a valid directory, the function returns ENOTDIR. If the *flag* argument is set to AT_SYMLINK_NOFOLLOW, the function will not automatically traverse a symbolic link at the position of the path. The fchownat() function is a multi-purpose function that can be used in place of chown(), lchown(), or fchown(). See chown(2).

The function call chown(*path*, *owner*, *group*) is equivalent to fchownat(AT_FDCWD, *path*, *owner*, *group*, 0).

The function call lchown(*path*, *owner*, *group*) is equivalent to fchownat(AT_FDCWD, *path*, *owner*, *group*, AT_SYMLINK_NOFOLLOW).

Set file access and modification times

```
int futimesat (int fd, const char *path, const struct timeval \
            times[2])
```

The futimesat(2) function sets the access and modification times for a file. If the *path* argument is relative, it is resolved relative to the *fd* argument file descriptor; otherwise the *fd* argument is ignored. If the *fd* argument is the special value AT_FDCWD, the path is resolved relative to the current working directory. If the *path* argument is a null pointer, the function sets the access and modification times of the file referenced by the *fd* argument. In all other relative path cases, if the *fd* argument does not refer to a valid directory, the function returns ENOTDIR. The futimesat() function can be used in place of utimes(2).

The function call utimes(*path*, *times*) is equivalent to futimesat(AT_FDCWD, *path*, *times*).

Determine accessibility of a file

```
int faccessat(int fd, const char *path, int amode, int flag);
```

The faccessat() function checks the file named by the pathname pointed to by the *path* argument for accessibility according to the bit pattern contained in *amode*, using the real user ID in place of the effective user ID and the real group ID in place of the effective group ID. This allows a setuid process to verify that the user running it would have had permission to access this file.

If *path* specifies a relative path, the file whose accessibility is to be determined is located relative to the directory associated with the file descriptor *fd* instead of the current working directory. If *path* specifies an absolute path, the *fd* argument is ignored.

If faccessat() is passed in the *fd* parameter the special value AT_FDCWD, defined in <fcntl.h>, the current working directory is used and the behavior is identical to a call to access(2).

New pathconf() functionality

```
long int pathconf(const char *path, int name)
```

Two variables have been added to pathconf(2) to provide enhanced support for extended attribute manipulation. The XATTR_ENABLED variable allows an application to determine if attribute support is currently enabled for the file in question. The XATTR_EXISTS variable allows an application to determine whether there are any extended attributes associated with the supplied path.

Open/Create an attribute file

```
int attropen (const char *path, const char *attrpath, int oflag \
         [, mode_t mode])
```

The attropen(3C) function returns a file descriptor for the named attribute, *attrpath*, of the file indicated by *path*. The *oflag* and *mode* arguments are identical to the open(2) arguments and are applied to the open operation on the attribute file (for example, using the O_CREAT flag creates a new attribute). Once opened, all normal file system operations can be used on the attribute file descriptor. The attropen() function is a convenience function and is equivalent to the following sequence of operations:

```
fd = open (path, O_RDONLY);
attrfd = openat(fd, attrpath, oflag|O_XATTR, mode);
close(fd);
```

The set of existing attributes can be browsed by calling attropen() with "." as the attribute name. The list of attributes is obtained by calling getdents(2) (or fdopendir(3C) followed by readdir(3C), see below) on the returned file descriptor.

Convert an open file descriptor for a directory into a directory descriptor

```
DIR * fdopendir (const int fd)
```

The fdopendir(3C) function promotes a file descriptor for a directory to a directory pointer suitable for use with the readdir(3C) function. The originating file descriptor should not be used again following the call to fdopendir(). The directory pointer should be closed with a call to closedir(3C). If the provided file descriptor does not reference a directory, the function returns ENOTDIR. This function is useful in circumstances where the only available handle on a directory is a file descriptor. See attropen(3C) and openat(2).

Using the API

The following examples demonstrate how the API might be used to perform basic operations on extended attributes:

示例 **1** List extended attributes on a file.

```
attrdirfd = attropen("test", ".", O_RDONLY);
dirp = fdopendir(attrdirfd);
while (dp = readdir(dirp)) {
...
```

示例 **2** Open an extended attribute.

```
attrfd = attropen("test", dp->d_name, O_RDONLY);
```

or

```
attrfd = openat(attrdirfd, dp->d_name, O_RDONLY);
```

示例 **3** Read from an extended attribute.

```
while (read(attrfd, buf, 512) > 0) {
...
```

示例 **4** Create an extended attribute.

```
newfd = attropen("test", "attr", O_CREAT|O_RDWR);
```

or

```
newfd = openat(attrdirfd, "attr", O_CREAT|O_RDWR);
```

示例 **5** Write to an extended attribute.

```
count = write(newfd, buf, length);
```

示例 **6** Delete an extended attribute.

```
error = unlinkat(attrdirfd, "attr");
```

Applications intending to access the interfaces defined here as well as the POSIX and X/Open specification-conforming interfaces should define the macro `_ATFILE_SOURCE` to be 1 and set whichever feature test macros are appropriate to obtain the desired environment. See `standards(5)`.

Extended Archive Formats

As noted above in the description of command utilities modified to provide support for extended attributes, the archive formats for `tar(1)` and `cpio(1)` have been extended to provide support for archiving extended attributes. This section describes the specifics of the archive format extensions.

Extended tar format

The `tar` archive is made up of a series of 512 byte blocks. Each archived file is represented by a header block and zero or more data blocks containing the file contents. The header block is structured as shown in the following table.

Field Name	Length (in Octets)	Description
Name	100	File name string
Mode	8	12 file mode bits
Uid	8	User ID of file owner
Gid	8	Group ID of file owner
Size	12	Size of file
Mtime	12	File modification time
Chksum	8	File contents checksum
Typeflag	1	File type flag
Linkname	100	Link target name if file linked
Magic	6	“ustar”
Version	2	“00”
Uname	32	User name of file owner
Gname	32	Group name of file owner
Devmajor	8	Major device ID if special file
Devminor	8	Minor device ID if special file
Prefix	155	Path prefix string for file

The extended attribute project extends the above header format by defining a new header type (for the `Typeflag` field). The type 'E' is defined to be used for all extended attribute files. Attribute files are stored in the `tar` archive as a sequence of two `<header ,data>` pairs. The first file contains the data necessary to locate and name the extended attribute in the file system. The second file contains the actual attribute file data. Both files use an 'E' type header. The prefix and name fields in extended attribute headers are ignored, though they should be set to meaningful values for the benefit of archivers that do not process these headers. Solaris archivers set the prefix field to “`/dev/null`” to prevent archivers that do not understand the type 'E' header from trying to restore extended attribute files in inappropriate places.

Extended cpio format

The `cpio` archive format is octet-oriented rather than block-oriented. Each file entry in the archive includes a header that describes the file, followed by the file name, followed by the contents of the file. These data are arranged as described in the following table.

Field Name	Length (in Octets)	Description
c_magic	6	70707
c_dev	6	First half of unique file ID
c_ino	6	Second half of unique file ID
c_mode	6	File mode bits
c_uid	6	User ID of file owner
c_gid	6	Group ID of file owner
c_nlink	6	Number of links referencing file
c_rdev	6	Information for special files
c_mtime	11	Modification time of file
c_namesize	6	Length of file pathname
c_filesize	11	Length of file content
c_name	c_namesize	File pathname
c_filedata	c_filesize	File content

The basic archive file structure is not changed for extended attributes. The file type bits stored in the c_mode field for an attribute file are set to 0xB000. As with the tar archive format, extended attributes are stored in cpio archives as two consecutive file entries. The first file describes the location/name for the extended attribute. The second file contains the actual attribute file content. The c_name field in extended attribute headers is ignored, though it should be set to a meaningful value for the benefit of archivers that do not process these headers. Solaris archivers start the pathname with "/dev/null/"to prevent archivers that do not understand the type 'E' header from trying to restore extended attribute files in inappropriate places.

Attribute identification data format

Both the tar and cpio archive formats can contain the special files described above, always paired with the extended attribute data record, for identifying the precise location of the extended attribute. These special data files are necessary because there is no simple naming mechanism for extended attribute files. Extended attributes are not visible in the file system name space. The extended attribute name space must be "tunneled into" using the openat() function. The attribute identification data must support not only the flat naming structure for extended attributes, but also the possibility of future extensions allowing for attribute directory hierarchies and recursive attributes. The data file is therefore composed of a sequence of records. It begins with a fixed length header describing the content. The following table describes the format of this data file.

Field Name	Length (in Octets)	Description
h_version	7	Name file version
h_size	10	Length of data file
h_component_len	10	Total length of all path segments
h_link_comp_len	10	Total length of all link segments
path	h_component_len	Complex path
link_path	h_link_comp_len	Complex link path

As demonstrated above, the header is followed by a record describing the "path" to the attribute file. This path is composed of two or more path segments separated by a null character. Each segment describes a path rooted at the hidden extended attribute directory of the leaf file of the previous segment, making it possible to name attributes on attributes. The first segment is always the path to the parent file that roots the entire sequence in the normal name space. The following table describes the format of each segment.

Field Name	Length (in Octets)	Description
h_namesz	7	Length of segment path
h_typeflag	1	Actual file type of attribute file
h_names	h_namesz	Parent path + segment path

If the attribute file is linked to another file, the path record is followed by a second record describing the location of the referencing file. The structure of this record is identical to the record described above.

另请参见

cp(1), cpio(1), find(1), ls(1), mv(1), pax(1), runat(1), tar(1), du(1), fsck(1M), access(2), chown(2), link(2), open(2), pathconf(2), rename(2), stat(2), unlink(2), utimes(2), attropen(3C), standards(5)

引用名

grub – Oracle Solaris 上的 GRand Unified Bootloader 2 软件

描述

Oracle Solaris 操作系统的当前发行版附带了 GRUB (GRand Unified Bootloader) 2 软件。GRUB 由自由软件基金会开发并提供支持。

GRUB 手册概述（可从 `www.gnu.org` 访问）对 GRUB 进行了介绍：

> 简单地说，引导装载程序就是在计算机启动时最先运行的软件程序。引导装载程序负责装入控件并将其传输到操作系统内核软件（如 Linux 或 GNU Mach），而内核又会初始化操作系统（例如，GNU [编辑注释：或 Solaris] 系统）的其余部分。
>
> GNU GRUB 是一种十分强大的引导装载程序，可通过链式装入的方式装入各种免费以及专用的操作系统。GRUB 设计用于解决引导个人计算机的复杂性问题；该程序和此手册均与 GNU 计算机平台紧密相关，但将来可能会解决移植到其他平台的问题。注意：Oracle 已将 GRUB 移植到 Solaris 操作系统。
>
> 灵活性是 GRUB 的主要特点之一；GRUB 可识别文件系统和内核可执行文件格式，这使您可以随意装入任意操作系统，无需记录内核在磁盘上的物理位置。因此，您只需指定内核的文件名以及内核所在的驱动器和分区即可装入内核。

在 Solaris 计算机中，x86 平台上支持 GRUB。Solaris 附带的 GRUB 软件新增了一个实用程序，它们未在开源分发软件中提供：

`bootadm(1M)`　　用于管理引导归档文件以及更改 GRUB 菜单。

除了此 Solaris 特定的实用程序之外，在 GRUB 手册中还对 GRUB 软件进行了介绍，可从 Oracle Web 站点获取该手册的 PDF 版本。

因为 `bootadm(1M)` 提供了用于修改 GRUB 菜单的接口，所以大多数管理员无需手动执行 GRUB 实用程序（在具有 BIOS 固件的系统中位于 `/usr/lib/grub2/bios`，在具有 64 位 UEFI 固件的系统中位于 `/usr/lib/grub2/uefi64`）。

管理员不应手动编辑 GRUB 配置文件 `grub.cfg`，该文件位于系统根池中顶层 ZFS 数据集的 `/boot/grub` 子目录中（即 `/rpool/boot/grub`）。当管理员使用 `bootadm(1M)` 更改引导配置时，会自动生成此文件。要直接创建 GRUB 菜单项的高级管理员必须编辑补充 GRUB 配置文件 named `custom.cfg`，该文件与 `grub.cfg` 文件存储在同一目录中。请注意，在刚安装的系统上不存在此文件，因此管理员需要对其进行创建。

另请参见

`boot(1M)`、`bootadm(1M)`、`installgrub(1M)`

《安装 Oracle Solaris 11.1 系统》

《Oracle Solaris 管理：常见任务》

http://www.gnu.org/software/grub

附注

Solaris 的以前版本介绍了使用 installgrub(1M) 命令安装 GRUB 引导装载程序的方法。此命令已过时，提供此命令只为方便使用或用于灾难恢复和降级到 GRUB Legacy 作为系统引导装载程序的 Solaris 引导环境。有关更多详细信息，请参考"引导和关闭 Oracle Solaris"文档。

引用名

gss_auth_rules – overview of GSS authorization

描述

The establishment of the veracity of a user's credentials requires both authentication (Is this an authentic user?) and authorization (Is this authentic user, in fact, authorized?).

When a user makes use of Generic Security Services (GSS) versions of the `ftp` or `ssh` clients to connect to a server, the user is not necessarily authorized, even if his claimed GSS identity is authenticated, Authentication merely establishes that the user is who he says he is to the GSS mechanism's authentication system. Authorization is then required: it determines whether the GSS identity is permitted to access the specified Solaris user account.

The GSS authorization rules are as follows:

- If the mechanism of the connection has a set of authorization rules, then use those rules. For example, if the mechanism is Kerberos, then use the `krb5_auth_rules`(5), so that authorization is consistent between raw Kerberos applications and GSS/Kerberos applications.
- If the mechanism of the connection does not have a set of authorization rules, then authorization is successful if the remote user's `gssname` matches the local user's `gssname` exactly, as compared by `gss_compare_name`(3GSS).

文件

`/etc/passwd` System account file. This information may also be in a directory service. See `passwd`(4).

属性

See `attributes`(5) for a description of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed

另请参见

`ftp`(1), `ssh`(1), `gsscred`(1M), `gss_compare_name`(3GSS), `passwd`(4), `attributes`(5), `krb5_auth_rules`(5)

引用名

hal – 硬件抽象层概述

描述

硬件抽象层 (Hardware Abstraction Layer, HAL) 提供了连接到系统的各种硬件的视图。当硬件配置通过热插拔或其他机制发生更改时，此视图也将随之进行动态更新。HAL 将硬件表示为设备对象。设备对象由唯一标识符进行标识并带有一组键/值对，这些键/值对称为设备属性。有些属性源自实际硬件，有些属性由设备信息文件（.fdi 文件）合并而来，有些属性与实际设备配置相关。

HAL 通过 D-Bus 提供简单易用的 API。D-Bus 是一种 IPC 框架，在其功能中提供了使应用程序可以相互通话的系统范围的消息总线。具体来说，D-Bus 提供了异步通知，使 HAL 可以在添加和删除设备以及设备的属性发生更改时通知消息总线上的其他对等方。

在 Solaris 操作系统上，HAL 由守护进程 hald(1M) 和一组实用程序（用于添加和删除设备以及修改设备的属性）提供支持。

另请参见

hald(1M)、hal-device(1M)、hal-fdi-validate(1M)、hal-find(1M)、hal-find-by-capability(1M

http://freedesktop.org。

引用名

iconv_1250 – code set conversion tables for MS 1250 (Windows Latin 2)

描述

The following code set conversions are supported:

Code Set Conversions Supported

Code	Symbol	Target Code	Symbol	Target Output
MS 1250	win2	ISO 8859-2	iso2	ISO Latin 2
MS 1250	win2	MS 852	dos2	MS-DOS Latin 2
MS 1250	win2	Mazovia	maz	Mazovia
MS 1250	win2	DHN	dhn	Dom Handlowy Nauki

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

MS 1250 to ISO 8859-2

For the conversion of MS 1250 to ISO 8859-2, all characters not in the following table are mapped unchanged.

Conversions Performed

MS 1250	ISO 8859-2	MS 1250	ISO 8859-2
24-211	40	235	273
212	251	236	276
213	40	237	274
214	246	241	267
215	253	245	241
216	256	246-267	40
217	254	271	261
221-231	40	273	40
232	271	274	245
233	40	276	265
234	266	247	365

MS 1250 to MS 852

For the conversion of MS 1250 to MS 852, all characters not in the following table are mapped unchanged.

Conversions Performed			
MS 1250	**MS 852**	**MS 1250**	**MS 852**
200-211	40	311	220
212	346	312	250
213	40	313	323
214	227	314	267
215	233	315	326
216	246	316	327
217	215	317	322
220-231	40	320	321
232	347	321	343
233	40	322	325
234	230	323	340
235	234	324	342
236	247	325	212
237	253	326	231
240	377	327	236
241	363	330	374
242	364	331	336
243	235	332	351
244	317	333	353
245	244	334	232
246	40	335	355
247	365	336	335
250	371	337	341
251	40	340	352
252	270	341	240
253	256	342	203
254	252	343	307

Conversions Performed			
MS 1250	MS 852	MS 1250	MS 852
255	360	344	204
256	40	345	222
257	275	346	206
260	370	347	207
261	40	350	237
262	362	351	202
263	210	352	251
264	357	353	211
265-267	40	354	330
270	367	355	241
271	245	356	214
272	255	357	324
273	257	360	320
274	225	361	344
275	361	362	345
276	226	363	242
277	276	364	223
300	350	365	213
301	265	366	224
302	266	367	366
303	306	370	375
304	216	371	205
305	221	372	243
306	217	374	201
307	200	375	354
310	254	376	356

MS 1250 to Mazovia

For the conversion of MS 1250 to Mazovia, all characters not in the following table are mapped unchanged.

Conversions Performed			
MS 1250	Mazovia	MS 1250	Mazovia
200-213	40	310-311	40
214	230	312	220
215-216	40	313-320	40
217	240	321	245
220-233	40	322	40
234	236	323	243
235-236	40	324-325	40
237	246	326	231
240	377	327-333	40
241-242	40	334	232
243	234	335-336	40
244	40	337	341
245	217	340-341	40
246-252	40	342	203
253	256	343	40
254	252	344	204
255-256	40	345	40
257	241	346	215
260	370	347	207
261	361	350	40
262	40	351	202
263	222	352	221
264	40	353	211
265	346	354-355	40
266	40	356	214

Conversions Performed			
MS 1250	Mazovia	MS 1250	Mazovia
267	372	357-360	40
270	40	361	244
271	206	362	40
272	40	363	242
273	257	364	223
274-276	40	365	40
277	247	366	224
300-303	40	367	366
304	216	370-373	40
305	40	374	201
306	225	375-376	40
307	200		

MS 1250 to DHN

For the conversion of MS 1250 to DHN, all characters not in the following table are mapped unchanged.

Conversions Performed			
MS 1250	DHN	MS 1250	DHN
200-213	40	306	201
214	206	307-311	40
215-216	40	312	202
217	207	313-320	40
220-233	40	321	204
234	217	322	40
235-236	40	323	205
237	220	324-325	40
240	377	326	231
241-242	40	327-333	40
243	203	334	232

Conversions Performed			
MS 1250	DHN	MS 1250	DHN
244	40	335-336	40
245	200	337	341
246-252	40	340	40
253	256	341	240
254	252	342-345	40
255-256	40	346	212
257	210	347-351	40
260	370	352	213
261	361	353-354	40
262	40	355	241
263	214	356-360	40
264	40	361	215
265	346	362	40
266	40	363	216
267	372	364	223
270	40	365	40
271	211	366	224
272	40	367	366
273	257	370-371	40
274-276	40	372	243
277	221	373-376	40
300-305	40		

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见

iconv(1), iconv(3C), iconv(5)

引用名

iconv_1251 – code set conversion tables for MS 1251 (Windows Cyrillic)

描述

The following code set conversions are supported:

Code Set Conversions Supported				
Code	**Symbol**	**Target Code**	**Symbol**	**Target Output**
MS 1251	win5	ISO 8859-5	iso5	ISO 8859-5 Cyrillic
MS 1251	win5	KOI8-R	koi8	KOI8-R
MS 1251	win5	PC Cyrillic	alt	Alternative PC Cyrillic
MS 1251	win5	Mac Cyrillic	mac	Macintosh Cyrillic

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

MS 1251 to ISO 8859-5

For the conversion of MS 1251 to ISO 8859-5, all characters not in the following table are mapped unchanged.

Conversions Performed			
MS 1251	**ISO 8859-5**	**MS 1251**	**ISO 8859-5**
24	4	310	270
200	242	311	271
201	243	312	272
202	40	313	273
203	363	314	274
204-207	40	315	275
210	255	316	276
211	40	317	277
212	251	320	300
213	40	321	301
214	252	322	302
215	254	323	303
216	253	324	304
217	257	325	305

Conversions Performed			
MS 1251	ISO 8859-5	MS 1251	ISO 8859-5
220	362	326	306
221-227	40	327	307
230	255	330	310
231	40	331	311
232	371	332	312
233	40	333	313
234	372	334	314
235	374	335	315
236	373	336	316
237	377	337	317
241	256	340	320
242	376	341	321
243	250	342	322
244-247	40	343	323
250	241	344	324
251	40	345	325
252	244	346	326
253-254	40	347	327
255	55	350	330
256	40	351	331
257	247	352	332
260-261	40	353	333
262	246	354	334
263	366	355	335
264-267	40	356	336
270	361	357	337
271	360	360	340

Conversions Performed			
MS 1251	ISO 8859-5	MS 1251	ISO 8859-5
272	364	361	341
273	40	362	342
274	370	363	343
275	245	364	344
276	365	365	345
277	367	366	346
300	260	367	347
301	261	370	350
302	262	371	351
303	263	372	352
304	264	373	353
305	265	374	354
306	266	375	355
307	267	376	356

MS 1251 to KOI8-R

For the conversion of MS 1251 to KOI8-R , all characters not in the following table are mapped unchanged.

Conversions Performed			
MS 1251	KOI8-R	MS 1251	KOI8-R
24	4	310	351
200	261	311	352
201	262	312	353
202	40	313	354
203	242	314	355
204-207	40	315	356
210	255	316	357
211	40	317	360
212	271	320	362

Conversions Performed			
MS 1251	KOI8-R	MS 1251	KOI8-R
213	40	321	363
214	272	322	364
215	274	323	365
216	273	324	346
217	277	325	350
220	241	326	343
221-227	40	327	376
230	255	330	373
231	40	331	375
232	251	332	377
233	40	333	371
234	252	334	370
235	254	335	374
236	253	336	340
237	257	337	361
241	276	340	301
242	256	341	302
243	270	342	327
244-247	40	343	307
250	263	344	304
251	40	345	305
252	264	346	326
253-254	40	347	332
255	55	350	311
256	40	351	312
257	267	352	313
260-261	40	353	314

Conversions Performed			
MS 1251	KOI8-R	MS 1251	KOI8-R
262	266	354	315
263	246	355	316
264-267	40	356	317
270	243	357	320
271	260	360	322
272	244	361	323
273	40	362	324
274	250	363	325
275	265	364	306
276	245	365	310
277	247	366	303
300	341	367	336
301	342	370	333
302	367	371	335
303	347	372	337
304	344	373	331
305	345	374	330
306	366	375	334
307	372	376	300

MS 1251 to PC Cyrillic

For the conversion of MS 1251 to PC Cyrillic, all characters not in the following table are mapped unchanged.

Conversions Performed			
MS 1251	PC Cyrillic	MS 1251	PC Cyrillic
24	4	332	232
200-207	40	333	233
210	260	334	234
211-227	40	335	235

Conversions Performed			
MS 1251	PC Cyrillic	MS 1251	PC Cyrillic
230	260	336	236
231-247	40	337	237
250	360	340	240
251-254	40	341	241
255	55	342	242
256-267	40	343	243
270	361	344	244
271-277	40	345	245
300	200	346	246
301	201	347	247
302	202	350	250
303	203	351	251
304	204	352	252
305	205	353	253
306	206	354	254
307	207	355	255
310	210	356	256
311	211	357	257
312	212	360	340
313	213	361	341
314	214	362	342
315	215	363	343
316	216	364	344
317	217	365	345
320	220	366	346
321	221	367	347
322	222	370	350

Conversions Performed			
MS 1251	PC Cyrillic	MS 1251	PC Cyrillic
323	223	371	351
324	224	372	352
325	225	373	353
326	226	374	354
327	227	375	355
330	230	376	356
331	231		

MS 1251 to Mac Cyrillic

For the conversion of MS 1251 to Mac Cyrillic, all characters not in the following table are mapped unchanged.

Conversions Performed			
MS 1251	Mac Cyrillic	MS 1251	Mac Cyrillic
24	4	260	241
200	253	262	247
201	256	263	264
202	40	264	266
203	257	266	246
204	327	267	245
205	311	270	336
206	240	271	334
207-211	40	272	271
212	274	273	310
213	40	274	300
214	276	275	301
215	315	276	317
216	40	277	273
217	332	300	200
220	254	301	201

Conversions Performed			
MS 1251	Mac Cyrillic	MS 1251	Mac Cyrillic
221	324	302	202
222	325	303	203
223	322	304	204
224	323	305	205
225	40	306	206
226	320	307	207
227	321	310	210
230	40	311	211
231	252	312	212
232	275	313	213
233	40	314	214
234	277	315	215
235	316	316	216
236	40	317	217
237	333	320	220
240	312	321	221
241	330	322	222
242	331	323	223
243	267	324	224
244	377	325	225
245	242	326	226
246	40	327	227
247	244	330	230
250	335	331	231
252	270	332	232
253	307	333	233
254	302	334	234

Conversions Performed			
MS 1251	Mac Cyrillic	MS 1251	Mac Cyrillic
255	55	335	235
256	250	336	236
257	272	337	237
355	316		

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见

iconv(1), iconv(3C), iconv(5)

引用名

iconv – code set conversion tables

描述

The following code set conversions are supported:

Code Set Conversions Supported

Code	Symbol	Target Code	Symbol	Target Output
ISO 646	646	ISO 8859-1	8859	US ASCII
ISO 646de	646de	ISO 8859-1	8859	German
ISO 646da	646da	ISO 8859-1	8859	Danish
ISO 646en	646en	ISO 8859-1	8859	English ASCII
ISO 646es	646es	ISO 8859-1	8859	Spanish
ISO 646fr	646fr	ISO 8859-1	8859	French
ISO 646it	646it	ISO 8859-1	8859	Italian
ISO 646sv	646sv	ISO 8859-1	8859	Swedish
ISO 8859-1	8859	ISO 646	646	7 bit ASCII
ISO 8859-1	8859	ISO 646de	646de	German
ISO 8859-1	8859	ISO 646da	646da	Danish
ISO 8859-1	8859	ISO 646en	646en	English ASCII
ISO 8859-1	8859	ISO 646es	646es	Spanish
ISO 8859-1	8859	ISO 646fr	646fr	French
ISO 8859-1	8859	ISO 646it	646it	Italian
ISO 8859-1	8859	ISO 646sv	646sv	Swedish
ISO 8859-16	iso16	ISO 8859-2	iso2	ISO Latin 2
ISO 8859-2	iso2	ISO 8859-16	iso16	ISO Latin 10
ISO 8859-16	iso16	IBM 850	ibm850	IBM 850 code page
ISO 8859-16	iso16	IBM 870	ibm870	IBM 870 code page
ISO 8859-2	iso2	MS 1250	win2	Windows Latin 2
ISO 8859-2	iso2	MS 852	dos2	MS-DOS Latin 2
ISO 8859-2	iso2	Mazovia	maz	Mazovia
IBM 850	ibm850	ISO 8859-16	iso16	ISO Latin 10
IBM 870	ibm870	ISO 8859-16	iso16	ISO Latin 10
MS 1250	win2	DHN	dhn	Dom Handlowy Nauki
MS 852	dos2	ISO 8859-2	iso2	ISO Latin 2
MS 852	dos2	MS 1250	win2	Windows Latin 2
MS 852	dos2	Mazovia	maz	Mazovia
MS 852	dos2	DHN	dhn	Dom Handlowy Nauki
Mazovia	maz	ISO 8859-2	iso2	ISO Latin 2
Mazovia	maz	MS 1250	win2	Windows Latin 2
Mazovia	maz	MS 852	dos2	MS-DOS Latin 2
Mazovia	maz	DHN	dhn	Dom Handlowy Nauki
DHN	dhn	ISO 8859-2	iso2	ISO Latin 2
DHN	dhn	MS 1250	win2	Windows Latin 2
DHN	dhn	MS 852	dos2	MS-DOS Latin 2
DHN	dhn	Mazovia	maz	Mazovia
ISO 8859-5	iso5	KOI8-R	koi8	KOI8-R
ISO 8859-5	iso5	PC Cyrillic	alt	Alternative PC Cyrillic

ISO 8859-5	iso5	MS 1251	win5	Windows Cyrillic
ISO 8859-5	iso5	Mac Cyrillic	mac	Macintosh Cyrillic
KOI8-R	koi8	ISO 8859-5	iso5	ISO 8859-5 Cyrillic
KOI8-R	koi8	PC Cyrillic	alt	Alternative PC Cyrillic
KOI8-R	koi8	MS 1251	win5	Windows Cyrillic
KOI8-R	koi8	Mac Cyrillic	mac	Macintosh Cyrillic
PC Cyrillic	alt	ISO 8859-5	iso5	ISO 8859-5 Cyrillic
PC Cyrillic	alt	KOI8-R	koi8	KOI8-R
PC Cyrillic	alt	MS 1251	win5	Windows Cyrillic
PC Cyrillic	alt	Mac Cyrillic	mac	Macintosh Cyrillic
MS 1251	win5	ISO 8859-5	iso5	ISO 8859-5 Cyrillic
MS 1251	win5	KOI8-R	koi8	KOI8-R
MS 1251	win5	PC Cyrillic	alt	Alternative PC Cyrillic
MS 1251	win5	Mac Cyrillic	mac	Macintosh Cyrillic
Mac Cyrillic	mac	ISO 8859-5	iso5	ISO 8859-5 Cyrillic
Mac Cyrillic	mac	KOI8-R	koi8	KOI8-R
Mac Cyrillic	mac	PC Cyrillic	alt	Alternative PC Cyrillic
Mac Cyrillic	mac	MS 1251	win5	Windows Cyrillic

Conversions

The conversions are performed according to the tables contained in the manual pages cross-referenced in the `Index of Conversion Code Tables` below.

Index of Conversion Code Tables

Code	Target Code	See Manual Page
ISO 646	ISO 8859-1	iconv_646 (5)
ISO 646de	ISO 8859-1	
ISO 646da	ISO 8859-1	
ISO 646en	ISO 8859-1	
ISO 646es	ISO 8859-1	
ISO 646fr	ISO 8859-1	
ISO 646it	ISO 8859-1	
ISO 646sv	ISO 8859-1	
ISO 8859-1	ISO 646	iconv_8859-1 (5)
ISO 8859-1	ISO 646de	
ISO 8859-1	ISO 646da	
ISO 8859-1	ISO 646en	
ISO 8859-1	ISO 646es	

Index of Conversion Code Tables		
ISO 8859-1	ISO 646fr	
ISO 8859-1	ISO 646it	
ISO 8859-1	ISO 646sv	
ISO 8859-2	MS 1250	iconv_8859-2 (5)
ISO 8859-2	MS 852	
ISO 8859-2	Mazovia	
ISO 8859-2	DHN	
MS 1250	ISO 8859-2	iconv_1250 (5)
MS 1250	MS 852	
MS 1250	Mazovia	
MS 1250	DHN	
MS 852	ISO 8859-2	iconv_852 (5)
MS 852	MS 1250	
MS 852	Mazovia	
MS 852	DHN	
Mazovia	ISO 8859-2	iconv_maz (5)
Mazovia	MS 1250	
Mazovia	MS 852	
Mazovia	DHN	

Index of Conversion Code Tables		
Code	Target Code	See Manual Page
DHN	ISO 8859-2	iconv_dhn (5)
DHN	MS 1250	
DHN	MS 852	
DHN	Mazovia	
ISO 8859-5	KOI8-R	iconv_8859-5 (5)
ISO 8859-5	PC Cyrillic	

Index of Conversion Code Tables		
ISO 8859-5	MS 1251	
ISO 8859-5	Mac Cyrillic	
KOI8-R	ISO 8859-5	iconv_koi8-r (5)
KOI8-R	PC Cyrillic	
KOI8-R	MS 1251	
KOI8-R	Mac Cyrillic	
PC Cyrillic	ISO 8859-5	iconv_pc_cyr (5)
PC Cyrillic	KOI8-R	
PC Cyrillic	MS 1251	
PC Cyrillic	Mac Cyrillic	
MS 1251	ISO 8859-5	iconv_1251 (5)
MS 1251	KOI8-R	
MS 1251	PC Cyrillic	
MS 1251	Mac Cyrillic	
Mac Cyrillic	ISO 8859-5	iconv_mac_cyr (5)
Mac Cyrillic	KOI8-R	
Mac Cyrillic	PC Cyrillic	
Mac Cyrillic	MS 1251	

文件

`/usr/lib/iconv/*.so`
conversion modules

`/usr/lib/iconv/*.t`
Conversion tables.

`/usr/lib/iconv/geniconvtbl/binarytables/*.bt`
Conversion binary tables.

`/usr/lib/iconv/iconv_data`
List of conversions supported by conversion tables.

另请参见

iconv(1), iconv(3C), iconv_1250(5), iconv_1251(5), iconv_646(5), iconv_852(5), iconv_8859-1(5), iconv_8859-2(5), iconv_8859-5(5), iconv_dhn(5), iconv_koi8-r(5), iconv_mac_cyr(5), iconv_maz(5), iconv_pc_cyr(5), iconv_unicode(5)

引用名

iconv_646 – code set conversion tables for ISO 646

描述

The following code set conversions are supported:

Code Set Conversions Supported				
Code	**Symbol**	**Target Code**	**Symbol**	**Target Output**
ISO 646	646	ISO 8859-1	8859	US ASCII
ISO 646de	646de	ISO 8859-1	8859	German
ISO 646da	646da	ISO 8859-1	8859	Danish
ISO 646en	646en	ISO 8859-1	8859	English ASCII
ISO 646es	646es	ISO 8859-1	8859	Spanish
ISO 646fr	646fr	ISO 8859-1	8859	French
ISO 646it	646it	ISO 8859-1	8859	Italian
ISO 646sv	646sv	ISO 8859-1	8859	Swedish

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

ISO 646 (US ASCII) to ISO 8859-1

For the conversion of ISO 646 to ISO 8859-1, all characters in ISO 646 can be mapped unchanged to ISO 8859-1

ISO 646de (GERMAN) to ISO 8859-1

For the conversion of ISO 646de to ISO 8859-1, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 646de	**ISO 8859-1**	**ISO 646de**	**ISO 8859-1**
100	247	173	344
133	304	174	366
134	326	175	374
135	334	176	337

ISO 646da (DANISH) to ISO 8859-1

For the conversion of ISO 646da to ISO 8859-1, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 646da	ISO 8859-1	ISO 646da	ISO 8859-1
133	306	173	346
134	330	174	370
135	305	175	345

ISO 646en (ENGLISH ASCII) to ISO 8859-1

For the conversion of ISO 646en to ISO 8859-1, all characters not in the following table are mapped unchanged.

Conversions Performed	
ISO 646en	ISO 8859-1
043	243

ISO 646es (SPANISH) to ISO 8859-1

For the conversion of ISO 646es to ISO 8859-1, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 646es	ISO 8859-1	ISO 646es	ISO 8859-1
100	247	173	260
133	241	174	361
134	321	175	347
135	277		

ISO 646fr (FRENCH) to ISO 8859-1

For the conversion of ISO 646fr to ISO 8859-1, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 646fr	ISO 8859-1	ISO 646fr	ISO 8859-1
043	243	173	351
100	340	174	371
133	260	175	350
134	347	176	250
135	247		

ISO 646it (ITALIAN) to ISO 8859-1

For the conversion of ISO 646it to ISO 8859-1, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 646it	ISO 8859-1	ISO 646it	ISO 8859-1
043	243	140	371
100	247	173	340
133	260	174	362
134	347	175	350
135	351	176	354

ISO 646sv (SWEDISH) to ISO 8859-1

For the conversion of ISO 646sv to ISO 8859-1, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 646sv	ISO 8859-1	ISO 646sv	ISO 8859-1
100	311	140	351
133	304	173	344
134	326	174	366
135	305	175	345
136	334	176	374

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见

iconv(1), iconv(3C), iconv(5)

引用名

iconv_852 – code set conversion tables for MS 852 (MS-DOS Latin 2)

描述

The following code set conversions are supported:

Code Set Conversions Supported				
Code	**Symbol**	**Target Code**	**Symbol**	**Target Output**
MS 852	dos2	ISO 8859-2	iso2	ISO Latin 2
MS 852	dos2	MS 1250	win2	Windows Latin 2
MS 852	dos2	Mazovia	maz	Mazovia
MS 852	dos2	DHN	dhn	Dom Handlowy Nauki

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

MS 852 to ISO 8859-2

For the conversion of MS 852 to ISO 8859-2, all characters not in the following table are mapped unchanged.

Conversions Performed			
MS 852	**ISO 8859-2**	**MS 852**	**ISO 8859-2**
24-177	40	271-274	40
200	307	275	257
201	374	276	277
202	351	277-305	40
203	342	306	303
204	344	307	343
205	371	310-316	40
206	346	317	244
207	347	320	360
210	263	321	320
211	353	322	317
212	325	323	313
213	365	324	357
214	356	325	322

Conversions Performed			
MS 852	ISO 8859-2	MS 852	ISO 8859-2
215	254	326	315
216	304	327	316
217	306	330	354
220	311	331-334	40
221	305	335	336
222	345	336	331
223	364	337	40
224	366	340	323
225	245	341	337
226	265	342	324
227	246	343	321
230	266	344	361
231	326	345	362
232	334	346	251
233	253	347	271
234	273	350	300
235	243	351	332
236	327	352	340
237	350	353	333
240	341	354	375
241	355	355	335
242	363	356	376
243	372	357	264
244	241	360	255
245	261	361	275
246	256	362	262
247	276	363	267

Conversions Performed			
MS 852	ISO 8859-2	MS 852	ISO 8859-2
250	312	364	242
251	352	365	247
252	40	366	367
253	274	367	270
254	310	370	260
255	272	371	250
256-264	40	372	377
265	301	374	330
266	302	375	370
267	314	376	40
270	252		

MS 852 to MS 1250

For the conversion of MS 852 to MS 1250, all characters not in the following table are mapped unchanged.

Conversions Performed			
MS 852	MS 1250	MS 852	MS 1250
200	307	270	252
201	374	271-274	40
202	351	275	257
203	342	276	277
204	344	277-305	40
205	371	306	303
206	346	307	343
207	347	310-316	40
210	263	317	244
211	353	320	360
212	325	321	320
213	365	322	317

Conversions Performed			
MS 852	**MS 1250**	**MS 852**	**MS 1250**
214	356	323	313
215	217	324	357
216	304	325	322
217	306	326	315
220	311	327	316
221	305	330	354
222	345	331-334	40
223	364	335	336
224	366	336	331
225	274	337	40
226	276	340	323
227	214	341	337
230	234	342	324
231	326	343	321
232	334	344	361
233	215	345	362
234	235	346	212
235	243	347	232
236	327	350	300
237	350	351	332
240	341	352	340
241	355	353	333
242	363	354	375
243	372	355	335
244	245	356	376
245	271	357	264
246	216	360	255

Conversions Performed			
MS 852	MS 1250	MS 852	MS 1250
247	236	361	275
250	312	362	262
251	352	363	241
252	254	364	242
253	237	365	247
254	310	366	367
255	272	367	270
256	253	370	260
257	273	371	250
260-264	40	372	377
265	301	374	330
266	302	375	370
267	314	376	40

MS 852 to Mazovia

For the conversion of MS 852 to Mazovia, all characters not in the following table are mapped unchanged.

Conversions Performed			
MS 852	Mazovia	MS 852	Mazovia
205	40	246-247	40
206	215	250	220
210	222	251	221
212-213	40	253	246
215	240	254-270	40
217	225	275	241
220-226	40	276	247
227	230	306-336	40
230	236	340	243
233-234	40	342	40

Conversions Performed			
MS 852	Mazovia	MS 852	Mazovia
235	234	343	245
236-243	40	344	244
244	217	345-375	40
245	206		

MS 852 to DHN

For the conversion of MS 852 to DHN, all characters not in the following table are mapped unchanged.

Conversions Performed			
MS 852	DHN	MS 852	DHN
200-205	40	244	200
206	212	245	211
207	40	246-247	40
210	214	250	202
211-214	40	251	213
215	207	253	220
216	40	254-270	40
217	201	275	210
220-226	40	276	221
227	206	306-336	40
230	217	340	205
233-234	40	342	40
235	203	343	204
236-237	40	344	215
242	216	345-375	40
252	254		

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见　iconv(1), iconv(3C), iconv(5)

引用名

iconv_8859-1 – code set conversion tables for ISO 8859-1 (Latin 1)

描述

The following code set conversions are supported:

Code Set Conversions Supported				
Code	**Symbol**	**Target Code**	**Symbol**	**Target Output**
ISO 8859-1	8859	ISO 646	646	7 bit ASCII
ISO 8859-1	8859	ISO 646de	646de	German
ISO 8859-1	8859	ISO 646da	646da	Danish
ISO 8859-1	8859	ISO 646en	646en	English ASCII
ISO 8859-1	8859	ISO 646es	646es	Spanish
ISO 8859-1	8859	ISO 646fr	646fr	French
ISO 8859-1	8859	ISO 646it	646it	Italian
ISO 8859-1	8859	ISO 646sv	646sv	Swedish

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

ISO 8859-1 to ISO 646 (7-bit ASCII)

For the conversion of ISO 8859-1 to ISO 646, all characters not in the following table are mapped unchanged.

```
Converted to Underscore ’_’ (137)
---------------------------------
200 201 202 203 204 205 206 207
210 211 212 213 214 215 216 217
220 221 222 223 224 225 226 227
230 231 232 233 234 235 236 237
240 241 242 243 244 245 246 247
250 251 252 253 254 255 256 257
260 261 262 263 264 265 266 267
270 271 272 273 274 275 276 277
300 301 302 303 304 305 306 307
310 311 312 313 314 315 316 317
320 321 322 323 324 325 326 327
330 331 332 333 334 335 336 337
340 341 342 343 344 345 346 347
350 351 352 353 354 355 356 357
360 361 362 363 364 365 366 367
370 371 372 373 374 375 376 377
```

ISO 8859-1 to ISO 646de (GERMAN)

For the conversion of ISO 8859-1 to ISO 646de, all characters not in the following tables are mapped unchanged.

Conversions Performed			
ISO 8859-1	ISO 646de	ISO 8859-1	ISO 646de
247	100	337	176
304	133	344	173
326	134	366	174
334	135	374	175

```
Converted to Underscore '_' (137)
---------------------------------
100 133 134 135 173 174 175 176
200 201 202 203 204 205 206 207
210 211 212 213 214 215 216 217
220 221 222 223 224 225 226 227
230 231 232 233 234 235 236 237
240 241 242 243 244 245 246
250 251 252 253 254 255 256 257
260 261 262 263 264 265 266 267
270 271 272 273 274 275 276 277
300 301 302 303     305 306 307
310 311 312 313 314 315 316 317
320 321 322 323 324 325     327
330 331 332 333     335 336 337
340 341 342 343     345 346 347
350 351 352 353 354 355 356 357
360 361 362 363 364 365     367
370 371 372 373     375 376 377
```

ISO 8859-1 to ISO 646da (DANISH)

For the conversion of ISO 8859-1 to ISO 646da, all characters not in the following tables are mapped unchanged.

Conversions Performed			
ISO 8859-1	ISO 646da	ISO 8859-1	ISO 646da
305	135	345	175
306	133	346	173
330	134	370	174

```
Converted to Underscore '_' (137)
---------------------------------
133 134 135 173 174 175
```

```
200 201 202 203 204 205 206 207
210 211 212 213 214 215 216 217
220 221 222 223 224 225 226 227
230 231 232 233 234 235 236 237
240 241 242 243 244 245 246 247
250 251 252 253 254 255 256 257
260 261 262 263 264 265 266 267
270 271 272 273 274 275 276 277
300 301 302 303 304         307
310 311 312 313 314 315 316 317
320 321 322 323 324 325 326 327
    331 332 333 334 335 336 337
340 341 342 343 344         347
350 351 352 353 354 355 356 357
360 361 362 363 364 365 366 367
371 372 373 374         376 377
```

ISO 8859-1 to ISO 646en (ENGLISH ASCII)

For the conversion of ISO 8859-1 to ISO 646en, all characters not in the following tables are mapped unchanged.

Conversions Performed

ISO 8859-1	ISO 646en
243	043

```
Converted to Underscore '_' (137)
---------------------------------
043
200 201 202 203 204 205 206 207
210 211 212 213 214 215 216 217
220 221 222 223 224 225 226 227
230 231 232 233 234 235 236 237
240 241 242     244 245 246 247
250 251 252 253 254 255 256 257
260 261 262 263 264 265 266 267
270 271 272 273 274 275 276 277
300 301 302 303 304 305 306 307
310 311 312 313 314 315 316 317
320 321 322 323 324 325 326 327
330 331 332 333 334 335 336 337
340 341 342 343 344 345 346 347
350 351 352 353 354 355 356 357
360 361 362 363 364 365 366 367
370 371 372 373 374 375 376 377
```

ISO 8859-1 to ISO 646fr (FRENCH)

For the conversion of ISO 8859-1 to ISO 646fr, all characters not in the following tables are mapped unchanged.

Conversions Performed			
ISO 8859-1	ISO 646fr	ISO 8859-1	ISO 646fr
243	043	347	134
247	135	350	175
250	176	351	173
260	133	371	174
340	100		

```
Converted to Underscore ’_’ (137)
---------------------------------
043
100 133 134 135 173 174 175 176
200 201 202 203 204 205 206 207
210 211 212 213 214 215 216 217
220 221 222 223 224 225 226 227
230 231 232 233 234 235 236 237
240 241 242     244 245 246
    251 252 253 254 255 256 257
    261 262 263 264 265 266 267
270 271 272 273 274 275 276 277
300 301 302 303 304 305 306 307
310 311 312 313 314 315 316 317
320 321 322 323 324 325 326 327
330 331 332 333 334 335 336 337
    341 342 343 344 345 346
        352 353 354 355 356 357
360 361 362 363 364 365 366 367
370     372 373 374 375 376 377
```

ISO 8859-1 to ISO 646it (ITALIAN)

For the conversion of ISO 8859-1 to ISO 646it, all characters not in the following tables are mapped unchanged.

Conversions Performed			
ISO 8859-1	ISO 646it	ISO 8859-1	ISO 646it
243	043	350	175
247	100	351	135
260	133	354	176

Conversions Performed			
ISO 8859-1	ISO 646it	ISO 8859-1	ISO 646it
340	173	362	174
347	134	371	140

```
Converted to Underscore '_' (137)
---------------------------------
043
100 133 134 135 173 174 175 176
200 201 202 203 204 205 206 207
210 211 212 213 214 215 216 217
220 221 222 223 224 225 226 227
230 231 232 233 234 235 236 237
240 241 242     244 245 246
250 251 252 253 254 255 256 257
    261 262 263 264 265 266 267
270 271 272 273 274 275 276 277
300 301 302 303 304 305 306 307
310 311 312 313 314 315 316 317
320 321 322 323 324 325 326 327
330 331 332 333 334 335 336 337
    341 342 343 344 345 346
        352 353 354 355 356 357
360 361 362 363 364 365 366 367
370     372 373 374 375 376 377
```

ISO 8859-1 to ISO 646es (SPANISH)

For the conversion of ISO 8859-1 to ISO 646es, all characters not in the following tables are mapped unchanged.

Conversions Performed			
ISO 8859-1	ISO 646es	ISO 8859-1	ISO 646es
241	133	321	134
247	100	347	175
260	173	361	174
277	135		

```
Converted to Underscore '_' (137)
---------------------------------
100 133 134 135 173 174 175
200 201 202 203 204 205 206 207
210 211 212 213 214 215 216 217
220 221 222 223 224 225 226 227
230 231 232 233 234 235 236 237
```

```
240     242 243 244 245 246
250 251 252 253 254 255 256 257
    261 262 263 264 265 266 267
270 271 272 273 274 275 276
300 301 302 303 304 305 306 307
310 311 312 313 314 315 316 317
320     322 323 324 325 326 327
330 331 332 333 334 335 336 337
340 341 342 343 344 345 346
350 351 352 353 354 355 356 357
360     362 363 364 365 366 367
370 371 372 373 374 375 376 377
```

ISO 8859-1 to ISO 646sv (SWEDISH)

For the conversion of ISO 8859-1 to ISO 646sv, all characters not in the following tables are mapped unchanged.

Conversions Performed			
ISO 8859-1	**ISO 646sv**	**ISO 8859-1**	**ISO 646sv**
304	133	344	173
305	135	345	175
311	100	351	140
326	134	366	174
334	136	374	176

```
Converted to Underscore '_' (137)
---------------------------------
100 133 134 135 136 140
173 174 175 176
200 201 202 203 204 205 206 207
210 211 212 213 214 215 216 217
220 221 222 223 224 225 226 227
230 231 232 233 234 235 236 237
240 241 242 243 244 245 246 247
250 251 252 253 254 255 256 257
260 261 262 263 264 265 266 267
270 271 272 273 274 275 276 277
300 301 302 303         306 307
310     312 313 314 315 316 317
320 321 322 323 324 325     327
330 331 332 333     335 336 337
340 341 342 343         346 347
350     352 353 354 355 356 357
360 361 362 363 364 365     367
370 371 372 373     375 376 377
```

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见

iconv(1), iconv(3C), iconv(5)

引用名

iconv_8859-2 – code set conversion tables for ISO 8859-2 (Latin 2)

描述

The following code set conversions are supported:

Code Set Conversions Supported				
Code	**Symbol**	**Target Code**	**Symbol**	**Target Output**
ISO 8859-2	iso2	MS 1250	win2	Windows Latin 2
ISO 8859-2	iso2	MS 852	dos2	MS-DOS Latin 2
ISO 8859-2	iso2	Mazovia	maz	Mazovia
ISO 8859-2	iso2	DHN	dhn	Dom Handlowy Nauki

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

ISO 8859-2 to MS 1250

For the conversion of ISO 8859-2 to MS 1250, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 8859-2	**MS 1250**	**ISO 8859-2**	**MS 1250**
24	4	261	271
177-237	40	265	276
241	245	266	234
245	274	267	241
246	214	271	232
251	212	273	235
253	215	274	237
254	217	276	236
256	216	266	236

ISO 8859-2 to MS 852

For the conversion of ISO 8859-2 to MS 852, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 8859-2	**MS 852**	**ISO 8859-2**	**MS 852**
24	4	316	327
177-237	40	317	322
240	377	320	321
241	244	321	343
242	364	322	325
243	235	323	340
244	317	324	342
245	225	325	212
246	227	326	231
247	365	327	236
250	371	330	374
251	346	331	336
252	270	332	351
253	233	333	353
254	215	334	232
255	360	335	355
256	246	336	335
257	275	337	341
260	370	340	352
261	245	341	240
262	362	342	203
263	210	343	307
264	357	344	204
265	226	345	222
266	230	346	206
267	363	347	207
270	367	350	237

Conversions Performed			
ISO 8859-2	MS 852	ISO 8859-2	MS 852
271	347	351	202
272	255	352	251
273	234	353	211
274	253	354	330
275	361	355	241
276	247	356	214
277	276	357	324
300	350	360	320
301	265	361	344
302	266	362	345
303	306	363	242
304	216	364	223
305	221	365	213
306	217	366	224
307	200	367	366
310	254	370	375
311	220	371	205
312	250	372	243
313	323	374	201
314	267	375	354
315	326	376	356
366	367		

ISO 8859-2 to Mazovia

For the conversion of ISO 8859-2 to Mazovia, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 8859-2	Mazovia	ISO 8859-2	Mazovia
24	4	323	243

Conversions Performed			
ISO 8859-2	Mazovia	ISO 8859-2	Mazovia
177-237	40	324-325	40
240	377	326	231
241	217	327-333	40
242	40	334	232
243	234	335-336	40
244-245	40	337	341
246	230	340-341	40
247-253	40	342	203
254	240	343	40
255-256	40	344	204
257	241	345	40
260	370	346	215
261	206	347	207
262	40	350	40
263	222	351	202
264-265	40	352	221
266	236	353	211
267-273	40	354-355	40
274	246	356	214
275-276	40	357-360	40
277	247	361	244
300-303	40	362	40
304	216	363	242
305	40	364	223
306	225	365	40
307	200	366	224
310-311	40	367	366

Conversions Performed			
ISO 8859-2	Mazovia	ISO 8859-2	Mazovia
312	220	370-373	40
313-320	40	374	201
321	245	375-376	40
322	40		

ISO 8859-2 to DHN

For the conversion of ISO 8859-2 to DHN, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 8859-2	DHN	ISO 8859-2	DHN
24	4	322	40
177-237	40	323	205
240	377	324-325	40
241	200	326	231
242	40	327-333	40
243	203	334	232
244-245	40	335-336	40
246	206	337	341
247-253	40	340	40
254	207	341	240
255-256	40	342-345	40
257	210	346	212
260	370	347-351	40
261	211	352	213
262	40	353-354	40
263	214	355	241
264-265	40	356-360	40
266	217	361	215
267-273	40	362	40

Conversions Performed			
ISO 8859-2	DHN	ISO 8859-2	DHN
274	220	363	216
275-276	40	364	223
277	221	365	40
300-305	40	366	224
306	201	367	366
307-311	40	370-371	40
312	202	372	243
313-320	40	373-376	40
321	204		

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见

`iconv`(1), `iconv`(3C), `iconv`(5)

引用名

iconv_8859-5 – code set conversion tables for ISO 8859-5 (Cyrillic)

描述

The following code set conversions are supported:

Code Set Conversions Supported				
Code	**Symbol**	**Target Code**	**Symbol**	**Target Output**
ISO 8859-5	iso5	KOI8-R	koi8	KOI8-R
ISO 8859-5	iso5	PC Cyrillic	alt	Alternative PC Cyrillic
ISO 8859-5	iso5	MS 1251	win5	Windows Cyrillic
ISO 8859-5	iso5	Mac Cyrillic	mac	Macintosh Cyrillic

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

ISO 8859-5 to KOI8-R

For the conversion of ISO 8859-5 to KOI8-R, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 8859-5	**KOI8-R**	**ISO 8859-5**	**KOI8-R**
24	4	320	301
241	263	321	302
242	261	322	327
243	262	323	307
244	264	324	304
245	265	325	305
246	266	327	332
247	267	330	311
250	270	331	312
251	271	332	313
252	272	333	314
253	273	334	315
254	274	335	316
256	276	336	317

Conversions Performed			
ISO 8859-5	KOI8-R	ISO 8859-5	KOI8-R
257	277	337	320
260	341	340	322
261	342	341	323
262	367	342	324
263	347	343	325
264	344	344	306
265	345	345	310
266	366	346	303
267	372	347	336
270	351	350	333
271	352	351	335
272	353	352	337
273	354	353	331
274	355	354	330
275	356	355	334
276	357	356	300
277	360	357	321
300	362	360	260
301	363	361	243
302	364	362	241
303	365	363	242
304	346	364	244
305	350	365	245
306	343	366	246
307	376	367	247
310	373	370	250
311	375	371	251

Conversions Performed			
ISO 8859-5	KOI8-R	ISO 8859-5	KOI8-R
312	377	372	252
313	371	373	253
314	370	374	254
315	374	375	255
316	340	376	256
317	361		

ISO 8859-5 to PC Cyrillic

For the conversion of ISO 8859-5 to PC Cyrillic, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 8859-5	PC Cyrillic	ISO 8859-5	PC Cyrillic
24	4	307	227
200-240	40	310	230
241	360	311	231
242-254	40	312	232
255	260	313	233
256-257	40	314	234
260	200	315	235
261	201	316	236
262	202	317	237
263	203	320	240
264	204	321	241
265	205	322	242
266	206	323	243
267	207	324	244
270	210	325	245
271	211	326	246
272	212	327	247

Conversions Performed			
ISO 8859-5	PC Cyrillic	ISO 8859-5	PC Cyrillic
273	213	330	250
274	214	331	251
275	215	332	252
276	216	333	253
277	217	334	254
300	220	335	255
301	221	336	256
302	222	337	257
303	223	360-374	40
304	224	375	260
305	225	376	40
306	226	365	40

ISO 8859-5 to MS 1251

For the conversion of ISO 8859-5 to MS 1251, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 8859-5	MS 1251	ISO 8859-5	MS 1251
24	4	317	337
200-237	40	320	340
241	250	321	341
242	200	322	342
243	201	323	343
244	252	324	344
245	275	325	345
246	262	326	346
247	257	327	347
250	243	330	350
251	212	331	351

Conversions Performed			
ISO 8859-5	**MS 1251**	**ISO 8859-5**	**MS 1251**
252	214	332	352
253	216	333	353
254	215	334	354
255	210	335	355
256	241	336	356
257	217	337	357
260	300	340	360
261	301	341	361
262	302	342	362
263	303	343	363
264	304	344	364
265	305	345	365
266	306	346	366
267	307	347	367
270	310	350	370
271	311	351	371
272	312	352	372
273	313	353	373
274	314	354	374
275	315	355	375
276	316	356	376
277	317	357	377
300	320	360	271
301	321	361	270
302	322	362	220
303	323	363	203
304	324	364	272

Conversions Performed			
ISO 8859-5	MS 1251	ISO 8859-5	MS 1251
305	325	365	276
306	326	366	263
307	327	367	277
310	330	370	274
311	331	371	232
312	332	372	234
313	333	373	236
314	334	374	235
315	335	375	210
316	336	376	242
376	331		

ISO 8859-5 to Mac Cyrillic

For the conversion of ISO 8859-5 to Mac Cyrillic, all characters not in the following table are mapped unchanged.

Conversions Performed			
ISO 8859-5	Mac Cyrillic	ISO 8859-5	Mac Cyrillic
24	4	317	237
200-237	40	320	340
240	312	321	341
241	335	322	342
242	253	323	343
243	256	324	344
244	270	325	345
245	301	326	346
246	247	327	347
247	272	330	350
250	267	331	351
251	274	332	352

Conversions Performed			
ISO 8859-5	**Mac Cyrillic**	**ISO 8859-5**	**Mac Cyrillic**
252	276	333	353
253	40	334	354
254	315	335	355
255	40	336	356
256	330	337	357
257	332	340	360
260	200	341	361
261	201	342	362
262	202	343	363
263	203	344	364
264	204	345	365
265	205	346	366
266	206	347	367
267	207	350	370
270	210	351	371
271	211	352	372
272	212	353	373
273	213	354	374
274	214	355	375
275	215	356	376
276	216	357	337
277	217	360	334
300	220	361	336
301	221	362	254
302	222	363	257
303	223	364	271
304	224	365	317

Conversions Performed			
ISO 8859-5	Mac Cyrillic	ISO 8859-5	Mac Cyrillic
305	225	366	264
306	226	367	273
307	227	370	300
310	230	371	275
311	231	372	277
312	232	373	40
313	233	374	316
314	234	375	40
315	235	376	331
316	236		

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见

iconv(1), iconv(3C), iconv(5)

引用名

iconv_dhn – code set conversion tables for DHN (Dom Handlowy Nauki)

描述

The following code set conversions are supported:

Code Set Conversions Supported

Code	Symbol	Target Code	Symbol	Target Output
DHN	dhn	ISO 8859-2	iso2	ISO Latin 2
DHN	dhn	MS 1250	win2	Windows Latin 2
DHN	dhn	MS 852	dos2	MS-DOS Latin 2
DHN	dhn	Mazovia	maz	Mazovia

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

DHN to ISO 8859-2

For the conversion of DHN to ISO 8859-2, all characters not in the following table are mapped unchanged.

Conversions Performed

DHN	ISO 8859-2	DHN	ISO 8859-2
24-177	40	222	40
200	241	223	364
201	306	224	366
202	312	225-230	40
203	243	231	326
204	321	232	334
205	323	233-237	40
206	246	240	341
207	254	241	355
210	257	242	363
211	261	243	372
212	346	244-340	40
213	352	341	337
214	263	342-365	40

Conversions Performed			
DHN	ISO 8859-2	DHN	ISO 8859-2
215	361	366	367
216	363	367	40
217	266	370	260
220	274	371-376	40
221	277		

DHN to MS 1250

For the conversion of DHN to MS 1250, all characters not in the following table are mapped unchanged.

Conversions Performed			
DHN	MS 1250	DHN	MS 1250
200	245	233-237	40
201	306	240	341
202	312	241	355
203	243	242	363
204	321	243	372
205	323	244-251	40
206	214	252	254
207	217	253-255	40
210	257	256	253
211	271	257	273
212	346	260-340	40
213	352	341	337
214	263	342-345	40
215	361	346	265
216	363	347-360	40
217	234	361	261
220	237	362-365	40
221	277	366	367

Conversions Performed			
DHN	MS 1250	DHN	MS 1250
222	40	367	40
223	364	370	260
224	366	371	40
225-230	40	372	267
231	326	373-376	40
232	334		

DHN to MS 852

For the conversion of DHN to MS 852, all characters not in the following table are mapped unchanged.

Conversions Performed			
DHN	MS 852	DHN	MS 852
200	244	212	206
201	217	213	251
202	250	214	210
203	235	215	344
204	343	216	242
205	340	217	230
206	227	220	253
207	215	221	276
210	275	222-375	40
211	245		

DHN to Mazovia

For the conversion of DHN to Mazovia, all characters not in the following table are mapped unchanged.

Conversions Performed			
DHN	Mazovia	DHN	Mazovia
200	217	212	215
201	225	213	221

Conversions Performed			
DHN	Mazovia	DHN	Mazovia
202	220	214	222
203	234	215	244
204	245	216	242
205	243	217	236
206	230	220	246
207	240	221	247
210	241	222-247	40
211	206		

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见

iconv(1), iconv(3C), iconv(5)

引用名

iconv_koi8-r – code set conversion tables for KOI8-R

描述

The following code set conversions are supported:

Code Set Conversions Supported				
Code	**Symbol**	**Target Code**	**Symbol**	**Target Output**
KOI8-R	koi8	ISO 8859-5	iso5	ISO 8859-5 Cyrillic
KOI8-R	koi8	PC Cyrillic	alt	Alternative PC Cyrillic
KOI8-R	koi8	MS 1251	win5	Windows Cyrillic
KOI8-R	koi8	Mac Cyrillic	mac	Macintosh Cyrillic

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

KOI8-R to ISO 8859-5

For the conversion of KOI8-R to ISO 8859-5, all characters not in the following table are mapped unchanged.

Conversions Performed			
KOI8-R	**ISO 8859-5**	**KOI8-R**	**ISO 8859-5**
24	4	320	337
241	362	321	357
242	363	322	340
243	361	323	341
244	364	324	342
245	365	325	343
246	366	327	322
247	367	330	354
250	370	331	353
251	371	332	327
252	372	333	350
253	373	334	355
254	374	335	351
256	376	336	347

Conversions Performed			
KOI8-R	ISO 8859-5	KOI8-R	ISO 8859-5
257	377	337	352
260	360	340	316
261	242	341	260
262	243	342	261
263	241	343	306
264	244	344	264
265	245	345	265
266	246	346	304
267	247	347	263
270	250	350	305
271	251	351	270
272	252	352	271
273	253	353	272
274	254	354	273
275	255	355	274
276	256	356	275
277	257	357	276
300	356	360	277
301	320	361	317
302	321	362	300
303	346	363	301
304	324	364	302
305	325	365	303
306	344	366	266
307	323	367	262
310	345	370	314
311	330	371	313

Conversions Performed			
KOI8-R	ISO 8859-5	KOI8-R	ISO 8859-5
312	331	372	267
313	332	373	310
314	333	374	315
315	334	375	311
316	335	376	307
317	336		

KOI8-R to PC Cyrillic

For the conversion of KOI8-R to PC Cyrillic, all characters not in the following table are mapped unchanged.

Conversions Performed			
KOI8-R	PC Cyrillic	KOI8-R	PC Cyrillic
24	4	333	350
200-242	40	334	355
243	361	335	351
244-254	40	336	347
255	260	337	352
256-262	40	340	236
263	360	341	200
264-274	40	342	201
275	260	343	226
276-277	40	344	204
300	356	345	205
301	240	346	224
302	241	347	203
303	346	350	225
304	244	351	210
305	245	352	211
306	344	353	212

Conversions Performed			
KOI8-R	PC Cyrillic	KOI8-R	PC Cyrillic
307	243	354	213
310	345	355	214
311	250	356	215
312	251	357	216
313	252	360	217
314	253	361	237
315	254	362	220
316	255	363	221
317	256	364	222
320	257	365	223
321	357	366	206
322	340	367	202
323	341	370	234
324	342	371	233
325	343	372	207
326	246	373	230
327	242	374	235
330	354	375	231
331	353	376	227
332	247		

KOI8-R to MS 1251

For the conversion of KOI8-R to MS 1251, all characters not in the following table are mapped unchanged.

Conversions Performed			
KOI8-R	MS 1251	KOI8-R	MS 1251
24	4	317	356
200-237	40	320	357
241	220	321	377

Conversions Performed			
KOI8-R	MS 1251	KOI8-R	MS 1251
242	203	322	360
243	270	323	361
244	272	324	362
245	276	325	363
246	263	326	346
247	277	327	342
250	274	330	374
251	232	331	373
252	234	332	347
253	236	333	370
254	235	334	375
255	210	335	371
256	242	336	367
257	237	337	372
260	271	340	336
261	200	341	300
262	201	342	301
263	250	343	326
264	252	344	304
265	275	345	305
266	262	346	324
267	257	347	303
270	243	350	325
271	212	351	310
272	214	352	311
273	216	353	312
274	215	354	313

Conversions Performed			
KOI8-R	**MS 1251**	**KOI8-R**	**MS 1251**
275	210	355	314
276	241	356	315
277	217	357	316
300	376	360	317
301	340	361	337
302	341	362	320
303	366	363	321
304	344	364	322
305	345	365	323
306	364	366	306
307	343	367	302
310	365	370	334
311	350	371	333
312	351	372	307
313	352	373	330
314	353	374	335
315	354	375	331
316	355	376	327
376	227		

KOI8-R to Mac Cyrillic

For the conversion of KOI8-R to Mac Cyrillic, all characters not in the following table are mapped unchanged.

Conversions Performed			
KOI8-R	**Mac Cyrillic**	**KOI8-R**	**Mac Cyrillic**
24	4	317	356
200-237	40	320	357
240	312	321	337
241	254	322	360

Conversions Performed			
KOI8-R	**Mac Cyrillic**	**KOI8-R**	**Mac Cyrillic**
242	257	323	361
243	336	324	362
244	271	325	363
245	317	326	346
246	264	327	342
247	273	330	374
250	300	331	373
251	275	332	347
252	277	333	370
253	40	334	375
254	316	335	371
255	40	336	367
256	331	337	372
257	333	340	236
260	334	341	200
261	253	342	201
262	256	343	226
263	335	344	204
264	270	345	205
265	301	346	224
266	247	347	203
267	272	350	225
270	267	351	210
271	274	352	211
272	276	353	212
273	40	354	213
274	315	355	214

Conversions Performed			
KOI8-R	Mac Cyrillic	KOI8-R	Mac Cyrillic
275	40	356	215
276	330	357	216
277	332	360	217
300	376	361	237
301	340	362	220
302	341	363	221
303	366	364	222
304	344	365	223
305	345	366	206
306	364	367	202
307	343	370	234
310	365	371	233
311	350	372	207
312	351	373	230
313	352	374	235
314	353	375	231
315	354	376	227
316	355		

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见

`iconv(1)`, `iconv(3C)`, `iconv(5)`

引用名

iconv_mac_cyr – code set conversion tables for Macintosh Cyrillic

描述

The following code set conversions are supported:

Code Set Conversions Supported				
Code	**Symbol**	**Target Code**	**Symbol**	**Target Output**
Mac Cyrillic	mac	ISO 8859-5	iso5	ISO 8859-5 Cyrillic
Mac Cyrillic	mac	KOI8-R	koi8	KOI8-R
Mac Cyrillic	mac	PC Cyrillic	alt	Alternative PC Cyrillic
Mac Cyrillic	mac	MS 1251	win5	Windows Cyrillic

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

Mac Cyrillic to ISO 8859-5

For the conversion of Mac Cyrillic to ISO 8859-5, all characters not in the following table are mapped unchanged.

Conversions Performed			
Mac Cyrillic	**ISO 8859-5**	**Mac Cyrillic**	**ISO 8859-5**
24	4	276	252
200	260	277	372
201	261	300	370
202	262	301	245
203	263	302-311	40
204	264	312	240
205	265	313	242
206	266	314	362
207	267	315	254
210	270	316	374
211	271	317	365
212	272	320-327	40
213	273	330	256
214	274	331	376

Conversions Performed			
Mac Cyrillic	ISO 8859-5	Mac Cyrillic	ISO 8859-5
215	275	332	257
216	276	333	377
217	277	334	360
220	300	335	241
221	301	336	361
222	302	337	357
223	303	340	320
224	304	341	321
225	305	342	322
226	306	343	323
227	307	344	324
230	310	345	325
231	311	346	326
232	312	347	327
233	313	350	330
234	314	351	331
235	315	352	332
236	316	353	333
237	317	354	334
240-246	40	355	335
247	246	356	336
250-252	40	357	337
253	242	360	340
254	362	361	341
255	40	362	342
256	243	363	343
257	363	364	344

Conversions Performed			
Mac Cyrillic	ISO 8859-5	Mac Cyrillic	ISO 8859-5
260-263	40	365	345
264	366	366	346
265-266	40	367	347
267	250	370	350
270	244	371	351
271	364	372	352
272	247	373	353
273	367	374	354
274	251	375	355
275	371	376	356
375	370		

Mac Cyrillic to KOI8-R

For the conversion of Mac Cyrillic to KOI8-R, all characters not in the following table are mapped unchanged.

Conversions Performed			
Mac Cyrillic	KOI8-R	Mac Cyrillic	KOI8-R
24	4	276	272
200	341	277	252
201	342	300	250
202	367	301	265
203	347	302-311	40
204	344	312	240
205	345	313	261
206	366	314	241
207	372	315	274
210	351	316	254
211	352	317	245
212	353	320-327	40

Conversions Performed			
Mac Cyrillic	**KOI8-R**	**Mac Cyrillic**	**KOI8-R**
213	354	330	276
214	355	331	256
215	356	332	277
216	357	333	257
217	360	334	260
220	362	335	263
221	363	336	243
222	364	337	321
223	365	340	301
224	346	341	302
225	350	342	327
226	343	343	307
227	376	344	304
230	373	345	305
231	375	346	326
232	377	347	332
233	371	350	311
234	370	351	312
235	374	352	313
236	340	353	314
237	361	354	315
240-246	40	355	316
247	266	356	317
250-252	40	357	320
253	261	360	322
254	241	361	323
255	40	362	324

Conversions Performed			
Mac Cyrillic	KOI8-R	Mac Cyrillic	KOI8-R
256	262	363	325
257	242	364	306
260-263	40	365	310
264	246	366	303
265-266	40	367	336
267	270	370	333
270	264	371	335
271	244	372	337
272	267	373	331
273	247	374	330
274	271	375	334
275	251	376	300
375	370		

Mac Cyrillic to PC Cyrillic

For the conversion of Mac Cyrillic to PC Cyrillic, all characters not in the following table are mapped unchanged.

Conversions Performed			
Mac Cyrillic	PC Cyrillic	Mac Cyrillic	PC Cyrillic
24	4	355	255
240-334	40	356	256
335	360	357	257
336	361	360	340
337	357	361	341
340	240	362	342
341	241	363	343
342	242	364	344
343	243	365	345
344	244	366	346

Conversions Performed			
Mac Cyrillic	PC Cyrillic	Mac Cyrillic	PC Cyrillic
345	245	367	347
346	246	370	350
347	247	371	351
350	250	372	352
351	251	373	353
352	252	374	354
353	253	375	355
354	254	376	356
303	366		

Mac Cyrillic to MS 1251

For the conversion of Mac Cyrillic to MS 1251, all characters not in the following table are mapped unchanged.

Conversions Performed			
Mac Cyrillic	MS 1251	Mac Cyrillic	MS 1251
24	4	255	40
200	300	256	201
201	301	257	203
202	302	260-263	40
203	303	264	263
204	304	266	264
205	305	267	243
206	306	270	252
207	307	271	272
210	310	272	257
211	311	273	277
212	312	274	212
213	313	275	232
214	314	276	214

Conversions Performed			
Mac Cyrillic	MS 1251	Mac Cyrillic	MS 1251
215	315	277	234
216	316	300	274
217	317	301	275
220	320	302	254
221	321	303-306	40
222	322	307	253
223	323	310	273
224	324	311	205
225	325	312	240
226	326	313	200
227	327	314	220
230	330	315	215
231	331	316	235
232	332	317	276
233	333	320	226
234	334	321	227
235	335	322	223
236	336	323	224
237	337	324	221
240	206	325	222
241	260	326	40
242	245	327	204
243	40	330	241
244	247	331	242
245	267	332	217
246	266	333	237
247	262	334	271

Conversions Performed			
Mac Cyrillic	MS 1251	Mac Cyrillic	MS 1251
250	256	335	250
252	231	336	270
253	200	337	377
254	220	362	324

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见

`iconv`(1), `iconv`(3C), `iconv`(5)

引用名

iconv_maz – code set conversion tables for Mazovia

描述

The following code set conversions are supported:

Code Set Conversions Supported				
Code	**Symbol**	**Target Code**	**Symbol**	**Target Output**
Mazovia	maz	ISO 8859-2	iso2	ISO Latin 2
Mazovia	maz	MS 1250	win2	Windows Latin 2
Mazovia	maz	MS 852	dos2	MS-DOS Latin 2
Mazovia	maz	DHN	dhn	Dom Hanlowy Nauki

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

Mazovia to ISO 8859-2

For the conversion of Mazovia to ISO 8859-2, all characters not in the following table are mapped unchanged.

Conversions Performed			
Mazovia	**ISO 8859-2**	**Mazovia**	**ISO 8859-2**
24–177	40	230	246
200	307	231	326
201	374	232	334
202	351	233	40
203	342	234	243
204	344	235	40
205	40	236	266
206	261	237	40
207	347	240	254
210	40	241	257
211	353	242	363
212-213	40	243	323
214	356	244	361
215	346	245	321

Conversions Performed			
Mazovia	**ISO 8859-2**	**Mazovia**	**ISO 8859-2**
216	304	246	274
217	241	247	277
220	312	250-340	40
221	352	341	337
222	263	342-365	40
223	364	366	367
224	366	367	40
225	306	370	260
226-227	40	371-376	40
256	201		

Mazovia to MS 1250

For the conversion of Mazovia to MS 1250, all characters not in the following table are mapped unchanged.

Mazovia	MS 1250	Mazovia	MS 1250
200	307	236	234
201	374	237	40
202	351	240	217
203	342	241	257
204	344	242	363
205	40	243	323
206	271	244	361
207	347	245	321
210	40	246	237
211	353	247	277
212-213	40	250-251	40
214	356	252	254
215	346	253-255	40
216	304	256	253

Mazovia	MS 1250	Mazovia	MS 1250
217	245	257	273
220	312	260-340	40
221	352	341	337
222	263	342-345	40
223	364	346	265
224	366	347-360	40
225	306	361	261
226-227	40	362-365	0
230	214	366	367
231	326	367	40
232	334	370	260
233	40	371	40
234	243	372	267
235	40	373-376	40
274	212		

Mazovia to MS 852

For the conversion of Mazovia to MS 852, all characters not in the following table are mapped unchanged.

Conversions Performed			
Mazovia	MS 852	Mazovia	MS 852
205	40	234	235
206	245	235	40
210-213	40	236	230
215	206	237	40
217	244	240	215
220	250	241	275
221	251	243	340
222	210	244	344
225	217	245	343

Conversions Performed			
Mazovia	MS 852	Mazovia	MS 852
226-227	40	246	253
230	227	247	276
233	40	250-375	40
227	327		

Mazovia to DHN

For the conversion of Mazovia to DHN, all characters not in the following table are mapped unchanged.

Conversions Performed			
Mazovia	DHN	Mazovia	DHN
200-205	40	234	203
206	211	236	217
207-214	40	240	207
215	212	241	210
216	40	242	216
217	200	243	205
220	202	244	215
221	214	246	220
225	201	247	221
230	206		

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见

iconv(1), iconv(3C), iconv(5)

引用名

iconv_pc_cyr – code set conversion tables for Alternative PC Cyrillic

描述

The following code set conversions are supported:

Code Set Conversions Supported				
Code	**Symbol**	**Target Code**	**Symbol**	**Target Output**
PC Cyrillic	alt	ISO 8859-5	iso5	ISO 8859-5 Cyrillic
PC Cyrillic	alt	KOI8-R	koi8	KOI8-R
PC Cyrillic	alt	MS 1251	win5	Windows Cyrillic
PC Cyrillic	alt	Mac Cyrillic	mac	Macintosh Cyrillic

Conversions

The conversions are performed according to the following tables. All values in the tables are given in octal.

PC Cyrillic to ISO 8859-5

For the conversion of PC Cyrillic to ISO 8859-5, all characters not in the following table are mapped unchanged.

Conversions Performed			
PC Cyrillic	**ISO 8859-5**	**PC Cyrillic**	**ISO 8859-5**
24	4	231	311
200	260	232	312
201	261	233	313
202	262	234	314
203	263	235	315
204	264	236	316
205	265	237	317
206	266	240	320
207	267	241	321
210	270	242	322
211	271	243	323
212	272	244	324
213	273	245	325
214	274	246	326

Conversions Performed			
PC Cyrillic	ISO 8859-5	PC Cyrillic	ISO 8859-5
215	275	247	327
216	276	250	330
217	277	251	331
220	300	252	332
221	301	253	333
222	302	254	334
223	303	255	335
224	304	256	336
225	305	257	337
226	306	260-337	255
227	307	360	241
230	310	362-376	255

PC Cyrillic to KOI8-R

For the conversion of PC Cyrillic to KOI8-R, all characters not in the following table are mapped unchanged.

Conversions Performed			
PC Cyrillic	KOI8-R	PC Cyrillic	KOI8-R
24	4	242	327
200	341	243	307
201	342	244	304
202	367	245	305
203	347	246	326
204	344	247	332
205	345	250	311
206	366	251	312
207	372	252	313
210	351	253	314
211	352	254	315

Conversions Performed			
PC Cyrillic	KOI8-R	PC Cyrillic	KOI8-R
212	353	255	316
213	354	256	317
214	355	257	320
215	356	260-337	255
216	357	340	322
217	360	341	323
220	362	342	324
221	363	343	325
222	364	344	306
223	365	345	310
224	346	346	303
225	350	347	336
226	343	350	333
227	376	351	335
230	373	352	337
231	375	353	331
232	377	354	330
233	371	355	334
234	370	356	300
235	374	357	321
236	340	360	263
237	361	361	243
240	301	362-376	255
241	302		

PC Cyrillic to MS 1251

For the conversion of PC Cyrillic to MS 1251, all characters not in the following table are mapped unchanged.

Conversions Performed			
PC Cyrillic	**MS 1251**	**PC Cyrillic**	**MS 1251**
24	4	242	342
200	300	243	343
201	301	244	344
202	302	245	345
203	303	246	346
204	304	247	347
205	305	250	350
206	306	251	351
207	307	252	352
210	310	253	353
211	311	254	354
212	312	255	355
213	313	256	356
214	314	257	357
215	315	260-337	210
216	316	340	360
217	317	341	361
220	320	342	362
221	321	343	363
222	322	344	364
223	323	345	365
224	324	346	366
225	325	347	367
226	326	350	370
227	327	351	371
230	330	352	372
231	331	353	373

Conversions Performed			
PC Cyrillic	MS 1251	PC Cyrillic	MS 1251
232	332	354	374
233	333	355	375
234	334	356	376
235	335	357	377
236	336	360	250
237	337	361	270
240	340	362-376	210
241	341		

PC Cyrillic to Mac Cyrillic

For the conversion of PC Cyrillic to Mac Cyrillic, all characters not in the following table are mapped unchanged.

Conversions Performed			
PC Cyrillic	Mac Cyrillic	PC Cyrillic	Mac Cyrillic
24	4	341	361
240	340	342	362
241	341	343	363
242	342	344	364
243	343	345	365
244	344	346	366
245	345	347	367
246	346	350	370
247	347	351	371
250	350	352	372
251	351	353	373
252	352	354	374
253	353	355	375
254	354	356	376
255	355	357	337

Conversions Performed			
PC Cyrillic	**Mac Cyrillic**	**PC Cyrillic**	**Mac Cyrillic**
256	356	360	335
257	357	361	336
260-337	40	362-376	40
340	360		

文件

`/usr/lib/iconv/*.so`	conversion modules
`/usr/lib/iconv/*.t`	conversion tables
`/usr/lib/iconv/iconv_data`	list of conversions supported by conversion tables

另请参见

`iconv`(1), `iconv`(3C), `iconv`(5)

引用名

iconv_unicode – code set conversion tables for Unicode

描述

The following code set conversions are supported:

CODE SET CONVERSIONS SUPPORTED

FROM Code Set		TO Code Set	
Code	FROM	Target Code	TO
	Filename		Filename
	Element		Element
ISO 8859-1 (Latin 1)	8859-1	UTF-8	UTF-8
ISO 8859-2 (Latin 2)	8859-2	UTF-8	UTF-8
ISO 8859-3 (Latin 3)	8859-3	UTF-8	UTF-8
ISO 8859-4 (Latin 4)	8859-4	UTF-8	UTF-8
ISO 8859-5 (Cyrillic)	8859-5	UTF-8	UTF-8
ISO 8859-6 (Arabic)	8859-6	UTF-8	UTF-8
ISO 8859-7 (Greek)	8859-7	UTF-8	UTF-8
ISO 8859-8 (Hebrew)	8859-8	UTF-8	UTF-8
ISO 8859-9 (Latin 5)	8859-9	UTF-8	UTF-8
ISO 8859-10 (Latin 6)	8859-10	UTF-8	UTF-8
Japanese EUC	eucJP	UTF-8	UTF-8
Chinese/PRC EUC (GB 2312-1980)	gb2312	UTF-8	UTF-8
ISO-2022	iso2022	UTF-8	UTF-8
Korean EUC	ko_KR-euc	Korean UTF-8	ko_KR-UTF-8
ISO-2022-KR	ko_KR-iso2022-7	Korean UTF-8	ko_KR_UTF-8
Korean Johap (KS C 5601-1987)	ko_KR-johap	Korean UTF-8	ko_KR-UTF-8
Korean Johap (KS C 5601-1992)	ko_KR-johap92	Korean UTF-8	ko_KR-UTF-8
Korean UTF-8	ko_KR-UTF-8	Korean EUC	ko_KR-euc
Korean UTF-8	ko_KR-UTF-8	Korean Johap (KS C 5601-1987)	ko_KR-johap
Korean UTF-8	ko_KR-UTF-8	Korean Johap (KS C 5601-1992)	ko_KR-johap92
KOI8-R (Cyrillic)	KOI8-R	UCS-2	UCS-2
KOI8-R (Cyrillic)	KOI8-R	UTF-8	UTF-8
PC Kanji (SJIS)	PCK	UTF-8	UTF-8
PC Kanji (SJIS)	SJIS	UTF-8	UTF-8
UCS-2	UCS-2	KOI8-R (Cyrillic)	KOI8-R
UCS-2	UCS-2	UCS-4	UCS-4

CODE SET CONVERSIONS SUPPORTED

FROM Code Set		TO Code Set	
Code	FROM	Target Code	TO
	Filename		Filename

	Element		Element
UCS-2	UCS-2	UTF-7	UTF-7
UCS-2	UCS-2	UTF-8	UTF-8
UCS-4	UCS-4	UCS-2	UCS-2
UCS-4	UCS-4	UTF-16	UTF-16
UCS-4	UCS-4	UTF-7	UTF-7
UCS-4	UCS-4	UTF-8	UTF-8
UTF-16	UTF-16	UCS-4	UCS-4
UTF-16	UTF-16	UTF-8	UTF-8
UTF-7	UTF-7	UCS-2	UCS-2
UTF-7	UTF-7	UCS-4	UCS-4
UTF-7	UTF-7	UTF-8	UTF-8
UTF-8	UTF-8	ISO 8859-1 (Latin 1)	8859-1
UTF-8	UTF-8	ISO 8859-2 (Latin 2)	8859-2
UTF-8	UTF-8	ISO 8859-3 (Latin 3)	8859-3
UTF-8	UTF-8	ISO 8859-4 (Latin 4)	8859-4
UTF-8	UTF-8	ISO 8859-5 (Cyrillic)	8859-5
UTF-8	UTF-8	ISO 8859-6 (Arabic)	8859-6
UTF-8	UTF-8	ISO 8859-7 (Greek)	8859-7
UTF-8	UTF-8	ISO 8859-8 (Hebrew)	8859-8
UTF-8	UTF-8	ISO 8859-9 (Latin 5)	8859-9
UTF-8	UTF-8	ISO 8859-10 (Latin 6)	8859-10
UTF-8	UTF-8	Japanese EUC	eucJP
UTF-8	UTF-8	Chinese/PRC EUC (GB 2312-1980)	gb2312
UTF-8	UTF-8	ISO-2022	iso2022
UTF-8	UTF-8	KOI8-R (Cyrillic)	KOI8-R
UTF-8	UTF-8	PC Kanji (SJIS)	PCK
UTF-8	UTF-8	PC Kanji (SJIS)	SJIS
UTF-8	UTF-8	UCS-2	UCS-2
UTF-8	UTF-8	UCS-4	UCS-4
UTF-8	UTF-8	UTF-16	UTF-16
UTF-8	UTF-8	UTF-7	UTF-7
UTF-8	UTF-8	Chinese/PRC EUC (GB 2312-1980)	zh_CN.euc

CODE SET CONVERSIONS SUPPORTED

FROM Code Set		TO Code Set	
Code	FROM	Target Code	TO
	Filename		Filename
	Element		Element
UTF-8	UTF-8	ISO 2022-CN	zh_CN.iso2022-7
UTF-8	UTF-8	Chinese/Taiwan Big5	zh_TW-big5
UTF-8	UTF-8	Chinese/Taiwan EUC (CNS 11643-1992)	zh_TW-euc

UTF-8	UTF-8	ISO 2022-TW	zh_TW-iso2022-7
Chinese/PRC EUC (GB 2312-1980)	zh_CN.euc	UTF-8	UTF-8
ISO 2022-CN	zh_CN.iso2022-7	UTF-8	UTF-8
Chinese/Taiwan Big5	zh_TW-big5	UTF-8	UTF-8
Chinese/Taiwan EUC (CNS 11643-1992)	zh_TW-euc	UTF-8	UTF-8
ISO 2022-TW	zh_TW-iso2022-7	UTF-8	UTF-8

示例

示例 1 The library module filename

In the conversion library, `/usr/lib/iconv` (see iconv(3C)), the library module filename is composed of two symbolic elements separated by the percent sign (`%`). The first symbol specifies the code set that is being converted; the second symbol specifies the *target code*, that is, the code set to which the first one is being converted.

In the conversion table above, the first symbol is termed the “FROM Filename Element”. The second symbol, representing the target code set, is the “TO Filename Element”.

For example, the library module filename to convert from the *Korean EUC* code set to the *Korean UTF-8* code set is

```
ko_KR-euc%ko_KR-UTF-8
```

文件

`/usr/lib/iconv/*.so` conversion modules

另请参见

iconv(1), iconv(3C), iconv(5)

Chernov, A., Registration of a Cyrillic Character Set, RFC 1489, RELCOM Development Team, July 1993.

Chon, K., H. Je Park, and U. Choi, Korean Character Encoding for Internet Messages, RFC 1557, Solvit Chosun Media, December 1993.

Goldsmith, D., and M. Davis, UTF-7 – A Mail-Safe Transformation Format of Unicode, RFC 1642, Taligent, Inc., July 1994.

Lee, F., HZ – A Data Format for Exchanging Files of Arbitrarily Mixed Chinese and ASCII characters, RFC 1843, Stanford University, August 1995.

Murai, J., M. Crispin, and E. van der Poel, Japanese Character Encoding for Internet Messages, RFC 1468, Keio University, Panda Programming, June 1993.

Nussbacher, H., and Y. Bourvine, Hebrew Character Encoding for Internet Messages, RFC 1555, Israeli Inter-University, Hebrew University, December 1993.

Ohta, M., Character Sets ISO-10646 and ISO-10646-J-1, RFC 1815, Tokyo Institute of Technology, July 1995.

Ohta, M., and K. Handa, ISO-2022-JP-2: Multilingual Extension of ISO-2022-JP, RFC 1554, Tokyo Institute of Technology, December 1993.

Reynolds, J., and J. Postel, ASSIGNED NUMBERS, RFC 1700, University of Southern California/Information Sciences Institute, October 1994.

Simonson, K., Character Mnemonics & Character Sets, RFC 1345, Rationel Almen Planlaegning, June 1992.

Spinellis, D., Greek Character Encoding for Electronic Mail Messages, RFC 1947, SENA S.A., May 1996.

The Unicode Consortium, The Unicode Standard, Version 2.0, Addison Wesley Developers Press, July 1996.

Wei, Y., Y. Zhang, J. Li, J. Ding, and Y. Jiang, ASCII Printable Characters-Based Chinese Character Encoding for Internet Messages, RFC 1842, AsiaInfo Services Inc., Harvard University, Rice University, University of Maryland, August 1995.

Yergeau, F., UTF-8, a transformation format of Unicode and ISO 10646, RFC 2044, Alis Technologies, October 1996.

Zhu, H., D. Hu, Z. Wang, T. Kao, W. Chang, and M. Crispin, Chinese Character Encoding for Internet Messages, RFC 1922, Tsinghua University, China Information Technology Standardization Technical Committee (CITS), Institute for Information Industry (III), University of Washington, March 1996.

附注

ISO 8859 character sets using Latin alphabetic characters are distinguished as follows:

`ISO 8859-1 (Latin 1)` For most West European languages, including:

Albanian
Catalan
Danish
Dutch
English
Faeroese
Finnish
French
German
Galician
Irish
Icelandic
Italian
Norwegian
Portuguese
Spanish
Swedish

`ISO 8859-2 (Latin 2)` For most Latin-written Slavic and Central European languages:

Czech
Polish
Slovak

German	Rumanian	Slovene
Hungarian	Croatian	

ISO 8859-3 (Latin 3) Popularly used for Esperanto, Galician, Maltese, and Turkish.

ISO 8859-4 (Latin 4) Introduces letters for Estonian, Latvian, and Lithuanian. It is an incomplete predecessor of ISO 8859-10 (Latin 6).

ISO 8859-9 (Latin 5) Replaces the rarely needed Icelandic letters in ISO 8859-1 (Latin 1) with the Turkish ones.

ISO 8859-10 (Latin 6) Adds the last Inuit (Greenlandic) and Sami (Lappish) letters that were not included in ISO 8859-4 (Latin 4) to complete coverage of the Nordic area.

引用名

ieee802.11 – 802.11 kernel statistics

描述

This page describes the kernel statistics that can be used to monitor attributes specific to the 802.11 physical layer. These statistics can be retrieved using `kstat`(1M). Not all 802.11 devices will support all statistics.

`tx_frags` Count of data and management fragments transmitted.

`rx_frags` Count of data and management fragments received.

`rx_dups` Count of duplicate frames received. Duplicates are determined by the sequence control field.

`mcast_tx` Count of broadcast and multicast frames transmitted.

`mcast_rx` Count of broadcast and multicast frames received.

`tx_failed` Count of frames that could not be transmitted due to the retransmission limit being reached.

`tx_retrans` Count of frames successfully retransmitted after one or more retransmissions.

`tx_reretrans` Count of frames successfully retransmitted after more than one retransmission.

`rts_success` Count of times a CTS was received in response to an RTS.

`rts_failure` Count of times a CTS was not received in response to an RTS.

`ack_failure` Count of times an ACK was expected but was not received.

`fcs_errors` Count of frames received with FCS errors.

`wep_errors` Count of frames received with the WEP bit set but that either should not have been encrypted or that were discarded due to WEP not being supported.

另请参见

`kstat`(1M)

引用名

ieee802.3, cap_autoneg, cap_1000fdx, cap_1000hdx, cap_100fdx, cap_100hdx, cap_10fdx, cap_10hdx, cap_rem_fault, cap_pause, cap_asym_pause, adv_cap_autoneg, adv_cap_1000fdx, adv_cap_1000hdx, adv_cap_100fdx, adv_cap_100hdx, adv_cap_10fdx, adv_cap_10hdx, adv_cap_pause, adv_cap_asym_pause, adv_rem_fault, lp_cap_autoneg, lp_cap_1000fdx, lp_cap_1000hdx, lp_cap_100fdx, lp_cap_100hdx, lp_cap_10fdx, lp_cap_10hdx, lp_cap_pause, lp_cap_asym_pause, lp_rem_fault, xcvr_addr, xcvr_id, xcvr_inuse, link_up, link_duplex, link_tx_pause, link_rx_pause – Ethernet mii kstat and dladm parameters

描述

This page describes the kernel statistics and the `dladm`(1M) configuration parameters used to monitor and configure the Ethernet physical layer.

The `cap_*` parameters exist in the kernel statistics for an Ethernet device. The parameters describe the maximum capability of a device. When the value of a statistic is `1`, the device has the capability described. When the value is `0`, the device does not have the capability.

The exceptions to this rule are the `cap_asym_pause` and `cap_pause` parameters which are explained later in this page.

`cap_autoneg`	Capable of auto-negotiation
`cap_1000fdx`	Capable of 1000 full duplex operation
`cap_1000hdx`	Capable of 1000 half duplex operation
`cap_100fdx`	Capable of 100 full duplex operation
`cap_100hdx`	Capable of 100 half duplex operation
`cap_10fdx`	Capable of 10 full duplex operation
`cap_10hdx`	Capable of 10 half duplex operation
`cap_rem_fault`	Capable of reporting locally detected faults to link partner

The `adv_cap_*` parameters exist in the kernel statistics and represent a mirror image of the `dladm adv_*_cap` parameter list for an Ethernet device. The `dladm adv_*_cap` tuning parameters allow fine grain control of the Ethernet device physical layer. The parameters are also a subset of the `cap_*` statistics. If the `cap_*` value is `0`, the corresponding `adv_cap_*` must also be `0`. The exceptions to this rule are the `adv_cap_asym_pause` and `adv_cap_pause` parameters.

When auto-negotiation is enabled, the `adv_*_cap` statistics show which capabilities are advertised to the link partner. When auto-negotiation is disabled in *forced mode*, the statistics precisely show how a link should function and that it must be matched on the link partner to achieve a valid link up.

Statistics with values other than `0` and `1` are also described in the following.

`adv_cap_autoneg` Advertise auto-negotiation capability

`adv_cap_1000fdx` Advertise 1000 full duplex capability

`adv_cap_1000hdx` Advertise 1000 half duplex capability

`adv_cap_100fdx` Advertise 100 full duplex capability

`adv_cap_100hdx` Advertise 100 half duplex capability

`adv_cap_10fdx` Advertise 10 full duplex capability

`adv_cap_10hdx` Advertise 10 half duplex capability

`adv_rem_fault` Fault value reported by the local system to the peer

- `0` Link is good
- `1` Fault

The `lp_cap_*` parameters exist as kernel statistics for an Ethernet device. The statistics are the advertised capabilities provided by the link partner on completion of auto-negotiation. If the capabilities match the capabilities provided in the local advertisement, the link can proceed to a link up state. If no match is found, the link remains down. In two other instances, `lp_cap_*` values might all be zero: (1) when a cable is not present and (2) when forced mode is enabled.

`lp_cap_autoneg` Link partner advertises auto-negotiation capability

`lp_cap_1000fdx` Link partner advertises 1000 full duplex capability

`lp_cap_1000hdx` Link partner advertises 1000 half duplex capability

`lp_cap_100fdx` Link partner advertises 100 full duplex capability

`lp_cap_100hdx` Link partner advertises 100 half duplex capability

`lp_cap_10fdx` Link partner advertises 10 full duplex capability

`lp_cap_10hdx` Link partner advertises 10 half duplex capability

`lp_rem_fault` Fault value the remote system reports

- `0` Link is good
- `1` Fault

The `xcvr_*` kernel statistics provide information about the physical layer device that is in use.

`xcvr_addr` MII address in the `0` to `31` range of the physical layer device in use for a given Ethernet device

`xcvr_id` MII transceiver manufacturer and device ID

`xcvr_inuse` MII transceiver type, based on the following list:

0 other	Undefined
1 none	MII present, but nothing connected
2 10Mb/s	10Mb/s Manchester encoding
3 100BaseT4	100 Mb/s 8B/6T
4 100BaseX	100 Mb/s 4B/5B
5 100BaseT2	100 Mb/s PAM5X5
6 1000BaseX	1000 Mb/s 8B/10B
7 1000BaseT	1000 Mb/s 4D-PAM5

The above values define maximum capability. In many cases, lower speeds can occur. The cap_* statistics must be viewed to establish the range of capability.

The link_* kernel statistics show the link state at the local end of the connection.

link_up	1	Link is up
	0	Link is down
link_duplex	2	Full duplex link
	1	Half duplex link
	0	Unknown

The cap_asym_pause, cap_pause, adv_cap_asym_pause, and adv_cap_pause parameters do not follow the rules of other cap_* and adv_cap_* kstats or parameters. The cap_*pause kstats provide information about the capabilities supported by the device and constrain the values that may be set to the corresponding adv_cap_*pause parameters.

cap_pause	Symmetric pause capability.
cap_asym_pause	Asymmetric pause capability.

The adv_cap_pause and adv_cap_asym_pause statistics are limited by the available settings for cap_pause and cap_asym_pause. These statistics are read-only values whose settings may be administratively controlled by setting the flowctrl property supported by dladm(1M). For a device that is fully capable of pausing both Rx (receive) and Tx (transmit) operations, the settings available are defined in the truth table that follows the adv_cap_pause and adv_cap_asm_pause parameter descriptions below.

adv_cap_pause — When adv_cap_pause is 1, the device can both assert and respond to flow control. This is the pre-Gigabit, symmetric mode of operation, and implies a full (both send and receive) implementation of the PAUSE mechanism within the device. In addition, if adv_cap_asym_pause is 1, the device can operate either symmetrically or asymmetrically in either direction.

If adv_cap_pause is 0, advertised, flow-control behavior is determined by adv_cap_asym_pause. If the value of adv_cap_asym_pause is 1, the device can assert flow control, but cannot resend.

No flow control is available when bot adv_cap_pause and adv_cap_asym_pause are 0.

`adv_cap_asym_pause` Asymmetric pause capability.

The `cap_asym_pause` and `cap_pause` statistics show the capability of a device and also limit the legal setting for `adv_cap_asym_pause` and `adv_cap_pause`. The following truth table describes the available `adv_cap_asym_pause` and `adv_cap_pause` settings limited by `cap_asym_pause` and `cap_pause statistics`. The abbreviations below are used in the table.

CA	`cap_asym_pause`
CP	`cap_pause`
AA	`adv_cap_asym_pause`
AP	`adv_cap_pause`

CP	CA	AP	AA	Description
0	0	0	0	No pause in use.
0	0	x	x	Device not pause capable, cannot set.
0	1	0	0	Asymmetric Rx pause capable, but not advertised.
0	1	0	1	Asymmetric Rx pause capable and advertised.
0	1	1	0	Asymmetric Rx pause capable, but not advertised. Not capable of symmetric pause.
0	1	1	1	Asymmetric Rx pause capable and advertised. No symmetric pause capability or asymmetric Tx pause.
1	0	0	0	Symmetric pause capable, but not advertised.
1	0	0	1	Symmetric pause capable, advertising asymmetric Rx pause only.
1	0	1	0	Symmetric pause capable, advertising symmetric Rx and Tx pause capability.

1	0	1	1	Symmetric pause capable and advertised.
1	1	0	0	Symmetric and asymmetric pause capable, but not advertised.
1	1	0	1	Symmetric and asymmetric Tx pause capable. Only asymmetric Tx pause advertised.
1	1	1	0	Symmetric and symmetric Tx pause capable. Only symmetric pause advertised.
1	1	1	1	Asymmetric Tx pause capable and advertised.

In the cases above, an error is posted when a device driver cannot advertise. A new setting is ignored and values revert to the previous setting.

The `lp_cap_pause` and the `lp_cap_asym_pause` provide the advertised capabilities of the link partners.

`lp_cap_pause` When `lp_cap_pause` is 1, the link-partner can both assert and respond to flow control. This is the pre-Gigabit, symmetric mode of operation, and implies a full (both send and receive) implementation of the PAUSE mechanism within the device. In addition, if `lp_cap_asym_pause` is 1, the link-partner can operate either symmetrically or asymmetrically in either direction.

If `lp_cap_pause` is 0, the flow-control behavior supported by the link-partner is determined by `lp_cap_asym_pause`. If the value of `lp_cap_asym_pause` is 1, the link-partner can assert flow control, but cannot respond to any pause-frames sent to it.

No flow control is available when both `lp_cap_pause` and `lp_cap_asym_pause` are 0.

`lp_cap_asym_pause` Asymmetric pause capability

When `adv_*pause_cap` and `lp_*pause_cap` are compared on completion of auto-negotiation, the chosen flow control mechanism for the link depends on what is most meaningful.

`link_tx_pause` Link partner can assert flow control by sending pause frames when congestion is experienced.

`link_rx_pause` Link partner can respond to pause frames received.

The following truth table illustrates the meaningful flow control combinations related to local and link partner configurations. The abbreviations below are used in the table.

AA	`adv_cap_asym_pause`
AP	`adv_cap_pause`

LAC	lp_cap_asym_pause
LPC	lp_cap_pause
LA	link_asym_pause
LP	link_pause

AA	AP	LAC	LPC	LA	LP	Description
1	0	1	1	1	0	Local station will Tx a pause when Rx is congested.
0	1	0	1	0	1	Flow control in both Rx and Tx directions.
x	1	1	0	1	1	Local station honors received Pause frames by temporarily suspending Transmit.
x	x	x	x	0	0	All other combinations: Flow control not available on the link

When forced mode is enabled, the current setting of adv_cap_asym_pause and adv_cap_pause are used for the link. The link_asym_pause and link_pause become equal to the current adv_cap_asym_pause and adv_cap_pause settings. The above table also applies in forced mode, but the link partner configuration must be checked to verify that flow control is operating on the link.

另请参见

dladm(1M), driver.conf(4), bge(7D), dlpi(7P), eri(7D), gld(7D), hme(7D), qfe(7d)

附注

When adv_cap_autoneg is set to 0, the highest priority speed and duplex is used for forced mode.

The highest priority is the highest speed at full duplex. The lowest priority is the lowest speed at half duplex.

MII transceivers can exist internally to a system or can be connected to an external MII connector. Typically, an internal transceiver has an xcvr_addr of 1, while an external connection has an xcvr_addr of 0.

引用名

ipfilter – IP 包过滤软件

描述

IP 过滤器是 Solaris 系统上提供包过滤功能的软件。在设置正确的系统上，IP 过滤器可用于构建防火墙。

Solaris IP 过滤器随 Solaris 操作系统一起安装。但是，缺省情况下不启用包过滤。有关启用和激活 IP 过滤器功能的过程，请参见 ipf(1M)。

与位置配置文件交互

在位置配置文件中管理 IP 过滤器配置和激活（有关位置配置文件的更多信息，请参见 netcfg(1M)）。这些配置文件要么是固定的，表示以传统方式管理网络配置；要么是活性的，表示自动管理网络配置，并会根据配置文件中指定的策略规则对网络环境的变化作出反应。

当固定位置（当前可能只有一个 DefaultFixed 位置）处于活动状态时，对 SMF 系统信息库所做的更改会在对其禁用时应用于此位置，因此如果在稍后重新启用该位置会恢复这些更改。

当活性位置处于活动状态时，不应将更改直接应用于 SMF 系统信息库；这些更改不会保留在位置配置文件中，因此如果禁用该位置，或者刷新或重新启动系统的网络配置（由 `svc:/network/physical:default` 和 `svc:/network/location:default` 管理），更改将会丢失。应使用 `netcfg` 命令将更改应用于位置本身；这样会将更改保存到位置配置文件系统信息库中，还会将其应用于 SMF 系统信息库（如果更改是针对当前活动的位置进行的）。

如果在 `ipfilter-config-file` 属性中指定 IPv4 过滤器配置文件，则将启用 `ipfilter` SMF 服务。要启用其他类型的过滤，还可以指定 `ipfilter-v6-config-file`、`ipnat-config-file` 和 `ippool-config-file` 属性。

服务

`ipfilter` SMF 服务支持 `start`、`stop`、`restart` 和 `refresh` 方法。这些方法可以通过使用 svcadm(1M) 进行调用。

`start`　装入 `ipfilter` 内核模块并根据配置激活任何防火墙或 NAT 规则。

`stop`　清除应用的所有防火墙和 NAT 规则以及创建的任何活动会话信息。仅当能够进入主机的任何网络通信流量不存在风险时，才能停止启用联网功能的服务。

`restart`　停止然后启动 `ipfilter` 服务。对活动防火墙使用该方法会导致一定时段的暴露，在该时段内，通信流量可能在未经过滤的情况下进入和/或通过防火墙。

`refresh`　装入当前配置并从旧配置切换为新配置，而在这期间系统一直都在积极使用安全策略，没有间断。

基于主机的防火墙

为简化 IP 过滤器配置管理，创建了一个防火墙框架，以便用户可以通过在系统和服务级别指定防火墙策略来配置 IP 过滤器。在指定用户定义的防火墙策略后，该框架会生成一组 IP 过滤器规则来执行所需的系统行为。用户可以指定系统和服务防火墙策略，以允许或拒绝来自特定主机、子网和接口的网络通信流量。这些策略将转换成一组活动的 IPF 规则来执行指定的防火墙策略。

如果用户选择不使用该框架，仍可以指定他们自己的 ipf 规则文件。有关如何启用定制规则以及 ipf(4) 找出 ipf 规则语法的信息，请参见 ipf(1M)。

IPF 使用 ipmon(1M) 服务来记录防火墙事件。ipmon SMF 服务依赖于 ipfilter SMF 服务。在 ipfilter 由 “svcadm enable ipfilter” 自动启用后，ipmon 服务会立即由 ipfilter 服务启动方法临时启用。

型号

本节介绍基于主机的防火墙框架。有关如何配置防火墙策略的详细信息，请参见 svc.ipfd(1M)。

有一种具有不同优先级的三层方法可帮助用户实现所需行为。

全局缺省
全局缺省—缺省的系统范围防火墙策略。所有服务都将自动继承此策略，除非有服务修改了其防火墙策略。

网络服务
其优先级高于全局缺省。无论全局缺省策略如何，服务的策略都将允许/禁止其特定端口的通信。

全局覆盖
另一种系统范围的策略，优先于网络服务层中特定服务的需求。

```
Global Override
      |
      |
Network Services
      |
      |
Global Default
```

防火墙策略包括一种防火墙模式和一组可选的网络源。网络源包括 IP 地址、子网和本地网络接口，所有这些网络源都可供系统从中接收传入通信。基本防火墙模式组包括：

无
无防火墙，允许所有传入通信。

拒绝
允许所有传入通信，只拒绝来自指定源的通信。

允许
拒绝所有传入通信，只允许来自指定源的通信。

有关层的详细说明

第一个系统范围层全局缺省定义应用于**任何**传入通信的防火墙策略；例如，允许或阻止来自某一 IP 地址的所有通信。这使得很容易设置阻止所有传入通信或阻止来自不需要源的所有传入通信的策略。

网络服务层包含向远程客户机提供服务的本地程序（例如，telnetd、sshd 和 httpd）的防火墙策略。其中每个程序（即一个网络服务）都有其自己的防火墙策略

来控制对其服务的访问。最初，服务的策略设置为继承全局缺省策略，即“使用全局缺省”模式。这使得很容易在全局缺省层设置一个可由所有服务继承的策略。

如果某一服务的策略不同于全局缺省策略，则该服务的策略具有较高的优先级。如果全局缺省策略设置为阻止来自子网的所有通信，则 SSH 服务可配置为允许从该子网中的特定主机进行访问。所有网络服务的所有策略的集合构成了网络服务层。

第二个系统范围层全局覆盖的防火墙策略也应用于任何传入网络通信。此策略具有最高优先级，将覆盖其他层中的策略，具体来说是覆盖网络服务的需求。例如，无论服务的策略如何，都能理想地阻止已知的恶意源。

用户交互

此框架利用 IP 过滤器功能，仅在启用 `svc:/network/ipfilter` 时才有效，而在禁用 `network/ipfilter` 时将无效。同样，网络服务的防火墙策略也仅在启用该服务时才有效，而在禁用该服务时也将无效。具有有效防火墙的系统具有 IP 过滤器规则（用于正在运行/已启用的每个网络服务）以及系统范围的策略（防火墙模式不是**无**）。

用户可通过设置系统范围策略和各网络服务的策略来配置防火墙。有关配置防火墙策略的方法，请参见 svc.ipfd(1M)。

防火墙框架由策略配置和一个机制组成，用以从策略生成 IP 过滤器规则并应用这些规则来获取所需的 IP 过滤器配置。下面对设计和用户交互进行了快速汇总：

- 系统范围策略存储在 `network/ipfilter` 中
- 网络服务的策略存储在各 SMF 服务中
- 用户通过启用 `network/ipfilter`（请参见 `ipf(1M)`）来激活防火墙
- 用户通过启用/禁用网络服务来激活/取消激活该服务的防火墙
- 对系统范围或每服务防火墙策略进行更改会导致系统防火墙规则的更新

属性

有关以下属性的说明，请参见 `attributes(5)`：

属性类型	属性值
接口稳定性	Committed（已确定）

另请参见

`svcs(1)`、`ipf(1M)`、`ipmon(1M)`、`ipnat(1M)`、`netcfg(1M)`、`svcadm(1M)`、`svc.ipfd(1M)`、`ipf(`

《系统管理指南：IP 服务》

附注

`ipfilter` 服务由服务管理工具 `smf(5)` 管理，其服务标识符为：

```
svc:/network/ipfilter:default
```

可以使用 `svcadm(1M)` 来对此服务执行管理操作（如启用、禁用或请求重新启动）。可以使用 `svcs(1)` 命令来查询服务的状态。

IP 过滤器启动配置文件存储在 `/etc/ipf` 中。

引用名

isalist – Solaris软件可识别的本机指令集

描述

本文列出了由isalist(1)以及SI_ISALIST命令（属于sysinfo(2)）返回的可能的指令集名称。

该列表按某种方式对指令集系列进行了排序，即排在后面的名称通常比前面的名称的运行速度快；请注意，这与isalist(1)和sysinfo(2)列出的顺序相反。在下面的值列表中，带编号的条目通常表示性能渐增；带字母的条目表示互斥或无法排序。

此功能已过时，可能会在以后的Solaris版本中删除。以下列表不会影响现代处理器已创建的所有扩展。要了解处理指令集扩展的更好方法，请参见getisax(2)。

SPARC 平台

在适当时，指出了Oracle的C 4.0编译器带有给定值的-xarch选项的对应项。其他编译器可能也有类似的选项。

1a. sparc
指示SPARC V8指令集，其定义详见《SPARC Architecture Manual》，第8版，Prentice-Hall, Inc.出版，1992年。在特定系统上，某些指令（如整数相乘和相除、FSMULD及对四元操作数执行的所有浮点运算）可由内核模拟。

1b. sparcv7
与sparc相同。此选项与使用Oracle C 4.0编译器的–xarch=v7选项生成的代码相对应。

2. sparcv8–fsmuld
与sparc类似，但必须在硬件中执行整数相乘和相除。此选项与使用Oracle C 4.0编译器的–xarch=v8a选项生成的代码相对应。

3. sparcv8
与sparcv8–fsmuld类似，但也必须在硬件中执行FSMULD。此选项与使用Oracle C 4.0编译器的–xarch=v8选项生成的代码相对应。

4. sparcv8plus
指示SPARC V8指令集以及SPARC V9指令集（其定义详见《SPARC Architecture Manual》，第9版，Prentice-Hall出版，1994年）中可按照V8+技术规范使用的那些指令。此选项与使用Oracle C 4.0编译器的–xarch=v8plus选项生成的代码相对应。

5a. sparcv8plus+vis
与sparcv8plus类似，但新增了可按照V8+技术规范使用的那些UltraSPARC I可视化指令。此选项与使用Oracle C 4.0编译器的–xarch=v8plusa选项生成的代码相对应。

5b. sparcv8plus+fmuladd
与sparcv8plus类似，但新增了Fujitsu SPARC64浮点相乘-相加和相乘-相减指令。

6. sparcv9
指示SPARC V9指令集，其定义详见《SPARC Architecture Manual》，第9版，Prentice-Hall出版，1994年。

7a. sparcv9+vis
与sparcv9类似，但新增了UltraSPARC I可视化指令。

7b. sparcv9+vis2
与sparcv9类似，但新增了UltraSPARC III可视化指令。

7c. `sparcv9+fmuladd` 与 sparcv9 类似，但新增了 Fujitsu SPARC64 浮点相乘-相加和相乘-相减指令。

x86 平台

1. `i386` Intel 80386 指令集，其说明详见《i386 Microprocessor Programmer's Reference Manual》。

2. `i486` Intel 80486 指令集，其说明详见《i486 Microprocessor Programmer's Reference Manual》。（实际上，这包括 i386 以及 CMPXCHG、BSWAP 和 XADD 指令。）

3. `Pentium` Intel Pentium 指令集，其说明详见《iPentium Processor User's Manual》。（实际上，这包括 i486、CPU_ID 指令以及 CPU_ID 指令指明存在的所有功能。）

4. `pentium+mmx` 与 pentium 类似，包含保证存在的 MMX 指令。

5. `pentium_pro` Intel PentiumPro 指令集，其说明详见《PentiumPro Family Developer's Manual》。（实际上，这包括 pentium 以及保证存在的 CMOVcc、FCMOVcc、FCOMI 和 RDPMC 指令。）

6. `pentium_pro+mmx` 与 pentium_pro 类似，包含保证存在的 MMX 指令。

7. `amd64` AMD Opteron 指令集，其说明详见《AMD64 Architecture Programmer's Manual》。

另请参见

`isalist(1)`、`getisax(2)`、`sysinfo(2)`

引用名

kerberos – Solaris Kerberos 实现概述

描述

Solaris Kerberos 实现（后文有时简称为“Kerberos”）会验证网络环境中的客户机，从而实现安全事务。（客户机可以是用户或网络服务。）Kerberos 会验证客户机的身份以及所传输数据的真实性。Kerberos 是*single-sign-on*系统，意味着用户仅需要在会话开始时提供口令。Solaris Kerberos 实现是基于 MIT 开发的 Kerberos(TM) 系统，并与异构网络上的 Kerberos V5 系统兼容。

Kerberos 通过授予客户机**票证**发挥作用，票证可唯一地标识客户机并具有有限生命周期。将针对票证所属网络服务自动验证拥有票证的客户机；例如，具有有效 Kerberos 票证的用户可登录运行 Kerberos 的其他计算机，而无需标识自身。由于每个客户机均具有唯一的票证，因此可保证其身份。

要获得票证，客户机必须首先使用 `kinit(1)` 命令或 PAM 模块初始化 Kerberos 会话。（请参见 `pam_krb5(5)`）。`kinit` 提示输入口令，然后与密钥分发中心 (*Key Distribution Center*, KDC) 通信。KDC 返回票证授予票证 (*Ticket-Granting Ticket*, TGT) 并提示输入确认口令。如果客户机确认口令，则它可以使用票证授予票证来获得特定网络服务的票证。因为以透明方式授予票证，所以用户无需担心票证管理。可使用 `klist(1)` 命令查看当前票证。

票证的有效性取决于安装时设置的系统**策略**。例如，票证具有缺省的有效生命周期。策略可以进一步指示特权票证（如属于 root 的票证）具有很短的生命周期。策略还可以允许覆盖某些缺省值；例如，客户机可以请求生命周期大于或小于缺省值的票证。

可使用 `kinit` 更新票证。票证也是**可转发的**，允许您将一台计算机上授予的票证用在其他主机上。可使用 `kdestroy(1)` 销毁票证。在 `.logout` 文件中包含 `kdestroy` 调用是个不错的选择。

在 Kerberos 中，客户机称为**主体**。主体的格式如下：

```
primary/instance@REALM
```

primary　用户、主机或服务。

instance　主体的限定。如果主体为主机（由关键字 `host` 表示），则实例是该主机的完全限定域名。如果主体是用户或服务，则实例是可选项。某些实例具有特权，如 `admin` 或 `root`。

realm　域的 Kerberos 等效项；事实上，在大部分情况下领域直接映射到 DNS 域名。提供的 Kerberos 领域仅可为大写。有关主体名称的示例，请参见“示例”。

通过利用一般安全服务 API (General Security Services API, GSS-API)，Kerberos 还提供用户验证以外的其他两种类型的安全服务：**完整性**（验证所传输数据的有效性）以及**保密性**（加密所传输的数据）。开发人员可通过使用 RPCSEC_GSS API 接口来利用 GSS-API（请参见 `rpcsec_gss(3NSL)`）。

示例

示例 1　有效主体名称的示例

以下是有效的主体名称示例：

```
joe
joe/admin
joe@ENG.ACME.COM
joe/admin@ENG.ACME.COM
rlogin/bigmachine.eng.acme.com@ENG.ACME.COM
host/bigmachine.eng.acme.com@ENG.ACME.COM
```

前四个例子是**用户主体**。在前两个例子中，假设用户 joe 与客户机处于相同领域，所以未指定领域。请注意，即使同一用户在使用 joe 和 joe/admin，它们仍是不同的主体；joe/admin 拥有与 joe 不同的特权。第五个例子是**服务主体**，而最后一个例子是**主机主体**。对于主机主体，需要 host 这一单词。有了主机主体，实例即是完全限定主机名。请注意，单词 admin 和 host 是保留关键字。

另请参见

kdestroy(1)、kinit(1)、klist(1)、kpasswd(1)、krb5.conf(4)、krb5envvar(5)

《Oracle Solaris 11.1 管理：安全服务》

附注

在以前版本的 Solaris 操作系统中，Solaris Kerberos 实现称为“Sun Enterprise 验证机制”(Sun Enterprise Authentication Mechanism, SEAM)。

如果输入您的用户名，kinit 会以下面的消息响应：

```
Principal unknown (kerberos)
```

您尚未注册为 Kerberos 用户。请与您的系统管理员联系或请参见《Oracle Solaris 11.1 管理：安全服务》。

引用名

krb5_auth_rules – Kerberos V5 授权概述

描述

当 ftp、rcp、rlogin、rsh、ssh、telnet 或 ssh 客户机的 Kerberos 版本用于连接到服务器时，发起方用户的身份必须针对 Kerberos V5 验证系统进行验证。如果适当的条目存在于 ~/.k5login 文件、gsscred 表中，或如果缺省 GSS/Kerberos 验证规则成功将 Kerberos 主体名称映射到 Unix 登录名，则可授权帐户访问权。

要避免安全问题，~/.k5login 文件必须由客户机尝试访问的服务器上的远程用户所有。文件应该包含专用授权列表，其中包含格式为 *principal/instance@realm* 的 Kerberos 主体名称。*/instance* 变量在 Kerberos 主体名称中是可选项。例如，jdb@ENG.ACME.COM 和 jdb/happy.eng.acme.com@ENG.ACME.COM 之类的不同主体名称均是合法的 Kerberos 主体（虽然并不等效）。如果 ~/.k5login 文件位于远程用户帐户的登录目录中且可以针对文件中指定的某个主体验证发起方用户，则可以授予客户机访问权限。有关 Kerberos 主体名称的更多信息，请参见 gkadmin(1M) 和 kadm5.acl(4)。

当远程用户的登录帐户中找不到 ~/.k5login 文件时，会针对 gsscred 表检查与发起方用户相关联的 Kerberos V5 主体名称。如果 gsscred 表存在且主体名称在表中有匹配项目，并且表中列出的 Unix 用户 ID 对应于客户机尝试访问的用户帐户，则会授予访问权。如果 Unix 用户 ID 不匹配，则会拒绝访问权。请参见 gsscred(1M)。

例如，如果 23154 也是用户帐户数据库中列出的 jdb-user 的 uid，则会授予 gsscred 表中列出的具有主体名称 jdb@ENG.ACME.COM 和 uid 23154 的发起方用户对 jdb-user 帐户的访问权。请参见 passwd(4)。

最后，如果不存在 ~/.k5login 文件且发起方用户的 Kerberos V5 身份不在 gsscred 表中，或 gsscred 表不存在，则在以下情况下会授予客户机对帐户的访问权（缺省 GSS/Kerberos 验证规则）：

- 经过验证的主体名称的用户部分与客户机指定的 Unix 帐户名相同。
- 客户机和服务器的领域部分相同，除非 krb5.conf(4) *auth_to_local_realm* 参数用于创建等效项。
- Unix 帐户名存在于服务器上。

例如，如果发起方用户的主体名称为 jdb@ENG.ACME.COM 且服务器位于 SALES.ACME.COM 领域中，则即使 jdb 是服务器上有效的帐户名也会拒绝客户机的访问权。这是因为领域 SALES.ACME.COM 和 ENG.ACME.COM 不同。

krb5.conf(4) *auth_to_local_realm* 参数也会影响授权。非缺省领域可等效于经过验证的 name-to-local name 映射的缺省领域。

文件

~/.k5login	每用户帐户的授权文件。
/etc/passwd	系统帐户文件。此信息也可在目录服务中。请参见 passwd(4)。

属性

有关以下属性的说明，请参见 attributes(5)：

属性类型	属性值
接口稳定性	Committed（已确定）

另请参见

ftp(1)、rcp(1)、rsh(1)、telnet(1)、gkadmin(1M)、gsscred(1M)、kadm5.acl(4)、krb5.conf(4

引用名

krb5envvar – Kerberos 环境变量

描述

Kerberos 机制提供了许多环境变量，可配置不同行为来满足应用程序的需求。Kerberos 机制中使用的环境变量包括：

KRB5_KTNAME

由机制用于指定密钥表文件的位置。该变量可设置为以下值：

`[[`*<kt type>*`:]`*<file name>*`]`

其中 *<kt type>* 可以为 `FILE` 或 `WRFILE`。`FILE` 表示读取操作；`WRFILE` 表示写入操作。*<file name>* 是 `keytab` 文件的位置。

r

如果未定义 `KRB5_KTNAME`，缺省值为：

```
FILE:/etc/krb5/krb5.keytab
```

`keytab` 文件用于永久存储凭证并且通常用于服务守护进程。

指定 `FILE` 类型即假设对关联文件的后续操作可由调用进程读取。务必小心确保文件仅可供需要检索未加密密钥的主体集读取。

`WRFILE` 类型由 `kadmin`(1M) 命令使用。指定此类型可允许管理员指定要写入的替代 `keytab` 文件，而无需对文件位置使用其他命令行参数。

KRB5CCNAME

由机制用于指定凭证高速缓存的位置。该变量可设置为以下值：

`[[`*<cc type>*`:]`*<file name>*`]`

其中，*<cc type>* 可以是 `FILE` 或 `MEMORY`。*<file name>* 是主体的凭证高速缓存的位置。

如果未定义 `KRB5CCNAME`，缺省值为：

`FILE:/tmp/krb5cc_`*<uid>*

其中，*<uid>* 是创建高速缓存文件的进程的用户 ID。

凭证高速缓存文件用于存储已授予主体的票证。

指定 `FILE` 类型即假设对关联文件的后续操作可由调用进程读取和写入。务必小心确保文件仅可供需要访问凭证的主体集访问。如果凭证文件位于其他用户拥有写入访问权的目录中，则需要设置该目录的 Sticky 位（请参见 `chmod`(1)）。

`MEMORY` 凭证高速缓存类型仅在特殊情况下使用，如为调用进程的生命周期创建临时高速缓存时。

KRB5RCNAME

由机制用于指定重放高速缓存的类型和位置。该变量可设置为以下值：

`[[`*<rc type>*`:]`*<file name>*`]`

其中，*<rc type>* 可以是 `FILE`、`MEMORY` 或 `NONE`。*<file name>* 仅在指定重放高速缓存文件类型时相关。

如果未定义，缺省值为：

`FILE:/var/krb5/rcache/root/rc_`*<service>*

...如果进程的所有者为 root，或：

`FILE:/var/krb5/rcache/rc_`*<service>*

...如果进程的所有者为 root 以外的用户。*<service>* 是与重放高速缓存文件相关联的服务进程名称。

重放高速缓存由 Kerberos 用于检测验证数据的重放。这可防止在网络上捕捉验证消息的人员通过重新发送这些消息来针对服务器进行验证。

当指定 `FILE` 重放高速缓存类型时，必须小心防止其他用户删除重放高速缓存文件。请确保重放高速缓存路径中的每个目录仅可由重放高速缓存的所有者写入，或者其他人具有写入权限的重放高速缓存路径中每个目录上均已设置 Sticky 位 (“t”)。

当指定 `MEMORY` 重放高速缓存类型时，您需要在性能与使用非持久性高速缓存产生的轻度安全风险之间进行权衡。系统重新引导期间发生以下状况即会产生风险：

- 从重新引导前上次写入重放高速缓存到 Kerberos 服务器应用程序运行时的持续时间少于 Kerberos 时钟相位差（请参见 `krb5.conf(4)`）。

当指定 `NONE` 重放高速缓存时间时，您需要了解这将会禁用重放高速缓存及其所带来的全部安全性风险。这包括手册页的此部分中所述的全部风险。

在此状况下，服务器应用程序可接受重放 Kerberos 验证数据（直到上次写入时与时钟相位差之间的差异为止）。通常，这是一小段时间而已。如果服务器应用程序开始接受连接所花的时间比时钟相位差长，则没有重放风险。

如果重放高速缓存位于 swap 文件系统（如 `/tmp` 和 `/var/run`）中时使用 `FILE` 重放高速缓存类型，仍存在上述风险。

通过 `FILE` 类型的 `MEMORY` 重放高速缓存类型性能改进得益于无需磁盘 I/O。即使 `FILE` 重放高速缓存位于内存备份文件系统上仍是如此，如 swap（`/tmp` 和 `/var/run`）。

请注意，`MEMORY` 类型高速缓存是每进程高速缓存，因此必须谨慎使用这些类型的高速缓存。`MEMORY` 类型高速缓存可能产生问题的示例之一即是应用程序使用多个进程来建立安全上下文。在此情况下，内存重放高速缓存不会在进程中共享，从而使重放攻击成为可能。

KRB5_CONFIG

允许您更改 `/etc/krb5/krb5.conf` 文件的缺省位置，以使 Kerberos 库代码从 KRB5_CONFIG 指定的其他文件中读取配置参数。例如（使用 `ksh(1)` 的 kinit）：

```
KRB5_CONFIG=/var/tmp/krb5.conf kinit
```

属性

有关以下属性的说明，请参见 attributes(5)：

属性类型	属性值
可用性	system/security/kerberos-5
接口稳定性	Uncommitted（未确定）

另请参见

chmod(1)、kinit(1)、klist(1)、ksh(1)、kadmin(1M)、kadmind(1M)、krb5.conf(4)、attributes(5)、

引用名

kssl, KSSL – 内核 SSL 代理

描述

KSSL 是用于 SSL/TLS 协议的透明服务器端代理。它可在内核中处理 SSL 通信流量，通过避免切换上下文并直接访问 Oracle Solaris Crypto Framework 的内核提供者从而提高性能。通过 KSSL，即使针对仅可以通过 TCP 以明文形式通信的应用程序，也可能提供 SSL 保护。

KSSL 在内核中配置并传递/接受应用程序的明文数据。客户机会将它们视为单一 SSL 服务器。

对其配置 KSSL 的服务器端应用程序并不知晓它正在接收先前由 SSL 保护的数据。KSSL 在一个端口上接收 SSL 通信流量（例如，端口 443），执行处理并将明文数据传递到在其他端口（例如，8080）上侦听的应用程序。类似地，对于出站方向，应用程序发送明文数据，KSSL 生成 SSL 记录并将其发送到客户机。因此，无需设置应用程序使用 SSL。

系统上可配置多个 KSSL 实例，每个实例均具有不同的一组属性，如端口、证书、密钥或加密套件。请参见 ksslcfg(1M)。内核中的每个 KSSL 实例都作为 SMF 服务受到跟踪。请参见 smf(5)。

KSSL 为使用 TCP 通过 IPv4 和 IPv6 传递的记录提供 SSL 处理。

KSSL 支持以下协议：SSLv3、TLSv1.0

另请参见

ksslcfg(1M)、smf(5)

由 T. Dierks 和 C. Allen 合著的《The TLS Protocol Version 1.0》，RFC 2246。The Internet Society 出版，1999 年。

引用名

labels – Solaris Trusted Extensions label attributes

描述

Labels are attributes that are used in mandatory policy decisions. Labels are associated, either explicitly or implicitly, with all subjects (generally processes) and objects (generally things with data such as files) that are accessible to subjects. The default Trusted Extensions mandatory policy labels are defined by a site's security administrator in label_encodings(4).

Mandatory Policy

Various mandatory policies might be delivered in the lifetime of Solaris Trusted Extensions.

The default mandatory policy of Trusted Extensions is a Mandatory Access Control (MAC) policy that is equivalent to that of the Bell-LaPadula Model of the Lattice, the Simple Security Property, and the *-Property (Star Property), with restricted write up. The default mandatory policy is also equivalent to the Goguen and Mesegeur model of Non-Inteference.

For this MAC policy, two labels are always defined: admin_low and admin_high. The site's security administrator defines all other labels in label_encodings(4). admin_low is associated with all normal user readable (viewable) Trusted Extensions objects. admin_high is associated with all other Trusted Extensions objects. Only administrative users have MAC read (view) access to admin_high objects and only administrative users have MAC write (modify) access to admin_low objects or admin_high objects.

Human Readable Labels

Users interact with labels as strings. Graphical user interfaces and command line interfaces present the strings as defined in label_encodings(4). Human readable labels are classified at the label that they represent. Thus the string for a label A is only readable (viewable, translatable to or from human readable to opaque m_label_t) by a subject whose label allows read (view) access to that label.

Internal Text Labels

In order to store labels in publicly accessible (admin_low) name service databases, an unclassified internal text form is used. This textual form is not intended to be used in any interfaces other than those that are provided with the Trusted Extensions software release that created this textual form of the label.

Labels and Applications

Applications interact with labels as opaque (m_label_t) structures. The semantics of these opaque structures are defined by a string to m_label_t translation. This translation is defined in label_encodings(4). Various Application Programming Interfaces (API) translate between strings and m_label_t structures. Various APIs test access of subject-related labels to object-related labels.

属性

See attributes(5) for description of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	See below.

The labels implementation is Committed for systems that implement the Defense Intelligence Agency (DIA) MAC policy of label_encodings(4). Other policies might exist in a future release of Trusted Extensions that might make obsolete or supplement label_encodings.

Internal text labels are Not-an-Interface and might change with any release of Trusted Extensions. They are intended only for input and generation on the same release of Trusted Extensions software.

As a potential porting aid for Trusted Solaris 8 applications, the opaque structure names `bslabel_t`, `blevel_t`, and `bclear_t` are defined to be equivalent to `m_label_t`. Like `m_label_t`, these types must be ported as opaque pointers. The same must be done with the various Trusted Solaris 8 label interfaces. These Trusted Solaris 8 structures and interfaces are Obsolete and might be removed from a future release of Trusted Extensions.

另请参见

`chk_encodings`(1M), `blcompare`(3TSOL), `label_to_str`(3TSOL), `m_label_alloc`(3TSOL), `m_label_dup`(3TSOL), `m_label_free`(3TSOL), `str_to_label`(3TSOL), `label_encodings`(4), `attributes`(5)

Bell, D. E., and LaPadula, L. J. Secure Computer Systems: Unified Exposition and Multics Interpretation, MTR-2997 Rev. 2, MITRE Corp., Bedford Mass., March 1976. NTIS AD-A023 588/7.

Goguen, J. A., and Mesegeur, J.: Security Policies and Security Models, Proceedings 1982 Symposium on Security and Privacy, IEEE Computer Society Press, 1982, p 11-20.

Goguen, J. A., and Mesegeur, J.: Unwinding and Interference Control, Proceedings 1984 Symposium on Security and Privacy, IEEE Computer Society Press, 1984, p 75-86.

《Compartmented Mode Workstation Labeling: Encodings Format》

附注

The functionality described on this manual page is available only if the system is configured with Trusted Extensions.

引用名

largefile – large file status of utilities

描述

A *large file* is a regular file whose size is greater than or equal to 2 Gbyte (2^{31} bytes). A *small file* is a regular file whose size is less than 2 Gbyte.

Large file aware utilities

A utility is called *large file aware* if it can process large files in the same manner as it does small files. A utility that is large file aware is able to handle large files as input and generate as output large files that are being processed. The exception is where additional files are used as system configuration files or support files that can augment the processing. For example, the `file` utility supports the `-m` option for an alternative "magic" file and the `-f` option for a support file that can contain a list of file names. It is unspecified whether a utility that is large file aware will accept configuration or support files that are large files. If a large file aware utility does not accept configuration or support files that are large files, it will cause no data loss or corruption upon encountering such files and will return an appropriate error.

The following `/usr/bin` utilities are large file aware:

adb	aliasadm	awk	bdiff	cat
chgrp	chmod	chown	cksum	cmp
compress	cp	csh	csplit	cut
dd	dircmp	du	egrep	fgrep
file	find	ftp	getconf	grep
gzip	head	join	jsh	ksh88
ksh	ln	ls	mailcompat	mailstats
mdb	mkdir	mkfifo	more	mv
nawk	page	paste	pathchck	pg
praliases	rcp	remsh	rksh88	rksh
rm	rmdir	rsh	sed	sh
sort	split	sum	tail	tar
tee	test	touch	tr	uncompress
uudcode	uuencode	vacation	wc	zcat

The following `/usr/xpg4/bin` utilities are large file aware:

awk	cp	chgrp	chown	du
egrep	fgrep	file	grep	ln

```
ls          more        mv          rm          sed
sh          sort        tail        tr
```

The following `/usr/xpg6/bin` utilities are large file aware:

```
getconf     ls          tr
```

The following `/usr/sbin` utilities are large file aware:

```
editmap     install     makemap     mkfile      mknod
mvdir       swap
```

The following `/usr/lib` utilities are large file aware:

```
mail.local  sendmail    smrsh
```

See the USAGE section of the `swap`(1M) manual page for limitations of `swap` on block devices greater than 2 Gbyte on a 32–bit operating system.

The following `/usr/ucb` utilities are large file aware:

```
chown       from        ln          ls          sed
sum         touch
```

The `/usr/bin/cpio` and `/usr/bin/pax` utilities are large file aware, but cannot archive a file whose size exceeds 8 Gbyte – 1 byte.

The `/usr/bin/truss` utilities has been modified to read a dump file and display information relevant to large files, such as offsets.

nfs file systems

The following utilities are large file aware for `nfs` file systems:

```
/usr/lib/autofs/automountd          /usr/sbin/mount
/usr/lib/nfs/rquotad
```

ufs file systems

The following `/usr/bin` utility is large file aware for `ufs` file systems:

`df`

The following `/usr/lib/nfs` utility is large file aware for `ufs` file systems:

`rquotad`

The following `/usr/xpg4/bin` utility is large file aware for `ufs` file systems:

`df`

The following `/usr/sbin` utilities are large file aware for `ufs` file systems:

`clri`	`dcopy`	`edquota`	`ff`	`fsck`
`fsdb`	`fsirand`	`fstyp`	`labelit`	`lockfs`
`mkfs`	`mount`	`ncheck`	`newfs`	`quot`
`quota`	`quotacheck`	`quotaoff`	`quotaon`	`repquota`
`tunefs`	`ufsdump`	`ufsrestore`	`umount`	

Large file safe utilities

A utility is called *large file safe* if it causes no data loss or corruption when it encounters a large file. A utility that is large file safe is unable to process properly a large file, but returns an appropriate error.

The following `/usr/bin` utilities are large file safe:

`audioconvert`	`audioplay`	`audiorecord`	`comm`	`diff`
`diff3`	`diffmk`	`ed`	`lp`	`mail`
`mailcompat`	`mailstats`	`mailx`	`pack`	`pcat`
`red`	`rmail`	`sdiff`	`unpack`	`vi`
`view`				

The following `/usr/xpg4/bin` utilities are large file safe:

`ed`	`vi`	`view`

The following `/usr/xpg6/bin` utility is large file safe:

```
ed
```

The following `/usr/sbin` utilities are large file safe:

lpfilter	lpforms

The following `/usr/ucb` utilities are large file safe:

`Mail`	`lpr`

另请参见

`lf64`(5), `lfcompile`(5), `lfcompile64`(5)

引用名

ldap – LDAP 用作命名系统信息库

描述

LDAP 是指 Lightweight Directory Access Protocol（轻量目录访问协议），是访问目录服务器的一项行业标准。通过使用 ldapclient(1M) 初始化客户机并在名称服务切换文件 /etc/nsswitch.conf 中使用关键字 ldap，Oracle Solaris 客户机可以从 LDAP 服务器获得命名信息。用户名、主机名和口令之类的信息存储在 LDAP 服务器上的目录信息树或 DIT 中。DIT 包含由属性构成的条目。每个属性都具有一个类型以及一个或多个值。

Oracle Solaris LDAP 客户机使用 LDAP v3 协议来访问 LDAP 服务器的命名信息。LDAP 服务器必须支持 RFC2307bis（草稿）中定义的对象类和属性，用于将命名服务模型映射到 LDAP。作为使用 RFC2307bis（草稿）中定义的模式的一种替代方法，可配置系统使用其他模式集并且配置模式映射功能以在二者之间映射。有关更多详细信息，请参见《Oracle Solaris Administration: Naming and Directory Services》。

ldapclient(1M) 实用程序可通过设置相应目录、文件以及配置信息将 Oracle Solaris 计算机变为 LDAP 客户机。LDAP 客户机在本地高速缓存文件中高速缓存这些配置信息。可通过 ldap_cachemgr(1M) 守护进程访问这些配置信息。此守护进程还会从 LDAP 服务器刷新配置文件中的信息，从而提高性能和安全性。ldap_cachemgr 必须一直运行，才能使命名服务正常运行。

存在两种类型的配置信息：通过配置文件获得的信息以及根据客户机配置的信息。配置文件包含有关客户机如何访问目录的所有信息。代理用户的凭证信息将根据客户机进行配置并且不会通过配置文件下载。

配置文件包含服务器特定的参数，并且所有客户机都需要这些参数以找出所需 LDAP 域的服务器。例如，这些信息可以是服务器的 IP 地址以及搜索基标识名 (Distinguished Name, DN)。配置文件会在客户机初始化期间根据缺省配置文件在客户机上进行配置，并且在过期时由 ldap_cachemgr 守护进程定期更新。

客户机配置文件可存储在 LDAP 服务器上并可由 ldapclient 实用程序用于初始化 LDAP 客户机。使用客户机配置文件是配置客户机的最简单方法。请参见 ldapclient(1M)。

凭证信息包括由客户机使用的客户机特定参数。这些信息可以是客户机的绑定标识名（LDAP“登录”名称）以及口令。如果这些参数是必需的，将通过 ldapclient(1M) 在初始化期间进行手动定义。

命名信息存储在 LDAP 服务器上的容器中。容器是包含命名服务信息的 DIT 中的非叶条目。容器与 NIS 中的映射类似。NIS 数据库与 LDAP 中的容器之间的缺省映射如下所示。可使用 serviceSearchDescriptors 覆盖这些容器的位置及名称。有关更多信息，请参见 ldapclient(1M)。

数据库	对象类	容器
passwd	posixAccount	ou=people,dc=...

数据库	对象类	容器
	shadowAccount	
group	posixGroup	ou=Group,dc=...
services	ipService	ou=Services,dc=...
protocols	ipProtocol	ou=Protocols,dc=...
rpc	oncRpc	ou=Rpc,dc=...
hosts	ipHost	ou=Hosts,dc=...
ipnodes	ipHost	ou=Hosts,dc=...
ethers	ieee802Device	ou=Ethers,dc=...
bootparams	bootableDevice	ou=Ethers,dc=...
networks	ipNetwork	ou=Networks,dc=...
netmasks	ipNetwork	ou=Networks,dc=...
netgroup	nisNetgroup	ou=Netgroup,dc=...
aliases	mailGroup	ou=Aliases,dc=...
publickey	nisKeyObject	
generic	nisObject	nisMapName=...,dc=...
printers	printerService	ou=Printers,dc=...
auth_attr	SolarisAuthAttr	ou=SolarisAuthAttr,dc=...
prof_attr	SolarisProfAttr	ou=SolarisProfAttr,dc=...
exec_attr	SolarisExecAttr	ou=SolarisProfAttr,dc=...
user_attr	SolarisUserAttr	ou=people,dc=...

客户机的安全模型由要使用的凭证级别、验证方法以及要使用的 PAM 模块共同定义。凭证级别定义客户机应使用何种凭证来验证目录服务器，而验证方法则定义所选择的方法。可使用多个值设置这两项。Oracle Solaris LDAP 支持将以下值用于凭证级别：

```
anonymous
proxy
self
```

Oracle Solaris LDAP 支持将以下值用于验证方法：

```
none
simple
sasl/CRAM-MD5
sasl/DIGEST-MD5
sasl/GSSAPI
tls:simple
tls:sasl/CRAM-MD5
tls:sasl/DIGEST-MD5
```

当凭证级别配置为 self 时，必须配置 DNS 且验证方法必须为 sasl/GSSAPI。/etc/nsswitch.conf 中的 hosts 和 ipnodes 必须配置为使用 DNS，例如 hosts: dns files 和 ipnodes: dns files。

sasl/GSSAPI 自动使用 GSSAPI 保密性和完整性选项（如果目录服务器上已配置这些选项）。

self 凭证级别允许以下类型的查找：根据用户的命名服务查找，或者连接到目录服务器时使用用户的 GSSAPI 凭证的查找。当前此模型中支持的唯一 GSSAPI 机制是 Kerberos V5。在使用此凭证级别之前必须配置 Kerberos。有关详细信息，请参见 kerberos(5)。

访问控制可提供更多保护，允许服务器授予某些容器或条目的访问权。访问控制由 LDAP 服务器中定义和存储的访问控制列表 (Access Control List, ACL) 指定。LDAP 服务器上的访问控制列表在 SunOne Directory Server 上称为访问控制指令 (Access Control Instruction, ACI)。每一个 ACL 或 ACI 均指定一或多个目录对象，例如特定容器中的 cn 属性，要授予或拒绝其访问权的一或多个客户机，以及决定客户机如何处理对象的一或多项访问权。客户机可以是用户或应用程序。例如，访问权可以指定为读写。有关使用 LDAP 作为命名系统信息库时 ACL 和 ACI 的限制，请参见《Oracle Solaris Administration: Naming and Directory Services》。

/etc 目录中提供了名为 nsswitch.ldap 的一个示例 nsswitch.conf(4) 文件。ldapclient(1M) 实用程序会将此文件复制到 /etc/nsswitch.conf。此文件将 LDAP 用作 nsswitch.conf 文件中不同数据库的系统信息库。

下面列出了与 LDAP 有关的用户命令：

idsconfig(1M)	使 SunOne Directory Server 可以支持 Solaris LDAP 客户机。
ldapaddent(1M)	从相应的 /etc 文件创建 LDAP 条目。
ldapclient(1M)	初始化 LDAP 客户机，或生成要存储在目录中的配置文件。
ldaplist(1)	列出 LDAP 命名空间的内容。

文件

/var/ldap/ldap_client_cred
/var/ldap/ldap_client_file　包含客户端的 LDAP 配置的文件。不要手动修改这些文件。其内容不保证是用户可读的。使用 ldapclient(1M) 可以更新这些文件。

`/etc/nsswitch.conf` 用于名称服务转换的配置文件。

`/etc/nsswitch.ldap` 使用 LDAP 和文件配置的名称服务转换的样例配置文件。

`/etc/pam.conf` PAM 框架配置文件。

`/etc/pam.d/`*service* 替代 PAM 框架配置文件。

另请参见 `ldaplist`(1)、`idsconfig`(1M)、`ldap_cachemgr`(1M)、`ldapaddent`(1M)、`ldapclient`(1M)、`nssw`

《Oracle Solaris Administration: Naming and Directory Services》

引用名

lf64 – transitional interfaces for 64-bit file offsets

描述

The data types, interfaces, and macros described on this page provide explicit access to 64-bit file offsets. They are accessible through the transitional compilation environment described on the `lfcompile64(5)` manual page. The function prototype and semantics of a transitional interface are equivalent to those of the standard version of the call, except that relevant data types are 64-bit entities.

Data Types

The following tables list the standard data or struct types in the left-hand column and their corresponding explicit 64-bit file offset types in the right-hand column, grouped by header. The absence of an entry in the left-hand column indicates that there is no existing explicit 32-bit type that corresponds to the 64–bit type listed in the right—hand column. Note that in a 64-bit application, the standard definition is equivalent to the 64-bit file offset definition.

<aio.h>

struct aiocb	struct aiocb64
off_t aio_offset;	off64_t aio_offset;

<sys/dirent.h>

struct dirent	struct dirent64
ino_t d_ino;	ino64_t d_ino;
off_t d_off;	off64_t d_off;

<sys/fcntl.h>

struct flock	struct flock64
off_t l_start;	off64_t l_start;
off_t l_len;	off64_t l_len;
F_SETLK	F_SETLK64
F_SETLKW	F_SETLKW64
F_GETLK	F_GETLK64
F_FREESP	F_FREESP64
F_ALLOCSP	F_ALLOCSP64
	O_LARGEFILE

<sys/stdio.h>

fpos_t	fpos64_t

<sys/resource.h>

rlim_t	rlim64_t
struct rlimit	struct rlimit64
rlim_t rlim_cur;	rlim64_t rlim_cur;
rlim_t rlim_max;	rlim64_t rlim_max;
RLIM_INFINITY	RLIM64_INFINITY
RLIM_SAVED_MAX	RLIM64_SAVED_MAX
RLIM_SAVED_CUR	RLIM64_SAVED_CUR

<sys/stat.h>

struct stat	struct stat64
ino_t st_ino;	ino64_t st_ino;
off_t st_size;	off64_t st_size;
blkcnt_t st_blocks;	blkcnt64_t st_blocks;

<sys/statvfs.h>

struct statvfs	struct statvfs64
fsblkcnt_t f_blocks;	fsblkcnt64_t f_blocks;
fsblkcnt_t f_bfree;	fsblkcnt64_t f_bfree;
fsblkcnt_t f_bavial;	fsblkcnt64_t f_bavial;
fsfilcnt_t f_files;	fsfilcnt64_t f_files;
fsfilcnt_t f_ffree;	fsfilcnt64_t f_ffree;
fsfilcnt_t f_favail;	fsfilcnt64_t f_favail;

<sys/types.h>

off_t;	off64_t;
ino_t;	ino64_t;
blkcnt_t;	blkcnt64_t;
fsblkcnt_t;	fsblkcnt64_t;
fsfilcnt_t;	fsfilcnt64_t;

<unistd.h>

	_LFS64_LARGEFILE
	_LFS64_STDIO

<sys/unistd.h>

	_CS_LFS64_CFLAGS
	_CS_LFS64_LDFLAGS
	_CS_LFS64_LIBS
	_CS_LFS64_LINTFLAGS

System Interfaces

The following tables display the standard API and the corresponding transitional interfaces for 64-bit file offsets. The interfaces are grouped by header. The interface name and the affected data types are displayed in courier font.

<aio.h>

int aio_cancel(...,	int aio_cancel64(...,
struct aiocb *);	struct aiocb64 *);
int aio_error(	int aio_error64(
const struct aiocb *);	const struct aiocb64 *);
int aio_fsync(...,	int aio_fsync64(...,
struct aiocb *);	struct aiocb64 *);
int aio_read(struct aiocb *);	int aio_read64(struct aiocb64 *);
int aio_return(struct aiocb *);	int aio_return64(struct aiocb64 *);
int aio_suspend(	int aio_suspend64(

const struct aiocb *, ...);	const struct aiocb64 *, ...);
int aio_waitn(aiocb_t *[],	int aio_waitn64(aiocb64_t *[],
...);	...);
int aio_write(struct aiocb *);	int aio_write64(struct aiocb64 *);
int lio_listio(...,	int lio_listio64(...,
const struct aiocb *, ...);	const struct aiocb64 *, ...);

<dirent.h>

int alphasort(	int alphasort64(
const struct dirent **,	const struct dirent64 **,
const struct dirent **)	const struct dirent64 **)
struct dirent *readdir();	struct dirent64 *readdir64();
struct dirent *readdir_r();	struct dirent64 *readdir64_r();
int scandir(...,	int scandir64(...,
struct dirent *(*[]),	struct dirent64 *(*[]),
int (*)(const struct dirent *),	int (*)(const struct dirent64 *),
int (*)(const struct dirent **,	int (*)(const struct dirent64 **,
const struct dirent **))	const struct dirent64 **))

<fcntl.h>

int attropen();	int attropen64();
int creat();	int creat64();
int open();	int open64();
int openat();	int openat64();
int posix_fadvise()	int posix_fadvise64()
int posix_fallocate()	int posix_fallocate64()

<ftw.h>

int ftw(...,	int ftw64(...,
const struct stat *, ...);	const struct stat64 *, ...);
int nftw(..	int nftw64(...,
const struct stat *, ...);	const struct stat64 *, ...);

<libgen.h>

char *copylist(..., off_t);	char *copylist64(..., off64_t);

<stdio.h>

int fgetpos();	int fgetpos64();
FILE *fopen();	FILE *fopen64();
FILE *freopen();	FILE *freopen64();
int fseeko(..., off_t, ...);	int fseeko64(..., off64_t, ...);
int fsetpos(...,	int fsetpos64(...,
const fpos_t *);	const fpos64_t *);
off_t ftello();	off64_t ftello64()();
FILE *tmpfile();	FILE *tmpfile64();

<stdlib.h>

int mkstemp();	int mkstemp64();

<sys/async.h>

int aioread(..., off_t, ...);	int aioread64(..., off64_t, ...);
int aiowrite(..., off_t, ...);	int aiowrite64(..., off64_t, ...);

<sys/dirent.h>

int getdents(..., dirent);	int getdents64(..., dirent64);

<sys/mman.h>

void mmap(..., off_t);	void mmap64(..., off64_t);

<sys/resource.h>

int getrlimit(...,	int getrlimit64(...,
struct rlimit *);	struct rlimit64 *);
int setrlimit(...,	int setrlimit64(...,
const struct rlimit *);	const struct rlimit64 *);

<sys/sendfile.h>

ssize_t sendfile(...,	ssize_t sendfile64(...,
off_t *, ...);	off64_t *, ...);
ssize_t sendfilev(..., const	ssize_t sendfilev64(..., const
struct sendfilevec *, ...);	struct sendfilevec64 *, ...);

<sys/stat.h>

int fstat(..., struct stat *);	int fstat64(..., struct stat64 *);
int fstatat(...,	int fstatat64(...,
struct stat *, int);	struct stat64 *, int);
int lstat(..., struct stat *);	int lstat64(..., struct stat64 *);
int stat(..., struct stat *);	int stat64(..., struct stat64 *);

<sys/statvfs.h>

int statvfs(...,	int statvfs64(...,
struct statvfs *);	struct statvfs64 *);
int fstatvfs(...,	int fstatvfs64(...,
struct statvfs *);	struct statvfs64 *);

<ucbinclude/stdio.h>

FILE *fopen()	FILE *fopen64()
FILE *freopen()	FILE *freopen64()

<ucbinclude/sys/dir.h>

int alphasort(	int alphasort64(
struct direct **,	struct direct64 **,
struct direct **);	struct direct64 **);
struct direct *readdir();	struct direct64 *readdir64();
int scandir(...,	int scandir64(...,
struct direct *(*[]);, ...);	struct direct64 *(*[]);, ...);

<unistd.h>

int lockf(..., off_t);	int lockf64(..., off64_t);
off_t lseek(..., off_t, ...);	off64_t lseek64(..., off64_t, ...);
int ftruncate(..., off_t);	int ftruncate64..., off64_t);
ssize_t pread(..., off_t);	ssize_t pread64..., off64_t);
ssize_t pwrite(..., off_t);	ssize_t pwrite64(..., off64_t);
int truncate(..., off_t);	int truncate64(..., off64_t);

另请参见

lfcompile(5), lfcompile64(5)

引用名

lfcompile – large file compilation environment for 32-bit applications

描述

All 64-bit applications can manipulate large files by default. The methods described on this page allow 32-bit applications to manipulate large files.

In the large file compilation environment, source interfaces are bound to appropriate 64-bit functions, structures, and types. Compiling in this environment allows 32-bit applications to access files whose size is greater than or equal to 2 Gbyte (2^{31} bytes).

Each interface named *xxx*`()` that needs to access 64-bit entities to access large files maps to a *xxx*`64()` call in the resulting binary. All relevant data types are defined to be of correct size (for example, `off_t` has a typedef definition for a 64-bit entity).

An application compiled in this environment is able to use the *xxx*`()` source interfaces to access both large and small files, rather than having to explicitly utilize the transitional *xxx*`64()` interface calls to access large files. See the `lfcompile64`(5) manual page for information regarding the transitional compilation environment.

Applications can be compiled in the large file compilation environment by using the following methods:

- Use the `getconf`(1) utility with one or more of the arguments listed in the table below. This method is recommended for portable applications.

argument	purpose
LFS_CFLAGS	obtain compilation flags necessary to enable the large file compilation environment
LFS_LDFLAGS	obtain link editor options
LFS_LIBS	obtain link library names
LFS_LINTFLAGS	obtain lint options

- Set the compile-time flag `_FILE_OFFSET_BITS` to 64 before including any headers. Applications may combine objects produced in the large file compilation environment with objects produced in the transitional compilation environment, but must be careful with respect to interoperability between those objects. Applications should not declare global variables of types whose sizes change between compilation environments.

Access to Additional Large File Interfaces

The `fseek()` and `ftell()` functions *do not* map to functions named `fseek64()` and `ftell64()`; rather, the large file additions `fseeko()` and `ftello()`, have functionality identical to `fseek()` and `ftell()` and *do* map to the 64-bit functions `fseeko64()` and `ftello64()`. Applications wishing to access large files should use `fseeko()` and `ftello()` in place of `fseek()` and `ftell()`. See the `fseek`(3C) and `ftell`(3C) manual pages for information about `fseeko()` and `ftello()`.

Applications wishing to access fseeko() and ftello() as well as the POSIX and X/Open specification-conforming interfaces should define the macro _LARGEFILE_SOURCE to be 1 and set whichever feature test macros are appropriate to obtain the desired environment (see standards(5)).

示例

In the following examples, the large file compilation environment is accessed by invoking the getconf utility with one of the arguments listed in the table above. The additional large file interfaces are accessed by specifying –D_LARGEFILE_SOURCE.

The examples that use the form of command substitution specifying the command within parentheses preceded by a dollar sign can be executed only in a POSIX-conforming shell such as the Korn Shell (see ksh(1)). In a shell that is not POSIX-conforming, such as the Bourne Shell (see sh(1)) and the C Shell (see csh(1)), the getconf calls must be enclosed within grave accent marks, as shown in the second example.

示例 1 Compile a program with a "large" off_t that uses fseeko(), ftello(), and yacc.

The following example compiles a program with a "large" off_t and uses fseeko(), ftello(), and yacc(1).

```
$ c89 -D_LARGEFILE_SOURCE                \
      -D_FILE_OFFSET_BITS=64 -o foo      \
      $(getconf LFS_CFLAGS) y.tab.c b.o  \
      $(getconf LFS_LDFLAGS)             \
      -ly $(getconf LFS_LIBS)
```

示例 2 Compile a program with a "large" off_t that does not use fseeko() and ftello() and has no application specific libraries.

```
% c89 -D_FILE_OFFSET_BITS=64     \
     ` getconf LFS_CFLAG`S a.c   \
     ` getconf LFS_LDFLAG`S      \
     ` getconf LFS_LIB`S         \
```

示例 3 Compile a program with a "default" off_t that uses fseeko() and ftello().

```
$ c89 –D_LARGEFILE_SOURCE  a.c
```

另请参见

csh(1), getconf(1), ksh(1), yacc(1), sh(1), fseek(3C), ftell(3C), lf64(5), lfcompile64(5), standards(5)

附注

Certain system-specific or non-portable interfaces are not usable in the large file compilation environment. Known cases are:

- Kernel data structures read from /dev/kmem.
- Interfaces in the kernel virtual memory library, –lkvm.
- Interfaces in the ELF access library, –lelf.
- Interfaces to /proc defined in <procfs.h>.
- The ustat(2) system call.

Programs that use these interfaces should not be compiled in the large file compilation environment. As a partial safeguard against making this mistake, including either of the `<libelf.h>` or `<sys/procfs.h>` header files will induce a compilation error when the large file compilation environment is enabled.

In general, caution should be exercised when using any separately-compiled library whose interfaces include data items of type `off_t` or the other redefined types either directly or indirectly, such as with '`struct stat`'. (The redefined types are `off_t`, `rlim_t`, `ino_t`, `blkcnt_t`, `fsblkcnt_t`, and `fsfilcnt_t`.) For the large file compilation environment to work correctly with such a library, the library interfaces must include the appropriate *xxx*`64()` binary entry points and must have them mapped to the corresponding primary functions when `_FILE_OFFSET_BITS` is set to 64.

Care should be exercised using any of the `printf()` or `scanf()` routines on variables of the types mentioned above. In the large file compilation environment, these variables should be printed or scanned using `long long` formats.

已知问题

Symbolic formats analogous to those found in `<sys/int_fmtio.h>` do not exist for printing or scanning variables of the types that are redefined in the large file compilation environment.

引用名

lfcompile64 – transitional compilation environment

描述

All 64-bit applications can manipulate large files by default. The transitional interfaces described on this page can be used by 32-bit and 64-bit applications to manipulate large files.

In the transitional compilation environment, explicit 64-bit functions, structures, and types are added to the API. Compiling in this environment allows both 32-bit and 64-bit applications to access files whose size is greater than or equal to 2 Gbyte (2^{31} bytes).

The transitional compilation environment exports all the explicit 64-bit functions (*xxx*`64()`) and types in addition to all the regular functions (*xxx*`()`) and types. Both *xxx*`()` and *xxx*`64()` functions are available to the program source. A 32-bit application must use the *xxx*`64()` functions in order to access large files. See the `lf64(5)` manual page for a complete listing of the 64-bit transitional interfaces.

The transitional compilation environment differs from the large file compilation environment, wherein the underlying interfaces are bound to 64-bit functions, structures, and types. An application compiled in the large file compilation environment is able to use the *xxx*`()` source interfaces to access both large and small files, rather than having to explicitly utilize the transitional *xxx*`64()` interface calls to access large files. See the `lfcompile(5)` manual page for more information regarding the large file compilation environment.

Applications may combine objects produced in the large file compilation environment with objects produced in the transitional compilation environment, but must be careful with respect to interoperability between those objects. Applications should not declare global variables of types whose sizes change between compilation environments.

For applications that do not wish to conform to the POSIX or X/Open specifications, the 64-bit transitional interfaces are available by default. No compile-time flags need to be set.

Access to Additional Large File Interfaces

Applications that wish to access the transitional interfaces as well as the POSIX or X/Open specification-conforming interfaces should use the following compilation methods and set whichever feature test macros are appropriate to obtain the desired environment (see `standards(5)`).

- Set the compile-time flag `_LARGEFILE64_SOURCE` to 1 before including any headers.
- Use the `getconf(1)` command with one or more of the following arguments:

argument	purpose
LFS64_CFLAGS	obtain compilation flags necessary to enable the transitional compilation environment
LFS64_LDFLAGS	obtain link editor options
LFS64_LIBS	obtain link library names
LFS64_LINTFLAGS	obtain lint options

示例

In the following examples, the transitional compilation environment is accessed by invoking the getconf utility with one of the arguments listed in the table above. The additional large file interfaces are accessed either by specifying –D_LARGEFILE64_SOURCE or by invoking the getconf utility with the arguments listed above.

The example that uses the form of command substitution specifying the command within parentheses preceded by a dollar sign can be executed only in a POSIX-conforming shell such as the Korn Shell (see ksh(1)). In a shell that is not POSIX-conforming, such as the Bourne Shell (see sh(1)) and the C Shell (see csh(1)), the command must be enclosed within grave accent marks.

示例 1 An example of compiling a program using transitional interfaces such as lseek64() and fopen64():

```
$ c89 -D_LARGEFILE64_SOURCE        \
      $(getconf LFS64_CFLAGS) a.c  \
      $(getconf LFS64_LDFLAGS)     \
      $(getconf LFS64_LIBS)
```

示例 2 An example of running lint on a program using transitional interfaces:

```
% lint -D_LARGEFILE64_SOURCE        \
     ` getconf LFS64_LINTFLAG`S ... \
     ` getconf LFS64_LIB`S
```

另请参见

getconf(1), lseek(2), fopen(3C), lf64(5), standards(5)

引用名

locale – 用户环境的子集，该子集取决于语言和文化习惯

描述

`locale` 定义了用户环境的子集，该子集取决于语言和文化习惯。它由一个或多个类别组成。每个类别都由其名称标识，用于控制系统组件行为的特定方面。类别名称对应于以下环境变量名称：

`LC_CTYPE`	字符分类和大小写转换。
`LC_COLLATE`	整理顺序。
`LC_TIME`	日期和时间格式。
`LC_NUMERIC`	数字格式。
`LC_MONETARY`	货币格式。
`LC_MESSAGES`	信息性和诊断性信息与交互式响应的格式。

标准实用程序的行为基于当前语言环境，如每个实用程序的 "ENVIRONMENT VARIABLES"（环境变量）部分中所定义。一些 C 语言函数的行为也会基于当前语言环境而改变，如最后对 `setlocale`(3C) 的调用所定义的那样。

由实现提供的语言环境之外的语言环境可以由应用程序通过 `localedef`(1) 实用程序创建。使用环境变量时用于指定语言环境的值将是创建语言环境时指定为 `localedef` 的 *name* 操作数的字符串。字符串 “C” 和 “POSIX” 保留为 POSIX 语言环境的标识符。

应用程序可以通过调用具有相应值的 `setlocale()` 函数来选择所需的语言环境。如果调用带空字符串的函数，例如：

```
setlocale(LC_ALL, "");
```

将使用对应环境变量的值。如果取消设置环境变量或者将环境变量设置为空字符串，则 `setlocale()` 函数将设置相应的环境。

语言环境定义

语言环境可以使用 `localedef` 实用程序接受的文件格式进行说明。

语言环境定义文件必须包含一个或多个语言环境类别源定义，且不得包含同一语言环境类别的多个定义。

类别源定义由类别头、类别正文和类别尾部组成。类别头包含类别的字符串命名，以字符 `LC_` 开头。类别尾部包含字符串 `END`，后跟一个或多个空白字符和对应类别头中使用的字符串。

类别正文包含一行或多行文本。每行包含一个标识符，后面跟可选的一个或多个操作数。标识符是标识特定语言环境元素或整理元素的关键字。语言环境内的每个关键字必须具有一个唯一名称（即两个类别不能有名称相同的关键字）。关键字不能以字符 `LC_` 开头。必须用一个或多个空白字符将标识符与操作数隔开。

操作数必须是字符、整理元素或字符串。字符串必须使用双引号 (") 括起来。字符串中的文本双引号前面必须带有 <**转义字符**>，如下所述。当关键字后跟多个操作数时，必须使用分号 (;) 分隔这些操作数。分号前后都允许使用空白字符。

文件中的第一个类别头前面可以有用于修改注释字符的行。它具有以下格式，从第 1 列开始：

"comment_char %c\n",<*comment character*>

注释字符缺省为井号 (#)。第一个位置中的空白行和包含 <**注释字符**> 的行将被忽略。

文件中的第一个类别头前面可以有用于修改要在文件中使用的转义符的行。它具有以下格式，从第 1 列开始：

"escape_char %c\n",<*escape character*>

转义符缺省为反斜杠。

将转义符作为行的最后一个字符可以续行；输入中的这种续行符将被丢弃。虽然实现不需要接受连续行中长度超过 {LINE_MAX} 个字节的部分，但它不会对连续行的累计长度施加限制。注释行无法使用转义的换行符进行续行。

单个字符、字符串中的字符和整理字符必须使用符号名称表示，如下面所定义。此外，字符可以使用该字符自身表示，或者表示为八进制、十六进制或十进制常量。使用非符号表示法时，所得到的语言环境定义在很多情况下无法在各系统之间移植。左尖括号 (<) 是保留符号，表示符号名称的开始，用于表示自身时，前面必须加转义字符。以下规则适用于字符表示形式：

1. 字符可以通过括在尖括号（< 和 >）内的符号名称表示。该符号名称（包括尖括号）必须与通过 localedef - f 选项指定的字符映射文件中定义的符号名称完全一致，并将替换为由与字符映射文件中的符号名称关联的值确定的字符值。使用字符映射文件中没有的符号名称将会导致错误，除非类别是 LC_CTYPE 或 LC_COLLATE，在这种情况下将会导致警告（有关由于错误和警告导致的操作的说明，请参见 localedef(1)）。如果 collating-element 或 collating-symbol 部分中指定的符号名称与字符映射文件（如果存在）中的符号名称重复，则是错误做法。在符号名称中使用转义符或右尖括号无效，除非该字符的前面带有转义符。

   示例：

   ```
   <C>;<c-cedilla> "<M><a><y>"
   ```

2. 一个字符可以用该字符自身表示，在这种情况下，该字符的值与实现相关。在字符串中，双引号字符、转义符和右尖括号字符必须进行转义（前面带有转义符），才能解释为该字符自身。在字符串外部，以下字符

   ,　;　<　> *escape_char*

   必须进行转义才能解释为该字符自身。

   示例：

```
c       "May"
```

3. 字符可以表示为八进制常量。八进制常量指定为转义符后跟两个或更多个八进制数字。每个常量表示一个字节值。多字节值可以用按字节顺序指定的串联常量表示，最后一个常量指定字符的最低有效字节。

   示例：

   ```
   \143;\347;\143\150    "\115\141\171"
   ```

4. 字符可以表示为十六进制常量。十六进制常量指定为转义符后跟 x，再跟两个或更多个十六进制数字。每个常量表示一个字节值。多字节值可以用按字节顺序指定的串联常量表示，最后一个常量指定字符的最低有效字节。

   示例：

   ```
   \x63;\xe7;\x63\x68    "\x4d\x61\x79"
   ```

5. 字符可以表示为十进制常量。十进制常量指定为转义字符后跟 d，再跟两个或更多个十进制数字。每个常量表示一个字节值。多字节值可以用按字节顺序指定的串联常量表示，最后一个常量指定字符的最低有效字节。

   示例：

   ```
   \d99;\d231;\d99\d104   "\d77\d97\d121"
   ```

   无论是使用符号名称、字符自身，还是八进制、十进制或十六进制常量，只能指定为其创建语言环境定义的字符集中存在的字符。如果存在字符映射文件，则只能使用八进制、十进制或十六进制常量指定字符映射文件中定义的字符。可以指定字符映射文件中不存在的符号名称，但这些名称将被忽略，如上面的第 1 项下所述。

LC_CTYPE

LC_CTYPE 类别定义字符分类、大小写转换和其他字符属性。此外，可以通过表示省略号的三个相邻句点 (...) 表示一系列字符。指定的省略号将解释为以下含义：省略号前面和后跟的值之间的所有值表示有效字符。指定的省略号仅在单编码字符集（即一组相同大小的字符）中有效。省略号将解释为在列表中包括以下所有字符，即编码值大于省略号前面字符的编码值、小于省略号后跟字符的编码值的所有字符。

示例：

```
\x30;...;\x39;
```

在字符类中包括编码值介于端点之间的所有字符。

可以识别以下关键字。在说明中，术语“自动包括”意味着包括或省略所引用的任何字符都不属于错误。

字符类 digit、xdigit、lower、upper 和 space 具有一组自动包括的字符。仅当字符值（即编码）不同于实现缺省值时，才需要指定这些字符。

upper 定义要归类为大写字母的字符。

在POSIX语言环境中，包括26个大写字母：

```
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
```

在语言环境定义文件中，不能指定为关键字cntrl、digit、punct或space指定的字符。大写字母A到Z自动包括在此类中。

`lower`

定义要归类为小写字母的字符。在POSIX语言环境中，包括26个小写字母：

```
a b c d e f g h i j k l m n o p q r s t u v w x y z
```

在语言环境定义文件中，不能指定为关键字cntrl、digit、punct或space指定的字符。可移植字符集中的小写字母a到z自动包括在此类中。

`alpha`

定义要归类为字母的字符。

在POSIX语言环境中，包括upper和lower类中的所有字符。

在语言环境定义文件中，不能指定为关键字cntrl、digit、punct或space指定的字符。归类为upper或lower的字符自动包括在此类中。

`digit`

定义要归类为数字的字符。

在POSIX语言环境中，仅包括

```
0 1 2 3 4 5 6 7 8 9
```

字符。

在语言环境定义文件中，只能指定数字0、1、2、3、4、5、6、7、8和9（以数值的连续升序顺序指定）。可移植字符集中的数字0到9自动包括在此类中。

定义字符类digit仅需要十个字符；可以指定定义数字的字符；不能在此处指定替代数字（例如印地语或日文汉字）。

`alnum`

定义要归类为字母和数字的字符。只能指定为alpha和digit关键字指定的字符。为关键字alpha和digit指定的字符自动包括在此类中。

`space`

定义要归类为空格字符的字符。

在POSIX语言环境中，至少包括字符SPACE、FORMFEED、NEWLINE、CARRIAGE RETURN、TAB和VERTICAL TAB。

在语言环境定义文件中，不能指定为关键字upper、lower、alpha、digit、graph或xdigit指定的字符。可移

植字符集中的 SPACE、FORMFEED、NEWLINE、CARRIAGE RETURN、TAB 和 VERTICAL TAB 字符以及 blank 类中包括的任何字符都自动包括在此类中。

cntrl

定义要归类为控制字符的字符。

在 POSIX 语言环境中，不包括 alpha 或 print 类中的字符。

在语言环境定义文件中，不能指定为关键字 upper、lower、alpha、digit、punct、graph、print 或 xdigit 指定的字符。

punct

定义要归类为标点字符的字符。

在 POSIX 语言环境中，不包括空格字符，也不包括 alpha、digit 或 cntrl 类中的任何字符。

在语言环境定义文件中，不能指定为关键字 upper、lower、alpha、digit、cntrl 或 xdigit 指定的字符或指定为空格字符的字符。

graph

定义要归类为可列显字符的字符，不包括空格字符。

在 POSIX 语言环境中，包括 alpha、digit 和 punct 类中的所有字符；不包括 cntrl 类中的字符。

在语言环境定义文件中，为关键字 upper、lower、alpha、digit、xdigit 和 punct 指定的字符自动包括在此类中。不能指定为关键字 cntrl 指定的字符。

print

定义要归类为可列显字符的字符，包括空格字符。

在 POSIX 语言环境中，包括 graph 类中的所有字符；不包括 cntrl 类中的字符。

在语言环境定义文件中，为关键字 upper、lower、alpha、digit、xdigit、punct 指定的字符和空格字符自动包括在此类中。不能指定为关键字 cntrl 指定的字符。

xdigit

定义要归类为十六进制数字的字符。

在 POSIX 语言环境中，仅包括：

```
0 1 2 3 4 5 6 7 8 9 A B C D E F a b c d e f
```

字符。

在语言环境定义文件中，只能指定为类 digit 定义的字符（以数值的连续升序顺序指定），后跟表示十六进制数字 10 到 15（包括这两个数字）的由六个字符组成的一个或多个字符集，每个字符

集都按升序排列（例如 A, B, C, D, E, F, a, b, c, d, e, f）。可移植字符集中的数字 0 到 9、大写字母 A 到 F 和小写字母 a 到 f 自动包括在此类中。

定义字符类 xdigit 要求此处同时包括字符类 digit 中包括的字符。

blank

定义要归类为空白字符的字符。

在 POSIX 语言环境中，仅包括空格字符和制表符。

在语言环境定义文件中，空格字符和制表符自动包括在此类中。

charclass

以分号分隔的字符串形式定义一个或多个语言环境特定的字符类名称。随后可以在 LC_CTYPE 定义中定义每个指定的字符类。字符类名称由可移植文件名字符集中的字母数字字符组成，至少一个字节，至多 {CHARCLASS_NAME_MAX} 个字节。字符类名称的第一个字符不能是数字。该名称不能与本文档中定义的任何 LC_CTYPE 关键字相同。

charclass-name

定义要归类为属于指定语言环境特定的字符类的字符。在 POSIX 语言环境中，指定的语言环境特定的字符类不需要存在。如果用 charclass 关键字定义了类名称，但随后未将任何字符分配给该名称，这不算是错误；它表示该类不包含任何属于它的字符。charclass-name 可以用作 wctype(3C) 函数的 *property* 参数，在正则表达式和 shell 模式匹配括号表达式中使用，以及供 tr(1) 命令使用。

toupper

定义小写字母与大写字母之间的映射。

在 POSIX 语言环境中，至少以下 26 个小写字符：

a b c d e f g h i j k l m n o p q r s t u v w x y z

将映射到对应的 26 个大写字符：

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

在语言环境定义文件中，操作数由分号分隔的字符对组成。每个字符对中的字符由逗号分隔，并且该对用括号括起。每对中的第一个字符是小写字母，第二个字符是对应的大写字母。只能指定为关键字 lower 和 upper 指定的字符。可移植字符集中的小写字母 a 到 z 及其对应的大写字母 A 到 Z 自动包括在此映射中，但仅当语言环境定义中省略了 toupper 关键字时才会包括。

tolower

定义大写字母与小写字母之间的映射。

在 POSIX 语言环境中，至少以下 26 个大写字符：

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

将映射到对应的 26 个小写字符：

```
a b c d e f g h i j k l m n o p q r s t u v w x y z
```

在语言环境定义文件中，操作数由分号分隔的字符对组成。每个字符对中的字符由逗号分隔，并且该对用括号括起。每对中的第一个字符是大写字母，第二个字符是对应的小写字母。只能指定为关键字 lower 和 upper 指定的字符。如果语言环境定义中省略了 tolower 关键字，则该映射将是为 toupper 指定的映射的反向映射。

LC_COLLATE

LC_COLLATE 类别为大量实用程序（例如 sort(1)、uniq(1) 等）、正则表达式匹配（请参见 regex(5)）以及 strcoll(3C)、strxfrm(3C)、wcscoll(3C) 和 wcsxfrm(3C) 函数提供整理序列定义。

整理序列定义用于定义语言环境中的整理元素（字符和多字符整理元素）之间的相对顺序。此顺序以整理值表示，即通过为每个元素分配一个或多个整理值（也称为整理权重）。提供了以下功能：

1. **多字符整理元素**。指定多字符整理元素（即要整理为一个实体的两个或更多个字符的序列）。
2. **用户定义的整理元素顺序**。每个整理元素将分配一个用于定义其在字符（或基本）整理序列中的顺序的整理值。此顺序用于正则表达式和模式匹配，还用作排序的整理权重（除非另外显式指定了整理权重）。
3. **多个权重和等效类**。可以为整理元素分配一个或多个要在排序中使用的整理权重（上限为 {COLL_WEIGHTS_MAX}）。以下将第一个权重称为主权重。
4. **一对多映射**。将单个字符映射到一个整理元素字符串。
5. **等效类定义**。两个或更多个整理元素具有相同的整理值（主权重）。
6. **按权重排序**。比较两个字符串以确定其相对顺序时，这两个字符串将先拆分为一系列整理元素。然后，将根据元素的相对主权重比较每个后续元素对中的元素。如果该权重相等且分配有多个权重，则将根据后续相对权重重新比较整理元素对，直至一对整理元素的比较结果不相同或权重用尽。

在整理序列定义中可识别以下关键字。以下各部分中详细介绍了这些关键字。

copy
指定用作此类别定义的现有语言环境的名称。如果指定了此关键字，则不能指定其他关键字。

collating-element
定义用于表示多字符整理元素的整理元素符号。此关键字是可选的。

collating-symbol
定义要在整理顺序声明中使用的整理符号。此关键字是可选的。

order_start
定义整理规则。此声明后跟一个或多个整理顺序声明，用于为整理元素分配字符整理值和整理权重。

order_end 指定整理顺序声明到此结束。

collating-element 关键字

除了字符集中的整理元素，collating-element 关键字也用于定义多字符整理元素。语法为：

"collating-element %s from \"%s\"\n",<*collating-symbol*>,<*string*>

<*collating-symbol*> 操作数是符号名称，括在尖括号（< 和 >）内，并且不得与当前字符映射文件（如果有）中的任何符号名称或此整理定义中定义的任何其他符号名称重复。该字符串操作数是由两个或更多个将整理为一个实体的字符组成的字符串。通过此关键字定义的 <*collating-element*> 将仅识别为 LC_COLLATE 类别。

示例：

```
collating-element <ch> from “<c><h>”
collating-element <e-acute> from “<acute><e>”
collating-element <ll> from “ll”
```

collating-symbol 关键字

此关键字用于定义要在整理序列声明中使用的符号；即，用在 order_start 和 order_end 关键字之间。语法为：

"collating-symbol %s\n",<*collating-symbol*>

<*collating-symbol*> 是符号名称，括在尖括号（< 和 >）内，并且不得与当前字符映射文件（如果有）中的任何符号名称或此整理定义中定义的任何其他符号名称重复。

通过此关键字定义的 collating-symbol 将仅识别为 LC_COLLATE 类别。

示例：

```
collating-symbol <UPPER_CASE>
collating-symbol <HIGH>
```

collating-symbol 关键字定义可以与字符顺序序列中的相对位置关联的符号名称。虽然此类符号名称不表示任何整理元素，但它可用作权重。

order_start 关键字

order_start 关键字必须位于整理顺序条目之前，同时定义此整理序列定义和其他整理规则的权重数目。

order_start 关键字的语法为：

"order_start %s;%s;...;%s\n",<*sort-rules*>,<*sort-rules*>

order_start 关键字的操作数是可选的。如果存在，则操作数定义要在比较字符串时应用的规则。操作数的数目定义为每个元素分配多少个权重。如果不存在任何操作数，则将假定使用一个 forward 操作数。如果存在，则第一个操作数定义在使用第一个（主）权重比较字符串时要应用的规则；第二个操作数定义在使用第二个权重比较

字符串时要应用的规则，依此类推。各个操作数使用分号分隔(;)。每个操作数由逗号(,)分隔的一个或多个整理指令组成。如果操作数的数目超过{COLL_WEIGHTS_MAX}限制，则实用程序将发出一条警告消息。支持以下指令：

forward　指定权重级别的比较操作从字符串开头向字符串结尾执行。

backward　指定权重级别的比较操作从字符串末尾向字符串开头执行。

position　指定权重级别的比较操作将考虑字符串中不受IGNORE影响的元素的相对位置。从比较的开始位置起，在受IGNORE影响的整理元素之后最先出现不受IGNORE影响的元素的那个字符串首先整理。如果两个字符串在同一相对位置包含不受IGNORE影响的字符，则由分配给元素的整理值确定排序顺序。在相等的情况下，将按照相同方式检查不受IGNORE影响的后续字符。

指令forward和backward互斥。

示例：

```
order_start    forward;backward
```

如果未指定任何操作数，则假定使用单个forward操作数。

整理顺序

order_start关键字后跟整理标识符条目。整理元素条目的语法为：

"%s %s;%s;...;%s\n"<*collating-identifier*>,<*weight*>,<*weight*>,...

每个*collating-identifier*都包含上面的“语言环境定义”中所述的一个字符、一个<*collating-element*>、一个<*collating-symbol*>、一个省略号或特殊符号UNDEFINED。所指定的整理元素顺序将确定字符顺序序列，这样每个整理元素比较的元素都少于其后跟元素的数目。NUL字符的比较优先级低于任何其他字符。

<*collating-element*>用于指定多字符整理元素，并指示通过<*collating-element*>指定的字符序列将作为一个单元整理且该序列的相对顺序由元素位置指定。

<*collating-symbol*>用于定义要在权重使用的元素在相对顺序中的位置。不使用<*collating-symbol*>指定权重。

省略号指定一系列要将根据其编码字符值进行整理的字符。省略号解释为，当前编码字符集中编码字符集值大于上一行中字符的值、小于下一行中字符的编码字符集值的所有元素，将放置在字符整理顺序中的上一个字符和下一个字符之间，并根据其编码字符集值按升序排列。以省略号开头解释为上一行指定了空字符，以省略号结尾解释为下一行指定了当前编码字符集中最高的编码字符集值。如果上一行或下一行未指定当前编码字符集中的字符，则省略号将视为无效。使用省略号会将定义与特定的编码字符集相关联，并可能会阻止该定义在不同实现之间移植。

符号UNDEFINED解释为包括未显式指定或通过省略号指定的所有编码字符集值。此类字符将按照该符号指示的位置插入字符整理顺序中，并根据其编码字符集值按升序排

列。如果未指定 UNDEFINED 符号，并且当前编码字符集包含此部分中未指定的字符，则实用程序将发出一条警告消息，并将此类字符放置在字符整理顺序结尾。

每个 collation-element 的可选操作数用于定义整理元素的主权重、次权重或后续权重。第一个操作数指定相对主权重，第二个操作数指定相对次权重，依此类推。可以为两个或更多个 collation-element 分配相同的权重；如果这些元素具有相同的主权重，则它们属于同一**等效类**。整理的行为方式如下，对于每个权重级别，受 IGNORE 影响的元素将被删除，除非使用 order_start 关键字为相应的级别指定了 position 整理指令。然后，根据元素的相对权重比较每个后续元素对。如果两个字符串的比较结果为相等，则将对下一个权重级别重复该过程，直到达到上限 {COLL_WEIGHTS_MAX}。

权重使用上述“语言环境定义”中说明的字符、<*collating-symbol*>、<*collating-element*>、省略号或特殊符号 IGNORE 表示。单个字符、<*collating–symbol*> 或 <*collating–element*> 表示在字符或符号的字符整理序列中的相对位置，而不是字符自身。因此，不为权重分配绝对值，而是基于整理元素在字符整理序列中的顺序，使用分配给该整理元素的相对顺序值表示特定权重。

一对多映射通过指定两个或更多个串联的字符或符号名称来指示。如果为字符 <eszet> 指定了字符串 “<s><s>” 作为权重，则执行比较时就像出现的所有 <eszet> 字符都替换为 <s><s>（假定 <s> 具有整理权重 <s>）。如果必须将 <eszet> 和 < s><s> 定义为等效类，则必须为字符串 ss 定义整理元素。

通过省略号指定的所有字符缺省情况下将会分配唯一的权重，该权重等于字符的相对顺序。通过显式或隐式 UNDEFINED 特殊符号指定的字符缺省情况下将会分配相同的主权重（即，属于同一等效类）。作为权重的省略号会解释为，该序列中的每个字符都有唯一权重，该权重等于字符整理序列中该字符的相对顺序。如果整理元素既不是省略号也不是特殊符号 UNDEFINED，则使用省略号作为权重将会视为错误。

将特殊关键字 IGNORE 作为权重表示，当在指定 IGNORE 的级别使用权重比较字符串时，将忽略整理元素，就像该字符串不包含整理元素一样。在正则表达式和模式匹配中，受其主权重的 IGNORE 影响的所有字符将构成一个等效类。

所有空操作数将解释为整理元素自身。

例如，顺序声明：

```
<a>       <a>;<a>
```

等效于：

```
<a>
```

如果整理元素为省略号，则可以使用省略号作为操作数，并且该省略号将解释为通过省略号定义的每个字符的值。

此部分中定义的整理顺序定义正则表达式中方括号表达式的解释方式。

示例：

```
order_start                               forward;backward
UNDEFINED                                 IGNORE;IGNORE
<LOW>
<space>                                   <LOW>;<space>
. . .                                     <LOW>;. . .
<a>                                       <a>;<a>
<a-acute>                                 <a>;<a-acute>
<a-grave>                                 <a>;<a-grave>
<A>                                       <a>;<A>
<A-acute>                                 <a>;<A-acute>
<A-grave>                                 <a>;<A-grave>
<ch>                                      <ch>;<ch>
<Ch>                                      <ch>;<Ch>
<s>                                       <s>;<s>
<eszet>                                   "<s><s>";"<eszet><eszet>"
order_end
```

此示例的解释如下：

1. UNDEFINED 意味着，整理时将忽略此定义中未指定的所有字符（未显式指定或通过省略号指定）；对于正则表达式，将首先对这些字符进行排序。
2. <space> 和 <a> 之间的所有字符具有相同的主等效类，各个字符的次权重基于其序数编码值。
3. 基于大写字符或小写字符 a 的所有字符属于相同的主等效类。
4. 多字符整理元素 <ch> 由整理符号 <ch> 表示，属于与多字符整理元素 <Ch> 相同的主等效类。

order_end 关键字

整理顺序条目必须使用 order_end 关键字结尾。

LC_MONETARY

LC_MONETARY 类别定义用于设置货币数值信息格式的规则和符号。通过 localeconv(3C) 函数可获取这些信息

在语言环境的此类别中定义了以下项。定义语言环境时，这些项的名称是 localedef(1) 实用程序可识别的关键字。这些名称也类似于 <locale.h> 中定义的

lconv 结构的成员名称。对于未指定的整数项，localeconv 函数返回 {CHAR_MAX}；对于未指定或大小为零的字符串项，则返回空字符串 ("")。

在语言环境定义文件中，这些操作数是字符串。对于某些关键字，字符串只能包含整数。未提供的关键字、设置为空字符串的字符串值 ("") 或者设置为 –1 的整数关键字用于指示该值在语言环境中不可用。

int_curr_symbol

国际货币符号。操作数是一个由四个字符组成的字符串，前三个字符包含根据 ISO 4217 标准指定的字母式国际货币符号。第四个字符是用于将国际货币符号与货币数量隔开的字符。

currency_symbol

用作本地货币符号的字符串。

mon_decimal_point

操作数是包含某个符号的字符串，该符号在货币格式的数量中用作十进制分隔符（基数字符）。

mon_thousands_sep

操作数是包含某个符号的字符串，该符号在货币格式的数量中用作划分十进制分隔符左侧数字的分隔符。

mon_grouping

定义货币格式的数量中每组数字的大小。操作数是使用分号分隔的一系列整数。每个整数指定每一组中数字的个数，第一个整数定义紧接在十进制分隔符前面的组的大小，后续整数定义后续组。如果最后一个整数不是 –1，则上一个组（如果有）的大小将重复用于其余数字。如果最后一个整数是 –1，则不执行进一步的分组。

下面是 mon_grouping 关键字的一个解释示例。假定要设置格式的值为 123456789，mon_thousands_sep 为 ’，下表显示了相应结果。第三列显示了 ISO C 标准的等效字符串，localeconv 函数将使用该字符串来适应此分组。

mon_grouping	Formatted Value	ISO C String
3;-1	123456’789	"\3\177"
3	123’456’789	"\3"
3;2;-1	1234’56’789	"\3\2\177"
3;2	12’34’56’789	"\3\2"
-1	1234567898	"\177"

在这些示例中，{CHAR_MAX} 的八进制值为 177。

positive_sign

用于指示非负值格式的货币数量的字符串。

negative_sign

用于指示负值格式的货币数量的字符串。

int_frac_digits

表示使用 int_curr_symbol 时写入货币格式的数量中的小数数字（即十进制分隔符右侧的数字）个数的整数。

frac_digits

表示使用currency_symbol时写入货币格式的数量中的小数数字（即十进制分隔符右侧的数字）个数的整数。

p_cs_precedes

在遵循SUSv3标准的应用程序中，如果currency_symbol位于具有非负值的货币数量值之前，则是设置为1的整数，如果该符号位于值之后，则是设置为0的整数。

在不遵循SUSv3标准的应用程序中，如果currency_symbol或int_currency_symbol位于具有非负值的货币数量值之前，则是设置为1的整数，如果该符号位于值之后，则是设置为0的整数。

p_sep_by_space

在遵循SUSv3标准的应用程序中，如果未使用空格将currency_symbol与具有非负值的货币数量值隔开，则是设置为0的整数，如果使用了空格将该符号与值隔开，则是设置为1的整数，如果使用了空格将该符号与符号字符串（如果相邻）隔开，则是设置为2的整数。

在不遵循SUSv3标准的应用程序中，如果未使用空格将currency_symbol或int_curr_symbol与具有非负值的货币数量值隔开，则是设置为0的整数，如果使用了空格将该符号与值隔开，则是设置为1的整数，如果使用了空格将该符号与符号字符串（如果相邻）隔开，则是设置为2的整数。

n_cs_precedes

在遵循SUSv3标准的应用程序中，如果currency_symbol位于具有负值的货币数量值之前，则是设置为1的整数，如果该符号位于值之后，则是设置为0的整数。

在不遵循SUSv3标准的应用程序中，如果currency_symbol或int_currency_symbol位于具有负值的货币数量值之前，则是设置为1的整数，如果该符号位于值之后，则是设置为0的整数。

n_sep_by_space

在遵循SUSv3标准的应用程序中，如果未使用空格将currency_symbol与具有负值的货币数量值隔开，则是设置为0的整数，如果使用了空格将该符号与值隔开，则是设置为1的整数，如果使用了空格将该符号与符号字符串（如果相邻）隔开，则是设置为2的整数。

在不遵循SUSv3标准的应用程序中，如果未使用空格将currency_symbol或int_curr_symbol与具有负值的货币数量值隔开，则是设置为0的整数，如果使用了空格将该符号与值隔开，则是设置为1的整数，如果使用了空格将该符号与符号字符串（如果相邻）隔开，则是设置为2的整数。

p_sign_posn

设置为某个值的整数，指示如何定位具有非负值的货币数量的 positive_sign。可识别下列 p_sign_posn 和 n_sign_posn 整数值：

在遵循 SUSv3 标准的应用程序中：

0 使用括号将数量和 currency_symbol 括起来。

1 符号字符串位于数量和 currency_symbol 之前。

2 符号字符串位于数量和 currency_symbol 之后。

3 符号字符串位于 currency_symbol 之前。

4 符号字符串位于 currency_symbol 之后。

在不遵循 SUSv3 标准的应用程序中：

0 使用括号将数量和 currency_symbol 或 int_curr_symbol 括起来。

1 符号字符串位于数量和 currency_symbol 或 int_curr_symbol 之前。

2 符号字符串位于数量和 currency_symbol 或 int_curr_symbol 之后。

3 符号字符串位于 currency_symbol 或 int_curr_symbol 之前。

4 符号字符串位于 currency_symbol 或 int_curr_symbol 之后。

n_sign_posn

设置为某个值的整数，指示如何定位负值格式的货币数量的 negative_sign。

int_p_cs_precedes

如果 int_curr_symbol 位于具有非负值的货币数量值之前，则是设置为 1 的整数，如果该符号位于值之后，则是设置为 0 的整数。

int_n_cs_precedes

如果 int_curr_symbol 位于具有负值的货币数量值之前，则是设置为 1 的整数，如果该符号位于值之后，则是设置为 0 的整数。

int_p_sep_by_space

如果未使用空格将 int_curr_symbol 与具有非负值的货币数量值隔开，则是设置为 0 的整数，如果使用了空格将该符号与值隔开，则是设置为 1 的整数，如果使用了空格将该符号与符号字符串（如果相邻）隔开，则是设置为 2 的整数。

int_n_sep_by_space

如果未使用空格将 int_curr_symbol 与具有负值的货币数量值隔开，则是设置为 0 的整数，如果使用了空格将该符号与值

隔开，则是设置为 1 的整数，如果使用了空格将该符号与符号字符串（如果相邻）隔开，则是设置为 2 的整数。

int_p_sign_posn

设置为某个值的整数，指示如何定位国际货币格式的正货币数量的 positive_sign。可识别下列 int_p_sign_posn 和 int_n_sign_posn 整数值：

0 使用括号将数量和 int_curr_symbol 括起来。

1 符号字符串位于数量和 int_curr_symbol 之前。

2 符号字符串位于数量和 int_curr_symbol 之后。

3 符号字符串位于 int_curr_symbol 之前。

4 符号字符串位于 int_curr_symbol 之后。

int_n_sign_posn

设置为某个值的整数，指示如何定位国际货币格式的负货币数量的 negative_sign。

下面显示了各种组合的结果：

		p_sep_by_space		
		2	1	0
p_cs_precedes=1	p_sign_posn=0	($1.25)	($1.25)	($1.25)
	p_sign_posn=1	+$1.25	+$1.25	+$1.25
	p_sign_posn=2	$1.25+	$1.25+	$1.25+
	p_sign_posn=3	+$1.25	+$1.25	+$1.25
	p_sign_posn=4	$+1.25	$+1.25	$+1.25
p_cs_precedes=0	p_sign_posn=0	(1.25 $)	(1.25 $)	(1.25$)
	p_sign_posn=1	+1.25 $	+1.25 $	+1.25$
	p_sign_posn=2	1.25$ +	1.25 $+	1.25$+
	p_sign_posn=3	1.25+ $	1.25 +$	1.25+$
	p_sign_posn=4	1.25$ +	1.25 $+	1.25$+

下面列出了 POSIX 语言环境的货币格式定义。代码列表描述了 localedef(1) 输入，表给出了相同信息，但添加了 localeconv(3C) 和 nl_langinfo(3C) 格式。POSIX 语言环境中未指定所有值。

```
LC_MONETARY
# This is the POSIX locale definition for
# the LC_MONETARY category.
```

```
#
int_curr_symbol       ""
currency_symbol       ""
mon_decimal_point     ""
mon_thousands_sep     ""
mon_grouping          -1
positive_sign         ""
negative_sign         ""
int_frac_digits       -1
frac_digits           -1
p_cs_precedes         -1
p_sep_by_space        -1
n_cs_precedes         -1
n_sep_by_space        -1
p_sign_posn           -1
n_sign_posn           -1
int_p_cs_precedes     -1
int_p_sep_by_space    -1
int_n_cs_precedes     -1
int_n_sep_by_space    -1
int_p_sign_posn       -1
int_n_sign_posn       -1
#
END LC_MONETARY
```

条目 n/a 指示 POSIX 语言环境中未提供该值。

LC_NUMERIC

LC_NUMERIC 类别定义将用于设置非货币数值信息格式的规则和符号。通过 localeconv(3C)函数可获取这些信息。

在语言环境的此类别中定义了以下项。定义语言环境时，这些项的名称是 localedef 实用程序可识别的关键字。这些名称也类似于 <locale.h> 中定义的 *lconv* 结构的成员名称。对于未指定的整数项，localeconv() 函数返回 {CHAR_MAX}；对于未指定或大小为零的字符串项，则返回空字符串 (“”)。

在语言环境定义文件中，这些操作数是字符串。对于某些关键字，字符串只能包含整数。未提供的关键字、设置为空字符串的字符串值 (“”) 或者设置为 –1 的整数关键字用于指示该值在语言环境中不可用。可以识别以下关键字：

decimal_point

操作数是包含某个符号的字符串，该符号在非货币格式的数值数量中用作十进制分隔符（基数字符）。此关键字不能省略并且不能设置为空字符串。在相应标准将 decimal_point 限制为单字节的环境中，指定多字节操作数的结果为未指定。

thousands_sep

操作数是包含某个符号的字符串，该符号在非货币格式的数值数量中用作划分十进制分隔符左侧数字的分隔符。在相应标准将 thousands_sep 限制为单字节的环境中，指定多字节操作数的结果为未指定。

grouping

定义非货币格式的数量中每组数字的大小。操作数是使用分号分隔的一系列整数。每个整数指定每一组中数字的个数，第一个整数定义紧接在十进制分隔符前面的组的大小，后续整数定义后续组。如果最后一个整数不是 –1，则上一个组（如果有）的大小将重复用于其余数字。如果最后一个整数是 –1，则不执行进一步的分组。下面列出了 POSIX 语言环境的非货币数值格式定义。代码列表描述了 localedef 输入，表给出了相同信息，但添加了 localeconv 值和 nl_langinfo 常量。

```
LC_NUMERIC
# This is the POSIX locale definition for
# the LC_NUMERIC category.
#
decimal_point     "<period>"
thousands_sep     ""
grouping    -1
#
END LC_NUMERIC
```

	POSIX 语言环境	langinfo	localeconv()	localedef
项	值	常量	值	值
decimal_point	"."	RADIXCHAR	"."	.
thousands_sep	n/a	THOUSEP	""	""
grouping	n/a	-	""	–1

条目 n/a 指示 POSIX 语言环境中未提供该值。

LC_TIME

LC_TIME 类别定义 date(1) 支持的字段描述符的解释方式，并影响 strftime(3C)、wcsftime(3C)、strptime(3C) 和 nl_langinfo(3C) 函数的行为。由于 C 语言访问和语言环境定义的接口大不相同，因此对其分别进行了说明。对于语言环境定义，可识别下列必需关键字：

abday

定义缩写工作日名称，与 %a 字段描述符（strftime()、wcsftime() 和 strptime() 函数中的转换规范）对应。操作数由七个分号分隔的字符串组成，每一个都括在双引号中。第一个字符串是与星期日对应的那一天的缩写名称，第二个字符串是与星期一对应的那一天的缩写名称，依此类推。

day

定义完整工作日名称，与 %A 字段描述符对应。操作数由七个分号分隔的字符串组成，每一个都括在双引号中。第一个字符串是与星期日对应的那一天的完整名称，第二个字符串是与星期一对应的那一天的完整名称，依此类推。

`abmon` 定义缩写月份名称，与`%b`字段描述符对应。操作数由十二个分号分隔的字符串组成，每一个都括在双引号中。第一个字符串是一年中第一个月（一月）的缩写名称，第二个字符串是第二个月的缩写名称，依此类推。

`mon` 定义完整月份名称，与`%B`字段描述符对应。操作数由十二个分号分隔的字符串组成，每一个都括在双引号中。第一个字符串是一年中第一个月（一月）的完整名称，第二个字符串是第二个月的完整名称，依此类推。

`d_t_fmt` 定义相应日期和时间表示形式，与`%c`字段描述符对应。操作数由一个字符串组成，可以包含字符和字段描述符的任意组合。此外，该字符串还可以包含转义序列`\\`、`\a`、`\b`、`\f`、`\n`、`\r`、`\t`、`\v`。

`date_fmt` 定义相应日期和时间表示形式，与`%C`字段描述符对应。操作数由一个字符串组成，可以包含字符和字段描述符的任意组合。此外，该字符串还可以包含转义序列`\\`、`\a`、`\b`、`\f`、`\n`、`\r`、`\t`、`\v`。

`d_fmt` 定义相应日期表示形式，与`%x`字段描述符对应。操作数由一个字符串组成，可以包含字符和字段描述符的任意组合。此外，该字符串还可以包含转义序列`\\`、`\a`、`\b`、`\f`、`\n`、`\r`、`\t`、`\v`。

`t_fmt` 定义相应时间表示形式，与`%X`字段描述符对应。操作数由一个字符串组成，可以包含字符和字段描述符的任意组合。此外，该字符串还可以包含转义序列`\\`、`\a`、`\b`、`\f`、`\n`、`\r`、`\t`、`\v`。

`am_pm` 定义**上午**和**下午**字符串的相应表示形式，与`%p`字段描述符对应。操作数由两个分号分隔的字符串组成，每一个都括在双引号中。第一个字符串表示**上午**指定方式，最后一个字符串表示**下午**指定方式。

`t_fmt_ampm` 以`am_pm`的12小时时钟格式定义相应时间表示形式，与`%r`字段描述符对应。操作数由一个字符串组成，可以包含字符和字段描述符的任意组合。如果该字符串为空，则语言环境中不支持12小时格式。

`era` 定义语言环境中的每个时代如何计数和显示年份。操作数由分号分隔的字符串组成。每个字符串都是具有下列格式的时代说明段：

direction:*offset*:*start_date*:*end_date*:*era_name*:*era_format*

其中的各部分遵循下面的定义。可以有多个时代说明段，具体数量取决于用户需要描述多少个不同的时代。

时代的开始时间可能不是最早的时间点，例如公元前基督教时代的开始时间为公元1年的1月1日的前一天，由此延续到更早的时间。

direction + 或 – 字符。+ 字符指示接近*start_date*的年份的数字小于接近*end_date*的年份的数字。– 字符指示接近*start_date*的年份的数字大于接近*end_date*的年份的数字。

offset 时代中最接近 *start_date* 的年份的数字，与 `%Eg` 和 `%Ey` 字段描述符对应。

start_date *yyyy*/*mm*/`dd` 格式的日期，其中 *yyyy*、*mm* 和 `dd` 分别是时代开始的年份、月份和日期数字。公元 1 年之前的年份以负数表示。

end_date 与 *start_date* 格式相同的时代结束日期，或者为两个特殊值 -* 或 +* 之一。值 -* 指示结束日期为时间的起点。值 +* 指示结束日期为时间的终点。

era_name 表示时代名称的字符串，与 `%EC` 字段描述符对应。

era_format 用于设置时代中年份的格式的字符串，与 `%EG` 和 `%EY` 字段描述符对应。

`era_d_fmt` 定义采用替代时代表示法的日期格式，与 `%Ex` 字段描述符对应。

`era_t_fmt` 定义语言环境的相应替代时间格式，与 `%EX` 字段描述符对应。

`era_d_t_fmt` 定义语言环境的相应替代日期和时间格式，与 `%Ec` 字段描述符对应。

`alt_digits` 定义数字的替代符号，与 `%O` 字段描述符修饰符对应。操作数由分号分隔的字符串组成，每一个都括在双引号中。第一个字符串是与零对应的替代符号，第二个字符串是与一对应的符号，依此类推。最多可以指定 100 个替代符号字符串。`%O` 修饰符指示将使用与通过字段描述符指定的值对应的字符串代替该值。

LC_TIME *C* 语言访问 可以访问以下信息。这些信息对应于 `<langinfo.h>` 中定义的常量并用作 `nl_langinfo`(3C) 函数的参数。

`ABDAY_`*x* 缩写的工作日名称（例如 Sun），其中 *x* 是 1 到 7 之间的一个数字。

`DAY_`*x* 完整的工作日名称（例如 Sunday），其中 *x* 是 1 到 7 之间的一个数字。

`ABMON_`*x* 缩写的月份名称（例如 Jan），其中 *x* 是 1 到 12 之间的一个数字。

`MON_`*x* 完整的月份名称（例如 January），其中 *x* 是 1 到 12 之间的一个数字。

`D_T_FMT` 相应的日期和时间表示形式。

`D_FMT` 相应的日期表示形式。

`T_FMT` 相应的时间表示形式。

`AM_STR` 相应的上午后缀。

`PM_STR` 相应的下午后缀。

`T_FMT_AMPM` `AM_STR` 和 `PM_STR.` 的 12 小时时钟格式的相应时间表示形式。

`ERA`

时代说明段，用于描述语言环境中的每个时代如何计数和显示年份。每个时代说明段都具有下列格式：

direction:*offset*:*start_date*:*end_date*:*era_name*:*era_format*

其中的各部分遵循下面的定义。可以有多个时代说明段，具体数量取决于用户需要描述多少个不同的时代。时代说明段使用分号分隔。

时代的开始时间可能不是最早的时间点，例如公元前基督教时代的开始时间为公元 1 年的 1 月 1 日的前一天，由此延续到更早的时间。

direction

+ 或 – 字符。+ 字符指示接近 *start_date* 的年份的数字小于接近 *end_date* 的年份的数字。– 字符指示接近 *start_date* 的年份的数字大于接近 *end_date* 的年份的数字。

offset

时代中最接近 start_date 的年份的数字。

start_date

yyyy/*mm*//*dd* 格式的日期，其中 *yyyy*、*mm* 和 dd 分别是时代开始的年份、月份和日期数字。公元 1 年之前的年份以负数表示。

end_date

与 *start_date* 格式相同的时代结束日期，或者为两个特殊值 –* 或 +* 之一。值 –* 指示结束日期为时间的起点。值 +* 指示结束日期为时间的终点。

era_name

时代名称，与 `%EC` 转换规范对应。

era_format

时代中年份的格式，与 `%EY` 和 `%EY` 转换规范对应。

`ERA_D_FMT`

年代日期格式。

`ERA_T_FMT`

语言环境的相应替代时间格式，与 `%EX` 字段描述符对应。

`ERA_D_T_FMT`

语言环境的相应替代日期和时间格式，与 `%Ec` 字段描述符对应。

`ALT_DIGITS`

数字的替代符号，与 `%O` 转换规范修饰符对应。该值由分号分隔的符号组成。第一个符号是对应于零的替代符号，第二个符号是对应于一的符号，依此类推。最多可以指定 100 个替代符号。下表显示了上述项与以下函数使用的转换说明符之间的对应关系：`date`(1)、`strftime`(3C)、`wcsftime`(3C) 和 `strptime`(3C)。

localedef	**langinfo**	**转换**
关键字	**常量**	**说明符**
abday	ABDAY_*x*	%a
day	DAY_*x*	%A
abmon	ABMON_*x*	%b

localedef	langinfo	转换
关键字	常量	说明符
mon	MON	%B
d_t_fmt	D_T_FMT	%c
date_fmt	DATE_FMT	%C
d_fmt	D_FMT	%x
t_fmt	T_FMT	%X
am_pm	AM_STR	%p
am_pm	PM_STR	%p
t_fmt_ampm	T_FMT_AMPM	%r
era	ERA	%EC, %Eg, %EG, %Ey, %EY
era_d_fmt	ERA_D_FMT	%Ex
era_t_fmt	ERA_T_FMT	%EX
era_d_t_fmt	ERA_D_T_FMT	%Ec
alt_digits	ALT_DIGITS	%O

LC_TIME 常规信息

尽管 POSIX 语言环境中的某些字段描述符（例如月份的名称）的第一个字母显示为大写，但在其他语言环境中不需要为大写。如果这些输出要在句子的开头使用，则使用这些字段的程序可能需要调整大写。

LC_TIME 中对 abday、day、mon 和 abmon 的说明隐含了采用公历风格的日历（一周包含 7 天、一年包含 12 个月、闰年等等）。本文档集不介绍其他日历类型的时间字符串的格式设置。

如“语言环境定义”中的 date 和 strftime(3C) 所指定的那样，与可选关键字对应的字段描述符由一个修饰符和后跟的传统字段描述符（例如 %Ex）组成。如果实现不支持可选关键字，或者没有为当前语言环境指定可选关键字，则这些字段描述符将视为传统字段描述符。例如，假定以下关键字：

```
alt_digits    "0th" ; "1st" ; "2nd" ; "3rd" ; "4th" ; "5th" ; \
"6th" ; "7th" ; "8th" ; "9th" ; "10th">
d_fmt    "The %Od day of %B in %Y"
```

在 7/4/1776，%x 字段描述符将生成 “The 4th day of July in 1776”，而 7/14/1789 的结果为 “The 14 day of July in 1789”。上述示例仅用于描述目的。%O 修饰符主要用于提供 date 格式的日文汉字数字或印地语数字。

LC_MESSAGES

LC_MESSAGES 类别定义肯定性和否定性响应的格式和值。

以下关键字会识别为语言环境定义文件的一部分。nl_langinfo(3C) 函数接受前四个关键字的大写版本。

yesexpr
操作数由一个扩展的正则表达式组成（请参见 regex(5)），该表达式描述了需要肯定性或否定性响应的问题的可接受肯定性响应。

noexpr
操作数由一个扩展的正则表达式组成，该表达式描述对需要肯定性或否定性响应的问题的可接受否定性响应。

yesstr
操作数由一个固定字符串（不是正则表达式）组成，应用程序可以使用该字符串编写一条列出可接受的肯定性响应的消息，例如在提示符中显示的消息。

nostr
操作数由一个固定字符串组成，应用程序可以使用该字符串编写一条列出可接受的否定性响应的消息。POSIX 语言环境的肯定性和否定性响应的格式和值如下所示；该代码列表描述了 localedef 输入，表给出了相同信息，但添加了 nl_langinfo() 常量。

```
LC_MESSAGES
# This is the POSIX locale definition for
# the LC_MESSAGES category.
#
yesexpr "<circumflex><left-square-bracket><y><Y>\
        <right-square-bracket>"
#
noexpr  "<circumflex><left-square-bracket><n><N>\
        <right-square-bracket>"
#
yesstr    "yes"
nostr    "no"
END LC_MESSAGES
```

localedef 关键字	langinfo 常量	POSIX 语言环境值
yesexpr	YESEXPR	"^[yY]"
noexpr	NOEXPR	"^[nN]"
yesstr	YESSTR	"yes"
nostr	NOSTR	"no"

在遵循 SUSv3 标准的应用程序中，未提供有关 yesstr 和 nostr 的信息。

另请参见

date(1)、locale(1)、localedef(1)、sort(1)、tr(1)、uniq(1)、localeconv(3C)、nl_langinfo(

引用名

locale_alias – 语言环境名称别名及其相应的规范语言环境名称

描述

系统中接受和支持两组语言环境名称别名：

1. 语言环境选择（如 setlocale(3C) 中所指定）和消息对象或消息目录处理（如 gettext(1)、catopen(3C) 和 gettext(3C) 中所指定）期间，接受并映射到相应的规范语言环境名称的语言环境名称别名（如果有）。

   在映射过程中，语言环境名称别名的代码集名称部分会通过以下方式标准化：仅提取字母数字字符并转换为小写，以便提高可能的映射的成功率。例如，通过这种标准化，AR_AA.UTF-8 之类的语言环境名称别名会标准化为 AR_AA.utf8 作为搜索域值，然后与内部映射表中一组预先标准化的语言环境别名进行实际比较，以找出规范语言环境名称。

   依计算机顺序的受支持语言环境名称别名如下所示。由于系统中目前尚未安装语言环境或者 Solaris 尚不支持语言环境，当前系统也可能没有相应的规范语言环境。

```
Locale Name Alias        Canonical Locale Name
-----------------        ---------------------
AR_AE                    ar_AE.UTF-8
AR_AE.UTF-8              ar_AE.UTF-8
AR_BH                    ar_BH.UTF-8
AR_BH.UTF-8              ar_BH.UTF-8
AR_DZ                    ar_DZ.UTF-8
AR_DZ.UTF-8              ar_DZ.UTF-8
AR_EG                    ar_EG.UTF-8
AR_EG.UTF-8              ar_EG.UTF-8
AR_JO                    ar_JO.UTF-8
AR_JO.UTF-8              ar_JO.UTF-8
AR_KW                    ar_KW.UTF-8
AR_KW.UTF-8              ar_KW.UTF-8
AR_MA                    ar_MA.UTF-8
AR_MA.UTF-8              ar_MA.UTF-8
AR_OM                    ar_OM.UTF-8
AR_OM.UTF-8              ar_OM.UTF-8
AR_QA                    ar_QA.UTF-8
AR_QA.UTF-8              ar_QA.UTF-8
AR_SA                    ar_SA.UTF-8
AR_SA.UTF-8              ar_SA.UTF-8
AR_TN                    ar_TN.UTF-8
AR_TN.UTF-8              ar_TN.UTF-8
AR_YE                    ar_YE.UTF-8
AR_YE.UTF-8              ar_YE.UTF-8
AS_IN                    as_IN.UTF-8
AS_IN.UTF-8              as_IN.UTF-8
AZ_AZ                    az_AZ.UTF-8
AZ_AZ.UTF-8              az_AZ.UTF-8
BE_BY                    be_BY.UTF-8
BE_BY.UTF-8              be_BY.UTF-8
```

BG_BG	bg_BG.UTF-8
BG_BG.UTF-8	bg_BG.UTF-8
BN_IN	bn_IN.UTF-8
BN_IN.UTF-8	bn_IN.UTF-8
CA_ES	ca_ES.UTF-8
CA_ES.UTF-8	ca_ES.UTF-8
CA_ES.UTF-8@euro	ca_ES.UTF-8
CA_ES@euro	ca_ES.UTF-8
CS_CZ	cs_CZ.UTF-8
CS_CZ.UTF-8	cs_CZ.UTF-8
DA_DK	da_DK.UTF-8
DA_DK.UTF-8	da_DK.UTF-8
DE_AT	de_AT.UTF-8
DE_AT.UTF-8	de_AT.UTF-8
DE_AT.UTF-8@euro	de_AT.UTF-8
DE_AT@euro	de_AT.UTF-8
DE_CH	de_CH.UTF-8
DE_CH.UTF-8	de_CH.UTF-8
DE_DE	de_DE.UTF-8
DE_DE.UTF-8	de_DE.UTF-8
DE_DE.UTF-8@euro	de_DE.UTF-8
DE_DE@euro	de_DE.UTF-8
DE_LU	de_LU.UTF-8
DE_LU.UTF-8	de_LU.UTF-8
DE_LU.UTF-8@euro	de_LU.UTF-8
DE_LU@euro	de_LU.UTF-8
EL_GR	el_GR.UTF-8
EL_GR.UTF-8	el_GR.UTF-8
EN_AU	en_AU.UTF-8
EN_AU.UTF-8	en_AU.UTF-8
EN_CA	en_CA.UTF-8
EN_CA.UTF-8	en_CA.UTF-8
EN_GB	en_GB.UTF-8
EN_GB.UTF-8	en_GB.UTF-8
EN_GB.UTF-8@euro	en_GB.UTF-8
EN_GB@euro	en_GB.UTF-8
EN_HK	en_HK.UTF-8
EN_HK.UTF-8	en_HK.UTF-8
EN_IE	en_IE.UTF-8
EN_IE.UTF-8	en_IE.UTF-8
EN_IE.UTF-8@euro	en_IE.UTF-8
EN_IE@euro	en_IE.UTF-8
EN_IN	en_IN.UTF-8
EN_IN.UTF-8	en_IN.UTF-8
EN_NZ	en_NZ.UTF-8
EN_NZ.UTF-8	en_NZ.UTF-8
EN_PH	en_PH.UTF-8
EN_PH.UTF-8	en_PH.UTF-8

EN_SG	en_SG.UTF-8
EN_SG.UTF-8	en_SG.UTF-8
EN_US	en_US.UTF-8
EN_US.UTF-8	en_US.UTF-8
ES_AR	es_AR.UTF-8
ES_AR.UTF-8	es_AR.UTF-8
ES_BO	es_BO.UTF-8
ES_BO.UTF-8	es_BO.UTF-8
ES_CL	es_CL.UTF-8
ES_CL.UTF-8	es_CL.UTF-8
ES_CO	es_CO.UTF-8
ES_CO.UTF-8	es_CO.UTF-8
ES_CR	es_CR.UTF-8
ES_CR.UTF-8	es_CR.UTF-8
ES_DO	es_DO.UTF-8
ES_DO.UTF-8	es_DO.UTF-8
ES_EC	es_EC.UTF-8
ES_EC.UTF-8	es_EC.UTF-8
ES_ES	es_ES.UTF-8
ES_ES.UTF-8	es_ES.UTF-8
ES_ES.UTF-8@euro	es_ES.UTF-8
ES_ES@euro	es_ES.UTF-8
ES_GT	es_GT.UTF-8
ES_GT.UTF-8	es_GT.UTF-8
ES_HN	es_HN.UTF-8
ES_HN.UTF-8	es_HN.UTF-8
ES_MX	es_MX.UTF-8
ES_MX.UTF-8	es_MX.UTF-8
ES_NI	es_NI.UTF-8
ES_NI.UTF-8	es_NI.UTF-8
ES_PA	es_PA.UTF-8
ES_PA.UTF-8	es_PA.UTF-8
ES_PE	es_PE.UTF-8
ES_PE.UTF-8	es_PE.UTF-8
ES_PR	es_PR.UTF-8
ES_PR.UTF-8	es_PR.UTF-8
ES_PY	es_PY.UTF-8
ES_PY.UTF-8	es_PY.UTF-8
ES_SV	es_SV.UTF-8
ES_SV.UTF-8	es_SV.UTF-8
ES_US	es_US.UTF-8
ES_US.UTF-8	es_US.UTF-8
ES_UY	es_UY.UTF-8
ES_UY.UTF-8	es_UY.UTF-8
ES_VE	es_VE.UTF-8
ES_VE.UTF-8	es_VE.UTF-8
ET_EE	et_EE.UTF-8
ET_EE.UTF-8	et_EE.UTF-8

FI_FI	fi_FI.UTF-8
FI_FI.UTF-8	fi_FI.UTF-8
FI_FI.UTF-8@euro	fi_FI.UTF-8
FI_FI@euro	fi_FI.UTF-8
FR_BE	fr_BE.UTF-8
FR_BE.UTF-8	fr_BE.UTF-8
FR_BE.UTF-8@euro	fr_BE.UTF-8
FR_BE@euro	fr_BE.UTF-8
FR_CA	fr_CA.UTF-8
FR_CA.UTF-8	fr_CA.UTF-8
FR_CH	fr_CH.UTF-8
FR_CH.UTF-8	fr_CH.UTF-8
FR_FR	fr_FR.UTF-8
FR_FR.UTF-8	fr_FR.UTF-8
FR_FR.UTF-8@euro	fr_FR.UTF-8
FR_FR@euro	fr_FR.UTF-8
FR_LU	fr_LU.UTF-8
FR_LU.UTF-8	fr_LU.UTF-8
FR_LU.UTF-8@euro	fr_LU.UTF-8
FR_LU@euro	fr_LU.UTF-8
GU_IN	gu_IN.UTF-8
GU_IN.UTF-8	gu_IN.UTF-8
HE_IL	he_IL.UTF-8
HE_IL.UTF-8	he_IL.UTF-8
HI_IN	hi_IN.UTF-8
HI_IN.UTF-8	hi_IN.UTF-8
HR_HR	hr_HR.UTF-8
HR_HR.UTF-8	hr_HR.UTF-8
HU_HU	hu_HU.UTF-8
HU_HU.UTF-8	hu_HU.UTF-8
ID_ID	id_ID.UTF-8
ID_ID.UTF-8	id_ID.UTF-8
IS_IS	is_IS.UTF-8
IS_IS.UTF-8	is_IS.UTF-8
IT_CH	it_CH.UTF-8
IT_CH.UTF-8	it_CH.UTF-8
IT_IT	it_IT.UTF-8
IT_IT.UTF-8	it_IT.UTF-8
IT_IT.UTF-8@euro	it_IT.UTF-8
IT_IT@euro	it_IT.UTF-8
JA_JP	ja_JP.UTF-8
JA_JP.UTF-8	ja_JP.UTF-8
KK_KZ	kk_KZ.UTF-8
KK_KZ.UTF-8	kk_KZ.UTF-8
KN_IN	kn_IN.UTF-8
KN_IN.UTF-8	kn_IN.UTF-8
KO_KR	ko_KR.UTF-8
KO_KR.UTF-8	ko_KR.UTF-8

LT_LT	lt_LT.UTF-8
LT_LT.UTF-8	lt_LT.UTF-8
LV_LV	lv_LV.UTF-8
LV_LV.UTF-8	lv_LV.UTF-8
MK_MK	mk_MK.UTF-8
MK_MK.UTF-8	mk_MK.UTF-8
ML_IN	ml_IN.UTF-8
ML_IN.UTF-8	ml_IN.UTF-8
MR_IN	mr_IN.UTF-8
MR_IN.UTF-8	mr_IN.UTF-8
MS_MY	ms_MY.UTF-8
MS_MY.UTF-8	ms_MY.UTF-8
MT_MT	mt_MT.UTF-8
MT_MT.UTF-8	mt_MT.UTF-8
NL_BE	nl_BE.UTF-8
NL_BE.UTF-8	nl_BE.UTF-8
NL_BE.UTF-8@euro	nl_BE.UTF-8
NL_BE@euro	nl_BE.UTF-8
NL_NL	nl_NL.UTF-8
NL_NL.UTF-8	nl_NL.UTF-8
NL_NL.UTF-8@euro	nl_NL.UTF-8
NL_NL@euro	nl_NL.UTF-8
OR_IN	or_IN.UTF-8
OR_IN.UTF-8	or_IN.UTF-8
PA_IN	pa_IN.UTF-8
PA_IN.UTF-8	pa_IN.UTF-8
PL_PL	pl_PL.UTF-8
PL_PL.UTF-8	pl_PL.UTF-8
PT_BR	pt_BR.UTF-8
PT_BR.UTF-8	pt_BR.UTF-8
PT_PT	pt_PT.UTF-8
PT_PT.UTF-8	pt_PT.UTF-8
PT_PT.UTF-8@euro	pt_PT.UTF-8
PT_PT@euro	pt_PT.UTF-8
RO_RO	ro_RO.UTF-8
RO_RO.UTF-8	ro_RO.UTF-8
RU_RU	ru_RU.UTF-8
RU_RU.UTF-8	ru_RU.UTF-8
SK_SK	sk_SK.UTF-8
SK_SK.UTF-8	sk_SK.UTF-8
SL_SI	sl_SI.UTF-8
SL_SI.UTF-8	sl_SI.UTF-8
SQ_AL	sq_AL.UTF-8
SQ_AL.UTF-8	sq_AL.UTF-8
SV_SE	sv_SE.UTF-8
SV_SE.UTF-8	sv_SE.UTF-8
TA_IN	ta_IN.UTF-8
TA_IN.UTF-8	ta_IN.UTF-8

TE_IN	te_IN.UTF-8
TE_IN.UTF-8	te_IN.UTF-8
TH_TH	th_TH.UTF-8
TH_TH.UTF-8	th_TH.UTF-8
TR_TR	tr_TR.UTF-8
TR_TR.UTF-8	tr_TR.UTF-8
UK_UA	uk_UA.UTF-8
UK_UA.UTF-8	uk_UA.UTF-8
VI_VN	vi_VN.UTF-8
VI_VN.UTF-8	vi_VN.UTF-8
ZH_CN	zh_CN.UTF-8
ZH_CN.UTF-8	zh_CN.UTF-8
ZH_HK	zh_HK.UTF-8
ZH_HK.UTF-8	zh_HK.UTF-8
ZH_SG	zh_SG.UTF-8
ZH_SG.UTF-8	zh_SG.UTF-8
ZH_TW	zh_TW.UTF-8
ZH_TW.UTF-8	zh_TW.UTF-8
Zh_CN	zh_CN.GB18030
Zh_CN.GB18030	zh_CN.GB18030
Zh_TW	zh_TW.BIG5
Zh_TW.big5	zh_TW.BIG5
af_ZA.utf8	af_ZA.UTF-8
ar	ar_EG.ISO8859-6
ar_AE.utf8	ar_AE.UTF-8
ar_BH.utf8	ar_BH.UTF-8
ar_DZ.utf8	ar_DZ.UTF-8
ar_EG	ar_EG.ISO8859-6
ar_EG.iso88596	ar_EG.ISO8859-6
ar_EG.utf8	ar_EG.UTF-8
ar_IQ.utf8	ar_IQ.UTF-8
ar_JO.utf8	ar_JO.UTF-8
ar_KW.utf8	ar_KW.UTF-8
ar_LY.utf8	ar_LY.UTF-8
ar_MA.utf8	ar_MA.UTF-8
ar_OM.utf8	ar_OM.UTF-8
ar_QA.utf8	ar_QA.UTF-8
ar_SA.utf8	ar_SA.UTF-8
ar_TN.utf8	ar_TN.UTF-8
ar_YE.utf8	ar_YE.UTF-8
as_IN.utf8	as_IN.UTF-8
az_AZ.utf8	az_AZ.UTF-8
be_BY.utf8	be_BY.UTF-8
bg_BG	bg_BG.ISO8859-5
bg_BG.utf8	bg_BG.UTF-8
bn_IN	bn_IN.UTF-8
bn_IN.utf8	bn_IN.UTF-8
bs_BA	bs_BA.ISO8859-2

bs_BA.iso88592	bs_BA.ISO8859-2
bs_BA.utf8	bs_BA.UTF-8
ca	ca_ES.ISO8859-1
ca_ES	ca_ES.ISO8859-1
ca_ES.8859-15	ca_ES.ISO8859-15
ca_ES.8859-15@euro	ca_ES.ISO8859-15
ca_ES.ISO8859-15@euro	ca_ES.ISO8859-15
ca_ES.iso88591	ca_ES.ISO8859-1
ca_ES.utf8	ca_ES.UTF-8
ca_ES@euro	ca_ES.ISO8859-15
cs	cs_CZ.ISO8859-2
cs_CZ	cs_CZ.ISO8859-2
cs_CZ.iso88592	cs_CZ.ISO8859-2
cs_CZ.utf8	cs_CZ.UTF-8
da	da_DK.ISO8859-1
da.ISO8859-15	da_DK.ISO8859-15
da_DK	da_DK.ISO8859-1
da_DK.8859-15	da_DK.ISO8859-15
da_DK.iso88591	da_DK.ISO8859-1
da_DK.iso885915	da_DK.ISO8859-15
da_DK.iso885915@euro	da_DK.ISO8859-15
da_DK.utf8	da_DK.UTF-8
de	de_DE.ISO8859-1
de.ISO8859-15	de_DE.ISO8859-15
de.UTF-8	de_DE.UTF-8
de_AT	de_AT.ISO8859-1
de_AT.8859-15	de_AT.ISO8859-15
de_AT.8859-15@euro	de_AT.ISO8859-15
de_AT.ISO8859-15@euro	de_AT.ISO8859-15
de_AT.iso88591	de_AT.ISO8859-1
de_AT.utf8	de_AT.UTF-8
de_AT@euro	de_AT.ISO8859-15
de_BE.utf8	de_BE.UTF-8
de_CH	de_CH.ISO8859-1
de_CH.iso88591	de_CH.ISO8859-1
de_CH.utf8	de_CH.UTF-8
de_DE	de_DE.ISO8859-1
de_DE.8859-15	de_DE.ISO8859-15
de_DE.8859-15@euro	de_DE.ISO8859-15
de_DE.ISO8859-15@euro	de_DE.ISO8859-15
de_DE.UTF-8@euro	de_DE.UTF-8
de_DE.iso88591	de_DE.ISO8859-1
de_DE.utf8	de_DE.UTF-8
de_DE@euro	de_DE.ISO8859-15
de_LU.utf8	de_LU.UTF-8
el	el_GR.ISO8859-7
el.UTF-8	el_CY.UTF-8
el.sun_eu_greek	el_GR.ISO8859-7

el_CY.utf8	el_CY.UTF-8
el_GR	el_GR.ISO8859-7
el_GR.ISO8859-7@euro	el_GR.ISO8859-7
el_GR.iso88597	el_GR.ISO8859-7
el_GR.utf8	el_GR.UTF-8
en_AU	en_AU.ISO8859-1
en_AU.iso88591	en_AU.ISO8859-1
en_AU.utf8	en_AU.UTF-8
en_BW.utf8	en_BW.UTF-8
en_CA	en_CA.ISO8859-1
en_CA.iso88591	en_CA.ISO8859-1
en_CA.utf8	en_CA.UTF-8
en_GB	en_GB.ISO8859-1
en_GB.8859-15@euro	en_GB.ISO8859-15
en_GB.IBM-1252	en_GB.ANSI1252
en_GB.iso88591	en_GB.ISO8859-1
en_GB.iso885915	en_GB.ISO8859-15
en_GB.iso885915@euro	en_GB.ISO8859-15
en_GB.utf8	en_GB.UTF-8
en_HK.utf8	en_HK.UTF-8
en_IE	en_IE.ISO8859-1
en_IE.8859-15	en_IE.ISO8859-15
en_IE.8859-15@euro	en_IE.ISO8859-15
en_IE.ISO8859-15@euro	en_IE.ISO8859-15
en_IE.iso88591	en_IE.ISO8859-1
en_IE.utf8	en_IE.UTF-8
en_IE@euro	en_IE.ISO8859-15
en_IN.utf8	en_IN.UTF-8
en_NZ	en_NZ.ISO8859-1
en_NZ.iso88591	en_NZ.ISO8859-1
en_NZ.utf8	en_NZ.UTF-8
en_PH.utf8	en_PH.UTF-8
en_SG.utf8	en_SG.UTF-8
en_US	en_US.ISO8859-1
en_US.8859-15	en_US.ISO8859-15
en_US.iso88591	en_US.ISO8859-1
en_US.iso885915	en_US.ISO8859-15
en_US.utf8	en_US.UTF-8
en_ZW.utf8	en_ZW.UTF-8
es	es_ES.ISO8859-1
es.ISO8859-15	es_ES.ISO8859-15
es.UTF-8	es_ES.UTF-8
es_AR	es_AR.ISO8859-1
es_AR.iso88591	es_AR.ISO8859-1
es_AR.utf8	es_AR.UTF-8
es_BO	es_BO.ISO8859-1
es_BO.iso88591	es_BO.ISO8859-1
es_BO.utf8	es_BO.UTF-8

es_CL	es_CL.ISO8859-1
es_CL.iso88591	es_CL.ISO8859-1
es_CL.utf8	es_CL.UTF-8
es_CO	es_CO.ISO8859-1
es_CO.iso88591	es_CO.ISO8859-1
es_CO.utf8	es_CO.UTF-8
es_CR	es_CR.ISO8859-1
es_CR.iso88591	es_CR.ISO8859-1
es_CR.utf8	es_CR.UTF-8
es_DO.utf8	es_DO.UTF-8
es_EC	es_EC.ISO8859-1
es_EC.iso88591	es_EC.ISO8859-1
es_EC.utf8	es_EC.UTF-8
es_ES	es_ES.ISO8859-1
es_ES.8859-15	es_ES.ISO8859-15
es_ES.8859-15@euro	es_ES.ISO8859-15
es_ES.ISO8859-15@euro	es_ES.ISO8859-15
es_ES.UTF-8@euro	es_ES.UTF-8
es_ES.iso88591	es_ES.ISO8859-1
es_ES.utf8	es_ES.UTF-8
es_ES@euro	es_ES.ISO8859-15
es_GT	es_GT.ISO8859-1
es_GT.iso88591	es_GT.ISO8859-1
es_GT.utf8	es_GT.UTF-8
es_HN.utf8	es_HN.UTF-8
es_MX	es_MX.ISO8859-1
es_MX.iso88591	es_MX.ISO8859-1
es_NI.utf8	es_NI.UTF-8
es_PA	es_PA.ISO8859-1
es_PA.iso88591	es_PA.ISO8859-1
es_PA.utf8	es_PA.UTF-8
es_PE	es_PE.ISO8859-1
es_PE.iso88591	es_PE.ISO8859-1
es_PE.utf8	es_PE.UTF-8
es_PR.utf8	es_PR.UTF-8
es_PY	es_PY.ISO8859-1
es_PY.iso88591	es_PY.ISO8859-1
es_PY.utf8	es_PY.UTF-8
es_SV	es_SV.ISO8859-1
es_SV.iso88591	es_SV.ISO8859-1
es_SV.utf8	es_SV.UTF-8
es_US.utf8	es_US.UTF-8
es_UY	es_UY.ISO8859-1
es_UY.8859-15	es_UY.ISO8859-15
es_UY.iso88591	es_UY.ISO8859-1
es_UY.utf8	es_UY.UTF-8
es_VE	es_VE.ISO8859-1
es_VE.iso88591	es_VE.ISO8859-1

es_VE.utf8	es_VE.UTF-8
et	et_EE.ISO8859-15
et_EE	et_EE.ISO8859-15
et_EE.iso885915	et_EE.ISO8859-15
et_EE.utf8	et_EE.UTF-8
fi	fi_FI.ISO8859-1
fi.ISO8859-15	fi_FI.ISO8859-15
fi_FI	fi_FI.ISO8859-1
fi_FI.8859-15	fi_FI.ISO8859-15
fi_FI.8859-15@euro	fi_FI.ISO8859-15
fi_FI.ISO8859-15@euro	fi_FI.ISO8859-15
fi_FI.iso88591	fi_FI.ISO8859-1
fi_FI.utf8	fi_FI.UTF-8
fi_FI@euro	fi_FI.ISO8859-15
fr	fr_FR.ISO8859-1
fr.ISO8859-15	fr_FR.ISO8859-15
fr.UTF-8	fr_FR.UTF-8
fr_BE	fr_BE.ISO8859-1
fr_BE.8859-15	fr_BE.ISO8859-15
fr_BE.8859-15@euro	fr_BE.ISO8859-15
fr_BE.ISO8859-15@euro	fr_BE.ISO8859-15
fr_BE.UTF-8@euro	fr_BE.UTF-8
fr_BE.iso88591	fr_BE.ISO8859-1
fr_BE.utf8	fr_BE.UTF-8
fr_BE@euro	fr_BE.ISO8859-15
fr_CA	fr_CA.ISO8859-1
fr_CA.iso88591	fr_CA.ISO8859-1
fr_CA.utf8	fr_CA.UTF-8
fr_CH	fr_CH.ISO8859-1
fr_CH.iso88591	fr_CH.ISO8859-1
fr_CH.utf8	fr_CH.UTF-8
fr_FR	fr_FR.ISO8859-1
fr_FR.8859-15	fr_FR.ISO8859-15
fr_FR.8859-15@euro	fr_FR.ISO8859-15
fr_FR.ISO8859-15@euro	fr_FR.ISO8859-15
fr_FR.UTF-8@euro	fr_FR.UTF-8
fr_FR.iso88591	fr_FR.ISO8859-1
fr_FR.utf8	fr_FR.UTF-8
fr_FR@euro	fr_FR.ISO8859-15
fr_LU.utf8	fr_LU.UTF-8
fr_LU@euro	fr_LU.ISO8859-15
gu_IN	gu_IN.UTF-8
gu_IN.utf8	gu_IN.UTF-8
he	he_IL.ISO8859-8
he_IL	he_IL.ISO8859-8
he_IL.iso88598	he_IL.ISO8859-8
he_IL.utf8	he_IL.UTF-8
hi_IN	hi_IN.UTF-8

```
hi_IN.utf8               hi_IN.UTF-8
hr_HR                    hr_HR.ISO8859-2
hr_HR.iso88592           hr_HR.ISO8859-2
hr_HR.utf8               hr_HR.UTF-8
hu                       hu_HU.ISO8859-2
hu_HU                    hu_HU.ISO8859-2
hu_HU.iso88592           hu_HU.ISO8859-2
hu_HU.utf8               hu_HU.UTF-8
hy_AM                    hy_AM.UTF-8
hy_AM.utf8               hy_AM.UTF-8
id_ID.utf8               id_ID.UTF-8
is_IS                    is_IS.ISO8859-1
is_IS.iso88591           is_IS.ISO8859-1
is_IS.utf8               is_IS.UTF-8
it                       it_IT.ISO8859-1
it.ISO8859-15            it_IT.ISO8859-15
it.UTF-8                 it_IT.UTF-8
it_CH.utf8               it_CH.UTF-8
it_IT                    it_IT.ISO8859-1
it_IT.8859-15            it_IT.ISO8859-15
it_IT.8859-15@euro       it_IT.ISO8859-15
it_IT.ISO8859-15@euro    it_IT.ISO8859-15
it_IT.UTF-8@euro         it_IT.UTF-8
it_IT.iso88591           it_IT.ISO8859-1
it_IT.utf8               it_IT.UTF-8
it_IT@euro               it_IT.ISO8859-15
ja                       ja_JP.eucJP
ja_JP                    ja_JP.eucJP
ja_JP.IBM-eucJP          ja_JP.eucJP
ja_JP.eucjp              ja_JP.eucJP
ja_JP.utf8               ja_JP.UTF-8
ka_GE.utf8               ka_GE.UTF-8
kk_KZ.utf8               kk_KZ.UTF-8
kn_IN                    kn_IN.UTF-8
kn_IN.utf8               kn_IN.UTF-8
ko                       ko_KR.EUC
ko.UTF-8                 ko_KR.UTF-8
ko_KR                    ko_KR.EUC
ko_KR.IBM-eucKR          ko_KR.EUC
ko_KR.euckr              ko_KR.EUC
ko_KR.utf8               ko_KR.UTF-8
ku_TR.utf8               ku_TR.UTF-8
ky_KG                    ky_KG.UTF-8
ky_KG.utf8               ky_KG.UTF-8
lt                       lt_LT.ISO8859-13
lt_LT                    lt_LT.ISO8859-13
lt_LT.iso885913          lt_LT.ISO8859-13
lt_LT.utf8               lt_LT.UTF-8
```

lv	lv_LV.ISO8859-13
lv_LV	lv_LV.ISO8859-13
lv_LV.iso885913	lv_LV.ISO8859-13
lv_LV.utf8	lv_LV.UTF-8
mk_MK	mk_MK.ISO8859-5
mk_MK.iso88595	mk_MK.ISO8859-5
mk_MK.utf8	mk_MK.UTF-8
ml_IN	ml_IN.UTF-8
ml_IN.utf8	ml_IN.UTF-8
mr_IN	mr_IN.UTF-8
mr_IN.utf8	mr_IN.UTF-8
ms_MY.utf8	ms_MY.UTF-8
mt_MT.utf8	mt_MT.UTF-8
nb_NO	nb_NO.ISO8859-1
nb_NO.iso88591	nb_NO.ISO8859-1
nb_NO.utf8	nb_NO.UTF-8
nl	nl_NL.ISO8859-1
nl.ISO8859-15	nl_NL.ISO8859-15
nl_BE	nl_BE.ISO8859-1
nl_BE.8859-15	nl_BE.ISO8859-15
nl_BE.8859-15@euro	nl_BE.ISO8859-15
nl_BE.ISO8859-15@euro	nl_BE.ISO8859-15
nl_BE.iso88591	nl_BE.ISO8859-1
nl_BE.utf8	nl_BE.UTF-8
nl_BE@euro	nl_BE.ISO8859-15
nl_NL	nl_NL.ISO8859-1
nl_NL.8859-15	nl_NL.ISO8859-15
nl_NL.8859-15@euro	nl_NL.ISO8859-15
nl_NL.ISO8859-15@euro	nl_NL.ISO8859-15
nl_NL.iso88591	nl_NL.ISO8859-1
nl_NL.utf8	nl_NL.UTF-8
nl_NL@euro	nl_NL.ISO8859-15
nn_NO	nn_NO.ISO8859-1
nn_NO.iso88591	nn_NO.ISO8859-1
nn_NO.utf8	nn_NO.UTF-8
no	nb_NO.ISO8859-1
no_NO	nb_NO.ISO8859-1
no_NO.ISO8859-1@bokmal	nb_NO.ISO8859-1
no_NO.ISO8859-1@nynorsk	nn_NO.ISO8859-1
no_NY	nn_NO.ISO8859-1
or_IN	or_IN.UTF-8
or_IN.utf8	or_IN.UTF-8
pa_IN	pa_IN.UTF-8
pa_IN.utf8	pa_IN.UTF-8
pl	pl_PL.ISO8859-2
pl.UTF-8	pl_PL.UTF-8
pl_PL	pl_PL.ISO8859-2
pl_PL.iso88592	pl_PL.ISO8859-2

```
pl_PL.utf8               pl_PL.UTF-8
pt                       pt_PT.ISO8859-1
pt.ISO8859-15            pt_PT.ISO8859-15
pt_BR                    pt_BR.ISO8859-1
pt_BR.iso88591           pt_BR.ISO8859-1
pt_BR.utf8               pt_BR.UTF-8
pt_PT                    pt_PT.ISO8859-1
pt_PT.8859-15            pt_PT.ISO8859-15
pt_PT.8859-15@euro       pt_PT.ISO8859-15
pt_PT.ISO8859-15@euro    pt_PT.ISO8859-15
pt_PT.iso88591           pt_PT.ISO8859-1
pt_PT.utf8               pt_PT.UTF-8
pt_PT@euro               pt_PT.ISO8859-15
ro_RO                    ro_RO.ISO8859-2
ro_RO.iso88592           ro_RO.ISO8859-2
ro_RO.utf8               ro_RO.UTF-8
ru                       ru_RU.ISO8859-5
ru.UTF-8                 ru_RU.UTF-8
ru.koi8-r                ru_RU.KOI8-R
ru_RU                    ru_RU.ISO8859-5
ru_RU.iso88595           ru_RU.ISO8859-5
ru_RU.koi8r              ru_RU.KOI8-R
ru_RU.utf8               ru_RU.UTF-8
ru_UA.utf8               ru_UA.UTF-8
sh                       bs_BA.ISO8859-2
sh_BA                    bs_BA.ISO8859-2
sh_BA.ISO8859-2@bosnia   bs_BA.ISO8859-2
sh_BA.UTF-8              bs_BA.UTF-8
sk_SK                    sk_SK.ISO8859-2
sk_SK.iso88592           sk_SK.ISO8859-2
sk_SK.utf8               sk_SK.UTF-8
sl_SI                    sl_SI.ISO8859-2
sl_SI.iso88592           sl_SI.ISO8859-2
sl_SI.utf8               sl_SI.UTF-8
sq_AL                    sq_AL.ISO8859-2
sq_AL.utf8               sq_AL.UTF-8
sr_CS.UTF-8              sr_RS.UTF-8
sr_CS.iso88595           sr_CS.ISO8859-5
sr_ME                    sr_ME.UTF-8
sr_ME.utf8               sr_ME.UTF-8
sr_RS                    sr_RS.UTF-8
sr_RS.utf8               sr_RS.UTF-8
sv                       sv_SE.ISO8859-1
sv.ISO8859-15            sv_SE.ISO8859-15
sv.UTF-8                 sv_SE.UTF-8
sv_SE                    sv_SE.ISO8859-1
sv_SE.8859-15            sv_SE.ISO8859-15
sv_SE.iso88591           sv_SE.ISO8859-1
```

sv_SE.iso885915	sv_SE.ISO8859-15
sv_SE.iso885915@euro	sv_SE.ISO8859-15
sv_SE.utf8	sv_SE.UTF-8
ta_IN	ta_IN.UTF-8
ta_IN.utf8	ta_IN.UTF-8
te_IN	te_IN.UTF-8
te_IN.utf8	te_IN.UTF-8
th	th_TH.TIS620
th_TH	th_TH.TIS620
th_TH.ISO8859-11	th_TH.TIS620
th_TH.TIS-620	th_TH.TIS620
th_TH.utf8	th_TH.UTF-8
tr	tr_TR.ISO8859-9
tr_TR	tr_TR.ISO8859-9
tr_TR.iso88599	tr_TR.ISO8859-9
tr_TR.utf8	tr_TR.UTF-8
uk_UA.utf8	uk_UA.UTF-8
vi_VN	vi_VN.UTF-8
vi_VN.utf8	vi_VN.UTF-8
zh	zh_CN.EUC
zh.GBK	zh_CN.GBK
zh.UTF-8	zh_CN.UTF-8
zh_CN	zh_CN.EUC
zh_CN.IBM-eucCN	zh_CN.EUC
zh_CN.gb18030	zh_CN.GB18030
zh_CN.gb2312	zh_CN.EUC
zh_CN.gbk	zh_CN.GBK
zh_CN.utf8	zh_CN.UTF-8
zh_HK	zh_HK.BIG5HK
zh_HK.big5hkscs	zh_HK.BIG5HK
zh_HK.utf8	zh_HK.UTF-8
zh_SG.utf8	zh_SG.UTF-8
zh_TW	zh_TW.EUC
zh_TW.IBM-eucTW	zh_TW.EUC
zh_TW.big5	zh_TW.BIG5
zh_TW.euctw	zh_TW.EUC
zh_TW.utf8	zh_TW.UTF-8

2. 会额外检查的已过时 Solaris 语言环境名称，以找到消息对象或消息目录文件（如 gettext(1)、catopen(3C) 和 gettext(3C) 中所指定）

   如果当前语言环境是列出的规范语言环境之一，并且不存在匹配的消息对象或消息目录文件供当前运行中的程序使用当前语言环境名称打开，则消息传送函数会使用下面显示的其他语言环境名称作为别名，来额外检查是否存在要打开的消息对象或消息目录文件：

Canonical Locale Name	Additional Locale Names Checked

ar_EG.ISO8859-6	ar
bg_BG.ISO8859-5	bg_BG
bs_BA.ISO8859-2	sh, sh_BA, sh_BA.ISO8859-2@bosnia
bs_BA.UTF-8	sh_BA.UTF-8
ca_ES.ISO8859-1	ca, ca_ES
ca_ES.ISO8859-15	ca_ES.ISO8859-15@euro
cs_CZ.ISO8859-2	cs, cs_CZ
da_DK.ISO8859-1	da, da_DK
da_DK.ISO8859-15	da.ISO8859-15
de_AT.ISO8859-1	de_AT
de_AT.ISO8859-15	de_AT.ISO8859-15@euro
de_CH.ISO8859-1	de_CH
de_DE.ISO8859-1	de, de_DE
de_DE.ISO8859-15	de.ISO8859-15, de_DE.ISO8859-15@euro
de_DE.UTF-8	de.UTF-8, de_DE.UTF-8@euro
el_CY.UTF-8	el.UTF-8
el_GR.ISO8859-7	el, el.sun_eu_greek, el_GR, el_GR.ISO8859-7@euro
en_AU.ISO8859-1	en_AU
en_CA.ISO8859-1	en_CA
en_GB.ISO8859-1	en_GB
en_IE.ISO8859-1	en_IE
en_IE.ISO8859-15	en_IE.ISO8859-15@euro
en_NZ.ISO8859-1	en_NZ
en_US.ISO8859-1	en_US
es_AR.ISO8859-1	es_AR
es_BO.ISO8859-1	es_BO
es_CL.ISO8859-1	es_CL
es_CO.ISO8859-1	es_CO
es_CR.ISO8859-1	es_CR
es_EC.ISO8859-1	es_EC
es_ES.ISO8859-1	es, es_ES
es_ES.ISO8859-15	es.ISO8859-15, es_ES.ISO8859-15@euro
es_ES.UTF-8	es.UTF-8, es_ES.UTF-8@euro
es_GT.ISO8859-1	es_GT
es_MX.ISO8859-1	es_MX
es_NI.ISO8859-1	es_NI
es_PA.ISO8859-1	es_PA
es_PE.ISO8859-1	es_PE
es_PY.ISO8859-1	es_PY
es_SV.ISO8859-1	es_SV
es_UY.ISO8859-1	es_UY
es_VE.ISO8859-1	es_VE
et_EE.ISO8859-15	et, et_EE
fi_FI.ISO8859-1	fi, fi_FI
fi_FI.ISO8859-15	fi.ISO8859-15, fi_FI.ISO8859-15@euro
fr_BE.ISO8859-1	fr_BE
fr_BE.ISO8859-15	fr_BE.ISO8859-15@euro

fr_BE.UTF-8	fr_BE.UTF-8@euro
fr_CA.ISO8859-1	fr_CA
fr_CH.ISO8859-1	fr_CH
fr_FR.ISO8859-1	fr, fr_FR
fr_FR.ISO8859-15	fr.ISO8859-15, fr_FR.ISO8859-15@euro
fr_FR.UTF-8	fr.UTF-8, fr_FR.UTF-8@euro
he_IL.ISO8859-8	he, he_IL
hr_HR.ISO8859-2	hr_HR
hu_HU.ISO8859-2	hu, hu_HU
is_IS.ISO8859-1	is_IS
it_IT.ISO8859-1	it, it_IT
it_IT.ISO8859-15	it.ISO8859-15, it_IT.ISO8859-15@euro
it_IT.UTF-8	it.UTF-8, it_IT.UTF-8@euro
ja_JP.eucJP	ja
ko_KR.EUC	ko
ko_KR.UTF-8	ko.UTF-8
lt_LT.ISO8859-13	lt, lt_LT
lv_LV.ISO8859-13	lv, lv_LV
mk_MK.ISO8859-5	mk_MK
nb_NO.ISO8859-1	no, no_NO, no_NO.ISO8859-1@bokmal
nl_BE.ISO8859-1	nl_BE
nl_BE.ISO8859-15	nl_BE.ISO8859-15@euro
nl_NL.ISO8859-1	nl, nl_NL
nl_NL.ISO8859-15	nl.ISO8859-15, nl_NL.ISO8859-15@euro
nn_NO.ISO8859-1	no_NO.ISO8859-1@nynorsk, no_NY
pl_PL.ISO8859-2	pl, pl_PL
pl_PL.UTF-8	pl.UTF-8
pt_BR.ISO8859-1	pt_BR
pt_PT.ISO8859-1	pt, pt_PT
pt_PT.ISO8859-15	pt.ISO8859-15, pt_PT.ISO8859-15@euro
ro_RO.ISO8859-2	ro_RO
ru_RU.ISO8859-5	ru, ru_RU
ru_RU.KOI8-R	ru.koi8-r
ru_RU.UTF-8	ru.UTF-8
sk_SK.ISO8859-2	sk_SK
sl_SI.ISO8859-2	sl_SI
sq_AL.ISO8859-2	sq_AL
sr_ME.ISO8859-5	sr_SP, sr_YU, sr_YU.ISO8859-5
sr_ME.UTF-8	sr_CS, sr_CS.UTF-8
sr_RS.ISO8859-5	sr_SP, sr_YU, sr_YU.ISO8859-5
sr_RS.UTF-8	sr_CS, sr_CS.UTF-8
sv_SE.ISO8859-1	sv, sv_SE
sv_SE.ISO8859-15	sv.ISO8859-15
sv_SE.UTF-8	sv.UTF-8
th_TH.TIS620	th, th_TH, th_TH.ISO8859-11
tr_TR.ISO8859-9	tr, tr_TR
zh_CN.EUC	zh
zh_CN.GBK	zh.GBK

```
zh_CN.UTF-8             zh.UTF-8
zh_TW.EUC               zh_TW
```

属性

有关下列属性的说明，请参见 attributes(5)：

属性类型	属性值
接口稳定性	Committed（已确定）

另请参见

gettext(1)、catopen(3C)、gettext(3C)、setlocale(3C)、attributes(5)、environ(5)

引用名

man – macros to format Reference Manual pages

用法概要

`nroff -man` *filename*...

`troff -man` *filename*...

描述

These macros are used to lay out the reference pages in this manual. If *filename* contains format input for a preprocessor, the commands shown above must be piped through the appropriate preprocessor. This is handled automatically by the `man(1)` command. See the `Conventions` section.

Any text argument *t* may be zero to six words. Quotes may be used to include SPACE characters in a “word”. If *text* is empty, the special treatment is applied to the next input line with text to be printed. In this way `.I` may be used to italicize a whole line, or `.SB` may be used to make small bold letters.

A prevailing indent distance is remembered between successive indented paragraphs, and is reset to default value upon reaching a non-indented paragraph. Default units for indents *i* are ens.

Type font and size are reset to default values before each paragraph, and after processing font and size setting macros.

These strings are predefined by `-man`:

`\*R`	‘®’, ‘(Reg)’ in `nroff`.
`\*S`	Change to default type size.

Requests

* n.t.l. = next text line; p.i. = prevailing indent

Request	*Cause*	*If no*	*Explanation*
	Break	*Argument*	
`.B`*t*	no	*t*=n.t.l.*	Text is in bold font.
`.BI`*t*	no	*t*=n.t.l.	Join words, alternating bold and italic.
`.BR`*t*	no	*t*=n.t.l.	Join words, alternating bold and roman.
`.DT`	no	.5i 1i...	Restore default tabs.
`.HP`*i*	yes	*i*=p.i.*	Begin paragraph with hanging indent. Set prevailing indent to *i*.
`.I`*t*	no	*t*=n.t.l.	Text is italic.
`.IB`*t*	no	*t*=n.t.l.	Join words, alternating italic and bold.
`.IP`*x i*	yes	*x*=""	Same as `.TP` with tag *x*.
`.IR`*t*	no	*t*=n.t.l.	Join words, alternating italic and roman.

Request	*Cause* *Break*	*If no* *Argument*	*Explanation*
`.IX`*t*	no	-	Index macro, for SunSoft internal use.
`.LP`	yes	-	Begin left-aligned paragraph. Set prevailing indent to .5i.
`.P`	yes	-	Same as `.LP`.
`.PD`*d*	no	*d*=.4v	Set vertical distance between paragraphs.
`.PP`	yes	-	Same as `.LP`.
`.RE`	yes	-	End of relative indent. Restores prevailing indent.
`.RB`*t*	no	*t*=n.t.l.	Join words, alternating roman and bold.
`.RI`*t*	no	*t*=n.t.l.	Join words, alternating roman and italic.
`.RS`*i*	yes	*i*=p.i.	Start relative indent, increase indent by *i*. Sets prevailing indent to .5i for nested indents.
`.SB`*t*	no	-	Reduce size of text by 1 point, make text bold.
`.SH`*t*	yes	-	Section Heading.
`.SM`*t*	no	*t*=n.t.l.	Reduce size of text by 1 point.
`.SS`*t*	yes	*t*=n.t.l.	Section Subheading.
`.TH`*n s d f m*	yes	-	Begin reference page *n*, of of section *s*; *d* is the date of the most recent change. If present, *f* is the left page footer; *m* is the main page (center) header. Sets prevailing indent and tabs to .5i.
`.TP`*i*	yes	*i*=p.i.	Begin indented paragraph, with the tag given on the next text line. Set prevailing indent to *i*.
`.TX`*t p*	no	-	Resolve the title abbreviation *t*; join to punctuation mark (or text) *p*.

Conventions

When formatting a manual page, `man` examines the first line to determine whether it requires special processing. For example a first line consisting of:

```
'\" t
```

indicates that the manual page must be run through the `tbl(1)` preprocessor.

A typical manual page for a command or function is laid out as follows:

.TH *title*[1-9]

The name of the command or function, which serves as the title of the manual page. This is followed by the number of the section in which it appears.

.SH NAME

The name, or list of names, by which the command is called, followed by a dash and then a one-line summary of the action performed. All in roman font, this section contains no `troff(1)` commands or escapes, and no macro requests.

.SH SYNOPSIS

Commands:

The syntax of the command and its arguments, as typed on the command line. When in boldface, a word must be typed exactly as printed. When in italics, a word can be replaced with an argument that you supply. References to bold or italicized items are not capitalized in other sections, even when they begin a sentence.

Syntactic symbols appear in roman face:

[ ]

An argument, when surrounded by brackets is optional.

|

Arguments separated by a vertical bar are exclusive. You can supply only one item from such a list.

. . .

Arguments followed by an ellipsis can be repeated. When an ellipsis follows a bracketed set, the expression within the brackets can be repeated.

Functions:

If required, the data declaration, or `#include` directive, is shown first, followed by the function declaration. Otherwise, the function declaration is shown.

.SH DESCRIPTION

A narrative overview of the command or function's external behavior. This includes how it interacts with files or data, and how it handles the standard input, standard output and standard error. Internals and implementation details are normally omitted. This section attempts to provide a succinct overview in answer to the question, *what does it do?*

Literal text from the synopsis appears in constant width, as do literal filenames and references to items that appear elsewhere in the reference manuals. Arguments are italicized.

If a command interprets either subcommands or an input grammar, its command interface or input grammar is normally described in a `USAGE` section, which follows the `OPTIONS` section. The `DESCRIPTION` section only describes the behavior of the command itself, not that of subcommands.

.SH OPTIONS
The list of options along with a description of how each affects the command's operation.

.SH RETURN VALUES
A list of the values the library routine returns to the calling program and the conditions that cause these values to be returned.

.SH EXIT STATUS
A list of the values the utility returns to the calling program or shell, and the conditions that cause these values to be returned.

.SH FILES
A list of files associated with the command or function.

.SH SEE ALSO
A comma-separated list of related manual pages, followed by references to other published materials.

.SH DIAGNOSTICS
A list of diagnostic messages and an explanation of each.

.SH BUGS
A description of limitations, known defects, and possible problems associated with the command or function.

SMF service `svc:/application/man-index` is used to generate the index files for `-f`, `-k`, and `-K` options of man(1). The SMF service is online by default. The index files for the options are generated during package/system installation if the FMRI is specified as an `restart_fmri` actuator. Any man page delivering packages that use the FMRI as `restart_fmri` actuator for its file actions must deliver a unique symbolic link to their man page directory at the following directory:

```
/usr/share/man/index.d/
```

The unique symbolic link names should be of package FMRI without scheme, slash character (`/`) replaced with "`%2F`", and optionally have some or all sequences of numbers of version. For instance, for the following package FMRI:

```
pkg://opensolaris.org/library/libc@5.11,5.11-0.75:20071001T163427
```

could supply one of the symbolic link names such as:

```
opensolaris.org%2Flibrary%2Flibc
opensolaris.org%2Flibrary%2Flibc@5.11
```

The method of the SMF service, upon requested to generate index files using `restart_fmri` during package install, checks the symbolic links. Based on the newness of the installed files, index files are generated only as necessary.

Unless used in an IPS package as restart_fmri, by default, it generates index files in /usr/share/man/ and /usr/gnu/share/man/. When used in and IPS package as restart_fmri, it generates and places index files in the same directory as the man page source directory of the package.

文件

/usr/share/man/index.d/
: Directory to save symbolic links to man page directories needing index files for -f, -k, and -K queries.

/usr/share/man/man_index/*
: Table of Contents and keyword database.

Generated files include:

- /usr/share/man/man_index/man.idx
- /usr/share/man/man_index/man.dic
- /usr/share/man/man_index/man.frq
- /usr/share/man/man_index/man.pos

/usr/share/lib/tmac/an

另请参见

man(1), nroff(1), troff(1), whatis(1)

Dale Dougherty and Tim O'Reilly, Unix Text Processing

引用名

mansun – macros to format Reference Manual pages

用法概要

```
nroff  -mansun filename...
troff  -mansun filename...
```

描述

These macros are used to lay out the reference pages in this manual. Note: if *filename* contains format input for a preprocessor, the commands shown above must be piped through the appropriate preprocessor. This is handled automatically by `man(1)`. See the “Conventions" section.

Any text argument *t* may be zero to six words. Quotes may be used to include SPACE characters in a “word”. If *text* is empty, the special treatment is applied to the next input line with text to be printed. In this way `.I` may be used to italicize a whole line, or `.SB` may be used to make small bold letters.

A prevailing indent distance is remembered between successive indented paragraphs, and is reset to default value upon reaching a non-indented paragraph. Default units for indents *i* are ens.

Type font and size are reset to default values before each paragraph, and after processing font and size setting macros.

These strings are predefined by `-mansun`:

`\*R` ‘®', ‘(Reg)' in `nroff`.

`\*S` Change to default type size.

Requests

* n.t.l. = next text line; p.i. = prevailing indent

Request	*Cause*	*If no*	*Explanation*
	Break	*Argument*	
`.B` *t*	no	*t*=n.t.l.*	Text is in bold font.
`.BI` *t*	no	*t*=n.t.l.	Join words, alternating bold and italic.
`.BR` *t*	no	*t*=n.t.l.	Join words, alternating bold and Roman.
`.DT`	no	.5i 1i...	Restore default tabs.
`.HP` *i*	yes	*i*=p.i.*	Begin paragraph with hanging indent. Set prevailing indent to *i*.
`.I` *t*	no	*t*=n.t.l.	Text is italic.
`.IB` *t*	no	*t*=n.t.l.	Join words, alternating italic and bold.
`.IP` *x i*	yes	*x*=""	Same as `.TP` with tag *x*.

Request	*Cause* *Break*	*If no* *Argument*	*Explanation*
`.IR` *t*	no	*t*=n.t.l.	Join words, alternating italic and Roman.
`.IX` *t*	no	-	Index macro, for SunSoft internal use.
`.LP`	yes	-	Begin left-aligned paragraph. Set prevailing indent to .5i.
`.P`	yes	-	Same as `.LP`.
`.PD` *d*	no	*d*=.4v	Set vertical distance between paragraphs.
`.PP`	yes	-	Same as `.LP`.
`.RE`	yes	-	End of relative indent. Restores prevailing indent.
`.RB` *t*	no	*t*=n.t.l.	Join words, alternating Roman and bold.
`.RI` *t*	no	*t*=n.t.l.	Join words, alternating Roman and italic.
`.RS` *i*	yes	*i*=p.i.	Start relative indent, increase indent by *i*. Sets prevailing indent to .5i for nested indents.
`.SB` *t*	no	-	Reduce size of text by 1 point, make text bold.
`.SH` *t*	yes	-	Section Heading.
`.SM` *t*	no	*t*=n.t.l.	Reduce size of text by 1 point.
`.SS` *t*	yes	*t*=n.t.l.	Section Subheading.
`.TH` *n s d f m*	yes	-	Begin reference page *n*, of of section *s*; *d* is the date of the most recent change. If present, *f* is the left page footer; *m* is the main page (center) header. Sets prevailing indent and tabs to .5i.
`.TP` *i*	yes	*i*=p.i.	Begin indented paragraph, with the tag given on the next text line. Set prevailing indent to *i*.
`.TX` *t p*	no	-	Resolve the title abbreviation *t*; join to punctuation mark (or text) *p*.

Conventions

When formatting a manual page, `mansun` examines the first line to determine whether it requires special processing. For example a first line consisting of:

```
'\" t
```

indicates that the manual page must be run through the `tbl(1)` preprocessor.

A typical manual page for a command or function is laid out as follows:

.TH *title* [1-8]
: The name of the command or function, which serves as the title of the manual page. This is followed by the number of the section in which it appears.

.SH NAME
: The name, or list of names, by which the command is called, followed by a dash and then a one-line summary of the action performed. All in Roman font, this section contains no `troff(1)` commands or escapes, and no macro requests. It is used to generate the `windex` database, which is used by the `whatis(1)` command.

.SH SYNOPSIS

Commands: The syntax of the command and its arguments, as typed on the command line. When in boldface, a word must be typed exactly as printed. When in italics, a word can be replaced with an argument that you supply. References to bold or italicized items are not capitalized in other sections, even when they begin a sentence.

Syntactic symbols appear in Roman face:

[ ] An argument, when surrounded by brackets is optional.

| Arguments separated by a vertical bar are exclusive. You can supply only one item from such a list.

. . . Arguments followed by an ellipsis can be repeated. When an ellipsis follows a bracketed set, the expression within the brackets can be repeated.

Functions: If required, the data declaration, or `#include` directive, is shown first, followed by the function declaration. Otherwise, the function declaration is shown.

.SH DESCRIPTION
: A narrative overview of the command or function's external behavior. This includes how it interacts with files or data, and how it handles the standard input, standard output and standard error. Internals and implementation details are normally omitted. This section attempts to provide a succinct overview in answer to the question, "what does it do?"

Literal text from the synopsis appears in constant width, as do literal filenames and references to items that appear elsewhere in the reference manuals. Arguments are italicized.

If a command interprets either subcommands or an input grammar, its command interface or input grammar is normally described in a `USAGE` section, which follows the OPTIONS section. The DESCRIPTION section only describes the behavior of the command itself, not that of subcommands.

.SH OPTIONS
The list of options along with a description of how each affects the command's operation.

.SH FILES
A list of files associated with the command or function.

.SH SEE ALSO
A comma-separated list of related manual pages, followed by references to other published materials.

.SH DIAGNOSTICS
A list of diagnostic messages and an explanation of each.

.SH BUGS
A description of limitations, known defects, and possible problems associated with the command or function.

文件

/usr/share/lib/tmac/ansun

/usr/share/man/windex

另请参见

man(1), nroff(1), troff(1), whatis(1)

Dale Dougherty and Tim O'Reilly, Unix Text Processing

引用名

me – macros for formatting papers

用法概要

```
nroff  -me [options] filename...
```

```
troff  -me [options] filename...
```

描述

This package of `nroff` and `troff` macro definitions provides a canned formatting facility for technical papers in various formats. When producing 2-column output on a terminal, filter the output through `col(1)`.

The macro requests are defined below. Many `nroff` and `troff` requests are unsafe in conjunction with this package, however, these requests may be used with impunity after the first `.pp`:

.bp	begin new page
.br	break output line here
.sp *n*	insert *n* spacing lines
.ls *n*	(line spacing) *n*=1 single, *n*=2 double space
.na	no alignment of right margin
.ce *n*	center next *n* lines
.ul *n*	underline next *n* lines
.sz +*n*	add *n* to point size

Output of the `eqn(1)`, `neqn(1)`, `refer(1)`, and `tbl(1)` preprocessors for equations and tables is acceptable as input.

Requests

In the following list, “initialization” refers to the first `.pp, .lp`, `.ip`, `.np`, `.sh`, or `.uh` macro. This list is incomplete.

Request	*Initial*	*Cause*	*Explanation*
	Value	*Break*	
`.(c`	-	yes	Begin centered block.
`.(d`	-	no	Begin delayed text.
`.(f`	-	no	Begin footnote.
`.(l`	-	yes	Begin list.
`.(q`	-	yes	Begin major quote.
`.(`*xx*	-	no	Begin indexed item in index *x*.
`.(z`	-	no	Begin floating keep.

Request	*Initial* *Value*	*Cause* *Break*	*Explanation*
`.)c`	-	yes	End centered block.
`.)d`	-	yes	End delayed text.
`.)f`	-	yes	End footnote.
`.)l`	-	yes	End list.
`.)q`	-	yes	End major quote.
`.)x`	-	yes	End index item.
`.)z`	-	yes	End floating keep.
`.++` *m H*	-	no	Define paper section. *m* defines the part of the paper, and can be `C` (chapter), `A` (appendix), `P` (preliminary, for instance, abstract, table of contents, etc.), `B` (bibliography), `RC` (chapters renumbered from page one each chapter), or `RA` (appendix renumbered from page one).
`.+c` *T*	-	yes	Begin chapter (or appendix, etc., as set by `.++`). *T* is the chapter title.
`.1c`	1	yes	One column format on a new page.
`.2c`	1	yes	Two column format.
`.EN`	-	yes	Space after equation produced by `eqn` or `neqn`.
`.EQ` *x y*	-	yes	Precede equation; break out and add space. Equation number is *y*. The optional argument *x* may be *I* to indent equation (default), *L* to left-adjust the equation, or

Request	*Initial* *Value*	*Cause* *Break*	*Explanation*
			C to center the equation.
`.GE`	-	yes	End *gremlin* picture.
`.GS`	-	yes	Begin *gremlin* picture.
`.PE`	-	yes	End `pic` picture.
`.PS`	-	yes	Begin `pic` picture.
`.TE`	-	yes	End table.
`.TH`	-	yes	End heading section of table.
`.TS` *x*	-	yes	Begin table; if *x* is *H* table has repeated heading.
`.ac` *A N*	-	no	Set up for ACM style output. *A* is the Author's name(s), *N* is the total number of pages. Must be given before the first initialization.
`.b` *x*	no	no	Print *x* in boldface; if no argument switch to boldface.
`.ba` *+n*	0	yes	Augments the base indent by *n*. This indent is used to set the indent on regular text (like paragraphs).
`.bc`	no	yes	Begin new column.
`.bi` *x*	no	no	Print *x* in bold italics (nofill only).
`.bu`	-	yes	Begin bulleted paragraph.
`.bx` *x*	no	no	Print *x* in a box (nofill only).
`.ef` *'x'y'z*	""""	no	Set even footer to *x y z*.
`.eh` *'x'y'z*	""""	no	Set even header to *x y z*.
`.fo` *'x'y'z*	""""	no	Set footer to *x y z*.
`.hx`	-	no	Suppress headers and footers on

Request	*Initial* *Value*	*Cause* *Break*	*Explanation*
			next page.
`.he` *'x'y'z*	"'''	no	Set header to *x y z*.
`.hl`	-	yes	Draw a horizontal line.
`.i` *x*	no	no	Italicize *x*; if *x* missing, italic text follows.
`.ip` *x y*	no	yes	Start indented paragraph, with hanging tag *x*. Indentation is *y* ens (default 5).
`.lp`	yes	yes	Start left-blocked paragraph.
`.lo`	-	no	Read in a file of local macros of the form `.*`*x*. Must be given before initialization.
`.np`	1	yes	Start numbered paragraph.
`.of` *'x'y'z*	"'''	no	Set odd footer to x y z.
`.oh` *'x'y'z*	"'''	no	Set odd header to x y z.
`.pd`	-	yes	Print delayed text.
`.pp`	no	yes	Begin paragraph. First line indented.
`.r`	yes	no	Roman text follows.
`.re`	-	no	Reset tabs to default values.
`.sc`	no	no	Read in a file of special characters and diacritical marks. Must be given before initialization.
`.sh` *n x*	-	yes	Section head follows, font automatically bold. *n* is level of section, *x* is title of section.
`.sk`	no	no	Leave the next page blank. Only one page is remembered ahead.

Request	*Initial* *Value*	*Cause* *Break*	*Explanation*
`.sm` *x*	-	no	Set *x* in a smaller pointsize.
`.sz` +*n*	10p	no	Augment the point size by *n* points.
`.th`	no	no	Produce the paper in thesis format. Must be given before initialization.
`.tp`	no	yes	Begin title page.
`.u` *x*	-	no	Underline argument (even in `troff`). (Nofill only).
`.uh`	-	yes	Like `.sh` but unnumbered.
`.xp` *x*	-	no	Print index *x*.

文件

`/usr/share/lib/tmac/e`

`/usr/share/lib/tmac/*.me`

另请参见

`col`(1), `eqn`(1), `nroff`(1), `refer`(1), `tbl`(1), `troff`(1)

引用名

mech_spnego – Simple and Protected GSS-API Negotiation Mechanism

用法概要

`/usr/lib/gss/mech_spnego.so.1`

描述

The SPNEGO security mechanism for GSS-API allows GSS-API applications to negotiate the actual security mechanism to be used in the GSS-API session. `mech_spnego.so.1` is a shared object module that is dynamically opened by applications that specify the SPNEGO Object Identifier (OID) in calls to the GSS-API functions (see `libgss`(3LIB)).

SPNEGO is described by IETF RFC 2478 and is intended to be used in environments where multiple GSS-API mechanisms are available to the client or server and neither side knows what mechanisms are supported by the other.

When SPNEGO is used, it selects the list of mechanisms to advertise by reading the GSS mechanism configuration file, `/etc/gss/mech` (see `mech`(4)), and by listing all active mechanisms except for itself.

选项

SPNEGO may be configured to function in two ways. The first way is to interoperate with Microsoft SSPI clients and servers that use the Microsoft "`Negotiate`" method, which is also based on SPNEGO. The Microsoft "`Negotiate`" mechanism does not strictly follow the IETF RFC. Therefore, use special handling in order to enable full interoperability. In order to interoperate, place option "`[ msinterop ]`" at the end of the SPNEGO line in `/etc/gss/mech`.

This is an example (from `/etc/gss/mech`):

```
spnego    1.3.6.1.5.5.2    mech_spnego.so    [ msinterop ]
```

Without the "`[ msinterop ]`" option, `mech_spnego` will follow the strict IETF RFC 2478 specification and will not be able to negotiate with Microsoft applications that try to use the SSPI "`Negotiate`" mechanism.

Interfaces

`mech_spnego.so.1` has no public interfaces. It is only activated and used through the GSS-API interface provided by `libgss.so.1` (see `libgss`(3LIB)).

文件

`/usr/lib/gss/mech_spnego.so.1`	shared object file
`/usr/lib/sparcv9/gss/mech_spnego.so.1`	SPARC 64-bit shared object file
`/usr/lib/amd64/gss/mech_spnego.so.1`	x86 64-bit shared object file

属性

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SUWNspnego
MT Level	Safe

另请参见

Intro(3), libgss(3LIB), mech(4), attributes(5)

《Oracle Solaris 11 开发者安全性指南》

引用名

mm – text formatting (memorandum) macros

用法概要

```
nroff  -mm [options] filename...
troff  -mm [options] filename...
```

描述

This package of nroff(1) and troff(1) macro definitions provides a formatting facility for various styles of articles, theses, and books. When producing 2-column output on a terminal or lineprinter, or when reverse line motions are needed, filter the output through col(1). All external -mm macros are defined below.

Note: this -mm macro package is an extended version written at Berkeley and is a superset of the standard -mm macro packages as supplied by Bell Labs. Some of the Bell Labs macros have been removed; for instance, it is assumed that the user has little interest in producing headers stating that the memo was generated at Whippany Labs.

Many nroff and troff requests are unsafe in conjunction with this package. However, the first four requests below may be used with impunity after initialization, and the last two may be used even before initialization:

.bp	begin new page
.br	break output line
.sp*n*	insert n spacing lines
.ce*n*	center next n lines
.ls*n*	line spacing: *n*=1 single, *n*=2 double space
.na	no alignment of right margin

Font and point size changes with \f and \s are also allowed; for example, \fIword\fR will italicize *word*. Output of the tbl(1), eqn(1) and refer(1) preprocessors for equations, tables, and references is acceptable as input.

Requests

Here is a table of macros.

Macro Name	Initial Value	Break? Reset?	Explanation
.1C	on	y,y	one column format on a new page
.2C [*l*]	–	y,y	two column format *l*=line length
.AE	–	y	end abstract
.AL [*t*] [*i*] [*s*]	*t*=1;*i*=.Li;*s*=0	y	Start automatic list type *t*=[1,A,a,I,i] 1=arabic numbers; A=uppercase letters a=lowercase letters; I=uppercase Roman numerals; i=lowercase Roman numerals indentation *i*; separation *s*

Macro Name	Initial Value	Break? Reset?	Explanation
`.AS` *m* [*n*]	*n*=`0`	y	begin abstract
`.AU`	–	y	author's name
`.AV` *x*	–	y	signature and date line of verifier *x*
`.B` *x*	–	n	embolden *x*; if no *x*, switch to boldface
`.BE`	–	y	end block text
`.BI` *x y*	–	n	embolden *x* and underline *y*
`.BL`	–	y	bullet list
`.BR` *x y*	–	n	embolden *x* and use Roman font for *y*
`.BS`	–	n	start block text
`.CN`	–	y	same as `.DE` (`nroff`)
`.CS`	–	y	cover sheet
`.CW`	–	n	same as `.DS I` (`nroff`)
`.DE`	–	y	end display
`.DF` [*p*] [*f*] [*rp*]	*p*=`L`;*f*=`N`	y	start floating display; position *p*=[`L`,`C`,`CB`] `L`=left; `I`=indent; `C`=center; `CB`=center block fill *f*=[`N`,`Y`]; right position *rp* (fill only)
`.DL` [*i*] [*s*]	–	y	start dash list
`.DS` [*p*] [*f*] [*rp*]	*p*=`L`;*f*=`N`	y	begin static display (see `.DF` for argument descriptions)
`.EC` *x* [*n*]	*n*=`1`	y	equation title; equation *x*; number *n*
`.EF` *x*	–	n	even footer appears at the bottom of even-numbered pages; *x*="*l*'*c*'*r*" *l*=left; *c*=center; *r*=right
`.EH` *x*	–	n	even header appears at the top of even-numbered pages; *x*="*l*'*c*'*r*" *l*=left; *c*=center; *r*=right
`.EN`	–	y	end displayed equation produced by `eqn`
`.EQ`	–	y	break out equation produced by `eqn`
`.EX` *x* [*n*]	*n*=`1`	y	exhibit title; exhibit *x*
			number *n*

Macro Name	Initial Value	Break? Reset?	Explanation
`.FD` [*f*] [*r*]	*f*=`10`;*r*=`1`	n	set footnote style format *f*=[`0-11`]; renumber *r*=[`0`,`1`]
`.FE`	–	y	end footnote
`.FG` *x* [*n*]	*n*=`1`	y	figure title; figure *x*; number *n*
`.FS`	–	n	start footnote
`.H` *l* [*t*]	–	y	produce numbered heading level *l*=[`1-7`]; title *t*
`.HU` *t*	–	y	produce unnumbered heading; title *t*
`.I` *x*	–	n	underline *x*
`.IB` *x y*	–	n	underline *x* and embolden *y*
`.IR` *x y*	–	n	underline *x* and use Roman font on *y*
`.LE` [*s*]	*s*=`0`	y	end list; separation *s*
`.LI` [*m*] [*p*]	–	y	start new list item; mark *m*
			prefix *p* (mark only)
`.ML` *m* [*i*] [*s*]	*s*=`0`	y	start marked list; mark *m* indentation *i*; separation *s*=[`0`,`1`]
`.MT` *x*		y	memo title; title *x*
`.ND` *x*		n	no date in page footer; *x* is date on cover
`.NE`	–	y	end block text
`.NS`	–	y	start block text
`.OF` *x*	–	n	odd footer appears at the bottom of odd-numbered pages; *x*="*l*'*c*'*r*" *l*=left; *c*=center; *r*=right
`.OF` *x*	–	n	odd header appears at the top of odd-numbered pages; *x*="*l*'*c*'*r*" *l*=left; *c*=center; *r*=right
`.OP`	–	y	skip to the top of an odd-number page
`.P` [*t*]	*t*=`0`	y,y	begin paragraph; *t*=[`0`,`1`] `0`=justified; `1`=indented
`.PF` *x*	–	n	page footer appears at the bottom of every page; *x*="*l*'*c*'*r*" *l*=left; *c*=center; *r*=right

Macro Name	Initial Value	Break? Reset?	Explanation
`.PH` *x*	–	n	page header appears at the top of every page; *x*="*l*'*c*'*r*" *l*=left; *c*=center; *r*=right
`.R`	on	n	return to Roman font
`.RB` *x y*	–	n	use Roman on *x* and embolden *y*
`.RI` *x y*	–	n	use Roman on *x* and underline *y*
`.RP` *x*	-	y,y	released paper format ? *x*=no stops title on first
`.RS`	5n	y,y	right shift: start level of relative indentation
`.S` *m n*	–	n	set character point size & vertical space character point size *m*; vertical space *n*
`.SA` *x*	*x*=`1`	n	justification; *x*=[`0`,`1`]
`.SK` *x*	–	y	skip *x* pages
`.SM`	–	n	smaller; decrease point size by 2
`.SP` [*x*]	–	y	leave *x* blank lines
`.TB` *x* [*n*]	*n*=`1`	y	table title; table *x*; number *n*
`.TC`	–	y	print table of contents (put at end of input file)
`.TE`	–	y	end of table processed by tbl
`.TH`	–	y	end multi-page header of table
`.TL`	–	n	title in boldface and two points larger
`.TM`	–	n	UC Berkeley thesis mode
`.TP` *i*	y	y	*i*=p.i. Begin indented paragraph, with the tag given on the next text line. Set prevailing indent to *i*.
`.TS` *x*	–	y,y	begin table; if *x*=`H` table has multi-page header
`.TY`	–	y	display centered title `CONTENTS`
`.VL` *i* [*m*] [*s*]	*m*=`0`;*s*=`0`	y	start variable-item list; indentation *i* mark-indentation *m*; separation *s*

Registers

Formatting distances can be controlled in `-mm` by means of built-in number registers. For example, this sets the line length to 6.5 inches:

```
.nr  LL  6.5i
```

Here is a table of number registers and their default values:

Name	Register Controls	Takes Effect	Default
`Cl`	contents level	table of contents	2
`De`	display eject	display	0
`Df`	display floating	display	5
`Ds`	display spacing	display	1v
`Hb`	heading break	heading	2
`Hc`	heading centering	heading	0
`Hi`	heading indent	heading	1
`Hi`	heading spacing	heading	1
`Hu`	heading unnumbered	heading	2
`Li`	list indentation	list	`6 (nroff) 5 (troff)`
`Ls`	list spacing	list	6
`Pi`	paragraph indent	paragraph	5
`Pt`	paragraph type	paragraph	1
`Si`	static indent	display	`5 (nroff) 3 (troff)`

When resetting these values, make sure to specify the appropriate units. Setting the line length to 7, for example, will result in output with one character per line. Setting `Pi` to 0 suppresses paragraph indentation

Here is a list of string registers available in `-mm`; they may be used anywhere in the text:

Name	String's Function
`\*Q`	quote (" in `nroff`, “ in `troff`)
`\*U`	unquote (" in `nroff`, '' in `troff`)
`\*—`	dash (-- in `nroff`, — in `troff`)
`\*(MO`	month (month of the year)

Name	String's Function
*(DY	day (current date)
**	automatically numbered footnote
*’	acute accent (before letter)
*‘	grave accent (before letter)
*^	circumflex (before letter)
*,	cedilla (before letter)
*:	umlaut (before letter)
*~	tilde (before letter)
\(BU	bullet item
\(DT	date (*month day*, *yr*)
\(EM	em dash
\(Lf	LIST OF FIGURES title
\(Lt	LIST OF TABLES title
\(Lx	LIST OF EXHIBITS title
\(Le	LIST OF EQUATIONS title
\(Rp	REFERENCES title
\(Tm	trademark character (TM)

When using the extended accent mark definitions available with .AM, these strings should come after, rather than before, the letter to be accented.

文件

/usr/share/lib/tmac/m

/usr/share/lib/tmac/mm.[nt]   nroff and troff definitions of mm.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	text/doctools

另请参见

col(1), eqn(1), nroff(1), refer(1), tbl(1), troff(1), attributes(5)

已知问题

Floating keeps and regular keeps are diverted to the same space, so they cannot be mixed together with predictable results.

引用名

ms – text formatting macros

用法概要

```
nroff  -ms [options] filename...
```

```
troff  -ms [options] filename...
```

描述

This package of nroff(1) and troff(1) macro definitions provides a formatting facility for various styles of articles, theses, and books. When producing 2-column output on a terminal or lineprinter, or when reverse line motions are needed, filter the output through col(1). All external -ms macros are defined below.

Note: this -ms macro package is an extended version written at Berkeley and is a superset of the standard -ms macro packages as supplied by Bell Labs. Some of the Bell Labs macros have been removed; for instance, it is assumed that the user has little interest in producing headers stating that the memo was generated at Whippany Labs.

Many nroff and troff requests are unsafe in conjunction with this package. However, the first four requests below may be used with impunity after initialization, and the last two may be used even before initialization:

.bp	begin new page
.br	break output line
.sp *n*	insert n spacing lines
.ce *n*	center next n lines
.ls *n*	line spacing: *n*=1 single, *n*=2 double space
.na	no alignment of right margin

Font and point size changes with \f and \s are also allowed; for example, \fIword\fR will italicize *word*. Output of the tbl(1), eqn(1) and refer(1) preprocessors for equations, tables, and references is acceptable as input.

Requests

Macro Name	Initial Value	Break? Reset?	Explanation
.AB *x*	–	y	begin abstract; if *x*=no do not label abstract
.AE	–	y	end abstract
.AI	–	y	author's institution
.AM	–	n	better accent mark definitions
.AU	–	y	author's name
.B *x*	–	n	embolden *x*; if no *x*, switch to boldface
.B1	–	y	begin text to be enclosed in a box

Macro Name	Initial Value	Break? Reset?	Explanation
`.B2`	–	y	end boxed text and print it
`.BT`	date	n	bottom title, printed at foot of page
`.BX` *x*	–	n	print word *x* in a box
`.CM`	if t	n	cut mark between pages
`.CT`	–	y,y	chapter title: page number moved to CF (TM only)
`.DA` *x*	if n	n	force date *x* at bottom of page; today if no *x*
`.DE`	–	y	end display (unfilled text) of any kind
`.DS` *x y*	I	y	begin display with keep; *x*=I, L, C, B; *y*=indent
`.ID` *y*	8n,.5i	y	indented display with no keep; *y*=indent
`.LD`	–	y	left display with no keep
`.CD`	–	y	centered display with no keep
`.BD`	–	y	block display; center entire block
`.EF` *x*	–	n	even page footer *x* (3 part as for `.tl`)
`.EH` *x*	–	n	even page header *x* (3 part as for `.tl`)
`.EN`	–	y	end displayed equation produced by `eqn`
`.EQ` *x y*	–	y	break out equation; *x*=L,I,C; *y*=equation number
`.FE`	–	n	end footnote to be placed at bottom of page
`.FP`	–	n	numbered footnote paragraph; may be redefined
`.FS` *x*	–	n	start footnote; *x* is optional footnote label
`.HD`	undef	n	optional page header below header margin
`.I` *x*	–	n	italicize *x*; if no *x*, switch to italics
`.IP` *x y*	–	y,y	indented paragraph, with hanging tag *x*; *y*=indent
`.IX` *x y*	–	y	index words *x y* and so on (up to 5 levels)
`.KE`	–	n	end keep of any kind
`.KF`	–	n	begin floating keep; text fills remainder of page
`.KS`	–	y	begin keep; unit kept together on a single page
`.LG`	–	n	larger; increase point size by 2
`.LP`	–	y,y	left (block) paragraph.

Macro Name	Initial Value	Break? Reset?	Explanation
`.MC` *x*	–	y,y	multiple columns; *x*=column width
`.ND` *x*	if t	n	no date in page footer; *x* is date on cover
`.NH` *x y*	–	y,y	numbered header; *x*=level, *x*=0 resets, *x*=S sets to *y*
`.NL`	10p	n	set point size back to normal
`.OF` *x*	–	n	odd page footer *x* (3 part as for `.tl`)
`.OH` *x*	–	n	odd page header *x* (3 part as for `.tl`)
`.P1`	if TM	n	print header on first page
`.PP`	–	y,y	paragraph with first line indented
`.PT`	- % -	n	page title, printed at head of page
`.PX` *x*	–	y	print index (table of contents); *x*=no suppresses title
`.QP`	–	y,y	quote paragraph (indented and shorter)
`.R`	on	n	return to Roman font
`.RE`	5n	y,y	retreat: end level of relative indentation
`.RP` *x*	–	n	released paper format; *x*=no stops title on first page
`.RS`	5n	y,y	right shift: start level of relative indentation
`.SH`	–	y,y	section header, in boldface
`.SM`	–	n	smaller; decrease point size by 2
`.TA`	8n,5n	n	set TAB characters to 8n 16n . . . (`nroff`) or 5n 10n . . . (`troff`)
`.TC` *x*	–	y	print table of contents at end; *x*=no suppresses title
`.TE`	–	y	end of table processed by `tbl`
`.TH`	–	y	end multi-page header of table
`.TL`	–	y	title in boldface and two points larger
`.TM`	off	n	UC Berkeley thesis mode
`.TS` *x*	–	y,y	begin table; if *x*=H table has multi-page header
`.UL` *x*	–	n	underline *x*, even in `troff`
`.UX` *x*	–	n	UNIX; trademark message first time; *x* appended
`.XA` *x y*	–	y	another index entry; *x*=page or no for none; y=indent

Macro Name	Initial Value	Break? Reset?	Explanation
`.XE`	–	y	end index entry (or series of `.IX` entries)
`.XP`	–	y,y	paragraph with first line indented, others indented
`.XS` *x y*	–	y	begin index entry; *x*=page or no for none; *y*=indent
`.1C`	on	y,y	one column format, on a new page
`.2C`	–	y,y	begin two column format
`.] –`	–	n	beginning of `refer` reference
`.[ 0`	–	n	end of unclassifiable type of reference
`.[ N`	–	n	N= 1:journal-article, 2:book, 3:book-article, 4:report

Registers

Formatting distances can be controlled in `-ms` by means of built-in number registers. For example, this sets the line length to 6.5 inches:

```
.nr  LL  6.5i
```

Here is a table of number registers and their default values:

Name	Register Controls	Takes Effect	Default
`PS`	point size	paragraph	10
`VS`	vertical spacing	paragraph	12
`LL`	line length	paragraph	6i
`LT`	title length	next page	same as `LL`
`FL`	footnote length	next `.FS`	5.5i
`PD`	paragraph distance	paragraph	1v (if n), .3v (if t)
`DD`	display distance	displays	1v (if n), .5v (if t)
`PI`	paragraph indent	paragraph	5n
`QI`	quote indent	next `.QP`	5n
`FI`	footnote indent	next `.FS`	2n
`PO`	page offset	next page	0 (if n), ≈1i (if t)
`HM`	header margin	next page	1i
`FM`	footer margin	next page	1i
`FF`	footnote format	next `.FS`	0 (1, 2, 3 available)

When resetting these values, make sure to specify the appropriate units. Setting the line length to 7, for example, will result in output with one character per line. Setting FF to 1 suppresses footnote superscripting; setting it to 2 also suppresses indentation of the first line; and setting it to 3 produces an .IP-like footnote paragraph.

Here is a list of string registers available in -ms; they may be used anywhere in the text:

Name	String's Function
*Q	quote (" in nroff, “ in troff)
*U	unquote (" in nroff, ” in troff)
*–	dash (-- in nroff, – in troff)
*(MO	month (month of the year)
*(DY	day (current date)
**	automatically numbered footnote
*’	acute accent (before letter)
*‘	grave accent (before letter)
*^	circumflex (before letter)
*,	cedilla (before letter)
*:	umlaut (before letter)
*~	tilde (before letter)

When using the extended accent mark definitions available with .AM, these strings should come after, rather than before, the letter to be accented.

文件

/usr/share/lib/tmac/s

/usr/share/lib/tmac/ms.???

另请参见

col(1), eqn(1), nroff(1), refer(1), tbl(1), troff(1)

已知问题

Floating keeps and regular keeps are diverted to the same space, so they cannot be mixed together with predictable results.

引用名

mutex – concepts relating to mutual exclusion locks

描述

Mutual exclusion locks (mutexes) prevent multiple threads from simultaneously executing critical sections of code which access shared data (that is, mutexes are used to serialize the execution of threads). All mutexes must be global. A successful call to acquire a mutex will cause another thread that is also trying to lock the same mutex to block until the owner thread unlocks the mutex.

Mutexes can synchronize threads within the same process or in other processes. Mutexes can be used to synchronize threads between processes if the mutexes are allocated in writable memory and shared among the cooperating processes (see mmap(2)), and have been initialized for this task.

The following table lists mutex functions and the actions they perform.

FUNCTION	ACTION
`mutex_init`	Initialize a mutex.
`mutex_destroy`	Destroy a mutex.
`mutex_lock`	Lock a mutex.
`mutex_trylock`	Attempt to lock a mutex.
`mutex_unlock`	Unlock a mutex.
`pthread_mutex_init`	Initialize a mutex.
`pthread_mutex_destroy`	Destroy a mutex.
`pthread_mutex_lock`	Lock a mutex.
`pthread_mutex_trylock`	Attempt to lock a mutex.
`pthread_mutex_unlock`	Unlock a mutex.

Initialization

Mutexes are either intra-process or inter-process, depending upon the argument passed implicitly or explicitly to the initialization of that mutex. A statically allocated mutex does not need to be explicitly initialized; by default, a statically allocated mutex is initialized with all zeros and its scope is set to be within the calling process.

For inter-process synchronization, a mutex needs to be allocated in memory shared between these processes. Since the memory for such a mutex must be allocated dynamically, the mutex needs to be explicitly initialized with the appropriate attribute that indicates inter-process use.

Locking and Unlocking

A critical section of code is enclosed by a call to lock the mutex and the call to unlock the mutex to protect it from simultaneous access by multiple threads. Only one thread at a time may possess mutually exclusive access to the critical section of code that is enclosed by the mutex-locking call and the mutex-unlocking call, whether the mutex's scope is intra-process or inter-process. A thread calling to lock the mutex either gets exclusive access to the code starting from the successful locking until its call to unlock the mutex, or it waits until the mutex is unlocked by the thread that locked it.

Mutexes have ownership, unlike semaphores. Only the thread that locked a mutex, (that is, the owner of the mutex), should unlock it.

If a thread waiting for a mutex receives a signal, upon return from the signal handler, the thread resumes waiting for the mutex as if there was no interrupt.

Caveats

Mutexes are almost like data – they can be embedded in data structures, files, dynamic or static memory, and so forth. Hence, they are easy to introduce into a program. However, too many mutexes can degrade performance and scalability of the application. Because too few mutexes can hinder the concurrency of the application, they should be introduced with care. Also, incorrect usage (such as recursive calls, or violation of locking order, and so forth) can lead to deadlocks, or worse, data inconsistencies.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT-Level	MT-Safe

另请参见

mmap(2), shmop(2), mutex_destroy(3C), mutex_init(3C), mutex_lock(3C), mutex_trylock(3C), mutex_unlock(3C), pthread_create(3C), pthread_mutex_destroy(3C), pthread_mutex_init(3C), pthread_mutex_lock(3C), pthread_mutex_trylock(3C), pthread_mutex_unlock(3C), pthread_mutexattr_init(3C), attributes(5), standards(5)

附注

In the current implementation of threads, pthread_mutex_lock(), pthread_mutex_unlock(), mutex_lock() mutex_unlock(), pthread_mutex_trylock(), and mutex_trylock() do not validate the mutex type. Therefore, an uninitialized mutex or a mutex with an invalid type does not return EINVAL. Interfaces for mutexes with an invalid type have unspecified behavior.

By default, if multiple threads are waiting for a mutex, the order of acquisition is undefined.

The system does not support multiple mappings to the same logical synch object if it is initialized as process-private (USYNC_THREAD for Solaris, PTHREAD_PROCESS_PRIVATE for POSIX). If you need to mmap(2)a synch object to different locations within the same address space, then the synch object should be initialized as a shared object (USYNC_PROCESS for Solaris, PTHREAD_PROCESS_SHARED for POSIX).

引用名

mwac, MWAC – Mandatory Write Access Control

描述

Mandatory Write Access Control (MWAC) implements a new policy in the Oracle Solaris operating environment, that allows for fine- grained control over the writability of objects on otherwise read-only file systems.

In the current instance of the Oracle Solaris operating environment, MWAC is available only to non-global zones. The global zone implements the MWAC policy for non-global zones, preventing any overruling of the policy from within the non-global zone.

Zones marked as `read-only` have their root file system write-protected by MWAC. Only the file system objects that are write-listed by the read-only-profile are writable. See `zonecfg`(1M). Other file system objects are read-only.

Creating links to objects that are read-only by virtue of the MWAC-policy is not allowed.

另请参见

`ln`(1), `zoneadm`(1M), `zonecfg`(1M), `link`(2), `pathconf`(2)

《Oracle Solaris 11.1 管理：安全服务》

引用名

nfssec – overview of NFS security modes

描述

The mount_nfs(1M) and share_nfs(1M) commands each provide a way to specify the security mode to be used on an NFS file system through the sec=*mode* option. *mode* can be sys, dh, krb5, krb5i, krb5p, or none. These security modes can also be added to the automount maps. mount_nfs(1M) allows you to specify a single security mode; share_nfs(1M) allows you to specify multiple modes (or none). With multiple modes, an NFS client can choose any of the modes in the list.

The sec=*mode* option on the share_nfs(1M) command line establishes the security mode of NFS servers. If the NFS connection uses the NFS Version 3 protocol, the NFS clients must query the server for the appropriate *mode* to use. If the NFS connection uses the NFS Version 2 protocol, then the NFS client uses the default security mode, which is currently sys. NFS clients may force the use of a specific security mode by specifying the sec=*mode* option on the command line. However, if the file system on the server is not shared with that security mode, the client may be denied access.

If the NFS client wants to authenticate the NFS server using a particular (stronger) security mode, the client wants to specify the security mode to be used, even if the connection uses the NFS Version 3 protocol. This guarantees that an attacker masquerading as the server does not compromise the client.

The NFS security modes are described below. Of these, the krb5, krb5i, krb5p modes use the Kerberos V5 protocol for authenticating and protecting the shared filesystems. Before these can be used, the system must be configured to be part of a Kerberos realm. See kerberos(5).

sys
Use AUTH_SYS authentication. The user's UNIX user-id and group-ids are passed in the clear on the network, unauthenticated by the NFS server. This is the simplest security method and requires no additional administration. It is the default used by Solaris NFS Version 2 clients and Solaris NFS servers.

dh
Use a Diffie-Hellman public key system (AUTH_DES, which is referred to as AUTH_DH in RFC 2695: Authentication Mechanisms for ONC RPC.

krb5
Use Kerberos V5 protocol to authenticate users before granting access to the shared filesystem.

krb5i
Use Kerberos V5 authentication with integrity checking (checksums) to verify that the data has not been tampered with.

krb5p
User Kerberos V5 authentication, integrity checksums, and privacy protection (encryption) on the shared filesystem. This provides the most secure filesystem sharing, as all traffic is encrypted. It should be noted that performance might suffer on some systems when using krb5p, depending on the computational intensity of the encryption algorithm and the amount of data being transferred.

`none`

Use null authentication (`AUTH_NONE`). NFS clients using `AUTH_NONE` have no identity and are mapped to the anonymous user `nobody` by NFS servers. A client using a security mode other than the one with which a Solaris NFS server shares the file system has its security mode mapped to `AUTH_NONE.` In this case, if the file system is shared with `sec=`*none,* users from the client are mapped to the anonymous user. The NFS security mode `none` is supported by `share_nfs`(1M).

`sec=`*mode*[`:`*mode*]. . .

Sharing uses one or more of the specified security modes. The *mode* in the `sec=`*mode* option must be a node name supported on the client. If the `sec=` option is not specified, the default security mode used is `AUTH_SYS`. Multiple `sec=` options can be specified on the command line, although each mode can appear only once.

Each `sec=` option specifies modes that apply to any subsequent `window=, rw, ro, rw=, ro=` and `root=` options that are provided before another `sec=`option. Each additional `sec=` resets the security mode context, so that more `window=, rw, ro, rw=, ro=` and `root=` options can be supplied for additional modes.

The NFSv4 server constructs a shared file system name space which is identical to the real file system name space on the server, including directories which are not actually shared, if they lead to shared directories. The constructed parts of the name space are known as the `pseudo-fs`. The `pseudo-fs` is always read-only.

As with NFSv3, the security mode of the shared directory is controlled using the `sec=`*mode* option of `share_nfs`(1M). However, the security mode of `pseudo-fs` objects is the union of the various security modes of the shared directories below.

When an NFSv4 client performs a mount, the client traverses the server's name space, from the root, down to the directory being mounted. Using the features of the NFSv4 protocol, the client may negotiate the security flavor of the directories as it proceeds down. If no `sec=`*mode* option is given to `mount_nfs` or an automounter map entry, then the client will do full negotiation for each directory down to the mount point, changing security flavors as needed. If `sec=`*mode* option is given, the client is constrained to use the requested security mode for all operations.

示例

示例 1 Sharing `/var` with Kerberos Authentication and Integrity Protection

The following example shares `/var` with Kerberos authentication and integrity protection:

```
share -F nfs -o sec=krb5i /var
```

示例 2　Sharing /var with Kerberos Authentication and Privacy Protection

The following example shares/var with Kerberos authentication and privacy protection:

```
share -F nfs -o sec=krb5p /var
```

示例 3　Sharing /var with Kerberos Authentication and Optionally Falling Back to AUTH_SYS Authentication

The following example shares /var with Kerberos authentication and optionally falls back to AUTH_SYS authentication:

```
share -F nfs -o sec=krb5:sys /var
```

示例 4　Sharing /var with Kerberos Authentication Allowing read/write Operations for Kerberos Authenticated Users and Optionally Falling Back to AUTH_SYS Authentication Allowing only Read Operations

The following example shares /var with Kerberos authentication allowing read/write operations for Kerberos authenticated users and optionally falls back to AUTH_SYS authentication allowing only read operations:

```
share -F nfs -o sec=krb5,rw,sec=sys,ro /var
```

文件

/etc/nfssec.conf　　NFS security service configuration file

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/file-system/nfs

另请参见

automount(1M), kclient(1M), mount_nfs(1M), share_nfs(1M), rpc_clnt_auth(3NSL), secure_rpc(3NSL), nfssec.conf(4), attributes(5), kerberos(5)

RFC 2695: Authentication Mechanisms for ONC RPC

附注

/etc/nfssec.conf lists the NFS security services. Do not edit this file. It is not intended to be user-configurable. See kclient(1M).

引用名

nis, NIS, NIS+ – network information name service

描述

NIS, formerly known as the Yellow Pages or YP, is the name of the network information name service in common use in networks on which Sun and other vendors' network nodes reside. The most recent version of NIS is version 2.

All commands and functions that use NIS version 2 are prefixed by the letters `yp` as in `ypmatch`(1), `ypcat`(1), `yp_match`(3NSL), and `yp_first`(3NSL).

The NIS+ name service is no longer shipped with the Solaris operating system. Tools to aid the migration from NIS+ to LDAP are available in the current Solaris release.

引用名

nwam – 网络自动配置管理

描述

可以根据系统定义的缺省策略规则或用户定义的策略规则自动管理网络配置。此管理由 nwamd 守护进程执行。

NWAM 工具由 network/physical:default SMF 服务提供。如果 netcfg/active_ncp 属性设置为活性 NCP 名称，则会启用 NWAM 配置管理。或者，如果 netcfg/active_ncp 属性设置为 DefaultFixed，则会执行传统的网络配置。

操作

在自动 NCP 中实施的缺省配置策略用于让所有以物理方式连接的以太网链路处于活动状态；如果未提供以太网链路，则使单个无线链路处于活动状态。DHCP 用于获取所有活动链路的 IP 地址。可以通过创建备用网络配置文件 (Network Configuration Profiles, NCP) 更改该缺省策略。nwam-manager GUI 工具或 nwamd(1M) 命令可用于创建和修改 NCP。

NWAM 管理也可用于高层网络配置，如名称服务和网络安全协议。此配置在位置配置文件中指定。

可使用 netcfg(1M) 管理配置文件。这是用于针对所有活性配置文件指定激活策略的主要工具。它也可用于指定位置以及简单 NCP 组件。

使用 dladm(1M) 或 ipadm(1M) 所进行的配置更改会应用于当前活动的 NCP，而不论 NCP 是活性的（即，由 nwamd 主动管理）还是固定的。

位置可管理一组更广泛的配置对象。这些对象的活动系统配置位于 SMF 中；而 NWAM 工具通过以下方式管理配置：在配置文件系统信息库中存储设置以及在激活位置时将这些设置应用于 SMF 系统信息库。

对于活性位置，禁用位置时会丢失位置处于活动状态时对 SMF 系统信息库所进行的更改；所以，应使用 netcfg 对 NWAM 配置文件进行更改。这些更改将保存至 NWAM 系统信息库，并且还会应用于活动系统配置（如果更改当前处于活动状态的位置）。

对于 DefaultFixed 位置，对 SMF 系统信息库所进行的更改将在禁用位置时保存至 NWAM 系统信息库。因此，将在下次启用 DefaultFixed 位置时恢复这些更改。

服务属性

以下列表的格式为：

property_group/property_name property_type default_value
    说明

nwamd/debug boolean false
    使用 daemon.debug 启用调试日志。

nwamd/autoconf boolean false
    指示在没有更好（更优）选择的情况下是否应自动连接打开的 WLAN。

`nwamd/ncu_wait_time` count `60`

在尝试下一个可用 NCU 之前等待 NCU（或链路/接口 NCU 对，视需要而定）启动的秒数。初启活动不会被取消，并且最终可能会成功，此时会根据指定的配置条件激活更优的 NCU 并可能会禁用备用 NCU。

`nwamd/condition_check_interval` count `120`

对有条件激活对象执行定期条件检查的间隔秒数。最小值为 30 秒。

`nwamd/scan_interval` count `120`

执行定期无线扫描的间隔秒数。

`nwamd/scan_level` astring `weak`

一个信号强度阈值；如果当前连接的 AP 低于此信号级别并以更高的信号强度提供了对等 AP（属于同一 ESSID），则将断开现有连接以便于连接到信号更强的 AP。

`nwamd/strict_bssid` boolean `false`

如果为 true，则必须同时匹配 ESSID 和 BSSID 才能连接到以前连接的 WLAN。如果为 false，则只需匹配 ESSID。

`netcfg/active_ncp` astring `Automatic`

当前活动的 NCP。此属性不应由用户设置；它由 NWAM 服务在内部使用以在多次重新启动之间保持持久性。更改当前活动 NCP 的正确用户界面是使用 `nwam-manager` GUI 或 `netadm` 命令的 `enable` 子命令。

如果启用保留的 NCP 名称 `DefaultFixed`，将禁用 NWAM 自动配置管理，并启用传统的固定网络配置。

属性

有关下列属性的说明，请参见 `attributes(5)`：

属性类型	属性值
可用性	system/core-os
接口稳定性	Volatile（可变）

另请参见

`svcs(1)`、`dladm(1M)`、`ipadm(1M)`、`netcfg(1M)`、`netcfgd(1M)`、`nwamd(1M)`、`svadm(1M)`、`attr`

另请参见 JDS/GNOME 手册页集合中的 `nwam-manager(1M)`。

附注

联网服务由服务管理工具 `smf(5)` 管理，其服务标识符为：

```
svc:/network/physical:default
```

可以使用 `svadm(1M)` 来对此服务执行管理操作（如启用、禁用或请求重新启动）。可以使用 `svcs(1)` 命令来查询服务的状态。

引用名

openssl – OpenSSL cryptographic and Secure Sockets Layer toolkit

描述

OpenSSL is a cryptography toolkit that implements the Secure Sockets Layer (SSLv2/v3) and Transport Layer Security (TLS v1) network protocols.

The following features are omitted from the binaries for issues including but not limited to patents, trademark, and US export restrictions: ECC, IDEA, MDC2, RC3, RC5, 4758_CCA Engine, AEP Engine, Atalla Engine, CHIL Engine, CSWIFT Engine, GMP Engine, NURON Engine, PadLock Engine, Sureware Engine, and UBSEC Engine.

The PKCS#11 Engine

A new PKCS#11 engine has been included with ENGINE name `pkcs11`. The engine was developed in Sun and is not integrated in the OpenSSL project.

The PKCS#11 engine is configured to use the Oracle Solaris Cryptographic Framework. See `cryptoadm`(1M) for configuration information.

The PKCS#11 engine can support the following set of mechanisms: `CKM_AES_CBC`, `CKM_AES_ECB`, `CKM_BLOWFISH_CBC`, `CKM_DES_CBC`, `CKM_DES_ECB`, `CKM_DES3_CBC`, `CKM_DES3_ECB`, `CKM_DSA`, `CKM_MD5`, `CKM_RC4`, `CKM_RSA_PKCS`, `CKM_RSA_X_509`, `CKM_SHA_1`, `CKM_SHA224`, `CKM_SHA256`, `CKM_SHA384`, `CKM_SHA512`, `CKM_SHA224_HMAC`, `CKM_SHA224_HMAC_GENERAL`, and `CKM_SHA224_KEY_DERIVATION`.

The set of mechanisms available depends on installed Crypto Framework providers. To see what mechanisms can be offloaded to the Cryptographic Framework through the PKCS#11 engine on a given machine, run the following command:

```
/usr/sfw/bin/openssl engine pkcs11 -vvv -t -c
```

Due to requirements of the PKCS#11 standard regarding `fork`(2) behavior, some applications that use the OpenSSL EVP interfaces and `fork()` with active `crypto` contexts might experience unexpected behavior.

The Dynamic Engine Support

The dynamic engine support has been enabled, which allows an external engine, in the form of a shared library, to be dynamically bound and used by an OpenSSL-based application.

Run the following command to see if the dynamic engine is supported:

```
$openssl engine dynamic
(dynamic) Dynamic engine loading support
```

The devcrypto Engine

This engine is implemented as a separate shared library, and it can be used by an OpenSSL application through the OpenSSL dynamic engine support. This engine was developed in Sun and is not integrated in the OpenSSL project.

Files for the `devcrypto` engine library are:

```
/lib/openssl/engines/libdevcrypto.so
/lib/openssl/engines/64/libdevcrypto.so
```

The devcrypto engine accesses only the kernel hardware providers from the Oracle Solaris Cryptographic Framework. To see the hardware provider information on a system, use the cryptoadm(1M) command.

The devcrypto engine supports the following set of mechanisms: CKM_AES_CBC, CKM_AES_CTR, CKM_AES_ECB, CKM_BLOWFISH_CBC, CKM_DES_CBC, CKM_DES_ECB, CKM_DES3_CBC, CKM_DES3_ECB, and CKM_RC4.

The set of mechanisms available depends on hardware providers installed and enabled in the Cryptographic Framework. To see what mechanisms are supported by the devcrypto engine on a particular machine, run the following command:

```
$openssl engine dynamic -pre SO_PATH:/lib/openssl/engines/libdevcrypto.so\
-pre LOAD -t -c
```

Using FIPS Mode

Currently a FIPS-140 certified mode is not available in Oracle Solaris.

Building an OpenSSL Application

To build an OpenSSL application, use the following cc command line options:

```
cc [ flag... ] file... -lcrypto -lssl [ library... ]
```

Accessing RSA Keys in PKCS#11 Keystores

OpenSSL can access RSA keys in PKCS#11 keystores using the following functions of the ENGINE API:

```
EVP_PKEY *ENGINE_load_private_key(ENGINE *e,
 const char *key_id, UI_METHOD *ui_method,
 void *callback_data)

EVP_PKEY *ENGINE_load_public_key(ENGINE *e,
 const char *key_id, UI_METHOD *ui_method,
 void *callback_data)
```

key_id, formerly for filenames only, can be now also set to a PKCS#11 URI. The EVP_PKEY structure is newly allocated and caller is responsible to free the structure later. To avoid clashes with existing filenames, file:// prefix for filenames is now also accepted but only when the PKCS#11 engine is in use. The PKCS#11 URI specification follows:

```
pkcs11:[token=<label>][:manuf=<label>][;serial=<label>]
   [;model=<label>][;object=<label>]
   [;objecttype=(public|private|cert)]
   [;passphrasedialog=(builtin|exec:<file>)]
```

The ordering of keywords is not significant. The PKCS#11 engine uses the keystore for the slot chosen for public key operations, which is metaslot on a standard configured machine. Currently, the PKCS#11 engine ignores the objecttype keyword. The only mandatory keyword is object which is the key object label. For information on how to use a different, possibly hardware, keystore with metaslot, seelibpkcs11(3LIB).

The token PIN is provided by way of the passphrasedialog keyword and is either read from the terminal (builtin) or from the output of an external command (exec:<file>). The PIN is used to log into the token and by default is deleted from the memory then. The keyword pin is intentionally not provided due to inherent security problems of possible use of a password in the process arguments.

Due to fork safety issues the application must re-login if the child continues to use the PKCS#11 engine. It is done inside of the engine automatically if fork is detected and in that case, exec:<file> option of the passphrasedialog keyword can be used. Alternatively, an environment variable OPENSSL_PKCS11_PIN_CACHING_POLICY can be used to allow the PIN to be cached in memory and reused in the child. It can be set to none which is the default, memory to store the PIN in memory, and mlocked-memory to keep the PIN in a locked page using mlock(3C). PRIV_PROC_LOCK_MEMORY privilege is required in that case.

Sensitive parts of private keys are never read from the token to the process memory no matter whether the key is tagged with sensitive flag or not. The PKCS#11 engine uses the public components as a search key to get a PKCS#11 object handle to the private key.

To use the RSA keys by reference, high level API functions such as RSA_public_decrypt(), EVP_PKEY_set1_RSA(), or EVP_SignInit() must be used. Low level functions might go around the engine and fail to make use of the feature.

Additional Documentation

Extensive additional documentation for OpenSSL modules is available in the /usr/share/man/man1openssl, /usr/share/man/man3openssl, /usr/share/man/man5openssl, and /usr/share/man/man7openssl directories.

To view the license terms, attribution, and copyright for OpenSSL, run pkg info --license library/security/openssl.

示例

示例 1 Generating and Printing a Public Key

The following example generates and prints a public key stored in an already initilized PKCS#11 keystore. Notice the use of -engine pkcs11 and -inform e.

```
$ pktool gencert keystore=pkcs11 label=mykey \
   subject="CN=test" keytype=rsa keylen=1024 serial=01
$ openssl rsa -in "pkcs11:object=mykey;passphrasedialog=builtin"\
   -pubout -text -engine pkcs11 -inform e
```

属性

See attributes(5) for a description of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	library/security/openssl, library/security/openssl
Interface Stability	Volatile

另请参见

crle(1), cryptoadm(1M), libpkcs11(3LIB), attributes(5), privileges(5)

/usr/share/man/man1openssl/openssl.1openssl, /usr/share/man/man1openssl/CRYPTO_num_locks.3openssl, /usr/share/man/man3openssl/engine.3, /usr/share/man/man3openssl/evp.3

引用名

pam_allow – PAM authentication, account, session and password management PAM module to allow operations

用法概要

```
pam_allow.so.1
```

描述

The pam_allow module implements all the PAM service module functions and returns PAM_SUCCESS for all calls. Opposite functionality is available in the pam_deny(5) module.

Proper Solaris authentication operation requires pam_unix_cred(5) be stacked above pam_allow.

The following options are interpreted:

debug Provides syslog(3C) debugging information at the LOG_AUTH | LOG_DEBUG level.

错误

PAM_SUCCESS is always returned.

示例

示例 1 Allowing ssh none

The following example is a pam.conf fragment that illustrates how to allow the SSHv2 userauth of “none”:

```
sshd-none  auth     required      pam_unix_cred.so.1
sshd-none  auth     sufficient    pam_allow.so.1
sshd-none  account  sufficient    pam_allow.so.1
sshd-none  session  sufficient    pam_allow.so.1
sshd-none  password sufficient    pam_allow.so.1
```

The equivalent configuration using /etc/pam.d/ would be the following entries in /etc/pam.d/sshd-none:

```
auth     required      pam_unix_cred.so.1
auth     sufficient    pam_allow.so.1
account  sufficient    pam_allow.so.1
session  sufficient    pam_allow.so.1
password sufficient    pam_allow.so.1
```

示例 2 Allowing Kiosk Automatic Login Service

The following example is a pam.conf fragment that illustrates how to allow gdm kiosk automatic login:

```
gdm-autologin auth  required    pam_unix_cred.so.1
gdm-autologin auth  sufficient  pam_allow.so.1
```

The equivalent configuration using /etc/pam.d/ would be the following entries in /etc/pam.d/gdm-autologin:

```
auth  required    pam_unix_cred.so.1
auth  sufficient  pam_allow.so.1
```

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

libpam(3LIB), pam(3PAM), pam_sm(3PAM), syslog(3C), pam.conf(4), attributes(5), pam_deny(5), pam_unix_cred(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

This module is intended to be used to either allow access to specific services names, or to all service names not specified (by specifying it as the default service stack).

引用名

pam_authtok_check – authentication and password management module

用法概要

```
pam_authtok_check.so.1
```

描述

pam_authtok_check provides functionality to the Password Management stack. The implementation of pam_sm_chauthtok() performs a number of checks on the construction of the newly entered password. pam_sm_chauthtok() is invoked twice by the PAM framework, once with flags set to PAM_PRELIM_CHECK, and once with flags set to PAM_UPDATE_AUTHTOK. This module only performs its checks during the first invocation. This module expects the current authentication token in the PAM_OLDAUTHTOK item, the new (to be checked) password in the PAM_AUTHTOK item, and the login name in the PAM_USER item. The checks performed by this module are:

length
: The password length should not be less that the minimum specified in /etc/default/passwd.

circular shift
: The password should not be a circular shift of the login name. This check may be disabled in /etc/default/passwd.

complexity
: The password should contain at least the minimum number of characters described by the parameters MINALPHA, MINNONALPHA, MINDIGIT, and MINSPECIAL. Note that MINNONALPHA describes the same character classes as MINDIGIT and MINSPECIAL combined; therefore the user cannot specify both MINNONALPHA and MINSPECIAL (or MINDIGIT). The user must choose which of the two options to use. Furthermore, the WHITESPACE parameter determines whether whitespace characters are allowed. If unspecified MINALPHA is 2, MINNONALPHA is 1 and WHITESPACE is yes

variation
: The old and new passwords must differ by at least the MINDIFF value specified in /etc/default/passwd. If unspecified, the default is 3. For accounts in name services which support password history checking, if prior history is defined, the new password must not match the prior passwords.

dictionary check
: The password must not be based on a dictionary word. The list of words to be used for the site's dictionary can be specified with DICTIONLIST. It should contain a comma-separated list of filenames, one word per line. The database that is created from these files is stored in the directory named by DICTIONDBDIR (defaults to /var/passwd). See mkpwdict(1M) for information on pre-generating the database. If neither DICTIONLIST nor DICTIONDBDIR is specified, no dictionary check is made.

upper/lower case
: The password must contain at least the minimum of upper- and lower-case letters specified by the MINUPPER and MINLOWER values in /etc/default/passwd. If unspecified, the defaults are 0.

maximum repeats — The password must not contain more consecutively repeating characters than specified by the MAXREPEATS value in /etc/default/passwd. If unspecified, no repeat character check is made.

The following option may be passed to the module:

force_check — If the PAM_NO_AUTHTOK_CHECK flag set, force_check ignores this flag. The PAM_NO_AUTHTOK_CHECK flag can be set to bypass password checks (see pam_chauthtok(3PAM)).

server_policy — If the account authority for the user, as specified by PAM_USER, is not files or NIS, and if server_policy is specified, this module does not perform any password-strength checks. Instead, it leaves it to the account authority to validate the new password against its own set of rules.

debug — syslog(3C) debugging information at the LOG_DEBUG level

返回值

If the password in PAM_AUTHTOK passes all tests, PAM_SUCCESS is returned. If any of the tests fail, PAM_AUTHTOK_ERR is returned.

文件

/etc/default/passwd — See passwd(1) for a description of the contents.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

passwd(1), pam(3PAM), mkpwdict(1M), pam_chauthtok(3PAM), syslog(3C), libpam(3LIB), pam.conf(4), passwd(4), shadow(4), attributes(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

引用名

pam_authtok_get – authentication and password management module

用法概要

```
pam_authtok_get.so.1
```

描述

The pam_authtok_get service module provides password prompting funtionality to the PAM stack. It implements pam_sm_authenticate() and pam_sm_chauthtok(), providing functionality to both the Authentication Stack and the Password Management Stack.

Authentication Service

The implementation of pam_sm_authenticate(3PAM) prompts the user name if not set and then tries to get the authentication token from the pam handle. If the token is not set, it then prompts the user for a password and stores it in the PAM item PAM_AUTHTOK. This module is meant to be the first module on an authentication stack where users are to authenticate using a keyboard.

Password Management Service

Due to the nature of the PAM Password Management stack traversal mechanism, the pam_sm_chauthtok(3PAM) function is called twice. Once with the PAM_PRELIM_CHECK flag, and one with the PAM_UPDATE_AUTHTOK flag.

In the first (PRELIM) invocation, the implementation of pam_sm_chauthtok(3PAM) moves the contents of the PAM_AUTHTOK (current authentication token) to PAM_OLDAUTHTOK, and subsequentially prompts the user for a new password. This new password is stored in PAM_AUTHTOK.

If a previous module has set PAM_OLDAUTHTOK prior to the invocation of pam_authtok_get, this module turns into a NO-OP and immediately returns PAM_SUCCESS.

In the second (UPDATE) invocation, the user is prompted to Re-enter his password. The pam_sm_chauthtok implementation verifies this reentered password with the password stored in PAM_AUTHTOK. If the passwords match, the module returns PAM_SUCCESS.

The following option can be passed to the module:

debug	syslog(3C) debugging information at the LOG_DEBUG level

错误

The authentication service returns the following error codes:

PAM_SUCCESS	Successfully obtains authentication token
PAM_SYSTEM_ERR	Fails to retrieve username, username is NULL or empty

The password management service returns the following error codes:

PAM_SUCCESS	Successfully obtains authentication token
PAM_AUTHTOK_ERR	Authentication token manipulation error

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

pam(3PAM), pam_authenticate(3PAM), syslog(3C), libpam(3LIB), pam.conf(4), attributes(5), pam_authtok_check(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

引用名

pam_authtok_store – password management module

用法概要

```
pam_authtok_store.so.1
```

描述

pam_authtok_store provides functionality to the PAM password management stack. It provides one function: pam_sm_chauthtok().

When invoked with flags set to PAM_UPDATE_AUTHTOK, this module updates the authentication token for the user specified by PAM_USER.

The authentication token PAM_OLDAUTHTOK can be used to authenticate the user against repositories that need updating (NIS, LDAP). After successful updates, the new authentication token stored in PAM_AUTHTOK is the user's valid password.

This module honors the PAM_REPOSITORY item, which, if set, specifies which repository is to be updated. If PAM_REPOSITORY is unset, it follows the nsswitch.conf(4).

The following option can be passed to the module:

debug — syslog(3C) debugging information at the LOG_DEBUG level

server_policy — If the account authority for the user, as specified by PAM_USER, is a server, do not encrypt the authentication token before updating.

错误

PAM_SUCCESS — Successfully obtains authentication token

PAM_SYSTEM_ERR — Fails to get username, service name, old password or new password, user name null or empty, or password null.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

pam(3PAM), pam_authenticate(3PAM), pam_chauthtok(3PAM), syslog(3C), libpam(3LIB), pam.conf(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

If the PAM_REPOSITORY *item_type* is set and a service module does not recognize the type, the service module does not process any information, and returns PAM_IGNORE. If the PAM_REPOSITORY *item_type* is not set, a service module performs its default action.

引用名

pam_deny – PAM authentication, account, session and password management PAM module to deny operations

用法概要

```
pam_deny.so.1
```

描述

The pam_deny module implements all the PAM service module functions and returns the module type default failure return code for all calls.

The following options are interpreted:

debug syslog(3C) debugging information at the LOG_AUTH|LOG_DEBUG levels

错误

The following error codes are returned:

PAM_ACCT_EXPIRED	If pam_sm_acct_mgmt is called.
PAM_AUTH_ERR	If pam_sm_authenticate is called.
PAM_AUTHOK_ERR	If pam_sm_chauthtok is called.
PAM_CRED_ERR	If pam_sm_setcred is called.
PAM_SESSION_ERR	If pam_sm_open_session or pam_sm_close_session is called.

示例

示例 1 Disallowing ssh none authentication

The following example is a pam.conf fragment that illustrates how to deny the SSHv2 userauth of “none”:

```
sshd-none      auth        requisite   pam_deny.so.1
sshd-none      account     requisite   pam_deny.so.1
sshd-none      session     requisite   pam_deny.so.1
sshd-none      password    requisite   pam_deny.so.1
```

The equivalent configuration in /etc/pam.d/ would be the following entries in /etc/pam.d/sshd-none:

```
auth       requisite   pam_deny.so.1
account    requisite   pam_deny.so.1
session    requisite   pam_deny.so.1
password   requisite   pam_deny.so.1
```

示例 2 Disallowing any service not explicitly defined

The following example is a pam.conf fragment that illustrates how to deny any PAM service which is not explicitly defined in the PAM configuration:

```
other          auth        requisite   pam_deny.so.1
other          account     requisite   pam_deny.so.1
other          session     requisite   pam_deny.so.1
other          password    requisite   pam_deny.so.1
```

示例 2 Disallowing any service not explicitly defined （续）

The equivalent configuration in /etc/pam.d/ would be the following entries in /etc/pam.d/other:

```
auth      requisite   pam_deny.so.1
account   requisite   pam_deny.so.1
session   requisite   pam_deny.so.1
password  requisite   pam_deny.so.1
```

属性

See attributes(5) for a description of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT-Level	MT-Safe with exceptions

另请参见

su(1M), libpam(3LIB), pam(3PAM), pam_sm_authenticate(3PAM), syslog(3C), pam.conf(4), nsswitch.conf(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5), privileges(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

The pam_deny module is intended to deny access to a specified service. The other service name may be used to deny access to services not explicitly specified.

引用名

pam_dhkeys – authentication Diffie-Hellman keys management module

用法概要

pam_dhkeys.so.1

描述

The pam_dhkeys.so.1 service module provides functionality to two PAM services: Secure RPC authentication and Secure RPC authentication token management.

Secure RPC authentication differs from regular Unix authentication because ONC RPCs use Secure RPC as the underlying security mechanism.

The following options may be passed to the module:

debug	syslog(3C) debugging information at LOG_DEBUG level
nowarn	Turn off warning messages

Authentication Services

If the user has Diffie-Hellman keys, pam_sm_authenticate() establishes secret keys for the user specified by the PAM_USER (equivalent to running keylogin(1)), using the authentication token found in the PAM_AUTHTOK item. If pam_sm_setcred() is called with PAM_ESTABLISH_CRED and the user's secure RPC credentials need to be established, these credentials are set. This is equivalent to running keylogin(1).

If the credentials could not be set and PAM_SILENT is not specified, a diagnostic message is displayed. If pam_setcred() is called with PAM_DELETE_CRED, the user's secure RPC credentials are unset. This is equivalent to running keylogout(1).

PAM_REINITIALIZE_CRED and PAM_REFRESH_CRED are not supported and return PAM_IGNORE.

Authentication Token Management

The pam_sm_chauthtok() implementation checks whether the old login password decrypts the users secret keys. If it doesn't this module prompts the user for an old Secure RPC password and stores it in a pam data item called SUNW_OLDRPCPASS. This data item can be used by the store module to effectively update the users secret keys.

错误

The authentication service returns the following error codes:

PAM_SUCCESS	Credentials set successfully.
PAM_IGNORE	Credentials not needed to access the password repository.
PAM_USER_UNKNOWN	PAM_USER is not set, or the user is unknown.
PAM_AUTH_ERR	No secret keys were set. PAM_AUTHTOK is not set, no credentials are present or there is a wrong password.
PAM_BUF_ERR	Module ran out of memory.

The authentication token management returns the following error codes:

PAM_SUCCESS	Old rpc password is set in SUNW_OLDRPCPASS
PAM_USER_UNKNOWN	User in PAM_USER is unknown.

PAM_AUTHTOK_ERR User did not provide a password that decrypts the secret keys.

PAM_BUF_ERR Module ran out of memory.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

keylogin(1), keylogout(1), pam(3PAM), pam_authenticate(3PAM), pam_chauthtok(3PAM), pam_setcred(3PAM), pam_get_item(3PAM), pam_set_data(3PAM), pam_get_data(3PAM), syslog(3C), libpam(3LIB), pam.conf(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

引用名

pam_dial_auth – authentication management PAM module for dialups

用法概要

```
pam_dial_auth.so.1
```

描述

The pam_dial_auth module implements pam_sm_authenticate(3PAM) which authenticates the user according to the dialups(4) and d_passwd(4) files configuration.

Authentication service modules must implement both pam_sm_authenticate() and pam_sm_setcred(). pam_sm_setcred() in this module always returns PAM_IGNORE.

The value of the PAM_TTY item is checked against entries in dialups(4). If there is a match, the user's shell is compared against entries in d_passwd(4). If there is a matching entry, the user is prompted for a password which is validated against the entry found.

The following option may be passed in to this service module:

debug syslog(3C) debugging information at LOG_DEBUG level.

错误

If dialups(4) is not present, PAM_IGNORE is returned. Upon successful completion of pam_sm_authenticate(), PAM_SUCCESS is returned. The following error codes are returned upon error:

PAM_AUTH_ERR Authentication failure.

PAM_SERVICE_ERR Error in the calling service, PAM_TTY is not set.

PAM_SYSTEM_ERR System error (d_passwd(4) is not present).

PAM_USER_UNKNOWN No account is present for *user*.

属性

See attributes(5) for a description of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT Level	MT-Safe with exceptions

另请参见

pam(3PAM), pam_authenticate(3PAM), pam_sm_authenticate(3PAM), d_passwd(4), dialups(4), libpam(3LIB), pam.conf(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

引用名

pam_krb5 – authentication, account, session, and password management PAM modules for Kerberos V5

用法概要

```
/usr/lib/security/pam_krb5.so.1
```

描述

The Kerberos V5 service module for PAM provides functionality for all four PAM modules: authentication, account management, session management, and password management. The service module is a shared object that can be dynamically loaded to provide the necessary functionality upon demand. Its path is specified in the PAM configuration file.

Kerberos Authentication Module

The Kerberos V5 authentication component provides functions to verify the identity of a user, `pam_sm_authenticate()`, and to manage the Kerberos credentials cache, `pam_sm_setcred()`.

`pam_sm_authenticate()` authenticates a user principal through the Kerberos authentication service. If the authentication request is successful, the authentication service sends a ticket-granting ticket (TGT) back to the service module, which then verifies that the TGT came from a valid Key Distribution Center (KDC) by attempting to get a service ticket for the local host service. For this to succeed, the local host's `keytab` file (`/etc/krb5/krb5.keytab`) must contain the entry for the local host service. For example, in the file `host/`*hostname.com@REALM*, *hostname.com* is the fully qualified local hostname and *REALM* is the default realm of the local host as defined in `/etc/krb5/krb5.conf`. If the host entry is not found in the keytab file, the authentication fails. Administrators can optionally disable this “strict” verification by setting “`verify_ap_req_nofail = false`” in `/etc/krb5/krb5.conf`. See `krb5.conf`(4) for more details on this option. This allows TGT verification to succeed in the absence of a `keytab` host principal entry.

If `pam_sm_authenticate()` is called and the `pkinit` module option is set, the Kerberos V5 authentication module tries to do `PKINIT` authentication, assuming that both the system and the `KDC` are configured to support this type of authentication. This form of authentication uses a user's certificate and private key to acquire the user's initial Kerberos credential (`TGT`). One of the keystore formats supported is `PKCS11` which supports use of any `PKCS11` compatible keystore capable of storing the required credential and private key needed for `PKINIT` authentication (`PKCS11` compatible smartcards are an example). See `krb5.conf`(4) for more details on `PKINIT` configuration. This form of authentication is typically useful for services where the system on which the `auth` stack is being processed has access to the user's certificate and private key.

If `pam_sm_authenticate()` is called and the `pkinit` module option is not set then the Kerberos V5 authentication module does password based authentication.

In either case, if the `PAM_AUTHTOK` password item has been set when `pam_sm_authenticate()` is called, which is the case when `pam_krb5` is stacked after `pam_authtok_get` in the `auth` stack, the Kerberos V5 authentication module uses that `PAM_AUTHTOK` password for either `PKINIT` or password based Kerberos authentication.

If the `PAM_USER` item is not set `pam_krb5` with the `pkinit` option prompts for and set that item.

If the PAM_AUTHTOK password item has not been set when pam_sm_authenticate() is called, which is the case when pam_krb5 is stacked before pam_authtok_get in the auth stack, and the pkinit option is present the Kerberos V5 authentication module allows the Kerberos pkinit preauth plugin to prompt for whatever information is needed to perform PKINIT (typically this is for the user's PIN). No PAM items are set by way of this prompting. See krb5.conf(4) for more information on PKINIT configuration options.

The pam_krb5 module sets the KRB5CCNAME shell environment variable upon successful authentication or password change to FILE:/tmp/krb5cc_*uid* where *uid* is the UID of the user that pam_krb5 authenticated. KRB5CCNAME is documented in krb5envvar(5).

If it is desirable to initially have the Kerberos V5 authentication module try PKINIT Kerberos authentication and fall back to password based Kerberos authentication then either the sufficient or optional control flags must be provided for the instance of pam_krb5 with the pkinit module option set and another instance of pam_krb5 without the pkinit module option must be stacked below pam_authtok_get. If there are PAM modules other than pam_krb5 that must be evaluated below pam_authtok_get then the control flag should be set to optional for the instance of pam_krb5 with the pkinit module option set otherwise the control flag should be set to sufficient.

Only two instances of pam_krb5 are supported in a auth stack.

pam_sm_authenticate(3PAM) can be passed the following flag:

PAM_DISALLOW_NULL_AUTHTOK
: This flag is ignored. The Kerberos authentication mechanism does not allow an empty password string by default.

pam_sm_setcred() creates and modifies the user's credential cache. This function initializes the user's credential cache, if it does not already exist, and stores the initial credentials for later use by Kerberos network applications. The following flags can be set in the flags field. They are best described by their effect on the user's credential cache.

PAM_ESTABLISH_CRED
: Stores the initial credentials in the user's credential cache so that the user can access Kerberos network services. If a successful authentication pass was made, the new credentials are stored in the credential cache, overwriting any existing credentials that were previously stored. If an unsuccessful authentication pass was made, PAM_CRED_UNAVAIL is returned.

PAM_DELETE_CRED
: This flag has no effect on the credential cache and always returns PAM_SUCCESS. The credential cache is not deleted because there is no accurate method to determine if the credentials are needed by another process. The credential cache can be deleted with the kdestroy(1) command.

PAM_REINITIALIZE_CRED
Deletes the user's existing credential cache, if it exists, and creates a new credential cache. The new credentials are stored in the new cache and the user's ticket lifetime and renewable life time values are reset.

PAM_REFRESH_CRED
Does not require a previous authentication pass, but if a successful one is made, the new credentials are stored in the credential cache. If a previous authentication pass was not made or was unsuccessful, an attempt to renew the existing credentials is made. This function fails if the user's renewable ticket lifetime is expired.

The following options can be passed to the Kerberos V5 authentication module:

debug Provides syslog(3C) debugging information at LOG_DEBUG level.

nowarn Turns off warning messages.

pkinit Indicates that the Kerberos V5 authentication module should try Kerberos PKINIT authentication instead of the default password based Kerberos authentication.

Kerberos V5 Account Management Module

The Kerberos account management component provides a function to perform account management, pam_sm_acct_mgmt(). This function checks to see if the pam_krb5 authentication module has noted that the user's password has not expired. The following options can be passed in to the Kerberos V5 account management module:

debug Provides syslog(3C) debugging information at LOG_DEBUG level

nowarn Turns off warning messages. Also, does not query KDC for impending password expiration information used to warn the user.

Kerberos V5 Session Management Module

The Kerberos V5 session management component provides functions to initiate pam_sm_open_session() and terminate pam_sm_close_session() Kerberos sessions. For Kerberos V5, both pam_sm_open_session and pam_sm_close_session() are null functions, returning PAM_IGNORE.

Kerberos V5 Password Management Module

The Kerberos V5 password management component provides a function to change passwords, pam_sm_chauthtok(), in the Key Distribution Center (KDC) database.

If the Kerberos V5 authentication module used PKINIT authentication in the auth stack then the Kerberos V5 password management module returns PAM_IGNORE in the following cases:

- The new password is NULL.
- The old password is NULL.
- Verification of the old password fails.

The rationale behind this is that the KDC can not allow a PKINIT user to change/set a password since the user can be expected to use PKINIT only. If all of the cases above are false the Kerberos V5 password management module tries to change the user's password in the KDC database.

If the KDC only supports PKINIT authentication then the Kerberos V5 password management module should not be present in any password stacks.

Related to PKINIT the Kerberos V5 password management module does not support changing the key store PIN used to access a user's private key and certificate.

The following flags can be passed to pam_sm_chauthtok(3PAM):

PAM_CHANGE_EXPIRED_AUTHTOK
The password service should only update the user's Kerberos password if it is expired. Otherwise, this function returns PAM_IGNORE. The default behaviour is to always change the user's Kerberos password.

PAM_PRELIM_CHECK
This is a null function that always returns PAM_IGNORE.

PAM_UPDATE_AUTHTOK
This flag is necessary to change the user's Kerberos password. If this flag is not set, pam_krb5 returns PAM_SYSTEM_ERR.

The following option can be passed to the Kerberos V5 password module:

debug Provides syslog(3C) debugging information at LOG_DEBUG level.

错误

The following error codes are returned for pam_sm_authenticate():

PAM_AUTH_ERR	Authentication failure
PAM_BUF_ERR	Memory buffer error.
PAM_IGNORE	The user is "root" and the root key exists in the default keytab.
PAM_SUCCESS	Successfully obtained Kerberos credentials .
PAM_SYSTEM_ERR	System error.
PAM_USER_UNKNOWN	An unknown Kerberos principal was requested.

The following error codes are returned for pam_sm_setcred():

PAM_AUTH_ERR	Authentication failure.
PAM_BUF_ERR	Memory buffer error.
PAM_IGNORE	The user is "root" and the root key exists in the default keytab.
PAM_SYSTEM_ERR	System error.
PAM_SUCCESS	Successfully modified the Kerberos credential cache.

The following error codes are returned for pam_sm_acct_mgmt():

PAM_AUTH_ERR	Authentication failure.

`PAM_IGNORE` Kerberos service module `pam_sm_authenticate()` was never called, or the user is “`root`” and the root key exists in the default keytab.

`PAM_NEW_AUTHTOK_REQD` Obtain new authentication token from the user.

`PAM_SERVICE_ERR` Error in underlying service module.

`PAM_SUCCESS` Kerberos principal account is valid.

`PAM_SYSTEM_ERR` System error.

`PAM_USER_UNKNOWN` An unknown Kerberos principal was requested.

The following error code is returned for `pam_sm_open_session()` and `pam_sm_close_session()`:

`PAM_IGNORE` These two functions are null functions in `pam_krb5`:

The following error codes are returned for `pam_sm_chauthtok()`:

`PAM_AUTH_ERR` Authentication failure.

`PAM_IGNORE` The user has not been authenticated by Kerberos service module `pam_sm_authenticate()`, or the user is “`root`” and the root key exists in the default keytab.

`PAM_NEW_AUTHTOK_REQD` User's Kerberos password has expired.

`PAM_SERVICE_ERR` Error in module. At least one input parameter is missing.

`PAM_SYSTEM_ERR` System error.

`PAM_USER_UNKNOWN` An unknown Kerberos principal was requested.

`PAM_SUCCESS` Successfully changed the user's Kerberos password.

示例

示例 1 Authenticating Users Through Kerberos as First Choice Using Password-based Authentication

The following is an excerpt of a sample `pam.conf` configuration file that authenticates users through the Kerberos authentication service and authenticates through the Unix login only if the Kerberos authentication fails. This arrangement is helpful when a majority of the users are networked by means of Kerberos and when there are only a few non-Kerberos type user accounts, such as root. The service illustrated below is for `gdm`.

```
gdm auth requisite          pam_authtok_get.so.1
gdm auth required           pam_dhkeys.so.1
gdm auth required           pam_unix_cred.so.1
gdm auth sufficient         pam_krb5.so.1
gdm auth required           pam_unix_auth.so.1
```

示例 1 Authenticating Users Through Kerberos as First Choice Using Password-based Authentication （续）

These changes should not be made to the existing krlogin, krsh, and ktelnet service entries. Those services require Kerberos authentication, so using a seemingly sufficient control flag would not provide the necessary functionality for privacy and integrity. There should be no need to change those entries.

The following entries check for password expiration when dealing with Kerberos and Unix password aging policies:

```
other   account requisite      pam_roles.so.1
other   account required       pam_unix_account.so.1
other   account required       pam_krb5.so.1
```

The following entries would change the Kerberos password of the user and continue to change the Unix login password only if the Kerberos password change had failed:

```
other   password required      pam_dhkeys.so.1
other   password requisite     pam_authtok_get.so.1
other   password requisite     pam_authtok_check.so.1
other   password sufficient    pam_krb5.so.1
other   password required      pam_authtok_store.so.1
```

示例 2 Authenticating Users Through Kerberos Only Using Password-based Authentication

The following example allows authentication only to users that have Kerberos-based accounts.

```
gdm auth requisite         pam_authtok_get.so.1
gdm auth required          pam_dhkeys.so.1
gdm auth required          pam_unix_cred.so.1
gdm auth required          pam_krb5.so.1
```

Typically, you would have another service specified in the pam.conf file that would allow local users, such as database, web server, system administrator accounts, to log in to the host machine. For example, the service name “login” could be used for these users. These users should not belong to any roles.

The rest of the module types look similar to that shown in the previous example:

```
other   account requisite      pam_roles.so.1
other   account required       pam_unix_account.so.1
other   account required       pam_krb5.so.1
```

With binding specified in the following, it is important that non-Kerberos users specify the repository in which they reside using the -r option with the passwd(1) command. This configuration is also based on the assumptions that:

- Kerberos users maintain only their Kerberos passwords;

示例 2 Authenticating Users Through Kerberos Only Using Password-based Authentication （续）

- changing their Unix password is not necessary, given that they are authenticated only through their Kerberos passwords when logging in.

```
other   password required      pam_dhkeys.so.1
other   password requisite     pam_authtok_get.so.1
other   password requisite     pam_authtok_check.so.1
other   password binding       pam_krb5.so.1
```

示例 3 Authenticating Through Kerberos Optionally Using Password-based Authentication

This configuration is helpful when the majority of users are non-Kerberos users and would like to authenticate through Kerberos if they happened to exist in the Kerberos database. The effect of this is similar to users voluntarily executing kinit(1) after they have successfully logged in:

```
gdm auth requisite          pam_authtok_get.so.1
gdm auth required           pam_dhkeys.so.1
gdm auth required           pam_unix_cred.so.1
gdm auth required           pam_unix_auth.so.1
gdm auth optional           pam_krb5.so.1
```

The rest of the configuration is as follows:

```
other   account requisite      pam_roles.so.1
other   account required       pam_unix_account.so.1
other   account optional       pam_krb5.so.1

other   password required      pam_dhkeys.so.1
other   password requisite     pam_authtok_get.so.1
other   password requisite     pam_authtok_check.so.1
other   password required      pam_authtok_store.so.1
other   password optional      pam_krb5.so.1
```

Non-Kerberos users should specify their respective repositories by using the -r option when changing their password with the passwd(1) command.

示例 4 Authenticating Users Through Kerberos PKINIT as First Choice

The following is an excerpt of a sample pam.conf configuration file that authenticates users through the Kerberos authentication service and authenticates through the Unix login only if the Kerberos authentication (using PKINIT) fails. This arrangement is helpful when a majority of the users are networked by means of Kerberos and when there are only a few non-Kerberos type user accounts, such as root. The service illustrated below is for login. The user is prompted once for the PIN by pam_krb5.

示例 4 Authenticating Users Through Kerberos PKINIT as First Choice （续）

```
login auth required           pam_unix_cred.so.1
login auth sufficient         pam_krb5.so.1 pkinit
login auth requisite          pam_authtok_get.so.1
login auth required           pam_dhkeys.so.1
login auth required           pam_unix_auth.so.1
```

示例 5 Authenticating Users Through Kerberos PKINIT Only

The following example allows authentication only to users that have kerberos-based accounts requiring PKINIT authentication.

```
login auth required           pam_unix_cred.so.1
login auth required           pam_krb5.so.1 pkinit
```

示例 6 Authenticating Users Through Kerberos PKINIT Optionally

The following example allows users to acquire a Kerberos credential using PKINIT authentication if they have a Kerberos account. Whether pam_krb5 succeeds or fails the user must provide their Unix password to login.

```
login auth required           pam_unix_cred.so.1
login auth optional           pam_krb5.so.1 pkinit
login auth requisite          pam_authtok_get.so.1
login auth required           pam_unix_auth.so.1
```

示例 7 Authenticating Users Through Kerberos PKINIT as a Requirement

The following example allows users to login if pam_krb5 is able to acquire a Kerberos credential using PKINT authentication and in addition must provide their Unix password to pam_unix_auth.

```
login auth required           pam_unix_cred.so.1
login auth required           pam_krb5.so.1 pkinit
login auth requisite          pam_authtok_get.so.1
login auth required           pam_unix_auth.so.1
```

示例 8 Authenticating Users Through Kerberos PKINIT as a Requirement

The following example allows users to login using their PAM_AUTHTOK password acquired by pam_authtok_get. This password is used by pam_krb5 to try PKINIT authentication and is also used by pam_unix_auth to authenticate the user using the user's Unix account. If PKINIT requires a password/PIN that differs from the user's Unix password then pam_krb5 must be stacked above pam_authtok_get.

```
login auth required           pam_unix_cred.so.1
login auth requisite          pam_authtok_get.so.1
```

示例 8 Authenticating Users Through Kerberos PKINIT as a Requirement （续）

```
login auth required            pam_krb5.so.1 pkinit
login auth required            pam_unix_auth.so.1
```

示例 9 Authenticating Users Through Kerberos PKINIT with a Fall Back to Password-based krb auth

The following example allows users to acquire a Kerberos credential using PKINIT authentication or using password based authentication if PKINIT fails. If PKINIT succeeds the user is not prompted for their password. If pam_krb5 PKINIT succeeds, the second instance of pam_krb5 does not try password authentication and returns success. If PKINIT fails the user is prompted for their Kerberos password.

```
login auth required            pam_unix_cred.so.1
login auth sufficient          pam_krb5.so.1 pkinit
login auth requisite           pam_authtok_get.so.1
login auth required            pam_krb5.so.1
```

示例 10 Authenticating Users Through Kerberos Requiring Users to Authenticate Either through Kerberos PKINIT or Fall Back to Password-based krb auth

The following example allows users to acquire a Kerberos credential using PKINIT authentication or using password based authentication if PKINIT fails. If pam_krb5 PKINIT succeeds, the second instance of pam_krb5 does not try password authentication and returns ignore. If pam_krb5 PKINIT fails the second instance of pam_krb5 tries password based authentication and return success or failure.

```
login auth required            pam_unix_cred.so.1
login auth optional            pam_krb5.so.1 pkinit
login auth requisite           pam_authtok_get.so.1
login auth required            pam_krb5.so.1
login auth required            pam_dhkeys.so.1
login auth required            pam_unix_auth.so.1
```

示例 11 Authenticating Users Through Kerberos Requiring Users to Authenticate Either through Kerberos PKINIT or Fall Back to pam_pkcs11

The following example allows users to acquire a Kerberos credential using PKINIT authentication or if that fails use pam_pkcs11 to validate the user's PIN using their certificate and private key.

```
login auth required            pam_unix_cred.so.1
login auth sufficient          pam_krb5.so.1 pkinit
login auth sufficient          pam_pkcs11.so
```

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed

另请参见

kdestroy(1), kinit(1), passwd(1), ktkt_warnd(1M), libpam(3LIB), pam(3PAM), pam_sm(3PAM), pam_sm_acct_mgmt(3PAM), pam_sm_authenticate(3PAM), pam_sm_chauthtok(3PAM), pam_sm_close_session(3PAM), pam_sm_open_session(3PAM), pam_sm_setcred(3PAM), syslog(3C), krb5.conf(4), pam.conf(4), attributes(5), kerberos(5), krb5envvar(5), pam_krb5_migrate(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

On successful acquisition of initial credentials (ticket-granting ticket), ktkt_warnd(1M) is notified, to alert the user when the initial credentials are about to expire.

引用名

pam_krb5_migrate – authentication PAM module for the KerberosV5 auto-migration of users feature

用法概要

`/usr/lib/security/pam_krb5_migrate.so.1`

描述

The KerberosV5 auto-migrate service module for PAM provides functionality for the PAM authentication component. The service module helps in the automatic migration of `PAM_USER` to the client's local Kerberos realm, using `PAM_AUTHTOK` (the PAM authentication token associated with `PAM_USER`) as the new Kerberos principal's password.

KerberosV5 Auto-migrate Authentication Module

The KerberosV5 auto-migrate authentication component provides the `pam_sm_authenticate`(3PAM) function to migrate a user who does not have a corresponding `krb5` principal account to the default Kerberos realm of the client.

`pam_sm_authenticate`(3PAM) uses a host-based client service principal, present in the local `keytab` (`/etc/krb5/krb5.keytab`) to authenticate to `kadmind`(1M) (defaults to the `host/nodename.fqdn` service principal), for the principal creation operation. Also, for successful creation of the `krb5` user principal account, the host-based client service principal being used needs to be assigned the appropriate privilege on the master KDC's `kadm5.acl`(4) file. `kadmind`(1M) checks for the appropriate privilege and validates the user password using PAM by calling `pam_authenticate`(3PAM) and `pam_acct_mgmt`(3PAM) for the `k5migrate` service.

If migration of the user to the KerberosV5 infrastructure is successful, the module will inform users about it by means of a `PAM_TEXT_INFO` message, unless instructed otherwise by the presence of the `quiet` option.

The authentication component always returns `PAM_IGNORE` and is meant to be stacked in the PAM configuration (see `pam.conf`(4)) with a requirement that it be listed below `pam_authtok_get`(5) in the authentication stack. Also, if `pam_krb5_migrate` is used in the authentication stack of a particular service, it is mandatory that `pam_krb5`(5) be listed in the PAM account stack of that service for proper operation (see EXAMPLES).

选项

The following options can be passed to the KerberosV5 auto-migrate authentication module:

`debug`

Provides `syslog`(3C) debugging information at `LOG_DEBUG` level.

`client_service=`<*service name*>

Name of the service used to authenticate to `kadmind`(1M) defaults to `host`. This means that the module uses `host/`<*nodename.fqdn*> as its client service principal name, KerberosV5 user principal creation operation or <*service*>/<*nodename.fqdn*> if this option is provided.

`quiet`

Do not explain KerberosV5 migration to the user.

This has the same effect as passing the PAM_SILENT flag to pam_sm_authenticate(3PAM) and is useful where applications cannot handle PAM_TEXT_INFO messages.

If not set, the authentication component will issue a PAM_TEXT_INFO message after creation of the Kerberos V5 principal, indicating that it has done so.

expire_pw
: Causes the creation of KerberosV5 user principals with password expiration set to now (current time).

示例

示例 1 Sample PAM entries for the pam_krb5_migrate.so.1 module

The following pam.conf(4) entries demonstrate the use of the pam_krb5_migrate.so.1 module:

```
login       auth requisite          pam_authtok_get.so.1
login       auth required           pam_dhkeys.so.1
login       auth required           pam_unix_cred.so.1
login       auth sufficient         pam_krb5.so.1
login       auth requisite          pam_unix_auth.so.1
login       auth optional           pam_krb5_migrate.so.1 expire_pw
login       auth required           pam_dial_auth.so.1

other    account requisite       pam_roles.so.1
other    account required        pam_krb5.so.1
other    account required        pam_unix_account.so.1
```

The equivalent PAM configuration in /etc/pam.d/ would be the following entries in /etc/pam.d/login:

```
auth    requisite          pam_authtok_get.so.1
auth    required           pam_dhkeys.so.1
auth    required           pam_unix_cred.so.1
auth    sufficient         pam_krb5.so.1
auth    requisite          pam_unix_auth.so.1
auth    optional           pam_krb5_migrate.so.1 expire_pw
auth    required           pam_dial_auth.so.1
```

and the following entries in /etc/pam.d/other:

```
account requisite              pam_roles.so.1
account required               pam_krb5.so.1
account required               pam_unix_account.so.1
```

The pam_krb5_migrate module can generally be present on the authentication stack of any service where the application calls pam_sm_authenticate(3PAM) and an authentication token (in the preceding example, the authentication token would be the user's Unix password) is available for use as a Kerberos V5 password.

示例 2 Sample Entries from kadm5.acl

The following entries from kadm5.acl(4) permit or deny privileges to the host client service principal:

```
host/*@ACME.COM U root
host/*@ACME.COM ui *
```

The preceding entries permit the pam_krb5_migrate add privilege to the host client service principal of any machine in the ACME.COM KerberosV5 realm, but denies the add privilege to all host service principals for addition of the root user account.

示例 3 Sample PAM entries for the Master KDC

The /etc/pam.conf entries below enable kadmind(1M) on the master KDC to use the k5migrate PAM service in order to validate Unix user passwords for accounts that require migration to the Kerberos realm.

```
k5migrate        auth    required        pam_unix_auth.so.1
k5migrate        account required        pam_unix_account.so.1
```

The equivalent PAM configuration in /etc/pam.d/ would be the following entries in /etc/pam.d/k5migrate:

```
auth    required        pam_unix_auth.so.1
account required        pam_unix_account.so.1
```

属性

See attributes(5) for a description of the following attribute:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed

另请参见

kadmind(1M), syslog(3C), pam_authenticate(3PAM), pam_acct_mgmt(3PAM), pam_sm_authenticate(3PAM), kadm5.acl(4), pam.conf(4), attributes(5), pam_authtok_get(5), pam_krb5(5)

引用名

pam_ldap – authentication and account management PAM module for LDAP

用法概要

`/usr/lib/security/pam_ldap.so.1`

描述

The `pam_ldap` module implements `pam_sm_authenticate`(3PAM) and `pam_sm_acct_mgmt`(3PAM), the functions that provide functionality for the PAM authentication and account management stacks. The `pam_ldap` module ties the authentication and account management functionality to the functionality of the supporting LDAP server. For authentication, `pam_ldap` can authenticate the user directly to any LDAP directory server by using any supported authentication mechanism, such as `DIGEST-MD5`. However, the account management component of `pam_ldap` will work only with the Sun Java System Directory Server. The server's user account management must be properly configured before it can be used by `pam_ldap`. Refer to the Sun Java System Directory Server Administration Guide for information on how to configure user account management, including password and account lockout policy.

`pam_ldap` must be used in conjunction with the modules that support the UNIX authentication, password, and account management, which are `pam_authtok_get`(5), `pam_passwd_auth`(5), `pam_unix_account`(5), and `pam_unix_auth`(5). `pam_ldap` is designed to be stacked directly below these modules. If other modules are designed to be stacked in this manner, the modules can be stacked below the `pam_ldap` module. The 示例 section shows how the UNIX modules are stacked with `pam_ldap`. When stacked together, the UNIX modules are used to control local accounts, such as `root`. `pam_ldap` is used to control network accounts, that is, LDAP users. For the stacks to work, `pam_unix_auth`, `pam_unix_account`, and `pam_passwd_auth` must be configured with the `binding` control flag and the `server_policy` option. This configuration allows local account override of a network account.

LDAP Authentication Module

The LDAP authentication module verifies the identity of a user. The `pam_sm_authenticate`(3PAM) function uses the password entered by the user to attempt to authenticate to the LDAP server. If successful, the user is authenticated. See NOTES for information on password prompting.

The authentication method used is either defined in the client profile , or the authentication method is configured by using the `ldapclient`(1M) command. To determine the authentication method to use, this module first attempts to use the authentication method that is defined, for service `pam_ldap`, for example, `serviceAuthenticationMethod:pam_ldap:sasl/DIGEST-MD5`. If no authentication method is defined, `pam_ldap` uses the default authentication method. If neither are set, the authentication fails. This module skips the configured authentication method if the authentication method is set to `none`.

The following options can be passed to the LDAP service module:

`debug` `syslog`(3C) debugging information at `LOG_DEBUG` level.

`nowarn` Turn off warning messages.

These options are case sensitive and must be used exactly as presented here.

LDAP Account Management Module

The LDAP account management module validates the user's account. The pam_sm_acct_mgmt(3PAM) function authenticates to the LDAP server to verify that the user's password has not expired, or that the user's account has not been locked. In the event that there is no user authentication token (PAM_AUTHTOK) available, the pam_sm_acct_mgmt(3PAM) function attempts to retrieve the user's account status without authenticating to the LDAP server as the user logging in. This procedure will succeed only if the LDAP server is Sun Java System Directory server 5.2 patch 4 or newer. The following options can be passed to the LDAP service module:

debug	syslog(3C) debugging information at LOG_DEBUG level.
nowarn	Turn off warning messages.

These options are case sensitive, and the options must be used exactly as presented here.

LDAP Password Management Module

LDAP password management is no longer supported by pam_ldap. Use pam_authtok_store(5) instead of pam_ldap for password change. pam_authtok_store(5) handles both the local and LDAP accounts and updates the passwords in all the repositories configured by nsswitch.conf(4).

错误

The authentication service returns the following error codes:

PAM_SUCCESS	The uthentication was successful.
PAM_MAXTRIES	The maximum number of authentication attempts was exceeded.
PAM_AUTH_ERR	The authentication failed.
PAM_USER_UNKNOWN	No account is present for the user.
PAM_BUF_ERR	A memory buffer error occurred.
PAM_SYSTEM_ERR	A system error occurred.
PAM_IGNORE	The user's account was inactivated.

The account management service returns the following error codes:

PAM_SUCCESS	The user was allowed access to the account.
PAM_NEW_AUTHTOK_REQD	A new authentication token is required.
PAM_ACCT_EXPIRED	The user account has expired.
PAM_PERM_DENIED	The user was denied access to the account at this time.
PAM_USER_UNKNOWN	No account is present for the user.
PAM_BUF_ERROR	A memory buffer error occurred.
PAM_SYSTEM_ERR	A system error occurred.

示例

示例 1 Using pam_ldap With Authentication

The following is a configuration for the login service when using pam_ldap. The service name login can be substituted for any other authentication service such as dtlogin or su. Lines that begin with the # symbol are comments and are ignored.

```
# Authentication management for login service is stacked.
# If pam_unix_auth succeeds, pam_ldap is not invoked.
# The control flag "binding" provides a local overriding
# remote (LDAP) control. The "server_policy" option is used
# to tell pam_unix_auth.so.1 to ignore the LDAP users.

login   auth requisite  pam_authtok_get.so.1
login   auth required   pam_dhkeys.so.1
login   auth required   pam_unix_cred.so.1
login   auth binding    pam_unix_auth.so.1 server_policy
login   auth required   pam_ldap.so.1
```

示例 2 Using pam_ldap With Account Management

The following is a configuration for account management when using pam_ldap. Lines that begin with the # symbol are comments and are ignored.

```
# Account management for all services is stacked
# If pam_unix_account succeeds, pam_ldap is not invoked.
# The control flag "binding" provides a local overriding
# remote (LDAP) control. The "server_policy" option is used
# to tell pam_unix_account.so.1 to ignore the LDAP users.

other   account  requisite      pam_roles.so.1
other   account  binding        pam_unix_account.so.1 server_policy
other   account  required       pam_ldap.so.1
```

示例 3 Using pam_authtok_store With Password Management For Both Local and LDAP Accounts

The following is a configuration for password management when using pam_authtok_store. Lines that begin with the # symbol are comments and are ignored.

```
# Password management (authentication)
# The control flag "binding" provides a local overriding
# remote (LDAP) control. The server_policy option is used
# to tell pam_passwd_auth.so.1 to ignore the LDAP users.

passwd  auth binding  pam_passwd_auth.so.1 server_policy
passwd  auth required pam_ldap.so.1

# Password management (updates)
# This updates passwords stored both in the local /etc
# files and in the LDAP directory. The "server_policy"
# option is used to tell pam_authtok_store to
```

示例 3　Using pam_authtok_store With Password Management For Both Local and LDAP Accounts　（续）

```
# follow the LDAP server's policy when updating
# passwords stored in the LDAP directory

other password required   pam_dhkeys.so.1
other password requisite  pam_authtok_get.so.1
other password requisite  pam_authtok_check.so.1
other password required   pam_authtok_store.so.1 server_policy
```

文件

/var/ldap/ldap_client_file
/var/ldap/ldap_client_cred　　The LDAP configuration files of the client. Do not manually modify these files, as these files might not be human readable. Use ldapclient(1M) to update these files.

/etc/pam.conf　　PAM configuration file.

/etc/pam.d/*service*　　Alternate PAM configuration files.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT-Level	MT-Safe with exceptions

另请参见

idsconfig(1M), ldap_cachemgr(1M), ldapclient(1M), libpam(3LIB), pam(3PAM), pam_sm_acct_mgmt(3PAM), pam_sm_authenticate(3PAM), pam_sm_chauthtok(3PAM), pam_sm_close_session(3PAM), pam_sm_open_session(3PAM), pam_sm_setcred(3PAM), syslog(3C), pam.conf(4), attributes(5), ldap(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

The previously supported use_first_pass and try_first_pass options are obsolete in this version, are no longer needed, can safely be removed from pam.conf(4), and are silently ignored. They might be removed in a future release. Password prompting must be provided for by stacking pam_authtok_get(5) before pam_ldap in the auth and password module stacks and pam_passwd_auth(5) in the passwd service auth stack (as described in the EXAMPLES section). The previously supported password update function is replaced in this release by the previously recommended use of pam_authtok_store with the server_policy option (as described in the EXAMPLES section).

The functions: pam_sm_setcred(3PAM), pam_sm_chauthtok(3PAM), pam_sm_open_session(3PAM), and pam_sm_close_session(3PAM) do nothing and return PAM_IGNORE in pam_ldap.

引用名

pam_list – PAM account management module for UNIX

用法概要

```
pam_list.so.1
```

描述

The pam_list module implements pam_sm_acct_mgmt(3PAM), which provides functionality to the PAM account management stack. The module provides functions to validate that the user's account is valid on this host based on a list of users and/or netgroups in the given file. The users and netgroups are separated by newline character. Netgroups are specified with character '@' as prefix before name of netgroup in the list. The maximum line length is 1023 characters.

The username is the value of PAM_USER. The host is the value of PAM_RHOST or, if PAM_RHOST is not set, the value of the localhost as returned by gethostname(3C) is used.

If neither of the allow, deny, or compat options are specified, the module will look for +/- entries in the local /etc/passwd file. If this style is used, nsswitch.conf(4) must not be configured with compat for the passwd database. If no relevant +/- entry exists for the user, pam_list is not participating in result.

If compat option is specified then the module will look for +/- entries in the local /etc/passwd file. Other entries in this file will be counted as + entries. If no relevant entry exits for the user, pam_list will deny the access.

The following options can be passed to the module:

allow=
The full pathname to a file of allowed users and/or netgroups. Only one of allow= or deny= can be specified.

compat
Activate compat mode.

deny=
The full pathname to a file of denied users and/or netgroups. Only one of deny= or allow= can be specified.

debug
Provide syslog(3C) debugging information at the LOG_AUTH | LOG_DEBUG level.

user
The module should only perform netgroup matches on the username. This is the default option.

nouser
The username should not be used in the netgroup match.

host
Only the host should be used in netgroup matches.

nohost
The hostname should not be used in netgroup matches.

norole
Return PAM_IGNORE if the account (PAM_USER) is a role. This is the default.

role
Evaluate the rules even if PAM_USER is a role account.

user_host_exact
The user and hostname must be in the same netgroup.

错误

The following error values are returned:

PAM_SERVICE_ERR	An invalid set of module options was specified in the PAM configuration (seepam.conf(4)) for this module, or the user/netgroup file could not be opened.
PAM_BUF_ERR	A memory buffer error occurred.
PAM_IGNORE	The module is ignored, as it is not participating in the result.
PAM_PERM_DENIED	The user is not on the allow list or is on the deny list.
PAM_SUCCESS	The account is valid for use at this time.
PAM_USER_UNKNOWN	No account is present for the user

示例

示例 1 Using pam_list in default mode

The changes to /etc/pam.conf would be:

```
other   account requisite       pam_roles.so.1
other   account required        pam_unix_account.so.1
other   account required        pam_list.so.1
```

The equivalent PAM configuration in /etc/pam.d/ would be the following entries in /etc/pam.d/other:

```
account requisite       pam_roles.so.1
account required        pam_unix_account.so.1
account required        pam_list.so.1
```

In the case of default mode or compat mode, the important lines in /etc/passwd appear as follows:

```
+loginname    - user is approved
-loginname    - user is disapproved
+@netgroup    - netgroup members are approved
-@netgroup    - netgroup members are disapproved
```

示例 2 Using pam_list with allow file

The changes to /etc/pam.conf would be:

```
other   account requisite       pam_roles.so.1
other   account required        pam_unix_account.so.1
other   account required        pam_list.so.1 allow=/etc/users.allow
```

The equivalent PAM configuration in /etc/pam.d/ would be the following entries in /etc/pam.d/other:

```
account requisite       pam_roles.so.1
account required        pam_unix_account.so.1
```

示例 2 Using pam_list with allow file （续）

```
account required        pam_list.so.1 allow=/etc/users.allow
```

/etc/users.allow contains:

```
root
localloginname
remoteloginname
@netgroup
```

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT-Level	MT-Safe with exceptions

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multithreaded application uses its own PAM handle.

另请参见

pam(3PAM), pam_authenticate(3PAM), pam_sm_acct_mgmt(3PAM), syslog(3C), libpam(3LIB), nsswitch.conf(4), pam.conf(4), attributes(5)

引用名

pam_passwd_auth – authentication module for password

用法概要

```
pam_passwd_auth.so.1
```

描述

pam_passwd_auth provides authentication functionality to the password service as implemented by passwd(1). It differs from the standard PAM authentication modules in its prompting behavior. It should be the first module on the password service authentication stack.

The name of the user whose password attributes are to be updated must be present in the PAM_USER item. This can be accomplished due to a previous call to pam_start(3PAM), or explicitly set by pam_set_item(3PAM). Based on the current user-id and the repository that is to by updated, the module determines whether a password is necessary for a successful update of the password repository, and if so, which password is required.

The following options can be passed to the module:

debug
: syslog(3C) debugging information at the LOG_DEBUG level

nowarn
: Turn off warning messages

server_policy
: If the account authority for the user, as specified by PAM_USER, is a server, do not apply the Unix policy from the passwd entry in the name service switch.

错误

The following error codes are returned:

PAM_BUF_ERR	Memory buffer error
PAM_IGNORE	Ignore module, not participating in result
PAM_SUCCESS	Successfully obtains authentication token
PAM_SYSTEM_ERR	System error

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

passwd(1), pam(3PAM), pam_authenticate(3PAM), pam_start(3PAM), pam_set_item(3PAM), syslog(3C), libpam(3LIB), pam.conf(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

This module relies on the value of the current real UID, this module is only safe for MT-applications that don't change UIDs during the call to pam_authenticate(3PAM).

引用名

pam_pkcs11 – PAM Authentication Module for the PKCS#11 token libraries

用法概要

`pam_pkcs11.so [debug] [config_file=`*filename*`]`

描述

The `pam_pkcs11` module implements `pam_sm_authenticate`(3PAM), which provides functionality to the PAM authentication stack. This module allows a user to login a system, using a X.509 certificate and its dedicated private key stored in a PKCS#11 token. This module currently supports the RSA algorithm only.

To verify the dedicated private key is truly associated with the X.509 certificate, the following verification procedure is performed in this module by default:

- Generate 128 random byte data
- Sign the random data with the private key and get a signature. This step is done in the PKCS#11 token.
- Verify the signature using the public key extracted from the certificate.

For the verification of the users' certificates, locally stored CA certificates as well as either online or locally accessible CRLs are used.

PAM CONFIGURATION

The `pam_pkcs11.so` service module can be used in the `<auth>` PAM chain. The program that needs a PAM service should be configured in `/etc/pam.conf` or `/etc/pam.d/`*service*. For details on how to configure PAM services, see `pam.conf`(4).

The following example uses only `pam_pkcs11` for authentication:

```
login auth requisite pam_pkcs11.so.1
login autho required pam_unix_cred.so.1
```

The following example uses `pam_pkcs11` for authentication with fallback to standard UNIX authentication:

```
login auth sufficient pam_pkcs11.so.1
login auth requisite  pam_authtok_get.so.1
login auth required   pam_dhkeys.so.1
login auth required   pam_unix_cred.so.1
login auth required   pam_unix_auth.so.1
```

PAM_PKCS11 CONFIGURATION

To configure the `pam_pkcs11` module, you must have the following information:

- Which PKCS #11 token you are going to use
- Which mapper(s) you need, and if needed, how to create and edit the related mapping files
- The root Certificate Authority files, and if required, the Certificate Revocation Lists files
- The list of authorized users to login, and their corresponding certificates

To configure the `pam_pkcs11` module, you need to modify the `pam_pkcs11.conf` configuration file which is in the `/etc/security/pam_pkcs11` directory by default. For detailed information on how to configure the pam_pkcs11 module, see the PAM-PKCS11 User Manual, available at the http://www.opensc-project.org/ web site, under the PAM PKCS#11 link.

The following example illustrates how to configure the pam_pkcs11 module for a user whose certificate and private key are stored in the Solaris pkcs11_softtoken keystore. This example uses the default certificate verification policy.

- Set up the PKCS#11 module.

  On Solaris, the PKCS#11 module should be set to /usr/lib/libpkcs11.so.1, the PKCS#11 Cryptographic Framework library.

- Set up the slot_description entry.

  Specifies the slot to be used. For example, slot_description = "Sun Crypto Softtoken". The default value for this entry is none which means to use the first slot with an available token.

  An administrator can use the cryptoadm list -vcommand to find all the available slots and their slot descriptions. For more information, see libpkcs11(3LIB) and cryptoadm(1M).

- Install or create user certificates and its dedicated private keys in the specific PKCS#11 token.

- Set up the certificate verification policy (cert_policy). If needed, set up CA certificate and CRL files.

  The certificate verification policy includes:

  none — Perform no verification

  ca — Perform CA check

  signature — Perform a signature check to ensure that private and public key matches

  crl_*xxx* — Perform various certificate revocation checking

  As this example uses the default policy, cert_policy = ca,signature, an administer needs to set up the CA certificates.

  - Copy the CA certificate to the /etc/security/pam_pkcs11/cacerts directory.

    A certificate that is self-signed is its own CA certificate. Therefore, in this example, the certificate is placed both in the Softtoken keystore and in the CA certificate directory.

  - Make hash links for CA certificates

    ```
    $ /etc/security/pam_pkcs11/make_hash_link.sh \
          /etc/security/pam_pkcs11/cacerts
    ```

- Set up the mappers and mapfiles.

When a X509 certificate is provided, there are no direct ways to map a certificate to a login. The `pam_pkcs11` module provides a configurable way with mappers to specify `cert-to-user` mapping.

Many mappers are provided by the `pam_pkcs11` module, for example, the common name (CN) mapper, the digest mapper, the Email mapper, or the LDAP mapper.

A user can configure a mapper list in the `pam_pkcs11.conf` file. The mappers in the list are used sequentially until the certificate is successfully matched with the user.

The default mapper list is as follows:

```
use_mappers = digest, cn, pwent, uid, mail, subject, null;
```

Some mappers do not require the specification of a mapfile, for example, the common name mapper. Other mappers require mapfiles, for example, the digest mapper. Some sample mapping files can be found in the `/etc/security/pam_pkcs11` directory.

选项

The following options are supported:

`config_file=`*filename*
Specify the configuration file. The default value is `/etc/security/pam_pkcs11/pam_pkcs11.conf`.

`debug`
Enable debugging output.

文件

`/usr/lib/security/pam_pkcs11.so`
`pam_pkcs11` module

`/usr/lib/pam_pkcs11/ldap_mapper.so`
Mapper module.

`/usr/lib/pam_pkcs11/opensc_mapper.so`
Mapper module.

`/usr/lib/pam_pkcs11/openssh_mapper.so`
Mapper module.

`/etc/security/pam_pkcs11/pam_pkcs11.conf`
Configuration file.

`/etc/security/pam_pkcs11/cacerts`
Configuration directory. Stores the CA certificates.

`/etc/security/pam_pkcs11/crls`
Configuration directory. Stores the CRL files.

`/etc/security/pam_pkcs11/digest_mapping.example`
Sample mapfile.

`/etc/security/pam_pkcs11/subject_mapping.example`
Sample mapfile.

/etc/security/pam_pkcs11/mail_mapping.example
Sample mapfile.

/etc/security/pam_pkcs11/make_hash_link.sh
Sample script.

Authors

PAM-pkcs11 was originally written by MarioStrasser , mast@gmx.net.

Newer versions are from Juan Antonio Martinez, jonsito@teleline.es

属性

See attributes(5) for a description of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	library/security/pam/module/pam-pkcs11, SUNWpampkcs11r, SUNWpampkcs11-docs
Interface Stability	Uncommitted

另请参见

pkcs11_inspect(1), pklogin_finder(1), cryptoadm(1M), libpkcs11(3LIB)libpkcs11(3LIB)pam_sm_authenticate(3PAM), pam.conf(4), attributes(5), pkcs11_softtoken(5)

PAM-PKCS11 User Manual, available at the http://www.opensc-project.org/ web site, under the PAM PKCS#11 link.

引用名

pam_rhosts_auth – authentication management PAM module using ruserok()

用法概要

```
/usr/lib/security/pam_rhosts_auth.so.1
```

描述

The rhosts PAM module, `/usr/lib/security/pam_rhosts_auth.so.1`, authenticates a user via the `rlogin` authentication protocol. Only `pam_sm_authenticate()` is implemented within this module. `pam_sm_authenticate()` uses the `ruserok`(3SOCKET) library function to authenticate the `rlogin` or `rsh` user. `pam_sm_setcred()` is a null function.

`/usr/lib/security/pam_rhosts_auth.so.1` is designed to be stacked on top of the `/usr/lib/security/pam_unix.so.1` module for both the `rlogin` and `rsh` services. This module is normally configured as *sufficient* so that subsequent authentication is performed only on failure of `pam_sm_authenticate()`. The following option may be passed in to this service module:

`debug` `syslog`(3C) debugging information at LOG_DEBUG level.

属性

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT Level	MT-Safe with exceptions

另请参见

`pam`(3PAM), `pam_authenticate`(3PAM), `ruserok`(3SOCKET), `syslog`(3C), `libpam`(3LIB), `pam.conf`(4), `attributes`(5)

附注

The interfaces in `libpam()` are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

引用名

pam_roles – Solaris Roles account management module

用法概要

pam_roles.so.1

描述

The pam_roles module implements pam_sm_acct_mgmt(3PAM). It provides functionality to verify that a user is authorized to assume a role. It also prevents direct logins to a role. The user_attr(4) database is used to determine which users can assume which roles.

The PAM items PAM_USER and PAM_AUSER, and PAM_RHOST are used to determine the outcome of this module. PAM_USER represents the new identity being verified. PAM_AUSER, if set, represents the user asserting a new identity. If PAM_AUSER is not set, the real user ID of the calling service implies that the user is asserting a new identity. Notice that root can never have roles.

This module is generally stacked above the pam_unix_account(5) module.

The following options are interpreted:

allow_remote
Allows a remote service to specify the user to enter as a role.

debug
Provides syslog(3C) debugging information at the LOG_DEBUG level.

错误

The following values are returned:

PAM_IGNORE
If the type of the new user identity (PAM_USER) is "normal". Or, if the type of the new user identity is "role" and the user asserting the new identity (PAM_AUSER) has the new identity name in its list of roles.

PAM_USER_UNKNOWN
No account is present for user.

PAM_PERM_DENIED
If the type of the new user identity (PAM_USER) is "role" and the user asserting the new identity (PAM_AUSER) does not have the new identity name in its list of roles.

示例

示例 1 Using the pam_roles.so.1 Module

The following example is a pam.conf(4) fragment that demonstrates the use of the pam_roles.so.1 module:

```
cron account required pam_unix_account.so.1

other account requisite pam_roles.so.1
other account required pam_unix_account.so.1
```

The equivalent configuration in /etc/pam.d/ would be the following entry in /etc/pam.d/cron:

```
account required pam_unix_account.so.1
```

and the following entries in /etc/pam.d/other:

示例 1 Using the pam_roles.so.1 Module （续）

```
account requisite pam_roles.so.1
account required pam_unix_account.so.1
```

The cron service does not invoke pam_roles.so.1. Delayed jobs are independent of role assumption. All other services verify that roles cannot directly login. The "su" service (covered by the "other" service entry) verifies that if the new user is a role, the calling user is authorized for that role.

示例 2 Allowing Remote Roles

Remote roles should only be allowed from remote services that can be trusted to provide an accurate PAM_AUSER name. This trust is a function of the protocol (such as sshd-hostbased).

The following example is a pam.conf(4) fragment that demonstrates the use of pam_roles configuration for remote roles for the sshd-hostbased service.

```
sshd-hostbased account requisite pam_roles.so.1 allow_remote
sshd-hostbased account required pam_unix_account
```

The equivalent configuration in /etc/pam.d/ would be the following entries in /etc/pam.d/sshd-hostbased:

```
account requisite pam_roles.so.1 allow_remote
account required pam_unix_account
```

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

roles(1), sshd(1M), su(1M), libpam(3LIB), pam(3PAM), pam_acct_mgmt(3PAM), pam_setcred(3PAM), pam_set_item(3PAM), pam_sm_acct_mgmt(3PAM), syslog(3C), pam.conf(4), user_attr(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_auth(5), pam_unix_session(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

This module should never be stacked alone. It never returns PAM_SUCCESS, as it never makes a positive decision.

The allow_remote option should only be specified for services that are trusted to correctly identify the remote user (that is, sshd-hostbased).

PAM_AUSER has replaced PAM_RUSER whose definition is limited to the rlogin/rsh untrusted remote user name. See pam_set_item(3PAM).

引用名

pam_sample – a sample PAM module

用法概要

```
/usr/lib/security/pam_sample.so.1
```

描述

The SAMPLE service module for PAM is divided into four components: authentication, account management, password management, and session management. The sample module is a shared object that is dynamically loaded to provide the necessary functionality.

Sample Authentication Component

The SAMPLE authentication module provides functions to test the PAM framework functionality using the `pam_sm_authenticate(3PAM)` call. The SAMPLE module implementation of the `pam_sm_authenticate(3PAM)` function compares the user entered password with the password specified in the PAM configuration (see `pam.conf(4)`), or the string `test` if a default test password has not been set. The following options can be passed in to the SAMPLE Authentication module:

`debug`
Syslog debugging information at the `LOG_DEBUG` level.

`pass=newone`
Sets the password to be newone.

`first_pass_good`
The first password is always good when used with the use_first_pass or try_first_pass option.

`first_pass_bad`
The first password is always bad when used with the use_first_pass or try_first_pass option.

`always_fail`
Always returns `PAM_AUTH_ERR.`

`always_succeed`
Always returns `PAM_SUCCESS.`

`always_ignore`
Always returns `PAM_IGNORE.`

`use_first_pass`
Use the user's initial password (entered when the user is authenticated to the first authentication module in the stack) to authenticate with the SAMPLE module. If the passwords do not match, or if this is the first authentication module in the stack, quit and do not prompt the user for a password. It is recommended that this option only be used if the SAMPLE authentication module is designated as *optional* in the PAM configuration (see `pam.conf(4)`).

`try_first_pass`
Use the user's initial password (entered when the user is authenticated to the first authentication module in the stack) to authenticate with the SAMPLE module. If the passwords do not match, or if this is the first authentication module in the stack, prompt the user for a password.

The SAMPLE module `pam_sm_setcred(3PAM)` function always returns `PAM_SUCCESS.`

Sample Account Management Component

The SAMPLE Account Management Component implements a simple access control scheme that limits machine access to a list of authorized users. The list of authorized users is supplied as option arguments to the entry for the SAMPLE account management PAM module in the PAM configuration (see pam.conf(4)). Note that the module always permits access to the root super user.

The option field syntax to limit access is shown below: allow= *name[,name]* allow= *name [allow=name]*

The example pam.conf show below permits only larry to login directly. rlogin is allowed only for don and larry. Once a user is logged in, the user can use su if the user are sam or eric.

```
login     account     require     pam_sample.so.1  allow=larry
gdm       account     require     pam_sample.so.1  allow=larry
rlogin    account     require     pam_sample.so.1  allow=don allow=larry
su        account     require     pam_sample.so.1  allow=sam,eric
```

The debug and nowarn options are also supported.

Sample Password Management Component

The SAMPLE Password Management Component function (pam_sm_chauthtok(3PAM)), always returns PAM_SUCCESS.

Sample Session Management Component

The SAMPLE Session Management Component functions (pam_sm_open_session(3PAM), pam_sm_close_session(3PAM)) always return PAM_SUCCESS.

属性

See attributes(5) for description of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT Level	MT-Safe with exceptions

另请参见

pam(3PAM), pam_sm_authenticate(3PAM), pam_sm_chauthtok(3PAM), pam_sm_close_session(3PAM), pam_sm_open_session(3PAM), pam_sm_setcred(3PAM), libpam(3LIB), pam.conf(4), attributes(5)

警告

This module should never be used outside of a closed debug environment. The examples of the use_first_pass and try_first_pass options are obsolete for all other Solaris delivered PAM service modules

附注

The interfaces in libpam() are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

引用名

pam_smbfs_login – PAM user credential authentication module for SMB/CIFS client login

用法概要

```
pam_smb_cred.so.1
```

描述

The pam_smbfs_login module implements pam_sm_setcred(3PAM) to provide functions that act equivalently to the smbadm(1M) add-key command.

This optional functionality is meant to be used only in environments that do not run Active Directory or Kerberos, but which synchronize passwords between Solaris clients and their CIFS/SMB servers.

This module permits the login password to be stored as if the smbadm(1M) add-key command was used to store a password for PAM_USER in the user or system default domain.

To use this functionality, add the following line to the /etc/pam.d/login file:

```
auth optional    pam_smbfs_login.so.1
```

Authentication service modules must implement both pam_sm_authenticate(3PAM) and pam_sm_setcred(3PAM). In this module, pam_sm_authenticate(3PAM) always returns PAM_IGNORE.

The pam_sm_setcred(3PAM) function accepts the following flags:

PAM_REFRESH_CRED
Returns PAM_IGNORE.

PAM_SILENT
Suppresses messages.

PAM_ESTABLISH_CRED
PAM_REINITIALIZE_CRED
Stores the authentication token for PAM_USER in the same manner as the smbadm(1M) add-key command.

PAM_DELETE_CRED
Deletes the stored password for PAM_USER in the same manner as the smbadm(1M) remove-key command.

The following options can be passed to the pam_smbfs_login module:

debug
Produces syslog(3C) debugging information at the LOG_AUTH or LOG_DEBUG level.

nowarn
Suppresses warning messages.

错误

Upon successful completion of pam_sm_setcred(3PAM), PAM_SUCCESS is returned. The following error codes are returned upon error:

PAM_USER_UNKNOWN
User is unknown.

PAM_AUTHTOK_ERR
Password is bad.

PAM_AUTH_ERR
Domain is bad.

PAM_SYSTEM_ERR
System error.

属性

See attributes(5) for descriptions of the following attribute:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

smbadm(1M), syslog(3C), libpam(3LIB), pam(3PAM), pam_setcred(3PAM), pam_sm(3PAM), pam_sm_authenticate(3PAM), pam_sm_chauthtok(3PAM), pam_sm_setcred(3PAM), pam.conf(4), attributes(5), smbfs(7FS)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

引用名

pam_smb_passwd – SMB password management module

用法概要

`pam_smb_passwd.so.1`

描述

The pam_smb_passwd module enhances the PAM password management stack. This functionality supports the changing or adding of SMB passwords for local Solaris users. The Solaris SMB server uses SMB passwords to authenticate connected Solaris users. This module includes the pam_sm_chauthtok(3PAM) function.

The pam_sm_chauthtok() function accepts the following flags:

PAM_PRELIM_CHECK
Always returns PAM_IGNORE.

PAM_SILENT
Suppresses messages.

PAM_UPDATE_AUTHTOK
Updates or creates a new SMB local LM/NTLM hash for the user that is specified in PAM_USER by using the authentication information found in PAM_AUTHTOK. The LM hash is only created if the smbd/lmauth_level property value of the smb/server service is set to 3 or less. PAM_IGNORE is returned if the user is not in the local /etc/passwd repository.

The following options can be passed to the pam_smb_passwd module:

debug
Produces syslog(3C) debugging information at the LOG_AUTH or LOG_DEBUG level.

nowarn
Suppresses warning messages.

文件

/var/smb/smbpasswd
Stores SMB passwords for Solaris users.

错误

Upon successful completion of pam_sm_chauthtok(), PAM_SUCCESS is returned. The following error codes are returned upon error:

PAM_AUTHTOK_ERR
Authentication token manipulation error

PAM_AUTHTOK_LOCK_BUSY
SMB password file is locked

PAM_PERM_DENIED
Permissions are insufficient for accessing the SMB password file

PAM_SYSTEM_ERR
System error

PAM_USER_UNKNOWN
User is unknown

属性

See the attributes(5) man page for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

smbd(1M), syslog(3C), libpam(3LIB), pam(3PAM), pam_chauthtok(3PAM), pam_sm(3PAM), pam_sm_chauthtok(3PAM), pam.conf(4), attributes(5)

附注

The interfaces in libpam(3LIB) are MT-Safe *only* if each thread within the multi-threaded application uses its own PAM handle.

The pam_smb_passwd.so.1 module should be stacked following all password qualification modules in the PAM password stack.

引用名

pam_tsol_account – PAM account management module for Trusted Extensions

用法概要

```
/usr/lib/security/pam_tsol_account.so.1
```

描述

The Solaris Trusted Extensions service module for PAM, /usr/lib/security/pam_tsol_account.so.1, checks account limitations that are related to labels. The pam_tsol_account.so.1 module is a shared object that can be dynamically loaded to provide the necessary functionality upon demand. Its path is specified in the PAM configuration file.

pam_tsol_account.so.1 contains a function to perform account management, pam_sm_acct_mgmt(). The function checks for the allowed label range for the user. The allowable label range is set by the defaults in the label_encodings(4) file. These defaults can be overridden by entries in the user_attr(4) database.

By default, this module requires that remote hosts connecting to the global zone must have a CIPSO host type. To disable this policy, add the allow_unlabeled keyword as an option to the entry in pam.conf(4), as in:

```
other  account required    pam_tsol_account allow_unlabeled
```

The equivalent PAM configuration in /etc/pam.d/ would be the following entry in /etc/pam.d/other:

```
account required           pam_tsol_account allow_unlabeled
```

选项

The following options can be passed to the module:

allow_unlabeled
Allows remote connections from hosts with unlabeled template types.

debug
Provides debugging information at the LOG_DEBUG level. See syslog(3C).

返回值

The following values are returned:

PAM_SUCCESS
The account is valid for use at this time and label.

PAM_PERM_DENIED
The current process label is outside the user's label range, or the label information for the process is unavailable, or the remote host type is not valid.

Other values
Returns an error code that is consistent with typical PAM operations. For information on error-related return values, see the pam(3PAM) man page.

属性

See attributes(5) for description of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT Level	MT-Safe with exceptions

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

另请参见

keylogin(1), libpam(3LIB), pam(3PAM), pam_sm_acct_mgmt(3PAM), pam_start(3PAM), syslog(3C), label_encodings(4), pam.conf(4), user_attr(4), attributes(5)

《Oracle Solaris 11.1 管理：安全服务》中的第 14 章 “使用可插拔验证模块”

附注

The functionality described on this manual page is available only if the system is configured with Trusted Extensions.

引用名

pam_tty_tickets – PAM authentication module

用法概要

```
pam_tty_tickets.so.1 [timeout=minutes] [sudo-compat] [debug]
```

描述

The pam_tty_tickets module provides a mechanism for checking a ticket that was created by a prior successful authentication. Tickets by default validity of 5 minutes.

The default ticket location includes both the source (PAM_AUSER) and destination (PAM_USER) as well as the tty (PAM_TTY) for which it is valid.

The module can be configured using the sudo–compat option to store the tickets in the same location as sudo, though use of sudo is not required to use this feature.

The pam_sm_setcred() function creates a ticket for the user in the tickets directory.

The pam_sm_authenticate() function checks the timestamp on the ticket is no older than the timeout value, if is then it returns PAM_SUCCESS. If it is older then the ticket is removed and the module returns PAM_IGNORE.

This module is intended to be placed in the auth stack with the sufficient control flag.

No messages are produced by this module using the PAM conversation function. Some messages are sent to syslog for error conditions as as well as messages at LOG_INFO for ticket validity checking

The following options can be passed to the module:

debug
: Debugging information is sent to syslog LOG_AUTH|LOG_DEBUG.

sudo-compat
: Location of the per user (per tty) tickets, matches the sudo location. When this option is set PAM_USER must be root other wise the module returns PAM_IGNORE and tickets are not read or created.

timeout
: Validity time in minutes for a ticket. The default is 5 minutes.

示例

示例 1 Using the Default Settings

The following is an excerpt of a sample pam.conf configuration file that has per tty tickets with the default time out (5 minutes) for users authenticating with su(1M):

```
su auth required    pam_unix_cred.so.1
su auth sufficient  pam_tty_tickets.so.1
su auth requisite   pam_authtok_get.so.1
su auth required    pam_dhkeys.so.1
su auth required    pam_unix_auth.so.1
```

示例 2 Changing the Default Settings

The following example changes the defaults so that tickets are valid for 10 minutes and uses the sudo location:

示例 2 Changing the Default Settings （续）

```
su auth required    pam_unix_cred.so.1
su auth sufficient  pam_tty_tickets.so.1 sudo-compat timeout=10
su auth requisite   pam_authtok_get.so.1
su auth required    pam_dhkeys.so.1
su auth required    pam_unix_auth.so.1
```

错误

PAM_SUCCESS Ticket is valid

PAM_IGNORE All other cases

文件

/system/volatile/tty_tickets/<PAM_AUSER>/<PAM_USER>/<PAM_TTY>
Default ticket location.

/system/volatile/sudo/<PAM_AUSER>/<PAM_TTY>
When used sudo-compat is set this file has the same format as those created by sudo.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	See below.

The syslog messages are Volatile. The module name, module options, and ticket locations are Committed.

另请参见

su(1M), sudo(1M), pam(3PAM), pam_sm_authenticate(3PAM), pam_sm_setcred(3PAM), attributes(5)

引用名

pam_unix_account – PAM account management module for UNIX

用法概要

```
pam_unix_account.so.1
```

描述

pam_unix_account module implements pam_sm_acct_mgmt(), which provides functionality to the PAM account management stack. The module provides functions to validate that the user's account is not locked or expired and that the user's password does not need to be changed. The module retrieves account information from the configured databases in nsswitch.conf(4).

The following options can be passed to the module:

debug
: syslog(3C) debugging information at the LOG_DEBUG level

nowarn
: Turn off warning messages

server_policy
: If the account authority for the user, as specified by PAM_USER, is a server, do not apply the Unix policy from the passwd entry in the name service switch.

错误

The following values are returned:

PAM_UNIX_ACCOUNT	User account has expired
PAM_AUTHTOK_EXPIRED	Password expired and no longer usable
PAM_BUF_ERR	Memory buffer error
PAM_IGNORE	Ignore module, not participating in result
PAM_NEW_AUTHTOK_REQD	Obtain new authentication token from the user
PAM_PERM_DENIED	The account is locked or has been inactive for too long
PAM_SERVICE_ERR	Error in underlying service module
PAM_SUCCESS	The account is valid for use at this time
PAM_USER_UNKNOWN	No account is present for the user

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

pam(3PAM), pam_authenticate(3PAM), syslog(3C), libpam(3LIB), pam.conf(4), nsswitch.conf(4), attributes(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

Attempts to validate locked accounts are logged via syslog(3C) to the LOG_AUTH facility with a LOG_NOTICE severity.

引用名

pam_unix_auth – PAM authentication module for UNIX

用法概要

```
pam_unix_auth.so.1
```

描述

The pam_unix_auth module implements pam_sm_authenticate(), which provides functionality to the PAM authentication stack. It provides functions that use crypt(3C) to verify that the password contained in the PAM item PAM_AUTHTOK is the correct password for the user specified in the item PAM_USER.

If PAM_AUSER and PAM_USER are both specified and PAM_USER is a role, the user_attr(4) keyword roleauth is checked to determine if the password that is checked is for the role (PAM_USER) or the assuming user (PAM_AUSER). If PAM_REPOSITORY is specified, the user's pass word is fetched from that repository. Otherwise, the default nsswitch.conf(4) repository is searched for that user.

For accounts in the name services which support automatic account locking, the account can be configured to be automatically locked (see user_attr(4) and policy.conf(4)) after multiple failed login attempts. For accounts that are configured for automatic locking, if authentication failure is to be returned, the failed login counter is incremented upon each failure. If the number of successive failures equals or exceeds the configured value, the account is locked and PAM_MAXTRIES is returned. The files (see passwd(4) and shadow(4)) and ldap (when configured with enableShadowUpdate true, see ldapclient(1M)), repositories support automatic account locking. A successful authentication by this module clears the failed login counter and reports the number of failed attempts since the last successful authentication.

Authentication service modules must implement both pam_sm_authenticate() and pam_sm_setcred(). To allow the authentication portion of UNIX authentication to be replaced, pam_sm_setcred() in this module always returns PAM_IGNORE. This module should be stacked with pam_unix_cred(5) to ensure a successful return from pam_setcred(3PAM).

The following options can be passed to the module:

nowarn
: Turn off warning messages.

server_policy
: If the account authority for the user, as specified by PAM_USER, is a server, do not apply the UNIX policy from the passwd entry in the name service switch.

nolock
: Regardless of the automatic account locking setting for the account, do not lock the account, increment or clear the failed login count. The nolock option allows for exempting account locking on a per service basis.

错误

The following error codes are returned from `pam_sm_authenticate()`:

`PAM_AUTH_ERR`
Authentication failure.

`PAM_BUF_ERR`
Memory buffer error.

`PAM_IGNORE`
Ignores module, not participating in result.

`PAM_MAXTRIES`
Maximum number of retries exceeded.

`PAM_PERM_DENIED`
Permission denied.

`PAM_SUCCESS`
Successfully obtains authentication token.

`PAM_SYSTEM_ERR`
System error.

`PAM_USER_UNKNOWN`
No account present for user.

The following error codes are returned from `pam_sm_setcred()`:

`PAM_IGNORE`
Ignores this module regardless of the control flag.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

login(1), passwd(1), ldapclient(1M), useradd(1M), usermod(1M), roleadd(1M), rolemod(1M), crypt(3C), libpam(3LIB), pam(3PAM), pam_authenticate(3PAM), pam_setcred(3PAM), syslog(3C), pam.conf(4), passwd(4), policy.conf(4), nsswitch.conf(4), shadow(4), user_attr(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5), pam_unix_session(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

If the `PAM_REPOSITORY` *item_type* is set and a service module does not recognize the type, the service module does not process any information, and returns `PAM_IGNORE`. If the `PAM_REPOSITORY` *item_type* is not set, a service module performs its default action.

引用名

pam_unix_cred – PAM user credential authentication module for UNIX

用法概要

```
pam_unix_cred.so.1
```

描述

The pam_unix_cred module implements pam_sm_setcred(3PAM). It provides functions that establish user credential information. It is a module separate from the pam_unix_auth(5) module to allow replacement of the authentication functionality independently from the credential functionality.

The pam_unix_cred module must always be stacked along with whatever authentication module is used to ensure correct credential setting.

Authentication service modules must implement both pam_sm_authenticate() and pam_sm_setcred().

pam_sm_authenticate() in this module always returns PAM_IGNORE.

pam_sm_setcred() initializes the user's project, privilege sets and initializes or updates the user's audit context if it hasn't already been initialized. The following flags may be set in the flags field:

PAM_ESTABLISH_CRED
PAM_REFRESH_CRED
PAM_REINITIALIZE_CRED

Initializes the user's project to the project specified in PAM_RESOURCE, or if PAM_RESOURCE is not specified, to the user's default project. Establishes the user's privilege sets.

If the audit context is not already initialized and auditing is configured, these flags cause the context to be initialized to that of the user specified in PAM_AUSER (if any) merged with the user specified in PAM_USER and host specified in PAM_RHOST. If PAM_RHOST is not specified, PAM_TTY specifies the local terminal name. Attributing audit to PAM_AUSER and merging PAM_USER is required for correctly attributing auditing when the system entry is performed by another user that can be identified as trustworthy.

If the audit context is already initialized, the PAM_REINITIALIZE_CRED flag merges the current audit context with that of the user specified in PAM_USER. PAM_REINITIALIZE_CRED is useful when a user is assuming a new identity, as with su(1M).

PAM_DELETE_CRED

This flag has no effect and always returns PAM_SUCCESS.

The following options are interpreted:

debug	Provides syslog(3C) debugging information at the LOG_DEBUG level.
nowarn	Disables any warning messages.

错误

Upon successful completion of pam_sm_setcred(), PAM_SUCCESS is returned. The following error codes are returned upon error:

PAM_CRED_UNAVAIL	Underlying authentication service cannot retrieve user credentials
PAM_CRED_EXPIRED	User credentials have expired
PAM_USER_UNKNOWN	User is unknown to the authentication service
PAM_CRED_ERR	Failure in setting user credentials
PAM_BUF_ERR	Memory buffer error
PAM_SYSTEM_ERR	System error

The following values are returned from pam_sm_authenticate():

PAM_IGNORE	Ignores this module regardless of the control flag

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

ssh(1), su(1M), settaskid(2), libpam(3LIB), getprojent(3PROJECT), pam(3PAM), pam_set_item(3PAM), pam_sm_authenticate(3PAM), syslog(3C), setproject(3PROJECT),pam.conf(4), nsswitch.conf(4), project(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_auth(5), pam_unix_account(5), pam_unix_session(5), privileges(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

If this module is replaced, the audit context and credential may not be correctly configured.

引用名

pam_unix_session – session management PAM module for UNIX

用法概要

```
pam_unix_session.so.1
```

描述

The pam_unix_session module implements pam_sm_open_session(3PAM) and pam_sm_close_session(3PAM).

pam_sm_open_session() updates the /var/adm/lastlog file with the information contained in the PAM_USER, PAM_TTY, and PAM_RHOST items. pam_unix_account(5) uses this account to determine the previous time the user logged in.

pam_sm_close_session() is a null function.

The following options can be passed to the module:

debug syslog(3C) debugging information at the LOG_DEBUG level

错误

Upon successful completion, PAM_SUCCESS is returned. The following error codes are returned upon error:

PAM_SESSION_ERR Cannot make or remove the entry for the specified session (PAM_TTY is not present).

PAM_USER_UNKNOWN No account is present for *user*.

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT Level	MT-Safe with exceptions

另请参见

pam(3PAM), pam_authenticate(3PAM), syslog(3C), libpam(3LIB), pam.conf(4), nsswitch.conf(4), attributes(5), pam_authtok_check(5), pam_authtok_get(5), pam_authtok_store(5), pam_dhkeys(5), pam_passwd_auth(5), pam_unix_account(5),pam_unix_auth(5),

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

引用名

pam_user_policy – PAM user authentication policy module

用法概要

`pam_user_policy.so.1`

描述

The pam_user_policy module causes a user-specific PAM configuration to be evaluated and returns the result of evaluating such a configuration.

The pam_user_policy module implements all PAM service module functions.

The PAM configuration to evaluate is determined by looking for a pam_policy key in a user's attributes (see user_attr(4)) or profiles (see prof_attr(4)), or failing that, by looking at the default profiles granted in policy.conf(4). If no PAM configuration is found, the “unix” policy is used.

Failure to obtain a user name is considered an error (see below).

This module should generally be stacked as the first module, possibly as the only module, in a PAM service configuration using a *control_flag* value of 'sufficient' or 'binding', depending on the contents of the user-specific PAM configuration.

The pathname to the user-specific PAM configuration file passed to pam_eval(3PAM) must be absolute so pam_user_policy prepends “/etc/security/pam_policy” to any non-absolute PAM configuration pathnames.

The following option can be passed to the module:

debug syslog(3C) debugging information at the LOG_DEBUG level

Authentication Module

The pam_get_user(3PAM) function is used to retrieve the current user name and sets this to be the value of PAM_USER if PAM_USER was not already set. If no user name can be obtained, PAM_USER_UNKNOWN is returned.

The pam_user_policy authentication module then looks up the name of a PAM configuration file to use for that user as described above and evaluates the named configuration by calling pam_eval() with the same flags as were passed to the pam_user_policy authentication module.

The configuration file name found or the default “unix” is saved as module data (see pam_set_data(3PAM) for use by other pam_user_policy modules.

Other Modules

If the PAM_USER item is not set to a non-empty string then pam_user_policy returns PAM_USER_UNKNOWN immediately. If a PAM configuration file name was saved as module data by a previous call to a pam_user_policy module, then that configuration will be used; otherwise a PAM configuration will be looked up as described above. The service module then evaluates the named configuration by calling pam_eval() with the same flags as were passed to the service module.

The configuration file name found or the default “unix” is saved as module data (see pam_set_data(3PAM)) for use by other pam_user_policy modules.

返回值

If PAM_USER is not set or cannot be obtained, the module's service functions return PAM_USER_UNKNOWN. If module-specific data cannot be stored, PAM_SERVICE_ERR is returned. Failure to allocate resources causes the module to return PAM_BUF_ERR. Otherwise the value returned by pam_eval() is returned.

文件

A number of pam.conf files for inclusion by pam_user_policy can be found in /etc/security/pam_policy:

unix
: Use only Unix passwords for authentication, account management, and password management.

krb5_only
: Use Kerberos V5 only for authentication, account management, and password management.

krb5_first
: Use Kerberos V5 for authentication with fallback on Unix authentication, use Kerberos V5 for account management and password management for Kerberos users and Unix for account management and password management for Unix users.

krb5_optional
: Use Unix for authentication, account management, and password management and then optionally using Kerberos V5 for authentication, account management and password management for Kerberos users.

ldap
: Use pam_ldap(5) for authentication, account management, and password management for LDAP users and Unix for authentication, account management, and password management for Unix users.

any
: Try Kerberos V, LDAP and Unix, in that order, and as sufficient, for authentication, account management, and password management.

示例

示例 1 Authenticate a user with Kerberos V5 for all PAM services.

In the following example, user 'larry' should only be authenticated with Kerberos V5 for all PAM services.

```
$ usermod -K pam_policy=krb5_only larry
```

示例 2 Use the PAM configuration /etc/security/pam_policy/custom for a user.

In the following example, the PAM configuration /etc/security/pam_policy/custom should be used for user 'curly'. This custom PAM configuration might have different configurations for different PAM services, such as requiring Unix authentication for console logins but Kerberos V5 for all other PAM services.

```
$ usermod -K pam_policy=custom curly
```

示例 3 Create a new profile.

The following example creates a new profile named “PAM Per-User Policy of LDAP” and assign it to user 'moe' indicating that pam_ldap(5) should be used for all PAM services. Alternatively the profile could be assigned to all users by adding it to PROFS_GRANTED in policy.conf(4).

```
$ profiles -p "PAM Per-User Policy of LDAP" \
   'set desc="Profile which sets pam_policy=ldap";
   set pam_policy=ldap; exit;'
$ usermod -P "PAM Per-User Policy of LDAP" moe
```

示例 4 Add a new user.

The following example adds a new user named 'shemp' who uses the PAM configuration /usr/local/etc/pam.conf for all PAM services.

```
$ useradd -K pam_policy=/usr/local/etc/pam.conf shemp
```

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT-Level	MT-Safe with exceptions

另请参见

libpam(3LIB), pam(3PAM), pam_eval(3PAM), pam_get_user(3PAM), pam_set_data(3PAM), syslog(3C), pam.conf(4), policy.conf(4), prof_attr(4), user_attr(4), attributes(5), pam_ldap(5)

附注

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multithreaded application uses its own PAM handle.

引用名

pam_zfs_key – PAM user credential module for ZFS

用法概要

```
pam_zfs_key.so.1 [create] [homes=]
```

描述

The pam_zfs_key module implements pam_sm_setcred(3PAM) and pam_sm_chauthtok(3PAM).

The pam_zfs_key module provides functions that allow loading and changing of the ZFS encryption passphrase for encrypted file systems that are mounted at the users home directory location. Authentication service modules must implement both pam_sm_authenticate() and pam_sm_setcred().

pam_sm_authenticate() in this module always returns PAM_IGNORE. If they are located at a different location then the module option, homes= can be used to specify that. It is the ZFS dataset name, not the mountpoint, which would usually be /export/home/.

The last component of the ZFS dataset name must match the value of PAM_USER, that is, the users login name. If the users home directory is a local ZFS filesystem with encryption enabled and the ZFS keysource property is set to passphrase,prompt, on pam_sm_setcred() this module manages the keys as follows:

PAM_DELETE_CRED

Attempts to umount the filesystem and unload the key. This often fails because there are still processes with the user's home directory as the current working directory. A force module option is provided to attempt a forced unmount first.

PAM_ESTABLISH_CRED

Attempts to use the value of PAM_AUTHTOK to load the key for the ZFS dataset and mount it.

If PAM_AUTHTOK is not the correct passphrase, the user is prompted once for an alternate. This value is never stored in PAM_AUTHTOK, even if it is the correct passphrase for the ZFS dataset that is the users home directory. This is equivalent to:

```
zfs key -l rpool/export/home/$USER
```

If no ZFS file system exists for the user and the create module option is provided, a new one is created. The ZFS encryption property defaults to on in this case unless the encryption= property is set for the module to override it.

If the create module option is not provided and no ZFS file system exists for the user, the module returns PAM_IGNORE.

The newly created ZFS file system has the following ZFS delegations specified for the user for which it is created: key,keychange,mount. In these cases keysource is always set to passphrase,prompt.

When pam_sm_chauthtok(3PAM) is called, for example, on password change, this module attempts to change the passphrase for the ZFS dataset to match the value in PAM_AUTHTOK. This is equivalent to running:

```
zfs key -c rpool/export/home/$USER
```

This requires that the user have the keychange delegation, as password change usually runs as the user.

The following mount options are supported:

create Create new ZFS datasets

encryption Set the ZFS encryption property for create

force Attempt a umount2(2) with a MS_FORCE of the dataset when doing PAM_DELETE_CRED.

homes= Alternate location of ZFS datasets for user home directories. The default is rpool/export/home.

nowarn Do not provide any error messages or warnings.

示例

示例 1 Using pam_zfs_key in Default Mode

The following example uses pam_zfs_key in default mode.

```
gdm     auth requisite          pam_authtok_get.so.1
gdm     auth required           pam_dhkeys.so.1
gdm     auth required           pam_unix_cred.so.1
gdm     auth required           pam_unix_auth.so.1
gdm     auth optional           pam_zfs_key.so.1

other   password required       pam_dhkeys.so.1
other   password requisite      pam_authtok_get.so.1
other   password requisite      pam_authtok_check.so.1
other   password required       pam_authtok_store.so.1
other   password optional       pam_zfs_key.so.1
```

示例 2 Specifying an Alternate ZFS Dataset

The following example specifies an alternate ZFS dataset location for the home directory file systems. New entries should be created if they are not present using aes-256-gcm as the ZFS encryption property setting.

```
gdm auth requisite  pam_authtok_get.so.1
gdm auth required   pam_dhkeys.so.1
gdm auth required   pam_unix_cred.so.1
gdm auth required   pam_unix_auth.so.1
gdm auth optional   pam_zfs_key.so.1 homes=tank/users \
create encryption=aes-256-gcm
```

示例 3 Making it Mandatory for the ZFS Dataset to Mount

The following example makes it mandatory for the ZFS dataset to mount and ensures the passphrase always stays in sync with the login password.

```
gdm     auth requisite          pam_authtok_get.so.1
gdm     auth required           pam_dhkeys.so.1
gdm     auth required           pam_unix_cred.so.1
gdm     auth required           pam_unix_auth.so.1
gdm     auth required           pam_zfs_key.so.1

other   password required       pam_dhkeys.so.1
other   password requisite      pam_authtok_get.so.1
other   password requisite      pam_authtok_check.so.1
other   password requisite      pam_zfs_key.so.1
other   password required       pam_authtok_store.so.1
```

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
MT-Level	MT-Safe with exceptions. See below.

The interfaces in libpam(3LIB) are MT-Safe only if each thread within the multi-threaded application uses its own PAM handle.

另请参见

zfs(1M), umount2(2), pam.conf(4), libpam(3LIB), pam(3PAM), pam_sm_chauthtok(3PAM), pam_sm_setcred(3PAM), attributes(5)

引用名

pkcs11_kernel – 内核加密框架的 PKCS#11 接口

用法概要

```
/usr/lib/security/pkcs11_kernel.so
/usr/lib/security/64/pkcs11_kernel.so
```

描述

pkcs11_kernel.so 对象使用专用接口与内核加密框架进行通信，从而实现 RSA Security Inc. PKCS#11 加密令牌接口 (Cryptoki) v2.20 规范。

每个唯一的硬件提供者均由一个 PKCS#11 插槽表示。在没有硬件内核加密框架提供者的系统中，此 PKCS#11 库不提供插槽。

此库所提供的 PKCS#11 机制由可用硬件提供者确定。

应用程序开发者应链接到 libpkcs11.so，而不是直接链接到 pkcs11_kernel.so。请参见 libpkcs11(3LIB)。

除以下函数之外，实现了 libpkcs11(3LIB) 中列出的所有标准 PKCS#11 函数：

```
C_DecryptDigestUpdate
C_DecryptVerifyUpdate
C_DigestEncryptUpdate
C_GetOperationState
C_InitToken
C_InitPIN
C_SetOperationState
C_SignEncryptUpdate
C_WaitForSlotEvent
```

对这些函数的调用将返回 CKR_FUNCTION_NOT_SUPPORTED。

缓冲区不能大于 2 兆字节。例如，可以使用 2 兆字节的纯文本缓冲区和 2 兆字节的密文缓冲区调用 C_Encrypt()。

对 C_FindObjects() 的调用可返回的最大对象句柄数为 512。

可用于加密操作的最大内核内存量受 project.max-crypto-memory 资源控制限制。内核中对缓冲区的分配以及与会话相关的结构分配都按照此资源控制执行。

返回值

在 RSA PKCS#11 v2.20 规范中定义并列出了其中每个已实现函数的返回值。请参见 http://www.rsasecurity.com。

属性

有关以下属性的说明，请参见 attributes(5)：

属性类型	属性值
接口稳定性	Committed（已确定）
MT 级别	MT-Safe with exceptions（多线程安全，但存在异常）。请参见 RSA PKCS#11 v2.20 的第 6.6.2 节

属性类型	属性值
标准	PKCS#11 v2.20

另请参见

cryptoadm(1M)、rctladm(1M)、libpkcs11(3LIB)、attributes(5)、pkcs11_softtoken(5)

RSA PKCS#11 v2.20 http://www.rsasecurity.com

附注

如果应用程序具有 PKCS#11 插槽的打开会话，则会导致相应硬件提供者不能将驱动程序设为无法装入。管理员必须关闭具有打开 PKCS#11 会话的应用程序后，硬件提供者才能将驱动程序设为无法装入。

引用名

pkcs11_kms – Oracle Key Manager 的 RSA PKCS#11 提供器

用法概要

```
/usr/lib/security/pkcs11_kms.so
```

```
/usr/lib/security/64/pkcs11_kms.so
```

描述

`pkcs11_kms.so` 对象使用 Oracle Key Manager (OKM) KMS 代理协议来实现 RSA Security Inc. PKCS#11 Cryptographic Token Interface (Cryptoki), v2.20, 规范， 从而与 Oracle Key Manager 设备 (Key Manager appliance, KMA) 对话。此提供器实现 PKCS#11 规范并使用（专用）KMS 客户机协议与远程 OKM 进行通信。

此提供器支持以下 PKCS#11 机制：`CKM_AES_KEY_GEN`、`CKM_AES_CBC_PAD` 和 `CKM_AES_CBC`。

此提供器支持以下 PKCS#11 接口：

```
C_Initialize
C_Finalize
C_GetInfo
C_GetAttributeValue
C_SetAttributeValue
C_GetFunctionList
C_GetSlotList
C_GetSlotInfo
C_GetTokenInfo
C_GetMechanismList
C_GetMechanismInfo
C_InitToken
C_SetPIN
C_Login
C_Logout
C_FindObjectsInit/C_FindObjects/C_FindObjectsFinal
C_GenerateKey
C_EncryptInit/C_Encrypt/C_EncryptFinal
C_DecryptInit/C_Decrypt/C_DecryptFinal
C_DestroyObject
C_OpenSession
C_CloseSession
C_CloseAllSessions
C_GetSessionInfo
C_CreateObject
C_CopyObject
C_GetObjectSize
C_EncryptUpdate
C_DecryptUpdate
```

在调用时，所有其他函数均返回 `CKR_FUNCTION_NOT_SUPPORTED`。

先决条件

pkcs11_kms 提供器只能用于有权访问 OKM 的系统。OKM 管理员必须为每个访问 OKM 的用户（或应用程序）配置一个代理 ID。此操作通过 OKM 实用程序完成，这些实用程序属于 OKM 管理工具，并不是 Oracle Solaris 随附的。

在 OKM 管理员配置了要使用的 KMA 并将参数传递给了客户机（即 Oracle Solaris 用户或应用程序）后，即可初始化 Oracle Solaris PKCS#11 KMS 提供器以供使用。

KMS 提供器的初始化是通过 kmscfg(1M) 实用程序完成的。要初始化本地提供器配置文件以供进一步使用，kmscfg 要求用户至少输入配置文件的名称、OKM 代理 ID、用于保护配置文件的初始口令以及 KMA 的 IP 地址。有关详细信息，请参见 kmscfg(1M) 手册页。

运行完 kmscfg 并配置了本地令牌名称空间后，用户即可初始化令牌以供使用。令牌初始化是使用 pktool(1) 命令（如下所示）执行的：

```
$ pktool inittoken currlabel=KMS
```

用户必须提供缺省的 SO（security officer，安全官）PIN（Personal Identification Number，个人识别码）才能初始化 KMS 提供器以供使用。缺省 SO PIN 就是 OKM 管理员在最初设置 OKM 代理时所使用的 PIN。初始化令牌的用户必须知道此口令短语才能初始化提供器。

在初始化提供器后，可以更改用户 PIN 的缺省值。可再次使用 pktool(1) 更改该 PIN 值。

使用以下命令可更改本地 PIN：

```
$ pktool setpin token=KMS
```

为 pktool setpin 操作提供的 PIN 或通过调用 C_Login() 和 C_SetPIN() 函数提供的 PIN 可以是长度介于 1 到 256 之间且不带嵌入空字符的任何字符串。

访问令牌

在用户初始化令牌后，即可通过 pktool(1)、decrypt(1)、encrypt(1) 或通过编写 PKCS11 应用程序并指定 KMS 令牌来开始使用他们的令牌。

示例

示例 1 在 Oracle Key Manager 上创建密钥

以下命令在 Oracle Key Manager 上创建密钥：

```
$ pktool genkey token=KMS label=mykey1 keytype=aes keylen=256
```

示例 2 使用来自 Oracle Key Manager 的密钥对文件进行加密

以下命令使用来自 Oracle Key Manager 的密钥对文件进行加密：

```
$ encrypt -a aes -K mykey1 -T KMS -i input.txt -o output.enc
```

示例 3 使用来自 Oracle Key Manager 的密钥对文件进行解密

以下命令使用来自 Oracle Key Manager 的密钥对文件进行解密：

```
$ decrypt -a aes -K mykey1 -T KMS -i output.enc -o output.txt
```

属性

有关以下属性的说明，请参见 attributes(5)：

属性类型	属性值
可用性	/system/library/security/crypto/pkcs11_kms
接口稳定性	Committed（已确定）
MT 级别	MT-Safe with Exceptions（多线程安全，但存在异常）。请参见下文。
标准	PKCS#11 v2.20

多线程安全属性的异常记录在 RSA PKCS#11 v2.20 的第 6.6.2 节中。

另请参见

decrypt(1)、encrypt(1)、pktool(1)、cryptoadm(1M)、kmscfg(1M)、libpkcs11(3LIB)、attrib

《KMS 2.2：管理指南》

《Oracle Key Manager (OKM) Administration Guide》

附注

pkcs11_kms.so 使用一个专用目录存放初始化与 KMA 的连接所需的配置文件及其他数据。对于首次创建此专用目录的主机来说，此目录是本地目录。缺省情况下，KMS 令牌目录空间位于 /var/user/$USERNAME/kms 中。可以覆盖缺省 KMS 目录，方法是：先设置 KMSTOKEN_DIR 环境变量，然后再使用 kmscfg(1M)、decrypt(1)、encrypt(1) 和 pktool(1) 命令。

PKCS#11 客户机需要安装 Oracle Key Manager 软件版本 2.4。

如果 PKCS#11 客户机使用来自多个系统的相同代理 ID，则创建该代理时不应设置 One Time Passphrase 标志。在其中一些成员运行低于 OKM 2.4 的软件版本的 OKM 群集中，此选项不可用。有关创建代理的帮助信息，请参见《Oracle Key Manager (OKM) Administration Guide》。

在 KCS#11 客户机使用 OKM 代理创建密钥之前，OKM 代理必须分配 Default Key Group。如果未向代理分配 Default Key Group，操作将失败，并显示 CKR_PIN_INCORRECT 错误。有关向代理分配密钥组的帮助，请参见《Oracle Key Manager (OKM) Administration Guide》。

引用名

pkcs11_softtoken – 软件 RSA PKCS#11 的软令牌

用法概要

```
/usr/lib/security/pkcs11_softtoken.so
/usr/lib/security/64/pkcs11_softtoken.so
```

描述

pkcs11_softtoken.so 对象在软件中实现了 RSA Security Inc. PKCS#11 加密令牌接口 (Cryptographic Token Interface, Cryptoki) v2.20 规范。令牌对象的持久性存储由此 PKCS#11 实现提供。

应用程序开发者应链接到 libpkcs11.so 而不是直接链接到 pkcs11_softtoken.so。请参见 libpkcs11(3LIB)。

实现了以下加密算法：DES、3DES、AES、Blowfish、RC4、MD5、SHA1、SHA224、SHA256、SHA384、SHA512、RS
和 ECC。

除以下函数之外，实现了 libpkcs11(3LIB) 中列出的所有标准 PKCS#11 函数：

```
C_GetObjectSize
C_InitPIN
C_InitToken
C_WaitForSlotEvent
```

对这些函数的调用将返回 CKR_FUNCTION_NOT_SUPPORTED。

支持以下 RSA PKCS#11 v2.20 机制：

```
CKM_RSA_PKCS_KEY_PAIR_GEN
CKM_RSA_PKCS
CKM_RSA_X_509

CKM_DSA_KEY_PAIR_GEN
CKM_DSA
CKM_DSA_SHA1

CKM_DH_PKCS_KEY_PAIR_GEN
CKM_DH_PKCS_DERIVE

CKM_EC_KEY_PAIR_GEN
CKM_ECDSA
CKM_ECDSA_SHA1
CKM_ECDH1_DERIVE

CKM_DES_KEY_GEN
CKM_DES_ECB
CKM_DES_CBC
CKM_DES_CBC_PAD

CKM_DES3_KEY_GEN
CKM_DES3_ECB
```

```
CKM_DES3_CBC
CKM_DES3_CBC_PAD

CKM_AES_KEY_GEN
CKM_AES_ECB
CKM_AES_CBC
CKM_AES_CBC_PAD
CKM_AES_CTR

CKM_BLOWFISH_KEY_GEN
CKM_BLOWFISH_CBC

CKM_RC4_KEY_GEN
CKM_RC4

CKM_MD5_RSA_PKCS
CKM_SHA1_RSA_PKCS
CKM_SHA224_RSA_PKCS
CKM_SHA256_RSA_PKCS
CKM_SHA384_RSA_PKCS
CKM_SHA512_RSA_PKCS

CKM_MD5
CKM_SHA_1
CKM_SHA224
CKM_SHA256
CKM_SHA384
CKM_SHA512

CKM_MD5_HMAC
CKM_MD5_HMAC_GENERAL
CKM_SHA_1_HMAC
CKM_SHA_1_HMAC_GENERAL
CKM_SHA224_HMAC
CKM_SHA256_HMAC
CKM_SHA224_HMAC_GENERAL
CKM_SHA256_HMAC_GENERAL
CKM_SHA384_HMAC
CKM_SHA384_HMAC_GENERAL

CKM_MD5_KEY_DERIVATION
CKM_SHA1_KEY_DERIVATION
CKM_SHA224_KEY_DERIVATION
CKM_SHA256_KEY_DERIVATION
CKM_SHA384_KEY_DERIVATION
CKM_SHA512_KEY_DERIVATION

CKM_SSL3_PRE_MASTER_KEY_GEN
```

```
CKM_SSL3_MASTER_KEY_DERIVE
CKM_SSL3_KEY_AND_MAC_DERIVE
CKM_SSL3_MASTER_KEY_DERIVE_DH
CKM_TLS_PRE_MASTER_KEY_GEN
CKM_TLS_MASTER_KEY_DERIVE
CKM_TLS_KEY_AND_MAC_DERIVE
CKM_TLS_MASTER_KEY_DERIVE_DH
```

以下每种类型的密钥对象都具有令牌特定的某些属性，缺省情况下，这些属性将因创建对象、生成密钥/密钥对以及派生密钥而设置为 true。

公钥对象　CKA_ENCRYPT、CKA_VERIFY、CKA_VERIFY_RECOVER

私钥对象　CKA_DECRYPT、CKA_SIGN、CKA_SIGN_RECOVER、CKA_EXTRACTABLE

密钥对象　CKA_ENCRYPT、CKA_DECRYPT、CKA_SIGN、CKA_VERIFY、CKA_EXTRACTABLE

支持以下证书对象：

CKC_X_509　对于 CKC_X_509 证书对象，支持以下属性：CKA_SUBJECT、CKA_VALUE、CKA_LABEL、CKA_ID、CKA_ISSUER、CKA_SER
和 CKA_CERTIFICATE_TYPE。

CKC_X_509_ATTR_CERT　对于 CKC_X_509_ATTR_CERT 证书对象，支持以下属性：CKA_OWNER、CKA_VALUE、CKA_LABEL、CKA_SERIAL_NUMBER、CKA_AC_ISS
和 CKA_CERTIFICATE_TYPE。

在 C_FindObjectsInit 中执行匹配模板的对象搜索操作。匹配的对象将进行缓存以供后续 C_FindObjects 操作使用。

pkcs11_softtoken.so 对象提供了一个基于文件系统的持久性令牌对象存储，用来存储令牌对象。该令牌对象存储的缺省位置是 getpwuid_r() 所返回的用户起始目录。用户可以使用 ${SOFTTOKEN_DIR} 环境变量覆盖该缺省位置。

如果令牌对象存储从未经过初始化，C_Login() 函数可能会返回 CKR_OK，但用户无法创建、生成、派生或查找任何专用令牌对象并会收到 CKR_PIN_EXPIRED。

用户必须使用 pktool(1) setpin 命令（其缺省口令短语 "changeme" 是旧口令短语）更改对象存储的口令短语。在初始化新创建的令牌对象存储并为其设置口令短语时，需要执行此操作。

在使用 pktool setpin 命令所设置的新口令短语登录到对象存储后，用户即可在此新创建的对象存储中创建和存储专用令牌对象。在 setpin 初始化令牌对象存储之前，允许使用 C_Login() 函数，但用户创建、生成、派生或查找任何专用令牌对象的所有尝试都会失败并会显示 CKR_PIN_EXPIRED 错误。

为 C_Login() 和 C_SetPIN() 函数提供的 PIN 可以是长度介于 1 到 256 之间且不带嵌入空字符的任何字符串。

令牌对象存储的缺省位置是 `/var/user/$USERNAME/pkcs11_softtoken`。

用户可以使用 `${SOFTTOKEN_DIR}` 环境变量覆盖该缺省位置。备用令牌对象存储的位置是 `${SOFTTOKEN_DIR}/pkcs11_softtoken/`。

返回值

每个已实现函数的返回值都在 RSA PKCS#11 v2.20 规范中定义和列出。请参见 http://www.rsasecurity.com

文件

`/var/user/$USERNAME/pkcs11_softtoken`	用户的缺省令牌对象存储
`${SOFTTOKEN_DIR}/pkcs11_softtoken`	备用令牌对象存储

属性

有关以下属性的说明，请参见 `attributes(5)`：

属性类型	属性值
接口稳定性	Committed（已确定）
MT 级别	MT-Safe with exceptions（多线程安全，但存在异常）。请参见 RSA PKCS#11 v2.20 的第 6.6.2 节。
标准	PKCS#11 v2.20

另请参见

`pktool(1)`、`cryptoadm(1M)`、`libpkcs11(3LIB)`、`attributes(5)`、`pkcs11_kernel(5)`

RSA PKCS#11 v2.20 http://www.rsasecurity.com

引用名

pkcs11_tpm – 可信平台模块 (Trusted Platform Modules, TPM) 的 RSA PKCS#11 令牌

用法概要

```
/usr/lib/security/pkcs11_tpm.so
```

```
/usr/lib/security/64/pkcs11_tpm.so
```

描述

pkcs11_tpm.so 对象使用可信计算组协议与 TPM 安全设备对话，从而实现 RSA Security Inc. PKCS#11 加密令牌接口 (Cryptographic Token Interface, Cryptoki) v2.20 规范。此提供器实现 PKCS#11 规范并在 SUNWtss 软件包中使用 TCG 软件栈 (TCG Software Stack, TSS) API。

应用程序开发者应链接到 libpkcs11.so.1，而不是直接链接到 pkcs11_tpm.so。请参见 libpkcs11(3LIB)。

实现了以下加密算法：RSA、SHA1 和 MD5。

除以下函数之外，实现了 libpkcs11(3LIB) 中列出的所有标准 PKCS#11 函数：

```
C_EncryptUpdate
C_EncryptFinal
C_DecryptUpdate
C_DecryptFinal
C_DigestEncryptUpdate
C_DecryptDigestUpdate
C_SignEncryptUpdate
C_DecryptVerifyUpdate
C_GetFunctionStatus
C_CancelFunction
C_WaitForSlotEvent
C_GenerateKey
C_DeriveKey
```

支持以下 RSA PKCS#11 v2.20 机制：

```
CKM_RSA_PKCS_KEY_PAIR_GEN
CKM_RSA_PKCS
CKM_RSA_PKCS_OAEP
CKM_RSA_X_509
CKM_MD5_RSA_PKCS
CKM_SHA1_RSA_PKCS
CKM_SHA_1
CKM_SHA_1_HMAC
CKM_SHA_1_HMAC_GENERAL
CKM_MD5
CKM_MD5_HMAC
CKM_MD5_HMAC_GENERAL
```

基于每个用户的初始化

pkcs11_tpm 提供器只能用于安装有 TPM 设备和 SUNWtss 软件包的系统上。如果满足这些先决条件，用户可以使用pktool(1) 创建自己的专用令牌，这样他们能够使用 TPM 设备执行操作并通过 TPM 保护的密钥保护自己的专用数据。

要准备并初始化用户的 TPM 令牌，必须执行以下步骤：

1. 初始化令牌。
2. 设置 SO（security officer，安全官） PIN（Personal Identification Number，个人识别码）。
3. 设置用户的唯一 PIN。

令牌初始化是使用 pktool(1) 命令执行的，如下所示：

```
$ pktool inittoken currlabel=TPM newlabel=tpm/myname
```

- 缺省情况下，未初始化的 TPM 按名称 TPM 识别。当用户初始化自己的专用令牌时，可以将其重命名为某一其他名称（例如，tpm/joeuser）或保留为 TPM（在这种情况下，将省略 newlabel 参数）。
- 用户必须提供缺省的 SO PIN 才能初始化其令牌。缺省 SO PIN 为 87654321。在上述步骤 2 中更改该 PIN。

在初始化令牌之后，必须更改 SO 和用户 PIN 的缺省值。同样，可使用 pktool(1) 更改这些 PIN 值。

更改 SO PIN：

```
$ pktool setpin token=tpm/joeuser so
```

so 选项指示此 "setpin" 操作将更改 SO PIN，因此 SO PIN 必须存在。随后，用户必须输入缺省的 SO PIN (87654321) 并输入（和确认）新的 PIN。

在重置了 SO PIN 的缺省值后，还必须更改用户的唯一 PIN。

更改用户的 PIN：

```
$ pktool setpin token=tmp/joeuser
```

非 SO 用户的缺省 PIN 是 12345678。用户必须输入缺省 PIN 并输入（和确认）新的唯一 PIN。

为 pktool setpin 操作提供的 PIN 或通过调用 C_Login() 和 C_SetPIN() 函数提供的 PIN 可以是长度介于 1 到 256 之间且不带嵌入空字符的任何字符串。

访问令牌

在用户初始化令牌之后，即可通过以下方法来开始使用该令牌：使用 pktool(1)，或者通过编写 PKCS11 应用程序并使用前面创建的名称（在上例中为 tpm/joeuser）来定位该令牌。

示例：

```
$ pktool gencert token=tpm/joeuser -i
$ pktool list token=tpm/joeuser
```

附注

pkcs11_tpm.so 在文件系统特定的令牌对象存储区域提供对象存储。专用对象由使用专用密钥的加密进行保护，只能通过将令牌的专用密钥装入 TPM 并完全在 TPM 中执行解密的方式进行解密。用户的专用密钥由 TPM 在用户设置其个人 PIN（请参见上文）时生成。SO 和用户的密钥都存储在 TSS 持久性存储数据库中并由唯一的 UUID 值引用。所有用户令牌都有唯一的 SO 密钥和唯一的用户密钥，因此一个用户令牌的 PIN 无法解锁同一计算机上其他用户令牌的专用数据。

每个 TPM 都是唯一的，在一个 TPM 上创建的令牌密钥不能用于其他 TPM。pkcs11_tpm.so 令牌数据完全由 TPM 所在的系统进行管理并且不能移到其他系统上。如果 TPM 重置且 SRK（Storage Root Key，存储库根密钥）发生更改，则以前为该 TPM 生成的所有密钥都将不再有效。

pkcs11_tpm.so 将创建一个专用工作区来管理每个已创建令牌的管理文件。缺省情况下，此区域创建为 /var/user/$USERNAME/tpm/。不过，用户可以覆盖此区域，方法是在初始化或使用令牌之前设置 PKCS11_TPM_DIR 环境变量。

返回值

每个已实现函数的返回值都在 RSA PKCS#11 v2.20 规范中定义和列出。请参见 http://www.rsasecurity.com。

文件

/var/user/$USERNAME/tpm/
用户的缺省令牌对象存储。

${PKCS11_TPM_DIR}
备用令牌对象存储。

属性

有关下列属性的说明，请参见 attributes(5)：

属性类型	属性值
接口稳定性	Committed（已确定）
MT 级别	MT-Safe with Exceptions（多线程安全，但存在异常）（请参见下文）
标准	PKCS#11 v2.20

多线程安全属性的异常记录在 RSA PKCS#11 v2.20 的第 6.6.2 节中。

另请参见

pktool(1)、cryptoadm(1M)、libpkcs11(3LIB)、attributes(5)

TCG 软件栈 (TCG Software Stack, TSS) 规范 https://www.trustedcomputinggroup.org/specs/TSS（截至出版日期）

RSA PKCS#11 v2.20, http://www.rsasecurity.com

引用名

pkg – 映像包管理系统

描述

映像包管理系统 pkg(5) 是用于提供软件生命周期管理（安装、升级和删除）的框架。映像包管理以软件包为单位对软件进行管理，软件包是由一组关键字/值对以及可能的数据有效负荷定义的操作的集合。在许多情况下，操作是在文件系统中找到的文件，但它们也表示其他可安装对象，例如驱动程序、服务和用户。

软件包 Fmri 和版本

每个软件包均由带有机制 pkg: 的故障管理资源标识符 (fault management resource identifier, FMRI) 表示。软件包的完整 FMRI 由机制、发布者、软件包名称和以下格式的版本字符串组成：

```
pkg://solaris/system/library/c++-runtime@0.5.11,5.11-0.175.0.0.0.2.1:20120921T190358Z
```

solaris 是发布者。system/library/c++-runtime 是软件包名称。虽然名称空间是有层次的且可具有任意深度，但不存在强制包含；名称从本质上而言是任意的。发布者信息为可选项，但必须位于 pkg:// 之后（如果存在）。包含发布者的 FMRI 通常称为“全限定”FMRI。如果不存在发布者信息，则软件包名称通常应该位于 pkg:/ 之后。

如果 FMRI 不包含发布者信息，打包客户端通常允许省略 FMRI 的机制。例如，pkg:/system/library/c++-runtime 可以写为 system/library/c++-runtime。如果省略了机制，客户端也允许省略除软件包名称的最后一个组成部分（用于匹配目的）之外的所有其他内容。例如，system/library/c++-runtime 可以写为 library/c++-runtime 或 c++-runtime，这将与名为 c++-runtime 的软件包或以 /c++-runtime 结尾的软件包名称匹配。

发布者名称将个人、个人组或组织标识为一个或多个软件包的源。为避免发布者名称冲突并有帮于标识发布者，最佳做法是使用代表发布软件包实体的域名作为发布者名称。

版本跟在软件包名称后面，由 @ 符号分隔。版本包含四个数字序列，由标点符号分隔。前三个序列中的元素由圆点分隔，各序列可具有任意长度。不允许版本的组成部分以零开头（例如 01.1 或 1.01）。允许以零结尾（例如 1.10）。

版本的第一个部分是组件版本。对于紧密绑定到操作系统上的组件，此序列通常是操作系统中该版本的 uname -r 值。对于具有自己的开发生命周期的组件，此序列是一个由小圆点分隔的发行编号，例如 2.4.10。

版本的第二个部分（如果存在）必须跟在逗号 (,) 后面，是内部版本。内部版本指定构建软件包内容的操作系统版本，提供了希望软件包内容可成功运行的最低操作系统版本。

版本的第三个部分（如果存在）必须跟在连字符 (-) 后面，是分支版本。分支版本是用于提供特定于供应商的信息的版本控制组件。分支版本可在打包元数据发生更改时增大，独立于组件版本。分支版本可能包含内部版本号或其他信息。

版本的第四个部分（如果存在）必须跟在冒号 (:) 后面，是一个时间戳。时间戳表示软件包的发布时间。

在版本之间进行比较时，不考虑完整版本的任何组件，除非左侧的组件与其相同。因此 "4.3-1" 大于 "4.2-7"，因为 "4.3" 大于 "4.2"；"4.3-3" 大于 "4.3-1"，因为 "3" 大于 "1"。

系统的许多部分（如果适用）会在显示 FMRI 时缩短 FMRI 并接受简短格式的输入，以减少显示的或所需的信息量。通常，可以省略机制、发布者、内部版本和时间戳。有时可以省略所有版本控制信息。

操作

操作表示系统上的可安装对象。在软件包的清单中对操作进行了描述。每个操作主要由其名称和一个关键属性组成。这些属性共同引用一个唯一的对象，与该对象一样遵循版本历史记录。操作可以具有其他属性。某些属性由包管理系统直接解释。其他属性可能仅对系统管理员或最终用户有用。

操作具有一种简单文本表示法：

action_name attribute1=value1 attribute2=value2 ...

属性名称中不能包含空格、引号或等号 (=)。第一个等号后的所有字符都属于值。值中可以包含所有这些符号，虽然空格必须括在单引号或双引号中。单引号处于括在双引号中的字符串内时不需要进行转义，双引号处于括在单引号中的字符串内时也不需要进行转义。可使用反斜杠 (\) 字符作为引号的前缀来避免终止带引号的字符串。反斜杠可使用反斜杠进行转义。

可以使用多个值对属性进行多次命名。这些值被视为无序列表。

具有多个属性的操作可以在清单文件中创建长行。可以通过使用反斜杠终止每个不完整的行来对此类行进行折行。请注意，属性/值对之间必须具有此接续字符。属性、属性值及其组合均不可分离。

下面列出的属性不是全部属性。事实上，任意属性都可附加到操作中，标准属性组易于扩大以合并将来的开发。

某些操作属性会导致在打包上下文外部执行其他操作。这些属性记录在下面的“执行器”部分中。

文件操作

`file` 操作表示普通文件。`file` 操作引用有效负荷，具有四个标准属性：

`path` 安装文件的文件系统路径。此属性是 `file` 操作的关键属性。

`mode` 文件的访问权限（采用数字格式）。这些只是简单权限，并非 ACL。

`owner` 拥有文件的用户的名称。

`group` 拥有文件的组的名称。

有效负荷是一个位置属性，因为它未命名。它是操作名称后面的第一个词。在发布的清单中，它是文件内容的 `SHA-1` 散列。如果存在于尚待发布的清单中，则它表示可以找到有效负荷的路径。请参见 pkgsend(1)。如果值包括一个等号，则可以使用散列属性代替位置属性。这两种属性可用于同一操作中。但是，散列必须完全相同。

其他属性包括：

preserve

此属性指定在升级时不应覆盖文件的内容（如果确定自文件安装或上次升级后其内容已发生了更改）。在初始安装时，如果找到现有文件，则挽救该文件（存储在 /var/pkg/lost+found 中）。

如果 preserve 的值是 renameold，则使用扩展名 .old 重命名现有文件，并将新文件放入相应位置。

如果 preserve 的值是 renamenew，则现有文件保持不变，并使用扩展名 .new 安装新文件。

如果 preserve 的值是 legacy，则在初始软件包安装时不会安装此文件。在升级时，会使用扩展名 .legacy 重命名任何现有文件，并在随后将新文件放入相应位置。

如果 preserve 的值是 true（或是上面未列出的值，例如 strawberry），则现有文件保持不变，而且也不安装新文件。

overlay

此属性指定操作是允许其他软件包在同一位置提供文件，还是提供用于覆盖其他文件的文件。此功能设计用于不参与任何自组装（例如 /etc/motd）且可安全覆盖的配置文件。

如果未指定 overlay，多个软件包将无法向同一位置提供文件。

如果 overlay 的值是 allow，则允许另外一个软件包向同一位置提供文件。除非也设置了 preserve 属性，否则此值没有效果。

如果 overlay 的值是 true，操作提供的文件将覆盖已指定 allow 的任何其他操作。基于覆盖文件的 preserve 属性值保留对已安装文件进行的更改。在删除时，如果仍要安装将覆盖的操作，则将保留文件的内容，无论是否指定了 preserve 属性。只能一个操作覆盖另一个操作，且 mode、owner 和 group 属性必须匹配。

也可以“体验”文件，而且文件可根据自身情况具有其他属性。对于 ELF 文件，可识别下列属性：

elfarch

ELF 文件的体系结构。此属性是 uname -p 查询文件所基于的体系结构后的输出。

elfbits

此属性为 32 或 64。

elfhash

此属性是文件中“相关”ELF 部分的散列。这些部分已在装入二进制文件时映射到内存中。在确定两个二进制文件的可执行行为是否将不同时，仅需要考虑这些部分。

original_name
此属性用于处理可编辑文件在软件包之间或在位置之间（或在这两者之间）的移动操作。此属性采用的格式为源软件包的名称后跟一个冒号和文件的原始路径。所删除的任何文件将使用其软件包和路径或 original_name 属性的值（如果指定）进行记录。所安装的已设置 original_name 属性的任何可编辑文件将使用具有该名称的文件（如果它在同一打包操作中被删除）。

release-note
该属性用于指明此文件包含发行说明文本。该属性的值为软件包 FMRI。如果 FMRI 指定的软件包名称存在于原始映像中，指定的版本比原始映像中的软件包版本更高，则此文件将成为发行说明的一部分。特殊 FMRI feature/pkg/self 是指包含软件包。如果 feature/pkg/self 的版本为 0，则此文件仅在初次安装时是发行说明的一部分。

revert-tag
此属性用于标记应恢复为一个组的可编辑文件。可以指定多个 revert-tag 值。在使用所指定的任意标记调用 pkg revert 时，文件将恢复为其清单定义的状态。请参见 pkg(1)。

目录操作

dir 操作类似于 file 操作，因为它表示文件系统对象。但 dir 操作表示目录而不是普通文件。dir 操作具有与 file 操作相同的四个标准属性，path 是关键属性。

目录是 IPS 中包括的引用。当显式或隐式引用某目录的最新软件包不再引用该目录时，将删除该目录。如果该目录包含未打包的文件系统对象，则将这些项移动到 $IMAGE_META/lost+found 中。有关 $IMAGE_META 的更多信息，请参见“文件”部分。

要将未打包的内容移动到新的目录中，以下属性可能会有用：

salvage-from
此属性指定所挽救项的目录。具有此属性的目录在创建时可继承所挽救目录的内容（如果存在）。

链接操作

link 操作表示符号链接。link 操作具有以下标准属性：

path
安装符号链接的文件系统路径。此属性是 link 操作的关键属性。

target
符号链接的目标。链接将解析到的文件系统对象。

mediator
指定由给定中介组（例如 python）中涉及的所有路径名称共享的中介名称空间中的条目。可基于 mediator-version 和/或 mediator-implementation 执行链接中介。给定路径名称的所有中介链接必须指定同一中介。但是，并非所有中介版本和实现都需要在给定路径上提供链接。如果中介不提供链接，则会在选定该中介时删除链接。mediator 与特定版本和/或实现组合起来表示可选择供包管理系统使用的中介。

mediator-version
指定 mediator 属性描述的接口的版本（表示为非负整数的点分序列）。如果指定了 mediator 而未指定 mediator-implementation，则此属性是必需的。本地系统管理员可以显式设置要使用的版本。指定的值通常应与提供链接的软件包版本相匹配（例如，runtime/python-26 应使用 mediator-version=2.6），尽管这不是必需的。

mediator-implementation
指定除 mediator-version 之外还使用中介实现，或使用中介实现代替 mediator-version。不认为实现字符串应进行排序，如果系统管理员未显式指定，则由 pkg(5) 任意选择一个字符串。

该值可以是由字母数字字符和空格组成的任意长度的字符串。如果实现本身可被版本化或已被版本化，则应在字符串的结尾处在 @ 之后指定版本（表示为非负整数的点分序列）。如果存在实现的多个版本，则缺省行为是选择最高版本的实现。

如果系统上仅安装了特定路径的实现中介链接的一个实例，则会自动选择该实例。如果以后安装了该路径的其他链接，除非应用供应商、站点或本地覆盖或者如果某一链接进行了版本中介，否则不会切换链接。

mediator-priority
在解决中介链接中的冲突时，pkg(5) 通常会选择 mediator-version 值最大的链接，如果不可能，则会基于 mediator-implementation 进行选择。此属性用于为常规冲突解决方案过程指定覆盖。

如果未指定此属性，则会应用缺省中介选择逻辑。

如果值为 vendor，则与未指定 mediator-priority 的链接相比，会优先选择该链接。

如果值为 site，则与值为 vendor 或未指定 mediator-priority 的链接相比，会优先选择该链接。

本地系统管理员可以覆盖上面所述的选择逻辑。

硬链接操作

hardlink 操作表示硬链接。它具有与 link 操作相同的属性，path 也是其关键属性。

驱动程序操作

driver 操作表示设备驱动程序。driver 操作不引用有效负荷。驱动程序文件自身必须作为 file 操作进行安装。可识别下列属性（有关更多信息，请参见 add_drv(1M)）：

name
驱动程序的名称。这通常是（但并不总是）二进制驱动程序文件的文件名。此属性是 driver 操作的关键属性。

alias
此属性表示驱动程序的别名。给定的驱动程序可以具有多个 alias 属性。无需任何特殊的引号规则。

class
此属性表示驱动程序类。给定的驱动程序可以具有多个 class 属性。

`perms`
此属性表示驱动程序的设备节点的文件系统权限。

`clone_perms`
此属性表示此驱动程序的克隆驱动程序次要节点的文件系统权限。

`policy`
此属性指定设备的其他安全策略。给定的驱动程序可以具有多个 `policy` 属性，但次要设备规范不可以存在于多个属性中。

`privs`
此属性指定驱动程序所用的特权。给定的驱动程序可以具有多个 `privs` 属性。

`devlink`
此属性指定 `/etc/devlink.tab` 中的条目。该值是定义了进入文件的确切行，带有由 `\\t` 表示的制表符。有关更多信息，请参见 devlinks(1M)。给定的驱动程序可以具有多个 `devlink` 属性。

依赖操作

`depend` 操作表示软件包间的依赖性。一个软件包可以依赖于另一个软件包，因为第一个软件包需要第二个软件包中的功能才能运行自身包含的功能或者甚至进行安装。依赖性可以是可选的。如果安装时未满足某个依赖性，包管理系统会尝试安装或更新依赖软件包至足够新的版本（受其他约束限制）。

可以识别下列属性：

`fmri`
表示依赖软件包的 FMRI。此属性是 `dependency` 操作的关键属性。`fmri` 值不得包括发布者。假定软件包名称是完整的。`require-any` 类型的依赖性可具有多个 `fmri` 属性。`fmri` 值中的版本是可选项，虽然对于某些类型的依赖性来说，不带版本的 `fmri` 没有任何意义。

`type`
依赖性的类型。

`require`
该依赖性是必需的，必须具有等于或高于 `fmri` 属性中所指定版本的版本。如果未指定版本，则任何版本都满足依赖性。如果不能满足其任一必需依赖性，则无法安装软件包。

`optional`
该依赖性（如果存在）必须处于指定的版本级别或更高级别。

`exclude`
如果该依赖性存在于指定的版本级别或更高级别，则无法安装包含软件包。如果未指定版本，则依赖软件包无法与指定依赖性的软件包同时安装。

`incorporate`
该依赖性是可选项，但是依赖软件包的版本会受到约束。请参见下文的“约束和冻结”。

`require-any`
多个 `fmri` 属性指定的多个依赖软件包中的任一个都可满足依赖性（遵循与 `require` 依赖性类型相同的规则）。

`conditional`
该依赖性仅在系统上存在 `predicate` 属性定义的软件包时才是必需的。

`origin`
该依赖性（如果存在）必须在要在安装之前修改的映像上具有指定值或更优值。如果 `root-image` 属性的值为 `true`，则依赖性必须存在于根目录为 / 的映像上，才能安装此软件包。

`group`
除非软件包出现在映像避免列表上，否则该依赖性是必需的。请注意，过时软件包会无提示地满足组依赖性。请参见 `pkg(1)` 中的 `avoid` 子命令。

`parent`
如果映像不是子映像，会忽略该依赖性。如果映像是子映像，则依赖性必须存在于父映像中。符合 `parent` 依赖性的软件包版本与用于 `incorporate` 依赖性的软件包版本相同。

`predicate`
表示 `conditional` 依赖性的谓词的 FMRI。

`root-image`
仅对 `origin` 依赖性有影响，如上所述。

许可证操作

`license` 操作表示许可证或其他与软件包内容相关联的信息文件。软件包可以通过 `license` 操作将许可证、免责声明或其他指南提供给软件包安装程序。

`license` 操作的有效负荷将提供到与软件包相关的映像元数据目录中，且应仅包含用户可读的文本数据。不应包含 HTML 或任何其他形式的标记。通过各属性，`license` 操作可以向客户端指示必须显示相关的有效负荷并/或要求接受它。显示并/或接受的方法由客户端决定。

可以识别下列属性：

`license`
此属性是 `license` 操作的关键属性。此属性为许可证提供有意义的描述，以帮助用户在无需阅读许可证文本本身的情况下确定内容。其中一些示例值包括：

- ABC Co. Copyright Notice
- ABC Co. Custom License
- Common Development and Distribution License 1.0 (CDDL)
- GNU General Public License 2.0 (GPL)
- GNU General Public License 2.0 (GPL) Only
- MIT License
- Mozilla Public License 1.1 (MPL)
- Simplified BSD License

license 值在软件包内必须唯一。建议在说明中包括许可证的版本，如上面的几个示例所示。如果软件包有对应多种不同许可证的代码，请使用多个 license 操作。许可证属性值的长度不得超过 64 个字符。

must-accept
如果为 true，则用户必须先接受此许可证，才能安装或更新相关软件包。省略此属性等效于 false。接受的方法（例如，交互式或基于配置）由客户端决定。

must-display
如果为 true，则在执行打包操作期间客户端必须显示操作的有效负荷。省略此值等效于 false。此属性不应用于版权声明，仅用于实际许可证或执行操作期间必须显示的其他材料。显示的方法由客户端决定。

传统操作

legacy 操作表示由传统包管理系统使用的软件包数据。与此操作相关联的属性将添加到传统系统的数据库中，以便查询这些数据库的工具可以像实际安装了传统软件包一样工作。需特别指出的是，这应足以使传统系统确信系统上已安装了 pkg 属性指定的软件包，如此便可使用软件包来满足依赖性。

可识别根据 pkginfo(4) 上的参数指定的下列属性：

category
CATEGORY 参数的值。缺省值为 system。

desc
DESC 参数的值。

hotline
HOTLINE 参数的值。

name
NAME 参数的值。缺省值为 none provided。

pkg
要安装的软件包的缩写。缺省值为软件包的 FMRI 中的名称。此属性是 legacy 操作的关键属性。

vendor
VENDOR 参数的值。

version
VERSION 参数的值。缺省值为软件包的 FMRI 中的版本。

设置操作

set 操作表示软件包级别的属性或元数据，例如软件包描述。

可以识别下列属性：

name	属性的名称。
value	提供给属性的值。

set 操作可以提供软件包作者选择的任何元数据。但是，存在大量定义明确的对包管理系统具有特定意义的属性名称。

pkg.fmri
请参见“描述”部分中的“软件包 FMRI 和版本”。

info.classification
一个 pkg(5) 客户端可以使用一个或多个标记对软件包进行分类。该值应具有一个机制（例如 "org.opensolaris.category.2008" 或 "org.acm.class.1998"）和实际分类（例如 "Applications/Games"），由冒号 (:) 分隔。

pkg.description
软件包的内容和功能的详细描述，长度通常约为一个段落。

pkg.obsolete
如果为 true，则将软件包标记为过时。过时的软件包除了设置操作外不能具有任何其他操作，且不得标记为已重命名。

pkg.renamed
如果为 true，则软件包已被重命名。软件包中还必须存在一个或多个 depend 操作，且指向此软件包已重命名到的软件包版本。软件包不能同时标记为已重命名和过时，但在其他情况下可以具有任意多个设置操作。

pkg.summary
软件包的一行简短描述。

组操作

group 操作定义 UNIX 组，如 group(4) 中所定义。不存在对于组口令的支持。使用此操作定义的组最初不具有用户列表。可以使用 user 操作添加用户。可以识别下列属性：

groupname
组名的值。

gid
组的唯一数字 ID。缺省值为 100 之下的第一个自由组。

用户操作

user 操作定义 UNIX 用户，如 /etc/passwd、/etc/shadow、/etc/group 和 /etc/ftpd/ftpusers 文件中所定义。使用此属性定义的用户具有添加到相应文件中的条目。

可以识别下列属性：

username
用户的唯一名称

password
用户的加密口令。缺省值为 *LK*。请参见 shadow(4)。

uid
用户的唯一 UID。缺省值为 100 之下的第一个自由值。

`group`
用户的主组名称。必须可在 `/etc/group` 中找到。

`gcos-field`
`/etc/passwd` 中 `gcos` 字段的值。缺省值为 `username`。

`home-dir`
用户的起始目录。缺省值为 `/`。

`login-shell`
用户的缺省 shell。缺省值为空。

`group-list`
用户所属的辅助组。请参见 `group`(4)。

`ftpuser`
可设置为 `true` 或 `false`。缺省值 `true` 指示允许用户通过 FTP 登录。请参见 `ftpusers`(4)。

`lastchg`
1970 年 1 月 1 日至上次修改口令的日期之间的天数。缺省值为空。请参见 `shadow`(4)。

`min`
所需的相邻两次更改口令之间的最小天数。必须将此字段设置为 0 或更大值才能启用口令有效期。缺省值为空。请参见 `shadow`(4)。

`max`
口令的最大有效天数。缺省值为空。请参见 `shadow`(4)。

`warn`
用户在口令到期之前多少天收到警告。请参见 `shadow`(4)。

`inactive`
允许该用户不活动的天数。按每台计算机对此进行计数。可从计算机的 `lastlog` 文件获取有关上次登录的信息。请参见 `shadow`(4)。

`expire`
表示为自 UNIX 纪元（1970 年 1 月 1 日）后的天数的绝对日期。达到此数字时，将无法再进行登录。例如，到期值为 13514 指定登录将在 2007 年 1 月 1 日失效。请参见 `shadow`(4)。

`flag`
设置为空。请参见 `shadow`(4)。

执行器

在某些上下文中，附加操作可能适合在为特定操作做准备时执行或者在引入特定操作后执行。这些附加操作通常仅在实时系统映像上才需要，而且特定于操作系统。当软件包安装或删除过程中涉及的多个操作具有相同的执行器时，会为该安装或删除过程执行一次与执行器存在情况相对应的操作。

错误指定的执行器可能会导致软件包安装失败，如果该执行器无法确定进行安全安装的方法。

系统定义了以下执行器：

`reboot-needed`
可设置为 `true` 或 `false`。此执行器声明：如果软件包系统在实时映像上操作，则必须在新引导环境中更新或删除标记的操作。`be-policy` 映像属性控制新引导环境的创建。有关 `be-policy` 属性的更多信息，请参见 `pkg`(1) 手册页的“映像属性”部分。

`disable_fmri`、`refresh_fmri`、`restart_fmri`、`suspend_fmri`
其中每个执行器在软件包安装或删除过程中均使用服务实例的 FMRI 值进行操作。根据 `svcadm`(1M) 的 `disable` 子命令，`disable_fmri` 会导致给定的 FMRI 在删除操作之前被禁用。根据 `svcadm`(1M) 的各个子命令，`refresh_fmri` 和 `restart_fmri` 会导致给定的 FMRI 在安装、更新或删除操作后被刷新或重新启动。最后，`suspend_fmri` 会导致给定的 FMRI 在安装操作阶段之前被临时禁用，并在该阶段完成后被重新启用。

该值可以包含与多个服务实例匹配的模式。但是，它必须使用 `svcs`(1) 所接受的 glob 显式执行此操作，而不是通过不指示任何实例来隐式执行此操作。

中介

中介是代表一组相关符号链接或硬链接的名称。如果两个或更多个链接操作拥有相同的路径和中介名称，则用户或软件包系统会根据版本、实现或优先级选择链接目标。有关中介属性的信息，请参见“链接操作”。

以下示例显示名为 `java` 的中介的两个不同实例（其中，链接选择的区别为版本）。这两个 `link` 操作会显示在两个不同软件包中。

```
link mediator=java mediator-version=1.6 path=usr/java target=jdk/jdk1.6.0_31
link mediator=java mediator-version=1.7 path=usr/java target=jdk/jdk1.7.0_02
```

有关如何为此链接路径选择所需版本的信息，请参见 `pkg`(1) 手册页中的 `set-mediator` 子命令。要选择版本，必须安装两个软件包。

约束和冻结

在将软件包转换为新版本、添加到系统中或从系统中删除时，所选的版本或是否允许删除由对软件包施加的各种约束确定。这些约束可由其他软件包以依赖性的形式进行定义，或者由管理员以冻结的形式进行定义。

最常见的约束形式由 `require` 依赖性提供，如上面的“依赖操作”中所述。此类约束可防止软件包被降级或删除。

操作系统的大多数部分由名为 *incorporation* 的软件包进行封装。这些软件包主要提供由 `incorporate` 依赖性表示的约束。

如上所述，合并的软件包不需要存在于系统上，但如果存在，它将同时指定一个非独占最低版本和一个独占最高版本。例如，如果依赖性的 FMRI 具有版本 1.4.3，则低于 1.4.3 的版本不能满足依赖性，且任何高于或等于 1.4.4 的版本也不能满足依赖性。但是，可以允许仅仅扩展了点分序列的版本，如 1.4.3.7。

Incorporation 用于强制系统的各部分进行同步升级。对于某些组件（例如 C 库和内核），这是一项基本要求。对于其他组件（例如，不具有任何其他依赖性的简单用户级组件），同步升级仅仅用来提供一组经过测试的已知软件包版本，这些软件包版本可由 incorporation 的特定版本进行引用。

因为 incorporation 只是一个软件包，所以可将其删除，它提供的所有约束也将随之解除。但是，Oracle Solaris 提供的许多 incorporation 是其合并的软件包所必需的，因为解除其约束可能会不安全。

尝试将软件包升级到已安装的 incorporation 所不允许的版本，将不会尝试查找更高的 incorporation 版本来满足该请求，反而会失败。如果必须移动约束本身，而又无法删除指定它的 incorporation，则必须将 incorporation 升级到指定所需约束版本的版本。升级 incorporation 会导致不能满足新版本提供的约束的所有合并软件包也进行升级。

系统管理员可以使用 `pkg freeze` 命令约束软件包。在未提供版本的情况下，将指定软件包约束为系统上已安装的版本。如果提供了版本化软件包，则此管理约束或冻结会像已安装了合并依赖性（其中 `fmri` 属性具有所提供的软件包版本值）一样进行操作。

包管理系统永远不会自动解除冻结。要解除约束，请使用 `pkg unfreeze` 命令。

发布者和系统信息库

如上所述，发布者只是软件包客户端用来标识软件包提供者的一个名称。发布者可使用软件包系统信息库和/或软件包归档来发行其软件包。软件包系统当前支持以下两种类型的系统信息库：源系统信息库和镜像系统信息库。

源是包含一个或多个软件包的所有元数据（例如，目录、清单和搜索索引）和内容（文件）的软件包系统信息库。如果在一个映像中为给定发布者配置了多个源，则软件包客户端 API 会尝试选择从其检索软件包数据的最佳源。这是最常见的系统信息库类型，当每次在软件包系统信息库上使用 `pkgsend` 或 `pkgrecv` 时进行隐式创建。

mirror 是仅包含软件包内容（文件）的软件包系统信息库。如果在一个映像中为给定发布者配置了一个或多个镜像，则客户端 API 会优先使用镜像进行软件包内容检索并尝试选择从其检索软件包内容的最佳镜像。如果镜像不可访问、不具有所需内容或者运行缓慢，客户端 API 将从任何已配置的源系统信息库检索内容。镜像设计用于通过 `pkg.depotd`(1M) 的动态镜像功能在一组可信客户端之间共享内容。镜像还设计用于验证对软件包元数据的访问，但发行软件包内容不需要进行验证。例如，客户端可能配置有需要具有 SSL 密钥和证书对才能访问的 `https` 源，以及提供软件包内容的 `http` 镜像。这样，只有经过授权的客户端才可以安装或更新软件包，同时避免了验证软件包内容检索的开销。通过删除系统信息库中除名为 `file` 的子目录及其父目录之外的所有子目录可创建镜像。通过使用 `pkg.depotd`(1M) 的镜像模式还可将源系统信息库置备为镜像。

全局和非全局区域更新

`pkg` 系统会强制使非全局区域与全局区域保持同步。这表示某些软件包在全局区域和所有非全局区域中必须处于同一版本，以确保运行相同的内核。为此，`pkg` 会使用 `parent` 依赖性对非全局区域施加某些约束。有关 `parent` 依赖性的更多信息，请参见“依赖操作”。

由于全局区域对非全局区域施加的限制，非全局区域必须可以访问全局区域的软件包且必须具有相似的发布者配置。使用**系统信息库**可实现上述两个目标（请参见 pkg.sysrepo(1M) 手册页）。系统信息库可提供全局区域中配置的发布者的访问权，以及有关如何配置这些发布者的信息。要阻止非全局区域在安装或更新期间选择不同软件包，在发布者搜索顺序中系统发布者的排列顺序需要高于非全局区域中配置的发布者的顺序。有关发布者搜索顺序的信息，请参见 pkg(1) 手册页中的 pkg set-publisher 命令。

要更新系统上的所有非全局区域，请使用不带全局区域中参数的 pkg update 命令。此命令会在全局区域上执行并以递归方式在每个非全局区域上执行。将对非全局区域仅进行最少的必要更改，以使其与全局区域中所进行的更改同步。例如，假设全局区域和非全局区域中都安装了版本 1 的软件包 foo，并假设系统信息库中存在版本 2。如果 foo 具有父项依赖性，则 pkg update foo 会在全局区域和非全局区域中将 foo 更新至版本 2，这是因为 parent 依赖性会强制使软件包保持同步。如果 foo 不具有父项依赖性，则 foo 在全局区域中会更新至版本 2，但在非全局区域中仍为版本 1。

侧面和变量

软件可以具有可选组件和互斥组件。可选组件的示例包括语言环境和文档。互斥组件的示例包括 SPARC 或 x86 和调试或非调试二进制文件。

在 IPS 中，可选组件称为**侧面**，互斥组件称为**变量**。侧面和变量指定为软件包操作中的标记。每个侧面和变量标记都有一个名称和值。单个操作可以具有多个侧面和变量标记。具有多个侧面和变量标记的组件有多种，例如供开发者使用的特定于体系结构的头文件，或仅用于 SPARC 全局区域的组件。

以下是变量标记的一个示例 variant.arch=sparc。以下是侧面标记的一个示例 variant.arch=sparc。引用侧面和变量时，通常不带前导字符串 facet. 和 variant.。

侧面和变量是映像的特殊属性，无法在单个软件包上设置。要查看映像上设置的侧面和变量的当前值，请按 pkg(1) 手册页中所示，使用 pkg facet 和 pkg variant 命令。要修改映像上设置的侧面和变量的值，请使用 pkg change-facet 和 pkg change-variant 命令。

侧面为布尔型：只能设置为 true（启用）或 false（禁用）。缺省情况下，映像中的所有侧面都被视为 true。某一操作的侧面标记只能具有 true 值；其他值代表的行为不确定。映像上设置的侧面可以为完整侧面（如 doc.man）或模式（如 locale.*）。要禁用侧面名称空间的一部分，仅启用其中的几个侧面时，此方式很有用。例如，您可以禁用所有语言环境，然后仅启用一个或两个特定语言环境，如以下示例所示：

```
# pkg change-facet locale.*=false
[output about packages being updated]
# pkg change-facet locale.en_US=true
[output about packages being updated]
```

大多数变量可以具有任意数量的值。例如，arch 变量可以设置为 i386、sparc、ppc、arm 或分发支持的任何体系结构。（Oracle Solaris 中仅使用 i386

和 sparc。）但 debug 变量例外。debug 变量只能设置为 true 或 false；其他值的行为不确定。如果文件操作同时具有非调试和调试版本，则必须针对这两个版本明确设置适用的 debug 变量，如以下示例所示：

```
file group=sys mode=0644 overlay=allow owner=root \
  path=etc/motd pkg.csize=115 pkg.size=103 preserve=true \
  variant.debug.osnet=true

file group=sys mode=0644 overlay=allow owner=root \
  path=etc/motd pkg.csize=68 pkg.size=48 preserve=true \
  variant.debug.osnet=false
```

要安装使用变量的软件包，必须在映像上设置变量值。arch 和 zone 变量由创建映像和安装其初始内容的程序设置。缺省情况下，debug.* 变量在映像中设置为 false。

映像中设置的侧面和变量会影响是否安装特定操作。

- 始终会安装不带侧面或变量标记的操作。
- 仅当与侧面标记匹配的所有侧面或侧面模式在映像中都设置为 false 时，才不会安装带这些标记的操作。只要有任何侧面设置为 true 或未明确设置（true 为缺省值），就会安装该操作。
- 仅当所有变量标记的值与映像中设置的值相同时，才会安装带这些变量标记的操作。
- 如果侧面和变量都允许安装操作，则会安装带有这两种标记的操作。

您可以创建自己的侧面和变量标记。以下是 Oracle Solaris 中的常用标记。

变量名	可能值
variant.arch	sparc,i386
variant.opensolaris.zone	global,nonglobal
variant.debug.*	true、false

以下列表描述了 Oracle Solaris 中使用的一小部分侧面标记：

```
facet.devel            facet.doc
facet.doc.html         facet.doc.info
facet.doc.man          facet.doc.pdf
facet.locale.de        facet.locale.en_GB
facet.locale.en_US     facet.locale.fr
facet.locale.ja_JP     facet.locale.zh_CN
```

映像策略

映像策略由具有布尔值的映像属性所定义。有关 flush-content-cache-on-success 和 send-uuid 属性的说明以及如何查看和修改其值的信息，请参见 pkg(1) 手册页中的“映像属性”。

文件

因为 pkg(5) 映像可位于任意一个较大的文件系统内，需要使用标记 $IMAGE_ROOT 来区分相对路径。对于典型的系统安装，$IMAGE_ROOT 等效于 /。

$IMAGE_ROOT/var/pkg
　完整或部分映像的元数据目录。

$IMAGE_ROOT/.org.opensolaris,pkg
　用户映像的元数据目录。

在特定映像的元数据中，某些文件和目录可能包含修复和恢复期间有用的信息。标记 $IMAGE_META 用于指示包含元数据的顶层目录。 $IMAGE_META 通常是以上给出的两个路径之一。

$IMAGE_META/lost+found
　在软件包操作期间移动的有冲突目录和文件的位置。

$IMAGE_META/publisher
　为每个发布者包含一个目录。每个目录存储特定于发布者的元数据。

$IMAGE_META 目录分层结构中的其他路径是专用的，但可以进行更改。

属性

有关下列属性的说明，请参见 attributes(5)：

属性类型	属性值
可用性	package/pkg
接口稳定性	Uncommitted（未确定）

另请参见

pkg(1)、pkgsend(1)、pkg.depotd(1M)、pkg.sysrepo(1M)、svcs(1)、svcadm(1M)

《添加和更新 Oracle Solaris 11.1 软件包》

《复制和创建 Oracle Solaris 11.1 软件包系统信息库》

《在 Oracle Solaris 11.1 中使用映像包管理系统打包和交付软件》

http://hub.opensolaris.org/bin/view/Project+pkg/

引用名

privileges – 进程特权模型

描述

Oracle Solaris 软件实现了一组特权，用于提供对进程操作的精细控制。如果拥有某种特权，则进程可以执行一组特定的受限操作。

现在 Oracle Solaris 操作系统转变为一个主要基于特权的安全模型，这使开发人员可以限制进程只执行实际需要的那些特权操作，而不是授予所有特权（超级用户）或不授予任何特权（非零 UID）。此外，一组先前不受限制的操作现在需要特权；这类特权被称为“基本”特权，会缺省提供给所有进程。

除了“基本”特权以外，定义的所有特权构成了过去与 root 用户关联的那组特权。“基本”特权是过去无特权的进程拥有的“特权”。

定义的特权包括：

`PRIV_CONTRACT_EVENT`

允许进程请求将事件可靠地提交到事件端点。

允许进程将事件包括在模板的关键事件集术语中，该模板可以由用户批量生成。

`PRIV_CONTRACT_IDENTITY`

允许进程设置进程合同模板的服务 FMRI 值。

`PRIV_CONTRACT_OBSERVER`

允许进程观察由进程的有效用户 ID 以外的用户创建和拥有的合同生成的合同事件。

允许进程打开属于进程的有效用户 ID 以外的用户创建和拥有的合同的合同事件端点。

`PRIV_CPC_CPU`

允许进程访问每个 CPU 的硬件性能计数器。

`PRIV_DTRACE_KERNEL`

允许执行 DTrace 内核级跟踪。

`PRIV_DTRACE_PROC`

允许执行 DTrace 进程级跟踪。允许在用户具有权限的进程中放置和启用进程级跟踪探测。

`PRIV_DTRACE_USER`

允许执行 DTrace 用户级跟踪。允许使用 syscall 和配置文件 DTrace 提供者检查用户具有权限的进程。

`PRIV_FILE_CHOWN`

允许进程更改文件的所有者用户 ID。允许进程将文件的组 ID 更改为进程的有效组 ID 之外的某个 ID 或进程的某个补充组 ID。

`PRIV_FILE_CHOWN_SELF`

允许进程放弃其文件。具有此特权的进程运行时就像 {`_POSIX_CHOWN_RESTRICTED`} 未生效一样。

PRIV_FILE_DAC_EXECUTE
允许进程执行其权限位或 ACL 禁止进程执行权限的可执行文件。

PRIV_FILE_DAC_READ
允许进程读取其权限位或 ACL 禁止进程读取权限的文件或目录。

PRIV_FILE_DAC_SEARCH
允许进程搜索其权限位或 ACL 禁止进程搜索权限的目录。

PRIV_FILE_DAC_WRITE
允许进程写入其权限位或 ACL 禁止进程写入权限的文件或目录。缺少有效的 UID 0 时，写入 UID 0 拥有的文件需要所有特权。

PRIV_FILE_DOWNGRADE_SL
允许进程将文件或目录的敏感标签设置为不影响现有敏感标签的敏感标签。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_FILE_FLAG_SET
允许进程设置 immutable、nounlink 或 appendonly 文件属性。

PRIV_FILE_LINK_ANY
允许进程创建指向进程有效 UID 以外的 UID 拥有的文件的硬链接。

PRIV_FILE_OWNER
允许不是文件的所有者的进程修改该文件的访问权限和修改时间。允许不是目录的所有者的进程修改该目录的访问权限和修改时间。允许不是文件或目录的所有者的进程删除或重命名其父目录设置了“执行后保存文本图像”(sticky) 位的文件或目录。允许不是文件的所有者的进程将 namefs 挂载到该文件上。允许不是文件或目录的所有者的进程修改该文件或目录的权限位或 ACL。

PRIV_FILE_READ
允许进程读取其权限或 ACL 允许进程读取权限的文件或目录。

PRIV_FILE_SETID
允许进程更改文件的所有权，或写入未清除 set-user-ID 和 set-group-ID 位的文件。允许进程对所属组不是进程的有效组或进程的某个补充组的文件或目录设置 set-group-ID 位。允许进程在存在 PRIV_FILE_OWNER 时对具有不同所有权的文件设置 set-user-ID 位。当创建或修改 setuid 0 文件时，还适用其他限制。

PRIV_FILE_UPGRADE_SL
允许进程将文件或目录的敏感标签设置为影响现有敏感标签的敏感标签。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_FILE_WRITE
允许进程写入其权限或 ACL 允许进程写入权限的文件或目录。

PRIV_GRAPHICS_ACCESS
允许进程对图形设备创建带特权的 ioctl。通常仅 xserver 进程需要具有此特权。具有此特权的进程也允许执行带特权的图形设备映射。

PRIV_GRAPHICS_MAP
允许进程通过图形设备执行带特权的映射。

PRIV_IPC_DAC_READ
允许进程读取其权限位禁止进程读取权限的 System V IPC 消息队列、信号集或共享内存区段。

PRIV_IPC_DAC_WRITE
允许进程写入其权限位禁止进程写入权限的 System V IPC 消息队列、信号集或共享内存区段。

PRIV_IPC_OWNER
允许不是 System V IPC 消息队列、信号集或共享内存区段的所有者的进程删除、更改该消息队列、信号集或共享内存区段的所有权或权限位。

PRIV_NET_ACCESS
允许进程打开 TCP、UDP、SDP 或 SCTP 网络端点。

PRIV_NET_BINDMLP
允许进程绑定到配置为进程区域的多级端口 (multi-level port, MLP) 的端口。此特权适用于共享地址和区域特定的地址 MLP。有关配置 MLP 端口的信息，请参见 Trusted Extensions 手册页中的 tnzonecfg(4)。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_NET_ICMPACCESS
允许进程发送和接收 ICMP 包。

PRIV_NET_MAC_AWARE
允许进程使用 setpflags(2) 设置 NET_MAC_AWARE 进程标志。此特权还允许进程使用 setsockopt(3SOCKET) 设置 SO_MAC_EXEMPT 套接字选项。NET_MAC_AWARE 进程标志和 SO_MAC_EXEMPT 套接字选项都允许本地进程与无标签的对等进程通信（如果本地进程的标签影响对等进程的缺省标签，或本地进程在全局区域中运行）。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_NET_OBSERVABILITY
只允许进程打开设备接收网络通信，禁止发送通信。

PRIV_NET_PRIVADDR
允许进程绑定到特权端口号。特权端口号是 1-1023（传统 UNIX 特权端口）和标记为 “udp/tcp_extra_priv_ports” 的端口（保留供 NFS 和 SMB 使用的端口除外）。

PRIV_NET_RAWACCESS
允许进程直接访问网络层。

PRIV_PROC_AUDIT
允许进程生成审计记录。允许进程获取自身的审计预选信息。

PRIV_PROC_CHROOT
允许进程更改其根目录。

PRIV_PROC_CLOCK_HIGHRES
允许进程使用高分辨率计时器。

PRIV_PROC_EXEC
允许进程调用 exec(2)。

PRIV_PROC_FORK
允许进程调用 fork(2)、fork1(2) 或 vfork(2)。

PRIV_PROC_INFO
允许进程检查它可以向其发送信号的进程以外的进程的状态。不能被检查的进程在 /proc 中不可见，并且显示为不存在。

PRIV_PROC_LOCK_MEMORY
允许进程锁定物理内存中的页。

PRIV_PROC_OWNER
允许进程向其他进程发送信号，以及检查和修改其他进程中的进程状态，而不管所有权如何。修改其他进程时，还适用其他限制：访问进程的有效特权集必须是目标进程的有效特权集、允许特权集和可继承特权集的超集；其限制特权集必须是目标的限制特权集的超集；如果目标进程的任何 UID 设置为 0，则必须声明所有特权，除非有效 UID 为 0。允许进程将任意进程绑定到 CPU。

PRIV_PROC_PRIOCNTL
允许进程将其优先级提升到其当前级别之上。允许进程将其调度类更改为任何调度类，包括 RT 类。

PRIV_PROC_SESSION
允许进程向其会话之外的进程发送信号或对其跟踪。

PRIV_PROC_SETID
允许进程随意设置其 UID，设置为 UID 0 要求声明所有特权。

PRIV_PROC_TASKID
允许进程将一个新任务 ID 分配给其调用进程。

PRIV_PROC_ZONE
允许进程跟踪其他区域中的进程或向其发送信号。请参见 zones(5)。

PRIV_SYS_ACCT
允许进程通过 acct(2) 启用、禁用和管理记帐。

PRIV_SYS_ADMIN
允许进程执行系统管理任务，例如设置节点和域名称、指定 coreadm(1M) 和 nscd(1M) 设置等

PRIV_SYS_AUDIT
允许进程启动（内核）审计守护进程。允许进程查看和设置审计状态（审计用户 ID、审计终端 ID、审计会话 ID、审计预选掩码）。允许进程禁用和启用审计。允许进程配置审计参数（高速缓存和队列大小、事件到类的映射以及策略选项）。

PRIV_SYS_CONFIG
允许进程执行各种系统配置任务。允许执行文件系统特定的管理过程，例如文件系统配置 ioctl、配额调用、创建和删除快照以及处理 PCFS 引导扇区。

PRIV_SYS_DEVICES
允许进程创建设备特殊文件。允许进程成功调用某个调用内核 drv_priv(9F) 函数的内核模块以检查允许的访问权限。允许进程直接打开真实控制台设备。允许进程打开已以独占方式打开的设备。

PRIV_SYS_DL_CONFIG
允许进程配置系统的数据链路接口。

PRIV_SYS_IP_CONFIG
允许进程配置系统的 IP 接口和路由。允许进程配置 TCP/IP 参数。允许进程弹出具有匹配 zoneid 的固定 STREAM 模块。

PRIV_SYS_IPC_CONFIG
允许进程增加 System V IPC 消息队列缓冲区的大小。

PRIV_SYS_LINKDIR
允许进程链接目录以及解除该链接。

PRIV_SYS_MOUNT
允许进程挂载和卸载本来操作受限的文件系统（即，除 namefs 之外的大多数文件系统）。允许进程添加和删除交换设备。

PRIV_SYS_NET_CONFIG
允许进程执行 PRIV_SYS_IP_CONFIG、PRIV_SYS_DL_CONFIG 和 PRIV_SYS_PPP_CONFIG 允许的所有操作以及以下操作：使用 rpcmod STREAMS 模块和插入/删除模块栈顶部之外的位置上的 STREAMS 模块。

PRIV_SYS_NFS
允许进程提供 NFS 服务：启动 NFS 内核线程、执行 NFS 锁定操作、绑定到 NFS 保留端口：端口 2049 (nfs) 和端口 4045 (lockd)。

PRIV_SYS_PPP_CONFIG
允许进程使用 pppd(1M) pppd(1M) 创建、配置和销毁 PPP 实例，以及使用 sppptun(1M)sppptun(1M) 控制 PPPoE 激活。缺省情况下此特权会授予独占 IP 栈实例区域。

PRIV_SYS_RES_BIND
允许进程将进程绑定到处理器集。

PRIV_SYS_RES_CONFIG
允许进程将进程绑定到处理器集（如同 PRIV_SYS_RES_BIND 特权），还允许执行此段落中所述的下列操作。允许进程创建和删除处理器集，将 CPU 分配给处理器集，以及覆盖 PSET_NOESCAPE 属性。允许进程使用 p_online(2) 更改系统中 CPU 的运行状态。允许进程配置文件系统配额。允许进程配置资源池和将进程绑定到池。

PRIV_SYS_RESOURCE
允许进程超出通过 setrlimit(2) 和 setrctl(2) 对其施加的资源限制。

`PRIV_SYS_SHARE`
允许进程共享文件系统和取消共享。

`PRIV_SYS_SMB`
允许进程提供 NetBIOS 或 SMB 服务：启动 SMB 内核线程或绑定到 NetBIOS 或 SMB 保留端口：端口 137、138、139 (NetBIOS) 和 445 (SMB)。

`PRIV_SYS_SUSER_COMPAT`
允许进程成功调用某个调用内核 `suser()` 函数的第三方可装入模块以检查允许的访问权限。仅当兼容第三方可装入模块时才存在此特权，Oracle Solaris 属性不使用该特权。

`PRIV_SYS_TIME`
允许进程使用以下任何合适的系统调用管理系统时间：`stime(2)`、`adjtime(2)` 和 `ntp_adjtime(2)`。

`PRIV_SYS_TRANS_LABEL`
允许进程将不受进程的敏感标签影响的标签转换为外部字符串格式或反之。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

`PRIV_VIRT_MANAGE`
允许进程管理虚拟化环境。

`PRIV_WIN_COLORMAP`
允许进程覆盖颜色映射限制。

允许进程安装或删除颜色映射。

允许进程检索其他进程分配的颜色映射单元条目。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

`PRIV_WIN_CONFIG`
允许进程配置或销毁 X 服务器永久保留的资源

允许进程使用 SetScreenSaver 设置屏幕保护程序超时值

允许进程使用 ChangeHosts 修改显示访问控制列表。

允许进程使用 GrabServer。

允许进程使用 SetCloseDownMode 请求保留窗口、像素映射、颜色映射、属性、光标、字体或图形文本资源。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

`PRIV_WIN_DAC_READ`
允许进程从不归其所有的窗口资源中读取（具有不同的用户 ID）。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_WIN_DAC_WRITE

允许进程写入或创建不归其所有的窗口资源（具有不同的用户 ID）。新创建的窗口属性使用窗口的用户 ID 创建。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_WIN_DEVICES

允许进程对窗口输入设备执行各种操作。

允许进程获取和设置键盘和指针控制。

允许进程修改指针按钮和按键映射。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_WIN_DGA

允许进程使用直接图形访问 (DGA) X 协议扩展。仍需要对帧缓存器有直接进程访问权限。因此，该进程必须具有可访问帧缓存器的 MAC 和 DAC 特权，或者必须为该进程分配了帧缓存器。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_WIN_DOWNGRADE_SL

允许进程将窗口资源的敏感标签设置为不影响现有敏感标签的敏感标签。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_WIN_FONTPATH

允许进程设置字体路径。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_WIN_MAC_READ

允许进程从其敏感标签不等于进程敏感标签的窗口资源中读取。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_WIN_MAC_WRITE

允许进程创建其敏感标签不等于进程敏感标签的窗口资源。新创建的窗口属性使用窗口的敏感标签创建。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_WIN_SELECTION

允许进程在没有选择确认器干预的情况下请求在窗口间移动数据。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

PRIV_WIN_UPGRADE_SL

允许进程将窗口资源的敏感标签设置为影响现有敏感标签的敏感标签。

仅当系统配置有 Trusted Extensions 时，才会解释该特权。

在上面列出的特权中，特权 PRIV_FILE_LINK_ANY、PRIV_FILE_READ、PRIV_FILE_WRITE、PRIV_PROC_INFO、PRIV_PROC_SESSI 和 PRIV_PROC_EXEC 视为“基本”特权。过去，没有特权的进程始终具有这些特权。缺省情况下，进程仍具有基本特权。

进程的限制特权集（请参见下文）中必须存在特权 PRIV_PROC_SETID、PRIV_PROC_AUDIT 和 PRIV_SYS_RESOURCE，setuid root exec 才能成功，即获取值为 0 的有效 UID 和其他特权。

Oracle Solaris 中的特权实现使用四个特权集扩展进程凭证：

I，可继承特权集　exec 中继承的特权。

P，允许特权集　进程的最大特权集。

E，有效特权集　当前有效的特权。

L，限制特权集　进程及其子级可以获取的特权的上限。对 L 的更改将在下一个 exec 中生效。

对于没有特权的进程，I、P 和 E 集通常与基本特权集相同。限制特权集通常是整个特权集。

每个进程都有权限识别状态 (Privilege Awareness State, PAS)，该状态可以接受的值为 PA（privilege-aware，可识别特权）和 NPA（非 PA）。PAS 是一种过渡机制，用于允许用户在与旧超级用户模型完全兼容的方式和完全忽略有效 UID 的方式之间进行选择。

为了方便讨论，下面引入了“观察到的有效特权集”(oE) 和“观察到的允许特权集”(oP) 概念以及实现特权集 iE 和 iP。

通过 setppriv(2) 或者使用 setpflags(2) 处理有效特权集、允许特权集或限制特权集，进程将变为可识别特权的进程。在所有情况下，在变为可识别特权的过程中 oE 和 oP 都是不变的。在变为可识别特权的过程中，将进行以下赋值：

```
iE = oE
iP = oP
```

如果进程可识别特权，oE 和 oP 在更改 UID 时保持不变。如果进程不识别特权，观察到的 oE 和 oP 如下：

```
oE = euid == 0 ? L : iE
oP = (euid == 0 || ruid == 0 || suid == 0) ? L : iP
```

如果不识别特权的进程具有值为 0 的有效 UID，它可以行使其限制特权集中包含的特权（特权的上限）。如果不识别特权的进程具有任何值为 0 的 UID，则它表现为有能力行使 L 中的所有特权。

使用 setpflags() 可以使进程恢复到不识别特权的状态。内核在执行 exec(2) 时始终会尝试此操作。仅当满足以下条件时，才允许执行此操作。

- 如果有任何 UID 等于 0，则 P 必须等于 L。
- 如果有效 UID 等于 0，则 E 必须等于 L。

当进程放弃特权识别时，将进行以下赋值：

```
if (euid == 0) iE = L & I
if (any uid == 0) iP = L & I
```

在 UID 不为 0 时获取的特权是受限制特权集限制的进程可继承特权集。

只有进程的（观察到的）有效特权集中的特权才允许进程执行受限的操作。进程可以使用任何特权处理函数来在特权集中添加或删除特权。特权始终是可删除的。只能将允许特权集中的特权添加到有效特权集和可继承特权集中。限制特权集无法扩展。可继承特权集可以大于允许特权集。

当进程执行 exec(2) 时，内核将首先尝试取消特权识别，然后进行以下特权集修改：

```
E’ = P’ = I’ = L & I
L is unchanged
```

如果进程未处理其特权，则特权集实际上保持不变，因为 E、P 和 I 已相同。

在执行 exec 时，将强制使用限制特权集。

要以向后兼容的方式运行不识别特权的应用程序，可识别特权的应用程序应以 I=basic 启动不识别特权的应用程序。

对于大多数特权，缺少特权仅会导致失败。在某些情况下，缺少特权会导致系统调用的行为方式发生变化。在其他情况下，删除特权会导致 setuid root 应用程序出现严重故障。此类型的特权被视为“不安全的”。当进程的限制特权集中缺少任何不安全的特权时，系统将不会接受 setuid root 应用程序的 setuid 位。setuid root 应用程序可继续执行，不会更改有效用户 ID 或增加特权。已经确定以下特权是不安全的：PRIV_PROC_SETID、PRIV_SYS_RESOURCE 和 PRIV_PROC_AUDIT。

特权升级

在某些情况下，某一个特权会导致进程获取一个或多个未显式授予给该进程的其他特权。为了阻止此类特权升级，安全策略应要求显式授予这些其他特权。

升级通常源自那些允许通过“原始”接口修改系统资源的机制；例如，通过 /dev/kmem 更改内核数据结构或者通过 /dev/dsk/* 更改文件。当受控制的进程比控制进程具有更多特权时，也会发生升级。上述这种特殊情况包括：处理或创建 UID 0 所有的对象或尝试使用 setuid(2) 获取 UID 0。由于 UID 0 拥有所有系统配置文件，并且普通文件保护机制允许 UID 为 0 的进程修改系统配置，因此需要对 UID 0 进行特殊处理。通过进行相应的文件修改，以值为 0 的有效 UID 运行的给定进程可以获取所有特权。

在进程可能获取 UID 0 的情况下，安全策略应要求具有其他特权，最高为整个特权集。在有其他机制保护系统文件的情况下，可以放松或删除此类限制。当前的 Oracle Solaris 发行版中没有此类机制。

应尽可能限制使用 UID 0 进程。这些进程应替换为以其他 UID 运行的程序，并且该 UID 具有恰好需要的特权。

不再需要执行 (exec) 子进程的守护进程应从其允许特权集和限制特权集中删除 PRIV_PROC_EXEC 特权。

分配的特权和保护

向用户分配特权时，系统管理员可能会为该用户提供比所需更多的权限。管理员应考虑是否需要添加保护。例如，如果为用户指定了 PRIV_PROC_LOCK_MEMORY 特权，则管理员还应考虑设置 project.max-locked-memory 资源控制，以阻止该用户锁定所有内存。

扩展策略

在配置文件或命令行中列出特权时，有时可以使用扩展策略语法。在扩展策略中，特权集括在花括号中，后跟一个分号和一个对象。扩展策略可为指定的对象添加使用该列表中特权的能力。当前网络端口、UID 和文件对象支持扩展策略。

例如，

{file_dac_read}:/var/core/*
  访问 /var/core 下的文件时允许使用特权 file_dac_read。

{net_privaddr}:80/tcp,{net_privaddr}:443/tcp
  允许进程将网络端点绑定到 TCP 端口 80 和 443。

{proc_setid}:80-100
  允许进程将 UID 更改为 UID 80 到 100（包括两端）。

{proc_setid}:casper
  允许进程将 UID 更改为用户名 "casper" 的 UID。

安装扩展策略时，扩展策略中列出的特权集将从可继承特权集中删除，并随后从允许特权集和有效特权集中删除，除非该策略中设置的特权包括该区域中可用的所有特权。例如，当安装 {zone}:/etc/shadow 格式的扩展策略时，将不会删除任何特权；而如果扩展策略的格式为 {file_dac_read}:/etc/shadow，将会删除 PRIV_FILE_DAC_READ 特权。

仅当有效特权集中缺少相应特权时，扩展策略才会生效。

虽然可以指定 {all}:/some/file 这样的扩展策略，但系统仍会对某些应用进行限制，例如添加 setuid 位。

当进程安装扩展策略时，该策略中列出的所有特权都需要在该进程中有效。例如，当使用 ppriv(1) 安装策略时，需要所有特权都列在其有效特权集中。不会对作为 ppriv 的对象的进程应用此类限制。但是，其限制特权集将覆盖该扩展策略中的所有特权。

在某些上下文中，需要对部分扩展策略语法进行转义。例如，在 exec_attr(4) 中，需要使用反斜杠 (\) 转义分号 (;)。文件名中的某些字符可能也需要使用反斜杠进行转义，具体取决于上下文。

会在文件系统中的每一层对扩展策略进行评估；对于 lofs(7FS) 文件系统，指定的策略需要考虑以下方面：该策略需要同时指定 lofs 文件系统和底层文件系统。

以下列表包含对象类型和相关特权。

对象	语法	特权
用户名	name	proc_setid
Uid	uid	proc_setid
uid 范围	uid1-uid2	proc_setid
网络端口[1]	port/udp, port/tcp, port/sctp, port/*	net_privaddr
端口范围	port1-port2/<proto>	net_privaddr
文件名	pathname	文件特权 proc_exec
通配符[2]	pathname*	文件特权 proc_exec

[1] 如 services(4) 中所定义的数字

[2] 匹配以指定路径名开头的所有文件名

特权调试

当系统调用由于权限错误失败时，导致该问题的原因并不总是显而易见。要调试此类问题，可以使用名为**特权调试**的工具。当对进程启用特权调试时，内核会在进程的控制终端报告缺少的特权。（使用 ppriv(1) 的 -D 选项可对进程启用调试。）此外，管理员可以通过以下方式设置 system(4) 变量 priv_debug 来启用系统范围的特权调试：

```
set priv_debug = 1
```

在运行系统上，可以使用 mdb(1) 更改此变量。

特权管理

使用 usermod(1M) 或 rolemod(1M) 可分别为用户或角色分配特权或修改其特权。使用 ppriv(1) 可枚举系统上支持的特权，使用 truss(1) 可确定程序需要哪些特权。

另请参见

mdb(1)、ppriv(1)、add_drv(1M)、ifconfig(1M)、lockd(1M)、nfsd(1M)、pppd(1M)、rem_drv(1M)

《Oracle Solaris 11.1 管理：安全服务》

附注

删除进程的任何基本特权会使进程处于不符合标准的状态，可能导致意外的应用程序故障，只能在完全了解可能的副作用的情况下执行该操作。

引用名

prof – profile within a function

用法概要

```
#define MARK
#include <prof.h>

void MARK(name);
```

描述

MARK introduces a mark called *name* that is treated the same as a function entry point. Execution of the mark adds to a counter for that mark, and program-counter time spent is accounted to the immediately preceding mark or to the function if there are no preceding marks within the active function.

name may be any combination of letters, numbers, or underscores. Each *name* in a single compilation must be unique, but may be the same as any ordinary program symbol.

For marks to be effective, the symbol MARK must be defined before the header prof.h is included, either by a preprocessor directive as in the synopsis, or by a command line argument:

```
cc –p –DMARK work.c
```

If MARK is not defined, the MARK(*name*) statements may be left in the source files containing them and are ignored. prof –g must be used to get information on all labels.

示例

In this example, marks can be used to determine how much time is spent in each loop. Unless this example is compiled with MARK defined on the command line, the marks are ignored.

```
#include <prof.h>
work( )
{
        int i, j;
        . . .
        MARK(loop1);
        for (i = 0; i < 2000; i++) {
                . . .
        }
        MARK(loop2);
        for (j = 0; j < 2000; j++) {
                . . .
        }
}
```

另请参见

profil(2), monitor(3C)

引用名

rbac, RBAC – 基于角色的访问控制

描述

基于角色的访问控制允许系统管理员将部分系统的管理控制委托给用户。可以通过以下两种方式使用户能够以附加特权运行命令：

- 将配置文件直接分配给用户，在此情况下无需进行其他验证
- 创建角色并将配置文件分配给角色。此外，还可用于针对用户构建限制性环境，使用户无法运行通常允许其运行的命令。

配置文件

配置文件是命令和授权的命名集合，其中的命令和授权是以附加特权和/或真实有效的特定 UID 与 GID 运行的。例如，大部分打印机系统可通过运行带 UID 的 lp 命令或通过 lp 命令进行管理。某些命令需要 privileges(5) 中定义的特权方可运行。例如，“Process Management”（进程管理）配置文件允许用户以 proc_owner 特权运行 kill 命令，以便其将信号发送给它并不拥有的进程。

有关管理员如何扩展系统提供的配置文件并创建自己的配置文件的信息，请参见 exec_attr(4) 和 prof_attr(4)。配置文件配置可存储在任何当前支持的名称服务中（文件、NIS、LDAP）。

配置文件还可搭配服务管理工具 (Service Management Facility, SMF) 使用，以控制运行服务时所使用的特权和 UID/GID。有关更多信息，请参见 smf_security(5)。

角色

角色是一种特殊的共享帐户，无法直接登录系统，而该系统只能由授权用户使用 su(1M) 命令或以 ssh(1) 通过网络（使用基于主机的验证或 GSS-API 验证时）进行访问。角色无法以 rlogin(1)、telnet(1) 或 gdm 进行登录。

角色与普通用户一样具有 UID、口令和起始目录。可以使用用户自己的口令或按角色的口令（user_attr(4) 中的 roleauth 关键字控制基于角色的行为）对角色进行验证。通常，角色的登录 shell 是授予其一个或多个配置文件的配置文件 shell 之一（pfsh(1)、pfksh(1)、pfcsh(1)），从而允许角色始终以特权执行命令。

一般情况下，只有在需要共享帐户环境时才需要角色。通常将配置文件直接分配给用户即可。

可使用 usermod(1M) 命令将 root 用户配置为角色。这样可确保即使 root 口令更广为人知，也只有授权用户才能成为 root 用户。

```
# usermod -K type=role root
```

将 root 设置为角色不会限制对单一用户模式的访问。应使用其他方法保护系统控制台，如使用 eeprom(1M) 设置安全口令。

授权

授权是一个唯一字符串，代表用户执行某些操作或某类操作的权限。通常只有始终以某些特权运行的程序会检查授权，例如 setuid(2) 程序（如 cdrw(1) 或系统 cron(1M) 守护进程）。

授权定义存储在 auth_attr(4) 数据库中。对于编程授权检查，只有授权名称才很重要。

auth_attr 数据库中部分典型值如下所示：

```
solaris.jobs.:::Cron and At Jobs::help=JobHeader.html
solaris.jobs.grant:::Delegate Cron & At \
    Administration::help=JobsGrant.html
solaris.jobs.admin:::Manage All Jobs::help=AuthJobsAdmin.html
solaris.jobs.user:::Cron & At User::help=JobsUser.html
```

以 grant 后缀结尾的授权名称字符串为特殊授权，可让用户将具有相同前缀和功能区域的授权委托给其他用户。

所有以 solaris 开头的授权名称均会保留以供操作系统供应商进行分配。开发人员和管理员可以创建自己的顶层名称空间；建议使用唯一标识符，如公司名称、DNS 域名或应用程序名称。

授权检查

要通过 C 代码检查授权，开发人员应使用 chkauthattr(3C) 库函数，该函数将会验证用户是否具有给定的授权。

可在 Shell 脚本中明确检查授权，方法是检查 auths(1) 实用程序的输出。例如，

```
for auth in ‘auths       | tr , " "‘ NOTFOUND
do
    ["$auth" = "solaris.date" ] && break      # authorization found
done

if [ "$auth" != "solaris.date" ]
then
    echo >&2 "$PROG: ERROR: you are not authorized to set the date"
    exit 1
fi
```

授权还可供服务管理工具 (Service Management Facility, SMF) 用于控制可更改服务状态或重新配置服务的用户。有关更多信息，请参见 smf_security(5)。

与 sudo(1M) 进行比较

Solaris 中的 RBAC 可提供一组与 sudo(1M) 类似的功能，该命令通常随 UNIX 或 UNIX 型系统一起提供。它是在 Solaris 的配套 CD 上提供的。

Solaris RBAC 和 sudo 之间最显著的区别之一就是验证模型。在 sudo 中，用户以自身身份重新进行验证。在 Solaris RBAC 中，无需进行其他验证（当配置文件直接分配给用户时），或用户针对称为角色的共享帐户进行验证。

使用 sudo 中的 NOPASSWD 功能类似于将配置文件分配给用户并让用户使用 pfexec(1) 执行命令。例如，如果 sudoers(4) 允许用户以 UID 0 身份运行 kill(1) 但不进行验证 (NOPASSWD)，则用户可运行：

```
$ sudo kill -HUP 1235
```

在 Solaris RBAC 中，如果用户具有正常（即，无配置文件）登录 shell，则通过向用户分配 “Process Management”（进程管理）配置文件，用户将执行等效操作并使用 pfexec，如下所示：

```
$ pfexec kill -HUP 1235
```

如果用户将某个配置文件 shell（如 pfsh）作为登录 shell，则 kill 始终会使用附加特权来运行，而不需要“前缀”。例如，

```
$ kill -HUP 1235
```

RBAC 角色在概念上与 sudoers(4) 中的 User_Alias 类似，只是需要角色口令而不是用户口令。

RBAC 中的执行配置文件 exec_attr(4) 条目与 sudoers 中的 Cmnd_Alias 类似。

目前，Solaris RBAC 中并没有 Host_Aliassudo(1M) 功能的等效项。

另请参见

auths(1)、ld.so.1(1)、pfcsh(1)、pfexec(1)、pfksh(1)、pfsh(1)、roles(1)、sudo(1M)、exec_attr(

《Oracle Solaris 11.1 管理：安全服务》

引用名

regex – internationalized basic and extended regular expression matching

描述

Regular Expressions (REs) provide a mechanism to select specific strings from a set of character strings. The Internationalized Regular Expressions described below differ from the Simple Regular Expressions described on the `regexp(5)` manual page in the following ways:

- both Basic and Extended Regular Expressions are supported
- the Internationalization features—character class, equivalence class, and multi-character collation—are supported.

The Basic Regular Expression (BRE) notation and construction rules described in the `BASIC REGULAR EXPRESSIONS` section apply to most utilities supporting regular expressions. Some utilities, instead, support the Extended Regular Expressions (ERE) described in the `EXTENDED REGULAR EXPRESSIONS` section; any exceptions for both cases are noted in the descriptions of the specific utilities using regular expressions. Both BREs and EREs are supported by the Regular Expression Matching interfaces `regcomp(3C)` and `regexec(3C)`.

Basic Regular Expressions

BREs Matching a Single Character

A BRE ordinary character, a special character preceded by a backslash, or a period matches a single character. A bracket expression matches a single character or a single collating element. See `RE Bracket Expression`, below.

BRE Ordinary Characters

An ordinary character is a BRE that matches itself: any character in the supported character set, except for the BRE special characters listed in `BRE Special Characters`, below.

The interpretation of an ordinary character preceded by a backslash (\) is undefined, except for:

1. the characters), (, {, and }
2. the digits 1 to 9 inclusive (see `BREs Matching Multiple Characters`, below)
3. a character inside a bracket expression.

BRE Special Characters

A BRE *special character* has special properties in certain contexts. Outside those contexts, or when preceded by a backslash, such a character will be a BRE that matches the special character itself. The BRE special characters and the contexts in which they have their special meaning are:

. [\ The period, left-bracket, and backslash are special except when used in a bracket expression (see `RE Bracket Expression`, below). An expression containing a [that is not preceded by a backslash and is not part of a bracket expression produces undefined results.

* The asterisk is special except when used:

- in a bracket expression
- as the first character of an entire BRE (after an initial ^, if any)

- as the first character of a subexpression (after an initial ^, if any); see `BREs Matching Multiple Characters`, below.

^ The circumflex is special when used:

- as an anchor (see `BRE Expression Anchoring`, below).
- as the first character of a bracket expression (see `RE Bracket Expression`, below).

$ The dollar sign is special when used as an anchor.

Periods in BREs

A period (`.`), when used outside a bracket expression, is a BRE that matches any character in the supported character set except NUL.

RE Bracket Expression

A bracket expression (an expression enclosed in square brackets, [ ]) is an RE that matches a single collating element contained in the non-empty set of collating elements represented by the bracket expression.

The following rules and definitions apply to bracket expressions:

1. A *bracket expression* is either a matching list expression or a non-matching list expression. It consists of one or more expressions: collating elements, collating symbols, equivalence classes, character classes, or range expressions (see rule 7 below). Portable applications must not use range expressions, even though all implementations support them. The right-bracket (]) loses its special meaning and represents itself in a bracket expression if it occurs first in the list (after an initial circumflex (^), if any). Otherwise, it terminates the bracket expression, unless it appears in a collating symbol (such as [.].]) or is the ending right-bracket for a collating symbol, equivalence class, or character class. The special characters:

   ```
   .    *    [    \
   ```

   (period, asterisk, left-bracket and backslash, respectively) lose their special meaning within a bracket expression.

   The character sequences:

   ```
   [.   [=    [:
   ```

   (left-bracket followed by a period, equals-sign, or colon) are special inside a bracket expression and are used to delimit collating symbols, equivalence class expressions, and character class expressions. These symbols must be followed by a valid expression and the matching terminating sequence .], =] or :], as described in the following items.
2. A *matching list* expression specifies a list that matches any one of the expressions represented in the list. The first character in the list must not be the circumflex. For example, `[abc]` is an RE that matches any of the characters `a`, `b` or `c`.
3. A *non-matching list* expression begins with a circumflex (^), and specifies a list that matches any character or collating element except for the expressions represented in the list after the leading circumflex. For example, [^abc] is an RE that matches any character or

collating element except the characters `a, b`, or `c`. The circumflex will have this special meaning only when it occurs first in the list, immediately following the left-bracket.

4. A *collating symbol* is a collating element enclosed within bracket-period ([..]) delimiters. Multi-character collating elements must be represented as collating symbols when it is necessary to distinguish them from a list of the individual characters that make up the multi-character collating element. For example, if the string `ch` is a collating element in the current collation sequence with the associated collating symbol <ch>, the expression [[.ch.]] will be treated as an RE matching the character sequence `ch`, while [ch] will be treated as an RE matching `c` or `h`. Collating symbols will be recognized only inside bracket expressions. This implies that the RE `[[.ch.]]*c` matches the first to fifth character in the string chchch. If the string is not a collating element in the current collating sequence definition, or if the collating element has no characters associated with it, the symbol will be treated as an invalid expression.
5. An *equivalence class expression* represents the set of collating elements belonging to an equivalence class. Only primary equivalence classes will be recognised. The class is expressed by enclosing any one of the collating elements in the equivalence class within bracket-equal ([==]) delimiters. For example, if a and b belong to the same equivalence class, then [[=a=]b], [[==]b] and [[==]b] will each be equivalent to [ab]. If the collating element does not belong to an equivalence class, the equivalence class expression will be treated as a *collating symbol.*
6. A *character class expression* represents the set of characters belonging to a character class, as defined in the `LC_CTYPE` category in the current locale. All character classes specified in the current locale will be recognized. A character class expression is expressed as a character class name enclosed within bracket-colon ([::]) delimiters.

   The following character class expressions are supported in all locales:

`[:alnum:]`	`[:cntrl:]`	`[:lower:]`	`[:space:]`
`[:alpha:]`	`[:digit:]`	`[:print:]`	`[:upper:]`
`[:blank:]`	`[:graph:]`	`[:punct:]`	`[:xdigit:]`

   In addition, character class expressions of the form:

   ```
   [:name:]
   ```

   are recognized in those locales where the *name* keyword has been given a `charclass` definition in the `LC_CTYPE` category.
7. A *range expression* represents the set of collating elements that fall between two elements in the current collation sequence, inclusively. It is expressed as the starting point and the ending point separated by a hyphen (–).

Range expressions must not be used in portable applications because their behavior is dependent on the collating sequence. Ranges will be treated according to the current collating sequence, and include such characters that fall within the range based on that collating sequence, regardless of character values. This, however, means that the interpretation will differ depending on collating sequence. If, for instance, one collating sequence defines as a variant of `a`, while another defines it as a letter following `z`, then the expression [–z] is valid in the first language and invalid in the second.

In the following, all examples assume the collation sequence specified for the POSIX locale, unless another collation sequence is specifically defined.

The starting range point and the ending range point must be a collating element or collating symbol. An equivalence class expression used as a starting or ending point of a range expression produces unspecified results. An equivalence class can be used portably within a bracket expression, but only outside the range. For example, the unspecified expression [[=e=]–f] should be given as [[=e=]e–f]. The ending range point must collate equal to or higher than the starting range point; otherwise, the expression will be treated as invalid. The order used is the order in which the collating elements are specified in the current collation definition. One-to-many mappings (see `locale(5)`) will not be performed. For example, assuming that the character eszet is placed in the collation sequence after `r` and `s`, but before `t`, and that it maps to the sequence `ss` for collation purposes, then the expression [r–s] matches only `r` and `s`, but the expression [s–t] matches `s`, `beta`, or `t`.

The interpretation of range expressions where the ending range point is also the starting range point of a subsequent range expression (for instance [a–m–o]) is undefined.

The hyphen character will be treated as itself if it occurs first (after an initial ^, if any) or last in the list, or as an ending range point in a range expression. As examples, the expressions [–ac] and [ac–] are equivalent and match any of the characters `a`, `c`, or –; [^–ac] and [^ac–] are equivalent and match any characters except `a`, `c`, or –; the expression [%– –] matches any of the characters between % and – inclusive; the expression [– –@] matches any of the characters between – and @ inclusive; and the expression [a– –@] is invalid, because the letter `a` follows the symbol – in the POSIX locale. To use a hyphen as the starting range point, it must either come first in the bracket expression or be specified as a collating symbol, for example: [][.–.]–0], which matches either a right bracket or any character or collating element that collates between hyphen and 0, inclusive.

If a bracket expression must specify both – and], the] must be placed first (after the ^, if any) and the – last within the bracket expression.

Note: Latin-1 characters such as or are not printable in some locales, for example, the `ja` locale.

BREs Matching Multiple Characters

The following rules can be used to construct BREs matching multiple characters from BREs matching a single character:

1. The concatenation of BREs matches the concatenation of the strings matched by each component of the BRE.

2. A *subexpression* can be defined within a BRE by enclosing it between the character pairs \(and \) . Such a subexpression matches whatever it would have matched without the \(and \), except that anchoring within subexpressions is optional behavior; see `BRE Expression Anchoring`, below. Subexpressions can be arbitrarily nested.
3. The *back-reference* expression \n matches the same (possibly empty) string of characters as was matched by a subexpression enclosed between \(and \) preceding the \n. The character *n* must be a digit from 1 to 9 inclusive, *n*th subexpression (the one that begins with the *n*th \(and ends with the corresponding paired \)). The expression is invalid if less than *n* subexpressions precede the \n. For example, the expression `^\(.*\)\1$` matches a line consisting of two adjacent appearances of the same string, and the expression `\(a\)*\1` fails to match `a`. The limit of nine back-references to subexpressions in the RE is based on the use of a single digit identifier. This does not imply that only nine subexpressions are allowed in REs. The following is a valid BRE with ten subexpressions:

   ```
   \(\(\(ab\)*c\)*d\)\(ef\)*\(gh\)\{2\}\(ij\)*\(kl\)*\(mn\)*\(op\)*\(qr\)*
   ```
4. When a BRE matching a single character, a subexpression or a back-reference is followed by the special character asterisk (*), together with that asterisk it matches what zero or more consecutive occurrences of the BRE would match. For example, `[ab]*` and `[ab][ab]` are equivalent when matching the string `ab`.
5. When a BRE matching a single character, a subexpression, or a back-reference is followed by an *interval expression* of the format \{*m*\}, \{*m*,\} or \{*m*,*n*\}, together with that interval expression it matches what repeated consecutive occurrences of the BRE would match. The values of *m* and *n* will be decimal integers in the range 0 ≤ *m* ≤ *n* ≤ {`RE_DUP_MAX`}, where *m* specifies the exact or minimum number of occurrences and *n* specifies the maximum number of occurrences. The expression \{*m*\} matches exactly *m* occurrences of the preceding BRE, \{*m*,\} matches at least *m* occurrences and \{*m*,*n*\} matches any number of occurrences between *m* and *n*, inclusive.

   For example, in the string `abababccccccd`, the BRE c\{3\} is matched by characters seven to nine, the BRE \(ab\)\{4,\} is not matched at all and the BRE c\{1,3\}d is matched by characters ten to thirteen.

The behavior of multiple adjacent duplication symbols (* and intervals) produces undefined results.

BRE Precedence

The order of precedence is as shown in the following table:

BRE Precedence (from high to low)	
collation-related bracket symbols	[= =] [: :] [. .]
escaped characters	\<*special character*>
bracket expression	[ ]
subexpressions/back-references	\(\) \n

single-character-BRE duplication	* \{*m*,*n*\}
concatenation	
anchoring	^ $

BRE Expression Anchoring

A BRE can be limited to matching strings that begin or end a line; this is called *anchoring*. The circumflex and dollar sign special characters will be considered BRE anchors in the following contexts:

1. A circumflex (`^`) is an anchor when used as the first character of an entire BRE. The implementation may treat circumflex as an anchor when used as the first character of a subexpression. The circumflex will anchor the expression to the beginning of a string; only sequences starting at the first character of a string will be matched by the BRE. For example, the BRE `^ab` matches `ab` in the string `abcdef,` but fails to match in the string `cdefab`. A portable BRE must escape a leading circumflex in a subexpression to match a literal circumflex.
2. A dollar sign (`$`) is an anchor when used as the last character of an entire BRE. The implementation may treat a dollar sign as an anchor when used as the last character of a subexpression. The dollar sign will anchor the expression to the end of the string being matched; the dollar sign can be said to match the end-of-string following the last character.
3. A BRE anchored by both `^` and `$` matches only an entire string. For example, the BRE `^abcdef$` matches strings consisting only of `abcdef`.
4. `^` and `$` are not special in subexpressions.

Note: The Solaris implementation does not support anchoring in BRE subexpressions.

Extended Regular Expressions

The rules specifed for BREs apply to Extended Regular Expressions (EREs) with the following exceptions:

- The characters `|`, `+`, and `?` have special meaning, as defined below.
- The `{` and `}` characters, when used as the duplication operator, are not preceded by backslashes. The constructs `\{` and `\}` simply match the characters `{` and `}`, respectively.
- The back reference operator is not supported.
- Anchoring (`^$`) is supported in subexpressions.

EREs Matching a Single Character

An ERE ordinary character, a special character preceded by a backslash, or a period matches a single character. A bracket expression matches a single character or a single collating element. An *ERE matching a single character* enclosed in parentheses matches the same as the ERE without parentheses would have matched.

ERE Ordinary Characters

An *ordinary character* is an ERE that matches itself. An ordinary character is any character in the supported character set, except for the ERE special characters listed in `ERE Special Characters` below. The interpretation of an ordinary character preceded by a backslash (`\`) is undefined.

ERE Special Characters

An *ERE special character* has special properties in certain contexts. Outside those contexts, or when preceded by a backslash, such a character is an ERE that matches the special character itself. The extended regular expression special characters and the contexts in which they have their special meaning are:

. [\ (The period, left-bracket, backslash, and left-parenthesis are special except when used in a bracket expression (see `RE Bracket Expression`, above). Outside a bracket expression, a left-parenthesis immediately followed by a right-parenthesis produces undefined results.

) The right-parenthesis is special when matched with a preceding left-parenthesis, both outside a bracket expression.

* + ? { The asterisk, plus-sign, question-mark, and left-brace are special except when used in a bracket expression (see `RE Bracket Expression`, above). Any of the following uses produce undefined results:

- if these characters appear first in an ERE, or immediately following a vertical-line, circumflex or left-parenthesis
- if a left-brace is not part of a valid interval expression.

| The vertical-line is special except when used in a bracket expression (see `RE Bracket Expression`, above). A vertical-line appearing first or last in an ERE, or immediately following a vertical-line or a left-parenthesis, or immediately preceding a right-parenthesis, produces undefined results.

^ The circumflex is special when used:

- as an anchor (see `ERE Expression Anchoring`, below).
- as the first character of a bracket expression (see `RE Bracket Expression`, above).

$ The dollar sign is special when used as an anchor.

Periods in EREs

A period (`.`), when used outside a bracket expression, is an ERE that matches any character in the supported character set except NUL.

ERE Bracket Expression

The rules for ERE Bracket Expressions are the same as for Basic Regular Expressions; see `RE Bracket Expression`, above).

EREs Matching Multiple Characters

The following rules will be used to construct EREs matching multiple characters from EREs matching a single character:

1. A *concatenation of EREs* matches the concatenation of the character sequences matched by each component of the ERE. A concatenation of EREs enclosed in parentheses matches whatever the concatenation without the parentheses matches. For example, both the ERE `cd` and the ERE `(cd)` are matched by the third and fourth character of the string `abcdefabcdef`.

2. When an ERE matching a single character or an ERE enclosed in parentheses is followed by the special character plus-sign (+), together with that plus-sign it matches what one or more consecutive occurrences of the ERE would match. For example, the ERE `b+(bc)` matches the fourth to seventh characters in the string `acabbbcde; [ab] +` and `[ab][ab]*` are equivalent.
3. When an ERE matching a single character or an ERE enclosed in parentheses is followed by the special character asterisk (`*`), together with that asterisk it matches what zero or more consecutive occurrences of the ERE would match. For example, the ERE `b*c` matches the first character in the string `cabbbcde,` and the ERE `b*cd` matches the third to seventh characters in the string `cabbbcdebbbbbbcdbc`. And, `[ab]*` and `[ab][ab]` are equivalent when matching the string `ab`.
4. When an ERE matching a single character or an ERE enclosed in parentheses is followed by the special character question-mark (`?`), together with that question-mark it matches what zero or one consecutive occurrences of the ERE would match. For example, the ERE `b?c` matches the second character in the string `acabbbcde`.
5. When an ERE matching a single character or an ERE enclosed in parentheses is followed by an *interval expression* of the format {*m*}, {*m*,} or {*m*,*n*}, together with that interval expression it matches what repeated consecutive occurrences of the ERE would match. The values of *m* and *n* will be decimal integers in the range 0 ≤ *m* ≤ *n* ≤ `{RE_DUP_MAX}`, where *m* specifies the exact or minimum number of occurrences and *n* specifies the maximum number of occurrences. The expression {*m*} matches exactly *m* occurrences of the preceding ERE, {*m*,} matches at least *m* occurrences and {*m*,*n*} matches any number of occurrences between *m* and *n*, inclusive.

For example, in the string `abababccccccd` the ERE c{3} is matched by characters seven to nine and the ERE (ab){2,} is matched by characters one to six.

The behavior of multiple adjacent duplication symbols `(+, *, ?` and intervals) produces undefined results.

ERE Alternation

Two EREs separated by the special character vertical-line (|) match a string that is matched by either. For example, the ERE a((bc)|d) matches the string abc and the string ad. Single characters, or expressions matching single characters, separated by the vertical bar and enclosed in parentheses, will be treated as an ERE matching a single character.

ERE Precedence

The order of precedence will be as shown in the following table:

ERE Precedence (from high to low)	
collation-related bracket symbols	[= =] [: :] [. .]
escaped characters	\<*special character*>
bracket expression	[ ]

grouping	( )
single-character-ERE duplication	* + ? {*m*,*n*}
concatenation	
anchoring	^ $
alternation	|

For example, the ERE `abba | cde` matches either the string `abba` or the string `cde` (rather than the string `abbade` or `abbcde`, because concatenation has a higher order of precedence than alternation).

ERE Expression Anchoring

An ERE can be limited to matching strings that begin or end a line; this is called *anchoring*. The circumflex and dollar sign special characters are considered ERE anchors when used anywhere outside a bracket expression. This has the following effects:

1. A circumflex (^) outside a bracket expression anchors the expression or subexpression it begins to the beginning of a string; such an expression or subexpression can match only a sequence starting at the first character of a string. For example, the EREs ^ab and (^ab) match ab in the string abcdef, but fail to match in the string cdefab, and the ERE a^b is valid, but can never match because the `a` prevents the expression ^b from matching starting at the first character.
2. A dollar sign (`$`) outside a bracket expression anchors the expression or subexpression it ends to the end of a string; such an expression or subexpression can match only a sequence ending at the last character of a string. For example, the EREs ef$ and (ef$) match ef in the string abcdef, but fail to match in the string cdefab, and the ERE e$f is valid, but can never match because the `f` prevents the expression e$ from matching ending at the last character.

另请参见

`localedef`(1), `regcomp`(3C), `attributes`(5), `environ`(5), `locale`(5), `regexp`(5)

引用名

regexp, compile, step, advance – simple regular expression compile and match routines

用法概要

```
#define INIT declarations
#define GETC(void) getc code
#define PEEKC(void) peekc code
#define UNGETC(void) ungetc code
#define RETURN(ptr) return code
#define ERROR(val) error code

extern char *loc1, *loc2, *locs;

#include <regexp.h>

char *compile(char *instring, char *expbuf, const char *endfug, int eof);

int step(const char *string, const char *expbuf);

int advance(const char *string, const char *expbuf);
```

描述

Regular Expressions (REs) provide a mechanism to select specific strings from a set of character strings. The Simple Regular Expressions described below differ from the Internationalized Regular Expressions described on the regex(5) manual page in the following ways:

- only Basic Regular Expressions are supported
- the Internationalization features—character class, equivalence class, and multi-character collation—are not supported.

The functions step(), advance(), and compile() are general purpose regular expression matching routines to be used in programs that perform regular expression matching. These functions are defined by the <regexp.h> header.

The functions step() and advance() do pattern matching given a character string and a compiled regular expression as input.

The function compile() takes as input a regular expression as defined below and produces a compiled expression that can be used with step() or advance().

Basic Regular Expressions

A regular expression specifies a set of character strings. A member of this set of strings is said to be matched by the regular expression. Some characters have special meaning when used in a regular expression; other characters stand for themselves.

The following *one-character REs* match a *single* character:

1.1 An ordinary character (*not* one of those discussed in 1.2 below) is a one-character RE that matches itself.

1.2 A backslash (\ ) followed by any special character is a one-character RE that matches the special character itself. The special characters are:

a. `.`, `*`, `[` , and `\` (period, asterisk, left square bracket, and backslash, respectively), which are always special, *except* when they appear within square brackets (`[ ]`; see 1.4 below).

b. `^` (caret or circumflex), which is special at the *beginning* of an *entire* RE (see 4.1 and 4.3 below), or when it immediately follows the left of a pair of square brackets (`[ ]`) (see 1.4 below).

c. `$` (dollar sign), which is special at the `end` of an *entire* RE (see 4.2 below).

d. The character used to bound (that is, delimit) an entire RE, which is special for that RE (for example, see how slash (`/`) is used in the `g` command, below.)

1.3 A period (`.`) is a one-character RE that matches any character except new-line.

1.4 A non-empty string of characters enclosed in square brackets (`[ ]`) is a one-character RE that matches *any one* character in that string. If, however, the first character of the string is a circumflex (`^`), the one-character RE matches any character *except* new-line and the remaining characters in the string. The `^` has this special meaning *only* if it occurs first in the string. The minus (–) may be used to indicate a range of consecutive characters; for example, `[0–9]` is equivalent to `[0123456789]`. The – loses this special meaning if it occurs first (after an initial `^`, if any) or last in the string. The right square bracket (`]`) does not terminate such a string when it is the first character within it (after an initial `^`, if any); for example, `[ ]a–f]` matches either a right square bracket (`]`) or one of the ASCII letters `a` through `f` inclusive. The four characters listed in 1.2.a above stand for themselves within such a string of characters.

The following rules may be used to construct REs from one-character REs:

2.1 A one-character RE is a RE that matches whatever the one-character RE matches.

2.2 A one-character RE followed by an asterisk (`*`) is a RE that matches `0` or more occurrences of the one-character RE. If there is any choice, the longest leftmost string that permits a match is chosen.

2.3 A one-character RE followed by `\{`*m*`\}`, `\{`*m,*`\}`, or `\{`*m,n*`\}` is a RE that matches a *range* of occurrences of the one-character RE. The values of *m* and *n* must be non-negative integers less than 256; `\{`*m*`\}` matches *exactly m* occurrences; `\{`*m,*`\}` matches *at least m* occurrences; `\{`*m,n*`\}` matches *any number* of occurrences *between m* and *n* inclusive. Whenever a choice exists, the RE matches as many occurrences as possible.

2.4 The concatenation of REs is a RE that matches the concatenation of the strings matched by each component of the RE.

2.5 A RE enclosed between the character sequences `\ (` and `\ )` is a RE that matches whatever the unadorned RE matches.

2.6 The expression `\ n` matches the same string of characters as was matched by an expression enclosed between `\ (` and `\ )` *earlier* in the same RE. Here *n* is a digit; the sub-expression specified is that beginning with the *n*-th occurrence of `\ (` counting from the left. For example, the expression `^ \ ( . * \ ) \ 1 $` matches a line consisting of two repeated appearances of the same string.

An RE may be constrained to match words.

3.1 `\ <` constrains a RE to match the beginning of a string or to follow a character that is not a digit, underscore, or letter. The first character matching the RE must be a digit, underscore, or letter.

3.2 `\ >` constrains a RE to match the end of a string or to precede a character that is not a digit, underscore, or letter.

An *entire RE* may be constrained to match only an initial segment or final segment of a line (or both).

4.1 A circumflex (`^`) at the beginning of an entire RE constrains that RE to match an *initial* segment of a line.

4.2 A dollar sign (`$`) at the end of an entire RE constrains that RE to match a *final* segment of a line.

4.3 The construction `^`*entire RE* `$` constrains the entire RE to match the entire line.

The null RE (for example, `//`) is equivalent to the last RE encountered.

Addressing with REs

Addresses are constructed as follows:

1. The character "`.`" addresses the current line.
2. The character "`$`" addresses the last line of the buffer.
3. A decimal number *n* addresses the *n*-th line of the buffer.
4. *'x* addresses the line marked with the mark name character *x*, which must be an ASCII lower-case letter (`a`–`z`). Lines are marked with the `k` command described below.
5. A RE enclosed by slashes (`/`) addresses the first line found by searching *forward* from the line *following* the current line toward the end of the buffer and stopping at the first line containing a string matching the RE. If necessary, the search wraps around to the beginning of the buffer and continues up to and including the current line, so that the entire buffer is searched.
6. A RE enclosed in question marks (`?`) addresses the first line found by searching *backward* from the line *preceding* the current line toward the beginning of the buffer and stopping at the first line containing a string matching the RE. If necessary, the search wraps around to the end of the buffer and continues up to and including the current line.

7. An address followed by a plus sign (+) or a minus sign (–) followed by a decimal number specifies that address plus (respectively minus) the indicated number of lines. A shorthand for .+5 is .5.
8. If an address begins with + or –, the addition or subtraction is taken with respect to the current line; for example, –5 is understood to mean `.–5`.
9. If an address ends with + or –, then 1 is added to or subtracted from the address, respectively. As a consequence of this rule and of Rule 8, immediately above, the address – refers to the line preceding the current line. (To maintain compatibility with earlier versions of the editor, the character ^ in addresses is entirely equivalent to –.) Moreover, trailing + and – characters have a cumulative effect, so — refers to the current line less 2.
10. For convenience, a comma (`,`) stands for the address pair `1,$`, while a semicolon (`;`) stands for the pair `.,$`.

Characters With Special Meaning

Characters that have special meaning except when they appear within square brackets (`[ ]`) or are preceded by `\` are: `.`, `*`, `[` , `\` . Other special characters, such as `$` have special meaning in more restricted contexts.

The character `^` at the beginning of an expression permits a successful match only immediately after a newline, and the character `$` at the end of an expression requires a trailing newline.

Two characters have special meaning only when used within square brackets. The character – denotes a range, `[` *c*–*c*`]`, unless it is just after the open bracket or before the closing bracket, `[` –*c*`]` or `[` *c*–`]` in which case it has no special meaning. When used within brackets, the character `^` has the meaning *complement of* if it immediately follows the open bracket (example: `[^`*c*`]`); elsewhere between brackets (example: `[`*c*`^]`) it stands for the ordinary character `^`.

The special meaning of the `\` operator can be escaped only by preceding it with another `\` , for example `\\` .

Macros

Programs must have the following five macros declared before the `#include <regexp.h>` statement. These macros are used by the `compile()` routine. The macros `GETC`, `PEEKC`, and `UNGETC` operate on the regular expression given as input to `compile()`.

`GETC` This macro returns the value of the next character (byte) in the regular expression pattern. Successive calls to `GETC` should return successive characters of the regular expression.

`PEEKC` This macro returns the next character (byte) in the regular expression. Immediately successive calls to `PEEKC` should return the same character, which should also be the next character returned by `GETC`.

`UNGETC` This macro causes the argument `c` to be returned by the next call to `GETC` and `PEEKC`. No more than one character of pushback is ever needed and this character is guaranteed to be the last character read by GETC. The return value of the macro UNGETC(c) is always ignored.

RETURN(*ptr*) This macro is used on normal exit of the compile() routine. The value of the argument *ptr* is a pointer to the character after the last character of the compiled regular expression. This is useful to programs which have memory allocation to manage.

ERROR(*val*) This macro is the abnormal return from the compile() routine. The argument *val* is an error number (see ERRORS below for meanings). This call should never return.

compile()

The syntax of the compile() routine is as follows:

compile(*instring*, *expbuf*, *endbuf*, *eof*)

The first parameter, *instring*, is never used explicitly by the compile() routine but is useful for programs that pass down different pointers to input characters. It is sometimes used in the INIT declaration (see below). Programs which call functions to input characters or have characters in an external array can pass down a value of (char *)0 for this parameter.

The next parameter, *expbuf*, is a character pointer. It points to the place where the compiled regular expression will be placed.

The parameter *endbuf* is one more than the highest address where the compiled regular expression may be placed. If the compiled expression cannot fit in (endbuf–expbuf) bytes, a call to ERROR(50) is made.

The parameter *eof* is the character which marks the end of the regular expression. This character is usually a /.

Each program that includes the <regexp.h> header file must have a #define statement for INIT. It is used for dependent declarations and initializations. Most often it is used to set a register variable to point to the beginning of the regular expression so that this register variable can be used in the declarations for GETC, PEEKC, and UNGETC. Otherwise it can be used to declare external variables that might be used by GETC, PEEKC and UNGETC. (See EXAMPLES below.)

step(), advance()

The first parameter to the step() and advance() functions is a pointer to a string of characters to be checked for a match. This string should be null terminated.

The second parameter, *expbuf*, is the compiled regular expression which was obtained by a call to the function compile().

The function step() returns non-zero if some substring of *string* matches the regular expression in *expbuf* and 0 if there is no match. If there is a match, two external character pointers are set as a side effect to the call to step(). The variable loc1 points to the first character that matched the regular expression; the variable `loc2` points to the character after the last character that matches the regular expression. Thus if the regular expression matches the entire input string, `loc1` will point to the first character of *string* and `loc2` will point to the null at the end of *string*.

The function `advance()` returns non-zero if the initial substring of *string* matches the regular expression in *expbuf*. If there is a match, an external character pointer, `loc2`, is set as a side effect. The variable `loc2` points to the next character in *string* after the last character that matched.

When `advance()` encounters a `*` or `\{ \}` sequence in the regular expression, it will advance its pointer to the string to be matched as far as possible and will recursively call itself trying to match the rest of the string to the rest of the regular expression. As long as there is no match, `advance()` will back up along the string until it finds a match or reaches the point in the string that initially matched the `*` or `\{ \}`. It is sometimes desirable to stop this backing up before the initial point in the string is reached. If the external character pointer `locs` is equal to the point in the string at sometime during the backing up process, `advance()` will break out of the loop that backs up and will return zero.

The external variables `circf`, `sed`, and `nbra` are reserved.

示例

示例 1 Using Regular Expression Macros and Calls

The following is an example of how the regular expression macros and calls might be defined by an application program:

```
#define INIT       register char *sp = instring;
#define GETC()     (*sp++)
#define PEEKC()    (*sp)
#define UNGETC(c)  (—sp)
#define RETURN(c)  return;
#define ERROR(c)   regerr()

#include <regexp.h>
 . . .
      (void) compile(*argv, expbuf, &expbuf[ESIZE],'\0');
 . . .
      if (step(linebuf, expbuf))
                        succeed;
```

诊断

The function `compile()` uses the macro `RETURN` on success and the macro `ERROR` on failure (see above). The functions `step()` and `advance()` return non-zero on a successful match and zero if there is no match. Errors are:

11 range endpoint too large.

16 bad number.

25	\ *digit* out of range.
36	illegal or missing delimiter.
41	no remembered search string.
42	`\( \)` imbalance.
43	too many `\(`.
44	more than 2 numbers given in `\{ \}`.
45	`}` expected after `\`.
46	first number exceeds second in `\{ \}`.
49	`[ ]` imbalance.
50	regular expression overflow.

另请参见

`regex(5)`

引用名

resource_controls – 通过项目数据库提供的资源控制

描述

资源控制工具通过项目数据库进行配置。请参见 project(4)。您可以通过以下实用程序设置和修改资源控制：

- prctl(1)
- projadd(1M)
- projmod(1M)
- rctladm(1M)

在程序中，您可以使用 setrctl(2) 来设置资源控制值。

除了前述资源控制以外，还存在可通过 pooladm(1M) 和 poolcfg(1M) 实用程序访问的资源池。在程序中，可通过 libpool(3LIB) 库操作资源池。

下面列出了一些可用的资源控制：

`process.max-address-space`
此进程可用的最大地址空间量，即段大小的总和，以字节数表示。

`process.max-core-size`
此进程创建的最大核心文件大小，以字节数表示。

`process.max-cpu-time`
此进程可用的最长 CPU 时间，以秒数表示。

`process.max-data-size`
此进程可用的最大堆内存，以字节数表示。

`process.max-file-descriptor`
此进程可用的最大文件描述符索引，以整数表示。

`process.max-file-size`
此进程可写入的最大文件偏移，以字节数表示。

`processs.max-itimers`
允许的最大间隔计时器数目，以整数表示。

`process.max-msg-messages`
消息队列中的最大消息数（在 msgget() 时间从资源控制复制的值），表示为整数。

`process.max-msg-qbytes`
消息队列中消息的最大字节数（在 msgget() 时间从资源控制复制的值），以字节数表示。

`process.max-port-events`
每事件端口允许的最大事件数，以整数表示。

`process.max-sem-nsems`
每信号集允许的最大信号数目，以整数表示。

process.max-sem-ops
: 每 semop 调用允许的最大信号操作数（在 semget() 时间从资源控制复制的值）。以整数表示，用于指定操作数目。

process.max-sigqueue-size
: 进程可具有的待处理排队信号最大数目。

process.max-stack-size
: 此进程可用的最大栈内存区段，以字节数表示。

project.cpu-caps
: 项目可以使用的 CPU 资源最大量。使用的单位是项目中所有用户线程可使用的单个 CPU 的百分比。以整数表示。上限不适用于实时调度类中运行的线程。此资源控制不支持 syslog 操作。

project.cpu-shares
: 授予项目的 CPU 份额数，用于公平份额调度器（请参见 FSS(7)）。使用的单位是共享数目（整数）。此资源控制不支持 syslog 操作。

project.max-contracts
: 项目中允许的最大合同数目，以整数表示。

project.max-crypto-memory
: 可用于加密操作的最大内核内存量。内核中对缓冲区的分配以及与会话相关的结构分配都按照此资源控制执行。

project.max-locked-memory
: 设备驱动器和用户进程（包括 D/ISM）锁定的物理内存总量，以字节数表示。

project.max-lwps
: 项目可同时使用的最大 LWP 数目，以整数表示。

project.max-msg-ids
: 项目允许的最大消息队列 ID 数目，以整数表示。

project.max-port-ids
: 允许的最大事件端口数目，以整数表示。

project.max-processes
: 项目可同时使用的最大进程数目，以整数表示。

project.max-sem-ids
: 项目允许的最大信号 ID 数目，以整数表示。

project.max-shm-ids
: 项目允许的最大共享内存 ID 数目，以整数表示。

project.max-shm-memory
: 项目允许的共享内存总量，以字节数表示。

project.max-tasks
: 项目中允许的最大任务数目，以整数表示。

`project.pool`
将指定的资源池与项目绑定。

`rcap.max-rss`
项目中的进程可用的物理内存总量（字节）。

`task.max-cpu-time`
此任务的进程可用的最长 CPU 时间，以秒数表示。

`task.max-lwps`
此任务的进程可同时使用的最大 LWP 数目，以整数表示。

`task.max-processes`
任务可同时使用的最大进程数目，以整数表示。

以下是可用的区域范围的资源控制：

`zone.cpu-cap`
设置可供某个区域使用的 CPU 时间量的限制。使用的单位是区域中所有用户线程可使用的单个 CPU 的百分比。以整数表示。当设置有上限的区域中的项目有其自己的上限时，优先采用最小值。此资源控制不支持 `syslog` 操作。

`zone.cpu-shares`
对区域的公平份额调度器 (fair share scheduler, FSS) CPU 份额数设置限制。CPU 份额首先分配给区域，然后在区域内的项目之间进一步分配，如 `project.cpu-shares` 项中所述。以整数表示。此资源控制不支持 `syslog` 操作。

`zone.max-locked-memory`
区域可用的锁定物理内存的总量

`zone.max-lofi`
区域可用的最大 `lofi`(7D) 设备数目。

`zone.max-lwps`
通过阻止一个区域中的过多 LWP 影响其他区域来增强资源隔离。区域的 LWP 总数可以使用 `project.max-lwps` 条目在区域内的项目之间进一步细分。以整数表示。

`zone.max-msg-ids`
区域允许的最大消息队列 ID 数目，以整数表示。

`zone.max-processes`
区域可同时使用的最大进程数目，以整数表示。

`zone.max-sem-ids`
区域允许的最大信号 ID 数目，以整数表示。

`zone.max-shm-ids`
区域允许的最大共享内存 ID 数目，以整数表示。

`zone.max-shm-memory`
区域允许的共享内存总量，以字节数表示。

zone.max-swap
可用于此区域的用户进程地址空间映射和 tmpfs 挂载的交换空间总量

请参见 zones(5)。

资源控制中使用的单位

资源控制可以采用大小（字节）、时间（秒）或计数（整数）单位来表示。这些单位使用以下指定的字符串。

Category	Res Ctrl	Modifier	Scale
	Type String		
Size	bytes	B	1
		KB	2^10
		MB	2^20
		GB	2^30
		TB	2^40
		PB	2^50
		EB	2^60
Time	seconds	s	1
		Ks	10^3
		Ms	10^6
		Gs	10^9
		Ts	10^12
		Ps	10^15
		Es	10^18
Count	integer	none	1
		K	10^3
		M	10^6
		G	10^9
		T	10^12
		P	10^15
		Es	10^18

标度值可用于资源控制。以下示例显示了标度阈值：

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

在 project 文件中，值 1K 展开为 1000：

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

第二个示例使用较大的比例值：

```
process.max-file-size=(priv,5G,deny)
```

在 project 文件中，值 5G 展开为 5368709120：

```
process.max-file-size=(priv,5368709120,deny)
```

前述示例使用上表中指定的比例系数。

请注意，prctl(1)、projadd(1M) 和 projmod(1M) 命令接受单位修饰符（例如，5G）。您不能在project数据库本身中使用单位修饰符。

资源控制值和特权级别

资源控制的阈值相当于可能会引发本地操作或进行全局操作（如记录）的某个点。

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

basic
可由调用进程的所有者修改。

privileged
可由当前进程（需要 sys_resource 特权）或由 prctl(1)（需要 proc_owner 特权）修改。

system
在操作系统实例的持续时间内固定不变。

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

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

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

在指定 privileged 的特权级别时，您可以使用缩写 priv。例如：

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

针对资源控制值的全局和本地操作

针对资源控制值可执行两种类别的操作：全局操作和本地操作。

全局操作应用于系统中每个资源控制的资源控制值。您可以使用 rctladm(1M) 执行以下操作：

- 显示活动系统资源控制的全局状态。
- 设置全局日志操作。

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

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

- emerg

缺省情况下，没有资源控制违规的全局日志。

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

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

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

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

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

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

以下列出了可用于资源控制值的信号：

SIGABRT
终止进程。

SIGHUP
发送挂起信号。当载波在断开的线路上停止时出现。发送给控制终端的进程组的信号。

SIGTERM
终止进程。由软件发送的终止信号。

SIGKILL
终止进程并中止程序。

SIGSTOP
停止进程。作业控制信号。

SIGXRES
超过了资源控制限制。由资源控制功能生成。

SIGXFSZ

终止进程。超过了文件大小限制。仅可用于具有 RCTL_GLOBAL_FILE_SIZE 属性的资源控制 (process.max-file-size)。请参见 rctlblk_set_value(3C)。

SIGXCPU

终止进程。超过了 CPU 时间限制。仅可用于具有 RCTL_GLOBAL_CPUTIME 属性的资源控制 (process.max-cpu-time)。请参见 rctlblk_set_value(3C)。

资源控制标志和属性

系统的每个资源控制都有一组特定的关联属性。这组属性定义为一组标志，这些标志与此资源的所有受控实例关联。不能修改全局标志，但是可以使用 rctladm(1M) 或 setrctl(2) 系统调用检索这些标志。

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

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

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

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

全局标志表示以下内容：

lowerable

不需要超级用户特权来减小此控制的特权值。

no-deny

即使当超过阈值时，也从不拒绝对资源的访问。

cpu-time

当达到此资源的阈值时可发送 SIGXCPU。

seconds

资源控制的时间值。

使用 prctl 命令可以显示资源控制的本地值和操作。例如：

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

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

资源控制执行

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

属性

有关以下属性的说明，请参见 `attributes`(5)：

属性类型	属性值
接口稳定性	Committed（已确定）

另请参见

`prctl`(1)、`pooladm`(1M)、`poolcfg`(1M)、`projadd`(1M)、`projmod`(1M)、`rctladm`(1M)、`setrctl`(2)、`r`

《Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management》

引用名

sgml, solbook – Standard Generalized Markup Language

描述

Standard Generalized Markup Language (SGML) is the ISO standard 8879:1986 that describes a syntax for marking up documents with tags that describe the purpose of the text rather than the appearance on the page. This form of markup facilitates document interchange between different platforms and applications. SGML allows the management of information as data objects rather than text on a page.

In an SGML document the main structural components are called `elements`. The organization and structure of a document and the meaning of elements are described in the Document Type Definition (DTD). Elements are the *tags* that identify the content. Element names may be descriptive of the content for ease of use. For example `<para>` for paragraphs. Elements can have `attributes` which are used to modify or refine the properties or characteristics of the element. Within the DTD a valid context for each element is defined and a framework is provided for the types of elements that constitute a compliant document.

Another component of the DTD is `entities`. Entities are a collection of characters that can be referenced as a unit. Entities are similar to constants in a programming language such as C. They can be defined and referenced. An entity can represent one character or symbol which does not appear on a standard keyboard, a word or group of words, or an entire separate sgml marked-up file. Entities allow reuse of standard text.

There is no single standard DTD , but the de facto standard for the computer industry is the DocBook DTD , developed and maintained by the Davenport Group. Within Sun, the SolBook DTD , which is a proper subset of DocBook DTD , is used when writing reference manual pages. The SolBook DTD contains a number of tags that are designed for the unique needs of the reference pages.

SolBook Elements

Elements are defined with a hierarchical structure that gives a structure to the document. The following is a description of some of the elements from the SolBook DTD which are used for reference pages.

DOCTYPE

The first line in an SGML file that identifies the location of the DTD that is used to define the document. The `<!DOCTYPE` string is what the SGML -aware `man(1)` command uses to identify that a file is formatted in SGML rather than `nroff(1)`.

RefEntry

The top layer element that contains a reference page is `<refentry>`. All of the text and other tags must be contained within this tag.

RefMeta

The next tag in a reference page is `<refmeta>`, which is a container for several other tags. They are:

<refentrytitle> This is the title of the reference page. It is equivalent to the name of the reference page's file name, without the section number extension.

<manvolnum> This is the section number that the reference page resides in. The contents may be a text entity reference.

<refmiscinfo> There are one or more `<refmiscinfo>` tags which contain *meta* information. Meta information is information about the reference page. The `<refmiscinfo>` tag has the `class` attribute. There are four classes that are routinely used.

date — This is the date that the file was last modified. By consensus this date is changed only when the technical information on the page changes and not simply for an editorial change.

sectdesc — This is the section title of the reference page; for example `User Commands`. The value of this attribute may be a text entity reference.

software — This is the name of the software product that the topic discussed on the reference page belongs to. For example UNIX commands are part of the SunOS x.x release. The value of this attribute may be a text entity reference.

arch — This is the architectural platform limitation of the subject discussed on the reference page. If there are no limitations the value used is `generic`. Other values are `sparc` and `x86`.

copyright — This attribute contains the Sun Microsystems copyright. Any other copyrights that may pertain to the individual reference page file should be entered as separate `<refmiscinfo>` entries. The value of this attribute may be a text entity reference.

RefNameDiv — This tag contains the equivalent information to the `.TH` macro line in an `nroff(1)` reference page. `<refnamediv>` contains three tags. These tags contain the text that is before and after the '–' (dash) on the `NAME` line.

<refname> These are the names of the topics that are discussed in the file. There may be more than one `<refname>` for a page. The first `<refname>` must match the name of the file and the `<refentrytitle>`. If there are more than one `<refname>` tags, each is separated by a ',' (comma). The comma is generated by the publisher of sgml files, so it should not be typed. This is referred to as *auto-generated* text.

<refpurpose> The text after the dash on the `NAME` line is contained in this tag. This is a short summary of what the object or objects described on the reference page do or are used for. The dash is also auto-generated and should not be typed in.

<refdiscriptor> In some cases the `<refentrytitle>` is a general topic descriptor of a group of related objects that are discussed on the same page. In this case the first tag after the `<refnamediv>` is a `<refdiscriptor>`. The `<refname>` tags follow. Only one `<refdiscriptor>` is allowed, and it

should match the <refentrytitle>.

RefSynopsisDiv

The SYNOPSIS line of the reference page is contained by this tag. There is a <title> that usually contains an entity reference. The text is the word SYNOPSIS. There are several tags within <refsynopsisdiv> that are designed specifically for the type of synopsis that is used in the different reference page sections. The three types are:

<cmdsynopsis>	Used for commands and utilities pages.
<funcsynopsis>	Used for programming interface pages.
<synopsis>	Used for pages that do not fall into the other two categories.

RefSect1

This tag is equivalent to the .SH nroff macro. It contains a <title> element that is the title of the reference page section. Section names are the standard names such as DESCRIPTION, OPTIONS, PARAMETERS, SEE ALSO, and others. The contents of the <title> may be a text entity reference.

RefSect2

This tag is equivalent to the .SS nroff macro. It contains a <title> element that contains the text of the sub-section heading. <refsect2> tags may also be used within a <refsynopsisdiv> as a sub-section heading for the SYNOPSIS section.

Block Elements

There are a number of block elements that are used for grouping text. This is a list of some of these elements.

<para>

This tag is used to contain a paragraph of text.

<variablelist>

This tag is used to create two column lists. For example descriptions for command options, where the first column lists the option and the second column describes the option.

<orderedlist>

An list of items in a specific order.

<itemizedlist>

A list of items that are marked with a character such as a bullet or a dash.

<literallayout>

Formatted program output as produced by a program or command. This tag is a container for lines set off from the main text in which line breaks, tabs, and leading white space are significant.

<programlisting>

A segment of program code. Line breaks and leading white space are significant.

<table>

This tag contains the layout and content for tabular formatting of information. <table> has a required <title>.

<informaltable>

This tag is the same as the <table> tag except the <title> is not required.

<example> This tag contains examples of source code or usage of commands. It contains a required `<title>`.

<informalexample> This tag is the same as the `<example>` tag except the `<title>` is not required.

Inline Elements

The inline elements are used for tagging text.

<command> An executable program or the entry a user makes to execute a command.

<function> A subroutine in a program or external library.

<literal> Contains any literal string.

<parameter> An argument passed to a computer program by a function or routine.

<inlineequation> An untitled mathematical equation occurring in-line.

<link> A hypertext link to text within a book, in the case of the reference manual it is used to cross reference to another reference page.

<olink> A hypertext link used to create cross references to books other than the reference manual.

<xref> A cross reference to another part of the same reference page.

另请参见

`man(1)`, `nroff(1)`, `man(5)`

引用名

smf – 服务管理工具

描述

Solaris 服务管理工具定义了一种编程模型，用以提供持久运行的应用程序，这些应用程序称为**服务**。此工具还提供运行服务的基础结构。服务可以表示运行的应用程序、设备软件状态或一组其他服务。服务在框架中由**服务实例**对象表示，这些对象是服务对象的子项。实例对象可以继承或覆盖父服务对象的配置，这使多个服务实例可以共享配置信息。所有服务和实例对象都包含在一个**范围**内，该范围表示配置信息的集合。本地 Solaris 实例的配置称为“本地主机”范围，这是当前唯一支持的范围。

每个服务实例都使用故障管理资源标识符 (fault management resource identifier, FMRI) 来命名（模式为 `svc:`）。例如，在系统启动时启动的 syslogd(1M) 守护进程是具有以下名称的缺省服务实例：

```
svc://localhost/system/system-log:default
svc:/system/system-log:default
system/system-log:default
```

许多命令也允许使用 FMRI 缩写。有关此类示例，请参见 svcs(1) 手册页。

在上面的示例中,`default` 是实例名称，`system/system-log` 是服务名称。服务名称可包含多个用斜杠 (/) 分隔的组成部分。所有组成部分（最后一个组成部分除外）构成了服务的**类别**。站点特定的服务应当用以 `site` 开头的类别来命名。

服务实例可以启用，也可以禁用。所有服务都可以通过 svcadm(1M) 命令来启用或禁用。

可通过 svcs(1) 命令来显示系统上管理服务实例的列表。

当管理员删除由标准位置中的清单或配置文件支持的条目时，该条目会被屏蔽，且通过向 SMF 发送普通查询无法看到该条目。使用 `svccfg listcust` 可以浏览已屏蔽的条目，将 `delcust` 子命令与 `svccfg` 结合使用可将其删除。有关详细信息，请参见 svccfg(1M)。

依赖性

服务实例可以与一组**实体**具有依赖关系，这些实体可包含服务、实例和文件。依赖性控制着启动和自动停止服务的时间。当未满足已启用服务的依赖性时，服务就保持脱机状态。当满足其依赖性时，就会启动服务。如果启动成功，服务将转换为联机状态。不同于服务和实例，不会在创建或删除文件时动态地评估文件依赖性。只对它们评估一次。

依赖性是否满足将由其**分组**确定：

`require_all` 当所有引用的服务正在运行（联机或降级）或者所有指示的文件都存在时满足。

`require_any` 当引用的任一服务正在运行（联机或降级）或者至少有一个指示的文件存在时满足。

optional_all　当引用的服务正在运行（联机或降级）或因没有管理操作而未运行（禁用、维护、不存在或脱机等待必须有管理操作才能启动的依赖项）时满足。不完整的服务也满足可选的依赖性。

exclude_all　当所有引用的服务都已禁用、处于维护状态或者引用的服务或文件不存在时满足。

在运行（联机或降级）期间，如果由 require_all 、require_any 或 optional_all 依赖性引用的服务停止或刷新，SMF（Service Management Facility，服务管理工具）将考虑服务停止的原因以及依赖性的 restart_on 属性以确定是否停止该服务。

```
                  |  restart_on value
event             |  none  error restart refresh
------------------+------------------------------
stop due to error |  no    yes   yes     yes
non-error stop    |  no    no    yes     yes
refresh           |  no    no    no      yes
```

如果服务出现硬件错误或软件错误（如核心转储），则认为该服务已经因出错而停止。对于 exclude_all 依赖性，如果引用的服务已启动，但 restart_on 属性不是 none，也将停止该服务。

服务的依赖性可通过 svcs(1) 或 svccfg(1M) 列出，也可通过svccfg(1M) 进行修改。

重启程序

每个服务都由一个重启程序来管理。主重启程序 svc.startd(1M) 管理整个服务实例集及其依赖项的状态。主重启程序代表其服务操作，其操作对象是委托重启程序，委托重启程序可以为特定的应用程序类提供特定的执行环境。例如，inetd(1M) 是一个委托重启程序，它为其服务实例提供一个由网络连接构成的初始环境作为输入和输出文件描述符。委托给 inetd(1M) 的每个实例都处于联机状态。虽然特定实例的守护进程可能未运行，但该实例已经可运行。

由于当实例转为联机状态时将满足依赖性，因此 svc.startd(1M) 将调用其他实例的启动方法或指示委托重启程序执行此类操作。这些操作可能会重叠。

可使用 svcs(1) 检查当前的服务集及关联的重启程序。所有重启程序使用的公共配置的说明在 smf_restarter(5) 中指定。

方法

每个服务或服务实例都必须定义一组启动、停止和（可选）刷新该服务的方法。有关 svc.startd(1M) 以及类似 fork(2)-exec(2) 重启程序的方法约定的更完整说明，请参见 smf_method(5)。

管理方法（如将传统配置信息捕获到系统信息库的方法）将在 svccfg(1M) 手册页中讨论。

服务的方法可使用 svccfg(1M) 命令列出和修改。

状态

每个服务实例始终处于某一明确定义的状态，具体取决于其依赖性、其方法的执行结果以及其潜在合同事件。系统定义了以下状态：

UNINITIALIZED
: 这是所有服务实例的初始状态。实例将根据 svc.startd(1M) 或相应重启程序的计算结果转为维护、脱机或禁用状态。

OFFLINE
: 实例已启用，但尚未运行或无法运行。如果重启程序成功执行服务启动方法或等效方法，实例将转为联机状态。如果执行失败，则会导致实例处于降级或维护状态。管理操作会导致实例处于初始化状态。

ONLINE
: 实例已启用且正在运行或可以运行。联机状态的具体特性是应用程序/模型特定的，由负责服务实例的重启程序进行定义。当满足所有依赖性时，联机是正确配置的服务的预期运行状态。实例失败会导致其处于降级或维护状态。如果实例所依赖的服务失败，则会导致其处于脱机或降级状态。

DEGRADED
: 实例已启用且正在运行或可以运行。不过，相对于正常运行而言，实例是以有限的功能运行的。实例失败会导致其处于维护状态。如果实例所依赖的服务失败，则会导致其处于脱机或降级状态。功能恢复后，应会将实例转为联机状态。

MAINTENANCE
: 实例已启用，但无法运行。需要管理操作（通过 svcadm clear 执行）才能使实例脱离维护状态。如果管理操作正在进行中，维护状态可能是临时所处的状态。

DISABLED
: 实例已禁用。使服务转换为脱机状态，最终只有满足所有依赖性时才会恢复联机状态。

LEGACY-RUN
: 此状态表示不通过服务管理工具进行管理的传统实例。处于此状态的实例已在某一时间点启动，可能正在运行，也可能未在运行。只能使用该工具观察实例，不能将实例转换为其他状态。

状态也会发生转换，但会导致返回到最初的状态。

事件通知

SMF 允许使用 SNMP（Simple Network Management Protocol，简单网络管理协议）或 SMTP（Simple Mail Transfer Protocol，简单邮件传输协议）通知状态转换。该工具会发布状态转换的信息事件，以供通知守护进程（如 snmp-notify(1M) 和 sasinfo(1M)）使用。已禁用服务的 SMF 状态转换不会生成通知，除非转换的最终状态为禁用且该转换存在通知参数。对于初始和最终状态相同的转换，也不会为其生成通知。

通知参数

FMA（Fault Management Architecture，故障管理体系结构）事件的通知参数存储在 svc:/system/fm/notify-params:default 中，但 SMF 状态转换所生成的信息事件除外。这些事件存储在服务中或存储在转换服务的实例中。在 svc:/system/svc/global:default 中，可在系统范围设置 SMF 状态转换所生成事件的通知参数。当在转换实例中找不到组合查找（如 scf_instance_get_pg_composed(3SCF) 中所述）时，将使用系统范围通知参数。可使

用 svccfg(1M) 处理通知参数。可使用 DTD（document type definition，文档类型定义）中描述的 notification_parameters 元素在服务清单或配置文件中配置通知参数。下面列举了一个示例：

```
<notification_parameters>
     <event  value='from-online' />
     <type name='smtp' active="false">
        <parameter name='to'>
            <value_node value='root@local' />
            <value_node value='admin-alias@eng' />
        </parameter>
     </type>
     <type name='snmp' />
</notification_parameters>
```

events 是一个用逗号分隔的 SMF 状态转换集列表或一个用逗号分隔的 FMA 事件类列表。*events* 不能混合包含 SMF 状态转换集和 FMA 事件类。

为了方便起见，标记 problem- {diagnosed,updated,repaired,resolved} 描述了 FMA 子系统诊断出的问题的生命周期：包括初始诊断、中期更新和最终问题解决。这些标记是基础 FMA 协议事件类 (全部位于 list.* 分层结构中) 的别名，但在配置通知首选项时不能使用后者。

problem-diagnosed
FMA 子系统诊断出了新问题。诊断包含由一个或多个可疑项组成的列表，系统可能（根据需要）已将这些可疑项自动隔离以防止进一步发生错误。问题在事件有效载荷中问题由 UUID（Universally Unique Identifier，通用唯一标识符）进行标识，描述此问题解决生命周期的其他事件会引用匹配的 UUID。

problem-updated
问题诊断中的一个或多个可疑资源已修复、替换或被视为没有故障（或再次出现了故障），但列表中至少保留了一个故障资源。修复可能是 fmadm 命令行（fmadm repaired、fmadm acquit、fmadm replaced）的结果，也可能是自动检测到的（如通过检测部件序列号更改）。

problem-repaired
问题诊断中的所有可疑资源均已修复、解决或被视为无故障。在此阶段，部分或全部资源可能仍处于隔离状态。

problem-resolved
问题诊断中的所有可疑资源均已修复、解决或被视为无故障**并且**不再处于隔离状态（例如，曾经是可疑项并处于脱机状态的 CPU 现在又重新恢复联机状态；这种取消隔离操作通常自动执行）。

状态转换集的定义如下：

to-<state>　最终转换状态为 <state> 的所有转换的集合。

from-<state>　初始转换状态为 <state> 的所有转换的集合。

<状态> 初始转换状态为 <state> 的所有转换的集合。

`all` 所有转换的集合。

状态的有效值包括：维护、脱机、禁用、联机和降级。转换集定义的示例如下：`maintenance`、`from-online`、`to-degraded`。

属性和属性组

上述依赖性、方法、委托重启程序和实例状态以服务或服务实例的属性或属性组形式表示。服务或服务实例具有任意数量的属性组，这些属性组用于存储应用程序数据。以这种方式使用特性组时，可以使应用程序的配置派生可供系统信息库提供给该工具中的所有数据的属性。应用程序也可以在框架中使用适当的 `service_bundle(4)` DTD 子集表示其配置数据。

属性查找是组合查找。如果在服务实例中未找到属性组/属性组合，大部分命令以及 `libscf(3LIB)` 的高级接口将在包含该实例的服务中搜索相同的属性组/属性组合。这样可以在服务实例之间共享公共配置。组合可视为服务实例与其父服务之间的继承关系。

属性将受到保护，以防被未经授权的进程修改。请参见 `smf_security(5)`。

通用属性组

通用属性组适用于所有服务实例。该属性组包含以下属性：

enabled (boolean) 指定是否启用实例。如果实例上不存在此属性，SMF 将无法告知实例的重启程序有关实例的存在情况。

restarter (fmri) 此服务的重启程序。有关更多信息，请参见"重启程序"一节。如果未设置此属性，将使用缺省的系统重启程序。

complete (astring) 该服务是完整的，还是不应启动的部分定义。此属性会在清单导入时自动设置。此外，没有此属性但顺利针对模板定义进行验证（请参见 `scf_tmpl_validate_fmri(3SCF)`）的实例会由 `svcadm(1M)` 在启用时创建此属性。

层

系统信息库由标准位置中文件的管理定制、当前状态和缺省值的组合集合而成。由 SMF 管理的文件系统位置中的清单定义的服务、实例、属性组和属性始终在系统信息库中准确表示。运行时由管理员或其他程序执行的定制被捕获并存储在系统信息库中。

属性在系统信息库中可以具有不同的值，用于反映清单、配置文件和管理定制的不同设置。缺省情况下向用户和服务提供哪一个值由名为 `layers` 的简单优先级机制仲裁。

SMF 跟踪四个层。按优先级降序，它们是：

admin 由交互使用 SMF 命令或库进行的任何更改。此层具有最高优先级。

site-profile（站点配置文件）

/etc/svc/profile/site 目录中的文件，或传统 /etc/svc/profile/site.xml 和 /var/svc/profile/site.xml 文件中的任何值。

system-profile（系统配置文件）

系统配置文件位置 /etc/svc/profile/generic.xml 和 /etc/svc/profile/platform.xml 中的任何值。

manifest（清单）

系统清单位置 /lib/svc/manifest 和 /var/svc/manifest 中的任何值。

任何单独的层中不能有属性冲突。admin 层中的冲突属性只是覆盖之前的属性。如果任何其他层中的多个文件提供该同一属性，并且该属性未在更高的层中设置，则整个实例标记为冲突，并且不会由 svc.startd(1M) 启动，直到冲突定义被去除，或在更高的层设置该属性。请求单个值（包括 svccfg 和 svcprop）的其他 libscf 使用者看到所有适当的值中的随机属性设置。我们不保证返回哪个冲突值。

快照

有关系统信息库中每个实例的历史数据由服务管理工具进行维护。此数据以只读快照形式提供，用于管理检查和回滚。可能会提供以下快照类型集：

`initial` 由管理员创建的或安装软件包期间生成的实例的初始配置。

`previous` 执行管理撤消操作时捕获的当前配置。

`running` 正在运行的实例配置。

`start` 成功转换为联机状态期间所捕获的配置。

可使用 svccfg(1M) 命令与快照进行交互。

专用属性组

某些属性组将标记为 non-persistent。这些组不会在快照中进行备份，其内容在系统引导期间会被清除。此类组通常存放活动程序状态，此状态无需在系统重新启动后存在。

配置系统信息库

每个服务实例的当前状态以及与服务和服务实例关联的属性都存储在由 svc.configd(1M) 管理的系统信息库中。

服务管理工具数据的系统信息库由 svc.configd(1M) 管理。

服务束、清单和配置文件

与服务或服务实例关联的信息存储在配置系统信息库中，可作为 XML 文件导出。此类 XML 文件称作服务束，是可移植的，很适合用于备份。服务束分为以下几种类型：

清单 包含与特定服务或服务实例集关联的完整属性集的文件。

配置文件 包含每个实例的 enabled 属性（在通用属性组中为**布尔型**）的一组服务实例和值的文件。

配置文件也可以包含服务和实例中属性的配置值。模板元素不能在配置文件中定义。

配置文件可以使用来自 DTD 的一组宽松元素，如 service_bundle(4) 中所述。要使用这些元素，应向 DOCTYPE 条目添加以下定义：

```
<!ENTITY % profile "INCLUDE">
<!ENTITY % manifest "IGNORE">
```

可使用 svccfg(1M) 命令在系统信息库中导入或导出服务束。有关服务束文件格式的说明以及有关制作服务束的准则，请参见 service_bundle(4)。

里程碑

smf 里程碑是一种服务，它聚合了多个服务依赖性。通常，里程碑本身并不执行任何有用的操作，但它声明一种其他服务可以依赖的特定系统就绪状态。一个示例就是**名称/服务**里程碑，它仅依赖当前启用的名称服务。

传统启动脚本

/etc/rc?.d 目录中的启动程序作为相应运行级别的里程碑的一部分执行：

/etc/rcS.d	milestone/single-user:default
/etc/rc2.d	milestone/multi-user:default
/etc/rc3.d	milestone/multi-user-server:default

每个程序的执行都表示为程序路径所指定的功能简化的服务实例。这些实例保持特殊的 legacy-run 状态。

这些实例没有已启用属性（在通用属性组中为 boolean 类型），通常无法使用 svcadm(1M) 命令进行处理。系统不会对这些程序执行错误诊断或重新启动。

另请参见

svcs(1)、inetd(1M)、snmp-notify(1M)、sasinfo(1M)、svcadm(1M)、svccfg(1M)、svc.conf

引用名

smf_bootstrap – 服务管理工具的引导、打包和兼容性行为

描述

服务管理工具在提供服务清单、合并服务清单更改、描述服务配置稳定性、使用服务配置覆盖以及使用服务配置文件等方面建立了相关约定。

引导时的清单装入

在系统引导期间，以及管理员或程序运行时，处理标准目录树 /lib/svc/manifest 和 /var/svc/manifest 的清单：

```
$ svcadm restart manifest-import
```

处理之前未导入，或自上一次导入以来发生更改的清单。使用散列来确定清单是否已更改。

当标准位置中的清单第一次导入时，它的属性、实例和服务作为 manifest 层的一部分添加到系统信息库。

当标准位置中的清单更新时，将自动导入它们。添加新的服务和实例，升级属性（如果它们发生变化），并在服务、实例和属性被移除时将其删除。

在引导期间，将在两个不同的阶段处理清单。

服务 svc:/system/early-manifest-import:default 是一项伪服务，负责首次清单处理。此服务仅在 svc.startd(1M) 初始化任何服务之前处理 /lib/svc/manifest 目录树中的清单，从而使 /lib/svc/manifest 中提供的服务始终以最新定义启动。由于此服务是伪服务，因此将忽略 svcadm(1M) 命令，但可以使用 svcs(1) 观察状态及获取日志文件信息。

svc:/system/manifest-import:default 服务负责第二次清单处理并按相应顺序依次导入 /lib/svc/manifest 和 /var/svc/manifest 目录树中的清单文件。

对 /var/svc/manifest 的支持是对引入 system/early-manifest-import:default 之前该目录树中所提供清单的兼容性支持。当服务可能以旧系统信息库配置启动时，由于其更新的清单尚未导入，因此 /var/svc/manifest 中所提供的服务可以会出现与升级相关的问题。同样，由于其清单文件尚未得到处理，因此新添加的服务也可能不可用或者已删除的服务仍在引导期间启动。为避免这些问题，强烈建议开发者将清单移至 /lib/svc/manifest。

请仅考虑将 *.xml 文件用于本节中介绍的操作。

配置文件应用

early-manifest-import 和 manifest-import 服务还应用配置文件。

/etc/svc/profile/generic.xml 和 /etc/svc/profile/platform.xml 中的系统提供的配置文件导入到 system-profile 层。

/etc/svc/profile/site 目录中的站点特定的配置文件和传统站点文件 /etc/svc/profile/site.xml 和 /var/svc/profile/site.xml 导入到 site-profile 层。

管理员可以通过运行以下命令请求重新应用这些配置文件：

```
$ svcadm restart manifest-import
```

配置文件定义的属性、实例和服务的行为与由清单定义的相同。

请仅考虑将 *.xml 文件用于本节中介绍的操作。

打包操作期间的清单处理

软件包中的所有服务清单都应由类 manifest 来标识。安装和移除服务清单的类操作脚本包含在软件包子系统中。调用 pkg install 时，会导入服务清单。

调用 pkg uninstall 时，会删除清单中禁用的实例。清单中联机或降级的实例将先被禁用，然后再被删除。清单中没有剩余实例的所有服务也将被删除。

稳定性声明

清单中所提供的每个服务组和每个属性组都应根据 attributes(5) 定义声明稳定性级别。通过了解稳定性级别，应用程序开发者可以根据服务或对象的组件是否存在或者它们是否可能跨发行版边界仍能正常使用，确定功能开发的可能性。

在 smf(5) 上下文中，稳定性值还可以跨服务的发行版边界（可包含该服务的修补程序）标识属性组中属性的预期更改范围。以下两部分对这一点进行了更详细地讨论。

属性组删除

service_bundle(4) 文档类型定义包含一个删除属性，该属性适用于服务清单中的每个属性组。如果该删除属性设置为 true，它会指示 svccfg(1M) 及其他清单导入工具从系统信息库中删除此属性组。如果该删除属性缺失，或者存在但设置为 false，将保留系统信息库中的属性组。

不会删除声明为稳定或发展中的属性组。可跨任何发行版边界删除声明为不稳定的属性组。

另请参见

svcs(1)、svcadm(1M)、svccfg(1M)、svc.startd(1M)、libscf(3LIB)、service_bundle(4)、at

pkg(1)

附注

smf(5) 的现有版本不支持多个系统信息库。

引用名

smf_method – 方法的服务管理框架约定

描述

在服务管理框架smf(5) 中通过 svc.startd(1M) 管理的服务类由符合简单 fork(2)-exec(2) 模型的应用程序组成。svc.startd(1M) 主守护进程及其他重启程序支持 fork(2)-exec(2) 模型（可能带有附加功能）。svc.startd(1M) 守护进程及其他重启程序需要按照本手册页中所述的约定激活、处理或检查服务实例的方法。

调用格式

方法调用的格式不受约定的控制。在某些情况下，方法调用可能由守护进程或提供服务的其他二进制可执行文件的直接调用组成。对于使用可执行脚本或其他间接可执行文件的情况，约定建议采用以下格式：

```
/path/to/method_executable abbr_method_name
```

该建议格式中使用的 *abbr_method_name* 是支持的方法，如 start 或 stop。在相关重启程序页中提供了重启程序支持的方法集。svc.startd(1M) 守护进程支持 start、stop 和 refresh 方法。

除了本页中引用的方法之外，重启程序还可能会定义其他种类的方法。围绕此类扩展的约定将由重启程序定义，可能与此处给出的定义不同。

环境变量

重启程序向方法提供了四个环境变量，这些变量用于确定调用方法的上下文。

SMF_FMRI
　为其调用方法的实例的服务故障管理资源标识符 (Fault Management Resource Identifier, FMRI)。

SMF_METHOD
　所调用方法的完整名称，如 start 或 stop。

SMF_RESTARTER
　调用方法的重启程序的服务 FMRI

SMF_ZONENAME
　正在其中运行方法的区域的名称。也可使用 zonename(1) 命令获取此名称。

在方法调用任何持久性进程之前，应从环境中删除这些变量。在下述包含文件中，为使用 Bourne 兼容 shell 脚本编写服务方法的服务作者提供了便利 shell 函数 smf_clear_env。

方法上下文可能会导致设置其他环境变量（如下文所述）。

方法定义

方法至少由 method 类型的属性组中的三个属性定义。

这些属性包括：

exec (*astring*)	方法可执行字符串。
timeout_seconds (*count*)	方法超时之前所经过的秒数。有关更多详细信息，请参见**超时**部分。
type (*astring*)	方法类型。当前始终设置为 method。

要进一步完善方法的执行环境，可定义方法上下文。有关更多信息，请参见**方法上下文**部分。

方法标记

当重启程序 svc.startd 在方法的 exec 字符串中定义一组标记时，将使用适当的值对这些标记进行解析和扩展。其他重启程序可能不支持方法标记。inet 服务 inetd(1M) 的委托重启程序不支持以下方法扩展。

%%
: %

%r
: 重启程序的名称，如 svc.startd

%m
: 所调用方法的完整名称，如 start 或 stop。

%s
: 服务的名称

%i
: 实例的名称

%f
: 实例的 FMRI

%{prop[:,]}
: 属性的值。prop 可能是以 / 分隔的属性 FMRI、属性组名称和属性名，也可能是 application 属性组中的属性名。这些值可以后跟 ,（逗号）或 :（冒号）。使用分隔符（如果提供）分隔多个值。如果未提供分隔符，将使用空格。出现在字符串值中的以下 shell 元字符将用 \（反斜杠）引起来：

```
; & ( ) | ^ < > newline space tab  \  " '
```

无效扩展会导致方法失败。

有两个显式标记可用来代替方法命令。

:kill [-signal]
: 将指定的信号（缺省情况下为 SIGTERM）发送到主实例合同中的所有进程。始终返回 SMF_EXIT_OK。此标记应当用于替换常见的 pkill 调用。

:是
: 始终返回 SMF_EXIT_OK。此标记应当用于对重启程序是必需的但对特定服务是不必要的方法。

正在退出和退出状态

start 方法的必需行为是延迟退出，直到服务实例已经准备好应答请求或以其他方式工作。

以下退出状态代码在 <libscf.h> 和 shell 支持文件中定义。

`SMF_EXIT_OK`	`0`	方法已退出，已成功执行其操作。
`SMF_EXIT_ERR_FATAL`	`95`	方法出现致命性失败，必须通过管理干预才能恢复。
`SMF_EXIT_ERR_CONFIG`	`96`	无法恢复的配置错误。返回此退出状态的常见情况是已启用的服务实例缺少必需的配置文件。
`SMF_EXIT_ERR_NOSMF`	`99`	方法被错误地在 smf(5) 工具外部调用。依赖于 smf(5) 功能的服务应退出并显示此状态值。
`SMF_EXIT_ERR_PERM`	`100`	方法所需的某种形式的权限（如文件访问、特权、授权或其他凭证）在调用时不可用。
`SMF_EXIT_ERR_OTHER`	非零	方法的任何非零退出状态均被视为未知错误。一系列未知错误可能会被重启程序或以重启程序的名义识别为故障。

除了上述退出代码以外，方法还可结合使用以下退出代码与 `smf_method_exit()`，可通过 smf_method_exit(3SCF) 和 `smf_include.sh` 获得以下代码：

`SMF_EXIT_TEMP_DISABLE`	`101`	方法顺利退出并请求临时禁用。
`SMF_EXIT_TEMP_TRANSIENT`	`105`	方法顺利退出并请求按照服务模型为“瞬态”一样来进行处理。

利用精确的退出代码，使负责的重启程序可以将错误响应分类为间歇性的（值得执行重新启动）或永久性的（要求管理干预）。

超时

每个方法都可以有独立的超时（以秒为单位指定）。方法超时由 `timeout_seconds` 属性指定。

超时是服务的重启程序用来确定方法已挂起或未在继续执行的最后手段。如果已超时，多数重启程序会将服务置于维护状态。请参见 svc.startd(1M)。在指定超时时，建议考虑一定的误差范围，以避免出现方法正在执行但系统因为内存、CPU 或 I/O 负载等原因而暂时响应非常缓慢的情况时，过早地判断为执行失败。

对于预期需要用一两秒的方法，`60` 秒是一个较好的初始值。`300` 秒（5 分钟）适用于通常需要用 `30` 秒的方法。对于经常要用较长时间的方法，应相应地增大超时值。

如果希望快速判断为故障以便提示进行管理干预，则可以使用较短的超时。如果管理干预可能只是清除服务并重新启动方法，请考虑使用较长的超时。

如果 `timeout_seconds` 设置为 `0`，则服务无超时。不建议使用此设置，但它适用于绝对需要它的服务。`-1` 也可用于指定无超时，但已过时。

Shell 编程支持

定义上述退出状态值的一组环境变量通过文件 /lib/svc/share/smf_include.sh 中的便利 shell 函数提供。此文件是 Bourne shell 脚本，可通过源运算符包含在任何 Bourne 兼容 shell 中。

为帮助编写可充当 SMF 方法的脚本以及 /etc/init.d 脚本，提供了 smf_present() shell 函数。如果 smf(5) 工具不可用，smf_present() 会返回非零退出状态。

此类脚本的一个可能的结构如下所示：

```
if smf_present; then
      # Shell code to run application as managed service
      ....

      smf_clear_env
else
      # Shell code to run application as /etc/init.d script
      ....
fi
```

此示例显示了所提供的两个便利函数的用法。

方法上下文

服务管理工具提供了一种常见机制，可用于设置在其中执行 fork(2)-exec(2) 模型服务的上下文。

所需的方法上下文应由服务开发者提供。所有服务实例都应尽可能地以最低权限级别运行，以限制潜在的安全危害。

方法上下文可包含以下属性：

use_profile
: 一个布尔值，指定是否应使用配置文件来代替 user、group、privileges 和 limit_privileges 属性。

环境
: 要以一种包含多个 NAME=value 字符串的形式插入方法所在环境的环境变量。

profile
: RBAC（Role-Based Access Control，基于角色的访问控制）配置文件的名称，该配置文件与方法可执行文件一起用于标识 exec_attr(4) 中的条目。

user
: 数字或文本形式的用户 ID。

group
: 数字或文本形式的组 ID。

supp_groups
: 一个可选字符串，按数字或文本形式的 ID 指定补充组成员资格。

privileges
一个可选字符串，指定 privileges(5) 中所定义的特权集。可在此处指定一个扩展策略。

limit_privileges
一个可选字符串，指定 privileges(5) 中所定义的限制特权集。

working_directory
从中启动方法的起始目录。可将 :home 用作一个标记，指示使用其 uid 启动方法的用户的起始目录。如果未设置该属性，将使用 :home。

corefile_pattern
一个可选字符串，按照 coreadm(1M) 指定要用于服务的核心文件模式。多数重启程序都提供了缺省值。设置此属性会覆盖全局核心模式的本地定制。

project
数字或文本形式的项目 ID。可将 :default 用作一个标记，为使用其 uid 启动方法的用户指示由 getdefaultproj(3PROJECT)标识的项目。

resource_pool
启动方法的资源池的名称。可将 :default 用作一个标记，指示在上述 project 属性中给出的 project(4) 条目中所指定的池。

通过为服务或实例指定 method_context 属性组，可为整个服务实例设置方法上下文。通过在方法属性组上提供方法上下文属性，方法可能会覆盖实例方法上下文。

无效方法上下文设置始终会导致方法失败，但导致发出警告的无效环境变量除外。

除了上述定义的上下文之外，许多 fork(2)-exec(2) 模型重启程序在作为方法调用可执行文件时还使用以下约定：

参数数组
argv[] 中参数的设置与 exec 字符串的结果 /bin/sh -c 一致。

文件描述符
文件描述符 0 为 /dev/null。文件描述符 1 和 2 建议为基于服务日志文件。

文件

/lib/svc/share/smf_include.sh
退出状态值的定义。

/usr/include/libscf.h
退出状态代码的定义。

示例

示例 1 在服务日志中报告服务特定的配置错误消息。

启动方法可能要使用 smf_method_exit()，以在服务日志中报告服务特定的配置错误消息。

```
if [ ! -s "$my_config_file" ]; then
   smf_method_exit $SMF_EXIT_ERR_CONFIG \
       missing_or_empty_config_file \
```

示例 1 在服务日志中报告服务特定的配置错误消息。 （续）

```
       "$my_config_file is missing or empty"
fi
```

示例 2 在非全局区域中启动时禁用仅应在全局区域中运行的服务。

仅应在全局区域中运行的服务在非全局区域中启动时，可能要禁用其本身。

```
if smf_is_nonglobalzone; then
    smf_method_exit $SMF_EXIT_TEMP_DISABLE global_zone_only \
    "$SMF_FMRI is not supported in a local zone" SUNW_OST_OSCMD
fi
```

另请参见

zonename(1)、coreadm(1M)、inetd(1M)、svccfg(1M)、svc.startd(1M)、exec(2)、fork(2)、g

附注

smf(5) 的现有版本不支持多个系统信息库。

当服务配置为以 root 身份但以不同于 limit_privileges 的权限启动时，所生成的进程可识别权限。这可能会出乎开发者的意料，他们本以为 seteuid(<非零 UID>) 能够将权限降为基本或更低的级别。

引用名

smf_restarter – 重启程序的服务管理工具约定

描述

服务管理工具中的所有服务实例都必须由重启程序来管理。本手册页介绍该框架中所有重启程序共有的配置、功能和报告特性。特定重启程序独有的特性将在重启程序的手册页中介绍。

对于每个管理服务，重启程序依赖在服务实例上检索属性来确定配置。重启程序管理一组属性组，以此向显示工具（如 svcs(1)）传送服务的当前处置情况。

服务配置

所有服务的常见重启程序配置均在 general 属性组中捕获。此组包含以下必需和可选的属性设置。

enabled（已启用）
: 此属性是必需的。如果设置此属性，实例的重启程序会尝试维护服务的可用性。

restarter
: 此属性是可选的，允许指定备用重启程序管理服务实例。如果 restarter 属性为空或缺失，则重启程序缺省为 svc.startd(1M)。

single_instance
: 此属性是可选的。如果设置此属性，在任意时间将只允许一个服务实例转换为联机或降级状态。

服务报告

所有重启程序均使用 restarter 属性组报告状态，该属性组包含以下属性：

next_state
: 存储在这些属性中的实例的当前状态和下一状态（如果当前处于转换中）。有关可能的状态的说明，请参见 smf(5)。

auxiliary_state
: 一个不带空格的 astring，包含描述完整重启程序特定的状态以及重启程序状态属性的准确术语。不能一直设置辅助状态，在从任何状态进行转换期间应始终将其清除。每个重启程序都必须定义它所使用的辅助状态的精确列表。

state_timestamp
: 访问当前状态的时间。

contract
: 执行服务实例所依据的主进程合同 ID（如果有）。

另请参见

svcs(1)、svc.startd(1M)、service_bundle(4)、smf(5)、smf_method(5)

引用名

smf_security – 服务管理工具的安全行为

描述

服务管理工具 smf(5) 的配置子系统需要修改服务配置的权限。通过 user_attr(4) 和 prof_attr(4) 将下文所述的授权与用户相关联，可以向用户授予适当的权限。请参见 rbac(5)。

以下授权用于处理服务和服务实例。

solaris.smf.modify　有权添加、删除或修改服务、服务实例或其属性以及读取受保护的属性值。

属性组授权

smf(5) 配置子系统将属性与每个服务和服务实例相关联。相关属性将进行分组。组可以表示执行方法、凭证信息、应用程序数据或重启程序状态。创建或修改属性组的能力可能会导致 smf(5) 组件执行可能需要系统权限的操作。相应地，框架也需要处理属性组的适当权限。

每个属性组都有与其用途相对应的类型。核心属性组类型有 method、dependency、application 和 framework。也可以引入其他属性组类型，但前提是它们必须符合 smf(5) 中的扩展命名约定。不过，以下基本授权仅适用于核心属性组类型：

solaris.smf.modify.method
　有权更改值或者创建、删除或修改 method 类型的属性组。

solaris.smf.modify.dependency
　有权更改值或者创建、删除或修改 dependency 类型的属性组。

solaris.smf.modify.application
　有权更改值、读取受保护的值以及创建、删除或修改 application 类型的属性组。

solaris.smf.modify.framework
　有权更改值或者创建、删除或修改 framework 类型的属性组。

solaris.smf.modify
　有权添加、删除或修改服务、服务实例或其属性以及读取受保护的属性值。

属性组特定的授权可以由包含在该属性组中的属性来指定。

modify_authorization
　授权允许在属性组中添加、删除或修改属性以及从受保护的属性组检索属性值。

value_authorization
　授权允许更改属性组的任何属性值（modify_authorization 除外）以及从受保护的属性组检索任何属性值（modify_authorization 除外）。

read_authorization
　授权允许在属性组中检索属性值。如果存在具有此名称的字符串值属性，则会将所在属性组标识为受保护。此属性不适用于非 application 类型的属性组。请参见“受保护的属性组”。

上述授权属性仅当具有 astring 类型时才能使用。如果实例属性组没有其中任一属性，但实例的服务具有与该属性同名的属性组，则将使用该属性的值。

受保护的属性组

通常，任何用户无需显式授权即可读取系统信息库中的所有属性值。非 framework 类型的属性组可用于存储其值需要保护的属性。除非具有适当的授权，否则不能公开这些属性组。通过字符串值的 read_authorization 属性存在与否来指示属性组的受保护状态。如果此属性存在，只能按照**属性组授权**中的说明来检索属性组中的所有属性值。

如果管理域所使用的策略禁止备份被视为敏感的数据，应从其备份中排除 SMF 系统信息库数据库。即使存在此类策略，也可备份未受保护的属性值，方法是使用 svccfg(1M) 归档命令为不包含受保护属性值的系统信息库创建归档文件。

服务操作授权

对服务实例执行的某些操作可能会导致服务中断或停用。这些操作需要授权才能确保任何拒绝服务操作都是有意的管理操作。此类操作包含执行 refresh 或 restart 方法的请求或将服务实例置于维护或其他非运行状态的请求。以下授权允许请求此类操作：

solaris.smf.manage　　有权请求任何服务实例的重新启动、刷新或其他状态修改。

此外，general/action_authorization 属性还可以指定允许对该服务实例请求服务操作的其他授权。需要 solaris.smf.manage 授权才能修改此属性。

定义的权限配置文件

包含了两个权限配置文件，它们可以提供用于处理典型 smf(5) 操作的分组授权。

服务管理

服务管理员可以通过任意方式处理系统信息库中的任何服务。服务管理员与 solaris.smf.manage 和 solaris.smf.modify 授权相对应。

要使用 pkg(1) 命令添加或删除其服务清单文件中包含服务清单的软件包，至少需要服务管理配置文件。

服务操作员

服务操作员能够在系统上启用或禁用任何服务实例以及请求执行其 restart 或 refresh 方法。服务操作员与 solaris.smf.manage 和 solaris.smf.modify.framework 授权相对应。

站点可根据需要定义其他权限配置文件。

远程系统信息库修改

远程系统信息库服务器可能会由于附加权限检查而拒绝修改尝试。请参见“附注”部分。

示例

示例 1　允许用户以非 root 用户的身份修改 system/cron 服务。

通过将以下行添加到 /etc/user_attr 可允许用户 “johndoe” 以非 root 用户身份对 system/cron 服务执行重新启动、启用、禁用或其他状态修改操作。

```
johndoe::::auths=solaris.smf.manage.cron
```

示例 2 允许用户以非 root 用户身份修改任意服务上的任意属性以及修改 system/cron 服务。

通过将以下行添加到 /etc/user_attr 可允许用户 “janedoe” 以非 root 用户身份修改任意服务上的任意属性，以及对 system/cron 服务执行重新启动、启用、禁用或其他状态修改操作。

```
janedoe::::auths=solaris.smf.modify,solaris.smf.manage.cron
```

另请参见

auths(1)、profiles(1)、svccfg(1M)、prof_attr(4)、user_attr(4)、rbac(5)、smf(5)

pkg(1)

附注

smf(5) 的现有版本不支持远程系统信息库。

当服务配置为以 root 身份但以不同于 limit_privileges 的权限启动时，所生成的进程可识别权限。这可能会出乎开发者的意料，他们本以为 seteuid(<非零 UID>) 能够将权限降为基本或更低的级别。

引用名

smf_template – 对服务元数据的服务管理框架支持

描述

模板由服务开发者定义，用于在整体上描述关于服务的元数据或描述服务的各个配置属性，包括便于阅读的说明以及有效配置的定义。

通过 SMF 命令为管理员提供访问模板的权限，这些命令用于描述配置值以及根据模板验证配置。

工具开发者可以使用模板为服务配置提供更有帮助的用户界面。

模板数据

服务元数据在模板中定义为服务清单的一部分。

使用模板数据

可以使用 `svcs -lv` 和 `svccfg describe` 命令以便于阅读的格式访问关于属性的元数据。

可以使用 svccfg(1M) 的 `validate` 子命令根据模板数据验证服务实例或清单。可以使用一组 libscf(3LIB) 接口访问模板数据。

模板定义

服务清单是定义模板的唯一接口。

服务作者应该为服务特定的属性组以及它们所引入的属性提供模板元数据，包括 `common_names`、`descriptions`、`choices` 和 `constraints`。在 C 语言环境中，服务作者必须至少为属性组和属性提供说明。服务作者不能为框架附带的属性组（如方法和相关项）提供模板元数据。

有关为服务制作模板定义的示例，请参见“示例”部分。

模板编写

对属性组的相关模板数据的所有模板接口搜索依次在实例、服务、服务的重启程序上执行，最后将全局执行。

属性组模板由其作者定义，可以应用于特定实例、服务及其所有实例、重启程序的委托或者全局应用。典型服务作者在实例或服务上定义模板。在实例上定义的模板仅应用于该实例，可覆盖在服务上定义的该属性组的模板。在服务上定义的模板应用于该服务的所有实例。

重启程序作者可以在其清单中定义模板，这些模板应用于使用其重启程序的任何服务（也称为**委托**）。SMF 框架作者在 `svc:/system/svc/global` 的清单中为整个 SMF 框架熟知其含义的属性组定义了模板。

如果模板已在全局定义或由重启程序定义，但又由服务或实例重新定义，将被标记为验证错误。服务作者可以只为其服务特定的且 SMF 框架未使用的属性组创建模板，从而避免这些错误。

属性组模板也可以按名称或类型设置通配符。只有适用于属性组的最特定的模板定义才受支持。

模板详细信息

服务和实例模板

template 元素定义模板块的开始。以下所有更深入的定义可包含在模板块中。template 元素可包含在 service 或 instance 元素中。如果它包含在 service 元素中，则应用于服务及该服务的所有实例。如果它包含在 instance 元素中，则仅应用于服务的该实例。

我们建议您尽可能地为整个服务定义模板数据。

```
<service ... >
   <template>
   </template>
</service>
```

服务和实例通用名称

整个服务或实例可以定义一个通用名称，用来描述服务/实例的用途。

```
<template>
  <common_name>
      <loctext xml:lang='C'>console login</loctext>
  </common_name>
<template>
```

common_name 是一个自由格式的字符串，但专门用作 GUI 或 CLI 中的标签。

定义通用名称时，请遵循以下准则：

- 简短。通常一两个字词就很合适。将名称限定为 40 个字符以内。
- 清晰。服务名、属性组名或属性名对用户可能没什么帮助，但 *common_name* 有助于阐明实体的用途。
- 无标点符号。*common_name* 不是句子或段落。它不应包含分句或短语。只有在为了满足商标需求时才应使用标点符号。
- 大写字母只能用于首字母缩略词或正确的名称。对于英语以外的语言环境，对句子分段使用适当的大写。

服务和实例说明

description 元素包含属性组的较长说明，适用于状态行或工具提示：

```
<template>
  <description>
     <loctext xml:lang='C'>Provide the text login prompt on console.
     </loctext>
  </description>
<template>
```

description 准则

- 使用正确的语法。*description* 是旨在供用户阅读的句子。
- 简短。几个句子通常最合适。

文档

可明确地定义此服务的文档，这样在服务出现问题或者服务的使用者需要更多相关信息时，就可以轻松地查找文档。

属性组

pg_pattern 元素包含属性组的定义：

```
<template>
  <pg_pattern name="pgname" type="pgtype" target="this" required="true">
  </pg_pattern>
</template>
```

name 是属性组的名称，*type* 是属性组的类型。

target 指定此定义的目标。"this" 是指定义服务或实例。"instance" 只能在服务的模板块中使用，这意味着该定义应用于此服务的所有实例。"delegate" 只能在重启程序的模板块中使用，并且应用于委托给该重启程序的所有实例。"all" 只能由主重启程序使用，是指系统上的所有服务。目标的缺省值为 "this"。

required 指示此属性组是否为必需。*required* 的缺省值为 false。如果 *required* 为 true，则必须指定 *name* 和 *type*。

name 和/或 *type* 可忽略。如果这两个属性中的任何一个被忽略，则将其视为通配符。例如，如果在 pg_pattern 定义中忽略 name 属性，则 pg_pattern 将适用于具有指定类型的所有特性组。

属性组名称

common_name 元素包含属性组的便于阅读的本地化名称：

```
<pg_pattern ...>
  <common_name>
    <loctext xml:lang='C'>start method</loctext>
  </common_name>
</pg_pattern>
```

common_name 是一个自由格式的字符串，但专门用作 GUI 或 CLI 中的标签。

请参见上文“服务实例和通用名称”中的 *common_name* 准则。

属性组说明

description 元素包含属性组的较长说明，适用于状态行或工具提示：

```
<pg_pattern ...>
  <description>
    <loctext xml:lang='C'>A required method which starts the service.
    </loctext>
    </description>
</pg_pattern>
```

请参见上文“服务和实例说明”中用于指定 *description* 的准则。

属性

prop_pattern 元素包含特定属性的定义：

```
<pg_pattern ...>
  <prop_pattern name="propname" type="proptype" required="true">
  </prop_pattern>
</pg_pattern>
```

name 是属性的名称，*type* 是属性的类型。

`required` 指示属性是否为必需。`required` 的缺省值为 `false`。

name 始终为必需。*type* 只有在 `required` 为 `false` 时才可选。

属性名

common_name 元素包含属性的便于阅读的本地化名称：

common_name 是一个自由格式的字符串，但专门用作 GUI 或 CLI 中的标签。

```
<prop_pattern ...>
<common_name>
  <loctext xml:lang='C'>retry interval</loctext>
</common_name>
</prop_pattern>
```

请参见上文“服务实例和通用名称”中的 *common_name* 准则。

属性单位

units 元素包含数字属性的便于阅读的本地化单位：

```
<prop_pattern ...>
  <units>
    <loctext xml:lang='C'>seconds</loctext>
  </units>
</prop_pattern>
```

units 准则

- 简短。尽量只使用单个字词或标签。复数形式通常最合适。
- 无标点符号。*units* 不是句子或段落。它不应包含分句或短语。只有在为了满足商标需求时才应使用标点符号。

属性说明

description 元素包含属性的较长说明，适用于状态行或工具提示：

```
<prop_pattern ...>
   <description> <loctext xml:lang='C'>
    The number of seconds to wait before retry.
    </loctext> </description>
</prop_pattern>
```

请参见上文“服务和实例说明”中用于指定 *description* 的准则。

属性可见性

visibility 元素指定更高级别软件中的简化视图是否要显示此属性。

```
<prop_pattern ...>
  <visibility value="hidden | readonly | readwrite"/>
</prop_pattern>
```

有些属性是内部实现的详细信息，不应显示为配置设置。其他属性可能仅为只读。此属性用于指定这些限制。值 hidden 指示不应显示属性，`readonly` 指示属性不能修改，`readwrite` 指示属性可修改。

此属性不是一种安全机制，它只是专门帮助用户避免因粗心大意而造成破坏以及从 CLI 输出或 GUI 显示中删除不必要的杂乱信息。在许多命令和 UI 中的完全公开模式下，隐藏属性是可见的。

属性格式

cardinality 和 *internal_separators* 元素用于限制属性的结构：

```
<prop_pattern ...>
  <cardinality min="1" max="1"/>
  <internal_separators>,</internal_separators>
</prop_pattern>
```

cardinality 指示可接受的属性值数量。*min* 是最小数量，*max* 是最大数量。这两者均可选。如果这两者均未指定，则 `<cardinality/>` 等同于缺省数量的值、零个值或零个以上的值。

internal_separators 指定在打包了多个实际值的属性值中使用的分隔符。

值约束

constraints 元素指定属性可接受哪些值：

```
<prop_pattern ...>
<constraints>
       <value name="blue" />
       <range min="1" max="7"/>
       <include_values type="values"/>
</constraints>
</prop_pattern>
```

value 元素包含可能的属性值。range 包含整数范围。

value 和 *range* 可以按任何组合形式使用，限制其使用将会禁止许多有效的说明。如果未指定值约束，则属性可采用任何值。

include_values 包含由值块指定的所有值（请参见"值说明"部分）。

值选择

选择块指示 UI 应向用户提供哪些值：

```
<prop_pattern ...>
<choices>
      <range min="1" max="3"/>
      <value name="vt100" />
      <value name="xterm" />
      <include_values type="constraints"/>
      <include_values type="values"/>
</choices>
</prop_pattern>
```

与用于约束时一样，`range` 和 `value` 也分别包含范围和各个值。

include_values 包含由约束块或值块（请参见下一部分）指定的所有值。

值说明

与属性名一样，属性可采用的值也可能具有难以理解的表示形式。values 元素包含特定属性值的便于阅读的本地化说明：

```
<prop_pattern>
<values>
      <value name="blue">
              <common_name>
                      <loctext xml:lang='C'>blue</loctext>
              </common_name>
              <description>
                      <loctext xml:lang='C>
                              The color between green and indigo.
                      </loctext>
              </description>
      </value>
</values>
</prop_pattern>
```

common_name 是一个自由格式的字符串， 但专门用作 GUI 或 CLI 中的标签。

请参见上文"服务实例和通用名称"中的 *common_name* 准则。

示例

假定要定义基本模板数据的基本服务如下所示：

```
<?xml version="1.0"?
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type='manifest' name='FOOfoo:foo'>
<service name='system/foo' type='service' version='1'>
      <dependency>
              name='multi-user'
              type='service'
              grouping='require_all'
              restart_on='none'
              <service_fmri value='svc:/milestone/multi-user' />
      </dependency>
      <exec_method
              type='method'
              name='start'
              exec='/opt/foo/food'
              timeout_seconds='60'>
      </exec_method>
      <exec_method
              type='method'
              name='stop'
              exec=':kill'
              timeout_seconds='60'>
      </exec_method>
      <property_group name='config' type='application'>
              <propval name='local_only' type='boolean' value='false' />
              <propval name='config_file' type='astring'
                  value='/opt/foo/foo.conf' />
      <property name='modules' type='astring'>
         <astring_list>
               <value_node value='bar'/>
               <value_node value='baz'/>
         </astring_list>
        </property>
      </property_group>

      <instance name='default' enabled='false' />
</service>
</service_bundle>
```

该服务可定义一些基本模板数据，从而为在 `<service>` 标记内使用此服务的管理员提供帮助。最有用的就是记录服务本身的用途以及服务特定的配置。

```
<template>
       <common_name> <loctext xml:lang='C'>
           all-purpose demonstration
       </loctext> </common_name>
       <documentation>
```

```
         <manpage title=’food’ section=’1M’
                  manpath=’/opt/foo/man’ />
</documentation>

<pg_pattern name=’config’ type=’application’ target=’this’
    required=’true’>
        <description> <loctext xml:lang=’C’>
            Basic configuration for foo.
         </loctext> </description>
         <prop_pattern name=’local_only’ type=’boolean’
             required=’false’>
                <description> <loctext xml:lang=’C’>
                    Only listen to local connection requests.
                </loctext> </description>
         </prop_pattern>
         <prop_pattern name=’config_file’ type=’astring’
              required=’true’>
                 <cardinality min=’1’ max=’1’/>
                 <description> <loctext xml:lang=’C’>
                      Configuration file for foo.
                  </loctext> </description>
          </prop_pattern>
          <prop_pattern name=’modules’ type=’astring’
              required=’false’>
                  <description> <loctext xml:lang=’C’>
                      Plugin modules for foo.
                   </loctext> /description>
                   <values>
                            <value name=’bar’>
                            <description> <loctext xml:lang=’C’>
                                Allow foo to access the bar.
                             </loctext> </description>
                             </value>
                             <value name=’baz’>
                             <description> <loctext xml:lang=’C’>
                                Allow foo to access baz functions.
                             </loctext> </description>
                             </value>
                             <value name=’qux’>
                             <description> <loctext xml:lang=’C’>
                                Allow foo to access qux functions.
                              </loctext> </description>
                              </value>
                  </values>
                  <choices>
                      <include_values type=’values’/>
                  </choices>
           <prop_pattern>
        </pg_pattern>
</template>
```

文件

/usr/share/lib/xml/dtd/service_bundle.dtd.1

另请参见

svcs(1)、svccfg(1M)、libscf(3LIB)、service_bundle(4)、smf(5)

/usr/apache2/2.2/man

http://httpd.apache.org

引用名

solaris10 – Solaris 10 标记区域

描述

solaris10 标记使用 brands(5) 中介绍的标记区域框架，使得 Solaris 10 二进制应用程序无需经过修改即可在具有最新 Solaris 操作系统内核的计算机上运行。

Oracle Solaris 10 区域是用于承载在 Oracle Solaris 11 内核上运行的 x86 和 SPARC Solaris 10 9/10（或以后发行的 Oracle Solaris 10 更新）用户环境的 solaris10 标记区域。

请注意，如果首先在原始系统上安装内核修补程序 142909-17 (SPARC) 或 142910-17 (x86/x64) 或更高版本，则可以使用较早的 Oracle Solaris 10 发行版。

solaris10 标记包括将 Solaris 10 系统映像安装到非全局区域所必需的工具。它还支持将 Solaris 10 本地区域迁移到 solaris10 标记区域所必需的工具。该标记支持在运行最新 Solaris 操作系统的 SPARC 或 x86 计算机上执行 32 位和 64 位 Solaris 10 应用程序。

配置与管理

solaris10 标记支持完全根非全局区域模型。所有必需的 Solaris 10 软件及任何其他软件包都安装在区域的专用文件系统中。

区域只能位于其自己的 zfs(1M) 数据集中并且仅支持 ZFS。在安装或连接区域时，将自动创建 ZFS 数据集。如果无法创建 ZFS 数据集，区域也无法安装或连接。

zonecfg(1M) 实用程序可用于配置 solaris10 标记区域。在创建区域时可以使用 SYSsolaris10 模板，也可以手动设置配置。一旦安装了标记区域，便不能更改或删除该区域的标记。zoneadm(1M) 实用程序可用于报告区域的标记类型和管理区域。zlogin(1) 实用程序可用于登录到区域。

当前对于委托 ZFS 数据集配置的支持是试验性的，尚未经过测试。对于在超虚拟化 xVM 域中运行这些区域的支持是试验性的，并且在区域中运行的 64 位 x86 应用程序存在已知问题。不能在标记区域中配置 /dev/sound 设备。此外，在全局区域中用于检查 solaris10 标记区域中执行的进程时，mdb(1) 和 dtrace(1M) 无法完全正常运行。

solaris10 标记安装程序支持通过已安装的 Solaris 10 系统的映像安装区域。这可以是完整的 flash_archive(4)、cpio(1) 或 pax(1) xustar 归档文件。cpio 归档文件可以使用 gzip(1) 或 bzip2(1) 进行压缩。此映像也可以是级别为 0 的ufsdump(1M)，或者是到 Solaris 10 系统根目录树顶级的路径。不能从标准的 Solaris 10 分发介质中安装区域。

为了将本地区域从 Solaris 10 系统迁移到最新的 Solaris 操作系统内核，attach 子命令支持从已安装的 Solaris 10 本地区域的归档文件中安装区域。与安装程序一样，这可以是 zonepath 的 cpio(1) 或 pax(1) xustar 归档文件。cpio 归档文件可以使用 gzip(1) 或 bzip2(1) 进行压缩。映像还可以是到 Solaris 10 区域的 zonepath 目录树顶级的路径。除了从 Solaris 10 本地区域迁移外，相同的迁移选项也可以在将 solaris10 标记区域从一台主机迁移到另一台主机时使用。从 Solaris 10 进行迁移时，可以将区域配置为 sparseroot 区域。这种情况下，在生成归档文件前，应该在主机上已配置好区域。这样可以确保在归档文件中包含继承的目录。

子命令

支持 zoneadm(1M) 标记特定子命令的下列参数：

attach [-a *archive* | -d *path*] [-c *sysidcfg*]
将指定的 Solaris 10 本地区域映像连接到标记区域。如果 -a 或 -d 都未指定，则假定已使用区域文件正确安装区域的 zonepath。

-a *archive* 到已安装的 Solaris 10 本地区域或 solaris10 标记区域的 zonepath 的 cpio(1)、pax(1) xustar 或 zfs 归档文件的路径。cpio 和 zfs 归档文件可以使用 gzip 或 bzip2 进行压缩。

如果区域状态为“不可用”，则可能无法使用此选项。

警告：未来的 Solaris 发行版中可能会删除此选项。建议使用“zoneadm install -a archive”。

-c *sysidcfg* 指定 *sysidcfg* 文件将导致连接后在区域上发生 sys-unconfig。然后 *sysidcfg* 文件应用到区域。

-d *path* 到已安装的 Solaris 10 本地区域或 solaris10 标记区域的 zonepath 的 zonepath 目录的路径。

如果区域状态为“不可用”，则可能无法使用此选项。

警告：未来的 Solaris 发行版中可能会删除此选项。建议使用“zoneadm install -a archive”。

clone [-c sysidcfg]
安装某个区域时，可直接复制现有的已安装区域。该子命令是一种备选的区域安装方法。

-c sysidcfg 提供取消配置克隆的区域后将应用的 *sysidcfg* 文件。

install [-a *archive*] [-d *path*] [-p] [-s] [-u] [-v] \
[-c sysidcfg]
将指定的 Solaris 10 系统映像安装到区域中。要求使用 -u 或 -p 选项**以及** -a 或 - d 选项。

-a *archive*
到已安装的 Solaris 10 系统、已安装的 Solaris 10 本地区域或 Solaris 10 标记区域的 flash_archive(4)、cpio(1)、pax(1) xustar、zfs 归档文件或级别 0 ufsdump(1M) 的路径。cpio 和 zfs 归档文件可以使用 gzip 或 bzip2 进行压缩。

-c *sysidcfg*
提供安装后将应用的 *sysidcfg* 文件。

-d *path*
到已安装的 Solaris 10 系统的根目录的路径。

-p
在安装区域后保留系统配置。

-s
: 无提示安装。

-u
: 安装区域后对区域运行 sys-unconfig。

-v
: 安装过程的详细输出。

应用程序支持

solaris10 区域只支持用户级的 Solaris10 应用程序。不能使用 solaris10 区域中的 Solaris 10 设备驱动程序或 Solaris 10 内核模块。不过，取决于内核模块，您可能能够将最新的 Solaris 内核模块版本用于 Solaris 10 用户级应用程序。

示例

示例 1 创建用于安装的 ZFS Flash 归档文件

以下示例显示了如何为物理到虚拟 (P2V) 迁移创建归档文件。这是在运行 Solaris 10 的系统的全局区域中执行的。在 Solaris 10 系统中绝对不能配置、安装或运行任何非全局区域。Solaris 10 系统可以使用 ZFS 或 UFS 作为其根文件系统。

```
# flarcreate -n s10box -c /net/somehost/p2v/s10box.flar
```

示例 2 使用 Flash 归档文件安装 solaris10 标记区域

以下示例使用示例 1 中的归档文件安装区域。它假设已使用 zonecfg(1M) 配置了区域并将 brand 属性设置为 solaris10。

```
# zoneadm -z s10p2v install -a /net/somehost/p2v/s10box.flar -p
```

示例 3 创建用于附加的 ZFS 归档文件

以下示例显示了如何为虚拟到虚拟 (V2V) 迁移创建归档文件。它假设 solaris10 标记区域的 zonepath 为 /zones/v2vzone。

首先，确定 zonepath dataset 的名称。

```
# dataset=$(zfs list -H -o name /zones/v2vzone)
```

然后，创建区域数据集的快照。

```
# zfs snapshot -r $dataset@v2v
```

最后，生成使用 bzip2 压缩的 ZFS 自包含递归流。

```
# zfs send -rc $dataset@v2v | bzip2 > /net/somehost/v2v/v2v.zfs.bz2
```

示例 4 使用 ZFS 归档文件连接区域

以下示例使用 ZFS 归档文件连接区域。它假设已使用 zonecfg(1M) 配置了区域并将 brand 属性设置为 solaris10。

```
# zoneadm -z v2vzone attach -a /net/somehost/v2v/v2v.zfs.bz2
```

属性

有关以下属性的说明，请参见 attributes(5)：

属性类型	属性值
可用性	system/zones/brand/brand-solaris10
接口稳定性	Committed（已确定）

另请参见

cpio(1)、mdb(1)、pax(1)、zlogin(1)、dtrace(1M)、ufsdump(1M)、zfs(1M)、zoneadm(1M)、zonecf

引用名

solaris, ipkg – Solaris 标记区域

描述

solaris 标记使用标记区域框架（如 brands(5) 中所述）运行与安装在全局区域中的同一软件一起安装的区域。使用 solaris 标记时，系统软件必须始终与全局区域保持同步。区域中的系统软件包使用映像包管理系统进行管理。请参见 pkg(5)。

配置与管理

solaris 标记支持完全根非全局区域模型。所有必需的系统软件及任何其他软件包都安装在区域的专用文件系统中。区域只能位于其自己的 zfs(1M) 数据集中并且仅支持 ZFS。在安装或连接区域时，将自动创建 ZFS 数据集。如果无法创建 ZFS 数据集，区域也无法安装或连接。

子命令

以下 solaris 标记特定的子命令选项受 zoneadm(1M) 支持。

attach [-a *archive* | -d *path* | -z *ZBE*] [-u | -U] [-c *config_profile* .xml | *dir*]

将指定的 solaris 标记区域映像连接到区域。如果 -a 和 -d 均未指定，则假定区域的区域路径中已经正确安装区域文件。zoneadm 会在计算机上检查区域要关联的软件包级别。如果区域通过全局区域所依赖的包（具有不同的版本号）不同于源计算机上的相关包，zoneadm 将报告这些冲突并且不执行连接。

如果目标系统只有比源系统更新的相关软件包（修订版本号较高)，则可以使用 -u 或 -U 选项更新相关软件包以匹配位于新系统上的软件包修订版。

连接区域时，可能会存在多个区域引导环境 (zone boot environment, ZBE)，attach 子命令必须确定要连接其中哪一个。选择条件如下所示，并使用第一个匹配项。

- 如果使用 -z 选项来指定 ZBE，则选择它。
- 如果只有一个 ZBE，则选择它。
- 如果只有一个 ZBE 与该全局区域引导环境关联，则选择它。
- 如果只有一个活动的 ZBE 与该全局区域引导环境关联，则选择它。
- 如果只有一个活动的 ZBE，则选择它。

如果选定的 ZBE 与其他全局区域引导环境关联，则 ZBE 将被克隆，并且连接选定 ZBE 的克隆。有关引导环境的更多信息，请参见 beadm(1M)。

-a *archive*

警告：未来的 Solaris 发行版中可能会删除此选项。建议使用“zoneadm install -a archive”。

到已安装 Oracle Solaris 标记区域的 cpio(1) 或 pax(1) xustar 或 zfs(1M) 归档文件的路径。

cpio 和 pax 归档文件必须采用区域路径或区域根路径，并且不得包含绝对路径（路径不得以 / 开头）。ZFS 归档文件可以是 zonepath dataset、zone rpool dataset 或单独区域引导环境。

如果 ZFS 归档文件包含多个 ZBE，则连接可能失败，并显示一条消息，指示必须使用 -z 选项连接特定的 ZBE。在这种情况下，仍会提取归档文件中的 ZBE，-a 和 -d 选项均无法继续用于此区域，除非使用 -z 选项附加了所提取的 ZBE 之一。如

果不是希望附加所提取的 ZBE 之一，而是希望删除所提取的 ZBE，请在 zoneadm -z <zone> uninstall 前面使用 zoneadm -z <zone> mark incomplete。

cpio 和 ZFS 归档文件可以使用 gzip 或 bzip2 进行压缩。

如果区域状态为“不可用”，则可能无法使用此选项。

-c *config_profile*.xml | *dir*
提供从系统信息库安装后将应用的配置文件或配置文件的目录。

所有配置文件必须具有 .xml 扩展名。

-d *path*
警告：未来的 Solaris 发行版中可能会删除此选项。建议使用“zoneadm install -a archive”。

区域路径目录的路径，该目录属于 solaris 标记区域的区域路径。

如果区域状态为“不可用”，则可能无法使用此选项。

-u
更新区域中的最少数量的软件包，以允许区域的软件包与在全局区域中安装的软件包兼容。

-U
将区域中的所有软件包更新为与安装在全局区域中的软件包兼容的最新版本。

-z *ZBE*
连接指定的现有区域引导环境。如果指定的区域引导环境与其他全局区域关联，则指定的 ZBE 将被克隆，并连接 ZBE 的克隆。

clone [-c *config_profile*.xml | *dir*]

-c *config_profile*.xml | *dir*
提供从系统信息库安装后将应用的配置文件或配置文件的目录。

所有配置文件必须具有 .xml 扩展名。

install [-m *manifest.xml*] [-c *config_profile*.xml | *dir*]
install [-a *archive* | -d *path*] [-U] [-p] [-s] [-u] [-v] [-c *config_profile*.xml | *dir*]
solaris 标记安装程序支持从软件系统信息库或从运行同一发行版的已安装系统映像安装区域。此映像可以是 cpio(1)、pax(1) xustar 或 ZFS 归档文件。cpio 或 ZFS 归档文件可以使用 gzip 或 bzip2 进行压缩。该映像也可以是到系统根目录树顶级的路径，或者是预存在的区域路径。

如果未指定 -a 和 -d 选项，则从系统信息库安装区域。要在区域中安装其他软件包，可以复制并编辑缺省区域清单 /usr/share/auto_install/manifest/zone_default.xml 以包含所需的软件包。应使用 -m 选项指定安装该修改的清单。

要从系统或区域映像安装区域，需要使用 -a 或 -d 选项。如有必要，映像的活动 ZBE 中的软件将会进行最低限度的必要更新，以使其与全局区域的软件包兼容。如果指定 -U 选项，则会将映像的活动 ZBE 中的所有软件更新至与活动 ZBE 兼容的最新版本。如果使用 -a 或 -d 选项，则还需要使用 -u 或 -p 选项。

-a *archive*
: 到已安装全局区域或非全局区域的 cpio(1) 或 pax(1) xustar 或 ZFS 归档文件的路径。

  如果 ZFS 归档文件包含多个引导环境，则安装了活动的引导环境。如果安装无法确定哪个引导环境是活动的引导环境，安装程序将提供提取的引导环境列表，并建议一个使用 -z 选项的连接命令来连接特定的引导环境。

  cpio 和 ZFS 归档文件可以使用 gzip 或 bzip2 进行压缩。

-c *config_profile*.xml | *dir*
: 提供从系统信息库安装后将应用的配置文件或配置文件的目录。

  所有配置文件必须具有 .xml 扩展名。

-d *path*
: 区域路径目录的路径，该目录属于 solaris 标记区域的区域路径或 Solaris 11 全局区域根目录。

-m *manifest.xml*
: 将指定给自动化安装程序的清单文件。

-p
: 从归档文件或路径安装区域后，保留系统配置。

-s
: 无提示安装

-u
: 安装后取消配置系统。

-U
: 将区域中的所有软件包更新为与安装在全局区域中的软件包兼容的最新版本。只有使用了 -a 或 -d 选项，才能使用 -U 选项。

-v
: 安装过程的详细输出。

-z *ZBE*
: 连接指定的现有区域引导环境。如果指定的区域引导环境与其他全局区域关联，则指定的 ZBE 将被克隆，并连接 ZBE 的克隆。

示例

示例 1 为 P2V 创建 ZFS 归档文件

以下示例显示如何创建归档文件，以便进行物理到虚拟 (P2V) 迁移。这将在未配置、安装或运行非全局区域的系统的全局区域中执行。它假设根池名为 rpool。

首先，创建整个根池的快照。

```
# zfs snapshot -r rpool@p2v
```

示例 1 为 P2V 创建 ZFS 归档文件 （续）

然后，销毁与交换设备和转储设备关联的快照，因为在目标系统上不需要它们。

```
# zfs destroy rpool/swap@p2v
# zfs destroy rpool/dump@p2v
```

最后，生成使用 gzip 压缩的 ZFS 复制流归档文件。在本示例中，它存储在远程 NFS 服务器上。

```
# zfs send -R rpool@p2v | gzip > /net/somehost/p2v/p2v.zfs.gz
```

示例 2 为 V2V 创建 ZFS 归档文件

以下示例显示如何创建归档文件，以便进行虚拟到虚拟 (V2V) 迁移。它假设该区域的 zonepath 为 /zones/v2vzone。

首先，确定 zonepath 数据集的名称。

```
# dataset=$(zfs list -H -o name /zones/v2vzone)
```

然后，创建区域数据集的快照。

```
# zfs snapshot -r $dataset@v2v
```

最后，生成使用 bzip2 压缩的 ZFS 自包含递归流。

```
# zfs send -rc $dataset@v2v | bzip2 > /net/somehost/v2v/v2v.zfs.bz2
```

示例 3 使用 ZFS 归档文件安装区域

以下示例使用 ZFS 归档文件安装区域。

```
# zoneadm -z v2vzone install -p -a /net/somehost/v2v/v2v.zfs.bz2
```

可使用类似的命令来安装示例 1 中创建的 P2V 归档文件。

属性

有关以下属性的说明，请参见 attributes(5)：

属性类型	属性值
可用性	system/zones
接口稳定性	Uncommitted（未确定）

另请参见

cpio(1)、pax(1)、beadm(1M)、sysconfig(1M)、zfs(1M)、zlogin(1)、zonename(1)、zoneadm(1M)、

pkg(5)，在 IPS 整合中提供

引用名

standards, ANSI, C, C++, ISO, POSIX, POSIX.1, POSIX.2, SUS, SUSv2, SUSv3, SVID, SVID3, XNS, XNS4, XNS5, XPG, XPG3, XPG4, XPG4v2 – Oracle Solaris 支持的标准和规范

描述

Oracle Solaris 支持 IEEE 标准 1003.1 和 IEEE 标准 1003.2（通常分别称为 POSIX.1 和 POSIX.2）。下表列出了这些标准的每个版本及其简要说明，以及最先遵从这些标准的 SunOS 或 Solaris 发行版。

POSIX 标准	说明	发行版
POSIX.1-1988	系统接口和系统头	SunOS 4.1
POSIX.1-1990	POSIX.1-1988 更新	Solaris 2.0
POSIX.1b-1993	实时扩展	Solaris 2.4
POSIX.1c-1996	线程扩展	Solaris 2.6
POSIX.2-1992	shell 和实用程序	Solaris 2.5
POSIX.2a-1992	交互式 shell 和实用程序	Solaris 2.5
POSIX.1–2001	POSIX.1-1990、POSIX.1b-1993、POSIX.1c-1996、POSIX.2-1992 和 POSIX.2a-1992 更新	Solaris 10
POSIX.1-2004	POSIX.1-2001 更新	Oracle Solaris 11
POSIX.1-2008	POSIX.1-2004 更新	Oracle Solaris 11

Oracle Solaris 还支持 X/Open Common Applications Environment (CAE) Portability Guide Issue 3 (XPG3) 和 Issue 4 (XPG4)、单一 UNIX 规范（Single UNIX Specification，SUS，亦称为 XPG4v2）、单一 UNIX 规范版本 2 (Single UNIX Specification, Version 2, SUSv2) 以及单一 UNIX 规范版本 3 (Single UNIX Specification, Version 3, SUSv3)。XPG4 和 SUS 均包括 Networking Services Issue 4 (XNS4)。SUSv2 包括 Networking Services Issue 5 (XNS5)。

下表列出了每个 X/Open 规范及其简要说明，以及最先遵从这些标准的 SunOS 或 Solaris 发行版。

X/Open CAE		
规范	说明	发行版
XPG3	POSIX.1-1988 的超集，包含 SVID3 中的实用程序	SunOS 4.1
XPG4	POSIX.1-1990、POSIX.2-1992 和 POSIX.2a-1992 的超集，包含从 XPG3 到 POSIX 标准的扩展	Solaris 2.4
SUS (XPG4v2)	XPG4 的超集，包含常见应用程序软件包广泛使用的旧式 BSD 接口	Solaris 2.6

X/Open CAE 规范	说明	发行版
XNS4	套接字和 XTI 接口	Solaris 2.6
SUSv2	SUS 的超集，已扩展为支持 POSIX.1b-1993、POSIX.1c-1996 和 ISO/IEC 9899（C 标准）修订 1	Solaris 7
XNS5	XNS4 的超集和干净的 LP64 派生接口。	Solaris 7
SUSv3	与 POSIX.1–2001 相同	Solaris 10

XNS4 规范只有在 ILP32（32 位）环境中使用才是安全的，不应该用于 LP64（64 位）应用程序环境。在这种环境中，应使用 XNS5 或 SUSv3，XNS5 或 SUSv3 具有可在 ILP32 和 LP64 环境中移植的干净的 LP64 接口。Solaris 发行版 7 到 Oracle Solaris 11 支持 ILP32 和 LP64 两种环境。

Solaris 发行版 7 到 10 标为符合 The Open Group 的 UNIX 98 产品标准。Solaris 10 到 Oracle Solaris 11 标为符合 The Open Group 的 UNIX 03 产品标准。

Solaris 发行版 2.0 到 Oracle Solaris 11 支持 System V 接口定义第 3 版的卷 1 至卷 4 (SVID3) 指定的接口。然而请注意，由于此规范的开发者 (UNIX Systems Laboratories) 已不再运营并且此规范遵从 POSIX 和 X/Open CAE 规范，因此对符合此规范当前所要达到的要求有一些分歧。

安装 Oracle Solaris Studio 12.3 C 编译器后，Oracle Solaris 11 即支持 ANSI X3.159-1989 编程语言 - C 和 ISO/IEC 9899:1990 编程语言 - C (C) 接口。

安装 Oracle Solaris Studio 12.3 C 编译器后，Oracle Solaris 11 即支持 ISO/IEC 9899:1990 修订 1:1995: C Integrity。

安装 Oracle Solaris Studio 12.3 C 编译器后，Oracle Solaris 11 即支持 ISO/IEC 9899:1999 编程语言 - C。

安装 Oracle Solaris Studio 12.3 C++ 编译器后，Oracle Solaris 11 即支持 ISO/IEC 14882:1998 编程语言 - C++。编译器自述文件中介绍了该标准不支持的功能。

实用程序

如果 POSIX.2、POSIX.2a、XPG4、SUS 或 SUSv2 要求的行为与旧式 Solaris 实用程序行为冲突，将不会更改实用程序的原始 Solaris 版本；`/usr/xpg4/bin` 中提供了符合标准的新版本。如果 POSIX.1–2001 或 SUSv3 要求的行为与旧式 Solaris 实用程序行为冲突，`/usr/xpg4/bin` 或 `/usr/xpg6/bin` 中提供了符合标准的新版本。如果 POSIX.1–2001 或 SUSv3 要求的行为与 POSIX.2、POSIX.2a、SUS 或 SUSv2 冲突，`/usr/xpg6/bin` 中提供了符合 SUSv3 标准的新版本。

如果应用程序要使用符合标准的实用程序，则必须设置 `PATH`（`sh(1)` 或 `ksh(1)`）或者 `path` (`csh(1)`) 环境变量，以指定顺序指定下列目录来获取相应的实用程序：

SVID3、XPG3

1. `/usr/bin`
2. 包含编译器的二进制文件的目录
3. 包含应用程序所需的二进制文件的其他目录

POSIX.2、POSIX.2a、SUS、SUSv2、XPG4

1. `/usr/xpg4/bin`
2. `/usr/bin`
3. 包含编译器的二进制文件的目录
4. 包含应用程序所需的二进制文件的其他目录

POSIX.1–2001、SUSv3

1. `/usr/xpg6/bin`
2. `/usr/xpg4/bin`
3. `/usr/bin`
4. 包含编译器的二进制文件的目录
5. 包含应用程序所需的二进制文件的其他目录

当应用程序使用 `execlp()` 或 `execvp()`（请参见 `exec(2)`）来执行 shell 文件或使用 `system(3C)` 时，用于解释 shell 文件的 shell 取决于调用者遵循的标准：

标准	使用的 Shell
1989 ANSI C、1990 ISO C、1999 ISO C、POSIX.1 (1990–2001)、SUS、SUSv2、SUSv3、XPG4	`/usr/xpg4/bin/sh`
POSIX.1 (1988)、SVID3、XPG3 或未指定标准	`/usr/bin/sh`

功能测试宏

功能测试宏供应用程序用来指示用户需要的 C 标准指定的那些功能之外的其他功能集。如果应用程序仅使用特定标准（如 POSIX 或 X/Open CAE）定义的接口和头，则仅需要定义该标准指定的相应功能测试宏。如果应用程序要使用该标准未定义的接口和头，则除了定义相应的标准功能测试宏之外，还必须定义 `__EXTENSIONS__`。通过定义 `__EXTENSIONS__`，应用程序便可以访问与指定的标准不冲突的所有接口和头。应用程序必须在编译命令行或应用程序源文件中定义 `__EXTENSIONS__`。

1989 ANSI C、1990 ISO C、1999 ISO C

无需定义功能测试宏来指示应用程序符合 C 标准。

ANSI/ISO C++

ANSI/ISO C++ 未定义任何功能测试宏。如果编译器根据编译器缺省值和命令行选项预定义的标准 C++ 声明宏 `__cplusplus` 设置为 199711 或更大值，则编译器会以符合标准的模式运行，并指示符合 C++ 标准。值 199711 指示符合该标准要求的 ISO/IEC 14882:1998。（如上所述，该标准的符合性并不完整。）

未针对 POSIX 或 X/Open CAE 定义 C++ 绑定，因此指定功能测试宏（如 _POSIX_SOURCE、_POSIX_C_SOURCE 和 _XOPEN_SOURCE）会由于与 C++ 标准和其他规范的要求冲突而导致编译错误。

POSIX

要符合 POSIX.1 的应用程序必须在加入任何头之前定义该标准指定的功能测试宏。对于以下列出的标准，应用程序必须定义所列出的功能测试宏。应用程序编写者必须在对应的标准中查找其他可查询的宏，以确定实现是否支持所需的选项。

POSIX 标准	功能测试宏
POSIX.1-1990	_POSIX_SOURCE
POSIX.1-1990 和 POSIX.2-1992 C 语言绑定选项	_POSIX_SOURCE 和 _POSIX_C_SOURCE=2
POSIX.1b-1993	_POSIX_C_SOURCE=199309L
POSIX.1c-1996	_POSIX_C_SOURCE=199506L
POSIX.1–2001	_POSIX_C_SOURCE=200112L

SVID3

SVID3 规范未指定任何功能测试宏用于指示编写的应用程序符合 SVID3 要求。SVID3 规范是在 C 标准完成之前编写的。

X/Open CAE

要构建或编译符合 X/Open CAE 规范之一的应用程序，请遵循以下准则。如果应用程序同时需要 CAE 和 POSIX 功能，则不需要设置 POSIX 功能测试宏。

XPG3　应用程序必须定义 _XOPEN_SOURCE。如果 _XOPEN_SOURCE 定义了值，该值必须小于 500。

XPG4　应用程序必须定义 _XOPEN_SOURCE 并设置 _XOPEN_VERSION=4。如果 _XOPEN_SOURCE 定义了值，该值必须小于 500。

SUS (XPG4v2)　应用程序必须定义 _XOPEN_SOURCE 并设置 _XOPEN_SOURCE_EXTENDED=1。如果 _XOPEN_SOURCE 定义了值，该值必须小于 500。

SUSv2　应用程序必须定义 _XOPEN_SOURCE=500。

SUSv3　应用程序必须定义 _XOPEN_SOURCE=600。

编译

Oracle Solaris Studio 12.3 C 编译器提供符合 ISO/IEC 99899:1999（1999 ISO C 语言）标准的编译系统以及 c99 实用程序。

将 ld 直接用于链接应用程序时，必须在任何 link/load 命令行中指定 /usr/lib/values-xpg4.o，除非应用程序符合 POSIX.1–2001 或 SUSv3，在这种情况下必须在任何 link/load 编译行中指定 /usr/lib/values-xpg6.o。将 cc 或 CC 用于链接应用程序时，编译器会自动添加相应的文件。但是，构建应用程序的首选方法如下表所述。

符合 XNS4 或 XNS5 的应用程序必须在任何 link/load 命令行中包括 -l XNS，并且要分别定义针对 SUS 或 SUSv2 指定的功能测试宏。

如果编译器支持 redefine_extname pragma 功能（Oracle Solaris Studio 12.3 C 编译器和 Oracle Solaris Studio 12.3 C++ 编译器定义了 __PRAGMA_REDEFINE_EXTNAME 宏来指示它们支持此功能），则标准头会使用 #pragma redefine_extname 指令将函数名称正确映射到库入口点名称。此映射完全支持 ISO C、POSIX 和 X/Open 名称空间保留项。

如果编译器不支持此 pragma 功能，则标准头会使用 #define 指令将内部函数名称映射到相应的库入口点名称。在这种情况下，应用程序应避免显式使用 lf64(5) 手册页上列出的 64 位文件偏移符号，因为实现会使用这些名称来命名替代的入口点。

使用 Oracle Solaris Studio 12.3 C 编译器时，符合以上所列规范的应用程序应使用下表中指出的实用程序和标志进行编译：

Specification	Compiler/Flags	Feature Test Macros
1989 ANSI C and 1990 ISO C	c89	none
1999 ISO C	c99	none
SVID3	cc -Xt -xc99=none	none
POSIX.1-1990	c89	_POSIX_SOURCE
POSIX.1-1990 and POSIX.2-1992 C-Language Bindings Option	c89	_POSIX_SOURCE and POSIX_C_SOURCE=2
POSIX.1b-1993	c89	_POSIX_C_SOURCE=199309L
POSIX.1c-1996	c89	_POSIX_C_SOURCE=199506L
POSIX.1-2001	c99	_POSIX_C_SOURCE=200112L
POSIX.1c-1996	c89	_POSIX_C_SOURCE=199506L
CAE XPG3	cc -Xa -xc99=none	_XOPEN_SOURCE
CAE XPG4	c89	_XOPEN_SOURCE and _XOPEN_VERSION=4

SUS (CAE XPG4v2) (includes XNS4)	c89	_XOPEN_SOURCE and _XOPEN_SOURCE_EXTENDED=1
SUSv2 (includes XNS5)	c89	_XOPEN_SOURCE=500
SUSv3	c99	_XOPEN_SOURCE=600

对于支持 LP64（64 位）编程环境的平台，应使用以下格式的命令行来构建符合 SUSv2 的使用 XNS5 库调用的 LP64 应用程序：

```
c89 $(getconf XBS5_LP64_OFF64_CFLAGS) -D_XOPEN_SOURCE=500 \
    $(getconf XBS5_LP64_OFF64_LDFLAGS) foo.c -o foo \
    $(getconf XBS5_LP64_OFF64_LIBS) -lxnet
```

应使用以下格式的命令行来构建符合 SUSv3 的类似 LP64 应用程序：

```
c99 $(getconf POSIX_V6_LP64_OFF64_CFLAGS) -D_XOPEN_SOURCE=600 \
    $(getconf POSIX_V6_LP64_OFF64_LDFLAGS) foo.c -o foo \
    $(getconf POSIX_V6_LP64_OFF64_LIBS) -lxnet
```

SUSv3

c99	_XOPEN_SOURCE=600

另请参见

csh(1)、ksh(1)、sh(1)、exec(2)、sysconf(3C)、system(3C)、environ(5)、lf64(5)

引用名

sticky – mark files for special treatment

描述

The *sticky bit* (file mode bit `01000`, see `chmod(2)`) is used to indicate special treatment of certain files and directories. A directory for which the sticky bit is set restricts deletion of files it contains. A file in a sticky directory can only be removed or renamed by a user who has write permission on the directory, and either owns the file, owns the directory, has write permission on the file, or is a privileged user. Setting the sticky bit is useful for directories such as `/tmp`, which must be publicly writable but should deny users permission to arbitrarily delete or rename the files of others.

If the sticky bit is set on a regular file and no execute bits are set, the system's page cache will not be used to hold the file's data. This bit is normally set on swap files of diskless clients so that accesses to these files do not flush more valuable data from the system's cache. Moreover, by default such files are treated as swap files, whose inode modification times may not necessarily be correctly recorded on permanent storage.

Any user may create a sticky directory. See `chmod` for details about modifying file modes.

另请参见

`chmod(1)`, `chmod(2)`, `chown(2)`, `mkdir(2)`, `rename(2)`, `unlink(2)`

已知问题

The `mkdir(2)` function will not create a directory with the sticky bit set.

引用名

suri – storage URIs

描述

Storage URIs uniquely identify storage objects across different nodes.

The following Storage URIs are supported:

Dev URI

```
dev:<local-path-under-/dev>
dev:///<path-with-dev>
dev:<absolute-path-with-dev>
```

Examples:

```
dev:dsk/c0t0d0s0
dev:///dev/dsk/c0t0d0s0
dev:/dev/dsk/c0t0d0s0
```

The dev URI specifies an object in the form of an existing device. All forms of the dev URI are functionally equivalent. The “`dev:dsk`” format with no authority section and an implied “`/dev/`” prefix is usually preferred. The path must be a device in the `/dev` name space.

Components of the device path may only include characters from POSIX.1-2008 Portable Filename Character Set, 0-9a-zA-Z._-.

Logical Unit URI

```
lu:luname.naa.<ID>
lu:initiator.naa.<ID>,target.naa.<ID>,luname.naa.<ID>
```

Examples:

```
lu:luname.naa.5000c5000288fa25
lu:initiator.naa.2100001d38089fb0,target.naa.2100001d38089fb0,luname.naa.5000c5000288fa25
```

Logical unit URI specifies a logical unit attached via fibre channel or serial-attached SCSI. In the luname-only URI form, the ID describes a logical unit name. In the initiator,target,luname form, an initiator specifies an initiator port and a target specifies a target port, and together they specify a path to the logical unit whose name must match the URI's luname part as in the luname-only form.

IDs are represented by using the SCSI-3 VPD page (0x83) global unique identifier (GUID) formatted as hexadecimal numbers (64/128 bits). IDs are case insensitive. Only GUIDs based on NAA identifiers are supported. See NOTES for information on NAA identifier format.

In general it is recommended using multipathing in conjunction with luname-only URIs. If multipathing is disabled and a luname-only URI is used, a random path to the specified logical unit will be chosen. To avoid this random path assignment an initiator,target,luname URI form can be used to select a specific path to a logical unit. If multipathing is enabled and an initiator,target,luname URI form is used then the multipathing framework controls which paths are used to access the logical unit and the URI is only used to identify the unit, not the access path.

If an initiator or/and a target IDs are not matched but the logical unit is found based on a logical unit name, the unit will be accessed via an alternate path.

iSCSI URI

```
iscsi:///luname.naa.<ID>
iscsi://<host>[:<port>]/luname.naa.<ID>
```

Examples:

```
iscsi:///luname.naa.600144f03d70c80000004ea57da10001
iscsi://[::1]/luname.naa.600144f03d70c80000004ea57da10001
iscsi://127.0.0.1/luname.naa.600144f03d70c80000004ea57da10001
iscsi://127.0.0.1:3620/luname.naa.600144f03d70c80000004ea57da10001
iscsi://hostname:3620/luname.naa.600144f03d70c80000004ea57da10001
```

The iSCSI URI specifies an object based on a logical unit number accessible via iSCSI protocol. Optional `hostname`[:<`port`>] authority section provides information for a consumer to set up a send-targets discovery address. IPv6 addresses must be enclosed in square brackets.

Format of an ID is explained in the “Logical Unit URI” section. See NOTES for information on NAA identifier format.

Getting URIs

To avoid errors by generating URIs manually, the `suriadm`(1M) subcommand `lookup-uri` is recommended to be used to generate URIs based on existing system device paths. Such device paths must be identified via other means. For Fibre Channel devices, `fcadm`(1M) may be used. For Serial Attached SCSI devices, `sasinfo`(1M) may be used. For iSCSI devices, `iscsiadm`(1M) may be used.

属性

See `attributes`(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/library/storage/suri
Interface Stability	Commited

另请参见

`fcadm`(1M), `iscsiadm`(1M), `sasinfo`(1M), `suriadm`(1M), `attributes`(5)

Small Computer System Interface-3 (SCSI-3)

附注

NAA stands for “Name Address Authority” and it is a field within an identifier that specifies the format and length of that identifier. NAA identifier format is defined in SCSI Primary Commands - 3 specification, section “7.6.3.6 NAA identifier format”. WWN (World Wide Name) identifiers use an NAA identifier format.

引用名

tecla, teclarc – User interface provided by the tecla library.

描述

This man page describes the command-line editing features that are available to users of programs that read keyboard input via the tecla library. Users of the `tcsh shell` will find the default key bindings very familiar. Users of the `bash` shell will also find it quite familiar, but with a few minor differences, most notably in how forward and backward searches through the list of historical commands are performed. There are two major editing modes, one with `emacs`-like key bindings and another with `vi`-like key bindings. By default `emacs` mode is enabled, but `vi(1)` mode can alternatively be selected via the user's configuration file. This file can also be used to change the bindings of individual keys to suit the user's preferences. By default, tab completion is provided. If the application hasn't reconfigured this to complete other types of symbols, then tab completion completes file names.

Key Sequence Notation

In the rest of this man page, and also in all tecla configuration files, key sequences are expressed as follows.

`^A` or `C-a`
: This is a 'CONTROL-A', entered by pressing the CONTROL key at the same time as the 'A' key.

`\E` or `M-`
: In key sequences, both of these notations can be entered either by pressing the ESCAPE key, then the following key, or by pressing the META key at the same time as the following key. Thus the key sequence `M-p` can be typed in two ways, by pressing the ESCAPE key, followed by pressing 'P', or by pressing the META key at the same time as 'P'.

up
: This refers to the up-arrow key.

down
: This refers to the down-arrow key.

left
: This refers to the left-arrow key.

right
: This refers to the right-arrow key.

a
: This is just a normal 'A' key.

The Tecla Configuration File

By default, tecla looks for a file called `.teclarc` in your home directory (ie. `~/.teclarc`). If it finds this file, it reads it, interpreting each line as defining a new key binding or an editing configuration option. Since the `emacs` key-bindings are installed by default, if you want to use the non-default `vi` editing mode, the most important item to go in this file is the following line:

```
edit-mode vi
```

This will re-configure the default bindings for `vi`-mode. The complete set of arguments that this command accepts are:

vi
: Install key bindings like those of the `vi` editor.

emacs
: Install key bindings like those of the `emacs` editor. This is the default.

none
: Use just the native line editing facilities provided by the terminal driver.

To prevent the terminal bell from being rung, such as when an unrecognized control-sequence is typed, place the following line in the configuration file:

```
nobeep
```

An example of a key binding line in the configuration file is the following.

```
bind M-[2~ insert-mode
```

On many keyboards, the above key sequence is generated when one presses the insert key, so with this key binding, one can toggle between the `emacs`-mode insert and overwrite modes by hitting one key. One could also do it by typing out the above sequence of characters one by one. As explained above, the `M-` part of this sequence can be typed either by pressing the ESCAPE key before the following key, or by pressing the META key at the same time as the following key. Thus if you had set the above key binding, and the insert key on your keyboard didn't generate the above key sequence, you could still type it in either of the following 2 ways.

1. Hit the ESCAPE key momentarily, then press '[', then '2', then finally '~'.
2. Press the META key at the same time as pressing the '[' key, then press '2', then '~'.

If you set a key binding for a key sequence that is already bound to a function, the new binding overrides the old one. If in the new binding you omit the name of the new function to bind to the key sequence, the original binding becomes undefined.

Starting with versions of `libtecla` later than 1.3.3 it is now possible to bind key sequences that begin with a printable character. Previously key sequences were required to start with a CONTROL or META character.

Note that the special keywords "up", "down", "left", and "right" refer to the arrow keys, and are thus not treated as key sequences. So, for example, to rebind the up and down arrow keys to use the history search mechanism instead of the simple history recall method, you could place the following in your configuration file:

```
bind up history-search-backwards
bind down history-search-backwards
```

To unbind an existing binding, you can do this with the bind command by omitting to name any action to rebind the key sequence to. For example, by not specifying an action function, the following command unbinds the default beginning-of-line action from the `^A` key sequence:

```
bind ^A
```

If you create a `~/.teclarc` configuration file, but it appears to have no effect on the program, check the documentation of the program to see if the author chose a different name for this file.

Filename and Tilde Completion

With the default key bindings, pressing the TAB key (aka. ^I) results in tecla attempting to complete the incomplete file name that precedes the cursor. Tecla searches backwards from the cursor, looking for the start of the file name, stopping when it hits either a space or the start of the line. If more than one file has the specified prefix, then tecla completes the file name up to the point at which the ambiguous matches start to differ, then lists the possible matches.

In addition to literally written file names, tecla can complete files that start with ~/ and ~user/ expressions and that contain $envvar expressions. In particular, if you hit TAB within an incomplete ~user, expression, tecla will attempt to complete the username, listing any ambiguous matches.

The completion binding is implemented using the cpl_complete_word() function, which is also available separately to users of this library. See the cpl_complete_word(3TECLA) man page for more details.

Filename Expansion

With the default key bindings, pressing ^X* causes tecla to expand the file name that precedes the cursor, replacing ~/ and ~user/ expressions with the corresponding home directories, and replacing $envvar expressions with the value of the specified environment variable, then if there are any wildcards, replacing the so far expanded file name with a space-separated list of the files which match the wild cards.

The expansion binding is implemented using the ef_expand_file() function. See the ef_expand_file(3TECLA) man page for more details.

Recalling Previously Typed Lines

Every time that a new line is entered by the user, it is appended to a list of historical input lines maintained within the GetLine resource object. You can traverse up and down this list using the up and down arrow keys. Alternatively, you can do the same with the ^P, and ^N keys, and in vi command mode you can alternatively use the k and j characters. Thus pressing up-arrow once, replaces the current input line with the previously entered line. Pressing up-arrow again, replaces this with the line that was entered before it, etc.. Having gone back one or more lines into the history list, one can return to newer lines by pressing down-arrow one or more times. If you do this sufficient times, you will return to the original line that you were entering when you first hit up-arrow.

Note that in vi mode, all of the history recall functions switch the library into command mode.

In emacs mode the M-p and M-n keys work just like the ^P and ^N keys, except that they skip all but those historical lines which share the prefix that precedes the cursor. In vi command mode the upper case 'K' and 'J' characters do the same thing, except that the string that they search for includes the character under the cursor as well as what precedes it.

Thus for example, suppose that you were in emacs mode, and you had just entered the following list of commands in the order shown:

```
ls ~/tecla/
cd ~/tecla
ls -l getline.c
emacs ~/tecla/getline.c
```

If you next typed:

```
ls
```

and then hit M-p, then rather than returning the previously typed emacs line, which doesn't start with "ls", tecla would recall the "ls -l getline.c" line. Pressing M-p again would recall the "ls ~/tecla/" line.

Note that if the string that you are searching for, contains any of the special characters, *, ?, or '[', then it is interpretted as a pattern to be matched. Thus, cotinuing with the above example, after typing in the list of commands shown, if you then typed:

```
*tecla*
```

and hit M-p, then the "emacs ~/tecla/getline.c" line would be recalled first, since it contains the word tecla somewhere in the line, Similarly, hitting M-p again, would recall the "ls ~/tecla/" line, and hitting it once more would recall the "ls ~/tecla/" line. The pattern syntax is the same as that described for file name expansion, in the ef_expand_file(3TECLA).

History Files

Authors of programs that use the tecla library have the option of saving historical command-lines in a file before exiting, and subsequently reading them back in from this file when the program is next started. There is no standard name for this file, since it makes sense for each application to use its own history file, so that commands from different applications don't get mixed up.

International Character Sets

Since libtecla version 1.4.0, tecla has been 8-bit clean. This means that all 8-bit characters that are printable in the user's current locale are now displayed verbatim and included in the returned input line. Assuming that the calling program correctly contains a call like the following,

```
setlocale(LC_CTYPE, "");
```

then the current locale is determined by the first of the environment variables LC_CTYPE, LC_ALL, and LANG, that is found to contain a valid locale name. If none of these variables are defined, or the program neglects to call setlocale, then the default C locale is used, which is US 7-bit ASCII. On most unix-like platforms, you can get a list of valid locales by typing the command:

```
locale -a
```

at the shell prompt.

Meta Keys and Locales

Beware that in most locales other than the default C locale, META characters become printable, and they are then no longer considered to match M-c style key bindings. This allows international characters to be entered with the compose key without unexpectedly triggering META key bindings. You can still invoke META bindings, since there are actually two ways to do this. For example the binding M-c can also be invoked by pressing the ESCAPE key momentarily, then pressing the c key, and this will work regardless of locale. Moreover, many modern terminal emulators, such as gnome's gnome-terminal's and KDE's konsole terminals, already generate escape pairs like this when you use the META key, rather than a real meta character, and other emulators usually have a way to request this behavior, so you can continue to use the META key on most systems.

For example, although xterm terminal emulators generate real 8-bit meta characters by default when you use the META key, they can be configured to output the equivalent escape pair by setting their EightBitInput X resource to False. You can either do this by placing a line like the following in your ~/.Xdefaults file,

```
XTerm*EightBitInput: False
```

or by starting an xterm with an -xrm '*EightBitInput: False' command-line argument. In recent versions of xterm you can toggle this feature on and off with the 'Meta Sends Escape' option in the menu that is displayed when you press the left mouse button and the CONTROL key within an xterm window. In CDE, dtterms can be similarly coerced to generate escape pairs in place of meta characters, by setting the Dtterm*KshMode resource to True.

Entering International Characters

If you don't have a keyboard that generates all of the international characters that you need, there is usually a compose key that will allow you to enter special characters, or a way to create one. For example, under X windows on unix-like systems, if your keyboard doesn't have a compose key, you can designate a redundant key to serve this purpose with the xmodmap command. For example, on many PC keyboards there is a microsoft-windows key, which is otherwise useless under Linux. On a laptop, for example, the xev program might report that pressing this key generates keycode 115. To turn this key into a COMPOSE key, do the following:

```
xmodmap -e ’keycode 115 = Multi_key’
```

Type this key followed by a " character to enter an 'I' with a umlaut over it.

The Available Key Binding Functions

The following is a list of the editing functions provided by the tecla library. The names in the leftmost column of the list can be used in configuration files to specify which function a given key or combination of keys should invoke. They are also used in the next two sections to list the default key bindings in emacs and vi modes.

user-interrupt	Send a SIGINT signal to the parent process.
suspend	Suspend the parent process.

stop-output
Pause terminal output.

start-output
Resume paused terminal output.

literal-next
Arrange for the next character to be treated as a normal character. This allows control characters to be entered.

cursor-right
Move the cursor one character right.

cursor-left
Move the cursor one character left.

insert-mode
Toggle between insert mode and overwrite mode.

beginning-of-line
Move the cursor to the beginning of the line.

end-of-line
Move the cursor to the end of the line.

delete-line
Delete the contents of the current line.

kill-line
Delete everything that follows the cursor.

backward-kill-line
Delete all characters between the cursor and the start of the line.

forward-word
Move to the end of the word which follows the cursor.

forward-to-word
Move the cursor to the start of the word that follows the cursor.

backward-word
Move to the start of the word which precedes the cursor.

goto-column
Move the cursor to the 1-relative column in the line specified by any preceding digit-argument sequences (see Entering Repeat Counts below).

find-parenthesis
If the cursor is currently over a parenthesis character, move it to the matching parenthesis character. If not over a parenthesis character move right to the next close parenthesis.

forward-delete-char
Delete the character under the cursor.

backward-delete-char
Delete the character which precedes the cursor.

list-or-eof
This is intended for binding to `^D`. When invoked when the cursor is within the line it displays all possible completions then redisplays the line unchanged. When invoked on an empty line, it signals end-of-input (EOF) to the caller of `gl_get_line()`.

del-char-or-list-or-eof
This is intended for binding to `^D`. When invoked when the cursor is within the line it invokes forward-delete-char. When invoked at the end of the line it displays all possible

completions then redisplays the line unchanged. When invoked on an empty line, it signals end-of-input (EOF) to the caller of `gl_get_line()`.

forward-delete-word
: Delete the word which follows the cursor.

backward-delete-word
: Delete the word which precedes the cursor.

upcase-word
: Convert all of the characters of the word which follows the cursor, to upper case.

downcase-word
: Convert all of the characters of the word which follows the cursor, to lower case.

capitalize-word
: Capitalize the word which follows the cursor.

change-case
: If the next character is upper case, toggle it to lower case and vice versa.

redisplay
: Redisplay the line.

clear-screen
: Clear the terminal, then redisplay the current line.

transpose-chars
: Swap the character under the cursor with the character just before the cursor.

set-mark
: Set a mark at the position of the cursor.

exchange-point-and-mark
: Move the cursor to the last mark that was set, and move the mark to where the cursor used to be.

kill-region
: Delete the characters that lie between the last mark that was set, and the cursor.

copy-region-as-kill
: Copy the text between the mark and the cursor to the cut buffer, without deleting the original text.

yank
: Insert the text that was last deleted, just before the current position of the cursor.

append-yank
: Paste the current contents of the cut buffer, after the cursor.

up-history
: Recall the next oldest line that was entered. Note that in `vi` mode you are left in command mode.

down-history
: Recall the next most recent line that was entered. If no history recall session is currently active, the next line from a previous recall session is recalled. Note that in vi mode you are left in command mode.

history-search-backward
: Recall the next oldest line who's prefix matches the string which currently precedes the cursor (in `vi` command-mode

the character under the cursor is also included in the search string). Note that in `vi` mode you are left in command mode.

history-search-forward

Recall the next newest line who's prefix matches the string which currently precedes the cursor (in `vi` command-mode the character under the cursor is also included in the search string). Note that in `vi` mode you are left in command mode.

history-re-search-backward

Recall the next oldest line who's prefix matches that established by the last invocation of either history-search-forward or history-search-backward.

history-re-search-forward

Recall the next newest line who's prefix matches that established by the last invocation of either history-search-forward or history-search-backward.

complete-word

Attempt to complete the incomplete word which precedes the cursor. Unless the host program has customized word completion, file name completion is attempted. In `vi` commmand mode the character under the cursor is also included in the word being completed, and you are left in `vi` insert mode.

expand-filename

Within the command line, expand wild cards, tilde expressions and dollar expressions in the file name which immediately precedes the cursor. In `vi` commmand mode the character under the cursor is also included in the file name being expanded, and you are left in `vi` insert mode.

list-glob

List any file names which match the wild-card, tilde and dollar expressions in the file name which immediately precedes the cursor, then redraw the input line unchanged.

list-history

Display the contents of the history list for the current history group. If a repeat count of `> 1` is specified, only that many of the most recent lines are displayed. See the Entering Repeat Counts section.

read-from-file

Temporarily switch to reading input from the file who's name precedes the cursor.

read-init-files

Re-read `teclarc` configuration files.

beginning-of-history

Move to the oldest line in the history list. Note that in `vi` mode you are left in command mode.

end-of-history

Move to the newest line in the history list (ie. the current line). Note that in `vi` mode this leaves you in command mode.

digit-argument

Enter a repeat count for the next key binding function. For details, see the Entering Repeat Counts section.

newline

Terminate and return the current contents of the line, after appending a newline character. The newline character is normally '\n', but will be the first character of the key sequence that invoked the newline action, if this happens to be a printable character. If the action was invoked by the '\n' newline character or the '\r' carriage return character, the line is appended to the history buffer.

repeat-history

Return the line that is being edited, then arrange for the next most recent entry in the history buffer to be recalled when tecla is next called. Repeatedly invoking this action causes successive historical input lines to be re-executed. Note that this action is equivalent to the 'Operate' action in ksh.

ring-bell

Ring the terminal bell, unless the bell has been silenced via the nobeep configuration option (see The Tecla Configuration File section).

forward-copy-char

Copy the next character into the cut buffer (NB. use repeat counts to copy more than one).

backward-copy-char

Copy the previous character into the cut buffer.

forward-copy-word

Copy the next word into the cut buffer.

backward-copy-word

Copy the previous word into the cut buffer.

forward-find-char

Move the cursor to the next occurrence of the next character that you type.

backward-find-char

Move the cursor to the last occurrence of the next character that you type.

forward-to-char

Move the cursor to the character just before the next occurrence of the next character that the user types.

backward-to-char

Move the cursor to the character just after the last occurrence before the cursor of the next character that the user types.

repeat-find-char

Repeat the last backward-find-char, forward-find-char, backward-to-char or forward-to-char.

invert-refind-char

Repeat the last backward-find-char, forward-find-char, backward-to-char, or forward-to-char in the opposite direction.

delete-to-column
Delete the characters from the cursor up to the column that is specified by the repeat count.

delete-to-parenthesis
Delete the characters from the cursor up to and including the matching parenthesis, or next close parenthesis.

forward-delete-find
Delete the characters from the cursor up to and including the following occurence of the next character typed.

backward-delete-find
Delete the characters from the cursor up to and including the preceding occurence of the next character typed.

forward-delete-to
Delete the characters from the cursor up to, but not including, the following occurence of the next character typed.

backward-delete-to
Delete the characters from the cursor up to, but not including, the preceding occurence of the next character typed.

delete-refind
Repeat the last *-delete-find or *-delete-to action.

delete-invert-refind
Repeat the last *-delete-find or *-delete-to action, in the opposite direction.

copy-to-column
Copy the characters from the cursor up to the column that is specified by the repeat count, into the cut buffer.

copy-to-parenthesis
Copy the characters from the cursor up to and including the matching parenthesis, or next close parenthesis, into the cut buffer.

forward-copy-find
Copy the characters from the cursor up to and including the following occurence of the next character typed, into the cut buffer.

backward-copy-find
Copy the characters from the cursor up to and including the preceding occurence of the next character typed, into the cut buffer.

forward-copy-to
Copy the characters from the cursor up to, but not including, the following occurence of the next character typed, into the cut buffer.

backward-copy-to
Copy the characters from the cursor up to, but not including, the preceding occurence of the next character typed, into the cut buffer.

copy-refind
Repeat the last *-copy-find or *-copy-to action.

copy-invert-refind
: Repeat the last `*`-copy-find or `*`-copy-to action, in the opposite direction.

vi-mode
: Switch to `vi` mode from emacs mode.

emacs-mode
: Switch to `emacs` mode from `vi` mode.

vi-insert
: From `vi` command mode, switch to insert mode.

vi-overwrite
: From `vi` command mode, switch to overwrite mode.

vi-insert-at-bol
: From `vi` command mode, move the cursor to the start of the line and switch to insert mode.

vi-append-at-eol
: From `vi` command mode, move the cursor to the end of the line and switch to append mode.

vi-append
: From `vi` command mode, move the cursor one position right, and switch to insert mode.

vi-replace-char
: From `vi` command mode, replace the character under the cursor with the next character entered.

vi-forward-change-char
: From `vi` command mode, delete the next character then enter insert mode.

vi-backward-change-char
: From vi command mode, delete the preceding character then enter insert mode.

vi-forward-change-word
: From `vi` command mode, delete the next word then enter insert mode.

vi-backward-change-word
: From vi command mode, delete the preceding word then enter insert mode.

vi-change-rest-of-line
: From `vi` command mode, delete from the cursor to the end of the line, then enter insert mode.

vi-change-line
: From `vi` command mode, delete the current line, then enter insert mode.

vi-change-to-bol
: From `vi` command mode, delete all characters between the cursor and the beginning of the line, then enter insert mode.

vi-change-to-column
: From `vi` command mode, delete the characters from the cursor up to the column that is specified by the repeat count, then enter insert mode.

vi-change-to-parenthesis
: Delete the characters from the cursor up to and including the matching parenthesis, or next close parenthesis, then enter `vi` insert mode.

vi-forward-change-find

From `vi` command mode, delete the characters from the cursor up to and including the following occurence of the next character typed, then enter insert mode.

vi-backward-change-find

From vi command mode, delete the characters from the cursor up to and including the preceding occurence of the next character typed, then enter insert mode.

vi-forward-change-to

From `vi` command mode, delete the characters from the cursor up to, but not including, the following occurence of the next character typed, then enter insert mode.

vi-backward-change-to

From `vi` command mode, delete the characters from the cursor up to, but not including, the preceding occurence of the next character typed, then enter insert mode.

vi-change-refind

Repeat the last vi-*-change-find or vi-*-change-to action.

vi-change-invert-refind

Repeat the last vi-*-change-find or vi-*-change-to action, in the opposite direction.

vi-undo

In `vi` mode, undo the last editing operation.

vi-repeat-change

In `vi` command mode, repeat the last command that modified the line.

Default Key Bindings In emacs Mode

The following default key bindings, which can be overriden by the tecla configuration file, are designed to mimic most of the bindings of the unix `tcsh shell` shell, when it is in `emacs` editing mode.

This is the default editing mode of the tecla library.

Under UNIX the terminal driver sets a number of special keys for certain functions. The tecla library attempts to use the same key bindings to maintain consistency. The key sequences shown for the following 6 bindings are thus just examples of what they will probably be set to. If you have used the stty command to change these keys, then the default bindings should match.

`^C`	user-interrupt
`^\literal>`	abort
`^Z`	suspend
`^Q`	start-output
`^S`	stop-output
`^V`	literal-next

The cursor keys are refered to by name, as follows. This is necessary because different types of terminals generate different key sequences when their cursor keys are pressed.

right	cursor-right
left	cursor-left
up	up-history
down	down-history

The remaining bindings don't depend on the terminal setttings.

`^F`	cursor-right
`^B`	cursor-left
`M-i`	insert-mode
`^A`	beginning-of-line
`^E`	end-of-line
`^U`	delete-line
`^K`	kill-line
`M-f`	forward-word
`M-b`	backward-word
`^D`	del-char-or-list-or-eof
`^H`	backward-delete-char
`^?`	backward-delete-char
`M-d`	forward-delete-word
`M-^H`	backward-delete-word
`M-^?`	backward-delete-word
`M-u`	upcase-word
`M-l`	downcase-word
`M-c`	capitalize-word
`^R`	redisplay
`^L`	clear-screen
`^T`	transpose-chars
`^@`	set-mark

`^X^X`	exchange-point-and-mark
`^W`	kill-region
`M-w`	copy-region-as-kill
`^Y`	yank
`^P`	up-history
`^N`	down-history
`M-p`	history-search-backward
`M-n`	history-search-forward
`^I`	complete-word
`^X*`	expand-filename
`^X^F`	read-from-file
`^X^R`	read-init-files
`^Xg`	list-glob
`^Xh`	list-history
`M-<`	beginning-of-history
`M->`	end-of-history
`\n`	newline
`\r`	newline
`M-o`	repeat-history
`M-^V`	`vi`-mode
`M-0, M-1, ... M-9`	digit-argument (see below)

Note that `^I` is what the TAB key generates, and that `^@` can be generated not only by pressing the CONTROL key and the @ key simultaneously, but also by pressing the CONTROL key and the space bar at the same time.

Default Key Bindings in `vi` Mode

The following default key bindings are designed to mimic the `vi` style of editing as closely as possible. This means that very few editing functions are provided in the initial character input mode, editing functions instead being provided by the `vi` command mode. The `vi` command mode is entered whenever the ESCAPE character is pressed, or whenever a key sequence that starts with a meta character is entered. In addition to mimicing `vi`, `libtecla` provides bindings for tab completion, wild-card expansion of file names, and historical line recall.

To learn how to tell the tecla library to use `vi` mode instead of the default `emacs` editing mode, see the earlier section entitled The Tecla Configuration File.

Under UNIX the terminal driver sets a number of special keys for certain functions. The tecla library attempts to use the same key bindings to maintain consistency, binding them both in input mode and in command mode. The key sequences shown for the following 6 bindings are thus just examples of what they will probably be set to. If you have used the `stty` command to change these keys, then the default bindings should match.

`^C`	user-interrupt
`^\`	abort
`^Z`	suspend
`^Q`	start-output
`^S`	stop-output
`^V`	literal-next
`M-^C`	user-interrupt
`M-^\literal>`	abort
`M-^Z`	suspend
`M-^Q`	start-output
`M-^S`	stop-output

Note that above, most of the bindings are defined twice, once as a raw control code like `^C` and then a second time as a META character like `M-^C`. The former is the binding for `vi` input mode, whereas the latter is the binding for `vi` command mode. Once in command mode all key sequences that the user types that they don't explicitly start with an ESCAPE or a META key, have their first key secretly converted to a META character before the key sequence is looked up in the key binding table. Thus, once in command mode, when you type the letter i, for example, the tecla library actually looks up the binding for `M-i`.

The cursor keys are refered to by name, as follows. This is necessary because different types of terminals generate different key sequences when their cursor keys are pressed.

`right`	cursor-right
`left`	cursor-left
`up`	up-history
`down`	down-history

The cursor keys normally generate a key sequence that start with an ESCAPE character, so beware that using the arrow keys will put you into command mode (if you aren't already in command mode).

The following are the terminal-independent key bindings for `vi` input mode.

`^D`	list-or-eof
`^G`	list-glob
`^H`	backward-delete-char
`^I`	complete-word
`\r`	newline
`\n`	newline
`^L`	clear-screen
`^N`	down-history
`^P`	up-history
`^R`	redisplay
`^U`	backward-kill-line
`^W`	backward-delete-word
`^X*`	expand-filename
`^X^F`	read-from-file
`^X^R`	read-init-files
`^?`	backward-delete-char

The following are the key bindings that are defined in `vi` command mode, this being specified by them all starting with a META character. As mentioned above, once in command mode the initial meta character is optional. For example, you might enter command mode by typing ESCAPE, and then press 'H' twice to move the cursor two positions to the left. Both 'H' characters get quietly converted to `M-h` before being compared to the key binding table, the first one because ESCAPE followed by a character is always converted to the equivalent META character, and the second because command mode was already active.

`M-\`	cursor-right (META-space)
`M-$`	end-of-line
`M-*`	expand-filename
`M-+`	down-history

M--	up-history
M-<	beginning-of-history
M->	end-of-history
M-^	beginning-of-line
M-	repeat-find-char
M-,	invert-refind-char
M-|	goto-column
M-~	change-case
M-.	vi-repeat-change
M-%	find-parenthesis
M-a	vi-append
M-A	vi-append-at-eol
M-b	backward-word
M-B	backward-word
M-C	vi-change-rest-of-line
M-cb	vi-backward-change-word
M-cB	vi-backward-change-word
M-cc	vi-change-line
M-ce	vi-forward-change-word
M-cE	vi-forward-change-word
M-cw	vi-forward-change-word
M-cW	vi-forward-change-word
M-cF	vi-backward-change-find
M-cf	vi-forward-change-find
M-cT	vi-backward-change-to
M-ct	vi-forward-change-to
M-c;	vi-change-refind
M-c,	vi-change-invert-refind
M-ch	vi-backward-change-char

M-c^H	vi-backward-change-char
M-c^?	vi-backward-change-char
M-cl	vi-forward-change-char
M-c\	vi-forward-change-char (META-c-space)
M-c^	vi-change-to-bol
M-c0	vi-change-to-bol
M-c$	vi-change-rest-of-line
M-c|	vi-change-to-column
M-c%	vi-change-to-parenthesis
M-dh	backward-delete-char
M-d^H	backward-delete-char
M-d^?	backward-delete-char
M-dl	forward-delete-char
M-d	forward-delete-char (META-d-space)
M-dd	delete-line
M-db	backward-delete-word
M-dB	backward-delete-word
M-de	forward-delete-word
M-dE	forward-delete-word
M-dw	forward-delete-word
M-dW	forward-delete-word
M-dF	backward-delete-find
M-df	forward-delete-find
M-dT	backward-delete-to
M-dt	forward-delete-to
M-d;	delete-refind
M-d,	delete-invert-refind
M-d^	backward-kill-line
M-d0	backward-kill-line

M-d$	kill-line
M-D	kill-line
M-d|	delete-to-column
M-d%	delete-to-parenthesis
M-e	forward-word
M-E	forward-word
M-f	forward-find-char
M-F	backward-find-char
M--	up-history
M-h	cursor-left
M-H	beginning-of-history
M-i	vi-insert
M-I	vi-insert-at-bol
M-j	down-history
M-J	history-search-forward
M-k	up-history
M-K	history-search-backward
M-l	cursor-right
M-L	end-of-history
M-n	history-re-search-forward
M-N	history-re-search-backward
M-p	append-yank
M-P	yank
M-r	vi-replace-char
M-R	vi-overwrite
M-s	vi-forward-change-char
M-S	vi-change-line
M-t	forward-to-char
M-T	backward-to-char

`M-u`	vi-undo
`M-w`	forward-to-word
`M-W`	forward-to-word
`M-x`	forward-delete-char
`M-X`	backward-delete-char
`M-yh`	backward-copy-char
`M-y^H`	backward-copy-char
`M-y^?`	backward-copy-char
`M-yl`	forward-copy-char
`M-y\`	forward-copy-char (META-y-space)
`M-ye`	forward-copy-word
`M-yE`	forward-copy-word
`M-yw`	forward-copy-word
`M-yW`	forward-copy-word
`M-yb`	backward-copy-word
`M-yB`	backward-copy-word
`M-yf`	forward-copy-find
`M-yF`	backward-copy-find
`M-yt`	forward-copy-to
`M-yT`	backward-copy-to
`M-y;`	copy-refind
`M-y,`	copy-invert-refind
`M-y^`	copy-to-bol
`M-y0`	copy-to-bol
`M-y$`	copy-rest-of-line
`M-yy`	copy-line
`M-Y`	copy-line
`M-y|`	copy-to-column
`M-y%`	copy-to-parenthesis

`M-^E`	emacs-mode
`M-^H`	cursor-left
`M-^?`	cursor-left
`M-^L`	clear-screen
`M-^N`	down-history
`M-^P`	up-history
`M-^R`	redisplay
`M-^D`	list-or-eof
`M-^I`	complete-word
`M-\r`	newline
`M-\n`	newline
`M-^X^R`	read-init-files
`M-^Xh`	list-history
`M-0, M-1, ... M-9`	digit-argument (see below)

Note that `^I` is what the TAB key generates.

Entering Repeat Counts

Many of the key binding functions described previously, take an optional count, typed in before the target key sequence. This is interpreted as a repeat count by most bindings. A notable exception is the goto-column binding, which interprets the count as a column number.

By default you can specify this count argument by pressing the META key while typing in the numeric count. This relies on the digit-argument action being bound to 'META-0', 'META-1' etc. Once any one of these bindings has been activated, you can optionally take your finger off the META key to type in the rest of the number, since every numeric digit thereafter is treated as part of the number, unless it is preceded by the literal-next binding. As soon as a non-digit, or literal digit key is pressed the repeat count is terminated and either causes the just typed character to be added to the line that many times, or causes the next key binding function to be given that argument.

For example, in `emacs` mode, typing:

```
M-12a
```

causes the letter 'a' to be added to the line 12 times, whereas

```
M-4M-c
```

Capitalizes the next 4 words.

In vi command mode the meta modifier is automatically added to all characters typed in, so to enter a count in vi command-mode, just involves typing in the number, just as it does in the vi editor itself. So for example, in vi command mode, typing:

```
4w2x
```

moves the cursor four words to the right, then deletes two characters.

You can also bind digit-argument to other key sequences. If these end in a numeric digit, that digit gets appended to the current repeat count. If it doesn't end in a numeric digit, a new repeat count is started with a value of zero, and can be completed by typing in the number, after letting go of the key which triggered the digit-argument action.

文件

/usr/lib/libtecla.so	The tecla library
/usr/include/libtecla.h	The tecla header file
~/.teclarc	The personal tecla customization file

属性

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	library/libtecla
Interface Stability	Committed

另请参见

vi(1), cpl_complete_word(3TECLA), ef_expand_file(3TECLA), gl_get_line(3TECLA), gl_io_mode(3TECLA), libtecla(3LIB), pca_lookup_file(3TECLA), attributes(5)

引用名

term – conventional names for terminals

描述

Terminal names are maintained as part of the shell environment in the environment variable `TERM`. See `sh(1)`, `profile(4)`, and `environ(5)`. These names are used by certain commands (for example, `tabs`, `tput`, and `vi`) and certain functions (for example, see `curses(3CURSES)`).

Files under `/usr/share/lib/terminfo` are used to name terminals and describe their capabilities. These files are in the format described in `terminfo(4)`. Entries in `terminfo` source files consist of a number of comma-separated fields. To print a description of a terminal *term*, use the command `infocmp -I` *term*. See `infocmp(1M)`. White space after each comma is ignored. The first line of each terminal description in the `terminfo` database gives the names by which `terminfo` knows the terminal, separated by bar (`|`) characters. The first name given is the most common abbreviation for the terminal (this is the one to use to set the environment variable `TERMINFO` in `$HOME/.profile`; see `profile(4)`), the last name given should be a long name fully identifying the terminal, and all others are understood as synonyms for the terminal name. All names but the last should contain no blanks and must be unique in the first 14 characters; the last name may contain blanks for readability.

Terminal names (except for the last, verbose entry) should be chosen using the following conventions. The particular piece of hardware making up the terminal should have a root name chosen, for example, for the AT&T 4425 terminal, `att4425`. This name should not contain hyphens, except that synonyms may be chosen that do not conflict with other names. Up to 8 characters, chosen from the set `a` through `z` and `0` through `9`, make up a basic terminal name. Names should generally be based on original vendors rather than local distributors. A terminal acquired from one vendor should not have more than one distinct basic name. Terminal sub-models, operational modes that the hardware can be in, or user preferences should be indicated by appending a hyphen and an indicator of the mode. Thus, an AT&T 4425 terminal in 132 column mode is `att4425–w`. The following suffixes should be used where possible:

Suffix	Meaning	Example
–w	Wide mode (more than 80 columns)	att4425–w
–am	With auto. margins (usually default)	vt100–am
–nam	Without automatic margins	vt100–nam
–*n*	Number of lines on the screen	aaa–60
–na	No arrow keys (leave them in local)	c100–na
–np	Number of pages of memory	c100–4p
–rv	Reverse video	att4415–rv

To avoid conflicts with the naming conventions used in describing the different modes of a terminal (for example, `-w`), it is recommended that a terminal's root name not contain hyphens. Further, it is good practice to make all terminal names used in the `terminfo(4)` database unique. Terminal entries that are present only for inclusion in other entries via the `use=` facilities should have a '+' in their name, as in `4415+nl`.

Here are some of the known terminal names: (For a complete list, enter the command `ls -C /usr/share/lib/terminfo/?`).

2621,hp2621	Hewlett-Packard 2621 series
2631	Hewlett-Packard 2631 line printer
2631–c	Hewlett-Packard 2631 line printer, compressed mode
2631–e	Hewlett-Packard 2631 line printer, expanded mode
2640,hp2640	Hewlett-Packard 2640 series
2645,hp2645	Hewlett-Packard 2645 series
3270	IBM Model 3270
33,tty33	AT&T Teletype Model 33 KSR
35,tty35	AT&T Teletype Model 35 KSR
37,tty37	AT&T Teletype Model 37 KSR
4000a	Trendata 4000a
4014,tek4014	TEKTRONIX 4014
40,tty40	AT&T Teletype Dataspeed 40/2
43,tty43	AT&T Teletype Model 43 KSR
4410,5410	AT&T 4410/5410 in 80-column mode, version 2
4410–nfk,5410–nfk	AT&T 4410/5410 without function keys, version 1
4410–nsl,5410–nsl	AT&T 4410/5410 without pln defined
4410–w,5410–w	AT&T 4410/5410 in 132-column mode
4410v1,5410v1	AT&T 4410/5410 in 80-column mode, version 1
4410v1–w,5410v1–w	AT&T 4410/5410 in 132-column mode, version 1
4415,5420	AT&T 4415/5420 in 80-column mode
4415–nl,5420–nl	AT&T 4415/5420 without changing labels
4415–rv,5420–rv	AT&T 4415/5420 80 columns in reverse video

4415–rv–nl,5420–rv–nl	AT&T 4415/5420 reverse video without changing labels
4415–w,5420–w	AT&T 4415/5420 in 132-column mode
4415–w–nl,5420–w–nl	AT&T 4415/5420 in 132-column mode without changing labels
4415–w–rv,5420–w–rv	AT&T 4415/5420 132 columns in reverse video
4418,5418	AT&T 5418 in 80-column mode
4418–w,5418–w	AT&T 5418 in 132-column mode
4420	AT&T Teletype Model 4420
4424	AT&T Teletype Model 4424
4424-2	AT&T Teletype Model 4424 in display function group ii
4425,5425	AT&T 4425/5425
4425–fk,5425–fk	AT&T 4425/5425 without function keys
4425–nl,5425–nl	AT&T 4425/5425 without changing labels in 80-column mode
4425–w,5425–w	AT&T 4425/5425 in 132-column mode
4425–w–fk,5425–w–fk	AT&T 4425/5425 without function keys in 132-column mode
4425–nl–w,5425–nl–w	AT&T 4425/5425 without changing labels in 132-column mode
4426	AT&T Teletype Model 4426S
450	DASI 450 (same as Diablo 1620)
450–12	DASI 450 in 12-pitch mode
500,att500	AT&T-IS 500 terminal
510,510a	AT&T 510/510a in 80-column mode
513bct,att513	AT&T 513 bct terminal
5320	AT&T 5320 hardcopy terminal
5420_2	AT&T 5420 model 2 in 80-column mode
5420_2–w	AT&T 5420 model 2 in 132-column mode
5620,dmd	AT&T 5620 terminal 88 columns
5620–24,dmd–24	AT&T Teletype Model DMD 5620 in a 24x80 layer
5620–34,dmd–34	AT&T Teletype Model DMD 5620 in a 34x80 layer
610,610bct	AT&T 610 bct terminal in 80-column mode
610–w,610bct–w	AT&T 610 bct terminal in 132-column mode

630,630MTG	AT&T 630 Multi-Tasking Graphics terminal
7300,pc7300,unix_pc	AT&T UNIX PC Model 7300
735,ti	Texas Instruments TI735 and TI725
745	Texas Instruments TI745
dumb	generic name for terminals that lack reverse line-feed and other special escape sequences
hp	Hewlett-Packard (same as 2645)
lp	generic name for a line printer
pt505	AT&T Personal Terminal 505 (22 lines)
pt505–24	AT&T Personal Terminal 505 (24-line mode)
sync	generic name for synchronous Teletype Model 4540-compatible terminals

Commands whose behavior depends on the type of terminal should accept arguments of the form `-T`*term* where *term* is one of the names given above; if no such argument is present, such commands should obtain the terminal type from the environment variable `TERM`, which, in turn, should contain *term*.

文件

`/usr/share/lib/terminfo/?/*` compiled terminal description database

另请参见

`sh`(1), `stty`(1), `tabs`(1), `tput`(1), `vi`(1), `infocmp`(1M), `curses`(3CURSES), `profile`(4), `terminfo`(4), `environ`(5)

引用名

threads, pthreads – POSIX pthread 和 Solaris 线程概念

用法概要

POSIX

```
cc –mt [ flag... ] file... [ -lrt library... ]
```

```
#include <pthread.h>
```

Solaris

```
cc –mt [ flag... ] file... [ library... ]
```

```
#include <sched.h>
```

```
#include <thread.h>
```

描述

POSIX 和 Solaris 线程在 libc(3LIB) 中均具有各自的实现。这两个实现可交互操作，其功能相似并可在同一应用程序中使用。只有 POSIX 线程保证可完全移植到符合 POSIX 的其他环境。POSIX 和 Solaris 线程需要不同的源，包括文件和链接系统信息库。请参见“用法概要”部分。

相似之处

大部分 POSIX 和 Solaris 线程函数互相对应。POSIX 函数名称具有 “pthread” 前缀（信号名称除外）。相似的 POSIX 和 Solaris 函数的函数名称也具有相似的结尾。通常，相似的 POSIX 和 Solaris 函数的参数数目和用法相同。

不同之处

POSIX pthread 和 Solaris 线程的不同之处在于：

- POSIX 线程的可移植性更好。
- POSIX 线程根据可配置的属性对象为每个线程建立特性。
- POSIX pthread 实现了线程取消。
- POSIX pthread 强制执行调度算法。
- POSIX pthread 允许使用 fork(2) 调用的清理处理程序。
- 可暂停和继续执行 Solaris 线程。
- Solaris 线程实现守护进程线程，进程不会等待其终止。

函数比较

下表对 POSIX pthread 和 Solaris 线程函数进行了比较。当 POSIX pthread 或 Solaris 线程中没有可比较的接口时，列中会显示连字符 (–)。

与创建有关的函数

POSIX	Solaris
pthread_create()	thr_create()
pthread_attr_init()	–
pthread_attr_setdetachstate()	–
pthread_attr_getdetachstate()	–
pthread_attr_setinheritsched()	–
pthread_attr_getinheritsched()	–
pthread_attr_setschedparam()	–

pthread_attr_getschedparam()	–
pthread_attr_setschedpolicy()	–
pthread_attr_getschedpolicy()	–
pthread_attr_setscope()	–
pthread_attr_getscope()	–
pthread_attr_setstackaddr()	–
pthread_attr_getstackaddr()	–
pthread_attr_setstacksize()	–
pthread_attr_getstacksize()	–
pthread_attr_getguardsize()	–
pthread_attr_setguardsize()	–
pthread_attr_destroy()	–
–	thr_min_stack()

与退出有关的函数

POSIX	Solaris
pthread_exit()	thr_exit()
pthread_join()	thr_join()
pthread_detach()	–

与线程特定数据有关的函数

POSIX	Solaris
pthread_key_create()	thr_keycreate()
pthread_setspecific()	thr_setspecific()
pthread_getspecific()	thr_getspecific()
pthread_key_delete()	–

与信号有关的函数

POSIX	Solaris
pthread_sigmask()	thr_sigsetmask()
pthread_kill()	thr_kill()

与 ID 有关的函数

POSIX	Solaris
pthread_self()	thr_self()
pthread_equal()	–
–	thr_main()

与调度有关的函数

POSIX	Solaris
–	thr_yield()
–	thr_suspend()
–	thr_continue()
pthread_setconcurrency()	thr_setconcurrency()
pthread_getconcurrency()	thr_getconcurrency()
pthread_setschedparam()	thr_setprio()
pthread_setschedprio()	thr_setprio()
pthread_getschedparam()	thr_getprio()

与取消有关的函数

POSIX	Solaris
pthread_cancel()	–
pthread_setcancelstate()	–
pthread_setcanceltype()	–
pthread_testcancel()	–
pthread_cleanup_pop()	–
pthread_cleanup_push()	–

与互斥锁有关的函数

POSIX	Solaris
pthread_mutex_init()	mutex_init()
pthread_mutexattr_init()	–
pthread_mutexattr_setpshared()	–
pthread_mutexattr_getpshared()	–
pthread_mutexattr_setprotocol()	–

pthread_mutexattr_getprotocol()	–
pthread_mutexattr_setprioceiling()	–
pthread_mutexattr_getprioceiling()	–
pthread_mutexattr_settype()	–
pthread_mutexattr_gettype()	–
pthread_mutexattr_setrobust()	–
pthread_mutexattr_getrobust()	–
pthread_mutexattr_destroy()	–
pthread_mutex_setprioceiling()	–
pthread_mutex_getprioceiling()	–
pthread_mutex_lock()	mutex_lock()
pthread_mutex_trylock()	mutex_trylock()
pthread_mutex_unlock()	mutex_unlock()
pthread_mutex_destroy()	mutex_destroy()

与条件变量有关的函数

POSIX	Solaris
pthread_cond_init()	cond_init()
pthread_condattr_init()	–
pthread_condattr_setpshared()	–
pthread_condattr_getpshared()	–
pthread_condattr_destroy()	–
pthread_cond_wait()	cond_wait()
pthread_cond_timedwait()	cond_timedwait()
pthread_cond_signal()	cond_signal()
pthread_cond_broadcast()	cond_broadcast()
pthread_cond_destroy()	cond_destroy()

与读取器/写入器锁定有关的函数

POSIX	Solaris
pthread_rwlock_init()	rwlock_init()

pthread_rwlock_rdlock()	rw_rdlock()
pthread_rwlock_tryrdlock()	rw_tryrdlock()
pthread_rwlock_wrlock()	rw_wrlock()
pthread_rwlock_trywrlock()	rw_trywrlock()
pthread_rwlock_unlock()	rw_unlock()
pthread_rwlock_destroy()	rwlock_destroy()
pthread_rwlockattr_init()	–
pthread_rwlockattr_destroy()	–
pthread_rwlockattr_getpshared()	–
pthread_rwlockattr_setpshared()	–

与信号有关的函数

POSIX	Solaris
sem_init()	sema_init()
sem_open()	–
sem_close()	–
sem_wait()	sema_wait()
sem_trywait()	sema_trywait()
sem_post()	sema_post()
sem_getvalue()	–
sem_unlink()	–
sem_destroy()	sema_destroy()

与fork( )清理有关的函数

POSIX	Solaris
pthread_atfork()	–

与限制有关的函数

POSIX	Solaris
pthread_once()	–

与调试有关的函数

POSIX	Solaris

–	thr_stksegment()

锁定

同步

多线程行为是异步的，因此针对并发和并行处理进行了优化。由于线程（始终来自同一进程，有时来自多个进程）相互之间共享全局数据，因此无法保证它们对任何时间点的共享数据具有独占访问权限。要获得对共享数据的独占访问权限，需要在线程之间进行同步。POSIX 和 Solaris 均实现了四种同步机制：互斥锁、条件变量、读取器/写入器锁定（**优化的频繁读、少量写互斥锁**）以及信号。

同步多个线程会大大降低线程并发性。同步的粒度越粗，即锁定的代码块越大，并发性就越小。

MT fork()

如果线程程序调用 fork(2)，它会隐式调用只复制调用线程的 fork1(2)。如果整个进程中还有任何待处理的互斥锁，则应用程序应在调用 fork() 之前调用 pthread_atfork(3C) 等待并获取这些互斥锁。

调度

POSIX 线程

Solaris 支持以下三种 POSIX 调度策略：

SCHED_OTHER　传统分时调度策略。该策略基于分时 (timesharing, TS) 调度类。

SCHED_FIFO　先入先出调度策略。如果不被更高优先级抢占，根据此策略调度的线程将会继续执行直至完成。此类线程属于实时 (real-time, RT) 调度类。调用进程必须在其有效集中声明 {PRIV_PROC_PRIOCNTL} 特权。

SCHED_RR　循环调度策略。如果不被更高优先级抢占，根据此策略调度的线程将在系统确定的时段内执行。此类线程属于实时 (real-time, RT) 调度类，调用进程必须在其有效集中声明 {PRIV_PROC_PRIOCNTL} 特权。

除了上述 POSIX 指定的调度策略以外，Solaris 还支持以下调度策略：

SCHED_IA　根据 priocntl(2) 中所述的交互式类 (IA) 策略调度线程。

SCHED_FSS　根据 priocntl(2) 中所述的公平份额类 (FSS) 策略调度线程。

SCHED_FX　根据 priocntl(2) 中所述的固定优先级类 (FSS) 策略调度线程。

Solaris 线程

唯一受支持的调度策略是 SCHED_OTHER，即基于 TS 调度类的分时。

错误

在多线程应用程序中，当有其他线程调用 forkall(2) 时，将会从阻止的系统调用返回 EINTR。

用法

-mt 编译器选项

-mt 编译器选项会编译和链接多线程代码。它使用 –D_REENTRANT 编译源文件并适当扩大支持系统信息库集。

属性

有关下列属性的说明，请参见 attributes(5)：

属性类型	属性值
MT 级别	MT-Safe、Fork 1-Safe

另请参见

crle(1)、fork(2)、priocntl(2)、libpthread(3LIB)、librt(3LIB)、libthread(3LIB)、pthread_atfo

《链接程序和库指南》

引用名

trusted_extensions – Solaris Trusted Extensions

描述

Solaris Trusted Extensions 软件是 Solaris 操作系统 (Solaris OS) 的特定配置。Solaris Trusted Extensions（下称 Trusted Extensions）为本地对象和进程、桌面和窗口系统、区域和文件系统以及网络通信提供标签。这些标签用于实现多级别安全性 (MLS) 策略，以便根据标签关系来限制信息流。与基于所有权的自主访问控制 (DAC) 相反，Trusted Extensions 实施的 MLS 策略是强制访问控制 (MAC) 的一个示例。

缺省情况下，Trusted Extensions 软件是禁用的。该软件可以通过 FMRI 标识的 `labeld(1M)` 服务启用和禁用（但不能配置）：

```
svc:/system/labeld:default
```

有关使用前必须执行的 Trusted Extensions 软件配置的信息，请参见下文列出的管理员指南。在启用或禁用 `labeld` 后，必须重新引导系统才能激活或取消激活 Trusted Extensions 软件。

另请参见

`labeld(1M)`、`label_encodings(4)`、`labels(5)`

《Trusted Extensions 配置和管理》

《Trusted Extensions 用户指南》

引用名

vgrindefs – vgrind's language definition data base

用法概要

```
/usr/lib/vgrindefs
```

描述

vgrindefs contains all language definitions for vgrind(1). Capabilities in vgrindefs are of two types: Boolean capabilities which indicate that the language has some particular feature and string capabilities which give a regular expression or keyword list. Entries may continue onto multiple lines by giving a \ as the last character of a line. Lines starting with # are comments.

Capabilities

The following table names and describes each capability.

Name	Type	Description
ab	str	Regular expression for the start of an alternate form comment
ae	str	Regular expression for the end of an alternate form comment
bb	str	Regular expression for the start of a block
be	str	Regular expression for the end of a lexical block
cb	str	Regular expression for the start of a comment
ce	str	Regular expression for the end of a comment
id	str	String giving characters other than letters and digits that may legally occur in identifiers (default '_')
kw	str	A list of keywords separated by spaces
lb	str	Regular expression for the start of a character constant
le	str	Regular expression for the end of a character constant
oc	bool	Present means upper and lower case are equivalent
pb	str	Regular expression for start of a procedure
pl	bool	Procedure definitions are constrained to the lexical level matched by the 'px' capability
px	str	A match for this regular expression indicates that procedure definitions may occur at the next lexical level. Useful for lisp-like languages in which procedure definitions occur as subexpressions of defuns.
sb	str	Regular expression for the start of a string
se	str	Regular expression for the end of a string
tc	str	Use the named entry as a continuation of this one
tl	bool	Present means procedures are only defined at the top lexical level

Regular Expressions

`vgrindefs` uses regular expressions similar to those of `ex(1)` and `lex(1)`. The characters ‘^', ‘$', ‘:', and ‘\' are reserved characters and must be ‘quoted' with a preceding \ if they are to be included as normal characters. The metasymbols and their meanings are:

`$` The end of a line

`^` The beginning of a line

`\d` A delimiter (space, tab, newline, start of line)

`\a` Matches any string of symbols (like ‘.*' in lex)

`\p` Matches any identifier. In a procedure definition (the ‘pb' capability) the string that matches this symbol is used as the procedure name.

`()` Grouping

`|` Alternation

`?` Last item is optional

`\e` Preceding any string means that the string will not match an input string if the input string is preceded by an escape character (\). This is typically used for languages (like C) that can include the string delimiter in a string by escaping it.

Unlike other regular expressions in the system, these match words and not characters. Hence something like ‘(tramp|steamer)flies?' would match ‘tramp', ‘steamer', ‘trampflies', or ‘steamerflies'. Contrary to some forms of regular expressions, `vgrindef` alternation binds very tightly. Grouping parentheses are likely to be necessary in expressions involving alternation.

Keyword List

The keyword list is just a list of keywords in the language separated by spaces. If the ‘oc' boolean is specified, indicating that upper and lower case are equivalent, then all the keywords should be specified in lower case.

示例

示例 1 A sample program.

The following entry, which describes the C language, is typical of a language entry.

```
C|c|the C programming language:\
    :pb=^\d?*?\d?\p\d?(\a?\)(\d|{):bb={:be=}:cb=/*:ce=*/:sb=":se=\e":\
    :le=\e’:tl:\
    :kw=asm auto break case char continue default do double else enum\
    extern float for fortran goto if int long register return short\
    sizeof static struct switch typedef union unsigned void while #define\
    #else #endif #if #ifdef #ifndef #include #undef # define endif\
    ifdef ifndef include undef defined:
```

Note that the first field is just the language name (and any variants of it). Thus the C language could be specified to `vgrind(1)` as ‘c' or ‘C'.

文件

`/usr/lib/vgrindefs` file containing vgrind descriptions

另请参见

`ex`(1), `lex`(1), `troff`(1), `vgrind`(1)

引用名

wbem – Web-Based Enterprise Management

描述

Web-Based Enterprise Management (WBEM) is a set of management and Internet-related technologies intended to unify the management of enterprise computing environments. Developed by the Distributed Management Task Force (DMTF), WBEM enables organizations to deliver an integrated set of standards-based management tools that support and promote World Wide Web technology. The DMTF has developed a set of standards that make up WBEM. This set of standards includes:

Common Information Model (CIM)

CIM is an object-oriented data model that describes the overall management of information in an enterprise network environment. CIM consists of a CIM specification and a CIM schema:

CIM Specification
: Consists of the language and methodology that describes management data.

CIM Schema
: Provides actual model descriptions of systems, applications, large area networks, and devices. The CIM Schema enables applications from different developers on different platforms to describe management data in a standard format. As a result, a variety of management applications can share this information.

CIM Operations Over HyperText Transport Protocol (HTTP) 1.1 is a transport mechanism that maps CIM operations to HTTP to allow implementations of CIM to interoperate in an open, standardized manner.

CIM Operations Over HTTP 1.1 uses eXtensible Markup Language (XML), which is a markup language that represents management information in textual form.

In addition to the XML representation, CIM information is also represented textually by the managed object format (MOF). These MOF representations are typically stored as text files that developers compile into a CIM Object Manager.

WBEM Tools and Services

Tools and services that enable developers to create and Services management applications and instrumentation that manage heterogeneous computer environments include:

- Solaris WBEM Services
- Solaris WBEM Software Development Kit (SDK)

Solaris WBEM Services

These services consist of a set of value-added Services components. These services make it easier for developers to create management applications that run in the Solaris operating environment. They also make the Solaris operating environment easier to manage. Solaris WBEM Services consists of:

- CIM Object Manager, CIM Repository, and MOF Compiler

- CIM and Solaris Schema, which is an extension schema of CIM. CIM and Solaris Schema is a collection of CIM classes that describe managed elements in the Solaris operating environment. These classes are available from the CIM Object Manager at start up.
- Solaris Providers, which are programs that communicate information between the Solaris operating environment and the CIM Object Manager (providers get and set dynamic information about managed elements, acting as an intermediary between the CIM Object Manager and the managed elements).

  Solaris software providers have been developed for a variety of areas: users, roles, file systems, and network configuration, for example. A remote provider is also available to distribute agents away from the CIM Object Manager when required. Because of the incremental development capabilities of the WBEM instrumentation framework, developers can progressively and consistently add more providers for additional Solaris software services.
- SNMP Adapter for WBEM, which enables Simple Network Management Protocol (SNMP) management applications to access system management information that is provided by Solaris WBEM Services. Used with the Solstice Enterprise Agent (SEA) Master Agent `snmpdx(1M)`, the SNMP Adapter for WBEM maps SNMP requests into equivalent WBEM Common Information Model (CIM) properties or instances.

  The SNMP Adapter for WBEM also remaps the response from the CIM Object Manager into an SNMP response, which is returned to the management application.

  A mapping file contains the corresponding Object Identifier (OID), class name, property name, and Abstract Syntax Notation One (ASN.1) type for each object. Developers can create their own mapping files.
- SNMP Provider, which enables WBEM services to deliver SNMP information.

Solaris WBEM SDK

The Solaris WBEM SDK is a set of application programming interfaces (APIs) that contain the components necessary to write management applications. These applications communicate with WBEM-enabled management devices by using XML and HTTP communication standards.

Solaris WBEM applications request information or services from the Common Information Model (CIM) Object Manager through the WBEM APIs. These APIs represent CIM objects as Java classes. The APIs are used to describe managed objects and to retrieve information about managed objects in a system environment. The advantage of modeling managed resources by using CIM is that those objects can be shared across any system that is CIM-compliant.

For more information on the Solaris WBEM SDK, see the Solaris WBEM Developer's Guide. The Solaris WBEM API documentation is available in Javadoc format with the Solaris OS installation at `/usr/sadm/lib/wbem/doc/index.html`.

Compatibility of Solaris WBEM Services with Existing Protocols

Adapters and converters enable Solaris WBEM Services of Solaris to work compatibly with existing protocols by mapping WBEM information to these protocols. One such protocol is Simple Network Management Protocol (SNMP).

Legacy management applications can administer WBEM-enabled software in the Solaris operating environment. Developers can write agents or providers that convert information from these protocols to WBEM, and they can write adapters that convert WBEM information into these protocols.

属性

See `attributes(5)` for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	SPARC and x86
Architecture	SUNWwbapi, SUNWwbco, SUNWwbco, SUNWwbdev, SUNWwbdoc, SUNWwbpro
CSI	Enabled

另请参见

`appletviewer`(1), `cimworkshop`(1M), `init.wbem`(1M), `mofcomp`(1M), `mofreg`(1M), `snmpdx`(1M), `wbemadmin`(1M), `wbemconfig`(1M), `wbemlogviewer`(1M), `attributes`(5)

引用名

zones – Solaris 应用程序容器

描述

Solaris 中的区域工具提供了用于运行应用程序的隔离环境。在区域中运行的进程可以免受系统中其他活动的监视或干扰。对其他进程、网络接口、文件系统、设备和进程间通信设备的访问受到限制，防止不同区域中的进程之间相互交互。

在区域中可用的权限受到限制，以防止操作造成系统范围的影响。请参见 privileges(5)。

可以使用 zoneadm(1M) 和 zonecfg(1M) 实用程序配置和管理区域。可以为区域指定配置详细信息，在区域中安装文件系统内容（包括软件包）以及管理区域的运行时状态。可以使用 zlogin(1) 在活动区域中运行命令。您不需要通过基于网络的登录服务器（例如 in.rlogind(1M) 或 sshd(1M)）进行登录，即可执行此操作。

区域的自动引导可以通过 FMRI 标识的区域服务来启用和禁用：

svc:/system/zones:default

请参见 zoneadm(1M)。请注意，区域有一个 autoboot 属性，该属性可以设置为 true（始终自动引导）。不过，如果禁用了区域服务，将不发生自动引导，而无论指定区域的 autoboot 属性设置如何。请参见 zonecfg(1M)。

每个活动区域使用一个字母数字名称和一个数字 ID 来标识。字母数字名称是使用 zonecfg(1M) 实用程序配置的。数字 ID 是在引导区域时自动分配的。zonename(1) 实用程序可以报告当前区域名称，而 zoneadm(1M) 实用程序可用于报告已配置区域的名称和 ID。

区域可以处于多种状态之一：

CONFIGURED
指示区域配置已完全指定并且已提交到稳定的存储设备。

INCOMPLETE
指示区域正在安装或卸载过程中，或者已在此类转换期间中断。

INSTALLED
指示区域配置已在系统上实例化：软件包已安装在区域的根路径下。

READY
指示区域的“虚拟平台”已建立。例如，文件系统已挂载，设备已配置，但尚未启动任何与区域关联的进程。

RUNNING
指示正在运行与区域应用程序环境关联的用户进程。

SHUTTING_DOWN
DOWN
指示正在停止区域。如果区域无法退出应用程序环境状态（例如已挂载的文件系统），或者无法销毁虚拟平台的某个部分，则区域会在其中一个状态中挂起。在这些情况下，需要操作者进行干预。

UNAVAILABLE

指示已安装区域但无法引导。出现以下情况时区域会进入无法使用的状态：区域的存储不可用而 svc:/system/zones:default 正在联机，或区域尝试引导；成功提取归档文件之后基于归档文件的安装失败；区域的软件与全局区域的软件不兼容（如不正确的强制附加之后）。

进程访问限制

在区域（不包括全局区域）中运行的进程对其他进程拥有受限的访问权限。只有相同区域中的进程才能通过 /proc（请参见 proc(4)）或者通过接受进程 ID 的系统调用接口（例如 kill(2) 和 priocntl(2)）来查看。尝试访问存在于其他区域（包括全局区域）中的进程时将失败，并且出现与当指定进程不存在时发布的错误代码相同的错误代码。

权限限制

在非全局区域中运行的进程被限制为拥有一部分权限，从而防止某个区域能够执行可能会影响其他区域的操作。这组权限限制了区域中的特权用户（例如超级用户或 root 用户）的能力。可以使用 ppriv(1) 实用程序显示某个区域中可用的权限列表。有关权限的详细信息，请参见 privileges(5)。

设备限制

对区域中可用的一组设备进行了限制，以防止某个区域中的进程干扰在其他区域中运行的进程。例如，某个区域中的进程应该无法使用 /dev/kmem 修改内核内存或修改根磁盘的内容。因此，缺省情况下，只有一些被认为可安全在区域中使用的 pseudo 设备可用。在特定区域内，可使用 zonecfg(1M) 实用程序使其他设备变得可用。

设备和权限限制对于可在非全局区域中运行的实用程序产生重大影响。例如，eeprom(1M)、prtdiag(1M) 和 prtconf(1M) 实用程序无法在区域中使用，因为它们依赖于通常不可用的设备。

标记

在最初创建区域时，会向该区域分配一个标记。标记区域中的软件与在全局区域中发现的软件不匹配。软件可以包含按照不同的方式配置或布局的 Solaris 软件，也可以包含非 Solaris 软件。这种特定的软件集合称为“标记”（请参见 brands(5)）。在安装后，除非首先卸载区域，否则无法更改区域的标记。

文件系统

每个区域都有自己的文件系统分层结构部分，根目录称为区域 root。区域中的进程只能访问分层结构的该部分中的文件，即位于区域根路径下的文件。这样可以防止一个区域中的进程损坏或检查与另一个区域关联的文件系统数据。chroot(1M) 实用程序可以在区域中使用，但只能将进程限制为区域中可访问的根路径。

为了保留文件系统空间，可以使用 lofs(7FS) 文件系统的只读选项，将文件系统部分挂载到一个或多个区域中。这样可以在多个区域中共享相同的文件系统数据，同时保留区域提供的安全保证。

在区域中建立的 NFS 和 autofs 挂载是该区域的本地挂载；它们不能从其他区域（包括全局区域）中访问。在停止或重新引导区域时，将删除挂载。

委托给某个区域的 ZFS 数据集可以在该区域中进行管理。在委托数据集中可以创建子数据集。在委托数据集中创建的数据集是自我委托的。可以销毁除顶级委托数据集之外的委托数据集。大多数属性都可以在委托数据集上设置，但并不是所有属性都可以。有关详细信息，请参见 zfs(1M)。

每个区域都有一个顶级委托数据集，该数据集又包含 ROOT，并且可能包含其他数据集，例如 .../export 和 .../export/home。存在于 ROOT 数据集下方的数据集构成了区域的引导环境。只应使用 zoneadm(1M) 或 beadm(1M) 命令来创建或销毁引导环境数据集。

联网

区域对于 TCP、UDP 和 SCTP 应用程序使用自己的端口号范围，通常有一个或多个单独的 IP 地址（不过，某些 Trusted Extensions 配置在区域之间共享 IP 地址）。

对于 IP 层（IP 路由、ARP、IPsec、IP 过滤器等），区域可以与全局区域共享配置和状态（共享 IP 区域），也可以具有独特的 IP 层配置和状态（独占 IP 区域）。

如果区域将连接到相同的数据链接，即与全局区域相同的 IP 子网，该区域适合使用共享 IP 实例。

如果区域必须在网络上的 IP 层中隔离（例如连接到与全局区域和其他非全局区域不同的 VLAN 或不同的 LAN），出于隔离原因，区域应该具有其独占 IP。

共享 IP 区域无法在网络中执行某些操作（例如更改 IP 地址或者发送欺骗性 IP 或 Ethernet 包），而与连接到相同网络接口的单独主机在网络中执行的功能相比，独占 IP 区域具有或多或少的相同功能。特别是，这种区域中的超级用户可以更改 IP 地址和发送欺骗性 ARP 包。

在 zonecfg(1M) 中，会向共享 IP 的区域分配一个或多个网络接口名称和 IP 地址。网络接口名称还必须在全局区域中配置。

在 zonecfg(1M) 中，会向独占 IP 的区域分配一个或多个网络接口名称。网络接口名称必须专门分配给该区域，即名称不能分配给其他某个正在运行的区域，也无法由全局区域使用。

以 DHCP 客户机、IPsec 和 IP 过滤器形式提供的完整 IP 级别功能在独占 IP 的区域中可用，而在共享 IP 的区域中不可用。

主机标识符

区域能够模拟 32 位主机标识符，该标识符可以通过 zonecfg(1M) 针对系统整合目的进行配置。如果区域模拟主机标识符，在区域上下文中执行的 hostid(1) 和 sysdef(1M) 等命令以及 sysinfo(2) 和 gethostid(3C) 等 C 接口将显示或返回区域的模拟主机标识符，而不是主机计算机的标识符。

属性

有关下列属性的说明，请参见 attributes(5)：

属性类型	属性值
可用性	system/core-os

另请参见

hostid(1)、zlogin(1)、zonename(1)、beadm(1M)、in.rlogind(1M)、sshd(1M)、sysdef(1M)、zfs(1