AIX 5L V5.3

命令参考大全，卷 2，d - h

S151-0036-03

AIX 5L V5.3

命令参考大全，卷 2，d - h

S151-0036-03

注意

在使用本资料及其支持的产品之前，请阅读第 649 页的『声明』中的信息。

第四版（2006 年 7 月）

本版本适用于 AIX 5L V5.3 及其所有后续发行版，直到在新版本中另有声明为止。

在本出版物的后面提供了读者意见表。如果该表已被删除，请将意见寄往 IBM 中国公司上海分公司，汉化部；中国上海市淮海中路 333 号瑞安广场 10 楼；邮政编码：200021。要通过电子形式发送意见，请使用以下商业互联网地址：ctscrcf@cn.ibm.com。我们可以使用您提供的任何信息，而无需对您承担任何责任。

目录

关于本书

本书向最终用户提供有关 AIX® 操作系统命令的完整详细信息。按字母顺序和类别列出命令并给出命令与其可用标志的完整描述。每个命令列表包含示例（如果适用）。本卷包含以字母 d 到 h 开头的 AIX 命令。本出版物还可在随操作系统一起提供的文档 CD 上获得。

如何使用本书

命令是执行某项操作或者运行某个程序的请求。通过使用命令来告诉操作系统您想执行什么任务。输入命令后，这些命令通过命令解释器（也称为 shell）进行解释，然后执行该任务。

有些命令可以仅输入一个单词来完成。也可以将命令组合起来使得一个命令的输出成为另一个命令的输入。这也称为流水线技术。

标志进一步定义了命令的操作。标志是和命令名一起在命令行中使用的修饰符，它通常位于连字符之后。

这些命令也可编成一组并存储在一个文件当中。这些被称作 shell 过程或 shell 脚本。您可以运行包含了这些命令的文件，而不是单个执行这些命令。

某些命令可以通过使用基于 Web 的系统管理器应用程序或系统管理接口工具（SMIT）来构造。

突出显示

本书中使用以下突出显示约定:

粗体	标识命令、子例程、关键字、文件、结构、目录以及其他名字由系统预定义的项。也识别图形对象例如用户选择的按钮、标签和图标。
斜体	识别实际名称或值将由用户提供的参数。
等宽字体	识别特定数据值的示例，文本与您可能看到的显示相似的示例、程序代码部分与您可能作为程序员写的相似的示例、系统消息或者您将实际输入的信息。

格式

每个命令可能包含以下任一部分:

用途	每个命令主要功能的描述。
语法	语法语句显示了命令行选项。
描述	命令的讨论描述了命令功能和用法的详细信息。
标志	带有解释标志如何修改命令操作的命令行标志和相关变量列表。
参数	命令行参数和它们的描述的列表。
子命令	解释这些命令用法的子命令（用于交互式命令）的列表。
退出状态	命令返回的退出值的描述。
安全性	指定运行命令需要的任何许可权。
示例	指定如何使用该命令的示例。
文件	由该命令使用的文件的列表。
相关信息	本书中相关命令和其他书中的相关讨论的列表。

阅读语法语句

语法语句是表现命令语法和组成的符号（例如方括号（[]）、花括号（{}）和竖条（|））的方法。以下是语法说明 **unget** 命令的示例:

unget [-r*SID*] [**-s**] [**-n**] *File* ...

以下约定用于命令语法语句当中:

- 必须在命令行中逐字输入的项使用**粗体**。这些项包含命令名、标志以及文字字符。
- 必须由一个名称替换的变量所表示的项以*斜体*显示。这些项包括标志后的参数和命令读取的参数，例如*文件*和*目录*。
- 方括号中的参数是可选的参数。
- 花括号中的参数是必需的参数。
- 既不在方括号也不在花括号中的参数是必需的参数。
- 竖条说明您仅可选择一个参数。例如，[a | b] 表明您*可以*选择 a、b 或不作任何选择。与此相似，{ a | b } 表明您*必须*选择 a 或者 b。
- 省略号（...）表示该参数可以在命令行上重复。
- 短划线（-）表示标准输入。

可安装软件包的列表

要列出单个命令的可安装软件包（文件集），使用带有 **-w** 标志的 **lslpp** 命令。例如，要列出拥有命令 **installp** 的文件集，请输入:

```
lslpp -w /usr/sbin/installp
```

输出类似于以下所显示的内容:

```
File                            Fileset                 Type
-----------------------------------------------------------------------
/usr/sbin/installp                bos.rte.install           File
```

要列出具有包含 installp 的所有文件名的文件集，请输入:

```
lslpp -w "*installp*"
```

输出类似于以下所显示的内容:

```
File                            Fileset                 Type
-----------------------------------------------------------------------
/usr/sbin/installp                bos.rte.install           File
/usr/clvm/sbin/linstallpv         prpq.clvm                 File
/usr/lpp/bos.sysmgt/nim/methods/c_installp
                                  bos.sysmgt.nim.client     File
```

在后台运行命令

如果您要运行一个需要长时间处理的命令，您可以指定该命令在后台运行。后台进程是运行较慢的程序的一个非常有用的方法。要在后台运行某个命令，应在该命令的结尾使用 & 运算符:

```
命令 &
```

一旦进程在后台运行，您可以继续工作并在系统中输入其他的命令。

有时，您可能想在某个特定时间或者特定的日期运行某个命令。使用 **cron** 守护程序，您可以调度命令使其自动运行。或者使用 **at** 和 **batch** 命令，您可以稍后运行或者当系统装入级别许可时运行命令。

输入命令

当使用操作系统时，通常在命令行的 shell 提示符后输入命令。shell 提示符可以不同。在以下示例中，$ 便是该提示符。

要显示当前目录内容的列表，输入 ls 然后按回车键:

```
$ ls
```

当您输入一个命令并运行时，操作系统将不会显示 shell 提示符。当命令执行完毕，系统再次显示提示符。这说明您可以输入另外一条命令。

输入操作系统命令的一般格式是:

命令 标志 参数

标志更改命令工作的方式。很多命令有若干标志。例如，如果您在 **ls** 命令之后输入 **-l**（长型）标志，系统将提供关于当前目录内容的附加信息。下边的示例显示如何在 **ls** 命令之后使用 **-l** 标志:

```
$ ls -l
```

参数由字符串组成，接在命令和标志之后。它指定了数据如文件或目录的名称或者数值。在下面的示例中，目录名 **/usr/bin** 就是一个参数:

```
$ ls -l /usr/bin
```

输入命令时，重要的是记住以下几点:

- 命令通常用小写输入。
- 标志通常有一个前缀 -（减号）。
- 命令行中可输入多条命令，它们之间用 **;**（分号）隔开。
- 长命令可以用 \（反斜杠）在下一行继续输入。反斜杠位于第一行的最后。以下的示例说明了反斜杠的位置:

```
$ cat /usr/ust/mydir/mydata > \
/usr/usts/yourdir/yourdata
```

当输入某些命令时，shell 提示符将会发生更改。由于一些命令是实际的程序（例如 **telnet** 命令），在执行这些命令时提示符将会更改。任何在程序中发生的命令都是子命令。当您退出程序后，提示符将会返回到 shell 提示。

操作系统可以运行不同的 shell（例如 Bourne、C 或 Korn），并且您输入的命令将会由 shell 来解释。因此，您必须知道您用的是什么 shell 以便您能以正确的格式输入命令。

停止命令

如果您输入一条命令然后要停止命令的运行，则可中止命令的运行。要停止命令的运行，按下中断程序键（通常是 Ctrl-C 或者 Alt-Pause）。当进程停止后，将返回 shell 提示并且您可以继续输入其他命令。

ISO 9000

在此产品的开发和制造中使用了 ISO 9000 注册质量体系。

Single UNIX Specification 的 32 位和 64 位支持

从版本 5.2 开始，此操作系统被设计用来支持 The Open Group 的 Single UNIX Specification V3（UNIX 03），提供了对基于 UNIX 操作系统的可移植性。添加了许多新接口或增强了某些当前接口以满足此规范，使版本 5.2 对应用程序具有更强的开放性和可移植性，同时保留了与先前 AIX 发行版的兼容性。
要确定开发可移植到 UNIX 03 的应用程序的正确方法，可能需要参考 The Open Group 的 UNIX 03 规范，可以通过在线访问或从 http://www.unix.org/ 下载此规范。

相关信息

以下书籍包含命令的相关信息:

- 《*AIX 5L V5.3 命令参考大全，卷 1*》
- 《*AIX 5L V5.3 命令参考大全，卷 3*》
- 《*AIX 5L V5.3 命令参考大全，卷 4*》
- 《*AIX 5L V5.3 命令参考大全，卷 5*》
- 《*AIX 5L V5.3 命令参考大全，卷 6*》
- *AIX 5L Version 5.3 Files Reference*
- 《*打印机和打印指南*》
- 《*安装与迁移*》
- 《*AIX 5L V5.3 分区环境中的 AIX 安装*》
- *AIX 5L Version 5.3 Network Information Services (NIS and NIS+) Guide*
- 《*性能管理*》
- *AIX 5L Version 5.3 Performance Tools Guide and Reference*
- 《*安全性*》
- 《*网络与通信管理*》
- 《*操作系统与设备管理*》
- *AIX 5L Version 5.3 Technical Reference: Base Operating System and Extensions Volume 1*
- *AIX 5L Version 5.3 Technical Reference: Base Operating System and Extensions Volume 2*
- *AIX 5L Version 5.3 Technical Reference: Communications Volume 1*
- *AIX 5L Version 5.3 Technical Reference: Communications Volume 2*
- *AIX 5L Version 5.3 Technical Reference: Kernel and Subsystems Volume 1*
- *AIX 5L Version 5.3 Technical Reference: Kernel and Subsystems Volume 2*
- 《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》
- *Performance Toolbox Version 2 and 3 for AIX: Guide and Reference*

按字母排序的命令清单

dacinet 命令

用途

管理 CAPP/EAL4+ 配置中 TCP 端口的安全性。

语法

dacinet aclflush

dacinet aclclear *Service* | *Port*

dacinet acladd *Service* | [**-**] *addr* [*/prefix_length*] [**u:***user* | *uid* | **g:***group* | *gid*]

dacinet acldel *Service* | [**-**] *addr* [*/prefix_length*] [**u:***user* | *uid* | **g:***group* | *gid*]

dacinet aclls *Service* | *Port*

dacinet setpriv *Service* | *Port*

dacinet unsetpriv *Service* | *Port*

dacinet lspriv

描述

dacinet 命令用来管理 TCP 端口的安全性。请参阅『子命令』一节以获得 **dacinet** 各种功能的详细信息。

子命令

acladd 在保存由 DACinet 使用的访问控制列表的内核表中添加 ACL 条目。**acladd** 子命令的参数的语法为:

[**-**]*addr*[**/***length*][**u**:*user*|*uid*| **g**:*group*|*gid*] 参数定义如下:

addr DNS 主机名或者某个 IP v4/v6 的地址。地址前的 "-" 表示该 ACL 条目是用来拒绝访问而不是允许访问。

length 表示 *addr* 将会用来作为一个网络地址而不是主机地址，其中位于第一的 *length* 位取自于 *addr*。

u:*user*|*uid*
可选的用户标识符。如果没有指定 *uid*，则指定主机或者子网上的所有的用户都将获得访问该服务的许可。如果提供，则只有指定的用户获得访问许可。

g:*group*|*gid*
可选的分组标识符。如果没有指定 *gid*，则指定主机或者子网上的所有用户都将获得访问该服务的许可。如果提供，则只有指定的分组获得访问许可。

aclclear 清除指定服务或端口的 ACL。

acldel 从保存由 DACinet 使用的访问控制列表的内核表中删除 ACL 条目。**dacinet acldel** 子命令从 ACL 中删除某个条目，但仅当它使用的参数正好和曾经用来向 ACL 添加该条目的参数完全一致时才会执行。**acldel** 子命令参数的语法为:

[**-**]*addr*[**/***length*][**u**:*user*|*uid*| **g**:*group*|*gid*] 参数定义如下:

addr DNS 主机名或者某个 IP v4/v6 的地址。地址前的 "-" 表示该 ACL 条目是用来拒绝访问而不是允许访问。

length 表示 *addr* 将会被用来作为一个网络地址而不是主机地址，其中处于第一的 *length* 位取自于 *addr*。

u:*user*|*uid*
可选的用户标识符。如果没有指定 *uid*，则指定主机或者子网上的所有的用户都将获得访问该服务的许可。如果提供，则只有指定的用户获得访问许可。

g:*group*|*gid*
可选的分组标识符。如果没有指定 *gid*，则指定主机或者子网上的所有用户都将获得访问该服务的许可。如果提供，则只有指定的分组获得访问许可。

aclflush 清除系统中定义的所有 ACL，放弃所有不可达到的 TCP 端口的连接请求除了当前主机是使用的 root 用户。它还清除特权端口，使任何进程可以绑定到大于 1024 的端口。

aclls 列出指定服务或端口的 ACL。dacinet aclls 0 列出了缺省的 ACL。对于认证处理，从逻辑观点来看，缺省的 ACL 追加到服务的 ACL 中。如果 ACL 中没有条目与试图连接服务的用户相匹配，访问将会被拒绝。如果存在一个或者更多的条目，列表中带有 *user*|*group@host*|*subnet* 的匹配连接请求者的第一个决定用户能连接到服务的能力。这样可以在对分组添加访问允许之前加入对某位成员访问的拒绝就可以实现对分组中这位成员访问的拒绝。

lspriv 列出了所有特权服务或者没有永久特权的端口（也就是说，它只列出了端口号大于 1024 的特权服务）。

setpriv 将指定服务/端口设置为特权服务／端口，使得只有具有超级用户特权的进程才能绑定到该端口并在该端口提供服务。端口号小于 1024 的端口将会被忽略，因为它们的特权是永久性的。

unsetpriv 将指定服务/端口设置为非特权服务／端口使得任何进程可以将其绑定。任何进程可以在当前临时端口范围内绑定任何端口，不管端口是否标志为有特权。

文件

/usr/sbin/dacinet 包含 **dacinet** 命令。

---

dadmin 命令

用途

用来查询和修改 DHCP 服务器的状态。

语法

dadmin [**-?**] [**-v**] [**-h** *Hostname*] [**-n** *interval*] [**-f**] **-d** *IpAddress* | [**-x**] **-i** | [**-x**] **-s** | **-t on**|**off**|*Value* | **-q** *IpAddress* | **-r** *IpAddress* | **-p** *IpAddress* | **-c** *Clientld*

描述

dadmin 命令允许 DHCP 管理员查询和修改 DHCP 服务器数据库的状态。它使得管理员能够在本机或者远程查询 DHCP 服务器 IP 地址的状态、查询 IP 地址组、查询客户机、删除 IP 地址映射、刷新服务器以及更改服务器的跟踪级别。

dadmin 命令向前兼容前发行版 DHCP 服务器来列出 IP 地址状态并且刷新。

查询一个 IP 地址信息时，**dadmin** 命令返回 IP 地址的状态。根据 IP 地址的状态，**dadmin** 命令可以返回租用的持续时间、开始租用时间、最后租用时间，服务器是否支持该 IP 地址的 DNS A 记录更新，以及映射到该 IP 地址的客户机标识。

在查询客户器信息时，**dadmin** 命令返回客户机的 IP 地址和 IP 地址状态、客户机被分配任何 IP 地址的最后时间、客户机使用的主机名和域名，以及服务器是否支持该 IP 地址的 DNS A 记录更新。

修改服务器的跟踪级别时，**dadmin** 命令按跟踪掩码格式设置并返回服务器的跟踪级别。该掩码代表二进制位串，其中每个位代表特定日志项是否正在被服务器跟踪（请参阅联机文档中的『DHCP 服务器配置文件』）。从最低级到最高级顺序，这些日志项分别是 LOG_NONE、LOG_SYSERR、LOG_OBJERR、LOG_PROTOCOL 和 LOG_PROTERR（同样的值）、LOG_WARN 和 LOG_CONFIG（同样的值）、LOG_EVENT 和 LOG_PARSEERR（同样的值）、LOG_ACTION、
LOG_INFM、LOG_ACNTING、LOG_STAT、LOG_TRACE、LOG_START 和 LOG_RTRACE。

注: LOG_START 不能禁用。这说明掩码范围是从 0x0800 到 0x1FFF。

标志

-c *Clientld* 返回 DHCP 服务器已知的特定客户机的状态。*Clientld* 代表 DHCP 客户机用来标识它自身的客户机标识符，或者是仅由十六进制字符指定的或是在由 DHCP 服务器使用的 TYPE-STRING 表示中的字段。

-d *IpAddress* 删除和 IP 地址 *IpAddress* 关联的租用信息。结果该地址将会移动到 FREE 状态并且可以再一次进行绑定。

-f 和标志 **-d** 一起使用。**-f** 标志将会不经任何提示就强行删除该地址。删除和 IP 地址关联的租用信息。

-h *Hostname* 用来指定目标 DHCP 服务器。*Hostname* 可以输入名称或者 IP 地址。

-i 重新初始化 DHCP 服务器。该标志发出信号通知服务器同步其数据库并且通过重新读入配置文件来重新启动。

-n *interval* 显示服务器统计信息、摘要和所有所请求的时间间隔。

-p *IpAddress* 返回子网的每个 IP 地址的状态。*IpAddress* 用于对列表标识该子网。

-q *IpAddress* 返回特定 IP 地址的状态。

-r *IpAddress* 将 IP 地址置于空闲状态。

-s 返回 DHCP 服务器的配置池的每个地址的状态。

-t on|off|*Value* 修改 DHCP 服务器的跟踪级别。跟踪值以十六进制格式显示，代表服务器中使用的跟踪掩码。*Value* 可以指定为十进制或十六进制格式。关键字 **on** 和 **off** 在跟踪掩码中一次启用或者禁用单个位。

-v 以详细方式执行命令。

-x 使用 **dadmin** 协议的第一版。**-x** 标志用来连接前发行版 DHCP 服务器并且只能对 **-i** 和 **-s** 标志有效。与 DHCPv6 服务器连接时后面要跟 6。

-? 显示用法语法。

退出状态

0 成功结束。

>0 发生错误。

安全性

为了保证 dadmin 客户机的安全连接，DHCP 服务器只允许从服务器自身或者从包含在超级用户的 **.rhosts** 文件中的远程机器的连接。要防止普通用户修改 DHCP 服务器的地址映射，管理员必须确保 **dadmin** 命令的执行只限于那些允许访问的机器上的合法用户。

文件

/usr/sbin/dadmin 包含了 dadmin 命令。

相关信息

.rhosts 文件格式，在 *AIX 5L Version 5.3 Files Reference* 中的 **DHCP Server Configuration File**。

dhcpsd 守护程序。

《*网络与通信管理*》中的『TCP/IP 地址和参数分配 － 动态主机配置协议』。

《*网络与通信管理*》中的『TCP/IP 守护程序』。

---

date 命令

用途

显示或者设置日期或时间。

语法

作为 root 用户设置时间和日期

/usr/bin/date [**-n**] [**-u**] [*Date*] [**+***FieldDescriptor ...*]

显示日期和时间

/usr/bin/date [**-u**] [**+***FieldDescriptor ...*]

作为 root 用户以秒为单位调节时间

/usr/bin/date [**-a**][**+** | **-**]*sss*[.*fff*]

描述

警告: 当系统正有一个以上的用户在运行时，请勿更改日期。

如果不带标志调用或者以 **+**（加号）开头的标志列表进行调用，**date** 命令将当前的日期和时间写到标准的输出。否则，它将设置当前的日期。只有 root 用户能够更改时间和日期。遇到任何不认识的标志或输入时，**date** 命令将打印输出使用消息。

当以 *Date* 参数设置日期时，可以使用以下格式:

- *mmddHHMM*[*YYyy*]
- *mmddHHMM*[*yy*]

Date 参数的变量定义如下:

mm 表示月份数。
dd 表示月份中的天数。
HH 表示一天中的小时数（使用 24 小时制）。
MM 表示分钟数。

YY 表示年份的头两个数字。

注: 如果您没有指定年份的头两个数字，取值范围从 69 到 99，指的是 20 世纪，包括 1969 年到 1999 年，取值范围从 00 到 68 指的是 21 世纪，包括 2000 年到 2068 年。

yy 表示年份的后两个数字。

注: **date** 命令接受 4 个数字的年份的输入。例如，如果指定了 4 个数字的年份，**date** 命令将会尝试按照“YYyy”设置年份，如果数值超出了范围（小于 1970 年和大于 2037 年）则会造成设置失败。

若没有指定年份，当前的年份将作为缺省值使用。系统按照全球标准时间（CUT）操作。

如果您在 **date** 命令后加上 **+**（加号）和一个字段描述符，您可以控制命令的输出。您必须在每个描述符之前加上一个 **%**（百分号）。系统用指定的值替换字段描述符。输入一个文字 % 作为 %%（两个百分号）。**date** 命令复制任何其他的字符到输出中而不加更改。**date** 命令始终以换行符作为字符串的结尾。

标志

-a [**+** | **-**]*sss*[*.fff*] 缓慢调整时间 ***sss.fff*** 秒（*fff* 代表秒片断）。该调整可以是正向或负向的。系统时钟将会加快或者减慢直到它走到指定的秒数。

n 请勿在时钟同步的局域网内的所有机器上设置全局时间。

-u 按照全球标准时间（CUT）显示或者设置时间。

字段描述符

%a 显示语言环境的简略周日名称。

%A 显示语言环境的完整周日名称。

%b 显示语言环境的简略月份名称。

%B 显示语言环境的完整月份名称。

%c 显示语言环境适当的日期和时间表示。此为缺省值。

%C 按十进制数（00-99）显示四个数字代表的年份的前两个数字。一年除以 100 并且只取整数部分。

%d 按照十进制数显示月份中的日期（01-31）。在两个数字的字段，0 表示前导空间填充。

%D 按照等同于 **%m/%d/%y** 的格式显示日期。

%e 按照十进制数显示月份中的日期（1-31）。在两个数字的字段，空白空间用作前导空格填充。

%h 显示语言环境的简略月份名称（**%b** 的同义词）。

%H 以十进制数（00-23）显示小时（24 小时制）。

%I 用十进制数（01-12）显示小时（12 小时制）。

%j 以十进制数（001-366）显示一年中的日期。

%k 将 24 小时计时法小时时钟显示为右对齐、空格填充的数字（0 到 23）。

%m 以十进制数（01-12）显示一年中的月份。

%M 以十进制数（00-59）显示分钟。

%n 插入 <换行> 字符。

%p 显示语言环境中 AM 或者 PM 的等价物。

%r 使用 AM-PM 符号显示 12 小时制（01-12）；在 POSIX 语言环境中这个等同于 **%I:%M:%S %p**。

%S 以十进制数（00-59）显示秒。

%s 显示从全球标准时间（CUT）1970 年 1 月 1 日起的秒数。

%t 插入 <tab> 字符。

%T 显示 24 小时制（00-23），按照等同于 **HH:MM:SS** 的格式。

%u 按照十进制数从 1 到 7 显示周日（星期日 = 7）。请参考 **%w** 字段描述符。

%U 按照十进制数 [00 - 53] 显示一年中的每一周（星期日是一周的第一天）。新的一年中第一个星期日之前的所有天数都认为属于第 0 周。

%V 按照十进制数从 01 到 53 显示一年中的每一周（星期一是一周的第一天）。如果包含 1 月 1 日的那一周有四天或超过四天在新的一年，则认为该周是第 01 周；否则认为是前一年的第 53 周。

%w 按照十进制从 0 到 6 显示周日（星期日 = 0）。请参阅 **%u** 的字段描述符。

%W 按照十进制以星期一作为一周的第一天来显示一年中的周数（00-53）。

%x 显示语言环境的适当的日期的表示。
%X 显示语言环境的适当的时间的表示。
%y 显示年份的最后两个数字（00-99）。
%Y 按照十进制以四个数字显示年份。
%Z 显示时区名称，或者如果没有指定时区则没有字符。
%% 显示一个 %（百分号）字符。

修改的字段描述符

可以修改 **%E** 和 **%O** 字段描述符来指出不同的格式或说明，按照 *AIX 5L Version 5.3 Files Reference* 中的 Locale Definition Source File Format 的 **LC_TIME** Category 中描述的那样。如果没有指定对应的关键字（请参阅 **era**、**era_year**、**era_d_fmt** 和 **alt_digits** 关键字），或者当前语言环境不支持，则使用没有修改的字段描述符的值。

%Ec 显示语言环境的适当备用日期和时间表示。
%EC 在语言环境的备用表示中显示基准年份名（或者其他时间周期）。
%Ex 显示语言环境的备用日期表示。
%EX 显示语言环境的备用时间表示。
%Ey 在语言环境的备用表示中显示 **%EC** 字段描述符的偏移量（只针对年）。
%EY 显示完整的备用年份表示。
%Od 使用语言环境的备用数字符号显示月份中的日期。
%Oe 使用语言环境的备用数字符号显示月份中的日期。
%OH 用语言环境的备用数字符号显示小时（24 小时制）。
%OI 用语言环境的备用数字符号显示小时（12 小时制）。

%Om 使用语言环境的备用数字符号显示月份。
%OM 使用语言环境的备用数字符号显示分钟。
%OS 使用语言环境的备用数字符号显示秒。
%Ou 使用语言环境的备用表示以数字显示周日（星期一 = 1）。
%OU 使用语言环境的备用数字符号显示一年当中的周号。星期日认为是一周中的第一天。
%OV 使用语言环境的备用数字符号显示一年当中的周号。星期一认为是一周中的第一天。
%Ow 使用语言环境的备用表示以数字显示周日（星期日 = 0）。
%OW 使用语言环境的备用数字符号显示一年当中的周号。星期一认为是一周中的第一天。
%Oy 使用备用表示显示年份（%C 的偏移量）。

退出状态

此命令返回以下退出值:

0 该日期已经成功写入。
>0 发生错误。

示例

1. 要显示当前日期和时间，请输入:

   ```
   date
   ```

2. 要设置日期和时间，请输入:

   ```
   date 0217142590
   ```

   对于使用 CST 时区的系统，这项输入设置日期和时间为: CST 1990 年 2 月 17 日 14:25:00 星期六。

   注: 您必须具有 root 用户权限才可以更改日期和时间。

3. 要以指定的格式显示日期和时间，请输入:

```
date +"%r %a %d %h %y (Julian Date: %j)"
```

示例 2 中显示的日期显示如下:

```
02:25:03 PM Fri 17 Feb 90 (Julian Date: 048)
```

环境变量

以下环境变量将会影响到 **date** 命令的执行效果。

LANG	确定在 **LC_ALL** 和相应的环境变量（以 **LC_** 开头）都不指定语言环境的时候将要使用的缺省语言环境。
LC_ALL	确定用于覆盖由 **LANG** 或任何以 **LC_** 开头的环境变量设置的任何语言环境类别值的语言环境。
LC_CTYPE	确定文本数据的字节序列解释为字符时使用的语言环境（例如参数中单对多字节字符）。
LC_MESSAGES	决定写信息应使用的语言。
LC_TIME	确定由 **date** 写入的日期和时间字符串的内容。
NLSPATH	为处理 **LC_MESSAGES** 决定消息目录的位置。
TZ	指定时间和日期使用的时区，除非指定了 **-u** 选项。如果未设置 **TZ** 变量且未指定 **-u** 标志，则使用非指定的系统缺省时区。

相关信息

environment 文件。

localtime 子例程、**strftime** 子例程和 **time** 子例程。

AIX 5L Version 5.3 Files Reference 中 Locale Definition Source File Format 的 **LC_TIME** Category。

《*AIX 5L V5.3 本地语言支持指南和参考大全*》中的『理解语言环境』讨论了语言环境值。

---

dbts 命令

用途

调试瘦服务器。

语法

dbts [**-v**] *ThinServer*

描述

dbts 命令使瘦服务器引导为调试方式。该命令通过搜索为瘦服务器创建的调试引导映像，检查瘦服务器是否先前曾引导为调试方式。如果未找到该映像，则克隆瘦服务器所使用的公共映像，并从该克隆创建调试引导映像，以允许瘦服务器引导为调试方式。调试引导映像克隆使用以下命名约定:

{*COSI name*}_{*thin server name*}-debug

在瘦服务器使用调试公共映像结束之后，必须运行 **swts** 命令以将瘦服务器切换到另一个公共映像。**rmcosi** 命令除去从 **dbts** 命令创建的调试公共映像。**dbts** 命令可在 NIM 主服务器或瘦服务器上运行。

标志

-v 在 **dbts** 命令运行时启用详细调试输出。

退出状态

0 命令成功完成。
>0 发生错误。

安全性

访问控制: 必须拥有 root 用户权限才能运行 **dbts** 命令。

示例

1. 要调试引导名为 lobo、并且正在使用名为 cosi1 的公共映像的瘦服务器，请输入:

   ```
   dbts lobo
   ```

   这将创建名为 cosi1_lobo-debug 的调试引导映像，以将 lobo 引导为调试方式。

位置

/usr/sbin/dbts

文件

/etc/niminfo 包含 NIM 所使用的变量。

相关信息

lsts 命令、**mkcosi** 命令、**mkts** 命令、**nim** 命令、**nim_clients_setup** 命令、**nim_master_setup** 命令、**nimconfig** 命令、**rmts** 命令，以及 **swts** 命令。

---

dbx 命令

用途

提供了一个调试和运行程序的环境。

语法

dbx [**-a** *ProcessID*] [**-c** *CommandFile*] [**-d** *NestingDepth*] [**-I** *Directory*] [**-E** *DebugEnvironment*] [**-p** *oldpath=newpath:...*| *pathfile*] [**-k**] [**-u**] [**-F**] [**-r**] [**-x**] [*ObjectFile* [*CoreFile*]]

描述

dbx 命令为 C、C++、Pascal 和 FORTRAN 语言程序提供了一个符号调试程序，允许您按照下面的内容执行操作:

- 检验对象和核心文件。
- 为程序运行提供一个控制环境。

- 在选中的语句处设置断点或单步执行程序。
- 使用符号变量进行调试并将它们以正确的格式进行显示。

ObjectFile 参数是一个由编译器产生的对象（可执行）文件。在编译您的程序时，使用 **-g**（生成符号表）标志产生 **dbx** 命令需要的信息。

> **注**: **cc** 命令的 **-g** 标志应该在对象文件进行编译时使用。如果没有使用 **-g** 标志或者如果符号引用由 **strip** 命令从 **xcoff** 文件中删除，则 **dbx** 命令的符号性能将受到限制。此外，请勿使用 **-O** 编译器选项来优化您打算用 **dbx** 进行调试的可执行文件。优化操作将重新编排代码，并会破坏调试数据，进而限制了用 **dbx** 调试可执行文件的值。

如果没有指定 **-c** 标志，**dbx** 命令将会在用户的 **$HOME** 目录中检查 **.dbxinit** 文件。然后它在用户当前目录中检查 **.dbxinit** 文件。如果在当前目录中存在一个 **.dbxinit** 文件，该文件将覆盖用户的 **$HOME** 目录中的 **.dbxinit** 文件。如果在发现了 **.dbxinit** 文件存在于用户的 **$HOME** 目录中或者当前目录中，该文件的子命令将在调试部分开始时运行。使用编辑器创建一个 **.dbxinit** 文件。

如果没有指定 *ObjectFile*，则 **dbx** 将询问要进行检查的对象文件的名称。缺省值是 **a.out**。如果当前目录中存在 **core** 文件或者指定了 *CoreFile* 参数，则 **dbx** 将报告程序发生错误的位置。在开始执行 *ObjectFile*之前，可能会检验保留在核心映像中的变量、寄存器和内存。那时 **dbx** 调试程序提示输入命令。

表达式句柄

dbx 程序可以显示各种各样的表达式。您可以在 **dbx** 调试程序中使用 C 和 Pascal 语法的公共子集和一些 FORTRAN 扩展的子集来指定表达式。

以下运算符在调试程序中有效:

*****（星号）或者 **^**（脱字符号）	表示正在取消引用间接寻址或者指针。
[ ]（方括号）或者 **( )**（圆括号）	表示数组表达式的下标。
.（句点）	通过指针和结构来使用该字段引用运算符。这将使得 C 运算符 **->**（箭头）变得没有必要，尽管它仍允许使用。
&（与符号）	获得变量的地址。
..（两个句点）	在指定数组子部分时将上下界分开。例如: **n[1..4]**。

以下类型的运算在调试程序的表达式中有效:

代数运算	**=**、**-**、*****、**/**（浮点运算的除法）、**div**（整数运算的除法）、**mod** 和 **exp**（乘幂运算）。
按位运算	**-**、**|**、**bitand**、**xor**、**~**. **<<** 和 **>>**
逻辑运算	**or**、**and**、**not**、**||** 和 **&&**
比较运算	**<**、**>**、**<=**、**>=**、**< >** 或 **!=** 以及 **=** 或 **==**
其他	（**类型名**），**sizeof**

逻辑和比较表达式在 **stop** 和 **trace** 中作为条件使用的。

检验表达式类型。您可以通过重命名或者重载运算符的方式重设一个表达式类型。三种重命名的格式分别是: *类型名*（*表达式*），*表达式* |*类型名*以及（*类型名*）*表达式*。以下是一个示例，其中 *x* 变量是一个值为 97 的整数:

```
(dbx) print x
97
(dbx) print char (x), x \ char, (char) x, x
'a' 'a' 'a' 97
```

命令行编辑

dbx 命令提供一种命令行编辑功能类似于由 Korn Shell 提供的功能。**vi** 方式提供**类似 vi** 的编辑功能，而 **emacs** 方式赋予您类似于 **emacs** 的控制功能。

这些功能可以通过使用 **dbx** 子命令 **set -o** 或者 **set edit** 来开启。要打开 vi 风格的命令行编辑，您可以输入子命令 **set edit vi** 或者 **set -o vi**。

您也可以使用 **EDITOR** 环境变量来设置编辑方式。

dbx 命令将保存输入到历史文件 **.dbxhistory** 当中的命令。如果没有设置 **DBXHISTFILE** 环境变量，使用的历史文件是 **$HOME/.dbxhistory**。

按照缺省值，**dbx** 将保存最近输入的 128 个命令。**DBXHISTSIZE** 环境变量可以从来增加该限制。

标志

-a *ProcessID*
将调试程序和正在运行的进程连接起来。要连接调试程序，您需要向此进程发送信号的权限。使用 **ps** 命令来决定进程的 ID。如果您有许可权，**dbx** 程序将使用 **ptrace** 系统调用中断该进程以向其发送 **SIGTRAP** 信号，该进程不可忽略 **SIGTRAP** 信号。然后，dbx 确定对象文件的全名、读入符号信息，并提示输入命令。

-c *CommandFile*
读入标准输入之前，在文件中运行 **dbx** 子命令。**$HOME** 目录中指定的文件将首先被处理；然后处理当前目录中的文件。当前目录中的命令文件将会覆盖 **$HOME** 目录中的命令文件。如果指定的文件既不存在于 **$HOME** 目录也不存在于当前目录，将会显示警告消息。**source** 子命令可以在 **dbx** 程序运行后使用。

-d *NestingDepth*
设置程序块的嵌套限制。缺省的嵌套深度限制为 25。

-E *DebugEnvironment*
指定调试程序的环境变量。

-p *oldpath=newpath:...| pathfile*
以 *oldpath=newpath* 的格式在检查核心文件时指定替换的库的路径。*oldpath* 指定了将被替换的值（存储在核心文件中）而 *newpath* 指定了将要替换的新的值。这些可能是全部或者部分的，绝对路径或者相对路径。可以指定若干个替换，它们之间用冒号隔开。相反，**-p** 标志可以指定文件名，映射以前读入的描述格式。每行中只允许有一个映射从文件中读出。

-F
可以用来关闭缓慢读入方式并使 **dbx** 命令在启动时便读入全部符号。缺省情况下，缓慢读入方式是打开的：它在 **dbx** 会话初始化时读入需要的符号表信息。在该方式下，**dbx** 将不会读入那些符号信息尚未被读入的变量和类型。因此，诸如 **whereis i** 等命令并不列出在所有函数中的局部变量 **i** 的全部实例。

-I *Directory*
（大写 i）将 *Directory* 变量指定的目录包含到搜索源文件目录列表中。搜索源文件的缺省目录为：

- 文件编译时该源文件所在的目录。只有编译器设置了对象中的源路径时才能搜索目录。
- 当前目录。
- 当前程序所在的目录。

-k
映射内存地址；这对于内核调试是非常有用的。

-r
立即运行对象文件。如果它成功结束，**dbx** 调试程序将会退出。否则，将会进入调试程序并报告中断的原因。

注：除非指定了 **-r**，否则 **dbx** 命令将会提示用户并等待命令输入。

-u
让 **dbx** 命令为文件名加上 @（at 符号）前缀。该标志减少符号名混淆的可能性。

-x 防止 **dbx** 命令跳过来自于 FORTRAN 源代码的 _（下划线）字符。该标志允许 **dbx** 在符号之间区别哪些除了下划线以外都是相同的，例如 xxx 和 xxx_。

示例

1. 以下示例解释如何在启动 **dbx** 调试程序时同时启动一个进程。本例使用了一个名为 **samp.c** 的程序。该 C 程序首先经过 **-g** 标志的编译生成一个包含符号列表引用的对象文件。在此情况下，该程序命名为: **samp**:

   ```
   $ cc -g samp.c -o samp
   ```

   samp 程序运行后，操作系统会报告一个总线错误并将核心映像写入到您当前的工作目录中，如下所示:

   ```
   $ samp
   Bus Error - core dumped
   ```

   要确定发生错误的位置，请输入:

   ```
   $ dbx samp
   ```

   系统将会返回如下信息:

   ```
   dbx version 3.1
   Type 'help' for help.
   reading symbolic information . . . [
   using memory image in core]
     25   x[i] = 0;
   (dbx) quit
   ```

2. 该示例解释了如何将 **dbx** 连接到进程。该示例使用了如下程序 **looper.c**:

   ```
   main()
   {
         int i,x[10];

         for (i = 0; i < 10;);
   }
   ```

   由于 **i** 不会增加，因此该程序不会中止。使用 **-g** 标志编译 **looper.c** 以获取符号调试能力:

   ```
   $ cc -g looper.c -o looper
   ```

   在命令行中运行 **looper** 并执行以下步骤便会在它运行的时候将 **dbx** 连接到程序:

   a. 要将 **dbx** 连接到 **looper**，则必须确定进程的 ID。如果您没有运行 **looper** 作为后台程序，您必须启动另外一个 Xwindow 窗口。在该 Xwindow 窗口中，输入:

      ```
      ps -u UserID
      ```

      其中 *UserID* 是您的登录标识。所有属于您的活动的进程将会显示如下:

      ```
      PID     TTY      TIME    COMMAND
      68      console   0:04    sh
      467     lft3     10:48    looper
      ```

      在该示例中，与 **looper** 相关的进程 ID 是 467。

   b. 要将 **dbx** 连接到 **looper**，请输入:

      ```
      $ dbx -a 467
      ```

      系统将返回如下的信息:

      ```
      Waiting to attach to process 467 . . .
      Successfully attached to /tmp/looper.
      dbx is initializing
      Type 'help' for help.
      ```

```
reading symbolic information . . .

attached in main at line 5
5     for (i = 0; i < 10;);
(dbx)
```

现在，您便可查询和调试进程，如同该程序最初由 **dbx** 进行启动。

3. 要将目录添加到用来搜索可执行文件 **objfile** 的源文件的目录列表中，可以输入:

```
$dbx -I /home/user/src -I /home/group/src
objfile
```

use 子命令可以在 **dbx** 启动之后用来实现此功能。**use** 命令重新设置目录清单，而 **-I** 标志则向清单中添加目录。

4. 要使用 **-r** 标志，请输入:

```
$ dbx -r samp
```

系统将会返回如下信息:

```
Entering debug program . . .
dbx version 3.1
Type 'help' for help.
reading symbolic information . . .
bus error in main at line 25
  25   x[i] = 0;
(dbx) quit
```

虽然没有设置核心映像，但 **-r** 标志仍然允许您检查进程的状态。

5. 要指定调试程序的环境变量，请输入:

```
dbx -E LIBPATH=/home/user/lib -E LANG=Ja_JP objfile
```

dbx 子命令

注: 这些子命令只能在运行 **dbx** 调试程序时使用。

/ 在当前源文件中向前搜索某种模式。
? 在当前源文件中向后搜索某种模式。
addcmd 将 **dbx** 子命令添加到指定的事件号。
alias 创建 **dbx** 子命令的别名。
assign 为一个变量赋值。
attribute 显示所有或者选中属性对象的信息。
call 运行与指定的过程或函数相关的对象代码。
case 修改 dbx 调试程序解释符号的方式。
catch 在一个信号送到应用程序之前启动捕获这个信号。
clear 删除所有源指定行的停止。
cleari 除去地址中所有的断点。
condition 显示全部或者选中的条件变量的信息。
cont 从当前停止点继续应用程序的执行直到程序终止或者遇到下一个断点。
corefile 显示有关 corefile 的高级数据。
delcmd 删除与指定事件号关联的 **dbx** 子命令。
delete 除去与指定事件号对应的跟踪和停止，以及线程的 **tskip** 计数。
detach 继续应用程序的执行并退出调试程序。
disable 禁用与指定事件号对应的跟踪和停止。
display memory 显示内存中的内容。
down 将当前函数在堆栈中向下移动。
dump 显示指定过程中的变量的名称和值。
edit 启动编辑器编辑指定文件。

enable 启用与指定事件号对应的跟踪和停止。
fd 显示文件描述符信息。
file 将当前源文件修改成指定的文件。
frame 将当前函数更改为对应指定堆栈帧号的函数。
func 将当前函数修改成指定的过程或者函数。
goto 使指定行成为下一个运行的行。
gotoi 修改程序计数器地址。
handler 显示有关 pthreads atfork 或 cancellation cleanup 处理程序的信息。
help 显示 **dbx** 子命令或主题的帮助信息。
ignore 在一个信号送到应用程序之前停止捕获这个信号。
kthread 显示有关内核线程的信息。

list 显示当前源文件的行。
listi 列出应用程序的指令清单。
malloc 显示有关程序对 malloc 子系统的使用情况的信息。
map 显示关于应用程序装入特征的信息。
move 跳到下一行并显示。
multproc 启用或禁用多线程调试。
mutex 显示全部或者选中的互斥信息。
next 运行应用程序直到下一个源程序行。
nexti 运行应用程序直到下一个机器指令。
onceblock 显示有关 once 块的信息。
plugin 调用插件子命令或显示可用插件名称。
pluginload 装入插件。
pluginunload 卸装插件。
print 打印表达式的值或运行一个过程并打印过程的返回代码。
proc 显示有关进程的信息。
prompt 修改 dbx 命令提示符。
quit 停止 dbx 调试程序。
registers 显示所有通用寄存器、系统控制寄存器、浮点寄存器和当前指令寄存器的值。
rerun 按照以前设置的参数开始运行应用程序。
resource 显示有关 pthread 拥有或等待的资源的信息。
return 继续应用程序的运行直到达到返回一个指定的过程。
rwlock 显示 rwlocks 的信息。
run 开始运行一个应用程序。
screen 打开一个 Xwindow 作为 dbx 命令的交互。
set 为 dbx 调试程序变量定义一个值。
sh 传递命令到 shell 去运行。
skip 从当前的停止处继续运行应用程序。
source 从文件中读取 **dbx** 子命令。
status 显示活动的跟踪、停止子命令，以及剩余线程 **tskip** 计数。

step 运行一个源命令行。
stepi 运行一个机器指令。
stophwp 设置一个硬件的观察点停止。
stop 停止运行应用程序。
stopi 在指定位置设置停止点。
thread 显示和控制线程。
tls 显示 TLS 初始化模板信息。
tnext 运行线程直到下一个源程序行。
tnexti 运行线程直到下一个机器指令。
trace 打印跟踪信息。
tracehwp 设置硬件观察点跟踪。
tracei 打开跟踪。

tskip	跳过线程断点。
tstep	为一个源程序行运行一个线程。
tstepi	为一个机器指令运行一个线程。
tstop	为线程设置源程序级断点停止。
tstophwp	设置线程级硬件观察点停止。
tstopi	为线程设置指令级断点停止。
ttrace	为线程设置源程序级跟踪。
ttracehwp	设置线程级硬件观察点跟踪。
ttracei	为线程设置指令级跟踪。
unalias	删除别名。
unset	删除一个变量。
up	将当前函数在堆栈中向上移动。
use	设置在搜索源文件时要搜索的目录列表。
whatis	显示应用程序组件的声明。
where	显示活动过程和函数的清单。
whereis	显示所有名字匹配指定标识符的符号的全限定。
which	显示给定标识符的全限定。

/ 子命令

/ [*RegularExpression* [**/**]]

/ 子命令在当前源文件中向前搜索由 *RegularExpression* 参数指定的模式。不带参数输入 **/** 子命令将会使 **dbx** 命令向前搜索以前的正则表达式。搜索将会在文件的结尾折返。

示例:

1. 要向前搜索当前源文件中的数字 12，请输入:

   ```
   / 12
   ```

2. 要重复以前的搜索，请输入:

   ```
   /
   ```

请参阅 **?**（搜索）子命令和 **regcmp** 子例程。

? 子命令

? [*RegularExpression* [**?**]]

? 子命令在当前源文件中向后搜索由 *RegularExpression* 参数指定的模式。不带参数输入 **?** 子命令将会使 **dbx** 命令向后搜索以前的正则表达式。搜索将会在文件的结尾折返。

示例:

1. 要向后搜索当前源文件中的字符 z，请输入:

   ```
   ?z
   ```

2. 要重复以前的搜索，请输入:

   ```
   ?
   ```

请参阅 **/**（搜索）子命令和 **regcmp** 子例程。

addcmd 子命令

addcmd { *Number...* | **all** } "*commands_string*"

addcmd 子命令将 **dbx** 子命令添加到指定事件。每当遇到对应该事件的断点、跟踪点或观察点时，都将执行这些子命令。**dbx** 子命令可通过 "*commands_string*" 参数指定，这是一组由分号（;）分隔的 **dbx** 子命令。**dbx** 子命令所添加到的事件可通过 *Number* 参数指定，或者可使用 **all** 标志将 **dbx** 子命令添加到所有事件。

标志:

all 将 **dbx** 子命令添加到所有事件。

示例:

1. 要将 **where** 子命令添加到事件号 1，请输入:
   ```
   addcmd 1 "where"
   ```
2. 要将 **registers** 子命令添加到事件号 2，请输入:
   ```
   addcmd 2 "registers"
   ```
3. 要将 **where** 和 **registers** 子命令添加到事件号 3，请输入:
   ```
   addcmd 3 "where;registers"
   ```

请参阅 **clear** 子命令、**delcmd** 子命令、**delete** 子命令、**disable** 子命令、**enable** 子命令、**stop** 子命令、**status** 子命令，以及 **trace** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『设置和删除断点』。

alias 子命令

alias [*Name* [[(*Arglist*)] *String* | *Subcommand*]]

alias 子命令为 **dbx** 子命令创建别名。*Name* 参数是要创建的别名。*String* 参数是 **dbx** 子命令的序列，在执行完该子命令后可以通过 *Name* 引用这些序列。如果 **alias** 子命令没有指定参数，则它将显示所有当前的别名。

示例:

1. 要用 `rr` 替换 `rerun`，请输入:
   ```
   alias rr rerun
   ```
2. 要在命令行输入 `printandstep` 时运行 `print n` 和 `step` 两个子命令，请输入:
   ```
   alias printandstep "print n; step"
   ```
3. `alias` 子命令也可以用作一个有限的宏工具。例如:
   ```
   (dbx) alias px(n) "set $hexints; print n; unset $hexints"
   (dbx) alias a(x,y) "print symname[x]->symvalue._n_n.name.Id[y]"
   (dbx) px(126)
   0x7e
   ```

   在本示例中，别名 `px` 显示了一个十六进制的值而不会永久影响到调试环境。

assign 子命令

assign *Variable* **=***Expression*

assign 子命令将 *Expression* 参数指定的值分配到由 *Variable* 参数指定的变量。

示例:

1. 要将数值 `5` 分配到 `x` 变量，请输入:
   ```
   assign x = 5
   ```

2. 要将变量 y 的值分配到变量 x，请输入:

```
assign x =  y
```

3. 要将字符值 'z' 分配到变量 z，请输入:

```
assign  z  =  'z'
```

4. 要将布尔值 false 分配到逻辑类型变量 B，请输入:

```
assign  B  =  false
```

5. 要将字符串 "Hello World" 分配到字符指针 Y，请输入:

```
assign  Y  =  "Hello  World"
```

6. 要禁用类型检查，将 **dbx** 调试程序变量 $unsafeassign 进行设置，请输入:

```
set $unsafeassign
```

请参阅『显示和修改变量』。

attribute 子命令

attribute [*AttributeNumber* ...]

attribute 子命令显示关于 *AttributeNumber* 参数定义的用户线程、互斥或者条件属性对象的相关信息。如果没有指定参数，则会列出所有的属性对象。

对于列出的每个属性对象，将显示如下信息:

attr	按照 $a*AttributeNumber* 格式指出属性对象的符号名。
obj_addr	指出属性对象的地址。
type	指出属性对象的类型；对于用户线程、互斥以及条件变量的类型分别是 thr、mutex 或者 cond。
state	指出属性对象的状态。该状态可能是 valid 或 inval。
stack	指出一个线程属性对象的堆栈大小属性。
scope	指出线程属性对象的作用域属性。这将决定线程的争用域并定义必须争用进程资源的线程的集合。对于系统或者进程争用域，作用域的值可以是 sys 或 pro。
prio	指出线程属性对象的优先级属性。
sched	指出一个线程属性对象的调度策略属性。该属性控制了调度策略，可选的值为 fifo、rr（循环法）或 other。
p-shar	指定互斥或条件属性对象的进程共享属性。如果可以由不同进程的线程访问，该互斥或条件是进程共享的。该属性值可以是 yes 或 no。
protocol	指出互斥的协议属性。该属性决定了保持线程优先级互斥的作用。可选的值有 no_prio、prio 或 protect。
clock	显示条件属性对象的时钟属性。该属性确定将等待条件变量的线程指定为超时的时候必须使用的时钟。该值可以是 realtime 或 monotonic。

注:

1. **dbx** 调试程序的 **print** 子命令识别符号属性名称，还可用于显示相应对象的状态。
2. 可用的属性取决于 POSIX 选项的实现。

示例:

1. 要列出所有属性的信息，请输入:

```
attribute
```

输出类似于:

```
attr   obj_addr   type  state  stack   scope    prio
sched p-shar
$a1    0x200035c8  mutex valid                                 no
$a2    0x20003628  cond  valid                                 no
$a3    0x200037c8  thr   valid  57344    sys      126 other
$a4    0x200050f8  thr   valid  57344    pro      126 other
```

2. 要列出属性 1 和 3 的信息，请输入:

```
attribute 1 3
```

输出类似于:

```
attr   obj_addr   type  state  stack   scope    prio
sched p-shar
$a1    0x200035c8  mutex valid                                 no
$a3    0x200037c8  thr   valid  57344    sys      126 other
```

请参阅 **dbx** 命令中 **condition** 子命令、**mutex** 子命令、**print** 子命令和 **thread** 子命令。

另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『创建线程』、『使用互斥对象』和『使用条件变量』。

call 子命令

call *Procedure* ([*Parameters*])

call 子命令运行由 *Procedure* 参数指定的过程。返回代码不会显示。如果指定了参数，它们将传给正在运行的过程。

注: call 子命令不能用来调用使用向量参数的函数。

示例: 运行 dbx 时要调用一个命令，请输入:

```
(dbx) call printf("hello")
hello
```

printf 成功返回。

case 子命令

case [**default** | **mixed** | **lower** | **upper**]

case 子命令修改 **dbx** 调试程序解释符号的方式。缺省的符号处理视当前使用的语言而定。如果当前的语言是 C、C++ 或没有定义，这些符号将不会合并；如果当前的语言是 FORTRAN 或 Pascal，这些符号会合并成小写格式。如果符号需要使用与当前语言不一致的方法解释，则使用该子命令。

不带参数输入 **case** 子命令将会显示当前的 case 方式。

标志:

default 随当前语言变化。
mixed 按实际显示的情况解释符号。
lower 按小写来解释符号。
upper 按大写来解释符号。

示例:

1. 要显示当前的大小写方式，请输入:

```
case
```

2. 要指示 **dbx** 命令按照符号实际形式解释符号，请输入:

   ```
   case mixed
   ```

3. 要指示 **dbx** 按照大写形式解释符号，请输入:

   ```
   case upper
   ```

请参阅 Folding Variables to Lowercase and Uppercase。

catch 子命令

catch [*SignalNumber* | *SignalName*]

catch 子命令在信号送到应用程序之前启动捕获这个信号。该子命令在调试应用程序处理例如中断之类的信号时非常有用。待捕获的信号可以分别用 *SignalNumber* 参数按号码或者用 *SignalName* 参数按名称来指定。信号名不区分大小写，而前缀 **SIG** 为可选。如果既没有指定 *SignalNumber* 也没有指定 *SignalName* 参数，则将默认捕获除了 **SIGHUP**、**SIGCLD**、**SIGALARM** 和 **SIGKILL** 之外的所有信号。如果没有指定参数，则将显示当前能够捕获的信号的列表。

示例:

1. 要显示当前能够由 **dbx** 捕获的信号的列表，请输入:

   ```
   catch
   ```

2. 要捕获 SIGALARM 信号，请输入:

   ```
   catch SIGALARM
   ```

请参阅 **ignore** 子命令和信号处理。

clear 子命令

clear *SourceLine*

clear 子命令删除所有给定源文件行的停止。*SourceLine* 参数可以按照两种格式指定:

- 作为一个整数
- 作为一个文件名字符串，后面跟一个 :（冒号）和一个整数

示例: 要删除第 19 行中设置的断点，请输入:

```
clear 19
```

请参阅 **cleari** 子命令和 **delete** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『设置和删除断点』。

cleari 子命令

cleari *Address*

cleari 子命令清除由 *Address* 参数指定的地址的所有断点。

示例:

1. 要删除设置在地址 0x100001b4 处的断点，请输入:

   ```
   cleari 0x100001b4
   ```

2. 要删除设置在 main() 过程地址处的断点，请输入:

   ```
   cleari &main
   ```

请参阅 **clear** 子命令、**delete** 子命令和《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『设置和删除断点』。

condition 子命令

condition [**wait** | **nowait** | *ConditionNumber ...*]

condition 子命令显示一个或多个条件变量的相关信息。如果指定了一个或多个 *ConditionNumber* 参数，**condition** 子命令将会显示指定条件变量的信息。如果没有指定标志或者参数，**condition** 子命令列出所有条件变量的清单。

各个条件的信息列表如下:

cv	按照 $c*ConditionNumber* 的格式指出条件变量的符号名称。
obj_addr	设定条件变量的内存地址。
clock	显示条件变量的时钟属性。
num_wait	设定条件变量中等待的线程的数量。
waiters	列出等待条件变量的用户线程。

注: **dbx** 调试程序的 **print** 子命令识别符号条件变量名称并用来显示相应对象的状态。

标志:

wait	显示有等待线程的条件变量。
nowait	显示没有等待线程的条件变量。

示例:

1. 要显示关于所有条件变量的信息，请输入:

   ```
   condition
   ```

2. 要显示有等待线程的条件变量的信息，请输入:

   ```
   condition  wait
   ```

3. 要显示关于条件变量 3 的信息，请输入:

   ```
   condition 3
   ```

   输出类似于:

   ```
   cv      obj_addr     num_wait  waiters
   $c3     0x20003290         0
   ```

请参阅 **attribute** 子命令、**mutex** 子命令、**print** 子命令以及 **thread** 子命令。

另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》的 『使用条件变量』。

cont 子命令

cont [*SignalNumber* | *SignalName*]

cont 子命令从当前的断点开始继续执行应用程序直到该程序结束或者遇到另外一个断点。如果通过 *SignalNumber* 参数（指定信号数）或 *SignalName* 参数（指定信号名），该程序将假定已经收到该信号并继续执行。信号名不区分大小写而且前缀 **SIG** 是可选的。如果没有指定信号，该程序将如同没有停止下来过一样继续执行。

示例

1. 要从当前断点位置继续执行程序，请输入:

   ```
   cont
   ```

2. 要如同收到 SIGQUIT 信号一样继续执行程序，请输入:

   ```
   cont SIGQUIT
   ```

请参阅 **dbx** 命令的 **detach** 子命令、**dbx** 的 **goto** 子命令、**dbx** 的 **next** 子命令、**dbx** 的 **skip** 子命令、**dbx** 的 **step** 子命令。

corefile 子命令

corefile 子命令从 corefile 的报头显示信息，包括可执行文件名称、corefile 格式版本控制信息、指示可用数据的标志、导致崩溃的信号以及堆放核心的进程的执行方式。

delcmd 子命令

delcmd *EventNumber* { *Number...* | **all** }

delcmd 子命令除去与指定事件关联的 **dbx** 子命令。可通过 *Number* 参数指定要除去的 **dbx** 子命令，或者可使用 **all** 标志除去与指定事件关联的所有 **dbx** 子命令。*EventNumber* 参数指定要从中除去 **dbx** 子命令的事件。

标志:

all 除去与指定事件关联的所有 **dbx** 子命令。

示例:

1. 要从事件号 2 除去所有 dbx 子命令，请输入:

   ```
   delcmd 2 all
   ```

2. 要从事件号 3 除去 1 号 **dbx** 子命令，请输入:

   ```
   delcmd 3 1
   ```

3. 要从事件号 2 除去 1 号和 2 号 **dbx** 子命令，请输入:

   ```
   delcmd 2 1 2
   ```

请参阅 **addcmd** 子命令、**clear** 子命令、**delete** 子命令、**disable** 子命令、**enable** 子命令、**stop** 子命令、**status** 子命令，以及 **trace** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『设置和删除断点』。

delete 子命令

delete { *Number ...* | **all** | **tskip** [**for $t***threadnumber*]}

delete 子命令从应用程序除去跟踪和停止以及线程的 **tskip** 计数。可通过 *Number* 参数指定要除去的跟踪和停止，或者可使用 **all** 标志除去所有跟踪和停止。使用 **status** 子命令来显示 dbx 调试程序为跟踪和停止生成的相应跟踪号和停止号。

剩余 **tskip** 计数（使用 **tskip** 子命令为线程设置）可使用 **tskip** 标志删除。使用 **status** 子命令可显示剩余线程 **tskip** 计数。如果未指定任何线程，则使用当前线程。

标志

all	删除所有的跟踪和停止。
for $t *threadnumber*	指定线程号。

示例

1. 要从应用程序中删除所有的跟踪和停止，请输入:

   ```
   delete all
   ```

2. 要删除事件号为 4 的跟踪和停止，请输入:

   ```
   delete 4
   ```

3. 要除去线程 3 的 **tskip** 计数，请输入:

   ```
   delete tskip for $t3
   ```

4. 要除去当前线程的 **tskip** 计数，请输入:

   ```
   delete tskip
   ```

请参阅 **clear** 子命令、**cleari** 子命令、**status** 子命令、**tskip** 子命令，以及《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『设置和删除断点』。

detach 子命令

detach [*SignalNumber* | *SignalName*]

detach 子命令将继续应用程序的执行并退出调试程序。信号可以通过以下任意一项指定:

- 名称，使用 *SignalName* 参数
- 序号，使用 *SignalNumber* 参数

信号名不区分大小写而且前缀 **SIG** 是可选的。

如果指定了信号，程序将如同接收到该信号一样继续执行。如果没有指定信号，该程序将如同没有停止下来过一样继续执行。

示例

1. 要退出 **dbx** 而继续执行应用程序，请输入:

   ```
   detach
   ```

2. 要退出 **dbx** 并如同收到 SIGREQUEST 信号那样继续执行程序，请输入:

   ```
   detach SIGREQUEST
   ```

请参阅 Using dbx Debug Program。

disable 子命令

disable { *Number* ... **all** }

disable 子命令禁用与调试事件关联的跟踪和停止。要禁用的跟踪和停止可通过 *Number* 参数指定，或者可使用标志 **all** 来禁用所有跟踪和停止。使用 **status** 子命令来显示由 dbx 调试程序将其和跟踪或停止关联的事件号。

标志:

all 删除所有的跟踪和停止。

示例:

1. 要从应用程序禁用所有跟踪和停止，请输入:

```
disable all
```

2. 要禁用事件号 4 的跟踪和停止，请输入:

```
disable 4
```

有关更多信息，请参阅第 24 页的『enable 子命令』、第 20 页的『delete 子命令』和第 43 页的『status 子命令』。

另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『设置和删除断点』。

display memory 子命令

{ *Address,Address*/ | *Address*/ [*Count*] } [*Mode*] [**>***File*]

不带任何关键字进行初始化的 **display memory** 子命令将显示由如下因素控制的内存的部分内容:

显示的内存范围通过指定以下某个方法进行控制:

- 两个 *Address* 参数，其中将显示这两个地址之间的所有内容,

  或

- 一个 *Address* 参数指定显示起始地址以及一个 *Count* 参数指定从 *Address* 开始显示的行数。

在名称之前加上一个 &（与符号）指定符号地址。地址可以是其他地址和 +（加号）、-（减号）以及 *（间接乘法）运算组成的表达式。任何包含在括号中间的表达式都将认为是一个地址。

- 显示内存的格式是由 *Mode* 参数控制的。*Mode* 参数的缺省值为当前方式。*Mode* 参数的初始值为 **X**。可能存在的方式包括:

b 打印八进制的字节。
c 按字符打印一个字节。
d 按十进制打印一个短字。
D 按十进制打印一个长字。
f 打印单精度的实数。
g 打印一个双精度的实数。
h 按照十六进制打印一个字节。
i 打印机器指令。
lld 打印带符号的 8 字节十进制数。
llu 打印无符号的 8 字节十进制数。
llx 打印无符号的 8 字节十六进制数。
llo 打印无符号的 8 字节八进制数。
o 按八进制打印一个短字。
O 按八进制打印一个长字。
q 打印一个扩展精度的浮点数。
s 打印一个空字节结尾的字符串。
x 按十六进制打印一个短字。
X 按十六进制打印一个长字。

标志:

>*File* 重定向输出到指定的文件。

示例:

1. 要按十六进制显示一个内存内容的长字，且该长字以地址 `0x3fffe460` 起始，请输入:
   ```
   0x3fffe460 / X
   ```
2. 要按照字符显示起始地址由变量 `y` 决定的两个字节内存内容，请输入:
   ```
   &y / 2c
   ```
3. 要显示 FORTRAN 字符串 a_string 中的第六个到第八个元素，请输入:
   ```
   &a_string + 5, &a_string + 7/c
   ```

请参阅《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『测试存储器地址』。

down 子命令

down [*Count*]

down 子命令将当前函数在堆栈中向下移动 *Count* 级。当前函数用来解析名称。*Count* 参数的缺省值为 1。

示例:

1. 要在堆栈中向下移动一级，请输入:
   ```
   down
   ```
2. 要在堆栈中向下移动三级，请输入:
   ```
   down 3
   ```

请参阅 **up** 子命令、**where** 子命令和《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『显示堆栈跟踪』。

dump 子命令

dump [*Procedure* | "**PATTERN**"] [**>***File*]

dump 子命令显示指定过程中的所有变量以及匹配指定模式的那些变量的名称和值。如果 *Procedure* 参数为句点（`.`），则所有活动变量都将显示。如果 *Procedure* 和 "**PATTERN**" 参数都未指定，则使用当前过程。"**PATTERN**" 参数为带有 `*`、`?` 和 `[]` 元字符的通配符表达式。当使用 "**PATTERN**" 时，它将显示全局空间中所有匹配的符号（来自所有过程）。如果使用了 **>***File* 标志，输出将会重定向到指定的文件。

标志:

>*File* 重定向输出到指定的文件。

示例:

1. 要显示当前过程中的变量的名称和值，请输入:
   ```
   dump
   ```
2. 要显示过程 **add_count** 中的变量的名称和值，请输入:
   ```
   dump add_count
   ```
3. 要显示以字符 `s` 开头的变量的名称和值，请输入:
   ```
   dump "s*"
   ```

4. 要将当前过程中的变量的名称和值重定向输出到 **var.list** 文件中，请输入:

```
dump > var.list
```

请参阅《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『显示和修改变量』。

edit 子命令

edit [*Procedure* | *File*]

edit 子命令为指定文件启动编辑器。文件可以通过 *File* 参数或者指定 *Procedure* 参数来指定，其中编辑器将启动并打开包含该过程的文件中。如果没有指定文件，则编辑器将会根据当前的源文件来调用。缺省值为 **vi** 编辑器。通过将 **EDITOR** 环境变量重新设置成您想要的编辑器名称来覆盖缺省值。

示例:

1. 要编辑当前源文件，请输入:

```
edit
```

2. 要编辑 `main.c` 文件，请输入:

```
edit main.c
```

3. 要编辑包含在 `do_count()` 过程中的文件，请输入:

```
edit do_count
```

请参阅 **list** 子命令、**vi** 或者 **vedit** 命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『更改当前文件或进程』和『显示当前文件』。

enable 子命令

enable { *Number* ... **all** }

enable 子命令启用与调试事件关联的跟踪和停止。要启用的跟踪和停止可通过 *Number* 参数指定，或者可使用标志 **all** 来启用所有跟踪和停止。使用 **status** 子命令来显示由 dbx 调试程序将其和跟踪或停止关联的事件号。

标志:

all 删除所有的跟踪和停止。

示例:

1. 要从应用程序启用所有跟踪和停止，请输入:

```
enable all
```

2. 要启用事件号 4 的跟踪和停止，请输入:

```
enable 4
```

有关更多信息，请参阅第 21 页的『disable 子命令』、第 20 页的『delete 子命令』、第 43 页的『status 子命令』。

另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『设置和删除断点』。

fd 子命令

fd [**raw**] [*start* [*end*]]

fd 子命令显示文件描述符信息。使用 **raw** 选项可导致输出以原始十六进制格式显示。其他可选参数包括 *start* 和 *end* 下标。如果未给出下标，则会显示有关所有可用文件描述符的信息。使用一个下标会显示单个文件描述符，使用两个下标则会显示一个范围值。

示例:

1. 要以十六进制查看有关所有文件描述符的信息，请输入:

   ```
   fd raw
   ```

2. 要在 3 到 5 的范围内查看有关文件描述符的信息，请输入:

   ```
   fd 3 5
   ```

file 子命令

file [*File*]

file 子命令将当前的源文件更改为 *File* 参数指定的文件；它不会写入源文件。*File* 参数可以向文件指定完整的路径名。如果 *File* 参数没有指定路径，则 **dbx** 程序通过搜索使用路径试图找到该文件。如果没有指定 *File* 参数，则 **file** 子命令将会显示当前的源文件名。如果路径已知，则 **file** 子命令还显示文件的完整或者相对路径名。

示例:

1. 要将当前源文件更改为 `main.c` 文件，请输入:

   ```
   file main.c
   ```

2. 要显示当前源文件的名称，请输入:

   ```
   文件 file
   ```

请参阅 **func** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『更改当前文件或进程』和『显示当前文件』。

frame 子命令

frame [*num*]

frame 子命令将当前函数更改为对应于指定堆栈帧号 *num* 的函数。当前函数用来解析名称。堆栈帧的编号从当前活动函数的堆栈帧开始（当前活动的函数帧总是编号为 `0`）。如果存在 *n* 个帧，则 **main** 函数的帧将编号为 `n-1`。如果未指定帧号，则显示有关与当前帧关联的函数的信息。

示例:

1. 要移动到帧号 `2`，请输入:

   ```
   frame 2
   ```

2. 要显示堆栈上的当前函数，请输入:

   ```
   frame
   ```

请参阅 **up** 和 **down** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『更改当前文件或过程』以及『显示堆栈跟踪』。

func 子命令

func [*Procedure*]

func 子命令将当前函数更改为由参数 *Procedure* 指定的过程或函数。如果没有指定 *Procedure* 参数，则缺省值为显示当前的函数。更改当前函数将当前源文件更改为包含新函数的文件；当前名称解析的作用域也发生更改。

示例:

1. 要将当前的函数更改为 do_count 过程，请输入:

   ```
   func do_count
   ```

2. 要显示当前函数的名称，请输入:

   ```
   func
   ```

请参阅 **file** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『更改当前文件或进程』。

goto 子命令

goto *SourceLine*

goto 子命令使源行成为下一个要运行的行。通常指定的源行必须与当前行在同一函数中。要重设该约束，使用 **set** 子命令并带有 **$unsafegoto** 标志。

示例: 要使下一个执行的行为第六行，请输入:

```
goto 6
```

请参阅 **cont** 子命令、**gotoi** 子命令、以及**set** 子命令。

gotoi 子命令

gotoi *Address*

gotoi 子命令修改程序计数器地址为 *Address* 参数指定的地址。

示例: 要将程序计数器地址更改为 0x100002b4，请输入:

```
gotoi 0x100002b4
```

请参阅 **goto** 子命令。

handler 子命令

handler { **atfork** | **cancel_cleanup** [**all** | *pthread id*] }

handler 子命令显示有关使用 **pthread_atfork** 注册的 atfork 处理程序或使用 **pthread_cleanup_push** 注册的 cancellation cleanup 处理程序的信息。如果使用 *atfork* 选项，则会显示注册为 pre、parent 和 child atfork 处理程序的例程的名称（在兼容非 POSIX 的 atfork 处理程序情况下，还附带各自的参数）。**cancel_cleanup** 选项会显示所有已注册的 cancellation cleanup 处理程序，还附带可选的 *pthread id* 参数以指定特定的 pthread 或 **all** 参数以指定所有 pthread。如果未给出任何参数，则会显示当前 pthread 的 cancellation cleanup 处理程序（如果存在）。

示例:

1. 要查看有关所有已注册 atfork 处理程序的信息，请输入:

   ```
   handler atfork
   ```

2. 要查看有关当前 pthread 的任何已注册 cancellation cleanup 处理程序的信息，请输入:

   ```
   handler cancel_cleanup
   ```

3. 要查看有关称为 $t2 的 pthread 对象的任何已注册 cancellation cleanup 处理程序的信息，请输入:

```
handler cancel_cleanup 2
```

help 子命令

help [*Subcommand* | *Topic*]

根据您指定的参数 **help** 子命令显示 **dbx** 子命令或主题的帮助信息。输入 **help** 子命令并带有 *Subcommand* 参数显示语法语句和指定子命令的描述。输入 **help** 子命令并带有 *Topic* 参数显示指定主题的详细描述。您无需对 **help** 子命令提供完整的主题字符串。如果您提供从主题开头部分开始的子串，**dbx** 程序可识别该主题。以下是可用的主题:

startup	列出 **dbx** 启动选项列表。
execution	列出和程序执行相关的 **dbx** 子命令。
breakpoints	列出和断点和跟踪相关的 **dbx** 子命令。
files	列出用于访问源文件的 **dbx** 子命令。
data	列出用于访问程序变量和数据的 **dbx** 子命令。
machine	列出用于机器级别的调试的 **dbx** 子命令描述。
environment	列出用于 **dbx** 的配置和环境设置的 **dbx** 子命令。
threads	列出 **dbx** 子命令获得线程相关的对象。
expressions	描述 **dbx** 表达式语法和运算符。
scope	描述 **dbx** 如何解析不同于作用域的名称。
set_variables	列出 **dbx** 带有用法说明的调试变量。
usage	列出带有简单描述的 **dbx** 公共子命令。

示例:

1. 要列出所有可用的 **dbx** 子命令和主题，请输入:

   ```
   帮助
   ```

2. 要显示 **dbx** 子命令 **list** 的描述，请输入:

   ```
   help list
   ```

3. 要显示 **dbx** 命令的主题 **set_variables** 的描述，请输入:

   ```
   help set_variables
   ```

ignore 子命令

ignore [*SignalNumber* | *SignalName*]

ignore 子命令在信号送到应用程序之前停止指定信号的捕捉。该子命令在调试应用程序处理例如中断之类的信号时非常有用。

要捕捉的信号可以通过以下任意一项指定:

- *SignalNumber* 参数指定信号数
- *SignalName* 参数指定信号名称

信号名不区分大小写。前缀 **SIG** 为可选。

如果既没有指定 *SignalNumber* 也没有指定 *SignalName* 参数，则缺省情况下将捕捉除了 **SIGHUP**、**SIGCLD**、**SIGALRM** 以及 **SIGKILL** 以外的所有信号。如果 **SIGTRAP** 信号来自调试器以外的进程，则 **dbx** 调试程序无法忽略该信号。如果没有指定参数，则将会显示当前所有忽略的信号的列表。

示例: 要使 **dbx** 忽略发送到应用程序的警报时钟超时信号，请输入:

```
ignore alrm
```

请参阅 **catch** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『处理信号』。

kthread 子命令

kthread [**raw**] [**info** | **ru**] [*tid*]

kthread 子命令显示有关内核线程的信息。使用 **raw** 选项将导致所有输出均以十六进制显示，而无论其显示格式是否更适于人类阅读。如果不使用参数，则会打印有关所有内核线程的摘要信息。如果提供数字的线程标识，则会导致 dbx 显示单线程上的信息。**info** 选项从用户线程结构产生有关线程的更详细输出。使用 **ru** 选项会显示 `ti_ru` 数据成员，它包含了资源使用信息。

有关用户线程的更多信息，请参阅第 48 页的『thread 子命令』。

示例:

1. 要查找有关当前正在运行的线程的信息，必须首先在命令行输入以下内容以获取有关所有线程的信息。

   ```
   kthread
   ```

   就在 dbx 停止进程前运行（或可运行）的线程标以星号。根据输出选择正确的线程标识并输入:

   ```
   kthread info tid
   ```

2. 要以十六进制查看有关所有线程的资源信息，请输入:

   ```
   kthread raw ru
   ```

list 子命令

list [*Procedure* | *SourceLine-Expression* [**,***SourceLine-Expression*]]

list 子命令显示指定数目的源文件的行。行数由以下两种方法之一来指定:

- 通过使用 *Procedure* 参数指定一个过程。

在这种情况下，**list** 子命令显示从指定过程开始前的几行开始直到列表窗口已满。

- 通过使用 *SourceLine-Expression* 参数指定源文件开始和结束行。

SourceLine-Expression 参数由一个有效的行号，加上可选的 +（加号）或 -（减号），以及一个整数组成。另外，*SourceLine* $（美元符号）可以用来表示当前行号；*SourceLine* @ （at 符号）可以用来表示要列出的下一行的行号。

所有在第一个指定的行号和第二个指定的行号之间（包括这两行）的行都将被显示出来。

如果没指定第二个源行，则只打印第一行。

如果 **list** 子命令没有指定参数，则将从当前源文件的行开始，打印 $listwindow 所指定的行数。

要修改列表行数的缺省值，可以设置特定的调试程序变量，*$listwindow*，为您想要的行数。*$listwindow* 的初始值为 10。

示例:

1. 要列出当前文件中第 1 到第 10 行，请输入:

   ```
   list 1,10
   ```

2. 要列出 10 行，或者 $listwindow 行的 main 程序，请输入:

```
list main
```

3. 要以当前行为中间行，列出前后共 11 行，请输入:

```
list $-5,$+5
```

4. 您可以在 *SourceLineExpression* 表达式中使用包括加法和减法在内的简单整数表达式。例如:

```
(dbx) list $
4 {

(dbx) list 5
5 char i = '4';

(dbx) list sub
23 char *sub(s,a,k)
24 int a;
25 enum status k;  .  .  .

(dbx) move
25
(dbx) list @ -2
23 char *sub(s,a,k)
```

请参阅 **edit** 子命令、**listi** 子命令以及 **move** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『显示当前文件』。

listi 子命令

listi [*Procedure* | **at** *SourceLine* | *Address* [**,** *Address*]]

listi 子命令显示源文件指定的指令集合。显示的指令可以通过以下方法指定:

- 提供 *Procedure* 参数，其中 **listi** 子命令将从指定过程开始列出指令列表，直到列表窗口填满为止。
- 使用 **at** *SourceLine* 标志，其中 **listi** 从指定行开始显示指令列表，直到列表窗口被填满为止。*SourceLine* 变量可指定为整数，也可指定为后跟一个 :（冒号）和一个整数的文件名字符串。
- 通过 *Address* 参数来指定开始和结束的地址，则两个地址之间（包括这两个地址）所有的指令都将显示。

如果 **listi** 子命令没有使用标志或参数，则显示下一个 **$listwindow** 指令。要修改当前列表窗口的大小，可以使用 **set $listwindow=***Value* 子命令。

反汇编方式: **dbx** 程序可反汇编 POWER 系列或 PowerPC® 体系结构的指令。在缺省方式下，**dbx** 程序显示正在运行的结构的指令。

dbx 命令中的 **set** 子命令的 **$instructionset** 以及 **$mnemonics** 变量在允许您覆盖缺省的反汇编方式。要获得更多的信息，请参阅 **dbx** 命令中的 **set** 子命令。

标志:

SourceLine　　指定了列表的开始命令行。

示例:

1. 要列出下 10 条或 $listwindow 条指令，请输入:

```
listi
```

2. 要列出第 10 行的机器指令，请输入

```
listi at 10
```

3. 要列出 sample.c 文件中源代码第 5 行的机器指令，请输入:

```
listi at "sample.c":5
```

4. 要列出地址位于 0x10000400 和 0x10000420之间的指令，请输入:

```
listi 0x10000400, 0x10000420
```

请参阅 **list** 子命令以及 **set** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『使用 dbx 进行机器级别的调试』。

malloc 子命令

malloc [> *File*]

不带选项的 **malloc** 子命令会打印已启用选项和分配策略的列表以及进程启动以来 malloc 使用的统计摘要。

malloc [**allocation** [{ *address* | *size* | *heap* | *pid* | *tid* | *time* } { ″<″ | ″==″ | ″>″ ″!=″] } *Value*]] [> *File*]

malloc 子命令的 **allocation** 选项显示进程当前保留的所有分配的已排序列表。如果使用可选的 attribute RELOP value 参数，则允许对活动分配的更详细选择。

malloc [**freespace** [{ *address* | *size* | *heap* } { ″<″ | ″==″ | ″>″ | ″!=″] } *Value*]] [> *File*]

malloc 子命令的 **freespace** 选项显示进程堆中可用的所有可用空间的已排序列表。如果使用可选的 attribute RELOP value 参数，则允许对可用空间节点的更详细选择。

标志:

> *File* 重定向输出到指定的文件。

有关更多信息，请参阅《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『使用 malloc 子系统分配系统内存』。

map 子命令

map { [*Format*] [**entry** *ModuleNumber* [, *ModuleNumber*] | *Address* | *SymbolName*] [**for $t***threadnumber*] [**>** *File*] }

map 子命令显示应用程序装入部分的特征。该信息可以包括每个装入模块的模块名称、成员名、文本起点、文本终点、文本长度、数据起点、数据终点、数据长度、TLS 数据起点、TLS 数据终点、TLS 数据长度以及文件描述符。可用以下方式指定要显示的项:

- 通过使用 *ModuleNumber* 参数指定单个项。
- 通过使用两个逗号分隔的 *ModuleNumber* 参数指定某个范围的项。
- 通过使用 *Address* 参数向装入模块指定要解析的地址。
- 通过使用 *SymbolName* 参数向装入模块指定要解析的符号名称。

如果不带以上任何说明调用 map 子命令，则该子命令显示应用程序所有装入部分的信息。

Format 参数指定装入模块描述的输出方式。以下表包含 *Format* 参数的可能值:

abbr 指定简略输出方式，该方式包含每个装入模块的包含项编号、模块名称和该模块的可选成员名的单行。

normal 指定常规输出方式，该方式包含每个装入模块的项编号、模块名称、成员名、文本起点、文本长度、数据起点、数据长度和文件描述符。如果装入模块有 TLS 数据，则 TLS 数据起点和 TLS 数据长度也将显示。

raw 指定原始输出方式，该方式由每个模块的包含以下空格分隔字段的无格式单行组成：项编号、带有可选成员名的模块名称、文本起点、文本终点、文本长度、数据起点、数据终点、数据长度和文件描述符。如果装入模块有 TLS 数据，则 TLS 数据起点、TLS 数据终点和 TLS 数据长度也将显示。

verbose 指定详细输出方式，该方式包含每个装入模块的项编号、模块名称、成员名、文本起点、文本终点、文本长度、数据起点、数据终点、数据长度和文件描述符。如果装入模块有 TLS 数据，则 TLS 数据起点、TLS 数据终点和 TLS 数据长度也将显示。

如果未指定 *Format* 参数，则 DBX 使用 **$mapformat** 内部变量的值。如果未指定 *Format* 参数且未设置 **$mapformat**，则 DBX 以正常方式显示装入模块信息。

如果装入模块有 TLS 数据，则指定线程的 TLS 数据信息会显示。如果未指定任何线程，则使用当前线程。

标志:

> *File* 重定向输出到指定的文件。
entry *ModuleNumber* [, *ModuleNumber*] 指定要显示的模块或模块范围。
for $t *threadnumber* 指定线程号。

示例:

1. 要以简略方式列出所有已装入模块，请输入以下命令:
   ```
   map abbr
   ```
2. 要以详细方式列出装入模块 3 到 5，请输入以下命令:
   ```
   map verbose entry 3,5
   ```
3. 要列出包含地址 0x20001000 的装入模块，请输入以下命令:
   ```
   map 0x20001000
   ```
4. 要列出包含变量 foo 的装入模块，请输入以下命令:
   ```
   map foo
   ```
5. 要以常规方式列出线程 2 的装入模块（包含各模块的 TLS 数据信息），请输入以下命令:
   ```
   map normal for $t2
   ```

有关更多信息，请参阅 **$mapformat** 内部变量。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『使用 dbx 进行机器级别的调试』。

move 子命令

move *SourceLine*

move 子命令将下一个显示行更换为由 *SourceLine* 参数指定的行。该子命令修改 @ （at 符号）变量的值。

SourceLine 变量可以通过一个整数或者代表文件名的字符串后跟 :（冒号）和一个整数的方法指定。

示例:

1. 要想将下一行修改为第 12 行，请输入:
   ```
   move 12
   ```
2. 要将下一行修改为文件 sample.c 的第 5 行，请输入:
   ```
   move "sample.c":5
   ```

请参阅 **list** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『显示当前文件』。

multproc 子命令

multproc [**on** | **parent** | **child** | **off**]

multproc 子命令指定 **dbx** 调试程序在创建派生或者越权进程时的行为。**on** 标志用来指定一个新的 **dbx** 会话将会被创建以调试派生进程的子路径。原先的 **dbx** 将会继续调试原来的父路径。**parent** 以及 **child** 标志用来指定派生进程的单独路径。除了 **off** 的所有标志都能使 **dbx** 跟随越权进程。**off** 标志禁用多进程调试。如果没有指定标志，则 **multproc** 子命令返回当前调试的多进程的状态。

dbx 程序使用 Xwindows 来进行多进程调试。**dbx** 程序将会打开足够多的窗口供多进程调试使用。每个子窗口的标题为子进程 ID（pid）。要在进程之间切换，使用 Xwindows 处理技术来激活 dbx 会话显示的窗口。如果系统不支持 Xwindows，则在调试派生进程时将会出现一个警告信息而 dbx 程序继续仅调试父进程。多进程调试也可能因如下原因失败:

- **dbx** 程序没有运行在 Xwindows 环境下。
- Xwindows 已经运行但是 **dbx** 通用 **$xdisplay** 变量没有设置一个有效的显示名。**$xdisplay** 变量被初始化成 shell 的 **DISPLAY** 环境变量。**set** *名称* = *表达式* **dbx** 子命令可以用来修改显示名称的值。
- **/tmp** 目录不允许对调试程序进行读写访问。**dbx** 程序要求在该目录中有小量空间控制 Xwindow 环境。
- 该系统没有足够的资源满足一个新的 Xwindow 窗口。

如果 **$xdisplay** 设置成远程显示，用户可能看不见新创建的 Xwindow。如果 **$xdisplay** 设置不正确，则 Xwindows 或者其他的系统资源将会报告发生错误的原因。

dbx 程序不能区分不同类型错误的区别，但是将会在子命令失败时发送如下的信息:

```
Warning: dbx subcommand multiproc fails. dbx
continued with multproc disabled.
```

新创建窗口的用户定义的配置可以在 **.Xdefaults** 文件中由 **dbx_term** 应用程序名定义。

标志:

on 启用多进程调试。
off 禁用多进程调试。

示例:

1. 要检查当前多进程的调试状态，请输入:

   ```
   multproc
   ```

2. 要启用多进程调试，请输入:

   ```
   multproc on
   ```

3. 要禁用多进程调试，请输入:

   ```
   multproc off
   ```

请参阅 **screen** 子命令以及 **fork** 子例程。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『多个进程中的调试程序』。

mutex 子命令

mutex [**lock** | **unlock** | **thnum** | **utid** | *MutexNumber ...*]

mutex 子命令显示关于互斥的信息。如果给定 *MutexNumber* 参数，则 **mutex** 子命令将显示指定互斥的信息。如果没有指定标志或者参数，则 **mutex** 子命令将会显示所有的互斥信息。

每个互斥信息列出如下:

```
mutex          指定互斥的符号名，遵循 $m MutexNumber 格式。
type           指出互斥类型: non-rec（非递归），recursi（递归）或 fast（快速）。
obj_addr       指出互斥的内存地址。
lock           指出互斥的锁定状态: yes 如果互斥已经锁定，no 如果没有锁定。
owner          如果锁定互斥，指出拥有该互斥的用户线程的符号名。
blockers       列出在互斥变量中锁定的用户线程。
```

注: **dbx** 调试程序的 **print** 子命令识别互斥符号名，然后能用来显示相应对象的状态。

标志:

lock 显示关于锁定互斥的信息。
unlock 显示关于未锁定互斥的信息。
thnum 显示特定线程的所有互斥的信息。
utid 显示用户线程 id 与给定用户线程 id 符合的用户线程的所有互斥信息。

示例:

1. 要显示所有互斥的信息，请输入:

   ```
   mutex
   ```

2. 要显示所有锁定的互斥的信息，请输入:

   ```
   mutex lock
   ```

3. 要显示互斥号为 4、5、6的信息，请输入:

   ```
   mutex 4 5 6
   ```

   输出类似于:

   ```
   mutex   obj_addr         type     lock owner  blockers
   $m4    0x20003274        non-rec   no
   $m5    0x20003280        recursi   no
   $m6    0x2000328a        fast      no
   ```

4. 要显示所有线程 1 的互斥信息，请输入:

   ```
   mutex thnum 1
   ```

5. 要显示线程号为 0x0001 的线程拥有的所有互斥信息，请输入:

   ```
   mutex utid 0x0001
   ```

请参阅 **attribute** 子命令、**condition** 子命令、**print** 子命令以及 **thread** 子命令。

另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『使用互斥对象』。

next 子命令

next [*Number*]

next 子命令运行应用程序到下一行。*Number* 参数指定 **next** 子命令运行的次数。如果没有指定 *Number* 参数，则 **next** 只运行一次。

如果在多线程应用程序中使用 **next** 子命令，则所有用户线程都在该操作期间运行，但该程序仍继续执行，直到正在运行的线程到达指定的源程序行。如果您希望只单步执行正在运行的线程，请使用 **set** 子命令设置变量 **$hold_next**。由于运行中的线程可能等待某个阻塞的线程拥有的锁，因此设置该变量将可能导致死锁。

示例:

1. 要继续执行到下一源行，请输入:

   ```
   next
   ```

2. 要继续执行到当前源行后的第三行，请输入:

   ```
   next 3
   ```

请参阅 **cont** 子命令、**goto** 子命令、**nexti** 子命令、**set** 子命令以及 **step** 子命令。

nexti 子命令

nexti [*Number*]

nexti 子命令运行应用程序到下一个指令。*Number* 参数指定 **nexti** 子命令运行的次数。如果没有指定 *Number* 参数，则 **nexti** 只运行一次。

如果在多线程应用程序中使用 **nexti** 子命令，则所有用户线程都在该操作期间运行，但该程序仍继续执行，直到正在运行的线程到达指定的机器指令。如果您想单步运行正在运行的线程，请使用 **set** 子命令对 **$hold_next** 变量进行设置。由于运行中的线程可能等待某个阻塞的线程拥有的锁，因此设置该变量将可能导致死锁。

示例:

1. 要继续执行到下一个机器指令，请输入:

   ```
   nexti
   ```

2. 要继续执行到从当前机器指令起的第三个机器指令，请输入:

   ```
   nexti 3
   ```

请参阅 **gotoi** 子命令、**next** 子命令、**set** 子命令以及 **stepi** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『运行机器水平中的程序』。

onceblock 子命令

onceblock [**uninit** | **done**]

onceblock 子命令显示有关使用 **pthread_once** 例程注册的初始化代码块的信息。如果不带参数，则会显示有关所有已注册 once 块的信息。可选的 **uninit** 和 **done** 标志仅分别显示未执行或已执行的 once 块，而提供数字 once 标识则显示单个 once 块的信息。

注: 要在调试活动进程时使用 **onceblock** 子命令，应该将环境变量 AIXTHREAD_ONCE_DEBUG 设置为等于 ON。类似的情况，在调试核心文件时，如果进程运行时未启用 said 变量，则 **onceblock** 子命令将无法获取任何信息。

示例:

1. 要了解是否尚未执行任何 once 块，请输入:

   ```
   onceblock uninit
   ```

plugin 子命令

plugin [*Name* [*Command*]]

plugin 子命令将 *Command* 参数指定的命令传递到 *Name* 参数指定的插件。如果未指定参数，将显示所有可用插件的名称。

示例:

1. 要列出所有可用插件，请输入:

   ```
   plugin
   ```

2. 要调用名称为 “sample” 的插件的子命令 “help” ，请输入:

   ```
   plugin sample help
   ```

3. 要调用名称为 “xyz” 的插件的子命令 “interpret 0x20000688” ，请输入:

   ```
   plugin xyz interpret 0x20000688
   ```

请参阅 **pluginload** 子命令和 **pluginunload** 子命令。另见《*AIX 5L V5.3 通用编程概念*》中的 『开发 dbx 插件框架』。

pluginload 子命令

pluginload *File*

pluginload 子命令装入 *File* 参数指定的插件。*File* 参数应指定该插件的路径。

示例: 要装入位于 “/home/user/dbx_plugins/libdbx_sample.so” 的，名称为 “sample” 的插件，请输入:

```
pluginload /home/user/dbx_plugins/libdbx_sample.so
```

请参阅 **plugin** 子命令和 **pluginunload** 子命令。另见《*AIX 5L V5.3 通用编程概念*》中的 『开发 dbx 插件框架』。

pluginunload 子命令

pluginunload *Name*

pluginunload 子命令卸装 *Name* 参数指定的插件。

示例: 要卸装名称为 “sample” 的插件，请输入:

```
pluginunload sample
```

请参阅 **plugin** 子命令和 **pluginload** 子命令。另见《*AIX 5L V5.3 通用编程概念*》中的 『开发 dbx 插件框架』。

print 子命令

print *表达式 ...*

print *Procedure* ([*Parameters*])

print 子命令打印执行以下任务:

- 打印由 *Expression* 参数指定的表达式列表的值。
- 执行 *Procedure* 参数指定的过程并将过程的返回值打印出来。包括的参数传递给过程。

示例:

1. 要显示 x 以及 y 左移两位的值，请输入:

   ```
   print x, y << 2
   ```

2. 要显示带参数 0 调用 sbrk 例程的返回值，请输入:

```
print sbrk(0)
```

请参阅 **assign** 子命令、**call** 子命令以及 **set** 子命令。

proc 子命令

proc [**raw**] [**cred** | **cru** | **ru** | **sigflags** | **signal**]

proc 子命令显示有关进程的信息。如果使用 **raw** 选项，则会导致输出以原始十六进制显示，而不是以更人类可读的方式解释值。如果不带其他参数使用 **proc** 子命令，则会输出用户进程数据结构中存储的有关进程的一般信息。**cred** 选项显示 pi_cred 数据成员的内容，其中该数据成员描述进程的凭证。**cru** 和 **ru** 选项分别显示数据成员 pi_cru 和 pi_ru，包含资源使用信息。**sigflags** 和 **signal** 选项显示与当前信号状态和已注册信号处理程序相关的信息，这些信息包含在 pi_sigflags 和 pi_signal 数据成员中。

示例:

1. 要以原始十六进制查看当前进程（或核心文件）的资源使用信息，请输入:

   ```
   proc raw ru
   ```

2. 要查看信号处理程序信息，请输入:

   ```
   proc signal
   ```

prompt 子命令

prompt [*"String"*]

prompt 子命令将 **dbx** 命令的提示符修改为 *String* 参数指定的字符串。

示例: 要将提示符更改为 dbx>，请输入:

```
prompt "dbx>"
```

请参阅《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『定义新的 dbx 提示』。

quit 子命令

quit

quit 子命令结束所有在 **dbx** 调试对话中的进程。

请参阅 **detach** 子命令。

registers 子命令

registers [**>***File*]

registers 子命令显示通用寄存器、系统控制寄存器、浮点数寄存器、向量寄存器以及当前指令寄存器的值。

- 通用寄存器由符号 **$r***Number* 变量表示，其中 *Number* 参数指定寄存器的序号。

  注: 寄存器的值可以设置成十六进制数 **0xdeadbeef**。十六进制数 **0xdeadbeef** 是一个在进程初始化时就赋到通用寄存器中的初始值。

- 浮点寄存器由 **$fr***Number* 变量表示。按照缺省设置，浮点寄存器不会被显示出来。要显示浮点寄存器，使用 **unset $noflregs dbx** 子命令。
- 浮点寄存器由 **$vr***Number* 变量表示。**$novregs** 内部变量控制是否显示向量寄存器。将按缺省值设置 **$novregs** 变量，不显示向量寄存器。当未设置 **$novregs**，且向量寄存器有效时（在能处理向量的处理器上调试程序，或者分析包含向量寄存器状态的核心文件），将显示所有的向量寄存器（vr0–vr31、vrsave 和

vscr）。还可以按类型引用向量寄存器。例如，**$vrNf**（浮点）、**$vrNs**（短整型）和 **$vrNc**（字符型）向量寄存器变量可以与 **print** 和 **assign** 子命令一起用来按类型显示和设置向量寄存器。

注：如果当前线程正处于内核方式，**registers** 子命令将不能显示寄存器。

标志:

>*File* 重定向输出到指定的文件。

请参阅 **set** 子命令和 **unset** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『使用机器寄存器』。

rerun 子命令

rerun [*Arguments*] [**<** *File*] [**>** *File*] [**> >** *File*] [**2>** *File*] [**2> >** *File*] [**>&** *File*] [**> >&** *File*]

rerun 子命令开始对象文件的执行。*Arguments* 参数将会作为一个命令行参数使用。如果没有指定 *Arguments* 参数，最近一次 **run** 或者 **rerun** 子命令使用的参数将会被再次使用。

标志:

<*File*	将输入重定向使得从 *File* 文件中获得输入。
>*File*	重定向输出到 *File* 文件中。
> >*File*	添加重定向输出到 *File* 中。
2>*File*	将标准错误重定向到 *File* 中。
2> >*File*	添加重定向标准错误到 *File* 中。
>&*File*	将输出和标准错误重定向到 *File* 中。
> >&*File*	添加输出和标准错误到 *File* 中。

请参阅 **run** 子命令。

resource 子命令

resource { **owner** | **waiter** } [**all** | *pthread id*]

resource 子命令显示有关当前 pthreads 保留或正在等待哪些资源的信息。第一个参数（必需）指示您是否有兴趣查看拥有资源或正在等待资源的 pthread。第二个参数可用于指示所有 pthread 或某个特定 pthread。如果未给出任何参数，则仅显示与当前 pthread 相关的信息（如果适用）。

注：**resource** 子命令仅可用于调试在多个调试环境变量设置为 ON 的情况下运行的进程。这些变量包括 AIXTHREAD_MUTEX_DEBUG、AIXTHREAD_COND_DEBUG、AIXTHREAD_RWLOCK_DEBUG、AIXTHREAD_READ_OWNER 和 AIXTHREAD_WAITLIST_DEBUG。如果在调试活动的进程时或在调试器核心文件生成时未开启这些变量，则 **resource** 子命令只能检索到很少的信息，或者根本没有信息。由于使用这些功能可能降低性能，所以建议仅为调试用途才将其激活。

示例:

1. 要确认当前 pthread 是否占用任何资源，请输入:

   ```
   resource owner
   ```

2. 要查看 pthread 正在等待哪些资源，请输入:

   ```
   resource waiter all
   ```

return 子命令

return [*Procedure*]

return 子命令使应用程序执行直到返回到由 *Procedure* 参数指定的过程。如果没有指定 *Procedure* 参数，执行将会在当前过程返回时停止。

示例:

1. 要继续执行例程的调用，请输入:

   ```
   return
   ```

2. 要继续执行至 `main` 过程，请输入:

   ```
   return main
   ```

rwlock 子命令

rwlock [read | write | *RwlockNumber*....]

rwlock 子命令显示 rwlocks 的信息。如果给定 *RwlockNumber* 参数，**rwlock** 子命令将会显示指定 rwlocks 的信息。如果没有指定标志或者参数，**rwlock** 子命令将会显示所有 rwlocks 的信息。

每个 **rwlock** 的信息如下:

`rwl`	设定 rwlock 的符号名称，格式为 $rw *RwlockNumber*。
`flag_value`	指出标志的值。
`owner`	指出 rwlock 的所有者。
`status`	指出谁拥有该 rwlock。这个值可以是读入（如果由读入者拥有）、写出（如果由写出者拥有）或者空闲（如果空闲）。
`wsleep[#]`	指出写入的线程阻塞。# 说明写入的线程阻塞的总数。
`rsleep[#]`	指出读入的线程阻塞。# 说明读入的线程阻塞的总数。

注: dbx 调试程序的 **print** 子命令识别 rwlock 的符号名称并能用于显示其对应的对象的状态。

标志:

read	显示所有处于读状态的 rwlock 的信息。
write	显示所有处于写状态的 rwlock 的信息。

示例:

1. 要显示所有 rwlock 的信息，请输入:

   ```
   rwlock
   ```

   输出类似于:

   ```
   rwl     flag_value    owner status
   $rwl        1         $t1    write
           rsleeps[     0]:
           wsleeps[     0]:
   ```

2. 要显示处于写状态的所有 rwlock 的信息，请输入:

   ```
   rwlock write
   ```

   输出类似于:

```
rwl     flag_value    owner status
$rwl        1         $t1    write
        rsleeps[     0]:
        wsleeps[     0]:
```

请参阅 **attribute** 子命令、**condition** 子命令、 **mutex** 子命令、**print** 子命令以及 **thread** 子命令。

run 子命令

run [*Arguments*] [**<***File*] [**>***File*] [**> >***File*] [**2>***File*] [**2> >***File*] [**>&***File*] [**> >&***File*]

run 子命令打开对象文件。*Arguments* 参数将会作为一个命令行参数使用。

标志:

<*File*	将输入重定向使得从 *File* 文件中获得输入。
>*File*	重定向输出到 *File* 文件中。
2>*File*	将标准错误重定向到 *File* 中。
> >*File*	添加重定向输出到 *File* 中。
2> >*File*	添加重定向标准错误到 *File* 中。
>&*File*	将输出和标准错误重定向到 *File* 中。
> >&*File*	添加输出和标准错误到 *File* 中。

示例: 要带参数 `blue` 和 `12` 运行应用程序，请输入:

```
run blue 12
```

请参阅 **rerun** 子命令。

screen 子命令

screen

screen 子命令为 **dbx** 命令交互打开一个 Xwindow 窗口。您可以在进程发生的窗口继续操作。

screen 子命令必须在 **dbx** 调试程序运行在一个 Xwindows 环境下时运行。如果 **screen** 子命令运行于非 Xwindow 环境下，**dbx** 程序将会显示一个警告信息并且继续执行调试过程如同没有设定 **screen** 子命令。**screen** 子命令也可能在下述情况下失败:

- **dbx** 程序没有运行在 Xwindows 环境下。
- Xwindows 已经运行但是 **dbx** 通用 **$xdisplay** 变量没有设置一个有效的显示名。**$xdisplay** 变量被初始化成 **DISPLAY** 环境变量的值。**dbx** 子命令 **set** *Name=表达式*修改显示名称的值。
- Xwindows 正在运行，但是 **TERM** 环境变量没有设置成一个有效的命令名称以调用新窗口。
- **/tmp** 目录不允许对调试程序进行读写访问。**dbx** 程序要求在 screen 命令执行时该目录中有小量空间。
- 该系统没有足够的资源满足一个新的 Xwindow 窗口。

dbx 程序不能区分不同类型错误的区别，但是将会在子命令失败时发送如下的信息:

```
Warning: dbx subcommand screen fails. dbx
continues.
```

如果 **$xdisplay** 设置成远程显示，用户可能看不见新创建的 Xwindow。如果 **$xdisplay** 设置不正确，Xwindows 或者其他的系统资源将会报告发生该问题。

新创建窗口的用户定义的配置可以在 **.Xdefaults** 文件中由 **dbx_term** 应用程序名定义。

示例: 要打开一个 **dbx** 命令交互的 Xwindow 窗口，请输入:

```
screen
```

请参阅《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『区分 dbx 输出和程序输出』和 *AIX 5L Version 5.3 AIXwindows Programming Guide* 中的 AIXwindows Overview。

set 子命令

set [*Variable*=*Expression*]

set 子命令为 **dbx** 调试程序变量定义一个值。其值由 *Expression* 参数指定; 而程序变量由 *Variable* 参数指定。变量名不得和正在调试的程序中的使用名称相冲突。在其他命令内部扩展变量至相应的表达式。如果 **set** 子命令没有参数使用，将会显示当前的变量。

以下变量可以用 **set** 子命令设置:

$catchbp
在下一个命令执行期间捕捉断点。

$deferevents
开启延迟事件功能。

$expandunions
显示变体记录或变体联合的每个部分的值。

$frame
使用由 **$frame** 的值指定的地址所指向的堆栈框架，来执行堆栈跟踪和访问局部变量。

$hexchars
按照十六进制值打印字符。

$hexin
按照十六进制解释地址。

$hexints
按照十六进制数值打印整数。

$hexstrings
按照十六进制打印字符指针。

$hold_next
控制在运行 **cont**、**next**、**nexti** 和 **step** 子命令时获得除了正在运行线程之外的所有线程。由于运行中的线程可能等待某个阻塞的线程拥有的锁，因此设置该变量将可能导致死锁。

$ignoreifhandler
您的程序接收到带已注册处理程序的信号时，请勿停止。

$ignoreload
在您运行 **load**、**unload** 或者 **loadbind** 子例程时不会停止。

$ignorenonbptrap
您的程序遇到非断点捕获指令且具有已注册的 **SIGTRAP** 处理程序，请勿停止。

$instructionset
重设缺省反汇编方式。以下列表包含了 *Expression* 参数可能取的值:

"**default**"
指定 **dbx** 程序运行的结构。

"**com**" 指定 PowerPC 和 POWER 系列体系结构的公共交集方式的指令集。**dbx** 程序缺省设置为基于 POWER 的助记符。

"**pwr**" 指定 POWER 系列体系结构的指令集和助记符。

"**pwrx**" 指定用于 AIX 5.1 和更早版本 POWER 系列体系结构的 POWER2 实施的指令集和助记符。

"**601**" 指定 AIX 5.1 和更早版本的 PowerPC 601 RISC 微处理器的指令集和助记符。

"**603**" 指定 AIX 5.1 和更早版本的 PowerPC 603 RISC 微处理器的指令集和助记符。

"**604**" 指定 PowerPC 604 RISC 微处理器的指令集和助记符。

"**970**" 指定 PowerPC 970 微处理器的指令集和助记符。

"**ppc**" 指定由基于 POWER 体系结构定义的指令集和助记符，除了可选指令。这些指令在所有的基于基于 POWER 的实现的体系中有效，AIX 5.1 和更早版本的 PowerPC 601 RISC 微处理器除外。

"**any**" 指定任何有效的、基于 POWER 的或 POWER 系列系列的指令。对于重叠的指令，缺省助记符为基于 POWER 的助记符。

如果没有设置 *Expression* 参数，**dbx** 将会使用缺省反汇编方式。

$java
设置了该变量时，还将设置以下变量，并将 **dbx** 置为某种方式来调试 Java™ 应用程序。如果取消设置，还将取消设置以下变量:

$ignorenonbptrap
禁止 Java Just-In-Time（JIT）编译器生成的捕获指令的通知。

$listwindow
指定一个函数列表的行数以及不加参数使用 **list** 子命令时的列表的行数。缺省值是 10 行。

$mapaddrs
开始映射地址。取消 **$mapaddrs** 设置将停止地址映射。

$mapformat
指定 **map** 子命令的缺省输出方式。

″**abbr**″ 指定简略输出方式，该方式包含每个装入模块的包含项编号、模块名称和该模块的可选成员名的单行。

″**normal**″
指定常规输出方式，该方式包含每个装入模块的项编号、模块名称、成员名、文本起点、文本长度、数据起点、数据长度和文件描述符。如果装入模块有 TLS 数据，则 TLS 数据起点和 TLS 数据长度也将显示。

″**raw**″ 指定原始输出方式，该方式由每个模块的包含以下空格分隔字段的无格式单行组成: 项编号、带有可选成员名的模块名称、文本起点、文本终点、文本长度、数据起点、数据终点、数据长度和文件描述符。如果装入模块有 TLS 数据，则 TLS 数据起点、TLS 数据终点和 TLS 数据长度也将显示。

″**verbose**″
指定详细输出方式，该方式包含每个装入模块的项编号、模块名称、成员名、文本起点、文本终点、文本长度、数据起点、数据终点、数据长度和文件描述符。如果装入模块有 TLS 数据，则 TLS 数据起点、TLS 数据终点和 TLS 数据长度也将显示。

如果没有设置 *Expression* 参数的值，则 dbx 程序将使用“正常”输出方式。

$mnemonics
修改反汇编时 **dbx** 程序的助记符设置。

″**default**″
指定最接近指定的指令的助记符。

″**pwr**″ 指定 POWER 系列结构的助记符。

″**ppc**″ 指定基于 POWER 的结构中定义的助记符，不包括可选指令。

如果没有设置 *Expression* 参数值 **dbx** 程序将会最接近指定指令集的助记符。

$noargs
省略来自子命令的参数，例如 *where*、*up*、*down* 和 *dump*。

$noflregs
省略 **registers** 子命令中浮点寄存器的显示。

$novregs
省略 **registers** 子命令中向量寄存器的显示。

$octin
解释八进制的地址。

$octints
打印八进制的整数。

$pretty
与 **print** 子命令连用，按照 *pretty printed* 格式显示复杂的 C 及 C++ 数据结构（结构、联合、数组）的值。

″**on**″ 指定值不换行打印、并带有表示值的静态范围的限定名称的美化打印。

″**verbose**″
指定各自行中每个值的恰当的打印并带有表示值的静态范围的限定名称。一个限定名称包含一个由点隔开与值相关外部块的列表。

″**off**″ 指定关闭恰当打印。此为缺省设置。

$repeat
如果没有输入命令则重复前一个命令。

$sigblock
程序的阻塞信号。

$stack_details
显示 **where** 子命令所显示的每个活动函数或过程的帧号和寄存器集。

$stepignore 控制在 **step/tstep** 子命令运行到某一源程序行，并且该行调用无调试信息可用的另一个例程时，**dbx** 命令的行为。该变量允许 **step/tstep** 子命令单步跳过无调试信息可用的大例程。以下列表包含了 *Expression* 参数可能取的值:

"**function**"
执行 **dbx** 命令的 **next/tnext** 子命令的函数。这是缺省值。

"**module**"
如果 **next/tnext** 子命令的函数在无调试信息可用的装入模块（如系统库）中，则执行该函数。

"**none**" 在后台执行 **dbx** 命令的 **stepi/tstepi** 子命令的函数，直到它到达有源程序信息可用的指令。在该点上 **dbx** 将会显示运行到哪儿停止了。

$thcomp 当设置了 **$thcomp** 将会按照压缩格式显示线程命令 **th-** 的显示信息。
$unsafeassign 关闭对 **assign** 语句两端严格的类型检查语句。即使设置了 **$unsafeassign** 变量，**assign** 语句两边也不一定包含不同大小的存储类型。
$unsafebounds 关闭数组的下标检查。
$unsafecall 关闭子例程或者函数调用参数的严格类型检查。
$unsafegoto 关闭 **goto** 子命令目的地检查。
$vardim 指定在打印未知边界的数组时将使用的维度。缺省值是 10。
$xdisplay 指定显示的 Xwindows 的名称，用于 multproc 或者 screen 子命令。缺省值为 shell **DISPLAY** 变量的值。

$unsafe 变量限制了 **dbx** 调试程序在错误检测中的用途。

示例:

1. 要修改缺省的列表的行数为 20，请输入:
   ```
   set $listwindow=20
   ```
2. 要禁用 assign 子命令的类型检查，请输入:
   ```
   set $unsafeassign
   ```
3. 要反汇编 AIX 5.1 和更早版本的 PowerPC 601 RISC 微处理器的机器指令，请输入:
   ```
   set $instructionset="601"
   ```

请参阅 **unset** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『更改带有特殊调试程序变量的打印输出』。

set edit [vi, emacs] 或 set -o [vi, emacs] 子命令

set 子命令带有 **-o** 或者 **edit** 选项可以用来打开某一行编辑方式。如果给定 **set-o vi** 或 **set edit vi** 命令，则您将处于 *vi* 行编辑器的输入方式。如果给了 **set -o emacs** 或者 **set edit emacs** 命令，您将处于 *emacs* 行编辑器的输入方式。

示例:

1. 要打开 vi 行编辑器，请输入:
   ```
   set-o vi
   ```
   或
   ```
   set edit vi
   ```

sh 子命令

sh [*Command*]

sh 子命令将 *Command* 参数指定的命令传递到 shell 执行。**SHELL** 环境变量决定使用哪个 shell。缺省值为 **sh** shell。如果没有指定参数，则将控制权交给 shell。

示例:

1. 要运行 `ls` 命令，请输入:
   ```
   sh ls
   ```
2. 要进入 shell，请输入:
   ```
   sh
   ```
3. 要使用 `SHELL` 环境变量，请输入:
   ```
   sh echo $SHELL
   ```

请参阅《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『从 dbx 运行 Shell 命令』。

skip 子命令

skip [*Number*]

skip 子命令从当前的停止点继续执行程序。数目等于 *Number* 参数的值的断点将会被跳过并将在下一个断点处或者程序结束时停止执行。如果没有指定 *Number* 参数，缺省值为 1。

示例: 要继续执行程序直到遇到第二个断点，请输入:

```
skip 1
```

也可请参阅 **cont** 子命令。

source 子命令

source *File*

source 子命令从 *File* 参数指定的文件读入 **dbx** 子命令。

示例: 要从 `cmdfile` 文件中读 **dbx** 子命令，请输入:

```
source cmdfile
```

请参阅《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『从文件阅读 dbx 子命令』。

status 子命令

status [**more**] [**>***File*]

status 子命令显示用户定义的所有断点、跟踪点和观察点，此外还有剩余线程 **tskip** 计数（使用 **tskip** 子命令设置）。如果指定了 more 参数，则 **status** 子命令还将显示与这些断点、跟踪点和观察点关联的 **dbx** 子命令。**status** 子命令用方括号（[]）括住事件号列出启用的事件，用句点（..）括住事件号列出禁用的事件，并用尖括号（<>）括住事件号列出延迟的事件。

> 标志将 **status** 子命令的输出发送到 *File* 参数中指定的文件中。

标志:

>*File* 重定向输出到 *File*文件中。

示例:

1. 要显示用户定义的所有断点、跟踪点和观察点，以及剩余线程 **tskip** 计数，请输入:

   ```
   status
   ```

   输出类似于:

   ```
   [1] stop at 13
   [2] stop at 14
   .3. stop at 15
   .4. stop at 16
   [5] stop at 17<6> stop at 18 if g > 10
   <7> stop in func

   Remaining tskip counts:
   tskip 2 for $t1
   tskip 1 for $t5
   ```

   在上面的示例输出中，事件 3 和 4 已禁用，事件 6 和 7 已延迟。

2. 要显示用户定义的所有断点、跟踪点和观察点，以及关联的 dbx 子命令，请输入:

   ```
   status more
   ```

   其输出类似于下面的形式:

   ```
   [1] stop at 13
       [1] where
   .2. stop at 14
       [1] where
       [2] registers
   <3> stop at 15 if g > 10
       [1] where; registers
   ```

请参阅 **dbx** 命令的 **addcmd** 子命令、**clear** 子命令、**delete** 子命令、**delcmd** 子命令、**tskip** 子命令、**stop** 子命令，以及 **trace** 子命令。

另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『设置和删除断点』。

step 子命令

step [*Number*]

step 子命令运行应用程序源命令行。通过 *Number* 参数指定执行的行数。如果省略了 *Number* 参数，则缺省值为 1。

如果在多线程应用程序中使用 **step** 子命令，则所有用户线程都在该操作期间运行，但该程序仍继续执行，直到正在运行的线程到达指定的源程序行。如果您希望只单步执行正在运行的线程，请使用 **set** 子命令设置变量 **$hold_next**。由于运行中的线程可能等待某个阻塞的线程拥有的锁，因此设置该变量将可能导致死锁。

> **注:** 使用 **set** 子命令的 **$stepignore** 变量来控制 **step** 子命令的行为。**$stepignore** 变量使 **step** 子命令单步跳过大的无法获得调试信息的例程。

示例:

1. 要继续执行一个源程序行，请输入:

```
step
```

2. 要继续执行五行，请输入:

```
step 5
```

3. 要防止 **dbx** 程序单步执行 **printf** 函数，如下述示例代码所述:

```
60 printf ("hello world \n");
```

请输入:

```
set $stepignore="function"; step
```

请参阅 **cont** 子命令、**goto** 子命令、**next** 子命令、**set** 子命令以及 **stepi** 子命令。

stepi 子命令

stepi [*Number*]

stepi 子命令运行应用程序的指令。通过 *Number* 参数指定执行的指令数。如果省略了 *Number* 参数，则缺省值为 1。

如果在多线程应用程序上使用，**stepi** 子命令只单步执行当前运行的线程。所有其他的线程都将停止。

示例:

1. 要继续执行 1 个机器指令，请输入:

```
stepi
```

2. 要继续指令 5 个机器指令，请输入:

```
stepi 5
```

请参阅 **gotoi** 子命令、**nexti** 子命令和 **step** 子命令。

stop 子命令

stop { [*Variable*] [**at** *SourceLine* | **in** *Procedure* | **on load** [*"ModuleName"*]] [**if** *Condition*]}

stop 子命令在满足特定条件时停止应用程序。应用程序将会在以下条件停止:

- 在使用了 **if** *Condition* 标志的情况下 *Conditoin* 为 “true”。
- 在使用了 **in** *Procedure* 标志的情况下调用了 *Procedure*。
- 在指定了 *Variable* 参数的情况下更改了 *Variable*。
- 在使用了 **at** *SourceLine* 标志的情况下达到了 *SourceLine* 行号。

  SourceLine 变量可以通过一个整数或者代表文件名的字符串后跟 :（冒号）和一个整数的方法指定。
- 在使用了 **on load** 标志并指定了 *ModuleName* 参数的情况下装入或卸装了 *ModuleName* 装入模块。

  可选的 *ModuleName* 变量可指定为单个模块名称或以以下格式与成员名配对的模块名称。

  ModuleName(*MemberName*)
- 在使用了 on load 标志且未指定 *ModuleName* 参数的情况下装入或卸装了任何装入模块。

完成任意命令后，**dbx** 调试程序产生一个消息报告它已作为命令结果产生。该消息包括和您的断点相关的事件的 ID 以及您的命令的解释。解释的语法并不一定和您的命令相同。例如:

```
stop in main
[1] stop in main
stop at 19 if x == 3
[2] stop at "hello.c":19 if x = 3
stop in func
<3> stop in func
stop g
<4> stop g
```

方括号（[]）中的数字为与断点关联的事件标识。**dbx** 调试程序将事件号和每个 **stop** 子命令联系起来。当程序由于某个事件而停止时，该事件标识将与当前行一起显示，以表明哪个事件造成程序停止。尖括号（<>）中的数字为延迟事件的事件标识。延迟事件是没有任何与之关联的断点、跟踪点或观察点的事件，每当输入命令涉及当前未装入内存的符号时，将创建延迟事件。每当对应于常规事件的模块卸载时，在方括号（[]）中显示的常规事件也会转换为延迟事件。每当对应于延迟事件的模块装入内存时，延迟事件将转换为常规事件，并创建对应的断点、跟踪点或观察点。您创建的事件与 **dbx** 创建的内部事件共存，因此事件号可能并不总是连续的。

使用 **status** 子命令来查看这些数字。您可以重定向 **status** 的输出到一个文件当中。使用 **delete** 或 **clear** 子命令来关闭 **stop** 子命令，或使用 **enable** 或 **disable** 子命令。使用 **addcmd** 子命令将 **dbx** 子命令添加到指定的事件号，使用 **delcmd** 从指定的事件号删除关联的 **dbx** 子命令。

在多线程应用程序中，所有用户线程都将在任一用户线程遇到断点时停止。在源文件行和函数中设置的断点将会被任何执行该行和该函数的用户线程遇到，除非您按照下述例 9 的方式指定条件。以下别名自动指定条件:

- **bfth**(*Function*, *ThreadNumber*)
- **blth**(*SourceLine*, *ThreadNumber*)

ThreadNumber 是线程符号名的数字部分，由 **thread** 子命令报告（例如，5 是名为 $t5 的线程的 *ThreadNumber*）。这些别名实际上是宏，它们产生扩展子命令如下:

```
stopi at &Function    if ($running_thread == ThreadNumber)
stop at SourceLine  if ($running_thread == ThreadNumber)
```

标志:

at *SourceLine*	指定行号。
if *Condition*	指定条件，例如 true。
in *Procedure*	指定调用的过程。
on load *ModuleName*	指定要监视的装入模块。

示例:

1. 要在 main 过程中第一个语句处停止，请输入:

   ```
   stop in main
   ```

2. 当 x 变量的值在第 12 的行执行时更改时要停止执行，请输入:

   ```
   stop x at 12
   ```

3. 要在 sample.c 文件的第五行停止执行，请输入:

   ```
   stop at "sample.c":5
   ```

4. 要在 **dbx** 命令每次运行 func1 的子例程时检查 x 的值，请输入:

   ```
   stop in func1 if x = 22
   ```

5. 要检查 **dbx** 命令每次开始运行 func1 函数时 x 的值，请输入:

   ```
   stopi at &func1 if x = 22
   ```

6. 当更改 *Variable* 变量的值时要停止程序，请输入:

```
stop Variable
```

7. 要随时在 *Condition* 变为 true 时停止程序的运行，请输入:

```
stop if (x > y) and (x < 2000)
```

8. 以下示例列出了如何显示活动的事件并将它们删除:

```
status
[1] stop in main
[2] stop at "hello.c":19 if x = 3
delete 1
status
[2] stop at "hello.c":19 if x = 3
clear 19
status
(dbx)
```

delete 命令用事件标识符删除事件。**clear** 命令通过行号删除了断点。

9. 要将断点设置在 func1 的开始处，并且只有由线程 $t5 执行时有效，请输入以下等价命令之一:

```
stopi at &func1 if ($running_thread == 5)
```

或

```
bfth(func1, 5)
```

10. 要在装入或卸装了任何模块的情况下停止程序，请输入:

```
stop on load
```

11. 要在装入或卸装了模块 Module 的情况下停止程序，请输入:

```
stop on load "Module"
```

12. 要在装入或卸装了模块 Module 的成员 Member 的情况下停止程序，请输入:

```
stop on load "Module(Member)"
```

请参阅 **addcmd** 子命令、**clear** 子命令、**delete** 子命令、**delcmd** 子命令、**disable** 子命令、**enable** 子命令、**stopi** 子命令，以及 **trace** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『设置和删除断点』。

stophwp 子命令

stophwp *Address Size*

stophwp 子命令为指定的内存区域设置一个硬件监视点停止。如果区域的内容更改，则程序将停止。

注:

1. **stophwp** 子命令的成功之处在于硬件依赖性。此功能只能在 630 型、PowerPC 机型以及后来的型号中使用。
2. 由于硬件只能设置一个监视点的限制，在试图创建另一个硬件观察点时活动的观察点事件将会在 **stophwp** 和 **tracehwp** 发生冲突。同样的，以前的事件必须在创建一个新的事件之前删除。同样，因为存在一个活动的软件监视点（由一些 **stop** 和 **trace** 子命令生成）取消硬件监视点的性能，这些类型的事件也可以作为创建一个硬件监视点之前必须删除的冲突事件。

示例:

1. 要在起始地址为 0x200004e8 的 4 字节内存区域的内容更改时停止程序，请输入:

```
stophwp 0x200004e8 4
```

请参阅 **tracehwp** 子命令。

stopi 子命令

stopi { [*Address*] [**at** *Address* | **in** *Procedure*] [**if** *Condition*]}

stopi 子命令在指定的位置设置一个停止点:

- **通过 if** *Condition* 标志可以在指定条件为 true 时停止程序。
- 通过 *Address* 参数，当 *Address* 的内容被修改时程序停止运行。
- 通过 **at** *Address* 标志，在指定的地址设置停止点。
- 通过 **in** *Procedure* 标志，程序在调用 *Procedure* 时停止。

标志:

if *Condition*	指定条件，例如 true。
in *Procedure*	指定调用的过程。
at *Address*	指定机器指令地址。

示例:

1. 在地址为 0x100020f0 处停止执行，请输入:

   ```
   stopi at 0x100020f0
   ```

2. 要在地址为 0x100020f0 的内容被修改时停止，请输入:

   ```
   stopi 0x100020f0
   ```

3. 要在地址为 0x200020f0 的内容被线程 $t1 修改时停止，请输入:

   ```
   stopi 0x200020f0 if ($running_thread == 1)
   ```

请参阅 **stop** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『使用 dbx 进行机器水平调试』。

thread 子命令

显示选中的线程: **thread** { [**info**] [**-**] [*ThreadNumber ...*] } | **current** | **run** | **susp** | **term** | **wait**

选中一个单独线程: **thread current** [-] *ThreadNumber*

保持或者释放线程: **thread** { **hold** | **unhold** } [-] [*ThreadNumber ...*]

获得显示选项的帮助: **thread** { **help**}

thread 子命令显示和控制用户线程。

thread 子命令的第一种形式可以两种格式显示信息。如果 **thread** 子命令是 **th**，则会使用第一种格式显示信息。如果 **thread** 子命令是 **th -**，则会按照第二种格式显示信息。如果没有指定参数，则会显示所有用户线程的信息。如果指定了一个或者更多的 *ThreadNumber* 参数，则会显示相关用户线程的信息。**thread** 子命令显示线程时，当前线程行将以 **>** 开头。如果运行中的线程和当前的线程不一致，则该行前会加上 ***** 符号。由 **thread** 子命令按照两种方式显示的信息如下。

由 **thread** 子命令按照第一种方式显示的信息如下:

thread	按照 $t*ThreadNumber* 格式指出用户线程的符号名。
state-k	指出核心线程的状态（如果用户线程连接了核心线程）。它可以是 run、wait、susp 或者 term，分别对应运行、等待、挂起和终止。

wchan　　指出核心线程正在等待或者睡眠的事件（如果用户线程连接了核心线程）。
state-u　　指出用户线程的状态。可能的状态有 running、blocked 或者 terminated。
k-tid　　指出核心线程标识符（如果用户线程连接到核心线程）。
mode　　指出用户线程停止的方式（如果用户线程连接到了核心线程），它可以是核心方式或用户方式。
held　　指出是否保持用户线程。
scope　　指出用户线程的争用作用域；对于系统或者进程争用域分别是 sys 或者 pro。
function　　指出用户线程函数的名称。

由 **thread** 子命令按照第二种方式显示的信息如下。按照缺省值，对于 **thread** 子命令 **th -** 参数，显示的信息是长格式。

thread　　按照 $t*ThreadNumber* 格式指出用户线程的符号名。

核心线程的相关信息

tid　　指出用户线程标识符（如果用户线程连接了核心线程）。
pri　　指出了核心线程的优先级。
sched　　指出了核心线程的调度策略。它可以是 fif、oth 或 rr，分别对应 fifo、其他和环（round robin）调度策略。
state　　指出核心线程的状态（如果用户线程连接了核心线程）。它可以是 run、wait、susp 或 zomb，分别对应运行、等待、挂起和僵尸（zombie）状态。

用户线程相关信息

tid　　指出用户线程标识符。
pri　　指出用户线程的优先级。
sched　　指出了用户线程的调度策略。它可以是 fif、oth 或 rr，分别对应 fifo、其他和环（round robin）调度策略。
state　　指出用户线程的状态。可以是 running、 creating、 suspended、 blocked、 runnable 或 terminated。
state　　按照十六进制指出用户状态。
flags　　按照十六进制指出 pthread 标志的值。
wchan　　指出核心线程正在等待或者睡眠的事件（如果用户线程连接了核心线程）。
mode　　指出用户线程停止的方式（如果用户线程连接到了核心线程），它可以是核心方式或用户方式。
held　　指出是否保持用户线程。
scope　　指出用户线程的争用作用域；对于系统或者进程争用域分别是 sys 或者 pro。
cancellation

pending
　　指出是否取消处于挂起状态。

state　　指出取消的方式和状态。

　　如果取消没有被挂起而状态和方式分别为启用和推迟，则用 **ed** 表示，如果取消状态和方式分别为启用和异步，则用 **ea** 代表，而如果方式没有启用，则用 **d** 代表。

　　如果取消被挂起而取消的状态和方式分别为启用和推迟，则用 **ED** 代表，如果取消状态和方式分别为启用和异步，则用 **EA** 代表，而如果方式没有启用，则用 **D** 代表。

joinable　　指出线程是否可以连接。
boosted　　指出线程 boosted 值。
function　　指出用户线程函数的名称。
cursig　　指出当前信号的值。

如果设置了可选项 $thcomp，则将会按照如下的压缩格式显示信息。

```
m           mode           (k)ernel (u)ser
k           k-state        (r)unning (w)aiting (s)uspended (z)ombie
u           u-state        (r)unning (R)unnable (s)uspended (t)erminated

                           (b)locked (c)reating
h           held           (yes) (n)o
s           scope          (s)ystem (p)rocess
c           cancellation   not pending:  (e)nabled & (d)eferred,
                                         (e)nabled & (a)sync, (d)isabled
                           pending    :  (E)nabled & (D)eferred,
                                         (E)nabled & (A)sync, (D)isabled
j           joinable       (yes) (n)o
b           boosted        value of boosted field in pthread structure
plk         kernel thread  (oth)er  (fif)o  (rr)-> round-robin
            policy
plu         user thread    (oth)er  (fif)o  (rr)-> round-robin
            policy
prk         kernel thread  hex number
            policy
pru         user thread    hex number
            policy
k-tid                      kernel thread id in hex
u-tid                      pthread id in hex
fl                         value of flags field in pthread structure in hex
sta                        value of state field in pthread structure in hex
cs                         value of the current signal
wchan                      event for which thread is waiting
function                   function name
```

thread 子命令的第二种格式用来选择当前线程。**dbx** 调试程序的 **print**、**registers** 以及 **where** 子命令都在当前线程的上下文中工作。如果当前的线程处于核心方式 **registers** 子命令不能显示寄存器。

thread 子命令的第三种格式用来控制线程的执行。可以使用 **hold** 标志保持线程，或者使用 **unhold** 标志释放线程。保持的线程将不会再继续直到它被释放。

注: **dbx** 调试程序的 **print** 子命令识别线程的符号名并可以显示相应对象的状态。

标志:

current 如果没有指定 *ThreadNumber* 参数，将会显示当前的线程。如果指定了 *ThreadNumber* 参数，选择指定的用户线程作为当前线程。

help 显示所有在使用 **th -** 命令时显示的线程选项的相关信息。

hold 如果没有 *ThreadNumber* 参数，则会保持和显示所有的用户线程。如果指定了一个或者更多的 *ThreadNumber* 参数，将会保持并显示指定用户线程。

unhold 如果没有指定 *ThreadNumber* 参数将会释放和显示所有以前保持的用户线程。如果指定了一个或者更多的 *ThreadNumber* 参数，将会释放并显示指定的用户线程。

info 如果没有指定 *ThreadNumber* 参数，将显示所有用户线程的长格式列表。如果指定了一个或者更多的 *ThreadNumber* 参数将会显示指定用户线程的长格式列表。

所有上述标志使用 [-] 选项。如果给定了该选项，则将按照第二种格式以及长格式显示信息，除非设置了 **set** `$thcomp` 选项。

run 显示处于 `run` 状态的线程。

susp 显示处于 `susp` 状态的线程。

term 显示处于 `term` 状态的线程。

wait 显示处于 `wait` 状态的线程。

示例:

1. 要显示所有处于 wait 状态的线程信息，请输入:

```
thread  wait
```

输出类似于:

```
thread  state-k   wchan state-u   k-tid mode held scope function
 $t1     wait           running   17381   u   no   pro  main
 $t3     wait            running    8169   u   no   pro  iothread
```

2. 要显示若干给定线程的信息，请输入:

```
thread 1 3 4
```

输出类似于:

```
thread  state-k   wchan state-u   k-tid mode held scope function
 $t1     wait           running   17381   u   no   pro  main
 $t3     wait           running    8169   u   no   pro  iothread
>$t4     run            running    9669   u   no   pro  save_thr
```

3. 要使线程 4 成为当前线程，请输入:

```
thread  current 4
```

4. 要保持 2 号线程，请输入:

```
thread  hold 2
```

5. 要以第二种格式显示处于等待状态的线程的信息，请输入:

```
thread wait -
```

输出类似于:

```
thread m k u h s c  j b  kpl  upl  kpr upr k_tid   u_tid   fl  sta wchan function
*$t1   u r w n p ed y 0  oth  oth  61   1  0043e5  000001  51  004       main
 $t3   u r w n p ed y 0  oth  oth  61   1  001fe9  000102  51  004        iothread
>$t4   u r r n p ed y 0  oth  oth  61   1  0025c5  000203  50  064       save_thr
```

6. 要按照第二种格式显示若干给定线程的信息，请输入:

```
thread - 1 2 3
```

输出类似于:

```
thread m k u h s c  j b  kpl  upl  kpr upr k_tid   u_tid   fl  sta wchan function
*$t1   u r w n p ed y 0  oth  oth  61   1  0043e5  000001  51  004       main
 $t3   u r w n p ed y 0  oth  oth  61   1  00fe9   000102  51  004       iothread
>$t4   u r r n p ed y 0  oth  oth  61   1  0025c5  000203  50  064       save_thr
```

请参阅 **attribute** 子命令、**condition** 子命令、**mutex** 子命令、**print** 子命令、**registers** 子命令以及 **where** 子命令。

另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『创建线程』。

tls 子命令

tls map

tls 子命令只接受一个标志，它使用该标志来显示每个装入的 TLS 模块的 TLS 初始化模板起点和长度。

tnext 子命令

tnext [*Number*]

tnext 子命令将正在运行的线程一直运行至下一个源程序行。*Number* 参数指定 **tnext** 子命令运行的次数。如果未指定 *Number* 参数，则 **tnext** 只运行一次。该子命令只能在系统范围线程上调用。

所有线程都在此操作期间运行。要在此操作期间捕获断点，请设置 `$catchbp` **dbx** 变量。如果 `$catchbp` 变量已设，并且已到达了另一个线程的断点，则 **tnext** 子命令不会重复剩余的次数。

示例:

1. 要继续执行正在运行的线程，直到执行至下一个源程序行，请输入:

   ```
   tnext
   ```

2. 要继续执行正在运行的线程，直到执行至当前源程序行的后面第三行，请输入:

   ```
   tnext 3
   ```

请参阅 **tnexti** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『调试涉及多个线程的程序』。

tnexti 子命令

tnexti [*Number*]

tnexti 子命令将正在运行的线程一直运行至下一条指令。*Number* 参数指定 **tnexti** 子命令运行的次数。如果未指定 *Number* 参数，则 **tnexti** 只运行一次。该子命令只能在系统范围线程上调用。

所有线程都在此操作期间运行。要在此操作期间捕获断点，请设置 `$catchbp` **dbx** 变量。如果 `$catchbp` 变量已设，并且已到达了另一个线程的断点，则 **tnexti** 子命令不会重复剩余的次数。

示例:

1. 要继续执行正在运行的线程，直到执行至下一条机器指令，请输入:

   ```
   tnexti
   ```

2. 要继续执行正在运行的线程，直到执行至当前机器指令后面的第三条机器指令，请输入:

   ```
   tnexti 3
   ```

请参阅 **tnext** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『调试涉及多个线程的程序』。

trace 子命令

trace [*SourceLine* | *Expression* **at** *SourceLine* | *Procedure* | [*Variable*] [**at** *SourceLine* | **in** *Procedure*] | **on load** *ModuleName*] [**if** *Condition*]

程序运行时 **trace** 子命令打印指定过程、函数、命令行、表达式或变量的跟踪信息。*SourceLine* 变量可以通过一个整数或者代表文件名的字符串后跟 :（冒号）和一个整数的方法指定。可以指定条件。**dbx** 调试程序将一个数字和每个 **trace** 子命令联系起来。使用 **status** 子命令来查看这些数字。使用 **delete** 子命令来关闭跟踪。可分别使用 **enable** 和 **disable** 子命令来启用及禁用跟踪。

trace 子命令可以在被调试进程装入或卸装了模块时显示跟踪信息。可选的 *ModuleName* 参数可指定为单个模块名称或以以下格式与成员名配对的模块名称。

ModuleName(*MemberName*)

如果不带 *ModuleName* 参数使用 **on load** 标志，dbx 将跟踪所有模块的装入和卸装。

根据缺省值，跟踪是基于进程的。要创建基于线程的跟踪，按照示例 8 的条件指定线程。

标志:

at *SourceLine*	指定发现被跟踪的表达式的源行。
if *Condition*	指定跟踪开始的条件。跟踪只有在 **if** *Condition* 为 true 时开始执行。
in *Procedure*	指定用来发现被跟踪的过程或者变量的过程。
on load *ModuleName*	指定要监视的装入模块。

示例:

1. 要跟踪每个 printf 过程的调用，请输入:

   ```
   trace printf
   ```

2. 要跟踪 hello.c 文件第 22 行的每次执行，请输入:

   ```
   trace "hello.c":22
   ```

3. 要跟踪 x 变量在 main 过程中的修改，请输入:

   ```
   trace x in main
   ```

4. 要跟踪数据地址 0x2004000，请输入:

   ```
   set $A=0x2004000
   trace $A
   ```

   注: **tracei** 子命令用来跟踪地址的。

5. 您可以将源行的打印限制在指定的 *Procedure* 活动的时候。您也可以指定可选择的 *Condition* 来控制何时产生跟踪信息。例如:

   ```
   (dbx) trace in sub2
   [1] trace in sub2
   (dbx) run
   trace in hellosub.c:  8  printf("%s",s);
   trace in hellosub.c:  9  i = '5';
   trace in hellosub.c:  10  }
   ```

6. 您可以在过程每次被调用或返回时显示一个消息。当调用了一个过程，该消息包括传递的参数和调用的例程的名称。在返回中，该信息包括从 *Procedure* 返回的值。例如:

   ```
   (dbx) trace sub
   [1] trace sub
   (dbx) run
   calling sub(s = "hello", a = -1, k = delete) from function main
   returning "hello" from sub
   ```

7. 您可以在程序运行到指定的行时打印 *Expression* 的值。行号和文件会打印出来，但是不会打印源命令行。例如:

   ```
   (dbx) trace x*17 at "hellosub.c":8 if (x > 0)
   [1] trace x*17 at "hellosub.c":8 if x > 0
   (dbx) run
   at line 8 in file "hellosub.c": x*17 = 51

   (dbx) trace x
   [1] trace x
   initially (at line 4 in "hello.c"):  x = 0
   after line 17 in "hello.c":  x = 3
   ```

8. 要跟踪 **x** 变量在 $t1 线程中的变化，请输入:

   ```
   (dbx) trace x if ($running_thread == 1)
   ```

9. 要跟踪所有模块的装入或卸装，请输入:

   ```
   trace on load
   ```

10. 要跟踪模块 Module 的装入或卸装，请输入:

```
trace on load "Module"
```

11. 要跟踪模块 Module 中成员 Member 的装入或卸装，请输入:

```
trace on load "Module(Member)"
```

也可请参阅 **tracei** 子命令。

tracehwp 子命令

tracehwp *Address Size*

tracehwp 子命令为指定的内存区域设置一个硬件监视点停止。如果区域的内容发生了变化，**dbx** 调试程序打印跟踪信息。

注:

1. **tracehwp** 命令的成功之处在于硬件依赖性。此功能只能在 630 型、PowerPC 机型以及后来的型号中使用。
2. 由于硬件只能设置一个监视点的限制，在试图创建另一个硬件观察点时活动的观察点事件将会在 **stophwp** 和 **tracehwp** 发生冲突。同样的，以前的事件必须在创建一个新的事件之前删除。同样，因为存在一个活动的软件监视点（由一些 **stop** 和 **trace** 子命令生成）取消硬件监视点的性能，这些类型的事件也可以作为创建一个硬件监视点之前必须删除的冲突事件。

示例:

1. 要跟踪起始地址为 0x200004e8 的 4 个字节的内存内容，请输入:

```
tracehwp 0x200004e8 4
```

请参阅 **stophwp** 子命令。

tracei 子命令

tracei [[*Address*] [**at** *Address* | **in** *Procedure*] | *Expression* **at** *Address*] [**if** *Condition*]

tracei 子命令在如下情况下开启跟踪:

- 由 *Address* 参数指定的地址内容更改，如果包含了 *Address* 标志。
- 如果指定了 **at** *Address* 参数，则运行 **at** Address 指令。
- 由 *Procedure* 指定的进程处于活动状态如果包含了 **in** *Procedure* 标志。
- 由 *Condition* 指定的条件为 true 如果包含了 **if** *Condition* 标志。

标志:

at *Address*	指定一个地址。当该地址处的指令被运行时可以启用跟踪。
if *Condition*	指定条件。当满足该条件时启用跟踪。
in *Procedure*	指定一个过程。该过程处于活动时启用跟踪。

示例:

1. 要跟踪每个指令的执行，请输入:

```
tracei
```

2. 要跟踪每次在地址 0x100020f0 处执行的命令，请输入:

```
tracei at 0x100020f0
```

3. 要在 main 过程活动时跟踪内存地址 0x20004020 内容每次发生的变化，请输入:

```
tracei 0x20004020 in main
```

4. 要跟踪线程 $t4 对地址 0x100020f0 处的指令的每次执行，请输入:

```
tracei at 0x100020f0 if ($running_thread == 4)
```

请参阅 **trace** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『使用 dbx 进行机器水平调试』。

tskip 子命令

tskip [*Number*]

tskip 子命令从当前停止点继续执行正在运行的线程。对于正在运行的线程，将跳过由 *Number* 参数指定的数量的线程级断点。该子命令只能对系统范围线程调用。

所有其他线程在此操作期间都运行，并且用户指定的所有断点和观察点将被捕获。当任何线程遇到断点或观察点时，执行会停止。即使 **tskip** 子命令启动的执行可由于另一个线程的事件而停止，然而当进程继续执行时，为上一个线程指定的 **tskip** 计数仍将有效，并且 **tskip** 计数指定的线程级断点的数量对于该线程将被忽略。当该线程结束时，将删除与之关联的 **tskip** 计数。

使用 **status** 子命令可查看该线程的剩余 **tskip** 计数。使用 **delete** 子命令可删除该线程的剩余 **tskip** 计数。

示例: 要继续执行，直至遇到从正在运行的线程的当前停止点开始的第二个线程级断点，请输入:

```
tskip 1
```

请参阅 cont 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『调试涉及多个线程的程序』。

tstep 子命令

tstep [*Number*]

tstep 子命令从正在运行的线程的当前源程序行运行指定数量的源程序行。*Number* 参数指定 **tstep** 子命令运行的次数。如果未指定 *Number* 参数，则 **tstep** 只运行一次。该子命令只能在系统范围线程上调用。

所有线程都在此操作期间运行。如果设置了 $hold_next，则除了正在运行的线程外，所有线程都将挂起。

注: 使用 **set** 子命令的 $stepignore 变量可控制 **tstep** 子命令的行为。$stepignore 变量允许 **tstep** 子命令单步跳过无调试信息可用的大例程。

示例:

1. 要使正在运行的线程继续执行一个源程序行，请输入:

   ```
   tstep
   ```

2. 要使正在运行的线程继续执行五个源程序行，请输入:

   ```
   tstep 5
   ```

3. 要防止 **dbx** 程序单步执行 **printf** 函数，如下面的示例代码所示:

   ```
   60 printf ("hello world /n");
   ```

   请输入:

   ```
   set $stepignore="function"; step
   ```

请参阅 cont 子命令、goto 子命令、**tnext** 子命令、set 子命令，以及 **tstepi** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『调试涉及多个线程的程序』。

tstepi 子命令

tstepi [*Number*]

tstepi 子命令从正在运行的线程的当前指令运行指定数量的指令。*Number* 参数指定 **tstepi** 子命令运行的次数。如果未指定 *Number* 参数，则 **tstepi** 只运行一次。该子命令只能在系统范围线程上调用。

所有线程都在此操作期间运行。如果设置了 `$hold_next`，则除了正在运行的线程外，所有线程都将挂起。

示例:

1. 要使正在运行的线程继续执行一条机器指令，请输入:

   ```
   tstepi
   ```

2. 要使正在运行的线程继续执行五条机器指令，请输入:

   ```
   tstepi 5
   ```

请参阅 gotoi 子命令、**tnexti** 子命令，以及 **tstep** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『调试涉及多个线程的程序』。

tstop 子命令

tstop { **in** *Procedure* | [*Variable*] **at** *SourceLine* [**if** *Condition*] } [**for $t***threadnumber*]

tstop 子命令为线程设置源程序级断点停止，并在指定线程到达断点时停止应用程序。创建事件时，指定的线程应已存在。如果未指定任何线程，则使用当前线程。当发生以下任一情况时，指定线程将停止:

- 使用了 **if** *Condition* 标志，并且 *Condition* 为 true。
- 使用了 **in** *Procedure* 标志，并且调用了 *Procedure*。
- 使用了 **at** *SourceLine* 标志，并且到达了 *SourceLine* 行号。*SourceLine* 变量可指定为整数，也可指定为后跟一个冒号（:）和一个整数的文件名字符串。

线程级断点只能在系统范围线程上设置。当同时遇到线程级断点和进程级断点时，两个断点都将得到处理，并且将报告线程级断点。当该线程终止时，将删除与之关联的事件。

标志:

at*SourceLine*	指定行号。
for $t *threadnumber*	指定线程号。
if *Condition*	指定条件（例如，`true`）。
in *Procedure*	指定调用的过程。

示例:

1. 要在运行线程 2 时在 **func** 过程中的第一个语句处停止执行，请输入:

   ```
   tstop in func for $t2
   ```

2. 要在 x 变量的值在执行过程的第 12 行处更改时停止当前线程的执行，请输入:

   ```
   tstop x at 12
   ```

请参阅 **ttrace** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『调试涉及多个线程的程序』。

tstophwp 子命令

tstophwp *address size* [**for $t***threadnumber*]

tstophwp 子命令为指定的内存区域设置线程级硬件观察点停止。在运行指定线程时，如果区域的内容发生更改，则程序停止。创建事件时，指定的线程应已存在。如果未指定任何线程，则使用当前线程。线程级观察点事件只能对系统范围线程设置。当该线程终止时，将删除与之关联的事件。

注:

1. **tstophwp** 子命令是否成功取决于硬件。此功能只在 PowerPC 机器的 630 型和更高型号上可用。
2. 由于存在“只允许设置一个观察点”的硬件限制，当试图使用 **tstophwp** 和 **ttracehwp** 为同一个线程创建另一个硬件观察点事件时，活动的线程观察点事件将成为冲突。为避免此问题，在创建新事件前必须删除前一个事件。由于活动软件观察点（由 **stop** 和 **trace** 子命令的某些调用创建）的存在可能抵消硬件观察点的性能增益，因此在创建硬件观察点之前还必须删除这些类型的事件，以避免冲突。
3. 当进程级观察点存在时，没有线程级观察点的线程将观察进程观察点位置。如果线程有线程级观察点，则该线程将观察线程观察点位置。
4. 线程级硬件观察点和进程级硬件观察点可共存，并且不会相互冲突。
5. 如果存在针对同一个地址的进程级观察点和线程级观察点，则报告进程级观察点事件。

标志:

for $t *threadnumber*　　指定线程号。

示例:　要在线程 2 正在运行、且起始地址为 0x200004e8 的 4 字节内存区域的内容更改时停止程序，请输入:

```
tstophwp 0x200004e8 4 for $t2
```

请参阅 **ttracehwp** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『调试涉及多个线程的程序』。

tstopi 子命令

tstopi { **in** *Procedure* | [*Address*] **at** *Address* [**if** *Condition*] } [**for $t***threadnumber*]

tstopi 子命令为线程设置指令级断点停止。创建事件时，指定的线程应已存在。如果未指定任何线程，则使用当前线程。当发生以下任一情况时，指定线程将停止:

- 使用了 **if** *Condition* 标志，并且 *Condition* 为 true。
- 使用了 **in** *Procedure* 标志，并且调用了 *Procedure*。
- 使用了 **at** *Address* 标志，并且到达了 *Address*。

线程级断点只能在系统范围线程上设置。当同时遇到线程级断点和进程级断点时，两个断点都将得到处理，并且将报告线程级断点。当该线程终止时，将删除与之关联的事件。

标志:

at *Address*　　指定机器指令地址。
for $t *threadnumber*　　指定线程号。
if *Condition*　　指定条件。
in *Procedure*　　指定调用的过程。

示例:

1. 要在运行线程 2 时在地址 0x100020f0 处停止执行，请输入:

   ```
   tstopi at 0x100020f0 for $t2
   ```

2. 在运行当前线程时，要在进入 **func** 过程时停止执行，请输入:

   ```
   tstopi in func
   ```

请参阅 **ttracei** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『调试涉及多个线程的程序』。

ttrace 子命令

ttrace { [*Variable*] **at** *SourceLine* | *Procedure* } [**if** *Condition*] [**for $t***threadnumber*]

ttrace 子命令在指定线程运行时显示指定过程、函数、源程序行和变量的跟踪信息。*SourceLine* 变量可指定为整数，也可指定为后跟一个冒号（:）和一个整数的文件名字符串。**dbx** 调试程序将一个编号与每个 **ttrace** 子命令相关联。使用 **status** 子命令来查看这些数字。使用 **delete** 子命令来关闭跟踪。可分别使用 **enable** 和 **disable** 子命令来启用及禁用跟踪。

如果未指定任何线程，则使用当前线程。线程级跟踪只能对系统范围线程设置。创建事件时，指定的线程应已存在。当该线程结束时，将删除与之关联的事件。

标志:

at*SourceLine*	指定发现被跟踪的表达式的源行。
for $t *threadnumber*	指定线程号。
if *Condition*	指定跟踪开始的条件。仅在 *Condition* 为 true 时跟踪才会开始。
in *Procedure*	指定用来查找要跟踪的过程或变量的过程。

示例:

1. 要在运行线程 2 时跟踪对 **printf** 过程的每个调用，请输入:

   ```
   ttrace printf for $t2
   ```

2. 要在当前线程运行时跟踪 **hello.c** 文件中第 22 行的每次执行，请输入:

   ```
   ttrace "hello.c":22
   ```

请参阅 **ttracei** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『调试涉及多个线程的程序』。

ttracei 子命令

ttracei [*Address*] **at** *Address* [**if** *Condition*] } [**for $t***threadnumber*]

当发生以下任一情况时，**ttracei** 子命令开启对指定线程的跟踪:

- 包含了 **if** *Condition* 标志，并且 *Condition* 为 true。
- 指定了 **at** *Address* 标志，并且 *Address* 处的指令运行。

如果未指定任何线程，则使用当前线程。线程级跟踪只能对系统范围线程设置。创建事件时，指定的线程应已存在。当该线程结束时，将删除与之关联的事件。

标志:

at *Address* 指定一个地址。当该地址处的指令被运行时可以启用跟踪。
for $t *threadnumber* 指定线程号。
if *Condition* 指定条件。当满足该条件时启用跟踪。

示例:

1. 在线程 3 运行时，要在每次执行地址 0x100020f0 处的指令时进行跟踪，请输入:

   ```
   tracei at 0x100020f0 for $t3
   ```

2. 要在当前线程每次执行地址 0x100020f0 处的指令时进行跟踪，请输入:

   ```
   tracei at 0x100020f0
   ```

请参阅 **ttrace** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『调试涉及多个线程的程序』。

ttracehwp 子命令

ttracehwp *address size* [**for $t***threadnumber*]

ttracehwp 子命令为指定的内存区域设置线程级硬件观察点跟踪。在运行指定线程时，如果区域的内容发生更改，则 **dbx** 调试程序显示跟踪信息。创建事件时，指定的线程应已存在。如果未指定任何线程，则使用当前线程。线程级观察点事件只能对系统范围线程设置。当该线程终止时，将删除与之关联的事件。

注:

1. **ttracehwp** 子命令成功与否取决于硬件。此功能只在 PowerPC 机器的 630 型和更高型号上可用。
2. 由于存在“只允许设置一个观察点”的硬件限制，当试图使用 **tstophwp** 和 **ttracehwp** 为同一个线程创建另一个硬件观察点事件时，活动的线程观察点事件将成为冲突。为避免此问题，在创建新事件前必须删除前一个事件。由于活动软件观察点（由 **stop** 和 **trace** 子命令的某些调用创建）的存在可能抵消硬件观察点的性能增益，因此在创建硬件观察点之前还必须删除这些类型的事件，以避免冲突。
3. 当进程级观察点存在时，没有线程级观察点的线程将观察进程观察点位置。如果线程有线程级观察点，则该线程将观察线程观察点位置。
4. 线程级硬件观察点和进程级硬件观察点可共存，并且不会相互冲突。
5. 如果存在针对同一个地址的进程级观察点和线程级观察点，则报告进程级观察点事件。

标志:

for $t *threadnumber* 指定线程号。

示例: 在运行线程 2 时，要在起始地址为 0x200004e8 的 4 字节内存区域的内容每次发生更改时进行跟踪，请输入:

```
ttracehwp 0x200004e8 4 for $t2
```

请参阅 **tstophwp** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『调试涉及多个线程的程序』。

unalias 子命令

unalias *Name*

unalias 子命令删除由 *Name* 参数指定的别名。

示例: 要删除名称为 printx 的别名，请输入:

```
unalias printx
```

请参阅 **alias** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『创建子命令别名』。

unset 子命令

unset *Name*

unset 子命令删除和由参数 *Name* 指定的名称相关的 **dbx** 调试程序变量。

示例: 要删除禁止显示浮点数寄存器的变量，请输入:

```
unset $noflregs
```

请参阅 **set** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『更改带有特殊调试变量的确打印输出』。

up 子命令

up [*Count*]

up 子命令将当前函数在堆栈内向上移动 *Count* 级。当前函数用来解析名称。*Count* 参数的缺省值为 1。

示例:

1. 要将当前函数在堆栈中向上移动两级，请输入:

   ```
   up 2
   ```

2. 要显示堆栈中的当前函数，请输入:

   ```
   up 0
   ```

请参阅 **down** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『更改当前文件或进程』、『显示堆栈跟踪』。

use 子命令

use [{ **+** | *Directory* | '['*RegularExpression* = *NewPath*']' } ...]

use 子命令设置 **dbx** 调试程序寻找源文件时要搜索的目录和要应用的路径映射的列表。如果不带参数指定 **use** 子命令，则会显示要搜索的目录和要应用的路径映射的当前列表。

@（符号“@”）是一个特殊目录，它指示 **dbx** 程序查找目标文件（如果存在）中的完整路径名信息。如果您有一个名为 @ 的相关目录要搜索，您可以在搜索路径中使用 **./@** 。

use 子命令使用 **+**（加号）来向要搜索的目录列表中添加更多目录或映射。指定为 **use** 子命令的输入时，+ 代表目录和映射的当前列表。要将目录或映射附加到当前列表的末尾，应该将 + 指定到新的目录或映射之前。要将目录预置到当前列表的开头，应该将 + 指定到新的目录或映射之后。如果您有一个名为 **+** 的目录，则请指定该目录的完整路径名（例如，**./+** 或 **/tmp/+**）。

use 子命令将括在 [和]（方括号）中的包含 =（等号）的字符串解释为路径映射。这些路径映射和特殊的 @ 目录结合使用。它们使得用户可以在编译后源文件全部目录结构发生更改的情况下表示源文件位置。

调试过程中尝试找到源文件时以下规则适用:

- 以指定的顺序评估列表中的目录。
- 评估完列表中的目录之后，将在目录中搜索给定的文件。如果该文件存在于目录中且可读，则会使用该文件。
- 评估完特殊的 @ 目录之后，如果已指定了一个或多个路径映射且一个路径映射的 *RegularExpression* 部分与目标文件中文件的完整路径名称信息的前几个 *n* 字符匹配，且该路径映射的 *NewPath* 部分的替换生成一个可读文件，则会使用该文件。
- 评估完特殊的 @ 目录之后，如果未指定路径映射或无匹配情况，则会搜索相应于该文件的完整路径名称信息的目录。如果该文件存在于目录中且可读，则会使用该文件。
- 如果多个路径映射生成一个可读文件，则会应用其 *RegularExpression* 与该文件的完整路径名称信息的最多字符匹配（即，最专有的）路径映射且会使用生成的文件。
- 如果多个路径映射生成一个可读文件且每个路径映射具有相同的特殊性，则会应用最接近列表开头的路径映射并会使用生成的文件。

注: 如果特殊的 @ 目录并非列表的一部分，则将完全忽略可能已指定的所有路径映射。

示例:

1. 要将搜索目录列表更改为当前目录（.）、父目录（..）和 **/tmp** 目录，请输入:

   ```
   use . .. /tmp
   ```

2. 要将搜索目录列表更改为当前目录（.）、源文件编译时（@）所在的目录和 **../source** 目录，请输入:

   ```
   use . @ ../source
   ```

3. 要将 **/tmp2** 目录添加到待搜索的目录列表中，请输入:

   ```
   use + /tmp2
   ```

4. 要向要搜索目录的列表开头添加 **/tmp3** 目录，请输入:

   ```
   use /tmp3 +
   ```

5. 要想表示其完整路径名称信息以 **/home/developer** 开头的源文件现在位于 **/mnt** 下，请输入:

   ```
   use + [/home/developer=/mnt]
   ```

6. 要指引 dbx 程序首先在 **/latest** 下查找，并在该处不存在该文件的情况下在 **/stable** 下查找其完整路径名称信息以 **/home/developer** 开头的文件，请输入:

   ```
   use + [/home/developer=/latest] [/home/developer=/stable]
   ```

另见 **edit** 子命令和 **list** 子命令。

whatis 子命令

whatis *Name*

whatis 子命令显示 *Name* 的声明，其中 *Name* 参数指定了变量、过程或者函数的名称，块名称可选。

注: 只有在 **dbx** 调试程序运行时使用 **whatis** 子命令。

示例:

1. 要显示 x 变量的声明，请输入:

   ```
   whatis x
   ```

2. 要显示 main 过程的声明，请输入:

```
whatis main
```

3. 要显示 x 变量在 main 函数中的声明，请输入:

```
whatis main.x
```

4. 要打印枚举、结构或者联合（或者 Pascal 中等价的结构）类型的标记，使用 $$TagName:

```
(dbx) whatis $$status
enum $$status { run, create, delete, suspend };
```

where 子命令

where [*startframe endframe*] [**>***File*]

where 子命令显示与帧号 *startframe* 到 *endframe* 关联的活动过程和函数的列表。堆栈帧的编号从当前活动函数的堆栈帧（该帧总是编号为 0）开始。如果存在 *n* 个帧，则 **main** 函数的帧将编号为 n-1。通过使用 **>***File* 标志可将此子命令的输出重定向到指定文件。

标志:

>*File* 重定向输出到指定的文件。

请参阅 **frame** 子命令、**up** 子命令和 **down** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『显示堆栈跟踪』。

whereis 子命令

whereis *Identifier*

whereis 子命令显示所有名称符合指定标识符的符号的完整修饰符。符号打印的顺序并不重要。

示例: 要显示名称为 x 的所有符号的修饰符，请输入:

```
whereis x
```

另见 **which** 子命令。

which 子命令

which *Identifier*

which 子命令显示给定标识符的完整的修饰符。完整的修饰符包括和该标识符相关的外部块的列表。

示例: 要显示 x 符号的完整修饰符，请输入:

```
which x
```

请参阅 **whereis** 子命令。另见《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『名称范围<』/l>。

文件

a.out 对象文件；包含对象代码。
core 包含核心转储。
.dbxinit 包含初始命令。

相关信息

adb 命令和 **cc** 命令。

ptrace 子例程。

a.out 文件和 **core** 文件。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『dbx 符号调试程序概述』和『使用 dbx 调试程序』。

---

dc 命令

用途

提供了一种交互式桌面计算器来完成任意精度的整数计算。

语法

dc [*File*]

描述

dc 命令是一个任意精度的算术计算器。**dc** 命令从 *File* 参数或者标准输入得到其输入直到它读到一个文件结束符。一旦 **dc** 命令接收到输入，它将求出计算值并将计算值写入到标准输出当中。它按十进制整数计算，但是您可以指定输入和输出的基数，以及小数部分保留的位数。**dc** 命令结构如同一个堆栈、逆波兰表示法计算。

bc 命令是 **dc** 命令的一个预处理器。它提供插入式符号以及和 C 语言类似的语法，能够实现程序的功能和控制结构。

子命令

c	清洁堆栈: **dc** 命令弹出堆栈中的所有的值。
d	复制栈顶值。
f	显示堆栈上的所有值。
i	弹出栈顶值并将其作为进一步输入的基数。
I	将输入基数压入栈顶。
k	弹出栈顶值并将其作为非负比例因子。位置的恰当数目将会显示在输出中并在乘、除和求幂中保留下来。如果所有的数值一起修改，比例因子、输入基数、输出基数的交互作用是合理的。
l*x*	将 *x* 变量代表的寄存器中的值压入堆栈。由 *x* 变量代表的寄存器是不能修改的。所有的寄存器都以为 0 的值启动。
L*x*	将 *x* 变量视为一个堆栈并将其顶部的值弹出到主堆栈中。
o	将栈顶值弹出并且将其作为进一步输出的基数。
O	将输出基数压入栈顶。
p	显示栈顶值。顶部值不会发生更改。
P	将栈顶作为一个字符串来进行解释，除去并显示它。
q	退出程序。如果 **dc** 命令正在运行一个字符串，它将弹出递归级别为 2。
Q	弹出栈顶值并将其作为字符串运行的级别。
s*x*	弹出栈顶值并将其存放在名称为 *x* 的寄存器中，其中 *x* 变量可以是任何字符。
S*x*	将 *x* 变量视为一个堆栈。它弹出主栈顶并将其值压入到 *x* 变量所代表的堆栈。
v	将堆栈顶部的元素用它的平方根来代替。选项中任何现有的小数部分将会计算在内，但是相反，比例因子将会被忽略。
x	将堆栈顶部的元素视为一个字符串并将其作为 **dc** 命令的一个字符串来运行。
X	将堆栈顶部的数字用它的比例因子来代替。

z	将堆栈中的元素数压入到堆栈中。
Z	将堆栈顶部的数字用这个数字的位数来代替。
Number	将指定的值压入堆栈。*Number* 是一个完整的字符串其数字从 0 到 9。要表示一个负数，在它前面加上一个 _（下划线）。一个数字可以包含小数点。
+ - / * % ^	对堆栈顶部的两个数进行加（+）、减（-）、乘（*）、除（/）、求余（%）或者取幂（^）。**dc** 命令弹出顶部的两个条目并将其结果压回到堆栈中。**dc** 命令忽略指数的小数部分。
[*String*]	将用方括号里的 *String* 参数放到堆栈的顶部。
[**=** | **>** | **<**] *x*	将堆栈顶部的两个元素弹出并进行比较。计算 *x* 变量代表的寄存器的值就像它们遵循规定的关系。
!	将其作为一个操作系统命令来解释该行中剩下的部分。
?	获取并且运行一个输入行。
;:	**bc** 命令使用这些字符用来进行数组操作。

示例

1. 要将 **dc** 命令作为计算器使用，请输入:

```
You:    1 4 / p
System: 0
You:    1 k   [ Keep 1 decimal place ]s.
        1 4 / p
System: 0.2
You:    3 k   [ Keep 3 decimal places ]s.
        1 4 / p
System: 0.250
You:    16 63 5 / + p
System: 28.600
You:    16 63 5 + / p
System: 0.235
```

可以在 **dc** 命令中如本例所示那样使用注释。包含在方括号中的注释和可能在其后附加 **s.**（**[** *Comment***]** **s.**）都会被 **dc** 命令忽略。只包含在方括号中的注释存储在堆栈的顶部。

如果是直接从键盘输入 **dc** 命令，按下 Ctrl-D 可以结束 **bc** 命令会话并返回到 shell 命令行。

2. 要装入并且运行一个 **dc** 程序文件，请输入:

```
You:    dc prog.dc
        5 lf x p [ 5 factorial ]s.
System: 120
You:    10 lf x p [ 10 factorial ]s.
System: 3628800
```

该项解释 **dc** 程序，它保存在 **prog.dc** 程序文件当中，然后从工作站的键盘中读入。`lf x` 计算存储在寄存器 `f` 中的函数，它可以由 **prog.c** 程序文件定义如下:

```
[ f: compute the factorial of n ]s.
[ (n = the top of the stack) ]s.
[ If 1>n do b; If 1<n do r ]s.
 [d 1 >b d 1 <r] sf
[ Return f(n) = 1 ]s.
 [d - 1 +] sb
[ Return f(n) = n * f(n-1) ]s.
 [d 1 - lf x *] sr
```

您可以使用任何文本编辑器或者使用 **bc** 的 **-c**（编译）标志来创建 **dc** 程序文件。当您直接从键盘输入 **dc** 命令表达式时，按下 Ctrl-D 可以结束 **bc** 命令会话并返回到 shell 命令行。

文件

/usr/bin/dc 包含 **dc** 命令。

相关信息

bc 命令。

---

dd 命令

用途

转换并复制文件。

语法

dd [**bs=***BlockSize*][**cbs=***BlockSize*] [**conv=** [ascii | block | ebcdic | ibm | unblock] [lcase | ucase] [iblock] [noerror] [swab] [sync] [oblock] [notrunc]] [**count=***InputBlocks*] [**files=***InputFiles*] [**fskip=***SkipEOFs*] [**ibs=***InputBlockSize*] [**if=***InFile*] [**obs=***OutputBlockSize*][**of=***OutFile*] [**seek=***RecordNumber*] [**skip=***SkipInputBlocks*][**span=***yes*|*no*]

dd [*Option*=*Value*]

描述

dd 命令读取 *InFile* 参数或者标准输入，进行指定的转换，然后将转换后的结果复制到 *OutFile* 参数或者标准输出中。可以指定输入输出块的大小以利用原始的物理 I/O。

注: *Block* 项指的是 **dd** 命令在一个操作中读取或者写的数据的数量，并且无需和磁盘块的大小相等。

指定了大小的地方需要一定字节数。由 **w**、**b** 或 **k** 结尾的数字表示分别与 2、512 或者 1024 相乘；由 **x** 或 *****（星号）隔开的两个数字说明是乘积。count 参数表示待复制的块的数量，而不是字节的数量。

由 **conv=ascii** 和 **conv=ebcdic** 标志关联的字符集映射是求补运算。这些标志在 ASCII 字符和大多数工作站和键控穿孔机上找到的 EBCDIC 字符子集间进行映射。

如果指定了任何 **block**、**unblock**、**ascii**、**ebcdic** 或 **ibm** 转换，则使用 **cbs** 参数值。如果指定了 **unblock** 或 **ascii** 参数，则 **dd** 命令将会执行一个固定长度到可变长度的转换。否则它将执行从可变长度到固定长度的转换。**cbs** 参数决定了该固定长度。

警告: 如果 **cbs** 参数指定的值小于最小输入块的值，则转换块将会被截短。

转换完成后，**dd** 命令将报告完整和部分输入输出块的数量。

注:

1. 通常您只需对输出文件的写许可。但是，当输出文件是不能直接访问的设备并且使用了 **seek** 标志，您还需要对文件的读访问。
2. 仅当以 **conv=ascii** 或 **conv=unblock** 标志集转换时，**dd** 命令才会插入新行字符；仅当以 **conv=ebcdic**、**conv=ibm** 或者 **conv=block** 标志集转换时，才能填充。
3. 在将文件复制到磁带时，尽可能使用 **backup**、**tar** 或者 **cpio** 命令来取代 **dd** 命令。这些命令被设计用来和磁带设备一起使用。要获得更多的使用磁带设备的信息，请参阅 **rmt** 特殊文件。

4. 由 **bs**、**ibs** 和 **obs** 标志指定的块大小值必须始终是使用的介质物理块大小的整数倍。
5. 当指定了 **conv=sync** 标志后，**dd** 命令用空值填充所有的部分输入块。这样，如果任何读取没有接收到一个完整的数据块（由标志 **ibs** 指定），则 **dd** 命令将在数据流中间插入空值。在从管道中读入时，这是经常发生的事情。
6. 如果 **bs** 标志由它自身指定而除了 **sync**、**noerror** 或 **notrunc** 没有指定其他转换，则从每个输入块得到的数据将会被作为一个单独的输出块写入；如果读入返回的值小于一个完整的块且没有指定 **sync** 标志，则结果输出块将会和输入块的大小相同。如果没有指定 **bs** 标志或者指定了 **sync**、**noerror** 或 **notrunc** 之外的其他转换，则输入将会被处理并收集到完整的输出块中，直到输入结束。

设备范围

如果输入文件比输出设备的物理大小要大，则 **dd** 命令可以跨设备工作。

注: 在指定块大小时注意 *bs* 必须为设备物理大小的整数倍因为不正确的块大小将会导致数据的不一致或者重叠。

如果 InFile 或 OutFile 参数是 stdin 或 stdout 时，则 **dd** 的跨设备范围将不会出现。

在写的过程中如果输出设备已经写满，则 **dd** 命令提示输入下一个设备时，将发生跨范围操作。从输入设备读入时，如果数据已从设备中完全读取（即使设备没有到达结尾），则 **dd** 也将提示下一个设备。在这种情况下，需要按下 ‘n’ 退出。

标志

bs=*BlockSize*
指定输入输出块的大小，取代 **ibs** 和 **obs** 标志。由 **bs** 标志指定的块大小值必须始终是使用的媒介物理块大小的整数倍。

cbs=*BlockSize*
指定由可变长度到固定长度和从固定长度到可变长度（如 **conv=block**）的转换块大小。

count=*InputBlocks*
仅复制由 *InputBlocks* 变量指定的输入块的数量。

conv= *Conversion,....*

指定一个或者多个转换选项。多个转换之间用逗号隔开。以下清单描述了可能的选项:

ascii 由 EBCDIC 转换为 ASCII。该选项与 **ebcdic**、**ibm**、**block** 和 **unblock** 选项冲突。

block 将变长记录转换到固定长度。长度由转换块大小（cbs）决定。该选项与 **ascii**、**ebcdic**、**ibm** 和 **unblock** 选项冲突。

ebcdic 由 ASCII 转换到标准 EBCDIC。该选项与 **ascii**、**ibm**、**block** 和 **unblock** 选项冲突。

ibm 将 ASCII 转换为 IBM® 版本的 EBCDIC。该选项与 **ascii**、**ebcdic**、**block** 和 **unblock** 选项冲突。

iblock、 **oblock**
在直接访问设备发生读写错误时使数据损失最小。如果您指定了 **iblock** 变量而在块读取期间发生了错误（其中块大小为 512 或者由 **ibs=***InputBlockSize* 变量指定大小），**dd** 命令试图以较小的大小单元重新读取数据块。如果 **dd** 命令可以确定输入设备的扇区大小，它将每次读取损坏块的一个扇区。否则它将每次读取 512 字节。输入块大小（**ibs**）必须是重试大小的整数倍。该选项包含与读入单个扇区错误有关的数据丢失。**oblock** 转换在输出时进行类似工作。

lcase 使得所有的字母字符转换为小写。

noerror
发生错误也不停止进程。

notrunc
不截取输出文件。相反，将保留不显式写入输出的块。

ucase 使得所有的字母字符转换为大写。

swab 交换每对字节。

sync 将每个输入块填充到由 **ibs** 值指定的长度。

unblock
将固定长度的块转化为可变长度。长度由转换块大小（cbs）决定。该选项与 **ascii**、**ebcdic**、**ibm** 和 **block** 选项冲突。

files=*InputFiles*
在结束之前复制由 *InputFiles* 变量值指定的文件数（只对磁带或类似设备输入的有用）。

fskip=*SkipEOFs*
在开始复制之前跳过由 *SkipEOFs* 变量指定数量的文件结束符；该 *SkipEOFs* 变量在多文件磁带中进行定位时很有用。

ibs=*InputBlockSize*
指定输入块的大小；缺省值为 512 字节或者一个块。由 **ibs** 标志指定的块大小值必须始终是使用的媒介物理块大小的整数倍。

if=*InFile*
指定输入文件的名称；标准输入是缺省值。

obs=*OutputBlockSize*
指定输出块的大小；缺省值为 512 字节或者一个块。由 **obs** 标志指定的块大小值必须始终是使用的媒介物理块大小的整数倍。

of=*OutFile*
指定输出文件的名称；标准输出是缺省值。

seek=*RecordNumber*
在复制之前从输出文件的开始寻找由 *RecordNumber* 变量指定的记录。

skip=*SkipInputBlocks*
在开始复制之前跳过指定的 *SkipInputBlocks* 值的输入块。

span=*yes|no*
如果指定为 `yes` 则允许设备范围而如果指定为 `no` 则按缺省工作。请参阅 Spanning Across Devices 以获得更多信息。

退出状态

此命令返回以下退出值:

0 输入文件已经成功复制。
>0 发生错误。

示例

1. 要将 ASCII 文本文件转化为 EBCDIC，请输入:

   ```
   dd  if=text.ascii  of=text.ebcdic  conv=ebcdic
   ```

   该命令将 text.ascii 文件转化为 EBCDIC 版本并存在 text.ebcdic 文件中。

   注: 如果您指定了 **conv=ebcdic** 参数，则 **dd** 命令将会将 ASCII ^（弯曲号）字符转换为未使用的 EBCDIC 字符（十六进制的 9A），将 ASCII ~（代字号）转换为 EBCDIC ^（NOT 符号）。
2. 要将变长记录的 ASCII 文件 **/etc/passwd** 转换为一个固定长度为 132 字节的 EBCDIC 纪录，请输入:

   ```
   dd  if=/etc/passwd  cbs=132  conv=ebcdic  of=/tmp/passwd.ebcdic
   ```
3. 要将每个记录为 132 字节的 EBCDIC 文件转换为小写的、可变长度的 ASCII 行，请输入:

   ```
   dd  if=/tmp/passwd.ebcdic  cbs=132  conv=ascii  of=/tmp/passwd.ascii
   ```
4. 要将变长记录的 ASCII 文件 **/etc/passwd** 转换为一个固定长度为 132 字节的 IBM 版本的 EBCDIC 文件，请输入:

   ```
   dd  if=/etc/passwd  cbs=132  conv=ibm  of=/tmp/passwd.ibm
   ```
5. 要从块大小为 1kB 的磁带中将块复制到块大小为 2kB 的磁带，请输入:

   ```
   dd  if=/dev/rmt0  ibs=1024  obs=2048  of=/dev/rmt1
   ```
6. 要将 **dd** 命令作为一个过滤器使用，请输入:

   ```
   ls -l | dd  conv=ucase
   ```

   该命令将用大写字母显示当前目录的长列表。

   注: **dd** 和 **cpio** 命令在 12 型 9348 磁带单元上的性能可以通过修改缺省块大小来得到改善。要修改块大小，请按下面的方法使用 **chdev** 命令:

   ```
   chdev -l Device_name -a block_size=32k
   ```
7. 要使用 36 个 512 字节块来有效的实现向 3.5 英寸 1.4MB 软盘的传输，请输入:

   ```
   dd  if=Filename  of=/dev/rfd0  bs=36b  conv=sync
   ```

   该命令将 *Filename* 参数的值一次一个柱面地写入到软盘设备。当从磁盘读入或者当文档的大小不是软盘块大小的整数倍时，需要 conv=sync 参数。如果 **dd** 命令的输入是一个管道而不是一个文件，请勿尝试使用本方法，它将在大多数输入中填充入空格而不只是最后一个块。
8. 要将块从块大小为 720 字节的输入文件复制到 1.44MB 大小的软盘当中，请输入:

   ```
   dd if=testfile of=/dev/fd0 bs=720b conv=sync
   ```

   注: 如果输入文件大于输出设备的物理大小，则 dd 命令将会提示您使用另外一个设备。

9. 要将块从块大小为 32k 字节的输入文件复制到磁带中，请输入:

   ```
   dd if=inputfile of=/dev/rmt0 bs=32k conv=sync
   ```

10. 要将数据的块从磁带复制到当前目录中的文件中（块大小设置为 32k），按照下面的方法输入:

    ```
    dd if=/dev/rmt0 of=outfile bs=32k conv=sync
    ```

11. 要将块从块大小为 720 字节的输入文件复制到 1.44MB 大小的软盘当中，请输入:

    ```
    dd if=testfile of=/dev/fd0 bs=720b conv=sync span=yes
    ```

    注: 如果输入文件大于输出设备的物理大小，则 **dd** 命令将会提示您使用另外一个设备。

12. 要将块从块大小为 32k 字节的输入文件复制到磁带中，请输入:

    ```
    dd if=inputfile of=/dev/rmt0 bs=32k conv=sync span=yes
    ```

13. 要将块数据从块大小为 32k 的磁带复制到当前目录中的文件中，请输入:

    ```
    dd if=dev/rmt0 of=outfile bs=32k conv=sync span=yes
    ```

文件

/usr/bin/dd 包含 **dd** 命令。

相关信息

backup、**cp**、**cpio**、**tar**、**tr** 命令。

rmt 特殊文件。

《*操作系统与设备管理*》中的『备份方法』提供了有关使用备份和使用内存设备的信息。

《*操作系统与设备管理*》中的『文件』提供了有关如何处理文件的信息。

defaultbrowser 命令

用途

启动缺省 Web 浏览器并可选择装入指定的 URL。

语法

defaultbrowser [*URL* [**new-window**, **new-tab**]]

描述

defaultbrowser 命令运行 DEFAULT_BROWSER 环境变量中指定的浏览器启动命令。

如果 *URL* 作为参数给出，则将 URL 装入到浏览器中。要正确执行这项工作，浏览器命令必须将 URL 作为一个参数接受。

如果将要启动的浏览器是 Mozilla Web 浏览器，则可以使用可选的 **new-window** 和 **new-tab** 参数。必须始终带 URL 指定这两个参数。该 URL 随后将在新浏览器窗口或新标签中打开。如果浏览器不是 Mozilla Web 浏览器，则将忽略这两个参数。

defaultbrowser 命令的主要用途在于应用程序需要打开浏览器来显示 HTML 文档或基于 Web 的应用程序时使用该命令。使用这种方法，系统管理员只需要在安装了一个新的浏览器时修改 DEFAULT_BROWSER 环境变量就可以使所有的应用程序自动使用这个新的浏览器。

DEFAULT_BROWSER 环境变量必须设置成启动期望的浏览器的命令。并将任何必须包含在命令后以启动指定 URL 地址的参数包含进去。例如，如果启动浏览器并打开了一个指定 URL 的命令是 `wonderbrowser -r URL`，则 DEFAULT_BROWSER 环境变量将设成 `wonderbrowser -r`。

如果未定义 DEFAULT_BROWSER 环境变量，则 **defaultbrowser** 命令会运行 Mozilla Web 浏览器（如果已安装）。

示例

1. 要启动指定的缺省浏览器并打开缺省主页，请输入:

   ```
   defaultbrowser
   ```

2. 要启动指定的缺省浏览器并且用它打开 URL http://machine/path/file.html，请输入:

   ```
   defaultbrowser http://machine/path/file.html
   ```

3. 要启动指定的缺省浏览器并让它打开 URL http://machine/path/file.html；并且如果缺省浏览器是 Netscape，则该页将会显示在名为 **webpage** 的窗口中，请输入:

   ```
   defaultbrowser http://machine/path/file.html webpage
   ```

4. 要启动指定的缺省浏览器并且如果浏览器是 Mozilla Web 浏览器，在新浏览器窗口中用它打开 URL http://machine/path/file.html，请输入:

   ```
   defaultbrowser http://machine/path/file.html new-window
   ```

5. 要启动指定的缺省浏览器并且如果浏览器是 Mozilla Web 浏览器，在新浏览器标签中用它打开 URL http://machine/path/file.html，请输入:

   ```
   defaultbrowser http://machine/path/file.html new-tab
   ```

文件

/usr/bin/defaultbrowser	**defaultbrowser** 命令

---

defif 方法

用途

在配置数据库中定义一个网络接口。

语法

defif [**-c** *Class* **-s** *Subclass*] **-t** *Type*

描述

defif 方法定义了网络接口的指定实例。它只定义当前配置适配器的接口。要定义指定的实例，**defif** 方法执行以下操作:

1. 在配置数据库中创建一个定制接口实例。
2. 派生接口实例的逻辑名。
3. 检索预定义属性。

4. 更新定制相关性对象类来反映已定义接口实例的相关性。
5. 设置接口实例的状态标志为 **defined**。

标志

-c *Class* 指定要定义的接口类。有效值是 **if**。

-s *Subclass* 指定要定义的接口子类。有效的值为:

TR	令牌环
EN	以太网
SL	串行线路接口协议
XT™	X.25 协议
LO	回送

-t *Type* 指定要定义的接口的类型。有效的值为:

tr	令牌环
en	以太网
sl	串行线路接口协议
ie3	IEEE 802.3 以太网
lo	回送
xt	X.25 协议

示例

要定义一个令牌环网络接口实例，按以下格式输入方法:

```
defif -t tr
```

相关信息

mkdev 命令。

odm_run_method 子例程。

《*网络与通信管理*》中的『TCP/IP 网络接口』。

General Programming Concepts 中的 Object Data Manager (ODM) Overview for Programmers。

Kernel Extensions and Device Support Programming Concepts 中的 Writing a Device Method。

---

definet 方法

用途

在系统配置数据库中定义一个 inet 实例。

语法

definet [**-c** *Class*]

描述

definet 方法在 ODM 配置数据库中创建一个对象指定 inet 实例的定制属性。它执行以下操作:

1. 创建一个定制的 inet 实例。
2. 将 inet 实例的状态标志设置为已定义。

该方法由 **mkdev** 高级命令调用而且并不会出现在命令行中。

注: **definet** 方法是一个编程工具，并不需要从命令行中执行该工具。

标志

-c *Class* 指定 inet 实例为已定义。*Class* 变量唯一可用的值为 **tcpip**。

示例

要定义 inet0 实例，按照如下方法:

```
definet
```

相关信息

mkdev 命令。

odm_run_method 子例程。

General Programming Concepts 中的 Object Data Manager (ODM) Overview for Programmers。

Kernel Extensions and Device Support Programming Concepts 中的 Writing a Device Method。

---

defragfs 命令

用途

增加文件系统的邻接可用空间。

语法

defragfs [**-q** | **-r** | **-s**] { *Device* | *FileSystem* }

描述

defragfs 命令通过重新组织分配使空间邻接而不是散布在整个磁盘从而增加了文件系统邻接可用空间。要整理碎片的文件系统可以通过指定 *Device* 变量来指定，它是逻辑卷的路径名（例如 **/dev/hd4** ）。它同样可以用 *FileSystem* 变量来指定，它是 **/etc/filesystems** 文件中的安装点。

defragfs 命令用于片段和压缩的文件系统。然而，您可以使用 **defragfs** 命令来增加无碎片文件系统的邻接可用空间。

您必须安装文件系统的读写以便成功地运行此命令。使用 **-q**、**-r** 或 **-s** 标志生成一个分解片段报告。这些标志不会更改文件系统。

defragfs 命令使用快照慢慢地比较 JFS2 文件系统，因为必须将数据复制到快照存储对象。如果存在快照，**defragfs** 命令将会发出一个警告信息。**snapshot** 命令可以用来删除快照，然后在 **defragfs** 命令完成之后再次使用该命令以创建一个新的快照。

标志

-q 报告文件系统的当前状态。
-r 报告文件系统的当前状态以及不带 **-q**、**-r** 或 **-s** 标志运行 **defragfs** 命令会导致的状态。
-s 报告文件系统中的分解片段。该选项将使 **defragfs** 通过文件系统中可能导致降低性能的元数据。

输出

在 JFS 文件系统中，由 defragfs 命令报告的消息定义如下:

可用片段的数量
文件系统中可用片段的数量。

已分配片段的数量
文件系统中已分配片段的数量。

小于一个块的可用空间的数量
文件系统中小于一个块的可用空间的数量。可用空间是一组没有被分配的邻接片段。

小的可用空间中可用片段的数量
所有小的可用空间的片段总数。小的可用空间是指小于一个块的空间。

移动的片段数量
移动的片段总数。

移动的逻辑块的数量
移动的逻辑块的总数。

试图分配空间的次数
重新分配可用片段的次数。

精确匹配的数量
移动的片段正好适合某个可用空间的次数。

片段的总数
文件系统中片段总数。

可以被迁移的片段的数量
整理碎片过程中可以被移动的片段的数量。

FileSystem 文件系统有百分之 n 片段。
用百分比显示文件系统的片段范围。

在 JFS2 文件系统中 defragfs 命令报告的消息定义如下:

总的分配组
文件系统中分配组的数量。分配组将文件系统空间分成了若干大块。分配组允许 JFS2 资源分配策略使用常见方法获得好的 I/O 性能。

有碎片的分配组
具有碎片的分配组的组数。

跳过的分配组 - 完全可用
因为空间完全可用而跳过的分配组的组数。

跳过的分配组 - 太少的可用块
因为太少可用块可重新分配而跳过的分配组的组数。

分配组跳过 - 含有大的邻接可用空间
跳过分配组（因为包含大的邻接可用空间而不值得整理碎片）的数目。

分配组作为磁盘整理的候选
适合进行磁盘碎片整理的分配组的数量。

候选分配组中的平均可用运行空间数
适合整理碎片的分配组中平均每一分配组的自由运行空间数。可用运行空间是那些没有分配的邻接块的集合。

块总数 文件系统中总的模块数。

可以被迁移的块数
整理碎片过程中可以被迁移的块的数量。

FileSystem 文件系统片段占百分之 n。
用百分比显示文件系统的片段范围。

示例

1. 要进行整理 **/data1** 文件系统（它位于 **/dev/lv00** 逻辑卷中），请输入:
   ```
   defragfs /data1
   ```
2. 要通过指定安装点来整理 **/data1** 文件系统，请输入:
   ```
   defragfs /data1
   ```
3. 要生成关于 **/data1** 文件系统包括当前状态以及整理后的状态的报告，请输入:
   ```
   defragfs  -r /data1
   ```
4. 要生成关于 **/data1** 文件系统分解片段的报告，请输入:
   ```
   defragfs -s /data1
   ```

文件

/etc/filesystems 列出已知的文件系统并且定义它们的特征。

相关信息

crfs 命令、**lsfs** 命令、**mkfs** 命令。

《*操作系统与设备管理*》一书中的『JFS 数据压缩』、『JFS 碎片』以及『可变数量的 i 节点』。

---

defvsd 命令

用途

指定某个节点拥有或使用虚拟共享磁盘。

语法

defvsd *logical_volume_name global_group_name vsd_name*

描述

运行此命令来将驻留在可全局访问的卷组上的逻辑卷指定用作虚拟共享磁盘。

您可以使用系统管理接口工具（SMIT）来运行 **defvsd** 命令。要使用 SMIT，请输入:

```
smit vsd_data
```

并选择**定义虚拟共享磁盘**选项。

标记

–r 复位在运行该命令的节点上所指定节点的外发和预期序号。当已重新引导、抛出另一个节点，或者已在该节点上重新配置所有的虚拟共享磁盘时，请使用此标志。同时抛入指定的节点。

注: 此选项只应该在 IBM 服务人员的直接指导下使用。切勿在常规环境下使用。

–R 复位运行该命令的节点上所有节点的外发和预期序号。在重新引导节点后使用此标志。将抛入虚拟共享磁盘网络中的所有节点。

注: 此选项只应该在 IBM 服务人员的直接指导下使用。切勿在常规环境下使用。

–p 将虚拟共享磁盘并行性级别设置为指定的数字。有效范围为 1 到 9。缺省值为 9。较大的值可能会给大请求比较长的响应时间。（请参阅 *RSCT for AIX 5L™: Managing Shared Disks* 以获取更多关于调节虚拟共享磁盘性能的信息。）

此值是虚拟共享磁盘 IP 设备驱动程序在内核中进行 **uphysio** 调用时使用的 *buf_cnt*参数。使用 **statvsd** 显示运行命令的节点上的当前值。

–k 抛出在本地节点上指定的节点数。本地节点忽略来自抛出节点的请求。使用 **–r** 将节点抛回来。

注:

1. 在使用此标志之前，请参考以下的“限制”部分。
2. 此选项只应该在 IBM 服务人员的直接指导下使用。切勿在常规环境下使用。

–t 列出当前路由表和虚拟共享磁盘驱动程序高速缓存的 mbuf 头。

–T 清除或释放所有高速缓存的路由。

–v *vsd_name* **...**

复位读写请求数中关于指定的虚拟共享磁盘的统计信息。

–V 复位读写请求数中所有已配置虚拟共享磁盘的统计信息。

–C 复位由 **statvsd** 命令显示的虚拟共享磁盘设备驱动程序计数器。客户机和服务器节点中外发的和预期的请求序号除外。

–K 抛出本地节点上的所有节点。本地请求仍会得到肯定。

注:

1. 在使用此标志之前，请参考以下的“限制”部分。
2. 此选项只应该在 IBM 服务人员的直接指导下使用。切勿在常规环境下使用。

–M 设置虚拟共享磁盘最大 IP 消息大小。这是虚拟共享磁盘通过网络为一个 I/O 请求发送的最大数据块。此限制还影响本地虚拟共享磁盘 I/O 块大小。该值以字节计算且必须不大于网络最大的传输单元（MTU）大小。所有节点应该使用相同的值。建议的值为:

- 61440（60KB）（对于交换机）

- 8192（8KB）（对于巨型帧以太网）
- 1024（1KB）（对于 1500 字节 MTU 以太网）

参数

logical_volume_name

是想要指定为虚拟共享磁盘的逻辑卷的名称。该逻辑卷必须驻留在指明的全局卷组上。名称的长度必须小于或等于 15 个字符。

global_group_name

是由您希望在其中指定虚拟共享磁盘的 **vsdvg** 命令先前所定义的可全局访问卷组的名称。名称的长度必须小于或等于 31 个字符。

vsd_name

为新的虚拟共享磁盘指定唯一的名称。该名称在 RSCT 对等域中必须唯一，并且要避免将来可能产生的名称冲突，该名称在整个集群中也应当是唯一的。建议使用的命名约定为 **vsdnn***gvg_name*。名称的长度必须小于或等于 31 个字符。

注: 如果指定的 *vsd_name* 已经是另一台设备的名称，则 **cfgvsd** 命令对于该虚拟共享磁盘将失败。此错误确保为该名称创建的特殊设备文件不会覆盖和毁坏代表其他某种设备类型（例如逻辑卷）的同名文件。

安全性

您必须有 **root** 权限才能运行该命令。

限制

必须从对等域中的联机节点上发出此命令。要使对等域联机，请使用 **startrpdomain** 命令。要使特定节点在现有对等域中联机，请使用 **startrpnode** 命令。有关创建和管理 RSCT 对等域的更多信息，请参考 *RSCT Administration Guide*。

示例

1. 以下示例指定在可全局访问的卷组 **vg1n1** 上，称为 **lv1vg1n1** 的逻辑卷作为名为 **vsd1vg1n1** 虚拟共享磁盘使用。

```
defvsd lv1vg1n1 vg1n1 vsd1vg1n1
```

位置

/opt/rsct/vsd/bin/defvsd

相关信息

命令: **vsdatalst**、**vsdvg**、**undefvsd**

---

deleteX11input 命令

用途

从 ODM（对象数据管理器）数据库中删除 X11 输入扩展记录。

语法

deleteX11input *DeviceName ...*

描述

deleteX11input 命令用来从 ODM 数据库中删除 X11 输入扩展记录。对于每一个指定的 *DeviceName* ODM 数据库找到尽可能多的对象实例。该命令将询问用户验证是否删除发现的每个指定的设备。也可以指定部分名称。

该命令是一条 root 用户或者系统用户才能使用的命令。如果存在没有权限的用户企图删除记录，将会发生许可错误。

参数

DeviceName 指定 X11 输入扩展设备名称。

错误代码

没有在 ODM 数据库找到设备名。 在 ODM 数据库中没有符合指定模式的对象。

用法: deleteX11input DeviceName 该用户没有指定设备名。

相关信息

addX11input 命令、**listX11input** 命令。

---

delta 命令

用途

在 SCCS 文件中创建一个 delta。

语法

delta [**-r** *SID*] [**-s**] [**-n**] [**-g** *List*] [**-p**] [**-m** *ModificationRequestList*] [**-y** [*Comment*]] *File ...*

描述

delta 命令将通过 **get -e** 命令获得文件版本发生的任何修改信息引进到指定的源代码控制系统（SCCS）文件。

delta 命令读入符合指定文件 *s*（请参阅 **get** 命令获得由 SCCS 创建和使用文件的描述）的 g-file 文件并创建一个新的 delta。g-file 文件中每行都不超过 512 个字符。

如果您为 *File* 的值指定了文件的目录，**delta** 命令将对该目录中那些先前检出来编辑的所有 SCCS 文件（即所有前缀为 **s.** 的文件）执行请求的操作。如果您指定了 **-**（减号）来取代 *File* 值，**delta** 命令读入标准输入并且将每一行作为 SCCS 的文件名来解释。当 **delta** 命令从标准输入读入时，您必须提供 **-y** 标志。如果设置了 **v** 头标志，还必须提供 **-m** 标志。**delta** 命令读入标准输入直到它读到文件结束符。

> **注:** 以 SOH ASCII（二进制 001）字符开头的行不能放到 SCCS 文件中，除非使用 ****（反斜杠）将 SOH 引用起来。对于 SCCS，SOH 含有特殊含义并会造成错误。

当 **get** 命令产生大量的数据时，应该避免在对 SCCS 文件使用 **get** 命令后紧接着在同样的文件上使用 **delta** 命令。相反，您应该切换使用 **get** 命令和 **delta** 命令。

delta 命令保存对一个特殊版本的 SCCS 文件所作的修改。要使用 **delta** 命令:

1. 使用 **get -e** 命令获得文件的可编辑版本。
2. 编辑该文件。
3. 使用 **delta** 命令创建一个新版本的 SCCS 文件。

如果没有指定 **-y** 选项，**delta** 命令将提示您输入注释。注释应用于该特定 delta 并将出现在 SCCS 文件头中。在您使用 **get** 命令得到 delta 时并不会检索该注释，并且不会出现在检索文件的文本中。使用注释保持跟踪创建 delta 的原因。

要查看注释，使用编辑器查看 SCCS 文件，通过 **cat** 命令将 SCCS 文件显示在屏幕上，或者通过 **prs** 命令将选择的文件部分打印到标准输出。记住请勿在 SCCS 文件中直接更改内容。要更改 delta 注释，使用 **cdc** 命令。

> **注:** 如果文件包含了扩展标识关键字，请勿使用 **delta** 命令。只读文件版本用文本值替换关键字。对只读文件使用 **delta** 命令将会导致关键字的丢失。要从该状态恢复，除去 delta 或者再次编辑文件并替换标识关键字。

除非存在一个能够编辑的文件副本，否则 SCCS 不允许使用 **delta** 命令。

要防止关键字的丢失，使用 **admin** 命令并带有 **-f** 标志来指定 **i** 头标志。这样，缺少关键字的文件版本将会产生错误。

标志

-g *List*
指定 **get** 命令创建 g-file 文件时将要忽略的 SID 列表（deltas）。您使用这个标志之后，**get** 命令会在建立 g-file 文件时忽略指定的 delta。

-m *ModificationRequestList*
如果设置了 SCCS 文件的 **v** 头标志，则必须提供修改请求（MR）号作为创建新的 delta 的原因。

如果您没有指定 **-m** 标志，而设置了 **v** 头标志，则 **delta** 命令从标准输入中读取 MR。如果标准输入是一个工作站，**delta** 命令提示您输入 MR。**delta** 命令继续接受输入直到它读到文件结尾字符。它将始终在注释之前读入 MR（请参阅 **-y** 标志）。您可以使用空格、制表符或者两者共用来在列表中分开 MR。

如果 **v** 头标志有值，它将解释为验证 MR 数的程序的名称。如果 **delta** 命令从 MR 验证程序返回一个非零的退出值，则 **delta** 命令认为某些 MR 号是无效的并停止其运行。

-n
保留 g-file 文件，它通常在完成 **delta** 命令处理后就被删除。

-p
将运行 delta 命令之前和之后 SCCS 文件的区别写到标准输出（以 **diff** 命令格式）中去。请参阅 **diff** 命令获得更多关于格式的解释。

-r *SID*
指定在 SCCS 文件中创建哪个 delta。仅当由同一用户在相同的 SCCS 文件上执行了两次或多次未完成的 **get -e** 命令时，您才能使用该标志。*SID* 的值可以是 **get** 命令行中指定的 SID 也可以是将要创建的 SID（由 **get** 命令报告）。如果指定的 SID 无法专门地标识或者如果 SID 必须指定而没有指定，将会导致错误。

-s
禁止在完成 **delta** 之后在标准输出中写入常规信息。

-y[*Comment*]

指定表述生成 delta 的原因的文本。空字符串将会认为是一个有效的 *Comment* 值。如果您的注释行包括了特殊字符或者空格，该行必须放入单引号或者双引号中。

如果您不指定 **-y** 标志，则 **delta** 命令从标准输入中读入注释直到遇到空行或者文件结束符。

对于键盘输入，**delta** 命令将会提示注释内容。如果一行的最后一个字符是 \ （反斜杠），它将被忽略。注释不得超过 512 个字节。

退出状态

此命令返回以下退出值:

0 成功结束。
>0 发生错误。

示例

1. 要将您做出的修改记录到 SCCS 文件中，请输入:

   ```
   delta s.prog.c
   ```

   这添加了一个 delta 到 SCCS 文件 `s.prog.c` 中，通过编辑 `prog.c` 来记录发生的修改。`delta` 程序将提示您输入注释总结所作的修改。输入注释，然后输入文件结束符或者按下两次返回键说明您已经完成了注释。
2. 要带简单描述注释将所做的修改记录到一个 SCCS 文件，请输入:

   ```
   delta -y "This delta contains the payroll function" s.prog.c
   ```

文件

/usr/bin/delta 包含 **delta** 命令。

相关信息

admin 命令、**cat** 命令、**cdc** 命令、**diff** 命令、**get** 命令、**prs** 命令、**rmdel** 命令、**sccsdiff** 命令以及 **sccshelp** 命令。

sccsfile 文件格式。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『SCCS 命令列表』。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『源代码控制系统（SCCS）概述』。

---

deroff 命令

用途

除去文件中 **nroff**、**troff**、**tbl** 以及 **eqn** 命令构造。

语法

deroff { **-ma -me -ms** [**-mm** [**-ml**]] } [**-i** | **-l**] [**-k**] [**-p**] [**-u**] [**-w**] [*File ...*]

描述

deroff 命令读入指定的文件（缺省为标准输入）其中包括英文文本，命令所有的 **troff** 请求、宏调用、反斜杠构造、**eqn** 命令构造（包含在 **.EQ** 和 **.EN** 命令行之间以及定界符之间）以及 **tbl** 命令描述，然后将文件剩余的部分写入到标准输出中。

deroff 命令后通常跟有一串包含文件（**.so** 和 **.nx troff** 命令请求）。如果一个文件已经包括在其中，一个命名它的 **.so** 请求将会被忽略，而命名该文件的 **.nx** 请求将会结束执行。

> **注:** **deroff** 命令不是完整的 **troff** 命令解释器，因此它可能会因微妙的构造变混乱。大多数错误导致输出太多而不是太少。

参数

File 为 **deroff** 命令除去 **troff**、**eqn** 和 **tbl** 命令执行的影响指定英文文本文件。缺省文件是标准输入。

标志

-ma 在文本中忽略 **MA**（**man**）宏使得只有运行中的文本才输出。
-me 在文本中忽略 **ME** 宏使得只有在运行中的文本才输出。此为缺省值。
-ml 在文本中忽略 **MM** 宏（**-mm** 标志）并且删除 **MM** 列表结构。**-mm** 标志必须与该标志同时使用。
> **注:** 请勿将**-ml** 标志和嵌套列表同时使用。

-mm 忽略 **MM** 宏。
-ms 在文本中忽略 **MS** 宏使得只有运行的文本才输出。
-i 不进行包含文件的处理。
-l 不进行以 **/usr/lib** 开头的包含文件的处理，例如 **/usr/lib/tmac** 中的宏文件。
-k 保留指定放在一起的块。缺省值为除去保留的文本块；例如 **.ne** 结构就会被除去。
-p 处理特殊的段落。
-u 除去 ASCII 下划线和粗体字控制序列。该标志自动设置 **-w** 标志。
-w 生成输出单词列表，每行一个单词而其他的字符将会删除。否则，输出遵循初始设置。

在文本中，一个字就是指任何以字母开头的字符串，包含至少两个字母并且是由字母、数字、与符号（&）以及省字号（'）组成。然而在宏调用中一个字是以至少两个字母开头并且至少有 3 个字母组成的字符串。定界符是除了字母、数字、标点、省略号以及与符号之外的任意字符。从字中除去尾部的省略符号以及与符号。

相关信息

eqn 命令、**neqn** 命令、**nroff** 命令、**tbl** 命令、**troff** 命令。

---

detachrset 命令

用途

从进程中拆离 rset。

语法

detachrset [**-P**] *pid*

描述

detachrset 命令从一个进程中拆离一个 rset。从进程中拆离 rset 将会允许进程使用系统中的任何处理器与 / 或内存区域。

标志

-P 从指定的进程（*pid*）中拆离部分 rset。

参数

pid 进程标识。

安全性

用户必须拥有 root 权限或者具有 CAP_NUMA_ATTACH 能力并且目标进程必须有与命令发出者同样有效的 **userid**。该用户必须拥有 root 权限才能从进程中除去部分 rset（**-P** 选项）。

示例

要从进程 21414 中拆离 rset，请输入:

```
detachrset 21414
```

文件

/usr/bin/detachrset 包含 **detachrset** 命令。

相关信息

attachrset、**execrset**、**lsrset**、**mkrset** 以及 **rmrset** 命令。

---

devinstall 命令

用途

为设备安装软件支持。

语法

devinstall **-f** *file* **-d** *device* [**-s**] [**-v**]

描述

devinstall 命令为设备安装软件支持。它将安装由 **-f** 标志指定的文件中列出的软件包。

对于多数在初始软件安装后需要被添加的新设备来说，可以通过使用 **cfgmgr** 的 **-i** 标志来安装新设备的软件。

在某些情况下，新设备替换启动机器所需的设备。例如，您可以替换支持根卷组的 SCSI 适配卡，或者支持控制台的图形适配卡。在这种情况下，除非您已经安装了支持新设备的软件，否则计算机将不能以正常方式启动。为此，关闭您的系统然后按照硬件说明书安装新硬件。接着，以维护方式启动您的计算机。在启动过程

中，会检测到新的适配器，而且文件 **/tmp/device.pkgs** 将被创建，它包含支持新硬件所需的软件包的名字。只要计算机在维护方式下，您就能通过运行命令 **devinstall** 来安装支持新设备的软件。

标志

-f *File* 指定包含待装软件包列表的文件。通常情况下，这个文件就是 **/tmp/device.pkgs**，它通过命令 **cfgmgr** 创建。

-d *Device* 指明安装介质的位置。这可以是一个硬件设备，例如磁带或软盘；也可以是包含安装映像的目录；或者就是安装映像文件本身。当安装介质是 IBM 安装磁带，或 IBM 校正服务磁带，磁带设备应该被指定为 no-rewind-on-close 和 no-retension-on-open。例如，文件 **/dev/rmt0.1** 针对高密磁带，而文件 **/dev/rmt0.5** 针对低密磁带。对于非 IBM 供应的磁带，请使用磁带供应商指定的选项。

-s 覆盖 **/var/adm/dev_pkg.fail** 文件。这个文件包含所有未成功安装的软件包列表，可以用来简化通过其他数据结束恢复或者安装的过程。

-v 指定详细选项，使得命令 **devinstall** 在执行过程中显示附加信息。

命令 **devinstall** 会将命令行所指定的文件中列举的设备软件包逐一安装。它运行命令 **geninstall**，使用参数 **-I** "acXge **/var/adm/ras/devinst.log**"，其中 a: 应用、c: 提交、X: 扩展文件系统、e: 日志（日志文件的全路径名是 **/var/adm/ras/devinst.log**）、g: auto_include。（请参阅 **geninstall** 命令以获得这些标志的更多信息）。**devinstall** 命令检查由 **geninstall** 命令生成的、描述每个软件包安装尝试结果的摘要文件，在该信息的基础上，它将创建两个文件。**/var/adm/dev_pkg.fail** 文件列举所有安装失败的软件包（如果存在）。**/var/adm/dev_pkg.success** 文件列出所有安装成功的软件包。

返回值

返回值 0 表示没有软件包安装成功。

返回值 1 表示至少一个包成功安装，而且命令 **bosboot** 应该被执行。

返回值 2 表示 **devinstall** 命令失败。

/var/adm/dev_pkg.success 文件列出所有安装成功的软件包。**/var/adm/dev_pkg.fail** 文件列举所有安装失败的软件包。

安全性

特权控制: 只有 root 用户可以执行这个命令。

示例

您可以在从设备安装磁带启动计算机并进入维护方式之后安装支持新设备的软件，请输入:

```
devinstall -f /../tmp/device.pkgs -d /dev/rmt0.1
```

然后运行命令 **bosboot**。

```
bosboot -ad /dev/ipldevice
```

文件

/dev/rmtn 指定原始的流式磁带接口。

相关信息

bosboot 命令、**cfgmgr** 命令、**installp** 命令。

---

devnm 命令

用途

命名一个设备

语法

devnm *Path ...*

描述

命令 **devnm** 读取 *Path* 参数，标识与 *Path* 参数所驻留的文件系统相关的一个特定文件，并将该特殊文件名写到标准输出。每个 *Path* 参数必须是一个全路径名。

devnm 命令的最常用用法是通过 **/etc/rc** 命令文件为根设备构造一个装载表项。

注: 该命令只用于本地文件系统。

示例

1. 要标识文件所驻留的设备，请输入:

   ```
   devnm /diskette0/bob/textfile
   ```

   这将显示 /diskette0/bob/textfile 文件所驻留的特定设备文件的名字。如果一个软盘安装为 /diskette0 设备，则 **devnm** 命令显示:

   ```
   fd0 /diskette0/bob/textfile
   rfd0 /diskette0/bob/textfile
   ```

   这表示 /diskette0/bob/textfile 文件驻留在 **/dev/fd0** 软盘驱动器上。
2. 要标识文件系统所驻留的设备，请输入:

   ```
   devnm /
   ```

   这将显示根文件系统（/）所驻留的设备名。屏幕上将显示以下列表:

   ```
   hd0 /
   ```

   这意味着根文件系统（/）驻留在 **/dev/hd0** 设备上。

文件

/dev	指定目录。
/usr/sbin/devnm	包含命令 **devnm**。

相关信息

rc 命令。

df 命令

用途

报告文件系统上的空间信息。本文档描述了 AIX **df** 命令，以及 System V 版本的 **df**。

语法

df [[**-P**] | [**-l** | **-M** | **-i** | **-t** | **-v**]] [**-k**] [**-m**] [**-g**] [**-s**] [*FileSystem ...* | *File...*]

描述

df 命令显示文件系统的总空间和可用空间信息。*FileSystem* 参数指定文件系统驻留的设备的名称，文件系统的安装目录或文件系统的相对路径名。*File* 参数指定非安装点的文件或目录。如果指定 *File* 参数，**df** 命令显示该文件或目录所在文件系统的信息。如果您未指定 *FileSystem* 或 *File* 参数，命令 **df** 显示当前已安装的所有文件系统信息。在缺省情况下，文件系统的统计信息以 512 字节的块单元显示。

df 命令通过 **statfs** 系统调用得到文件系统的空间统计信息。然而，如果指定了 **-s** 标志，则从虚拟文件系统（VFS）的文件系统帮助中取得统计信息。如果您不用 **-s** 标志指定参数，而且帮助系统无法获取统计信息，则采用 **statfs** 系统调用统计信息。在某些例外情况下，例如运行 **df** 命令时，文件系统正在被修改，则 **df** 命令显示的统计信息可能并不正确。

注: 一些远程文件系统，例如网络文件系统（NFS），并不提供 **df** 命令所需的所有信息。**df** 命令对于服务器不提供的统计信息打印空格。

df 命令不完全支持 NFSv4 文件系统。请使用 **nfs4cl** 命令来抽取块和空间信息。

标志

-g 以 GB 块为单位显示统计信息。由于文件系统统计信息的输出值为浮点数形式，因此每个单元字节的值较大。

-i 显示文件系统可用和已用的索引节点的数目；当指定文件系统已被安装时，这是缺省输出设置。

-l 显示总块数、已用空间、可用空间、使用空间的百分比、文件系统的安装点的信息。

-k 以 1024 字节块为单位显示统计信息。

-m 以 MB 块为单位显示统计信息。文件系统统计信息的输出值为浮点数形式，因为每个单元字节的值较大。

-M 在第 2 列中显示文件系统的安装点信息。

-P 以 POSIX 轻便格式显示文件系统的信息。

当指定 **-P** 标志时，标题行类似于如下显示:

```
Filesystem 512-blocks Used Available Capacity Mounted on\n
```

如果 **-k**、**-m** 或 **-g** 标志在 **-P** 标志外被指定，列标题的 512 块将被相对单位所代替，取决于哪些标志同 **-P** 标志一起使用。

文件系统的统计信息将按下面顺序一行显示:

文件系统、*总空间*、*已用空间*、*可用空间*、*使用百分比*、*安装点*。

-s 从 VFS 的文件系统帮助获取文件系统统计信息，而不是 **statfs** 系统调用。当使用 **-s** 标志时，任何给定参数必须是一个 JFS 或高级 JFS 文件系统的安装点或设备。文件系统也必须被列在 **/etc/filesystems** 中。

-t 在输出中包含总共分配空间的数量。

-v 显示指定文件系统的所有信息。

带标志 **-m** 和 **-g** 的输出参数值将被四舍五入到第 2 位十进制小数位。如果 **-k**、**-m** 和 **-g** 标志被同时指定或指定任意两个，则最后指定的一个标志有效。

退出状态

此命令返回以下退出值:

0 成功结束。
>0 发生错误。

示例

1. 要显示所有已安装文件系统的信息，请输入:

   ```
   df
   ```

   如果您的系统安装了 **/**、**/usr**、**/site** 和 **/usr/venus** 文件系统，则 **df** 命令的输出类似于下面:

   ```
   Filesystem 512-blocks Free   %Used   Iused  %Iused  Mounted on
   /dev/hd0    19368     9976    48%     4714    5%     /
   /dev/hd1    24212     4808    80%     5031   19%     /usr
   /dev/hd2     9744     9352     4%     1900    4%     /site
   /dev/hd3     3868     3856     0%      986    0%     /usr/venus
   ```

2. 要以 1024 字节块为单位显示文件系统 /test 的信息，请输入:

   ```
   df -k /test
   ```

   ```
   Filesystem    1024 blocks     Free    %Used   Iused  %Iused  Mounted on
   /dev/lv11          16384     15824       4%      18      1%  /tmp/ravi1
   ```

   即以 1024 字节块为单位显示文件系统的统计信息。

3. 要以 MB 为单位显示文件系统 /test 的信息，请输入:

   ```
   df -m /test
   ```

   ```
   Filesystem    MB blocks     Free    %Used    Iused  %Iused  Mounted on
   /dev/lv11       16.00     15.46       4%       18      1%  /tmp/ravi1
   ```

   即以 MB 为单位显示文件系统统计信息，并四舍五入到小数点第 2 位。

4. 要以 GB 为单位显示文件系统 **/test** 的信息，请输入:

   ```
   df -g /test
   ```

   ```
   Filesystem    GB blocks   Free     %Used    Iused  %Iused  Mounted on
   /dev/lv11          0.02   0.02        0%       18      1%  /tmp/ravi1
   ```

   即以 GB 为单位显示文件系统统计信息，并四舍五入到小数点第 2 位。

5. 要显示当前目录所在文件系统的可用空间，请输入:

   ```
   cd/
   df .
   ```

   该命令的输出类似以下格式:

   ```
   Device   512-blocks  free   %used   iused   %iused  Mounted on
   /dev/hd4    19368    9976    48%     4714    5%     /
   ```

文件

/etc/filesystems 列出已知的文件系统并且定义它们的特征。
/var/spool/mail/* 包含虚拟文件系统类型的描述。

相关信息

fsck 命令。

filesystems 文件。

《*操作系统与设备管理*》中的『文件系统』说明了文件系统类型、管理、结构和维护。

《*操作系统与设备管理*》中的『安装』说明了安装文件和目录、安装点和自动安装。

System V df 命令

用途

报告可用磁盘块和文件的数量。

语法

/usr/sysv/bin/df [**-a**] [**-l**] [[[**-e**] [**-g**] [**-n**]] | [[**-i**] [**-v**]] | [**-t**]] [*FileSystem* ...] [*File* ...]

描述

命令 **df** 显示文件系统的总空间和可用空间信息。文件系统的统计信息以 512 字节块为单位显示。

标志

-a 运行缺省操作，打印安装点、设备名、可用块数目和已用节点（文件）的数量。
-e 只打印可用文件的数量。
-g 打印 **statvfs** 的完整结构。该选项覆盖 **-a**、**-e**、**-i**、**-n**、**-t** 和 **-v** 选项。可用块数、总块数和可用块数以 512 字节块为单位报告。
-i 显示总节点数、可用节点数、已用节点数，以及节点使用百分比。
-l 只报告本地文件系统。
-n 打印文件系统类型。
-t 报告已分配的总块数。
-v 报告已用块的百分比，和已用块及可用块的数目。

参数

File *File* 参数指定非安装点的文件或目录。如果指定了 *File* 参数，**df** 命令显示该文件或目录所在文件系统的信息。
FileSystem *FileSystem* 参数指定文件系统所在的设备名、文件系统所安装的目录，或者文件系统的相对路径名。

注: 如果未指定 *FileSystem* 或者 *File* 参数，**df** 命令显示当前安装的所有文件系统信息。

退出状态

0 成功完成命令。
>0 发生错误。

示例

1. 要显示所有已安装文件系统的信息，请输入:

   ```
   /usr/sysv/bin/df
   ```

   输出类似以下格式:

   ```
   /              (/dev/hd4       ):     19656 blocks    1504 files
   /usr           (/dev/hd2       ):   1139904 blocks   20254 files
   /var           (/dev/hd9var    ):     23096 blocks     512 files
   /tmp           (/dev/hd3       ):      2464 blocks     204 files
   /home          (/dev/hd1       ):     44208 blocks     146 files
   /proc          (/proc          ):         0 blocks       0 files
   /opt           (/dev/hd10opt   ):     13880 blocks     310 files
   ```

2. 要显示当前目录所在文件系统的信息，请输入:

   ```
   /usr/sysv/bin/df .
   ```

3. 要显示所有已安装文件系统的总节点数、空闲节点数和可用节点数，请输入:

   ```
   /usr/sysv/bin/df -i
   ```

   输出类似以下格式:

   ```
   Mount Dir  Filesystem      iused    avail    itotal  %iused
   /          /dev/hd4         1504     6688      8192     19%
   /usr       /dev/hd2        20254   127202    147456     14%
   /var       /dev/hd9var       512     3584      4096     13%
   /tmp       /dev/hd3          204     5940      6144      4%
   /home      /dev/hd1          146    14190     14336      2%
   /proc      /proc               0        0         0       0
   /opt       /dev/hd10opt      310     5834      6144      6%
   ```

4. 要显示文件系统 **/tmp** 的总块数、已用块数和可用块数，请输入:

   ```
   /usr/sysv/bin/df -v /tmp
   ```

5. 要显示文件系统类型，请输入:

   ```
   /usr/sysv/bin/df -n
   ```

6. 要显示所有本地文件系统的索引节点信息，请输入:

   ```
   /usr/sysv/bin/df -i -l
   ```

7. 要显示所有文件系统的 statvfs 结构信息，请输入:

   ```
   /usr/sysv/bin/df -g
   ```

8. 要显示文件系统上的可用文件数，请输入:

   ```
   /usr/sysv/bin/df -e
   ```

文件

/usr/sysv/bin/df	包含 System V 的 **df** 命令。
/etc/filesystems	包含文件系统信息。

相关信息

/usr/bin/df 命令。

dfmounts 命令

用途

显示已安装的资源信息。

语法

dfmounts [**-F** *fstype*] [**-h**] [*server* ...]

描述

dfmounts 命令显示客户机通过网络文件系统（NFS）由客户机远程装载的本地系统。它也显示装载资源的客户机列表。**dfmounts** 命令显示一个头，其后是以空白字符分隔的资源信息字段列表。

对于每一个资源，显示以下字段:

RESOURCE
对于 NFS 而言，标记连字符号 “-”。

SERVER
指示资源从哪台机器装载。

PATHNAME
指示共享资源的路径。

CLIENTS
目前已安装资源的系统列表，以逗号分隔。

标志

-F *fstype*　指明文件系统类型（fstype）。仅支持 **nfs** 文件系统类型。

-h　禁止 **dfmounts** 输出中的头行。

参数

Server　代表网络上的一个系统，其上资源可以为本地系统所用。*Server* 显示机器上的可用资源，以及当前使用每一资源的客户机。如果未指定该参数，则 **dfmounts** 命令通过假定服务器为本地系统来打印信息。**dfmounts** 命令可提供多个服务器的名字。

退出状态

0　成功完成命令。

>0　发生错误。

安全性

示例

1. 要打印系统 “mercury” 在文件系统 “**nfs**” 上的已安装资源信息，请输入:

```
dfmounts -F nfs mercury
```

2. 要打印系统在文件系统“**nfs**”上的已安装资源信息（省略报头），请输入:

```
dfmounts -hF nfs
```

文件

/usr/bin/dfmounts	包含通用 System V 的 **dfmounts** 命令。
/usr/lib/fs/nfs/dfmounts	包含 System V 的 NFS **dfmounts** 命令。
/var/spool/mail/*	包含已知虚拟文件系统实现的描述。

相关信息

dfshares 命令。

---

dfpd 命令

用途

向负载管理器提供关于负载平衡的服务器的负载统计信息。

语法

/usr/sbin/dfpd [**-d**] [**-f** *ConfigurationFile*]

描述

DFP 守护程序（**dfpd**）运行于负载平衡的服务器上，并向负载管理器提供服务器的负载统计信息。这使得负载管理器能够为更加可用的服务器发送未来的连接，这有助于实现负载平衡。

当 **dfpd** 守护程序启动时，它从 *ConfigurationFile* 参数指定的文件读取配置信息。如果参数未指定，**dfpd** 守护程序从 **/etc/dfpd.conf** 文件读取配置信息。

一旦启动，**dfpd** 守护程序在配置文件所指定的端口上侦听来自负载管理器上的连接。

DFP 守护程序配置文件

/etc/dfpd.conf 文件可以通过编辑来更新。**/etc/dfpd.conf** 文件中的项包括以下信息:

密钥项 MD5 指定了密钥（最多 64 个字符）应该在 DFP 客户机、服务器和负载管理器之间保持一致。密钥项 MD5 的一个示例为:

```
md5key 1234567890abcdefabcdef12345678901234567890abcdefabcdef1234567890
```

负载管理器侦听器项指定了 DFP 服务器在哪个端口侦听负载管理器连接。一个负载管理器条目的示例为:

```
ldlistener 9503
```

轮询空闲时间项指定了 CPU 空闲时间的连续计算周期。轮询空闲时间条目的一个示例为:

```
pollidletime 30
```

空闲时间值在发送给负载管理器之前，需要乘以 *mfactor* 参数。这有助于不同容量的计算机之间的权重合理化。缺省值为主机上的 CPU 数。一个 mfactor 条目的实例为:

```
mfactor 1
```

标志

-d 在调试方式下运行，不成为一个守护程序。
-f *ConfigurationFile* 使守护程序使用指定的 *ConfigurationFile*。

---

dfsck 命令

用途

在不同驱动器上同时检查和修复两个文件系统。

语法

dfsck [*FlagList1*] *FileSystem1* [*FlagList2*] *FileSystem2*

描述

dfsck 命令可以使您同时检查两个不同的驱动器上的两个文件系统。利用 *FlagList1* 和 *FlagList2* 参数为两组文件系统传递标志和参数。可以通过介绍标志的部分查看 *FlagList1* 和 *FlagList2* 的一系列有效标志。如果您指定一些标志作为部分参数，用 **-**（负号）来分开文件系统组。

dfsck 命令允许您同时与两个 **fsck** 命令交互。为了实现该目的，**dfsck** 命令为每个消息显示文件系统名称。当响应一个来自 **dfsck** 命令的问题时，在您的响应上加前缀 1 或 2，表示响应是针对第一个还是第二个文件系统组。

注: 请勿利用 **dfsck** 命令检查根文件系统。

标志

-d *BlockNumber* 搜索指定磁盘块的引用。无论何时 **fsck** 命令遇到包含指定块的文件，它将会显示索引节点数和所有指向它的路径名称。

-f 进行快速检查。在正常情况下，唯一可能被不适当关机停止系统影响的文件系统是系统停止时安装的那些文件系统。**-f** 标志会提示 **fsck** 命令不要检查那些已经成功卸载的文件系统。**fsck** 命令通过检查文件系统超块中的 **s_fmod** 标志来决定这件事。当安装了文件系统时，将设置该标志；在其被成功卸载时，清除该标志。如果文件系统被成功地卸载，这不大可能会存在什么问题。由于多数文件系统都能成功卸载，不检查这些文件系统能减少检查时间。

-i-*NodeNumber* 搜索指定索引节点的引用。无论何时 **fsck** 命令遇到一个指定索引节点的目录引用，它都会显示这个引用的完整路径名称。

-n 对 **fsck** 命令提出的所有问题假定一个否定回应；不打开指定的文件系统来写。

-o *Options* 向 **fsck** 命令传递逗号分隔的选项。假定这些选项是文件系统特定执行的，除了以下当前支持所有文件系统的选项:

mountable

如果询问的文件系统是可安装的（清除），则会使 **fsck** 命令成功地退出，返回一个为 0 的值。如果文件系统不可安装，**fsck** 命令退出并返回一个为 8 的值。

mytype

如果有问题的文件系统和在 **/etc/filesystems** 文件中或者在命令行中通过 **-V** 标志指定的类型相同，则使 **fsck** 命令退出成功（0）。否则，返回 8。例如，如果 **/**（根文件系统）是一个日志文件系统，则 `fsck -o mytype -V jfs /` 将退出并返回为 0 的值。

-p 不显示次要问题的消息但是可以自动修复问题。这个标志并不象 **-y** 标志那样授权大规模许可，当系统正常启动的时候对执行自动检查有用。无论系统在何时自动运行，您应该将这个标志作为系统启动过程的一部分来使用。也允许通过分组并行检查。

-t*File* 如果 **fsck** 命令得不到足够的内存来保存它的表的话，指定一个 *File* 参数而不是被检查的文件作为文件系统的暂存文件。如果没有指定 **-t** 标志而 **fsck** 命令需要一个临时文件，则它会提示您输入暂存文件的名字。但是，如果指定了 **-p** 标志，**fsck** 命令不会成功。如果暂存文件不是一个特殊的文件，当 **fsck** 命令结束的时候，它就会被除去。

-V *VfsName* 使用文件系统的 *VFSName* 变量指定的虚拟文件系统描述而不是用 **/etc/filesystems** 文件来决定描述。如果 **-V** *VfsName* 标志没有在命令行中指定，就会检查 **/etc/filesystems** 文件并且匹配节的 **vfs=***Attribute* 被假定是正确的文件系统类型。

-y 对 **fsck** 命令提出的所有问题假定一个肯定响应。这个标志使 **fsck** 命令采取它认为必要的行动。仅在损坏严重的文件系统中使用这个标志。

示例

1. 要同时检查在两个不同驱动器上的两个文件系统，请输入:

   ```
   dfsck  -p /dev/hd1 -  -p /dev/hd7
   ```

   如果 **/dev/hd1** 和 **/dev/hd7** 设备上的文件系统位于两个不同的驱动器上，该命令将同时检查两个文件系统。您也可以指定在 **/etc/filesystems** 文件中找到的文件系统名称。

文件

/usr/sbin/dfsck 包含 **dfsck** 命令。
/etc/filesystems 列出已知的文件系统并且定义它们的特征。
/var/spool/mail/* 包含虚拟文件系统类型的描述。
/usr/bin/from 包含当系统启动的时候运行的命令（包括 **fsck** 命令）。

相关信息

fsck 命令、**fsdb** 命令、**istat** 命令、**mkfs** 命令、**ncheck** 命令、**rc** 命令、**shutdown** 命令。

filesystems 文件、**filsys.h** 文件。

《*操作系统与设备管理*》中的『文件系统』说明了文件系统类型、管理、结构和维护。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

《*操作系统与设备管理*》中的『系统管理界面工具』说明了 SMIT 结构、主菜单和任务。

---

dfshares 命令

用途

列举远程系统上的可用资源。

语法

dfshares [**-F** *FileSystemType*] [**-h**] [*Server ...*]

描述

dfshares 命令提供通过网络文件系统可被主机获取的资源信息。**dfshares** 命令显示头行，后跟包含以空格为字段分隔符的行列表。

每一资源包含以下字段:

RESOURCE
显示以服务器: 路径格式导出的资源名。

SERVER
显示提供资源的计算机。

ACCESS
显示授权给客户机系统的访问许可权。然而，**dfshares** 命令无法确定 NFS 资源的该信息，因此将连字符 “-” 植入该字段。

TRANSPORT
显示共享资源所在的传输提供程序。然而，**dfshares** 命令无法确定 NFS 资源的该信息，因此将连字符 “-” 植入该字段。

标志

-F *FileSystemType*	指定文件系统类型。仅支持 **nfs** 文件系统类型。
-h	禁用 **dfmounts** 命令输出中的头行。

参数

Server	表示网络上向本地机提供资源的系统。如果未指定该参数，则 **dfshares** 命令显示本地机的信息。**dfshares** 命令可以指定多个服务器名称。

退出状态

0 命令成功结束

>0 发生错误。

示例

1. 要打印系统 “mercury” 上 **nfs** 类型文件系统的资源信息，请输入:

   ```
   dfshares -F nfs mercury
   ```

2. 要不带报头打印系统资源信息，请输入:

   ```
   dfshares -hF nfs
   ```

文件

/usr/bin/dfshares	包含通用 System V 的 **dfshares** 命令。
/usr/lib/fs/nfs/dfshares	包含针对 **nfs** 类型文件系统的 System V **dfshares** 命令。
/var/spool/mail/*	包含已知虚拟文件系统实现的描述。

相关信息

dfmounts 命令。

---

dhcpaction 命令

用途

提供客户机每次更新其租用时运行的脚本。

语法

/usr/sbin/dhcpaction *HostName DomainName IPAddress LeaseTime ClientID* { **A** | **PTR** | **BOTH** | **NONE** } { **NONIM** | **NIM** }

描述

dhcpaction 命令提供了更新 DNS 服务器的方法，它通过用恰当顺序的事件调用 **nsupdate** 命令来更新 A 记录、PTR 记录或者两者同时更新。**dhcpaction** 命令被 DHCP 客户机和服务器守护程序调用。它从 DNS 更新字符串调用。这是可配置的，因为某些环境（主要是异构环境）下，一些客户机可能无法更新 A 记录或者 PTR 纪录。缺省动作是客户机更新 A 记录，服务器更新 PTR 记录。这些选项可以在守护程序配置文件中设置以允许网络管理员想要的任何策略。

dhcpaction 命令也允许您同时运行 NIM 和 DHCP。当给定 NIM 参数时，**dhcpaction** 命令在 NIM 对象的 IP 地址更改时，将向它们尝试和发送更新。这保持对象同步。为此，一些挂起的操作可能必须被取消。对象将被注释，并且一条消息将被发给主机控制台。对象不应该经常重新设置。在 DHCP 环境下，地址不应该经常更改。只有客户机应该设置 NONIM 选项。

参数

ClientID	指定要在更新 DNS 服务器时使用的客户机标识。
DomainName	更新 DNS 服务器时指定所用的域名。
HostName	指定 DNS 服务器中要尝试和更新的主机名。
IPAddress	指定 DNS 服务器中要与主机名关联的 IP 地址。
LeaseTime	以秒为单位指定 DNS 服务器中主机名和 IP 地址之间关联的持续时间。

选项

A | **PTR** | **BOTH** | **NONE**	如果 DNS 服务器中的任何记录需要更新，则指定需要更新哪个记录。
NONIM | **NIM**	指定脚本是否应该采取动作以帮助 NIM 和 DHCP 正确交互。这在 DHCP 服务器上只能设置为 NIM。

退出状态

此命令返回以下退出值:

0	成功结束。
>0	发生错误。

安全性

访问控制: 任何用户，但是某些 NIM 操作只有 root 用户才可以执行。

文件

/usr/sbin/dhcpaction	包含 **dhcpaction** 命令。
/etc/dhcpcd.ini	包含 DHCP 客户机配置文件

相关信息

inetd 守护程序、**dhcpsd** 守护程序、**dhcprd** 守护程序。

DHCP 客户机配置文件

DHCP 服务器配置文件

bootp 配置文件

TCP/IP 地址和参数分配 - 动态主机配置协议

《*网络与通信管理*》中的『TCP/IP 守护程序』。

---

dhcpcd 守护程序

用途

实现动态主机配置协议（DHCP）客户机。服务器地址和 DHCP 服务器的配置信息。

语法

通过系统资源控制器实现一个 DHCP 客户机:

startsrc -s dhcpcd [**-a***Argument*] ...

不通过系统资源控制器实现一个 DHCP 客户机:

dhcpcd [**-f** *ConfigurationFile*] [**-i** *IPAddress*] [**-l** *LeaseFile*] [**-n**] [**-o** *OptionsFile*] [**-r**] [**-t** *Seconds*] [**-t** *Minutes*]

描述

dhcpcd 守护程序通过设置 IP（因特网协议）地址来实现 DHCP 客户机，并通过使用 DHCP 协议来实现其他参数。

dhcpcd 守护程序正常情况下由通常运行于系统引导时的 **/etc/rc.net** 文件启动。在缺省情况下，这将被注释掉，而且不在机器启动时运行。smit 选项可以启用 DHCP 客户机。

dhcpcd 守护程序读取它的配置文件，并在配置文件中试图提取并得到一个 IP 地址，以及其他针对特定接口的配置选项。**dhcpcd** 守护程序在系统启动后运行于后台。它将根据需要更新一个已经接收的地址。

在使用 **-i** 标志时，**dhcpcd** 守护程序也以 DHCP Inform 方式运行。该方式允许客户机在没有 IP 地址的情况下检索服务器 DHCP 上的配置信息。这对静态地址很有用，但是对诸如打印服务器等动态项和其他选项没用。**dhcpcd** 守护程序对于指定地址将运行一次。

refresh 命令可以使 **dhcpcd** 守护程序重新读取配置文件。SIGHUP 也可用于获取相同的响应。

dhcpcd 的缺省配置文件是 **/etc/dhcpcd.ini**。它包含日志和网络接口信息。

您可以通过基于 Web 的系统管理器应用程序（**wsm network** 快速路径）来运行这个命令。您也可以使用系统管理接口工具（SMIT）**smit usedhcp** 快速路径来运行该命令。

标志

-f *ConfigurationFile*	指定要用的配置文件。缺省为 **/etc/dhcpcd.ini** 文件。
-i *IPAddress*	指定 **dhcpcd** 守护程序使用 DHCP Inform 方式。IP 地址告诉 DHCP 到哪个接口去获取配置信息。
-l *LeaseFile*	指定不同的租约文件。当客户机获取租约时生成租约文件。在缺省情况下，租约文件为 **/etc/dhcpc.db**。
-n	防止在收到新的地址时重新配置接口。
-o *OptionsFile*	指定选项文件。在缺省情况下，选项文件为 **/etc/dhcpc.opt**。
-r	启动客户机守护程序，然后在运行一次后关闭。
-t *Seconds*	指定 **dhcpcd** 置其自身于后台之前将等待的秒数。如果没有找到 DHCP 服务器，这允许机器继续引导。
-t *Minutes*	指定时间（以分钟计）。如果 dhcp 客户机无法在此超时值内为接口配置地址（例如，由于 dhcp 服务器不可用），它将停止进一步的尝试。

退出状态

此命令返回以下退出值:

0	成功结束。
>0	发生错误。

安全性

访问控制: 您必须拥有 root 用户权限才能运行这个命令。

文件

/usr/sbin/dhcpcd	包含 **dhcpcd** 守护程序。
/etc/dhcpcd.ini	包含缺省客户机配置文件
/etc/services	定义用于网络服务的套接字和协议。
/etc/inetd.conf	定义 **inetd** 守护程序控制的服务。

相关信息

dhcpsconf 命令。

startsrc 命令、**stopsrc** 命令。

inetd 守护程序、**dhcpsd** 守护程序、**dhcprd** 守护程序。

/etc/inetd.conf 文件格式、**/etc/services** 文件格式。

DHCP 客户机配置文件

DHCP 服务器配置文件

bootp 配置文件

TCP/IP 地址和参数分配 – 动态主机配置协议

《*操作系统与设备管理*》中的『系统资源控制器』说明了子系统、子服务器以及系统资源控制器。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』 。

《*网络与通信管理*》中的『TCP/IP 守护程序』。

---

dhcpcd6 守护程序

用途

实现 IPv6 动态主机配置协议（DHCPv6）客户机。从 DHCPv6 服务器获取 IPv6 节点的 IPv6 地址和配置信息。

语法

要使用系统资源控制器启动 DHCPv6 客户机:

startsrc -s dhcpcd6 [**-a** *Argument*] ...

不使用系统资源控制器启动 DHCPv6 客户机:

dhcpcd6 [**-f** *ConfigurationFileName*] [**-u** *Client_duid_File*] [**-p** *ClientPort*] [**-t** *SolicitTimeout*]

描述

dhcpcd6 守护程序通过使用 DHCPv6 协议设置 IPv6（因特网协议 V6）地址和其他参数实现 DHCPv6 客户机。

dhcpcd6 守护程序正常情况下由通常在系统引导时运行的 **/etc/rc.net** 文件启动。在缺省情况下，这将被注释掉，而且不在机器启动时运行。**dhcpcd6** 守护程序在系统启动后在后台运行。

dhcpcd6 守护程序读取它的配置文件并尝试启动并得到配置文件中指定的接口的一个或多个 IPv6 地址和其他配置选项。从服务器获取的地址如服务器所要求进行更新。

如果 DHCPv6 客户机不需要让 DHCPv6 服务器向其分配 IPv6 地址，则该客户机可以仅获取例如可用 DNS 服务器或 NTP 服务器列表等配置信息。这在使用静态地址配置节点时非常有用。

可以使用 **refresh** 命令来使 **dhcpcd6** 守护程序重新读取配置文件。SIGHUP 也可用于获取相同的响应。

缺省的 **dhcpcd6** 配置文件是 **/etc/dhcpv6/dhcpc6.cnf**。它包含日志和网络接口信息。

标志

-f *ConfigurationFileName*	指定要用的配置文件。缺省值是 **/etc/dhcpv6/dhcpc6.cnf**。
-p *ClientPort*	指定要使用的客户机端口。缺省值是 546。
-t *SolicitTimeout*	指定退出前直到客户机从服务器请求配置信息的时间。
-u *Client_duid_File*	指定要使用的客户机标识文件。缺省值是 **/etc/dhcpv6/dhcpc6.duid**。

退出状态

此命令返回以下出口值:

0	成功完成。
>0	发生错误。

安全性

访问控制: 您必须拥有 root 用户权限才能运行这个命令。

示例

1. 要使用位于 **/usr/local** 的配置文件 **dhcpcd6.cnf** 来启动 DHCPv6 客户机，请输入:

   ```
   startsrc -s dhcpcd6 -a "-f /usr/local/dhcpcd6.cnf"
   ```

位置

/usr/sbin/dhcpcd6

文件

/usr/sbin/dhcpcd6	包含 **dhcpv6** 客户机守护程序。
/etc/dhcpv6/dhcpc6.cnf	包含缺省配置文件。
/etc/dhcpv6/dhcpc6.db	包含客户机租约文件。该文件由客户机守护程序创建并且为不可配置。
/etc/dhcpv6/dhcpc6.duid	包含客户机标识文件。该文件由客户机守护程序创建并且为不可配置。

相关信息

第 102 页的『dhcpsdv6 守护程序』。

《*AIX 5L V5.3 命令参考大全，卷 5*》中的 **startsrc** 命令。

---

dhcprd 守护程序

用途

转发本地网络的 BOOTP 和 DHCP 分组。

语法

用系统资源控制器向 DHCP 服务器转发信息:

startsrc -s dhcprd [**-a** *Argument*] [**-a** *Argument*] ...

不用系统资源控制器向 DHCP 服务器转发信息:

dhcprd [**-f** *ConfigurationFile*]

描述

dhcprd 守护程序侦听、接收广播分组，并转发它们到适当的服务器。这将可以避免广播分组必须传到其他网络的要求。DHCP 中继代理负责将 DHCP 和 BOOTP 客户机广播分组从本地网络传到一组服务器。BOOTP 或者 DHCP 客户机发送的初始分组是客户机所在的本地网络接口上的广播。这些分组不允许通过网关和路由器。作为 BOOTP/DHCP 中继代理，**dhcprd** 守护程序会将这些分组传给适当的服务器。

DHCP 服务器读取 **/etc/services** 文件以决定使用哪个端口来接收请求。缺省服务为 **dhcps**。由于这是与 **bootpd** 守护程序所用的同样的端口，因此您只能运行一个（**dhcprd** 或 **bootpd**）守护程序。如果您选择 **dhcprd** 守护程序，您就需要从 **/etc/inetd.conf** 文件中取消注释 **bootp**，接着在命令行上输入 **refresh -s inetd**。

注: 如果运行了 **bootpd**，则该程序在启动守护程序前必须停止。

标志

-f *ConfigurationFile* 指定要用的配置文件。缺省值为 **/etc/dhcpcd.ini** 文件。

退出状态

此命令返回以下退出值:

0 成功结束。
>0 发生错误。

安全性

访问控制: 您必须拥有 root 用户权限才能运行这个命令。

文件

/usr/sbin/dhcprd	包含 **dhcprd** 守护程序。
/etc/dhcprd.cnf	包含缺省配置文件。
/etc/services	定义用于网络服务的套接字和协议。
/etc/inetd.conf	定义 **inetd** 守护程序控制的服务。

相关信息

dhcpsconf 命令、**startsrc** 命令、**stopsrc** 命令。

dhcpcd 守护程序、**dhcpsd** 守护程序、**inetd** 守护程序。

DHCP 客户机配置文件

DHCP 服务器配置文件

TCP/IP 地址和参数分配 – 动态主机配置协议

《*操作系统与设备管理*》中的『系统资源控制器』说明了子系统、子服务器以及系统资源控制器。

有关安装基于 Web 的系统管理器安装的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

《*网络与通信管理*》中的『TCP/IP 守护程序』。

---

dhcpsconf 命令

用途

通过图形用户界面简化 DHCP（动态主机配置协议）服务器配置。

语法

dhcpsconf

描述

dhcpsconf 命令提供一个 X-windows GUI（图形用户界面）使得网络管理员可以读取、保存和修改配置文件。它也允许您启动、停止和检索一台处于运行状态的服务器的统计信息。

dhcpsconf 命令显示一组列表。左边的列表显示可用的选项和按键。**dhcpsconf** 命令读取 **/etc/options.file** 文件以确定它的基本选项和按键，并利用它们以通用资源类型启动。GUI 使得网络管理员可以通过选择资源菜单按钮来定义一些指定资源。

资源定义对话框允许网络管理员制定网络上的所有选项和细节。网络管理员可以定义和命名网络、打印机、名称服务器、DHCP 服务器和其他合法资源对象。一旦此步骤完成，这些新资源被添加到显示在主面板上的键和选项。这些信息也可以被用于创建一个或者一系列服务器配置文件。

GUI 以一个空的控制文件启动。控制文件包含单个或多个服务器的定义，以及一个服务器可读的实际文件。控制文件只有一个 DHCP 服务器可读，但是可以存放多台服务器的信息。这使得网络管理员可以配置网络上的单个服务器映像，创建多个服务器处理相同的数据集合，而只在一个文件查看和维护。

通过选择选项或按键，选择它们在编辑窗口的位置，和选择与之对应的添加按钮，选项和按键就可以被添加到服务器窗口。选项被添加到编辑窗口的指定位置。如果它是一个指定的资源，就按此名添加。如果是标准缺省设置之一，就会弹出一个要求赋值的窗口。

DHCP 服务器就像其他按键一样可被添加，除非它们指定了在其作用域内负责对应项的网络上的机器。这些键具有作用域和语法顺序。注释不是真正的键，但是它们允许出现在任何地方。

一台服务器可以有一个网络、类、客户机或者内部指定的选项。一个网络可以有一个子网络、类、客户机、或者选项。一个子网络可以有一个类、客户机、或者选项。一个类和客户机只能有选项。

服务器拥有一系列专用配置参数。它们通过 DHCP 键列表中的服务器键指定，或者使用服务器菜单栏上的缺省服务器选项指定。缺省服务器选项适用于主文件。在主文件上指定的 DHCP 服务器接收缺省选项，但是可能要修改。

置于编辑窗口上的任何项可以更改、重命名、查看、或者删除。这将允许您放置项，看它是否正确，必要时可以更改。

完成配置文件后，单个控制文件可能被保存，和／或一组服务器文件被创建。文件菜单按钮和服务器菜单按钮都有保存选项。文件保存按钮用于保存主文件。服务器保存按钮用于将一个特殊的服务器存到一个文件中。

文件菜单按钮还包含一个 quit 选项、检索文件的 open 选项、清除目前为止创建的所有信息的 new 选项。

操作菜单按钮包含一个状态按钮、启动按钮、停止按钮、刷新按钮和发送配置文件按钮。利用这些按钮，一个远程服务器可以报告状态，利用新的配置文件刷新自身，也可被停止、发送配置文件和重启。

帮助按钮包含描述每个窗口项的一系列帮助文档。

退出状态

此命令返回以下退出值:

0 成功结束。
>0 发生错误。

安全性

访问控制: 任何用户

文件

/usr/sbin/dhcpsconf	包含 **dhcpsconf** 命令。
/etc/dhcpcd.cnf	包含缺省客户机配置文件

相关信息

dhcpcd 守护程序、**dhcprd** 守护程序、**dhcpsd** 守护程序、**inetd** 守护程序。

DHCP 客户机配置文件

DHCP 服务器配置文件

TCP/IP 地址和参数分配 － 动态主机配置协议

---

dhcpsd 守护程序

用途

实现一个动态主机配置协议（DHCP）的服务器。向 DHCP 客户机提供地址和配置信息服务。

语法

利用系统资源控制器向 DHCP 客户机提供信息服务:

startsrc -s dhcpsd [**-a** *Argument*] [**-a** *Argument*] ...

向 DHCP 客户机提供信息服务，但不利用系统资源控制器:

dhcpsd [**-f** *ConfigurationFile*]

描述

DHCP 服务器处理动态地址的分配和维护。它还处理附加配置信息的分发。**dhcpsd** 守护程序运行于后台并维护一个服务器信息数据库，包含日志参数、IP（因特网协议）地址范围、其他网络配置信息和可访问信息。初始数据库由配置文件指定。配置文件包含启动 DHCP 客户机配置所需的所有数据。

DHCP 服务器维护一个由它所分发的地址以及这些地址的所有者的数据库。这些数据库保存在 **/etc/dhcpsd.ar** 和 **/etc/dhcpsd.cr** 文件中。一台正在启动的服务器读取配置文件，设置可用地址的初始数据库。服务器接收 **refresh** 命令或者 SIGHUP 信号重新读取配置文件。

DHCP 服务器读取 **/etc/services** 命令以决定使用哪个端口来接收请求。缺省服务为 dhcp。由于这是与 **bootpd** 守护程序所用的相同的端口，因此您只能运行一个（**dhcpsd** 或 **bootpd**）守护程序。如果您选择 **dhcpsd** 守护程序，您需要对 **/etc/inetd.conf** 文件中的 **bootp** 进行注释，然后在命令行上输入 **refresh -s inetd**。

注: 如果运行 **bootpd**，这个程序必须在启动守护程序前终止。

标志

-f *ConfigurationFile* 指定要用的配置文件。

退出状态

此命令返回以下退出值:

0 成功结束。
>0 发生错误。

安全性

访问控制: 您必须拥有 root 用户权限才能运行这个命令。

文件

/usr/sbin/dhcpsd	包含 **dhcpsd** 守护程序。
/etc/services	定义用于网络服务的套接字和协议。
/etc/inetd.conf	定义 **inetd** 守护程序控制的服务。

相关信息

dhcpsconf 命令

startsrc 命令、**stopsrc** 命令。

dhcpcd 守护程序、**dhcprd** 守护程序、**inetd** 守护程序。

DHCP 客户机配置文件

DHCP 服务器配置文件

TCP/IP 地址和参数分配 - 动态主机配置协议

《*操作系统与设备管理*》中的『系统资源控制器』说明了子系统、子服务器以及系统资源控制器。

《*网络与通信管理*》中的『TCP/IP 守护程序』。

dhcpsdv6 守护程序

用途

实现动态主机配置协议（DHCPv6）服务器。向 DHCPv6 客户机提供地址和配置信息。

语法

要使用系统资源控制器向 DHCPv6 客户机提供信息:

startsrc -s dhcpsdv6 [**-a** *"Argument"*]

向 DHCP 客户机提供信息服务，但不利用系统资源控制器:

dhcpsdv6 [**-d**] [**-f** *ConfigurationFile*] [**-a** *DadminPort*] [**-p** *ServerPort*]

描述

DHCPv6 服务器处理动态地址分配的分配和维护。它还处理附加配置信息的分发。**dhcpsd** 守护程序运行于后台并维护一个服务器信息数据库，其中包含日志参数、IP（因特网协议）地址范围、其他网络配置信息和可访问性信息。初始数据库由配置文件指定。配置文件包含启动 DHCP 客户机配置所需的所有数据。

DHCPv6 服务器维护一个由它所分发的地址以及这些地址的所有者的数据库。这些数据库保存在文件 **/etc/dhcpv6/db_file.crbk** 和 **/etc/dhcpv6/db_file.cr** 中。一台正在启动的服务器读取配置文件，设置可用地址的初始数据库。服务器接受 refresh 命令或 SIGHUP 信号以重新读取配置文件。

标志

-a	指定 Dadmin 端口；在缺省情况下为 942。
-d	显示调试信息。
-f *ConfigurationFile*	指定要用的配置文件。在缺省情况下，配置文件是 **/etc/dhcpv6/dhcpsdv6.cnf**。
-p	指定服务器用于侦听入局请求的端口；缺省情况下为 547。

退出状态

此命令返回以下出口值:

0	成功完成。
>0	发生错误。

安全性

访问控制: 您必须拥有 root 用户权限才能运行这个命令。

示例

1. 要使用位于 **/usr/local** 的配置文件 **dhcpsdv6.cnf** 来启动 DHCPv6 服务器，请输入:

```
startsrc -s dhcpsdv6 -a "-f /usr/local/dhcpsdv6.cnf"
```

位置

/usr/sbin/dhcpsdv6

文件

/usr/sbin/dhcpsdv6	包含 **dhcpsdv6** 守护程序。
/etc/dhcpv6/db_file.cr	包含客户机记录。该文件由服务器守护程序创建并且为不可配置。
/etc/dhcpv6/db_file.crbk	包含客户机记录。该文件由服务器守护程序创建并且为不可配置。
/etc/dhcpv6/dhcpsdv6.duid	包含服务器标识文件。该文件由服务器守护程序创建并且为不可配置。

相关信息

第 96 页的『 dhcpcd6 守护程序 』。

《*AIX 5L V5.3 命令参考大全, 卷 5*》中的 **startsrc** 命令。

---

diag 命令

用途

执行硬件问题确定。

语法

diag [[**-a**] | [**-s** [**-c**]] [**-E** *days*] [**-e**] | [**-d** *Device* [**-c**] [**-v**] [**-e**] [**-A**]] | [**-B** [**-c**]] | [**-t** *taskname*] | [**-S** *testsuite*] | [**-c -d** *Device* **-L pending | complete**]

描述

diag 命令是运行广泛选择的任务和服务帮助的起点。大多数任务和服务帮助是特定于平台的。以下任务和服务帮助是可用的:

- 运行诊断
- 显示或者更改诊断运行时选项
- 显示服务提示
- 显示先前的诊断结果
- 显示硬件错误报告
- 显示软件产品数据
- 显示配置和资源列表
- 显示硬件的重要产品数据
- 显示资源属性
- 更改硬件的重要产品数据
- 格式化介质
- 认证介质
- 显示测试模式
- 局域网分析器
- 在资源列表中添加资源

- 从资源列表删除资源
- SCSI 总线分析器
- 下载微码
- 显示或更改引导列表
- 定期诊断
- 备份与恢复介质
- 磁盘维护
- 配置拨号和 LPF 键
- 添加或删除抽屉配置
- 创建定制配置软盘
- 更新基于磁盘的诊断
- 配置 ISA 适配器
- 操作系统 Shell 提示（只适用于联机服务方式）
- 显示或更改多处理器配置
  - 启用和禁用个别处理器
- 显示或更改 BUMP 配置
  - 用新的二进制映像更新闪存 EPROM
  - 显示或更改诊断方式
  - 显示或更改远程电话号码和调制解调器配置
- 显示或更改电子方式切换
- 处理补充介质（只适用于单机方式）
- 类属微码下载
- 运行错误日志分析
- 用于以太网的服务帮助
- 在 AIX 5.1 和较早版本（RSPC）中更新系统闪存
- 在 AIX 5.1 和较早版本（RSPC）中配置铃声指示电源启动
- 在 AIX 5.1 和较早版本（RSPC）中配置服务处理器
- 在 AIX 5.1 和较早版本（RSPC）中保存或恢复服务处理器配置
- 在 AIX 5.1 和较早版本（RSPC）中显示机器检查错误日志
- 7135 RAIDiant 阵列服务帮助
- SCSI 设备识别和除去
- SCSD 磁带机服务帮助
- Escon 位错误率服务帮助
- SSA 服务帮助
- PCI RAID 物理磁盘标识
- 配置铃声提示电源启动策略（CHRP）
- 配置监视策略（CHRP）
- 配置重新引导策略（CHRP）
- 配置远程维护策略（CHRP）
- 保存或恢复硬件管理策略（CHRP）

- 显示固件设备节点信息（CHRP）
- 提供扇区可用性
- 更新系统或服务处理器闪存（CHRP）
- 显示系统环境传感器（CHRP）
- 显示检查停止分析结果
- 分析适配器内部日志
- 日志维修操作
- 闪烁 SK-NET FDDI 固件
- 显示微码级别

可以使用基于 Web 的系统管理器中的设备应用程序更改设备的特征。还可以使用系统管理接口工具（SMIT）**smit diag** 快速路径运行此命令。

标志

注: 大多数用户无需使用任何标志，因为 **diag** 命令是一个菜单驱动程序。

-A 指定高级方式。也必须使用 **-d** 标志指定设备。

-a 通过询问丢失的资源是否已除去或关闭等等来处理硬件配置的任何更改。在 AIX 5.2 和更高版本中，丢失的资源（以 “M” 表示）和丢失的资源路径（以 “P” 表示）将被整合入诊断资源选择列表。

-B 指示诊断运行基本系统测试。错误日志分析也会在支持错误日志分析的基本系统中的区域上运行。

-c 指示机器将不受照管。不进行任何提问。结果写到标准输出。也必须用指定要测试的设备的可选标志（**d**、**B**、**s**）。

-d *Device* 指定其上运行诊断的设备。

-E *Days* 指定运行错误日志分析期间搜索错误日志所用的天数。此标志可与其他任何标志一起使用。

-e 如果错误日志分析在选定设备上受到支持，则执行错误日志分析。不执行测试。必须与 **-d** 标志一起使用，否则显示资源选择菜单。如果使用 **-v** 标志，则 **-v** 具有优先权而忽略 **-e** 标志。

-S *testsuite* 指示进行测试的特定设备 “测试组”：

1. 基本系统
2. I/O 设备
3. 异步设备
4. 图形设备
5. SCSI 设备
6. 存储设备
7. 通信设备
8. 多媒体设备

-L pending | complete 以 **-d** 和 **-c** 选项指定资源的日志维修操作。如果该部分已被替换，但还不知道是否将在系统中保留此部分，则使用 **pending**。如果该部分已被替换且已知此部分将在系统中保留，则使用 **complete**。

-s 在所有资源上运行诊断。

-t *taskname*　到要运行的特定任务的快速路径。当前的快速路径任务如下:

format　格式化介质任务

certify　认证介质任务

download
下载微码任务

disp_mcode
显示微码级别任务

chkspares
节约扇区可用性任务

identifyRemove
热插拔任务

注: 任务取决于平台和设备。某些任务在系统上可能不可用。

-v　在“系统验证方式”下运行诊断，不执行错误日志分析。缺省值为“问题确定”方式，它测试设备并运行错误日志分析。如果与 **-e** 标志一起使用，则 **-v** 标志具有优先权而忽略 **-e** 标志。必须与 **-d** 标志一起使用以指定运行诊断的设备。

安全性

访问控制: 只有 root 用户可以运行此命令。

特权控制: 系统组。

示例

要以不提问的方式在 scdisk0 设备上运行诊断，请输入:

```
diag -d scdisk0 -c
```

文件

/usr/sbin/diag　包含 **diag** 命令。

相关信息

diaggetrto 命令、**diagsetrto** 命令。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

---

diaggetrto 命令

用途

显示诊断运行时选项。

语法

diaggetrto [[**-a**] [**-d**] [**-l**] [**-m**] [**-n**] [**-p**] [**-s**]]

描述

diaggetrto 命令显示一个或多个诊断运行时选项的值。以下运行时选项可用 **diaggetrto** 命令显示:

显示诊断方式选择菜单
当此选项关闭时，诊断仅以“问题确定”方式运行。缺省值为打开。

包含高级诊断
当此选项打开时，在从“任务选择菜单”或命令行运行时，诊断以高级方式运行。其缺省值为关闭。

用于搜索错误日志的天数
此选项在诊断不再分析错误日志条目之前控制这些条目必须多旧。缺省值为 7。

显示进度指示器
当此选项打开时，支持进度指示器的诊断应用程序将显示它们。缺省值为打开。

诊断事件记录
当此选项打开时，诊断记录事件。缺省值为打开。

诊断事件日志文件大小
此选项控制诊断事件日志的最大大小。可允许大小是成百上千的千字节的增量。缺省值为 100K。

标志

-a	显示**包含高级诊断**的值。
-d	显示**诊断事件记录**的值。
-l	显示**诊断事件日志文件大小**的值。
-m	显示**显示诊断选择菜单**的值。
-n	显示**用于搜索错误日志的天数**的值。
-p	显示**显示进度指示器**的值。
-s	显示所有诊断运行时选项。

退出状态

0 命令成功结束。

>0 发生错误。

示例

1. 要显示诊断事件日志大小，请输入:
   ```
   /usr/lpp/diagnostics/bin/diaggetrto -l
   ```
2. 要检查进度指示器是否打开以及检查诊断事件日志是否打开，请输入:
   ```
   /usr/lpp/diagnostics/bin/diaggetrto -p -d
   ```
3. 要显示搜索错误日志的天数，请输入:
   ```
   /usr/lpp/diagnostics/bin/diaggetrto -n
   ```

文件

/usr/lpp/diagnostics/bin/diaggetrto	包含 **diagsetrto** 命令。

相关信息

diagsetrto 命令、**diag** 命令。

diagrpt 命令

用途
显示先前的诊断结果。

语法

diagrpt [[**-o**] | [**-s** *mmddyy*] | [**-a**] | [**-r**]]

描述
diagrpt 命令显示先前的诊断会话结果。有三种类型的结果可以查看:

- 诊断结果文件存储在 **/etc/lpp/diagnostic/data** 目录。
- 诊断事件日志信息。
- 诊断结果存储在 CHRP 系统上的 NVRAM。

标志

-o	显示存储在 **/etc/lpp/diagnostics/data** 目录的最近诊断结果文件。
-s *mmddyy*	显示指定日期之后记录的所有诊断结果文件。
-a	以长版本显示诊断事件日志。
-r	以短版本显示诊断事件日志。

示例
1. 要列举 1999 年 1 月 31 日之后所有以前的诊断结果文件，请输入:

   ```
   /usr/lpp/diagnostics/bin/diagrpt -s 013199
   ```

2. 以短版本查看诊断事件日志，请输入:

   ```
   /usr/lpp/diagnostics/bin/diagrpt -r
   ```

文件

/usr/lpp/diagnostics/bin/diagrpt	包含 **diagrpt** 命令。

相关信息
diag 命令。

diagsetrto 命令

用途
设置诊断运行时选项。

语法
diagsetrto [[**-a on** | **off**] [**-d on** | **off**] [**-l** *Size*] [**-m on** | **off**] [**-n** *Days*] [**-p on** | **off**]]

描述

diagsetrto 命令设置任何数目的诊断运行时选项的值。以下运行时选项可以用 **diagsetrto** 命令更改:

显示诊断方式选择菜单
当此选项关闭时，诊断仅以“问题确定”方式运行。缺省值为打开。

包含高级诊断
当此选项打开时，在从“任务选择菜单”或命令行运行时，诊断以高级方式运行。其缺省值为关闭。

用于搜索错误日志的天数
此选项在诊断不再分析错误日志条目之前控制这些条目必须多旧。缺省值为 7。

显示进度指示器
当此选项打开时，支持进度指示器的诊断应用程序将显示它们。缺省值为打开。

诊断事件记录
当此选项打开时，诊断记录事件。缺省值为打开。

诊断事件日志文件大小
此选项控制诊断事件日志的最大大小。可允许大小是成百上千的千字节的增量。缺省值为 100K。

标志

-a on | off	设置**包含高级诊断**的值。
-d on | off	设置**诊断事件记录**的值。
-l *Size*	设置**诊断事件日志文件大小**的值。
-m on | off	设置**显示诊断方式选择菜单**的值。
-n *Days*	设置**用于搜索错误日志的天数**的值。
-p on | off	设置**显示进度指示器**的值。

退出状态

0 命令成功结束。

>0 发生错误。

示例

1. 要将诊断事件日志大小设置为 500K，请输入:
   ```
   /usr/lpp/diagnostics/bin/diagsetrto -l 500
   ```
2. 要关闭进度指示器并关闭诊断事件记录，请输入:
   ```
   /usr/lpp/diagnostics/bin/diagsetrto -p off -d off
   ```
3. 要将搜索错误日志的天数设置为 50，请输入:
   ```
   /usr/lpp/diagnostics/bin/diagsetrto -n 50
   ```

文件

/usr/lpp/diagnostics/bin/diagsetrto	包含 **diagsetrto** 命令。

相关信息

diaggetrto 命令、**diag** 命令。

diction 命令

用途
突出显示不清楚或者冗长的句子。

语法
diction [**-ml**] [**-mm**] [**-f** *PatternFile*] [**-n**] *File ...*

描述
diction 命令在包含来自于模糊和冗长语句数据库的短语的英文文档中查找所有语句。每个短语用 [ ]（方括号）括起来。因为 **diction** 命令在查找文本前首先运行 **deroff** 命令，包括含有适当的格式信息的头文件也是输入的一部分。**explain** 命令为 **diction** 命令找到的语句提供交互式同义字。

使用非标准化的格式宏可能导致不正确的语句中断。特别情况下，**diction** 命令不识别 **-me** 标识。

标志

-f *PatternFile*	指定包含模糊句法示例的文件；该文件用于缺省文件的补充。
-ml	使得 **deroff** 命令跳过 **mm** 宏列表；如果一个文档包含许多语句片段列表，也可以使用它。
-mm	覆盖缺省 **ms** 宏数据包。
-n	当与 **-f** 标志一起使用时禁用缺省文件，仅使用 *PatternFile* 参数指定的文件。

文件

/usr/lib/dict.d	包含缺省模式。

相关信息
deroff 命令、**explain** 命令。

ms 宏数据包。

diff 命令

用途
比较文本文件。

语法

比较两个文件的内容
diff [**-c**| **-C** *Lines* | **-D** [*String*] | **-e** | **-f** | **-n**] [**-b**] [**-i**] [**-t**] [**-w**] *File 1 File2*

diff [**-h**] [**-b**] *File 1 File2*

排序字典的内容并比较不同的文件

diff [**-c** | **-C** *Lines* | **-e** | **-f** | **-n**] [**-b**] [**-i**] [**-l**] [**-r**] [**-s**] [**-S** *File*] [**-t**] [**-w**] *Directory1 Directory2*

diff [**-h**] [**-b**] *Directory1 Directory2*

描述

diff 命令比较文本文件。它能比较单个文件或者目录内容。

注: **diff** 命令仅当输入为文本文件时才有效。

如果指定了 *Directory1* 和 *Directory2* 参数，**diff** 命令比较两个目录下名字相同的文本文件。列出不同的二进制文件、公共子目录和只在一个目录出现的文件。

当 **diff** 命令运行于常规文件时，且当目录比较期间比较不同的文本文件时，**diff** 命令显示文件中哪些行必须更改以保持它们一致。如果 *File1* 和 *File2* 参数都不是目录，其中之一可能给定负号 "-"，以采用标准输入。如果 *File1* 参数是目录，则使用目录中与 *File2* 参数指定的文件名一致的那个文件。

典型的输出包含这些格式的行:

```
Lines Affected in File1   Action        Lines Affected in File2
Number1                   a             Number2[,Number3]
Number1[,Number2]         d             Number3
Number1[,Number2]         c             Number3[,Number4]
```

这些行类似于 **ed** 子命令将 *File1* 文件转换成 *File2* 文件。Action 字母之前的数字指 *File1*；后面的数字则指 *File2*。因此，通过将 **a** 替换成 **d**，从右往左读，您就能知道如何将 *File2* 转换成 *File1*。在 **ed** 命令下，相同的对（即 *Number1* = *Number2*）会简略为单个数字。

以下每一行，**diff** 命令显示以 **<:**（小于符号，冒号）开始的第一个文件中的所有受影响行，然后显示以 **>**（大于符号）开始的第二个文件中的所有受影响行。

退出值 0 表示没有不同，1 表示找到不同处，2 表示出错。

注: 如果指定了多于一个的 **-c**、**-C**、**-D**、**-e**、**-f** 或 **-n** 标志，命令行上的最后一个具有优先权。系统不发出错误消息。

标志

-b 将每行末尾当作单个新行标记处理（忽略新行字符前的空格字符）的任意数量空格字符与其他空格字符串（除了新行标记）同样地比较。

-C *Lines* 启动 **diff** 命令，但只比较 *Lines* 变量指定的行数。**-C** 标志稍微修改输出。输出以文件的相同部分和创建日期开始。每个更改以 12 个 *（星号）组成的行分隔。从 *File1* 中要删的行以 -（减号）标记，*File2* 中要添加的行以 +（加号）标记。从一个文件更改到另一个文件的行在两个文件中都以 !（惊叹号）标记。在每一个文件的指定上下文行中的更改会被整组一起输出。

-c 启动 **diff** 命令，比较三行上下文。**-c** 标志稍微修改输出。输出以涉及文件的标识和它们的创建日期开始。每个更改以 12 个 *（星号）组成的行分隔。*File1* 中要删的行以 -（减号）标记，要被添加到 *File2* 的行以 +（加号）标记。从一个文件更改到另一个文件的行在两个文件中都以 !（惊叹号）标记。在每一个文件的指定上下文行中的更改会被整组一起输出。

-D [*String*] 使得 **diff** 命令在标准输出上建立一个 *File1* 和 *File2* 的合并版本。包含了 C 预处理器控件以便没有定义 *String* 的结果编译等同于编译 *File1*，同时定义 *String* 产生 *File2*。

-e 以适合 **ed** 编辑器的格式进行输出，将 *File1* 转换成 *File2*。当使用这个标志时，以下 shell 程序可以帮助维护一个文件的多个版本。手头仅需要由 **diff** 命令生成的一个祖先文件（**$1**）和一系列版本的 **ed** 脚本（**$2**、**$3**...）。标准输出上的最近版本如下:

```
(shift; cat $*; echo '1,$p') | ed - $1
```

当使用 **-e** 标志比较目录时，输出上添加了额外的命令，因此，结果是一个 shell 脚本，将两个目录上的共有文本文件从 *Directory1* 上的状态转换到 *Directory2* 上的状态。
注: 由 **-e** 或 **-f** 标志产生的编辑脚本无法创建由单个组成的行。(周期)。

-f 以不适合 **ed** 编辑器的格式创建输出，按照在 **-e** 标志下产生的逆向顺序显示从 *File1* 到 *File2* 的转换的必要修改。

-h 如果要更改的部分比较短而且分隔清晰，则执行备用的比较可能会更快。**-h** 标志可用于任意长度的文件。**-c**、**-C**、**-D**、**-e**、**-f** 和 **-n** 标志无法与 **-h** 标志一起使用。当使用 **-h** 标志时，除了 **-b** 标志，其他标志一律忽略。

-i 忽略字母大小写。例如，小写 **a** 被认为同大写 **A** 一样。

-l 长输出格式。每个由文本文件比较 **diff** 命令获得的结果通过命令 **pr** 输送分页。在报告所有文本文件不同之处后，其他不同之处将被记忆和总结。

-n 产生类似于 **-e** 标志创建的输出，但是顺序相反，而且在每一插入或删除命令上进行更改计数。这是修订控件系统（RCS）所用的格式。

-r 使 **diff** 命令的应用程序递归到遇到的公共子目录。

-s 报告相同的文件，否则不提。

-S [*File*] 当比较目录时，忽略在 *File* 变量指定的文件之前整理名称的文件。**-S** 标志只用于 *Directory1* 和 *Directory2* 参数指定的目录。如果您将 **-r** 标志与 **-S** 标志一起使用，**-S** 标志在 *Directory1* 和 *Directory2* 子目录中不进行递归。

-t 在输出行扩展制表符。典型输出或者 **-c** 标志输出会添加字符到每一行首，这会影响初始行的缩进，使得输出列表难以解释。该标志则保留原始源的缩进。

-w 忽略所有空格和制表符，将所有其他空白字符串视为一致。例如，`if ( a == b )` 与 `if(a==b)` 相等。

退出状态

此命令返回以下退出值:

0 未找到不同处。
1 找到不同处。
>1 发生错误。

示例

1. 要比较两个文件，请输入:

   ```
   diff chap1.back chap1
   ```

   这显示文件 `chap1.bak` 和 `chap1` 的不同处。

2. 要比较两个文件，但是忽略空格字符个数的区别，请输入:

   ```
   diff  -w prog.c.bak prog.c
   ```

   如果两行仅仅空格和制表符数目不同，则 **diff -w** 目录认为它们相等。

3. 要创建一个包含 **ed** 命令可以用于从另一个文件重构文件的命令的文件，请输入:

   ```
   diff  -e chap2 chap2.old >new.to.old.ed
   ```

这将创建一个名为 new.to.old.ed 的文件，它包含 **ed** 子命令，将 chap2 转换回在 chap2.old 中找到的文本版本。在大多数情况下，new.to.old.ed 比 chap2.old 要小很多。您可以通过删除 chap2.old 以节省磁盘空间，您也可以在任何时候重建它，通过输入:

```
(cat new.to.old.ed ; echo '1,$p') | ed - chap2 >chap2.old
```

括号中的命令将 1,$p 添加到编辑命令的末尾，发送给 **ed** 编辑器。1,$p 使得 **ed** 命令在编辑完成后将文件写到标准输出。该修改的命令序列然后从管道传给 **ed** 命令（| ed），编辑器读取它作为标准输入。**-** 标志使 **ed** 命令不显示文件大小以及其他附加信息，因为它们易与 chap2.old 的文本混淆。

文件

/usr/bin/diff 包含 **diff** 命令。

相关信息

bdiff 命令、**cmp** 命令、**diff3** 命令、**ed** 命令和 **pr** 命令。

《*操作系统与设备管理*》中的『文件』介绍了文件以及处理文件的方法。

《*操作系统与设备管理*》中的『输入和输出重定向』描述了操作系统如何处理输入和输出。

---

diff3 命令

用途

比较三个文件。

语法

diff3 [**-e** | **-x** | **-E** | **-X** | **-3**] *File1 File2 File3*

描述

diff3 命令比较三个文件，并将不同文本的范围写到标准输出，以以下代码为标记:

==== 三个文件都不同。
====1 *File1* 不同。
====2 *File2* 不同。
====3 *File3* 不同。

需要用以将给定文件的指定范围转换以匹配另一文件的更改类型，在输出中由以下 2 种方式之一指明:

File:*Number1* **a**
File:*Number1*[,*Number2*]**c**

文本添加到 *File* 中的行号 *Number1* 后，其中 *File* 可以是 **1**、**2** 或 **3**。第 *Number1* 行与第 *Number2* 行之间的文本将被更改。如果 *Number1* 同 *Number2* 相等，范围减缩为第 *Number1* 行。

相应范围的内容在 **c** 指示之后。当两文件内容相同时，命令 **diff3** 不显示较小编号文件的内容，虽然它显示相同行的位置。

注: **-e** 标志所产生的编辑脚本无法创建包含 .（句点）的行。

标志

-3 建立只合并标志为 ====3 的修改的编辑脚本。

-E, **-X** 这些分别类似于 **-e** 和 **-x** 标志，但是对重叠修改（就是在正常列表中被标记为 ==== 的修改处）的处理不同。两文件的重叠行被编辑脚本插入，并由 <<<<<< 和 >>>>>> 行括起来。**-E** 选项用于修订控件系统（RCS）合并，使得合并文件中的重叠修改受到保留并引起注意。

-e 建立和 **ed** 命令一起使用的编辑脚本，将所有修改合并到 *File1*，即 *File2* 和 *File3* 之间的修改（正常情况下标记为 ==== 和 ====3 的修改处）。

-x 建立只合并标记为 ==== 的修改的编辑脚本。

示例

列举三个文件的不同处:

```
diff3 fruit.a fruit.b fruit.c
```

如果文件 fruit.a、fruit.b 和 fruit.c 包含以下数据:

fruit.a	**fruit.b**	**fruit.c**
banana	apple	grape
grape	banana	grapefruit
kiwi	grapefruit	kiwi
lemon	kiwi	lemon
mango	orange	mango
orange	peach	orange
peach	pear	peach
pare		

然后 **diff3** 命令的输出显示了这些文件的差异，如下所示。（右边的注释不会出现在输出中。）

```
====            3 个文件都不同。
1:1,2c          第一个文件 fruit.a 的 1、2 行
  banana
  grape
2:1,3c          fruit.b 的第 1 到 3 行
  apple
  banana
  grapefruit
3:1,2c          文件 fruit.c 的第 1、2 行
  grape
  grapefruit
====2           第 2 个文件 fruit.b 的不同处。
1:4,5c          文件 fruit.a 和 fruit.c 第 4、5 行相同。
2:4a            为使文件 fruit.b 看起来相同，在第 4 行后添加。
3:4,5c
  lemon
  mango
====            第一个文件 fruit.a 的不同处。
1:8c
  pare
2:7c            fruit.b 的第 7 行和 fruit.c 的第 8 行相同
  pear
3:7a
```

文件

/usr/bin/diff3 指示 **diff3** 命令。

/usr/lbin/diff3prog 被 **diff3** shell 脚本调用。

相关信息

diff 命令、**ed** 命令。

《*操作系统与设备管理*》中的『文件』介绍了文件以及处理文件的方法。

《*操作系统与设备管理*》中的『输入和输出重定向』描述了操作系统如何处理输入和输出。

---

diffmk 命令

用途

标记文件的不同处。

语法

diffmk [{ **-ab***X* | **-ae***X* } [**-b**] [**-cb***X* | **-ce***X*] [**-db***X* | **-de***X*] *File1 File2* [*File3*]

描述

diffmk 命令比较 *File1* 参数指定的英文文件和 *File2* 参数指定的文件。然后它创建包含 **.mc** 请求的（建立更改标记）**nroff** 和 **troff** 命令的第三方文件。*File1* 和 *File2* 参数分别指定文件的新旧版本。**diffmk** 命令将新建文件写到 *File3* 参数（如果指定的话），或写到标准输出。*File3* 文件包含加上已插入格式化程序 **.mc** 请求的 *File2* 文件的行。当 *File3* 文件被格式化时，修改的文本或插入文本用竖线“|”在每行的右边空白处标记。空白处的星号“*”表示该行已被删。

如果定义了 **DIFFMARK** 环境变量，它指定 **diffmk** 命令用于比较文件的命令字符串。（正常情况下，命令 **diffmk** 使用命令 **diff**）。例如，为更好处理特大文件，您可以为 `diff -h` 设定变量 **DIFFMARK**。

参数

File1 指定与 *File2* 参数指定的文件比较的英文文件。比较结果包含由 *File3* 参数指定的文件。*File1* 被当作“旧”文件。

File2 指定一个英文文件，与 *File1* 参数指定的文件比较。比较结果包含由 *File3* 参数指定的文件。*File2* 被当作“新”文件。

File3 指定一个文件，包含 *File2* 文件中的行，以及插入格式化程序 **.mc** 请求，该请求是对于 **nroff** 和 **troff**。该文件的内容是参数 *File1* 和 *File2* 指定的两个文件的比较结果。格式化时，不同处用竖线“|”在每行的右边空白处标记。星号“*”表示该行已被删除。如果未指定 *File3*，则比较结果写到标准输入。

标记

-ab*X* 用 *X* 标记从哪行开始添加。
-ae*X* 用 *X* 标记在哪行结束。
-b 忽略只有制表符和空格符区别的行。
-cb*X* 用 *X* 标记不同的行从哪里开始。
-ce*X* 用 *X* 标记不同行在哪里结束。
-db*X* 用 *X* 标记删除行从哪里开始。
-de*X* 用 *X* 标记删除行在哪里结束。

示例

1. 要标记一个文本文件的两个版本的不同，请输入:

   ```
   diffmk chap1.old chap1 chap1.nroff
   ```

   这将创建一个 chap1 副本，包含 **nroff** 和 **troff** 更改标记请求，以识别添加的、更改的和从 chap1.old. 删除的文本。副本存放在 chap1.nroff 文件中。

2. 要用非 **nroff** 和 **troff** 消息标记不同，请输入:

   ```
   diffmk -ab'>>New:' -ae'<<End New' \
   chap1.old chap1 chap1.nroff
   ```

   这将使得 **diffmk** 命令在新添加部分的前一行写 >>New: 到 chap1 中，并在添加部分之后写 <<End New。不同的部分和删除部分仍然建立命令 **nroff** 和 **troff**，以在空白处添加竖线 “|” 或者星号 “*”。

3. 要使用不同的 **nroff** 和 **troff** 命令标记请求，忽略空格字符区别，请输入:

   ```
   diffmk -b -cb'.mc %' chap1.old chap1 chap1.nroff
   ```

   这将嵌入如下命令，用 % （百分号）标记更改部分，用竖线 “|” 标记添加部分，用星号 “*” 标记删除部分。它不标记仅仅是单词间的制表符或空格符数目不同的差异（-b）。

相关信息

diff 命令、**nroff** 命令、**troff** 命令。

---

dig 命令

用途

DNS 查询实用程序。

语法

dig [*@server*] [**-b** *address*] [**-c** *class*] [**-f** *filename*] [**-k** *filename*] [**-n**][**-p** *port#*] [**-t** *type*] [**-x** *addr*] [**-y** *name:key*] [*name*] [*type*] [*class*] [*queryopt...*]

dig [**-h**]

dig [*global-queryopt...*] [*query...*]

描述

dig（域信息搜索器）命令是一个用于询问 DNS 域名服务器的灵活的工具。它执行 DNS 搜索，显示从受请求的域名服务器返回的答复。多数 DNS 管理员利用 **dig** 作为 DNS 问题的故障诊断，因为它灵活性好、易用、输出清晰。虽然通常情况下 **dig** 使用命令行参数，但它也可以按批处理方式从文件读取搜索请求。不同于早期版本，**dig** 的 BIND9 实现允许从命令行发出多个查询。除非被告知请求特定域名服务器，**dig** 将尝试 **/etc/resolv.conf** 中列举的所有服务器。当未指定任何命令行参数或选项时，**dig** 将对 “**.**” （根）执行 NS 查询。

标志

-b *address*　　设置所要询问地址的源 IP 地址。这必须是主机网络接口上的某一合法的地址。

-c *class* 默认查询类（IN for internet）由选项 **-c** 重设。***class*** 可以是任何合法类，例如查询 Hesiod 记录的 HS 类或查询 CHAOSNET 记录的 CH 类。

-f *filename* 使 **dig** 在批处理方式下运行，通过从文件 ***filename*** 读取一系列搜索请求加以处理。文件包含许多查询；每行一个。文件中的每一项都应该以和使用命令行接口对 **dig** 的查询相同的方法来组织。

-h 当使用选项 **-h** 时，显示一个简短的命令行参数和选项摘要。

-k *filename* 要签署由 **dig** 发送的 DNS 查询以及对它们使用事务签名（TSIG）的响应，用选项 **-k** 指定 TSIG 密钥文件。

-n 缺省情况下，使用 IP6.ARPA 域和 RFC2874 定义的二进制标号搜索 IPv6 地址。为了使用更早的、使用 IP6.INT 域和 **nibble** 标签的 RFC1886 方法，指定选项 **-n**（nibble）。

-p *port#* 如果需要查询一个非标准的端口号，则使用选项 **-p**。port# 是 dig 将发送其查询的端口号，而不是标准的 DNS 端口号 53。该选项可用于测试已在非标准端口号上配置成侦听查询的域名服务器。

-t *type* 设置查询类型为 ***type***。可以是 BIND9 支持的任意有效查询类型。缺省查询类型是 **A**，除非提供 **-x** 选项来指示一个逆向查询。通过指定 AXFR 的 ***type*** 可以请求一个区域传输。当需要增量区域传输（IXFR）时，***type*** 设置为 **ixfr=N**。增量区域传输将包含自从区域的 SOA 记录中的序列号改为 **N** 之后对区域所做的更改。

-x *addr* 逆向查询（将地址映射到名称）可以通过 **-x** 选项加以简化。***addr*** 是一个以小数点为界的 IPv4 地址或冒号为界的 IPv6 地址。当使用这个选项时，无需提供 *name*、*class* 和 *type* 参数。**dig** 自动运行类似 `11.12.13.10.in-addr.arpa` 的域名查询，并分别设置查询类型和类为 PTR 和 IN。

-y *name:key* 您可以通过命令行上的 **-y** 选项指定 TSIG 密钥；***name*** 是 TSIG 密码的名称，***key*** 是实际的密码。密码是 64 位加密字符串，通常由 **dnssec-keygen**（8）生成。当在多用户系统上使用选项 **-y** 时应该谨慎，因为密码在 ps（1）的输出或 shell 的历史文件中可能是可见的。当同时使用 **dig** 和 TSCG 认证时，被查询的名称服务器需要知道密码和解码规则。在 BIND 中，通过提供正确的密码和 **named.conf** 中的服务器声明实现。

参数

global-queryopt... 全局查询选项（请参阅多个查询）。

query 查询选项（请参阅查询选项）。

查询选项

dig 提供查询选项号，它影响搜索方式和结果显示。一些在查询请求报头设置或复位标志位，一部分决定显示哪些回复信息，其他的确定超时和重试战略。每个查询选项被带前缀（+）的关键字标识。一些关键字设置或复位一个选项。通常前缀是求反关键字含义的字符串 no。其他关键字分配各选项的值，例如超时时间间隔。它们的格式形如 **+keyword=value**。查询选项是:

+[no]tcp

查询域名服务器时使用 [不使用] TCP。缺省行为是使用 UDP，除非是 AXFR 或 IXFR 请求，才使用 TCP 连接。

+[no]vc

查询名称服务器时使用 [不使用] TCP。**+[no]tcp** 的备用语法提供了向下兼容。**vc** 代表虚电路。

+[no]ignore

忽略 UDP 响应的中断，而不是用 TCP 重试。缺省情况运行 TCP 重试。

+domain=somename

设定包含单个域 ***somename*** 的搜索列表，好像被 **/etc/resolv.conf** 中的域伪指令指定，并且启用搜索列表处理，好像给定了 **+search** 选项。

+[no]search

使用 [不使用] 搜索列表或 **resolv.conf** 中的域伪指令（如果存在）定义的搜索列表。缺省情况不使用搜索列表。

+[no]defname

不建议看作 **+[no]search** 的同义词。

+[no]aaonly

该选项不做任何事。它用来提供对设置成未实现解析器标志的 **dig** 的旧版本的兼容性。

+[no]adflag

在查询中设置 [不设置] AD（真实数据）位。目前 AD 位只在响应中有标准含义，而查询中没有，但是出于完整性考虑在查询中这种性能可以设置。

+[no]cdflag

在查询中设置 [不设置] CD（检查禁用）位。它请求服务器不运行响应信息的 DNSSEC 合法性。

+[no]recursive

切换查询中的 RD（要求递归）位设置。在缺省情况下设置该位，也就是说 dig 正常情形下发送递归查询。当使用查询选项 **+nssearch** 或 **+trace** 时，递归自动禁用。

+[no]nssearch

这个选项被设置时，**dig** 试图寻找包含待搜名称的网段的权威域名服务器，并显示网段中每台域名服务器的 SOA 记录。

+[no]trace

切换为待查询名称从根名称服务器开始的代理路径跟踪。缺省情况不使用跟踪。一旦启用跟踪，**dig** 使用迭代查询解析待查询名称。它将按照从根服务器的参照，显示来自每台使用解析查询的服务器的应答。

+[no]cmd

设定在输出中显示指出 **dig** 版本及其所用的查询选项的初始注释。缺省情况下显示注释。

+[no]short

提供简要答复。缺省值是以冗长格式显示答复信息。

+[no]identify

当启用 **+short** 选项时，显示 [或不显示] 提供应答的 IP 地址和端口号。如果请求简短格式应答，缺省情况不显示提供应答的服务器的源地址和端口号。

+[no]comments

切换输出中的注释行显示。缺省值是显示注释。

+[no]stats

该查询选项设定显示统计信息：查询进行时，应答的大小等。缺省显示查询统计信息。

+[no]qr

显示 [不显示] 发送的查询请求。缺省不显示。

+[no]question

当返回应答时，显示 [不显示] 查询请求的问题部分。缺省作为注释显示问题部分。

+[no]answer

显示 [不显示] 应答的回答部分。缺省显示。

+[no]authority

显示 [不显示] 应答的权限部分。缺省显示。

+[no]additional

显示 [不显示] 应答的附加部分。缺省显示。

+[no]all

设置或清除所有显示标志。

+time=T

为查询设置超时时间为 ***T*** 秒。缺省值为 5 秒。如果将 ***T*** 设置为小于 1 的数，则以 1 秒作为查询超时时间。

+tries=A

设置向服务器发送 UDP 查询请求的重试次数为 ***A***，代替缺省的 3 次。如果将 ***A*** 小于或等于 0，则采用 1 为重试次数。

+ndots=D

出于完全考虑，设置必须出现在名称 ***D*** 的点数。缺省值是使用在 **/etc/resolv.conf** 中的 **ndots** 语句定义的，或者是 1，如果没有 **ndots** 语句的话。带更少点数的名称被解释为相对名称，并通过搜索列表中的域或文件 **/etc/resolv.conf** 中的域伪指令进行搜索。

+bufsize=B

设置使用 EDNS0 的 UDP 消息缓冲区大小为 ***B*** 字节。缓冲区的最大值和最小值分别为 65535 和 0。超出这个范围的值自动舍入到最近的有效值。

+[no]multiline

以详细的多行格式显示类似 SOA 的记录，并附带可读注释。缺省值是每单个行上显示一条记录，以便于计算机解析 **dig** 的输出。

多条查询

dig 的 BIND9 支持在命令行上指定多个查询（支持 **-f** 批处理文件选项的附加功能）。每条查询可以使用自己的标志位、选项和查询选项。

在这种情况下，在上面描述的命令行语法中，每条查询自变量代表一个个别查询。每一条由任意标准选项和标志、待查询名称、可选查询类型和类以及任何适用于该查询的查询选项。

也可以使用对所有查询均有效的查询选项全局集合。全局查询选项必须位于命令行上第一个名称、类、类型、选项、标志和查询选项的元组之前。任何全局查询选项（除了 **+[no]cmd** 选项）可以被下面的查询特别选项重设。例如:

```
dig +qr www.isc.org any -x 127.0.0.1 isc.org ns +noqr
```

显示 **dig** 如何从命令行出发进行三个查询: 一个针对 **www.isc.org** 的任意查询、一个 127.0.0.1 的逆向查询，以及一个 **isc.org** 的 NS 记录查询。应用了 **+qr** 的全局查询选项，以便 **dig** 显示进行每条查询的初始查询。最后那个查询有一个本地查询选项 **+noqr**，表示 **dig** 在搜索 **isc.org** 的 NS 记录时不显示初始查询。

示例

一个典型的 **dig** 调用类似:

```
dig @server name type
```

其中:

server 待查询名称服务器的名称或 IP 地址。可以是用点分隔的 IPv4 地址或用冒号分隔的 IPv6 地址。当由主机提供服务器参数时，**dig** 在查询域名服务器前先解析那个名称。如果没有服务器参数可以提供，**dig** 参考 **/etc/resolv.conf**，然后查询列举在那里的域名服务器。显示来自域名服务器的应答。

name 将要查询的资源记录的名称。

type 显示所需的查询类型 – **ANY**、**A**、**MX**、**SIG**，以及任何有效查询类型等。如果不提供任何类型参数，**dig** 将对纪录 ***A*** 执行查询。

文件

/etc/resolv.conf

相关信息

host 和 dnssec-keygen 命令。

named8 守护程序。

RFC1035

---

digest 命令

用途

将 ASCII 码从 **/etc/qconfig** 文件转换成 **/etc/qconfig.bin** 文件，由 **qdaemon** 命令使用的队列配置二进制版本。该命令不在命令行输入；由 **qdaemon** 命令调用。

语法

/usr/lib/lpd/digest *ASCIIFile BinaryFile*

描述

digest 命令接收一个 ASCII 字符输入文件，将其转换成二进制文件。该命令只被 **qdaemon** 命令用作将 **/etc/qconfig** 文件译成该文件的二进制版本，**/etc/qconfig.bin**。

文件

/etc/qconfig	包含队列配置文件。
/usr/sbin/qdaemon	包含队列守护程序。
/etc/qconfig.bin	包含已转换过的 **/etc/qconfig** 文件的二进制版本。

相关信息

qdaemon 命令。

---

dircmp 命令

用途

比较二个目录和其公共文件的内容。

语法

dircmp [**-d**] [**-s**] [**-w** *num*] *Directory1 Directory2*

描述

dircmp 命令比较 *Directory1* 和 *Directory2* 参数指定的两个目录，并将其内容相关信息写到标准输出。首先，**dircmp** 命令比较每个目录中的文件名。如果相同的文件名同时出现在两个目录，**dircmp** 命令比较两文件的内容。

在输出中，**dircmp** 命令首先列举各目录独有的文件。然后列举文件名相同但是内容不同的文件。如果不指定任何标志，命令还列举文件名和内容都相同的文件。

diff -r 命令提供类似于 **dircmp** 命令所拥有的功能。

标记

-d 显示公共文件中的不同内容。显示格式与 **diff** 命令相同。
-s 不列出相同文件的名称。
- w 更改输出宽度为 *num* 个字符数。
num

退出状态

此命令返回以下退出值:

0 成功结束。
>0 发生错误。

注: 目录内容中的不同不会当作错误。

示例

1. 要总结两目录中的文件的不同处，请输入:

   ```
   dircmp proj.ver1 proj.ver2
   ```

   这将显示目录 proj.ver1 和 proj.ver2 的不同处摘要。分别列举只在一个目录出现的文件，以及在两目录都出现的文件。如果一文件同时出现在两目录，命令 **dircmp** 则分析两文件内容是否相同。
2. 要显示文件不同处的细节，请输入:

   ```
   dircmp  -d  -s proj.ver1 proj.ver2
   ```

   -s 标志禁止有关相同文件的信息。**-d** 标志显示在两个目录中找到的每个不同文件的 **diff** 列表。
3. 要以输出行宽度设置为 90 个字符来显示文件之间的不同处细节，请输入:

   ```
   $dircmp -w 90 dir1 dir2
   ```

文件

/usr/bin/dircmp 包含 **dircmp** 命令。

相关信息

cmp 命令、**diff** 命令。

《*操作系统与设备管理*》中的『目录』 描述了文件系统中目录的结构和特征。

《*操作系统与设备管理*》中的『输入和输出重定向』描述了操作系统如何处理输入和输出。

dirname 命令

用途

将指定路径除了最后以外的部分写到标准输出。

语法

dirname *Path*

描述

dirname 命令读取指定路径名保留最后一个“/”（斜杠）及其后面的字符，删除其他部分，并写结果到标准输出。如果最后一个“/”后无字符，**dirname** 命令使用倒数第二个“/”，并忽略其后的所有字符。**dirname** 命令在创建路径名的时候遵从以下规则:

1. 如果 *Path* 参数为“//”（双斜杠），或者参数 *Path* 全部由斜杠组成，将其转换为单斜杠“/”。跳过步骤 2 到 7。
2. 从指定路径删除尾部的“/”字符。
3. 如果参数 *Path* 中没有剩下的“/”，则将路径转换成 **.**（点）。跳过步骤 4 到 7。
4. 从路径中删除尾部的所有非斜杠字符。
5. 如果剩下的路径为“//”（双斜杠），跳到步骤 6。
6. 删除路径尾部的斜杠字符。
7. 如果剩下的路径为空，则转换成单斜杠“/”。

例如，输入:

```
dirname //
```

结果为 **/**（斜杠）。输入:

```
dirname /a/b/
```

结果为: **/a**。输入:

```
dirname a
```

结果为 **.**（点）。输入:

```
dirname a/b
```

结果为路径名 **a**。

命令 **dirname** 和 **basename** 通常在 shell 内部命令替换使用，以指定一个与指定输入文件名略有差异的输出文件名。

退出状态

此命令返回以下退出值:

0 成功完成
>0 发生错误。

示例

要构造和另一文件位于相同目录的一个文件名，请输入:

```
AOUTFILE=`dirname $TEXTFILE`/a.out
```

这将设置 shell 变量 AOUTFILE 作为文件 **a.out** 的名称，它与 TEXTFILE 位于同一目录。如果 TEXTFILE 是 **/home/fran/prog.c**，则 dirname $TEXTFILE 的值成了 **/home/fran**，且 AOUTFILE 变成了 **/home/fran/a.out**。

文件

/usr/bin/dirname	包含令 **dirname** 命令。

相关信息

basename 命令、**sh** 命令。

---

disable 命令

disable 命令包含 AIX 打印子系统 **disable** 和 System V 打印子系统 **disable** 的信息。

用途

禁用打印机队列设备。

语法

disable [**-c**] [**-r***Reason*] *PrinterName* ...

描述

disable 命令禁用或脱机由 *PrinterName* 参数指定的打印机队列设备。

注: 您只有拥有 root 用户权限或属于 printq 组，才可使用该命令。

标志

-c	取消所有任务请求。使用该标志与输入 **enq -K** 命令相同。
-r*Reason*	用 *Reason* 变量指定禁用打印机队列设备的理由。该标志是一个『无操作』标志，意思为系统忽略该标志。

示例

1. 要不等到当前打印任务结束就使打印机队列 lp0 脱机，请输入:

   ```
   disable  -c lp0
   ```

2. 要等到所有打印任务结束才使打印机队列 lp0 脱机，请输入:

   ```
   disable lp0
   ```

文件

/usr/sbin/qdaemon	队列守护程序
/etc/qconfig	队列配置文件

/etc/qconfig.bin	摘要，**/etc/qconfig** 文件的二进制版本
/var/spool/lpd/qdir/*	队列请求
/var/spool/lpd/stat/*	设备状态信息
/var/spool/qdaemon/*	队列文件的临时副本

相关信息

cancel 命令、**enable** 命令、**enq** 命令、**lp** 命令、**lpstat** 命令。

《*打印机和打印指南*》中的『启动和停止打印队列』。

System V 打印子系统 disable 命令

用途

禁用 LP 打印机

语法

disable [*flags*] *printers*

描述

disable 命令禁用指定的 *printers*，使 **lp** 提交的打印请求停止。缺省情况下，位于指定打印机的当前的打印请求，将在本打印机或另一台同类打印机全部重新执行。如果是远程打印机，该命令只终止到远程系统的任务传送。**disable** 命令必须在远程系统上运行以禁用打印机。（运行 **lpstat -p** 获取打印机的状态。）

打印机名是*系统定义*字并且限定成 ASCII 字符的大小写。

标志

-c 取消指定打印机上的当前打印请求。该标志不能与 **-W** 标志同时使用。如果是远程打印机，则忽略 **-c** 标志。

-r *reason*

给禁用的打印机分配 *reason*。该 *reason* 应用于所有指定 *printers*。该 *reason* 由 **lpstat -p** 报告。*reason* 如果包含空格，必须用引号括住。缺省的 reason 值取已存在打印机的 *unknown reason*，以及刚添加到系统却还未启用的打印机的 *new printer*。

-W 一直等到当前打印任务结束才禁用指定打印机。

该标志不能和 **-c** 标志同时使用。如果是远程打印机，忽略 **-W** 标志。

文件

/var/spool/lp/*

参考

lp 命令，**lpstat** 命令。

diskusg 命令

用途

根据用户标识生成磁盘记帐数据。

语法

diskusg [**-X**] [**-U** *MaxUsers*] [**-i** *FileListName*] [**-p** *File*] [**-u** *File*] [**-v**] { **-s** [*File ...*] | *FileSystem ...* }

描述

diskusg 命令从以 *File* 或 *FileSystem* 参数指定的文件中的数据或从标准输入生成中间磁盘记帐信息。**diskusg** 命令按用户将记录写到标准输出。该命令由 **dodisk** 命令调用，它可以在 **cron** 守护程序下运行。输出按照以下的格式:

UID	包含数字形式的用户标识。
Login	包含用户的登录名
Blocks	包含分配给用户的 512 字节磁盘块总数。

此命令的输出作为 **acctdisk** 命令的输入，它将信息转换成一个完整的记帐记录。这个完整记帐记录和其他的完整记帐记录合并以生成日报。

如果您指定 *FileSystem* 参数，则 **diskusg** 命令读取指定文件系统的索引节点以生成使用情况数据。*FileSystem* 参数必须是文件系统设备的特别文件名称。例如，用 **/dev/hd4** 设备代替 **/**（根目录）为根文件系统构建使用情况统计数据。

如果您指定了 *File* 参数，输入必须是 **diskusg** 输出格式。

若需要更多的磁盘使用情况信息，请参阅 **acctdusg** 命令。

注: 此命令只用于本地设备。

标志

-i *FileListName*	忽略 *FileListName* 文件系统中的数据。*FileListName* 变量指定用引号括起或逗号分隔的文件系统名列表。
-p *File*	使用由 *File* 变量指定的密码文件生成登录名。缺省值为 **/etc/passwd** 文件。
-s [*File*]	合并来自输入文件或标准输入的所有记录为一个记录。输入数据已经是 **diskusg** 的输出格式了。
-U *MaxUsers*	设定 **diskusg** 命令可以处理的最大用户数。只有在用户数多于缺省值 5000 时，您才需要使用该标志。
-u *File*	对每一不属于任何用户的文件，写一条记录到指定的 *File* 变量。每一记录由特别文件名称、索引节点数和用户标识组成。
-v	将不属于任何用户的文件列表写到标准错误输出。
-X	打印并处理每个用户名的全部可用字符，而不是截断为前 8 个字符。

安全性

访问控制: 该命令的执行权限只授予用户组 **adm** 中的成员。

示例

要生成每天的磁盘记帐信息，添加一行类似下面的数据到 **/var/spool/cron/crontab/root** 文件:

```
0 2 * * 4 /usr/sbin/acct/dodisk
```

该命令让 **cron** 守护程序在每个星期四（4）的凌晨 2 点（02）运行 **dodisk** 命令。**dodisk** 命令调用 **diskusg** 和 **acctdisk** 命令两者。

注: 您必须拥有 root 用户权限，才可运行这个命令。

文件

/usr/sbin/acct/diskusg	包含 **diskusg** 命令。
/etc/passwd	包含基本用户属性。

相关信息

acctdisk 命令、**acctmerg** 命令、**dodisk** 命令、**runacct** 命令。

acct 子例程。

acct 文件格式和 **utmp** 文件格式。

《*操作系统与设备管理*》中的『记帐命令』、『系统记帐』以及『设置记帐系统』。

---

dispgid 命令

用途

显示所有合法用户组名列表。

语法

dispgid

描述

dispgid 命令可以显示系统上的所有用户组名列表（每行一个）。此命令无参数。以下文件以只读方式被访问以检索数据:

- **/etc/passwd**
- **/etc/group**
- **/etc/security/user**
- **/etc/security/limits**
- **/etc/security/group**
- **/etc/security/environ**

退出状态

0 命令成功结束。

>0 发生错误。

示例

1. 要列举机器上所有的合法用户组，输入命令 **dispgid** 如下:

   ```
   dispgid
   ```

   输出类似以下格式:

   ```
   system
   staff
   bin
   bin
   sys
   adm
   mail
   security
   cron
   printq
   audit
   ecs
   guest
   usr
   perf
   ```

文件

/usr/sbin/dispgid	包含命令 **dispgid**
/etc/group	包含组信息

相关信息

dispuid 命令、**lsgroup** 命令。

---

dispuid 命令

用途

显示所有合法用户名列表。

语法

dispuid

描述

该命令显示系统上的所有用户名列表（每行一个）。此命令无参数。以下文件以只读方式被访问以检索用户数据:

- **/etc/passwd**
- **/etc/security/user**
- **/etc/security/user.roles**
- **/etc/security/limits**
- **/etc/security/environ**
- **/etc/group**
- **/etc/group**

退出状态

0 命令成功结束

>0 发生错误。

示例

1. 要列举机器上的所有合法用户列表，输入命令 dispuid 如下:

```
dispuid
```

输出类似以下格式:

```
root
daemon
bin
bin
sys
adm
uucp
guest
nobody
lpd
invscout
imnadm
user1
```

文件

/usr/sbin/dispuid	包含 dispuid 命令。
/etc/passwd	包含密码信息。

相关信息

dispgid 命令和 **lsuser** 命令。

---

dist 命令

用途

重新分发消息给附加地址。

语法

dist [**+** Folder] [**-nodraftfolder** | **-draftfolder** **+***Folder*] [*Message* | **-draftmessage** *Message*] [**-annotate** [**-inplace** | **-noinplace**] | **-noannotate**] [**-form** *FormFile*] [**-editor** *Editor* | **-noedit**] [**-nowhatnowproc** | **-whatnowproc** *Program*]

描述

dist 命令提供一个接口，重新分发已有消息给新的地址列表。在缺省情况下，**dist** 命令复制当前文件夹的当前消息给 *UserMHDirectory***/draft** 文件，并启动一个编辑器。可以用 *Message* 参数指定一个不同于缺省设置的当前文件夹的消息。

一旦启动，编辑器提示您为每个头字段赋值。**dist** 命令使用 *UserMHDirectory*/**distcomps** 文件中定义的报头格式。（如果该文件不存在，则系统使用 **/etc/mh/distcomps** 文件。）因为消息主体是您要重新分发的消息，所以请勿填写消息体。如果要定义不同于 *UserMHDirectory*/**distcomps** 文件的格式文件，可以使用 **-form** 标志。

若要更换缺省编辑器，可以使用 **-editor** 标志或在 **$HOME/.mh_profile** 文件中定义 `Editor:` 项。

按下 Ctrl-D 按键顺序可以退出编辑器。一旦退出编辑器，**dist** 命令会启动消息处理程序（MH）`What Now?` 提示。按下 Enter 键可以看见可用的 **whatnow** 子命令列表。这些子命令使您能够继续编辑报文头、列举报文头、指导报文的部署，或结束 **dist** 命令的处理进程。

注: 在报头和报文体之间的左边必须有一行破折号或空格，以便于报文发送时的确认。

重新分发的报文需要在初始报文前添加一个新的报头。您用 **dist** 命令编辑的 **draft** 文件仅由报头字段构成。带新草稿的原始报文副本不会自动存储。

若要用重新分发信息注释原始报文，可使用 **-annotate** 标志。该标志在原始报文上追加 `Resent:` 字段以及当前的日期和时间。

标志

-annotate
重新分发的报文的注释行如下:

```
重新发送:    日期
重新发送:    地址
```

因为在命令多重执行的情况下，不会保留 **-annotate** 标志，所以只有从 **dist** 命令直接发送的报文才有完整的注释。**-inplace** 标志在指定位置强制注释，以保存到注释报文的链接。

-draftfolder *+Folder*
将草稿消息放在指定的文件夹中。如果 **-draftfolder** *+Folder* 标志后紧跟 *Message* 变量，其作用等同于使用 **-draftmessage** 标志。如果未指定 *+Folder*，草稿报文放置于 *Current-Folder* 中。

-draftmessage *Message*
指定报文草稿。在缺省情况下，系统在当前目录创建一个新的报文草稿。草稿报文变成了当前报文。

-editor *Editor*
指定用于准备报文重分发的初始编辑器。

+*Folder*
标识包含要重新分发报文的文件夹。如果未指定文件夹，则假定为 *Current-Folder*。

-form *FormFile*
决定报文格式。**dist** 命令处理指定格式文件的每一行。

-help
列出命令语法、可用的转换（切换）和版本信息。

注: 对于 MH，这个标志的名称必须拼写完全。

-inplace
为了保留到已注释报文的链接，采取强制注释。

Message
标识要重新分发的报文。使用以下参考格式指定报文:

Number 消息的数目。

cur or . (period)
当前的消息。此为缺省值。

first 文件夹中第一个消息。

last 文件夹中最后一个消息。

next 紧跟着当前消息的下一个消息。

prev 在当前消息之前的消息。

-noannotate
禁止注释。这个标志是缺省值。

-nodraftfolder
将草稿放入 *UserMHDirectory*/**draft** 文件中。

-noedit
禁止初始编辑。

-noinplace 在适当的位置上预防注释。这个标志是缺省值。

-nowhatnowproc 禁止 **dist** 命令的交互式处理。**-nowhatnowproc** 标志阻止任何编辑发生。

-whatnowproc *Program* 启动指定程序，以引导重分发任务的完成。如果您指定 **whatnow** 命令为 *Program* 变量，则 **dist** 命令启动一个内部 **whatnow** 过程，而不是文件名为 **whatnow** 的程序。

概要文件条目

将以下条目输入到 *UserMHDirectory***/.mh_profile** 文件:

`Current-Folder:`	设定缺省的当前文件夹。
`Draft-Folder:`	设定缺省的草稿文件夹。
`Editor:`	设定缺省的编辑器。
`fileproc:`	指定用于接力传送报文的程序。
`Path:`	指定用户的 MH 目录。
`whatnowproc:`	指定用于提示 `What now?` 问题的程序。

示例

1. 要从当前文件夹重新分发当前报文，请输入:

   ```
   dist
   ```

   系统提示为报头字段赋值。输入值后按 Enter 键。要跳过该条目，不输入值，按下 Enter 键。您必须填写 `Resent-to:` 字段。完成报头后，请勿修改文本主体。按下 Ctrl-D 按键顺序可以退出编辑器。系统提示:

   ```
   What now?
   ```

   按 Enter 键查看可用选项列表。如果您想重新分发此报文，请输入 `send`。 您的报文就会重新分发到新地址列表。

2. 当存在报文草稿时，要向新地址列表重新分发报文，请输入:

   ```
   dist
   ```

   系统反应与以下报文类似:

   ```
   Draft "$HOME/Mail/draft" exists (43 bytes).
   Disposition? _
   ```

   要重新分发该草稿，请输入:

   ```
   replace
   ```

   系统提示为报头字段赋值。输入值后按 Enter 键。要跳过该条目，不输入值，按下 Enter 键。您必须填写 `Resent-to:` 字段。完成报头后，请勿修改文本主体。按下 Ctrl-D 按键顺序可以退出编辑器。系统提示:

   ```
   What now?
   ```

   按 Enter 键查看可用选项列表。如果您想重新分发此报文，请输入 `send`。 您的报文就会重新分发到新地址列表。

3. 要重新分发来自 `schedules` 文件夹的报文 `15`，请输入:

   ```
   dist  +schedules 15
   ```

   系统提示为报头字段赋值。输入值后按 Enter 键。要跳过该条目，不输入值，按下 Enter 键。您必须填写 `Resent-to:` 字段。完成报头后，请勿修改文本主体。按下 Ctrl-D 按键顺序可以退出编辑器。系统提示:

```
What now?
```

按 Enter 键查看可用选项列表。要重新分发报文，请输入 send 并按下 Enter 键。

文件

/etc/mh/distcomps	包含系统缺省报文格式。
UserMHDirectory/**distcomps**	包含缺省用户报文格式。
UserMHDirectory/**draft**	包含当前草稿文件。
/usr/bin/dist	包含 **dist** 命令的可执行文件格式。

相关信息

ali 命令、**anno** 命令、**comp** 命令、**forw** 命令、**prompter** 命令、**refile** 命令、**repl** 命令、**send** 命令、**whatnow** 命令。

mh_alias 文件、**mh_profile** 文件。

《*网络与通信管理*》中的『邮件应用程序』。

dmadm 命令

用途

在管理服务器上操作网络数据管理工具（NDAF）。

语法

dmadm [**param=***val*]

描述

dmadm 命令通过对应的参数，在 NDAF 域中的管理服务器上设置缺省目录、超时值、日志记录级别、所用安全方法、Kerberos 密钥表路径、Kerberos 主体以及通信端口。

参数

dmadm 命令接受以下可选参数值之一:

[**-rpc_timeout=***val*]
设置 RPC 连接或调用的超时。缺省值为 300 秒。

[**-log_level=***val*]
设置日志文件的日志记录级别。缺省值为 0。可能值包括:

0 严重错误

1 错误

2 警告

3 通知

4 参考

[**-security=***val*]
设置所用的安全方法类型。缺省值为 krb5。值包括:

auth_sys
表示 **uid/gid** 认证

krb5 表示 Kerberos 认证

krb5i 表示 Kerberos 完整性认证

krb5p 表示 Kerberos 隐私认证

[**-krb5_principal=***val*]
设置用于 **kinit** 的 Kerberos 主体。

[**-admin_port=***val*]
设置等待 **dmf** 客户机的 RPC 的 **dmadm** 端口。缺省值为 28000。

[**-serv_port=***val*]
设置等待 **dmadm** RPC 的 **dms** 端口。缺省值为 28001。

[**-ndaf_dir=***val*]
设置 NDAF 的基目录。它包含单元、数据集和副本的缺省数据库、日志和目录。基目录的缺省值为 **/var/dmf**。其他缺省值包括以下目录:

- **${ndaf_dir}/var/dmf/log**（针对日志）
- **${ndaf_dir}/var/dmf/admin**（针对管理数据库）

[**-krb5_keytab=***val*]
指示 Kerberos 密钥表路径。如果在不使用 SRC 的情况下未指定，则，如果已经设定，则使用 **KRB5_KTNAME** 变量；否则，使用 **/etc/krb5/krb5.conf** 文件中指定的缺省密钥表文件。如果在使用 SRC 的情况下未指定，则使用 **/etc/krb5/krb5.conf** 文件中指定的缺省密钥表文件。

[**-admin_cb_port=***val*]
设置等待 **dms** RPC 回调的 **dmadm** 端口。缺省值为 28002。

退出状态

0 命令成功完成。

>0 发生错误。

示例

1. 要在管理服务器上使用 SRC 启动 **dmadm**，请输入:
   ```
   startsrc -s dmadm
   ```
2. 要使用 SRC 启动 **dmadm** 并指定 **auth_sys** 安全性，请输入:
   ```
   startsrc -a "-security=auth_sys" -s dmadm
   ```

位置

/usr/sbin/dmadm

相关信息

『dmf 命令』、第 161 页的『dms 命令』和第 163 页的『dms_enable_fs 命令』。

---

dmf 命令

用途

实现网络数据管理工具（NDAF）管理客户机可执行文件。

语法

dmf *verb object parameter flag*

描述

dmf 命令实现 NDAF CLI，这是 NDAF（网络数据管理工具）管理客户机可执行文件的程序名。NDAF 是一种跨机器网络对文件系统数据进行集中式创建、放置、复制、持续管理和名称空间联合的 AIX 解决方案。

dmf 是 NDAF 中的所有 CLI 命令的前缀。这些命令遵循一致的结构：一个通用前缀，即可执行文件的实际名称（**dmf**）、**create** 或 **delete** 之类的动词、操作所应用于的 *object*，以及任何后续参数（如 *names*）。这些参数是位置相关的。

动词

以下动词与 **dmf** 命令结合使用。

add_to	将键 / 值项添加到对象的基于列表的属性中。
check_adm	检测并报告 NDAF 管理数据库中的不一致性。
check_adm_serv	检查管理数据服务器数据库一致性。
check_serv	检测并报告数据服务器数据库中的不一致性。
clear	当与 “status” 对象一起使用时，清除管理服务器日志。
create	创建逻辑对象。
destroy	破坏对象及其所有内容。
enumerate	获取容器中对象的列表。
master	将另一个副本位置选为主位置。
mount	在联合名称空间中安装 **dset** 或副本。
place	将对象放置在服务器上。
remove_from	从对象的基于列表的属性中除去键 / 值项。
resolve	查找哪个 **dset** 或副本对应于单元中的路径。
set	设置对象的非列表属性的值。
show	显示对象的属性或先前 **dmf** 命令请求的状态。
source	更改副本的源数据集。
unmount	取消导出并破坏服务器上不包含数据的引用。
unplace	在服务器上取消放置对象。
update	使副本及其克隆位置用原始源数据集的内容刷新。
validate	检查某个对象在管理服务器上和该对象的服务器上的一致性。

add_to

dmf add_to *object* [*params*]

add_to 动词将键 / 值项添加到对象的基于列表的属性中。**add_to** 动词的参数为键 / 值对。

参数:

object 指定对象的类型。值包括以下各项（其他参数取决于对象）:

admin 此对象参数采用以下语法:

```
dmf add_to admin key=value [-r] [-a admin_server]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

key=*value*
指定属性以及要赋给它的值。有效键为 **DmPrincipal**。

-r 显示分配给该请求的 **uuid**。

server 此对象参数采用以下语法:

```
dmf add_to server key=value [-r] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定服务器名称。

key=*value*
指定属性以及要赋给它的值。有效键为 **DmPrincipal**、**DmClientDnsName** 和 **DmTransferTable**。

-r 显示分配给该请求的 **uuid**。

cell 此对象参数采用以下语法:

```
dmf add_to cell key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为 **DmPrincipal**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

dset 此对象参数采用以下语法:

```
dmf add_to dset key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为: **DmPrincipal**、**DmOwningRole** 和 **DmTransferTable**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

replica 此对象参数采用以下语法:

```
dmf add_to replica key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为: **DmPrincipal**、**DmOwningRole** 和 **DmTransferTable**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

role 此对象参数采用以下语法:

```
dmf add_to role key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为: **DmPrincipal**、**DmOwningRole**、**DmServer** 和 **DmMember**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

check_adm

dmf check_adm admin [**-a** *machine*]

check_adm 动词检测并报告 NDAF 管理数据库中的不一致性。

该工具比较每条记录，并在每次遇到不匹配时填写错误报告。只要客户机命令已正确执行，返回码就将为 0。所有其他问题（如 CLI 和管理服务器之间的通信问题）将返回非空错误。

注: 当其他 NDAF 操作正在运行时，不应该使用 **check_adm** 动词，因为这可能导致不准确的报告结果。

check_adm_serv

dmf check_adm_serv admin [**-a** *machine*] [**-c** *server*]

dmf check_adm_serv admin [**-a** *machine*]

或

dmf check_adm_serv server [**-a** *machine*] [**-c** *server*]

check_adm_serv 动词检查管理数据服务器数据库一致性。

该工具比较每条记录，并在每次遇到不匹配时填写错误报告。只要客户机命令已正确执行，返回码就将为 0。所有其他问题（如 CLI 和管理服务器之间的通信问题）将返回非空错误。

注: 当其他 NDAF 操作正在运行时，不应使用 **check_adm_serv** 动词，因为这可能导致不准确的报告结果。

check_serv

dmf check_serv server [**-a** *machine*] [**-c** *server*]

check_serv 动词检测并报告数据服务器数据库中的不一致性。

该工具比较每条记录，并在每次遇到不匹配时填写错误报告。只要客户机命令已正确执行，返回码就将为 0。所有其他问题（如 CLI 和管理服务器之间的通信问题）将返回非空错误。

注: 当其他 NDAF 操作正在运行时，不应使用 **check_serv** 动词，因为这可能导致不准确的报告结果。

clear

dmf clear status [**-r**] [**-a** *admin_server*]

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-r 显示分配给该请求的 **uuid**。

clear 动词在与 **status** 对象一起使用时，将清除管理服务器日志。所有的历史活动随后都将失去。

create

dmf create *object* [*params*]

create 动词创建逻辑对象。随之一起指定的 address 参数必须指向该对象的容器。根据创建的对象的类型，需要不同数量的参数。

参数:

object 指定创建的对象的类型。值包括以下各项（其他参数取决于对象）：

admin 需要将要赋予管理服务器的名称作为参数。此对象参数采用以下语法:

```
dmf create admin name [-r] [-a admin_server]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

name 指定要创建的管理服务器的名称。

-r 显示分配给该请求的 **uuid**。

注: 输入 `dmf create admin my_admin` 也会创建 **my_admin** 服务器对象。

server 需要服务器的名称、其 DNS 名称或 IP 地址和端口。此对象参数采用以下语法:

```
dmf create server name dns_target [-e] [-r] [-a admin_server]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

dns_target
指定服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-e 指定该对象对于 NDAF 是外部对象。

name 指定要创建的数据服务器的名称。

-r 显示分配给该请求的 **uuid**。

cell 需要赋予单元的名称。此对象参数采用以下语法:

```
dmf create cell name  [-w timeout] [-r] [-a admin_server]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

name 指定要创建的单元的名称。

-r 显示分配给该请求的 **uuid**。

-w *timeout*
指定该命令在完成之前可等待多长时间。

dset 需要 **dset** 的名称、宿主服务器以及（可选）服务器上的本地路径。此对象参数采用以下语法:

```
dmf create dset name  server  [path] [-r] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定容器（例如，单元名称）。

name 指定要创建的数据集的名称。

path 指定服务器上的本地路径。如果省略了 *path* 参数，则服务器将 **dset** 放置在其缺省池中。

-r 显示分配给该请求的 **uuid**。

server 指定服务器名称。

replica 需要副本的名称、宿主服务器以及（可选）服务器上的本地路径。此对象参数采用以下语法:

```
dmf create replica name  server  [path] [-d | -w timeout] [-r] [-a admin_server]
[-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定容器（例如，单元名称）。

-d 指定该命令必须异步运行。

name 指定要创建的副本的名称。

-o *object*
指定此命令所寻址到的对象的名称。

path 指定服务器上的本地路径。如果省略了 *path* 参数，则服务器将把副本放置在其缺省副本池中。

-r 显示分配给该请求的 **uuid**。

server 指定服务器名称。

-w *timeout*
指定该命令在完成之前可等待多长时间。

role 需要将创建的角色的名称。此对象参数采用以下语法:

```
dmf create role name  [-r] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定容器（例如，单元名称）。

-r 显示分配给该请求的 **uuid**。

destroy

dmf destroy *object* [*params*]

destroy 动词破坏对象及其所有内容。依赖于该对象的那些对象也将被破坏。例如，如果数据集被破坏，则其所有内容也被破坏。如果单元被破坏，则其所有数据集和副本也被破坏。*address* 参数指向要破坏的对象。

参数:

object 指定被破坏的对象的类型。值包括以下各项（其他参数取决于对象）：

admin 此对象参数采用以下语法:

```
dmf destroy admin  [-r] [-f] [-a admin_server]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-f 强制执行该操作而无需确认。

-r 显示分配给该请求的 **uuid**。

server 此对象参数采用以下语法:

```
dmf destroy server [-r] [-f] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定服务器名称。

-f 强制执行该操作而无需确认。

-r 显示分配给该请求的 **uuid**。

cell 此对象参数采用以下语法:

```
dmf destroy cell [-r] [-f] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-f 强制执行该操作而无需确认。

-r 显示分配给该请求的 **uuid**。

dset 此对象参数采用以下语法:

```
dmf destroy dset [-r] [-f] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-f 强制执行该操作而无需确认。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

replica 此对象参数采用以下语法:

```
dmf destroy replica [-r] [-f] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-f 强制执行该操作而无需确认。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

role 此对象参数采用以下语法:

```
dmf destroy role [-r] [-f] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-f 强制执行该操作而无需确认。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

enumerate

dmf enumerate *object* [*params*]

enumerate 动词获取容器内对象的列表，如单元内的数据集。它接受一个两部分参数。第一部分是以下列表中的某个关键字。

对象	选择器	
admin	**cell**	在该管理服务器上创建的单元的列表。
	server	依赖于该管理服务器的服务器的列表。
	admin	在该管理服务器上创建的管理员的列表。
server	**dset**	在该服务器上创建的数据集的列表。
	replica	在该服务器上创建的副本的列表。
cell	**dset**	属于该单元的数据集的列表。
	replica	属于该单元的副本的列表。
	role	已为该单元定义的角色的列表。
dset	**server**	放置了该数据集的服务器的列表。
replica	**server**	放置了该副本的服务器的列表。

第二部分可选并且可省略，这是一个以文本匹配模式存在的过滤器，它使用 ? 匹配单个字符，使用 * 匹配多个字符。这个第二部分用于将列表限制为只包含与过滤器匹配的对象。

参数:

object 指定对象的类型。值包括以下各项（其他参数取决于对象）：

admin 此对象参数采用以下语法:

```
dmf enumerate admin type [pattern] [-r] [-a admin_server]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

pattern 可选的匹配文本模式。有效值为 **?** 和 *****。

-r 显示分配给该请求的 **uuid**。

type 指定要返回的对象的类型。有效值为 **server**、**cell** 和 **admin**。

server 此对象参数采用以下语法:

```
dmf enumerate server type [pattern] [-r] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定服务器名称。

pattern 可选的匹配文本模式。有效值为 **?** 和 *****。

-r 显示分配给该请求的 **uuid**。

type 指定要返回的对象的类型。有效值为 **dset** 和 **replica**。

cell 此对象参数采用以下语法:

```
dmf enumerate cell type [pattern] [-r] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

pattern 可选的匹配文本模式。有效值为 **?** 和 *****。

-r 显示分配给该请求的 **uuid**。

type 指定要返回的对象的类型。有效值为 **dset**、**replica** 和 **role**。

master

dmf master replica *server* [*path*] [**-r**] [**-a** *admin_server*] [**-c** *container*] [**-o** *object*]

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-o *object*
指定此命令所寻址到的对象的名称。

path 指定要查找的路径。

-r 显示分配给该请求的 **uuid**。

server 指定服务器名称。

master 动词将另一个副本位置选为主位置。如果您想更新副本的主位置，则可使用此项。主位置是在执行任何 **update** 命令时所更新的第一个位置。其他位置随后异步更新。

mount

dmf mount *object* [*params*]

mount 动词在联合名称空间中安装数据集或副本，并使其在单元中对 NFS 客户机可见。实际上，会将对数据集（创建时在 NFSv4 中导出）或副本的 NFSv4 引用添加到单元中。

参数:

object 指定创建的对象的类型。值包括以下各项（其他参数取决于对象）：

dset 此对象参数采用以下语法:

```
dmf mount dset mount_path  [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

mount_path
指定名称空间中的安装路径。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

replica 此对象参数采用以下语法:

```
dmf mount replica mount_path  [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

mount_path
指定名称空间中的安装路径。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

place

place 动词在服务器上放置对象。它可应用于单元、副本或数据集。其参数取决于对象的类型。该操作对于单元和对于副本或数据集是完全不同的。

对于单元，**place** 动词用于使单元通过服务器可见。单元导出在服务器的 **nfsroot** 下，并且包含对安装的数据集和副本的引用。**dmf** 命令 **place cell** 只接受一个参数，即服务器的名称。单元的根名称空间随后放置在服务器上。

对于副本，**place** 动词在服务器上的指定位置创建副本的克隆。如果副本已安装在单元中，则对此克隆位置的引用将添加到返回给 NFS 客户机的引用列表中。此列表中的引用顺序取决于网络亲缘关系。副本的每个克隆位置在更新操作请求时异步更新。**dmf place replica** 命令将服务器或者服务器上的本地路径（可选）作为参数。例如:

```
dmf place replica my_server local_path -a my_admin -c my_cell -o my_replica
```

对于数据集，**place** 动词用在集群文件系统环境（如 GPFS）中，通过集群中的不同服务器提供相同的数据集统一视图。**-m** 标志必须准确。对目标数据集不执行任何 NDAF 管理或操作。例如:

```
dmf place dset my_external_server external_server_path -m -a my_admin -c my_cell -o my_dset
```

place dset 操作仅可用于集群文件系统环境（如 GPFS），其中底层数据的统一视图由该系统而并非由 NDAF 来呈现。

参数:

cell 此对象参数采用以下语法:

```
dmf place cell server  [-r] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-r 显示分配给该请求的 **uuid**。

server 指定服务器名称。

dset 此对象参数采用以下语法:

```
dmf place dset server  [path] [-d | -w timeout] [-r] [-m] [-a admin_server] [-c container]
 [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-d 指定该命令必须异步运行。

-m 指定已创建数据集的数据将在 NDAF 外部管理（用于集群机器）。

path 指定要查找的路径。

-r 显示分配给该请求的 **uuid**。

server 指定服务器名称。

-w *timeout*
指定该命令在完成之前可等待多长时间。

replica

此对象参数采用以下语法:

```
dmf place replica server  [path] [-d | -w timeout] [-r] [-a admin_server] [-c container]
 [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-d 指定该命令必须异步运行。

path 指定要查找的路径。

-r 显示分配给该请求的 **uuid**。

server 指定服务器名称。

-w *timeout*
指定该命令在完成之前可等待多长时间。

remove_from

dmf remove_from *object* [*params*]

remove_from 动词从对象的基于列表的属性中除去键 / 值项。**remove_from** 动词的参数为键 / 值对。

参数:

object 指定对象的类型。值包括以下各项（其他参数取决于对象）：

admin 此对象参数采用以下语法:

```
dmf remove_from admin key=value [-r] [-a admin_server]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

key=*value*
指定属性以及要赋给它的值。有效键为 **DmPrincipal**。

-r 显示分配给该请求的 **uuid**。

server 此对象参数采用以下语法:

```
dmf remove_from server key=value [-r] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定服务器名称。

key=*value*
指定属性以及要赋给它的值。有效键为 **DmPrincipal**、**DmClientDnsName** 和 **DmTransferTable**。

-r 显示分配给该请求的 **uuid**。

cell 此对象参数采用以下语法:

```
dmf remove_from cell key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为 **DmPrincipal**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

dset 此对象参数采用以下语法:

```
dmf remove_from dset key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为: **DmPrincipal**、**DmOwningRole** 和 **DmTransferTable**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

replica 此对象参数采用以下语法:

```
dmf remove_from replica key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为: **DmPrincipal**、**DmOwningRole** 和 **DmTransferTable**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

role 此对象参数采用以下语法:

```
dmf remove_from role key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为: **DmPrincipal**、**DmOwningRole**、**DmServer** 和 **DmMember**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

resolve

dmf resolve cell *path* [**-r**] [**-a** *admin_server*] [**-c** *container*] [**-o** *object*]

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-o *object*
指定此命令所寻址到的对象的名称。

path 指定要查找的路径。

-r 显示分配给该请求的 **uuid**。

resolve 动词查找哪个数据集或副本对应于单元中的路径。其参数为要查找的路径（给定路径为 **dmf mount** 命令所指定的 NDAF 安装点的路径）。

set

dmf set *object* [*params*]

set 动词设置对象的非列表属性的值。这些属性是单独的属性，与 **add_to** 动词有别。**set** 动词的参数为键/值对。

参数:

object 指定对象的类型。值包括以下各项（其他参数取决于对象）:

server 此对象参数采用以下语法:

```
dmf set server key=value [-r] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定服务器名称。

key=*value*
指定属性以及要赋给它的值。有效键为: **DmMinRpcPort**、**DmMaxRpcPort**、**DmDefaultRepPath**、**DmDefaultDsetPath**、**DmDTAPort** 和 **DmLogLevel**。

-r 显示分配给该请求的 **uuid**。

cell 此对象参数采用以下语法:

```
dmf set cell key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为: **DmLogLevel** 以及 **DmLocsMax**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

dset 此对象参数采用以下语法:

```
dmf set dset key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为: **DmOwner**、**DmGroup**、**DmMode** 以及 **DmLocsMax**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

replica 此对象参数采用以下语法:

```
dmf set replica key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为: **DmOwner**、**DmGroup**、**DmMode** 以及 **DmLocsMax**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

role 此对象参数采用以下语法:

```
dmf set role key=value [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

key=*value*
指定属性以及要赋给它的值。有效键为: **DmCreateDs**、**DmDestroyDs**、**DmModifyDs**、**DmDuplicateDs**、**DmCreateRole**、**DmDestroyRole** 以及 **DmModifyRole**。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

show

dmf show *object* [*params*]

show 动词显示对象的属性或者先前的 **dmf** 命令请求的状态。当不带任何标志使用 **dmf show** 命令时，所用的缺省值为 **~/.dmf/address** 中存在的那些值（通过 **-h** 标志显示）。

参数:

object 指定被破坏的对象的类型。值包括以下各项（其他参数取决于对象）：

admin 此对象参数采用以下语法:

```
dmf show admin  [-r] [-a admin_server]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-r 显示分配给该请求的 **uuid**。

server 此对象参数采用以下语法:

```
dmf show server [-r] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定服务器名称。

-r 显示分配给该请求的 **uuid**。

cell 此对象参数采用以下语法:

```
dmf show cell [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

dset 此对象参数采用以下语法:

```
dmf show dset [-r] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-r 显示分配给该请求的 **uuid**。

replica 此对象参数采用以下语法:

```
dmf show replica [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

role 此对象参数采用以下语法:

```
dmf show role [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

status 此对象参数采用以下语法:

```
dmf show status depth [-r] [-a admin_server]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

depth 指定要返回多少条记录。

-r 显示分配给该请求的 **uuid**。

source

dmf source replica *source_dset* [**-r**] [**-a** *admin_server*] [**-c** *container*] [**-o** *object*]

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-o *object*
指定此命令所寻址到的对象的名称。

source_dset
指定成为副本新源的数据集。

-r 显示分配给该请求的 **uuid**。

source 动词更改副本的源数据集。

unmount

dmf unmount *object* [*params*]

unmount 动词取消导出并破坏服务器上不包含数据的引用。其唯一的参数是数据集的现有安装点。

注: 如果 NFS 客户机已经解析了某个引用，则即使在卸装之后，它仍然能够访问此引用。如果该引用是在复制的数据集中，则在该引用因卸装而被除去之前，副本必须进行更新。

参数:

object 指定创建的对象的类型。值包括以下各项（其他参数取决于对象）：

dset 此对象参数采用以下语法:

```
dmf unmount dset mount_path  [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

mount_path
指定名称空间中的安装路径。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

replica 此对象参数采用以下语法:

```
dmf unmount replica mount_path  [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

mount_path
指定名称空间中的安装路径。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

unplace

单元和副本均可从服务器中取消放置。视 **dmf unplace** 用于单元或副本而定，该操作会有所不同。

对于单元，**unplace** 操作将避免 NFS 用户将单元安装在该服务器上。

dmf unplace cell *server* [**-r**] [**-f**] [**-a** *admin_server*] [**-c** *container*]

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-f 强制执行该操作而无需确认。

-r 显示分配给该请求的 **uuid**。

server 指定服务器名称。

unplace cell 动词将从服务器中除去单元。它不从数据集或副本中除去数据，并且不会在命令行所指定的服务器上取消导出这些数据的路径。数据将一直可供此服务器上其他客户机访问。**unplace cell** 动词将通过该服务器上的 NFS 取消导出单元。该对象在其他服务器上仍保持可见。

对于副本，**unplace** 操作将除去副本的位置。

dmf unplace replica *server* [*path*] [**-r**] [**-f**] [**-a** *admin_server*] [**-c** *container*]

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-f 强制执行该操作而无需确认。

path 指定要查找的路径。

-r 显示分配给该请求的 **uuid**。

server 指定服务器名称。

unplace replica 动词将从服务器中除去副本。**unplace replica** 动词不从数据集或副本中除去数据，并且不会在命令行所指定的服务器上取消导出这些数据的路径。数据将一直可供此服务器上其他客户机访问。

对于副本，**unplace** 动词除去该服务器上副本的克隆位置。此位置将取消导出并且其内容将遭破坏。对此位置的引用将从 NFS 针对单元中的此副本所返回的引用列表中除去。该副本的其他位置保持不变。第一个副本位置（用 **create** 命令创建）称为副本的*主位置*。此位置不可取消放置，除非另一个位置被选作主位置以替代第一个位置（有关更多信息，请参阅 master 动词）。

如果要除去副本，必须指定服务器和本地路径。确认之后，该路径指定的位置上的数据将受到破坏并且该路径取消导出。如果服务器上的副本只有一个位置，则路径的指定不是必需的。如果存在多个可用位置，则会返回一条错误消息。

注:

1. 副本无法从管理服务器进行更新。
2. 副本的最后一个位置不可取消放置。请改用 **destroy** 动词。

update

dmf update replica [**-d** | **-w** *timeout*] [**-r**] [**-a** *admin_server*] [**-c** *container*] [**-o** *object*]

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-d 指定该命令必须异步运行。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

-w *timeout*
指定该命令在完成之前可等待多长时间。

update 动词使副本及其克隆位置用原始源数据集的内容刷新。

validate

dmf validate *object* [*params*]

validate 动词检查对象在管理服务器上以及该对象的服务器上的一致性。查询将发送到管理服务器守护程序,它将查询其数据库以及对象的服务器的数据库。发现的任何一致的内容都将返回。

参数:

object 指定验证的对象的类型。值包括以下各项（其他参数取决于对象）:

server 此对象参数采用以下语法:

```
dmf validate server [-r] [-a admin_server] [-c container]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定服务器名称。

-r 显示分配给该请求的 **uuid**。

dset 此对象参数采用以下语法:

```
dmf validate dset [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

replica 此对象参数采用以下语法:

```
dmf validate replica [-r] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定管理服务器的 DNS 名称或 IP 地址。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

role 此对象参数采用以下语法:

```
dmf validate role [-r] [-f] [-a admin_server] [-c container] [-o object]
```

其中:

-a *admin_server*
指定要创建的数据服务器的名称。可使用冒号分隔符添加端口。

-c *container*
指定单元名称。

-f 强制执行该操作而无需确认。

-o *object*
指定此命令所寻址到的对象的名称。

-r 显示分配给该请求的 **uuid**。

对象

admin 表示管理服务器守护程序，并且用于配置管理服务器。此类的单个对象可在运行管理服务器守护程序的服务器上创建。运行管理服务器守护程序的机器还必须运行数据服务器守护程序。创建管理服务器对象时，还会创建同名的数据服务器对象。

server 表示系统中所有数据服务器。同时还设置此服务器上托管的数据的缺省属性以及常规服务器配置的缺省属性。

cell 表示单元。单元是管理和名称空间单位，由管理服务器托管，但独立于由该管理服务器托管的所有其他单元。一个单元包含其自己的名称空间、数据集组成，还有其自己的基于角色的安全性对象。单元可将其数据集放置在任何服务器上，只要该服务器是为托管该单元的管理服务器定义的。

dset 表示读 / 写数据集，包括那些在本地或集群文件系统上托管的数据集。此对象类创建数据集并管理其属性、安装和移动。

replica 表示数据集的只读副本，该副本可跨多个服务器分布。此对象类创建副本并管理其属性、安装和移动。

role 表示分配给一组 Kerberos 主体的一组特权，用于管理单元内的资源。

status 表示发送给服务器的给定请求的状态。

标志

-a 标识管理服务器，命令与附加的字符串参数一起应发送到该服务器。
-c 标识容器，该容器中含有此命令与附加的字符串参数一起所发送到的对象。
-d 启动异步运行的服务器请求。请求一旦启动，该命令即会返回。
-e 在服务器创建时，指定所创建的对象引用外部 NDAF 服务器，并且实际上未在 NDAF 数据服务器上运行。
-f 当与 **destroy** 或 **unplace** 一起使用时，强制执行命令，并且不提示确认。
-m 指定创建的数据集的数据将在 NDAF 外部管理（通常用于集群机器）。
-o 与附加的字符串参数一起，指定此命令所寻址到的对象的名称（**dset**、**replica** 或 **role**）。
-r 使 CLI 向控制台显示分配给管理服务器所生成的请求的 UUID。这对于通过 **dmf show status** 跟踪请求完成情况非常有用。
-w 指定 CLI 等待操作的异步部分在超时之前完成的持续时间（缺省值为 120 秒）。此标志接受数字参数。单位为秒。

退出状态

0 命令成功完成。
>0 发生错误。

示例

1. 要在运行 **dms** 和 **dmadm** 守护程序的主机名上创建 **admin** 对象及其链接的数据服务器，请输入:

   ```
   dmf create admin my_admin -a admin_host
   ```

2. 要在联合中以逻辑方式创建一个新服务器（将服务器添加到联合中），其中该新服务器名为 *server_name*，其 DNS 名称为 *server_dns_name*:

   ```
   dmf create server server_name server_dns_name -a admin_host
   ```

   该机器上必须运行 **dms** 守护程序。

3. 要在管理服务器上创建一个单元，该单元将作为 NFS 客户机所安装的名称空间的根，请输入:

   ```
   dmf create cell my_cell -a admin_host
   ```

4. 要在该单元中创建一个数据集，并且该数据集名为 *my_dset*，请输入:

   ```
   dmf create dset my_dset server_name server_path -a admin_host -c my_cell
   ```

   数据集数据将位于 *server_dns_name* 上，在 *server_path* 中。

5. 要创建该数据集的副本，请输入:

   ```
   dmf create replica my_replica server_name replica_path -a admin_host -c my_cell -o my_dset
   ```

位置

/usr/bin/dmf

相关信息

第 131 页的『dmadm 命令』、第 161 页的『dms 命令』和第 163 页的『dms_enable_fs 命令』。

dmpuncompress 命令

用途
恢复堆压缩文件。

语法
/usr/bin/dmpuncompress [**-f**] [*File*]

描述
dmpuncompress 命令恢复在堆放时压缩的原始堆文件。

File 参数指定的每个压缩文件已被除去并由已展开的副本替换。展开的文件具有与已压缩版本相同的名称，但是没有 **.BZ** 扩展名。如果用户具有 root 权限，则展开的文件保留与原始文件相同的所有者、组、方式和修改时间。如果用户没有 root 用户权限，则文件保留相同的方式和修改时间，但获得新的所有者和组。

标志
-f *File*　强制扩展。如果文件已存在则覆盖它。系统不提示用户将覆盖现有文件。文件大小实际不会缩小。

退出状态
0　成功完成。
>0　发生错误。

示例
要解压 **dump.BZ** 文件，请输入:

```
/usr/lib/ras/dmpuncompress dump.BZ
```

dump.BZ 文件被解压缩并重命名为 **dump**。

位置
/usr/bin/dmpuncompress

相关信息
savecore 命令、**snap** 命令和 **uncompress** 命令。

AIX 5L Version 5.3 Kernel Extensions and Device Support Programming Concepts 中的 System Dump Facility。

dms 命令

用途
在客户机数据服务器上操作网络数据管理工具（NDAF）。

语法

dms [**param=***val*]

描述

dms 命令通过对应的参数，在 NDAF 域中的数据服务器上设置缺省目录、超时值、日志记录级别、所用安全方法、Kerberos 密钥表路径、Kerberos 主体以及通信端口。

参数

dms 命令接受以下可选参数值之一:

[**-rpc_timeout=***val*]
设置 RPC 连接或调用的超时。缺省值为 300 秒。

[**-log_level=***val*]
设置日志文件的日志记录级别。缺省值为 0。可能值包括:

0 严重错误

1 错误

2 警告

3 通知

4 参考

[**-security=***val*]
设置所用的安全方法类型。缺省值为 krb5。值包括:

auth_sys
表示 **uid/gid** 认证

krb5 表示 Kerberos 认证

krb5i 表示 Kerberos 完整性认证

krb5p 表示 Kerberos 隐私认证

[**-krb5_principal=***val*]
设置用于 **kinit** 的 Kerberos 主体。

[**-serv_port=***val*]
设置等待 **dmadm** RPC 的 **dms** 端口。缺省值为 28001。

[**-serv_serv_port=***val*]
设置等待其他 **dms** RPC 的 **dms** 端口。缺省值为 28003。

[**-ndaf_dir=***val*]
设置 NDAF 的基目录。它包含单元、数据集和副本的缺省数据库、日志和目录。基目录的缺省值为 **/var/dmf**。其他缺省值包括以下目录:

- **${ndaf_dir}/log**（针对日志）
- **${ndaf_dir}/server**（针对数据服务器数据库）
- **${ndaf_dir}/server/dsets**（针对数据集，如果未设置 **-ndaf_dataset_default** 参数）
- **${ndaf_dir}/server/replicas**（针对副本，如果未设置 **-ndaf_replica_default** 参数）

注: 至少必须指定 **-ndaf_dataset_default** 和 **-ndaf_replica_default** 参数，或者 **-ndaf_dir** 参数。在包含指定目录（用来存储数据集和副本）的文件系统上，必须已经使用 **dms_enable_fs** 命令允许创建单元、数据集和副本。

[**-ndaf_dataset_default=***val*]
设置数据集的缺省目录。

注: 至少必须指定 **-ndaf_dataset_default** 和 **-ndaf_replica_default** 参数，或者 **-ndaf_dir** 参数。在包含指定目录（用来存储数据集和副本）的文件系统上，必须已经使用 **dms_enable_fs** 命令允许创建单元、数据集和副本。

[**-ndaf_replica_default=***val*]	设置副本的缺省目录。 **注**：至少必须指定 **-ndaf_dataset_default** 和 **-ndaf_replica_default** 参数，或者 **-ndaf_dir** 参数。在包含指定目录（用来存储数据集和副本）的文件系统上，必须已经使用 **dms_enable_fs** 命令允许创建单元、数据集和副本。
[**-krb5_keytab=***val*]	指示 Kerberos 密钥表路径。如果在不使用 SRC 的情况下未指定，则，如果已经设定，则使用 **KRB5_KTNAME** 变量；否则，使用 **/etc/krb5/krb5.conf** 文件中指定的缺省密钥表文件。如果在使用 SRC 的情况下未指定，则使用 **/etc/krb5/krb5.conf** 文件中指定的缺省密钥表文件。
[**-admin_cb_port=***val*]	设置等待 **dms** RPC 回调的 **dmadm** 端口。缺省值为 28002。

退出状态

0	命令成功完成。
>0	发生错误。

示例

1. 要使用 SRC 在数据集服务器上启动 **dms**，请输入：
   ```
   startsrc -s dms
   ```
2. 要使用 SRC 启动 **dms** 并指定 **auth_sys** 安全性，请输入：
   ```
   startsrc -a "-security=auth_sys" -s dms
   ```

位置

/usr/sbin/dms

相关信息

第 131 页的『dmadm 命令』、第 132 页的『dmf 命令』和『dms_enable_fs 命令』。

---

dms_enable_fs 命令

用途

启用、禁用或查询在文件系统上创建单元、数据集和副本的能力。

语法

dms_enable_fs [**-sqh**] *pathname*

描述

dms_enable_fs 命令启用、禁用或查询在文件系统上创建单元、数据集和副本的能力。它在文件系统的根目录中生成 **.DSETINFO** 目录。该目录不得删除。

标志

-h 显示 **dms_enable_fs** 命令的用法。

-q 检查 VFS（VFS 内的路径名）是否已启用。如果启用，则返回 0。否则，返回非零值。

-s 为文件集启用 VFS（VFS 的路径名）。

退出状态

0 命令成功完成。

>0 发生错误。

示例

1. 要为数据集启用 **/ndafexp** 文件系统，请输入：

```
dms_enable_fs -s /ndafexp
```

位置

/usr/sbin/dms_enable_fs

相关信息

第 131 页的『dmadm 命令』、第 132 页的『dmf 命令』和第 161 页的『dms 命令』。

---

dnssec-keygen 命令

用途

DNSSEC 密钥生成工具。

语法

dnssec-keygen [**-a** *algorithm*] [**-b** *keysize*] [**-n** nametype] [**-c** *class*] [**-e**] [**-g** *generator*] [**-h**] [**-p** *protocol*] [**-r** *randomdev*] [**-s** *strength*] [**-t** *type*] [**-v** *level*] [*name*]

描述

dnssec-keygen 命令为 DNSSEC（安全 DNS）生成密钥，见 RFC 2535 定义。也可以生成与 TSIG（事务特征符）一起使用的密钥，见 RFC 2845 中定义。

标志

-a *algorithm* 选择加密算法。algorithm 的值必须为 RSAMD5、RSA、DSA、DH（Diffie Hellman）或 HMAC-MD5 之一。这些值不区分大小写。注意对于 DNSSEC 而言，DSA 是一个强制性的实现算法，推荐使用 RSA。对 TSIG 而言，HMAC-MD5 是强制性的。

-b *keysize* 指定密钥位数。密钥大小的选项取决于所用的算法。RSA 密钥大小必须在 512 到 2048 位之间。Diffie Hellman 密钥必须在 128 和 4096 位之间。DSA 密钥必须在 512 到 1024 位之间，而且必须是 64 的整数倍。HMAC-MD5 密钥必须在 1 到 512 位之间。

-n *nametype* 指定密钥的所有者类型。nametype 的值必须是 ZONE（针对 DNSSEC 区域密钥）、HOST 或 ENTITY（针对主机关联密钥）、USER（针对用户关联密钥）。这些值不区分大小写。

-c *class*	指定包含密钥的 DNS 记录应该有指定类。如果未指定，则使用类 IN。
-e	如果生成 RSA 密钥，使用大指数。
-g *generator*	如果生成 Diffie Hellman 密钥，则使用该发生器。允许值为 2 和 5。如果不指定生成器，可能的话使用来自于 RFC 2539 的已知质数；否则缺省值为 2。
-h	打印 dnssec-keygen 的选项和参数的简短摘要。
-p *protocol*	为生成的密钥设定协议值。协议为 0 到 255 间的某个数。类型 USER 的缺省值是 2（email），其他类型的缺省值是 3（DNSSEC）。该参数的其他可能值列在 RFC 2535 和它的后续作业中。
-r *randomdev*	指定随机源。如果操作系统不提供 **/dev/random** 或等价设备，缺省随机源为键盘输入。randomdev 指定包含要使用随机数据的字符设备名或文件名而不是缺省值。特殊值键盘指示应该使用的键盘输入。
-s *strength*	指定密钥的强度值。该强度为 0 到 15 间的某个数，目前在 DNSSEC 中尚无定义用途。
-t *type*	指定密钥的使用。类型必须是 AUTHCONF、NOAUTHCONF、NOAUTH 或 NOCONF 之一。缺省值是 AUTHCONF。AUTH 指验证数据的能力，CONF 指加密数据的能力。
-v *level*	设置调试级别。

生成密钥

当 **dnssec-keygen** 成功完成时，它显示如 **Knnnn.+aaa+iiiii** 的字符串到标准输出。这是生成的密钥的标识字符串。这些字符串可作为 **dnssec-makekeyset** 的参数。

- **nnnn**是密钥名。
- **aaa** 算法的数字表示。
- **iiiii** 是密钥标识符（或占地面积）。

dnssec-keygen 创建两个名称基于所显示字符串的文件。**Knnnn.+aaa+iiiii.key** 包含公用密钥，而 **Knnnn.+aaa+iiiii.private** 包含专用密钥。**.key** 文件包含一个 DNS 密钥记录，它可以插入到一个区域文件（直接或使用 $INCLUDE 语句）。**.private** 文件包含算法特定字段。出于安全性原因，该文件一般没有读许可权。**.key** 和 **.private** 文件由诸如 HMAC-MD5 等对称加密算法创建，即使公共密钥和专用密钥相同。

示例

要为域 example.com 生成 768-bit 的 DSA 密钥，请输入以下命令:

```
dnssec-keygen -a DSA -b 768 -n ZONE example.com
```

此命令显示如下格式的字符串:

```
Kexample.com.+003+26160
```

在这个示例中，**dnssec-keygen** 创建了文件 **Kexample.com.+003+26160.key** 和 **Kexample.com.+003+26160.private**。

相关信息

dnssec-makekeyset 命令、**dnssec-signkey** 命令、**dnssec-signzone** 命令。

BIND 9 管理员参考手册。

RFC 2535、RFC 2845 和 RFC 2539。

dnssec-makekeyset 命令

用途

DNSSEC 区域签名工具。

语法

dnssec-makekeyset [**-a**] [**-s** *start-time*] [**-e** *end-time*] [**-h**] [**-p**] [**-r** *randomdev*] [**-t** *ttl*] [**-v** *level*] {*key...*}

描述

dnssec-makekeyset 命令生成 **dnssec-keygen** 创建的一个或多个密钥组成的密钥集。它创建一个文件，包含针对每个密钥的密钥记录，用每个区域密钥自签署密钥集。输出文件的形式为 **keyset-***nnnn***.**，其中 ***nnnn*** 是区域名。

标志

-a 验证所有生成的签名。

-s *start-time* 在生成的 SIG 记录合法化时，指定日期和时间。可以是绝对或相对时间。绝对起始时间格式为 YYYYMMDDHHMMSS 数字符号；20000530144500 表示 2000 年 5 月 30 日 14:45:00 UTC。相对起始时间用 +N 指示，表示截至到现在为止的秒数为 N 秒。如果 ***start-time*** 未指定，使用当前时间。

-e *end-time* 生成的 SIG 记录到期时，指定日期和时间。若指定 *start-time*，绝对时间的格式为 YYYYMMDDHHMMSS。相对于启动时间的时间用 **+N** 标记，表示距启动时间 N 秒。相对当前时间的时间用 **now+N** 标记。如果未指定 ***end-time***，缺省值为从启动时间算起 30 天。

-h 显示 **dnssec-makekeyset** 的选项和参数的简短摘要。

-p 签署区域时使用伪随机数据。这比使用真正的随机数据快，但安全性差。这个选项在标记大区域或者熵源有限时非常有用。

-r *randomdev* 指定随机源。如果操作系统不提供 **/dev/random** 或类似设备，缺省随机源为键盘输入。randomdev 指定包含要使用的随机数据的字符设备名或文件名而不是缺省值。特定的键盘值指定使用键盘输入。

-t *ttl* 指定记录 KEY 和 SIG 的生存时间。缺省值为 3600 秒。

-v *level Sets* 调试级别。

参数

key 密钥集文件要包含的密钥列表。这些密钥的格式为 Knnnn.+aaa+iiiii，由 **dnssec-keygen** 生成。

示例

下面的命令针对 **example.com** 生成一个包含 DSA 密钥的密钥集，该密钥创建在 **dnssec-keygen** 的联机帮助页。

```
dnssec-makekeyset -t 86400 -s 20000701120000 -e +2592000 Kexample.com.+003+26160
```

在这个示例中，**dnssec-makekeyset** 创建文件 **keyset-example.com.**。该文件包含指定密钥和自建签名。**example.com** 的 DNS 管理器可以发送 **keyset-example.com.** 给 **.com** 的 DNS 管理器要求签名，如果 **.com** 区域是 DNSSEC 识别的，而且两个区域的管理器拥有相互认证、安全的进行密钥和签名交换的机制。

相关信息

dnssec-keygen 命令和 **dnssec-signkey** 命令。

BIND 9 管理员参考手册。

RFC 2535。

---

dnssec-signkey 命令

用途

DNSSEC 密钥集签名工具。

语法

dnssec-signkey [**-a**] [**-c** *class*] [**-s** *start-time*] [**-e** *end-time*] [**-h**] [**-p**] [**-r** *randomdev*] [**-v** *level*] *keyset key*

描述

dnssec-signkey 命令签署一个密钥集。典型的密钥集是子区域的，并且将由 **dnssec-makekeyset** 创建。子区域的密钥集由父区域用区域键签名。输出文件的格式为 **signedkey-***nnnn*，其中 ***nnnn*** 区域名称。

标志

-a 验证所有生成的签名。

-c *class* 指定密钥集的 DNS 类。

-s *start-time* 在生成的 SIG 记录变有效时，指定日期和时间。可以是绝对或相对时间。绝对启动时间格式为 YYYYMMDDHHMMSS 数字符号；20000530144500 表示 2000 年 5 月 30 日 14:45:00 UTC。相对启动时间用 **+N** 标记，其中 N 表示离现在的秒数。如果 *start-time* 未指定，使用当前时间。

-e *end-time* 生成 SIG 记录到期时，指定日期和时间。若带有启动时间，绝对时间的格式为 YYYYMMDDHHMMSS 符号。相对启动时间用 **+N** 标记，表示距启动时间 N 秒。相对当前时间的时间记录用 **now+N** 标记。如果未指定 *end-time*，缺省值为从启动时间算起 30 天。

-h 显示 **dnssec-signkey** 的选项和参数的简短摘要。

-p 签署区域时使用伪随机数据。这比使用真正的随机数据快，但安全性差。这个选项在标记大区域或者熵源有限时非常有用。

-r *randomdev* 指定随机源。如果操作系统不提供 **/dev/random** 或类似设备，缺省随机源为键盘输入。*randomdev* 指定包含随机数的字符设备名或文件名而不是缺省值。特定的键盘值指定使用键盘输入。

-v *level* 设置调试级别。

参数

keyset 包含子密钥集的文件。

key 用于子密钥集签名的密钥。

示例

DNSSEC 识别的 **.com** 区域的 DNS 管理员将使用以下命令来签署由 **dnssec-makekeyset** 带 **dnssec-keygen** 生成的密钥创建的 **example.com**:

```
dnssec-signkey keyset-example.com. Kcom.+003+51944
```

在这个示例中，**dnssec-signkey** 创建文件 **signedkey-example.com.**，它包含 **example.com** 密钥和 **.com** 密钥生成的签名。

相关信息

dnssec-keygen 命令、**dnssec-makekeyset** 命令、**dnssec-signzone** 命令。

---

dnssec-signzone 命令

用途

DNSSEC 区域签名工具。

语法

dnssec-signzone [**-a**] [**-c** *class*] [**-d** *directory*] [**-s** *start-time*] [**-e** *end-time*] [**-h**] [**-i** *interval*] [**-n** *nthreads*] [**-o** *origin*] [**-p**] [**-r** *randomdev*] [**-t**] [**-v** *level*] *zonefile key...*

...

描述

dnssec-signzone 命令签署一个区域。它生成 NXT 和 SIG 记录，并产生区域的已签署的版本。如果父区域有文件 **signedkey**，父域的签名将合并到生成的已签署区域文件。已签署区域的授权的安全状态（即，无论子域是否安全）由每个子域是否含有 **signedkey** 文件决定。

标志

-a 验证所有生成的签名。

-c *class* 指定区域的 DNS 类。

-d *directory* 作为目录在目录中查找 **signedkey** 文件。

-s *start-time* 在生成的 SIG 记录变有效时，指定日期和时间。可以是绝对或相对时间。绝对启动时间格式为 YYYYMMDDHHMMSS 数字符号；20000530144500 表示 2000 年 5 月 30 日 14:45:00 UTC。相对启动时间用 **+N** 标记，表示距当前时间 N 秒。如果 *start-time* 未指定，使用当前时间。

-e *end-time* 生成 SIG 记录到期时，指定日期和时间。若带有启动时间，绝对时间的格式为 YYYYMMDDHHMMSS 符号。相对启动时间用 **+N** 标记，表示距启动时间 N 秒。相对当前时间的时间记录用 **now+N** 标记。如果未指定 *end-time*，缺省值为从启动时间算起 30 天。

-f *output-file* 包含签名区域的输出文件名。缺省为送到输入文件的 **append .signed**。

-h 显示 **dnssec-signzone** 的选项和参数的简短摘要。

-i *interval* 当前一个签名区域作为输入传入时，记录被重新签名。时间间隔选项以距离目前时间的偏移量（秒数）指定间隔周期。如果在间隔周期后记录 SIG 过期了，则保留它。否则，认为它立即会过期，并替换它。缺省间隔周期为签名启动时间和结束时间的间隔的四分之一。因此如果未指定 *end-time* 和 *start-time*，**dnssec-signzone** 生成合法期 30 天、间隔周期 7.5 天的签名。因此，任何存在的记录 SIG 如果将在 7.5 天内过期，则会被替换。

-n *ncpus* 指定使用的线程数。缺省情况下每个检测到的 CPU 启动一个线程。

-o *origin* 区域原点。若不指定，假定区域文件的名称为原点。

-p 签署区域时使用伪随机数据。这比使用真正的随机数据快，但安全性差。这个选项在标记大区域或者熵源有限时非常有用。

-r *randomdev* 指定随机源。如果操作系统不提供 **/dev/random** 或类似设备，缺省随机源为键盘输入。*randomdev* 指定包含随机数的字符设备名或文件名而不是缺省值。特定的键盘值指定使用键盘输入。

-t 完成时显示统计信息。

-v *level* 设置调试级别。

参数

zonefile 包含需被签名的区域的文件。设置调试级别。
key 用于签署子密钥集的密钥。

示例

以下命令以在 **dnssec-keygen** 联机帮助页中生成的 DSA 密钥签署 **example.com** 区域。区域密钥必须在区域中。如果存在与本区域或任何子域关联的 **signedkey** 文件，它们必须在当前目录 **example.com**，发出以下命令:

```
dnssec-signzone -o example.com db.example.com Kexample.com.+003+26160
```

这个示例中，**dnssec-signzone** 创建文件 **db.example.com.signed**。该文件必须在文件 **named.conf** 中的区域语句中引用。

相关信息

dnssec-keygen 命令、**dnssec-makekeyset** 命令、**dnssec-signkey** 命令。

BIND 9 管理员参考手册。

RFC 2535。

---

dodisk 命令

用途

启动磁盘使用计数。

语法

/usr/sbin/acct/dodisk [**-X**] [**-o**] [*File* ...]

描述

dodisk 命令通过调用 **diskusg** 命令和 **acctdisk** 命令来启动磁盘使用计数。如果您指定带 **dodisk** 命令的 **-o** 标志，则会通过使用 **acctdusg** 命令启动登录目录进行更加全面却更慢的磁盘计数。正常情况下，**cron** 守护程序运行 **dodisk** 命令。

在缺省情况下，**dodisk** 命令只对 **/etc/filesystems** 文件中的章节或包含属性 **account=true** 的指定文件进行磁盘计数。如果您用 *File* 参数指定文件名，只对这些文件进行磁盘计数。

如果不指定标志 **-o**，则 *File* 参数应该包含可安装文件系统的特别文件名称。如果同时指定 **-o** 标志和 *File* 参数，则文件应该是已安装文件系统的安装点。

注: 您不应该在分布式环境中共享节点中需要记帐的文件。每一节点应该拥有不同记帐文件的自身副本。

标志

-o 调用 **acctdusg** 命令（而不是 **diskusg** 命令）来通过登录目录启动磁盘计数。
-X 处理每个用户名的全部可用字符，而不是截断为前 8 个字符。

安全性

访问控制: 该命令的执行权限只授予用户组 **adm** 中的成员。

示例

1. 启动自动磁盘使用计数，添加以下内容到文件 **/var/spool/cron/crontabs/root** file:

   ```
   0 2 * * 4 /usr/sbin/acct/dodisk
   ```

   这个示例表明了 **cron** 守护程序将读取和执行的指示信息。**dodisk** 命令在每个星期四（4 上午 2 点（0 2）运行。此命令仅是通常情况下给予 **cron** 的记帐指令之一。请参阅《*操作系统与设备管理*》中『设置记帐系统』，以获取关于典型 **cron** 记帐条目的更多信息。

2. 要在包含大于 8 个字符用户名的系统上运行磁盘使用情况记帐，请将以下行添加至 **/var/spool/cron/crontabs/root** 文件中:

   ```
   0 2 * * 4 /usr/sbin/acct/dodisk -X
   ```

文件

/usr/sbin/acct	记帐命令路径
/etc/filesystems	包含文件系统信息。

相关信息

acctdisk 命令、**acctdusg**、**diskusg** 命令。

cron 守护程序。

有关记帐系统、日报表和月报表的准备以及记帐文件的更多信息，请参阅《*操作系统与设备管理*》中的『系统记帐』。

《*操作系统与设备管理*》中的『设置记帐系统』说明了建立记帐系统必须采取的步骤。

---

domainname 命令

用途

显示或设置当前网络信息服务（NIS）域的名称。

语法

/usr/bin/domainname [*DomainName*]

描述

domainname 命令显示或设置当前网络信息服务（NIS）域的名称。如果不指定参数，**domainname** 命令显示当前 NIS 域的名称。一个域通常包含同一管理器下的一组主机。

只有 root 用户可以通过 **domainname** 命令的参数设置域名。

示例

1. 要加入新域，请输入:

   ```
   domainname caesar
   ```

   这个示例中，domainname 命令设置 NIS 的域名为 caesar。
2. 要找出您的主机所在域的名称，请输入:

   ```
   domainname
   ```

相关信息

ypinit 命令。

ypbind 守护程序、**ypserv** 守护程序。

AIX 5L Version 5.3 Network Information Services (NIS and NIS+) Guide 中的 Network Information Service(NIS) Overview for System Management。

NIS Reference。

---

dosdel 命令

用途

删除 DOS 文件。

语法

dosdel [**-v**] [**-D** *Device*] *File ...*

描述

dosdel 命令删除由 *File* 参数指定的 DOS 文件。使用 **-v** 标志获取有关磁盘的格式化信息。

DOS 文件命名约定用于异常情况。因为 \ （反斜杠）字符对操作系统而言具有特殊含义，所以需要用 / （斜杠）字符作为 DOS 路径名中的子目录定界符。**dosdel** 命令在检查磁盘前，将文件或目录名中的小写字符转换成大写字符。因为假定所有的文件名是全（不是相对）路径名，所以无需添加 / （斜杠）。

标志

-D*Device* 指定 DOS 设备名为 **/dev/fd0** 或 **/dev/fd1**。缺省设备是 **/dev/fd0**。
-v 将有关该磁盘格式的信息写到标准输出。用此标志验证此设备为一个 DOS 磁盘。

示例

要在缺省设备上删除 DOS 文件，请输入:

```
dosdel file.ext
```

文件

/usr/bin/dosdel 包含 **dosdel** 命令。

相关信息

dosdir 命令、**dosformat** 命令、**dosread** 命令、**doswrite** 命令。

《*操作系统与设备管理*》中的『文件』描述了文件、文件类型以及如何命名文件。

---

dosdir 命令

用途

列出 DOS 文件的目录。

语法

dosdir [**-l** [**-e**]] [**-a**] [**-d**] [**-t**] [**-v**] [**-D** *Device*] [*File ...* | *Directory ...*]

描述

dosdir 命令显示指定 DOS 文件或目录的信息。如果您未指定 **-d** 标志的同时指定了目录，则 **dosdir** 命令显示该目录中的文件信息。

DOS文件命名约定用于异常情况。因为字符 \ （反斜杠）对操作系统而言具有特殊含义，所以需要用字符 / （斜杠）作为 DOS 路径名中的子目录定界符。**dosdir** 目录在检查磁盘前将文件名或目录名中的小写字符转换成大写。因为假定所有的文件名是全（不是相对）路径名，所以无需添加 / （斜杠）。

标志

-a 写所有文件的信息。包括隐藏文件和系统文件，以及 .（点）和 ..（点点）文件。

-d 将 *File* 参数作为文件处理，即便指定了目录。当用 *Directory* 参数指定目录时，则不显示该目录下的文件信息，但显示该目录的本身信息。

-D*Device* 指定 DOS 设备名为 **/dev/fd0** 或 **/dev/fd1**。缺省设备为 **/dev/rfd0**。

-e 使用 **-l** 标志来将集群的列表分配给文件。

-l 建立包括创建日期、大小（以字节计）和文件属性等的集群列表。子目录的大小为指定为 0 字节。这些属性包含以下意义:

A（归档）
自从文件最后一次修改后没有进行备份。

D（目录）
该文件是一个子目录，且不在正常的 DOS 目录搜索范围内。

H（隐藏）
该文件不在正常的 DOS 目录搜索范围内。

R（只读）
文件不能被修改。

S（系统）
文件为系统文件，且不在正常的 DOS 目录搜索范围内。

-t 从指定目录开始列举整个目录树。

-v 将磁盘格式信息写到标准输出。用此标志验证此设备为一个 DOS 磁盘。

示例

要读取在 **/dev/fd0** 上的 DOS 文件的目录，请输入:

```
dosdir
```

该命令返回文件名和磁盘空间信息。

```
PG3-25.TXT
PG4-25.TXT
PG5-25.TXT
PG6-25.TXT
Free space: 312320 bytes
```

要读取在 **/dev/fd1** 上的 DOS 文件的目录，请输入:

```
dosdir -D/dev/fd1
```

该命令返回文件名和磁盘空间信息。

```
PG7-25.TXT
PG8-25.TXT
PG9-25.TXT
PG10-25.TXT
Free space: 312320 bytes
```

文件

/usr/bin/dosdir 包含 **dosdir** 命令。

相关信息

dosdel 命令、**dosformat** 命令、**dosread** 命令、**doswrite** 命令。

《*操作系统与设备管理*》中的『文件』描述了文件、文件类型以及如何命名文件。

---

dosformat 命令

用途

格式化 DOS 软盘。

语法

dosformat [**-V** *Label*] [**-D** *Device* | **-4**]

描述

dosformat 命令以 DOS 格式格式化软盘。

缺省设备和 DOS 软盘驱动器格式为 3.5 英寸软盘 **/dev/fd0**。密度通常为 1.44M 或 2.88M 字节，取决于驱动器支持的密度。其他 DOS 软驱格式由 **-D** 或 **-4** 标志实现。

要包含卷标，使用 **-V** 标志。

> **注**: 该命令的用途是方便操作系统和 DOS 系统之间的文件传输。不推荐使用该命令格式化需要 DOS 系统启动文件的软盘。

标志

-V 将 *Label* 参数作为 DOS 卷标写到软盘。

-D*Device* 指定软驱类型和大小。指定 *Device* 参数如下:

对于 3.5-英寸、1.44M 软驱:

/dev/fd0
1.44MB （缺省）

/dev/fd0h
1.44MB

/dev/fd0l
720KB

/dev/fd0.18
1.44MB

/dev/fd0.9
720KB

对于 3.5 英寸，2.88M 的软驱:

/dev/fd0
2.88MB （缺省）

/dev/fd0h
2.88MB

/dev/fdol
720KB

/dev/fd0.36
2.88MB

/dev/fd0.18
1.44MB

/dev/fd0.9
720KB

对于 5.25 英寸，1.2M 软驱:

/dev/fd0
1.2MB （缺省）

/dev/fd0.15
1.2MB

/dev/fd0.9
360KB

-4 指定软盘大小为低密度。

示例

1. 要以卷标 “homework” 格式化 3.5 英寸、1.44M 软盘，请输入:

   ```
   dosformat -V homework
   ```

2. 要格式化 5.25 英寸、360K 软盘，请输入:

```
dosformat -D /dev/fd1.9
```

或者

```
dosformat -D /dev/fd1 -4
```

文件

/usr/bin/dosformat 包含 **dosformat** 命令。

相关信息

dosdel 命令、**dosdir** 命令、**dosread** 命令、**doswrite** 命令。

---

dosread 命令

用途

复制 DOS 文件。

语法

dosread [**-a**] [**-v**] [**-D** *Device*] *File1* [*File2*]

描述

dosread 命令复制由 *File1* 变量指定的 DOS 文件到标准输出或到由 *File2* 变量指定的文件。如果不为 *File2* 变量指定路径名，则将 DOS 文件复制到根目录。

除非另行说明，**dosread** 命令为 *File1* 变量指定的文件复制由目录条目指定数目的字节数。这就是说，您不能复制目录，因为目录约定记录大小为 0。

您可以使用 DOS 文件命名约定异常处理: \（反斜杠）。因为 \ 字符对于 DOS 而言具有特殊含义，所以用 /（斜杠）作为 DOS 路径名的子目录定界符。**dosdir** 目录在检查磁盘前将文件名或目录名中的小写字符转换成大写。因为假定所有的文件名是全（不是相对）路径名，所以无需在前面添加 / （斜杠）。

注:

1. **dosread** 命令不会将 * 和 ?（星号和问号）通配符解释为具有特殊含义的字符。如果不指定文件扩展名，则该文件名将认为您已指定了一个空的扩展名。
2. 您不能定制命令的名称。命令必须被命名为 **dosread**。
3. **dosread** 命令从包含 DOS 软盘的缺省驱动器读取文件。**dosread** 命令以操作系统可辨认的文件形式将文件复制到当前目录。如果 DOS 软盘包含子目录，**dosread** 目录不在操作系统上创建对应的新子目录。您必须创建子目录，指定您想复制到新的子目录的 DOS 文件。

标志

-a 以新行字符替换每一 CR-LF（回车符、换行）按键顺序，并将 Ctrl-Z（ASCII SUB）按键顺序解释成行结束字符。

-D*Device* 指定 DOS 设备名为 **/dev/fd0** 或 **/dev/fd1**。*Device* 变量的缺省值为 **/dev/fd0**。该设备必须为 DOS 磁盘格式。

-v 将磁盘格式文件信息写到标准输出。用此标志验证此设备为一个 DOS 磁盘。

示例

1. 要从 DOS 复制一个文本文件，请输入:

   ```
   dosread  -a chap1.doc chap1
   ```

   该命令序列将 DOS 文本文件 \CHAP1.DOC 从缺省设备 **/dev/fd0** 复制到当前目录的 chap1。
2. 要从 DOS 软盘复制一个二进制文件，请输入:

   ```
   dosread -D/dev/fd1 /survey/test.dta /home/fran/testdata
   ```

   该命令序列将 DOS 数据文件 \SURVEY\TEST.DTA 从 **/dev/fd1** 复制到 /home/fran/testdata。
3. 要复制软盘上的所有 DOS 文件，请输入:

   ```
   dosdir | awk '!/There are/ {print $1}'|xargs -t -i dosread {} {}
   ```

   该命令序列从包含 DOS 磁盘的缺省设备获取文件，并复制到当前目录。

文件

/usr/bin/dosread	包含 **dosread** 命令。
/dev/fd0	包含软驱的设备名。

相关信息

awk 命令、**dosdel** 命令、**dosdir** 命令、**dosformat** 命令、**doswrite** 命令、**xargs** 命令。

《*操作系统与设备管理*》中的『目录』。

《*操作系统与设备管理*》中的『文件类型』描述了文件、文件类型以及如何命名文件。

---

doswrite 命令

用途

复制文件到 DOS 文件

语法

doswrite [**-a**] [**-v**] [**-D***Device*] *File1 File2*

描述

doswrite 命令将 *File1* 参数指定的文件复制到由 *File2* 参数指定的 DOS 文件。**doswrite** 命令复制文件到单个 DOS 软盘。**doswrite** 命令不能跨多个 DOS 软盘复制文件。

doswrite 命令将 *File2* 参数指定的文件用标准 DOS 命名约定写到 DOS 设备。因为 DOS \ （反斜杠）字符对于 DOS 操作系统而言具有特定含义，当指定 *File2* 参数内子目录名时不使用 \（反斜杠）。而用 /（斜杠）代替。

doswrite 命令在它检查 DOS 设备前将 *File1* 参数中的小写字符转换成大写。因为假定所有的文件名是全（不是相对）路径名，所以无需在开始添加 / （斜杠）。

如果 *File2* 参数指定的文件包含 / （斜杠），每一中间部分必须作为目录存在，但是末尾部分（文件名）不一定要存在。任何同名的现有文件将被覆盖。

注:

1. 通配符 * 和 ?（星号和问号）不会由该命令以特殊方式来处理（尽管它们由 shell 处理）。如果不指定文件扩展名，则此文件名将认为您已指定了一个空的扩展名。
2. 该命令必须命名为 **doswrite**。
3. 一个 DOS 目录最多可包含 244 个文件。

标志

-a 用 CR-LF（回车符、换行）序列替换 NL（新建行）。添加 Ctrl-Z 到文件末尾的输出。
-D *Device* 指定 DOS 设备名为 **/dev/fd0** 或 **/dev/fd1**。缺省设备为 **/dev/rfd0**。该设备必须为 DOS 磁盘格式。
-v 将磁盘格式信息写到标准输出。用此标志验证此设备为一个 DOS 磁盘。

示例

1. 要将一个文本文件复制到 DOS 软盘，请输入:

   ```
   doswrite  -a chap1 chap1.doc
   ```

   该命令将当前目录上的文件 chap1 复制到缺省设备 **/dev/fd0** 上的 DOS 文本文件 \CHAP1.DOC。
2. 要将一个二进制文件复制到 DOS 软盘，请输入:

   ```
   doswrite -D/dev/fd1 /home/fran/testdata /survey/test.dta
   ```

   这会将数据文件 /home/fran/testdata 复制到 **/dev/fd1** 上的 DOS 文件 \SURVEY\TEST.DTA。
3. 要将当前目录的所有文件复制到缺省驱动器上的 DOS 软盘，请输入:

   ```
   for i in *
   执行
   doswrite $i $i
   done
   ```

文件

/usr/bin/doswrite 包含 **doswrite** 命令。
/dev/fd0 包含软驱的设备名。

相关信息

《*操作系统与设备管理*》中的『文件』描述了文件、文件类型以及如何命名文件。

dosdel 命令、**dosdir** 命令、**dosformat** 命令、**dosread** 命令。

---

dp 命令

用途

解析和重新格式化日期。

语法

dp [**-form** *File* | **-format** *String*] [**-width** *Number*] *Date*

描述

dp 命令解析并重新格式化日期。**dp** 命令不是由用户启动。**dp** 命令由其他程序调用，通常通过它的全路径名 **/usr/lib/mh/dp**。

dp 命令将每个指定的邮件报头字符串解析为日期，并试图重新格式化该字符串。**dp** 命令的缺省输出格式是 ARPA RFC 822 标准。对每个无法解析的字符串，**dp** 命令都显示一条错误消息。

参数

Date 指定要解析的日期。

标志

-form *File* 将 *Date* 参数中指定的日期重新格式化为由 *File* 变量描述的备用格式。

-format *String* 将 *Date* 参数中指定的日期重新格式化为由 *String* 变量指定的备用格式。缺省格式字符串如下:

```
%<(nodate{text})error:%{text}%|%(putstr(pretty{text}))%>
```

-help 列出命令语法、可利用的转换（切换）和版本信息。

注: 对于消息处理程序（MH）来说，标志的名字必须完整地拼写出来。

-width *Number* 设定 **dp** 命令用于显示日期和错误消息的最大列数。缺省为显示器宽度。

文件

$HOME/.mh_profile 包含 MH 用户概要文件。

/etc/mh/mtstailor 包含 MH 命令定义。

相关信息

ap 命令。

《*网络与通信管理*》中的『邮件应用程序』。

---

dpid2 守护程序

用途

作为后台进程启动 **dpid2** DPI-SMUX 转换器守护程序。

语法

dpid2 [**-d** [*Level*]]

描述

dpid2 命令启动 **dpid2** DPI-SMUX 转换器守护程序。这个命令只能由具有根特权的用户或系统组的成员发出。

dpid2 DPI-SMUX 转换器守护程序遵守由 RFC 1592 定义的标准简单网络管理协议分布式协议接口 V2.0，以及由 RFC 1227 定义的 SNMP MUX 协议和 MIB。

dpid2 作为 SMUX 转换器的 DPI® 2.0。它用于使 DPI 子代理程序（如 **/usr/sbin/hostmibd**）可与 AIX SNMP V1 代理程序通信。转换器将 DPI2 消息转换成 SMUX 协议消息，反之亦然。**dpid2** 本身作为 SMUX 同级实现。它连接作为 **snmpd** 代理程序一部分的 SMUX 服务器的 TCP 端口 199。对于 DPI2 子代理程序而言（例如 **/usr/sbin/hostmibd**），**dpid2** 相当于一个 DPI2 代理程序。它在任意 TCP 端口侦听来自 DPI2 子代理程序的连接请求。端口号由带 **snmpd** 代理程序的 **dpid2** 守护程序通过 MIB 变量 dpiPortForTCP（1.3.6.1.4.1.2.2.1.1.1）注册。DPI2 子代理程序通过对 dpiPortForTCP.0（1.3.6.1.4.1.2.2.1.1.1.0）实例发送一个查询请求到 **snmpd** 代理程序，从 **snmpd** 代理程序获知这个端口号。在 DPI2 子代理程序知道 DPI2 代理程序正在侦听的 TCP 端口号后，会尝试连接它。

dpid2 守护程序通常在系统启动期间且调用 **/etc/rc.tcpip** shell 脚本时执行。

dpid2 守护程序应该使用系统资源控制器（SRC）控制。不推荐在命令行输入 **dpid2** 命令。

使用以下 SRC 命令操作 **dpid2** 守护程序:

startsrc
启动一个子系统、子系统组，或是一个子服务器。

stopsrc
停止一个子系统、子系统组，或是一个子服务器。

refresh
使子系统或子系统组重新读取适当的配置文件。

lssrc 获取一个子系统、子系统组，或是一个子服务器的状态。

注: 在 AIX 发行版 5.2.0 及其之后的版本上，**snmpdv3** 代理程序本身作为一个 DPI2 代理程序并侦听 dpiPortForTCP.0 TCP 端口。因此，**dpid2** 在使用 **snmpdv3** 代理程序时没有必要使用。所以，**dpid2** 守护程序在系统启动时不执行，并且 **/etc/rc.tcpip** 中的 **dpid2** 行会被注释出。

标志

-d *Level* 指定跟踪 / 调试级别。

8 DPI 级别 1

16 DPI 级别 2

32 Internal 级别 1

64 Internal 级别 2

128 Internal 级别 3

增加多跟踪级别。

注: 如果指定了 **-d** 标志，但未指定级别，则缺省级别为 56。如果未指定 **-d** 标志，则缺省级别为 0。

示例

1. 要启动 **dpid2** 守护程序，输入以下类似命令:

```
startsrc -s dpid2 -a "-f /tmp/dpid2.log"
```

此命令在调试级别 0 上启动 **dpid2** 守护程序和日志文件到 **/tmp/dpid2.log** 文件。

2. 要停止 **dpid2** 守护程序，通常输入:

```
stopsrc -s dpid2
```

该命令停止 **dpid2** 守护程序。**-s** 标志指定随后停止的子系统。

3. 要从 **hostmbid** 获取短状态，请输入:

```
lssrc -s dpid2
```

该命令返回了守护程序的名字、进程标识和状态（活动或未活动）。

文件

/etc/snmpd.conf	在 snmpd v1 代理程序配置文件中指定 SMUX 同级条目。
/etc/snmpd.peers	为同级 smux 指定配置信息。
/etc/mib.defs	定义 SNMP 代理和管理器应该识别和处理的管理信息库（MIB）变量。

相关信息

snmpdv1 守护程序、**hostmibd** 命令。

/etc/snmpd.peers 文件。

---

drm_admin 命令

用途

管理基于数据复制管理器（DRM）的服务器，例如 **glbd**，全局位置代理（GLB）的复制版本。

语法

drm_admin [**-version**]

描述

drm_admin 工具管理基于数据复制管理器（DRM）的服务器，例如 **glbd**，全局位置代理（GLB）的复制版本。

通过 **drm_admin**，您可以检查或修改副本列表，合并数据库来强制汇合副本，终止服务器和删除副本。

drm_admin 的功能是管理数据库备份，不是更改其包含的数据。例如您可以使用 **drm_admin** 合并两个 GLB 数据库的副本，但您必须通过 **lb_admin** 在数据库中添加一个新条目。而且，虽然 **drm_admin** 可以停止或删除一个 GLB 副本，但是如果您想启动或创建一个副本，必须直接调用 **glbd**。

一旦调用，**drm_admin** 就进入交互方式，它接收以下描述的命令。

标志

-version 显示 **glbd** 所属的 NCS 版本，但不启动该守护程序。

子命令

多数 **drm_admin** 命令操作一个在缺省主机上（*DefaultHost*）的缺省对象（*DefaultObj*）。*DefaultObj* 和 *DefaultHost* 一起指定一个缺省副本。缺省值由设置命令建立，一直保存到被另一个设置更改。

目前为止，GLB 是唯一的已知对象。

一些 **drm_admin** 命令操作不同于缺省值的主机。我们用 *OtherHost* 标识该主机。

您所用的 *DefaultHost* 或 *OtherHost* 主机名采用 *Family:Host* 格式，主机可用名称或网络地址指定。例如，`ip:jeeves`、`ip:bertie` 和 `ip:#192.5.5.5` 都是合法主机名。

addrep *OtherHost*

将 *OtherHost* 添加到位于 *DefaultHost* 的副本列表。*DefaultHost* 中的副本会为 *DefaultObj* 将 *OtherHost* 发布到所有其他副本列表。

chrep -from *OtherHost* **-to** *NewOtherHost*

更改 *DefaultHost* 副本列表中的 *OtherHost* 网络地址为 *NewOtherHost*。*DefaultHost* 中的副本会将该更改传播到所有其他关于 *DefaultObj* 的副本列表。**chrep** 命令可能失败，如果 *DefaultObj* 的一个副本在 *OtherHost* 上运行，或者 *OtherHost* 不在 *DefaultHost* 上的副本列表。

delrep *OtherHost*

删除位于 *OtherHost* 上 *DefaultObj* 的副本。**delrep** 命令告诉 *OtherHost* 上的副本:

1. 将它的传播队列中的所有条目传播出去。
2. 向所有其他副本传播一个删除请求，使得 *OtherHost* 从关于 *DefaultObj* 的所有其他副本列表中删除。
3. 删除 *DefaultObj* 的副本。
4. 终止运行。

delrep 命令立刻返回 **drm_admin** 提示符，但是在既不稳定又不完整的配置中真正删除副本可能花费较长时间。通过列举主机上正运行的进程，您可以查看副本删除守护程序是否终止。

info

获取 *DefaultHost* 上 *DefaultObj* 副本的状态信息。

lrep [**-d**] [**-clocks**] [**-na**]

列举存放在 *DefaultHost* 的副本列表中的 *DefaultObj* 副本。

-d 列举现存和已删除的副本。

-clocks
显示每台主机的当前时间，表明副本间的时间差异。

-na 列举每台主机的网络地址。

merge {**-from** | **-to**} *OtherHost*

从一个副本到另一个副本将条目复制到 *DefaultObj* 数据库和副本列表中。如果目标数据库中的对应条目不存在或者有更早的时间戳记，则复制条目。

合并过程不引起条目的传播。源数据库和副本列表不发生更改。

-from 选项复制 *DefaultObj* 数据库和副本列表（位于 *OtherHost*）中的条目到 *DefaultObj* 数据库和副本列表（位于 *DefaultHost*）。

-to 选项将 *DefaultHost* 上的数据库和副本列表的条目复制到 *OtherHost* 上的数据库和副本列表。

跟随在 **merge -to** 之后的 **merge -from** 使两台主机上的副本合并。

merge_all
将 *DefaultHost* 作为所有 *DefaultObj* 副本的全局合并中心。对于 *DefaultHost* 副本列表上的每台主机，**merge_all** 首先执行 **merge -from**，接着又执行 **merge -to**。从而所有 *DefaultObj* 副本就强制为一致的状态了。**merge_all** 操作不引起条目的传播。

您应该在如下情况使用 **merge_all**:

清除一个副本。

复位一个副本。

一个副本已经两周或更长时间不可访问了。

一个副本已经在物理上不可访问了（例如，磁盘损坏引起的数据库破坏）。

monitor [**-r n**]
该命令使 **drm_admin** 每隔 n 分钟读取每个 *DefaultObj* 副本的时钟，并报告时钟差异或没有响应的副本。如果不指定**-r**，周期为 15 分钟。

purgerep *OtherHost*
从 *DefaultHost* 上的副本列表清除 *OtherHost*。*DefaultHost* 上的副本因此向保留在列表中的主机副本发送一个删除请求，然后将 *OtherHost* 从所有其他 *DefaultObj* 副本列表中删去。删除请求不发到 *OtherHost*。

purgerep 可能导致数据丢失，且仅在副本已经物理不可访问时使用。您应该在 **purgerep** 后进行 **merge_all** 操作，以防止剩下的 *DefaultObj* 数据库副本不一致。如果已清除副本仍在运行，应该进行复位处理。

我们推荐使用 **chrep**（而不是 **addrep** 和 **purgerep**）更改副本列表上的条目。

quit
退出 **drm_admin** 会话。

reset *OtherHost*
复位 *OtherHost* 上的 *DefaultObj* 副本。

reset 命令通知 *OtherHost* 上的副本删除 *DefaultObj* 的副本，并且停止运行。它不会导致其他副本列表上的 *OtherHost* 被删除。该命令可能引起数据丢失，除非先成功的进行 **merge_all** 操作。

set [**-o** *ObjName*] **-h** *HostName*
设置缺省对象和主机。所有后来的命令将在 *ObjName* 上运行。不指定主机的后继命令将被发送到 *HostName*。如果不指定 **-o** 选项，则 **drm_admin** 保持当前 *DefaultObj*。

如果设置 **-o** 选项，则 **drm_admin** 用指定对象的副本检查所有主机上的时钟。

stop
为运行在 *DefaultHost* 上的 *DefaultObj* 停止服务器。

示例

下面的示例启动 **drm_admin**，设置缺省对象为 GLB，缺省主机为 mars:

```
/etc/ncs/drm_admin drm_admin: set -o glb -h dds:mars
 Default object: glb default host: dds:mars
 state: in service
 Checking clocks of glb replicas
 dds:mars 1987/04/09.17:09
 dds:pluto 1987/04/09.17:09
 dds:mercury  1987/04/09.17:07
```

相关信息

lb_admin 命令。

glbd（NCS）守护程序。

drmgr 命令

用途

drmgr 命令用于安装和配置动态逻辑分区（DLPAR）脚本。

语法

drmgr { **-i***script_name* [**-w** *minutes*] [**-f**] | **-u** *script_name* } [**-D***hostname*]

drmgr [**-b**]

drmgr [**-R** *script_install_root_directory*]

drmgr [**-S** *syslog_ID*]

drmgr [**-l**]

描述

DLPAR 脚本由系统管理员和供应商提供，通过添加或删除与操作系统相关的应用程序和 / 或中间件，以协调资源的消耗（例如，特定处理器和大量固定内存）。DLPAR 脚本在 DLPAR 操作之前和之后均调用。提供了 DLPAR 脚本以便应用程序可以清楚的停顿并重新启动。

安装脚本时，**drmgr** 将脚本复制到一个专用库。它的缺省位置是 **/usr/lib/dr/scripts/all**。用户可以通过 **-r** *base_script_directory* 选项为该库指定一个备用位置。另外，用户也可以通过指定 **-D** *hostname* 选项，安装仅可在选定主机上执行的脚本。*hostname* 参数作为基本路径的扩展，并使用 '**uname -n**' 命令与当前主机名比较。如果使用 **-D** 参数安装脚本，则卸载时也必须使用它。

注意无法组合以上指定的不同操作标志。也就是说，用户无法组合 **-r** 和 **-S** 标志，**-l** 和 **-r** 标志，等等。

标志

-b	该选项将重建由 **drmgr** 管理的脚本信息文件。通常情况下，该选项只在从另一个系统恢复脚本时使用。
-D *hostname*	此标志指定可以调用脚本的主机名。
-f	强制现有脚本的替换。
-i *script_name*	此标志用于安装脚本。*script_name* 是以全路径安装的脚本。如果未指定路径，则假定是当前目录。如果出现任何名称冲突，**drmgr** 将给出警告，不再安装脚本。可以通过指定 **-f** 标志，可以覆盖任何现有脚本。
-l	该选项显示当前安装的 DLPAR 脚本的相关详细信息。
-R *base_script_directory*	该选项用于更改基本脚本的安装目录。
-S *syslog_ID*	这个标识字符串在记录 syslog 消息时，用作 syslog 标识字符串。注意这个标识字符串通过 **drmgr** 命令追加到每一个登录到 syslog 中的条目。
-u *script_name*	卸载 DLPAR 脚本。如果脚本是以 **-D** 选项安装，则在卸载时也必须使用相同的参数。如果未指定目录，**drmgr** 将试图从 "all" 安装目录中除去 DLPAR 脚本。
-w *minutes*	覆盖供应商为脚本指定的时间限制值。脚本在它超过指定时间限制时会终止。

退出状态

0 成功完成请求操作

>0 命令失败。失败原因可能为以下之一:

- 文件 / 目录不存在。
- 参数长度超过系统限制（PATH_MAX）。
- 指定了太多参数。
- 您必须有 root 用户权限来运行该命令。

相关信息

《*AIX 5L V5.3 通用编程概念：编写并调试程序*》中的动态逻辑分区。

---

drslot 命令

用途

管理可动态重配置插槽，例如热拔插插槽。

语法

标识一个热拔插插槽

drslot -i { -s *Slot* | **-l** *DeviceName* } **-c** *ConnectorType*

为配置设备准备一个热拔插插槽

drslot -a -s *slot* **-c** *ConnectorType* [**-l**]

为设备删除准备一个热拔插插槽

drslot -r { **-s** *slot* | **-l** *DeviceName*} **-c** *ConnectorType* [**-l**]

为设备删除和替换准备一个热拔插插槽

drslot -r { **-s** *slot* | **-l** *DeviceName* } **-c** *ConnectorType* [**-l**]

描述

drslot 命令管理可动态重配置插槽，也就是支持热拔插的插槽。热拔插插槽是连接实体的插件点，可在不关闭电源或重新引导操作系统的情况下进行配置。对于添加（**-a**）操作，必须用 **-s** 标志直接指定插槽，并赋予该插槽一个唯一标识。对于识别（**-i**）、除去（**-r**）和替换（**-r**）操作，可以直接用 **-s** 标志指定或间接指定。可以用 **-l** 标志（给出连接到插槽的设备的逻辑名）间接指定插槽。**drslot** 命令确定连接到指定设备的插槽，并对其进行管理。

注:

1. 除非指定插槽上的设备取消配置，否则除去和替换操作将失败。关于如何成功取消配置某个设备的更多信息，请参阅《*操作系统与设备管理*》中的『管理热插拔接口』。
2. 添加或替换操作后，您必须运行 **cfgmgr** 命令以激活新设备，供操作系统使用。

标志

注: 请勿同时使用 **-a**、**-i**、**-r** 和 **-r** 标志。

-a 为连接到热拔插插槽的设备配置进行准备。第一次识别插槽时，会给出它的确认提示信息。接着，提示您确认设备已经连接到插槽上了。确认设备已经连接到插槽后，准备好插槽和要配置的设备。

-c *ConnectorType* 指定您正操作的 *Slot* 的 *ConnectorType* 属性。例如，针对 PCI 热拔插插槽的 *ConnectorType* 是 pci。此标志必须和 **-a**、**-i**、**-r** 和 **-r** 标志一起使用。

-i 识别热拔插插槽。插槽的识别依赖于硬件。例如，如果插槽上连接了 LED，发出 **drslot -i** 命令可引起 LED 的闪烁。

-I 在使用 **-a**（添加）、**-r**（卸载）和 **-r**（替换）标志时，指定应该跳过的识别步骤。该标志只有在您确信已经识别正确插槽时方可使用。

-l *DeviceName* 指定 *DeviceName*，它是连接到受管插槽的设备逻辑名。如果没有使用 **-s** 标志，该标志用于 **-i**（识别）、**-r**（卸载）或 **-r**（替换）标志。

-r 在用 **rmdev** 命令、或 SMIT 或等价基于 Web 的系统管理器预先除去设备的配置信息后，为热插拔插槽准备设备卸载。插槽被识别时，会给出它的确认提示信息。如果连接到插槽的是一个可视指示符，它将被关闭。最后，准备插槽上的设备卸载，提示您去确认设备已从插槽上卸载。

-r 在除去设备的配置信息后准备设备卸载，并用相同设备替换。设备必须用 **rmdev** 命令、SMIT 或等效的基于 Web 的系统管理器来取消配置。**drslot** 识别插槽，提示您进行插槽的确认。接着，准备插槽进行设备的替换。您会被提示确认设备已经替换。确认热插拔插槽上的设备替换完成后，插槽和设备的配置就准备完毕了。

-s *Slot* 指定 **drslot** 操作的 *Slot*。进行添加（**-a**）操作时必须使用该标志。如果未使用标志 **-l**，则进行识别（**-i**）、卸载（**-r**）或替换（**-R**）操作时，必须使用该标志。*Slot* 的格式依赖于平台和 *ConnectorType*。

示例

1. 要标识指定 PCI 热插拔插槽，请输入:

   ```
   drslot  -i  -c pci  -s U0.1-P1-I3
   ```

   这个示例中，插槽上有一个 LED 指示灯。系统显示以下类似消息:

   ```
   指定 PCI 插槽的可视指示符已经设置成了识别状态。按 Enter 键继续或输入 x 退出。
   ```

   在输入回车键之前，U0.1-P1-I3 指定的插槽上的 LED 指示灯会一直闪烁。
2. 要添加一个热插拔以太网适配器到一个热插拔插槽，不进行插槽确认，请输入:

   ```
   drslot  -a  -I  -c pci  -s U0.1-P1-I3
   ```

   不会出现识别插槽的提示确认信息。当可以插入新适配器的时候，会给出类似下面的确认提示消息:

   ```
   指定 PCI 插槽的可视指示符已被设置成了操作状态。将 PCI 卡插到指定插槽，连接要配置的设备，然后按回车继续。输入 x 退出。
   ```

   连接适配器后按回车键，插槽准备完毕。
3. 在替换 scsi 卡前，识别其所在的特定 PCI 插槽，请输入:

   ```
   drslot  -R  -c pci  -s U0.2-P1-I3
   ```

   系统显示类似下面的消息:

   ```
   指定 PCI 插槽的可视指示符就被设置成了识别状态。输入回车继续或输入 x 退出。
   ```

   PCI 插槽上的 LED 指示灯不停闪烁以识别插槽。输入除 Enter 键以外的任何键退出命令。按 Enter 键继续。如果继续操作，PCI 插槽的 LED 指示灯转换成操作状态，系统显示类似以下消息:

指定 PCI 插槽的可视指示符就被设置成了操作状态。将 PCI 卡插到指定插槽，重新连接要配置的设备，然后按 Enter 键继续。输入 x 退出。现在退出使 PCI 插槽处于卸载状态。

文件

/usr/sbin/drslot

相关信息

lsslot 命令、**rmdev** 命令、**cfgmgr** 命令。

有关 PCI 适配器的热插拔管理和 PCI 热插拔支持的信息，请参阅《*操作系统与设备管理*》中的『PCI 热插拔管理』 。

---

dscreen 命令

用途

启动动态屏幕实用程序。

语法

dscreen [**-i** *InfoFile*] [**-t** *TermType*]

描述

dscreen 命令启动了“动态屏幕实用程序”，它允许单个物理终端同时连接到多个虚拟会话或屏幕。

如果没有指定的标志，**dscreen** 命令就从 **DSINFO** 环境变量中指定的文件读取在 **TERM** 环境变量中指定的终端的描述。如果没有指定 **DSINFO** 环境变量，则从 **/etc/dsinfo** 文件读取终端描述。终端描述通常包含以下配置信息:

- 用于“动态屏幕实用程序”的键及其功能
- 终端可用的屏幕内存页数
- 必须发送或接收以访问和使用“动态屏幕功能”的代码序列。

标志

-i *InfoFile*
: 指定包含和“动态屏幕实用程序”一起使用的备用键映射的文件。当最初定义的“动态屏幕”键与应用程序之一相冲突时，该选项非常有用。
如果没有指定该标志，就从 **DSINFO** 环境变量（如果设置的话）中指定的文件读取终端配置信息。否则，就从 **/etc/dsinfo** 文件读取信息。

-t*TermType*
: 标识要从包含键映射的文件读取的终端描述。当期望的终端类型与 **TERM** 环境变量的设置不相匹配时，该选项非常有用。

示例

1. 要使用键映射缺省值启动“动态屏幕实用程序”，请输入:

   ```
   dscreen
   ```

   这按照缺省 **/etc/dsinfo** 文件所指定的，设置了 **DSINFO** 和 **TERM** 环境变量。

2. 要启动“动态屏幕实用程序”并指定一个包含备用键映射的文件，同时标识从该文件读取的终端描述，请输入:

   ```
   dscreen -i mygile -t myterm
   ```

   这使用了从用户创建的 **dsinfo-** 类型文件，称为 myinfo 的信息用来处理特殊的键映射需要。myinfo 文件还包含一个称为 myterm 的终端定义。
3. 要启动“动态屏幕实用程序”并指定一个备用终端设置，请输入:

   ```
   dscreen -t wy60-wp
   ```

   该终端定义（保存在 **/etc/dsinfo** 文件中）设置了 **dscreen** 分配键的功能，这样就避免了与正在使用的字处理应用程序的控制键命令序列相冲突。

文件

/etc/dsinfo 包含用于“动态屏幕实用程序”的终端描述。

相关信息

《*网络与通信管理*》一书中的『动态屏幕实用程序』。

---

dslpaccept 命令

用途

为目录启用的 System V 打印系统接受打印队列请求。

语法

dslpaccept *PrintQueueName*

描述

dslpaccept 和 **dslpreject** 命令是用来设置一个打印队列，这样它就可以接受或者拒绝为它等待的打印请求。与 **accept** 和 **reject** 命令不同，只要是目录启用的命令它们就可以控制远程打印系统。这是因为它们在目录服务器上直接写到打印队列对象中。

这个命令的用户必须是目录启用的，并且在他是管理员的目录上下文中，拥有在目录上写、修改、搜索以及读取的许可权设置。

参数

PrinterQueueName *PrintQueueName* 参数为打印队列对象的 RDN。多个打印队列名可以通过一个以逗号隔开的列表指定。

退出状态

0 表明成功。

1 表明无效的选项。

2 表明指定的打印队列是未知的。

3 表明用户不具有修改许可权。

4 表明提供了无效 RDN。

5 表明已经设置了值。

6 表明命令无法连接目录服务。

7 其他任何错误的说明。

示例

1. 设置打印队列 “hpcolor” 来接受请求。

```
dslpaccept hpcolor
```

相关信息

dslpaccess 命令、**dslpadmin** 命令、**dslpdisable** 命令、**dslpenable** 命令、**dslpprotocol** 命令、**dslreject** 命令、**dslpsearch** 命令和 **lpstat** 命令。

---

dslpaccess 命令

用途

允许或者拒绝非目录启用的用户和系统访问对于 System V 打印子系统的打印队列。

语法

dslpaccess -q *QueueName* **-a** *AllowList* | **-d** *DenyList*

描述

dslpaccess 命令可能允许，或拒绝用户和系统访问目录启用的打印队列。这在 **lpadmin** 命令的 **-u** 选项中构成模型。

允许和拒绝由逗号分开的条目列表组成的列表，每一个可能指定一个登录标识或者是指定一个系统名和登录标识，如下所示:

```
[[LoginID]|[System!LoginID]],[[LoginID]|[System!Login-ID]],...
```

LoginID 或者 *System*，或者两者，都可以设置为通配符 **all**，允许或者拒绝所有适当的条目。使用 **all** 要当心。当 **all** 项添加到一个列表，对于 *LoginID* 或者 *System* 的适当值，所有非 **all** 条目都从其他列表中除去。*System* 的缺省值是本地主机。

这个命令的用户必须是目录启用的，并且在他们是管理员的目录上下文中，拥有在目录上写、修改、搜索以及读取的许可权设置。

标志

-a *AllowList* 指定一系列用户，添加到许可列表。如果存在，就将其从拒绝列表中删除。该选项不能和 **-d** 选项一起使用。

-d *DenyList* 指定一系列用户，添加到拒绝列表。如果已经存在，就将其从许可列表中删除。该选项不能和 **-d** 选项一起使用。

-q *QueueName* 该队列名称参数为打印队列的 RDN。如果打印队列名在目录的上下文中不存在，该命令失败。

退出状态

0 表明成功。

1 表明无效的选项。

2 表明指定的打印队列是未知的。

3 表明用户不具有适当的访问控制许可权。

4 表明提供了无效 RDN。

5 表明已经设置了值。

6 表明其他任何错误。

示例

1. 以下授权用户 fredp 在主机 systemX 上访问打印队列 printq1:

   ```
   dslpaccess -q printq1 -a systemX!fredb
   ```

2. 以下的则对所有主机拒绝用户 tomt 访问打印队列 printq1:

   ```
   dslpaccess -q printq1 -d all!tomt
   ```

相关信息

dslpaccept 命令、**dslpadmin** 命令、**dslpdisable** 命令、**dslpenable** 命令、**dslpprotocol** 命令、**dslreject** 命令、**dslpsearch** 命令和 **lpstat** 命令。

---

dslpadmin 命令

用途

为 System V 打印子系统配置目录启用的打印服务。

语法

dslpadmin [[**-q** *PrintQueueName* [**-D** *QueueDescription*] [**-n** *LocalQueueName*] [**-o banner** | **nobanner**] [**-A mail** | **none**] [**-F** *FaultRecovery*] [[**-P** *PhysicalPrinterName*] [**-s** *NetworkEntityName*]]] [**-P** *PhysicalPrinterName* [**-t** *PrinterType*] [**-l** *Location*] [**-L** *PDLList*]] [**-q** *PrintQueueName* **-P** *PhysicalPrinterName* [**-I** *ContentType*] [[**-i** *InterfaceScript*] | [**-m** [**Standard** | **PS**]]] [**-o** *PrintOptions*]] [**-q** *PrintQueueName* [**-I** *ContentType*]]] [**-q** *PrintQueueName* **-s** *NetworkEntityName* [**-a** *PrintSystemDNSName* | *PrinterSystemAddress*] [**-t BSD** | **HPNP**]]

dslpadmin [**-q** *PrintQueueName* [**-u** *PhysicalPrinterName*] [**-U** *objectRDN*]]

dslpadmin [**-x** *PrintQueueName*] [**-X** *PhysicalPrinterName*] [**-r**]

dslpadmin [**-h**]

描述

为了配置目录启用的打印服务，使用 **dslpadmin** 命令来执行以下功能:

- 向系统添加打印队列和具体的打印机。
- 修改打印队列和物理打印机。

- 从系统删除打印队列和物理打印机。
- 为网络打印机添加或者删除网络实体对象。

dslpadmin 命令提供了由 **lpadmin**（非目录明确）提供的功能性的目录明确版本，并且继续使用传统的“平面文件”配置系统。注意在两个系统同时使用的情况下，打印机子系统首先使用在目录中找到的信息。管理员有责任确保在两个配置系统之间不会产生命名冲突。

目录启用的命令使用相对专有名称（RDN）而不是专有名称（DN）。例如，要创建 DN 为“cn=test,ou=printq,ou=print,cn=aixdata”的目录启用的队列，对于 *PrintQueueName* 只要使用 RDN“test”。

当配置管理员不在主管打印队列的系统的打印队列时，就不会检查 **-i** 的 *InterfaceScript* 参数以及 **-t** 的 *PrinterType* 参数。这是因为远程系统为了执行检查而无法被访问。因此管理员有责任确保指定的 *InterfaceScript* 和 *PrinterType* 存在于远程主管系统。

命令行可以是任何 **-q**、**-P** 和 **-s** 标志的组合或者是 **-x**、**-X** 和 **-r** 的组合，但每种标志只能有一个。当同时创建或者修改多个目录对象时，在三个对象类型（打印机、打印队列和网络实体）之间就要设置适当的链接。

标志

-a *PrinterSystemDNSName* | *PrinterSystemAddress*
将一个 DNS 名或者网络地址与系统相关联。如果给定自变量可以解释为 IPv4 或者 Pv6 地址，则它就是一个地址，如果不可以的话，就假定它是一个 DNS 名。标志 -a 导致修改由 **-s** 指定的网络实体对象，如果它还不存在的话，则创建它。管理员应该确保网络实体对象都给定了唯一的名称，这样就可以避免修改现有的 UNIX® 系统对象，而不是避免添加新的打印系统对象。该标志要求 **-s** 标志。

-A [**mail** | **none**]
如果打印请求失败，指示打印系统生成邮件消息。邮件被发送到物理打印机的所有者，如果打印机没有所有者或者用户没有邮件地址，邮件就发送到主管打印队列系统的 root 用户。缺省值是 **none**。该标志需要 **-q** 标志。

-D *QueueDescription*
以 **-q** 标志为指定的打印队列对象定义一个描述注释。无论何时当用户用 **lpstat** 命令请求一个打印队列的完全描述，都会显示该描述。包含空格的字符串应该用双引号括起来。该标志需要 **-q** 标志。

-F *FaultRecovery*
定义打印队列的故障恢复方式。如果打印一个打印请求失败时打印机在打印队列上，该标志指定要使用的恢复。*FaultRecovery* 的值可以是以下任何一个:

继续 在打印停止的页面上方继续打印。在自动继续之前，这需要一个过滤器来等待清除故障。

开始 从开始处再次启动打印请求。

等待 在 *PhysicalPrinterName* 上禁用打印并且等待管理员或者用户再次启用打印。

在等待期间提交停止的打印请求的管理员或者用户可以发出一个更改请求，来指定打印从哪里重新开始。如果在打印请求启用之前没有发出更改请求，并且如果过滤器允许的话，打印就从停止处页码顶部继续进行。否则，就从开头打印请求。

FaultRecovery 的缺省值是**开始**。该标志需要 **-q** 标志。

-h
显示简单的帮助屏幕。

-i *InterfaceScript*
当通过指定的打印队列访问时，打印机的 *InterScript* 路径。如果没有指定 **-P** 标志，该标志是无效的。接口脚本通常是由用户提供。当 **-m** 也被指定时，该标志就无法应用。该标志需要 **-q** 和 **-P** 标志。

-I *ContentType*[, , ...]
指定打印队列的目录类型。允许打印队列通过列表中的目录类型处理打印请求。如果列表中包含多于一个 *ContentType*，则 *ContentType* 参数就必须用逗号隔开。请参阅关于格式完全描述的 **lpadmin** 手册页。这也需要 **-P** 标志和 **-q** 标志。

-l *Location*
定义打印机的位置。该字符串标识打印机物理上位于何处，例如 “X 号楼，6 号房间”。它可以通过命令 **dslpsearch** 搜索。一旦设置，该值就只能被覆盖，而不能被删除。该标志需要 **-P** 标志。

-L *PDL*[, *PDL*, ...]
指定由打印机支持的页面描述语言（PDL）的列表。通过使用 **dslpsearch** 命令，这被用作宣传打印机支持的任何 PDL，也可以被在上面搜索。**AUTOSW**、**PCL**、**PCLXL**、**PostScript**、**TEXT**、**ESCP**、**PJL**、**SIMPLE** 以及 **OTHER** PDL 都得到支持。如果 **-L** 标志是用来修改现有物理打印机对象，该列表就替换现有的列表。该标志需要 **-P** 标志。

-m [**standard** | **PS**]
当通过指定的打印队列访问时，为打印机模拟接口程序。这选择了由打印队列使用的模拟接口脚本。在创建一个具体的打印机对象时，并且 **-m** 和 **-i** 标志都没有指定时，缺省值就是 **standard**。当也指定了 **-i** 时，无法使用该标志。该标志需要 **-q** 和 **-P** 标志。

-n *LocalQueueName*
定义打印队列局部名。此名称通常仅在队列位于非目录启用主机上时，才不同于该队列的 RDN。通过引入远程网络连接，用它来识别接收系统上的打印队列。缺省值为打印队列的 RDN。该标志要求 **-q** 标志。

-o [**banner** | **nobanner**]
定义是否标志页总是由该打印队列产生。尽管 **nobanner** 允许用户提交一份指定没有标志页要打印的打印作业，但是缺省值 **banner** 强制为所有的打印请求打印标志页。该标志需要 **-q** 标志。

-o *PrintOption=Value*[, …]
为打印选项指定值。请参阅 **lpadmin** 文档获得和 **-o** 标志一起可用的打印选项的详细描述。该标志需要 **-q** 和 **-P** 标志。

-P *PhysicalPrinterName*
创建或者修改物理打印机对象。*PhysicalPrinterName* 参数指定打印机对象的 RDN。如果对象尚未存在，**dslpadmin** 就创建它。

-q *PrintQueueName*
dlspadmin 创建或者修改一个打印队列对象。*PrintQueueName* 参数指定打印队列对象的 RDN。在添加新的打印队列时，必须指定 **-s** 和 **-P** 标志这样命令就知道添加的打印队列的 *NetworkEntityName* 和 *PhysicalPrinterName*。如果打印队列对象不存在，**dslpadmin** 就创建它。

命令行可以包含**-q**、**-P** 和 **-s** 标志的任意组合，或者 **-x**、**-X** 和 **-r** 标志的任意组合，但是每个标志只能有一个。当同时创建或者修改多个目录对象时，在三个对象类型（打印机、打印队列和网络实体）之间就要设置适当的链接。

-r *NetworkEntityName*
删除网络实体系统对象。务必小心避免删除非打印机系统对象。管理员有责任确保删除了正确的对象。

-s *NetworkEntityName*
指定主管打印队列的网络实体系统对象。如果也给定了 **-a**，就创建或修改了对象。*NetworkEntityName* 参数指定当前目录上下文中的对象的 RDN。网络实体对象定义了远程客户机访问打印队列时需要使用的网络地址。

-t [**BSD** | **HPNP**]
定义了 “网络打印机” 打印队列使用的打印协议。设置网络打印机重试和超时值为其缺省值。要更改这些值，需要使用 **dslpprotocol** 命令。注意该标志只能用于支持 BSD 和 HPNP 协议的网络打印机。该标志需要 **-q** 标志。

-t *PrinterType*[, *PrinterType*, ...]
打印机类型列表。它识别打印机为一种或者多种类型例如 “hplaserjet”。请参阅 **lpadmin** 手册内容，获取详细信息。该标志需要 **-s** 标志。

-u *PhysicalPrinterName*
从打印队列解除指定的具体打印机的链接（它是由 **-q** 标志指定的），而不删除它的对象。该标志需要 **-q** 标志。

-U *objectRDN*
从打印队列（由 **-q** 指定）解除物理的打印机或者打印队列对象（由 *ObjectRDN* 指定）的链接，而不删除其对象。该标志需要 **-q** 标志。

-x *PrinterQueueName*
删除打印队列对象。

-X *PhysicalPrinterName*
删除物理打印机对象。

退出状态

0　表示成功

255 （或 -1）
表示配置中出错。显示错误消息，说明错误或故障。

示例

以下示例说明了当用户登录到目录启用的 UNIX 系统时 dslpadmin 命令的使用情况。

1. 下面将添加使用 BSD 远程打印协议的 HP LaserJet 网络打印机，其打印队列 RDN 为“denlj5n”，物理打印机 RDN 为“denplj5n”。它为打印队列描述了“HP JetDirect（PostScript）”，打印机类型“PS-b”，以及作为“PS”的典型接口脚本。打印机的网络地址是“p_hplj.ibm.com”：

   ```
   dslpadmin -q denlj5n -P denplj5n -T PS-b -D "HP JetDirect (PostScript)" \
   -I PS -m PS -A mail -o nobanner  -s denslj5n -a p_hplj.ibm.com -t BSD
   ```

   打印系统允许对于这种打印队列的目录类型 PS 的打印请求，并且允许禁用标志页。

2. 下面将添加使用 HPNP 远程打印协议的 HP LaserJet PostScript 网络打印机，其打印队列 RDN 为“dehpnp”，物理打印机 RDN 为“dephpnp”。它为打印队列描述了“HPNP（PCL）”，打印机类型“hplaserjet”，以及作为“标准”的模型接口脚本。打印机的网络地址是“p_hplj.ibm.com”。

   ```
   dslpadmin -q dehpnp -P dephpnp -T hplaserjet -D "HPNP (PCL)" -I pcl \
   -m standard -A mail -s deshpnp -a p_hplj.ibm.com -t HPNP
   ```

   打印系统允许对于该打印队列的目录类型 PCL 的打印请求，并且拒绝没有请求标志页的请求。如果打印机出现故障，打印系统就会通知打印机的所有者。

3. 以下删除了 HP LaserJet PostScript 打印机:

   ```
   dslpadmin -x delj5n  -X deplj5n
   ```

4. 以下删除了 HPNP 打印机:

   ```
   dslpadmin -x dehpnp  -X dephpnp -r deshpnp
   ```

相关信息

cancel 命令，**dslpaccept** 命令，**dslpaccess** 命令，**dslpenable** 命令，**dslpprotocol** 命令，**dslpsearch** 命令，**lp** 命令，**lpadmin** 命令，**lpstat** 命令。

---

dslpdisable 命令

用途

为 System V 打印子系统禁用打印队列请求。

语法

dslpdisable [**-r** *Reason*] *PrintQueueName*

描述

使用 **dslpenable** 和 **dslpdisable** 命令来启用或者禁用一个来自于已经排队等待的处理打印请求的打印队列。与 **enable** 和 **disable** 命令不同，只要是目录启用的命令它们就可以控制远程打印系统。这是因为它们直接写到目录服务器上的打印队列对象。

标志

-r *Reason*　　指明禁用打印队列的原因。包含空格的字符串应该用双引号括起来。

Reason 是由 lpstat 命令显示的一个字符串。如果没有指定，就不会设置缺省原因。

参数

PrinterQueueName — *PrintQueueName* 参数为打印队列的 RDN。这可以是打印队列的一个列表。如果打印队列名在目录的上下文中不存在，该命令就失败。

退出状态

0 表明成功。

1 表明无效的选项。

2 表明指定的打印队列是未知的。

3 表明用户不具有修改许可权。

4 表明提供了无效 RDN。

5 表明值已经设置。

6 表明命令无法联系目录服务。

7 表明其他任何错误。

示例

要禁用指定原因为“例行维护”的打印队列“printer1”，输入如下内容:

```
dslpdisable -r "routine maintenance" printer1
```

相关信息

dslpaccept 命令、**dslpaccess** 命令、**dslpadmin** 命令、**dslpenable** 命令、**dslpprotocol** 命令、**dslreject** 命令、**dslpsearch** 命令、**lpstat** 命令。

---

dslpenable 命令

用途

为 System V 打印子系统启用打印队列请求。

语法

dslpenable *PrinterQueueName*

描述

使用 **dslpenable** 和 **dslpdisable** 命令启用或者禁用一个来自于已经排队等待的处理打印请求的打印队列。与 **enable** 和 **disable** 命令不同，只要是目录启用的命令它们就可以控制远程打印系统。这是因为它们直接写到目录服务器上的打印队列对象。

参数

PrinterQueueName — *PrintQueueName* 参数为打印队列的 RDN。这可以是打印队列的一个列表。如果打印队列名在目录的上下文中不存在，该命令就失败。

子命令

退出状态

0 表明成功。

1 表明无效的选项。

2 表明指定的打印队列是未知的。

3 表明用户不具有修改许可权。

4 表明提供了无效 RDN。

5 表明值已经设置。

6 表明命令无法联系目录服务。

7 表明其他任何错误。

示例

1. 要启用打印队列 “hpcolor”，输入如下内容:

```
dslpenable hpcolor
```

相关信息

dslpaccept 命令、**dslpaccess** 命令、**dslpadmin** 命令、**dslpdisable** 命令、**dslpprotocol** 命令、**dslreject** 命令、**dslpsearch** 命令、**lpstat** 命令。

---

dslpprotocol 命令

用途

为 System V 打印子系统配置打印队列的远程打印协议。

语法

dslpprotocol -t *RemoteProtocol* [**-t** *Timeout*] [**-r** *Retry*] [**-r**] *PrinteQueueName*

dslpprotocol -l [**-S**] *PrintQueueName*

描述

dslpprotocol 命令是用于配置 “远程打印协议”，当向打印队列发送打印请求时，远程打印客户机可以使用该协议。

在目录启用的打印中，要打印一个远程打印队列，客户机必须首先获取它可以使用的远程打印协议。从目录中打印队列对象可以找到它。这可以是 BSD 和 HPNP 中的一个，也可以两个都是。在为打印队列配置了多个协议时，UNIX 打印系统使用读取的第一个值，因此一个队列通常只配置一个协议。

PrintQueueName 参数是打印队列的相对专有名称（RDN）。如果分派到 *PrintQueueName* 的值不存在，则命令失败。

这个命令的用户必须是目录启用的，并且在他们是管理员的目录上下文中，拥有在目录上写、修改、搜索以及读取的许可权设置。

标志

-l 打印出与打印队列相关的远程打印协议参数的描述。

-t *RemoteProtocol* 指定在向该打印队列发送打印请求时可用的远程打印协议。支持的协议类型值是 **bsd** 和 **hpnp**。缺省值是 **bsd**。

-t *TimeOut* 为指定的协议设置网络连接超时值，即断开连接之前网络连接在空闲状态应该保持的时间。为了禁止超时，也可以指定这个值 n。值 0 使得连接只要一空闲，立刻断掉。缺省值是 10 分钟，没有实际的上限。请参阅关于 **-t** 选项完全定义的 **lpsystem** 手册页。

-r 该选项用来从打印队列对象中删除指定的协议。该选项要求指定 **-t** 选项。

-r *Retry* 为指定协议设置网络连接重试时间，即网络在连接失败后试图重新建立前的等待时间（以分钟计）。缺省值是 2 分钟。该值如果是 0，意味着连接立即重试。注意，该值必须短于通过 **-t** 选项指定的超时值。当没有工作可用时，为了避免断开的连接重试也可以指定该值为 n。该值没有实际上限。对于“网络打印机”，重试时间应该设为 0。请参阅关于 **-r** 选项完全定义的 **lpsystem** 手册页。

-S 同 **-l** 选项一起使用，通过简单的格式来显示打印队列的协议设置。

参数

PrinterQueueName *PrintQueueName* 参数是打印队列的相对专有名称（RDN）。如果分派到 *PrintQueueName* 的值不存在，则命令失败。

退出状态

0 表明成功。

1 表明无效的选项。

2 表明指定的打印队列是未知的。

3 表明用户不具有修改许可权。

4 表明提供了无效 RDN。

5 表明值已经设置。

6 表明其他任何错误。

示例

1. 要设置打印队列“printq1”，以允许 BSD 远程打印协议，请输入如下内容:

   ```
   dslpprotocol -t BSD printq1
   ```

2. 要从打印队列“hpcolor”删除 BSD 协议，请输入如下内容:

   ```
   dslpprotocol -r -t BSD hpcolor
   ```

相关信息

dslpaccept 命令、**dslpaccess** 命令、**dslpadmin** 命令、**dslpenable** 命令、**dslpdisable** 命令、**dslpreject** 命令、**dslpsearch** 命令、**lpsystem** 命令。

dslpreject 命令

用途

为目录启用的 System V 打印系统拒绝打印队列请求。

语法

dslpreject [**-r** *Reason*] *PrinteQueueName*

描述

dslpaccept 和 **dslpreject** 命令是用来设置一个打印队列，以便接受或者拒绝为它排队的打印请求。与 **accept** 和 **reject** 命令不同，只要是目录启用的命令它们就可以控制远程打印系统。这是因为它们直接写到目录服务器上的打印队列对象。已经等待的打印请求不受 **dslpreject** 命令的影响。

这个命令的用户必须是目录启用的，并且在他们是管理员的目录上下文下，拥有在目录上写、修改、搜索以及读取的许可权设置。

标志

-r *Reason*　　为拒绝指派一个原因。包含空格的字符串应该在双引号之间。*Reason* 是由 **lpstat** 命令显示的一个字符串。如果没有指定，不会设置缺省原因。

参数

PrinterQueueName　　*PrintQueueName* 参数为打印队列对象的 RDN。多打印队列名可以通过一个以逗号隔开的列表指定。

退出状态

0　表明成功。

1　表明无效的选项。

2　表明指定的打印队列是未知的。

3　表明用户不具有修改许可权。

4　表明提供了无效 RDN。

5　表明值已经设置。

6　表明命令无法联系目录服务。

7　表明其他任何错误。

示例

1. 要设置一个打印队列来拒绝请求，以及指定原因是没有墨，输入如下内容:

```
dslpreject -r "no toner" printer1
```

相关信息

dslpaccept 命令、**dslpaccess** 命令、**dslpadmin** 命令、**dslpdisable** 命令、**dslpenable** 命令、**dslpprotocol** 命令、**dslpsearch** 命令、**lpstat** 命令。

---

dslpsearch 命令

用途

在 System V 打印子系统上为打印系统对象搜索目录。

语法

dslpsearch [**-q** [**-p**]] | [**-P**] [**-o** *SearchOption*]

描述

dslpsearch 命令允许用户和管理员为了打印系统对象搜索目录。例如，用户可以搜索任何能打印彩色 PostScript 文件的打印机。该命令的主要用途是搜索可以匹配搜索字符串的打印队列。

dslpsearch 命令返回匹配搜索字符串的任何对象的专有名称（DN）。然而，在其他目录启用的命令中，要求使用相对专有名称（RDN）。例如，如果 **dslpsearch** 命令返回 DN “cn=testqueue,ou=printq,ou=print,cn=aixdata”，则只有 RDN “testqueue"” 用来引用打印队列。

标志

-q 搜索匹配搜索选项的打印队列。搜索是在物理打印机对象上进行的，但是显示服务于那些打印机的打印队列。这是缺省搜索类型。**-q** 选项不能用 **-P** 指定。

-p 该选项与 **-q** 选项一起使用导致也显示出正服务于打印队列的一系列物理打印机。

-P 搜索匹配搜索字符串的物理打印机。**-P** 选项不能用 **-q** 指定。

-o *SearchOption* 多个搜索选项可以形成一个通过逗号隔开的列表。每个选项可以由以下内容构造:

- 一个或多个如下的页面描述语言（PDL）: **AUTOSW**、**PCL**、**PCLXL**、**POSTSCRIPT**、**TEXT**、**ESCP**、**PJL**、**SIMPLE**、**OTHER**
- 任何如下的打印机设施: **COLOR**、**DEPLEX**、**TRAYS**、**FINISH**
- 一个和多个物理打印机的位置由 `location=xxxxxxxx` 或者 `location='aaaa bbbbb'` 指定
- 由 location= 定义的字符串值通过放置于字符串两头的通配符搜索，这样 `location=Room1` 就可以搜索到在其位置上有 “Room1” 的任何打印机，例如 “Building X, Room1, Bay6”。字符串值内也可以有通配符（*），例如 `location=“Building X* Bay6”`。在搜索中多位置值被 OR’d。
- 以下是包含搜索字符串的有效命令行:

```
dslpsearch -q -o PCL,ESCP,location=room2,COLOR

dslpsearch -q -p -o "PS, location='Building 1, Room1', DUPLEX"
```

退出状态

0 表明成功。

1 表明无效的选项。

2 表明目录树上的搜索失败。

3 表明无效的目录上下文。

4 表明命令无法联系目录服务。

示例

1. 下面的命令行搜索匹配搜索选项的任何打印队列:

   ```
   dslpsearch -q -o search-options
   ```

2. 搜索匹配搜索选项的任何具体打印机。

   ```
   dslpsearch -P -o search-option
   ```

相关信息

dslpaccept 命令、**dslpaccess** 命令、**dslpadmin** 命令、**dslpdisable** 命令、**dslpenable** 命令、**dslpprotocol** 命令、**dslreject** 命令、**lpstat** 命令。

---

dspcat 命令

用途

显示消息编目的全部或者部分。

语法

在编目中显示消息

dspcat*CatalogName* [*SetNumber* [*MessageNumber*]]

为 gencat 命令格式化输出

dspcat -g *CatalogName* [*SetNumber*]

描述

dspcat 命令显示特定的消息，在一个设置中的所有消息，或者在一个编目中的所有消息。**dspcat** 命令引导消息到标准输出。

注: 如果在 C 或者 POSIX 语言环境中将 **LC_FASTMSG** 设置成 False，则 **dspcat** 命令就在 **NLSPATH** 下寻找目录文件。

LC__FASTMSG 指定用于 C 和 POSIX 语言环境的缺省消息，以及当 **LC__FASTMSG** 设置为 Ture 时，忽略 **NLSPATH**。

在 **/etc/environment** 中， **LC__FASTMSG** 的缺省值将为 Ture。

CatalogName 参数指定了一个消息编目。*SetNumber* 参数在由 *CatalogName* 参数指定的目录中，指定了一个设置。*MessageNumber* 参数在由 *SetNumber* 参数指定的设置中，指定了一个特定消息。如果包含了所有三个参数，**dspcat** 命令就显示指定的消息。如果不包含 *MessageNumber* 参数，**dspcat** 命令就显示设置中的所有消息。如果给 *SetNumber* 或者 *MessageNumber* 参数指定了一个不存在的值，**dspcat** 命令就显示一个错误消息，并且返回一个非零返回值。如果只是指定 *CatalogName* 参数，**dspcat** 命令就显示目录中的所有消息。如果包含了 *MessageNumber* 参数，就必须将 *SetNumber* 参数包含进去。

如果在 *CatalogName* 参数的值中，没有使用 /（斜杠）字符，则 **dspcat** 命令就通过 **NLSPATH** 环境变量和 **LC_MESSAGES** 目录查找指定的消息编目。

标志

-g 格式化输出被用作 **gencat** 命令的输入。在使用 **-g** 标志时，*MessageNumber* 参数无效。

示例

要在 test.cat 文件的设置数 1 中显示消息数 2，请输入:

```
dspcat test.cat 1 2
```

文件

/usr/bin/dspcat 包含 **dspcat** 命令。

相关信息

dspmsg 命令、**gencat** 命令、**mkcatdefs** 命令、**runcat** 命令。

catclose 子例程、**catgets** 子例程和 **catopen** 子例程。

关于“消息工具”的更多信息，请参阅《*AIX 5L V5.3 本地语言支持指南和参考大全*》中的『消息设施』。

---

dspmsg 命令

用途

从消息编目中显示一条选定的消息。

语法

dspmsg [**-s** *SetNumber*] *CatalogNameMessageNumber* ['DefaultMessage' [*Arguments*]]

描述

dspmsg 命令显示以 **gencat** 命令生成的消息编目中的一个特殊消息文本，或者作为命令参数提供的缺省消息（如果无法检索到该消息的话）。**dspmsg** 命令将消息指向标准输出。该命令意在 shell 脚本中使用，以作为 **echo** 命令的替换。

注: 如果在 C 或者 POSIX 语言环境中将 **LC_FASTMSG** 设置成 False，则 **dspmsg** 命令就在 **NLSPATH** 下寻找目录文件。

LC__FASTMSG 指定用于 C 和 POSIX 语言环境的缺省消息，并且当 **LC__FASTMSG** 设置为 True 时，指定忽略 **NLSPATH**。

在 **/etc/environment** 中， **LC__FASTMSG** 的缺省值为 True。

如果在 *CatalogName* 参数的值中，没有使用 /（斜杠）字符，则使用 **NLSPATH** 环境变量和 **LC_MESSAGES** 目录查找指定的消息编目。如果没找到由 *CatalogName* 参数指定的目录，或者没有找到 *MessageNumber* 参数（以及可选的 *SetNumber* 值）指定的消息，则显示提供的 *DefaultMessage* 值。如果没有指定 *DefaultMessage* 值，就会显示一个系统产生的错误消息。

如果 **dspmsg** 命令包含 **%s**、%n$s、**%ld**，或者 %n$ld **printf** 子例程转换规范，它就允许最多十个字符串变量代入消息中。丢失的用于转换规范的自变量会导致 **dspmsg** 错误消息。识别正常 **printf** 子例程控制字符转义（例如，**\n**）。

在目录中，推荐使用 **printf** 子例程格式字符串。即使在这个消息中的格式字符串与缺省消息的顺序不同，该格式也支持自变量的正确插入。如果通过 **%n$s** 符号插入消息，您必须用单引号将缺省消息括起来。

标志

-s *SetNumber*　　指定一个可选的设置数。*SetNumber* 变量的缺省值是 1。

示例

要显示 `test.cat` 目录的设置数 1 和消息数 2，请输入:

```
dspmsg  -s  1  test.cat  2  'message  %s  not  found'  2
```

如果没有找到消息，则显示 `message 2 not found`。

文件

/usr/bin/dspmsg　　包含 **dspmsg** 命令。

相关信息

dspcat 命令、**gencat** 命令、**mkcatdefs** 命令、**runcat** 命令。

catclose 子例程、**catgets** 子例程和 **catopen** 子例程。

关于“消息工具”的更多信息，请参阅《*AIX 5L V5.3 本地语言支持指南和参考*大全》中的『消息设施』。

---

dtaction 命令

用途

以指定参数调用 CDE 操作。

语法

dtaction [**-contextDir** *context_dir*] [**-execHost** *host_name*] [**-termOpts** *terminal_arguments*] [**-user** *user_name*] *action_name* [*action_arg*] ...

描述

dtaction 命令允许在其他情况下未连接到 CDE 开发环境中的应用程序或 shell 脚本调用操作请求。名为 *action_name* 的操作是使用命令行上提供的 *action_arg* 调用的。一个 *action_name* 是必需的；用户可提供任意

数量的 *action_arg*。对 *action_name* 和 *action_arg* 的解释取决于操作数据库中该操作的定义。操作可能定义在某个系统操作数据库文件中，或某个用户专用操作数据库文件中。

action_arg 为文件的绝对或相对路径名。**dtaction** 命令将此文件列表传递给指定的操作。

当检测到以下情况时，将发布错误对话:

- 桌面环境无法初始化
- 用户或密码无效
- 无法将标识更改为所请求的用户
- 未指定操作名称

标志

contextDir *context_dir*
指定缺省目录上下文（如果 *action_name* 的定义未定义命令操作的当前工作目录）。

execHost *host_name*
指定命令操作的替代执行主机 *host_name*。如果该操作不是命令操作，则 **dtaction** 命令忽略此选项。将在 *host_name* 上尝试该操作，而不是在该操作的 EXEC_HOST 所指定的主机上。如果在任何适用主机上无法调用指定的操作，则会发布错误。

termOpts *terminal_arguments*
指定参数，这些参数用于为非 NO_STDIO 类型的命令操作提供的终端仿真器。如果在 *terminal_arguments* 字符串中有空格字符，则该字符串必须括在引号内，以防 shell 将其错误解释。这些参数按原样传递给终端仿真器，因此用户必须确保这些字符串是合理的。尤其是，*terminal_arguments* 不允许用于指定命令的参数在终端仿真器窗口中运行（即：使用 **dtterm1** 时带 **-e** 标志）。

user *user_name*
指定用户名。如果 **dtaction** 当前并不是使用该用户身份运行的，则显示的提示对话框将用于收集指定的用户密码或 root 用户密码。输入有效密码之后，**dtaction** 命令改为使用所请求用户的身份运行，然后启动所请求的操作。

参数

action_name
指定要调用的操作的名称。

action_arg
指定文件的绝对或相对文件名。

环境变量

DTDATABASESEARCHPATH
逗号分隔的目录列表（带有可选的 host: 前缀），它告诉操作服务在何处查找操作数据库。

退出状态

将返回以下退出值:

0
成功完成。

>0
发生错误。

安全性

dtaction 命令是由 PAM 启用的服务名为 **dtaction** 的应用程序。如果 **user** *user_name* 选项指定的用户名不同于登录用户名，则 **dtaction** 命令将在调用指定操作前先认证用户。它可执行 PAM 认证，也可执行传统认证。

要在系统范围内使用 PAM 进行认证，请建立 root 用户许可权，然后在 **/etc/security/login.cfg** 文件的 **usw** 节中，将 *auth_type* 属性的值修改为 PAM_AUTH。

启用 PAM 时使用的认证机制取决于 **/etc/pam.conf** 中的登录服务的配置。**dtaction** 命令需要对应于 **auth** 模块类型的 **/etc/pam.conf** 条目。以下配置是 **/etc/pam.conf** 中针对 **dtaction** 服务的建议配置:

```
dtaction        auth            required        /usr/lib/security/pam_aix
```

示例

1. 要调用操作，请输入:

   ```
   dtaction Xterm
   ```

   这将启动 X Windows 终端仿真器（Xterm）。
2. 要在远程主机上调用操作，请输入:

   ```
   dtaction -execHost hostname Xterm
   ```

   这将在指定远程主机上执行 Xterm。
3. 要使用另一个用户的身份调用操作，请输入:

   ```
   dtaction -user username Xterm
   ```

   这将以指定用户的身份执行 Xterm。

位置

/usr/dt/bin/dtaction

标准错误

dtaction 命令将诊断错误消息写入标准错误，标准错误重定向至 **$HOME/.dt/errorlog**。

文件

/etc/pam.conf	确定 PAM 认证机制。
/etc/security/login.cfg	确定系统范围 PAM 认证。

相关信息

第 204 页的『dtlogin 命令』和第 227 页的『dtsession 命令』。

---

dtappintegrate 命令

用途

公共桌面环境应用程序集成工具。

语法

dtappintegrate -s *ApplicationRoot* [**-t** *TargetPath*] [**-l** *Language*] [**-u**]

描述

dtappintegrate 命令从专用位置链接应用程序 CDE 配置文件到系统位置，并且为受影响的语言更新系统浏览器帮助卷。在应用程序的安装过程期间使用 **dtappintegrate** 命令。应用程序安装脚本应该在末尾调用 **dtappintegrate** 命令。

在应用程序根目录下有四个由 CDE 策略规定的关键子目录（请参考 **$APP_ROOT**）。这些目录有:

$APP_ROOT/dt/appconfig/types/*Language*
对于文件类型、前面板和操作文件。

$APP_ROOT/dt/appconfig/appmanager/*Language*
对于应用程序组文件。

$APP_ROOT/dt/appconfig/icons/*Language*
对于 CDE 管理器使用的图标

$APP_ROOT/dt/appconfig/help/*Language*
对于应用程序帮助。例如，默认语言应用程序 **SpreadSheet** 将在 **/opt/SpreadSheet/dt/appconfig/icons/C/*.bm** 和 **/opt/SpreadSheet/dt/appconfig/icons/C/*.pm** 下装入桌面图标，其中 **/opt/SpreadSheet** 是 **$APP_ROOT** 的值。

注: **$APP_ROOT** 是这个文档的一个语法约定，并且运行时环境没有使用它。所有这些 CDE 配置文件和子目录都放在公共上边框的下面，并且应该始终包含缺省语言子目录 **C**。

在最简单的情况下，命令作为输入应用程序根目录，例如 **/opt/thisapp**。从该操作的输出是应用程序服务器上的相应子目录和文件，在下面的系统位置上，它们包含如上描述的应用程序 CDE 配置文件的相关符号链接:

/etc/dt/appconfig
顶级应用程序配置子目录是由以下子目录组成的:

/etc/dt/appconfig/types/*语言***/**
包含 ***.dt** 以及任何 ***.fp** 链接。

/etc/dt/appconfig/appmanager/*语言***/**
在应用程序管理器下，包含到应用程序组子目录的链接，以及到以操作出现的操作脚本文件的链接。

/etc/dt/appconfig/help/*语言***/**
包含到安装在应用程序根目录下帮助文件的符号链接。

/etc/dt/appconfig/icons/*语言***/**
包含到应用程序的 CDE 图标的符号链接。

标志

-s *ApplicationRoot*
集成位于 *ApplicationRoot* 上的应用程序。该标志是必需的。

-t *TargetPath*
从特定应用程序位置的应用程序 CDE 配置文件链接到 *TargetPath*，而不是链接到系统位置。该标志是可选的。

如果提供了 **-t** 标志，文件在指定子目录下链接。例如，指定 **-t /etc/dt/private** 将会导致在 **/etc/dt/private/help/***Language* 下符号地链接应用程序帮助文件。通常只有想创建独立应用程序的管理员才会使用该标志，在应用程序安装后脚本中并不使用。通过缺省值（没有指定 **-t**），应用程序子目录的根目录对应用程序主机是全局的。安装在主机上的所有应用程序将它们的配置文件复制到合并其他应用程序配置文件的相同地方。

-l *Language*
指定要集成的语言。基本上，该标志指明了在哪些目录里可以查找应用程序 CDE 的配置文件。如果没有指定该参数，所有的语言都将集成。该参数是可选的。

-u
取消应用程序集成。该标志是可选的。

dtlogin 命令

用途

执行 CDE 登录服务。

语法

dtlogin [-**config** *configuration_file*] [-**daemon**] [-**debug** *debug_level*] [-**error** *error_log_file*] [-**nodaemon**] [-**resources** *resource_file*] [-**server** *server_entry*] [-**session** *session_program*] [-**udpPort** *port_number*]

描述

dtlogin 命令支持以下关键任务:

- 对显式管理的本地和远程显示以及 XDMCP 管理的远程显示启动 **dtgreet** 登录屏幕。
- 从 GUI 登录屏幕访问传统终端（字符）登录
- 对依赖于系统的用户执行认证和登录
- 启动所选会话

dtlogin 命令提供的服务类似于字符终端上由 **init**、**getty** 和 **login** 提供的服务，包括提示输入登录名和密码、对用户进行认证以及运行会话。*会话*由特定进程的生存期定义。在传统的基于字符的终端领域中，会话是用户的登录 shell 程序进程；而在 DT 环境中，它是 DT 会话管理器。如果不使用 DT 会话管理器，则典型的替代是带退出选项的窗口管理器，或者是运行 shell 的终端仿真器（终端仿真器的生存期就是它所运行的 shell 进程的生存期）。这将把 X 会话简化为对基于字符的终端会话进行仿真。当会话终止时，**dtlogin** 将复位 X 服务器，并（可选）重新启动整个进程。

dtlogin 命令支持使用 X 显示管理器控制协议 V1.0（XDMCP）管理远程显示。当 **dtlogin** 收到来自 XDMCP 的间接查询时，它可运行选择器进程以代表显示执行 XDMCP BroadcastQuery（或对指定主机执行 XDMCP Query），并提供可能提供 XDMCP 显示管理的主机的菜单。此功能对于不提供主机菜单的 X 终端很有用。

由于 **dtlogin** 提供了用户可看到的第一个界面，因此它的设计目的是易于使用且可根据特定站点的需求方便地进行定制。

登录窗口

登录窗口允许用户输入用户标识和密码、选择启动会话以及选择启动语言环境。用户还可复位 X 服务器，或者临时暂挂 X 服务器以访问字符登录提示。

Login 窗口的内容如下:

登录字段
提供用户可用来输入其标识的输入字段。

密码字段
提供用户可用来输入其密码的输入字段（无回显）。

OK 按钮
对用户进行认证并启动会话。

Clear 按钮
清空登录和密码字段。

Options
让用户选择语言环境名称和登录会话类型。它还允许用户重新启动 X 服务器或切换到字符登录提示（针对本地显示）。Options 菜单的内容如下:

Languages
显示 Languages 菜单。从登录屏幕 Options 菜单中选择语言将使登录屏幕立即本地化，并为接下来的会话设置 **LANG** 变量。登录屏幕本地化和 **LANG** 在会话结束后将恢复为缺省值。此菜单的内容可能因系统上安装的语言环境而异。这可通过使用 **languageList** 资源进行覆盖。使用语言资源可覆盖 C 的缺省语言环境。系统语言环境或指定的 **languageList** 语言环境作为菜单项显示在 Languages 菜单中。使用 **languageName** 资源可为给定的语言环境名称指定要显示的替代文本。

无窗口 显示字符登录提示（仅限本地显示）。

重新装入登录
重新启动 X 服务器并返回登录屏幕。

资源 列出要使用的资源。

会话 显示 Sessions 菜单。允许用户选择登录时应启动的会话类型。菜单项包含以下各项:

DT Session
启动常规桌面会话（Xsession）。

Fail-safe Session
启动故障防护会话（Xfailsafe）。

帮助 显示帮助消息。

控制服务器

dtlogin 命令使用 POSIX 信号控制本地服务器。SIGHUP 信号应复位服务器、关闭所有客户机连接，并执行其他清除工作。SIGTERM 信号应终止服务器。如果这些信号未执行应执行的操作，则 **resetSignal** 和 **termSignal** 资源可指定替代信号。

为了控制不使用 XDMCP 的远程服务器，**dtlogin** 搜索显示上的窗口层次结构，并使用 KillClient X 协议请求来尝试清除终端以进行下一会话。这可能不会实际杀死所有客户机，因为只有那些创建了窗口的客户机才会被注意到。XDMCP 提供了更加保险的机制；当 **dtlogin** 关闭其初始连接时，会话结束并要求终端关闭所有其他连接。

控制 dtlogin

dtlogin 命令响应两种信号: SIGHUP 和 SIGTERM。当收到 SIGHUP 时，**dtlogin** 将重新读取配置文件以及服务器资源指定的文件，并确定是否添加或除去了条目。如果添加了新条目，则 **dtlogin** 将在关联的显示上启动会话。除去的条目将立即被禁用，这意味着正在进行的任何会话都将终止而不会通知，并且不会启动新的会话。如果收到的是 SIGTERM，则 **dtlogin** 终止正在进行的所有会话并退出。这可在关闭系统时使用。

国际化

所有标签和消息都可本地化。**dtlogin.cat** 消息目录包含缺省标签和消息的本地化表示形式。**dtlogin** 命令读取 **LANG** 环境变量所指定的相应消息目录，并显示本地化字符串。认证屏幕上有一个选项允许用户对后续会话覆

盖缺省语言。如果认证屏幕已针对所选语言进行了本地化，则屏幕将以该语言重新显示；否则，它将以缺省语言显示。两种情况下，都会对结果会话相应地设置 **LANG** 环境变量。

资源 language 在 dtlogin 配置文件中提供，用于更改显示的缺省语言。**languageList** 资源在 **dtlogin** 配置文件中提供，用于覆盖认证屏幕上显示的缺省语言集。**languageName** 资源用于提供从语言环境名称到“语言”菜单上所显示文本的映射。

认证和审计

dtlogin 命令是由 PAM 启用的、服务名为 **dtlogin** 的登录服务。**dtlogin** 客户机除了支持传统的本地 UNIX 登录和审计外，还支持 PAM 认证。各供应商可添加其他认证或审计功能，如 Kerberos 或 B1。

要将 PAM 用于系统范围的认证，请建立 root 用户许可权，然后在 **/etc/security/login.cfg** 文件的 **usw** 节中，将 *auth_type* 属性的值修改为 PAM_AUTH。

启用 PAM 时使用的认证机制取决于 **/etc/pam.conf** 中的登录服务的配置。**dtlogin** 命令需要对应于 **auth**、**account**、**password** 和 **session** 模块类型的 **/etc/pam.conf** 条目。在 **/etc/pam.conf** 中，**dtlogin** 服务的建议配置如下:

```
dtlogin        auth              required       /usr/lib/security/pam_aix
dtlogin        account           required       /usr/lib/security/pam_aix
dtlogin        password          required       /usr/lib/security/pam_aix
dtlogin        session           required       /usr/lib/security/pam_aix
```

X 服务器安全性

X 服务器提供了基于用户和基于主机的访问控制。缺省情况下，**dtlogin** 对 X 服务器使用基于用户的访问控制（MIT-MAGIC-COOKIE-1）。该级别的安全性允许根据每个用户实施访问控制。它所基于的方案是：如果客户机传送的授权数据与服务器所拥有的授权数据匹配，则允许该客户机访问。当用户登录时，这些授权数据缺省情况下存储在 **$HOME/.Xauthority** 文件中，并受到保护。

但是，使用基于主机的访问控制机制可能更适合于含有非安全网络的环境，因为基于用户的访问控制允许任何主机进行连接（只要该主机发现了专用密钥）。基于用户的访问控制的另一个缺陷是 R2 或 R3 客户机无法连接服务器。

authorize 资源用于控制 **dtlogin** 使用的是基于用户的访问控制还是基于主机的访问控制。有关更多信息，请参阅 xhost 和 xauth 命令。

资源

dtlogin 命令受 **dtlogin** 配置文件内容的控制，该文件缺省为 **/usr/dt/config/Xconfig**。某些资源控制 **dtlogin** 的常规行为，并且可以针对特定显示指定其他资源。

常规资源

以下 **dtlogin** 常规资源并不特定于显示，并且在适当情况下适用于所有显示。

accessFile

类: AccessFile

类类型:
String

缺省值:
Null

描述: 为防止未经授权的 XDMCP 服务，并允许转发 XDMCP IndirectQuery 请求，此文件包含一个主机名数据库，这些主机或者经过允许可直接访问此机器，或者含有查询应转发到的主机的列表。请参阅 Xaccess 文件部分以了解格式的描述。如果未设置此资源，则对所有主机允许 XDMCP 服务。

authDir

类: AuthDir

类类型:
String

缺省值:
/var/dt

描述: **dtlogin** 用来为使用 XDMCP 的显示临时存储授权文件的目录名。

autoRescan

类: AutoRescan

类类型:
Boolean

缺省值:
True

描述: 控制 **dtlogin** 在会话终止并且配置文件和服务器文件更改之后是否重新扫描这两个文件。可通过向主进程发送 SIGHUP 信号来强制 **dtlogin** 重新读取这些文件。

daemonMode

类: DaemonMode

类类型:
Boolean

缺省值:
False

描述: **dtlogin** 命令可使自己成为无关联的守护进程。这是通过派生并使父进程退出，然后关闭文件描述符并释放控制终端来完成的。这在尝试调试 **dtlogin** 时非常不便。将此资源设置为 False 将禁用 **daemonMode**。

debugLevel

类: DebugLevel

类类型:
Int

缺省值:
0

描述: 对此整数资源指定非零值将启用调试信息打印。它还会禁用守护程序方式，这将把信息重定向到通常无用的位存储区中。

errorLogFile

类: ErrorLogFile

类类型:

String

缺省值:

NULL

描述: 错误输出通常定向至系统控制台上。要重定向错误输出，请将此资源设置为任意文件名。此文件将包含由 **Xsetup**、**Xstartup** 和 **Xreset** 定向到 **stderr** 的任何输出。

errorLogSize

类: errorLogSize

类类型:

Int

缺省值:

50

描述: 此资源指定错误日志文件的最大大小（单位：千字节）。当达到限制时，**dtlogin** 删除该文件中最早的条目，直至文件大小减小到最大值的 75%。文件截短后，为了看到记入错误日志文件中的后续信息，正在访问错误日志文件（例如使用 cat 或 tail）的任何用户需要关闭该文件，然后再重新打开以进行访问。

exportList

类: ExportList

类类型:

String

缺省值:

NULL

描述: 包含一组由空格或制表符分隔的变量名。每个指定的变量是从 **dtlogin** 环境中获取的，并装入服务器和会话的环境中。有关详细信息，请参阅环境部分。

fontPathHead

类: FontPathHead

类类型:

String

缺省值:

NULL

描述: 附加在缺省 X 服务器字体路径前面的值。

fontPathTail

类: fontPathTail

类类型:

String

缺省值:

NULL

描述: 附加在缺省 X 服务器字体路径后面的值。

keyFile

类: KeyFile

类类型:

String

缺省值:

/usr/dt/config/Xkeys

描述: XDM-AUTHENTICATION-1 样式的 XDMCP 认证要求专用密钥在 **dtlogin** 和终端之间共享。此资源指定了包含这些值的文件。该文件中的每个条目由显示名和共享密钥所组成。缺省情况下，**dtlogin** 不包括对 XDM-AUTHENTICATION-1 的支持，因为它需要 DES，后者通常不可分发。

lockPidFile

类: LockPidFile

类类型:

Boolean

缺省值:

True

描述: 控制 **dtlogin** 是否使用文件锁定来防止 **dtlogin** 的多个实例同时执行。

networkDevice

类: NetworkDevice

类类型:

String

缺省值:

/dev/dtremote

描述: 对于远程连接，**/etc/utmp** 中的 `line` 的值还必须作为 **/dev** 目录中的设备存在，这样才能使 **finger** 之类的命令正确运行。此资源指定了在远程显示连接时 **dtlogin** 所创建的 **/dev** 文件的路径名。对于大多数平台，该文件创建为 **/dev/null** 的符号链接。该指定值必须以 `/dev/`开头，否则该值将被废弃，并且不会创建任何文件。

pidFile

类: PidFile

类类型:

STring

缺省值:

NULL

描述: 按指定的文件名创建文件，其中包含 **dtlogin** 主进程的进程标识的 ASCII 表示。这可在向 **dtlogin** 发送信号时使用。**dtlogin** 客户机还使用文件锁定来尝试防止多个 dtlogin 在同一机器上运行。有关更多信息，请参阅 **lockPidFile** 资源。

removeDomainname

类: RemoveDomainname

类类型:

Boolean

缺省值:

True

描述: 在计算 XDMCP 客户机的显示名时，**dtlogin** 通常创建终端的标准主机名称。由于这有时会令人困惑，因此在设置此变量时，如果主机名的域名与本地主机的域名相同，则 **dtlogin** 将除去主机名的域名部分。

requestPort

类: RequestPort

类类型:

int

缺省值:

177

描述: 指示 **dtlogin** 用来侦听传入的 XDMCP 请求的 UDP 端口号。除非系统需要调试，否则应保留此资源的缺省值。

servers

类: Servers

类类型:

String

缺省值:

:0 Local local /system_dependent_path/X :0

描述: 指定填满服务器条目的文件名，每行一个服务器条目（如果值以斜杠开头），或者指定单个服务器条目。每个条目表示一个应持续受到管理并且未使用 XDMCP 的显示。每个条目的通用语法如下:

DisplayName DisplayClass DisplayType[@ite] [Command [options]]

此处:

DisplayName

可通过 **-display** 选项传给任何 X 程序的值。此字符串用在特定于显示的资源中，用于指定特定显示，因此必须小心谨慎以与名称匹配。例如，如果其他资源指定为 Dtlogin._0.session，则使用 :0 local /usr/bin/X11/X :0 而不是 localhost:0 local /usr/bin/X11/X :0。此字段中的星号（*）由 **dtlogin** 扩展为 **hostname** :0。

DisplayClass

显示类部分也用在特定于显示的资源中，它作为资源的类部分。如果您有大量相似显示（例如，一组 X 终端）并且想为这些组设置资源，则此项很有用。使用 XDMCP 时，要求显示指定显示类。有关您设备的标准显示类字符串的信息，请参阅 X 终端文档。

DisplayType

如果指定为 local，则表示 X 服务器应该对应此条目启动。remote 值表示应连接现有 X 服务器。

@ite 在本地位图上，用户可使用登录屏幕选择**命令行登录**选项，该选项将临时暂挂 X 服务器，并显示传统的字符 login: 提示。用户随后可登录并执行非 X 相关的任务。当用户完成并注销后，X 服务器重新启动，并且登录屏幕重新显示。为了支持**命令行登录**方式，显示必须有关联的内部终端仿真器（ITE）设备。缺省情况下，**dtlogin** 将 ITE 设备 “console”（**/dev/console**）与显示 :0 相关联。如果您的配置不匹配此缺省值，则对带有关联 ITE 的任何显示指定 @device，而对服务器文件中列出的所有其他显示指定 @none。

Command [options]

启动 X 服务器的字符串。**dtlogin** 客户机将总是使用指定的 *DisplayName* 连接到 X 服务器，因此您可能需要指定显式连接号作为 X 服务器的选项（前面示例中的 :0）。

sysParmsFile

类: SysParmsFile

类类型:

String

缺省值:

/system_dependent_path

描述: 指定包含 shell 命令的文件，其中某个命令设置系统的时区环境变量（**TZ**）。如果时区是使用 shell 语法 `TZ=` 设置的，则 **dtlogin** 可使用此信息设置用户会话的时区。

timeZone

类: TimeZone

类类型:

String

缺省值:

NULL

描述: 指定 **dtlogin** 的本地时区。它作为 **TZ** 变量的值装入 **dtlogin** 环境，并由所有后续会话继承。某些系统保留包含时区设置的配置文件（例如，**/etc/src.sh**）。另见 **sysParmsFile** 资源。

wakeupInterval

类: WakeupInterval

类类型:

Int

缺省值:

10

描述: 如果用户从登录屏幕选择**命令行登录**方式，则 **dtlogin** 终止 X 服务器，并允许传统的基于字符的登录提示 `login:` 可见。如果用户未在 **wakeupInterval** 秒数的两倍时间内登录，则 X 服务器重新启动。用户登录后，**dtlogin** 每 **wakeupInterval** 秒检查一次，以查看用户是否注销。如果已注销，则 X 服务器重新启动，并且登录屏幕重新显示。

显示资源

dtlogin 命令显示资源可为所有显示指定，也可为特定显示指定。要指定特定显示，显示名应插在资源名中，位于 `Dtlogin` 和最后的资源名段之间。例如，`Dtlogin.expo_0.startup` 是资源的名称，该资源定义 `expo:0` 显示上的启动 shell 文件。资源管理器用冒号将资源名称与其资源值分隔开，并用点分隔资源名称的各个部分，因此在生成资源名称时，**dtlogin** 使用下划线（`_`）代替点（`.`）和冒号（`:`）。

还可以通过插入类名而不是显示名，为一类显示指定资源。对于不受 XDMCP 管理的显示，可在服务器资源所引用的文件中指定其类关联。使用 XDMCP 的显示将其类关联作为 XDMCP 包的一部分提供。

以下 **dtlogin** 常规资源并不特定于显示，并且在适当情况下适用于所有显示。

authorize

类类别:
Authorize

类型: Boolean

缺省值:
False

描述: Authorize 是 Boolean 资源，它控制 **dtlogin** 是否对服务器连接生成和使用授权。另见 **authName** 资源。

authName

类类别:
AuthName

类型: String

缺省值:
MIT-MAGIC-COOKIE-1

描述: 如果使用了 **authorize** 资源，则 **authName** 指定要使用的授权的类型。目前，**dtlogin** 只支持 MIT-MAGIC-COOKIE-1 授权。XDM-AUTHORIZATION-1 可以受支持，但是 DES 通常是不可分发的。XDMCP 连接会动态地声明哪些授权类型是受支持的，因此在此情况下将忽略 **authName**。另见 **authorize** 资源。

authFile

类类别:
AuthFile

类型: String

缺省值:
NULL

描述: 使用 **-auth** 服务器命令行选项将授权数据从 **dtlogin** 传达到服务器。请将此资源保存在写保护目录中以防误删，如果误删，将禁用服务器中的授权机制。如果为 NULL，则 **dtlogin** 生成文件名。

chooser

类类别:
Chooser

类型:

缺省值:

描述: 指定运行程序来为重定向至特殊主机名 CHOOSER 的间接查询提供主机菜单。缺省值为 **/usr/dt/bin/dtchooser**。请参阅 Xaccess 文件部分。

cpp

类类别:
Cpp

类型: String

缺省值:
因系统而异

描述: 指定 **xrdb** 所使用的 C 预处理器的路径。

environment

类类别:

Environment

类型: String

缺省值:

因系统而异

描述: 包含一组由空格或制表符分隔的 *name=value* 对。每一项都将装入服务器和会话的环境中。有关更多信息，请参阅环境部分。

failsafeClient

类类别:

FailsafeClient

类型: String

缺省值:

/system_dep./xterm

描述: 如果缺省会话未能执行，则 **dtlogin** 退回到此程序。执行此程序时不带任何参数，但使用与会话相同的环境变量执行。

grabServer

类类别:

GrabServer

类型: Boolean

缺省值:

True

描述: 为提高安全性，**dtlogin** 在读取名称和密码时将抓取服务器和键盘。**grabServer** 资源指定在读取名称和密码时是否应挂起服务器。如果为 FALSE，则在键盘抓取成功后会取消抓取服务器；否则一直抓取服务器，直到会话开始之前。

grabTimeout

类类别:

GrabTimeout

类型: Int

缺省值:

3 秒

描述: 指定 **dtlogin** 将等待抓取成功的最大时间。如果另一个客户机已经使服务器处于抓取状态，或者网络等待时间可能极高，则抓取会失败。**grabTimeout** 资源的缺省值为 3 秒；使用此资源时应小心，因为用户可能被显示上貌似相同的窗口所误导。如果抓取失败，**dtlogin** 将杀死并重新启动服务器（如有可能）和会话。某些 X 终端在服务器被抓取时无法显示其登录屏幕。将 **grabServer** 设置为 FALSE 将允许屏幕显示，但是也带来了用户登录名遭窃的可能性，复制登录屏幕的内容即可窃取登录名。因为键盘仍处于抓取状态，并且密码不回显，所以密码不会被窃。

language

类类别:

Language

类型: String

缺省值:

因系统而异

描述: 指定 **LANG** 环境变量的缺省设置。如果 **dtlogin** 屏幕已针对该语言进行了本地化，则它将相应显示；否则，它将以 C 语言显示。用户可使用登录屏幕上的选项暂时覆盖此设置。当后续会话终止时，**LANG** 变量还原为此设置。

languageList

类类别:
LanguageList

类型: String

缺省值:
NULL

描述: 允许用户覆盖登录屏幕的“语言”菜单中所显示的缺省语言集。如果某个特定显示上实际使用的语言集小于系统上安装的语言集，则此项非常有用。该资源值是 **LANG** 环境变量有效值的列表。语言值应由一个或多个空格或制表符分隔。

languageName

类类别:
LanguageName

类型: String

缺省值:
NULL

描述: 允许用户用替代文本覆盖登录屏幕的“语言”菜单中所显示的缺省语言环境名称。这样，用户就不会看到 `En_US` 项，他们可能看到 `English (United States)` 项。此资源指定为 **Dtlogin** ***local_name*. **languageName**: *text*，如下所示:

```
Dtlogin*En_US.languageName: English (United States)
Dtlogin*Fr_CA.languageName: French (Canadian)
```

openDelay

类类别:
OpenDelay

类型: Int

缺省值:
5 秒

描述: 指定打开受阻服务器的两次连续尝试之间的持续时间（以秒数计）。

openRepeat

类类别:
OpenRepeat

类型: Int

缺省值:
5 秒

描述: 指定打开受阻服务器的连续尝试次数。

openTimeout

类类别:
OpenTimeout

类型: Int

缺省值:
30 秒

描述: 指定实际尝试打开受阻服务器时的等待时间。这个时间与用于 **connect** 系统调用的最大时间相同。

pingInterval

类类别:

PingInterval

类型: Int

缺省值:

5 分钟

描述: 为了发现远程显示何时消失，**dtlogin** 将对远程显示偶尔进行 ping 操作，ping 时使用 X 连接并发送 XSync 请求。**pingInterval** 资源指定连续 ping 尝试之间的时间（以分钟计）。

pingTimeout

类类别:

PingTimeout

类型: int

缺省值:

5 分钟

描述: 指定终端响应请求的最大等待时间（以分钟计）。如果终端未响应，则会话终止。**dtlogin** 客户机不会 ping 本地显示。本地会话不应由于服务器等待（例如，等待远程文件系统服务）并且未响应 ping 而终止。

reset

类类别:

Reset

类型: String

缺省值:

NULL

描述: 指定在会话终止之后运行的程序（作为 root）。如果未设置此资源，则没有程序运行。约定名称为 **Xreset**。请参阅 Xreset 文件。

resetForAuth

类类别:

ResetForAuth

类型: Boolean

缺省值:

False

描述: 在样本服务器中授权的原始实施期间，授权文件是在服务器复位时重新读取的，而不是在检查初始连接时。由于 **dtlogin** 恰在连接到显示之前生成授权信息，因此旧服务器不会获得当前授权信息。此资源使 **dtlogin** 在设置文件之后向服务器发送 SIGHUP，从而导致再次发生服务器复位，在这段时间内，将读取新的授权信息。

resetSignal

类类别:

Signal

类型: Int

缺省值:

1 SIGHUP

描述: 指定 **dtlogin** 发送的用来复位服务器的信号。

resources

类类别:

Resource

类型: String

缺省值:

NULL

描述: 指定文件名称，该文件将由 **xrdb** 作为资源数据库装入显示的屏幕 0 的根窗口中。此资源数据库是在认证过程启动之前装入的，因此它可控制登录窗口的外观。请参阅有关认证屏幕的部分，那部分描述了适合于放在此文件中的各种资源。此资源没有缺省值，但是约定名称为 **Xresources**。

session

类类别:

Session

类型: String

缺省值:

/usr/dt/bin/Xsession

描述: 指定要为已认证的用户执行的会话。缺省情况下，运行 **/usr/dt/bin/Xsession** 文件。约定名称为 **Xsession**。请参阅 Xsession 文件。

setup

类类别:

Setup

类型: String

缺省值:

NULL

描述: 指定在显示认证屏幕之前运行的程序（作为 root）。缺省情况下，没有程序运行。约定名称为 **Xsetup**。请参阅 Xsetup 文件。

startAttempts

类类别:

StartAttempts

类型: Int

缺省值:

4

描述: 有四个数值资源控制在尝试打开受阻服务器时 dtlogin 的行为，它们是: **openDelay**、**openRepeat**、**openTimeout** 和 **startAttempts**。此资源指定整个过程在服务器上放弃之前所发生的次数。在进行了 **openRepeat** 次尝试之后，或者，如果任何特定尝试经历了 **openTimeout** 秒，**dtlogin** 将终止并重新启动服务器，同时尝试再次连接。此过程重复 **startAttempts** 次，在此之后，将声明该显示无效并已禁用。

startup

类类别:

Startup

类型: String

缺省值:

NULL

描述: 指定在认证过程成功后运行的程序（作为 root）。缺省情况下，没有程序运行。此处使用的文件的约定名称为 **Xstartup**。请参阅 Xstartup 文件部分。

systemPath

类类别:

SystemPath

类型: String

缺省值:

system_dep._path

描述: **dtlogin** 客户机将启动和复位脚本的 **PATH** 环境变量设置为此资源的值。请注意此条目中明显缺少 “.”。这是 root 应遵循的一种很好的做法，因为它避免了多种系统渗透方案。

systemShell

类类别:

SystemShell

类型: String

缺省值:

/bin/sh

描述: **dtlogin** 客户机将启动和复位脚本的 **SHELL** 环境变量设置为此资源的值。

terminateServer

类类别:

TerminateServer

类型: Boolean

缺省值:

False

描述: 指定 X 服务器是否应在会话结束后终止（而不是复位）。如果服务器运行时间倾向于无限增加，则可以使用此选项来限制服务器持续运行的时间量。

termSignal

类类别:

Signal

类型: Int

缺省值:

15 (SIGTERM)

描述: 指定 **dtlogin** 发送的用来终止服务器的信号。

userAuthDir

类类别:

UserAuthDir

类型: String

缺省值:

/var/dt

描述: 当 **dtlogin** 无法写入常规用户授权文件（**$HOME/.Xauthority**）时，它将在此目录中创建一个唯一的文件名，并将环境变量 **XAUTHORITY** 指向创建的文件。

userPath

类类别:

UserPath

类型: String

缺省值:

system_dep._path

描述: **dtlogin** 客户机将会话的 **PATH** 环境变量设置为此值。它应该是冒号分隔的目录列表。

xdmMode

类类别:
XdmMode

类型: Boolean

缺省值:
False

描述: 如果为 True，则 **$HOME/.xsession** 文件将在用户认证时从 **Xsession** 执行，而不是从 **dtsession**。

xrdb

类类别:
Xrdb

类型: String

缺省值:
/system_dep./xrdb

描述: 指定用于装入资源的程序。认证屏幕从键盘读取 *name-password* 对。因为这是 Motif 工具包客户机，所以颜色、字体和某些布局选项可通过资源进行控制。此屏幕的常规资源应放入由 **resources** 资源指定的文件中（缺省值为 **Xresources**）。在 **Dtlogin app-defaults** 文件中指定特定于语言的值，如文本或字体。

徽标资源

认证屏幕上的缺省徽标可更换为用户选择的位图或像素图。指定这些资源时，它们前面应加上字符串 `Dtlogin*logo*`。

bitmapFile

类类别:
BitmapFile

类型: String

缺省值:
NULL

描述: 指定要用作徽标的位图或像素图文件的绝对路径名称。

background

类类别:
Background

类型: Pixel

缺省值:
#a8a8a8

描述: 指定徽标的背景色。

topShadowPixmap

类类别:
topShadowPixmap

类型: String

缺省值:
25_foreground

描述: 指定要用作徽标边框阴影的像素图。

以下资源描述登录屏幕上所使用的问候语字符串。指定这些资源时，它们前面应加上字符串 `Dtlogin*greeting*`。

foreground

类类别:
Foreground

类型: Pixel

缺省值:
black

描述: 指定欢迎消息的前景色。

background

类类别:
Background

类型: Pixel

缺省值:
dynamic

描述: 指定欢迎消息的背景色。对于彩色系统缺省为浅灰色，对于单色系统则为白色。

fontlist

类类别:
FontList

类型: FontList

缺省值:
-*-*schoolbook-medium-i-normal--18-*

描述: 指定用于欢迎消息的字体。

labelString

类类别:
LabelString

类型: String

缺省值:
Welcome to %LocalHost%

描述: 指定用作欢迎消息的字符串。在文本中包含换行符 (0 可指定多行。如果文本中包含标记 %LocalHost%，则该标记将替换为提供登录服务的主机的名称。如果文本中包含标记 %DisplayName%，则该标记将替换为显示名称。

perLabelString

类类别:
LabelString

类型: String

缺省值:
Welcome %s

描述: 指定用于个性化欢迎消息的字符串。这是在输入用户名之后显示的消息。%s 将替换为输入的用户名。

alignment

类类别:
Alignment

类型: String

缺省值:
ALIGNMENT_CENTER

描述: 指定用于欢迎消息对齐的字符串。有效值为 ALIGNMENT_BEGINNING、ALIGNMENT_CENTER 和 ALIGNMENT_END。

Matte 资源

以下资源描述登录屏幕上使用的 matte 布局。指定这些资源时，它们前面应加上 `Dtlogin*matte.` 字符串。

width

类类别:
Width

类型: Int

缺省值:
对于高分辨率显示，为 806
对于中分辨率显示，为 755
对于低分辨率显示，为 585

描述: 指定对 **login_matte** 使用的宽度。

height

类类别:
Height

类型: Int

缺省值:
对于高分辨率显示，为 412
对于中分辨率显示，为 385
对于低分辨率显示，为 300

描述: 指定对 **login_matte** 使用的高度。

标签资源

以下资源描述登录屏幕上使用的字体布局。指定这些资源时，它们前面应加上 `string Dtlogin*.`。

labelFont

类类别:
LabelFont

类型: String

缺省值:
对于高分辨率显示，为 -*-swiss 742-medium-r-normal-*-140-*-p-110-*
对于低分辨率显示，为 -*-swiss 742-bold-r-normal-*-140-*-p-100-*

描述: 指定用于按钮和标签的 **labelFont**。

textFont

类类别:
TextFont

类型: String

缺省值:
对于高分辨率显示，为 -*-prestige-medium-r-normal-*-128-72-*
对于低分辨率显示，为 -*-helvetica-bold-r-normal-*-100-*

描述: 指定用于按钮和标签的 **textFont**。

标志

所有标志（除 **-config** 外）指定的值也可作为资源在配置文件中指定。通常，定制是使用配置文件（而不是命令行选项）完成的。这些标志对于调试和一次性测试最为有用。

-config *configuration_file*
指定一个资源文件，该文件指定了其余的配置参数。此文件替代 **dtlogin** 的缺省 **Xconfig** 文件。有关更多信息，请参阅 Xconfig 文件部分。

-daemon
指定 `true` 作为 **daemonMode** 资源的值。这将使 **dtlogin** 关闭所有文件描述符、取消与控制终端的关联，并在首次启动时将自己置于后台（就像是其他守护程序的宿主）。

-debug *debug_level*
指定用于 *debug_level* 资源的数字值。非零值将使 **dtlogin** 向终端显示调试语句；它还会禁用 **daemonMode** 资源，强制 **dtlogin** 同步运行。

-error *error_log_file*
指定 *error_log_file* 资源的值。有关更多信息，请参阅 Xerrors 文件部分。

-nodaemon
指定 `false` 作为该资源的值。

-resources *resource_file*
指定 *resource_file* 资源的值。有关更多信息，请参阅 Xresources 文件部分。

-server *server_entry*
指定 *server_entry* 资源的值。有关更多信息，请参阅 Xservers 文件部分。

-udpPort *port_number*
指定 **requestPort** 资源的值。此项设置 **dtlogin** 用于监视 XDMCP 请求的端口号。由于 XDMCP 使用公开的已注册 udp 端口 177，因此除进行调试外，请避免更改此资源。

-session *session_program*
指定 *session_program* 资源的值。有关更多信息，请参阅 Xconfig 文件部分。

环境变量

dtlogin 命令使用以下缺省环境调用用户会话:

DISPLAY
设置为相关显示名称。

EDITOR
设置为 **/usr/dt/bin/dtpad**。

HOME
设置为用户的主目录。

KBD_LANG
设置为对应适用语言的 **LANG** 值。

LANG
设置为当前 NLS 语言（如果存在）。

LC_ALL
设置为当前 NLS 语言（如果存在）。

LC_MESSAGES
设置为当前 NLS 语言（如果存在）。

LOGNAME
设置为用户名。

MAIL
设置为 **/usr/mail/$USER**（取决于系统）。

PATH
设置为 **userPath** 资源的值。

USER
设置为用户名。

SHELL
设置为用户的缺省 shell 程序（位于 **/etc/passwd**）。

TERM
设置为 **dtterm**。

TZ
设置为 *timeZone* 资源的值或系统缺省值。

XAUTHORITY
设置为权限文件。

添加到环境列表中

有四种方法可用于对上述列表进行修改或增补，具体取决于结果环境变量的所需作用域:

- **exportList** 资源可用于允许导出 **dtlogin** 的父进程提供给 dtlogin 进程的变量。通过此方法指定的变量既可用于显示的 X 服务器进程，也可用于用户会话，并且它们覆盖任何缺省设置。该资源接受至少由一个空格或制表符分隔的 *name=value* 字符串。
- **environment** 资源在 **dtlogin** 配置文件中提供，以允许以全局方式或基于每个显示方式设置环境变量。通过此方法指定的变量既可用于显示的 X 服务器进程，也可用于用户会话，并且它们覆盖任何缺省设置。该资源接受至少由一个空格或制表符分隔的 *name=value* 字符串。由于没有 shell 可用于对字符串进行语法分析，因此指定的值必须是常量。例如:

```
Dtlogin*environment:MAIL_HOST=blanco MAIL_SERVER=pablo
```

注: **LANG** 和 **TZ** 环境变量在配置文件中有其自己的专用资源，并且不应通过环境设置。

- 需要由 shell 进行处理或依赖于另一个环境变量值的环境变量可在启动脚本 **Xsession** 中指定。这些变量装入显示上所有用户的环境中，但不装入 X 服务器进程。它们覆盖同一变量的所有先前设置。**Xsession** 脚本接受使用 **ksh** 语法设置环境变量。例如:

  ```
  MAIL=/usr/mail/$USER
  ```

- 个人环境变量可在 **$HOME/.dtprofile** 脚本文件中以基于每个用户方式进行设置。**dtlogin** 命令对于此文件中的命令接受 **sh**、**ksh** 或 **csh** 语法。这些命令只能是设置环境变量的命令，而不是执行终端 I/O 的命令（**tset** 或 **stty** 除外）。如果 **.dtprofile** 的第一行为 `#!/bin/sh`、`#!/bin/ksh` 或 `#!/bin/csh`，则 **dtlogin** 将使用相应的 shell 对 **.dtprofile** 进行语法分析。否则，使用用户的缺省 shell 程序（**$SHELL**）。

退出状态

将返回以下退出值:

0	成功完成。
>0	发生错误。

示例

1. 要将 CDE 登录服务作为守护程序启动，请输入:

   ```
   /usr/dt/bin/dtlogin -daemon
   ```

2. 要以调试方式启动 CDE 登录服务，请输入:

   ```
   /usr/dt/bin/dtlogin -debug 1
   ```

位置

/usr/dt/bin/dtlogin

标准错误

dtlogin 命令返回以下错误消息:

- `Login incorrect; please try again.`
- `Unable to change to home directory.`
- `Sorry. Maximum number of users already logged in.`
- `Login error, invalid user ID.`
- `Login error, invalid group ID.`
- `Login error, invalid audit ID.`
- `Login error, invalid audit flag.`
- `Logins are currently disabled.`
- `Your current password has expired.`

文件

dtlogin 命令旨在用于广泛的各种环境中，并提供了一套可进行更改以适用于特定系统的配置文件。缺省的 **dtlogin** 配置文件可在 **/usr/dt/config** 中找到，但 **Xsession** 除外，它存储在 **/usr/dt/bin** 中。这些文件如下:

/usr/dt/config/Xconfig	指定其他 **dtlogin** 配置文件和 **dtlogin** 行为。
/usr/dt/config/Xaccess	控制从请求 XDMCP 服务的显示进行的访问。
/usr/dt/config/Xservers	包含供 **dtlogin** 进行显式管理的显示的列表。
/usr/dt/config/Xresources	包含指定登录屏幕外观的资源定义。
/usr/dt/config/Xsetup	在登录屏幕显示之前作为 root 执行的脚本。
/usr/dt/config/Xstartup	在成功认证用户后作为 root 执行的脚本。
/usr/dt/bin/Xsession	作为启动用户会话的已认证用户执行的脚本。
/usr/dt/config/Xfailsafe	作为启动故障防护会话的已认证用户执行的脚本。
/usr/dt/config/Xreset	在用户会话退出后作为 root 执行的脚本。

Xconfig 文件

Xconfig 文件包含 **dtlogin** 的常规资源，并且位于 **dtlogin** 配置文件树的顶端。**Xconfig** 指定其他 **dtlogin** 配置文件和日志文件的位置，并指定 **dtlogin** 行为。其他 **dtlogin** 配置文件和日志文件的位置由资源定义指定。缺省值如下:

Dtlogin.errorLogFile
: /var/dt/Xerrors

Dtlogin.pidFile
: /var/dt/Xpid

Dtlogin.accessFile
: Xaccess

Dtlogin.servers
: Xservers

Dtlogin*resources
: %L/Xresources

Dtlogin*setup
: Xsetup

Dtlogin*startup
: Xstartup

Dtlogin*reset
: Xreset

Dtlogin*failsafeClient
: Xfailsafe

Dtlogin*session
: /usr/dt/bin/Xsession

如果为 **accessFile**、**servers**、**resources**、**setup**、**startup**、**reset**、**failsafeClient** 或 **session** 指定的路径是相对路径，则 **dtlogin** 将首先在目录 **/etc/dt/config** 中查找文件，然后在 **/usr/dt/config** 中查找。

注: 某些资源在指定时含有分隔各个组成部分的 *。将 * 替换为显示名称可使这些资源对于每个不同的显示是唯一的。有关更多信息，请参阅显示资源。

缺省 **Xconfig** 文件为 **/usr/dt/config/Xconfig**。系统管理员可定制 **Xconfig**，方法是将 **/usr/dt/config/Xconfig** 复制为 **/etc/dt/config/Xconfig** 并修改 **/etc/dt/config/Xconfig**。缺省 **Xconfig** 文件包含上述配置文件和日志文件条目，还包含一些特定于供应商的资源定义和示例。

Xaccess 文件

accessFile 资源指定的数据库文件提供了一些信息，**dtlogin** 使用这些信息来控制从请求 XDMCP 服务的显示进行的访问。此文件包含三种类型的条目：控制对“直接”（Direct）和“广播”（Broadcast）查询的响应的条目、控制对“间接”（Indirect）查询的响应的条目，以及宏定义。

“直接”条目的格式为主机名或模式。模式不同于主机名的地方在于：它包含一个或多个元字符（* 匹配 0 个或多个字符的任意序列，而 ? 匹配任何单个字符），这些字符将与显示设备的主机名进行比较。如果该条目是主机名，则所有比较都使用网络地址进行，因此可转换为正确网络地址的任何名称都可以使用。对于模式，比较只使用规范主机名，因此请确保不要试图匹配别名。在主机名或模式前面放置一个惊叹号（!）字符将排除与该条目匹配的主机。

“间接”条目也包含主机名或模式，但在它后面是间接查询应发送到的主机名或宏的列表。间接条目还可指定 **dtlogin** 运行 **dtchooser** 以提供主机菜单，可以向这些主机显示登录屏幕。

宏定义包含宏名称和主机名列表，以及该宏扩展到的其他宏。宏和主机名的区别是：宏名以 % 字符开头。宏可以嵌套。

在检查对特定显示主机的访问时，将对每个条目依次进行扫描，第一个匹配的条目决定了响应。在扫描“间接”条目时将忽略“直接”和“广播”条目，反之亦然。空行将被忽略，# 作为注释定界符处理，它使该行的其余部分被忽略，\newline 使换行符被忽略，从而使间接主机列表可跨多行。

以下示例显示了一个 Xaccess 文件：

```
#
# Xaccess - XDMCP access control file
#

#
# Direct/Broadcast query entries
#
!xtra.lcs.mit.edu # disallow direct/broadcast service for xtra
bambi.ogi.edu     # allow access from this particular display
*.lcs.mit.edu     # allow access from any display in LCS

#
# Indirect query entries
#

#define %HOSTS macro
%HOSTS                expo.lcs.mit.edu xenon.lcs.mit.edu \
                      excess.lcs.mit.edu kanga.lcs.mit.edu

#force extract to contact xenon
extract.lcs.mit.edu xenon.lcs.mit.edu

#disallow indirect access by xtra
!xtra.lcs.mit.edu   dummy

#all others get to choose among %HOSTS
*.lcs.mit.edu       %HOSTS
```

如果授予 XDMCP 访问权，则可以在 **authDir** 所指定的目录中创建临时文件，该文件将包含 X 终端的授权信息。当会话开始后，将删除该文件。

对于不提供主机菜单用于“广播”或“间接”查询的 X 终端，**chooser** 程序可代它们提供。在 **Xaccess** 文件中，将 CHOOSER 指定为“间接”主机列表中的第一个条目。**chooser** 程序向列表中的其余每个主机名发送查询请求，并显示发出响应的所有主机的菜单。该列表可能含有词 BROADCAST，在此情况下，**chooser** 改为发送广播，并同样显示发出响应的所有主机的菜单。在某些操作系统中，UDP 包无法广播，所以此功能不起作用。

使用 **chooser** 程序的 **Xaccess** 文件的示例如下:

```
#offer a menu of these hosts to extract
extract.lcs.mit.edu CHOOSER %HOSTS

#offer a menu of all hosts to xtra
xtra.lcs.mit.edu    CHOOSER BROADCAST
```

用作 **chooser** 的程序由 **chooser** 资源指定。此程序的资源可放入由 resources 指定的文件中。缺省 **Xaccess** 文件为 **/usr/dt/config/Xaccess**。系统管理员可定制 **Xaccess**，方法是将 **/usr/dt/config/Xaccess** 复制为 **/etc/dt/config/Xaccess**，然后修改 **/etc/dt/config/Xaccess**。缺省 **Xaccess** 文件不包含任何条目。

Xservers 文件

Xservers 文件包含要管理的显示的列表。缺省 **Xservers** 文件为 **/usr/dt/config/Xservers**。系统管理员可定制 **Xservers**，方法是将 **/usr/dt/config/Xservers** 复制为 **/etc/dt/config/Xservers**，然后修改 **/etc/dt/config/Xservers**。缺省 **Xservers** 文件包含一条对应一个本机显示的条目。

Xresources 文件

Xservers 文件包含指定登录屏幕外观的资源定义。缺省 **Xresources** 文件为 **/usr/dt/config/Xresources**。系统管理员可定制 **Xresources**，方法是将 **/usr/dt/config/Xresources** 复制为 **/etc/dt/config/Xresources**，然后修改 **/etc/dt/config/Xresources**。

Xsetup 文件

Xsetup 文件通常是 shell 脚本。只有 root 用户才能运行该文件，并且他们应该非常注意安全性。此脚本在登录屏幕显示之前运行。没有任何类型的参数传入该脚本。**dtlogin** 命令一直等待，直到此脚本在显示登录屏幕之前退出。

缺省 **Xsetup** 文件为 **/usr/dt/config/Xsetup**。系统管理员可定制 **Xsetup**，方法是将 **/usr/dt/config/Xsetup** 复制为 **/etc/dt/config/Xsetup**，然后修改 **/etc/dt/config/Xsetup**。缺省 **Xsetup** 文件包含特定于供应商的代码，但是通常包含在登录屏幕显示之前设置 X 服务器的代码，例如设置键盘映射。

Xstartup 文件

Xstartup 文件通常是 shell 脚本。只有 root 用户才能运行该文件，并且他们应该非常注意安全性。该文件用来放置显示当日消息的命令或者代表用户执行其他系统级功能的命令。为使用此脚本，设置了以下环境变量:

DISPLAY
: 设置为相关显示名称。

HOME 设置为用户的主目录。

PATH 设置为 **systemPath** 资源的值。

USER 设置为用户名。

SHELL
: 设置为 **systemShell** 资源的值。

没有任何类型的参数传入该脚本。**dtlogin** 命令一直等待，直到此脚本在启动用户会话之前退出。如果此脚本的退出值非零，则 **dtlogin** 立即中断会话，并启动另一个认证周期。

缺省 **Xstartup** 文件为 **/usr/dt/config/Xstartup**。系统管理员可定制 **Xstartup**，方法是将 **/usr/dt/config/Xstartup** 复制为 **/etc/dt/config/Xstartup**，然后修改 **/etc/dt/config/Xstartup**。缺省 **Xstartup** 文件包含了一些代码，用来将 **/dev/console** 的所有权更改为控制台上正在运行的会话所属的用户。

Xsession 文件

Xsession 脚本初始化用户会话并调用桌面会话管理器。要具有授权用户的许可权才能运行该脚本，并且它具有多个预设的环境变量。请参阅环境变量以获得预设变量的列表。

缺省 **Xsession** 文件为 **/usr/dt/bin/Xsession**。系统管理员可定制 **Xsession**，方法是将 **/usr/dt/bin/Xsession** 复制为 **/etc/dt/config/Xsession**，然后修改 **/etc/dt/config/Xsession**。**Xconfig** 中定义的会话资源也必须进行更改以引用定制的 **Xsession** 文件。有关如何更新 **Xconfig** 文件的信息，请参阅 Xconfig 文件。缺省 **Xsession** 文件包含会话初始化代码。它确实包含特定于供应商的代码，但其常规功能如下:

- 将用户的 **$HOME/.dtprofile** 作为数据源
- 将任何 **/etc/dt/config/Xsession.d/*** 脚本作为数据源
- 将任何 **/usr/dt/config/Xsession.d/*** 脚本作为数据源
- 在后台启动桌面欢迎客户机 **dthello**
- 将应用程序搜索路径设置脚本 **dtsearchpath** 作为数据源
- 在后台启动帮助设置客户机 **dthelpgen**
- 在后台启动应用程序管理器目录设置客户机 **dtappgather**
- 执行桌面会话管理器 **dtsession**
-

建议系统管理员不要定制 **Xsession** 文件。

Xreset 文件

与 **Xstartup** 相对，**Xreset** 脚本在用户会话终止后运行。由于它由 root 用户运行，因此 **Xreset** 脚本中应该包含用于撤销 **Xstartup** 命令效果的命令，如从文件服务器上卸装目录。传给 **Xstartup** 的环境变量集也提供给 **Xreset**。

缺省 **Xreset** 文件为 **/usr/dt/config/Xreset**。系统管理员可定制 **Xreset**，方法是将 **/usr/dt/config/Xreset** 复制为 **/etc/dt/config/Xreset**，然后修改 **/etc/dt/config/Xreset**。缺省 **Xreset** 文件包含将 **/dev/console** 的所有权改回 root 用户的代码。

Xerrors 文件

Xerrors 脚本包含来自 **dtlogin** 的错误消息，以及由 **Xsetup**、**Xstartup** 或 **Xreset** 输出到 **stderr** 的任何内容。系统管理员可使用此文件的内容进行 **dtlogin** 故障诊断。**errorLogSize** 资源限制 **Xerrors** 文件的大小，可防止该文件无限增长。如果该文件确实增长得超过了所请求的大小，并且已由 **dtlogin** 截短，则任何正在访问该文件（例如，使用 **cat** 或 **tail**）的用户将需要关闭该文件（在文件截短后），然后重新打开进行访问，这样才能看到记入该文件中的后续信息。

系统管理员可通过设置 **Xconfig** 文件中的 **errorLogFile** 资源来更改 **Xerrors** 的路径名。

Xpid 文件

Xpid 脚本包含 **dtlogin** 主进程的进程标识，该标识可在向 **dtlogin** 发送信号时使用。系统管理员可通过设置 **Xconfig** 文件中的 **pidFile** 资源来更改 **Xpid** 的路径名。

相关信息

第 200 页的『dtaction 命令』、第 204 页的『dtlogin 命令』。

dtscript 命令

用途

构建 X Window System 环境中使用的简单对话框。

语法

dtscript [**-xrm***options*] [**-dir***Path*] [**-file***FileName*] [**-workspace***WorkspaceName*]。

注: 如果使用了 **-xrm** *选项*，在任何其他标志之前，就必须对它进行指定。

描述

台式机脚本支持 Motif 小窗口的一个子集，它就是您从选用区拖动和删除到对话框中的。在对话框中，您可以移动或者调整任何小窗口的大小。您也可以用提供的专门编辑器编辑小窗口属性。

您可以输入回调，给出小窗口的期望行为。完成对话框后，台式机脚本就会给它产生 dtksh 码。

标志

-dir*Path* 设置文件选择对话框中显示的台式机脚本当前目录到 *Path*。

-file*FileName* 装入一个名为 *FileName* 的现有的对话框。*FileName* 参数可以是一个绝对的路径名，或相对于当前目录的路径名，或者相对于 **-dir** 值的路径名。

-workspace*workspace* 将台式机脚本装入相应的 CDE 工作区。

-xrm*options* 使您能输入任何规范（*选项*），否则您就将进入一个资源文件。

示例

从窗口调用台式机脚本，输入:

```
dtscript
```

文件

/usr/dt/bin/dtscript 包含命令**dtscript**。

dtsession 命令

用途

管理 CDE 会话。

语法

dtsession [*options*] ...

描述

dtsession 命令在用户会话期间（从登录到注销）提供符合 ICCCM 1.1 的会话管理功能。它启动窗口管理器，并允许用户保存会话、恢复会话、锁定会话、启动屏幕保护程序，以及为兼容桌面的客户机分配颜色。

注: 桌面登录管理器 **dtlogin** 通过 **Xsession** 脚本自动调用 **dtsession** 客户机。**dtsession** 客户机还可以通过现有 X 服务器上的 **Xsession** 脚本启动。**dtsession** 会话管理器自动启动窗口管理器。

dtsession 命令支持以下任务:

- 初始化会话
- 启动窗口管理器
- 恢复主会话或当前会话
- 提供对命令或超时的会话锁定
- 提供对命令或超时的会话屏幕保存
- 充当其他桌面客户机的颜色分配服务器
- 保存主会话或当前会话
- 注销时显示确认对话
- 注销时显示会话选择对话
- 终止会话

会话

会话是显示在用户桌面上的应用程序、设置和资源的集合。会话管理是一组约定和协议，它们允许某个特殊的会话管理器（如 **dtsession**）保存和恢复用户会话。用户可登录到系统，并将看到与用户注销时所见相同的运行中的应用程序、设置和资源的集合。当用户首次登录到桌面时，将装入缺省初始会话。随后，**dtsession** 支持当前会话和主会话的概念。

定义了以下会话:

初始会话
当用户首次登录到桌面时，**dtsession** 将使用系统缺省值生成用户初始会话。有关更多信息，请参阅会话资源管理和会话应用程序管理。

当前会话
正在运行的用户会话总是视为当前会话，无论该会话是登录时从保存的主会话、从保存的当前会话，还是从系统缺省初始会话恢复的。根据用户的“样式管理器启动”设置，当用户退出会话时，当前会话将自动保存。当用户下一次登录到桌面时，先前保存的当前会话将重新启动。桌面恢复为用户上次注销时的相同状态。

主会话 这是另一种选项，不管用户注销时桌面是什么状态，该选项在用户每次登录时会将桌面恢复为同一状态。用户可保存当前会话的状态，然后设置“样式管理器启动”，以使桌面在每次用户登录时启动该会话。

特定于显示的会话
要为特定显示运行特定会话，用户可创建特定于显示的会话。为此，用户可将 **$HOME/.dt/sessions** 目录复制为 **$HOME/.dt/***display*，其中 display 是真实的非标准主机名（例如，`pablo:0` 是有效的，但 pablo.gato.com:0 或 `local:0` 无效）。当用户在显示 `pablo:0` 上登录时，该特定于显示的会话将优先。

ICCCM 会话管理协议

对于要在注销时保存并在登录时重新启动的应用程序，它必须使用简单会话管理协议。**dtsession** 命令支持 ICCCM 1.1 会话管理协议。

需要保存其状态的应用程序可使用 WM_SAVE_YOURSELF 协议。为此，应用程序需要在一个（且只能是一个）顶级窗口上设置 WM_SAVE_YOURSELF 属性。保存会话后，**dtsession** 将向应用程序的顶级窗口发送 WM_SAVE_YOURSELF 客户机消息。此时，应用程序将静默保存其状态。在应用程序保存其状态时，它无法

以任何方式与用户交互。由于应用程序可能将其状态保存到文件中，因此会话管理器提供了一个便利的函数 **DtSessionSavePath** ，它返回一个文件的完整路径名，应用程序可将其状态保存在该文件中。在应用程序保存其状态时，**dtsession** 等候应用程序在完成时发出通知。为了通知 **dtsession** 状态保存已完成，应用程序必须更新其顶级窗口上的 WM_COMMAND 属性。

应用程序顶级窗口上的 WM_COMMAND 属性提供两种作用。首先，此属性的更改指示 **dtsession**: 应用程序已完成其状态保存，并且 **dtsession** 可继续处理下一个应用程序。其次，WM_COMMAND 属性值应该包含 **dtsession** 在会话启动时用来重新启动应用程序的命令行。如果应用程序是用完整路径名启动的，则在设置 WM_COMMAND 时，它应使用完整路径名。不需要保存其状态，但希望重新启动的应用程序只需在应用程序启动期间设置 WM_COMMAND 一次。

恢复会话

在会话启动时，**dtsession** 确定恢复哪个会话。以下列表描述了优先顺序:

1. 特定于显示的当前会话或主会话
2. 当前会话或主会话
3. 初始会话

会话资源管理

会话管理器使用 X 服务器 RESOURCE_MANAGER 属性使桌面资源对所有应用程序可用。会话管理器按以下方式装入 RESOURCE_MANAGER:

1. 装入系统缺省资源。
2. 合并系统管理员指定的所有资源。
3. 合并用户指定的所有资源。

桌面缺省资源可在 **/usr/dt/config/$LANG/sys.resources** 文件中找到。这些资源通过 RESOURCE_MANAGER 属性对每个用户会话可用。请勿编辑此文件，因为在后续桌面安装期间将无条件覆盖该文件。

通过创建 **/etc/dt/config/$LANG/sys.resources** 文件，系统管理员可覆盖系统缺省资源或指定更多资源。由于此文件在会话启动期间合并到桌面缺省资源中，所以只能将关于新资源或更新资源的指定放置到此文件中。该做法比复制桌面缺省资源文件更好。此文件中指定的资源通过 RESOURCE_MANAGER 属性对每个用户会话可用。此文件中指定的资源优先于桌面缺省资源文件中指定的资源。

通过编辑 **$HOME/.Xdefaults** 文件，用户可覆盖桌面缺省资源和系统管理员资源。此文件中指定的资源通过 RESOURCE_MANAGER 属性仅对该用户的会话可用，并且优先于桌面缺省资源文件或系统管理员资源文件中指定的资源。

注: “X 工具箱内在机制”规定它将从 RESOURCE_MANAGER 或者从 **$HOME/.Xdefaults** 装入应用程序资源，但不能同时从两者装入。一般而言，这意味着将忽略用户的 **$HOME/.Xdefaults** 文件。但是，如前所述，会话管理器在会话启动时会把 **$HOME/.Xdefaults** 合并到 RESOURCE_MANAGER 中，以此来接受 $HOME/.Xdefaults。当用户更改其 **$HOME/.Xdefaults** 文件时，这些更改要在用户调用 **ReloadResources** 操作之后才能对新的应用程序可见。

ReloadResources 操作指示会话管理器将系统指定的资源、系统管理员指定的资源以及用户指定的资源重新装入 RESOURCE_MANAGER。这样可使对系统管理员指定的或用户指定的资源文件所作的更改对新的应用程序可用。

会话应用程序管理

在会话启动时，会话管理器将重新启动作为会话的一部分而保存的所有应用程序。要作为用户“初始会话”的一部分恢复的系统缺省应用程序集可在 **/usr/dt/config/$LANG/sys.session** 文件中找到。请勿编辑此文件，因为在后续桌面安装期间将无条件覆盖该文件。

通过创建 **/etc/dt/config/$LANG/sys.session** 文件，系统管理员可替换作为用户“初始会话”的一部分恢复的应用程序集。与资源文件不同，此文件可完全替代桌面缺省文件，因此您可以复制系统缺省文件并进行任何必要的修改。

窗口管理器

dtsession 命令启动窗口管理器。缺省情况下，启动 **/usr/dt/bin/dtwm**。可使用 **wmStartupCommand** 资源指定替代窗口管理器。有关更多信息，请参阅“工作空间管理器”规范。

样式管理器

样式管理器提供了界面，用户可通过该界面更改当前会话的各个桌面设置和 X 服务器设置。有关更多信息，请参阅“样式管理器”规范。

颜色服务器

dtsession 命令充当桌面的颜色服务器，并提供可用于进行配置的以下资源集:

foregroundColor
: 控制是否分配像素用于前景色。

dynamicColor
: 指定是否分配只读颜色。

shadowPixmaps
: 指定是否为顶部阴影或底部阴影分配颜色。

colorUse
: 限制颜色分配。

writeXrdbColors
: 指定资源数据库中是否放置 *background 和 *foreground 资源。

有关更多信息，请参阅颜色服务器资源部分。

会话锁定

dtsession 命令提供会话锁定。按下前面板上的锁图标可直接锁定当前会话。如果 X 服务器支持，则在当前会话处于不活动状态达到了指定时间后，可将它锁定。要解锁会话，用户必须输入其登录密码、root 用户的登录密码，或者 **keys** 资源所指定的任何用户的登录密码。有关 keys 资源的更多信息，请参阅屏幕锁和屏幕保护资源。

dtsession 命令是由 PAM 启用的会话管理器，服务名称为 **dtsession**。它支持将传统的本地 UNIX 认证以及 PAM 认证用于将会话解锁。各个供应商可增加其他再认证功能，如 DCE 要求的再认证功能。

将 PAM 用于认证的系统范围配置的设置方法如下: 建立 root 用户许可权，然后在 **/etc/security/login.cfg** 文件的 **usw** 节中，将 *auth_type* 属性的值修改为 PAM_AUTH。

启用 PAM 时使用的认证机制取决于 **/etc/pam.conf** 中的登录服务的配置。**dtsession** 命令需要对应于 **auth** 模块类型的 **/etc/pam.conf** 条目。以下配置是 **/etc/pam.conf** 中针对 **dtsession** 服务的建议配置:

```
dtsession        auth                required        /usr/lib/security/pam_aix
```

屏幕保护程序

dtsession 命令提供了对启动外部屏幕保护程序的支持，屏幕保护程序可作为从前面板锁定的会话的一部分启动，或者，如果 X 服务器支持，也可以在不活动状态达到了指定时间之后启动。有关屏幕保护程序如何集成到桌面的信息，请参阅屏幕保护程序规范。

X 服务器屏幕保护程序扩展

dtsession 命令在不活动状态达到指定时间后是否能够提供会话锁定或屏幕保护程序启动，取决于 X 服务器屏幕保护程序扩展的可用性。**dtsession** 命令支持 “X 协会样本 X11 屏幕保护程序扩展 1.0” 和 “HP X 屏幕保护程序扩展”。**dtsession** 命令是两种扩展都能识别、只能识别其中一种，还是都不能识别，这取决于供应商。

启动会话管理器

dtsession 命令应从 **Xsession** 脚本中启动。登录管理器规范中描述了 **Xsession**。虽然建议的做法是作为缺省登录序列的一部分从 **dtlogin** 启动 **Xsession**，但是某些系统允许代理程序（如 xinit、x11start 或 startx）启动 **Xsession**。

颜色服务器资源

colorUse

类类别:

ColorUse

类型: String

缺省值:

DEFAULT

描述: 指定用于用户界面的颜色的数量。颜色服务器将根据屏幕显示平面的数量确定监视器的类型，如下所示:

1、2 或 3 个平面（B_W）

指定黑白系统。调色板将两种颜色单元用于用户界面。在此配置中，只有两种调色板可用: BlackWhite 和 WhiteBlack。这两个调色板无法动态更改。要更改调色板，必须重新启动使用该调色板的所有应用程序。此资源值强制将 **ShadowPixmaps** 设置为 True，并将 **ForegroundColor** 设置为 black 或 white（取决于所选的调色板）。

4 或 5 个平面（LOW_COLOR）

指定低色系统。调色板有两个颜色集，并将最大 12 个颜色单元用于用户界面，包括黑和白（颜色单元 0 和 1）。可使用资源 **ShadowPixmaps** 和 **ForegroundColor** 减少颜色单元的数量。

6 个平面（MEDIUM_COLOR）

指定中色系统。调色板有四个颜色集，并将最大 22 个颜色单元用于用户界面，包括黑和白（颜色单元 0 和 1）。可使用资源 **ShadowPixmaps** 和 **ForegroundColor** 减少颜色单元的数量。

7 个以上的平面（HIGH_COLOR）

指定高色系统。调色板有八个颜色集，并将最大 42 个颜色单元用于用户界面，包括黑和白（颜色单元 0 和 1）。可使用资源 **ShadowPixmaps** 和 **ForegroundColor** 减少颜色单元的数量。

dynamicColor

类类别:

DynamicColor

类型: Boolean

缺省值:

True

描述: 此资源的值可为 True 或 False。**dynamicColor** 资源用于减少所用颜色单元的数量。选择了调色板之后，如果调色板不大可能更改，则 **dynamicColor** 可设置为 False。如果设置为 False，则不可使用桌面样式管理器动态更改颜色。所选调色板在下一次会话中生效。当下一次会话启动时，颜色服务器将使用所有客户机可共享的只读颜色单元，从而减少了所用颜色单元的数量。

foregroundColor

类类别:

ForegroundColor

类型: String

缺省值:

DYNAMIC

描述: 此资源的值可为 White、Black 或 Dynamic。**foregroundColor** 资源使所有文本（前景）使用像素 0 或 1（Black 或 White），或者使某个颜色单元专用于前景色，并相应地更改每个颜色集的背景色（Dynamic）。如果设置为 White 或 Black，则每个颜色集所使用的颜色单元数量将减 1。

shadowPixmaps

类类别:
ShadowPixmaps

类型: String

缺省值:
DEFAULT

描述: 对于彩色系统，此资源的值可为 True 或 False。如果为 True，则 **topShadowColor** 和 **bottomShadowColor** 使用与背景相同的像素，并且将指定 **topShadowPixmap** 和 **bottomShadowPixmap**（而非实心的颜色），以创建 3-D 外观。这将使每个颜色集的颜色单元数量减 2。对于有 4 个或更少颜色平面（16 或更少颜色单元）的系统，此资源缺省为 True，对于有四个以上颜色平面的系统，其缺省为 False。

writeXrdbColors

类类别:
WriteXrdbColors

类型: Boolean

缺省值:
True

屏幕锁和屏幕保护资源

keys

类类别:
Keys

类型: unsigned char

缺省值:
NULL

描述: 列出能够在屏幕由用户锁定的任何时候将屏幕解锁的钥匙持有者。该列表是由逗号分隔的用户标识列表。例如，如果用户 `kim` 在会话期间使以下资源有效，则用户 `fred` 和 `keith` 能够在 `kim` 锁定显示时将该显示解锁:

```
Dtsession*keys: fred,keith
```

passwordTimeout

类类别:
passwordTimeout

类型: unsigned int

缺省值:
10

描述: 指定密码对话框从屏幕上除去之前经过的时间（单位: 秒）。当显示锁定时，指针将显示一个锁光标，并且显示一个对话框询问用户密码。如果没有从指针或键盘上检测到任何活动的持续时间已达 *passwordTimeout* 秒，会从屏幕上除去该对话框。一旦检测到指针或键盘事件，该对话框将重新显示。*passwordTimeout* 为 0 将使密码对话框在显示锁定的整个时间内持续显示。缺省值为 10 秒。

其他资源

queryServerSettings

类类别:
QueryServerSettings

类型: Boolean

缺省值:
False

描述: 指定 **dtsession** 命令是否在注销时查询服务器以获取其所有设置，或者是否只保存使用桌面样式管理器设置的那些设置。查询服务器将确保所有设置都会保存；但是，当执行完全查询时，性能将下降。缺省值为 False，这表示不查询服务器。

saveFontPath

类类别:
SaveFontPath

类型: Boolean

缺省值:
False

wmStartupCommand

类类别:
WmStartupCommand

类型: executable path

缺省值:
NULL

描述: 允许在登录时启动替代窗口管理器。如果此资源为 NULL，则 **dtsession** 启动 **/usr/dt/bin/dtwm**。替代启动可能类似于下面的内容:

```
Dtsession*wmStartupCommand:   /usr/bin/X11/mwm
```

该命令不应包含任何 shell 命令，并且不应括在引号中。如果使用了非 **/usr/dt/bin/dtwm** 的任何其他窗口管理器，则客户机仍将恢复，但可能不会恢复到正确位置。缺省情况下，此资源包含 NULL 值。

标志

-migrate 指示 **dtsession** 迁移先前的会话中保存的资源信息。如果指定了此选项，可能会忽略先前保存的会话中的客户机重新启动信息。此选项应该只有在先前的会话保存在 AIX 4.1.1 或 AIX 4.1.2 上的情况下才是必需的，并且通常是通过修改 **/usr/dt/bin/Xsession** 脚本指定的。**/usr/dt/bin/Xsession** 脚本包含有关如何修改该脚本以便为 **dtsession** 指定此选项的信息。

退出状态

将返回以下退出值:

0 成功完成。
>0 发生错误。

示例

1. 要从命令行启动会话管理器，而不恢复先前的会话，请输入:

```
dtsession -norestore
```

位置

/usr/dt/bin/dtsession

文件

/usr/dt/config/$LANG/sys.session	用户初始会话的桌面缺省应用程序集。
/etc/dt/config/$LANG/sys.session	用户初始会话的系统管理员指定的应用程序集。
/usr/dt/config/$LANG/sys.resources	桌面缺省资源。
/etc/dt/config/$LANG/sys.resources	系统管理员指定的资源。
$HOME/.Xdefaults	用户指定的资源。 **注**: **dtsession** 命令在 **$HOME/.dt/display** 或 **$HOME/.dt/sessions** 中存储会话信息。这些目录的内容不应由用户直接编辑。
/usr/dt/app-defaults/$LANG/Dtsession	缺省 **dtsession** 资源。

相关信息

第 200 页的『dtaction 命令』、第 204 页的『dtlogin 命令』。

---

dtterm 命令

用途

为旧的应用程序提供运行时的支持。

语法

dtterm [Flags...]

描述

dtterm客户机为写给 ANSI X3.64-1979 和 ISO 6429:1992(E)一致的字符终端的旧应用程序提供运行时的支持。

标志

注: **dtterm** 终端仿真器接受所有的标准 X 工具箱命令行标志以及附加标志，所有这些都列在下面: （如果标志是以 **+** 开始，而不是以 **-** 开始则该标志恢复到它的缺省值）

-132	这样可以识别 DECCOLM 转义序列，也可以适当地调整 **dtterm** 窗口大小。通常情况下，在 80 到 132 柱状方式之间转换的 DECCOLM 转义序列被忽略。相关资源: c132。
+132	导致忽略 DECCOLM 转义序列。这是缺省行为。相关资源: c132。
-aw	说明应该允许自动环绕处理。当光标在一行的最右边，并且文本是输出这允许光标可以自动的换到下一行的开始。这是缺省行为。相关的资源: autoWrap。
+aw	说明应该禁止自动环绕处理。相关的资源: autoWrap。

-background*background_color* 指定终端窗口的背景和用于滚动条和 X11 指针光标的缺省背景。在 CDE 下，该标志是缺省的基本颜色设置选择像素或者背景像素可以参阅 -bs。如果没有 CDE，该标志用最终的黑色衰减色作为 *background/*Background 的缺省色。*background_color* 描述了所用的背景颜色。相关资源：background。

-bd *border_color* 指定所有窗口的边界色。如果使用了再生窗口管理器，例如 **dtwm** 和 **mwm**，shell 小窗口边界就可能不可见。缺省颜色为黑色。*border_color* 描述了所用的边界颜色。相关的资源：borderColor。

-bg *background_color* 等同于 -background。*background_color* 描述了所用的背景色。相关资源：background。

-bordercolor *border_color* 等同于上述 -bd。*border_color* 描述了所用的边界颜色。相关的资源：borderColor。

-borderwidth *border_width* 指定 shell 小窗口的边界宽度。这个值可能被再生窗口管理器如 **dtwm** 和 **mwm** 覆盖。缺省值为 0。*border_width* 用像素指定了窗口边界的宽度。相关的资源：borderWidth。

-bs 指定终端窗口应该使用 Motif 选择色，而不是用于终端窗口背景色的背景色。这是缺省行为。相关资源：backgroundIsSelect。

+bs 指定终端窗口应该使用 Motif 选择色，而不是用于终端窗口背景色的背景色。相关资源：backgroundIsSelect。

-bw *border_width* 等同于 -borderwidth。相关的资源：borderWidth。

-C 指定在 **/dev/console** 指引的输出应该指引到终端窗口。它作为一种方式提供，来避免那些通常在 ITE 上显示的输出覆盖 X 服务器上的显示。它不是作为一种指引输出从任意系统 **/dev/console** 到任意 X 服务器上的通用机理提供。

注：必须拥有 **/dev/console** 的读/写访问权才能使此标志有效。

-display *display_name* 指定 X11 显示服务器由 **dtterm** 使用。缺省值为 $DISPLAY 环境变量中的值。*display_name* 指定了连接到的 X11 服务器。

-e *program_argument...* 当启动 **dtterm** 后，指定一个要调用的可执行程序，作为一个子进程。该标志必须是命令行的最后一个标志。*program_argument* 指定了要运行的程序和命令行参数。

-fb *fontset* 在显示黑体终端文本时，指定要用的 XmFontSet。应该将其指定为 Motif XmFontList。只支持字符或者单空格的字体。使用均衡字体的行为没有定义。根据用户字体的 XLFD 名称就会生成缺省黑体。如果该字体不可用，通过使用一个像素偏移量叠印用户字体，就会生成黑体文本。*fontset* 指定了要用的黑体终端 XFontSet。相关资源：userFont。

-fg *foreground_color* 指定用于滚动条和 X11 指针光标的缺省前台色和终端窗口的前台色。在 CDE 下，该资源缺省为基本色集前景色像素。如果不在 CDE 下，该资源就会缺省为具有最终白色衰减色的 *foreground 或者 *Foreground。*foreground_color* 指定了所用的前景色。相关资源：foreground。

-fn *fontset* 在显示终端文本时，指定一个要用的 XFontSet。应该将其指定为 Motif XmFontList。只支持字符或者单空格的字体。使用均衡字体的行为没有定义。该字体不会用来显示非终端文本，例如菜单栏、弹出菜单和对话框等。缺省值是使用父公告牌的 XmNtextFontList 值（参阅 XmBulletinBoard），它的方式与 XmText 窗口一样。*fontset* 指定所要使用的终端 XFontSet。相关资源：userFont。

-font *fontset* 等同于 -fn。*fontset* 指定了要用的终端 XFontSet。相关资源：userFont。

-foreground *foreground* 等同于-fg。*foreground* 指定了所用的前景色。相关资源：foreground。

-geometry *geometry_string* 指定期望的终端窗口的尺寸和位置。缺省大小是 24 行，每行 80 字符。没有缺省位置。*geometry_string* 指定要用的终端几何结构。相关资源：geometry。

-help 显示一条消息，总结 **dtterm** 的用法。

-iconic 说明终端仿真器最初应该放置在图标化的显示上。相关资源：iconic。

+iconic 说明终端仿真器最初应该作为一个通常的窗口放置在显示上。这是缺省行为。相关资源：iconic。

-j 指定需要使用的跳转滚动。在跳转滚动下屏幕一次可以滚动多行。在文本的很多行正发送到终端时，它就提供了更快的屏幕更新。最大可以跳转滚动的行数是由终端窗口中的行数限定的。每一行都要显示。这是缺省行为。相关资源：jumpScroll。

+j 指定不应该使用的跳转滚动。关于跳转滚动的描述，请参阅 -j。相关资源：jumpScroll。

-kshMode 指定应该启动 **ksh** 方式。在 **ksh** 方式下，通过扩展修饰符位设置按下的键，会生成转义字符，后面是非扩展击键生成的一个字符。该标志可以与 emacs 以及 **ksh** 或 **ied** 的 emacs 命令行编辑器方式一起使用。它与生成扩展的单一字节字符，以及生成多字节 Asian 字符的 meta 键的 **** 正常使用相冲突。相关资源：kshMode。

+kshMode 指定不应该启用 **ksh** 方式。这是缺省行为。相关资源：kshMode。

-l 启用输出记录。启用记录后，所有从子进程收到的输出或者是记录到文件中，或者是记录到命令管道（这已通过 -lf 标志指定）。由于数据是从子进程直接记录的所以它就包含了由终端线路规范发送来的所有转义字符和回车或者是换行。通过转义序列可以启用或者禁用输出。相关资源：记录。

+l 禁用输出记录。关于输出记录的描述，参阅 -j。这个标志是缺省的。相关资源：记录。

-lf *file_name* 指定在 -l 标志中描述的输出记录的文件名。如果 *file_name* 以管道符号（|）开始，则字符串的其余部分就被看作是作为管道端点使用的一个命令。缺省文件名是 DttermLog*XXXXX*（其中 *XXXXX* 是 **dtterm** 的进程标识符），它是在 **dtterm** 开始的目录里创建的。如果最后五个字母是 *XXXXX*，就用进程标识符代替它们。*file_name* 指定了要用的日志文件名。相关资源：logFile。

-ls 指定启动的 shell 应该是一个登录 shell，例如 argv[0] 的第一个字符是短划线，意味着 shell 应该从系统的概要文件和用户的 **$HOME/.profile**（对 **ksh** 和 **sh** 而言）读取，或者是从系统的 **csh.login** 和用户的 **$HOME.login**（对 **csh** 而言）读取。相关资源：loginShell。

+ls 指定一个应该启动的通常（非登录）shell。这是缺省行为。相关资源：loginShell。

-map 说明如果 **dtterm** 尚未映射（已图标化），就应将其自身映射（脱去图标）到子进程输出。在 **dtterm** 未将其自身映射到子进程输出期间时间的初始化周期可以通过 mapOnOutputDelay 资源指定。相关资源：mapOnOutput。

+map 指定此处不能有特定的映射行为。这是缺省行为。相关资源：mapOnOutput。

-mb 表示当用户在右边空白处附近输入时，**dtterm** 应该鸣响边界铃。涉及到的实际距离通过 -nb 标志指定。相关资源：marginBell。

+mb 表示当用户在右边空白处附近输入时，边界铃不应鸣响。它是缺省值。相关资源：marginBell。

-ms *pointer_color* 说明给终端窗口（X11）的指针光标使用的前景色。缺省值是使用终端窗口的前景色。请参阅前景色。*pointer_color* 指定了所用的指针前景色。相关资源：pointerColor。

-name *prog_name* 指定 **dtterm** 窗口的 X11 名称。*prog_name* 使用的名称。

-nb *number* 如果边界空白响铃启用，指定从右边空白开始多少字符响铃会响。缺省值是 10。相关资源：nMarginBell。

-r 导致 **dtterm** 窗口前景色和背景色颠倒显示。这与 -rv 和 -reverse 标志是一致的。

+r 导致 **dtterm** 窗口以正常的前景色和背景色显示。这是缺省值，同时也与 +rv 标志一致。

-reverse 导致 **dtterm** 窗口前景色和背景色颠倒显示。这与 -r 和 -rv 标志是一致的。

-rv 导致 **dtterm** 窗口前景色和背景色颠倒显示。这与选择选项|全局选项是一致的，并且将“窗口背景”选项菜单变成“逆向”。以该标志开始的 **dtterm** 窗口，将“窗口背景”选项菜单设置成“逆向”。参阅“全局选项”。

+rv 导致 **dtterm** 窗口以正常的前景色和背景色显示。此为缺省设置。

-rw 说明应该启用反向环绕处理。相关资源：reverseWrap。

+rw 说明不应该启用反向环绕处理。它是缺省值。相关资源：reverseWrap。

-Sccn 说明终端仿真器应该逆着预先开放的 pty 或者流向装置。当 pty 或者流向设备的从属名属于 tty?? 形式，就要使用该标志。（例如，在 tty 后面恰好有两个字符）。该标志目的是在 **dtterm** 被计划性地从另一应用程序调用时使用。*cc* 指定当 pty 或者流向装置的从属名是 tty?? 形式时，该从属名最后两个字符。这个值是被忽略了，但必须在长度上恰巧是两个字符。*n* 指定相对于 pty 或流向装置已经公开的原版部分，文件描述符的数目。

-Sc.n 该标志与上面的 **-Sccn** 是一致的，但它是以更大的 pty 名称空间提供给系统。*c* 说明 pty 从属名的最后组成。这个值被忽略，可能是空值。*n*指定相对于 pty 已经公开的原版部分，文件描述符的数目。

-sb 说明应该显示一滚动条。它是缺省值。相关资源：滚动条。

+sb 说明不应该显示一滚动条。相关资源：滚动条。

-sf 说明 Sun 功能键转义码应该为功能键而不是标准 VT220 转义序列生成。相关资源：sunFunctionKeys。

+sf 说明标准转义序列应该为功能键而不是 Sun 功能键转义码生成。这是缺省行为。相关资源：sunFunctionKeys。

-sl*screens[s|l]* 说明在终端缓冲区内，超出窗口长度的行数。该标志值由一个数后面跟一个可选的后缀组成。如果不包含后缀，或者后缀是 **l**（ell），则终端缓冲区的总长度就是屏幕加上终端窗口的长度。如果后缀是 **s**（ess），终端缓冲区的总长度就是（屏幕数加一）倍的终端窗口长度。如果窗口调整得较大，**dtterm** 就会尽量维持缓冲区与窗口的比值不变。缺省值是 **4s**，*screens*说明了要保存的屏幕或者行的数目。相关资源：saveLines。

-ti *term_id* 提供用来为终端标识查询选择正确响应的名称。有效值为 vt100、vt101、vt102 和 vt220。缺省值是 vt220。*term_id* 说明要使用的终端标识符。

-title *title_string* 指定窗口标题。如果使用了 -e 标志，缺省值就是程序路径的最后组成部分。如果没有使用 -e 标志，缺省值就是用来运行 **dtterm**（例如，argv[0]）的名称的最后组成部分。*titel_string* 指定要用的标题。相关资源：标题。

-tm *term_modes* 说明包含终端设置关键字的一个字符串，以及可能绑定在一起的字符。允许的关键字包括 intr、quit、
erase、kill、eof、eol、swtch、start、stop、brk、susp、
dsusp、rprnt、flush、weras 以及 lnext。没有应用到特定结构的关键字都会被正确分析和忽略。控制字符可以用 ″^″，后面跟上字符（如 ″^c″ 或者 ″^u″）指定。而 ^? 可用来表示删除。这对于覆盖缺省终端设置是很有用的，而不必每次启动终端过程时进行 **stty**。缺省值是 NULL。*term_modes* 指定了终端方式字符串。相关资源：ttyModes。

-tn *term_name* 指定一个名称，设置 $TERM 环境变量。缺省值是 **vt220**。*term_name* 指定了要用的终端名称。相关资源：termName。

-usage 在屏幕上打印使用消息。

-vb 说明相对于听觉，更喜欢视觉铃声提示。不管何时收到 Control-G，都不响终端铃，而是闪烁窗口。相关资源：visualBell。

+vb 说明相对于视觉，更喜欢听觉铃声提示。这是缺省行为。相关资源：visualBell。

-w *border_width* 等同于 -borderwidth。*border_width* 用像素指定了窗口边界的宽度。

-xrm *resource_string* 允许在命令行指定 X11 资源管理器格式的资源。*resource_string* 指定了 X11 资源字符串。

资源

allowSendEvents 指定终端仿真器应该允许合成事件（它是由另一个应用程序生成和发送的）。启用该资源就可能承担一定的安全性风险。缺省值是 False。

appCursorDefault 如果是 True，光标键最初就在应用程序方式中。如果是 False它们初始在光标方式中。缺省值是 False。

appKeypadDefault 如果是 True，小键盘的键最初就在应用程序方式中。如果是 False它们最初就在数字方式中。缺省值是 False。

autoWrap 指定终端最初自动环绕处理是否启用。缺省值是 True。

background 指定终端窗口的背景色和用于滚动条的缺省背景色。在 CDE 下，该资源缺省为基本颜色设置选择像素或基本颜色设置背景像素，请参阅 backgroundIsSelect。缺省值是基本颜色设置背景像素。如果没有 CDE，则该资源缺省为 black。

backgroundIsSelect 如果是 true，则该资源指定终端窗口应该使用 Motif 选择色，而不是用于终端窗口背景的背景色。缺省值是 False。

blinkRate 指定光标在开和关状态闪烁时的毫秒数。如果该值是 250，即光标在每秒中闪烁两次。该值若是 0，即将闪烁关闭。缺省值是 250。

borderColor 给窗口定义边界颜色。当再生窗口管理器时，例如使用 **dtwm** 和 **mwm**，可能看不到窗口边界。缺省值是 “black”。

borderWidth 指定 shell 小窗口的边界宽度。通过再生窗口管理器（如 **dtwm** 和 **mwm**），可将此值覆盖。缺省值是 0。

c132 指定在 80 到 132 柱之间切换到窗口的 DECCOLM 转义序列是否应该得到肯定。缺省值是 False。

charCursorStyle 指定文本光标的形状。char_cursor_box 的值指定了光标和基本字体边界框中的宽度和高度。char_cursor_bar 的值指定了光标和基本字体边界框的宽度、两个像素的高度，以及在基线上顶部的拉伸。缺省值是 char_cursor_box。

consoleMode 指定在 **/dev/console** 指引的输出应该指引到终端窗口。它作为一种方式提供，来避免那些通常在 ITE 上显示的输出覆盖 X 服务器上的显示。它不作为将输出从任意系统 **/dev/console** 定向到任意 X 服务器的一种通用机制提供。注: 为使该标志起作用，您必须拥有 **/dev/console** 的所有权和读 / 写访问权。缺省值是 False。

foreground 指定用于滚动条的缺省前景色和用于指针光标的颜色以及终端窗口的前景色。在 CDE 下，该资源将缺省为基本前景色设置。否则缺省为 “white”。

geometry 指定期望的终端窗口的尺寸和位置。缺省大小是 24 行，每行 80 字符。没有缺省位置。

iconGeometry 指定期望的终端仿真器图标的位置。窗口管理器可能忽略这个值。没有缺省值。

iconic 如果是 true，指定终端仿真器最初应该放置在图标化的显示上。窗口管理器（包括 **dtwm** 和 **mwm**）可能忽略这个值。缺省值是 False。

iconicName 指定图标名称。如果使用了 -e 标志，缺省值就是程序路径的最后组成部分。如果没有使用 -e 标志，缺省值就是用来运行 **dtterm**（例如 argv[0]）的名称的基本名字。

jumpScroll 指定应当使用跳转滚动。在跳转滚动下屏幕一次可以滚动多行。在文本的很多行正发送到终端时，它就提供了更快的屏幕更新。最大可以跳转滚动的行数是由终端窗口中的行数限定的。可以保证所有行都会显示。缺省值是 True。

kshMode 指定应该启用 **ksh** 方式。在 **ksh** 方式下，通过扩展修饰符位设置按下的键将生成一个转义字符，后面是一个未扩展的击键生成的字符。该标志可以与 emacs 以及 **ksh** 或 **ied** 的 emacs 命令行编辑器方式使用。它与生成扩展的单一字节字符，以及生成多字节亚洲字符的元键的正常使用相冲突。缺省值是 False。

logFile 指定写有如下描述的输出日志的文件名。如果文件名以管道符号（|）开始，则字符串的其余部分就被看作是作为管道端点使用的一个命令。缺省文件名是 DttermLog*XXXXX*（其中 *XXXXX* 是一个独特的字符串），并且它是在启动子进程的目录内创建的。如果最后五个字符是 *XXXXX*，就用一个独特的字符串代替它们。

logging 启用输出记录。启用记录后，所有从子进程收到的输出或者是记录到文件中，或者是记录到命令管道（按照经由 logFile 标志指定的那样）。由于数据是从子进程直接记录的所以它就包含了由终端线路规范发送来的所有转义字符和回车或者是换行。通过转义序列可以启用或者禁用输出。缺省值是 False。

logInhibit 说明应该禁用设备和文件记录。缺省值是 False。

loginShell 指定启动的 shell 应该是一个登录 shell，例如 argv[0] 的第一个字符应该是短划线，意味着 shell 应该从系统的概要文件和用户的 **$HOME/.profile**（对 **ksh** 和 **sh** 而言）读取，或者是从系统的 **csh.login** 和用户的 **$HOME.login**读取（对 **csh** 而言）。缺省值是 False。

mapOnOutput 说明如果终端仿真器尚未映射（已图标化），则它就应该将其自身映射（脱去图标）到子进程输出。在它未将自身映射到子进程输出的最初时间，可由 mapOnOutputDelay 资源指定。缺省值是 False。

mapOnOutputDelay 启动后，指定 **dtterm** 不会接受 mapOnOutput 资源的秒数。对于最初的输出（例如 shell 提示），这允许不自动映射到窗口，而发送到终端。缺省值是 0（没有延迟）。

marginBell 指定当用户在靠近右边空白输入时，铃是否应该运行。缺省值是 False。

menuBar 指定应该显示的一个下拉菜单。缺省值是 True。

menuPopup 指定应该启用一个弹出菜单。缺省值是 True。

nMarginBell 如果空白响铃启用，指定从响铃处的右边空白开始多少字符铃响。缺省值是 10。

pointerBlank 指定指针光标应该置于空白方式。在这种方式，当指针移动，经过可选择的几秒钟，或者当击键输入时光标就会打开并封锁。延迟是经由 pointerBlankDelay 资源设置的。缺省值是 False。

pointerBlankDelay 在指针移动以后，定义在使指针光标封锁前等待的秒数。如果值是 0，仅当击键输入时调用指针封锁。缺省值是 2 秒。

pointerColor 说明给终端窗口指针（X11）的光标使用的前景色。缺省值是使用终端窗口的前景色。请参阅 foreground。

pointerColorBackground 说明给终端窗口指针（X11）的光标使用的背景色。缺省值是使用终端窗口背景色。参阅 background。

pointerShape 指定 X 光标字体字符，用作指针光标。它应该作为包含标题为 XC_removed 的文件的一个字符串被指定。缺省值为 **xterm**。

reverseVideo 指定是否应用反转视频。缺省值是 False。

reverseWrap 指定是否启用逆向环绕处理。缺省值是 False。

saveLines 说明在终端缓冲区内，超出窗口长度的行数。该值由一个数后面是一个可选的后缀构成。如果不包含后缀，或者后缀是 **l**（ell），则终端缓冲区的总长度就是屏幕加上终端窗口的长度。如果后缀是 **s**（ess），终端缓冲区的总长度就是（屏幕数加一）倍的终端窗口长度。如果窗口调整得较大，**dtterm** 就会尽量维持缓冲区与窗口的比值不变。缺省值为 **4s**。

scrollBar 指定滚动条是否可见。缺省值是 True。

sunFunctionKeys 说明 Sun Function Key 转义码是否应该为功能键而不是标准 VT220 转义序列生成。缺省值是 False。

termId 提供用来为终端标识查询选择正确响应的名称。有效值为 vt100、vt101、vt102 和 vt220。缺省值是 vt220。

termName 为 $TERM 环境变量定义名字。缺省值是 vt220。

title 指定窗口标题。如果使用了 -e 标志，缺省值就是程序路径的最后组成部分。如果没有使用 -e 标志，缺省值就是用来运行 **dtterm**（例如，argv[0]）的名称的最后组成部分。

ttyModes 说明包含终端设置关键字的一个字符串，以及可能绑定在一起的字符。允许的关键字包括：intr、quit、
erase、kill、eof、eol、swtch、start、stop、brk、susp、dsusp、
rprnt、flush、weras 和 Inext。没有应用到特定结构的关键字都会被正确分析和忽略。控制字符可以指定为 ^， 后面跟上字符（如 ^c 或者 ^u），而 ^? 可用来表示删除。这对于覆盖缺省终端设置是很有用的，而不必每次启动终端过程时进行 **stty**。缺省值是 NULL。

userBoldFont 在显示黑体终端文本时，指定要用的 XmFontSet。应该将其指定为 Motif XmFontList。只支持字符或者单空格的字体。没有定义使用比例字体时的行为。基于 userFont 的 XLFD 名称，将生成缺省粗体字。如果该字体不可用，通过使用一个像素偏移量叠印用户字体，就会生成黑体文本。

userFont 在显示终端文本时，指定一个要用的 XFontSet。应该将其指定为 Motif XmFontList。只支持字符或者单空格的字体。没有定义使用比例字体时的行为。该字体不会用来显示非终端文本，例如菜单栏、弹出菜单和对话框等。. 缺省值是使用父公告牌的 XmNtextFontList 值（参阅 XmBulletinBoard(3x)）它与 XmText 小窗口采用的方式相同。

visualBell 说明相对于听觉，更喜欢视觉提示。不管何时收到 CTRL-G，都不响终端铃声，而是闪烁窗口。缺省值是 False。

指针使用

注: **dtterm** 允许您选择文本区域。选择是基于 Inter-Client Communication Conventions Manual（ICCCM）中指定的模型进行的。**dtterm** 只支持基本的选择。通过基本的转换，可以复制或者粘贴所选择的文本。输入被当作键盘输入，在光标处插入。下面描述了选择／插入操作以及它们的缺省赋值。

select 左边按钮用来选择需要复制的文本。将指针移动到要复制文本的开头按住左边按钮，将光标移动到要复制文本末尾，然后松开按钮。任何当前选择的文本可以通过单击左键一次取消选择，不需要移动鼠标。

insert 中间按钮从基本选择粘贴文本，将其当作键盘输入。

操作

bell *([Percentage])* 该操作使得键盘铃在基础音量之上或之下指定百分比范围内鸣响。

break ( ) 该操作向子进程发送中断信号。

cancel ( ) 该操作向子进程发送 **CAN**（cancel）字符。

do ( ) 该操作向子进程发送与 **Do** 键相关的转义序列。

edit-key*(string)* 该操作向子进程发送与相应编辑键相关的转义序列。这些键的解释是特定于应用程序的。字符串的有效值是 find、insert、next、prior、remove 以及 select。

extend-start ( ) 开始扩展目前选择的文本。extend-end ( )

注: 扩展当前选择。选择文本数量取决于鼠标单击次数。

function-key-execute *(num [,type])* 该操作向子进程发送与相应功能键 *num* 相关的转义序列。对 *num* 的有效值是 1 到 35。如果类型设置成函数（或者干脆就没有设置）就会将与功能键 *num* 相关的转义序列发送到子进程。如果将 *type* 设置成 **UDK**，则与用户定义键 *num* 相关的字符串就被发送到子进程。

grab-focus ( ) 该操作依据多次鼠标单击的次数，进行下面工作之一。单击一次取消任何选定的文本，并且在指针位置设置选择支点；单击两次选择一个单词；单击三次选择文本的一行；单击四次选中全部文本。

hard-reset ( ) 该操作将在终端仿真器上进行硬复位。

help ( ) 该操作向子进程发送与 DEC VT220 帮助键相关的转义序列。这些键的解释是特定于应用程序的。

keymap *(name)* 该操作动态定义了一个新的翻译表，它的资源名后缀是 Keymap（大小写很重要）。名字 “None” 恢复最初的翻译表。

keypad-key-execute *(string)* 该操作向子进程发送与相应键区键相关的转义序列。这些键的解释是特定于应用程序的。对 *string* 的有效值包括：f1-f4、
space、tab、enter、equal、multiply、add、separator、
subtract、decimal、divide 以及 0 - 9。

move-cursor *(direction)* 该操作向子进程发送与相应光标移动相关的转义序列。这些键的解释是特定于应用程序的。*direction* 的有效值包括：up、down、backward 和 forward。

redraw-display ( ) 该操作刷新文本窗口的内容。

scroll *(count [,units])* 如果计数小于零，该操作将显示存储向下滚动；如果 *count* 大于零，则向上滚动。滚动的行数依据 *count* 和 *units*。*units* 的有效值是 page、halfpage或者是 line。*units* 的缺省值是 line。

select-adjust ( ) 该操作扩展选择。选择文本数量取决于鼠标单击次数。

1 次单击 = char

2 次单击 = word

3 次单击 = line

4 次单击 = buffer

select-all ( ) 该操作选择所有文本。

select-page ( ) 该操作选择屏幕上所有文本。

self-insert ( ) 该操作向子进程发送与被按下的键相关的转义序列。

soft-reset ( ) 该操作进行终端软复位。

stop *(state)* 该操作切换、启动或者停止从子进程读取数据的过程。*state* 的有效值是 toggle、on 和 off。

string *(string)* 该操作插入指定文本 *string*，就像输入的一样。如果 *string* 包含空格或非文字数字字符，必须对其加引号。如果 *string* 以字符 0x 开始，则就将其解释为十六进制字符常数。

tab ( ) 该操作向子进程发送一个制表符。

visual-bell ( ) 该操作迅速闪烁窗口。

Virtual Bindings 对虚拟键的绑定是特定于供应商的。当 **dtterm** 小窗口有输入焦点时，虚拟绑定不能应用。关于绑定虚拟按钮和键的信息，请参阅 VirtualBindings。

文件

/usr/bin/diff 包含 **diff** 命令。

相关信息

《*操作系统与设备管理*》中的『文件』介绍了文件以及处理文件的方法。

《*操作系统与设备管理*》中的『输入和输出重定向』描述了操作系统如何处理输入和输出。

---

du 命令

用途

概述磁盘使用。

语法

du [**-a** | **-s**] [**-k**] [**-m**] [**-g**][**-l**] [**-r**] [**-x**] [**-H** | **-L**][*File ...*]

描述

du 命令显示用于文件的块的数量。如果指定的 *File* 参数实际上是一个目录，就要报告该目录内的所有文件。如果没有提供 *File* 参数，**du** 命令使用当前目录内的文件。

如果 *File* 参数是一个目录，则报告的块的数量便是分配到目录中文件以及分配到目录自身的块之和。

指定 **-a** 标志，报告个体文件中块数量。不管是否使用了 **-a** 标志，由 *File* 参数指定的个体文件总是要列出。

指定 **-s** 标志，报告用于所有指定文件和目录中所有文件的全部块。

块计数包括每个文件的间接块。块计数是通过 512 字节单位计算的，它与系统使用的集群大小无关。指定 **-k** 标志，通过 1024 字节单位计算块数。

注:

1. 具有多个链接的文件只为一个条目计数和书写。
2. 由于块计数只基于文件大小，所以在报告的块数中，未分配的块是没有包含进去的。
3. 如果 **du** 得不到文件属性，或者无法读取目录，它就报告一个错误，并且会影响命令的退出状态。

标志

-a 对于每个指定的文件，显示该文件的磁盘使用情况。对于每个指定的目录，显示该目录（包括所有子目录）中单独每个文件的磁盘使用情况。将该标志与 **-s** 标志进行对比。

-g 用 GB 单位计算块数，而不是用缺省的 512 字节单位。对磁盘使用情况的输出值要用浮点数，这是因为如果用字节为单位的话，值会非常大。

-H 如果在命令行指定了符号链接，**du** 命令将统计链接引用的文件或文件层次结构的大小。

-k 用 1024 字节单位计算块数，而不是用缺省的 512 字节单位。

-l 在文件链接和多链接之间均匀地分配块。根据缺省值，有两个或者更多链接的文件只计数一次。

-L 如果在命令行指定了符号链接或者在文件层次结构的遍历中多次遇到符号链接，则 **du** 命令应统计链接引用的文件或文件层次结构的大小。

-m 用 MB 单位计算块数，而不是用缺省的 512 字节单位。对磁盘使用情况的输出值要用浮点数，这是因为如果用字节为单位的话，值会非常大。

-r 报告不可访问的文件或者目录名。此为缺省设置。

-s 对于每个指定的文件，显示该文件的磁盘使用情况。对于每个指定的目录，显示该目录（包括所有子目录）中所有文件的总体磁盘使用情况。将该标志与 **-a** 标志进行对比。

-x 在评估文件大小时，只评估那些与*File* 参数指定的文件或者目录驻留在相同设备上的文件。例如，您可以指定一个在多个设备上包含文件的目录。这种情况下，**-x** 标志就为与目录驻留在相同设备的所有文件显示块的大小。

如果指定了全部 **-k**、**-m** 和 **-g** 标志，或者是其中任意两个，最后指定的那个起作用。用标志 **-m** 和 **-g** 输出磁盘使用情况就会近似成最接近的第二位十进制数。

退出状态

此命令返回以下出口值:

0 成功结束。

>0 发生错误。

示例

1. 要概述一个目录树及其每个子树的磁盘使用情况，请输入:

   ```
   du /home/fran
   ```

   这在 /home/fran 该标志与目录及其每个子目录中显示了磁盘块数。
2. 要通过 1024 字节块概述一个目录树及其每个子树的磁盘使用情况，请输入:

   ```
   du -k /home/fran
   ```

   这在 **/home/fran** 目录及其每个子目录中显示了 1024 字节磁盘块数。
3. 要通过 MB 磁盘块概述一个目录树及其每个子树的磁盘使用情况，请输入:

   ```
   du -m /home/fran
   ```

   这在 **/home/fran** 目录及其每个子目录中显示了 MB 磁盘块数（近似到最接近的第二位十进制数）。
4. 要通过 GB 块概述一个目录树及其每个子树的磁盘使用情况，请输入:

   ```
   du -g /home/fran
   ```

   这在 **/home/fran** 目录及其每个子目录中显示了 GB 磁盘块数（近似到最接近的第二位十进制数）。
5. 要显示每个文件磁盘使用情况，请输入:

   ```
   du  -a /home/fran
   ```

   这显示了包含在每个文件以及目录 /home/fran 的子目录中的磁盘块数。在目录旁的数字就是该目录树的磁盘使用情况。在常规文件旁的数字就是该文件单独的磁盘使用情况。
6. 要只显示一个目录树的全部磁盘使用情况，请输入:

   ```
   du  -s /home/fran
   ```

   -s 标志控制 **du** 命令，只显示 /home/fran 目录和其中包含的文件的磁盘使用情况的总和。通过缺省值，如果 **du** 命令无法读取一个文件或者目录，就显示一条错误消息。
7. 除了在 /home/fran 的遍历中找到的常规文件，还要显示所有符号链接引用的文件或文件层次结构的磁盘使用，请输入:

   ```
   du -L /home/fran
   ```
8. 要报告符号链接 mylink 引用的文件或文件层次结构的磁盘使用，请输入:

```
du -H mylink
```

文件

/usr/bin/du 包含命令**du**。

相关信息

df 命令。

《*操作系统与设备管理*》中的『目录』说明了如何处理目录和路径名。

《*操作系统与设备管理*》中的『文件』提供了有关如何处理文件的信息。

---

dump 命令

用途

转储命令选择对象文件的几个部分。

语法

dump { **-a -c -d -g -h -l -n -o -p -r -s -t -u -v -H -R -T** } [**-z***Name* [,*Number*] [**+z***Number*]] [**-t***Index* [**+t***Index*]] [**-X** {**32**|**64**|**32_64**}] *File ...*

注: 请勿在 **-z** *Name* 标识和 *Number* 参数之间添加空格。

描述

dump 命令转储指定的 *File* 参数的所选部分。命令 **dump** 可以接受对象文件、压缩文件，以及可执行文件。

标志

-a	转储每个规定的压缩文件成员的压缩头。
-c	转储字符串表。
-d	转储每个区域中的原始数据。
-g	在压缩文档的符号表中转储全局符号。
-h	转储区域头。
-l	转储行号信息。
-n	转储所有装载者区域的信息。
-o	转储每个可选头。
-p	取消打印报头。
-r	转储重定位信息。
-s	转储每个选择的原始数据。
-t	转储符号表条目。
-t *Index*	仅转储索引符号表条目，由 *Index* 参数设定。使用 **-t** 标志和 **+t** 标志来指定符号表条目的范围。
+t *Index*	转储符号条目的范围以 *Index* 参数为结尾。范围开始于第一个符号表条目或由 **-t** 标志指定的条目。
-u	在 *File* 参数的名称下加下划线。
-v	转储符号表示的信息而不是数值。除了标志 **-o** 和标志 **-s** 外，任何标志都可以和 **-v** 标志连用。
-z*Name*[,*Number*]	转储 *Name* 参数的行号条目或者以指定数字开始的行号条目范围。

+z*Number* 将所有行号都转储到 *Number* 参数中。

-H 转储装载者区的报头。标志 **-H** 只能应用于可执行文件。

-r 转储引导区的重定位条目。标志 **-R** 只能应用于可执行文件。

-t 转储装载者区的符号表条目。标志 **-t** 只能应用于可执行文件。

-X *mode* 指定了 **damp** 应进行检查的对象文件的类型。方式必须是以下几种之一:

32 只能处理 32 位的对象文件

64 只能处理 64 位的对象文件

32_64 处理 32 位和 64 位对象文件

确省是处理 32 位对象文件（忽略 64 位的文件）。*mode* 也可以设定为 **OBJECT_MODE** 的环境变量。例如，**OBJECT_MODE=64** 使 **dump** 仅处理 64 位的对象而忽略 32 位的对象。标志 **-X** 覆盖 **OBJECT_MODE** 变量。

示例

1. 要将字符串表转储到 a.out 文件，请输入:
   ```
   dump -c a.out
   ```
2. 要将 XCOFF 数据段的内容转储到标准输出，请输入:
   ```
   dump -d a.out
   ```
3. 要转储对象文件头，请输入:
   ```
   dump -o a.out
   ```
4. 要为 a.out 文件转储行号信息，请输入:
   ```
   dump -l a.out
   ```
5. 要为 a.out 文件转储重定位信息，请输入:
   ```
   dump -r a.out
   ```
6. 要转储 a.out 对象文件文本段中的内容，请输入:
   ```
   dump -s a.out
   ```
7. 要转储 a.out 对象文件的符号表信息，请输入:
   ```
   dump -t a.out
   ```
8. 要打印从 20 到 31 的符号表条目而不带头信息时，请输入:
   ```
   dump -p -t20 +t30 a.out
   ```
9. 要只从 lib.a 中的 64 位对象转储对象文件头时，请输入:
   ```
   dump -X64 -o lib.a
   ```

相关信息

ar 命令、**size** 命令。

a.out 文件、**ar** 文件。

dumpcheck 命令

用途

检查转储装置和拷贝目录能够接收系统转储。如果资源可能不足以容纳这些转储，缺省记录一个错误。

语法

/usr/lib/ras/dumpcheck [[**-l**] [**-p**] [**-t** *TimeParameters*] [**-P**]] | [**-r**]

描述

/usr/lib/ras/dumpcheck 命令用来检查系统转储使用的磁盘资源。如果最大转储设备不足以接收转储，或者转储调页空间时，在拷贝目录中空间不够，该命令就记录一个错误。

dumpcheck 通常是每天在当地时间下午 3 点由守护程序运行。当使用 **-r** 标志从根目录的 **crontab** 中删除或者使用 **-t** *TimeParameters*，来更改转储检查的运行时间时，这可能有所变化。它也可以由 SMIT 配置。当安装了服务帮助后，**dumpcheck** 就自动添加到根目录的 **crontab** 中。

要取得最大效力，应该是在系统负载最大时运行 **dumpcheck**。这些时候，系统的转储最有可能是它的最大值。同样，即使使用 **dumpcheck** 观察转储的大小，仍然可能发生转储在转储装置或者是发生时在拷贝目录中不合适。如果在转储时间刚好有一个系统负载高峰，这也可能发生。

dumpcheck 函数是作为服务帮助文件设置的一部分安装的，也是自动安装的。

标志

-l	在错误日志中记录所有警告。如果没有指定参数，这就是缺省值。
-p	打印所有到标准输出的警告。
-P	说明将进行永久的变化。即它们应用到 **dumpcheck** 工具的后继执行中。**-P** 标志不一定要和 **-t** 和 **-r** 标志在一起。如果指定了 **-P** 标志，**dumpcheck** 只是更改了 **crontab** 条目，而没有进行任何检查。
-r	为该函数删除 **crontab** 条目，有效地取消配置。该命令通常是由 **cron** 运行的。**-r** 标志必须单独指定。它与任何其他标志一起都是无效的。
-t *TimeParameters*	当 **dumpcheck** 执行时，修改时间。*TimeParameters* 标志必须包含在单引号或者双引号中。它指定了 **crontab** 时间参数，以及在 **crontab** 文件中一行的最初五个参数。参阅关于时间参数格式的 **crontab** 命令。**-t** 标志与 **-r** 标志一起是无效的。如果指定了 **-t** 标志，**dumpcheck** 只是更改了 **crontab** 条目，而没有进行任何检查。

安全性

该命令只能由 root 用户执行。

示例

1. 要检查转储资源，并且将结果打印到标准输出，而不是记录，请输入:

   ```
   /usr/lib/ras/dumpcheck -p
   ```

   要永久进行更改，即在 **crontab** 条目中进行，请输入:

   ```
   /usr/lib/ras/dumpcheck -p -P
   ```

2. 要使得 **dumpcheck** 从星期一到星期五在上午 9:00 和下午 3:00 运行，请输入:

   ```
   /usr/lib/ras/dumpcheck -t "0 9,15 * * 1-5"
   ```

   要返回到缺省值，请输入:

   ```
   /usr/lib/ras/dumpcheck -t "0 15 * * *"
   ```

   当运行 **dumpcheck** 时，也可以用 SMIT 来配置时间。

3. 要停止运行该功能，请输入:

```
/usr/lib/ras/dumpcheck -r
```

也可以将 SMIT 应用到这项任务。

相关信息

sysdumpdev 命令。

AIX 5L Version 5.3 Kernel Extensions and Device Support Programming Concepts 中的 System Dump Facility。

---

dumpfs 命令

用途

转储文件系统信息

语法

dumpfs { *FileSystem* | *Device* }

描述

dumpfs 命令打印出指定的文件系统或特定设备的超级块、i-node 映射和磁盘映射信息。该列表用来查找文件系统信息。首先，**dumpfs** 命令是用于调试的。

命令 **dumpfs** 也可以遇到 JFS2 抽点转储。命令 **dumpfs** 将打印出指定抽点转储中的超级块、抽点转储映射以及块映射树的副本。

注: **dumpfs** 不能在 UDF、NFS 或 JFS 软盘上运行。

示例

打印 **/dev/hd4** 的信息，输入:

```
dumpfs /dev/hd4
```

相关信息

命令 **fsck mkfs** 命令。

---

echo 命令

用途

将字符串写到标准输出

语法

echo [*String* ...]

描述

命令 **echo** 将字符串写到标准输出中。多个字符串间由空格隔开，指定的最后一个 *String* 参数后跟有换行符。如果没有指定 *String* 参数，将显示一空白行（换行符）。

通常您可以通过 —（双连字符）来区分标志和一个以连字符开头的字符串。既然标志不被 **echo** 命令所支持，那么 —（双连字符）将以字面意义处理。

命令 **echo** 能够识别以下转义约定:

\a 显示警告字符。
\b 显示退格符。
\c 在输出中禁止另外跟在最终参数后面的换行字符。所有跟在 **\c** 序列后的字符都被忽略。
\f 显示走纸字符。
\n 显示换行字符。
\r 显示一个回车字符。
\t 显示制表符。
\v 显示垂直制表符。
**** 显示反斜杠符号。
\0数字 显示一个 ASCII 值为 0、1、2、3 位八进制数的八位字符。

> **注:** 命令 **bsh**、**ksh** 和 **csh** 各有一个内置的 **echo** 子命令。命令 **echo** 和命令 **bsh** 以及 **ksh echo** 子命令的工作方式是相同的。子命令 **csh echo** 的工作方式不同于 **echo** 命令。有关 **echo** 子命令的信息，请参阅《*操作系统与设备管理*》中的『Bourne shell 内置命令』、『Korn shell 或 POSIX shell 的常规内置命令描述』以及『C shell 内置命令』。

\（反斜杠）在 shell 中是一个引号字符。这意味着，除非 \ 和一个转义字符连用或者被引号括起（例如 ″\″ 或 '\'），否则当命令扩展时 shell 将除去反斜杠。

shell 扩展后，命令 **echo** 根据输入中的转义序列写到输出中。根据反斜杠简化表，比较反斜杠是如何在命令中首先由 shell 然后又由 **echo** 命令缩减的:

反斜杠缩减		
输入命令	**在 shell 扩展后**	**处理完 echo 命令后**
echo hi\\\\there	echo hi\\there	hi\there
echo 'hi\\\\there'	echo 'hi\\\\there'	hi\\there
echo ″hi\\\\there'	echo ″hi\\there″	hi\there

退出状态

此命令返回以下出口值:

0 成功结束。
>0 发生错误。

示例

1. 要将一个消息写到标准输出，请输入:

   ```
   echo Please insert diskette . . .
   ```

2. 要显示一个含有特殊字符的消息，请输入:

   ```
   echo "\n\n\nI'm at lunch.\nI'll be back at 1:00."
   ```

   该命令将跳过三行，然后显示报文如下:

   ```
   I'm at lunch.
   I'll be back at 1:00.
   ```

注: 如果报文中含有转义序列，则必须将它放在引号中。否则的话，shell 将会将 \（反斜杠） 翻译为一个元字符，并且用不同的方式处理。

3. 要使用 **echo** 命令的模式匹配字符功能，可以输入:

```
echo The back-up files are: *.bak
```

该用法将显示消息备份文件后跟有当前目录中以 .bak 结尾的文件名。

4. 如果想要在一个文件中添加单独的一行文本，可以输入:

```
echo Remember to set the shell search path to $PATH. >>notes
```

在 shell 代替了 shell 变量 **PATH** 的值以后，该用法会将消息添加到文件 notes 的末尾。

5. 要将报文写到标准错误输出，请输入:

```
echo Error: file already exists. >&2
```

这个命令将错误信息重定向到标准错误。如果 >&2 被省略了，则将报文将被写入标准输出中。

File

/usr/bin/echo 包含有 **echo** 命令。

相关信息

bsh 命令、**csh** 命令、**ksh** 命令、**printf** 命令。

《*操作系统与设备管理*》中的『输入和输出重定向』描述了操作系统如何处理输入和输出，以及如何使用重定向和管道符号。

《*操作系统与设备管理*》中的『Shell』描述了何为 shell、shell 的不同类型，以及 shell 如何影响解释命令的方式。

---

ed 或 red 命令

用途

用于文本文件的行编辑器。

语法

ed [**-p** *String*] [**-s** | **-**] [*File*]

red [**-p***String*] [**-s** | **-**] [*File*]

描述

命令 **ed** 启动 ed 编辑器行编辑程序。ed 编辑器一次只能作用于一个文件，它将文件拷贝到临时的编辑缓冲区中，对该副本进行更改。ed 编辑器属于一类也包含了edit 编辑器、ex 编辑器和 vi 编辑器的编辑器族。ed 编辑器在缓冲区中做出您所指定的更改。它直到您使用 write（**w**）子命令时才会更改文件本身。

当您使用 **ed** 命令启动 ed 编辑器时，您可以指定您想要编辑的文件的名称，或者您可以使用 **e** 子命令。当 **ed** 命令向缓冲区中读入新的文件时，该文件的内容将替代该缓冲区以前的内容。

命令 **red** 是命令 **ed** 的受限版本，是和受限的 shell（**rsh**）一起使用。使用 **red** 命令，您就只能编辑当前目录或 **/tmp** 目录中驻留的文件；您不能使用 **!**！（感叹号）提示符。

由 0 个、 1 个或 2 个地址，后接一个单字母子命令，并且再在这个子命令后接上其可选参数，这样就组成了一条 ed 编辑器的子命令。这些地址指定缓冲区内的一行或几行。因为每个子命令都有缺省地址，所以通常都不必指定地址。

ed 编辑器允许只编辑当前行，除非您指定缓冲区内的另一行。您只能移动和拷贝整行的数据。ed 编辑器对于编辑大文件或者对于在 shell 程序中进行编辑很有用。

ed 编辑器以下面的两种方式之一进行操作:

命令方式 在命令方式中，ed 编辑器识别并运行子命令。当您启动 ed 编辑器时，它处于命令方式。输入 **.**（句点）并按 Enter 键来确认您处于命令方式。

文本输入方式 在文本输入方式中，ed 编辑器允许您往文件缓冲区中输入文本，但是不识别子命令。请您用 **a**子命令、**c** 子命令，或 **i** 子命令进入文本输入方式。通过在行的开头单独输入一个 **.**（句点）以退出文本输入方式并返回命令方式。要在文本输入方式下将 **.**（句点）放入缓冲区中，请输入一个字符，后面跟上 **.**（点）。然后，退出文本输入方式并使用 **s** 子命令除去该字符。

以下列表提供了 **ed** 编辑器的最大限制。

- 每个文件名最多 64 个字符
- 每个全局子命令列表最多 256 个字符
- 缓冲区大小为 128,000 个字符

注: 该缓冲区包含了初始文件和编辑信息。

最大行数取决于可用的内存大小。最大的文件大小取决于可用的物理数据存储量（磁盘或磁带机），或是取决于用户内存中允许驻留的最大行数。

标志

-p *String* 将参数 *String* 设为编辑器的提示符。*String* 的缺省值是空值（无提示符）。

-s 用子命令 **e**、子命令 **r** 和子命令 **w** 禁止编辑器显示的字符计数。该标志还禁止显示 **e** 子命令和 **q** 子命令的诊断消息，并且禁止显示 **!** 子命令后的 !（惊叹号）提示符。

- 提供和 **-s** 标志一样的功能。

模式匹配

ed 编辑器支持有限格式的专门的模式匹配字符，这样，您就能将这些字符用作正则表达式（RE）来构造模式字符串。您可以用地址中的这些模式指定几行，以及在某些子命令中指定行的某些部分。

正则表达式

下面的 RE 与单个字符或者整理元素相匹配，如下所示:

Character 与自身匹配，并且可以是任何的普通字符（而不是专门的模式匹配符号之一）。

. 和换行字符之外的任何单个字符匹配。

[*String*]　和字符串中的任何一个字符匹配。某些模式匹配字符在括号中具有专门的意义，如下所示:

^　与除参数 *String* 中的字符和换行字符之外的任何字符相匹配，如果参数 *String* 的第一个字符是 ^（音调符号）的话。这个条件仅当 ^ 是字符串中的第一个字符，[^*String*] 时为 true。

-　根据当前的整理顺序指示某个范围的连续 ASCII 字符。例如，[a-f] 等价于 [abcdef] 或 [aAbBcCdDeEfF] 或 [abcdef] 并且可以包括带着重音的 a 和 e 字符。整理顺序可以给字符定义等价类。

如果减号作为字符串 [-*String*] 中的首字符；或者如果它紧跟在作为首字符的音调符号之后，[^-*String*]；或者如果它充当字符串的最后一个字符，则 [*String*-]，它就失去了其本身的意义。

]　当]（右括号）作为字符串的首字符，[]*String*]，或者当它紧跟在作为首字符的音调符号之后 [^]*String*]，它就作为该字符串的一部分而不是该字符串的终止符。

构造模式

下面的规则描述了如何从 RE 构造模式:

- 由单个的普通字符组成的 RE 与字符串中相同的字符相匹配。
- 后面紧接 *（星号）的 RE 匹配于 0 次或多次重复的、与该 RE 匹配的字符。例如下面的模式:

```
ab*cd
```

和下面的每个字符串相匹配:

```
acd
abcd
abbcd
abbbcd
```

但不匹配下面的字符串:

```
abd
```

如果存在一个选项，选择最长、最靠左的匹配字符串。例如，给定下面的字符串:

```
122333444
```

模式 .* 与 122333444 匹配，模式 .*3 与 122333 匹配，而模式 .*2 与 122 匹配。

- RE 后面紧跟着:

\{*m*\}　*精确*匹配 *m* 次出现的、与该 RE 匹配的字符。
\{*m*,\}　与*至少* *m* 次出现的、与该 RE 匹配的字符相匹配。
\{*m*,*n*\}　与重复从 *m* 到 *n*（含 n）之间*任意次数*的、与该 RE 匹配的字符相匹配。

数字 *m* 和 *n* 必须是从 0 到 255（含 255）的整数。只要存在选项，这个模式就匹配尽可能多的出现。

- 您可以将几个 RE 组合到模式中，这些模式与包含了相同字符序列的字符串匹配。例如，模式 AB*CD 匹配字符串 AB*CD 模式 [A-Za-z]*[0-9]* 匹配任何含有字母（包含无）组合、并紧跟任何数字（包含无）组合的字符串。
- 字符序列 \（*Pattern*\） 标志着一个子模式，如果这个子模式没有封闭，它就与该序列匹配同样的字符串。
- 字符 *Number* 匹配的字符串与该模式中某个子模式先前匹配的字符串相同（见前一条规则）。参数 *Number* 的模式代表一个阿拉伯数字。模式 *Number* 匹配的字符串被参数 *Number* 所指定（从左往右计数）的子模式的具体值所匹配。

例如下面的模式:

```
\(A\)\(B\)C\2\1
```

匹配字符串 ABCBA。您可以嵌套子模式。

限制模式匹配内容

您可以限制模式，让它只匹配一行中的第一段、最后一段或者整行。空模式（//（两个斜杠）），是重复上一个模式。

匹配行的第一段: 参数 ^*Pattern* 只匹配从行的首字符位置开始的字符串。

匹配行的最后一段: 参数 *Pattern*$ 只匹配以行的最后一个字符（不包括换行字符）为结尾的字符串。

匹配整行: 参数 ^*Pattern*$ 限制该模式，使其匹配整行。

寻找行

ed 编辑器使用三种类型的地址：行号地址、相对当前行的地址和模式地址。当前行（通常是子命令所作用的最后一行）是缓冲区中的参考点。

您可以用行号地址来做下面的工作:

- 指定新的当前行
- 显示指定的一行或几行
- 让某个命令作用于某一行或某几行

不接受地址的子命令将地址的出现认为是错误。接受地址的子命令可以使用赋予或缺省的地址。如果提供的地址过多，命令就使用最后（最右边）的那些地址。

在大多数情况下，用逗号（,）将地址分隔（例如 2,8）。分号（;）也可以分隔地址。地址之间的分号使得 ed 编辑器设置当前行为第一个地址，然后计算第二个地址（例如，为搜索设置起始行）。在一个地址对中，第一个地址的数值必须小于第二个地址。

您可以使用行号和符号地址来执行下面的任务:

- 寻址当前行
- 用数字寻址行
- 寻址第一行之前的行
- 寻址最后一行
- 寻址指定行上面一行
- 寻址指定行下面一行
- 寻址第一行到最后一行
- 寻址当前行到最后一行
- 寻址一组行
- 寻址含有指定模式的下一行
- 寻址含有指定模式的上一行
- 寻址标记过的行

寻址当前行

一个 .（句点）寻址当前行。.（句点）是多数 ed 编辑器子命令的缺省值，并且不需要进行指定。

按数字寻址行

要寻找缓冲区中指定的一行，请输入:

```
Number
```

其中参数 *Number* 代表行号。示例:

```
2253
```

寻址行号 2253 的行作为当前行。

寻址第一行之前的行

要寻找缓冲区内第一行之前的行，请输入:

```
0
```

寻找最后一行

要寻找缓冲区的最后一行，请输入:

```
$
```

寻找指定行上面的行

要指定当前行之上指定数目行的一个地址，请输入:

```
-Number
```

其中参数 *Number* 指的是您想要寻找的行是当前行之上的第多少行。示例:

```
-5
```

寻找当前行之上的第 5 行，并作为新的当前行。

您也可以只指定一个 - 来寻找当前行之上紧接当前行的一行。减号具有累计效果。例如，地址 - -（两个减号）寻找当前行之上两行的行。

寻找指定行之下的行

要指定当前行之下指定数目行的地址，请输入:

```
+Number
```

其中参数 *Number* 指的是您想要寻找的行是当前行之下第多少行。+（加号）是可选的。示例:

```
+11
```

寻找当前行之下的第 11 行，并将其作为新的当前行。

您也可以只指定一个 + 来寻找当前行之下紧接当前行的一行。+ 具有累计效果。例如，地址 + +（两个加号）寻找当前行之下两行的行。

寻址第一行到最后一行

要寻址第一行到最后一行，请输入:

```
,
```

,（逗号）代表了地址对 1,$（从第一行到最后一行）。第一行成为当前行。

寻址当前行到最后一行

要寻址当前行到最后一行，请输入:

```
;
```

;（分号）代表了地址对 .,$（从当前行到最后一行）。

寻址一组行

要寻址一组行，请输入:

```
FirstAddress,LastAddress
```

其中，参数 *FirstAddress* 是您想要寻址的组中的第一行的行号（或符号地址），而参数 *LastAddress* 是该组最后一行的行号（或符号地址）。该组中的第一行成为当前行。示例:

```
3421,4456
```

从行 3421 寻址到行 4456。行 3421 成为当前行。

寻址包含指定模式的下一行

要寻找含有匹配字符串的下一行，请输入:

```
/Pattern/
```

其中，参数 *Pattern* 是字符串或正则表达式。从当前行之后的行开始搜索，当找到该模式的匹配时停止。如果必要的话，搜索到达缓冲区的末尾，环绕到缓冲区的开始，并且继续搜索直到找到匹配或者返回到当前行。示例:

```
/Austin, Texas/
```

寻址含有 Austin、Texas 的下一行，并将其作为当前行。

寻址包含指定模式的上一行

要寻址包含模式匹配的上一行，请输入:

```
?Pattern?
```

其中，参数 *Pattern* 是字符串或正则表达式。**?***Pattern***?** 结构，和 **/***Pattern***/** 一样，可以搜索整个缓冲区，但是它往相反的方向搜索。示例:

```
?Austin, Texas?
```

寻址含有 Austin、Texas 的上一行，并将其作为当前行。

寻址标记过的行

要寻址用子命令 **k** 标记过的行，请输入:

```
'x
```

其中参数 *x* 是从 *a* 到 *z* 的一个小写字母。例如:

```
'c
```

是用子命令 **k** 寻找标记为 c 的行。

子命令

用 ed 编辑器子命令执行下面的操作:

- 编辑文件
- 处理文件
- 执行其他功能
  - 更改提示字符串
  - 输入系统命令
  - 退出 ed 编辑器
  - 请求帮助

在大多数情况下，您只能在一行上输入一条 ed 编辑器子命令。但是，可以将 **l**（列表）和 **p**（打印）子命令添加到除 **e**（编辑）、**E**（编辑）、**f**（文件）、**q**（退出）、**Q**（退出）、**r**（读）、**w**（写）和 **!**（操作系统命令）子命令之外的任何子命令。

子命令 **e**、**f**、**r** 和 **w** 接受文件名作为参数。ed 编辑器存储子命令使用过的最后一个文件的名称，并将其作为缺省文件名。如果不赋予文件名，下一个子命令 **e**、**E**、**f**、**r** 或 **w** 就使用该缺省的文件名。

ed 编辑器在出现错误的情况下给出两个消息之一: **?**（问号）或 `?File`。当 ed 编辑器收到中断信号（Ctrl-C 按键顺序）时，它显示一个 **?** 并返回命令方式。当 ed 编辑器读文件时，它废弃 ASCII 空字符和最后一个换行字符之后的所有字符。

编辑文件

您可以使用 ed 编辑器的子命令执行下面的任务:

- 添加文本
- 更改文本
- 拷贝文本
- 删除文本
- 显示文本
- 合并和分割行
- 进行全局更改
- 标记文本
- 移动文本
- 保存文本
- 搜索文本
- 替换文本
- 撤销文本更改

注: 在下面对 ed 编辑器子命令的描述中，缺省地址如括号中所示。请勿输入括号。地址 .（句点）指的是当前行。新的空行第一个位置出现的 .（句号）是返回命令方式的信号。

添加文本

(.)**a** [**l**] [**n**] [**p**] *Text***.**

子命令 **a**（添加）将文本添加到缓冲区中指定行之后。子命令 **a** 设置最后插入的行为当前行，或者，如果没有插入行的话，就设置为指定的行。0 地址将文本添加到缓冲区的起点。

如果您想要显示所添加的文本，请输入可选的子命令 **l**（列表）、**n**（行号）或 **p**（打印）。

输入您的文本，并在每一行的末尾按 Enter 键。如果您不在每一行的结束处按 Enter 键，ed 编辑器就在您的字符输满一行后自动将光标移到下一行。ed 编辑器将您按 Enter 键之前的所有字符认为是一行，不管这些字符在屏幕上占据了多少行。

在您输完所有文本后，请在新行的起始处输入一个 .（句点）。

(.)(.)**i** [**l**] [**n**] [**p**]*Text***.**

子命令 **i**（插入）将文本插入到指定行之*前*，并将当前行设为最后插入的一行。如果没有插入任何行，子命令 **i** 将指定行设为当前行。您不能对这个子命令使用 0 地址。

如果您想要显示插入的文本，请输入可选的子命令 **l**（列表）、**n**（行号）或 **p**（打印）。

输入您的文本，并在每一行的末尾按 Enter 键。如果您不在每一行的结束处按 Enter 键，ed 编辑器就在您的字符输满一行后自动将光标移到下一行。ed 编辑器将您按 Enter 键之前的所有字符认为是一行，不管这些字符在屏幕上占据了多少行。

在您输完所有文本后，请在新行的起始处输入一个 .（句点）。

注: 只有文本的放置上，子命令 **i** 与子命令 **a** 不同。

您可以使用不同的 ed 编辑器子命令在不同的位置添加文本。请使用前面的格式来执行下面的编辑任务:

- 在当前行之后添加文本
- 在当前行之前添加文本
- 在指定行之后添加文本
- 在指定行之前添加文本
- 在包含搜索模式的行之后添加文本
- 在包含搜索模式的行之前添加文本
- 在不包含搜索模式的行之后添加文本
- 在不包含搜索模式的行之前添加文本

在当前行之后添加文本:

1. 请输入下面的子命令:

   ```
   a[l][n][p]
   ```

   其中 **l**、**n** 和 **p** 是显示添加文本的可选子命令。
2. 请输入文本，并按 Enter 键。
3. 请输入一个 **.**（句点），并再次按 Enter 键以返回命令方式。

在当前行之前添加文本:

1. 请输入下面的子命令:

   ```
   i[l][n][p]
   ```

   其中 **l**、**n** 和 **p** 是显示添加文本的可选子命令。

2. 请输入文本，并按 Enter 键。
3. 请输入一个 **.**（句点），并再次按 Enter 键以返回命令方式。

在指定行之后添加文本:

1. 请输入下面的子命令:

   ```
   Addressa[l][n][p]
   ```

   其中参数 *Address* 是行号，插入的文本将在该行之后。可选的子命令 **l**、**n** 和 **p** 显示所添加的文本。
2. 请输入文本，并按 Enter 键。
3. 请输入一个 **.**（句点），并再次按 Enter 键以返回命令方式。

在指定行之前添加文本:

1. 请输入下面的子命令:

   ```
   Addressi[l][n][p]
   ```

   其中参数 *Address* 是行号，插入的文本应该在该行之前。可选的子命令 **l**、**n** 和 **p** 显示所添加的文本。
2. 请输入文本，并按 Enter 键。
3. 请输入一个 **.**（句点），并再次按 Enter 键以返回命令方式。

在包含搜索模式的行之后添加文本:

1. 请输入下面的子命令:

   ```
   [Address]g/Pattern/a[l][n][p]
   ```

   其中 *Address* 是可选的参数，它指定了搜索 *Pattern* 参数中指定的模式的行的范围。参数 *Pattern* 是字符串或正则表达式。如果您省略了参数 *Address*，ed 编辑器就在整个文件中搜索包含模式的行。可选的子命令 **l**、**n** 和 **p** 显示所添加的文本。
2. 请输入反斜杠:

   ```
   \
   ```
3. 请输入文本。要在所添加的文本中开始新的行，请输入反斜杠:

   ```
   \
   ```

   并按 Enter 键。您所输入的文本将被添加到包含了命令所指定的模式的每一行之后。
4. 要返回命令方式，请按 Enter 键。

在包含搜索模式的行之前添加文本:

1. 请输入下面的子命令:

   ```
   [Address]g/Pattern/i[l][n][p]
   ```

   其中 *Address* 是可选的参数，它指定了搜索 *Pattern* 参数中指定的模式的行的范围。参数 *Pattern* 是字符串或正则表达式。如果您省略了参数 *Address*，ed 编辑器就在整个文件中搜索包含模式的行。可选的子命令 **l**、**n** 和 **p** 显示所添加的文本。
2. 请输入反斜杠:

   ```
   \
   ```
3. 请输入文本。要在所添加的文本中开始新的行，请输入反斜杠:

   ```
   \
   ```

   并按 Enter 键。您输入的文本将被添加到包含了命令中所指定的模式的每一行之前。

4. 要返回命令方式，请按 Enter 键。

在不包含搜索模式的行之后添加文本:

1. 请输入下面的子命令:

   ```
   [Address]g/Pattern/a[l][n][p]
   ```

   其中 *Address* 是可选的参数，在它指定范围内的行中，搜索不包含 *Pattern* 参数所指定的模式的行。参数 *Pattern* 是字符串或正则表达式。如果您省略参数 *Address*，ed 编辑器就在整个文件中搜索不包含该模式的行。可选的子命令 **l**、**n** 和 **p** 显示所添加的文本。
2. 请输入反斜杠:

   ```
   \
   ```
3. 请输入文本。要在所添加的文本中开始新的行，请输入反斜杠:

   ```
   \
   ```

   并按 Enter 键。您所添加的文本被添加到不包含命令所指定的模式的每一行之后。
4. 要返回命令方式，请按 Enter 键。

在不包含搜索模式的行之前添加文本:

1. 请输入下面的子命令:

   ```
   [Address]g/Pattern/i[l][n][p]
   ```

   其中 *Address* 是可选的参数，在它指定范围内的行中，搜索不包含 *Pattern* 参数所指定的模式的行。参数 *Pattern* 是字符串或正则表达式。如果您省略了参数 *Address*，ed 编辑器就在整个文件中搜索不包含该模式的行。可选的子命令 **l**、**n** 和 **p** 显示所添加的文本。
2. 请输入反斜杠:

   ```
   \
   ```
3. 请输入文本。要在所添加的文本中开始新的行，请输入反斜杠:

   ```
   \
   ```

   并按 Enter 键。您输入的文本被添加到不包含命令中所指定的模式每一行之前。
4. 要返回命令方式，请按 Enter 键。

更改文本

(.,.)(.,.)**c** [**l**] [**n**] [**p**]*Text*.

子命令 **c**（更改）删除指定的、您想要替换的行，然后用您输入的新行替代它们。子命令 **c** 将新输入的最后一行设为当前行，或者，如果没有输入的话，就将没有删除的第一行设为当前行。

如果您想要显示插入的文本，请输入可选的子命令 **l**（列表）、**n**（行号）或 **p**（打印）。

请输入新的文本，并在每行的结束按 Enter 键。当您输完所有的新文本时，在行上输入一个 .（句点）。

您可以用 ed 编辑器以几种不同的方式更改文本。请使用前面的格式来执行下面的编辑任务:

- 更改当前行的文本
- 更改一行或一组行的文本
- 更改包含了指定模式的行的文本

- 更改不包含指定模式的行的文本

更改当前行的文本:

1. 请输入下面的子命令:

   ```
   c[l][n][p]
   ```

   其中 **l**、**n** 和 **p** 是显示更改的文本的可选子命令。
2. 请输入文本，并按 Enter 键。
3. 请输入一个 **.**（句点），并再次按 Enter 键以返回命令方式。

更改一行或一组行的文本:

1. 请输入下面的子命令:

   ```
   Addressc[l][n][p]
   ```

   其中参数 *Address* 是要更改的一行或一组行的地址。可选的子命令 **l**、**n** 和 **p** 显示更改的文本。
2. 请输入文本，并按 Enter 键。
3. 请输入一个 **.**（句点），并再次按 Enter 键以返回命令方式。

更改包含了指定模式的行的文本:

1. 请输入下面的子命令:

   ```
   Addressg/Pattern/c[l][n][p]
   ```

   其中参数 *Address* 是一组行的地址，您想要在这组行中搜索参数 *Pattern* 所指定的模式。可选的子命令 **l**、**n** 和 **p** 显示更改的文本。
2. 请输入反斜杠:

   ```
   \
   ```
3. 请输入新的文本。要在新的文本中开始新的行，请输入反斜杠:

   ```
   \
   ```

   并按 Enter 键。
4. 要返回命令方式，请再次按 Enter 键，输入一个 .（句点），并再次按 Enter 键。

更改不包含指定模式的行的文本:

1. 请输入下面的子命令:

   ```
   Addressv/Pattern/c[l][n][p]
   ```

   其中参数 *Address* 是一组行的地址，您想要在这组行中搜索参数 *Pattern* 所指定的模式。可选的子命令 **l**、**n** 和 **p** 显示更改的文本。
2. 请输入反斜杠:

   ```
   \
   ```
3. 请输入新的文本。要在新的文本中开始新的行，请输入反斜杠:

   ```
   \
   ```

   并按 Enter 键。
4. 要返回命令方式，请再次按 Enter 键，输入一个 .（句点），并再次按 Enter 键。

拷贝文本

(.,.)(.,.)**t***Address* [**p**] [**l**] [**n**]

子命令 **t**（转移）将指定行的副本插入到参数 *Address* 所指定的行之后。子命令 **t** 接受 0 地址，认为是将行插入到缓冲区的起始处。

子命令 **t** 将拷贝的最后一行设为当前行。

如果您想要显示转移的文本，请输入可选的子命令 **l**（列表）、**n**（行号）或 **p**（打印）。

拷贝一行或一个行的集合是将指定的行留在它们初始的位置，而将副本放入新的位置。您可以通过指定地址或模式来选择要拷贝的行。请使用前面的格式来执行下面的编辑任务:

- 拷贝当前行
- 拷贝指定地址的行
- 拷贝包含了指定模式的行
- 拷贝不包含指定模式的行

拷贝当前行:

1. 请输入下面的子命令:

   ```
   tAddress[l][n][p]
   ```

   其中参数 *Address* 是行号或符号地址，您想要将当前行的副本放置到该行之后。可选的子命令 **l**、**n** 和 **p** 显示拷贝的行。
2. 请输入文本，并按 Enter 键。
3. 请输入一个 **.**（句点），并再次按 Enter 键以返回命令方式。

拷贝指定地址的行:

1. 请输入下面的子命令:

   ```
   LineNumbertDestinationAddress[l][n][p]
   ```

   其中参数 *LineNumber* 是您想要拷贝的行的地址，参数 *DestinationAddress* 代表您想让副本位于该行之后。可选的子命令 **l**、**n** 和 **p** 显示拷贝的行。
2. 请输入文本，并按 Enter 键。
3. 请输入一个 **.**（句点），并再次按 Enter 键以返回命令方式。

拷贝包含了指定模式的行: 请输入下面的子命令:

```
[Address]g/Pattern/t[DestinationAddress][l][n][p]
```

其中 *Address* 是可选参数，在它指定的行的范围中搜索包含了指定模式的行，参数 *Pattern* 是您要搜索的文本，而 *DestinationAddress* 是可选的参数，指定您想要拷贝的文本行在其之后。可选的子命令 **l**、**n** 和 **p** 显示拷贝的行。

如果您省略了参数 *Address*，ed 编辑器就在整个文件中搜索包含模式的行。如果您省略了参数 *DestinationAddress*，拷贝后的文本就被放置到当前行之后。

拷贝不包含指定模式的行: 请输入下面的子命令:

```
[Address]v/Pattern/t[DestinationAddress][l][n][p]
```

其中 *Address* 是可选参数，在它指定的行的范围中搜索不包含了指定模式的行，参数 *Pattern* 是文本，而 *DestinationAddress* 是可选的参数，指定您想要拷贝的文本行在某行之后。可选的子命令 **l**、**n** 和 **p** 显示拷贝的行。

如果您省略了参数 *Address*，ed 编辑器就在整个文件中搜索不包含该模式的行。如果您省略了参数 *DestinationAddress*，拷贝后的文本就被放置到当前行之后。

删除文本

(.,.)(.,.)**d** [**l**] [**n**] [**p**]

子命令 **d**（删除）将指定的行从缓冲区中除去。最后一次删除的行之后的行就成为当前行。如果被删除的行原来位于该缓冲区的结束位置，新的最后一行就成为当前行。

如果您想要显示被删除的行，请输入可选的子命令 **l**（列表）、**n**（行号）或 **p**（打印）。

ed 编辑器提供了几种方式来删除文本。请使用前面的格式来执行下面的编辑任务:

- 删除当前行
- 删除一行或一组行
- 删除包含了指定模式的一行或一组行
- 删除不包含指定模式的一行或一组行
- 从当前行中删除文本
- 在选定的行中删除文本
- 从指定的行中删除文本
- 从包含了指定模式的行中删除文本
- 从包含了不同的指定模式的行中删除模式
- 从未包含不同指定模式的行中删除模式

删除当前行: 请输入下面的子命令:

```
d[l][n][p]
```

其中 **l**、**n** 和 **p** 是显示被删除的行的可选子命令。

删除一行或一组行: 请输入下面的子命令:

```
Addressd[l][n][p]
```

其中参数 *Address* 是您想要删除的行的行号或符号地址，而 **l**、**n** 和 **p** 是显示被删除的一行或几行的可选子命令。

删除包含了指定模式的一行或一组行: 请输入下面的子命令:

```
[Address]g/Pattern/d[l][n][p]
```

其中 *Address* 是可选参数，它指定了您想要搜索的行的行号或符号地址，而参数 *Pattern* 是字符串或正则表达式代表了您想要查找的文本。如果您省略了参数 *Address*，ed 编辑器就在整个文件中搜索包含了指定模式的行。可选的子命令 **l**、**n** 和 **p** 显示被删除的一行或几行。

删除不包含指定模式的一行或一组行: 请输入下面的子命令:

```
[Address]v/Pattern/d[l][n][p]
```

其中 *Address* 是可选参数，它指定了您想要搜索的行的行号或符号地址，而参数 *Pattern* 是字符串或正则表达式代表了您想要查找的文本。如果您省略了参数 *Address*，ed 编辑器就在整个文件中搜索不包含该模式的行。可选的子命令 **l**、**n** 和 **p** 显示被删除的一行或几行。

从当前行中删除文本:

1. 请输入下面的子命令:

   ```
   s/Pattern
   ```

   其中参数 *Pattern* 是字符串或者正则表达式，代表您想要删除的文本。
2. 要从行中删除模式的*第一个实例*，请输入:

   ```
   //
   ```

   OR

   要从行中删除模式的*每一个实例*，请输入:

   ```
   //g
   ```
3. 如果您想要显示删除的对象，请输入下面的可选子命令之一:

   ```
   l
   ```

   ```
   n
   ```

   ```
   p
   ```
4. 请按 Enter 键。

在选定的行中删除文本:

1. 请输入一组行的地址来选择（或者跳过该步来选择所有行）。
2. 要选择第 4 步中参数 *Pattern* 所指示的行，请输入:

   ```
   g
   ```

   或

   要选择第 4 步*未被* 参数 *Pattern* 所指示的行，请输入:

   ```
   v
   ```
3. 要进入您想要搜索的文本，请输入下面的子命令:

   ```
   /Pattern/s
   ```

   其中参数 *Pattern* 是您想要搜索的文本。
4. 请输入下面的子命令之一来做出所希望的删除:

   要在选中的每一行中删除参数 *Pattern* 的第一个实例，请输入:

   ```
   ///
   ```

   要在选中的每一行中删除参数 *Pattern* 的每一个实例，请输入:

   ```
   ///g
   ```

   要在选中的每一行上删除第一个出现了指定次数的参数 *Pattern*（其中参数 *Number* 为整数），请输入:

   ```
   ///Number
   ```

要在参数 *Pattern* 所选择的每一行中删除参数 *OtherPattern* 所指示的第一个字符串（其中参数 *OtherPattern* 是您想要搜索的模式），请输入:

```
/OtherPattern//
```

要在参数 *Pattern* 所选择的每一行中删除参数 *OtherPattern* 的每一个实例，请输入:

```
/OtherPattern//g
```

要在参数 *Pattern* 所选择的每一行中删除第一次出现了指定次数的参数 *OtherPattern*（其中参数 *Number* 为整数），请输入:

```
/OtherPattern//Number
```

5. 如果您想要显示删除的对象，请输入下面的可选子命令之一:

   ```
   l
   ```

   ```
   n
   ```

   ```
   p
   ```

6. 请按 Enter 键。

例如，要从*某范围的行*中删除某个模式的所有实例，请输入:

```
38,$g/tmp/s/gn
```

前面的示例在行 38 到最后一行间的所有行（`38,$`）中搜索 `tmp` 字符串并在这些行中删除该字符串的每一个实例（`/g`）。然后它显示文本被删除的行及它们的行号（`n`）。

要从包含该模式的*所有行*中删除该模式的所有实例，请输入:

```
g/rem/s///gl
```

前面的示例在整个文件中（地址参数被省略）搜索包含（`g`）字符串 `rem` 的所有行。它从每一行中删除字符串 `rem` 的所有实例（`///g`），然后显示删除文本的行，包括这些行中的非打印字符（`l`）。

从寻址的行中删除文本:

1. 请输入下面的子命令:

   ```
   Addresss/Pattern
   ```

   注: 参数 *Address* 后面是子命令 **s**。
   其中参数 *Address* 是您想要从中删除模式的行的行号、行号范围或符号地址，而参数 *Pattern* 是字符串或正则表达式，代表您想要删除的文本。

2. 要从每一行中删除模式的*第一个实例*，请输入:

   ```
   //
   ```

   OR

   要从每一行中删除该模式的*每一个实例*，请输入:

   ```
   //g
   ```

3. 如果您想要显示删除的对象，请输入下面的可选子命令之一:

   ```
   l
   ```

```
n

p
```

4. 请按 Enter 键。

从包含了指定模式的行中删除文本:

1. 请输入下面的子命令:

   ```
   [Address]g/Pattern/s
   ```

   其中 *Address* 是可选参数，它指定了包含了指定模式的行的行号、行号范围或符号地址，而参数 *Pattern* 是字符串或正则表达式，代表您想要查找和删除的文本。如果您省略了参数 *Address*，ed 编辑器就在文件的所有行中搜索该模式。
2. 要从包含该模式的每一行中删除它的*第一个实例*，请输入:

   ```
   ///
   ```

   OR

   要从包含该模式的每一行中删除它的*每一个实例*，请输入:

   ```
   ///g
   ```
3. 如果您想要显示删除的对象，请输入下面的可选子命令之一:

   ```
   l

   n

   p
   ```
4. 请按 Enter 键。

从包含了不同的指定模式的行中删除模式:

1. 请输入下面的子命令:

   ```
   [Address]g/SearchPattern/s
   ```

   其中 *Address* 是可选参数，它指定了包含了指定模式的行的行号、行号范围或符号地址，而参数 *SearchPattern* 是字符串或正则表达式，代表您想要更改的行中的文本。如果您省略了参数 *Address*，ed 编辑器就在文件的所有行中搜索该指定模式。
2. 要指定您想要删除的文本，请输入:

   ```
   /DeletePattern/
   ```
3. 要从每一行中删除模式的*第一个实例*，请输入:

   ```
   /
   ```

   OR

   要从每一行中删除该模式的*每一个实例*，请输入:

   ```
   /g
   ```

   注: 整条子命令字符串像这样:

   ```
   [Address]g/SearchPattern/s/DeletePattern//[g]
   ```
4. 如果您想要显示删除的对象，请输入下面的可选子命令之一:

l

n

p

5. 请按 Enter 键。

例如，要从包含了不同的指定模式的行中删除模式的第一个实例，请输入:

```
1,.g/rem/s/tmp//l
```

前面的示例搜索第一行到当前行（1,.）中所有包含（g）rem 字符串的行。它从这些行的每一行（/）中删除字符串 tmp 的第一个实例，然后显示文本被删除的行，包括这些行中的非打印字符（l）。

从不包含不同的指定模式的行中删除模式:

1. 请输入下面的子命令:

   ```
   [Address]v/SearchPattern/s
   ```

   其中 *Address* 是可选参数，它指定了包含了指定模式的行的行号、行号范围或符号地址，而参数 *SearchPattern* 是字符串或正则表达式，代表不在您想要查找并更改的行中的文本。如果您省略了参数 *Address*，ed 编辑器就在文件的所有行中搜索该指定模式。
2. 要指定您想要删除的文本，请输入:

   ```
   /DeletePattern/
   ```
3. 要删除该模式的*第一个实例*，请输入:

   /

   OR

   要从每一行中删除该模式的*每一个实例*，请输入:

   /g

   注: 整条子命令字符串像这样:

   ```
   [Address]v/SearchPattern/s/DeletePattern//[g]
   ```
4. 如果您想要显示删除的对象，请输入下面的可选子命令之一:

   l

   n

   p
5. 请按 Enter 键。

例如，要从不包含指定模式的行中删除模式的第一个实例，请输入:

```
1,.v/rem/s/tmp//l
```

前面的示例搜索第一行到当前行（1,.）中所有不包含（v）rem 字符串的行。它从这些行的每一行中（/）删除字符串 tmp 的第一个实例，然后显示文本被删除的行，包括这些行中的非打印字符（l）。

显示文本

(.,.)**l** 子命令 **l**（列表）以一种视觉上明确的格式将指定的行写到标准输出，并以相应的转义序列写字符 \\\、\\a、\\b、\\f、\\r、\\t 和 \\v。子命令 **l** 将打印不出的字符写作一个 3 位数的八进制数，在该字符的每个字节前面加上一个 \（反斜杠）（最重要的字节在第一位）。

子命令 **l** 将长的行换行，您可以通过写 \（反斜杠）/ 换行字符序列来指示换行点。换行发生在第 72 列的位置。$（美元符号）标记每一行的结束。您可以将 **l** 子命令附加给所有 ed 编辑器子命令，除了 **e**、**E**、**f**、**q**、**Q**、**r**、**w** 或 **!** 子命令。当前的行号被设置为被写的最后一行的地址。

(.,.)**n** 子命令 **n**（行号）显示指定的行，每行的前面是它自己的行号和一个跳格字符（显示为空白）；**n** 将被显示的最后一行设置为当前行。您可以将子命令 **n** 附加到除 **e**、**f**、**r** 或 **w** 之外的任何 ed 编辑器子命令之上。例如，子命令 **dn** 删除当前行并显示新的当前行和行号。

(.,.)(.,.)**p** 子命令 **P**（打印）显示指定的行并将被显示的最后一行设置为当前行。您可以将子命令 **p** 附加到除 **e**、**f**、**r** 或 **w** 之外的任何 ed 编辑器子命令之上。例如，子命令 **dp** 删除当前行并显示新的当前行。

(.)**=** 如果不提供地址，子命令 **=**（等号）就显示当前行的行号。当前面有 $ 地址的时候，子命令 **=** 显示缓冲区内最后一行的行号。子命令 **=** 不更改当前行，并且不能附加到子命令 **g** 或子命令 **v** 之上。

当您搜索包含或不包含指定模式的行时，您可以选择某个范围的行号来搜索。您可以用几种不同的方式选择并显示一个 ed 编辑器文件中的一行或一组行。请使用前面的格式来执行下面的编辑任务:

- 显示指定的一行或一组行
- 显示指定的一行或一组行以及它们的非打印字符
- 显示指定的一行或一组行以及它们的行号
- 显示包含了搜索模式的行
- 显示包含了搜索模式的行，包括它们的非打印字符
- 显示包含了搜索模式的行，包括它们的行号
- 显示不包含搜索模式的行
- 显示不包含搜索模式的行，包括它们的非打印字符
- 显示不包含搜索模式的行，包括它们的行号

显示指定的一行或一组行: 请输入下面的子命令:

```
Addressp
```

其中参数 *Address* 是您想要显示的行的行号或符号地址。

指定的一行或几行被显示到屏幕上。如果该组行太长而屏幕不能装下，ed 编辑器就从指定的第一行开始，尽量多地显示。

显示一行或一组行以及它们的非打印字符: 请输入下面的子命令:

```
Addressl
```

其中参数 *Address* 是您想要显示的行的行号或符号地址。

指定的一行或几行以及它们非打印字符被显示到屏幕上。如果该组行太长而屏幕不能装下，ed 编辑器就从指定的第一行开始，尽量多地显示。

显示指定的一行或一组行以及它们的行号: 请输入下面的子命令:

```
Addressn
```

其中参数 *Address* 是您想要显示的行的行号或符号地址。

指定的一行或几行被显示到屏幕上。每一行的行号被显示在该行的旁边。如果该组行太长而屏幕不能装下，ed 编辑器就从指定的第一行开始，尽量多地显示。

显示包含了搜索模式的行: 请输入下面的子命令:

```
Addressg/Pattern/p
```

其中参数 *Address* 是行的范围，参数 *Pattern* 是您想要搜索的字符串或者正则表达式。

包含了指定模式的一行或几行被显示到屏幕上。如果该组行太长而屏幕不能装下，ed 编辑器就从指定的第一行开始，尽量多地显示。

显示包含搜索模式的行，包括它们的非打印字符: 请输入下面的子命令:

```
[Address]g/Pattern/l
```

其中 *Address* 是可选参数，它指定了行的范围，参数 *Pattern* 是您想要搜索的字符串或正则表达式。如果您省略了参数 *Address*，ed 编辑器就搜索整个文件。

包含了指定模式的一行或几行被显示到屏幕上。非打印字符在屏幕上显现。如果该组行太长而屏幕不能装下，ed 编辑器就从指定的第一行开始，尽量多地显示。

显示包含了搜索模式的行，包括它们的行号: 请输入下面的子命令:

```
[Address]g/Pattern/n
```

其中 *Address* 是可选参数，它指定了行的范围，参数 *Pattern* 是您想要搜索的字符串或正则表达式。如果您省略了参数 *Address*，ed 编辑器就搜索整个文件。

包含了指定模式的一行或几行被显示到屏幕上。每一行的行号被显示在该行的旁边。如果该组行太长而屏幕不能装下，ed 编辑器就从指定的第一行开始，尽量多地显示。

显示不包含搜索模式的行: 请输入下面的子命令:

```
[Address]v/Pattern/p
```

其中 *Address* 是可选参数，它指定了行的范围，参数 *Pattern* 是您想要搜索的字符串或正则表达式。如果您省略了参数 *Address*，ed 编辑器就搜索整个文件。

不包含指定模式的一行或几行被显示到屏幕上。如果该组行太长而屏幕不能装下，ed 编辑器就从指定的第一行开始，尽量多地显示。

显示不包含搜索模式的行，包括它们非打印字符: 请输入下面的子命令:

```
[Address]v/Pattern/l
```

其中 *Address* 是可选参数，它指定了行的范围，参数 *Pattern* 是您想要搜索的字符串或正则表达式。如果您省略了参数 *Address*，ed 编辑器就搜索整个文件。

不包含指定模式的一行或几行被显示到屏幕上，包括非打印字符。如果该组行太长而屏幕不能装下，ed 编辑器就从指定的第一行开始，尽量多地显示。

显示不包含了搜索模式的行，包括它们的行号: 请输入下面的子命令:

```
[Address]v/Pattern/n
```

其中 *Address* 是可选参数，它指定了行的范围，参数 *Pattern* 是您想要搜索的字符串或正则表达式。如果您省略了参数 *Address*，ed 编辑器就搜索整个文件。

不包含指定模式的一行或几行被显示到屏幕上，还有它们的行号。如果该组行太长而屏幕不能装下，ed 编辑器就从指定的第一行开始，尽量多地显示。

合并和分割行

(.,.+1)**j** [**l**] [**n**] [**p**]

子命令 **j**（合并）通过除去行与行之间的换行符来合并相邻的行。如果仅给出一个地址，子命令 **j** 不进行任何操作。

如果您想要显示合并的行，请输入子命令 **l**（列表）、**n**（行号）或 **p**（打印）。这些子命令是可选的。

ed 编辑器提供了几种方式来合并或分割行。请使用前面的格式来执行下面的编辑任务:

- 合并当前行和之后的行
- 合并指定的行
- 分割当前行
- 分割指定的行

合并当前行和之后的行: 请输入下面的子命令:

```
j[l][n][p]
```

其中 **l**、**n** 和 **p** 是显示合并的行的可选子命令。

合并指定的行: 请输入下面的子命令:

```
Addressj[l][n][p]
```

其中参数 *Address* 是将要组成一行的连续行的集合，而 **l**、**n** 和 **p** 是显示合并的行的可选子命令。

分割当前行:

1. 要在指定的模式之后分割当前行，请输入下面的子命令:

   ```
   s/Pattern/Pattern\
   ```

   其中参数 *Pattern* 是字符串，您想要从它后面分割该行。

   注: 请保证参数 *Pattern* 所代表的两个字符串完全相同。
2. 请按 Enter 键。
3. 请输入下面的反斜杠:

   ```
   /
   ```
4. 要显示分割的行，请输入下面的可选子命令之一:

   ```
   l
   ```

   ```
   n
   ```

   ```
   p
   ```
5. 请按 Enter 键。

分割指定的行:

1. 要在指定的模式之后分割指定的行，请输入下面的子命令:

   ```
   Addresss/Pattern/Pattern\
   ```

其中参数 *Address* 是将被分割的行的地址，而参数 *Pattern* 是字符串，将从它之后分割该行。

注: 请保证参数 *Pattern* 所代表的两个字符串完全相同。

2. 请按 Enter 键。
3. 请输入下面的反斜杠:

   ```
   /
   ```

4. 要显示分割的行，请输入下面的可选子命令之一:

   ```
   l
   ```

   ```
   n
   ```

   ```
   p
   ```

5. 请按 Enter 键。

进行全局修改

(1,$)**g/***Pattern***/***SubcommandList* [**l**] [**n**] [**p**]

子命令 **g**（全局）首先给匹配参数 *Pattern* 的每一行都加上标记。该模式可以是一个固定的字符串或者正则表达式。然后，针对标记的每一行，该子命令都将该标记的行设置为当前行并运行参数 *SubcommandList*。请在相同的行上输入单个的子命令或一个子命令列表的第一个子命令，并加上子命令 **g**；请在不同的行顺序输入子命令。除最后一行之外，每一行都应该用一个 \（反斜杠） 结束。

参数 *SubcommandList* 可能包含子命令 **a**、**i** 和 **c** 以及它们的输入。如果 *SubcommandList* 参数中的最后一个命令通常都是终止输入方式的 **.**（句点），则 **.**（句点）是可选的。如果参数 *SubcommandList* 不存在，则显示当前行。参数 *SubcommandList* 不能包含子命令 **g**、**G**、**v** 或 **V**。

如果您想要显示改动，请输入子命令 **l**（列表）、**n**（行号）或 **p**（打印）。这些子命令是可选的。

注: 子命令 **g** 类似于子命令 **v**，它在不包含该模式的匹配的每一行上运行参数 *SubcommandList*。

(1,$)**G/***Pattern***/** [**l**] [**n**] [**p**]

交互式子命令 **g**（全局）将匹配参数 *Pattern* 的每一行都标记上，显示标记过的第一行，将该行设置为当前行，然后等待子命令。模式可以是一个固定的字符串或者正则表达式。

子命令 **g** 不接受子命令 **a**、**i**、**c**、**g**、**G**、**v** 和 **V**。该子命令完成后，子命令 **g** 显示下一个标记的行，如此反复。子命令 **g** 将换行符当作空子命令。:&（冒号 & 符号） 让子命令 **G** 再次运行上一个子命令。您可以通过按 Ctrl+C 来停止子命令 **g**。

如果您想要显示改动，请输入子命令 **l**（列表）、**n**（行号）或 **p**（打印）。这些子命令是可选的。

(1,$)**v/***Pattern***/***SubcommandList* [**l**] [**n**] [**p**]

子命令 **v** 对不包含参数 *Pattern* 的匹配的每一行都运行参数 *SubcommandList* 中的子命令。模式可以是一个固定的字符串或者正则表达式。

如果您想要显示更改，请输入子命令 **l**（列表）、**n**（行号）或 **p**（打印）。这些子命令是可选的。

子命令 **v** 不接受子命令**a**、**i**、**c**、**g**、**G** 和 **V**。

注: 子命令 **v** 与子命令 **g** 互补，后者在包含该模式的匹配的每一行上运行参数 *SubcommandList*。

(1,$)**V**/*Pattern*/ [**l**] [**n**] [**p**]

子命令 **v** 将不匹配参数 *Pattern* 的每一行标记上，显示标记过的第一行，将该行设置为当前行，然后等待子命令。模式可以是一个固定的字符串或者正则表达式。

如果您想要显示更改，请输入子命令 **l**（列表）、**n**（行号）或 **p**（打印）。这些子命令是可选的。

子命令 **V** 不接受子命令 **a**、**i**、**c**、**g**、**G** 和 **v**。
注: 子命令 **v** 与子命令 **G** 是互补的，后者将匹配该模式的行标记上。

标记文本

(.)(.)**k***x* [**l**] [**n**] [**p**]

子命令 **k**（标记）用参数 *x* 指定的名称标记指定的行，该参数必须是小写的 ASCII 字母。然后地址 '*x*（单引号标记在标记字符之前）寻址该行。子命令 **k** 不更改当前行。

如果您想要显示标记过的文本，请输入子命令 **l**（列表）、**n**（行号）或 **p**（打印）。这些子命令是可选的。

标记当前行: 请输入下面的子命令:

```
kLetter[l][n][p]
```

其中参数 *Letter* 是从字母 *a* 到 *z*，用作标记，而 **l**、**n** 和 **p** 是显示标记过的文本的可选子命令。

标记指定的行: 请输入下面的子命令:

```
AddresskLetter[l][n][p]
```

其中参数 *Address* 是您想要标记的行的行号或符号地址，参数 *Letter* 是从字母 *a* 到 *z*，用作标记。可选的子命令 **l**、**n** 和 **p** 显示标记过的文本。

移动文本

(.,.)(.,.)**m***A* [**l**] [**n**] [**p**]

子命令 **m**（移动）更改指定的一行或几行的位置。最先被移动的行跟在参数 *A* 所指定的行之后。参数 0 将指定的一行或几行移到该文件的开始。参数 *A* 所指定的地址不能是将被移动的行之一。子命令 **m** 将最后被移动的行设置为当前行。

如果您想要显示删除，请输入子命令 **l**（列表）、**n**（行号）或 **p**（打印）。这些子命令是可选的。

移动一行或一个行的集合会将指定的行从它们最初的位置删除，并将它们放置到一个新的位置。您可以用地址或模式选择移动哪些行。请使用前面的格式来执行下面的编辑任务:

- 移动当前行
- 移动指定地址的行
- 移动包含指定模式的行
- 移动不包含指定模式的行

移动当前行: 请输入下面的子命令:

```
mAddress[l][n][p]
```

其中参数 *Address* 是您想要让当前行在其后的行的行号或符号地址，而 **l**、**n** 和 **p** 是显示移动的行的可选子命令。

移动指定地址的行: 请输入下面的子命令:

```
LineNumbermDestinationAddress[l][n][p]
```

其中参数 *LineNumber* 是您想要移动的行的地址，参数 *DestinationAddress* 是您想要被移动的行放在其后的行。可选的子命令 **l**、**n** 和 **p** 显示移动的行。

移动包含指定模式的行: 请输入下面的子命令:

```
[Address]g/Pattern/m[DestinationAddress][l][n][p]
```

其中参数 *Address* 是可选的参数，在它指定的行中搜索包含指定模式的行，参数 *Pattern* 是您正在搜索的文本，而 *DestinationAddress* 是可选的参数，代表您想让被移动的行跟随其后的行。可选的子命令 **l**、**n** 和 **p** 显示被移动的行。

如果您省略了参数 *Address*，ed 编辑器就在整个文件中搜索包含模式的行。如果您省略了参数 *DestinationAddress*，移动后的文本就被放置到当前行之后。

移动不包含指定模式的行: 请输入下面的子命令:

```
[Address]v/Pattern/m[DestinationAddress][l][n][p]
```

其中 *Address* 是可选参数，在它指定的行的范围中搜索不包含指定模式的行，参数 *Pattern* 是文本，而 *DestinationAddress* 是可选参数，代表您想让被移动的行跟随其后的行。可选的子命令 **l**、**n** 和 **p** 显示被移动的行。

如果您省略了参数 *Address*，ed 编辑器就在整个文件中搜索不包含该模式的行。如果您省略了参数 *DestinationAddress*，移动后的文本就被放置到当前行之后。

保存文本

(1,$)**w** *File*

子命令 **w**（写）将指定的行从缓冲区中拷贝到参数 *File* 所指定的文件中。如果该文件不存在，子命令 **w** 用许可模式 666（允许每个人读和写）创建它，除非 **umask** 设置指定另外的文件创建方式。

子命令 **w** 不更改缺省文件名（除非参数 *File* 是自从您启动 ed 编辑器以来被用过的第一个文件名）。如果您不提供文件名，子命令 **w** 就使用缺省文件名。子命令 **w** 不更改当前行。

如果 ed 编辑器成功地从缓冲区中写入该文件，它就显示被写的字符数。如果您指定子命令 **!** *Command* 而不是文件名，子命令 **w** 读取参数 *Command* 指定的操作系统命令的输出。子命令 **w** 不将您指定的操作系统命令的名称保存作缺省的文件名。
注: 由于对于子命令 **w** 来说 0 不是合法的地址，所以您不能用命令 **ed** 创建一个空文件。

您可以以几种方式保存对文件的更改。请使用前面的格式来执行下面的编辑任务:

- 保存文件到当前文件
- 保存文件的一部分到当前文件
- 保存文件到不同的文件
- 保存文件的一部分到不同的文件

保存文件到当前文件: 请输入下面的子命令:

```
w
```

当前文件是以它当前的名称保存的，ed 编辑器显示写入的字符数。

保存文件的一部分到当前文件: 请输入下面的子命令:

```
Addressw
```

其中参数 *Address* 指定了要写的一行或一组行。ed 编辑器显示写入的字符数。

保存文件到不同的文件: 请输入下面的子命令:

```
w File
```

其中参数 *File* 是要写入的文件的名称。

当前文件被保存到参数 *File* 所指定的文件中。ed 编辑器显示写入的字符数。

保存文件的一部分到不同的文件: 请输入下面的子命令:

```
Addressw File
```

其中参数 *Address* 指定了要写的一行或一组行，而参数 *File* 指定了要写入的文件。

指定的行被保存到参数 *File* 所指定的文件。ed 编辑器显示写入的字符数。

搜索文本

您可以从当前行向前或向后搜索某种模式的文本。该模式可以是由文字字符和特殊字符 ^（音调符号），$（美元符号），.（句点），[（左方括号）、]（右方括号），*（星号），\（反斜杠），%（百分号）和 & 键组成的字符串或正则表达式。

您可以使用 ed 编辑器来执行下面的文本搜索:

- 向前搜索
- 向后搜索
- 往相同的方向重复一次搜索
- 往相反的方向重复一次搜索

向前搜索: 请输入下面的子命令:

```
/Pattern
```

其中参数 *Pattern* 是字符串或正则表达式，它指定了搜索的文本。

光标移动到该模式所指定的文本的第一个字符。

向后搜索: 请输入下面的子命令:

```
?Pattern
```

其中参数 *Pattern* 是字符串或正则表达式，它指定了搜索的文本。

光标移动到该模式所指定的文本的第一个字符。

往相同的方向重复一次搜索: 请输入下面的子命令:

```
/
```

光标移动到上一次搜索命令中的模式所指定的最近的文本实例的第一个字符。

往相反的方向重复一次搜索: 请输入下面的子命令:

```
?
```

光标移动到上一次搜索命令中的模式所指定的最近的文本实例的第一个字符。

替换文本

(.,.)(.,.)**s/***Pattern***/***Replacement***/** [**l**] [**n**] [**p**]
(.,.)(.,.)**s/***Pattern***/***Replacement***/ng** [**l**] [**n**] [**p**]

子命令 **s**（替换）在指定的每一行中搜索匹配参数 *Pattern* 的字符串，并用指定的参数 *Replacement* 替换该字符串。模式可以是一个固定的字符串或者正则表达式。如果没有全局子命令 （**g**），子命令 **s** 就只在指定的每一行上替换第一个匹配字符串。如果存在子命令 **g**，子命令 **s** 就在指定的每一行上替换出现的每一个匹配字符串。如果子命令 **s** 没有为该模式找到匹配，它就返回错误消息 **?**（问号）。

请输入子命令 **l**（列表）、**n**（行号）或 **p**（打印）来显示替换的文本。这些子命令是可选的。
注：除空格和换行符之外，任何字符都可以分隔（定界）参数 *Pattern* 和 *Replacement*。子命令 **s** 将最后更改的行设置为当前行。

如果指定了参数 *Number*（整数），则在指定的每一行中匹配字符串的第一个数被替换。

参数 *Replacement* 中所用的字符 *&*（与） 具有和参数 *Pattern* 相同的值。例如，子命令 **s/are/&n't/** 具有和子命令 **s/are/aren't/** 相同的效果，在当前行上将 **are** 替换为 **aren't**。\&（反斜杠，与） 除去了字符 *&* 在参数 *Replacement* 中的特殊意义。

子模式是由字符串 \)（反斜杠，左圆括号）和 \)（反斜杠，右圆括号）所封闭的模式的一部分；模式起作用仿佛封闭字符不存在。在参数 *Replacement* 中，[<]phsave;202;212*Number* 指的是匹配子模式的字符串。例如，如果当前行中有模式 **the** 的匹配，子命令 **s/\(t\)\(h\) \(e\)/t\1\2ose)** 就将 **the** 替换为 **those**。无论子模式是被嵌套还是位于一个系列中，*\Number* 都指的是参数 *Number* 所指定的具体值，即为从定界字符的左边数起的 \) （反斜杠，右圆括号）。

当 %（百分号）被单独用作参数 *Replacement* 时，它就让子命令 **s** 重复上一个参数 *Replacement*。如果 % 是较长的 *Replacement* 参数的一部分，或者如果 % 前面有一个 \（反斜杠），它就不具备这种特殊的意义。

您可以通过将换行符替换到行中来分割行。在参数 *Replacement* 中。按 \+Enter 的按键顺序引用是换行符（不显示），并将光标移动到下一行以为该字符串的剩余部分做好准备。换行符不能作为子命令 **g** 或子命令 **v** 的列表的一部分而被替换。

ed 编辑器提供了几种方式来替换文本。请使用前面的格式来执行下面的编辑任务:

- 在当前行中替换文本
- 在指定的一行或一组行中替换文本
- 在包含模式的行中替换指定的模式
- 在包含不同模式的行中替换模式
- 在未包含不同模式的行中替换模式

在当前行中替换文本:

1. 请输入下面的子命令:

   ```
   s/OldString/NewString
   ```

   其中参数 *OldString* 是存在的文本，而参数 *NewString* 是您想要用来替换的文本。
2. 请输入下面的操作之一:

要在当前行中用参数 *NewString* 替换参数 *OldString* 的第一个实例，请输入:

```
/
```

要在当前行中用参数 *NewString* 替换参数 *OldPattern* 的每一个实例，请输入:

```
/g
```

3. 要显示更改的文本，请输入下面的可选子命令之一:

```
l

n

p
```

4. 请按 Enter 键。

在指定的一行或一组行中替换文本:

1. 请输入下面的子命令:

```
Addresss/OldPattern/NewString
```

其中参数 *Address* 是您想要替换文本的一行或一组行的地址，参数 *OldPattern* 是存在的文本，而参数 *NewString* 是您想要用来替换的文本。

2. 请输入下面的操作之一:

要在每一行中用参数 *NewString* 替换参数 *OldPattern* 的第一个实例，请输入:

```
/NewString/
```

要在每一行中用参数 *NewString* 替换参数 *OldPattern* 的每一个实例，请输入:

```
/NewString/g
```

要在指定的每一行上用参数 *NewString* 替换参数 *NumberOldPattern* 的第一个实例，请输入:

```
/NewString/Number
```

3. 要显示更改的文本，请输入下面的可选子命令之一:

```
l

n

p
```

4. 请按 Enter 键。

在包含了指定模式的行中替换该模式:

1. 请输入下面的子命令:

```
Addressg/Pattern/s//NewString
```

其中参数 *Address* 是一组行的地址，您想要在这组行中搜索参数 *Pattern* 所指定的模式，而参数 *NewString* 是您想要用来替换参数 *Pattern* 的文本。

2. 请输入下面的操作之一:

要在每一行中用参数 *NewString* 替换参数 *Pattern* 的第一个实例，请输入:

```
/
```

要在每一行中用参数 *NewString* 替换参数 *Pattern* 的每一个实例，请输入:

```
/g
```

3. 要显示更改的文本，请输入下面的可选子命令之一:

```
l

n

p
```

4. 请按 Enter 键。

在包含了不同模式的行中替换模式:

1. 请输入下面的子命令:

```
Addressg/Pattern/s/OldString/NewString
```

其中参数 *Address* 是一组行的地址，您想要在这组行中搜索参数 *Pattern* 所指定的模式，参数 *OldString* 是您想要替换掉的文本，而参数 *NewString* 是您想要用来替换参数 *OldString* 的文本。

2. 请输入下面的操作之一:

要在包含了参数 *Pattern* 的每一行中用参数 *NewString* 替换参数 *OldString* 的第一个实例，请输入:

```
/
```

要在包含了参数 *Pattern* 的每一行中用参数 *NewString* 替换参数 *OldString* 的每一个实例，请输入:

```
/g
```

3. 要显示更改的文本，请输入下面的可选子命令之一:

```
l

n

p
```

4. 请按 Enter 键。

在未包含不同模式的行中替换模式:

1. 请输入下面的子命令:

```
Addressv/Pattern/s/OldString/NewString
```

其中参数 *Address* 是一组行的地址，您想要在这组行中搜索参数 *Pattern* 所指定的模式，参数 *OldString* 是您想要替换掉的文本，而参数 *NewString* 是您想要用来替换参数 *OldString* 的文本。

2. 请输入下面的操作之一:

要在不包含参数 *Pattern* 的每一行中用参数 *NewString* 替换参数 *OldString* 的第一个实例，请输入:

```
/
```

要在不包含参数 *Pattern* 的每一行中用参数 *NewString* 替换参数 *OldString* 的每一个实例，请输入:

```
/g
```

3. 要显示更改的文本，请输入下面的可选子命令之一:

```
l

n

p
```

4. 请按 Enter 键。

撤销文本更改

u [**l**] [**n**] [**p**]

子命令 **u**（撤销）将缓冲区恢复到它最近一次被 ed 编辑器的子命令修改之前的状态。子命令 **u** 不能撤销子命令 **e**、**f** 和 **w**。

如果您想要显示更改，请输入子命令 **l**（列表）、**n**（行号）或 **p**（打印）。这些子命令是可选的。

撤销文本更改: 请输入下面的子命令:

```
u[l][n][p]
```

其中 **l**、**n** 和 **p** 是显示更改的可选子命令。在最后一次保存之后的对该文本执行的所有添加、更改、移动、拷贝或删除的编辑功能都被撤销。

处理文件

您可以使用 ed 编辑器的子命令管理文件以执行下面的任务:

- 添加另外的文件到当前文件
- 更改缺省文件名
- 编辑另外的文件

添加另外的文件到当前文件

(**$**)**r** *File*

子命令 **r**（读）将文件读到缓冲区内指定行之后。子命令 **r** 不会删除该缓冲区以前的内容。如果输入时没有参数 *File*，子命令 **r** 读取缺省文件（如果存在）到该缓冲区中。子命令 **r** 不更改缺省文件名。

0 地址使得子命令 **r** 将文件读入到该缓冲区的开始位置。当子命令 **r** 成功读取文件后，它显示读入缓冲区内的字符数，并将读入的最后一行设置为当前行。

如果在子命令 **r** 中 ！（感叹号）替换了参数 *File*，该行剩下的部分就被认为是操作系统外壳命令，后者的输出将要被读取。子命令 **r** 不将操作系统命令名保存为缺省文件名。

将文件插入到当前行之后: 请输入下面的子命令:

```
r File
```

其中参数 *File* 是要被插入的文件名。

ed 编辑器将参数 *File* 指定的文件读入到当前文件的当前行之后，并显示读入到当前文件中的字符数。

在指定地址的行之后插入文件: 请输入下面的子命令:

```
Addressr File
```

其中参数 *Address* 指定了您想要放置其后的行，而参数 *File* 是将要被插入的文件名称。

ed 编辑器将参数 *File* 指定的文件读入到当前文件的指定行之后，并显示读入到当前文件中的字符数。

更改缺省文件名

f [*File*]　　子命令 **f**（文件名）将缺省文件名（存储下来的最近一次使用过的文件名）更改为参数 *File* 所指定的文件名。如果没有指定参数 *File*，子命令 **f** 显示缺省文件名。（子命令 **e** 存储缺省文件名。）

显示文件名:　　请输入下面的子命令:

```
f
```

ed 编辑器显示编辑缓冲区中的文件名称。

命名文件:　　请输入下面的子命令:

```
f File
```

其中参数 *File* 是该编辑缓冲区内的文件的新名称。

编辑缓冲区内的文件被重新命名。

编辑其他的文件

e *File*　　子命令 **e**（编辑）首先从该缓冲区删除任何内容，将该缓冲区的最后一行设置为当前行，并显示读入该缓冲区的字符数。如果自缓冲区的内容被保存（使用子命令 **w**）以来该缓冲区已被更改，ed 编辑器会在它清除该缓冲区之前显示一个 **?**（问号）。

子命令 **e** 将参数 *File* 存储为要使用的缺省文件名，如果必要的话，再随后使用子命令 **e**、**r** 或 **w**。（要更改缺省文件名的名称，请使用子命令**f**。）

当 **!**（感叹号）替换了参数 *File* 的时候，子命令 **e** 就认为该行剩下的部分是操作系统外壳命令，并读入该命令的输出。子命令 **e** 不将外壳命令的名称存储为缺省文件名。

E *File*　　子命令 **e**（编辑）的作用和子命令 **e** 相似，有一点例外；子命令 **e** 不检查最近一次子命令 **w** 之后对该缓冲区做出的更改。在重新编辑文件之前进行的所有更改都会丢失。

您可以使用子命令 **e** 或 **E** 执行下面的任务:

- 重新编辑当前文件而不保存它
- 在保存当前文件之后重新编辑它
- 保存当前文件之后编辑文件
- 编辑文件而不保存当前文件

重新编辑当前文件而不保存它:　　请输入下面的子命令:

```
E
```

ed 编辑器显示该文件中的字符数。在重新编辑文件之前进行的所有更改都会丢失。

保存当前文件之后重新编辑它:　　请输入下面的子命令:

```
e
```

ed 编辑器显示该文件中的字符数。

在保存当前文件之后编辑文件: 请输入下面的子命令:

```
e File
```

其中参数 *File* 是您想要编辑的新的或存在的文件的名称。

对于存在的文件，ed 编辑器显示该文件中的字符数。对于新文件，ed 编辑器显示一个 ?（问号）和文件名。

编辑文件而不保存当前文件: 请输入下面的子命令:

```
E File
```

其中参数 *File* 是您想要编辑的新的或存在的文件的名称。

对于存在的文件，ed 编辑器显示该文件中的字符数。对于新文件，ed 编辑器显示一个 ?（问号）和文件名。

ed 编辑器子命令的其他功能

您可以使用 edit 编辑器子命令执行下面的任务:

- 更改提示符字符串
- 输入系统命令
- 退出 ed 编辑器
- 请求帮助

更改提示符字符串

P 子命令 **P**（提示）打开或关闭用 *（星号）代表的 ed 编辑器提示符字符串。初始时，子命令 **P** 是关闭的。

启动或停止显示提示字符串: 请输入下面的子命令:

```
P
```

ed 编辑器的提示符 *（星号）是否显示取决于它先前的设置。

输入系统命令

! *Command* **!** 子命令允许您运行操作系统命令，而不用离开 ed 编辑器。在 ed 编辑器的某一子命令行上跟在 **!** 子命令后的所有内容将解释成操作系统命令。在该命令字符串的文本中，ed 编辑器将未转义的 %（百分号）替换为当前文件名，如果存在。

可以通过在 **!** ed 编辑器子命令后输入 !（感叹号）来重复先前的操作系统命令。如果操作系统的命令解释器（命令 **sh**）扩展了命令字符串，则 ed 编辑器回送扩展的行。**!** 子命令不更改当前行。

您可以使用 **!** 子命令执行以下操作:

- 运行一条操作系统命令
- 重复操作系统命令
- 运行几条操作系统命令

运行一条操作系统命令: 请输入下面的子命令:

```
!Command
```

其中参数 *Command* 指定一条通常在提示符处输入的操作系统命令。

该命令运行并显示其输出。命令完成后，编辑器显示一个 **!**（感叹号）。

重复一条操作系统命令: 请输入下面的子命令:

```
!
```

之前运行过的操作系统命令运行并显示其输出。命令完成后，编辑器显示一个 **!**（感叹号）。

运行几条操作系统命令:

1. 请输入下面的子命令来显示操作系统提示符:

   ```
   !sh
   ```
2. 请输入一条操作系统命令。
3. 按 Enter 键来运行该命令并显示其输出。
4. 重复步骤 2 和 3 来运行更多的操作系统命令。
5. 按 Ctrl+D 以返回命令方式。编辑器显示一个 **!**（感叹号）。

退出 ed 编辑器

q 子命令 **q**（退出）检查该缓冲区是否在最近一次执行的更改之后已被保存到文件中，然后退出 ed 编辑器。如果该缓冲区没有保存到文件中，子命令 **q** 显示 **?**（问号）消息。请再次输入子命令 **q** 以强行退出 ed 编辑器。对当前文件所做的更改丢失。

Q 子命令 **q**（退出）退出 ed 编辑器而不检查自从该缓冲区被保存到文件中以来是否做出了更改。自从最后一次保存以后对该缓冲区做出的任何更改都丢失。

检查编辑之后退出:

1. 请输入下面的子命令:

   ```
   q
   ```
2. 如果 ed 编辑器显示一个 **?**，请输入下面的子命令之一:

   要在退出之前保存更改，请输入:

   ```
   w
   ```

   然后按 Enter 键。

   要退出而不保存更改，请输入:

   ```
   q
   ```
3. 请按 Enter 键。

退出并废弃编辑:

1. 请输入下面的子命令:

   ```
   Q
   ```
2. 请按 Enter 键。自从最后一次保存以后对该缓冲区做出的任何更改都丢失。

请求帮助

h **h**（帮助）子命令为显示的最新 **?** 诊断或错误消息提供了简要的帮助消息。

H 子命令 **h**（帮助）让 ed 编辑器为所有后来发生的 ? 诊断消息显示帮助消息。子命令 **h** 也解释前面的 ? 如果存在。子命令 **h** 交替地打开或关闭这种方式；它最初是关闭的。

启动或停止显示帮助消息: 请输入下面的子命令:

```
H
```

是否为 ed 编辑器响应的 ? 显示帮助消息取决于前面的设置。

显示最后一条帮助消息: 请输入下面的子命令:

```
h
```

为 ed 编辑器最近一次响应的 ? 显示一条帮助消息。

ed 编辑器中的字符类支持

在标准表达式 *Patterns* 中，范围表达式匹配所有那些属于当前语言环境的整理顺序的两个字符之间的字符集合。范围表达式的语法如下:

[*character-character***]**

第一个字符在整理顺序中必须低于或等于第二个字符。例如 [a-c] 匹配 En_US 语言环境中 a、b 或 c 中的任何一个。

范围表达式一般被用来匹配一个字符类。例如 [0-9] 被用来指代所有的数字，而 [a-z A-Z] 被用来指代所有字母。当根据当前的语言环境的整理顺序来解释范围时，这种格式可能会产生不可预料的结果。

请用 [ ]（方括号）内的字符类表达式来匹配字符，而不采用前面的格式。系统根据当前语言环境中的字符类定义解释这种表达式的类型。但是，您不能在范围表达式中使用字符类表达式。

字符类表达式的语法如下:

[:*CharacterClass***:]**

也就是一个左括号、一个冒号、字符类的名称、另一个冒号，然后是右括号。

所有的语言环境中都支持下面的字符类:

[:upper:]	大写字母
[:lower:]	小写字母
[:alpha:]	大写字母和小写字母
[:digit:]	数字
[:alnum:]	字母数字字符
[:xdigit:]	十六进制数字
[:punct:]	标点字符（既不是控制字符，也不是字母数字）
[:space:]	空格、跳格符、回车、换行、垂直跳格或进纸符
[:print:]	可打印字符，包括空格
[:graph:]	可打印字符，不包括空格
[:cntrl:]	控制字符
[:blank:]	空格和跳格符

括号是字符类定义的一部分。要匹配任意大写 ASCII 字母或 ASCII 数字，请使用下面的正则表达式:

```
[[:upper:] [:digit:]]
```

请勿使用表达式 [A-Z0-9]。

一种语言环境可能支持其他的字符类。

换行符是字符类 [:space:] 的一部分，但是将不被该字符类所匹配。换行符只能用特殊搜索字符 $（美元符号）和 ^（脱字符）匹配。

退出状态

命令 **ed** 和 **red** 返回下面的出口值:

0 成功结束。
>0 发生错误。

相关信息

命令 **edit**、命令 **ex**、命令 **grep**、命令 **rsh**、命令 **sed**、命令 **sh**、命令 **stty**、命令 **vi** 或命令 **vedit**、命令 **view**。

---

edit 命令

用途

给新用户提供一种简单的行编辑器。

语法

edit [**-r**] [*File* ...]

描述

edit 命令启动一个行编辑器，该编辑器是为初级用户设计的，是 ex 编辑器的简化版。edit 编辑器属于一系列包含了 ed 编辑器、ex 编辑器和 vi 编辑器的编辑器族。了解 edit 编辑器可以帮助您学习其他编辑器更高级的功能。要编辑文件的内容，请输入:

```
edit  File
```

当 *File* 参数指定的是一个存在的文件时，**edit** 命令将该文件拷贝到缓冲区并显示文件中的行数和字符数。然后它显示 **:**（冒号）提示符来表明它准备从标准输入中读取子命令。

如果 *File* 参数中指定的文件尚未存在 **edit** 命令提示该信息并新建文件。您可以为 *File* 参数指定一个以上的文件名，在这种情况下 **edit** 命令将第一个文件拷入缓冲区，并将剩余的文件名储存在变量列表中以备将来使用。如果您不用 **w** 子命令做出更改的话，edit 编辑器不会更改编辑的文件。

edit 编辑器以下面的两种方式之一进行操作:

命令方式 识别并运行 edit 编辑器的子命令。当您启动 edit 编辑器时，它处于命令方式。要在其他时间进入命令方式，请仅在一行的开头输入 **.**（句点）。

文本输入方式 允许您往 edit 编辑器缓冲区中输入文本。请使用 **append (a)** 子命令、**change (c)** 子命令，或者 **insert (i)** 子命令进入文本输入方式。要结束文本输入方式，请仅在一行的开头输入 **.**（句点）。

标志

-r 在编辑器或系统故障后恢复正在编辑的文件。

文件中的寻址行

edit 编辑器使用下面的三种寻址类型:

- 行号寻址
- 相对位置寻址
- 模式寻址

行号寻址

行号寻址是利用行号或符号名称在文件找出某一行。这种方法是找到某一行或某些行的最简单方式。

要利用符号名称寻址第一行，请输入:

```
.
```

要利用符号名称寻址最后一行，请输入:

```
$
```

您也可以通过用逗号或分号将行号或符号地址分开，以此指定某个范围的行。第二个地址必须指向该范围第一个地址行后面的行。

示例:

```
1,5
```

寻址第一行到第五行。

```
.,$
```

寻址第一行到最后几行。

相对位置寻址

edit 编辑器可以通过某一行与当前行的相对位置找到该行的地址。以 *-Number* 或 *+Number* 参数开头的地址分别指向当前行之前第某行或之后第某行。

示例:

```
+8
```

寻址当前行之后第 8 行。

您也可以利用符号名称结合 *-Number* 或 *+Number* 的地址来找到相对于第一行或最后一行的某一行。

示例:

```
.+3
```

寻址第一行之后 3 行，以及:

```
$-10
```

寻址最后一行之前 10 行。

模式寻址

您可以通过在缓冲区中搜索某个特殊的模式来指定寻址行。edit 编辑器向前或向后搜索，并停止在第一次遇到的、含有与 *Pattern* 参数匹配内容的行的地方。如有必要，该搜索折回缓冲区的结束或起始位置，直到它找到匹配的对象或是返回到当前行。

要向前搜索，请输入:

```
/Pattern/
```

要向后搜索，请输入:

```
?Pattern?
```

您也可以将 *Pattern* 参数用逗号或者分号来分开，以此指定某个范围内的行。第二个地址必须指向该范围第一个地址行后面的行。

示例:

```
Pattern,Pattern
```

当被用作 *Pattern* 参数的一部分时，下面的字符具有特殊的意义:

^ 作为 *Pattern* 参数的第一个字符时，与行的开始相匹配。
$ 作为 *Pattern* 参数的最后一个字符时，与行的结尾相匹配。

使用 edit 编辑器的子命令

edit 编辑器子命令影响 **.**（句点）所表示的当前行。当您启动 edit 编辑器时，当前行是缓冲区中的最后一行。随着缓冲区被编辑，当前行变为最近一次被子命令所作用的行。为了处理文件的不同部分，您必须知道如何找到当前行以及在文件中如何寻址不同行。

您可以利用 edit 编辑器的子命令执行下面的任务:

- 添加文本
- 更改当前文件的文件名
- 修改文本
- 删除文本
- 显示当前文件的文件名和状态
- 显示文本并查找当前行
- 编辑另外的文件
- 结束并退出 edit 编辑器
- 做出全局改动
- 移动或拷贝文本
- 在系统崩溃之后保存文件
- 保存文本
- 替换文本
- 撤销更改

添加文本

在下面的子命令中，参数 *Address* 是可选的。如果您指定了地址，请勿输入括号。您可以使用完整的子命令，或是其缩写（如括号中所示）。

[*Address*]**append** (**a**) *Text* **.**

如果您没有指定 *Address* 参数，请附上您在当前行后面输入的文本。如果您在缓冲区中的位置不正确，您可能需要查找当前行或者指定地址。

如果您指定了地址，子命令 **a** 就将文本附加在指定的行之后。如果您指定地址 0，子命令 **a** 就将文本放置在缓冲区的起始位置。

输入文本，并在每一行的末尾处按 Enter 键。当您输完所有文本时，在行的起始处输入单独的一个 **.**（句点），以结束文本输入方式并返回命令方式。您可以使用子命令 **1,$p** 来显示缓冲区中所有的内容。

注: 在安置文本时，子命令 **a** 和子命令 **i** 不同。

[*Address*]**insert** (**i**)*Text***.**

如果您不指定参数 *Address*，则在当前行之前插入文本。如果您在缓冲区中的位置不正确，您可能需要查找当前行或者指定地址。

如果您指定地址，子命令 **i** 就在指定的行之前插入文本。您不能指定地址 0。

输入您的文本，并在每一行的末尾按 Enter 键。当输入所有文本后，请仅在一行的开头输入 **.**（句点），以结束文本输入方式并返回命令方式。您可以使用子命令 **1,$p** 来显示缓冲区中所有的内容。

注: 在安置文本时，子命令 **i** 与子命令 **a** 不同。

更改当前文件的文件名

file*File*　　将当前文件的文件名改为参数 *File* 所指定的文件名。edit 编辑器不会认为该文件要被编辑。

更改文本

在下面的子命令中，参数 *Address* 是可选的。如果您指定了地址，请勿输入括号。您可以使用完整的子命令，或是其缩写（如括号中所示）。

[*Address1*,*Address2*]**change** (**c**)**.**

Text

如果您不指定参数 *Address*，则用您输入的文本替换当前行。如果您在缓冲区中的位置不正确，您可能需要查找当前行或者指定地址。

如果您指定地址，子命令 **c** 将替换指定的某行或某几行。您可以通过用逗号将地址分开，以指定某个范围的行。

输入您的文本，并在每一行的末尾按 Enter 键。当您输完所有的文本时，在行的起始处输入单独的一个 **.**（句点），以结束文本输入方式并返回到命令方式。您可以使用子命令 **1,$p** 来显示缓冲区中的所有内容。最后输入的行成为当前行。

删除文本

在下面的子命令中，参数 *Address* 和参数 *Buffer* 是可选的。如果您指定地址或缓冲区，请勿输入括号。您可以使用完整的子命令，或是其缩写（在括号中显示）。

[*Address1*,*Address2*] **delete** [*Buffer*] (**d**)

如果您不指定参数 *Address*，则删除当前行。如果您在缓冲区中的位置不正确，您可能需要查找当前行或者指定地址。

如果您指定地址，子命令 **d** 就删除指定的某行或某些行。您可以通过用逗号将地址分开，以指定某个范围的行。最后删除的行之后的行就成为当前行。

如果您用从 a 到 z 的一个小写字母来指定一个缓冲区 edit 编辑器就将指定的几行保存在该缓冲区中。如果您用大写字母指定该缓冲区 edit 编辑器就将这些行添加到该缓冲区中。您可以使用子命令 **pu** 来将已删除的行放回缓冲区。

显示当前文件名称和状态

在下面的子命令中，您可以使用完整的子命令，或是其缩写（如括号中所示）。

file（**f**）

显示当前文件名称以及下面的相关信息:

- 自从上一个子命令 **w** 以来，该文件是否经过修改
- 当前行号
- 缓冲区中的行数
- 指示当前行位置的缓冲区的百分比

显示文本和查找当前行

在下面的子命令中，参数 *Address* 是可选的。如果您指定了地址，请勿输入括号。您可以使用完整的子命令，或是其缩写（如括号中所示）。

[*Address1*,*Address2*]**number**（**nu**）

显示指定在它的缓冲区行号之后的行或几行。如果您没有指定参数 *Address*，子命令 **nu** 就显示当前行和行号。

如果您指定了地址，子命令 **nu** 就显示指定的某行或某几行。您可以通过用逗号将地址分开，以指定某个范围的行。所显示的最后一行成为当前行。

[*Address1*,*Address2*]**print**（**p**）

显示指定的某行或某几行。如果您不指定参数 *Address*，子命令 **p** 显示当前行。

如果您指定地址，子命令 **p** 就显示指定的某行或某几行。您可以通过用逗号将地址分开，以指定某个范围的行。所显示的最后一行成为当前行。

[*Address*]**=**

显示指定行的行号。如果您没有指定参数 *Address*，子命令 **=** 显示当前行的行号。

[*Address*]**z**

从指定的行开始，显示一屏幕文本。如果没有指定参数 *Address*，子命令 **z** 从当前行开始，显示一屏幕文本。

[*Address*]**z-**

显示一屏幕文本，并且指定的行位于底部。如果没有指定参数 *Address*，子命令 **z-** 就显示一屏幕文本，并且当前行在最底。

[*Address*]**z.**

显示一屏幕文本，并且指定的行位于中央。如果没有指定参数 *Address*，子命令 **z.** 就显示一屏幕文本，并且当前行位于中央。

编辑其他的文件

在下面的子命令中，您可以使用完整的子命令，或是其缩写（如括号中所示）。

edit *File* (**e**)
在参数 *File*所指定的新文件上开始一个编辑会话。编辑器首先检查自最后一个 **write (w)** 子命令以来该缓冲区是否被编辑过。

如果自从最后一个 **w** 子命令以来该文件已被编辑，该 edit 编辑器就发出一个警告并取消子命令 **e**。否则 edit 编辑器删除编辑器缓冲区的内容，让指定的文件成为当前文件，并显示新的文件名。

在确保该文件可以被编辑后 edit 编辑器将该文件读入编辑器的缓冲区。如果 edit 编辑器读取文件时未出错，它就显示它所读取的行数和字符数。最后读取的行成为新的当前行。

next (**n**)
将命令行参数表中指定的下一个文件拷贝到缓冲区中进行编辑。

结束并退出 edit 编辑器

在下面的子命令中，您可以使用完整的子命令，或是其缩写（如括号中所示）。

quit (**q**)
在使用子命令 **write (w)** 后结束编辑回话。如果您修改了缓冲区并且未将更改写入磁盘，edit 编辑器就显示一条警告消息，并且不结束该编辑回话。

quit! (**q!**)
结束编辑回话，废弃从最后一个 **w** 子命令以来对该缓冲区所做出的任何更改。

进行全局修改

在下面的子命令中，参数 *Address* 是可选的。如果您指定了地址，请勿输入括号。您可以使用完整的子命令，或是其缩写（如括号中所示）。

[*Address1*,*Address2*]**global/***Pattern*/*SubcommandList* (**g**)
给与参数 *Pattern* 匹配的指定行做标记。然后 edit 编辑器对标记过的每一行都执行参数 *SubcommandList* 中指定的子命令列表。

如果您不指定参数 *Address*，子命令 **g** 就作用于当前行。如果您在缓冲区中的位置不正确，您可能需要查找当前行或者指定地址。

如果您指定了地址，子命令 **g** 就作用于指定的某行或某几行。您可以通过用逗号将地址分开，以指定某个范围的行。

单独一条子命令或子命令列表中第一条子命令与子命令 **g** 出现在相同的行。剩下的子命令必须出现在不同的行，并且每一行都（除了最后一行）以 ****（反斜杠）结尾。缺省子命令为 **print (p)**。

该子命令列表可能包括子命令 **append (a)**、子命令 **insert (i)** 和子命令 **change (c)**，以及与它们有关的输入。在这种情况下，表示结束的句号如果位于该命令列表的最后一行，就可以省略。

注：子命令 **undo (u)** 和子命令 **g** 不能出现在子命令列表中。

移动或拷贝文本

在下面的子命令中，参数 *Address1* 和参数 *Address2* 是可选的。如果您指定了地址，请勿输入括号。您必须指定参数 *Address3*。您可以使用完整的子命令，或是其缩写（如括号中所示）。

[*Address1*,*Address2*]**move** *Address3* (**m**)

如果您不指定地址或是地址范围，则将当前行移到参数 *Address3* 所指定的行之后。如果您在缓冲区中的位置不正确，您可能需要查找当前行或者指定地址。

如果您指定地址，子命令 **m** 就移动指定的一行或几行。您可以通过用逗号将地址分开，以指定某个范围的行。移动的行的第一行成为当前行。

[*Address1*,*Address2*]**yank** [*Buffer*] (**ya**)

拷贝指定的一行或几行到 *Buffer* 中，后者是可选参数，用 a 到 z 间的一个希腊字母来指定。您可以使用子命令 **pu** 将这些行放置到另外的文件中。

[*Address*]**put** [*Buffer*] (**pu**)

检索指定的参数 *Buffer* 的内容，如果您没有指定地址，则将其放置到当前行之后。如果您在缓冲区中的位置不正确，您可能需要查找当前行或者指定地址。

如果您指定地址，子命令 **pu** 检索指定缓冲区的内容，并将其放置到指定行之后。如果您没有指定参数 *Buffer*，子命令 **pu** 恢复最近一次删除或拷贝的文本。

您可以使用子命令 **pu** 加上子命令 **delete (d)** 在文件内部移动行，或者加上子命令 **yank (ya)** 在文件之间复制行。

在宏的内部，您不能使用子命令 **pu** 和子命令 **ya**。

在系统崩溃之后保存文件

preserve

保存当前编辑器缓冲区，仿佛系统刚刚崩溃一样。当子命令 **write (w)** 产生错误并且您不知道如何保存您的工作时使用该子命令。请使用子命令 **recover** 来恢复该文件。

recover *File*

从系统保存区域中恢复参数 *File* 所指定的文件。请在系统崩溃或者子命令 **preserve** 之后使用该子命令。

保存文本

在下面的子命令中，参数 *Address* 是可选的。如果您指定了地址，请勿输入括号。您可以使用完整的子命令，或是其缩写（如括号中所示）。

[*Address1*,*Address2*]**write** [*File*] (**w**)

如果您没有指定地址，则将缓冲区的所有内容写到参数 *File*所指定的文件中。

如果您指定了地址，子命令 **w** 就将指定的一行或几行写到指定的文件中。您可以通过用逗号将地址分开，以指定某个范围的行。edit 编辑器显示它所写入的行数和字符数。

如果您没有指定文件 edit 编辑器就使用当前的文件名。如果参数 *File* 不存在，该编辑器就创建一个。

替换文本

在下面的子命令中，参数 *Address* 是可选的。如果您指定了地址，请勿输入括号。您可以使用完整的子命令，或是其缩写（如括号中所示）。

[*Address1*,*Address2*] **substitute/***Pattern*/*Replacement*/ (**s**)

[*Address1*,*Address2*] **substitute/***Pattern*/*Replacement*/**g**

在每个指定的行上，替换指定的参数 *Pattern*的第一个实例。您可以替换参数 *Pattern* 的每一个实例，只需将子命令 **global (g)** 添加到子命令 **s** 的结束位置。

如果您不指定地址，子命令 **s** 就作用于当前行。如果您在缓冲区中的位置不正确，您可能需要查找当前行或者指定地址。如果您指定了地址，子命令 **s** 就作用于指定的一行或几行。您可以通过用逗号将地址分开，以指定某个范围的行。

撤销更改

在下面的子命令中，您可以使用完整的子命令，或是其缩写（如括号中所示）。

undo（**u**）　撤销最近一个缓冲区编辑子命令对缓冲区的更改。您不能撤销子命令 **write (w)** 或子命令 **edit (e)**。

注：子命令 **global** 被认为是独立于子命令 **u** 的。

相关信息

命令 **ed** 或 **red**、命令 **ex**、命令 **vi** 或 **vedit**。

---

edquota 命令

用途

编辑用户和组的配额。

语法

编辑用户指标

edquota [**-u**] [**-p** *Proto-UserName*] *UserName ...*

编辑组指标

edquota [**-g** [**-p** *Proto-GroupName*] *GroupName ...*]

编辑更改用户或组宽限期

edquota -t [**-u** | **-g**]

描述

edquota 命令用来创建并编辑指标。此命令创建一个含有每一个用户和组的当前磁盘份额的临时文件。用从 **/etc/filesystems** 文件已经建立的份额决定文件系统列表。**edquota** 命令也在临时文件中调用 vi 编辑器（或由环境变量 **EDITOR** 指定的编辑器），这样就可以添加并修改份额了。

注：如果在环境变量 **EDITOR** 中指定编辑器，必须指定编辑器的完整的路径名。

每一个文件系统的份额分别维护。当创建或编辑用户或组的份额时，份额就应用到指定的文件系统。在您想用份额的每一个文件系统中都要设置它。

缺省的情况或当和 **-u** 标志一起使用时，**edquota** 命令在命令行里编辑由 *UserName* 参数指定的一个或几个用户的份额。当带有 **-g** 标志时，**edquota** 命令编辑由 *GroupName* 参数指定的一个和几个组的份额。**-p** 标志确定原型的用户（*UserName*）或原型的组（*Proto-GroupName*）并且为指定的用户或组复制一个这些指标的副本。

用户能超过缺省 1 周宽限期的软限制。在宽限期满时，软限制就强制变成硬限制。宽限期能以天、小时、分钟和秒的形式指定。当其值为 0 时，说明使用缺省的宽限期，当其值为 1 秒时，没有宽限期。**-t** 标志更改宽限期。

在临时文件中显示的字段为:

使用的块	被此用户或组使用的 1KB 系统文件块的当前数目。
使用的节点	被此用户或组使用的当前文件数。
块软限制	在正常操作期间用户或组允许使用的 1KB 的块数。
块硬限制	用户或组可被允许使用的 1KB 的总块数，包括在 quota 宽限期内临时存储器。
节点软限制	在正常操作期间用户或组允许使用的文件数。
节点硬限制	用户或组被允许创建的总文件数，包括在 quota 宽限期内的临时文件。

注: 硬限制的值为 1 说明不允许分配。软限制的值为 1，而硬限制的值为 0，说明只有临时的基础空间允许分配。

当退出编辑器，**edquota** 命令读取临时文件并修改二进制 quota 文件以反映编辑器的修改。

硬限制或软限制只能指定为 1 KB 块的整数倍。

标志

-g 编辑一个或多个指定组的配额。

-p 当用标志 **-u** 调用时，复制为每一个指定的用户为一个原型用户建立的配额。当用标志 **-g**，**-p** 调用时，复制为每一个列出的组为一个原型组建立的配额。

-t 更改宽限期，在宽限期内，份额在软限制被强制成硬限制前能被超越。缺省的宽限期为 1 周。当用标志 **-u** 调用时，为所有的文件系统设定宽限期，文件系统的用户份额由 **/etc/filesystems** 文件指定。当用标志 **-g** 调用时，为所有带有由文件 **/etc/filesystems** 指定的组份额的文件系统设定宽限期。

注: 在用 **edquota** 命令更改宽限期后，新的宽限期直到 **quota.user** 和 **quota.group** 文件被刷新后才生效，通过运行 **quotaoff** 命令和其后的 **quotaon** 命令来刷新。为了使用新的宽限期，已经达到旧的宽限期的用户必须将文件系统的使用减小到软限制的水平以下。将来，当这些用户超过软限制时，新的宽限期会受影响。

-u 编辑一个或几个用户的份额。

注: 如果用户名或组名含有的都是数字，则就认为是用户或组标识符。份额这时候就为 ID 而不是名字编辑。

安全性

访问控制: 只有 root 用户能执行此命令。

示例

要为用户 `sharl` 建立份额，用已经为用户 `davec` 建立好的份额作为原型，请输入:

```
edquota -u -p davec sharl
```

文件

quota.user	指定用户份额。
quota.group	指定组份额。
/etc/filesystems	含有文件系统名和位置。

相关信息

quota 命令、**quotacheck** 命令、**quotaon** 和 **quotaoff** 命令、**repquota** 命令。

磁盘配额系统概述介绍磁盘配额系统，设置磁盘配额系统描述如何建立磁盘配额。两者都位于 《安全性》 中。

---

egrep 命令

用途

搜索文件获得模式。

语法

egrep [**-h**] [**-i**] [**-p** [*Separator*]] [**-s**] [**-v**] [**-w**] [**-x**] [**-y**] [[**-b**] [**-n**] | [**-c** | **-l** | **-q**]] { { **-e***Pattern* | **-f***StringFile* } ... | *Pattern* } [*File* ...]

描述

egrep 命令会在输入文件（缺省值为标准输入）中搜索与用 *Pattern* 参数指定的模式相匹配的行。这些模式是完整的正则表达式就像在 **ed** 命令中的那样（除了 \（反斜杠）和 \\（双反斜杠））。以下规则也应用于 **egrep** 命令:

- 一个正则表达式后面带一个 + （加号）会匹配一个或多个的正则表达式。
- 一个正则表达式后面带一个 ? （问号）会匹配零个或一个该正则表达式。
- 由 | （竖线）或者换行符隔开的多个正则表达式会匹配与任何一个正则表达式所匹配的字符串。
- 一个正则表达式可以被包括在 “()” （括号）中进行分组。

换行符将不会被正则表达式匹配。

运算符的优先顺序是 [,], *, ?, +, 合并, | 和换行符。

> **注:** **egrep** 命令与 **grep** 命令带 **-E** 标志是一样的，除了错误消息和使用消息不同以及 **-s** 标志的功能不同之外。

egrep 命令会显示包含该匹配行的文件，如果您指定了多于一个 *File* 参数的话。对 shell 有特殊含义的字符（$, *, [, |, ^, (,), \）出现在 *Pattern* 参数中时必须带双引号。如果 *Pattern* 参数不是简单字符串，通常必须用单引号将整个模式括起来。在表达式中例如 `[a-z]`，减号表示通过当前整理顺序。整理顺序可以定义等价的类以供在字符范围中使用。它使用了快速确定性的算法，有时需要外部空间。

注:

1. 行被限制在 2048 字节。
2. 段落（在 **-p** 标志下）当前限制在 5000 个字符的长度。
3. 请勿在特殊文件上运行 **grep** 命令，因为它会产生不可预测的结果。
4. 输入行不应该包含 NULL 字符。
5. 输入文件应该以换行符结束。
6. 尽管有些标志可以同时被指定，但是有些标志会覆盖其他的标志。例如，如果同时指定 **-l** 和 **-n** ，只有文件名写入到标准输出中去。

标志

-b	在每行前面加上找到该行的块号码。使用此标志有助于按照上下文查找磁盘块号码。**-b** 标志不能用于来自标准输入和管道的输入。
-c	仅显示匹配行的计数。
-e *Pattern*	指定一个模式。这象一个简单的模式，但是它在模式以一个 -（负号）开始时很有用。
-f *StringFile*	指定包含字符串的文件。
-h	当处理多个文件时排除文件名。
-i	当进行比较时忽略字符的大小写。
-l	列出包含匹配行的文件名（一次）。文件名之间用换行符加以分隔。如果搜索标准输入，会返回一个 "(StandardInput)" 路径名。
-n	在每行之前加上该行在文件中的相对行号。
-p[*Separator*]	显示包含匹配行的整个段落。段落之间将按照分*隔符 Separator* 参数指定的段落分隔符加以定界，这些分隔符是与搜索模式有着相同格式的模式。包含段落分隔符的行将仅用作分隔符；它们不会被包含在输出中。缺省的段落分隔符是空白行。
-q	禁止所有的输出到标准输出，不管匹配行。如果选中输入行，以 0 状态退出。
-s	仅显示出错消息。这点对检查状态有用。
-v	显示除了与指定的模式匹配的行之外的所有行。
-w	执行单词搜索。
-x	显示与指定模式精确匹配而不含其他字符的行。
-y	当进行比较时忽略字符的大小写。

退出状态

此命令返回以下出口值:

0	找到匹配项。
1	未找到匹配项。
>1	发现语法错误，或者文件不可访问（即使找到了匹配项）。

示例

要想使用包含模式匹配字 +、?、|、(和) 中的一个扩展模式，请输入:

```
egrep "\(([A-z]+|[0-9]+)\)" my.txt
```

它会显示那些包含在圆括号里字母或数字的行，而不是在圆括号里包含数字字母组合的行。它会匹配 (y) 和 (783902)，而不是 (alpha19c)。

> **注**: 当使用 **egrep** 命令时 \(（斜杠后跟着开括号）或者 \)（斜杠后跟着闭括号）匹配文本中的括号，但是 (（开括号）和)（闭括号）都是模式组部分中的特殊字符。在使用 **grep** 命令时，逆向也成立。

文件

/usr/bin/egrep	包含指向 **egrep** 命令的硬链接。
/bin/egrep	指定了指向 **egrep** 命令的符号链接。

相关信息

awk 命令、**ed** 命令、**fgrep** 命令、**grep** 命令、**sed** 命令。

《*操作系统与设备管理*》中的『文件』。

《*操作系统与设备管理*》中的『输入和输出重定向』。

《*操作系统与设备管理*》中的『Shell』。

《*AIX 5L V5.3 本地语言支持指南和参考大全*》中的『本地语言支持概述』。

---

eimadmin 命令

用途

管理企业标识映射（EIM）域。

语法

eimadmin -a | -p | -l | -m | -e -D | -R | -I | -A | -C [-s *switch*] **[-v** *verboseLevel*] **[-c** *accessType*] **[-f** *accessUserType*] **[-g** *registryParent*] **[-i** *identifier*] **[-j** *otherIdentifier*] **[-k** *URI*] **[-n** *description*] **[-o** *information*] **[-q** *accessUser*] **[-r** *registryName*] **[-t** *associationType*] **[-u** *registryUser*] **[-x** *registryAlias*] **[-y** *registryType*] **[-z** *registryAliasType*] **[-d** *domainDN*] **[-h** *ldapHost*] **[-b** *bindDN*] **[-w** *bindPassword*] **[-K** *keyFile* **[-P** *keyFilePassword*] **[-N** *certificateLabel*]] **[-S** *connectType*]

描述

eimadmin 命令是 AIX 系统服务 shell 工具。管理员可使用该命令定义 EIM 域并以注册表、标识以及标识和注册表用户之间的关联来填充域。管理员还可以使用 **eimadmin** 赋予用户（及其他管理员）访问 EIM 域的权限或者列出或除去 EIM 实体的权限。

管理员可以以两种方式使用 **eimadmin** 命令:

- 通过将信息包含到 **eimadmin** 命令上的命令行选项中
- 通过将信息包含到 **eimadmin** 命令引用的输入文件中

您可以手动创建文件或通过从数据库导出记录来创建。管理员通过指定命令行选项的组合指导实用程序处理。

eimadmin 命令可以执行以下操作:

- 添加对象（**-a**）
- 清除对象（**-d**）
- 列出对象（**-l**）
- 修改与对象关联的属性（**-m**）
- 擦除属性（**-e**）

在以下对象上:

- 域（**-D**）
- 注册表（**-R**）
- 标识（**-I**）
- 关联（**-A**）
- 访问权限（**-C**）

注:

1. 每个 **eimadmin** 命令必须包含一个操作和一个对象类型。根据对象及您在该对象上执行的操作，EIM 可能需要其他参数。

2. 一些选项针对多值属性，您可以多次指定这些属性。其他选项针对单值选项，对于这些属性您只可以指定一次。（如果您重复使用针对单值属性的选项，**eimadmin** 只处理在命令中碰到的第一个值）。除了这些规定之外，您指定参数的顺序就不重要了。
3. 您可以多种方式编写 **eimadmin** 命令的参数:
   - 连接操作与对象，省略嵌入的连字符: `-aD`
   - 包含两个连字符，并使用空格将两个选项分开: `-a -D`

   换言之，以下示例是无效的，因为它包含了两个连字符，且在 **-D** 之前没有空格: `-a-D`

标志

eimadmin 命令采用以下操作标志。

-a 添加对象。（创建对象定义及其属性。）
-e 擦除属性。（清除单值属性或除去多值属性。）
-l 列出对象。（检索对象定义及其属性。）
-m 修改属性。（通过更改单值属性或添加多值属性来更改现有对象的属性。）
-p 清除对象。（除去对象定义及其属性。）

eimadmin 命令采用以下对象标志。

-A 关联。这是 EIM 域中标识与用户标识之间的关系。
-C 访问权限。这是 EIM 定义的 LDAP 访问控制组。
-D 域。这是存储在 LDAP 目录中的标识、用户注册表及标识与用户标识之间关联的集合。
-I 标识。这是参与 EIM 域的人员或实体的名称。
-r 注册表。这是用户注册表的名称。在用户注册表中定义标识与用户标识之间的关联。

eimadmin 命令采用以下处理控制标志。

-s *switch* *switch* 指定一个影响 **eimadmin** 命令函数操作方式的值。您可以指定以下值:

RMDEPS

除去域或系统注册表时除去从属。这使得能够更方便地通过第一次除去所有为域定义的标识及注册表来除去域。也使得能够更加方便地通过第一次除去所有为注册表定义的应用程序注册表来除去系统注册表。

警告: 请注: **eimadmin** 命令不会在除去从属之前警告从属的存在，因此请小心使用该命令开关。

-v *verboseLevel* *verboseLevel* 参数是从 1 到 10 的整数，它控制 **eimadmin** 命令显示的跟踪详细信息的量。（它用于诊断 **eimadmin** 实用程序中的问题。）缺省值 0 表示无跟踪信息。您可以指定一个从 1 到 10 的整数值，指定从跟踪信息量的最小值到最大值。实用程序将检查值并显示为该级别及所有较低级别定义的跟踪信息。以下级别触发特定信息:

- 3 – 表示 EIM API 调用参数和返回值
- 6 – 表示选项值和输入文件标签
- 9 – 表示实用程序例程入口和出口语句

eimadmin 命令采用下表中列出的必需及可选的属性标志。标志选项是单值的，除非另外指出。如果您多次指定一个选项，则实用程序只处理第一次出现的值。

注:

1. 您可以指定这些属性作为命令选项或作为输入文件中的字段。如果您正在指定命令选项，您必须使用引号（″）或（'）将带有嵌入空白的值括起。引号对于单字值是可选的。指定不带引号的多字值实际上将截断命令行选项；第一个字之后的值将被截掉。
2. 以下特殊字符不允许出现在 *registryName*、*registryParent* 或 *identifier* 中:

   , = + < > # ; \ *

-c *accessType*
指定用户对于 EIM 域拥有的访问权限范围。*accessType* 必须为以下值之一:

ADMIN 指定管理访问权。

REGISTRY
指定注册表访问权。如果指定 **REGISTRY**，您还必须指定一个注册表值（**-r**）。注册表值可以是特定注册表名称，也可以是星号（*）以表示对所有注册表的访问权。

IDENTIFIER
指定标识访问权。

MAPPING
指定映射操作访问权。

-f *accessUserType*
指定访问用户名的类型。*accessUserType* 必须是以下类型之一:

DN *accessUser* 为专有名称。

KERBEROS
accessUser 为 Kerberos 标识。

-g *registryParent*
指定系统注册表的名称。应用程序注册表是系统注册表的子集。如果您正在添加应用程序注册表，您必须使用 **-r** 选项及 **-g** 选项。**-r** 值是您正在定义的应用程序注册表。**-g** 选项是预先存在的系统注册表。

-i *identifier*
指定唯一标识名称。例如: `John Day`。

-j *otherIdentifier*
指定非唯一标识名称。例如: `John`。
注: 您可以多次指定该选项以指定多个非唯一标识。

-k *URI*
为注册表（如果存在）指定统一资源标识（URI）。

-n *description*
指定要与域、注册表、标识或关联相关联的任意文本（您提供该文本）。
注: 您可以只为目标关联定义用户描述。

-o *information*
指定要与标识或关联相关联的附加信息。
注: 您可以只为目标关联定义用户信息。您可以多次指定该选项以指定多条信息。

-q *accessUser*
根据指定的 *accessUserType*，指定具有 EIM 访问权的用户专有名称（DN）或 Kerberos 标识。

-r *registryName*
指定注册表名称。添加新注册表时，**eimadmin** 将注册表作为系统注册表，除非您还指定了 **-g** 选项。如果您指定了 **-g** 选项，则 **eimadmin** 将注册表作为应用程序注册表。

-t *associationType*
指定标识与注册表之间的关系。*associationType* 必须为以下之一:

ADMIN 表示出于管理目的将用户标识与标识关联起来。

SOURCE
表示用户标识是查找操作的源。

TARGET
表示用户标识是查找操作的目标。
注: 您可以多次指定该选项以定义多个关系。

-u *registryUser*
指定在注册表中定义的用户的用户标识。

-x *registryAlias*
为注册表指定另一名称。您必须多次指定该选项以指定多个别名。

-y *registryType*

指定注册表类型。**eimadmin** 能够识别的预定义类型包括以下类型:

- RACF®
- OS/400®
- KERBEROS（对于 case ignore）
- KERBEROSX（对于 case exact）
- AIX
- NDS
- LDAP
- PD（Policy Director）
- WIN2K

您还可以通过使用以下两种规范化方法之一连接唯一 OID 来创建您自己的类型:

- caseIgnore
- caseExact

-z *registryAliasType*

为注册表别名指定类型。您可以创建自己的值或使用以下建议值之一:

- DNSHostName
- KerberosRealm
- IssuerDN
- RootDN
- TCPIPAddress
- LdapDnsHostName

注: 对于一组命令行选项或单个输入数据记录，**eimadmin** 命令只识别 *registryAliasType* 的第一个指定值。但是，**eimadmin** 命令确实能够识别多个注册表别名并将所有这些别名与单个 *registryAliasType* 关联起来。

eimadmin 命令采用以下连接类型标志。

-b *bindDN*

指定专有名称以用于到 LDAP 的简单绑定。

-d *domainDN*

指定 EIM 域的完整专有名称（DN）。*domainDN* 以 `'ibm-eimDomainName='` 开头并由以下元素组成:

域名　您正在创建的 EIM 域的名称。例如，**MyDomain**。

父级专有名称

条目的专有名称就在目录信息树型层次结构中给定的条目之上，例如 `o=ibm,c=us`。例如:

```
ibm-eimDomainName=MyDomain,o=ibm,c=us
```

-h *ldapHost*

为控制 EIM 数据的 LDAP 服务器指定 URL 及端口。格式为:

```
ldap://some.ldap.host:389
ldaps://secure.ldap.host:636
```

-K *keyFile*

指定 SSL 密钥数据库文件的名称，包括全路径名。如果找不到文件，则假定是包含认证证书的 RACF 密钥环的名称。该值对于与安全 LDAP 主机进行 SSL 通信是必需的（前缀为 `ldaps://`）。例如:

```
/u/eimuser/ldap.kdb
```

-N *certificateLabel*

指定要从密钥数据库文件或 RACF 密钥环中使用的证书。如果未指定该选项，则使用在文件或环中标记为缺省值的证书。

-P *keyFilePassword*
指定要访问密钥数据库文件中的加密信息所必需的密码。或者，您可以为该选项指定 SSL 密码存储文件，方法是将 `file://` 作为存储文件名的前缀。例如:

```
secret  or file:///u/eimuser/ldapclient.sth
```

注: 如果您在命令行上对 **-K** 选项而不是 **-P** 选项指定密钥数据库文件的名称，则 **eimadmin** 命令将提示输入密钥文件密码。

-S *connectType*
指定对 LDAP 服务器认证的方法。*connectType* 必须为以下值之一:

- **SIMPLE**（绑定 DN 和密码）
- **CRAM-MD5**（绑定 DN 及受保护密码）
- **EXTERNAL**（数字证书）
- **GSSAPI**（Kerberos）

若未指定，*connectType* 缺省值为 **SIMPLE**。对于连接类型 **GSSAPI**，使用缺省 Kerberos 凭证。运行 **eimadmin** 之前，必须使用诸如 kinit 的服务建立该凭证。关于 KINIT 及其相关的信息，请参阅 AIX Authentication Service Administration。

-w *bindPassword*
指定与绑定 DN 关联的密码。

实用程序需要的连接信息包括 EIM 域（**-d**）及其控制服务器（**-h**）、标识（**-b**、**-w** 或 **-K**、**-P**、**-N**，使用该标识认证（绑定）至服务器）及认证方法（**-S**）。

对于不是域（**-D**）的对象类型，域、服务器和绑定标识的指定是可选的。如果未指定，则从 RACF 概要文件检索信息。

注: 如果指定了任何连接信息，则还必须指定连接类型所必需的值的完整集合。省略一个或多个（并非全部）值将导致错误。下表显示用 **eimadmin** 命令指定时每个连接和主机类型的必需值和可选值。

连接类型 / 主机类型	必需值	可选值
SIMPLE 或 CRAM-MD5 / 安全（`ldaps://`）	-d、-h、-b、-w、-K、-P	-N
SIMPLE 或 CRAM-MD5 / 非安全（`ldap://`）	-d、-h、-b、-w	
EXTERNAL / 安全（`ldaps://`）	-d、-h、-K、-P、-S	-N
EXTERNAL / 非安全（`ldap://`）	不支持	不支持
GSSAPI / 安全（`ldaps://`）	-d、-h、-K、-P、-S	-N
GSSAPI / 非安全（`ldap://`）	-d、-h、-S	

注:

1. 上表有两个例外情况:
   - 如果通过输入文件指定值，则域选项（**-d**）对于域函数不是必需的。
   - 当 -K 指定 RACF 密钥环时，SSL 密钥数据库文件密码或存储文件（-P）不是必需的。
2. 如果简单绑定密码是必需的且命令行上未指定 **-w**，则 **eimadmin** 命令将提示输入简单绑定密码，并且如果 SSL 密钥数据库文件密码是必需的且命令行上未指定 **-P**，也会提示输入 SSL 密钥数据库文件密码。

下表概括了每个对象类型和操作对的必需及可选标志。您可以在输入文件中为大多数选项指定值，而不是在命令行上指定。

对象类型（操作）	标志	注释
D（**a**）	• 必需: **d**、**h** • 可选: **n**	添加域。

对象类型（操作）	标志	注释
D（**p**）	• 必需: **d**、**h** • 可选: **s**	除去域。如果域非空，则包含 -s RMDEPS。
D（**l**）	• 必需: **d**、**h** • 可选:	列出域。指定 -d* 列出所有域。
D（**m**）	• 必需: **d**、**h** • 可选: **n**	修改或添加域属性。
D（**e**）	• 必需: **d**、**h** • 可选: **n**	除去或清除域属性。
R（**a**）	• 必需: **r**、**y** • 可选: **g**、**k**、**n**、**x**、**z**	添加注册表。将假定为 **-r** 指定的值是新的系统注册表，除非还指定了 **-g**，在这种情况下，**-r** 值表示新的应用程序注册表。
R（**p**）	• 必需: **r** • 可选: **s**	除去注册表。
R（**l**）	• 必需: **r** • 可选: **y**	列出注册表。返回域中与指定的 **-r** 值搜索过滤器（可能包含通配符 *）匹配的所有注册表条目。
R（**m**）	• 必需: **r** • 可选: **k**、**n**、**x**、**z**	修改或添加注册表属性，包括注册表别名。
R（**e**）	• 必需: **r** • 可选: **k**、**n**、**x**、**z**	除去或清除注册表属性，包括注册表别名。
I（**a**）	• 必需: **i** • 可选: **j**、**n**、**o**	添加标识。
I（**p**）	• 必需: **i** • 可选:	除去标识。
I（**l**）	• 必需: **i** • 可选:	按唯一标识名称列出标识。返回域中与指定的 **-i** 值搜索过滤器（可能包含通配符 *）匹配的所有标识条目。
I（**l**）	• 必需: **j** • 可选:	按非唯一标识名称列出标识。返回域中具有与指定的 **-j** 值搜索过滤器（可能包含通配符 *）匹配的非唯一标识的所有标识条目。
I（**m**）	• 必需: **i** • 可选: **j**、**n**、**o**	修改或添加标识属性。
I（**e**）	• 必需: **i** • 可选: **j**、**n**、**o**	除去或清除标识属性。
A（**a**）	• 必需: **i**、**r**、**u**、**t** • 可选: **n**、**o**	添加关联。您可以重复 **-t** 选项以添加多个关联类型。**-n** 和 **-o** 标志只与 TARGET 关联有关。
A（**p**）	• 必需: **i**、**r**、**u**、**t** • 可选:	除去关联。您可以重复 **-t** 选项以除去多个关联类型。
A（**l**）	• 必需: **i** • 可选: **t**	列出关联。为指定的 **-i** 唯一标识返回域中所有的关联。指定 **-t** 值以限制返回到给定关联类型的条目。

对象类型（操作）	标志	注释
A（**m**）	• 必需: **r**、**u** • 可选: **n**、**o**	修改或添加关联属性。**-n** 和 **-o** 标志只与 TARGET 关联有关。
A（**e**）	• 必需: **r**、**u** • 可选: **n**、**o**	除去或清除关联属性。**-n** 和 **-o** 标志只与 TARGET 关联有关。
C（**a**）	• 必需: **c**、**q**、**f** • 可选: **r**	添加访问权。对于访问类型 REGISTRY，提供特定 **-r** 注册表值，或提供通配符 ***** 表示访问域中所有注册表。
C（**p**）	• 必需: **c**、**q**、**f** • 可选: **r**	除去访问权。对于访问类型 REGISTRY，提供特定 **-r** 注册表值，或提供通配符 ***** 表示访问域中所有注册表。
C（**l**）	• 必需: **c** • 可选: **r**	按类型列出访问权。对于访问类型 REGISTRY，提供特定 **-r** 注册表值，或提供通配符 ***** 表示访问域中所有注册表。
C（**l**）	• 必需: **q**、**f** • 可选:	按用户列出访问权。

退出状态

eimadmin 命令在完成时返回以下退出代码之一:

0 成功。
4 遇到一个或多个错误，但如果您指定了输入文件，则处理所有记录。
8 出现严重错误，该错误导致在到达输入文件（若指定）结束部分之前处理即停止。

示例

1. 要列出单一域，请输入:

   ```
   eimadmin -lD -h ldap://my.server -b "cn=EIM admin,o=MyCompany,c=US" -d "ibm-eimDomainName=My Employees,o=My Company,c=US"
   ```

   这将返回与以下输出类似的内容:

   ```
   domain name: My Employees
   domain DN: ibm-eimDomainName=My Employees,o=My Company,c=US
   description: employees in my company
   ```

2. 要列出单一注册表，请输入:

   ```
   eimadmin -lR -r MyRegistry
   ```

   这将返回与以下输出类似的内容:

   ```
   registry: MyRegistry
   registry kind: APPLICATION
   registry parent: MySystemRegistry
   registry type: RACF
   description: my racf registry
   URI: ldap://some.big.host:389/profileType=User,cn=RACFA,o=My Company,c=US
   registry alias: TCPGROUP
   registry alias type: DNSHostName
   ```

3. 要列出标识，请输入:

   ```
   eimadmin -lI -i "J.C.Smith"
   ```

   这将返回与以下输出类似的内容:

```
unique identifier: J.C.Smith
other identifier: J.C.Smith
other identifier: Joseph
other identifier: Joe
description: 004321
information: D01
information: 1990-04-11
```

4. 要列出目标关联，请输入:

```
eimadmin -lA -i "J.C.Smith" -t target
```

这将返回与以下输出类似的内容:

```
unique identifier: J.C.Smith
registry: MyRegistry
registry type: RACF
association: target
registry user: SMITH
description: TSO
information: 1989-08-01
information: ADMIN1
```

5. 要列出访问权，请输入:

```
eimadmin -lC -c admin
```

这将返回与以下输出类似的内容:

```
access user: cn=JoeUser,o=My Company,c=us
access user: cn=admin1,o=My Company,c=us
access user: cn=admin2,o=My Company,c=us
```

位置

/usr/bin/eimadmin

安全性

LDAP 管理员有权使用 **eimadmin** 命令并使用它提供的所有功能。只要以下条件为 true，EIM 管理员可以使用该命令:

- 他们拥有在包含 EIM 域的 LDAP 服务器上定义的绑定专有名称及密码。
- 他们的绑定专有名称拥有以下 EIM 权限之一:
  - EIM 管理员
  - EIM 注册表管理员
  - EIM 注册表 X 管理员
  - EIM 标识管理员

标准错误

eimadmin 命令发出一个消息来提示输入密码或表示错误。除非您使用输入文件，否则请勿期望会接收到成功完成的消息。处理输入文件中的记录时，当进程启动及停止时，**eimadmin** 会发出参考消息，此外，每处理 50 条记录将发送一条进度消息。

注: **eimadmin** 命令对于列出（-l）请求将返回一个或多个数据行，除非找不到匹配的 EIM 条目或未授权绑定标识访问该数据。

相关信息

eimadmin.conf 文件。

elogevent 命令、logevent 命令

用途

L 记录事件信息（由事件响应资源管理器（ERRM）产生）到一个指定的日志文件中。

语法

elogevent [**-h**] *log_file*

logevent [**-h**] *log_file*

描述

elogevent 脚本总是用英文返回消息。返回的 **logevent** 脚本消息的语言取决于语言环境的设置。

当事件发生时，这些脚本捕捉由 ERRM 产生的环境变量中由 ERRM 公布的事件信息。这些脚本可以当作由事件响应资源运行的操作来使用。还可以用作创建其他用户定义的操作的模板。

将返回有关 ERRM 环境变量的事件信息，还包含以下内容:

本地时 事件或者重新装备事件被发现的时间。由 ERRM 提供的实际的环境变量是 ERRM_TIME。该值是本地化的，在被显示出来之前会被转化成可读的形式。

这些脚本使用 **alog** 命令来将事件信息写入到指定的 *log_file* 中并从该文件中读取事件信息。

标志

–h 将脚本的用法声明写入到标准输出中。

参数

log_file

指定事件信息被记录的文件的名称。应该指定 *log_file* 参数的绝对路径。

log_file 被当作成一个循环标记，具有 64KB 的固定的大小。当 *log_file* 满了时，新的条目会覆盖存在最早的条目。

如果 *log_file* 已经存在，则事件信息会附加到它后面。如果 *log_file* 不存在，则将创建它，这样事件信息就可以写入到它里面。

退出状态

0 该脚本成功地运行。

1 所要求的 *log_file* 未被指定。

2 *log_file* 路径无效。

限制

- 这些脚本必须运行在 ERRM 运行的节点上。
- 当该事件信息被记录时，运行这些脚本的用户必须有 *log_file* 的写入权限。

标准输出

当 **-h** flag 被指定时，脚本的用法声明被写入到标准输出中。

示例

1. 要记录信息，请在在基于 Web 的系统管理器接口中指定 **/tmp/event.log**ERRM 运行这个命令:

   ```
   /usr/sbin/rsct/bin/logevent/tmp/event.log
   ```

   在该命令运行时，**/tmp/event.log** 文件不需要存在。

2. 要查看 **/tmp/event.log** 文件的内容，请运行这个命令:

   ```
   alog -f /tmp/event.log -o
   ```

   以下样本输出显示了一条 **/var** 文件系统（一个文件系统资源）的警告事件:

   ```
   ============================================================================
   Event reported at Mon Mar 27 16:38:03 2002Condition
   Name: /var space usedSeverity:     Warning
   Event  Expression:                              PercentTotUsed>90

   Resource Name:     /var
   Resource Class Name:    IBM.FileSystem
   Data Type:     CT_UINT32
   Data Value:     91
   ```

位置

/usr/sbin/rsct/bin/elogevent 包含了 **elogevent** 脚本

/usr/sbin/rsct/bin/logevent 包含了 **logevent** 脚本

相关信息

命令: **alog**

---

emgr 命令

用途

启动临时修订（临时修订）管理器，此管理器安装、除去、列出并检查系统的临时修订。

语法

要列出临时修订数据:

emgr -l [**-L** *Label* | **-n** *interim fix Number* | **-u** *VUID*] [**-v**{**1**|**2**|**3**}] [**-X**] [**-a** *path*]

要安装临时修订软件包:

emgr -e *interim fix package* | **-f** *ListFile* [**-w** *Directory*] [**-b**] [**-k**] [**-p**] [**-l**] [**-q**] [**-m**] [**-o**] [**-X**] [**-a** *path*]

要除去已安装的临时修订:

emgr -r -L *Label* | **-n** *interim fix Number* | **-u** *VUID* | **-f** *ListFile* [**-w** *Directory*] [**-a** *path*] [**-b**] [**-k**] [**-p**] [**-l**] [**-q**] [**-X**]

要检查已安装的临时修订:

emgr -c [**-L** *Label* | **-n** *interim fix Number* | **-u** *VUID* | **-f** *ListFile*] [**-w** *Directory*] [**-a** *path*] [**-v**{**1**|**2**|**3**}] [**-X**]

要加载或卸装已安装的临时修订:

emgr -M | **-U** [**-L** *Label* | **-n** *interim fix Number* | **-u** *VUID* | **-f** *ListFile*] [**-w** *Directory*] [**-a** *path*] [**-X**]

要强制除去已安装的临时修订:

emgr -R *interim fix Label* [**-w** *Directory*] [**-a** *path*] [**-X**]

查看由临时修订管理器锁定的软件包:

emgr -P [*Package*] [**-a** *path*] [**-X**]

要显示临时修订程序包的内容和拓扑:

emgr -d -e *interim fix Package* | **-f** *ListFile* [**-w** *Directory*] [**-v**{**1**|**2**|**3**}]

描述

emgr（临时修订管理器）命令可用于安装和管理系统临时修订。临时修订管理器安装由 **epkg** 命令创建的软件包并维护包含临时修订信息的数据库。**emgr** 命令执行以下操作:

- 临时修订软件包安装
- 临时修订除去
- 临时修订列出
- 临时修订检查
- 临时修订加载
- 临时修订卸装
- 显示软件包锁
- 已安装临时修订的强制除去

引用 Efix

引用临时修订的方法有如下几种:

按标号引用

在给定系统上安装的每个临时修订将具有一个唯一的临时修订标号。这是绑定所有不同数据库对象的唯一键。要按标号引用临时修订，请将标号用作为 **-L** 标志的参数。例如，要在具有标号 **ABC123** 的临时修订上运行检查操作，请输入:

```
emgr -cL ABC123
```

按 Efix 标识引用

在给定系统上安装的每个临时修订将具有一个临时修订标识。临时修订标识仅是在临时修订数据库中列出的临时修订的顺序号。如果在基于临时修订列表的临时修订上执行操作，使用此选项将很方便。**emgr** 命令将在执行给定操作前将临时修订标识转换为临时修订标号。要按标识引用临时修订，请将标识用作 **-n** 标志的参数。

注: Efix 标识可在除去或添加临时修订时更改。通过使用 **-l** 标志列出特定临时修订或所有临时修订以始终验证当前临时修订标识号。

例如，要在第一个标识等于 1 的临时修订上运行检查操作，请输入:

```
emgr -cn1
```

按 VUID 引用

因为临时修订软件包不被任何实体正式跟踪，所以相同的临时修订标号可能用于一个以上的临时修订软件包。然而，**emgr** 命令不接受同时安装一个以上具有相同临时修订标号的临时修订。VUID（虚拟唯一标识）可用于区分具有相同临时修订标号的软件包。**emgr** 命令将在执行给定操作前将 VUID 标识转换为临时修订标号。例如，要列出具有 VUID 等于 **000775364C00020316020703** 的已安装的临时修订，请输入:

```
emgr -l -u 000775364C00020316020703
```

注: VUID 在临时修订安装和除去过程的预览阶段显示。VUID 还在用 **-v** 标志列出设置为 2 或更高级别的冗长级别时显示。

Efix 记录

以下操作会记录到 **emgr** 命令的日志文件 **/var/adm/ras/emgr.log** 中:

- 安装
- 除去
- 检查
- 加载
- 卸装
- 强制除去

标志

-a *path* 指定安装的备用目录路径。

-b 使 **emgr** 命令跳过要求重新引导的临时修订的常规 AIX **bosboot** 进程。

-c 指定检查操作。指示 **emgr** 命令在指定的一个或多个临时修订上运行检查操作。

-d 显示内容和拓扑。此选项和 **-v** 标志一起在显示冗长输出时很有用。

-e *interim fix Package* 指定临时修订软件包文件的路径。临时修订软件包文件必须用 **epkg** 命令创建且必须以 16 位压缩扩展名 **.Z** 结束。

-f *ListFile* 指定包含以下内容之一的文件:

- 用于软件安装的软件包位置的列表（每行一个）
- 用于除去、加载、卸装和检查操作的临时修订标号的列表（每行一个）

emgr 命令忽略任何空白行或第一个非空白字符为 **#** 字符的行。

-l 通过使用 **bosboot** 命令的 **-l** 标志运行 AIX **bosboot** 低级别调试器。

-k 通过使用 **bosboot** 命令的 **-D** 标志在 AIX **bosboot** 的过程中载入低级别调试器。

-l 指示 **emgr** 命令在指定的一个或多个临时修订上运行列出操作。

-L *Label* 按临时修订标号选择用于该操作的临时修订。

-m 指示 **emgr** 命令执行加载安装。当加载安装了临时修订时，临时修订文件加载在目标文件上。

-M 指示 **emgr** 命令通过使用 **-m** 标志加载已经加载安装的一个或多个临时修订。**-M** 标志可用于加载使用 **-m** 标志安装并用 **-U** 标志或其他手段（例如重新引导系统）卸装的临时修订。

-n *interim fix ID* 通过指定临时修订标识选择用于此操作的临时修订。

-o 指定临时修订安装可以覆盖现有程序包。

-p 指示 **emgr** 命令执行安装或除去的预览。预览运行所有检查操作，但是不做任何更改。

-P [*Package*] 指定软件包查看操作，这将显示由临时修订管理器锁定的所有软件包、它们的安装程序和锁定标号。

-q 禁止除了错误和强烈警告外的所有输出。

-r 指示 **emgr** 命令在指定的一个或多个临时修订上运行除去操作。

-r *Label* 指示 **emgr** 命令运行强制除去操作。此选项除去与临时修订相关联的临时修订数据和数据包锁，而实际上不除去运行任何除去脚本或引导处理的临时修订文件。此选项一次仅可用于一个临时修订。标识目标临时修订时需要临时修订标号。

警告: 此临时修订除去方法应看作紧急过程。因为此方法可能在目标系统上创建不一致性，所以仅当所有其他除去临时修订的方法不成功时才应使用强制除去方法。

-u *VUID* 通过指定 VUID 选择用于此操作的临时修订。

-U 指示 **emgr** 命令卸装通过 **-m** 标志进行加载安装的一个或多个临时修订。

-v{**1**|**2**|**3**} 指定用于列出操作的冗长级别或用于检查操作的验证级别。有效级别是 1、2 和 3。

-w *Directory* 指示 **emgr** 命令使用指定的工作目录而不是缺省 **/tmp** 目录。

-X 尝试扩展任何没有足够空间执行请求的 **emgr** 操作的文件系统。此选项基于临时修订软件包和 **emgr** 命令提供的可用空间和大小估计扩展文件系统。

注:

1. 即使使用了 **-X** 标志，还是可能在安装过程中耗尽可用磁盘空间。如果安装期间在相同文件系统中创建或扩展其他文件，则更可能发生耗尽情况。
2. **emgr** 命令不能扩展远程文件系统。

退出状态

0 所有 **emgr** 命令操作成功完成。

>0 发生错误。

安全性

只有 root 用户可以运行 **emgr** 命令。只有 root 用户可以访问 efix 数据、保存文件和临时文件。

emgr 命令在系统中查找支持的 MD5 生成命令。如果找到一个此类命令，则 **emgr** 命令将 MD5 校验和显示给用户。然后用户可以用安全源交叉检查该 MD5 和。如果未找到生成 MD5 的命令，则 **emgr** 命令不执行进一步操作。

通过导出 EMGR_MD5_CMD shell 变量，用户可以将路径强制设置到 MD5 命令。该变量应该包含到生成 MD5 命令的绝对路径。

注: 临时修订管理的原始发行版中不支持该功能。建议用户通过将 **bos.rte.install** 更新到最新级别来更新到最新级别的临时修订管理。

示例

1. 要预览名为 **games.020303.epkg.Z** 的临时修订软件包的安装，请输入:

   ```
   emgr -p -e games.020303.epkg.Z
   ```

2. 要安装名为 **games.020303.epkg.Z** 的临时修订软件包并在需要额外空间时自动扩展文件系统，请输入:

   ```
   emgr -X -e games.020303.epkg.Z
   ```

3. 要列出系统上的所有临时修订，请输入:

   ```
   emgr -l
   ```

4. 要列出具有标号 **games** 的临时修订的第 3 层列表，请输入:

   ```
   emgr -lv3 -L games
   ```

5. 要除去具有标号 **games** 的临时修订，请输入:

```
emgr -r -L games
```

6. 要预览文件 **/tmp/myfixes** 中的临时修订标号的除去，请输入:

```
emgr -rp -f /tmp/myfixes
```

7. 要检查具有验证级别 2 的所有临时修订，请输入:

```
emgr -cv2
```

8. 要检查具有验证级别 1（缺省验证级别）的临时修订标识号 3，请输入:

```
emgr -c -n3
```

9. 要检查具有 VUID **000775364C00020316020703** 和验证级别 3 的临时修订，请输入:

```
emgr -u 000775364C00020316020703 -c -v3
```

10. 要列出所有的锁定软件包和它们的临时修订标号，请输入:

```
emgr -P
```

11. 要列出所有锁定了 **installp** 软件包 **bos.rte.lvm** 的临时修订标号，请输入:

```
emgr -P bos.rte.lvm
```

12. 要加载安装名为 **games.020303.epkg.Z** 的临时修订程序包并禁止 AIX **bosboot**，请输入:

```
emgr -e games.020303.epkg.Z -mb
```

13. 要加载所有已经通过 **-m** 选项加载安装到系统上的临时修订文件，请输入:

```
emgr -M
```

14. 要卸装与临时修订标号 **games** 相关联的所有临时修订文件，请输入:

```
emgr -U -L games
```

15. 要显示临时修订程序包 **test.102403.epkg.Z** 上级别 3 的详细输出，请输入:

```
emgr -v3 -d test.102403.epkg.Z
```

文件

/usr/sbin/emgr	包含 **emgr** 命令
/usr/emgrdata/DBS/efix.db	包含临时修订头数据库
/usr/emgrdata/DBS/files.db	包含临时修订文件数据库
/usr/emgrdata/DBS/pkglck.db	包含软件包锁数据库
/usr/emgrdata/DBS/prereq.db	包含先决条件数据库
/usr/emgrdata/DBS/e2eprereq.db	包含临时修订的必备数据库

相关信息

bosboot 命令、**epkg** 命令。

《*安装与迁移*》中的『安装可选软件产品和服务更新』。

---

emstat 命令

用途

显示仿真异常统计信息。

语法

emstat [**-a** | **-v**] [*Interval*] [*Count*]

描述

emstat 命令会显示仿真异常统计信息。仿真异常可能发生在某些旧的应用程序或者库在新的处理器上执行的时候，这些旧的应用程序和库包含了旧处理器结构中的已经被删除了的指令。这些指令可能导致:
非法指令程序异常。操作系统捕获这些异常并进行仿真。
这可能以程序的性能为代价。

仿真异常从上次机器重启开始计数，在当前时间间隔中的计数会被显示。用户可以可选地显示对齐排列的异常统计信息或者单个处理器仿真统计信息。

缺省输出显示每秒钟统计信息。采样间隔和迭代次数
也可以被指定。

参数

Interval	在两个样本之间的间隔.
Count	迭代次数.

标志

-a	显示校准异常统计信息。该标志不能与 **-v** 标志一起使用。
-v	显示单个处理器的统计信息。该标志不能与 **-a** 标志一起使用。

实例

1. 要想每秒钟显示仿真统计信息，请输入:

   ```
   emstat
   ```

   这产生了下面的输出:

   ```
     Emulation  Emulation
     SinceBoot      Delta
       8845591          0
       8845591          0
       8845591          0
       8845591          0
       8845591          0
       8845591          0
     ...
   ```

2. 要想每两秒显示仿真和校准异常统计信息总共显示 5 次，请输入:

   ```
   emstat -a 2 5
   ```

   这产生了下面的输出:

   ```
     Alignment  Alignment  Emulation  Emulation
     SinceBoot      Delta  SinceBoot      Delta
      21260604          0   70091846          0
      23423104    2162500   72193861    2102015
      25609796    2186692   74292759    2098898
      27772897    2163101   76392234    2099475
      29958509    2185612   78490284    2098050
   ```

3. 要想对每个处理器每 5 秒显示一次仿真统计信息，请输入:

   ```
   emstat -v 5
   ```

   这会产生以下输出:

```
Emulation  Emulation  Emulation  Emulation
SinceBoot      Delta    Delta00    Delta01
 88406295          0          0          0
 93697825    5291530          0    5291530
 98930330    5232505    5232505          0
102595591    3665261     232697    3432564
102595591          0          0          0
```

相关信息

alstat 命令。

---

emsvcsctrl 命令

用途

启动事件管理子系统。

语法

emsvcsctrl [**–a** | **–s** | **–k** | **–d** | **–c** | **–t** | **–o** | **–h**]

描述

emsvcsctrl 是一个启动事件管理子系统的控制脚本。事件管理是一个 RSTC 的分布式子系统，它为 IBM RS/6000® 服务器提供一组高可用性服务。此命令通过匹配关于系统资源的状态信息和有关客户机程序关注的资源条件的信息来创建事件。客户机程序会用事件检测和恢复系统故障，这样加强了系统可用性。emsvcsctrl 控制脚本控制事件管理子系统的操作。此子系统在系统资源控制器的控制下，同时属于一个叫 emsvcs 的子系统组。每个子系统都关联着一个守护程序。emsvcsctrl 还控制 AIX 资源监视器子系统的操作。此子系统在系统资源控制器的控制下，同时属于一个叫 emsvcs 的子系统组。每一个子系统都和一个守护程序相关联。

事件管理的实例和 AIX 资源监视器子系统在 HACMP/ES 集群的每个节点上执行。从操作的角度看，事件管理子系统组的组织如下:

子系统 事件管理

子系统组

emsvcs

SRC 子系统

emsvcs 子系统和 haemd 守护程序相关联。

emaixos

emaixos 和 harmad 守护程序相关联。

守护程序

haemd 守护程序提供事件管理服务。harmad 守护程序是 AIX 操作系统资源的资源监视器。

emsvcsctrl 脚本通常不在命令行下执行。通常在系统安装期间被 HACMP/ES 启动脚本命令调用。

emsvcsctrl 脚本为操作事件管理子系统提供各种各样的控制:

- 添加、启动、停止和删除子系统
- 清除子系统
- 打开或关闭跟踪

添加子系统: 当指定 -a 标志时，控制脚本使用 mkssys 命令将事件管理和 AIX 资源监视器子系统添加到 SRC。控制脚本运行如下:

1. 确保 emsvcs 和 emaixos 子系统已停止。
2. 从 SRC 中删除 emsvcs 和 emaixos 子系统（倘若它们仍然存在）。
3. 向 SRC 中添加 emsvcs 子系统。
4. 向 SRC 中添加 emaixos 子系统。
5. 如果 haemrm 组不存在，用 mkgroup 命令添加。发生的任何错误被写到日志文件 /var/ha/log/em.mkgroup 中。
6. 创建 /var/ha/lck/haem 和 /var/ha/soc/haem 目录，如果它们不存在。发生的任何错误被写到日志文件 /var/ha/log/em.mkdir 中。
7. 从安装的位置 /usr/sbin/rsct/install/config/em.HACMP.cdb 将事件管理配置数据库拷贝到运行时的位置 /etc/ha/cfg/em.HACMP.cdb。拷贝时的任何错误都被写进日志文件 /var/ha/log/em.cp 中。

启动子系统: 当指定 -s 标志时，控制脚本使用 startsrc 命令启动事件管理子系统 emsvcs 和 AIX 资源监视器子系统 emaixos。

停止子系统: 当指定 -k 标志时，控制脚本使用 stopsrc 命令停止事件管理子系统 emsvcs 和 AIX 资源监视器子系统 emaixos。

删除子系统: 当指定 -d 标志时，控制脚本使用 rmssys 命令从 SRC 中删除事件管理子系统和 AIX 资源监视器子系统。控制脚本运行如下:

1. 确保 emsvcs 和 emaixos 子系统已停止。
2. 用 rmssys 命令从 SRC 中删除 emsvcs 和 emaixos 子系统。

清除子系统: 当指定 -c 标志时，控制脚本从 SRC 中停止并除去所有系统分区的事件管理子系统。控制脚本运行如下:

1. 用 stopsrc -g emsvcs 命令停止子系统组中子系统的所有实例。
2. 用 rmssys 命令从 SRc 中除去子系统组中子系统的所有实例。
3. 从事件管理配置数据库（EMCDB）的运行时位置 /etc/ha/cfg/em.HACMP.cdb 除去事件管理配置数据库。

打开跟踪: 当指定 -t 标志时，控制脚本用 haemtrcon 命令为 haemd 守护程序打开跟踪功能。harmad 守护程序的跟踪功能用 traceson 命令启用。

关掉跟踪: 当指定 -o 标志时，控制脚本用 haemtrcoff 命令禁用 haemd 守护程序的跟踪功能。用 tracesoff 命令也可以关掉 harmad 守护程序的跟踪功能。

记录日志: 当运行时，事件管理守护程序通常提供关于其操作和错误的信息（通过向 AIX 错误日志写条目）。如果不能写入，错误就被写入日志文件 /var/ha/log/em.default.cluster_name 中。

标志

-a 添加子系统。

-s 启动子系统。

-k 停止子系统。

-d 删除子系统。

-c 清除子系统。

-t 启用子系统跟踪功能。

-o 禁用子系统的跟踪功能。

-h 显示用法信息。

安全性

您必须以有效的 **root** 用户标识来运行。

退出状态

0 表明命令已成功完成。

1 表明有错误产生。

限制

此命令仅在 HACMP™ 环境中有效。

标准错误

（必要时）此命令将错误信息写到标准错误中。

示例

1. 要将事件管理子系统添加到 SRC，请输入:

   ```
   emsvcsctrl -a
   ```

2. 要启动事件管理子系统，请输入:

   ```
   emsvcsctrl -s
   ```

3. 要停止事件管理子系统，请输入:

   ```
   emsvcsctrl -k
   ```

4. 要从 SRC 中删除事件管理子系统，请输入:

   ```
   emsvcsctrl -d
   ```

5. 要清理事件管理子系统，请输入:

   ```
   emsvcsctrl -c
   ```

6. 要打开事件管理守护程序的跟踪功能，请输入:

   ```
   emsvcsctrl -t
   ```

7. 要关闭事件管理守护程序的跟踪功能，请输入:

   ```
   emsvcsctrl -o
   ```

位置

/usr/sbin/rsct/bin/emsvcsctrl 包含 **emsvcsctrl** 脚本

文件

/var/ha/log/em.default.cluster_name
包含名为 cluster_name 的集群上的 haemd 守护程序的缺省日志。

/var/ha/log/em.cp 包含在复制事件管理配置数据库时所产生的任何错误的日志。

/var/ha/log/em.trace.cluster_name
包含名为 cluster_name 的集群上的 haemd 守护程序的跟踪日志。

/var/ha/log/em.mkgroup	包含在创建 haemrm 组时所产生的任何错误的日志。
/var/ha/log/em.mkdir	包含在创建 /var/ha/lck/haem 和 /var/ha/soc/haem 目录时所产生的任何错误的日志。

相关信息

命令: **haemtrcoff**、**haemtrcon**、**lssrc**、**startsrc**、**stopsrc**

守护程序: **haemd**

---

enable 命令

enable 命令包括 AIX 打印子系统 **enable** 和 System V 打印子系统 **enable** 的信息。

AIX 打印子系统 enble 命令

用途

启用打印机队列设备

语法

enable *PrinterName* ...

描述

enable 命令使由参数 *PrinterName* 指定的打印机队列设备在线，或者启用打印机队列设备，使之对系统可用。

注:

1. 您必须拥有 root 用户权限或属于 printq 组才能运行该命令。
2. 如果您输入 `enable -?`，系统显示以下错误消息:

   ```
   enq: (FATAL ERROR): 0781-048: Bad queue or device name: -?
   ```

示例

要启用打印队列设备 `lp0:lpd0`，请输入:

```
enable lp0:lpd0
```

文件

/etc/qconfig	包含队列配置文件。
/etc/qconfig.bin	包含 **/etc/qconfig** 文件的摘要的二进制版本。
/usr/sbin/qdaemon	包含队列守护程序。
/var/spool/lpd/qdir/*	包含队列请求。
/var/spool/lpd/stat/*	包含设备的状态信息。
/var/spool/qdaemon/*	包含队列文件的临时副本。

相关信息

cancel 命令、**disable** 命令、**lp** 命令、**lpstat** 命令。

《*打印机和打印指南*》中的『启动和停止打印队列』。

System V Print Subsystem enable 命令

用途

启用 LP 打印机

语法

enable ***printers***

描述

enable 命令激活指定的***打印机***，使之能打印由 **lp** 命令提交的打印请求。如果打印机是远程的，只启用远程系统请求的传送；**enable** 命令必须在远程系统再运行一遍以激活打印机。（运行 **lpstat -p** 获取打印机的状态。）

当打印设备的属性更改时，由 **enable** 识别。所以要更改设备的定义和分配，必须禁用那个设备上的打印机，更改设备，然后再运行 **enable**。新的设备属性在 **enable** 执行之后生效。

打印机名是*系统定义*字并且限定成 ASCII 字符的大小写。

文件

/var/spool/lp/*

参考

lp 命令、**lpstat** 命令。

---

enotifyevent 命令、notifyevent 命令

用途

M将由事件响应资源管理器（ERRM）生成的事件信息发邮件到一个指定用户标识。

语法

enotifyevent [**-h**] [*user-ID*]

notifyevent [**-h**] [*user-ID*]

描述

enotifyevent 脚本总是返回英文的信息。返回的 **notifyevent** 脚本信息的语言取决于语言环境设置。

当事件发生时，这些脚本捕捉由 ERRM 产生的环境变量中由 ERRM 公布的事件信息。这些脚本可以当作由事件响应资源运行的操作来使用。也会用作模板来创建其他用户定义的操作。

将返回有关 ERRM 环境变量的事件信息，还包含以下内容:

本地时间
: 事件或者重新装备事件被发现的时间。由 ERRM 提供的实际的环境变量是 ERRM_TIME。该值是本地化的，在被显示出来之前会被转化成可读的形式。/defn>

这些脚本 用 **mail** 命令发送 事件信息到指定的用户标识。当指定了用户标识，就假设它是有效的，在使用时不校验。如果不指定用户标识，运行此命令的用户就是缺省的用户。

user-ID 对于事件信息要发送到的用户来说时可选的。如果 *user-ID* 没有指定，运行此命令的用户是缺省的用户。

标志

–h 将该脚本的用法声明写到标准输出中。

参数

log_file

指定事件信息被记录的文件的名称。应该指定 *log_file* 参数的绝对路径。

log_file 被当作成一个循环标记，具有 64KB 的固定的大小。当 *log_file* 满了时，新的条目会覆盖存在最早的条目。

如果 *log_file* 已经存在，则事件信息会附加到它后面。如果 *log_file* 不存在，则将创建它，这样事件信息就可以写入到它里面。

退出状态

0 命令成功运行。

限制

1. 这些脚本必须运行在 ERRM 运行的节点上。
2. **mail** 命令用来读文件。

标准输出

当 **-h** flag 被指定时，脚本的用法声明被写入到标准输出中。

示例

1. 在基于 Web 的系统管理器中指定 **user1** 来给用户发邮件。ERRM 然后运行下面的命令:

   ```
   /usr/sbin/rsct/bin/notifyevent user1
   ```

2. 可以用 **mail** 命令来读取事件信息的内容。下面的示例表明对于 **/var** 文件系统（一个文件系统资源）警告事件是如何被格式化和记录的:

   ```
   ============================================================================
   Event reported at Sun Mar 26 16:38:03 2002

   Condition Name:      /var space used
   Severity:     Warning
   Event Type:       Event
   Expression:       PercentTotUsed>90

   Resource Name:     /var
   Resource Class Name:    IBM.FileSystem
   Data Type:     CT_UINT32
   Data Value:     91
   ```

位置

/usr/sbin/rsct/bin/enotifyevent
包含 **enotifyevent** 脚本

/usr/sbin/rsct/bin/notifyevent 包含 **notifyevent** 脚本

相关信息

命令: **mail**

---

enq 命令

用途

对一个文件排队。

语法

处理一个文件

enq [**-**] [**-B** *CharacterPair*] [**-c**] [**-C**] [**-G**] [**-j**] [**-m** *Text*] [**-M** *File*] [**-n**] [**-N** *Number*] [**-o** *Option*] [**-P** *Queue*] [**-r**] [**-R** *Number*] [**-t** *"User"*] [**-T** *Title*] [**-Y**] [**-Z** *Name*] *File*

更改打印作业的优先级

enq -a *Number* **-#** *JobNumber*

显示状态

enq [**-q** | **-A**] [**-L** | **-W**] [**-e**] [**-#** *JobNumber*] [**-u** *Name*] [**-w** *Seconds*] [**-s**]

更改队列和队列守护程序的状态

enq [**-d**] [**-D**] [**-G**] [**-K**] [**-L**] [**-q** | **-A**] [**-U**]

取消选项

enq [**-X**] [**-x***Number*] [**-P***Printer*]

将一个打印作业保持、释放或者移动到另外的队列

enq { **-h** | **-p** | **-Q** *NewQueue* } { **-#** *JobNumber* [**-P** *Queue*] | **-u** *User* | **-P** *Queue* }

排队并保持打印作业

enq -H *File ...*

描述

enq 命令是用来为向共享资源（典型是打印机设备）的请求进行排队的通用实用程序。使用 **enq** 命令对请求进行排队，取消请求，更改请求的优先级并显示队列和设备的状态。

enq 命令有五个不同的语法图，因为全部的标志不是说一起工作。一些标志用作文件处理并接受 *FileName* 作为一个选项。其他的标志用来更改打印作业的优先级、显示状态、更改队列或队列守护程序的状态以及取消打印作业。

要想对特定队列中的文件进行排队，请使用 **-P** 标志（**-P** *Queue*）。如果存在超过一个设备对一个队列服务，您还可以通过在队列名后面指定该设备（*:device*）来请求一个特定的设备。如果您没有指定一个设备，则该作业被送到第一个可用的设备中。如果您没有指定文件，**enq** 命令会将标准输入拷贝到一个文件中，并对它进行排队打印。

enq 命令的请求可以带有运算符消息。这个功能在分布式环境或者多用户系统中是很有用的。这些消息用来作为一个请求告诉打印机运算符这样的信息，在允许作业打印之前要往打印机装入特殊的表单或者不同颜色的纸

张。这些消息由 **-m** 和 **-M** 标志来指定。**qdaemon** 命令会处理 **enq** 命令的请求。当 **qdaemon** 准备开始一个具有相关消息的请求时，系统会在 **qdaemon** 进程运行所在的机器的控制台上显示这个消息。消息文本伴随着一个提示，告诉打印机操作程序该如何通知该请求继续执行或者如何取消该请求。

由 **enq -A** 生成的显示包含了远程队列的两个条目。第一个条目包含了客户机的本地队列和本地设备名以及它的状态信息。第二个条目紧跟其后；它包含了客户机的本地队列名（再次）和其后的远程队列名。任何提交给远程队列的作业在本地会首先显示，并因为该作业被远程机器处理移动到远程设备上。

由于状态命令会同远程机器进行通信，因此状态显示可能偶然会在等待远程机器的响应时挂起。如果不能在两台机器上建立连接的话，该命令将最终超时。

注:

1. 在您可以对一个文件排队之前，您必须拥有对它的读取访问权。要想除去一个文件（见 **-r** 标志）您也必须有对包含该文件的目录的写访问权
2. 如果在您发布 **enq** 命令后但是还没有打印之前您还想继续改动该文件的话，您必须使用 **-c** 标志。
3. 当在打印机上对文件进行排队时，标志可以按照任何顺序进行分布。
4. **-d** 和 **-G** 标志会立即执行。在该命令行中的标志出现之前的语法错误被报告出来。在命令行中的这些标志之后出现的语法错误会被忽略。

标志

文件处理选项

如果您给于 **enq** 命令一个文件名的列表，它会对它们全部进行排队，以在缺省的设备或者指定的设备上进行文件处理

-
导致 **enq** 命令当作过滤器来操作。如果您没有指定一个文件或者多个文件，**enq** 命令自动读取标准输入。然而，如果您指定了一个文件的话，您还可以使用短划线（**-**）来强制 **enq** 命令读取标准输入。短划线（**-**）实际上不是一个标志，而是一种特殊类型的文件名。因此，它必须位于该命令行所有指定的其他的标志之后。

-B *CharacterPair*
按照后面的方式，根据 *CharacterPair* 的值控制分段页面的打印。（**n** = never, **a** = always, **g** = group. 第一个字符用作报头，第二个字符用作报尾。）

HT	描述
nn	没有报头，没有报尾
na	没有报头，每个文件中有报尾
ng	没有报头，在该作业的末尾有报尾
an	在每个文件中有报头，没有报尾
aa	作业中每个文件中都有报头和报尾
ag	每个文件有报头，作业之后有报尾
gn	在作业起始处有报头，没有报尾
ga	在作业的起始处有报头，在每个文件的后面有报尾
gg	在作业的起始处有页头，在作业的结束处有报尾

/etc/qconfig 文件中的报头和报尾节定义了缺省的分段页面的处理方法。

注: 在远程打印环境中，缺省值是打印报头页而不是报尾页。

-c
拷贝文件。为了节省磁盘空间，**enq** 命令会记住文件名称但不会真的拷贝文件本身。如果您想在等待当前拷贝打印的时候继续修改文件，请使用 **-c** 标志。

-C 指定了 **mail** 命令用作错误消息和作业完成的通知，而不是 **write** 命令。（使用该标志对于写 PostScript 应用程序很有用，因为它允许从打印机得到更好的反馈）。错误消息和作业完成的消息（都是由 **piobe** 命令生成的）以及任何从打印机读入的数据也会用邮件送回。

-C 标志只适用于本地打印作业。如果在送到远程打印机的作业完成时您还想得到通知，请使用 **-n** 标志来接收邮件消息。

注: 有些消息用任何方式都不能从 **qdaemon** 和打印机后端重新定向。这些是系统错误，会被直接送到 **/dev/console** 文件中。

-j 指定将消息 `Job number is: nnn`，（其中 `nnn` 是指定的作业号）显示给标准输出。仅在作业提交到本地打印队列中时才会发生这种情况。

-m *Text* 用 **enq** 命令请求提交一个操作程序报文。指定的文本包含了这个报文。

-M *File* 用 **enq** 命令请求提交一个操作程序报文。指定的文件包含了该报文的文本。

-n 当您的作业完成时通知您。如果 **-t** 标志也被使用的话，**enq** 命令还会通知该请求的用户（见 **-t** 标志）。

-N *Number* 打印文件的 *Number* 个拷贝。通常，一个文件只打印一次。

-o *Option* 指定将针对后端的标志传送到后端。这样，对于每个队列，有本文章中没有介绍可以包括在 **enq** 命令行中的标志。参考 **piobe** 命令获取这些标志的列表。

-P *Queue* 指定作业要送到的队列。可以通过输入 `-P Queue:Device.` 来指定队列中的一个特定的设备。

-r 在成功完成打印后除去该文件。

-r *Number* 将当前作业的优先级设置成 *Number*。这个标志在作业提交的时候用到。利用 **-a** 标志来在作业提交后更改其优先级。越大的数指定越高的优先级。缺省的优先级为 15。对大多数用户来说最大的优先级为 20，有 root 用户权限的用户的最大的优先级为 30。

-t *"User"* 对要分发给 *User* 的输出加标签。通常，该输出被标记以分发给发布 **enq** 命令请求的用户名。*User* 的值必须是一个单字，符合规则用户标识的同样的要求。

-t *Title* 将标题放到报头页中，并在 **-q** 标志被指定时显示出来。通常，作业的标题就是该文件的名称。如果 **enq** 命令是从标准输入中读取的话，作业的标题就是 **STDIN.#**，这里 **#** 是 **enq** 命令的进程 ID。

-Y 告诉 **enq** 命令忽略该标志之后的命令行其余的部分。这对发现队列是否有效是很有用的（如果它是在 **/etc/qconfig** 文件中的话）。例如，输入 `enq -P lp4 -Y` 会返回出口值 0，如果行打印机 **lp4** 是有效队列的话；否则会返回一个非零值。使用这个标志对于强制 **qdaemon** 命令再消化 **/etc/qconfig** 文件是很有用的。

-Z *Name* 指定远程打印作业的始发站。

打印作业优先级选项

-a *Number* 将指定作业的优先级更改成 *Number*。该作业必须在输入带有该标志的 **enq** 命令之前已经被提交打印。参考 **-r** 标志获取优先级的描述。用 **-#** 标志来指定作业的编号。该标志仅对本地打印作业有效。

-# *JobNumber* 指定 **enq -q** 命令或者 **enq -a** 命令使用的作业编号，并只显示在状态输出中指定的作业。

注:

1. 指定 **-P** *Queue* 来覆盖缺省的目的打印机。
2. 如果作业 1、2 和 3 在打印机队列中，并且您指定了您想在作业 1 运行的时候得到作业 3 的状态的话，则状态信息将会显示作业 1 和作业 3 而不是仅仅是作业 3。
3. 如果您指定了一个不存在的作业编号的话，系统会显示队列里的当前作业编号，而不是错误消息。

显示状态选项

-A 为所有的队列提供状态。这就象对 **qconfig** 文件中每个队列运行一次 **enq -q** 命令一样。

-e 将不是在 **qdaemon** 命令的控制下的队列处来的状态信息排除在外。从这样的队列处得到的状态可能有不同的格式。**-e** 标志可以同任何标志的组合一起使用。

-L 指定长状态。该标志可以与 **-A** 标志或者 **-q** 标志一起使用。它不能同 **-W** 标志一起使用。如果 **-L** 标志和 **-W** 标志同时被使用的话，先指定的那个占先。使用 **-L** 标志来显示要在一个打印作业中打印的多个文件。

-q 显示缺省队列的状态。**LPDEST** 和 **PRINTER** 环境变量控制着缺省打印机的名。如果 **LPDEST** 环境变量包含了一个值的话，则这个值永远都是最先使用。如果 **LPDEST** 变量没有值，则 **enq** 命令会使用 **PRINTER** 环境变量。如果 **PRINTER** 环境变量没有包含任何值，则 **enq** 命令会使用系统缺省值。

声明:

1. 使用 **-P** *Queue* 标志和 **-q** 标志一起来显示一个特定队列的状态。
2. 任何目标命令行选项会覆盖 **LPDEST** 和 **PRINTER** 环境变量。

-s 获取打印队列的状态，而不列出任何文件。

-u *Name* 指定打印作业状态的用户名。

-w *Seconds* 指定队列状态的连续输出，每隔指定的 *Seconds* 就更新一次屏幕，直到队列为空（见 **lpq** 命令）。当队列为空时，进程停止。该标志只能与 **-q** 标志或者 **-A** 标志或者 **-L** 标志一起使用。

-W 指定宽状态格式，它具有更长的队列名、设备名和作业编号。作业编号的信息在 AIX 4.3.2 中或者更后面的版本中可得。该标志可以同 **-A** 标志或 **-q** 标志一起使用。它不能同 **-L** 标志一起使用。如果 **-L** 标志和 **-W** 标志同时被使用的话，先指定的那个占先。

更改队列和队列守护程序状态选项。

-d 运行 **digest** 命令，该命令存在于 **/etc/qconfig** 文件中。一旦 digest 完成，任何对 **/etc/qconfig** 文件的更改都会反应到 **/etc/qconfig.bin** 文件中。用户必须有 root 用户权限才能运行这个选项。

除了对所有用户可用的前面的那些标志之外 **enq** 命令还接受以下标志，当它们由具有 root 用户权限的用户输入时。root 用户权限意味着您是 root 用户或者您属于 **printq** 组。

注: 以下标志只能用在本地打印作业中。

-D 设备关闭。关闭与队列相关的设备。**qdaemon** 进程不再向该设备发送作业，并且输入 **enq -q** 命令会显示状态为 DOWN。任何当前运行在该设备上的作业都允许完成。

-G 优雅死亡。在所有当前运行的作业完成时结束 **qdaemon** 进程。这个标志的使用是唯一的将 **qdaemon** 进程关闭的干净的方式。**kill** 命令的使用可能会引发问题，如作业挂在队列中。

如果 **qdaemon** 进程正运行在 **srcmstr** 下的话（缺省配置），**enq -G** 不会防止 **qdaemon** 自动重启动。您必须使用 **chssys** 命令，这个命令会更改缺省的配置并且防止 **qdaemon** 进程的自动重启。以下命令：

```
chssys -s qdaemon -O
```

在 **enq -G** 之前发布，防止 **qdaemon** 的自动重启。

以下命令：

```
startsrc -s qdaemon
```

会手动重新启动 **qdaemon** 进程。

-K 与 **-D** 标志的操作相同，除了所有当前的作业被杀死之外。它们留在队列中，当设备打开时会重新运行。

-L 指定长状态。该标志可以与 **-A** 标志或者 **-q** 标志一起使用。使用 **-L** 标志来显示要在一个打印作业中的多个文件。

-U 提出与队列相关的设备。**qdaemon** 进程重新向它发送作业，输入 **enq -q** 命令会显示它的状态是 READY。

注: 如果存在多于一个设备同队列相关的话，当您使用 **-D** 标志，**-K** 标志和 **-U** 标志时您必须指定设备和队列。例如，仅当该队列中没有其他设备时输入 `-P lp:lpd` 才会指定相同的设备。

取消选项

-X 取消打印您的作业。如果您有 root 用户权限的话，则在该指定队列中的所有的作业都会被删除。该标志只有在本地打印作业中才有效。
-x *Number* 取消指定作业 *Number* 的打印
-P *Printer* 指定 *Printer*，这里或者所有的作业或者选定的作业编号要被取消。

注: 如果您拥有 root 用户权限并且没有指定队列的话，则在所有的队列中的所有的作业都会被删除。

保持和释放一个打印作业选项

-# *JobNumber* 指定要保持或释放的打印作业的编号。
-h 保持指定的打印作业
-H 对用 *File* 参数表示的文件进行排队和保持。
-p 释放指定的打印作业。
-P *Queue* 指定要被保持或释放的打印队列。
-u *User* 指定打印作业要被保持或释放的用户。

移动打印作业选项

-# *JobNumber* 指定要被移动的打印作业的编号
-P *Queue* 指定要被移动的打印队列。*Queue* 变量值可以是一个队列名或者有队列: 设备名的形式。
-Q *NewQueue* 指定打印作业要移动到的目标队列。*NewQueue* 变量值可以是队列名的形式或者是队列: 设备名的形式。
-u *User* 指定打印作业要被移动的用户。

安全性

审计的事件:

事件	信息
ENQUE_admin	队列名，设备名，作业名，用户名

示例

1. 要想在缺省打印机上打印文件 `memo` ，请输入:

   ```
   enq memo
   ```

2. 要想带页码打印文件 `prog.c`，请输入:

   ```
   pr prog.c | enq
   ```

   pr 命令在每页的顶部放置一个标题，该标题包含了文件最后修改日期、文件名和页码。**enq** 命令然后会打印该文件。

3. 要想带页码打印一个从标准输入读入的文件，请输入:

```
pr x | enq  -P bill  -n  -r fn1 - fn3
```

短划线 （**-**） 特殊文件名告诉 **enq** 命令要从标准输入读取。如果在命令行中有文件名的话，一般 **enq** 命令不会从标准输入中读取。它还表示打印的顺序。**pr** 命令会为 x 文件创建一个页面编号了的版本，并将其传送到 **enq** 命令，这个命令会创建一个包含了 **/var/spool/qdaemon** 文件中输出的临时文件。

enq 命令用四个文件创建了一个作业，并将其提交给名为 bill 的队列。它将打印 fn1 文件两次。然后它会打印，无论 **pr** 命令的输出是什么。最后它会打印 fn3 文件。那四个文件被当作用于分段页面的一个作业。当该作业完成时，会发送通知（-n 标志）。由于 -r 标志被指定，fn1 和 fn3 文件在作业完成时被除去。由短划线（**-**）文件创建的临时文件总要删除。

pr 命令在每页的顶部放置一个标题，该标题包含了文件最后修改日期，文件名和页码。**enq** 命令然后会打印该文件。

4. 要想将 report 文件打印在下一个可用的为 fred 队列配置的打印机上，请输入:

   ```
   enq -P fred report
   ```

5. 要想将以 sam 前缀开始的多个文件打印在为 fred 队列配置的下一个可用打印机上，请输入:

   ```
   enq -P fred sam*
   ```

   所有以 sam 前缀开头的文件都包含在一个打印作业中。通常状态命令只显示打印作业的标题，这种情况下就是队列中的第一个文件名除非 **-T** 标志指定了不同的值。要想列出打印作业中所有文件的名称，请使用长状态命令 **enq -A -L**.

6. 要想检查打印队列看看文件是否还在等待打印，请输入:

   ```
   enq  -q
   ```

   这个命令会显示用户缺省队列的状态。如果该文件还没有被打印，则它会出现在队列状态列表中。系统缺省的队列被定义成在 **/etc/qconfig[.bin]** 文件中的第一个队列。用户可以通过设置和导出用 **PRINTER** 环境变量将他们自己的缺省值覆盖。

7. 要想显示非缺省队列 lp0 的状态， 请输入:

   ```
   enq  -q  -P lp0
   ```

8. 要想获得长队列状态，请输入:

   ```
   enq  -L
   ```

9. 要想获取所有队列的状态，请输入:

   ```
   enq  -A
   ```

10. 要想获得所有队列的长状态，请输入:

    ```
    enq  -A  -L
    ```

11. 要获取缺省队列的状态，请以宽格式输入:

    ```
    enq  -W
    ```

12. 要获取所有队列的宽状态，请输入:

    ```
    enq  -W  -A
    ```

13. 要想停止打印一个作业（一个作业就是一个或者多个文件）， 请输入:

    ```
    enq  -x 413
    ```

该命令取消您以前的打印作业的请求。该编号是从输入 **enq -q** 命令获取的列表获得的。如果作业当前正在打印，则打印机会立即停止。如果作页还没有被打印，则它会从队列中除去这样它就不会被打印了。如果作业不在队列中的话，**enq** 命令会显示一条和下面相似的消息:

```
no such request from you -- perhaps it's done?
```

14. 要想将一个打印机从队列系统断开，请输入:

```
enq  -P lp0:dlp0  -D
```

输入这个命令会停止 **enq** 命令的请求被送到为 lp0 队列服务的打印机。如果一个文件当前正在打印，它会被允许打印完。您必须可以执行 **qadm** 命令来运行这个命令。

注: 为一个给定队列服务的打印机是按照设备节名命名的，出现在 **/etc/qconfig[.bin]** 文件中。

15. 要想使用缺省打印机上的 **piobe** 命令后端带页码打印文件，请输入:

```
enq  -o -p filename
```

-p 标志不被 **enq** 命令看见。-o 标志告诉 **enq** 命令将下一项（可以在引号内），传递到未更改的后端。因此，**enq** 命令会将 -p 标志传递给 **qdaemon** 进程，该进程接着传递到后端 **piobe** 上。-p 标志导致 **piobe** 执行 **/usr/bin/pr** 过滤器来在向设备提供数据之前将页码应用到文档中。多个选项可以在引号中给出，前面有 -o 标志，或者没有引号，每个前面有多个 -o 标志。

16. 假定 **qconfig** 文件有以下信息:

```
qname:
                device = fred
fred:
                file = /tmp/hello
                backend = /usr/bin/sh /usr/bin/diff
```

并提供了以下命令:

```
rm /tmp/hello
touch /tmp/hello
pr /etc/hosts|enq  -P qname:fred - /etc/hosts
```

qdaemon 进程用两个变量执行 /usr/bin/diff 程序，一个是临时文件名，另一个是 /etc/hosts 文件。两个文件之间唯一的区别就是其中一个是在 **pr** 命令中运行的。/tmp/hello 文件将包含两个文件之间的差异。如果它不存在，则 **qdaemon** 进程不会创建 /tmp/hello 文件。

17. 以下命令:

```
enq  -m'i want pink paper for this job' /etc/passwd
```

会在打印作用要打印之前将指定的操作程序消息发送到操作程序的控制台上。操作程序必须对这个消息进行响应，来取消或者继续该作业。

```
enq  -M pink /etc/passwd
```

这个命令完成同样的事，只有消息包含在名为 pink 的文件中。

18. 要想取消在 fred 队列中的所有的作业，请输入:

```
enq  -X  -P fred
```

如果输入该命令的用户有 root 用户权限，则所有 fred 队列中的作业都被删除。如果用户没有 root 用户权限，则只有该队列中的用户的作业会被删除。

19. 要想对名为 MyFile 的文件进行排队并返回 MyFile 作业编号到 **jdf** 文件中，请输入:

    ```
    enq  -j MyFile
    ```

20. 要想保持打印作业编号 310，请输入:

    ```
    enq  -h  -#310
    ```

    要想释放被保持的打印作业编号 310，请输入:

    ```
    enq  -p -#310
    ```

21. 要想保持队列 lp0 中的所有的打印作业，请输入:

    ```
    enq  -h  -P lp0
    ```

    要想释放 lp0 队列，请输入:

    ```
    enq  -p -P lp0
    ```

22. 要想保持由 fred 创建的所有打印作业，请输入:

    ```
    enq  -h  -u fred
    ```

    要想释放由 fred 创建的打印作业，请输入:

    ```
    enq  -p -u fred
    ```

23. 要想将作业编号 318 移动到队列 lp0 中，请输入:

    ```
    enq  -Q lp0  -#318
    ```

    控制移动打印作业的标志同保持打印文件的标志的工作方式是一样的。保持标志和变量在前面的示例中介绍了。

文件

/usr/sbin/qdaemon	对守护程序排队
/etc/qconfig	对配置文件排队.
/var/spool/lpd/qdir/*	对请求排队.
/var/spool/lpd/stat/*	设备状态的信息。
/var/spool/qdaemon/*	队列中文件的临时拷贝。
/etc/qconfig.bin	**/etc/qconfig** 文件的摘要的二进制版本

相关信息

chquedev 命令、**lsque** 命令、**mkque** 命令和 **rmque** 命令。

qconfig文件。

《*打印机和打印指南*》中的『更改或显示队列特征』。

《*打印机和打印指南*》中的『打印管理』。

《*打印机和打印指南*》中的『打印机特定信息』。

《*打印机和打印指南*》中的『安装对附加打印机的支持』。

《*打印机和打印指南*》中的『打印假脱机程序』。

《*打印机和打印指南*》中的『虚拟打印机定义和属性』。

《*打印机和打印指南*》中的『打印机冒号文件约定』。

enroll 命令

用途
建立用来实现安全通信通道的密码。

语法
enroll

描述
enroll 命令会建立一个密码，并保护通信通道，在该通道中消息只能被特定的接收端所阅读。该密码用来接收秘密邮件。

enroll 命令同 **xsend** 和 **xget** 命令一起使用，用来发送和接收秘密邮件。**xsend** 命令发送秘密邮件。**xget** 命令请求您输入密码并给您秘密邮件。

示例
要想建立一个密码，请输入:

```
enroll
```

当有提示时，请输入您的密码。这样就允许在您的系统上的其他用户发送给您秘密邮件。用 **xget** 命令来阅读秘密邮件。

文件

/var/spool/secretmail/User.key	包含了用户的加密后的密钥
/usr/bin/enroll	包含了 **enroll** 命令

相关信息
mail 命令、**xget** 命令、**xsend** 命令。

《*网络与通信管理*》中的『邮件』。

《*网络与通信管理*》中的『发送和接收秘密邮件』。

enscript 命令

用途
将文本文件转换成 PostScript 格式进行打印。

语法

enscript [**-1 -2 -c -g -k -l -m -o -q -r -B -G -K -R**] [**-b** *Header*] [**-f** *Font*] [**-f0** *CodeSet*:*Font*] [**-f1** *CodeSet*:*Font*] [**-p** *Out*] [**-F** *Hfont*] [**-F0** *CodeSet*:*Font*] [**-F1** *CodeSet*:*Font*] [**-L** *Lines*] [**-M** *MediaName*] [**-X** *CodesetName*] [*SpoolerOptions*] [*File ...*]

描述

enscript 命令会读入一个文本文件，将其转换成 PostScript 格式，对该文件进行假脱机工作以在 PostScript 打印机上打印。您可以用这个命令来指定字体、标题、限定的格式化选项和假脱机选项。

示例:

```
enscript -daleph bubble.txt
```

在名为 aleph 的打印机上打印 **bubble.txt** 文件的一个副本，和

```
enscript -2r finder.c
```

在缺省打印机上打印 **finder.c** 文件的双联印刷的景观列表。

ENSCRIPT 环境变量可以用来指定缺省值。**ENSCRIPT** 的值被当作在命令行显示的参数之前的参数字符串。例如:

```
ENSCRIPT='-fTimes-Roman8'
```

将您缺省的字体类型大小和字体设置成 8 点大小的 Times Roman 字体。

为 **psdit** 命令和 **enscript** 命令提供的包含不同的介质大小的信息包含在文件 **/usr/lib/ps/MediaSizes** 中。

在 **MediaSizes** 文件中的每个条目所需要的信息可以从 **PostScript Printer Description** 或者与 TranScrip 一起使用的 PostScript 打印机相匹配的 **PPD** 文件中获取。**PPD** 文件从 Adobe 公司的系统可用。从 **PPD** 文件提出的尺寸以打印机的度量点阵来表示。一个打印点就是一英寸的 1/72。

在 **MediaSizes** 文件中以 ASCII 码 ***** （星号）开头的任何行在将命令行提供的介质大小名称匹配到 **enscript** 命令和 **psdit** 命令的时候会被忽略。

在 **MediaSizes** 文件中的每个条目包含了 8 或者 9 个字段。头 8 个字段对所有的条目都是必需的。第 9 个字段是可选的。字段之间用空格隔开。每个条目的字段如下:

字段名	描述
EntryName	包含了与一个由 **-M** 标志和 **enscript** 命令或者 **psdit** 命令提供的介质名称相配的字符串。
MediaWidth	指定用点表示的介质的宽度。
MediaDepth	指定用点表示的介质的深度。
ImageableLLX	指定用点表示的可成像的左下角的 x 坐标。
ImageableLLY	指定用点表示的可成像的左下角的 y 坐标。
ImageableURX	指定用点表示的可成像的右上角的 x 坐标。
ImageableURY	指定用点表示的可成像的右上角的 y 坐标。
PageRegionName	为特定的打印机指定 PostScript 序列，来标识可成像区域的大小。
PaperTrayName	为特定打印机指定 PostScript 序列，来选择一个特定的纸张 / 介质盘。该字段是可选的。 **注**: 对于 PageRegionName 字段和 PaperTrayName 字段该序列可能是多个 PostScript 运算符或字。要想指定这样的一个序列，请使用 ASCII ″（双引号字符）来给整个序列定界。

下表显示了 **MediaSizes** 文件中字段条目的示例:

Name	Field Values
Letter	**Width** 612 **Depth** 792 **llx** 18 **lly** 17 **urx** 597 **ury** 776 **Page- Region- Name** Letter **Paper- Tray- Name** Letter
Legal	**Width** 612 **Depth** 1008 **llx** 18 **lly** 17 **urx** 597 **ury** 992 **Page- Region- Name** Legal **Paper- Tray- Name** Legal

PostScript 字体信息

Transcript 表的 PostScript 字体显示了 enscript 命令可用的字体。字体名用 **-F** 和 **-f encscipt** 命令标志来指定。字母字符区分大小写:

Transcript 的 PostScript 字体

字体名	字体系列
AvantGarde-Book	AvantGarde
AvantGarde-Demi	AvantGarde
AvantGarde-DemiOblique	AvantGarde
AvantGarde-BookOblique	AvantGarde
Bookman-Demi	Bookman
Bookman-DemiItalic	Bookman
Bookman-Light	Bookman
Bookman-LightItalic	Bookman
Courier	Courier
Courier-Bold	Courier
Courier-BoldOblique	Courier

Transcript 的 PostScript 字体

字体名	字体系列
Courier-Oblique	Courier
Garamond-Bold	Garamond
Garamond-BoldItalic	Garamond
Garamond-Light	Garamond
Garamond-LightItalic	Garamond
Helvetica	Helvetica
Helvetica-Bold	Helvetica
Helvetica-Oblique	Helvetica
Helvetica-BoldOblique	Helvetica
Helvetica-Narrow	Helvetica
Helvetica-Narrow-Bold	Helvetica
Helvetica-Narrow-BoldOblique	Helvetica
Helvetica-Narrow-Oblique	Helvetica
LubalinGraph-Book	Lubalin
LubalinGraph-BookOblique	Lubalin
LubalinGraph-Demi	Lubalin
LubalinGraph-DemiOblique	Lubalin

字体名	字体系列
Miryam-Iso	Miryam Iso
Miryam-IsoBold	Miryam Iso
Miryam-IsoBoldItalic	Miryam Iso
Miryam-IsoItalic	Miryam Iso
NarkissimIso	Narkissim Iso
NarkissimIso-Bold	Narkissim Iso
NarkissimIso-BoldItalic	Narkissim Iso
NarkissimIso-Italic	Narkissim Iso
NarkissTamIso	Narkiss Tam Iso
NarkissTamIso-Bold	Narkiss Tam Iso
NarkissTamIso-BoldItalic	Narkiss Tam Iso
NarkissTamIso-Italic	Narkiss Tam Iso
NewCenturySchlbk	NewCentury
NewCenturySchlbk-Bold	NewCentury
NewCenturySchlbk-Italic	NewCentury
NewCenturySchlbk-Roman	NewCentury
Optima	Optima
Optima-Bold	Optima
Optima-BoldOblique	Optima
Optima-Oblique	Optima
Palatino-Bold	Palatino

字体名	字体系列
Palatino-BoldItalic	Palatino
Palatino-Italic	Palatino
Palatino-Roman	Palatino
Rokaa	Rokaa
Rokaa-Bold	Rokaa
Rokaa-BoldItalic	Rokaa
Rokaa-Italic	Rokaa

字体名	字体系列
Setting	Setting
Setting-Bold	Setting
Setting-BoldItalic	Setting
Setting-Italic	Setting
ShalomIso	ShalomIso Iso
ShalomIso-Bold	ShalomIso Iso
ShalomIso-BoldItalic	ShalomIso Iso
ShalomIso-Italic	ShalomIso Iso
Souvenir-Demi	Souvenir
Souvenir-DemiItalic	Souvenir
Souvenir-Light	Souvenir
Souvenir-LightItalic	Souvenir
Times-Bold	Times
Times-BoldItalic	Times
Times-Italic	Times
Times-Roman	Times
Typing	Typing
Typing-Bold	Typing
Typing-BoldItalic	Typing
Typing-Italic	Typing
Symbol	(none)
ZapfChancery-MediumItalic	Zapf
ZapfDingbats	(none)

参数

SpoolerOptions 为对打印文件进行假脱机操作提供选项。后面是 *SpoolerOptions* 标志:

{-d | -P}*Queue*
将输出送到指定的队列中排队。

-n*Number*
产生指定的副本编号。缺省为 1。

-t*标题* 在第一个标题页中设置作业标题。

File 指定要被转化成 PostScript 格式的文本文件。如果您将该参数空着，**enscript** 命令会从标准输入中读入。

标志

-1 设置在 1 列中（缺省）。
-2 设置在 2 列中。
-c 截断那些比页宽还长的行。通常长行会被环绕折入页面中的下一行。
-g 不执行功能，但是 **-g** 标志仍然为了向下兼容被接受。
-k 允许预先提供页面（如果打印机支持的话）。通过使打印机在页与页之间保持运行，使得简单文档（如单字体的程序列表）打印得更快。
-l 模拟一个行打印机打印 66 行长的文档并忽略标题。
-m 在文件打印完后发送邮件。
-o 如果 **enscript** 命令不能在字体中找到字符的话，列出丢失的字符。
-q 导致 **enscript** 命令不报告它正在作什么。**enscript** 命令不能报告页面、目标、省略的字符等等。致命错误仍然报告到标准错误输出。
-r 将输出旋转 90 度（landscape 模式）。对要求有更大页宽的输出，或者对与 **-2** 标志结合使用的程序列表使用这个标志。下面的示例显示了获得程序列表的一种方法:

```
enscript -2r File . . .
```

-B 省略页头。
-G 以华丽方式打印，导致页眉、日期和页码以闪耀方式打印，要花费一点点性能的代价。
-K 禁止页面的预先提供（缺省）。
-r 以肖像方式（未旋转）打印，这是缺省方式。
-b*Header* 将用作页眉的字符串设置成 *Header* 变量值。缺省的页眉是从文件名、上次修改日期和页码构造的。
-f*Font* 将字体设置成被每个页面的主体使用。缺省的为 Courier10，除非用到了两列旋转的模式，在这种情况下缺省为 Courier7。

注:

1. 一种 PostScript 字体名 （如 Times-Roman, Times-BoldItalic, Helvetica, Courier）。
2. 点的大小（1 点 = 1/72 英寸）。字体按这种样式指定: Courier-Bold8 就是 8 点 Courier 粗体; Helvetica12 就是 12 点的 Helvetica。

-f0 *Codeset*:*Font* 设置写入到 PostScript 文件中的字符代码集名和 SBCS 字体以对每个页面的主体都使用。缺省的是对每个语言环境用 **/usr/lib/ps/transcript.conf** 配置文件来确定。
-f1 *Codeset*:*Font* 设置写入到 PostScript 文件中的字符代码集名和 MBCS 字体以对每个页面的主体都使用。缺省的是对每个语言环境用 **/usr/lib/ps/transcript.conf** 配置文件来确定。
-p*Out* 导致 PostScript 文件被写入到指定的文件中，而不是假脱机打印。作为一个特例，输入以下命令将会将 PostScript 文件发送到标准输出:

```
-p -
```

-F*Hfont* 将该字体设置成为页眉所用。缺省的是 Courier Bold10。
注: 字体规格有两部分:

- PostScript 字体名（如 Times-Roman, Times-BoldItalic, Helvetica, Courier）。
- 点的大小（1 点 = 1/72 英寸）。字体按这种样式指定: Courier-Bold8 就是 8 点 Courier 粗体; Helvetica12 就是 12 点的 Helvetica。

-F0 *Codeset*:*Font* 设置写入到 PostScript 文件中的字符代码集名和 SBCS 字体以对每个页眉使用。缺省的是对每个语言环境用 **/usr/lib/ps/transcript.conf** 配置文件来确定。

-F1 *Codeset*:*Font* 设置写入到 PostScript 文件中的字符代码集名和 MBCS 字体以对每个页眉使用。缺省的是对每个语言环境用 **/usr/lib/ps/transcript.conf** 配置文件来确定。

-L*Lines* 设置要在一页中打印的最大行数。**enscript** 命令通常会根据点的大小计算需要在一页中放多少行。（可能每页会比 **-L** 标志请求的放得少。）

-M*MediaName* 指定介质名以用来确定纸张上可成像区域的数量。提供的名称与 **MediaSizes** 文件中的条目相匹配。例如, `-M legal` 将请求将纸张的合法大小当作可成像区域。如果不使用该标志, 缺省大小为信纸大小, 即 8.5 英寸宽, 11.0 英寸长（21.6 厘米宽, 27.9 厘米长）。

-X*CodesetName* 为输入数据指定了代码集。缺省情况下, 输入的代码的集合由 **nl_langinfo** 子例程来确定。如果该标志被使用的话, 代码集合用 *CodesetName* 来确定。

国际字符支持

所有在字体中没有找到的字符用 ？字符（问号）来替代。要想获得没有找到的字符的完整的列表, 请使用 **-o** 标志。**NLSvec** 文件提供了关于字符翻译的信息。

环境变量

ENSCRIPT 指定了由 **enscript** 命令使用的选项的字符串。
LPDEST 指定了一个打印机目标。**-d** 假脱机程序选项会覆盖这个环境变量。
PSLIBDIR 为 **enscript** 命令序言和字体度量文件提供一个目录的路径名来代替 **/usr/lib/ps** 目录。
PSTEMPDIR 提供了一个临时目录的路径名来代替假脱机临时文件的 **/var/tmp** 目录。
TRANSCRIPT 为 MBCS 句柄提供了一个要使用的文件的绝对路径名, 来代替 **/usr/lib/ps/transcript.conf** 配置文件。

文件

/usr/lib/ps/*.afm 包含了 Adobe 字体度量文件。
/usr/lib/ps/font.map 包含了字体名及缩写列表。
/usr/lib/ps/enscript.pro 包含了 **enscript** 命令文件的序言。
/usr/lib/ps/MediaSizes 包含了介质大小使用的缺省文件。

相关信息

col 命令、**eqn** 命令、**lp** 命令、**managefonts** 命令、**nroff** 命令、**pic** 命令、**pr** 命令、**ps630** 命令、**psdit** 命令、**refer** 命令、**tbl** 命令、**troff** 命令。

nl_langinfo 子例程。

NLSvec 文件提供了有关字符翻译的信息。

entstat 命令

用途
显示以太网设备驱动器和设备统计信息。

语法
entstat [**-d -r -t**] *Device_Name*

描述
entstat 命令会显示由指定的以太网设备驱动器收集的统计信息。除了设备一般统计信息之外，用户可以有选择地指定要显示的特定于设备的统计信息。如果没有指定任何标志，则只显示设备的一般统计信息。

该命令当 **netstat** 命令与 **-v** 标志一起运行时也会被调用。**netstat** 命令不会发出任何 **entstat** 命令标志。

如果一个非法的 *Device_Name* 被指定的话，**entstat** 命令会产生一个错误消息，说明它不能连接到设备。

标志
-d 显示所有的统计信息，包括设备特定的统计信息。
-r 重新将所有的统计数据设置回它们的初始值。这个标志只能被特权用户调用。
-t 切换某些设备驱动程序中的调试跟踪。

参数
Device_Name 以太网设备名，例如，**ent0**。

统计信息字段
注: 某些适配器可能不支持特定的统计信息。不支持统计数据的字段的值总是 0。

在 **entstat** 命令的输出中显示的统计字段和它们的描述是:

标题字段

Device Type
显示适配器类型的描述。
Hardware Address
显示当前设备使用的以太网地址。
Elapsed Time
显示从上次统计信息被重置之后已经过去的实际的时间周期。当探测到硬件错误时在错误恢复的过程中，统计信息的一部分可能被设备驱动器重新设置。当这种情况发生时，为了反应统计信息之间的时间差，在输出的中间还会有另一个消耗时间被显示。

传送统计字段

Packets
成功的被设备传送的信息包的数量。
Bytes
成功的被设备传送的字节数目。
Interrupts
从适配器中被设备接收的传送中断的数目。
Transmit Errors
在这个设备上出现的输出错误数目。这是为由于硬件或网络错误传送不成功而设立的计数器。

Packets Dropped	设备驱动程序为传送而接受的信息包的数目，这些没有被送入设备（由于某种原因）。
Max Packets on S/W Transmit Queue	排队到软件传送队列的外发信息包的最大数量。
S/W Transmit Queue Overflow	已经溢出软件传送队列的外发包数。
Current S/W+H/W Transmit Queue Length	在软件传送队列或者在硬件传送队列中当前被暂挂的信息包数量。
Broadcast Packets	没有任何错误发送的广播包的个数。
Multicast Packets	未发生任何错误的多点广播的包的个数。
No Carrier Sense	由于无载波侦听错误导致的不成功发送的个数。
DMA Underrun	由于 DMA 欠载运行错误导致的不成功发送的个数。
Lost CTS Errors	由于清除发送（Clear-to-Send）信号丢失错误而导致的不成功发送的。
Max Collision Errors	由于存在太多冲突导致的不成功发送的个数。发生的冲突的个数超过了适配器重试的次数。
Late Collision Errors	由于迟冲突（late collision）错误导致的不成功发送的个数。
Deferred	在发送过程中推迟的外流包个数。被推迟意味着适配器在试图发送一帧时不得不推迟。在适配器准备发送时如果网络忙就会发生这种情况。适配器将只会推迟第一个发送数据包的尝试。此后适配器将无需验证就发送该包。如果网络仍然忙的话，就会记录一个冲突。
SQE Test	包含了在发送过程中成功执行的“信号质量错误”测试（即心跳）的个数。
Timeout Errors	由于适配器报告超时错误而导致的不成功发送的数目。
Single Collision Count	在发送过程中碰到单个冲突的外流包的个数。
Multiple Collision Count	在传送过程中遇到多个（2 到 15 个）冲突的外流包的个数。
Current HW Transmit Queue Length	在硬件传送队列当前存在的外流包个数。
CRC Errors	有校验和（FSC）错误的进站包个数。
DMA Overrun	有 DMA 过载错误的进站包个数。
Alignment Errors	有校准错误的进站包个数。
No Resource Errors	由于无资源错误导致被硬件放弃的进站包个数。这种错误通常发生的原因是适配器上的接收缓冲区被耗光。有些适配器可能将接收缓冲区的尺寸当作可配置的参数。检查设备配置属性（或者 SMIT 帮助）获取可能的调整信息。
Receive Collision Errors	在接收过程中有冲突错误的进站包个数。
Packet Too Short Errors	有包的大小小于以太网最小包大小的长度错误的进站包个数。
Packet Too Long Errors	有包的大小大于以太网最大包大小的长度错误的进站包个数。
Packets Discarded by Adapter	由于其他原因导致被硬件放弃的进站包个数。
Receiver Start Count	在适配器上接收器（接收单元）被启动的次数。

接收统计信息字段

Packets	被设备成功接收的包个数。
Bytes	成功的从设备中接收的字节数。
Interrupts	从适配器中通过驱动程序收到的中断数。
Receive Errors	该设备中发生的输入错误的个数。这是为由于硬件或网络错误产生接收不成功而设立的计数器。
Packets Dropped	驱动程序从此设备接收的、（由于某种原因）未给予网络多路分解器的种原因）。
Bad Packets	设备驱动程序接收（如保存）的坏包个数。
Broadcast Packets	没有任何错误接收的广播信息包数量。
Multicast Packets	没有任何错误接收的多点广播信息包的数量。
CRC Errors	有校验和（FSC）错误的进站包个数。
DMA Overrun	有 DMA 过载错误的进站包个数。
Alignment Errors	有校准错误的进站包个数。
No Resource Errors	由于无资源错误导致的硬件删除的进站包个数。
Receive Collision Errors	在接收过程中有冲突错误的进站包个数。
Packet Too Short Errors	有包的大小小于以太网最小包大小的长度错误的进站包个数。

Packet Too Long Errors	有包的大小大于以太网最大包大小的长度错误的进站包个数。
Packets Discarded by Adapter	由于其他原因导致被硬件放弃的进站包个数。
Receiver Start Count	在适配器上接收器（接收单元）被启动的次数。

一般统计信息字段

No mbuf Errors	对于设备驱动程序 mbufs 得不到的次数。当驱动程序必须获取 mbuf 缓冲区来处理入站信息包的时候，这种情况通常在接收操作期间发生。如果对于要求尺寸的 mbuf 池是空的话，则这个信息包就会废弃。**netstat -m** 命令可以用来确认这点。
Adapter Reset Count	适配器已经被重启动（重新初始化）的次数。
Driver Flags	设备驱动程序当前接通的内部状态标志。

设备特定的统计信息字段

这部分的显示对每种适配器可能都不一样。它可能包含适配器特定的信息和一些未包括在一般统计信息中的扩展统计信息。有些适配器可能没有任何设备特定统计信息。

实例

1. 要想显示 **ent0** 的设备一般统计信息，请输入:

   ```
   entstat ent0
   ```

   这会产生以下输出:

   ```
   ETHERNET STATISTICS (ent0) :
   Device Type: Ethernet High Performance LAN Adapter
   Hardware Address: 02:60:8c:2e:d0:1d
   Elapsed Time: 0 days 0 hours 8 minutes 41 seconds

   Transmit Statistics:     Receive Statistics:
   --------------------     -------------------
   Packets: 3               Packets: 2
   Bytes: 272               Bytes: 146
   Interrupts: 3            Interrupts: 2
   Transmit Errors: 0       Receive Errors: 0
   Packets Dropped: 0       Packets Dropped: 0
   Max Packets on S/W       Bad Packets: 0
   Transmit Queue:0
   S/W Transmit Queue
   Overflow: 0
   Current S/W+H/W Transmit
   Queue Length: 0

   Broadcast Packets: 2     CRC Errors: 0
   Multicast Packets: 0     Broadcast Packets: 1
   No Carrier Sense: 0      Multicast Packets: 0
   DMA Underrun: 0          DMA Overrun: 0
   Lost CTS Errors: 0       Alignment Errors: 0
   Max Collision Errors: 0  No Resource Errors: 0
   Late Collision Errors: 0 Receive Collision Errors: 0
   Deferred: 0              Packet Too Short Errors: 0
   SQE Test: 0              Packet Too Long Errors: 0
   Timeout Errors: 0        Packets Discarded by Adapter: 0
   Single Collision         Receiver Start Count: 1
   Count: 0
   Multiple Collision Count: 0
   Current HW Transmit Queue
   Length: 0

   General Statistics:
   ```

```
------------------
No mbuf Errors: 0
Adapter Reset Count: 0
Driver Flags: Up Broadcast Running Simplex
```

2. 要想显示 **ent0** 的以太网设备一般统计信息和以太网设备特定统计信息，请输入:

```
entstat -d ent0
```

这会产生以下输出:

```
ETHERNET STATISTICS (ent0) :
Device Type: Ethernet High Performance LAN Adapter
Hardware Address: 02:60:8c:2e:d0:1d
Elapsed Time: 0 days 2 hours 6 minutes 30 seconds

Transmit Statistics:      Receive Statistics:
--------------------      -------------------
Packets: 3                Packets: 2
Bytes: 272                Bytes: 146
Interrupts: 3             Interrupts: 2
Transmit Errors: 0        Receive Errors: 0
Packets Dropped: 0        Packets Dropped: 0
Max Packets on S/W        Receiver Start Count: 1
Transmit Queue:0
Bad Packets: 0
S/W Transmit Queue Overflow: 0
Current S/W+H/W Transmit Queue Length: 0

Broadcast Packets: 0      Broadcast Packets: 0
Multicast Packets: 0      Multicast Packets: 0
No Carrier Sense: 0       CRC Errors: 0
DMA Underrun: 0           DMA Overrun: 0
Lost CTS Errors: 0        Alignment Errors: 0
Max Collision Errors: 0   No Resource Errors: 0
Late Collision Errors: 0  Receive Collision Errors: 0
Deferred: 0               Packet Too Short Errors: 0
SQE Test: 0               Packet Too Long Errors: 0
Timeout Errors: 0         Packets Discarded by Adapter: 0
Single Collision Count: 0 Receiver Start Count: 1
Multiple Collision Count: 0
Current HW Transmit Queue Length: 0

General Statistics:
-------------------
No mbuf Errors: 0
Adapter Reset Count: 0
Driver Flags: Up Broadcast Running Simplex

Ethernet High Performance LAN Adapter Specific Statistics:
------------------------------------------------------------
Receive Buffer Pool Size: 37
Transmit Buffer Pool Size: 39
In Promiscuous Mode for IP Multicast: No
Packets Uploaded from Adapter: 0
Host End-of-List Encountered: 0
82586 End-of-List Encountered: 0
Receive DMA Timeouts: 0
Adapter Internal Data: 0x0 0x0 0x0 0x0 0x0
```

相关信息

atmstat 命令、**fddistat** 命令、**netstat** 命令，**tokstat** 命令。

env 命令

用途
显示当前环境或者为命令的执行设置环境。

语法

显示多个环境变量
env [**-i** | **-**] [*Name*=*Value*]... [*Command* [*Argument* ...]]

显示单个环境变量
env [*Name*]

描述
env 命令允许您显示您的当前环境或者在一个被更改了的环境下运行一个指定的命令。

如果没有标志或者参数被指定，**env** 命令会显示您的当前环境，每行显示一个 *Name*=*Value* 对。

标志

-i 忽略继承的环境，调用由 *Command* 参数指定的命令，在 *Name*=*Value* 参数指定的环境下。

参数

Name=*Value* 您可以在通过指定一个或多个 *Name*=*Value* 参数来在当前环境的一种修改版本下运行您的命令。使用 **-i** 标志，如果您想用指定的 *Name* =*Value* 参数代替整个当前环境的话。在其他情况下，环境的更改只有在指定命令正在运行时才有效。

Command *Command* 参数有一个可选的 *Argument* 变量。如果指定的命令是一种 Korn shell 特殊内置命令，结果就是不被指定的。Korn shell 内置命令在 **ksh** 命令中描述。

退出状态
如果 *Command* 参数被指定，**env** 命令的退出状态就是由 *Command* 参数指定的命令的退出状态。否则，**env** 命令用以下值中的一个来退出:

0 **env** 成功完成。
1-**125** 在 **env** 命令中发生了一个错误。
126 由 *Command* 参数指定的命令被发现了，但是不能被调用。
127 由 *Command* 参数指定的命令未被发现。

示例
1. 要想更改 **TZ** 环境变量（在 **date** 命令正运行时），请输入:

```
TZ=MST7MDT date
```

OR

```
env TZ=MST7MDT date
```

每个这种命令都会显示以山地时间显示的时间和当前日期。给出的两个命令是等价的。**date** 命令被完成时，**TZ** 环境变量的以前的值重新生效。

2. 要想在只包含为 **PATH**、**IDIR**、和 **LIBDIR** 环境变量定义的环境中运行 **make** 命令，请输入:

```
env -i PATH=$PATH IDIR=/$HOME/include LIBDIR=/$HOME/lib make
```

您必须指定 **PATH** 环境变量，这样 shell 就能找到 **make** 命令。当 **make** 命令被完成时，前面的环境就会生效。

文件

/usr/bin/env 包含了 **env** 命令。

相关信息

printenv 命令、**ksh** 命令。

环境文件。

概要文件文件格式。

exec 子例程。

《*操作系统与设备管理*》中的『命令』。

《*操作系统与设备管理*》中的『概要文件概述』。

《*操作系统与设备管理*》中的『Shell』。

---

epkg 命令

用途

创建可由临时修订管理器 **emgr** 安装的临时修订（临时修订）软件包。

语法

epkg [**-w** *WorkDirectory*] [**-p** *PrerequisiteFile*] [**-d** *DescriptionFile*] [**-e** *临时修订ControlFile*] [**-g** *PrerequisiteFile*] [**-l** *LockFile*] [**-S** *SupersedeFile*] [**-u** {**y**|**n**}]] [**-r** {**y**|**n**|**o**}] [**-s**] [**-T** {**y**|**n**}] [**-X**] [**-v**] *interim fix Label*

描述

epkg 工具可在两种方式中运行: *交互式*和*基于模板*。交互方式提示您几个问题并基于答案构造临时修订软件包。基于模板的方式使用提供交互方式中所问问题的答案的临时修订控制文件。临时修订软件包由临时修订管理器安装，此管理器由 **emgr** 命令启动。

交互方式

缺省情况下，**epkg** 命令在交互方式中运行。唯一的必需参数是临时修订标号。如果中断 **epkg** 会话，将保存临时修订控制文件。如果用相同的临时修订标号启动新的会话，将询问您是否您要继续使用先前的临时修订控制文件。要在启动交互式 **epkg** 会话之前提供此信息，请运行具有 **-u** 标志的 **epkg** 命令。

epkg 命令保留问题顺序的记录并允许您使用子命令在问题中浏览。同样，**epkg** 命令记住先前您提供的答案并将该答案设置为缺省答案。在子命令部分描述 **epkg** 子命令。

在回答了所有问题后，**epkg** 命令验证临时修订控制文件并创建可用 **emgr** 命令安装的压缩的 tar 软件包。

使用临时修订控制文件模板

可通过将临时修订控制文件用作模板非互动地创建临时修订软件包。以下是完整的临时修订控制文件的示例:

```
#interim fix control file complete example
ABSTRACT=This is a test of epkg.
PRE_INSTALL=/tmp/pre_install
POST_INSTALL=.
PRE_REMOVE=/tmp/pre_remove
POST_REMOVE=.
REBOOT=yes
PREREQ=.
DESCRIPTION=/tmp/description
EFIX_FILES=2

EFIX_FILE:
        EFIX_FILE_NUM=1
        SHIP_FILE=/home/test/ls
        TARGET_FILE=/usr/bin/ls
        TYPE= 1
        INSTALLER= 1
        ACL= DEFAULT
        AR_MEM=.

EFIX_FILE:
        EFIX_FILE_NUM=2
        SHIP_FILE=/home/test/mystrcat.o
        TARGET_FILE=/usr/ccs/lib/libc.a
        TYPE= 2
        INSTALLER= 1
        ACL= root:system:555
        AR_MEM=strcat.o
```

临时修订控制文件值，如下所示:

ABSTRACT
简短描述临时修订软件包。此摘要限制为 38 字节。

PRE_INSTALL
指定在安装预览之后和安装任何临时修订文件之前运行的脚本的位置。PRE_INSTALL 脚本中的失败将引起临时修订软件包安装异常中止。此组件是可选的。

POST_INSTALL
指定在已成功安装了所有临时修订文件后运行的脚本的位置。此组件是可选的。

PRE_REMOVE
指定在安装预览之后和在除去操作过程中除去任何临时修订文件之前运行的脚本的位置。此组件是可选的。

POST_REMOVE
指定在除去操作过程中除去临时修订文件之后运行的脚本的位置。此组件是可选的。

REBOOT
指定此临时修订是否需要重新引导。允许的值有 yes 和 no。如果此值设置为 yes，则 **emgr** 命令将根据需要对引导映像进行更改，并发出消息指示用户在安装之后重新引导。

PREREQ
指定包含 **installp** 先决条件的文件的位置。此组件是可选的。

DESCRIPTION
指定包含正安装的临时修订软件包的详细描述的文件的位置。

EFIX_FILES
指定临时修订中的文件总数。

EFIX_FILE_NUM
指定临时修订中文件的编号。临时修订中的每个文件必须具有唯一的号码，从 1 到 200。**epkg** 命令可以支持每临时修订最多 200 个文件。

SHIP_FILE
指定 **epkg** 将归档入临时修订软件包的文件的位置。可以为此文件指定绝对路径或相对路径。

TARGET_FILE
指定将安装 **SHIP_FILE** 的位置。此位置是系统上将安装临时修订软件包的位置。必须指定此文件的绝对路径。如果此文件的是注册软件包（例如 RPM Package Manager（RPM）或 **installp** 软件包）的一部分，则必须指定跟踪的位置。

TYPE 指定正在安装的文件的类型。有效的选项如下所示:

1 文件（标准或可执行文件）

2 库或压缩文档成员

INSTALLER
指定将跟踪临时修订软件包的安装程序的类型（如果存在）。有效的选项如下所示:

1 当前被 **installp** 跟踪

2 当前被 RPM 跟踪

3 当前被 **ISMP** 跟踪

4 当前被另一个安装程序跟踪

5 这是将被 **installp** 跟踪的新文件。

6 将被 RPM 跟踪的新文件

7 将被 **ISMP** 跟踪的新文件

8 将被另一个安装程序跟踪的新文件

9 不被任何安装程序跟踪

ACL 指定文件的访问属性（方式和所有权）。如果此属性设置为 **DEFAULT**，则 **emgr** 命令保留要替换的文件的当前权限。然而，如果目标文件是新文件或如果用户要用 **-v** 标志指定权限，则 **ACL** 属性可按语法 *Owner*:*Group*:*OctalModes* 输入，类似于以下内容:

```
ACL= root:system:555
```

AR_MEM
指定压缩文档成员的名称。此选项仅当 **TYPE**=2 时有效。此时，**SHIP_FILE** 是要提供的压缩文档成员的本地位置，**TARGET_FILE** 是目标压缩文档，且 **ACL** 适用于该压缩文档成员。例如，以下值设置将使文件 **myshr.o** 成为目标 **/usr/ccs/lib/libc.a** 中的成员 **shr.o**:

```
TYPE=2
SHIP_FILE=/home/myshr.o
TARGET_FILE=/usr/ccs/lib/libc.a
AR_MEM=shr.o
```

BUILD_BOOTIMAGE
指定是否需要重建引导映像。允许的值有 yes 和 no。如果将此字段设置为 yes，则需要进行重新引导。如果将此字段设置为 yes 且将 **REBOOT** 字段设置为 no，则 **epkg** 将返回错误。

E2E_PREREQ
指定临时修订先决条件文件在临时修订控制文件中的位置。

PKGLOCKS
指定程序包锁文件在临时修订控制文件中的本地文件位置。

SUPERSEDE
指定被取代的文件在临时修订控制文件中的本地文件位置。

FIXTESTED
指定该临时修订是否已经过测试。允许的值有 yes 和 no。

支持临时修订替换

该封装程序可以指定包含临时修订标签名称的文件，这些名称在安装 epkg 时将被取代。这将导致 **emgr** 命令在安装临时修订程序包之前除去该文件中指定的任何临时修订标签（如果已安装）。如果未能除去已安装、已取代的临时修订，则将终止临时修订程序包的安装。支持的最大被取代标签数是 32。该封装程序可以用以下方式，使用 **epkg** 命令指定取代文件:

- 使用 **-S** *supersede_file* 标志指定文件位置。示例:

  ```
  epkg -S /tmp/superseded.epkg myefix
  ```

- 如果在交互方式中使用了扩展选项标志（**-v**），则 **epkg** 命令将提示您提供被取代的文件。示例:

  ```
  输入取代文件的位置或输入“.”以跳过。-> /tmp/superseded.epkg
  ```

- 将 SUPERSEDE 属性设置为临时修订控制文件中被取代文件的本地文件位置。示例:

  ```
  SUPERSEDE=/tmp/superseded.epkg
  ```

被取代文件的格式是每行一个要取代的临时修订标签。注释以 # 号开头并将忽略前导空格。示例:

```
# Requisites for efix myefix3
myefix1
myefix2
```

支持临时修订 prereq 和 xreq

该封装程序可以指定包含正在安装的临时修订程序包必需的临时修订的临时修订标签名称的文件。这将导致 **emgr** 命令检查是否安装了临时修订标签（PREREQ）。如果未安装必需软件，则 **emgr** 命令将终止临时修订程序包的安装。用户还可以指定 XREQ 临时修订标签。这将导致 **emgr** 命令不安装临时修订（如果安装了指定的 xreq 临时修订）。

该封装程序可以用以下方式，使用 **epkg** 命令指定临时修订必备文件:

- 使用 **-g** *efix_prereq_file* 标志指定文件位置。示例:

  ```
  epkg -g /tmp/efixprereq.epkg myefix
  ```

- 如果在交互方式中使用了扩展选项标志（**-v**），则 **epkg** 命令将提示您提供临时修订 prereq 文件。示例:

  ```
  输入电子修订必备文件的位置或输入“.”以跳过。-> /tmp/efixprereq.epkg
  ```

- 将 E2E_PREREQ 属性设置为临时修订控制文件中临时修订必备文件的本地文件位置。示例:

  ```
  E2E_PREREQ=/tmp/efixprereq.epkg
  ```

临时修订必备文件条目的格式如下:

```
EfixLabel  RequisiteType: PREREQ/XREQ
```

示例:

```
oldefix1 PREREQ  # Make sure oldefix1 is already installed

oldefix4 XREQ    # Make sure oldefix4 is NOT installed
```

支持的最大临时修订必备软件数是 32。

临时修订输出和拓扑结构

emgr -d 标志显示临时修订程序包的内容和拓扑结构。**-d** 选项将与 **-v** 详细选项一起使用。有效的详细级别为 1-3。

详细级别 1（缺省值）将显示:

- 标签
- 电子修订文件
- 目标位置

详细级别 2 将显示:

- 所有级别 1 的输出
- 摘要
- 重新引导
- PRE-REQUISITES
- PRE_INSTALL
- POST_INSTALL
- PRE_REMOVE
- POST_REMOVE
- 文件类型

详细级别 3 将显示:

- 所有级别 2 的输出
- 封装日期
- VUID
- 大小
- ACL
- CKSUM
- 封装
- 电子修订描述
- 安装脚本的内容（如果为文本文件）

示例:

- 要获取临时修订程序包 `test.102403.epkg.Z` 上的级别 1 详细输出，请输入:

  ```
  emgr -d test.102403.epkg.Z
  ```

- 要获取临时修订程序包 `test.102403.epkg.Z` 上的级别 3 详细输出，请输入:

  ```
  emgr -v3 -d test.102403.epkg.Z
  ```

支持其他封装锁定

封装程序可以指定包含应该用 **emgr** 命令锁定的程序包名以及根据文件所有权自动锁定的程序包名的文件。封装程序必须指定程序包名、安装程序以及程序包锁定操作的类型（ALWAYS/IFINST）。该封装程序可以用以下方式，使用 **epkg** 命令指定程序包锁定文件:

- 使用 **-l** *pkg_locks_file* 标志指定文件位置。示例:

```
epkg -l /tmp/pkglock.epkg myefix
```

- 如果使用了扩展选项标志（**-v**），则 **epkg** 命令将提示您提供程序包锁定文件。示例:

  ```
  输入程序包锁定文件的位置或输入“.”以跳过。-> /tmp/pkglock.epkg
  ```

- 将 PKGLOCKS 属性设置为临时修订控制文件中程序包锁定文件的本地文件位置。示例:

  ```
  PKGLOCKS=/tmp/pkglock.epkg
  ```

程序包锁定文件的格式如下:

PackageName PackageAction PackageType

其中 *PackageName* 是要锁定的程序包名，*PackageAction* 是以下其中之一:

ALWAYS	始终尝试锁定该程序包。未能锁定程序包将导致安装失败。
IFINST	仅当安装了程序包时才尝试锁定该程序包。未能锁定已安装程序包将导致安装失败。

PackageType 是 installp（缺省值）、rpm、ISMP 和 other。

注: 仅支持 installp 锁定。

支持的最大程序包锁定条目数是 32。

示例:

```
bos.rte.lvm ALWAYS installp
bos.games   IFINST installp
```

在以上示例中，**emgr** 命令将始终尝试在安装过程中锁定 **bos.rte.lvm** 并在删除后将其解锁。当且仅当安装了 **bos.games** 时，**emgr** 命令才会将其锁定，该命令还将在删除后将其解锁（如果锁定）。

支持 bosboot 选项

epkg 命令的重新引导选项包括在不重建引导映像的情况下重新引导。

用户可用以下方式指定不带 bosboot 的重新引导:

- **epkg -r** 标志的 o 参数表示重新引导（“仅”）为必需，但是 **emgr** 命令不应该调用 bosboot（即，重建引导映像）。
- 交互方式下的重新引导提示符显示以下选项:

  ```
  Select reboot policy for this efix package:
  1) Reboot is NOT required.
  2) Reboot is required. The boot image will be rebuilt.
  3) Reboot is required. The boot image will NOT be rebuilt.
  ```

- 将临时修订控制文件中的 BUILD_BOOTIMAGE 和 REBOOT 属性设置为 “yes” 或 “no”。支持以下 REBOOT 和 BUILD_BOOTIMAGE 选项:

REBOOT=no & BUILD_BOOTIMAGE=no	重新引导并非必需。
REBOOT=yes & BUILD_BOOTIMAGE=yes	重新引导为必需。将重建引导映像。
REBOOT=yes & BUILD_BOOTIMAGE=no	重新引导为必需。不会重建引导映像。

注: REBOOT=no & BUILD_BOOTIMAGE=yes 将导致 **epkg** 命令出错。

标志

-d *DescriptionFile*	指定包含临时修订描述的文件。
-e *interim fix ControlFile*	指定控制如何构造临时修订的临时修订控制文件。

-g *PrerequisiteFile* 指定包含临时修订标签名的临时修订先决条件文件的位置。在安装临时修订程序包之前需要这些标签。

-l *LockFile* 指定包含程序包名的锁定文件的位置。这些程序包被 **emgr** 命令锁定或自动根据文件所有权锁定。

-p *PrerequisiteFile* 指定包含 installp 先决条件的文件。

-r {y|n|o} 设置 **epkg REBOOT** 属性。这将使 **emgr** 命令进行引导映像和发出指示用户在安装后重新引导的消息必需的更改。**y** 参数指定重新引导和 **bosboot** 是必需的。**n** 实参指定不需要重新引导。**o** 实参表示需要重新引导，但 **emgr** 不应该调用 **bosboot**。

-S *SupersedeFile* 指定包含临时修订标签名的临时修订取代文件的位置。安装 **epkg** 时将取代这些标签。

-s 使 **epkg** 命令跳过有关脚本和先决条件文件的问题。

-t 指定该临时修订是否已进行过测试。允许的值有 yes 和 no。缺省值为 no。

-u {**yes**|**no**} 指定您是否使用现有临时修订控制文件。

-v 使 **epkg** 命令询问更多扩展选项的问题。这包括要求您在所有临时修订文件上指定权限。

-w *WorkDirectory* 指定 **epkg** 命令将使用的备用工作目录。缺省工作目录是 **$HOME/epkgwork**。

-X 使 **emgr** 命令在安装了临时修订后，需要空间且可能进行扩展时自动扩展文件系统。

参数

interim fix Label

指定唯一标识该临时修订软件包的字符串。临时修订标号的最大长度是 10 个字节。

注: 临时修订管理器要求系统上的每个临时修订标号是唯一的。

子命令

b! 返回到前一个问题。

s! 显示当前临时修订控制文件的状态

q! 退出但不保存临时修订控制文件。（使用 Ctrl+C 按键顺序使 **epkg** 命令询问您是否要保存临时修订控制文件。）

h! 显示当前问题的帮助信息。

退出状态

0 **epkg** 命令操作成功完成。

>0 发生错误。

示例

1. 要在交互方式中运行 **epkg** 命令并创建临时修订标号为 **myfix** 的临时修订软件包，请输入:

   ```
   epkg myfix
   ```

2. 要使用名为 **/tmp/ecfile** 的现有的临时修订控制文件创建具有临时修订标号为 **myfix** 的临时修订软件包，请输入:

   ```
   epkg -e /tmp/ecfile myfix
   ```

3. 要创建临时修订标号为 **myfix** 的临时修订软件包并指定先决条件文件 **/tmp/prereq**、描述 **/tmp/description** 和扩展的选项，请输入:

   ```
   epkg -v -p /tmp/prereq -d /tmp/description myfix
   ```

文件

/usr/sbin/epkg 包含 **epkg** 命令。

相关信息

emgr 命令。

《*安装与迁移*》中的『安装可选软件产品和服务更新』。

---

eqn 命令

用途

为 **troff** 命令对数学文本进行格式化。

语法

eqn [**-d** *Delimiter1Delimiter2*] [**-f** *Font*] [**-p** *Number*] [**-s** *Size*] [**-T** *Name*] [**—**] [*File ...* | **-**]

描述

eqn 命令是一种在照相排字机上对数学文本进行排字的 **troff** 预处理器或可比较设备。**eqn** 命令的输出通常送到 **troff** 命令中，如下所示:

eqn [*Flag...*] *File...* | **troff** [*Flag...*] | [*Typesetter*]

eqn 命令读取由 *File* 参数指定的文件。当 - （负号）被指定成最后的参数时，它会读入标准输入。用 **.EQ** 宏开始的一行标志方程文本开头。方程文本的结尾会被以 **.EN** 宏开头的一行做上标记。这些行不能用 **troff** 命令更改，因此它们可以在宏软件包中定义，用来提供如对中、编号等附加的格式化命令。

关键字

下面是对于 **eqn** 和 **neqn** 命令的关键字。

```
above     dot      gsize     over      tdefine
back      dotdot   hat       pile      tilde
bar       down     italic    rcol      to
bold      dyad     lcol      right     under
ceiling   fat      left      roman     up
ccol      floor    lineup    rpile     vec
col       font     lpile     size
cpile     from     mark      sqrt
define    fwd      matrix    sub
delim     gfont    ndefine   sup
```

由 **eqn** 命令识别的关键字可以用空格、跳格符、换行符、括号、双引号、代字号和长音符号隔开。用 { } （大括号） 进行分组；在任何您可以使用单字符如 X 的地方，您都可以替换包括在大括号中的复杂结构。~ （代字号） 代表了输出中的一个全角的空格，而 ^ （长音符） 代表半角空格。

用 **sub** 和 **sup** 关键字产生出下标和上标。用 **over** 关键字产生分式。用 **sqrt** 关键字产生平方根。

用 **from** 和 **to** 关键字产生下限和上限。用 **left** 和 **right** 关键字产生定界符（如左右方括号和大括号）。在 **left** 和 **right** 关键字后面的有效字符为大括号、方括号、竖线、**c** 和加上划线和下划线用的 **f**，和什么都没有 **" "**（双引号）（对只有右边的中括号有用）。**left** 字符不需要一个匹配的 **right** 字符，但是 **right** 字符必须有一个匹配的 **left** 字符。

垂直排列（堆积）用 **pile**、**lpile**、**cpile** 和 **rpile** 关键字完成。堆积可以有任意个元素。**lpile** 关键字左调整 **pile** 和 **cpile** 关键字居中（可以有不同的垂直空间）**rpile** 关键字右调整。矩阵用 **matrix** 关键字产生。另外还有 **rcol** 关键字用来作为列的右调整。

区别标志用 **dot**、**dotdot**、**hat**、**tilde**、**bar**、**vec**、**dyad** 和 **under** 关键字产生。

使用 **size** *Number*（或 **size +/-***Number*）、**roman**、**italic**、**bold** 和 **font** *Number* 关键字来更改点大小和字体。您可以在一个文档中用 **gsize** *Number* 和 **gfont** *Number* 关键字或者用命令行的 **-s***Number* 和 **-f***Number* 标志来更改所有的点的大小和字体。

通常上标和下标比以前的大小减小 3 个点。您可以用命令行 **-p***Number* 标志来更改它。

您可以对连续显示的参数排队。将 **mark** 关键字放在第一个方程的目标排队点之前；将 **lineup** 关键字放在它要在后面的方程中垂直排列的位置。

您可以用 **define** 关键字定义缩写或重新定义已经存在的关键字；例如:

```
define  Thing%Replacement%
```

前面的示例定义了一个新的名为 *Thing* 的标志，在后面它一旦出现就用 *Replacement* 替代。% （百分号）可以是任何不在 *Replacement* 中的字符。

象 **sum**、**int**、**inf** 关键字和 **如 >=**、**!=** 以及 **->** 这样的缩写都被承认。希腊字母可以按照要求的大小写拼写出来，就像在 **alpha** 或者 **GAMMA** 中一样。如 **sin**、**cos** 和 **log** 之类的数学字符自动写成 Roman 字体。**troff** 命令的 4 字符的转义，如产生双脚注标号的 **\(dd** 可以在任何地方使用。包括在 **" "**（双引号）中的字符串要一点不变地传递。这就允许关键字以文本形式输入，并且总是用来与 **troff** 命令通信。

标志

-d*Delimiter1Delimiter2*
将两个 ASCII 字符，*Delimiter1* 和 *Delimiter2,* 设置成要由 **eqn** 命令处理的文本的定界符，此外还有 **.EQ** 和 **.EN** 宏括起来的输入。这些定界符之间的文本被当成 **eqn** 命令的输入。

> **注**: 在一个文件内，您也可以为 **eqn** 文本设置定界符，用 **delim** *Delimiter1Delimiter2* 命令实现。它们可以用 **delim off** 命令关闭。所有不在 **.EQ** 和 **.EN** 宏之间的文本要一点不变地传递。

-f*Font*
将所有 **eqn** 命令处理的文本中的字体更改成由 *Font* 变量指定的值。*Font* 值（字体名或者位置）必须是一个或两个 ASCII 字符。

-p*Number*
将上标和下标的大小减小指定的点数（缺省为 3）。

-s*Size*
将 **eqn** 命令处理的所有的文本的字体大小更改成由 *Size* 变量指定的值。

-t*Name*
为指定的打印设备准备输出。照相排字机或者可比较的设备的终端名提供了 *Name* 变量。缺省的是 **ibm3816**。

-
强制输入从标准输入中读取。

—
（双划线）表示了标志的结束。

文件

/usr/share/lib/pub/eqnchar
包含了特殊字符的定义。

相关信息

checkeq 命令、**mmt** 命令、**mvt** 命令、**neqn** 命令、**nroff** 命令、**tbl** 命令、**troff** 命令。

eqnchar 的文件格式包含了为 **eqn** 和 **neqn** 命令的特殊字符定义。

.EQ 和 **.EN** 宏、**mm** 宏软件包以及 **mv** 宏软件包。

---

errclear 命令

用途

从错误日志中删除记录。

语法

errclear [**-d** *ErrorClassList*] [**-i** *File*] [**-J** *ErrorLabel* [*,Errorlabel*]] | [**-K** *ErrorLabel* [*,Errorlabel*]] [**-l** *SequenceNumber*] [**-m** *Machine*] [**-n** *Node*] [**-N** *ResourceNameList*] [**-R** *ResourceTypeList*] [**-S** *ResourceClassList*] [**-T** *ErrorTypeList*] [**-y** *FileName*] [**-j** *ErrorID* [**,***ErrorID*]] | [**-k** *ErrorID* [**,***ErrorID*]] *Days*

描述

errclear 命令删除那些比由 *Days* 参数指定的天数还旧的错误日志记录。要删除所有的错误日志记录，指定 **0** 给 *Days* 参数即可。

如果 **-i** 标志没有和 **errclear** 命令一起使用，被 **errclear** 清除的错误日志文件就是在错误日志配置数据库中指定的那一个。（要想查看在错误日志配置数据库中的信息，请使用 **errdemon** 命令。）

> 注: **errclear** 命令清除指定的记录，但是并不减少错误日志文件的大小。

在基于 Web 的系统管理器（wsm）下，您可以使用系统应用程序来更改系统特征。也可以用系统管理界面工具（SMIT）**smit errclear** 快捷方式来运行此命令。

标志

-d *List*
删除由变量 *List* 指定的错误类中错误日志记录。*List* 变量的值可以用 ,（逗号）分隔或用 ″″（双引号）括起并用 ,（逗号）或空格分隔开。有效的 *List* 变量值是 **H** （硬件）、 **S** （软件）、**O** （**errlogger** 消息）和 **U** （未确定）。

-i *File*
使用由变量 *File* 指定的错误日志文件。如果不指定此标志，**errclear** 命令采用错误日志配置数据库中的值。

-j *ErrorID*[,*ErrorID*]
删除由变量 *ErrorID*（错误标识）指定的错误日志记录。*ErrorID* 变量的值可以用 ,（逗号）分隔或用 ″″（双引号）括起并用 ,（逗号）或空格分隔开。

-J *ErrorLabel*
删除由变量 *ErrorLabel* 指定的错误日志记录。*ErrorLabel* 变量的值可以用 ,（逗号）分隔或用 ″″（双引号）括起并用 ,（逗号）或空格分隔开。

-k *ErrorID*[,*ErrorID*]
删除除了由变量 *ErrorID* （错误标识符）指定的之外所有错误日志记录。*ErrorID* 变量的值可以用 ,（逗号）分隔或用 ″″（双引号）括起并用 ,（逗号）或空格分隔开。

-K *ErrorLabel*
删除除了由变量 *ErrorLabel* 指定之外的所有错误日志记录。*ErrorLabel* 变量的值可以用 ,（逗号）分隔或用 ″″（双引号）括起并用 ,（逗号）或空格分隔开。

-l *SequenceNumber* 用指定的序列号删除错误日志记录。*SequenceNumber* 变量的值可以用，（逗号）分隔或用 ″″（双引号）括起并用，（逗号）或空格分隔开。

-m *Machine* 对由变量 *Machine* 指定的机器删除错误日志记录。**uname -m** 命令返回 *Machine* 变量的值。

-n *Node* 删除由变量 *Node* 指定的节点的错误日志记录。**uname -n** 命令返回 *Node* 变量的值。

-N *List* 删除由变量 *List* 指定源名字的错误日志记录。*List* 变量是一个已经检测到错误的源名字的列表。对于软件错误，这些就是检测到错误的资源名。对于硬件错误，就是设备名或系统组件名。这并不表示组件是错误的或者需要更换。相反，它被用来确定要用合适的诊断模块来分析错误。*List* 变量的值可以用，（逗号）分隔或用 ″″（双引号）括起并用，（逗号）或空格分隔开。

-r *List* 删除由变量 *List* 指定的资源类型的错误日志记录。对硬件错误 *List* 变量是一个设备类型。对软件错误来说，*List* 变量的值是 **LPP**。*List* 变量的值可以用，（逗号）分隔或用 ″″（双引号）括起并用，（逗号）或空格分隔开。

-S *List* 删除 *List* 变量指定的资源类的错误日志记录。对于硬件错误，*List* 变量是一个设备类。*List* 变量的值可以用，（逗号）分隔或用 ″″（双引号）括起并用，（逗号）或空格分隔开。

-t *List* 删除由变量 *List* 指定的错误类型的错误日志记录。有效的 *List* 变量值有：**PERM**、**TEMP**、**PERF**、 **PEND**、**INFO** 和 **UNKN**。*List* 变量的值可以用，（逗号）分隔或用 ″″（双引号）括起并用，（逗号）或空格分隔开。

-y *FileName* 使用由变量 *FileName* 指定的错误记录模板文件。

安全性

访问控制：只有 root 用户才能运行这个命令。

示例

1. 要从错误日志中删除所有记录，请输入：

   ```
   errclear 0
   ```

2. 要从错误日志中删除所有软件错误类的条目，请输入：

   ```
   errclear -d S 0
   ```

3. 要从备用的错误日志文件 /var/adm/ras/errlog.alternate 中删除所有记录，请输入：

   ```
   errclear -i /var/adm/ras/errlog.alternate 0
   ```

4. 要从备份的错误日志文件 /var/adm/ras/errlog.alternate 中删除所有硬件记录，请输入：

   ```
   errclear -i /var/adm/ras/errlog.alternate -d H 0
   ```

文件

/etc/objrepos/SWservAt 包含软件服务帮助属性对象类，也就是错误日志配置数据库。

相关信息

errdead 命令，**errinstall** 命令，**errlogger** 命令，**errmsg** 命令、 **errpt** 命令，**errstop** 命令，**errupdate** 命令，**uname** 命令。

errdemon 守护程序。

errsave 内核服务。

errlog 子例程。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『错误日志概述』。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

---

errctrl 命令

用途
更改系统组件的错误检查参数。

语法
errctrl -P errcheckon | errcheckoff

描述
errctrl 命令更改所有系统组件的内核错误检查缺省参数。

注: 在执行 **bosboot** 命令之后，可以通过系统引导持久进行启用和禁用错误检查功能。

标志

-P 持久应用此命令。要持久地在每次系统重新引导时应用该命令，请运行 **bosboot** 命令。

参数

errcheckon 启用所有系统组件的错误检查功能的关键字。
errcheckoff 禁用所有系统组件的错误检查功能的关键字。

退出状态

0 成功完成。
-1 发生错误。

安全性
访问控制: 只有 root 用户可以运行此命令。

示例
1. 要打开所有系统组件的错误检查功能，请输入:

```
errctrl errcheckon
```

位置
/usr/sbin/errctrl

相关信息

bosboot 命令。

---

errdead 命令

用途

从系统转储解压错误记录。

语法

/usr/lib/errdead [**-i** *FileName*] *DumpFile*

描述

errdead 命令从含有内部缓冲区的系统转储中解压错误记录，此缓冲区由 **/dev/error** 文件维护。**errdead** 命令从转储文件中解压错误记录并直接将这些错误记录添加到错误日志中。

在 **errdead** 命令运行时不必运行错误日志守护程序。

标志

-i *FileName* 向由变量 *FileName* 指定的错误日志文件中添加解压过的错误记录。如果此文件不存在，**errdead** 命令创建之。如果不指定标志，使用错误日志配置数据库中的值。

安全性

访问控制: 只有 root 用户才能运行这个命令。

示例

要从文件 /dev/hd7 中驻留的转储映像捕捉错误日志信息，请输入:

```
/usr/lib/errdead /dev/hd7
```

当转储发生时，如果 **errdemon** 守护程序没有运行，错误日志消息在转储映像中。

文件

/etc/objrepos/SWservAt 包含了软件服务辅助属性对象类；即错误日志配置数据库。

相关信息

errclear 命令、**errinstall** 命令、**errlogger** 命令、**errmsg** 命令、**errpt** 命令、**errstop** 命令、**errupdate** 命令。

errdemon 守护程序。

errsave 内核服务。

errlog 子例程。

errdemon 守护程序

用途

启动错误日志记录守护程序（**errdemon**），并将条目写入错误日志。

语法

errdemon [[**-B** *BufferSize*] [**-d** | **-D**] [**-i** *File*] [**-s** *LogSize*] [**-t** *Time*] [**-m** *MaxDups*] | **-l**]

描述

错误日志记录守护程序从 **/dev/error** 文件中读取错误记录，并在系统错误日志中创建错误日志条目。除了在每次记录错误时将条目写入系统错误日志之外，错误日志记录守护程序还会执行错误通知数据库中指定的错误通知。**/etc/objrepos/errnotify** 文件为错误通知数据库。缺省系统错误日志保存在 **/var/adm/ras/errlog** 文件中。最新的错误条目放置在非易失随机存取存储器（NVRAM）中。系统启动期间，将从 NVRAM 中读取这条最新的错误条目，并在错误日志记录守护程序启动时添加到错误日志中。

如果错误记录模板指定 `Log=FALSE`，则错误日志记录守护程序不会为所记录的错误创建错误日志条目。

如果不带任何标志地使用错误日志记录守护程序，则系统将使用错误日志配置数据库中存储的配置值重新启动错误日志记录守护程序。缺省情况下，在极快地记录了重复的错误日志条目时，**errdemon** 守护程序会除去这些重复条目。这是为了防止失控的错误日志记录对系统性能产生不利影响。重复条目的数量可通过详细错误报告看到。

使用 **errclear** 命令从系统错误日志中除去条目。

> **注:** 错误日志记录守护程序通常是在系统初始化期间启动的。停止错误日志记录守护程序可能会使错误数据在可记录到错误日志文件中之前，先临时存储在将被覆盖的内部缓冲区中。

标志

-B *BufferSize*
对错误日志设备驱动程序的内存中缓冲区使用 *BufferSize* 参数所指定的字节数。指定的缓冲区大小保存在错误日志配置数据库中。如果 *BufferSize* 参数大于当前所用的缓冲区大小，则内存中缓冲区立即增大。如果 *BufferSize* 参数小于当前所用的缓冲区大小，则新的大小将在系统重新引导后下一次错误日志记录守护程序启动时生效。不能使缓冲区小于硬编码的缺省值 8KB。

如果此参数未指定，则错误日志记录守护程序将使用错误日志配置数据库中的缓冲区大小。

您指定的大小将向上取整为内存页大小（4KB）的下一个整数倍。用于错误日志设备驱动程序的内存中缓冲区的内存不可用于其他进程。（该缓冲区固定）。请注意，请勿使缓冲区过大而影响系统性能。另一方面，如果使缓冲区过小，则如果错误条目到达的速度快于从缓冲区中读取条目再写入日志文件的速度，则缓冲区可能会变满。当缓冲区满时，将丢弃新的条目，直到缓冲区中有空间变为可用为止。当发生此情况时，错误日志记录守护程序将创建一个错误日志条目以告知您出现的问题。增大缓冲区可纠正该问题。

-d
指定不可除去重复的错误日志条目。缺省行为是除去重复条目，这是通过 **-D** 标志指定的。

-D
指定要除去重复条目。这是缺省值。

-i *File*
使用由 *File* 变量指定的错误日志文件。指定名称的文件保存在错误日志配置数据库中，并且立即投入使用。

-l 显示错误日志配置数据库中的错误日志文件名、文件大小和缓冲区大小的值，以及重复处理值。

-m *MaxDups* 指定在强制除去重复错误之前允许的最大重复条目数。缺省值为 1000。当一条错误的重复次数达到 *MaxDups* 中指定的值时，重复错误将象记录唯一错误那样写入。*MaxDups* 的允许值为 1 到 2147483647。

-s *LogSize* 将 *LogSize* 变量所指定的大小用作错误日志文件的最大大小。指定的日志文件大小限制保存在错误日志配置数据库中，并且立即投入使用。如果该日志文件大小限制小于当前使用的日志文件的大小，则错误日志记录守护程序将重命名当前日志文件，在该文件名的后面加上 **.old**。错误日志记录守护程序将以指定的大小限制创建新的日志文件。使用 **errpt** 命令的 **-i** 标志可从旧日志文件中生成报告。

如果不指定此参数，则错误日志记录守护程序将使用错误日志配置数据库中的日志文件大小。

-t *Time* 指定一个大概的时间间隔（单位：毫秒），在这段时间内，如果某个错误与前个错误完全相同，则视作重复。在该时间间隔后发生的错误即使与前个错误完全相同，也不视为重复。缺省时间间隔为 10000，即 10 秒钟。Time 的允许值为 1 到 2147483647。

注： 此标志用于在错误记录器快速记录同一错误（这通常指示了循环条件）的情况下消除重复条目。它并不旨在捕获可能存在对应的错误通知对象的所有重复错误。使该值足够大可能会因为消除了过多错误而影响到错误通知。请参阅 **errpt** 命令以获得有关在错误报告中消除重复错误的描述。

安全性

访问控制：只有 root 用户可运行此守护程序。

示例

1. 要启动错误日志记录守护程序，请输入：

   ```
   /usr/lib/errdemon
   ```

2. 要查看当前的最大错误日志大小，请输入：

   ```
   /usr/lib/errdemon -l
   ```

3. 要将当前最大错误日志大小从 1MB 更改为 64KB，请输入：

   ```
   /usr/lib/errdemon -s 65536
   ```

4. 要只将在最后 10 毫秒内记录的错误视为重复，请输入：

   ```
   /usr/lib/errdemon -t 10
   ```

文件

/dev/error	错误记录的源文件。
/var/adm/ras/errtmplt	包含错误模板存储库。
/usr/lib/errdemon	包含 **errdemon** 守护程序。
/etc/objrepos/SWservAt	包含了软件服务辅助属性对象类；即错误日志配置数据库。

相关信息

errclear 命令、**errdead** 命令、**errinstall** 命令、**errlogger** 命令、**errmsg** 命令、**errpt** 命令、**errstop** 命令和 **errupdate** 命令。

errsave 内核服务。

error logging 特殊文件。

errlog 子例程。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『错误日志记录概述』。

---

errinstall 命令

用途

在错误日志消息集中安装消息。

语法

errinstall [**-c**] [**-f**] [**-q**] [**-z** *FileName*] *File*

描述

errinstall 命令是一个安装帮助，可以帮助您在错误日志消息编目的错误描述、可能原因、用户原因、安装原因、失败原因、推荐操作和详细数据 ID 消息集中添加或替换消息。

File 参数指定了含有要被添加或替换消息的输入文件。如果不指定 *File* 参数或将其指定为 **-**（减号），则 **errinstall** 命令从标准输入读取。

> **注**: 程序产品和内部应用程序应该使用错误日志消息集中的预定义消息。要列出预定义消息，用 **errmsg -w** 命令。要添加新消息，第三方软件供应商应该和 IBM 解决方案开发者联系来注册新消息。在内部应用程序的开发中，可以用 **errmsg** 命令来添加新消息，但是一定不能和为其他内部应用程序添加的消息冲突。

撤销功能

errinstall 命令在当前目录下创建一个撤销文件，名为 *File*.**undo** 文件。（如果 **errinstall** 命令从标准输入读入消息，则撤销文件的信息被写到标准输出。）*File*.**undo** 文件可以作为 **errinstall** 命令的输入来使用，用来撤销 **errinstall** 命令刚刚产生的更改。要撤销更改，运行 **errinstall** 命令，带有 **-f** 标志并指定 *File*.**undo** 文件为 *File* 参数。

输入文件（或标准输入）文件格式

要求有两行单独的信息来添加或替换错误日志消息编目中的单一的消息。您可以在单一文件中包含多条附加消息或替代消息。第一行是用来指定要被添加或替换到哪一个消息集的，用下面的格式:

```
SET MessageSetID
```

这里 *MessageSetID* 参数是以下单字符之一:

E 识别错误描述
P 识别可能原因
U 识别用户原因
I 识别安装原因
F 识别失败原因
R 识别推荐操作
D 识别详细信息数据

第二行列出要被添加或替换的消息和消息 ID。至少要求一行，也可以包含多行，跟在标志消息集的单行后面。如前面所说，用户应该和他们的服务代表联系来获得消息 ID，除非此消息只在内部应用程序中使用（在此情况下，用 **errmsg** 命令来安装不带预定义错误消息 ID 的错误消息）。

必须在消息 ID 和消息文本之间加一个空格，并且将消息文本用双引号括起来，如下:

```
消息 ID "消息文本"
```

除了必须的两行信息行以外，还可以包含注释行。一个注释行必须在第一列以 $ （美元符号）或*（星号）开头。优先选择星号开头。

> **注:** 添加到错误描述、可能原因和详细数据 ID 消息集中的消息不能超过 40 个字符长度。添加到用户原因、安装原因、失败原因和推荐操作消息集里的消息在长度上不能超过 128 个字符。如果超长，**errinstall** 命令会显示一个警告消息，但将消息添加到代码点目录里。这些消息会在用 summary errpt 命令显示时被截断。

标志

-c	检查输入的 *File* 参数的语法错误。
-f	替换有重复 ID 的消息。当试图用已经使用的消息 ID 来添加一条消息时，**-f** 标志强迫 **errinstall** 命令用新的消息文本替换旧的消息文本。如果不指定 **-f** 标志，则不替换旧的消息文本，并写警告消息到标准错误。**-f** 标志在撤销一个消息安装时也需要带上。
-q	禁止创建 **undo** 文件。
-z *FileName*	使用由参数 *FileName* 指定的错误日志消息编目。

安全性

访问控制: 只有 root 用户才能运行这个命令。

示例

1. 要为已经注册的产品 lpp 安装错误日志消息，输入:

   ```
   errinstall -f /tmp/lpp.desc
   ```

2. 要撤销由上面的示例 **errinstall** 命令对错误日志消息编目所作的更改，请输入:

   ```
   errinstall -f /tmp/lpp.desc.undo
   ```

3. 要在可能原因消息集中安装一错误消息，请输入:

   ```
   errinstall

   * Add a probable cause for widget failure:
   SET P
   E100 "widget adapter"
   ```

4. 要在可能原因消息集中用重复 ID 替换一条消息，请输入:

   ```
   errinstall -f

   * Replace the message associated with ID E100 in the
   * Recommended Action message set
   SET R
   E100 "Replace disk drive"
   ```

5. 如果命名输入文件为 **in_file** 然后想用它来安装新的错误消息，请输入:

   ```
   errinstall in_file
   ```

6. 要覆盖消息集中存在的错误消息，使用在 **in_file** 文件中以前定义的 ID 号码，并且指定带有的 **-f** 标志 **errinstall** 命令如下:

   ```
   errinstall -f in_file
   ```

7. 下面的示例说明要安装的输入文件的样本内容。

```
*
* Add these error messages to the Detailed Data message set:
*
SET D
8105 "Logical channel number"
8106 "Timer reference stamp"
*
* Add these error messages to the Probable Cause message set:
*
SET P
E861 "Bad memory card"
E865 "Unexpected System Halt"
E876 "Fiber Optic Cable"
*
* Add this message to the Recommended Action message set:
*
SET R
E850 "Install updated driver code"
```

文件

/usr/lib/nls/msg/$LANG/codepoint.cat	包含出错日志消息编目。在美国，环境变量 **$LANG** 的值是 **En_US**。

相关信息

errclear 命令、**errdead** 命令、**errlogger** 命令、**errmsg** 命令、**errpt** 命令、**errstop** 命令、**errupdate** 命令。

errdemon 守护程序。

errsave 内核服务。

errlog 子例程。

error logging 特殊文件。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『错误日志概述』

---

errlogger 命令

用途

记录操作程序信息。

语法

errlogger *Message*

描述

errlogger 命令创建一个操作程序错误日志条目，此条目包含可以长达 1024 字节的操作程序信息。

安全性

访问控制: 只有 root 用户才能运行这个命令。

示例

要为系统驱动重新配置创建操作程序信息，请输入:

```
errlogger system drive reconfigured
```

相关信息

errpt 命令。

errsave 内核服务。

errlog 子例程。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『错误日志概述』。

---

errmsg 命令

用途

在出错日志消息编目里添加一条消息。

语法

errmsg [**-c**] [**-z** *FileName*] [**-w** *Set_List* | *File*]

描述

errmsg 命令更新并显示出错日志消息编目，包括错误描述、可能的原因、用户原因、安装原因、失败原因、推荐的操作和详细的数据 ID 消息集。

那些对其要添加或删除消息的消息集在输入 *File* 参数里按如下方式列出:

***** 或 **$**	注释行在第一列必须用 * （星号）或 $（美元符号）注释符号。一般常选 *。
+	要添加的消息必须以 + （加号）开头。
-	要删除的消息必须以 - （减号）开头。
SET	消息集 ID。
″**Message Text**″	消息文本必须用双引号括起来。
Message ID	要删除的消息的消息 ID。

添加到出错描述、可能原因和详细数据 ID 消息集的消息在长度上不能超过 40 个字符。添加到用户原因、安装原因、失败原因和推荐操作消息集里的消息在长度上不能超过 128 个字符。最多可以向每个消息集里添加 2047 个用户自定义消息。

errmsg 命令被应用程序开发者用来创建在错误记录模板库中用到的新消息。如果可能的话，已存在的消息应该一直可用。

如果在命令行中没有指定标志，缺省的操作是更新。在输入的参数 *File* 里指定更新。如果输入的参数 *File* 没有指定或指定 **-**（减号）代替 *File* 参数，则 **errmsg** 命令从标准输入读取。对于每一条添加的消息，**errmsg** 命令分配一标识符。而且，在向消息编目添加消息外 **errmsg** 命令将此标识符和消息文本写到 *File***.out** 文件里。*File***.out** 在从消息编目删除消息时此文件也被创建。如果 **errmsg** 命令正在从标准输入中读取时，标识符和消息文本就被写到标准输出。

标志

-c 检测输入文件有无语法错误。

-w *Set_List* 显示由变量 *Set_List* 指定的错误日志消息集。此选项显示包含在出错日志消息集中的消息及其标识符。输出到标准输出。*Set_List* 变量可以用逗号分隔或者在双引号里用逗号或空格分隔。*Set_List* 变量是消息集 ID 或者如果 *Set_List* 变量**的值 all** 被指定的话，所有的错误日志消息集的内容都会显示出来。变量 *Set_List* 的有效值有:

all 显示所有消息集

D 显示详细的数据 ID 消息集

E 显示错误描述消息集

F 显示失败原因消息集

I 显示安装原因消息集

P 显示可能原因消息集

R 显示推荐操作消息集

U 显示用户原因消息集

-z *Filename* 使用由变量 *Filename* 指定的出错日志消息编目。

安全性

访问控制: 只有 root 用户才能运行这个命令。

示例

1. 要从可能原因消息集中删除消息，请输入:

```
errmsg
* Delete messages FF1A, FF1B, and FF1C from the Probable Cause
* message set
SET P
- FF1A
- FF1B
- FF1C
```

2. 要向可能原因消息集中添加一条小窗口失败错误的消息，请输入:

```
errmsg
* Add a Probable Cause for Widget Failure
SET P
+ "WIDGET ADAPTER"
```

File

/usr/lib/nls/msg/$LANG/codepoint.cat 包含出错日志消息编目。在美国，变量 **$LANG** 的值为 **En_US**。

相关信息

errclear 命令、**errdead** 命令、**errinstall** 命令、**errlogger** 命令、**errpt** 命令、**errstop** 命令、**errupdate** 命令。

errdemon 守护程序。

errsave 内核服务。

errlog 子例程。

error logging 特殊文件。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『错误日志概述』

---

errpt 命令

用途

生成一个记录下来的错误的报表。

语法

处理从错误日志得来的报表

errpt [**-a**] [**-A**] [**-c**] [**-d** *ErrorClassList*] [**-D**] [**-e** *EndDate*] [**-g**] [**-i** *File*] [**-l** *File*] [**-j** *ErrorID* [**,***ErrorID*]] | [**-k** *ErrorID* [**,***ErrorID*]] [**-J** *ErrorLabel* [**,***ErrorLabel*]] | [**-K** *ErrorLabel* [**,***ErrorLabel*]] [**-l** *SequenceNumber*] [**-m** *Machine*] [**-n** *Node*] [**-s** *StartDate*] [**-F** *FlagList*] [**-N** *ResourceNameList*] [**-P**] [**-r** *ResourceTypeList*] [**-S** *ResourceClassList*] [**-t** *ErrorTypeList*] [**-y** *File*] [**-z** *File*]

处理从错误记录模板资源库得来的报表

errpt [**-a**] [**-A**] [**-l** *File*] [**-t**] [**-d** *ErrorClassList*] [**-j** *ErrorID* [**,***ErrorID*]] | [**-k** *ErrorID* [**,***ErrorID*]] [**-J** *ErrorLabel* [**,***ErrorLabel*]] |
[**-K** *ErrorLabel* [**,***ErrorLabel*]] [**-F** *FlagList*] [**-P**] [**-t** *ErrorTypeList*] [**-y** *File*] [**-z** *File*]

描述

errpt 命令会从错误日志中的记录生成一个错误报表。它包含了所选符合特定标准的错误的标志。利用缺省的条件，您可以以错误发生及被记录相反的顺序显示错误日志的记录。利用 **- c** （并行）标志，您可以在错误发生时显示这些错误。如果 **-i** 标志没有同 **errpt** 命令一起使用的话，则 **errpt** 处理的错误日志文件就是在错误日志配置数据库中指定的那个。（要想查看在错误日志数据库中的信息，请使用 **errdemon** 命令。）

缺省的总结报表对每个错误包含一行数据。您可以使用标志来生成不同格式的报表。

> **注:** **errpt** 命令不会执行对错误日志的分析，要想分析它请使用 **diag** 命令。然而，当错误日志分析被执行时，诊断程序会将诊断信息加回到错误日志中去。这种信息会在对应的错误日志记录的详细数据后面出现。

您可以使用设备应用程序在基于 Web 的系统管理器中来更改设备的特征。（wsm）您还可以使用系统管理界面工具（SMIT）**smit errpt** 快速路径来运行这个命令。

标志

-a 以详细的格式显示错误日志文件中的错误信息。如果同 **-t** 标志结合起来使用的话，则模板文件中所有的信息都会显示出来。

-A 显示由 **-a** 标志产生的详细报表的缩减版本。**-A** 标志和 **-a-g** 或者 **-t** 标志在一起是非法的。报表中的项有标签、日期时间、类型、资源名称、描述和详细数据。该标志的示例的输出的格式如下:

```
LABEL:           STOK_RCVRY_EXIT
Date/Time:       Tue Dec 14 15:25:33
Type:            TEMP
Resource Name:   tok0
Description
Description
PROBLEM RESOLVED
Detail Data
FILE NAME
line: 273 file: stok_wdt.c
SENSE DATA
0000 0000 0000 0000 0000 0000
DEVICE ADDRESS
0004 AC62 25F1
```

-c 对每个错误记录并行地，也就是说在它们被记录下来的时候，进行格式化和显示。日志文件中存在的记录是按照它们被记录的顺序显示的。

-d *ErrorClassList* 将错误报告限定到几种特定的错误记录类型，由有效的 *ErrorClassList* 变量指定: **H** （硬件），**S** （软件），**0** （**errlogger** 命令消息）和 **U** （未确定）。在 *ErrorClassList* 变量中的错误记录可以用 **,**（逗号）隔开，或者用 ″ ″（双引号）括起来并用 **,**（逗号）或者空格符隔开。

-D 整理重复的错误。用 **-a** 标志得到的详细错误报表会报告编号和第一次重复及最后一次重复的时间。请参阅《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『错误日志概述』。

注: **-D** 标志与 **-c**、**-g**、**-l**、**-t** 和 **-P** 标志在一起无效。

-e *EndDate* 指定了所有的之前贴出的记录，包含 *EndDate* 变量，这里 *EndDate* 变量具有 *mmddhhmmyy* 的形式（月、日、小时、分钟和年）。

-g 显示未格式化的错误日志记录的 ASCII 表示。
该标志的输出格式如下:

el_sequence
错误记录戳记编号

el_label
错误标签

el_timestamp
错误日志记录的时间戳记

el_crcid
唯一的循环冗余检验（CRC）的错误标识符

el_machineid
机器标识变量

el_nodeid
节点标识变量

el_class
错误类

el_type
错误类型

el_resource
资源名称

el_rclass
资源类

el_rtype
资源类型

el_vpd_ibm
IBM 重要产品数据（VPD）

el_vpd_user
用户 VPD

el_in 设备的位置代码

el_connwhere
硬件连接标识（在特定设备中的位置，如插槽编号）

et_label
错误标签

et_class
错误类

et_type
错误类型

et_desc
错误描述

et_probcauses
可能的原因

et_usercauses
用户的原因

et_useraction
用户操作

et_instcauses
安装原因

et_instaction
安装操作

et_failcauses
故障原因

et_failaction
故障操作

et_detail_length
详细数据字段长度

et_detail_descid
详细数据标识符

et_detail_encode
详细数据输入格式的描述

et_logflg
日志标记

et_alertflg
可警告的错误标志

et_reportflg
错误报表标志

el_detail_length
详细数据输入长度

el_detail_data
详细数据输入

-F *FlagList*
根据该模板的 Alert、Log 或者 Report 字段的值来选择错误记录模板。*FlagList* 变量可以用 **,**（逗号）隔开，或者在 ″ ″（双引号）括起来并用 **,**（逗号）或者空格符隔开。**-F** 标志只能与 **-t** 标志一起使用。

FlagList 变量的有效值包括:

alert=0 Alert 字段设置成 False 时选择错误记录模板。

alert=1 Alert 字段设置成 true 时选择错误记录模板。

log=0 在 Log 字段设置成 False 时选择错误记录模板。

log=1 在 Log 字段设置成 True 时选择错误记录模板。

report=0
在 Report 字段设置成 False 选择错误记录模板。

report=1
Report 字段设置成 True 时选择错误记录模板。

-i *File*
使用 *File* 变量指定的错误日志文件。如果该标志没有被指定的话，就会使用从错误日志配置数据库中得来的值。

-l *File*
使用由 *File* 指定的诊断日志文件。如果该标志没有被指定的话，会使用缺省的路径名 **/var/adm/ras/diag_log**。

-j *ErrorID*[**,***ErrorID*]
只包括由 *ErrorID*（错误标识符）变量指定的错误日志记录。*ErrorID* 变量可以用 **,**（逗号）隔开，或者在 ″ ″（双引号）中括起来，然后用 **,**（逗号）或者空格符隔开。当它与 **-t** 标志结合在一起时，记录是在错误模板库中被处理的。（否则，记录就是在错误日志库中被处理。）

-J *ErrorLabel* 包括由 *ErrorLabel* 变量指定的错误日志记录。*ErrorLabel* 变量的值可以用 ,（逗号）隔开，或者括在 ″″（双引号）内然后用 ,（逗号）或者空格隔开。当它与 **-t** 标志结合在一起时，记录就是在错误模板库中被处理。（否则，记录就是在错误日志库中被处理。）

-k *ErrorID*[**,***ErrorID*] 将由 *ErrorID* 变量指定的错误日志记录排除在外。*ErrorID* 变量可以用 **,**（逗号）隔开，或者在 ″ ″（双引号）中括起来，然后用 **,**（逗号）或者空格符隔开。当它与 **-t** 标志结合在一起时，记录是在错误模板库中被处理的。（否则，记录就是在错误日志库中被处理。）

-K *ErrorLabel* 将 *ErrorLabel* 变量指定的错误日志记录排除在外。*ErrorLabel* 变量的值可以用 ,（逗号）隔开，或者括在 ″″（双引号）内然后用 ,（逗号）或者空格隔开。当它与 **-t** 标志结合在一起时，记录就是在错误模板库中被处理。（否则记录就是在错误日志库中被处理的。）

-l *SequenceNumber* 选择由 *SequenceNumber* 变量指定的唯一的错误日志记录。这个标志被错误通知对象类的方法所使用。*SequenceNumber* 变量可以用 **,**（逗号）隔开，或者在 ″ ″（双引号）括起来并用 **,**（逗号）或者空格符隔开。

-m *Machine* 包括指定的 *Machine* 变量的错误日志记录。**uname -m** 命令会返回 *Machine* 变量的值。

-n *Node* 包括了指定的 *Node* 变量的错误日志记录。**uname -n** 命令会返回 *Node* 变量的值。

-N *ResourceNameList* 生成由 *ResourceNameList* 变量指定的资源名称的报表。*ResourceNameList* 变量是已经被探测到有错误的资源的名称的列表。对于软件错误 *ResourceNameList* 变量会列出已经被检测到有错误的资源的名称。对于硬件错误，它会列出设备或者系统组件的名称。这并不表示组件是错误的或者需要更换。相反，它被用来确定要用合适的诊断模块来分析错误。

ResourceNameList 变量的名称可以用 **,**（逗号）隔开，或者在 ″ ″（双引号）中括起来并用 **,**（逗号）或者空格符隔开。

-P 只显示那些是前面的错误的重复的那些错误。**-P** 标志只用于由错误日志设备驱动程序生成的重复错误。这些错误是在由 **errlg_duptime** 错误记录属性指定的近似的时间间隔内发生的重复错误，该属性由 **errdemon** 守护程序 **-t** 标志所控制。**-P** 标志与 **-D** 标志一起是非法的。

-r *ResourceTypeList* 生成一个由 *ResourceTypeList* 变量指定的资源类型的报表。对于硬件错误，*ResourceTypeList* 变量就是设备类型。对于软件错误，它是 **LPP** 值。在 *ResourceTypeList* 变量中的项可以用 **,**（逗号）隔开，或者括在 ″ ″（双引号）内并用 **,**（逗号）或者空格符隔开。

-s *StartDate* 指定所有张贴的在*StartDate* 变量后面的记录，这里 *StartDate* 变量的形式为 *mmddhhmmyy*（月，日，小时，分和年）。

-S *ResourceClassList* 生成由 *ResourceClassList* 变量指定的资源类的报表。对于硬件错误，*ResourceClassList* 变量就是一个设备类。资源类必须用 **,**（逗号）两两隔开，或者括在 ″ ″（双引号）内并由 **,**（逗号）或空格符隔开。

-t 处理错误记录模板库而不是错误日志。**-t** 标志可以用来以报表的形式查看错误记录模板。

-t *ErrorTypeList* 将错误报表限定成由有效的 *ErrorTypeList* 变量指定的错误类型：**INFO**、**PEND**、**PERF**、**PERM**、**TEMP** 和 **UNKN**。错误类型可以用 **,**（逗号）隔开，或者括在 ″ ″（双引号）内，并用 **,**（逗号）或者空格符隔开。

-y *File* 使用由 *File* 变量指定的错误记录模板文件。当它与 **-t** 标志结合在一起时，记录就是在指定的错误模板库中被处理。（否则，记录就是在错误日志库中用指定的错误模板库处理。）

-z *File* 使用 *File* 变量指定的错误记录消息目录。当它与 **-t** 标志结合在一起时，记录就是在错误模板库中被处理的。（否则，记录就是在错误日志库中被处理。）

示例

1. 要想显示一个完整的摘要报告，请输入:

   ```
   errpt
   ```

2. 要想显示一个完整的详细报告，请输入:

   ```
   errpt  -a
   ```

3. 要想显示一个错误标识符 E19E094F 记录的所有的错误的报表，请输入:

   ```
   errpt  -a  -j E19E094F
   ```

4. 要想显示一个在过去 24 小时内记录下来的所有错误的详细报表，请输入:

```
errpt  -a  -s mmddhhmmyy
```

这里，mmddhhmmyy 字符串等于当前的月、日、小时、分和年再减去 24 小时。

5. 要想列出为其日志为任何错误日志记录关闭的错误记录模板，请输入:

```
errpt  -t  -F log=0
```

6. 要想从 /var/adm/ras/errlog.alternate 备用错误日志文件中查看所有的记录，请输入:

```
errpt  -i /var/adm/ras/errlog.alternate
```

7. 要想从 /var/adm/ras/errlog.alternate 备用错误日志文件中查看所有的硬件记录，请输入:

```
errpt  -i /var/adm/ras/errlog.alternate -d H
```

8. 要想显示对于错误标签 ERRLOG_ON 的所有记录下来的错误的详细报表，请输入:

```
errpt  -a  -J ERRLOG_ON
```

9. 要想显示所有错误的详细报表并给重复错误进行分组，请输入:

```
errpt -aD
```

10. 要显示 8 月份期间对于错误标签 DISK_ERR1 和 DISK_ERR2 记录下来的所有错误的详细报表，请输入:

```
errpt -a -J DISK_ERR1,DISK_ERR2 -s 0801000004 -e 0831235904"
```

文件

/etc/objrepos/SWservAt 包含了软件服务辅助属性对象类；即错误日志配置数据库。

相关信息

diag 命令、**errclear** 命令、**errinstall** 命令、**errupdate** 命令，**uname** 命令。

errsave 内核服务。

errlog 子例程。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『错误日志概述』。

《*AIX 5L V5.3 通用编程概念*》 中的『详细错误报告示例』、『错误报告概要示例』。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

---

errstop 命令

用途

终止错误记录守护程序。

语法

errstop

描述

注: 运行 **errstop** 命令会禁用诊断和恢复功能。通常 **errdemon** 命令在系统初始化过程中自动启动，在系统关闭过程中停止。在正常的操作过程中错误记录永远都不应被停止。**errstop** 命令只有在特殊的情况下才应被使用，只有在绝对需要和能够清楚地理解其后果的情况下才能使用。

errstop 命令会停止由 **errdemon** 命令初始化的错误记录守护程序。

安全性

访问控制: 只有 root 用户才能运行该命令。

示例

要想终止 **errdemon** 守护程序，请输入:

```
/usr/lib/errstop
```

相关信息

errclear 命令、**errdead** 命令、**errinstall** 命令、**errlogger** 命令、**errmsg** 命令、**errpt** 命令、**errupdate** 命令。

errdemon 守护程序。

errsave 内核服务。

errlog 子例程。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『错误日志概述』

---

errupdate 命令

用途

更新错误记录模板库。

语法

errupdate [**-c**] [**-f**] [**-h**] [**-n**] [**-p**] [**-q**] [**-y***FileName*] [*File*]

描述

errupdate 命令在错误记录模板库中增加或者删除记录，或者修改日志、报表，或者对现存记录的特征发出警报。 **errupdate** 命令从指定的 *File* 参数中读入。如果 *File* 参数没有指定的话，则 **errupdate** 命令从标准输入读入并写到标准输出。

每个要增加、删除或者修改的记录必须前面有一个运算符。有效的运算符为:

```
+       增加一个记录（加号）。
-       删除一个记录（删除运算符）。
=       修改日志、报表或者提醒记录的特征。
```

在输入文件中的记录必须由一个空行隔开。

在输入文件中的注释可以放置在模板之间，在第一列要用 *（星号）来表示。

如果 X/Open Portability Guide Issue 4 消息在错误模板中用到的话，则必须指定一个消息编目。这可以用这样一行来完成:

```
<*!catalog-name>
```

示例

```
*!mycat.cat
```

指定的目录应用于在后面的模板中找到的 XPG4 消息，直到遇到另一个 “*!” 目录说明符。并且， “*!” 说明符可以基于单个模板用 “catname” 关键字覆盖。

除非有到该目录的完整路径被指定，否则要遵守检索信息目录的一般规则。例如，在上面的示例中，mycat.cat 假定是在 **/usr/lib/nls/msg/%L** 中。

要增加的记录必须以特定的格式定义。错误记录模板的一般形式是:

```
Error Record Template

        + LABEL:
                        Comment=
                        Class=
                        Log=
                        Report=
                        Alert=
                        Err_Type=
                        Err_Desc=
                        Prob_Causes=
                        User_Causes=
                        User_Actions=
                        Inst_Causes=
                        Inst_Actions=
                        Fail_Causes=
                        Fail_Actions=
                        Detail_Data= <data_len>, <data_id>,
                        <data_encoding>
```

另外XPG4 消息的目录名可以用下面的来指定:

```
catname = <catalog>
```

任何包含 XPG4 消息、catname 关键字和多于 8 个的详细数据项的模板将会被当作一个 XPG4 模板。XPG4 模板是不可提醒的，并且使用了一个稍微不同的错误标识计算方法。

错误记录模板字段的描述如下:

Alert
: 表示错误日志记录可以被与 SNA 一般警告结构符合的产品处理。Alert 字段可以被设置成真或假。如果该字段从模板中省略，它的值会缺省为 False。如果 Alert 字段设置为 True，则 **errupdate** 命令不会添加模板，除非 Err_Desc、Inst_Actions、Fail_Cause，Fail_Actions 和 Detail_Data data_id 字段的内容是 SNA Generic Alert Architecture（在 GA27-3136 出版物中）认可的值。如果这些值中的任意一个不被 SNA 一般警告结构识别或者该模板是一个 XPG4 模板，并且 Alert 字段被设置成 True 的话，则 **-p** 标志必须被指定，才能增加或者更新模板。

Class

描述错误是发生在硬件还是在软件上，是一种操作程序消息还是没有确定的消息。以下类描述符中的一个必须被指定:

H 表示错误是硬件故障。

O 表示错误是操作程序消息。

S 表示错误是软件故障。

U 表示错误没有确定。

Comment

将注释指定成包括在 **#define** 语句中，该语句是为错误标识信息集创建的。注释不准超过 40 个字符，必须用双引号括起来。超过 40 个字符的注释会自动被截掉。**errupdate** 命令将注释包括在 C 语言的注释定界符内，/* （斜杠，星号）和 */ （星号，斜杠）。

Detail_Data

描述详细数据，如检测模块名称、检测数据或返回代码，它们是在错误发生时随错误一起记录下来的。如果没有详细数据同错误一起被记录下的话，则这个字段可以空着，或者通过将 **data_len** 的值指定为 0 来从详细数据 ID 消息集中显示一条消息。对于每个 **Detail_Data** 字段需要下面三个值，它们之间必须用逗号隔开:

data_len

与 **data_id** 值相关的数据的字节数目。**data_len** 值被解释成一个十进制值。要想指定与环境有关的大小，请使用 “W”。如果错误是从 64 位环境下记录下来的话则 “W” 将被当作是 8 个字节，否则当作是 4 个字节。

注: 在详细数据长度计算的过程中，每个 “W” 都被当作是 8 个字节，并且是不区分大小写的。

data_id

从详细数据 ID 消息集合 “D” 中识别一个文本消息，在该详细数据的前面的错误报表中打印出来。这个值被解释成一个无符号的十六进制数，长度最多可到 4 位。

data_encoding

描述了详细数据是如何在错误报表中打印出来的。有效的值为:

ALPHA

该详细数据是一个可打印的 ASCII 字符串。

DEC 该详细数据为一个整数值的二进制表示，并且要打印的是与其相当的十进制值。

LDEC 该详细数据为一个 64 位值的二进制表示，并且要打印的是与其相当的十进制数。/defn>

HEX 该详细数据要以十六进制数的形式打印。

每个模板最多可以指定 16 个 Detail_Data 记录。与错误一起记录的数据量不能超过 **/usr/include/sys/err_rec.h** 文件中的 ERR_REC_MAX。不能包含在错误日志记录中的错误数据应该保存在其他地方。在错误日志记录中的详细数据应该包含那些可以将错误数据和错误日志记录关连起来的信息。

Err_Desc

描述了已经发生的错误。错误描述消息标识符必须在该字段中被指定。这个值会从错误描述消息集合 “E” 中识别出一个文本消息，这个消息在该错误发生时会显示出来。该值被解释成一个无符号的十六进制数，长度最多 4 位。该字段还可以指定一个 T XPG4 样式的消息。这在以后再讨论。

Err_Type
描述已经发生的错误的严重性。以下值中的一个必须被指定:

PERF 设备或组件的性能已经降低到可接受的水平以下的条件（性能）。

PERM 不能恢复的条件（永久性）。

PEND 表示设备或组件的可用性的损失已经很危急的条件（迫切的）。/defn>

TEMP 从许多次不成功的尝试之后恢复过来的条件（临时的）。/defn>

UNKN 不可能确定错误的严重性时的条件（未知的）。

INFO 信息错误日志记录的条件。

Fail_Actions
描述了推荐的用于纠正一个由于故障原因导致的错误的操作。可以指定一个最多有四个推荐操作消息标识符的列表，用逗号隔开。该值从推荐操作消息集合 “R” 中识别一个文本消息，在错误发生时显示出来。该值被解释成一个无符号的十六进制数，长度最大 4 位。这个字段必须为空，如果 Fail_Causes 字段为空的话。

推荐操作列出的顺序要根据操作纠正错误的可能性以及代价来确定。永远都要将那些对系统有很小或者没有代价（很小，或者没有影响）的操作列在第一位。接下来列出那些纠正错误的可能性相等或者近似相等的操作，将最小代价的操作先列出来。按照可能性递减的顺序列出余下的操作。该字段还可以指定一个 XPG4 样式的消息。这会在以后再讨论。

Fail_Causes
描述了已经发生的错误的故障原因。故障原因定义成一个由资源失败而导致的条件。这个字段可以最多列出四个故障原因消息标识符，用逗号隔开。这个值会从故障原因消息集合 “F” 中指定一条文本消息，这个消息会在错误发生时被显示出来。该值被解释成一个无符号的十六进制数，长度最多 4 位。按照可能性递减的顺序列出故障原因。这个字段可以空着，如果它没有应用到已经发生的错误上的话。如果该字段为空，则 User_Causes 或者 Inst_Causes 字段不准为空。该字段还可以指定一个 T XPG4 样式的消息。这在以后再讨论。

Inst_Actions
描述了纠正一个安装导致的错误的推荐操作。这个字段可以列出最多 4 个推荐操作消息标识符，用逗号隔开。该值从推荐操作消息集合 “R” 中指定一个文本消息，在错误发生时显示出来。该值被解释成一个无符号的十六进制数，长度最大 4 位。这个字段必须为空，如果 Inst_Causes 字段为空的话。推荐操作被列出的顺序要根据该操作的代价以及该操作纠正错误的可能性来确定。那些代价很少或者没有代价，或者对系统影响很小或者没有影响的操作总是要首先列出来。其次要列出纠正错误的可能性相等或近似相等的操作，其中最小代价的操作在先。剩下的操作应该按照可能性递减的顺序列出来。该字段还可以指定一个 XPG4 样式的消息。这在以后再讨论。

Inst_Causes
描述了已经发生的错误的安装原因。安装原因被定义成由资源的初始化安装或者设置导致的条件。最多可以指定四个安装原因消息标识符，用逗号隔开。可以指定一个最多 4 个安装原因消息标识符列表，以逗号隔开。该值从安装原因消息集合 “I” 中指定一条文本消息，在错误发生的时候显示出来。该值被解释成一个无符号的十六进制数，长度最大 4 位。安装原因应按概率递减的顺序列出。该字段可以空着，如果它对已经发生的这个错误不适用的话。如果该字段空着，User_Causes 或 Fail_Causes 字段必须非空。该字段还可以指定一个 XPG4 样式的消息。这在以后再讨论。

LABEL
指定一个最多 19 个字符的唯一的标签，它必须对每个错误记录模板都提供。一个包含了 “ #define #ERRID_label Error_ID ”（这里 **Error_ID** 值为分配给错误记录模板的唯一 ID 值）的字符串会写到标准输出中去，如果 **-h** 标志在该命令行中被指定的话。

Log
指定当错误发生时是否应该为该错误创建一个错误日志记录。日志字段可以设置成 True 或 False。如果该字段从模板中省略的话，它的值会缺省成 True。当该字段被设置成 False 的话， Report 和 Alert 字段会被忽略。

Prob_Causes
描述了已经发生的错误的 1 个或多个可能的原因。可以指定一个最多 4 个可能原因消息标识符列表，以逗号隔开。该值会从可能原因消息集合 “P” 中指定一条文本消息，在错误发生时显示出来。该值被解释成一个无符号的十六进制数，长度最多 4 位。可能原因应该按照可能性递减的顺序列出。最少需要有一个可能的原因。该字段还可以指定一个 XPG4 样式的消息。这在以后再讨论。

Report	指定记录下来的错误的发生在错误报表打印时是否应该报告。 Report 字段可以设置成 True 或 False。如果该字段从模板省略，其值将缺省为 True。
User_Actions	描述了纠正用户导致的错误的推荐操作。可以指定一个最多有四个推荐操作消息标识符的列表，用逗号隔开。该值从推荐操作消息集合 “R” 中指定一个文本消息，在错误发生时显示出来。该值被解释成一个无符号的十六进制数，长度最多 4 位。该字段必须为空，如果 User_Causes 字段空着的话。推荐操作列出的顺序要由该错误的代价和该操作纠正错误的可能性来确定。那些成本很少或者没有成本、对系统影响很小或者没有影响的操作总是应该首先列出。其次要列出纠正错误的可能性相等或近似相等的操作，其中最小代价的操作在先。剩下的操作应该按照可能性递减的顺序列出来。该字段还可以指定一个 XPG4 样式的消息。这在以后会讨论。
User_Causes	描述了已经发生的错误的用户原因。用户原因定义成无需与服务机构联系就能纠正的情况。可以指定一个最多有 4 个用户原因的消息标识符的列表，之间用逗号隔开。该值从用户原因消息集合 “U” 中指定一条文本消息，在错误发生的时候显示出来。该值被解释成一个无符号的十六进制数，长度最大 4 位。用户原因应该按照可能性递减的顺序列出。该字段可以空着，如果它对已经发生的这个错误不适用的话。如果该字段空着，则 Inst_Causes 或 Fail_Causes 字段必须是非空的。该字段还可以指定一个 XPG4 样式的消息。这在以后再讨论。

catname 用来为当前的模板指定一个用来检索 XPG4 消息的消息目录。它会覆盖由前面的 “″*!” 目录标识符指定的目录。任何包含了 XPG4 消息的模板必须有一个用 catname 或者 “*!” 指定的目录. 目录名称必须用引号括起来。除非指定了该目录的完整路径，否则要遵守检索消息目录的一般规则。

例如，如果

```
catname = "mycat.cat"
```

被指定的话，mycat.cat 就假定是在 **/usr/lib/nls/msg/%L** 中。

错误描述、可能的原因、用户原因、安装原因、故障原因、推荐操作和详细数据 ID 消息必须要么是在错误日志消息目录中保持的一个错误消息标识符，要么是一个 XPG4 消息。

错误消息标识符包含最多四位的十六进制数，前面没有 ″Ox″。例如，1234 或者 ABCD。**errmsg -w** 命令可以用来将这些消息同它们的标识符一起打印。**errmsg** 命令可以用来增加新的消息。

XPG4 消息是通过这种形式来指定的:

```
{<set>, <number>, <"default text">}
```

集合、编号和缺省文本都是需要的。不支持符号消息引用。另外，包含了 XPG4 消息的模板是不可警告的。

必须为 XPG4 消息指定一个消息目录。这通过 “*!” 目录说明符或 catname 关键字完成。

错误记录不支持一般错误消息的全部功能。在错误日志模板中使用的字符串必须符合某些限制。

- 不支持变量替换。例如，这些字符串不可以用作格式说明符来打印值。字符串只能包含格式字符 ″\t″ 和 ″\n″。
- 缺省的文本字符串不可以超过 1kb，即 1024 字节。
- 必须注意的是，错误描述是在非详细报表中的 40 字符区域中打印的。对于这些报表没有完成字符串格式化，只有头 40 个字符才会被打印出来。
- 字符串后面不应该包含一个新行。这是由 errpt 提供的。

对每个增加进来的条目，errupdate 命令都指定了一个唯一的错误 ID，该 ID 被写入到由 *File*.h 指定的头文件中（这里 *File* 参数是 errupdate 命令的输入文件）。如果 errupdate 命令从标准输入中读入的话，则 #define 语句写到标准输出中。由 Class , Err_Desc , Err_Type , Fail_Actions , Fail_Causes , Inst_Actions ,

Inst_Causes，Prob_Causes，User_Actions，User_Causes 字段提供的值和 Detail_Data。data_id 的值用来计算该错误的唯一错误 ID 号。对于 XPG4 模板，该标签也包括在计算中。

Log、 Report 和 Alert 字段的内容不包括在唯一错误 ID 的计算之中；因此特定错误的日志、报表和警告特征可以在任何时候在存储在错误记录模板库中的错误记录定义中进行修改，用 **errupdate** 命令来实现。还要注意详细数据字段的 data_len 和 data_encode 部分没有使用到。

errupdate 命令还在名为 *File*.**undo** 的当前目录中创建了一个撤销文件。如果 **errupdate** 命令是从标准输入中读入的话 **undo** 文件就被写入到 **errids.undo** 文件中。**undo** 文件包含了 **errupdate** 命令的输入，用来撤销 **errupdate** 命令已经作出的改动。

errpt -t 命令可以用来查看错误记录模板库中的内容。该模板由于会在实际错误报表中出现，因此会被处理和打印。

> **注**: 如果您更改了错误模板，请注意这些模板可能会被后续的更新所更改。您应该保存一个所有进行的改动的记录，当您的系统被更新的时候重新进行这些改动。这种情况只有在进行了一些主要的系统更新如移动到一个新的操作系统级别等之后才有必要。还有，如果您重新安装，这样一个记录会允许您更改您的模板。保存这样的记录的最简单的方式就是让您的模板修改总是在同一个 errupdate 源文件进行。

标志

-c	检测输入文件有无语法错误。
-f	强制所有的模板进行更新，包括那些错误 ID 与输入模板中的相同的模板。
-h	为每个分配给错误模板的错误 ID 创建一个 #define 语句。如果在命令行中提供了一个文件名的话，则头文件就是那个提供的文件名后面带上 **.h**。否则， #define 语句就被写入到标准输出。
n	禁止将错误记录模板增加到错误记录模板库中。
-p	用 Alert 字段设置成 True 来对模板进行添加或者更新，该模板包含了错误描述、可能原因、用户原因、用户操作、安装原因、安装操作、故障原因、故障操作或者不被 SNA 一般警告结构（在出版物 GA27－3136 中）所承认的详细数据 ID 值。**errupdate** 命令将不会让您对模板增加这些特征，除非您指定了这个标志。
-q	禁止撤销文件的创建。
-y *FileName*	使用 *FileName* 参数指定的错误记录模板文件。

安全性

访问控制: 无，但是对于要更改的模板文件（缺省情况下为 /var/adm/ras/errtmplt），您必须具有写权限。

实例

1. 要想增加一个记录，请在输入文件中按下面的方式定义该记录:

```
+ CDROM_ERR22:
        Comment=        "Temporary CDROM read error"
        Class=  H
        Log=            True
        Report= True
        Report= True
        Alert=          False
         Err_Type=       TEMP
        Err_Desc=       E801
        Prob_Causes=    5004
        Fail_Causes=    E800, 6312
        Fail_Actions=   1601, 0000
        Detail_Data=    120, 11, HEX
        Detail_Data=    4, 8058, DEC
        Detail_Data=    4, 8059, DEC
```

要输入数据，

```
errupdate <input file>
```

2. 要想修改 99999999 记录的日志、报表和警告特征，请指定在唯一错误 ID 前面的修改运算符 =（等号）及该记录要被修改的新特征:

```
errupdate
 =99999999:
 Report = False
 Log = True
```

3. 要想从错误记录模板库中删除 99999999 记录的话，请指定在要删除的记录唯一错误 ID 前面的删除运算符 -（负号）:

```
errupdate
 -99999999:
```

4. 要用 “*!” 覆盖为该输入流指定的 XPG4 消息编目，请使用 “catname” 关键字。

```
*!mycat.cat
```

* mycat.cat 从现在起就用在所有的 XPG4 消息上了。

*除了这个:

```
+ CDROM_ERR23:
         Comment=        "Temporary CDROM read error"
         catname= "othercat.cat"
         Class=  H
         Log=            True
         Report= True
         Report= True
         Alert=          False
         Err_Type=       TEMP
         Err_Desc=       {1, 1, "CD ROM is broken"}
         Prob_Causes=    {2, 1, "cause 1"},\
                        {2, 2, "Cause 2"}
         Fail_Causes=    E800, 6312
         Fail_Actions=   1601, 0000
         Detail_Data=    120, 11, HEX
         Detail_Data=    4, 8058, DEC
         Detail_Data=    4, 8059, DEC
```

目录 othercat.cat 将会被只用作 CDROM_ERR23 模板。

注: 模板可以包含 XPG4 消息和传统的错误 ID 或代码点。

文件

/usr/include/sys/errids.h	包含了包含错误 ID 的头文件。
/usr/include/sys/err_rec.h	包含了含有记录错误的结构的头文件。

相关信息

errclear 命令、**errdead** 命令、**errinstall** 命令、**errlogger** 命令、**errmsg** 命令、**errpt** 命令和 **errstop** 命令。

errdemon 守护程序。

errsave 内核服务。

errlog 子例程。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『错误日志概述』。

ethchan_config 命令

用途

将适配器添加到 EtherChannel 或从 EtherChannel 删除适配器。

语法

ethchan_config { **-a** [**-b**] | **-d** } *EtherChannel Adapter*

ethchan_config -c *EtherChannel Attribute NewValue*

ethchan_config -f *EtherChannel*

描述

此命令向 EtherChannel 添加适配器，或从 EtherChannel 除去适配器。此命令还可用于修改 *EtherChannel* 属性。即使 EtherChannel 的接口当前已配置（也就是无需说，无需拆离 EtherChannel 的接口来添加或除去适配器或修改大多数 EtherChannel 属性），也可以执行这些添加、删除或修改操作。

标志

-a	将指定的 *Adapter* 添加到指定的 *EtherChannel*。如果必须将此适配器添加为备份适配器，则必须指定 **-b** 标志。
-b	指定此 *Adapter* 将添加为备份适配器。此标志仅在与 **-a** 标志一起使用时有效。
-c	将指定的 *EtherChannel* 属性的指定 *Attribute* 更改为指定的 *NewValue*。
-d	从指定的 *EtherChannel* 删除指定 *Adapter*。**-b** 标志无须和 **-d** 标志一起使用。
-f	强制指定 *EtherChannel* 的故障转移。请注意，只有在空闲通道中的适配器开启时才会真正发生故障转移；如果空闲通道中的适配器关闭，则 *EtherChannel* 将仍在活动通道上运行并且不会发生故障转移。

参数

Adapter	指定要添加或删除的适配器。
Attribute	指定指定 EtherChannel 的属性。
EtherChannel	指定 EtherChannel。
NewValue	指定指定 EtherChannel 的指定属性的新值。

退出状态

0	命令成功结束。
>0	发生错误。

示例

1. 要将适配器 `ent0` 添加为 EtherChannel（名称为 `ent7`）中的备份适配器，请输入:

   ```
   /usr/lib/methods/ethchan_config -a -b ent7 ent0
   ```

2. 要将名称为 `ent7` 的 EtherChannel 的 ping 属性地址更改为 `10.10.10.10`，请输入:

```
/usr/lib/methods/ethchan_config -c ent7 netaddr 10.10.10.10
```

3. 要将名称为 ent7 的 EtherChannel 的故障转移从当前活动通道强制到空闲通道，请输入:

```
/usr/lib/methods/ethchan_config -f ent7
```

限制

此命令无法修改 *use_jumbo_frame* 属性的使用。试图执行此操作将显示错误消息。

位置

/usr/lib/methods

---

ewallevent 命令、wallevent 命令

用途

B向所有登录用户广播一个事件或者一个重新装备的事件。

语法

ewallevent [**-c**] [**-h**]

wallevent [**-c**] [**-h**]

描述

ewallevent 脚本返回的永远是英文的消息。而 **wallevent** 脚本返回的消息的语言则取决于语言环境的设置。

当事件或者重新装备的事件发生时，这些脚本会对所有当前登录进主机的用户广播这个事件或重新装备事件的消息。当一个事件或者重新装备事件发生时，事件响应资源管理器会将事件或重新装备事件的信息捕获并贴到环境变量中，环境变量是由事件响应资源管理器生成的。这些脚本可以当作由事件响应资源运行的操作来使用。还可以用作创建其他用户定义的操作的模板。

当这些脚本为其是一种响应操作的事件或者重新装备事件发生时，在所有用户登录的控制台中消息按照这种格式显示:

```
Broadcast message from user@host (tty) at hh:mm:ss...

severity event_type occurred for Condition condition_name
on the resource resource_name of resource_class_name at hh:mm:ss mm/dd/yy
The resource was monitored on node_name and resided on {node_names}.
```

将返回有关 ERRM 环境变量的事件信息，还包含以下内容:

本地时间

事件或者重新装备事件被发现的时间。由 ERRM 提供的实际的环境变量是 ERRM_TIME。该值是本地化的，在被显示出来之前会被转化成可读的形式。/defn>

wallevent 脚本会捕获环境变量的值，使用 **wall** 命令将消息写到当前登录用户的控制台中。

标志

–c 指示 **wallevent** 广播 ERRM 事件的 **ERRM_VALUE**。如果指定了 **-c** 标志，**wallevent** 将广播此 SNMP 捕获消息。

–h 将该脚本的用法声明写到标准输出中。

参数

log_file

指定事件信息被记录的文件的名称。应该指定 *log_file* 参数的绝对路径。

log_file 被当作一个循环记录，具有 64KB 的固定的大小。当 *log_file* 满了时，新的记录会覆盖存在最早的记录。

如果 *log_file* 已经存在，则事件信息会附加到它后面。如果 *log_file* 不存在，则将创建它，这样事件信息就可以写入到它里面。

退出状态

0 脚本成功运行。

1 脚本运行时发生了错误。

限制

1. 这些脚本必须运行在 ERRM 运行的节点上。
2. **wall** 命令用来将消息写入到当前登录用户的控制台上。参考 **wall** 联机帮助页可以获取更多有关 **wall** 命令的信息。

标准输出

当 **-h** flag 被指定时，脚本的用法声明被写入到标准输出中。

示例

1. 假定 **wallevent** 脚本是在紧急通知响应中预先定义的操作，这个紧急通知响应是同 **/var space used** 条件有关的（在资源 **/var** 中）。为该条件定义的事件表达式的阈值达到，并且发生一个事件。紧急通知响应发生，**wallevent** 被运行 i。在所有登录的用户的控制台上会显示以下消息:

```
Broadcast message from joe@neverland.com (pts/6) at 18:42:03...

Critical event occurred for Condition /var space used
on the resource /var of filesys of IBM.FileSystem at 18:41:50 03/28/02
The resource was monitored on c174n05 and resided on {c174n05}.
```

2. 当重新装备事件因为 **/var space used** 条件而发生时（在资源 **/var** 上），所有登录用户的控制台上会显示以下消息:

```
Broadcast message from joe@neverland.com (pts/6) at 18:42:03...

Critical rearm event occurred for Condition /var space used
on the resource /var of filesys of IBM.FileSystem at 18:41:50 03/28/02
The resource was monitored on c174n05 and resided on {c174n05}.
```

位置

/usr/sbin/rsct/bin/ewallevent 包含了 **ewallevent** 脚本

/usr/sbin/rsct/bin/wallevent 包含了 **wallevent** 脚本

相关信息

命令: **wall**

ex 命令

用途

文本文件编辑器。

语法

ex[**-c** *Subcommand*] [**-l**] [**-r**] [**-s**] [**-t***Tag*] [**-V**] [**-w***Number*] [**-v**| **-**] [**+**[*Subcommand*]] [**-r**[*File*]] [*File...*]

描述

ex 命令会启动 ex 编辑器。ex 编辑器是编辑器系列的一部分，该系列包括了为初学者和临时使用准备的 ex 编辑器的简单版本 edit 编辑器和全屏幕的 vi 编辑器。直接调用 vi 编辑器会设置屏幕编辑的环境变量。ex 编辑器比简单的行编辑器更强，因为它是 vi 编辑器的子集并且可以访问 vi 编辑器的屏幕编辑功能。

File 参数要编辑的一个文件或者多个文件。如果您提供了多个文件名，则 ex 编辑器会按照指定的顺序编辑每个文件。

注:

1. 为了确定您的工作站如何更有效地执行，ex 编辑器使用工作站能力数据库 **terminfo** 以及从 **TERM** 环境变量中使用您正使用的工作站的类型。
2. **ex** 命令会影响当前的行，除非您指定别的方式。为了对文件的不同部分进行工作，您需要知道如何在一个文件中对行进行寻址。

标志

-c *Subcommand*
在编辑开始之前执行 ex 编辑器子命令。当输入一个空操作数时，如在 -c '' 中的那样，则编辑器会将当前行放置在文件的末尾。（通常，ex 编辑器会将当前行设置在文件的开始位置或者在某个指定的标记或模式处。）

-l
恰当地对 LISP 代码缩进，并将（）（开或闭括号），{ }（左右大括号），以及 [[ ]]（左右双方括号） 字符接受为文本，而不是将它们解释成 vi 子命令。这个标志在可见模式和打开模式下是活动的。

-r
设置 **readonly** 选项，防止您改动文件。

-s
禁止所有的交互式用户的反馈。如果您使用了该标志，文件输入／输出错误不会生成一个有帮助的错误信息。使用这个标志同使用 **-** 标志是一样的。

-t *Tag*
载入包含了由参数 *Tag* 指示的标记的文件，并将编辑器定位到该标记处。要想使用该标记，您必须首先利用 **ctags** 命令创建一个功能名称和它们的位置的数据库。

-w*Number*
将缺省的窗口大小设置成 *Number*。

-v
调用 vi 编辑器

> **注:** 当 **-v** 标志被选择时，会有一个放大的子命令集可用，包括了屏幕编辑和光标移动功能。请参阅 **vi** 命令。

-V
将编辑器在详细模式调用。

-
禁止所有的交互式用户的反馈。如果您使用了该标志，文件输入／输出错误不会生成一个有帮助的错误信息。使用该标志同 **-s** 标志是一样的。

+[*Subcommand*]
在指定的编辑器搜索或者子命令里面开始一个编辑。当没有输入参数时，**+** *subcommand* 会将当前行放置在文件的末尾。通常，ex 编辑器会将当前行设置在文件的起始位置，或者设置在特定的标记或模式处。

-r [*File*]
在编辑器或者系统崩溃后恢复文件。如果您没有指定 *File* 参数，则会显示所有保存了的文件的列表。

退出状态

以下出口值被返回:

0 成功结束。
>0 发生错误。

文件

/usr/lbin/exrecover	恢复子命令
/usr/lbin/expreserve	保留子命令
$HOME/.exrc	编辑器启动文件
./.exrc	编辑器启动文件
/var/tmp/Ex*nnnnn*	临时编辑器
/var/tmp/Rx*nnnnn*	命名临时的缓冲区
/var/preserve	保留目录

相关信息

ctags 命令、**ed** 命令、**edit** 命令、**vi** 命令。

---

execerror 命令

用途

将错误消息写到标准错误。

语法

execerror

描述

当载入实际程序没有成功时，**execerror** 命令由 **exec** 子例程执行。它会传递被执行的文件的名字和零个或更多的装载器错误消息的字符串。每个装载器错误消息字符串包含了一个错误编号，后面是出错的数据。

示例

execerror 命令的使用如下:

```
char *buffer[1024];
buffer[0] = "execerror" ;
buffer[1] = "name of program that failed to load";
loadquery(L_GETMESSAGES, &buffer[2], sizeof buffer -8);
execvp("/usr/sbin/execerror",buffer);
```

这个样本代码会导致在消息写到标准错误之后应用程序终止。

文件

/usr/sbin/execerror	包含了 **execerror** 命令。

相关信息

exec 子例程和 **loadquery** 子例程。

---

execrset 命令

用途

运行一个附加在 rset 的程序或者命令。

语法

execrset [**-P**] [**-F**] **-c** *CPUlist* [**-m** *MEMlist*] **-e** *Command* [*parameters*]

或

execrset [**-P**] [**-F**] [**-S**] *rsetname* [**-e**] *Command* [*parameters*]

描述

execrset 命令会执行一个带有 **rset** 的附件的命令。这会导致指定的命令仅限于在处理器上和 / 或 rset 中包含的存储器区域中运行。系统注册表中的 **rset** 名可以用来指定处理器和 / 或该命令允许使用的存储器区域。或者，包含了指定的处理器和存储器区域的 **rset** 可以附加到进程后。

标志

-F 强制 **execrset** 命令发生（运行）。此标志在发出该命令之前将除去 bindprocessor 绑定以及进程中所有线程的 **rset**。如果还指定了 **-P** 标志，则它将在发出命令之前分离有效的 **rset** 和所有线程的 **rset**。

-P 将 **rset** 作为一个分区 rset 连接。

-c *CPUlist* 在要附加在执行该程序或命令的进程上的 **rset** 中的 CPU 列表。可以是一个或者多个 CPU，也可以是 CPU 范围。

-m *MEMlist* 在 **rset** 中的存储器区域的列表。可以是一个或者多个存储器区域或范围。

-e *command* [*parameters*] 指定被任何参数跟随的命令运行。**-e** 标志必须是该命令使用的最后的标志。

-S 指示在单线程方式中必须调度进程以运行的提示。指定的 rset 中所包含的每个实际处理器的硬件线程中，只有一个线程将用于调度该作业。如果指定的 rset 中未包含物理处理器的所有硬件线程，则将忽略该处理器。指定的 rset 必须是唯一的，否则命令将失败。指定此标志将允许作业以单线程行为运行。

参数

rsetname 附加在执行程序或命令的进程后的系统注册表中的 **rset** 的名称

安全性

用户必须具有 root 用户权限，或者具有 **CAP_NUMA_ATTACH** 的能力。用户必须具有 root 用户权限来向命令的进程附加一个分区 rset(**-P** 标志）。

示例

1. 要想在 CPU 0 - 7 上运行 test1 程序，请输入:

```
execrset -c 0-7 -e test1
```

2. 当名为 **test/cpus0to15** 的 rset 带一个附件时想要运行 'test2 parm1 parm2' 程序的话，请输入:

```
execrset test/cpus0to15 test parm1 parm2
```

3. 要想在 CPU 0 上运行 **ls -l** 命令，请输入:

```
execrset -c 0 -e ls -l
```

文件

/usr/bin/execrset 包含了 **execrset** 命令。

相关信息

attachrset、**detachrset**、**lsrset**、**mkrset** 和 **rmrset** 命令。

---

expand 命令

用途

写入到标准输出中，将制表符转变成空格符。

语法

expand [**-t** *TabList*] [*File ...*]

expand [-*tabstop*]|[-*tab1*,*tab2*,...,*tabn*] [*File ...*]

描述

expand 命令将指定的文件或者标准输入写入到标准输出中，然后用一个或多个空格符替代制表符。任何退格符都被拷贝到输出中，并由于制表符停止计算导致列的位置计数缩减；列的位置计数不会缩减到小于零以下。

注: *File* 参数必须是一个文本文件。

标志

-t *TabList* 指定制表符停止位的位置。制表符停止位的缺省值是 8 个列位。

TabList 变量必须包含一个十进制正整数或者多个十进制正整数。多个整数必须按照递增的顺序，必须由逗号或者空格隔开，整数左右要有引号括起来。单个 *TabList* 变量设置制表符停止位为与列位置相等的数。多个 *TabList* 变量将制表符停止位设置为在与 *TabList* 变量中的整数对应的列位置。

如果 **expand** 命令处理在 *TabList* 变量中指定的最后一个之前的制表符停止位的话，该制表符停止位在输出中会被一个单空格符替代。

参数

tabstop 指定为单个参数。分开设置 *tabstop* 空格符，而不是缺省值 8。

tab1, tab2,..., tabn 在由 *tab1,tab2,...,tabn* 指定的列处设置 TAB 字符。

退出状态

此命令返回以下出口值:

0 成功结束。
>0 发生错误。

示例

1. 要想将制表符停止位调整一个 text.fil 中相等距离的量，请输入:

   ```
   expand -t 3 text.fil
   ```

   如果 text.fil 包含了:

   ```
   1  2        3456789
   ```

   则 **expand** 命令会显示:

   ```
   1  2        3456789
   ```

2. 要想将制表符停止位调整一个与 text.fil 中不同的量，请输入:

   ```
   expand -t 3,15,22  text.fil
   ```

   OR

   ```
   expand -t "3 15 22" text.fil
   ```

   如果 text.fil 包含了:

   ```
   1  2       3      456789
   ```

   则 **expand** 命令会显示:

   ```
   1  2       3      456789
   ```

文件

/usr/bin/expand 包含了 **expand** 命令。

相关信息

newform 命令，**tab** 命令，**unexpand** 命令，**untab** 命令。

《*操作系统与设备管理*》中的『文件』介绍了文件以及处理文件的方法。

《*操作系统与设备管理*》中的『输入和输出重定向』描述了操作系统如何处理输入和输出。

---

expfilt 命令

用途

向一个导出文件导出过滤规则。

语法

expfilt [**-p**] [**-q**] [**-r**] [**-v 4** | **6**] **-f** *directory* [**-l** *filt_id_list*]

描述

用 **expfilt** 命令来将过滤规则导出到导出文本文件中，该文件可以被 **impfilt** 命令使用。如果您想要在多个机器上面定义相似的规则的话，这个方法很有用。

注: 在一个机器上的过滤器的描述可能对另外一个机器没有意义或者会令人误解。这个字段将不会被导出。

可以使用 **genfilt** 命令、IPsec 系统管理界面程序（IP V4 或 IP V6）或“虚拟专用网”子菜单中的基于 Web 的系统管理器配置用于此命令的 IPsec 过滤规则。

标志

-f *directory*
为创建导出的文本文件指定目录。如果不存在该目录将会被创建。

-l *filt_id_list*
列出您要导出的过滤器规则的 ID。过滤器规则的 ID 可以用 ″**,**″ 或者 ″**-**″ 隔开。如果该标志未被使用的话，则所有在可用的 IP 版本的过滤器规则表中定义好的过滤器规则将被导出。

-p
允许预先定义规则。

-q
指定安静模式。禁止输出到 **stdout** 中。

-r
指定未加工模式。按原样导出过滤器规则，不在规则上更改方向。当过滤器规则按原样导入导出时使用这个标志；例如，存储存储一个配置或者将配置复制到另外机器中时。

利用 **-r** 标志，将会保存流量的方向。例如，如果在主机 10.0.0.1 上面有一个规则允许从 10.0.0.2 来的入站流量的话，带有 -r 标志的 expfilt 将会写入相同的过滤器规则。

忽略 **-r** 标志将导致在导出文件中的方向从入站切换到出站。

-v
您想导出的过滤器规则的 IP 版本。值为 **4** 将指定 IP V4，值为 **6** 将指定 IP V6。当该标志未被使用时，IP V4 和 IP V6 都会被导出。

相关信息

impfilt 命令。

---

explain 命令

用途

提供一种交互式的同义字。

语法

explain

描述

explain 命令为 **diction** 命令查找的英语短语提供一种交互式的同义字。在使用 **explain** 命令之前，请使用 **diction** 命令来获取一份表达得不好的短语列表。当您使用 **explain** 命令时，系统提示您输入一个短语，然后回应一个在语法上可以接受的替代短语。您可以继续输入短语，或者您也可以通过按下 Ctrl-D 按键顺序来退出。

explain 命令还从命令行中获取管道输入，如下例所示:

```
diction Filename | explain
```

没有其他命令行参数是有效的。

文件

/usr/lib/explain.d 包含了同义字。

相关信息

diction 命令。

---

explore 命令

用途

启动 WebExplorer® 万维网浏览器。

语法

explore [**-i***FileName*] [**-t***Number*] [**-q**] [[**-url**] *URL*]

描述

explore 命令打开 WebExplorer 主窗口并连接到统一资源定位符（URL）获取主文档。

标志

-i*FileName* 指定一个可更换的初始文件，其中 *FileName* 是该文件的完整路径名称，替代缺省的 **$HOME/.explore-preferences**。这就使您能够使用备用的一组用户首选项来启动 WebExplorer。

-t*Number* 指定用来装入图像的线程数目，这里 *Number* 图像装入线程的数目。每个线程会在主窗口的状态区显示出来。最大可以指定 8 个，缺省值是 4 个。

-q 指定安静模式。这将在您启动应用程序时禁止 WebExplorer 标题窗口，并在您退出时忽略确认窗口。

-url *URL* 在启动 WebExplorer 时指定要装入的特定文档，其中 *URL* 是要装入的文档的 URL。如果 WebExplorer 已定义了主文档，则此 URL 将覆盖它。您不必要在 URL前使用 **-url** 标志。如果您单独指定 URL，WebExplorer 将接受它。

退出状态

此命令返回以下出口值:

0 成功结束。
>0 发生错误。

安全性

访问控制: 任何用户

审计事件: N/A

示例

要想在启动浏览器时不让标题窗口出现而直接到 Dilbert Zone 的 URL 中，请输入:

```
explore -q  http://www.unitedmedia.com/comics/dilbert/
```

或者

```
explore -q  -url http://www.unitedmedia.com/comics/dilbert/
```

文件

/usr/lpp/explorer/bin/explore	包含 **explore** 命令。
$HOME/.explore-preferences	包含指定了用户的设置偏好如使用颜色的数目等的初始文件。
$HOME/.mailcap	包含了将 mime 类型映射到外部查看器的配置文件。
$HOME/.mimetypes	包含了将 mimetype 映射到外部查看器的用户定义的配置文件。它是在 **Configure Viewers** 对话框中设置的。该文件覆盖 **.mailcap** 设置。

---

exportfs 命令

用途

导出和不导出目录到 NFS 客户机。

语法

/usr/sbin/exportfs [**-a**] [**-v**] [**-u**] [**-i**] [**-f***File*] [**-F**] [**-o***Option* [,*Option* ...]] [*Directory*]

描述

exportfs 命令使得本地目录可以为网络文件系统（NFS）客户机安装。这个命令通常是在系统启动期间由 **/etc/rc.nfs** 文件调用的，并使用了 **/etc/exports** 文件中的信息来导出一个或者多个目录，这些目录必须用完整的路径名来指定。

/etc/xtab 文件列出了当前导出的目录。要想显示这个文件，请输入不带标志和参数的 **exportfs** 命令。要想更改文件或者更改它的一个目录中的特征的话，root 用户可以编辑 **/etc/exports** 文件并运行 **exportfs** 命令。这种更改可以在任何时间完成。永远都不要直接去编辑 **/etc/xtab** 文件。

注:

1. 您不能导出位于同一文件系统内的当前导出目录的父目录或子目录。
2. NFS V2 和 V3 允许既导出目录又导出文件。NFS V4 访问只能导出目录。

标志

-a	导出所有的列在导出文件中的目录。
-v	按照是导出还是不导出，打印每个目录的名称。
-u	不导出您指定的目录。当结合 **-a** 标志使用时，取消导出所有导出的目录。当同时结合 **-a** 和 **-f** 标志使用时，撤销导出指定导出文件中的所有目录。
-i	允许没有在导出文件中指定的目录的导出，或者忽略在导出文件中的选项。除非用到 **-f** 标志来指定一个可更替的文件，**exportfs** 命令通常将向 **/etc/exports** 文件查询来获得与要导出的目录的相关选项。
-f *File*	指定一个导出文件，而不是 **/etc/exports** 文件，该文件包含了一个您可以导出的目录的列表。这个文件应该遵守 **/etc/exports** 文件相同的格式。注: 这个可更替的文件将不会在系统和 NFS 启动的时候用来自动导出目录。**/etc/exports** 文件是支持在系统启动时指定导出目录的唯一的文件。
-F	指定应该执行强制的取消导出。请仅与 **-u** 标志结合使用该标志。在取消导出 V2/V3 导出时，该标志不起作用。V4 取消导出可能由于关联的状态而失败。该标志强制释放与 V4 导出相关联的任何状态。

-o*Options*

为要导出的目录指定可选的特征。您可以输入多个变量，中间用逗号隔开。对于采用了 *Client* 参数的选项，*Client* 可以指定主机名、点分式 IP 地址、网络名或子网标识符。子网标识符具有 *"@host/mask"* 格式，其中 *host* 是主机名或点分式 IP 地址，*mask* 指定了当检查访问时使用的位数。如果没有指定 *mask*，则使用完整的掩码。例如，标识符 `@client.group.company.com/16` 将匹配 `company.com` 子网上所有的客户机。`@client.group.company.com/24` 的标识符将仅匹配 `group.company.com` 子网上的客户机。从下面选项中选择:

ro 用只读许可方式导出目录。如果没有指定，则目录以读写许可权导出。

ro=*Client[:Client]*

将目录以只读许可权导出至指定的客户机。将目录以只读许可权导出至未在列表中指定的客户机。如果已经指定了一个读写列表的话，则无法指定一个只读列表。

rw 将目录以读写许可权导出至所有客户机。

rw=*Client [:Client]*

将目录以读写许可权导出至指定的客户机。将目录以只读方式导出至不在列表中的客户机。如果已经指定了一个只读列表，则无法指定一个读写列表。

anon =*UID*

如果请求来自于 root 用户的话，用 *UID* 值作为有效的用户 ID。

该选项的缺省值是 -2。在 NFS V2 和 NFS V3 中，将 *anon* 选项的值设置为 -1 将禁用匿名访问。这样，在缺省情况下，安全 NFS 作为匿名接受非安全的要求，并且要求更多安全性的用户可以禁止这个功能，通过将 *anon* 设置成值为 -1。

root=*Client[:Client]*

允许从指定的客户机的 root 访问。不在列表中的客户机不允许 root 访问。

access=*Client[:Client,...]*

给每个列出的客户机提供安装访问。客户机可以是主机名也可以是网络组名。列表中的每个客户机首先要在 **/etc/netgroup** 数据库进行检查，然后在 **/etc/hosts** 数据库中检查。缺省值允许任何机器安装给定的目录。

secure 在访问目录时要求客户机使用更安全的协议。

-o *Options* (continued)

sec=*flavor[:flavor...]*

该选项用于指定一系列可能用于访问导出目录下的文件的安全方法。大部分 exportfs 选项可以使用 **sec** 选项进行分群。**sec** 选项之后的选项被认为从属于其前面的 **sec** 选项。可能指定任何数量的 **sec** 节，但每个安全方法只能指定一次。在每个 **sec** 节内，可以指定一次 **ro**、**rw**、**root** 和 **access** 选项。只有 **public**、**anon** 和 **vers** 选项对于导出是全局性的。如果使用 **sec** 选项来指定任何安全方法，则必须使用它来指定所有安全方法。如有任何 **sec** 选项不存在，则允许所有认证风格。

允许的风格值为:

sys Unix 认证。这是缺省方法。

dh DES 认证。

无 如果安装请求使用导出中未指定的认证风格，则允许安装请求继续匿名凭证。

krb5 Kerberos。仅认证。

krb5i Kerberos。认证和整合。

krb5p Kerberos。认证、整合和隐私。

可能指定 **secure** 选项，但是没有结合 **sec** 选项。建议不要使用 **secure** 选项，并且可能消除它。使用 **sec=dh** 代替它。

vers=*version_number[:version_number...]*

指定允许哪些版本的 NFS 访问导出的目录。有效版本为 2、3 和 4。无法单独选择 V2 和 V3。指定 V2 或 V3 将使 NFS V2 和 NFS V3 都能够访问。可以单独选择 V4。缺省值是允许使用 NFS 协议 V2 和 V3 进行访问。

exname=*external-name*

用指定外部名导出目录。外部名必须以 **nfsroot** 名称开始。请参阅 **/etc/exports** 文件的描述以获取对 **nfsroot** 名称的描述。该选项仅用于导出以供 NFS V4 协议访问的目录。

deleg={yes | no}

对指定导出启用或禁用文件授权。此选项覆盖此导出的系统范围授权支持。系统范围支持是通过 **nfso** 实现的。

-o *Options* (continued)

refer=*rootpath@host***[+***host***][:***rootpath@host***[+***host***]]**

将在指定的路径中创建一个名称空间参照。该参照将指引客户机到指定的备用位置，客户机可以继续在其中进行操作。参照是一种特殊的对象。如果在指定的路径中存在非参照对象，将禁止导出并打印一条错误消息。如果在指定的路径中不存在任何内容，则将在该路径中创建一个包含通向该对象的路径名目录的参照对象。可以在文件系统中创建多个参照。不能为 **nfsroot** 指定参照。名称 `localhost` 不能被用作 *hostname*。仅允许 V4 的导出使用 **refer** 选项。如果导出规范允许 V2 或 V3 访问，将打印一条错误消息并禁止导出。不导出参照对象会影响从参照对象除去参照位置信息。不导出不会除去对象本身。如果要除去对象，请使用 **rm**。管理员必须确保在参照服务器上提供相应的数据。此选项仅在 带有 5300-03 Recommended Maintenance package 的 AIX 5L V5.3 或后续版本中可用。

注: 只有在服务器上启用复制时才能进行参照导出。使用 **chnfs -R on** 启用复制。

replicas=*rootpath@host***[+***host***][:***rootpath@host***[+***host***]]**

副本位置信息将与导出路径关联。如果当前服务器变为不可用，NFS V4 客户机可以使用副本信息将操作重定向到指定的另一个位置。管理员应该确保在副本服务器上提供相应的数据。因为副本信息适用于整个文件系统，所以指定的路径必须是文件系统的根目录。如果路径不是文件系统的根目录，将禁止导出并打印一条错误消息。名称 `localhost` 不能被用作 *hostname*。这个 **replicas** 选项只对 V4 的导出有意义。如果在允许 V2 或 V3 的导出上使用该选项，该选项是允许的，但 V2 或 V3 服务器将忽略副本信息。如果导出的目录不是副本列表中，条目 *exported directory@current host* 将被作为第一个副本位置添加。此选项仅在 带有 5300-03 的 AIX 5.3 或后续版本中可用。

只有在服务器上启用复制时才能进行副本导出。在缺省情况下，复制未启用。如果将在系统引导时进行副本导出，应该通过使用 **chnfs -R on** 命令启用复制。还可以为 **nfsroot** 指定副本位置。只能使用 **chnfs -R** [*+host*] 完成此操作。如果在列表中未指定当前主机，它将被作为第一个副本主机添加。在这种情况下不需要或不允许 *rootpath*，这是因为 **nfsroot** 只被复制到指定主机的 **nfsroots**。

可以使用 **chnfs** 程序启用或禁用复制。只有在没有 NFS V4 导出活动时才能更改复制方式。如果更改了服务器复制方式，则服务器将不执行在处于先前的复制方式期间服务器发出的文件句柄。这可能导致在保留旧的文件句柄的客户机上发生应用程序错误。更改服务器复制方式时请小心。如果可能，所有挂装到服务器的客户机都应该在更改服务器复制方式之前卸装。可以通过修改副本列表并重新导出目录，更改与该目录关联的副本位置信息。新的副本信息将替换旧的副本信息。

预期 NFS 客户机将在常规基础上刷新副本信息。如果服务器更改用于导出的副本信息，可能需要时间来引起客户机的注意。如果添加了新的副本位置，则这就不成问题，这是因为保留旧信息的客户机仍然拥有正确的（如果不完全）副本信息。除去副本信息可能会有问题，这是因为它可能导致客户机在一段时间内保留不正确的副本信息。要辅助客户机检测新的信息，**exportfs** 将试图接触被复制的目录。这将更改目录上的时间戳记，反过来将导致客户机重新获取该目录的属性。然而，如果被复制的文件系统是只读的，则此操作也许就不可能了。当更改某个目录的副本信息时，要明白可能在更改信息到客户机注意到新的信息之间有一段等待时间。

-o *Options* (continued)

noauto “按现状”接受副本规范。如果尚未指定主要主机名，则请勿自动将其作为一个副本位置插入。

无论何时客户机属性发生更改，所有包含该客户机作为参数的导出条目应该再次导出。可以更改客户机属性的事件包括修改网组或更改客户机的 IP 地址。更改失败会导致服务器使用原有的客户机信息。

Solaris 兼容性

exportfs 命令可能被调用为 **share**、**shareall**、**unshare** 或 **unshareall**。当 **exportfs** 命令作为 **share** 或 **shareall** 调用时，功能分别等同于 **exportfs** 和 **exportfs -a**，除非必须使用 **sec** 选项指定安全方法。当 **exportfs** 命令作为 **unshare** 或 **unshareall** 调用时，功能分别等同于 **exportfs -u** 和 **exportfs -u -a**。

示例

1. 要导出 **/etc/exports** 文件中的全部目录，请输入:

   ```
   exportfs -a
   ```

2. 要从 **/etc/exports** 文件导出一个目录，请输入:

   ```
   exportfs /home/notes
   ```

   在这个示例中，/home/notes 目录被导出。

   注: 要让这个命令工作，必须在 **/etc/exports** 文件中指定 **/home/notes** 目录。

3. 要取消导出目录，请输入:

   ```
   exportfs -u /home/notes
   ```

   在这个示例中，/home/notes 目录不导出。

4. 要显示当前正在导出的目录，请输入:

   ```
   exportfs -v
   ```

5. 要导出不是在 **/etc/exports** 文件中指定的目录，请输入:

   ```
   exportfs -i /home/zeus
   ```

   在这个示例中，/home/zeus 目录没有任何限制地被导出。

6. 要导出一个目录并为网组成员提供访问该目录的许可权，请输入:

   ```
   exportfs access=cowboys:oilers /home/notes -o
   ```

   在这个示例中，/home/notes 目录被导出，并且允许 cowboys 和 oilers 主机的用户有权访问。

7. 要从 **/etc/exports** 文件导出带不同选项的目录，请输入:

   ```
   exports -i -o -root=zorro:silver /directory
   ```

   在这个示例中，/directory 目录被导出，并且 root 用户访问 zorro 和 silver 主机，而不管 **/etc/exports** 文件中指定的访问权限是什么。

8. 要将 **/common/documents** 目录带写许可权导出到使用 Kerberos 认证的客户机或带只读许可权导出到使用 UNIX 认证的客户机，请将以下文本添加到 **/etc/exports** 文件中:

   ```
   /common/documents -sec=krb5,rw,sec=sys,ro
   ```

   然后，输入 exportfs /common/documents 执行导出。

9. 要在 **/usr/info** 上创建一个参照到主机 infoserver 上的 **/usr/info** 目录，将下行添加到 **/etc/exports**，然后导出 **/usr/info**:

   ```
   /usr/info -vers=4,refer=/usr/info@infoserver
   ```

10. 要为主机 backup1 和 backup2 上的 **/common/info** 目录指定副本，请将下行添加到 **/etc/exports**，然后导出 **/common/info**:

    ```
    /common/info -vers=4,replicas=/common/info@backup1:/common/info@backup2,<other options>
    ```

文件

/etc/exports	列出服务器可以导出的目录。
/etc/xtab	列出当前导出的目录。
/etc/hosts	包含网络中每个主机的一个条目。
/etc/netgroup	包含关于网络中每个用户组的信息。
/etc/rc.nfs	包含 NFS 和 NIS 守护程序的启动脚本。

相关信息

chnfsexp 命令、**mknfsexp** 命令、**rmnfsexp** 命令、**showmount** 命令，

《*安全性*》 中的『如何用安全 NFS 导出一个文件系统』。

《*网络与通信管理*》中的『NFS 命令列表』、『NFS 文件列表』和『网络文件系统』。

---

exportvg 命令

用途

从一个物理卷标集中导出卷标组的定义。

语法

exportvg *VolumeGroup*

描述

exportvg 命令会从系统中除去由 *VolumeGroup* 参数指定的卷标组的定义。由于关于卷标组及其内容的所有的系统知识都被除去了，导出的卷标组就不再可以访问了。**exportvg** 命令不会修改卷标组中的任何用户数据。

卷标组是在系统内的一种非共享的资源；它不应该被其他的处理器所访问，直到它被明确地从当前处理器中导出并在其他处理器上导入。**exportvg** 命令的主要的使用，与 **importvg** 命令结合在一起，是允许可移植的卷标在处理器之间交换。只有完整的卷标组才能够被导出，而不是单个的物理卷标。

利用 **exportvg** 命令和 **importvg** 命令，您也可以在两个处理器之间切换共享的物理卷标上的数据的所有权。

注: 要想使用这个命令，您必须要么有 root 用户权限或者要么是 **system** 组的成员。

您可以使用基于 Web 的系统管理器 (wsm) 中的卷应用程序来更改卷标特征。

您可以使用基于 Web 的系统管理器卷应用程序（**wsm lvm** 快速路径）来运行该命令。您也可以使用系统管理界面工具（SMIT）**smit exportvg** 快速路径来运行这个命令。

注:

1. 在其中有一个页面调度空间的卷标的组在该页面调度空间为活动时不能够被导出。在导出拥有活动页面调度空间的卷标的组之前，要确保页面调度空间在系统初始化的时候没有自动激活，然后再重新引导系统。
2. 逻辑卷标的安装点信息如果超过 128 字符长的话，会从 LVCB（逻辑卷标控制块）中丢失。请对超过 128 字符长的安装点作一个注释，因为您需要手动编辑 **/etc/filesystems** 文件，一旦执行了 **importvg** 命令来完整地导入该卷标组的话。

示例

要想从系统中除去卷标组 vg02 请输入:

```
exportvg vg02
```

注: 该卷标组在导出之前必须被断开。

vg02 的定义从系统中除去，该卷标组不能被访问。

文件

/usr/sbin **exportvg** 命令所在的目录.

相关信息

importvg 命令、**varyoffvg** 命令 c、**varyonvg** 命令。

《*操作系统与设备管理*》中的『逻辑卷存储器』说明了逻辑卷管理器、物理卷、逻辑卷、卷组、组织、确保数据完整性以及分配特征。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

《*操作系统与设备管理*》中的『系统管理界面工具』说明了 SMIT 的结构、主菜单以及通过 SMIT 完成的任务。

---

expr 命令

用途

求表达式变量的值。

语法

expr *Expression*

描述

expr 命令读入 *Expression* 参数，计算它的值，然后将结果写入到标准输出。

您必须对 *Expression* 参数应用以下规则:

- 用空格隔开每个项。
- 用 \ （反斜杠） 放在 shell 特定的字符前面。
- 对包含空格和其他特殊字符的字符串要用引号括起来。

整数前面可以放一个一元连字符。在内部，整数被当作 32 位，双互补数。

注: **expr** 命令返回一个 0 来指示一个零值，而不是空字符串。

以下项描述了 *Expression* 参数的运算符和关键字。需要转义的字符由一个 \ （反斜杠） 放在前面。这些项按照优先权递增的顺序列出，具有相等的优先权的运算符分组在 { } （大括号） 中:

Expression1 **\|** *Expression2*

返回 *Expression1*如果它不是空值或者 0 值的话，否则返回 *Expression2*。

Expression1 **\&** *Expression2*

返回 *Expression1*如果两个表达式都不是空值或者 0 值，否则返回一个 0 值。

Expression1 **{ =, \>, \>=, \<, \<=, != }** *Expression2*

如果两个表达式都是整数，返回整数比较的结果；否则它返回的是字符串比较的结果。

Expression1 **{+, -}** *Expression2*

整数值变量的加或者减。

Expression1 **{ *, /, % }** *Expression2*

整数值变量的乘、除或者提供除法的余数。

Expression1 **:** *Expression2*

将 *Expression1* 的运算得到的字符串与 *Expression2* 的运算结果的正则表达式模式进行比较。正则表达式语法与 **ed** 命令相同，除了所有的模式固定到字符串的开始之外（也就是说，只有以字符串的第一个字符开始的序列才被正则表达式匹配）。因此 ^ （插入符号） 在这种情况下就不是特殊字符。

一般地，匹配运算符返回匹配的字符个数（失败的时候返回 0）。如果模式包含了一个子表达式，也就是:

```
\( Expression \)
```

则会返回包含了实际匹配的字符的字符串。

整理顺序可以定义在字符范围内使用的等价类。有关整理顺序和等价类的更多信息，请参阅《*AIX 5L V5.3 本地语言支持指南和参考*大全》中的『了解语言环境的环境变量』。

注: 以下字符串变量会超过标准，其行为可能在不同的操作系统中会有不同。这些字符串变量是“不”可移植的。

match *String1 String2*

与 *Expression1* **:** *Expression2* 相同。

length *String1*

返回 *String1* 的长度。

index *String1 String2*

返回 *String1* 中包含 *String2* 中任意字符的第一个位置。

substr *String1 StartPosition Length*

返回一个以 *StartPosition* 的字符开始的在 *String1* 中的字符串，并且是 *Length* 长度的字符串。

退出状态

此命令返回以下出口值:

0 *Expression* 参数运算值既不为空也不为 0。
1 *Expression* 参数的计算值或者为空或者为 0。
2 *Expression* 参数是不合法的。
>2 发生错误。

注: 在 shell 处理完参数后，**expr** 命令除了利用数值外不能区分运算符和操作数。因此，如果 `$a` 的值是 `j` 的话，命令:

```
expr $a = j
```

就像:

```
expr j = j
```

在 shell 将所有的参数都传给 **expr** 命令之后。以下情况也为 true:

```
expr X$a = Xj
```

示例

1. 要修改一个 shell 变量，请输入:

   ```
   COUNT=`expr $COUNT + 1`
   ```

   这会加 1 到 shell 变量 $COUNT 上。 **expr** 命令以 grave accent 包装，这会导致 shell 将 **expr** 命令的标准输出替换到 COUNT= 命令的标准输出。$COUNT 变量使用前必须初始化。

2. 要想得到 **$STR** shell 变量的长度，请输入:

   ```
   LENGTH=`expr $STR : ".*"`
   ```

   这会将 LENGTH 变量设置成由 : （冒号）运算符给出的值。.*（点，星号）模式会对任何字符串从头到尾去匹配，因此冒号运算符给出 $STR 变量的长度作为匹配字符的数目。注，.* 必须在引号内，来防止 shell 将 *（星号）处理成模式匹配字符。引号不是模式中的一部分。

   如果 $STR 变量设置成空字符串或者包含了任何空格（空白或者制表键）的话，则该命令会显示错误信息 expr: syntax error。这是因为 shell 通常不会将空字符串传递给命令。在这种情况下，**expr** 命令只能看见:

   ```
   :.*
   ```

   shell 还会除去单个的引号。不起作用的原因是因为冒号运算符要求有两个值。这个问题可以通过将 shell 变量以双引号括起来解决:

   ```
   LENGTH=`expr "$STR" : ".*"`
   ```

   现在，如果 $STR 变量的值为空的话，LENGTH 变量被设置成 0 值。通常我们推荐将 shell 变量用双引号括起来。请勿将 shell 变量用单引号标记括起来。

3. 要想使用一个字符串的一部分，请输入:

   ```
   FLAG=`expr "$FLAG" : "-*\(.*\)"`
   ```

   只要 $FLAG shell 变量前面有连字符，就会除去连字符。冒号运算符给出 FLAG 变量的一部分，该变量由在 \(和 \) 字符（反斜杠，开括号和反斜杠，闭括号）中间的子表达式匹配。如果您忽略 \(和 \) 子表达式字符的话，冒号运算符会给出匹配的字符数目。

   如果 $FLAG 变量设置成 -（连字符），则该命令会显示一个语法错误消息。这是因为 shell 会将 $FLAG 变量的值替换，在运行 **expr** 命令之前。**expr** 命令不知道连字符是一个变量的值。它只能理解:

   ```
   - : -*\(.*\)
   ```

   并且它会将第一个连字符解释成减号运算符。要消除这种问题，请使用:

   ```
   FLAG=`expr "x$FLAG" : "x-*\(.*\)"`
   ```

4. 要想使用 **expr** 命令在 **if** 语句中，请输入:

   ```
   if expr "$ANSWER" : "[yY]" >/dev/null
   则返回 ANSWER 以 "y" 或者 "Y" 开始fi
   ```

   如果 $ANSWER 变量以 y 或者 Y 开始的话，**if** 语句的 then 部分会执行。如果匹配成功表达式的结果为 1，并且 **expr** 命令会返回一个为 0 的出口值，这个出口值被 **if** 语句识别成逻辑值 True。如果匹配失败，结果为 0，出口值为 1。

将 **expr** 命令的标准输出重新定向到 **/dev/null** 特殊文件，而废弃该表达式的结果。如果您没有对它重新定向，结果就会写到标准输出中，通常是您的工作站显示器。

5. 考虑以下表达式:

   ```
   expr "$STR" = "="
   ```

   如果 $STR 变量有值为 =（等号），则在 shell 处理完这个命令之后，**expr** 命令会看到表达式:

   ```
   = = =
   ```

   expr 命令将这个解释成在一行中的三个等号运算符，并显示一个语法出错消息。一旦 shell 变量的值与 **expr** 运算符中的一个相同，这种情况就会发生。避免这种问题的方法是，将该表达式写成:

   ```
   expr "x$STR" = "x="
   ```

6. 要想返回 $SHELL 环境变量 /usr/bin/ksh 的长度，请输入:

   ```
   expr length $SHELL
   ```

   显示如下:

   ```
   12
   ```

7. 要想返回 ″de″ 字符串中的任何一个字符在 ″abcdef″ 中出现的第一个位置，请输入:

   ```
   expr index abcdef de
   ```

   显示如下:

   ```
   4
   ```

8. 要想返回 ″fd″ 字符串的任何字符在 ″abcdef″ 中出现的第一个位置，请输入:

   ```
   expr index abcdef fd
   ```

   显示如下:

   ```
   4
   ```

9. 要想返回 ″Goodnight Ladies″ 字符串中从位置 11 开始的 6 个字符长的字符串，请输入:

   ```
   expr substr "Goodnight Ladies" 11 6
   ```

   显示如下:

   ```
   Ladies
   ```

文件

/usr/bin/expr 包含了 **expr** 命令。

相关信息

bsh 命令、**csh** 命令、**ed** 命令、**ksh** 命令。

《*操作系统与设备管理*》中的『命令』。

《*AIX 5L V5.3 本地语言支持指南和参考*大全》中的『本地语言支持概述』。

exptun 命令

用途

导出通道定义，还可以有选择地导出所有和该通道相关的用户定义过滤器规则。创建一个该通道伙伴可以使用的一个通道导出文件和一个可选的过滤器规则导出文件。

语法

exptun [**-v 4|6**] **-f** *directory* [**-t** *tid_list*] [**-r**] [**-l manual**]

描述

使用 **exptun** 命令来为通道伙伴创建一个要导入的通道上下文导出文件和一个可选择的过滤器规则附加文件。这个命令不会激活一个通道，它简单地为通道伙伴创建所需要的文件。

注: 生成的导出文件包含了由该通道使用的密钥。用操作系统的文件系统保护功能来保护这些文件。

标志

-f 定义导出文件要写入的目录。如果不存在，该目录将被创建。这些导出文件可以被送到通道伙伴让它导入。我们建议每个通道伙伴的导出文件应该具有不同的目录规格。

-l 您想要导出的通道类型。如果指定为手动，则只有手动 ibm 通道可以导出。

-r 导出与这些通道相关的所有用户定义的过滤器规则。如果这个标志没有被使用，则只有通道定义会导出。

-t 指定导出文件要用到的通道 ID 的列表。该列表可以指定成一个由 ″**,**″ 或者 ″**-**″ (1, 3, 10, 50-55)隔开的通道 ID 序列。如果该标志没有被使用，则从通道数据库来的所有的通道定义将会被导出。

-v 被导出通道的 IP 版本。值 **4** 说明 IP V4 的通道。值 **6** 说明 IP V6 的通道。如果该标志未被使用，IP V4 和 IP V6 的通道定义将被导出。

相关信息

chtun 命令、**gentun** 命令、**imptun** 命令、**lstun** 命令、**mktun** 命令和 **rmtun** 命令。

extendlv 命令

用途

通过在卷标组内增加未分配的物理分区来增加逻辑卷的大小。

语法

增加可用的物理分区

extendlv [**-a** *Position*] [**-e** *Range*] [**-u** *Upperbound*] [**-s** *Strict*] *LogicalVolume Partitions* [*PhysicalVolume* ...]

增加特定的物理分区

extendlv [**-m***MapFile*] *LogicalVolume Partitions*

描述

extendlv 命令可以增加分配给 *LogicalVolume* 的逻辑分区数，通过给 *Partitions* 参数代表的附加逻辑分区分配数目来实现。*LogicalVolume* 参数可以是一个逻辑卷名或者逻辑卷 ID。要限制对特定物理卷的分配，请在 *PhysicalVolume* 参数中使用一个或者多个物理卷的名称；否则，卷标组中的所有物理卷就都可以分配给新的物理分区。

缺省情况下，逻辑卷利用现存的特征进行扩展，这些特征在您使用 **lslv** 命令时会显示出来。要只覆盖新分区的这些现有特征，请使用标志为这些特征选择其他值。

逻辑卷的缺省最大分区数为 512。在将逻辑卷扩展到超过 512 个逻辑分区之前，先使用 **chlv** 命令增加该缺省值。

缺省的分配策略是使用每个逻辑卷拷贝的物理卷的最小数，将属于一个拷贝的物理分区放置得尽可能连续，然后将物理分区放置在 **-a** 标志指定的希望区域。另外，在缺省情况下，每个逻辑分区的拷贝是放在一个单独的物理卷中的。

当使用 **extendlv** 命令时，您可以指定逻辑卷的大小为 512 块 /KB/MB/GB。（见第 388 页的『示例』。）

注:

1. 在扩展一个带状逻辑卷时，分区数必须带状宽度的偶数倍。
2. 我们建议使用很大的分区数（超过 800MB）的逻辑卷应该分段逐步扩展。
3. 对逻辑卷进行的修改在文件系统中不会反应出来。要更改文件系统的特征，请使用 **chfs** 命令。
4. 您必须要么有 root 用户权限，要么是 **system** 组的成员才能使用这个命令。
5. **extendlv** 命令对于抽点转储式的卷标组不允许。

您可以使用在卷的应用程序基于 Web 的系统管理器来更改卷的特征。您还应该使用系统管理界面程序（SMIT）**smit extendlv** 的快速路径来运行这个命令。

标记

注: **-e**、**-s** 和 **-u** 标志与使用组合分割区的逻辑卷一起使用时无效。

-a *Position*
设置物理卷内的分配策略（在物理卷中的逻辑分区的位置）。*Position* 变量可以是以下中的一个:

- **m** 在每个物理卷的外部的中间段中分配逻辑分区。这是缺省的位置。
- **c** 在每个物理卷的中心段分配逻辑分区。
- **e** 在每个物理卷的外部边缘部分来分配逻辑分区。
- **ie** 在每个物理卷的内边缘部分来分配逻辑分区。
- **im** 在每个物理卷的内部中间部分来分配逻辑分区。

-e *Range*
设置物理卷内的分配策略（使用提供最优分配的卷，扩展物理卷的数目）。*Range* 变量的值由 *Upperbound* 变量所限制（用 -u 标志设置），可以是以下中的一个:

- **x** 覆盖最大数目的物理卷分配逻辑分区。
- **m** 覆盖最小数目的物理卷分配逻辑分区。

-m *MapFile*
指定了要分配的准确的物理分区。按照在 *MapFile* 参数给出的顺序使用分区。属于一个拷贝的所有的物理分区在分配下一个逻辑卷的拷贝之前进行分配。*MapFile* 参数的格式是: `PVname:PPnum1[-PPnum2]`。 在这个示例中，`PVname` 是物理卷名称（如 `hdisk0`）。它是每个物理分区一个记录或者是连续物理分区的一个范围。`PPnum` 是物理分区号。

-s *Strict*　　确定严格的分配策略。一个逻辑分区的拷贝可以分配成共享或不共享相同的物理卷。*Strict* 变量由以下中的一个来代表:

y　　设置一个严格的分配策略，因此一个逻辑分区的拷贝不能共享相同的物理卷。

n　　不设置严格的分配策略，因此一个逻辑分区的多个拷贝可以共享相同的物理卷。

s　　设置一个超严格的分配策略，因此对于一个镜像所分配的分区不能共享从其他镜像校验的物理卷。

注: 当更改一个非超严格的逻辑卷到一个超严格的逻辑卷时，您必须指定物理卷或者使用 **-u** 标志

-u *Upperbound*　　为新的分配设置物理卷的最大数目。*Upperbound* 变量的值应该在 1 到物理卷的总数之间。当使用超级限制时，上限表示每个镜像副本允许的最大物理卷数。当使用组合分割区的逻辑卷时，上限必须为 *Stripe_width* 的倍数。

示例

1. 要用三个逻辑分区来增加由 lv05 目录代表的逻辑卷的大小，请输入:

   ```
   extendlv lv05 3
   ```

2. 要请求一个名为 lv05 的最小为 10MB 的逻辑卷，请输入:

   ```
   extendlv lv05 10M #
   ```

   extendlv 命令将确定创建最小那种大小的逻辑卷所需要的分区数。

   您可以使用大写和小写字母，如下所示:

   ```
   B/b       512 byte blocks
   K/k       KB
   M/m       MB
   G/g       GB
   ```

文件

/usr/sbin/　　**extendlv** 命令所在的目录。

相关信息

chfs 命令、**chlv** 命令、**chpv** 命令、**lslv** 命令、**mklv** 命令、**mklvcopy** 命令。

《*操作系统与设备管理*》中的『逻辑卷存储器』说明了逻辑卷管理器、物理卷、逻辑卷、卷组、组织、确保数据完整性以及分配特征。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

《*操作系统与设备管理*》中的『系统管理界面工具』说明了 SMIT 的结构、主菜单以及通过 SMIT 完成的任务。

---

extendvg 命令

用途

向一个卷分组增加物理卷。

语法

extendvg [**-f**] *VolumeGroup PhysicalVolume ...*

描述

extendvg 命令会增加 *VolumeGroup* 的大小，通过增加一个或多个 *PhysicalVolumes* 来实现。

物理卷要被检查，来验证它不在其他卷组中。如果系统认为该物理卷属于正在要更改的卷组，它就会退出。但是如果系统从没有更改的卷标检测到一个描述区域，它会提示用户确认继续该命令。物理卷的以前的内容丢失，因此用户在使用覆盖功能的时候必须当心。

注: 要使用这个命令，您必须有 root 用户权限或者是 **system** 组的成员。

对于在 AIX 5.3 之前创建的卷组，或者对于在 AIX 5.3 上创建、但通过 **varyonvg -M** 标志激活的卷组，如果物理卷的最大传输大小小于该卷组的逻辑跟踪组大小，则 **extendvg** 将失败。对于在 AIX 5.3 上创建并且未用 **varyonvg -M** 标志激活的卷组，如果物理卷具有小于卷组的逻辑跟踪组大小的最大传送大小，则 extendvg 将动态地降低卷组的逻辑跟踪组大小。

注: 在快照卷组上不允许 **extendvg** 命令。

您可以使用基于 Web 的系统管理器 (wsm) 中的卷应用程序来更改卷特征。您还可以使用系统管理界面程序（SMIT）**smit extendvg** 快速路径来运行这个命令。

注: 该命令将不能向卷组中增加一个磁盘，如果该磁盘指示被一个第三方的卷管理器所管理的话。为了覆盖并清除第三方卷管理器的磁盘，要使用 **chpv -C** *HDiskName*。

标志

-f 强制物理卷增加到指定的卷组中，除非它是设备配置数据库中另外的卷组中的成员，或者是一个活动的卷组中的成员。

示例

要将物理卷 `hdisk3` 和 `hdisk8` 增加到卷组 `vg3` 中，请输入:

```
extendvg vg3 hdisk3 hdisk8
```

注: 卷组必须在扩展之前更改。

限制

extendvg 命令不能在快照卷组上运行。

文件

/usr/sbin/extendvg 包含了 **extendvg** 命令。

相关信息

reducevg 命令。

《*操作系统与设备管理*》中的『逻辑卷存储器』说明了逻辑卷管理器、物理卷、逻辑卷、卷组、组织、确保数据完整性以及分配特征。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

《*操作系统与设备管理*》中的『系统管理界面工具』说明了 SMIT 的结构、主菜单以及通过 SMIT 完成的任务。

---

f 命令

用途

显示用户信息。该命令和 **finger** 命令是一样的。

语法

{ **f** | **finger** } [[**-b**] [**-h**] [**-l**] [**-p**]] | [[**-i**] [**-q**] [**-s**] [**-w**]]

[**-f**] [**-m**] [*User* | *User* **@***Host* | **@***Host*]

描述

/usr/bin/f 命令显示了当前登录到某台主机上的用户的有关信息。输出格式随着显示信息的选项而更改。

缺省格式

缺省格式包括以下的条目:

- 登录名
- 用户全名
- 终端名称
- 写状态（终端名称前的 *（星号）指示没有写权限）

对主机的每个用户来说，如果已知，缺省信息列表也包括如下条目:

- 空闲时间（如果是一单个整数，则空闲时间以分钟来表示，如果存在 a:（冒号）符号，则以小时和分钟来表示，如果存在 “d” 符号，则以天和小时来表示。）
- 登录名
- 位置特定信息

  位置特定的信息将从 **/etc/passwd** 文件中的 gecos 字段检索。gecos 字段可能包含一个后跟逗号的用户全名。跟在逗号后面的所有信息由带有位置特定信息的 finger 命令显示。

长格式

无论用户的名称列表何时给出，**f** 命令使用长格式。（帐户名称以及用户的名和姓会被接受。）这种格式是多行的，包括上面描述的所有信息和如下信息:

- 用户的 **$HOME** 目录
- 用户的登录 shell
- 用户 **$HOME** 目录中 **.plan** 文件的内容。
- 用户目录 **$HOME** 中 **.project** 文件的内容

f 命令也可用于寻找远程系统上的用户。格式是指定用户为 *User@Host*。如果省略了用户名，**f** 命令提供了远程系统上的标准格式列表。

用您偏爱的文本编辑器创建 **.plan** 和 **.project** 文件并且将这些文件放到您的主目录中。**f** 命令在显示 **.plan** 文件和 **.project** 文件的内容时，使用 **toascii** 子例程去转换正常的 ASCII 字符范围以外的字符。**f** 命令在每一个被转换的字符前面显示一个 M-。

当您用 *User* 参数来指定用户的时候，您可以指定用户的名或者姓或者帐户名。当指定用户时，在指定的主机上，**f** 命令仅以长格式返回那些用户的有关信息。

有关 **f** 命令的其他信息，请参阅《*网络与通信管理*》中的『TCP/IP 的安装』。

标志

-b 给出一个简短、长格式的列表。
-f 禁止在输出中打印报头行（定义了将被显示的字段的第一行）。
-h 禁止以长格式和简短长格式打印 **.project** 文件。
-i 给出空闲时间的一个快速列表。
-l 给出一个长格式的列表。
-m 假定*用户* 参数指定了一个用户标识（用以任意的访问控制），*不是*一个用户的登录名称。
-p 禁止以长格式和简短长格式类型打印 **.plan** 文件。
-q 给出一个快速的列表。
-s 给出一个短格式列表。
-w 给出一个狭窄的、短格式的列表。

参数

@Host 指定远程主机上的所有登录进入的用户。
User 指定一个本地用户标识（用于任意的访问控制）或者本地的用户登录名称，正如在 **/etc/passwd** 文件中指定的一样。
User@Host 在远程主机上指定一个以长格式显示的用户标识。

示例

1. 要获取登录到主机 alcatraz 上的所有用户的有关信息，请输入:

   ```
   f @alcatraz
   ```

   就会显示与以下类似的信息:

   ```
   [alcatraz.austin.ibm.com]
   Login     Name         TTY Idle         When      Site Info
   brown     Bob Brown    console   2d   Mar 15 13:19
   smith     Susan Smith     pts0  11:   Mar 15 13:01
   jones     Joe Jones       tty0    3   Mar 15 13:01
   ```

   用户 brown 在控制台登录，用户 smith 从伪的电传线路 pts0 上登录，用户 jones 从 tty0 上登录。

2. 要获取 alcatraz 上的用户 brown 的有关信息，请输入:

   ```
   f brown@alcatraz
   ```

   就会显示与以下类似的信息:

```
Login name: brown
Directory: /home/brown    Shell: /home/bin/xinit -L -n Startup
On since May 8 07:13:49 on console
No Plan.
```

3. 要获取本地主机上用户 brown 的有关信息并以短格式显示，请输入:

```
f -q brown
```

就会显示与以下类似的信息:

```
Login           TTY             When
brown           pts/6           Mon Dec 17 10:58
```

文件

/usr/bin/f	包含了 **f** 命令。
/etc/utmp	包含当前登录的用户列表。
/etc/passwd	定义用户的帐户、名称和主目录。
/etc/security/passwd	定义用户密码。
/var/adm/lastlog	包含过去的登录时间。
$HOME/.plan	包含一个用户计划的一行描述的可选文件。
$HOME/.project	可选的包含一个用户的计划任务的文件。

相关信息

hostname 命令，**rwho** 命令，**finger** 命令。

fingerd 守护程序。

《*网络与通信管理*》中的『显示有关已登录用户的信息的命令』。

《*网络与通信管理*》中的『通信和网络』。

---

factor 命令

用途

因子分解一个数。

语法

factor [*Number*]

描述

当调用 **factor** 命令时没有为 *Number* 参数指定一个值的话，该命令将等待输入一个小于 1E14（100,000,000,000,000）的正数。 然后将那个数的素数因子输出到标准输出上去。它以相应的次数显示每一个因子。要退出命令，输入 0 或任何非数字字符。

当调用 **factor** 命令带有一个参数时，该命令将确定 *Number* 参数的素数因子并将结果输出到标准输出上去，然后退出。

示例

要计算 123 的素数因子，请输入:

```
factor 123
```

显示以下内容:

```
123
     3
     41
```

文件

/usr/bin/factor 包含 **factor** 命令。

相关信息

bc 命令

---

fc 命令

用途

处理命令历史列表。

语法

打开一个编辑器去修改或重新执行以前输入的命令

fc [**-r**] [**-e** *Editor*] [*First* [*Last*]]

生成一个以前输入的命令的列表

fc -l [**-n**] [**-r**] [*First* [*Last*]]

生成带执行时间的以前输入的命令的列表

fc -t [**-n**] [**-r**] [*First* [*Last*]]

重新执行以前输入的命令

fc -s [*Old***=** *New*] [*First*]

描述

fc 命令显示了历史命令文件内容或调用一个编辑器去修改并重新执行以前在 shell 中输入的命令。

历史命令文件按编号列出命令。列表中的第一个编号可以任意选择。每一个命令与编号的关系不会更改，除非用户登录进系统并且没有其他进程访问过该列表。在这种情况下，系统将重新设置编号，并将余下的最老的命令编号设置为 1。

如果历史命令文件中的编号达到了一个大于 **HISTSIZE** 环境变量值，或是大于 32767 的极限值，不管是哪一种情况，shell 会将编号重新设为 1。尽管可选编号重新开始设置，但 **fc** 命令会按时间顺序维护命令序列。例如，有三个命令按照编号 32766,32767 和 1（被折返）排序，则编号 32767 的命令仍被认为在编号为 1 的命令之前。

可以使用 **-l**（L 的小写）标志列出历史文件中的命令。当没有使用 **-l** 标志并且使用了 **-e***Editor* 标志编辑命令，其结果行将输入到历史文件的末尾并且被 shell 重新执行（**fc -e** *Editor* 命令不会被输入进历史命令列表中）。如果编辑器返回一个非零的退出状态值，这将禁止历史文件中的条目和命令重新执行。

与 **fc** 命令一起使用的命令行上的任何变量赋值或者重定向运算符将再一次调用前面的命令，并且抑制 **fc** 命令和前面的命令所产生的标准错误。示例:

```
fc -s -- -1 2>/dev/null
```

标志

-e *Editor*
使用指定的编辑器编辑命令。 *Editor* 参数应是一个命令名称。该命令用 **PATH** 环境变量指定位置。当没有指定 **-e** 标志时，环境变量 **FCEDIT** 中的值被用作缺省值。如果环境变量 **FCEDIT** 是空值或没有设置，则使用 ed 编辑器。

-l
（L 的小写）列出了历史文件中的命令。不调用编辑器去修改它们。按照 *First* 和 *Last* 参数所指定的顺序写命令，就象 **-r** 标志所作用的，在每一个命令的前面有一个命令编号。

-n
当与 **-l** 标志一起使用时，隐藏命令编号。

-r
逆转所列出命令的顺序（当使用 **-l** 标志）或者逆转所编辑的命令顺序（当没有指定 **-l** 标志时）。

-s
不调用编辑器重新执行一个命令。如果也没有指定 *First* 参数，则 **-s** 标志重新执行前一个命令。

-t
在历史文件中列出命令及其执行时间。工作过程与 **-l** 标志类似，但是还显示命令的执行时间。
注: 如果之前通过设置 EXTENDED_HISTORY=ON 记录了时间字段，则显示格式化的时间字段，否则显示“?”。

参数

First or *Last*
选择要列出或编辑的命令。**HISTSIZE** 环境变量值确定了能够访问到的以前所输入命令的数量。*First* 参数和 *Last* 参数必须具有以下数值中的一个:

[+] *Number*
表示一个特定的命令编号。**-l** 标志可以显示命令编号。缺省值为一个 +（加符号）字符。

-*Number*
表示一个以前执行的命令，由存储在历史列表中的命令编号指定。例如，-1 指出了前一个刚执行过的命令。

String
指出了最近所输入的命令，该命令以指定的字符串开头。如果指定了 *Old***=***New* 参数但没有指定 **-s** 标志，则来自 *First* 参数的字符串中不能包含一个嵌入的 **=**（等于符号）。

当使用 **-s** 标志时，省略 *First* 参数将导致使用前一个命令。

当没有指定 **-s** 标志时，将应用以下规则:

- 当使用 **-l** 标志时，省略 *Last* 参数将会导致缺省使用前一个命令。
- 当使用 **-r**、**-n** 和 **-e** 标志时，省略 *Last* 参数会导致缺省使用 *First* 参数。
- 如果 *First* 参数和 *Last* 参数都被省略了，则列出前 16 个命令或编辑前一个命令（取决于是否使用 **-l** 标志）。
- 如果使用了 *First* 参数和 *Last* 参数，则列出所有的命令（当指定了 **-l** 标志时）或编辑所有的命令（当没有指定 **-l** 标志）。通过将所有的命令同时显示在编辑器中，可以实现编辑多个命令，每个命令都另起一新行。如果 *First* 参数表示的命令比 *Last* 参数表示的命令要更新的话，则命令是以相反的顺序被列出或编辑的。这与使用 **-r** 标志的效果是一样的。例如，以下第一行的命令与第二行上相应的命令是等价的:

```
fc  -r  10  20          fc      30  40
fc      20  10          fc  -r  40  30
```

- 当使用某一范围中的命令时，如果 *First* 参数或 *Last* 参数指定了不在历史列表的值时，这并不是一个错误。**fc** 命令会替换表示列表中合适的最早或最近的命令的值。例如，如果在历史列表中仅有编号从 1 到 10 的十个命令，则命令:

  ```
  fc  -l
  fc   1   99
  ```

  分别列出和编辑所有这十个命令。

Old=*New*　　在要重新执行的命令中，用新字符串去代替第一个出现的老字符串。

环境变量

以下环境变量会影响 fc 命令的执行:

EXTENDED_HISTORY　　用来控制历史文件中命令执行时间的记录。如果变量设置为 ON，则记录时间，否则不记录时间。

FCEDIT　　当 shell 扩展该环境变量时，该变量确定了 **-e** *editor* 变量的缺省值。如果环境变量 **FCEDIT** 为空值或没有被设置，则缺省使用 ed 编辑器。

HISTDATEFMT　　它用于控制 **fc –t** 命令显示的时间格式。例如，如果 HISTDATEFMT=%Y，则 **fc -t** 将显示执行命令的年份。格式与 **date** 命令所做的相似。

HISTFILE　　确定历史命令文件的路径名。如果环境变量 **HISTFILE** 没有被设置，则 shell 可能会尝试访问或创建在用户主目录中的 **.sh_history** 文件。

HISTSIZE　　确定一个十进制数值，该数值表示了可以访问的以前输入的命令的数量限值。如果没有设置该变量，则使用缺省值 128。

退出状态

以下出口值被返回:

0　　成功完成列表。
>0　　发生错误。

否则退出状态是由 **fc** 命令执行的命令的状态。

示例

1. 要对最近所使用的命令调用环境变量 **FCEDIT** 所定义的编辑器（缺省的编辑器是 **/usr/bin/ed**），请输入:

   ```
   fc
   ```

   当完成编辑之后，执行该命令。
2. 要列出执行过的前两个命令，请输入:

   ```
   fc -l -2
   ```
3. 要找到以 cc 字符开始的命令，且将 foo 更改为 bar，并显示和执行该命令，请输入:

   ```
   fc -s foo=bar cc
   ```
4. 要列出先前执行的命令及其执行时间，请输入:

   ```
   fc –t
   ```

文件

/usr/bin/ksh　　包含了 Korn shell 的内置命令 **fc**。

/usr/bin/fc 包含了 **fc** 命令。

相关信息

ksh 命令。

---

fccheck 命令

用途

在首次故障数据捕捉实用程序（FFDC）实用程序中执行基本问题确定的任务。

语法

/usr/sbin/rsct/bin/fccheck [**-q**] | [**-h**]

描述

fccheck 为首次故障数据捕捉实用程序执行基本问题确定。该命令在本地节点上检查以下的情况和信息:

- 检查 FFDC 错误堆栈用法在当前进程环境中是否被禁用了。
- 获得 FFDC 当前所使用的 IP 地址去识别出本地节点。
- 检查 **/var/adm/ffdc/stacks** 是否可用，如果可用，其目录所驻留的文件系统还有多少可用空间。检查是否无足够的空间去创建 FFDC 错误堆栈。
- 检查 **/var/adm/ffdc/dumps** 是否可用，如果是，其目录所驻留的文件系统还有多少可用空间。

这些测试结果显示在标准输出上，除非指定了 ″quiet″ 选项。**fccheck** 设置了一个退出状态值，用于指示在测试过程中所检测出来的最严重的情况。

标志

-h 在标准输出上面显示帮助和用法信息。不执行其他处理。

-q 指定的 ″quiet″ 模式。该命令不将测试的结果显示在标准输出上。该命令的退出状态必须被用来确定测试的结果。如果检测到的情况多于一种以上，则退出状态反映了 **fccheck** 所检测到的最严重的情况。

退出状态

该命令可产生以下整数形式的退出状态代码:

0 **fccheck** 所测试的所有情况都处于正常的操作参数内。

2 成功显示了帮助信息。没有执行进一步的处理。

12 没有执行检查。该命令指定了无效的选项。

19 **/var/adm/ffdc/stacks** 目录没有被安装或不存在。

20 不能访问或检查 **/var/adm/ffdc/stacks** 路径上的一个或多个目录。在该路径上的一个或多个目录的许可可能已经被更改来禁止访问。

24 不能访问或检查 **/var/adm/ffdc/dumps** 路径上的一个或多个目录。在该路径上的一个或多个目录的许可可能已经被更改来以禁止访问。

32 **/var/adm/ffdc/dumps** 目录没有被安装或不存在。

40 在 **/var/adm/ffdc/stacks** 目录中没有足够的可用空间用于在本地节点上创建 FFDC 错误堆栈。

41 不能从操作系统中获得文件系统信息。这指出了操作系统本身一个潜在的问题。

42 在此进程环境中禁用了 FFDC 错误堆栈的创建和用法。

示例

要在本地节点上使用 FFDC 实用程序检查可能的问题:

```
fccheck
fccheck Status: All tests completed
```

如果本地节点禁用了 FFDC 错误堆栈的创建，则 **fccheck** 将指示这种情况为一个问题:

```
fccheck

fccheck Status: Creation and use of FFDC Error Stacks has been expressly
disabled in the current execution environment. Any processes created in
the current execution environment cannot create their own FFDC Error Stacks
or inherit use of existing FFDC Error Stacks.

fccheck Status: All checks completed. Examine the previous status output for
possible FFDC problem conditions and take the recommended actions listed in
these messages.
```

相关信息

命令: **fcclear**、**fcinit**

---

fcclear 命令

用途

除去本地节点上的 FFDC 错误堆栈和详细信息数据文件。

语法

/usr/sbin/rsct/bin/fcclear -h | [**-d** *filename* [,*filename*,...]] [**-D** *filename* [,*filename*,...]] [**-f** *FFDC_Failure_ID* [,*FFDC_Failure_ID*,...]] [**-F** *FFDC_Failure_ID* [,*FFDC_Failure_ID*,...]] [**-s** *file_name*[,*filename*,...]] [**-S** *file_name* [,*filename*,...]] [**-t** *days*]]

描述

fcclear 命令用于从本地节点上除去 FFDC 错误堆栈文件，这些文件不再需要被用来进行问题确定。包含特定 FFDC 故障标识符的记录的 FFDC 错误堆栈文件会被除去，另外也会除去特定的 FFDC 错误堆栈文件。FFDC 错误堆栈里的各个独立的条目不会被除去。

使用 **-t** 选项，**fcclear** 可用来除去日期超过了特定天数的 FFDC 错误堆栈文件。要使 **fcclear** 自动地清除不需要的 FFDC 错误堆栈，请参考 **cron** 命令获得有关自动执行命令的内容。

要从本地节点上除去所有的 FFDC 错误堆栈，指定天数选项的参数为 0。

标志

-d 通过指定一个或多个详细信息数据文件名的列表，来除去这些文件。文件名可以是绝对路径名，也可以是相对于 **/var/adm/ffdc/dumps** 目录的相对路径名。如果在本地节点上存在这些文件，则除去它们。该命令不能除去远程节点上的文件。如果给出了一个以上的文件名，必须用逗号（,）分隔文件名，中间不能有空格。

-D 通过给出一个或多个详细信息文件名的列表，来保存这些文件。文件名可以是绝对路径名，也可以是相对于 **/var/adm/ffdc/dumps** 目录的相对路径名。如果在本地节点上存在这些文件，则保留它们。该命令不能保留远程节点上的文件。如果给出了一个以上的文件名，必须用逗号（,）分隔文件名，中间不能有空格。

-f 通过指定一个或多个 FFDC 故障标识符的列表，来除去 FFDC 错误堆栈文件。如果在本地节点上存在与这些 FFDC 故障标识符相关的 FFDC 错误堆栈，则它们会被找到并被除去。远程节点上的 FFDC 错误堆栈将不被除去。如果给出了一个以上的 FFDC 故障标识符，必须用逗号（,）分隔标识符，中间不能有空格。

-F 通过给出一个或多个 FFDC 故障标识符的列表，来保存 FFDC 错误堆栈文件。如果在本地节点上存在与 FFDC 故障标识符相关的 FFDC 错误堆栈，则它们会被找到并被保留。远程节点上的 FFDC 错误堆栈将不被保留。如果给出了一个以上的 FFDC 故障标识符，必须用逗号（,）分隔标识符，中间不能有空格。

-h 在标准输出设备上面显示帮助和用法信息。不执行其他处理。

-s 通过给出一个或多个 FFDC 错误堆栈文件名的列表，来除去 FFDC 错误堆栈文件。这些文件名可能是绝对路径名，或是相对于 **/var/adm/ffdc/stacks** 目录的相对路径名。如果在本地节点上存在这些文件，则除去它们。该命令不能除去远程节点上的 FFDC 错误堆栈。如果给出了一个以上的文件名，必须用逗号（,）分隔文件名，中间不能有空格。

-S 通过给出一个或多个 FFDC 错误堆栈文件名的列表，来除去 FFDC 错误堆栈文件。这些文件名可能是绝对路径名，或是相对于 **/var/adm/ffdc/stacks** 目录的相对路径名。如果在本地节点上存在这些文件，则除去它们。该命令不能除去远程节点上的 FFDC 错误堆栈。如果给出了一个以上的文件名，必须用逗号（,）分隔文件名，中间不能有空格。

-t 指示超过了特定天数的 FFDC 错误堆栈和详细信息数据文件应该从本地节点上除去。该选择条件是独立于其他选择条件的。

退出状态

fcclear 命令在结束时生成以下的退出状态值:

0 命令成功结束。如果没有 FFDC 错误堆栈文件或详细信息数据文件与选择条件相匹配，则命令可能成功结束。

2 成功显示了帮助信息。没有执行进一步的处理。

10 没有文件从本地系统上除去。该命令没有指定一个需要的选项。

11 没有文件从本地系统上除去。-t 选项的参数值不是数字。

12 没有文件从本地系统上除去。调用者指定了一个未知选项。

19 **/var/adm/ffdc/stacks** 目录不存在或没有安装。

26 没有文件从本地系统上除去。相同的选项被指定了一次以上。

28 没有文件从系统上除去。调用者提供了选项，要求该命令除去和保留同一个文件。当该命令用户指定 FFDC 故障标识符时，会发生这种情况，该故障标识符是记录在通过命令名指定的一个 FFDC 错误堆栈文件中的。

示例

要从本地节点上除去任何超过七天的 FFDC 错误堆栈以及详细信息数据文件:

```
fcclear -t 7
```

要除去所有超过七天的 FFDC 错误堆栈和详细信息数据文件，而保留其中含有 FFDC 故障标识符 **/3Iv04ZVVfvp.wtY0xRXQ7.....**的信息的 FFDC 错误堆栈，请发布命令:

```
fcclear -t 7 -F /3Iv04ZVVfvp.wtY0xRXQ7...................
```

要除去包含 “FFDC 故障标识符” **/3Iv04ZVVfvp.wtY0xRXQ7.....** 的记录的 “FFDC 错误堆栈” 文件，请发出命令:

```
fcclear -f /3Iv04ZVVfvp.wtY0xRXQ7...................
```

要从系统中删除 FFDC 错误堆栈文件 **myprog.14528.19990204134809** 和 **a.out.5134.19990130093256** 以及详细信息数据文件 **myprog.14528.19990204135227**:

```
fcclear -s myprog.14528.19990204134809,a.out.5134.19990130093256
    -d myprog.14528.19990204135227
```

要扩展前一条命令去删除指定的文件以及任何超过了 14 天的 FFDC 错误堆栈和详细信息数据文件:

```
fcclear -s myprog.14528.19990204134809,a.out.5134.19990130093256
    -d myprog.14528.19990204135227 -t 14
```

相关信息

命令: **fccheck**、**fcreport** 和 **fcstkrpt**

---

fcdecode 命令

用途

将一个首次故障数据捕捉（FFDC）故障标识符从它的标准格式转换为它的组成部件，并将信息以可读的格式显示在标准输出设备上。

语法

/usr/sbin/rsct/bin/fcdecode *FFDC_Failure_ID* [,*FFDC_Failure_ID*,...] | **-h**

描述

fcdecode 将 42 个字符长的 FFDC 故障标识符解码到它的组成部件，并以可读的格式显示这些部件。该命令的输出结果显示了来自 FFDC 故障标识符中的以下信息:

- 该报告所驻留的节点的网络地址（ASCII 格式）
- 产生记录的时间，该时间以当前活动的时区设置来表示
- 以下之一，取决于在哪里记录了该信息:
- 如果信息记录归档在那个节点的 AIX 错误日志中，则为用于进行记录的 AIX 错误日志模板标识，或

- 如果信息记录归档在 FFDC 错误堆栈中且该 FFDC 错误堆栈驻留在这个节点上，则为包含该信息记录的 FFDC 错误堆栈文件名。
- 建议使用的一个命令，该命令可用来获取与 FFDC 故障标识符相关的特定报告。

标志

-h 向标准输出设备上显示帮助消息并退出。不管是否指定了选项，不执行其他的处理。

参数

FFDC_Failure_ID

对前面的 **fcpushstk** 和 **fclogerr** 命令调用所返回的一个 FFDC 故障标识符，或是对前面的 **fc_push_stack** 子例程或 **fc_log_error** 子例程调用所返回的一个 FFDC 故障标识符。 该标识符指示创建了一个条目来报告一个故障或其他值得注意的异常事件。可以提供一个以上的 FFDC 故障标识符作为该命令的参数，但每一个标识符必须用逗号（,）分隔，中间不能有空格。

退出状态

fcdecode 在结束后返回以下整数形式的状态代码之一:

0 成功解码了 FFDC 故障标识符。

2 显示帮助信息，处理结束。

10 没有提供某个 FFDC 故障标识符作为该命令的参数。

12 给该命令提供了无效或不支持的选项。

27 没有信息写入标准输出设备。FFDC 故障标识符参数不是一个有效参数。

示例

FFDC 故障标识符由一个 base-64 数值表示，该数值按从右往左的顺序读入。每一个点表示一个前导的零。要将 FFDC 故障标识符 **.3Iv04ZVVfvp.wtY0xRXQ7.**.....解码为它的组成部件:

```
fcdecode .3Iv04ZVVfvp.wtY0xRXQ7.....................

Information for First Failure Data Capture identifier
        .3Iv04ZVVfvp.wtY0xRXQ7.....................
        Generated by the local system
        Generated Thu Sep 3 11:40:17 1998 EDT
        Recorded to the AIX Error Log using template 460bb505
        To obtain the AIX Error Log information for this entry, issue
        the following command on the local system:
        TZ=EST5EDT errpt -a -j 460bb505 -s 0903114098 | more
        Search this output for an AIX Error Log entry that contains
        the following ERROR ID code:
         .3Iv04ZVVfvp.wtY0xRXQ7.....................
```

该命令运行在另一个不同的节点上时，会有以下结果:

```
fcdecode .3Iv04ZVVfvp.wtY0xRXQ7.....................

Information for First Failure Data Capture identifier
        .3Iv04ZVVfvp.wtY0xRXQ7.....................
        Generated on a remote system with the following Internet address:
          9.114.55.125
        Generated Thu Sep 3 11:40:17 1998 EDT
        Recorded to the AIX Error Log using template 460bb505
        TZ=EST5EDT errpt -a -j 460bb505 -s 0903114098 | more
```

```
Search this output for an AIX Error Log entry that contains
the following ERROR ID code:
 .3Iv04ZVVfvp.wtY0xRXQ7...................
```

相关信息

命令: **fcdispfid**、**fcreport** 和 **fcstkrpt**

---

fcdispfid 命令

用途

在标准错误设备上显示首次故障数据捕捉故障标识符（FFDC 故障标识符）。

语法

/usr/sbin/rsct/bin/fcdispfid [**-q**]*FFDC_Failure_ID* | **-h**

描述

脚本程序使用该命令，在标准错误设备上显示一个 FFDC 故障标识符的值。之所以提供这个接口的目的，是因为脚本程序除了通过退出状态码、信号、标准输出和标准错误以外，没有一种机制将数据传递回它的客户。在这种环境中要实现将一个 FFDC 故障标识符传递回一个客户的任务，**fcdispfid** 使用 XPG/4 编目消息号 **2615-000** 在标准错误设备上显示信息。脚本程序的客户能够捕捉标准错误信息，搜索特定的消息号以及从脚本程序中获得 FFDC 故障标识符。

脚本程序必须指示该脚本产生的任何 FFDC 故障标识符将被引导到脚本用户文档中的标准错误设备中去。缺省情况下，不能以为客户知道这种行为。

标志

-h 向标准输出设备上显示帮助信息并退出。不管指定的选项，不执行其他的处理。

-q 隐藏该命令产生的警告消息。如果没有提供该命令选项，当检测到一个无效的 FFDC 故障标识符时，该命令将显示消息。

参数

FFDC_Failure_ID

指定一个 FFDC 故障标识符。 这是调用前一个 **fcpushstk** 命令或 **fclogerr** 命令所返回的一个标识符，并指示创建了一个条目来报告脚本遇到的一个故障。使用 FFDC 消息 **2615-000** 将该标识符写到标准错误设备上。

退出状态

0 显示到标准错误的 FFDC 故障标识符。

2 显示帮助信息并结束进程。

12 没有信息写到标准错误设备上。指定的选项无效。

27 没有信息写到标准错误设备上。*FFDC_Failure_ID* 参数不像是一个有效格式。

示例

通过标准输出设备将一个 FFDC 故障标识符显示给客户:

```
   FID=$(fclogerr -e FFDC_ERROR -t ERRID_SP_FFDCEXMPL_ER -i /usr/lpp/ssp/inc/
myprog.h -r myprog -s myprog.ksh -p $LINEPOS -v "1.1" -l PSSP -d $MINUSDOPTS -x
$MINUSXOPTS -y $MINUSYOPTS -b "myprog Configuration Failure - Exiting")
    RC=$?
    if ((RC == 0))
    then
         fcdispfid $FID
         return 1
    else
         :
    fi
```

相关信息

命令: **fcdecode**、**fcfilter**、**fclogerr**、**fcpushstk**、**fcreport** 和 **fcstkrpt**

子例程: *RSCT First Failure Data Capture Programming Guide and Reference* 中的 **fc_display_fid**

---

fcfilter 命令

用途

在一个文件或标准输入中发现并显示任何一个首次故障数据捕捉（FFDC）故障标识符。可能指定了多个文件。

语法

/usr/sbin/rsct/bin/fcfilter [*file_name*] [. . .]

描述

该命令扫描作为参数列出的任何文件，以获得首次故障数据捕捉（FFDC）故障标识符。如果没有提供一个文件名作为参数，该命令从标准输入中搜索 FFDC 故障标识符。如果检测到一个故障标识符，**fcfilter** 在标准输出上该行显示标识符。

脚本程序能使用 **fcfilter** 命令获取子进程通过标准错误设备返回的 FFDC 故障标识符。

如果 **fcfilter** 在输入中检测到多于一个的 FFDC 故障标识符，该命令将显示所有被发现的 FFDC 故障标识符，在每一输出行中显示一个。

标志

-h 向标准输出设备上显示帮助信息并退出。不管指定的选项，不执行其他的处理。

-q 隐藏该命令产生的警告消息。如果没有提供该命令选项，当检测到一个无效的 FFDC 故障标识符时，该命令将显示信息。

参数

file_name

用于搜索一个 FFDC 故障标识符的文件名。可能提供多于一个的文件名。如果没有给出文件名，则 **fcfilter** 从标准输入中读入。

退出状态

fcfilter 在结束时返回以下整数形式的状态代码:

0 **fcfilter** 结束执行。该退出状态并不一定表示检测到了任何的 FFDC 故障标识符。

> 0 **fcfilter** 被一个信号所中断或停止。退出状态是一个整数信号值，正是该信号停止了该命令的运行。

示例

FFDC 故障标识符由一个 base-64 数值表示，该数值按从右往左的顺序读入。每一个点表示一个前导的零。要获得 *mycmd* 命令运行所产生的所有 FFDC 故障标识符的列表:

```
 mycmd 2> /tmp/errout
 fcfilter /tmp/errout
/.00...JMr4r.p9E.xRXQ7.....................
/.00...JMr4r.pMx.xRXQ7.....................
```

要获得来自父脚本程序中一个子进程的 FFDC 故障标识符，该脚本程序可如下使用 **fcfilter** 命令:

```
RESULTS=$(mychild 2> /tmp/errout)
if (($? != 0))               # mychild ended in failure, get FFDC ID
then
    cat /tmp/errout | fcfilter | read FIRST_FFDCID
else
    rm -f /tmp/errout
fi
```

相关信息

命令: **fcdispfid**、**fclogerr**、**fcpushstk**、 **fcreport** 和 **fcstkrpt**

子例程: **fc_display_fid**, **fc_log_error**, **fc_push_stack** （请参考 *RSCT First Failure Data Capture Programming Guide and Reference*）

---

fcinit 命令

用途

建立或继承一个首次故障数据捕捉执行环境。

语法

在 Bourne 和 Korn shell 中:

/usr/sbin/rsct/bin/fcinit.sh [[**-l**] [**-s** {**c** | **i** }]] | [**-h**]

在 C shell 中:

source /usr/sbin/rsct/bin/fcinit.csh [[**-l**] [-s { **c** | **i** }]] | [**-h**]

描述

如果一个脚本程序希望使用 FFDC 接口在 AIX 错误日志、BSD 系统日志或 FFDC 错误堆栈中记录信息，则必须使用该接口。

因下面的原因，应用程序希望建立一个 FFDC 环境:

- 脚本程序希望将信息记录到 AIX 错误日志中。脚本程序能使用 **fcinit** 建立一个基本的 FFDC 环境
- 脚本程序希望其自身以及自身或自身的子程序所创建的任何派生进程能在 FFDC 错误堆栈中记录故障信息。在这种情况下，脚本程序将自己看成是一个顶层应用程序，这个顶层应用程序将创建多个底层应用程序，而顶层应用程序要成功完成，需要取决于这些底层应用程序的成功完成。当以这种方式使用 **fcinit** 命令时，则称进程*建立*或*创建*了 FFDC 错误堆栈环境。
- 仅当脚本程序被其祖先进程调用，而该祖先进程欲将故障信息或跟踪信息记录到 FFDC 错误堆栈或 FFDC 跟踪中时，脚本程序才使用 FFDC 错误堆栈或 FFDC 跟踪。在其他情况下，脚本程序并不希望使用这些设备。当在这种方式下使用 **fcinit** 时，称进程*继承* 了 FFDC 错误堆栈环境。

希望通过 FFDC 接口将信息记录到 AIX 错误日志或 BSD 系统日志中的任何进程必须建立一个 FFDC 环境。如果该进程不想使用一个 FFDC 错误堆栈，则该进程能建立一个不使用 FFDC 错误堆栈的基本 FFDC 环境。当某个进程想将来自自身或者自身所创建的线程以及它所创建的任何派生进程的故障信息记录在一个 FFDC 错误堆栈中时，该进程将建立一个包含 FFDC 错误堆栈的 FFDC 错误堆栈环境。仅当某个进程的祖先进程之一要求它在一个 FFDC 错误堆栈文件中记录故障信息时，该进程将继承一个包含 FFDC 错误堆栈的 *FFDC 错误堆栈环境*；在其他情况下，该进程将不会在 FFDC 错误堆栈中记录故障信息。

在包含一个 FFDC 错误堆栈的 FFDC 错误堆栈环境中，保存了一个 FFDC 错误堆栈文件，这样故障信息记录在 **/var/adm/ffdc/stacks** 目录中的一个文件中。这些文件的命名格式是 ***script_name.PID.date_and_time***，其中 *script_name* 是脚本程序的名称，*PID* 是脚本程序的进程标识符，*date_and_time* 是脚本程序执行时的日期和时间值。无论何时当该脚本程序或其子进程在 FFDC 错误堆栈中记录故障信息时，它将记录在该文件中。

要使一个进程在 FFDC 错误堆栈中记录信息，该进程必须使用 **fcpushstk** FFDC 接口，并且进程必须在一个已建立的 FFDC 错误堆栈环境中运行。如果不存在一个 FFDC 错误堆栈环境，或者当存在一个 FFDC 错误堆栈环境而没有使用 **fcpushstk** 接口时，则该进程不会在 FFDC 错误堆栈中记录信息。当不需要故障调试信息时，该功能可以使进程处于正常或“安静”方式，而且当在一个特定环境中调用进程进行调试时，使用该功能也可以获得信息。

fcinit 必须在 FFDC 客户进程环境（“源”）中执行，以使得该命令能够正确地为脚本程序设置 FFDC 环境。使用该命令的基于脚本 FFDC 客户机必须获取此命令，以使得 **fcinit** 在客户进程映像中执行。如果没有作到该点，FFDC 接口程序会在自身的进程映像中执行；当 FFDC 接口程序执行结束后，FFDC 环境中的任何设置将会丢失。要演示一个基于脚本应用程序如何获取 **fcinit** 命令，一个 Korn Shell 将发出以下指令:

```
. fcinit.sh <options and arguments>
```

一个 C shell 脚本将会这样:

```
source fcinit.csh <options and arguments>
```

使用 **fclogerr** FFDC 接口的进程必须建立一个 *FFDC 环境*。如果进程仅希望使用 **fclogerr** 接口，则可以建立一个没有 FFDC 错误堆栈的 *FFDC* 环境。

如果已经存在一个 FFDC 环境，而脚本程序试图创建一个时，该脚本程序将会继承已存在的 FFDC 环境而不是创建自己的一个 FFDC 环境。

标志

-h 向标准输出设备上显示帮助信息并退出。不管指定的选项，不执行其他的处理。

-l 表示进程仅希望使用 AIX 错误日志。当指定了 **-s** 选项时，则没有必要指定本选项，这是因为在 FFDC 错误堆栈环境中允许使用 AIX 错误日志。

-s 指示要建立一个 FFDC 错误堆栈环境。要使用 **fcpushstk** 接口的应用程序必须指定该标志。当命令成功结束时，会为 **/var/adm/ffdc/stacks** 目录中的脚本程序保存一个 FFDC 错误堆栈文件。这个标志必须指定为下面两个可能的选项之一:

c 要求*创建* FFDC 错误堆栈环境。如果一个祖先进程没有建立一个 FFDC 错误堆栈环境，则将建立一个。如果一个祖先进程以前建立过这样的环境，则该进程将 *继承* FFDC 错误堆栈环境，就好像指定了 **i** 选项一样。

i 如果一个祖先进程以前建立了一个 FFDC 错误堆栈环境，则指定该环境要被*继承* 。如果祖先进程以前没有建立 FFDC 错误堆栈环境，则不会为该进程建立 FFDC 错误堆栈环境，且该进程不能使用 FFDC 错误堆栈（尽管该进程可能使用 AIX 错误日志和 BSD 系统日志）。

参数

file_name

用于搜索一个 FFDC 故障标识符的文件名。可能提供多于一个的文件。如果没有给出文件名，则 **fcfilter** 从标准输入中读入。

退出状态

fcinit 在结束时返回以下退出状态代码:

0 成功建立了 FFDC 环境。

1 成功继承了 FFDC 环境。

2 显示帮助信息并结束进程。

fcinit 在检测到一个故障时，返回以下退出状态代码:

12 没有建立或继承 FFDC 环境 — 提供了一个未知的功能参数。

13 没有建立或继承 FFDC 错误堆栈环境 – 调用者指示应创建和继承 FFDC 环境。

14 在调用中没有建立 FFDC 环境 – 调用者已经有一个为其建立的 FFDC 环境 – 该例程可能已经执行了多次。

15 没有建立或继承 FFDC 错误堆栈环境 – 不存在一个 FFDC 错误堆栈环境，且指定了 FC_INHERIT 选项。

16 没有建立或继承 FFDC 环境 – 该例程不能修改客户的进程环境。

17 没有建立或继承 FFDC 环境 – FFDC 环境看来被损坏了，应被认为不可用了。

18 没有建立或继承 FFDC 环境 – 该例程不能分配所需的内存去修改客户的进程环境。

19 没有建立或继承 FFDC 错误堆栈环境 – 不能为调用进程保留 FFDC 错误堆栈文件 – FFDC 错误堆栈目录不存在或不能被使用。

21 没有建立或继承 FFDC 错误堆栈环境 – 不能为调用进程保留 FFDC 错误堆栈文件 – 该文件已经存在

42 没有建立或继承 FFDC 错误堆栈环境 – 系统管理员已经禁用了 FFDC 错误堆栈的创建和使用。脚本只能建立使用 AIX 错误日志和 BSD 系统日志的基本 FFDC 环境。

99 没有建立或继承 FFDC 环境 - 在 **fcinit** 执行中发生了一个不可预料的内部故障。该情况需要引起客户和应用程序支持服务的注意。

示例

Korn shell 脚本要建立一个基本 FFDC 环境，仅用来使用 AIX 错误日志和 BSD 系统日志（将不使用或保留 FFDC 错误堆栈）：

```
# Set up an FFDC Environment to use the AIX Error Log only. An FFDC Error
# Stack is not needed for this script.
. fcinit.sh -1
rc=$?
if ((rc != 0))
    then         print "fcinit failed with exit code of $rc"
         exit 1
fi
# Normal processing starts
```

某个 Korn shell 脚本要建立一个 FFDC 错误堆栈环境，该环境会导致脚本程序以及任何派生进程在 FFDC 错误堆栈中记录故障信息:

```
# Set up FFDC Environment to record failure information to the FFDC Error
# Stack
. fcinit.sh -sc
rc=$?
if ((rc != 0))
then    print "fcinit failed with a code of $rc"
    exit 1
fi
# Normal processing starts
```

注: FFDC 客户程序可能会收到一个指示，指出 FFDC 错误堆栈环境是被继承的而不是由 **fcinit** 调用所创建的。这种情形发生在该进程的祖先之一已经建立了一个 FFDC 错误堆栈环境的情况下。

要从进程的父进程继承一个 FFDC 错误堆栈环境:

```
# Inherit an FFDC Environment from parent process if it exists - otherwise,
# operate in a normal "silent" mode
. fcinit.sh -si
rc=$?
if ((rc != 0))
then    print "fcinit failed with a code of $rc"
    exit 1
fi
# Normal processing starts
```

相关信息

命令: **fccheck**、**fclogerr**、**fcpushstk** 和 **fcteststk**

子例程: **fc_init**（请参考 *RSCT First Failure Data Capture Programming Guide and Reference*）

fclogerr 命令

用途

将故障信息或值得注意的情况记录到 AIX 错误日志和 BSD 系统日志中。

语法

/usr/sbin/rsct/bin/fclogerr { **-e** *event* **-t** *error_template_label* **-i** *error_template_headerfile* **-r** *resource* **-s** *source_filename* **-p** *line_of_code_pos* **-v** *sidlevel* **-l** *lpp_name* **-a** *assoc_fid* { [**-d** *detail_data_item*[,*detail_data_item*,...] **-x** *detail_data_type*[,*detail_data_type*,...] **-y** *detail_data_len*[,*detail_data_len*,...]] | [**-f** *detail_data_file*] } **-b** *BSD_syslog_message_text* } | **-h**

描述

此接口将由任何一个希望将信息记录到 AIX 错误日志和 BSD 系统日志记中的脚本程序使用。写入该设备的信息，是用于系统管理员或操作员去确定系统上已发生了哪些需要关注的故障情况或其他值得注意的情况。AIX 错误日志和 BSD 系统日志的用途是记录有关某种情况的足够信息，以便从报告中确定该情况的性质、影响和响应，而不需要重新创建这种情况来检测发生了什么情况以及发生地点。如果任何软件遇到了需要直接干预才能消除的永久性故障情况，或者是遇到了需要引起系统管理员注意的情况，则应该使用 **fclogerr** 将此信息记录在 AIX 错误日志和 BSD 系统日志中。

脚本程序应在使用 **fclogerr** 之前，通过直接创建或继承，建立一个基本的 FFDC 环境或一个 FFDC 错误堆栈环境。即使没有建立这些环境，**fclogerr** 也可以将信息记录到 AIX 错误日志和 BSD 系统日志中，但接口不能生成 FFDC 故障标识符，除非这些环境中的一种环境存在。

设计用来使用 FFDC 错误堆栈的进程也可以使用 **fclogerr** 接口，而且当进程遇到需要管理员注意或干预解决的情况时，也应使用它。

为确保正确识别发生的情况以及发生的位置，FFDC 策略建议应在脚本源代码模块中内联调用 **fclogerr**，并且一旦检测到情况发生时就调用。**fclogerr** 将记录源代码文件名和代码信息行，以帮助识别和定位遇到此情况的源代码。如果必要的话，**fclogerr** 能被一个子例程或自动加载的例程所调用来记录这些信息，这是在假设外部例程可以获得所有的位置信息以及必要的详细故障信息的情况下。外部的记录例程必须要记录下检测到的情况的真实位置。

虽然 **fclogerr** 会将信息报告到 AIX 错误日志和 BSD 系统日志中，但是还必须针对每种记录设备对此接口提供不同的选项。记录到 AIX 错误日志中的详细数据信息不会同时被记录到 BSD 系统日志中；BSD 系统日志信息是通过不同的命令选项提供的。这可能要求 **fclogerr** 的用户在该调用中复制一些信息。

标志

-a 包含了软件所报告一个故障情况的 FFDC 故障标识符，该软件被导致或引起了在此时被记录的该故障情况的应用程序使用。作为软件结果指示的一部分，该标识符应该已经返还给该应用程序。调用者在此提供该标识符是为了使 FFDC 错误堆栈能够将这个时刻生成的错误报告同先前记录的错误报告联系起来。这就能够使问题调查者从该应用程序产生的各种症状中跟踪故障的原因，并使他人找出其他软件中的根本原因。如果没有其他的软件故障和这种情况有关，或是如果其他软件没有返回一个 FFDC 故障标识符作为其结果信息的一部分时，应省略该选项。

-b 说明要被写入 BSD 系统日志的文本信息。

-d 一个或多个数据项，提供有关此情况的详细信息，用于在 AIX 错误日志的条目中提供详细数据。如果信息的细节太长了，则这些详细信息会被写入一个文件中，且那个文件名会作为 *detail_data_file* 参数。

如果给出了一个详细数据文件名，则应省略该选项。如果既没有提供 *detail_data* 参数也没有提供 *detail_data_file* 参数，或者这两个参数显示无效，则将不会在 AIX 错误日志中记录详细数据的信息。

该选项可以给出一个以上的数据项。 每一个数据项必须用逗号（,）分隔，中间不能有空格。如果在一个数据项中有空格字符，则该数据项必须用双引号（″″）括起来。数据项自身不能包含逗号（,），因为该命令会将命令解释为一个域分隔符。

该选项*必须*与 **-x** 选项和 **-y** 选项一起使用。

-e 说明了 FFDC 日志事件类型。当前有效的值有 FFDC_EMERG、FFDC_ERROR、FFDC_STATE、FFDC_TRACE、FFDC_RECOV 和 FFDC_DEBUG。该代码给出了所记录的事件类型的常规性描述（紧急情况，永久性情况，通知信息，调试信息等等。）以及发生情况的严重程度。如果没有该选项，则事件记录的事件类型被赋值为 FFDC_DEBUG。

-f 某个文件名，该文件包含了所报告情况的详细信息。当详细信息太长，而不能记录在由 **fclogerr** 命令留给应用程序用来记录详细数据信息的 100 个字节的空间中时，或是当存在一个能够分析详细信息的实用程序时，使用该选项。该文件的内容被复制到 **/var/adm/ffdc/dumps** 目录中，且该文件的新位置将作为详细数据记录到 AIX 错误日志的条目中。

-h 向标准输出设备上显示帮助信息并退出。无论指定什么选项，不执行其他处理。

-i 说明了头文件（. h）的绝对路径名，该头文件包含了错误日志模板标识号，该标识号是与用 **-l** 选项说明的 *error_template_label* 相对应的。这个模板必须也存在于节点的错误日志模板库中（**/var/adm/ras/errtmplt**）。该头文件是作为软件源代码的建立过程的一部分，由 **errupdate** 命令生成的，且该头文件应被包括在 LPP 包中，LPP 包随该软件安装在节点上。如果在执行脚本程序时没有指定这个选项，或找不到头文件，则 **fclogerr** 会使用自身缺省的错误模板来记录故障信息（标号是 FFDC_DEF_TPLT_TR，标识符代码是 2B4F5CAB）。

-l 指定在其中装载了该软件的许可编程产品名称的缩写。这个值对于用户和支持应用程序的服务都必须是可识别的，并且将它作为 LPP 可接受的名称。此类值的示例有: PSSP、GPFS、LoadLeveler® 和 RSCT。如果这个选项没有提供或者似乎无效的话，就要使用 **PPS_PRODUCT** 字符串。

-p 指定了源代码模块中的代码行位置，在该行位置处报告了某种情况。这个值必须是有效的整数值。要对情况做出正确的识别和定位，该值必须尽可能接近那个产生情况报告的代码行位置。Korn Shell 能够使用 **$LINENO** 值。不能提供一个特定行计数变量的脚本语言可以提供一个符号值，开发者可以使用该符号值，在使用 **fclogerr** 的源代码中定位那个地点。如果这个选项无效或者没有提供，则将会采用 **0** 值。

-q 禁止命令中警告消息的生成。当命令必须用缺省信息来替换丢失信息时，或者命令不能将 *detail_data_file* 复制到 **/var/adm/ffdc/dumps** 目录时，就会生成警告。

-r 指定软件组件名。这是一个产生报告的软件的符号名，该符号名应被客户和应用程序支持服务所识别。字符串的长度限制为 16 个字符。

-s 指定包含遇到报告情况的代码行的源文件名。对于 Korn 和 Borne shell 脚本来说，该选项的参数应设置为 **$0**; C shell 脚本应将这个参数值设为 **${0}**。如果没有提供该选项或是一个无效选项，则使用字符串 **unknown_file**。

-t 指示了一个符号标号，该标号被赋给了错误日志库中的 AIX 错误日志模板。建立错误日志模板的 **errupdate** 命令创建了一个宏，该宏将这个标号映射到一个整数代码。这个标号以字符 **ERRID_** 打头，最大长度为 19 个字符。当执行脚本程序时，如果没有指定该选项或没有找到头文件，则 **fclogerr** 将使用 OPMSG 模板，调用 **errlogger** 在 AIX 错误日志中创建一条消息。

-v 指示检测出所记录情况的源代码模块的 SCCS 版本号。对于在 SCCS 控制下建立的源代码，版本号将被设置为 ″**1.1**″（必须有双引号）。如果没有提供该选项或是一个无效选项，则使用字符串 **unknown**。

-x 当将此信息记录到 AIX 错误日志中时，指示将如何解释 **-d** 选项所指定的这些数据项。这些类型必须

与 **-t** 选项中指定的 AIX 错误日志模板的相应字段一致。每一种类型都指示了 **-d** 列表中相应数据项是如何被解释的。该选项可接受的值有 ALPHA、HEX 和 DEC。对于 **-d** 列表中的每一个参数，必须有一个在 **-x** 参数中列出的相匹配的类型。

如果给出了 **-d** 选项，则*必须*给出此选项。

-y 指示了由 **-d** 选项所指定的数据项长度（按字节数）。这些长度值必须与 **-t** 选项中指定的 AIX 错误日志模板的相应字段一致。对于 **-d** 列表中的每一个参数，必须有一个在 **-y** 参数中相匹配的类型。

如果给出了 **-d** 选项，则*必须*给出此选项。

参数

file_name

用于搜索一个 FFDC 故障标识符的文件名。可能提供多于一个的文件名。如果没有给出文件名，则 **fcfilter** 从标准输入中读入。

退出状态

fclogerr 在成功结束时，返回以下退出状态码:

0 已成功排队的信息将被写入 AIX 错误日志和 BSD 系统日志。这个记录的 FFDC 故障标识符显示在标准输出中。调用者应该捕获标准输出以获得该值。

2 显示帮助信息并结束进程。

12 没有将任何信息记录到 AIX 错误日志中，且该命令没有提供 FFDC 故障标识符。该命令的使用者给出了命令的一个无效选项。

在 AIX 之外的其他 UNIX 平台上，当发生故障时，**fclogerr** 返回以下退出状态代码:

38 BSD 系统日志不能为该事件产生一个记录。系统日志正遇到了某种故障。在 AIX 系统上，报告被记录到 AIX 错误日志中；在其他系统中，将认为此情况是一个故障。

当给 **fclogerr** 提供了不完整的信息时，它将用缺省信息去代替缺少的信息，并试图在 FFDC错误堆栈上产生一个记录。在这些情况下，将产生警告，并且生成警告信息，除非指定了 **-q** 选项。当检测到多于一个的警告情况时，该命令将返回一个退出状态码，该代码表示了该命令所认为的最严重情况。当检测到警告情况时，**fclogerr** 将返回以下退出状态代码:

10 命令的使用者没有成功为该命令提供一个 **-i** 选项，或不能定位作为 -i 选项参数指定的头文件。在这种情况下，该命令将使用“首次故障数据捕捉”缺省模板（标号是 FFDC_DEF_TPLT_TR，标识符代码是 2B4F5CAB），将一般性信息记录到 AIX 错误日志中。

26 向该例程提供一个有详细信息的数据字符串和一个详细的数据文件。例程选择了详细的数据字符串，并且忽略了有详细信息的数据文件。

28 没有提供检测到这个事件的资源名。使用缺省的资源名 **ffdc** 来替换缺少的资源名。

29 检测到的应用程序信息 — 源代码文件名、源代码文件版本、LPP 名称和代码位置行 — 没有提供。这些丢失信息被缺省信息所代替。

32 在 *detail_data_file* 参数中指定的文件不能被复制到 **/var/adm/ffdc/dumps** 目录中。FFDC 错误堆栈条目引用了该文件原来的版本。请勿废弃该文件原有的副本。

33 没有指定 **-e** 选项，或没有指定一个有效的 FFDC 事件类型。事件类型 FFDC_DEBUG 被赋值给了这个事件记录。

34 在 *format* 参数中没有给出消息。因此，在 BSD 系统日志中记录了此事件的一条一般性消息。

35 没有提供关于该事件的详细信息。如果没有这些详细的信息，则为了说明这次事件的细节，事后做问题分析时就可能会变得很困难。

36 详细数据串的长度超出了 AIX 错误日志条目的限制容量。将截短该详细数据以适应空间的大小。在信息的截短过程中，可能会丢失事件的一些信息。

37 不能为该例程所建立的报告去构造一个 FFDC 错误标识符。FFDC 故障标识符不会写至标准输出，但是该事件的相关信息被记录到 AIX 错误日志和 BSD 系统日志中。

38 在 BSD 系统日志中不能产生该事件的一个记录。系统日志可能处于禁用，或遇到了故障。在 AIX 系统上，报告被记录到 AIX 错误日志中；在其他系统中，将认为此情况是一个故障。

示例

例如，一个 Korn shell 脚本试图访问某个文件中的配置信息。如果访问没有成功，则脚本代码将使用以下模板源代码，将故障记录到 AIX 错误日志中:

```
*! mymesgcat.cat
+ SP_FFDCEXMPL_ER:
    + SP_FFDCEXMPL_ER:
    Comment        = "Configuration Failed - Exiting"
    Class          = S
    Log       = true
    Report         = true
    Alert          = false
    Err_Type  = PERM
    Err_Desc  = {3, 10, "CONFIGURATION FAILURE - EXITING"}
    Prob_Causes    = E89B
    User_Causes    = E811
    User_Actions   = 1056
    Fail_Causes    = E906, E915, F072, 108E
    Fail_Actions   = {5, 14, "VERIFY USER HAS CORRECT PERMISSIONS TO ACCESS FILE"},
              {5, 15, "VERIFY CONFIGURATION FILE"}
    Detail_Data    = 46, 00A2, ALPHA
    Detail_Data    = 42, EB2B, ALPHA
    Detail_Data    = 42, 0030, ALPHA
    Detail_Data    = 16, EB00, ALPHA
    Detail_Data    = 16, 0027, ALPHA
    Detail_Data    = 4, 8183, DEC
    Detail_Data    = 4, 8015, DEC
    Detail_Data    = 60, 8172, ALPHA
```

该定义产生了以下 AIX 错误日志模板:

```
LABEL:            ERRID_SP_FFDCEXMPL_ER
IDENTIFIER:        <calculated by errupdate during source code build>

Date/Time:         <filled in by AIX Error Log subsystem>
Sequence Number:   <filled in by AIX Error Log subsystem>
Machine Id:        <filled in by AIX Error Log subsystem>
Node Id:           <filled in by AIX Error Log subsystem>
Class:             S
Type:              PERM
Resource Name:     Resource Name:     <filled in by -r option to fclogerr>

Description
CONFIGURATION FAILURE - EXITING

Probable Causes
COULD NOT ACCESS CONFIGURATION FILE

User Causes
```

```
USER CORRUPTED THE CONFIGRATION DATABASE OR METHOD

    Recommended Actions
    RE-CREATE FILE

Failure Causes
COULD NOT ACCESS CONFIGURATION FILE
PERMISSIONS ERROR ACCESSING CONFIGURATION DATABASE
FILE READ ERROR
FILE IS CORRUPT

    Recommended Actions
    VERIFY USER HAS CORRECT PERMISSIONS TO ACCESS FILE
    VERIFY CONFIGURATION FILE

Detail Data
DETECTING MODULE
<filled in by fclogerr options>
ERROR ID
<The FFDC Failure Identifier created by fclogerr>
REFERENCE CODE
<The -a option value to fclogerr>
FILE NAME
<Must be supplied as part of -d option list to fclogerr>
FUNCTION
<Must be supplied as part of -d option list to fclogerr>
RETURN CODE<Must be supplied as part of -d option list to fclogerr>
ERROR CODE AS DEFINED IN sys/errno.h
<Must be supplied as part of -d option list to fclogerr>
USER ID<Must be supplied as part of -d option list to fclogerr>
```

前三个“细节数据字段”由 **fclogerr** 例程根据参数中传递的信息来构造。其余的“详细数据”必须用 **-d** 选项提供，且提供的数据类型必须由 **-x** 选项指出。下面的示例源代码段演示了这是如何实现的，以及如何调用 **fclogerr** 命令将信息记录到 AIX 错误日志和 BSD 系统日志中。

```
typeset CONFIG_FNAME
typeset INBUF
typeset MINUSDOPTS
typeset MINUSXOPTS
typeset MINUSYOPTS
typeset FID
integer MYCLIENT
integer RC
    :
MYCLIENT=$$
CONFIG_FNAME="/configfile.bin"
exec 3< $CONFIG_FNAME
    :
read -u3 INBUF
RC=$?
if ((RC != 0))
then
    # Create Detail Data Memory Block for AIX Error Log Template
    # Need to know the EXACT structure of the Template to do this correctly.
    #    Field 1 - filled in by fc_log_error
    #    Field 2 - filled in by fc_log_error
    #    Field 3 - filled in by fc_log_error
    #    Field 4 - name of configuration file being used - 16 bytes
    #    Field 5 - name of function call that failed - 16 bytes
    #    Field 6 - return code from failing function - 4 byte integer
    #    Field 7 - errno from failing function call (unused) - 4 byte integer
    #    Field 8 - user ID using this software - remaining space (62 bytes)
    # This source code supplied fields 4 through 8 in the "-d" option, and
    # describes the data types for each in the "-x" option.
    MINUSDOPTS=$CONFIG_FNAME
```

```
    MINUSXOPTS="ALPHA"
    MINUSYOPTS="16"
    MINUSDOPTS="$MINUSDOPTS,read"
    MINUSDOPTS="$MINUSDOPTS,configdabeast"
    MINUSXOPTS="$MINUSXOPTS,ALPHA"
    MINUSYOPTS="$MINUSYOPTS,16"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,$MYCLIENT"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,0"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,$MYCLIENT"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSYOPTS="$MINUSYOPTS,60"
    FID=$(fclogerr -e FFDC_ERROR -t ERRID_SP_FFDCEXMPL_ER -i /usr/lpp/ssp/inc/
myprog.h -r myprog -s myprog.ksh -p $LINEPOS -v "1.1" -l PSSP -d $MINUSDOPTS -x
$MINUSXOPTS -y $MINUSYOPTS -b "myprog Configuration Failure - Exiting")
    RC=$?
    if ((RC == 0))
    then
         fcdispfid $FID
         return 1
    else
         :
    fi
fi
```

现在使用相同的 AIX 错误日志模板，考虑一下关于上面示例的一点变化，不过这次将使用一个外部命令来获取由此源代码提供的某个文件中的配置信息。该命令如果遇到了任何故障情况，则以非零的退出状态值退出，并在标准输出上显示一个 FFDC 故障标识符。另外，要演示在 **-d** 列表中使用双引号的情况，配置文件在名字中要有一个嵌入的空格:

```
typeset CONFIG_FNAME
typeset INBUF
typeset MINUSDOPTS
typeset MINUSXOPTS
typeset MINUSYOPTS
typeset FID
typeset OUTPUT
integer MYCLIENT
integer RC
    :
MYCLIENT=$$
CONFIG_FNAME="This is a test"
OUTPUT=$(configdabeast $CONFIG_FNAME)
RC=$?
if ((RC != 0))
then
    # Create Detail Data Memory Block for AIX Error Log Template
    # Need to know the EXACT structure of the Template to do this correctly.
    #    Field 1 - filled in by fc_log_error
    #    Field 2 - filled in by fc_log_error
    #    Field 3 - filled in by fc_log_error
    #    Field 4 - name of configuration file being used - 16 bytes
    #    Field 5 - name of function call that failed - 16 bytes
    #    Field 6 - return code from failing function - 4 byte integer
    #    Field 7 - errno from failing function call (unused) - 4 byte integer
    #    Field 8 - user ID using this software - remaining space (62 bytes)
    # This source code supplied fields 4 through 8 in the "-d" option, and
    # describes the data types for each in the "-x" option.
```

```
    MINUSDOPTS="\""$CONFIG_FNAME"\""
    MINUSXOPTS="ALPHA"
    MINUSYOPTS="16"
    MINUSYOPTS="16"
    MINUSDOPTS="$MINUSDOPTS,configdabeast"
    MINUSXOPTS="$MINUSXOPTS,ALPHA"
    MINUSYOPTS="$MINUSYOPTS,16"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,$MYCLIENT"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,0"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSDOPTS="$MINUSDOPTS,$MYCLIENT"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSXOPTS="$MINUSXOPTS,DEC"
    MINUSYOPTS="$MINUSYOPTS,4"
    MINUSYOPTS="$MINUSYOPTS,60"
    FID=$(fclogerr -e FFDC_ERROR -t ERRID_SP_FFDCEXMPL_ER -i /usr/lpp/ssp/inc/
myprog.h -r myprog -s myprog.ksh -p $LINEPOS -v "1.1" -l PSSP -d $MINUSDOPTS -x
$MINUSXOPTS -y $MINUSYOPTS -a $OUTPUT -b "myprog Configuration Failure - Exiting")
    RC=$?
    if ((RC == 0))
    then
         fcdispfid $FID
         return 1
    else
         :
    fi
fi
```

相关信息

命令: **errpt**、**fcdecode**、**fcdispfid**、**fcinit**、**fcpushstk**、**fcreport**

文件: **ct_ffdc.h**

子例程: **fc_log_error**（请参考 *RSCT First Failure Data Capture Programming Guide and Reference*）

---

fcpushstk 命令

用途

向首次故障数据捕获错误堆栈（First Failure Data Capture Error Stack）记录故障信息或者值得注意的情况。

语法

/usr/sbin/rsct/bin/fcpushstk { [**-a** *assoc_fid*] **-c** *message_catalog_name* **-m** *message_set* **-n** *message_number* [**-o** *message_param* [,*message_param*,...]] **-l** *lpp_name* **-p** *line_of_code_pos* **-r** *resource* **-s** *source_filename* **-v** *sidlevel* {[**-d** *detail_data*] | [**-f** *detail_data_file*]} } *default_message* | **-h**

描述

fcpushstk 是脚本用来向 FFDC 错误堆栈中记录故障信息的命令。脚本会向 FFDC 错误堆栈记录描述性信息和调试数据，以便在今后确定问题时使用。

FFDC 错误堆栈是用来帮助理解故障情况的，这些情况是当多个相关的进程或线程在同一个节点执行共同任务时产生的。该设备最好被应用于那些创建一个或者多个线程或者子进程（它们接着自己创建线程或者子进程）

的应用程序。为了使用 FFDC 错误堆栈脚本通过 **fcinit** 接口建立了 *FFDC 错误堆栈环境*。这个环境建立之后，该应用程序或者任意一个它的派生程序就可以使用 FFDC 错误堆栈了。

并非所有的软件应用程序都会建立 FFDC 错误堆栈环境。但是，这些应用程序也许被其他已经建立 FFDC 错误堆栈环境的应用程序或者脚本调用。在这些情况下，调用这个软件的脚本或者应用程序也许希望从这个软件中捕获错误信息，并将这些错误信息同调用其他软件时获取的错误信息一起来分析，从而发现这些故障中的联系或者模式。出于这个原因，那些在正常操作条件下不使用 FFDC 错误堆栈的软件至少应该当任何客户机调用这个软件需要使用 FFDC 错误堆栈时支持其使用。通过 **fcint** 接口，这可以通过从父进程那里*继承* FFDC 错误堆栈环境来实现。

fcpushstk 记录了对于 FFDC 错误堆栈值得注意的情况的描述和详细信息。如果脚本没有通过创建或者继承来建立 *FFDC 错误堆栈环境*的话，**fcpushstk** 不会记录任何信息，并将控制返还给调用者。此项操作允许脚本当没有要求调试信息时在通常的“安静”方式下运行，但是也允许当要求调试信息的时候脚本支持使用 FFDC 错误堆栈。

当 FFDC 错误堆栈环境建立之后，脚本必须显式调用 **fcpushstk**，以便记录信息到 FFDC 错误堆栈。仅仅建立这个环境还不足以记录故障数据。**fclogerr** 命令还不能向 FFDC 错误堆栈写入任何记录。

要确保正确识别它所发生的情况和位置，当情况一被发现，**fcpushstk** 就应该立即在脚本源代码模块中被内嵌调用。**fcpushstk** 将会记录源代码文件名以及代码信息行，以便帮助识别遇到这个情况的源代码并对其进行定位。要记录这种信息，如果必要，子例程或者自动加载的例程可以调用 **fcpushstk**，只要所有的位置信息以及必要的故障细节信息对于这些外部例程来说可用。外部的记录例程必须记录下那些被检测到的故障发生的真实位置。

FFDC 错误堆栈记录的最大大小在 **<rsct/ct_ffdc.h>** 头文件中通过 FC_STACK_MAX 定义给出。FC_STACK_MAX 用字节定义了这个长度。这个值应该仅仅被当作大致的指导，因为这个长度包括将要被 **fcpushstk** 用来记录检查文件信息、描述信息和 FFDC 错误标识符信息的数据。任何超过 FC_STACK_MAX 字节的记录都将会被截断以适应 FC_STACK_MAX 的限定。

标志

-a 指定造成或者影响此时所记录的情况的、该应用程序所使用软件报告的故障情况的 FFDC 故障标识符。该标识符应已经返回给应用程序作为软件结果指示的一部分。调用程序在这里给出这个标识符，以便 FFDC 错误堆栈能够将此时所产生的故障报告与原来所记录的故障报告联系起来。这就能够使问题调查者从该应用程序产生的各种症状中跟踪故障的原因，并使他人找出其他软件中的根本原因。如果没有其他软件错误对这个情况负责，或者其他软件没有返回一个作为其结果信息一部分的 FFDC 故障标识符的话，则就不应该提供 **-a** 这个选项。

-c 指示包含对所记录故障的描述的、符合 XPG/4 的消息编目名称。这个名称同 **/usr/lib/nls/msg/$LANG** 目录有关。如果不能找到这个消息编目，则 *default_message* 将会显示来描述这个故障。注：那个 *default_message* 将不会在语言环境之间被转换。

-d 一个字符串，该字符串提供有关类似于 AIX 错误日志所用“详细数据”概念的情况的详细信息。如果这条信息的细节过长，这些详细信息将会被写入文件，而且那个文件名将会作为一个 **-f** 选项的参数被提供。不能同时指定 **-d** 和 **-f** 选项。如果既没有提供 **-d** 选项，也没有提供 **-f** 选项，或者这两个选项显示无效，则记录 **no detail data** 字符串。

-f 指定包含关于所报告情况详细信息的文件的名称，该情况类似于 AIX 错误日志所用的“详细数据”概念。这个选项被用于当详细信息过长而不能被 FFDC 错误堆栈本身记录时，或者当可以分析详细信息的实用程序存在的时候。文件的内容被复制到 **/var/adm/ffdc/dumps** 目录，而这个文件的新位置则作为在 FFDC 错误堆栈中的详细数据进行记录。如果包含这些情况细节的文件不存在，就不要指定这个选项。不能同时指定 **-d** 和 **-f** 选项。

-h 向标准输出设备上显示帮助信息并退出。不管指定的选项，不执行其他的处理。

-l 指定在其中装载了该软件的许可编程产品名称的缩写。这个值对于用户和支持应用程序的服务都必须是可识别的，并且将它作为 LPP 可接受的名称。此类值的示例有：PSSP、GPFS、LoadLeveler 和 RSCT。如果没有给出这个选项或似乎是一个无效的选项，则使用字符串 **PPS_PRODUCT**。

-m 指定在消息目录文件中包含故障描述消息的消息集。如果不能对这个消息集进行定位，则将会显示 *default_message* 来描述故障。注：**default_message** 不会被转换成用户的语言环境。

-n 指定描述记录的故障的消息号。如果不能对该消息进行定位，则将会显示 *default_message* 来描述故障。注：*default_message* 不会被转换成用户的语言环境。

-o 在 **-n** 选项指出的消息中，指定一个替换参数的列表。由于外壳操作环境的原因，**fcpushstk** 仅仅支持字符串作为替换参数（%s）。如果提供了多个替换参数，则每一个之间必须用一个逗号（,）隔开。如果任何一个替换参数包含嵌入的空白符，它们就必须用双引号（ “” ）圈起。

-q 禁止命令中警告消息的生成。当该命令必须使用缺省信息去代替缺少的信息，或是当该命令不能将 *detail_data_file* 复制到 **/var/adm/ffdc/dumps** 目录中的时候，生成警告消息。

-r 指定软件组件名。对于软件产生报告来说，这是一个符号化名称，对于用户和支持应用程序的服务都应该是可识别的。

-p 指定在源代码模块中产生情况报告的代码行位置。这个值必须是一个有效的整数值。要对情况作出正确的识别和定位，这个值应尽可能接近检测到发生情况的代码行位置。Korn shell 脚本可以使用 **$LINENO** 值。不提供特殊行计数变量的脚本语言在此可以向开发者提供符号值，开发者可以通过它在源代码中对使用 **fcpushstk** 的点进行定位。如果没有提供这个选项或是一个无效选项，则使用 **0** 值。

-s 指定包含遇到报告情况的代码行的源文件名。对于 Korn 和 Borne Shell 脚本，这个选项的参数应该设置为 **${0}**；C Shell 脚本对于这个参数要设置为 ${0}。如果这个选项没有提供或者无效的话，就要使用 **unknown_file** 字符串。

-v 指示检测出所记录情况的源代码模块的 SCCS 版本号。对于在 SCCS 控制下的源代码，这个值应设置为“**1.1**”（需要加双引号）。如果这个选项没有提供或者无效的话，就要使用 **unknown** 字符串。

参数

default_message

当信息不能从由 **-c** 、**-m** 和 **-n** 选项提供的消息目录信息中检索时，指示将用作故障描述的缺省消息。如果这个字符串中包含位置参数的话，则所有的位置参数都必须被指定成一个字符串（%s）。如果含有内嵌的空白时，这个消息需要用双引号（ “” ）括起。**fcpushstk** 限定字符串的全长度为 72 个字符。

退出状态

在成功完成后，**fcpushstk** 会返回以下退出状态码:

0 FFDC 错误堆栈环境存在，而且故障信息成功地记录在了 FFCD 错误堆栈中。该记录的 FFDC 故障标识符显示在标准输出上。调用者应该捕获标准输出以获得该值。

2 显示帮助信息并结束进程。

当故障产生时，**fcpushstk** 会返回以下退出状态码:

11 没有信息记录到 FFDC 错误堆栈中，并且该命令没有提供 FFDC 故障标识符。用户请求使用在 FFDC 软件发行版中不支持的选项。

12 没有信息记录到 FFDC 错误堆栈中，并且该命令没有提供 FFDC 故障标识符。给接口提供了未知的函数参数。

15 FFDC 错误堆栈环境不存在。没有信息记录在 FFDC 错误堆栈中。命令没有产生 FFDC 故障标识符。当 FFDC 错误堆栈环境不存在以通过 **fcinit** 被继承时，对于 FFDC 用户来说，这是一个很平常的返回码。

17 没有信息记录到 FFDC 错误堆栈中，并且该命令没有提供 FFDC 故障标识符。FFDC 错误堆栈环境可能被破坏，应该看作不能够使用。

19 没有信息记录到 FFDC 错误堆栈中 – FFDC 错误堆栈目录不存在或者不能够使用。该命令没有提供 FFDC 故障标识符。

20 没有信息记录到 FFDC 错误堆栈中，并且该命令没有提供 FFDC 故障标识符。不能够访问 FFDC 错误堆栈文件。文件可能被除去，或者为了禁止访问 FFDC 错误堆栈，该文件的或者它的目录的许可已经更改。

22 没有信息记录到 FFDC 错误堆栈中 – FFDC 错误堆栈文件不能锁定为供该接口专用。为了锁定该文件已经进行了反复尝试，但是所有的尝试都失败了。另外的一个进程也许已经锁定了该文件并且没能释放它，或者其他的进程也许已被挂起并且正阻止其他进程使用 FFDC 错误堆栈。该命令没有提供 FFDC 故障标识符。

24 没有信息记录到 FFDC 错误堆栈中，并且该命令没有提供 FFDC 故障标识符。FFDC 错误堆栈文件可能已被破坏。用户应当认为 FFDC 错误堆栈环境不可使用。

25 没有信息记录到 FFDC 错误堆栈中，并且该命令没有提供 FFDC 故障标识符。FFDC 错误堆栈文件的名称设置为了一个目录的名称。应当认为 FFDC 错误堆栈环境已被破坏而且不能使用。

32 一个转储文件不能被复制到 **/var/adm/ffdc/dumps** 目录。在包含 **/var/adm/ffdc** 目录的文件系统中没有足够的空间。应该使用 **fcclear** 命令来清除不需要的 FFDC 错误堆栈和转储文件，或者系统管理员需要向文件系统中加入更多的空间。该命令没有提供 FFDC 故障标识符。

40 没有信息记录到 FFDC 错误堆栈中 – 信息不能记录到 FFDC 错误堆栈中。在包含 **/var/adm/ffdc** 目录的文件系统中没有足够的空间。应该使用 **fcclear** 命令来清除不需要的 FFDC 错误堆栈和转储文件，或者系统管理员需要向文件系统中加入更多的空间。该命令没有提供 FFDC 故障标识符。

41 没有信息记录到 FFDC 错误堆栈中，并且该命令没有提供 FFDC 故障标识符。当从 FFDC 错误堆栈中读取控制信息或者向 FFDC 错误堆栈写入事件信息时，产生了一个故障。用户应该推断出该事件的条目没有被记录。

99 没有信息记录到 FFDC 错误堆栈中，并且该命令没有提供 FFDC 故障标识符。在 **fc_push_stack** 例程中产生了一个意外的内部故障。这个问题需要引起支持应用程序的服务的注意。

当 **fcpushstk** 该命令是同不完全的信息一同被提供时，它将用缺省信息代替丢失信息，并且会试图在 FFDC 错误堆栈中留下一个记录。在这些情况下会产生警告，而且除非指定了 **-q** 选项，这些警告消息会显示到标准错误设备上。如果在一个地方检测到不止一个警告情况，则该命令就会产生对应于检测到的最严重警告情况的退出状态码。当检测到警告情况时，**fcpushstk** 会返回以下退出状态码:

26 给该例程提供了一个详细数据串和一个详细数据文件。该例程选择详细数据串并忽略详细数据文件。

28 没有给出检测到事件的资源名。丢失资源名被缺省资源名替换。

29 检测应用程序的信息中至少有一项没有给出，这些项是源代码文件名、源代码文件版本、LPP 名、代码行位置。使用缺省信息去代替缺少的信息。

30 没有提供用来描述这个事件性质的缺省消息。如果不能定位包含描述消息的 XPG/4 消息目录的话，则通过 **fcstkrpt** 命令将不会显示对该情况的任何描述。

31 没有消息用来描述该事件的性质，或者没有提供 XPG/4 信息 — 目录文件名、消息集号、消息号。**fcstkrpt** 命令没有显示对该情况的任何描述。

32 在 *detail_data_file* 参数中指定的文件不能被复制到 **/var/adm/ffdc/dumps** 目录中。FFDC 错误堆栈条目引用了该文件的原始版本。请勿删除该文件的原始拷贝。

35 没有给出该事件的详细信息。没有这些信息，事后故障分析将很难进行，以说明该事件的详细情况。

37 不能为例程创建的报告建立一个 FFDC 故障标识符。该命令没有提供 FFDC 故障标识符，但是这次事件的信息已经记录到了 FFDC 错误堆栈中。

44 对该命令提供的信息将已经导致一个超出 FC_STACK_MAX 限定的 FFDC 错误堆栈记录。该记录会被截断以适应系统的限制，以便它能够记录。在这个截取的过程中，关于这个故障的重要信息也许会丢失。可修改脚本来提供更少的信息，或者将信息记录到一个详细信息数据文件中，同时向该命令提交那个详细信息数据文件的名称。

示例

当 FFDC 环境被进程建立或者继承时，向 FFDC 错误堆栈记录故障信息:

```
#!/bin/ksh
    :
    :
cp /tmp/workfile $FILENAME
RC=$?
if ((RC != 0))
then    FFDCID=$(fcpushstk -c mymsg.cat -m2 -n10 -o$FILENAME -r myprog
         -d"cp exit status $RC - file being copied /tmp/workfile" -s$0
         -p$LINENO -v"1.1" -lPSSP "Cannot update configuration file %1$s")
    if (($? == 0))
    then         fcdispfid $FFDCID
         return 1
    fi
fi
    :
    :
```

从没有代码行变量可用的脚本语言中进行同样的记录:

```
#!/bin/bsh
    :
    :
CODESCTN=14             # Used to identify where in the script code we are
cp /tmp/workfile $FILENAME
RC=$?
if test $RC -ne 0
then    FFDCID=`fcpushstk -c mymsg.cat -m2 -n10 -o$FILENAME -r myprog
         -d"cp exit status $RC - file being copied /tmp/workfile" -s$0
         -p$CODESCTN -v"1.1" -lPSSP "Cannot update configuration file %1$s"`
    if test $? -eq 0
    then         fcdispfid $FFDCID
         return 1
    fi
fi
CODESECTION=15              # New code section begins - a different task starts
    :
    :
```

记录同另外一个先前被采用 FFDC 的应用程序记录到 FFDC 错误堆栈中的故障情况相关的故障情况信息:

```
#!/bin/ksh
    :
    :
```

```
ASSOC_FID=$(/usr/lpp/ssp/bin/somecmd -a -b)
RC=$?if ((RC != 0))
then    FFDCID=$(fcpushstk -a$ASSOC_FID -c mymsg.cat -m2 -n10 -o$FILENAME -r myprog
         -d"cp exit status $RC - file being copied /tmp/workfile" -s$0
         -p$LINENO -v"1.1" -lPSSP "Cannot update configuration file %1$s")
    if (($? == 0))
    then         fcdispfid $FFDCID
         return 1
    fi
fi
    :
    :
```

相关信息

命令: **fcdecode**、**fcdispfid**、**fcinit**、**fcreport**、**fcstkrpt**、**fcteststk**

子例程: **fc_push_stack**（请参考 *RSCT First Failure Data Capture Programming Guide and Reference*）

---

fcreport 命令

用途

定位并显示故障和与该故障相关的任何故障的报告。

语法

/usr/sbin/rsct/bin/fcreport { [**-a**] *FFDC_Failure_ID* } | **-h**

描述

fcreport 对 FFDC 故障标识符进行解码，并且获取由它指定的故障报告。该命令还会检查是否存在故障同这个 FFDC 故障标识符相关联，如果是这样的话，就会获取这个故障的报告。该命令会继续检查它为相关故障定位的每一个故障的报告并且继续获取关于相关故障的报告，直到遇到以下情况中的一个:

- 没有检测到更多的相关故障。
- 找不到相关故障的报告。这种情况会在当相关的故障报告在远程节点上驻留，而在这个时候不能到达远程节点或者故障的记录已经从它驻留的节点那里除去了的时候产生。

利用该命令，用户可以获取导致特定故障的整个故障列表报告。如果故障是由提供给命令的初始故障引起的话，**fcreport** 不能对故障做出定位报告；它只能获取引起故障的故障报告。

标志

-a 显示包含在故障报告中的所有信息。缺省显示节点的网络地址（在此节点产生了故障报告）、故障报告的时间戳记和在故障报告中记录的事件描述。

-h 向标准输出设备上显示帮助信息并退出。不管指定的选项，不执行其他的处理。

参数

FFDC_Failure_ID

指定故障的 FFDC 故障标识符来开始报告。**fcreport** 将会试图获得该故障的故障信息，以及这个报告作为相关故障列出的任何故障。只有一个 FFDC 故障标识可以提供给这个命令。

安全性

fcreport 使用 **rsh** 来获得也许驻留在远程节点上的故障报告。用户必须有足够的权限来执行这些用于远程节点的 **rsh** 命令。如果用户没有这个许可，只要它们在本地节点存在，**fcreport** 就只能追踪到相关故障的列表。

退出状态

fcreport 一结束就会产生以下退出状态码中的一种:

0 为 FFDC 故障标识符提供的定位和显示的故障报告。零或者更多相关的故障报告可能已经被定位和显示了。

2 显示帮助信息并结束进程。

10 没有提供需求的选项或者参数。

11 一个后来的 FFDC 软件发行版产生了提供给该命令的 FFDC 故障标识符。命令不能正确解释这个标识符。

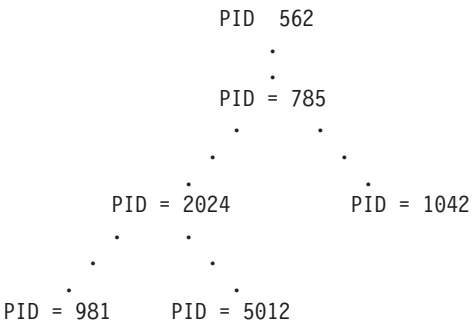
12 向该命令指定了未知的选项。

20 FFDC 故障标识符指的是系统上 “FFDC 错误堆栈” 中的条目，但是 “FFDC 错误堆栈” 文件不能被访问。文件可能已经除去，或者为了防止访问这个文件这个文件的许可可能已经被更改。

27 向此命令提供的 FFDC 故障标识符不是合法的标识符。

示例

考虑下面这种情况，几个进程按下面这种父子顺序被创建:



在这个示例中，进程 785 产生了 FFDC 故障标识符 **.3Iv04ZVVfvp.wtY0xRXQ7.....** 同时将它传递回进程 562。为了获得 FFDC 故障标识符 **.3Iv04ZVVfvp.wtY0xRXQ7.....** 的详细报告 和导致这个特定故障的先前的故障:

```
$ fcreport -a .3Iv04ZVVfvp.wtY0xRXQ7.....................
```

这个报告将会包括指定的 FFDC 故障标识符详细的信息，也包括可能引起它的进程 2024、1042、981 和 5012 中的故障的报告。报告将不会包括进程 562 中的故障，因为这些故障也许是进程 785 故障的结果。

相关信息

命令: **fcclear**、**fcdecode**、**fcdispfid**、**fcfilter**、**fclogerr**、**fcpushstk**、**fcstkrpt**

子例程: **fc_log_error**、**fc_push_stack**（参考 *RSCT First Failure Data Capture Programming Guide and Reference*）

fcstat 命令

用途

显示指定的光纤通道设备驱动程序收集的统计信息。

语法

fcstat *Device_Name*

描述

fcstat 命令用于显示由指定的光纤通道设备驱动程序收集的统计信息。它使用以下过程收集统计信息:

1. 打开 **fcstat** 的消息目录并检查参数列表。
2. 访问 ODM 数据库以获得有关所选适配器的信息。
3. 访问 ODM 数据库以获得有关所选适配器的端口的信息。
4. 打开并访问适配器统计信息。
5. 报告统计信息并退出。

如果指定了无效 *Device_Name*，则 **fcstat** 命令将返回一条错误消息，声明它无法在 ODM 数据库中找到该设备。如果指定的 *Device_Name* 未连接到网络（即: 链路已断），**fcstat** 命令也会报告错误。当 **fcstat** 命令无法从指定的 *Device_Name* 抽取统计信息时，它仍会报告从 ODM 数据库中抽取的信息。

参数

Device_Name 光纤通道设备的名称。例如，`fcs0`。

统计信息字段

注: 某些适配器可能不支持特定统计信息。不受支持的统计信息字段的值总是为 0。

在 **fcstat** 命令的输出中显示的统计信息字段及其描述如下:

`Device Type`	显示适配器的描述。
`Serial Number`	显示适配器中的序列号。
`Option ROM Version`	显示适配器上的选件 ROM 的版本。
`Firmware Version`	显示适配器上的固件的版本。
`Node WWN`	显示适配器的全球名称。
`Port FC ID`	显示适配器的 SCSI 标识。
`Port Type`	显示适配器的连接类型。
`Port Speed`	显示适配器的速度。
`Port WWN`	显示端口的全球名称。
`Seconds Since Last Reset`	显示自上次复位适配器上的统计信息以来经过的秒数。
`Frames`	显示传输和接收的帧数。
`Words`	显示传输和接收的字数。
`LIP Count`	显示 LIP 计数。
`NOS Count`	显示 NOS 计数。
`Error Frames`	显示出错帧的数量。
`Dumped Frames`	显示转储的帧数。
`Link Failure Count`	显示链路失败计数。
`Loss of Sync Count`	显示失去同步的次数。

Loss of Signal	显示丢失信号的次数。
Primitive Seq Protocol Err Count	显示原语序列出错的次数。
Invalid Tx Word Count	显示发生的无效传输次数。
Invalid CRC Count	显示发生的 CRC 错误数。
IP over FC Adapter Driver Information: No DMA Resource Count	显示 DMA 资源不可用的次数。
IP over FC Adapter Driver Information: No Adapter Elements Count	显示出现“没有适配器元素可用”情况的次数。
FC SCSI Adapter Driver Information: No DMA Resource Count	显示 DMA 资源不可用的次数。
FC SCSI Adapter Driver Information: No Adapter Elements Count	显示出现“没有适配器元素可用”情况的次数。
FC SCSI Adapter Driver Information: No Command Resource Count	显示出现“没有命令资源可用”情况的次数。
IP over FC Traffic Statistics: Input Requests	显示输入请求的数量。
IP over FC Traffic Statistics: Output Requests	显示输出请求的数量。
IP over FC Traffic Statistics: Control Requests	显示控制请求的数量。
IP over FC Traffic Statistics: Input Bytes	显示输入字节数。
IP over FC Traffic Statistics: Output Bytes	显示输出字节数。
FC SCSI Traffic Statistics: Input Requests	显示输入请求的数量。
FC SCSI Traffic Statistics: Output Requests	显示输出请求的数量。
FC SCSI Traffic Statistics: Control Requests	显示控制请求的数量。
FC SCSI Traffic Statistics: Input Bytes	显示输入字节数。
FC SCSI Traffic Statistics: Output Bytes	显示输出字节数。

退出状态

0	命令成功完成。
>0	发生错误。

示例

1. 要显示光纤通道设备驱动程序 fcs0 的统计信息，请输入:

   ```
   fcstat fcs0
   ```

   将显示类似于下面的输出。

   注: 不同 AIX 命令的输出格式并非总是固定不变。编写程序时，请勿期望 **fcstat** 命令的输出一直保持如下形式。

   ```
   FIBRE CHANNEL STATISTICS REPORT: fcs0

       Device Type: FC Adapter (df1000f9)
       Serial Number: 1E313BB001
       Option ROM Version: 02C82115
       Firmware Version: B1F2.10A5
       Node WWN: 20000000C9487B04
       Port WWN: 10000000C9416DA4

       FC4 Types
         Supported: 0x0000010000000000000000000000000000000000000000000000000000000000
         Active:    0x0000010000000000000000000000000000000000000000000000000000000000
       Class of Service: 4
       Port FC ID: 011400
       Port Speed (supported): 2 GBIT
       Port Speed (running):   1 GBIT
   ```

```
Port Type: Fabric

Seconds Since Last Reset: 345422

Transmit Statistics    Receive Statistics
-------------------    ------------------
Frames: 1              Frames: 1
Words: 1               Words: 1

LIP Count: 1
NOS Count: 1
Error Frames:  1
Dumped Frames: 1
Link Failure Count: 1
Loss of Sync Count: 1
Loss of Signal: 1
Primitive Seq Protocol Err Count: 1
Invalid Tx Word Count: 1
Invalid CRC Count: 1

IP over FC Adapter Driver Information
  No DMA Resource Count: 0
  No Adapter Elements Count: 0

FC SCSI Adapter Driver Information
  No DMA Resource Count: 0
  No Adapter Elements Count: 0
  No Command Resource Count: 0

IP over FC Traffic Statistics
  Input Requests:   0
  Output Requests:  0
  Control Requests: 0
  Input Bytes:  0
  Output Bytes: 0

FC SCSI Traffic Statistics
  Input Requests:   16289
  Output Requests:  48930
  Control Requests: 11791
  Input Bytes:  128349517
  Output Bytes: 209883136
```

位置

/usr/sbin/fcstat

相关信息

atmstat 命令、第 328 页的『entstat 命令』、**fddistat** 命令、**netstat** 命令和 **tokstat** 命令。

---

fcstkrpt 命令

用途

显示 FFDC 错误堆栈文件的内容。

语法

/usr/sbin/rsct/bin/fcstkrpt { [**-a**] [**-p** | **-r**] { **-f** *FFDC_Failure_Identifier* [**-i**] | **-s** *FFDC_Error_Stack_File_Name* } } | [**-h**]

描述

fcstkrpt 读取已经存在的 FFDC 错误堆栈文件并且向标准输出设备上显示它的内容。FFDC 错误堆栈文件通过文件名称本身指定，或者通过引用文件范围内特定记录的 FFDC 故障标识符来指定。

从 FFDC 错误堆栈得到的信息可以以下面两种格式中的一种来显示: 通过*相关故障情况*（缺省值）或者通过*软件层*。

标志

-a 指出对于 FFDC 错误堆栈中的条目所有将要显示的信息。缺省的操作就是显示记录时间戳记和事件的描述。

-f 指定 FFDC 故障标识符以便能够在定位 FFDC 错误堆栈的时候使用。**fcstkrpt** 解码了 FFDC 故障标识，定位与 FFDC 故障标识相关的 FFDC 错误堆栈并且处理 FFDC 错误堆栈。通过这个标志仅仅能指定一个 FFDC 故障标识。

-h 向标准输出设备上显示帮助信息并退出。不管指定的选项，不执行其他处理。

-i 仅仅显示与通过 **-f** 标志确定的特定故障报告相关的信息。在缺省情况下，显示所有在 FFDC 错误堆栈中的记录。

-p 沿着进程方向显示 FFDC 错误堆栈的信息。输出是有序的因此能够反映进程创建的顺序（父子关系）。首先显示子进程信息，紧跟着是父进程信息。这个视图被用来理解哪些事件先发生，哪些事件是由于它们而发生的。

-r 按照事件的关系显示 FFDC 错误堆栈中的信息。提供事件以及与之相关的事件。这个视图被用来理解哪些事件是由于其他事件的发生而发生的。此为缺省设置。

-s 指定要被检查的 FFDC 错误堆栈名称。这个名称可能是绝对的或者是相对的 FFDC 错误堆栈路径名称。只有一个 FFDC 错误堆栈文件名称可以通过这个标志来指定。如果用到了一个相对的文件名称，这个文件被认为是定位在节点的 **/var/adm/ffdc/stacks** 目录中，这里驻留着文件。

参数

FFDC_Failure_ID

指定故障的 FFDC 故障标识来开始报告。**fcreport** 将会试图为此故障获得故障信息，以及这个报告列出的作为相关故障的任何故障。只有一个 FFDC 故障标识可以提供给这个命令。

安全性

fcreport 利用 **rsh** 来获得也许驻留在远程节点上的故障报告。用户必须有足够的权限来执行用于远程节点的 **rsh** 命令。如果用户没有这个许可，只要它们在本地节点存在，**fcreport** 就只能追踪到相关故障的列表。

退出状态

fcstkrpt 完成之后会给出下面这些整型退出状态码:

0 FFDC 错误堆栈成功的被定位，并且已经将内容显示到了标准输出设备上。

2 显示帮助信息并结束进程。

12 指定的选项无效。

14 没有信息写入标准输出设备。使用了 **-f** 选项并且 “*FFDC 故障标识*” 参数是非法的。

20 没有信息写入标准输出设备。使用了 **-s** 选项并且没有找到 *FFDC 错误堆栈文件*参数。

27 没有信息写入标准输出设备。调用者提供了一个合法的 *FFDC 故障标识*，但是通过 FFDC 故障标识引用的文件没有在这个节点上记录。利用 **fcdecode** 命令来定位节点，这个节点上驻留着 FFDC 错误堆栈。

81 没有信息写入标准输出设备。当向标准输出设备上写入信息的时候产生了故障。应用程序应该得出结论：标准输出设备不能接受输出。

85 没有信息写入标准输出设备。调用者提供了一个合法的 FFDC 故障标识，但是 FFDC 故障标识引用的文件却不存在。

示例

为了获得存储在 FFDC 错误堆栈文件 **/var/adm/ffdc/stacks/myprog.562.19981001143052** 中信息的一个简短报告:

```
$ fcstkrpt -r -s myprog.562.19981001143052
```

为了获得包含在 FFDC 错误堆栈中信息的详细报告，在这里记录了 **.3Iv04ZVVfvp.wtY0xRXQ7.....** 错误故障标识符，并且以父子顺序显示了这些信息。

```
$ fcstkrpt -p -f .3Iv04ZVVfvp.wtY0xRXQ7...................
```

相关信息

命令：**fcclear**、**fcdecode**、**fcdispfid**、**fcfilter**、**fcpushstk**、**fcreport**

子例程：**fc_push_stack**（请参考 *RSCT First Failure Data Capture Programming Guide and Reference*）

---

fcteststk 命令

用途

测试“首次故障数据捕获错误堆栈（First Failure Data Capture Error Stack）”环境的存在。

语法

/usr/sbin/rsct/bin/fcteststk [**-q**] | [**-h**]

描述

fcteststk 可以被希望通过 FFDC 错误堆栈测试这些设备是否已被激活的任何应用程序调用。通过进行这种测试，应用程序可以避免收集故障信息时的性能负担，在 *FFDC 环境* 没有建立起来的情况。此接口主要通过库例程被提供，库例程对于它们的客户应用程序是否建立或者是否继承了 *FFDC 环境* 还不知道。

当进程希望从本身得到故障信息，创建的任何线程和任何派生进程记录到 FFDC 错误堆栈，*FFDC错误堆栈环境* 被进程建立了。当进程希望向 FFDC 错误堆栈文件记录故障信息的时候，仅当祖先进程之一要求进程这样做的时候，*FFDC Error Stack 环境* 才会被进程继承；在其他情况下，进程不会向 FFDC 错误堆栈中记录故障信息。进程使用 **fcinit** 建立或者继承“FFDC错误堆栈环境”。

“FFDC错误堆栈环境”保留一个 FFDC 错误堆栈文件，以便故障信息能够被记录在 **/var/adm/ffdc/stacks** 目录中的文件。这些文件采用 ***script_name.PID.date_and_time*** 的命名格式，这里 *script_name* 是脚本本身的

名字 *PID* 是脚本的进程标识，*date_and_time* 是脚本执行的日期和时间。无论何时脚本或者脚本的子进程向“FFDC 错误堆栈”中记录故障信息，它都将记录到这个文件中。

应用程序使用 **fcpushstk** 接口来向 FFDC 错误堆栈中记录故障信息。在记录这些信息之前，应用程序也许需要从各种不同的位置来收集信息，获得这些信息能影响应用程序的全面性能。如果 *FFDC Error Stack 环境*没有建立或者被继承，应用程序不必收集这些信息。为了避免这种性能影响，应用程序可以调用 **fcteststk** 来决定是否可以得到 *FFDC Error Stack 环境*如果可以得到就开始收集故障信息。如果 *FFDC Error Stack 环境*不存在，则应用程序可以避免收集这种信息。

当*FFDC 环境*存在的时候，使用 **fclogerr** FFDC 接口的进程可以使用 **fclogerr**，无论 FFDC 错误堆栈是否被 *FFDC 环境*在使用。无论何时使用 **fclogerr** ，不管是否保留 FFDC 错误堆栈，故障信息都将被记录到 AIX 错误日志和 BSD 系统日志中。使用 **fclogerr** 接口记录信息的应用程序必须总是收集故障信息并记录它，无论 FFDC 错误堆栈是否被使用。

标志

-h 对这个命令显示用法消息。不执行进一步的处理。

-q 禁止从解释 FFDC 环境是否被建立的命令的输出。命令的用户将会被要求从决定是否为这个进程建立了 FFDC 环境的命令来检查退出状态。

参数

FFDC_Failure_ID

指定故障的 FFDC 故障标识来开始报告。**fcreport** 将会试图为故障获得故障信息，以及这个报告作为相关故障列出的任何故障。可能只有一个 FFDC 故障标识提供给了这个命令。

安全性

fcreport 使用 **rsh** 来获得也许驻留在远程节点上的故障报告。用户必须有足够的权限来执行这些用于远程节点的 **rsh** 命令。如果用户没有这个许可，只要它们在本地节点存在，**fcreport** 就只能追踪到相关故障的列表。

退出状态

0 存在 FFDC 错误堆栈环境。

2 显示帮助信息并结束进程。

12 没有进行处理。指定的选项无效。

15 用户没有及时在该点建立或者继承 FFDC 错误堆栈环境。

17 FFDC 错误堆栈环境很可能损坏并且应该被视为是不可用的。

示例

要检测应用程序的 FFDC 错误堆栈环境是否存在:

```
fcteststk -q
if (($? == 0))
then    # Collect failure information
         :
         :
```

```
        # Use fcpushstk to record failure info
             :
             :
    fi
```

相关信息

命令: **fcinit**、**fcpushstk**

子例程: **fc_test_stack**（请参考 *RSCT First Failure Data Capture Programming Guide and Reference*）

---

fddistat 命令

用途

显示 “FDDI” 设备驱动程序和设备统计信息。

语法

fddistat [**-r** **-t**] *Device_Name*

描述

fddistat 命令显示了指定的 “FDDI” 设备驱动程序收集的统计信息。如果没有指定标志，则仅仅会显示设备驱动程序的统计信息。当 **netstat** 命令带有 **-v** 标志同这个命令一同运行的时候，这个命令也会被调用。**netstat** 命令并不会发出任何 **fddistat** 命令标志。

如果指定了非法的*设备名称*，**fddistat** 命令将会产生一个错误信息，说明不能够连接设备。

标志

-r 重新将所有的统计信息设置成为它们的初始值。这个标志只能被特权用户调用。
-t 切换某些设备驱动程序中的调试跟踪。

参数

Device_Name “FDDI” 设备的名称，例如，**fddi0**。

统计信息字段

注: 某些适配器可能不支持特定的统计信息。不支持统计数据的字段总是 0。

在 **fddistat** 命令的输出中显示的统计数据字段和它们的描述是:

标题字段

`Elapsed Time` 显示从上一次统计数据被重置开始到现在的真实时间周期。因为当检测到硬件错误的时候会有错误恢复阶段，在这期间部分统计数据也许会被设备驱动程序重置，当这个条件产生的时候，在输出中间会显示另一个过去的时间以便反映在统计数据之间时间上的不同。

传送统计数据字段

Packets	成功的被设备传送的信息包的数量。
Bytes	成功的被设备传送的字节数目。
Interrupt	从适配器中被设备接收的中断的数目。
Transmit Errors	在这个设备上出现的输出错误数目。这是为由于硬件或网络错误传送不成功而设立的计数器。
Packets Dropped	驱动程序由于传送而接受的信息包的数目，这次传送没有被送入设备（由于某种原因）。
Max Packets on S/W Transmit Queue	软件传送序列中能够排队的最大信息包的数量。
S/W Transmit Queue Overflow	软件传送队列溢出的信息包数量。
Current S/W+H/W Transmit Queue Length	在软件传送队列或者在硬件传送队列中被暂挂的信息包数量。
Broadcast Packets	没有任何错误传送出去的广播信息包数量。
Multicast Packets	没有任何错误传送出去的多点广播信息包的数量。

接收统计数据字段

Packets	成功的被设备接收的信息包的数量。
Bytes	成功的被设备接收的字节数。
Interrupts	从适配器被驱动程序收到的中断数。
Receive Errors	设备中遇到的输入错误的数目。这是为由于硬件或网络错误而产生接收不成功而设立的计数器。
Packets Dropped	设备驱动程序从此设备接收的信息包的数目，没有被送入网络设备（由于某种原因）。
Bad Packets	设备驱动程序接收的（例如，保存）坏的信息包数量。
Broadcast Packets	没有任何错误接收的广播信息包数量。
Multicast Packets	没有任何错误接收的多点广播信息包的数量。

常规的统计数据字段

No mbuf Errors	对于设备驱动程序 mbufs 不可得的次数。接收操作中当驱动程序必须获取 mbuf 来处理入站信息包的时候，这种情况通常会发生。如果要求尺寸的 mbuf 池是空的话，则这个信息包就会废弃。**netstat -m** 命令可以被用来确认这个动作。
SMT Error Word	适配器的 SMT 错误状态。
SMT Event Word	适配器的 SMT 事件状态。
Connection Policy Violation	适配器对响铃的连接状态。
Port Event	适配器的端口状态。
Set Count	当前设置计数值。
Adapter Check Code	适配器最近的检查状态。
Purged Frames	接收由于缺少可用描述符而被适配器删除的帧。
ECM State Machine	实体协调管理状态机器。
PCM State Machine: Port A	对于主要适配器状态机器的物理连接管理
PCM State Machine: Port B	对于次要适配器状态机器的物理连接管理
CFM State Machine: Port A	对于主要适配器状态机器的配置管理
CFM State Machine: Port B	对于次要适配器状态机器的配置管理
CF State Machine	全面配置状态机器。
MAC CFM State Machine	对于 MAC 状态机器的配置管理。
RMT State Machine	响铃管理状态机器。
Driver Flags	设备驱动程序当前打开的内部状态标志。

示例

要显示 **fddi0** 的设备驱动程序统计数据，请输入:

```
fddistat fddi0
```

这个动作产生了下面的输出:

```
------------------------------------------------------------
FDDI STATISTICS (fddi0) :
Elapsed Time: 0 days 0 hours 1 minutes 3 seconds

Transmit Statistics:                          Receive Statistics:
--------------------                          -------------------
Packets: 100                                  Packets: 100
Bytes: 113800                                 Bytes: 104700
Interrupts: 100                               Interrupts: 100
Transmit Errors: 0                            Receive Errors: 0
Packets Dropped: 0                            Packets Dropped: 0
Max Packets on S/W Transmit Queue: 0          Bad Packets: 0
S/W Transmit Queue Overflow: 0
Current S/W+H/W Transmit Queue Length: 0

Broadcast Packets: 0                          Broadcast Packets: 0
Multicast Packets: 0                          Multicast Packets: 0

General Statistics:
-------------------
No mbuf Errors: 0
SMT Error Word: 00040080                      SMT Event Word: 000004a0
Connection Policy Violation: 0000             Port Event: 0000
Set Count Hi: 0000                            Set Count Lo: 0003
Adapter Check Code: 0000                      Purged Frames: 0

ECM State Machine:         IN
PCM State Machine Port A: CONNECT
PCM State Machine Port B: ACTIVE
CFM State Machine Port A: ISOLATED
CFM State Machine Port B: CONCATENATED
CF State Machine:          C_WRAP_B
MAC CFM State Machine:     PRIMARY
RMT State Machine:         RING_OP

Driver Flags: Up Broadcast Running
        Simplex DualAttachStation
```

相关信息

atmstat 命令、**entstat** 命令、**netstat** 命令、**tokstat** 命令。

---

fdformat 命令

用途

fdformat 命令格式化软盘。

语法

fdformat [*Device*] [**-h**]

描述

注: 格式化软盘或者可读 / 写光盘会破坏它们上面存在的数据。

fdformat 命令格式化在指定软盘驱动器中的低密度软盘除非指定了 **-h** 标志。

所有新的、空白的软盘在使用前必须格式化。

在格式化软盘或者可读 / 写光盘之前 **fdformat** 命令会提示确认。这允许您直接地结束这个操作。

标志

-h 强制高密度格式化。这个标志只能随着 **fdformat** 命令使用。

参数

Device 指定包含要格式化的软盘的设备。缺省值是 **/dev/rfd0** 设备的 0 号驱动器。

示例

当使用 **fdformat** 命令的时候，要强制高密度格式化软盘，请输入:

```
fdformat -h
```

文件

/usr/sbin/fdformat	包含 **fdformat** 命令。
/dev/rfd*	指定设备参数。
/dev/fd*	指定设备参数。
/dev/romd*	指定设备参数。
/dev/omd*	指定设备参数。

相关信息

flcopy 命令、 **format** 命令。

fd 特殊的文件。

---

fdpr 命令

用途

改善用户级后期连接应用程序的执行时间和实际内存使用率的性能调整实用程序。

语法

常规用法:

fdpr -p *ProgramFile* **-x** *WrokLoadCommand*

详细用法:

fdpr -p *ProgramFile* [**-M** *Segnum*] [**-fd** *Fdesc*] [**-o** *OutputFile*] [**-armember** *ArchiveMemberList*] [OptimizationFlags] [**-map**] [**-disasm**] [**-disasm_data**] [**-disasm_bss**] [**-profcount**] [**-quiet**] [**-v**] [**-1 | -2 | -3 | -12 | -23 | -123**] [**-x** *WorkloadCommand*]

优化标志

[**-tb**] [**-pc**] [**-pp**] [**-O**][**-O2**] [**-O3**] [**-O4**] [**-selective_inline**] [**-sid_fac** *percent*] [**-inline_small_funcs** *size*] [**-inline_hot_funcs** *percent*] [**-hco_resched**] [**-killed_regs**] [**-lr_opt**] [**-align** *bytes*] [**-RD**] [**-dpnf** *factor*] [**-dpht** *threshold*] [**-build_dcg**] [**-tocload**] [**-ptrgl_opt**] [**-no_ptrgl_r11**] [**-dcbt_opt**] [**-ignore_info**] [**-dead_code_removal**] [**-bt_csect_anchor_removal**] [**-strip**] [**-analyse_asm_csects**] [**-extra_safe_analysis**] [**-inline**] [**-reduce_toc** *removal_factor*]

描述

fdpr 命令（反馈定向程序重构）是可以帮助改善用户级应用程序执行时间和内存实际使用率的性能调整程序。**fdpr** 程序通过收集程序在某些典型工作负荷下运行时的行为信息来优化某个程序的可执行映像，然后创建一个针对此工作负荷而优化的新版本的程序。通常，**fdpr** 所生成的新程序可以运行得更快并且实际内存使用得更少。

警告: **fdpr** 命令为程序应用了高级优化技术，可能会产生行为与预期不相符的程序；使用这种工具优化过的程序应当小心使用，同时应该进行严格的再次测试，至少应该使用测试原始程序的同一测试套件来验证期望的功能。不支持优化过的程序。

fdpr 命令在 3 个不同的阶段构建优化的可执行程序:

- 阶段 1（**-1** 标志）: 创建已检测的可执行程序和空模板概要文件。
- 阶段 2（**-2** 标志）: 运行已检测的程序并更新概要文件数据。
- 阶段 3（**-3** 标志）: 生成已优化的可执行程序文件。

这些阶段可以单独运行，也可以部分或者完全结合在一起来运行，但是必须按照顺序运行（即，**-1** 然后 **-2**，然后 **-3**，或者 **-12**，然后再 **-3**）。缺省值是运行全部三个阶段。

注: 在阶段 1 创建并且在阶段 2 运行的经过检测的可执行程序一般比原始程序运行速度慢好几倍。因为经过检测的程序在执行时间上的增长，可执行程序应该这样调用以最小化执行时间，同时在希望的代码区内仍然完全执行。**fdpr** 命令的用户应该在可行的情况下试图消除一切程序时间依赖的特性。

标志

-1-2 -3 指定要运行的阶段。缺省值是全部 3 个阶段（**-123**）。**-s** 标志必须在运行独立的阶段时使用，以便成功的阶段可以访问所需的中间文件。必须按顺序运行各个阶段（例如，先是 **-1**，然后是 **-2**，接下来是 **-3**，或者先是 **-1**，然后是 **-23**）。**-2** 标志必须与调用标志 **-x** 一起使用。

-M *SegNum* 指定为概要分析映射共享内存的位置。缺省值是 **0x30000000**。如果需要优化的程序或者任何用 **-x** 标志调用的工作负荷命令字符串使用冲突的共享内存地址，则要指定一个备用的共享内存地址。典型的可选值是 **0x40000000**，**0x50000000** ... 直到 **0xC0000000**）。

-fd *Fdesc* 指定为**概要文件**使用哪个文件描述符数字，概要文件会被映射到以上的共享内存区。Fdesc 的缺省值设置为 1999。

-o *OutFile* 从优化器中指定输出文件的名称。缺省值是 *program***.fdpr**

-p *ProgramFile* 包含要优化的可执行程序文件或者共享对象文件的名称，或者包含共享对象 / 可执行（文件）的共享库名称。这个程序必须是一个完整的可执行程序。

-armember *ArchiveMemberList* 要优化的压缩文档成员列表，位于 **-p** 标志指定的共享压缩文件中。如果 **-armember** 没有指定，压缩文档文件的所有成员都会被优化。

-map 将带各自的原有 -> 新地址的基本块和静态变量的映射打印为后缀为 **.mapper** 的文件。

-disasm 将已输出优化和检测的程序的反汇编的文本部分打印为后缀为 **.dis_text** 的文件。
-disasm_data 将已输出优化和检测的程序的反汇编数据部分打印为后缀为 **.dis_data** 的文件。
-disasm_bss 将已输出优化和检测的程序的反汇编的 bss 部分打印为后缀为 **.dis_bss** 的文件。
-profcount 将概要分析计数器打印为后缀为 **.ncounts** 的文件。
-quiet 安静输出方式。
-v 冗长的输出。
-x *WorkloadCommand* 指定用来调用经检测的程序的命令。**-x** 标志后的所有参数都用于调用。因此，命令行最后应该存在 **-x** 标志。如果使用 **-2** 标志，则 **-x** 标志是必需的。

优化标志

-analyse_asm_csects 分析以汇编语言编写的 csect（如果使用，必须同时在 -1 和 -3 阶段指定）。
-extra_safe_analysis 请勿尝试分析包含手写汇编代码的非传统 csect（如果使用，必须同时在 -1 和 -3 阶段指定）。
-ignore_info 忽略编译过程中使用 **-qfdpr** 选项生成的 **.info** 部分（如果使用，必须同时在 -1 和 -3 阶段指定）。
-align *bytes* 排列根据给定数量的字节频繁执行的代码，以便提高代码预取缓存率。如果忽略了此选项，**fdpr** 命令将排列可变的缺省字节数的代码。
-lr_opt 排除频繁执行的过程中链接寄存器的存储和恢复。
-bt_csect_anchor_removal 排除与代码中分支表的用法相关的装入指令。
-dead_code_removal 删除不可达代码。
-selective_inline 为从一个单独的主要调用位置频繁调用的函数执行选择性直接插入。
-sid_fac *percent* 为选择性直接插入优化设置主要因子百分比。允许的范围为 50 - 100（仅在使用 **-selective_inline** 标志时适用）。
-inline_small_funcs *size* 直接插入小于或等于以字节为单位的给定大小的所有函数。
-inline_hot_funcs *percent* 直接插入执行频率等于或大于给定百分比的所有函数。输入百分比范围为 0 - 100。
-inline 结合 **-selective_inline** 执行 **-inline_small_funcs 12**。
-hco_resched 如果可能，将来自频繁执行的代码的指令重新定位到很少执行的代码区域。
-dcbt_opt 插入 **dcbt** 指令来提高数据高速缓存的性能。
-killed_regs 排除频繁执行的函数调用后*被杀死*（覆盖）的寄存器的存储和恢复。
-tb 在重新排序的代码中强制重构回溯表。如果省略了 **-tb** 选项，将使用 Try & Catch 机制自动为 C++ 应用程序恢复回溯表。
-pc 在重新排序后的代码中保留 csect 的边界。
-pp 在重新排序后的代码中保留函数的边界。
-RD 执行静态数据重新排序。
-dpnf *factor* 0 - 1 之间的“数据放置规范化因子”；其中，0 将导致重新排序静态变量且不考虑其大小，而 1 将首先仅查找大小较小的变量（仅在结合 **-RD** 标志时适用）。
-dpht *threshold* 0 - 1 之间的“数据放置热阈值”；其中，0 将根据控制流重新排序大组中的静态变量，而 1 将根据极小组中的变量的访问频率重新排序这些变量（仅在结合 **-RD** 标志使用时适用）。
-build_dcg 为增强的数据重新排序构建 DCG（数据连接图）（仅在结合 **-RD** 标志使用时适用）。
-tocload 执行 tocload 优化。
-reduce_toc *removal_factor* 根据删除 0 - 1 之间的因子执行 TOC 条目，其中 0 将仅删除未访问的 TOC 条目，而 1 将删除未导出的 TOC 条目。
-strip 剥离输出文件（如果生成了的话）。
-ptrgl_opt 通过使用寄存器将间接调用指令替换为直接跳转来执行对对这些指令的优化。
-no_ptrgl_r11 请勿执行 _ptrgl csect 中的 R11 装入指令的删除（缺省情况下将应用 **-ptrgl_r11** 优化）。
-O 使用分支预测位设置、分支折叠和 NOOP 指令删除执行代码重新排序。缺省情况下将应用 **-O** 标志。
-O2 打开所有有较少影响的优化标志。

-O3	打开所有有影响的优化标志。
-O4	打开所有有影响的优化标志。

优化

fdpr 命令在缺省情况下执行级别最可能的代码重新排序优化，以及分支预测位设置、分支折叠、代码对齐和删除冗余的 NOOP 指令的优化。**-pc** 标志在保留 csect 的边界的同时重新排序整个代码，因此可能致使提高性能方面不如缺省的代码重新排序。与此类似，**-pp** 标志在保留过程的边界时重新排序整个代码。

上面的优化标志也可以对整个可执行程序文件执行其他优化。

使用 **-qfdpr** IBM xl 编译器标志构建的可执行程序包含在生成重新排序程序时辅助 **fdpr** 命令的信息。没有使用 -qfdpr 选项编译的模块 在符号表内的编译器签名的基础上被重新排序。

在重新排序程序时使用静态链接可能实现程序额外的性能提升。因为 **fdpr** 程序仅仅对指定的可执行程序重新排序指令，任何被程序调用的动态链接共享库例程都不会被优化。静态链接这些库例程到可执行程序，则允许优化在程序中的指令，也可以优化被程序使用的所有库例程。这些可能是构建静态链接程序的优点，也可能是缺点。请参阅《*性能管理*》以获取进一步的信息。

输出文件

所有被 fdpr 命令创建的文件被保存在当前的目录中，例外情况是可能在 -x 标志指定的工作负荷命令中被创建的文件。 在优化过程中，原始程序通过重新命名程序来保存，仅当最后一个阶段的工作都结束时才恢复原始的程序名。

因为运行此程序的脚本可能在执行程序之前更改工作目录，所以 **fdpr** 命令创建的概要文件显式地使用当前目录的全名。

fdpr 命令创建和 / 或使用的文件是:

program	要优化的无条纹可执行程序的名称。
program.**save**	原始可执行程序保存后的版本。
program.**nprof**	概要文件的名称。
program.**instr**	程序检测版本的名称。
program.**fdpr**	优化可执行输出文件的缺省名称。
program.**instr.dis_text**	检测阶段后由 **-disasm** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**fdpr.dis_text**	优化阶段后由 **-disasm** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**instr.dis_data**	检测阶段后由 **-disasm_data** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**fdpr.dis_data**	优化阶段后由 **-disasm_data** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**instr.dis_bss**	检测阶段后由 **-disasm_bss** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**fdpr.dis_bss**	优化阶段后由 **-disasm_bss** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**instr.mapper**	检测阶段后由 **-map** 标志生成的 ASCII 格式的缺省映射文件。
program.**fdpr.mapper**	优化阶段后由 **-map** 标志生成的 ASCII 格式的缺省映射文件。
program.**ncounts**	由 **-profcount** 标志生成的 ASCII 格式的缺省概要表计数器文件。

增强的调试能力

为了使优化后的程序具备一定程度的调试能力，**FDPR** 将更新“符号表”以反映 **.text** 部分中所作的更改。

在重新排序 **FDPR** 的过程中指定被重新定位的符号的地址的“符号表”中的输入字段被修改为指向它们在 **.text** 部分中的新地址。

另外，如果在重新排序过程中分割了函数或文件，**FDPR** 将在“符号表”中为被分割的函数 / 文件的每个新部分创建新条目。这些相同函数的新部分根据以下命名约定来产生符号表中新的符号名:

```
<original function name>__fdpr_<function's part number>
```

代码重新排序之后，所有新条目将带有后缀 __fdpr_ 字符串。

示例: 函数“main”最初在“符号表”中具有以下条目:

```
  [Index] m   Value       Scn     Aux   Sclass    Type    Name
   [456]  m  0x00000230    2       1     0x02    0x0000   .main
```

如果代码重新排序后，函数 main 被分割为 3 个部分，则它在“符号表”中将有 3 个条目；每个部分都具有一个如下条目:

```
[Index] m   Value       Scn     Aux   Sclass    Type    Name
   [456]  m  0x00000304    2       1     0x02    0x0000   .main
  [1447]  m  0x00003328    2       1     0x02    0x0000   .main__fdpr_1
  [1453]  m  0x000033b4    2       1     0x02    0x0000   .main__fdpr_2
```

示例

以下的是关于 fdpr 的典型使用示例。

1. 这个示例允许用户运行所有 3 个阶段。在这个示例中，test1 是无条纹的可执行程序 test2 是调用 test1 的 shell 脚本。 当前的工作目录是 **/tmp/fdpr** 。

   test2 脚本文件:

   ```
   # code to exercise test1
   test1 -expand 100 -root $PATH file.jpg -quit
   # the end of test2
   ```

   执行 **fdpr** 命令（使用缺省优化）:

   ```
   fdpr -p test1 -x test2
   ```

   这样导致新的重新排序的可执行程序 **test1.fdpr** 。
2. 要在任何时候运行某个阶段，请执行 **fdpr** 的阶段 1。

   ```
   fdpr -1 -p test1
   ```

   此命令字符串将创建一个名称为 test1.instr 的已检测版本和空模板概要文件 test1.nprof。

   要执行阶段 2:

   ```
   fdpr -2 -p test1 -x test2
   ```

   这个命令字符串执行脚本文件 test2 ， 此文件运行 test1 的检测版本来收集概要文件数据。

   要执行阶段 3:

   ```
   fdpr -3 -p test1
   ```

   这将再次导致新的重新排序的可执行文件 **test1.fdpr** 。
3. 要运行前两个阶段后再运行阶段 3，请执行阶段 1 和阶段 2。

   ```
   fdpr -12 -p test1 -x test2
   ```

   使用优化级别 3 执行阶段 3。

   ```
   fdpr -3 -O3 -p test1
   ```
4. 如果在运行使用 **fdpr** 优化过的程序时出错，可以如下方式使用 **dbx** 命令确定发生错误的是哪一个过程:

```
dbx program.fdpr
```

产生的输出类似于以下内容:

```
Type 'help' for help.
reading symbolic information ...warning: no source compiled with -g

[在核心中使用内存映像]

Segmentation fault in proc_d at 0x10000634
0x10000634 (???) 98640000        stb   r3,0x0(r4)
(dbx)
```

堆栈回溯可以用来确定程序是如何到达当前位置的，它如以下来生成:

```
(dbx)
```

生成如下输出的位置:

```
proc_d(0x0) at 0x10000634
proc_c(0x0) at 0x10000604
proc_b(0x0) at 0x100005d0
proc_a(0x0) at 0x1000059c
main(0x2, 0x2ff7fba4) at 0x1000055c
(dbx)
```

5. **dbx** 子命令 **stepi** 也可以用作重新排序后的可执行程序的单步遍历指令，如下:

```
(dbx) stepi
```

生成如下输出的位置:

```
stopped in proc_d at 0x1000061c
0x1000061c (???) 9421ffc0       stwu   r1,-64(r1)
(dbx)
```

在本示例中，**dbx** 表明例程 proc_d 中，程序在重新排序后的文本部分内的地址 0x1000061c 处停止。

具体实现

软件产品 / 选项: *AIX 性能助手 / 本地性能分析和控制命令*。

标准一致性: 无。

文件

/usr/bin/fdpr	包含 **fdpr** 命令。
program	要优化的完整的可执行程序名称。
program.**save**	原始可执行程序保存后的版本。
program.**nprof**	概要文件的名称。
program.**instr**	程序检测版本的名称。
program.**fdpr**	优化可执行输出文件的缺省名称。
program.**instr.dis_text**	检测阶段后由 **-disasm** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**fdpr.dis_text**	优化阶段后由 **-disasm** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**instr.dis_data**	检测阶段后由 **-disasm_data** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**fdpr.dis_data**	优化阶段后由 **-disasm_data** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**instr.dis_bss**	检测阶段后由 **-disasm_bss** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**fdpr.dis_bss**	优化阶段后由 **-disasm_bss** 标志生成的 ASCII 格式的缺省反汇编文件。
program.**instr.mapper**	检测阶段后由 **-map** 标志生成的 ASCII 格式的缺省映射文件。
program.**fdpr.mapper**	优化阶段后由 **-map** 标志生成的 ASCII 格式的缺省映射文件。
program.**ncounts**	由 **-profcount** 标志生成的 ASCII 格式的缺省概要表计数器文件。

相关信息

dbx 命令。

《*性能管理*》中的『用 fdpr 程序重构可执行程序』。

xlC 编译器。

---

fencevsd 命令

用途

阻止在一个或一组节点上运行的应用程序访问一个或一组虚拟共享磁盘。

语法

fencevsd {**–a** | **–v** *vsd_name_list*} **–n** *node_list*

描述

在某些情况下，当节点实际上在运行，但已切断与其他正在运行相同应用程序的节点之间的通信时，系统可能认为此节点已停止运行而开始恢复过程。在这种情况下，不得让问题节点为它通常提供服务的虚拟共享磁盘的请求提供服务，直到完全恢复为止，同时运行此应用程序的其他节点将认为问题节点在运行。**fencevsd** 命令阻止问题节点为其虚拟共享磁盘履行请求。

此命令可从运行可恢复虚拟共享磁盘子系统的 RSCT 对等域中的任何节点运行。

标记

–a 指定所有虚拟共享磁盘。

–v *vsd_name_list*
指定一个或多个虚拟共享磁盘名，用逗号分开。

–n *node_list* 指定一个或多个节点号，用逗号分开。

参数

logical_volume_name
是想要指定为虚拟共享磁盘的逻辑卷的名称。该逻辑卷必须驻留在指明的全局卷组上。名称的长度必须小于或等于 15 个字符。

global_group_name
是由您希望在其中指定虚拟共享磁盘的 **vsdvg** 命令先前所定义的可全局访问卷组的名称。名称的长度必须小于或等于 31 个字符。

vsd_name 为新的虚拟共享磁盘指定唯一的名称。该名称在 RSCT 对等域中必须唯一，并且要避免将来可能产生的名称冲突，该名称在整个集群中也应当是唯一的。建议使用的命名约定为 **vsdnn***gvg_name*。名称的长度必须小于或等于 31 个字符。

注: 如果指定的 *vsd_name* 已经是另一台设备的名称，则 **cfgvsd** 命令对于该虚拟共享磁盘将失败。此错误确保为该名称创建的特殊设备文件不会覆盖和毁坏代表其他某种设备类型（例如逻辑卷）的同名文件。

安全性

您必须有 **root** 权限来运行该命令.

限制

必须从具有活动的可恢复虚拟共享磁盘子系统的对等域中的节点上发出此命令。

示例

要使虚拟共享磁盘 vsd1 和 vsd2 与节点 5 隔开，请输入:

```
fencevsd -v vsd1,vsd2 -n 5
```

位置

/opt/rsct/vsd/bin/fencevsd

相关信息

命令: **lsvsd**、**unfencevsd**

有关可恢复虚拟共享磁盘子系统以及如何使用 **fencevsd** 和 **unfencevsd** 命令在应用程序恢复期间保持数据完整性的更多信息，请参考 *RSCT for AIX 5L: Managing Shared Disks*。

---

feprom_update 命令

用途

装入闪存 EPROM 并重新引导系统。

语法

feprom_update [**-f**] *FileName*

描述

注: 当系统在不止一个用户使用的情况下运行时，请勿使用该命令。

feprom_update 命令将系统的闪存同指定的文件一同装入，该文件必须包含一个合法的二进制闪存映像，然后重新引导系统。文件名也可以是包含闪存映像的软盘驱动器的设备名称。

在缺省情况下，**feprom_update** 命令警告系统会重新启动，并且在进行动作之前会要求确认。如果给定了 -f 标志，警告不会给出；闪存会更新，并且系统会不要求确认就重新启动。

当 feprom_update 命令运行时，系统必须处于服务方式和单用户 root 模式。

注: **feprom_update** 命令仅在具有 AIX 5.1 和更早版本的 Micro Channel® I/O 的多处理器系统上运行。对于 IBM 系统，这仅仅通过 AIX 5.1 包括 IBM 7012 型 G 系列、IBM 7013 型 J 系列和 IBM 7015 型 R 系列。

标志

-f 强制 **feprom_update** 命令来更新闪存并且不要求确认就重新启动系统。

示例

1. 要用 /tmp/eprom.new 文件的内容更新闪存，然后重新启动系统，输入以下命令:

   ```
   feprom_update /tmp/eprom.new
   ```

2. 要用 rfd0 驱动器中的软盘的内容更新闪存，然后没有警告的重新启动系统，输入以下命令:

   ```
   feprom_update -f /dev/rfd0
   ```

File

/usr/sbin/feprom_prom	包含 **feprom_prom** 命令。

相关信息

smit 命令。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

---

ff 命令

用途

列出文件名称和文件系统的统计数据。

语法

ff [**-a** *Number*] [**-c** *Number*] [**-l**] [**-l**] [**-m** *Number*] [**-n** *File*] [**-p** *Prefix*] [**-s**] [**-u**] [**-V** *VFSName*] [**-i** *I-Number* [**,***I-Number ...*]] [*FileSystem* | *DeviceName*]

描述

ff 命令读*FileSystem* 参数指定的文件系统中的 i 节点并且将它们的信息写入标准的输出当中去。命令认为 *FileSystem* 是一个文件系统，这个文件系统在 **/etc/filesystems** 文件中被引用并且保存通过标志指定文件的 i 节点数据。

从 **ff** 命令得到的输出包含每一个请求索引节点号的路径名称，另外还有您使用标志的其他文件信息。输出按照索引节点号的顺序列了出来，在所有的字段两两之间都有跳格隔开。通过 **ff** 命令产生的命令行包括路径名称和索引节点号字段。所有的标志都启用之后，输出字段包括路径名称、索引节点号、大小和“UID”（用户的 ID）。

Number 参数是一个指定了一定天数的十进制数。在它的前面加有“ +”或者“ -”（加号或减号）的前缀。因此 +3 代表这多于 3 天，-3 代表少于3 天，3 代表 3 天，这里的一天是 24 小时。

ff 命令在对于索引节点有多于一个的链接的情况下仅仅列出了众多可能链接中的一个，除非您特别指定 **-l** 标志。有了 **-l** 标志，**ff** 列出了所有的链接。

标志

-a *Number*	如果文件在 *Number* 参数指定的天数中被访问则显示文件。
-c *Number*	显示文件，如果文件在 *Number* 参数指定的天数之内索引节点被修改过。
-i *I-Number*	显示文件，文件同被 *I-Number* 参数指定的索引节点号相对应。列出的索引节点号必须以逗号分隔。

-I （这个标志是一个大写的 i。）在每一个路径名称之后不显示索引节点。

-l （这个标志是一个小写的 L。）另外显示多于一个链接的文件的一系列路径名称。
-m *Number* 显示文件，如果文件在 *Number* 参数指定的天数中被修改。
-n *File* 显示文件，如果文件比 *File* 参数中指定的参数更近的时间内被修改。
-p *Prefix* 向每一个路径名称前添加 *Prefix* 参数指定的前缀。缺省的前缀是 **.**（点）。
-s 在每一个路径名称之后以字节的形式写出文件的大小。
-u 在每一个路径名称之后写出所有者的登录名称。
-V *VFSName* 指示 **ff** 命令假定文件系统具有 *VFSName* 的类型，并且覆盖在 **/etc/filesystems** 文件中的值。

示例

1. 要列出给定文件系统中所有文件的路径名称，请输入:

   ```
   ff    -I    /dev/hd0
   ```

   将在 /dev/hd0 设备上显示文件的路径名称。如果您不指定 **-l** 标志，**ff** 命令同样也会显示每个文件的索引节点号。
2. 要列出最近修改过的文件，请输入:

   ```
   ff    -m    -2    -u    /dev/hd0
   ```

   将会显示在设备 /dev/hd0 中每个文件的路径名称、索引节点号和所有者用户的名称（-u 标志），这些文件在最近两天被修改过。（-m-2）。
3. 要列出最近*没有*被使用过的文件，请输入:

   ```
   ff    -a    +30    /dev/hd0
   ```

   将会显示每个文件的路径名称和索引节点号，这些文件至少有 30 天没有被访问过。（-a +30）。
4. 要找到同某个索引节点号对应的路径，请输入:

   ```
   ff    -l    -i    451,76    /dev/hd0
   ```

   将会显示所有与索引节点451 和 76 有关的路径名称（-l）。

文件

/var/spool/mail/* 包含虚文件系统类型的描述。
/etc/filesystems 列出已知的文件系统并且定义它们的特征。

相关信息

find 命令、**ncheck** 命令。

《*操作系统与设备管理*》中的『文件系统』说明了文件系统类型、管理、结构和维护。

fg 命令

用途
在前台运行作业。

语法
fg [*JobID*]

描述
如果启用了作业控制（请参阅《*操作系统与设备管理*》中的『Korn shell 或 POSIX shell 中的作业控制』），**fg** 命令将当前环境中的后台作业移至前台。使用 *JobID* 参数来指明在前台下要运行的特定作业。如果此参数没有提供，**fg** 命令使用最近在后台被暂挂的作业，或者作为后台作业运行。

JobID 参数可以是进程的标识号，或者您可以使用如下的符号组合:

%*Number* 通过作业编号引用作业。
%*String* 引用名称以特定字符串开头的作业。
%?*String* 引用名称中包含特定字符串的作业。
%+ OR **%%** 引用当前作业。
%- 引用前一个作业。

使用 **fg** 命令将作业放到前台将导致从列表中除去作业进程的标识符，此列表是那些当前外壳环境所知道的。

/usr/bin/fg 命令在它自己的命令执行环境下操作时不工作，因为此环境没有可使用的合适的作业。因为这个原因，**fg** 命令作为 Korn 或者 POSIX 外壳常规内置命令被执行。

退出状态
以下出口值被返回:

0 成功结束。
>0 发生错误。

如果禁用作业控制，**fg** 命令将出错退出，前台不放任何作业。

示例
如果 **jobs -l** 命令的输出显示以下在后台运行的作业:

```
[1] + 16477RunningSleep 100 &
```

可以在前台使用进程标识符来运行 `sleep 100 &` 命令，只需输入:

```
fg 16477
```

屏幕将显示:

```
sleep
```

文件
/usr/bin/ksh 包含 Korn 外壳 **fg** 内置命令。

/usr/bin/fg 包含 **fg** 命令。

相关信息

bg 命令、**csh** 命令、**jobs** 命令、**kill** 命令、**wait** 命令。

《*操作系统与设备管理*》中的『Korn shell 或 POSIX shell 中的作业控制』。

---

fgrep 命令

用途

为文件搜索文字字符串。

语法

fgrep [**-h**] [**-i**] [**-s**] [**-v**] [**-w**] [**-x**] [**-y**] [[**-b**] [**-n**] | [**-c** | **-l** | **-q**]] [**-p***Separator*] {*Pattern* | **-e***Pattern* | **-f***StringFile*} [*File...*]

描述

fgrep 命令搜索 *File* 参数指定的输入文件（缺省为标准输入）中的匹配模式的行。**fgrep** 命令特别搜索 *Pattern* 参数，它们是固定的字符串。如果在 *File* 参数中指定一个以上的文件 **fgrep** 命令将显示包含匹配行的文件。

fgrep 命令于 grep 和 egrep 命令不同，因为它搜索字符串而不是搜索匹配表达式的模式。**fgrep** 命令使用快速的压缩算法。$、*、[、|、(、) 和 \ 等字符串被 **fgrep** 命令按字面意思解释。这些字符并不解释为正则表达式，但它们在 **grep** 和 **egrep** 命令中解释为正则表达式。因为这些字符对于 shell 有特定的含义，完整的字符串应该加上单引号（'...'）。如果没有指定文件，则 **fgrep** 命令假定标准输入。通常情况下，找到的每行都复制到标准输出中去。如果不止一个输入文件，则在找到的每行前打印文件名。

注:

1. **fgrep** 命令和带 -F 标志的 **grep** 命令是一样的但出错和用法消息不同 -s 标志功能也不同。
2. 每行限制在 2048 个字节。
3. 段落（-p 标志下）目前限制在 5000 个字符的长度。
4. 请勿在特定的文件中运行 grep 命令，因为会产生不可预料的结果。
5. 输入行不能包含空字符。
6. 输入文件应该以换行字符结尾。
7. 虽然可以同时指定很多标志，但某些标志会覆盖其余的标志。例如，如果同时指定 **-l** 和 **-n** ，只有文件名写入到标准输出中去。

标志

-b	在找到的每行之前添加行所在的块编号。使用此标志有助于按照上下文查找磁盘块号码。**-b** 标志不能用于标准输入或者管道输入。
-c	仅显示匹配行的计数。
-e *模式*	指定模式。这个工作模式很简单，但当此模式以 a-（减号）开头时却是很有用的。
-f *StringFile*	指定包含字符串的文件。
-h	当多个文件被处理时隐藏文件名。
-i	当进行比较时忽略字母的大小写。

-l	只列出包含匹配行的文件名（一次）。文件名之间用换行符分隔。
n	将文件中每行的相对行号置于行前。
-p*Separator*	显示包含匹配行的整个段落。段落之间将按照 *Separator* 参数指定的段落分隔符加以分隔，这些分隔符是与搜索模式有着相同格式的模式。包含段落分隔符的行将仅用作分隔符；它们不会被包含在输出中。缺省的段落分隔符是空白行。
-q	禁止所有写入到标准输出的操作，不管是否为匹配行。如果选中输入行，以 0 状态退出。
-s	仅显示出错消息。这在检查状态时很有用。
-v	显示除了匹配特定模式的行以外的所有行。
-w	执行单词搜索。
-x	显示匹配模式的行，要求无额外的字符。
-y	当进行比较时忽略字符的大小写。

退出状态

此命令返回以下出口值:

0	找到匹配项。
1	未找到匹配项。
>1	发现语法错误，或者文件不可访问（即使找到了匹配项）。

示例

1. 搜索几个文件中的一个简单字符串:

   ```
   fgrep strcpy *.c
   ```

   在当前目录下所有以 .c 字符串结尾的文件中搜索字符串 strcpy。
2. 计数匹配某模式的行数:

   ```
   fgrep  -c "{"pgm.cfgrep -c "}"pgm.c
   ```

   显示在 pgm.c 中包含左括号和右括号的行的数目。

   如果在您的 C 程序中一行中没有包含多于一个 {（左括号）或者 }（右括号），并且括号正确匹配，则这两个数字将是一样的。如果这两个数字不一样，您可以将包含括号的行按照他们在文件中的位置顺序显示出来，使用以下命令:

   ```
   egrep {\|} pgm.c
   ```
3. 显示包含某模式的文件名:

   ```
   fgrep -l strcpy *.c
   ```

   搜索当前目录下以 .c 结尾的文件，然后显示包含 strcpy 字符串的文件名。

文件

/usr/bin/fgrep	包含 **fgrep** 命令。
/bin/fgrep	链接到 fgrep 命令的符号。

相关信息

ed 命令， **egrep** 命令， **grep** 命令， **sed** 命令。

《*操作系统与设备管理*》中的『文件』介绍了文件以及处理文件的方法。

《*操作系统与设备管理*》中的『输入和输出重定向』描述了操作系统如何处理输入和输出。

---

file 命令

用途

确定文件类型。

语法

对文件类型分类

file [**-m** *MagicFile*] [**-d**] [**-h**] [**-i**] [**-M** *MagicFile*] [**-f** *FileList*] [*File...*]

检查 Magic 文件的格式错误

file -c [**-m** *MagicFile*]

描述

file 命令读取用 *File* 参数或者 *FileList* 变量指定的文件，在每个文件上执行一系列测试，然后将它们按照类型分类。然后此命令将文件类型写入标准输出。文件可以是常规文件、目录、FIFO（指定的管道）、块特殊文件、字符特别文件、符号链接或者套接字类型。

- 对于长度为零的常规文件，将识别为空文件。
- 对于符号链接文件，缺省情况下此链接后跟符号链接引用的文件。

如果文件是 ASCII 码的格式，则 **file** 命令将检查前 1024 个字节然后确定文件类型。如果文件不是 ASCII 格式，则 **file** 命令将尝试区分二进制数据文件和包含扩展字符的文本文件。

如果 *File* 参数指定了可执行程序或者对象模块文件且版本号大于 0，则 **file** 命令将显示版本戳记。**ld** 命令说明了 **a.out** 文件的使用。

file 命令使用 **/etc/magic** 来标识包含某种 magic 数字的文件；即，任何包含可以表示类型的数字或字符串常量的文件。

如果文件不存在，无法读取或者文件状态无法确定，则文件将不看作会影响退出状态的错误。输出表明文件已被处理，但是类型仍无法确定。

当使用了 **-i** 标志时，应使用以下格式标识每个 *file* 指定的操作数:

```
"%s: %s\n", file, type
```

除非在 POSIX 语言环境，否则不指定 *type* 的值，如果 *file* 指定为下表中列出的类型之一，*type* 应包含（但不限于）对应字符串。在字符串中显示的每个空格应是一个空格。

表 1. File 实用程序输出字符串

如果 *file* 是:	*type* 应包含字符串:
目录	目录

表 1. File 实用程序输出字符串 (续)

如果 *file* 是:	***type* 应包含字符串:**
FIFO	fifo
套接字	套接字
块特殊文件	块特殊文件
字符特别文件	字符特别文件
可执行文件（二进制）	可执行文件
空常规文件	空
符号链接	符号链接到
ar 归档库	压缩文档
扩展的 *cpio* 格式	cpio 压缩文档
扩展的 *tar* 格式	tar 压缩文档
Shell 脚本	命令文本
C 语言源	C 程序文本
FORTRAN 源	fortran 程序文本

如果 *file* 标识为符号链接，则应使用以下备用输出格式:

```
"%s: %s %s\n", file, type, contents of link"
```

如果 *file* 操作数指定的文件不存在或不可读，则字符串 `cannot open` 应包含为 *type* 字段的一部分，但这不应当作影响退出状态的错误。如果 *file* 操作数指定的文件的类型不能确定，则字符串 `data` 应包含为 *type* 字段的一部分，但这不应当作影响退出状态的错误。

标志

-c 检查指定的 magic 文件（缺省情况下，**/etc/magic** 文件）有无格式错误。此验证一般不进行。在此标志下文件输入不执行。

-d 将任何缺省系统测试应用到文件。

-f *FileList* 读取指定的文件列表。文件必需在每行列出一个文件，不包含开开头和拖尾空格。

-h 当遇到符号链接时，将文件标识为符号链接。如果未指定 **-h** 标志且 *file* 是指向不存在文件的符号链接，*file* 应将文件标识为符号链接，就好像指定了 **-h** 标志。

-i 如果文件不是常规文件，则不尝试进一步对文件类型进行分类，但如第 442 页的『描述』中指定的那样标识文件。

-m *MagicFile* 指定 magic 文件的文件名（缺省情况下，**/etc/magic**）。

-M *MagicFile* 指定包含应应用于文件以对其分类的测试的文件的名称。不应应用缺省系统测试。

退出状态

此命令返回以下出口值:

0 成功结束。

>0 发生错误。

示例

1. 要显示文件中包含的信息类型，请输入:

   ```
   file myfile
   ```

这将显示文件 *myfile* 的类型（例如目录、数据、ASCII 文本、C 程序源和压缩文档）。

2. 要显示在文件名列表中指定的每个文件的类型，请输入:

```
file -f filenames
```

这将显示在 `filenames` 列表中指定的每个文件的类型。每个文件必须单独出现在一行上。

注: 要从 **file** 命令获取定制的信息，请和 **-m** 标志一起使用独立的 magic 文件。不建议编辑只读的 **/etc/magic** 文件。

文件

/usr/bin/file	包含 **file** 命令。
/etc/magic	包含文件类型数据库。

相关信息

find 命令、 **ld** 命令。

《*操作系统与设备管理*》中的『文件』描述了文件、文件类型以及如何命名文件。

《*操作系统与设备管理*》中的『输入和输出重定向概述』描述了操作系统如何处理输入和输出。

《*操作系统与设备管理*》中的『文件和目录访问方式』介绍了文件所有权以及用来访问文件和目录的许可权。

---

filemon 命令

用途

监视文件系统的性能，并且报告代表逻辑文件、虚拟内存段、逻辑卷和物理卷的 I/O 活动。

语法

filemon [**-d**] [**-i** *Trace_File* **-n** *Gennames_File*] [**-o** *File*] [**-O** *Levels*] [**-P**] [**-T** *n*] [**-u**] [**-v**]

描述

filemon 命令监控文件系统和 I/O 系统事件的跟踪，并且报告一个周期内的文件和 I/O 的访问性能。

在它的一般模式中，当一个或多个应用程序或者系统命令正在被执行和监控的时候，**filemon** 命令在后台运行。**filemon** 命令自动启动并且实时监控程序的文件系统和 I/O 事件的跟踪。作为缺省值，跟踪是立刻启动的；可选的是，跟踪可能被延缓直到用户发出一个 **trcon** 命令。用户可以在 **filemon** 命令正在运行时发出 **trcoff** 和 **trcon** 命令以便按需要关闭和打开监视。当跟踪被 **trcstop** 命令中止的时候，**filemon** 命令生成一个 I/O 活动报告并退出。

filemon 命令也可以处理一个跟踪文件，这个文件已经先前被跟踪工具所记录。文件和 I/O 活动报告将会基于那个文件中记录的事件。

为了更完整的提供一个应用程序的文件系统性能的理解，**filemon** 命令以下面的四种水平来监控文件和 I/O 活动:

逻辑文件系统	**filemon** 命令监控在逻辑文件上的逻辑 I/O 操作。被监控的操作包括所有的**读**、**写**、**打开**和 **lseek** 等系统调用，这个可能会还是不会导致实际的物理 I/O，取决于这些文件是否在内存中已经被缓冲。I/O 统计信息被保存在一个基本文件中。对异步 I/O 系统调用的调用不受 **filemon** 命令监视，因此 **filemon** 逻辑文件报告不包含异步 I/O（AIO）请求。
虚拟内存系统	**filemon** 命令监控磁盘上段和它们的映像之间的物理 I/O 操作（也就是，页面调度）。I/O 统计信息被保存在一个基本段中。
逻辑卷	**filemon** 命令监控逻辑卷上的 I/O 操作。I/O 统计信息被保存在一个基本的逻辑卷中。
物理卷	**filemon** 命令监控物理卷上的 I/O 操作。在这个级别上，获取了物理资源的使用。I/O 统计信息被保存在一个基本的物理卷中。

正如命令行标志所指定的那样，这四种级别的任何组合都可以被监控。作为缺省值，**filemon** 命令只监控虚拟内存、逻辑卷和物理卷级别的 I/O 操作。这些级别都与真实的磁盘 I/O 的请求相关。

filemon 命令将它的报告写到一个标准的输出或者一个指定的文件上。这个报告以每个被监控级别的 I/O 活动的摘要作为开始，以每个被监控级别的 I/O 活动的详细统计信息为结束。摘要和详细报告内容表述在 Reports 部分中。

注:

1. **filemon** 命令产生的报告可能会相当长。因此，**-o** 选项经常被用来将报告写到一个输出文件上。当打开一个物理设备并被应用程序直接访问时，只有那些读和写的完整的 512 个字节的块会被反映到报告中。设备驱动程序为了发出设备命令和读取设备状态所使用的“简短”读和写都被忽略。光盘驱动器没有同心的“磁道”或“柱面”，象在固定文件中那样。（只有一个螺旋磁道。）因此，不可能根据柱面来报告光盘驱动器的搜索间隔统计信息。
2. **-u** 标志被用来产生先前为了启动 **trace** 守护程序而打开的文件上的报告。这个数据的一些部分可能是很有用的，但是大部分的都应用到了守护程序和其他不相关的活动中去了。这个背景信息可以是压倒性的，特别是在大的系统中。如果 **/unix** 文件和正在运行的内核不是一样的，则内核地址可能是不正确的，会导致 **filemon** 命令退出。当从一个 shell 脚本中使用 **filemon** 命令时，允许在显示 **filemon** 输出文件的内容之前有一个轻微的延迟。**filemon** 命令可能会花费几秒钟来产生这个报告。

系统跟踪工具

filemon 命令使用系统跟踪工具获得原始的 I/O 性能资料。通常地，跟踪工具只支持一个输出流。因此，只有一个 **filemon** 或者跟踪进程能在一个时间是激活的。如果另外一个 **filemon** 或者跟踪进程已经在运行，**filemon** 命令会响应以下的消息:

```
/dev/systrace: Device busy
```

当监控很强的 I/O 应用程序时，**filemon** 命令可能不能实时地处理产生的跟踪事件。当上面的情况发生时，错误消息如下:

```
Trace kernel buffers overflowed, N missed entries
```

这个消息将显示在 `stderr` 上，标志着当跟踪缓冲区满的时候到底有多少跟踪事件被丢失。**filemon** 命令将继续监控 I/O 活动，但是报告的精确性降低到了某个未知程度。一个避免溢出的方法是监控文件和 I/O 子系统的较少的级别：跟踪事件产生的数目是与监控级别的个数成比例的。另外，跟踪缓冲区的大小可以通过使用 **-t** 选项来增加，这样就能在溢出前提供更大的跟踪事件容量。记住增加跟踪缓冲区的大小会导致更多的引脚内存，并且因此可能影响 I/O 和页面调度的行为。

在存储受限的环境（要求的存储容量比可供给的内存要多），**-P** 选项可以被用来引脚内存中的实时 **filemon** 进程中的文本和资料页，这样这些页面就不会被替换掉。如果不用 **-P** 选项，允许 **filemon** 进程被替换掉，

filemon 命令的前进可能被延迟，地点是在它不能足够快地处理跟踪事件的地方。这个情况导致跟踪缓冲区如上面情况描述的那样溢出。当然，插入这个进程将从应用程序中夺走内存（尽管 **filemon** 命令不是一个大程序，但是它的进程映像也能耗用 500KB）。

在使用 **filemon** 命令去处理一个存在的跟踪数据文件前，您必须使用 **-r** 选项在 **trcrpt** 命令中去将跟踪资料顺序的重写入一个新的文件。否则，**filemon** 命令产生以下的错误消息然后退出:

```
error: run 'trcrpt -r' on logfile first
```

-i *Trace_File* 和 **-n** *Gennames_File* 标志允许跟踪数据文件的 **filemon** 脱机处理，这些文件是利用 **trace** 命令创建的。如果存在一个，则两个标志都必须指定。当必须后期处理一个来自远程机器的跟踪文件或者在一个时间执行跟踪资料收集工作而在另一个时间后期处理它的时候，这些标志是很有效的。当系统负载很大或者 **filemon** 丢失了跟踪 hook 时，这些标志也是很有用的。

gennames 文件（包括文件系统信息）必须使用在跟踪来源处的机器上。而且，在与系统跟踪文件创建接近的时刻运行 **gennames** 是明智的，这样两个系统配置就是一样的了。

与 **filemon** 相关的跟踪 hook 必须被 **trace** 命令所收集并且被 **trace -j** 标志所指定。当 **filemon** 以 **-v** 标志调用时，相关的跟踪 hook 会列出。包含 **-f** 选项的 **gennames** 命令将被执行，它的输出被保存在 *Gennames_File* 中以收集 **filemon** 的附加消息。**-f** 选项被用来和 gennames 命令一起去收集逻辑卷和物理卷的设备信息。它也用来获取脱机的 **filemon** 使用的虚拟文件系统的信息。一旦执行了 **trace** 命令，**trcrpt -r** 必须在跟踪日志文件上运行并重定向到另外一个文件中。那么该文件和 *Gennames_File* 就可能提供给 **filemon**。

报告

每个由 **filemon** 命令生成的报告有一个指示日期、机器名称和按秒计数的监控周期长度的报头。在监控周期内的 CPU 使用情况也在报告中体现。

下一个，对每个被监控的文件系统级别，都会生成摘要报告。在缺省情况下，逻辑文件和虚拟内存报告分别限制在 20 个最活跃的文件和段中，可以通过转换数据的总数量来测量。如果指定了 **-v** 标志，所有文件和段的活动都会被报告。每个报告文件、段或者卷都有一行。四个摘要报告的每一行的栏都描述如下:

最活动的文件报告

栏	描述
#MBS	到／从文件中传输的兆字节的总数量。这些行以递减的顺序按该字段排序。
#opns	在评估周期内，文件被打开的次数。
#rds	对文件的系统读取调用数目。
#wrs	对文件的系统写入调用数目。
file	文件名（完整的路径名称在详细报告中）。
volume:inode	包含文件和文件 i 节点数目的卷的名称。该字段可以用来将一个文件和它的相应的持久段联系起来，在虚拟内存 I/O 报告中显示。该字段可以是空的；例如，对于在执行过程中创建和删除的临时文件。

最活动的段报告

栏	描述
#MBS	从/到段的传输的兆字节的总共数目。这些行以递减的顺序按该字段排序。
#rpgs	从磁盘（也就是页）读入到段的那些 4096 个字节的页的数目。
#wpgs	从段写入到磁盘（到处页）的那些 4096 个字节的页的数目。
segid	段的内部标识。

segtype	输入段: 工作段、持久段（本地文件）、客户机段（远程文件）、页面表段、系统段、或者特殊的永久段，这些特殊的段包含文件系统数据（日志、根目录、.inode、.inodemap、.inodex、.inodexmap、.indirect、.diskmap）。
volume:inode	对持久的段来说，包含关联文件的卷的名字和文件的 i 节点数目。该字段可以用来将一个永久段和它的相应文件联系起来，在文件 I/O 报告中显示。该字段对非持久段来说是空的。 **注**: 虚拟内存分析工具，**svmon** 可以用来显示关于一个给定段标志（segid）的段的更多信息，如下: `svmon -S <segid>`

最活动的逻辑卷报告

栏	描述
util	卷的使用情况（忙的时间片）。这些行以递减的顺序按该字段排序。
#rblk	从卷中读入的 512 字节的块的数目。
#wblk	写入卷的 512 字节的块的数目。
KB/sec	总共的传输吞吐量，按千字节每秒。
volume	卷的名称。
description	卷的内容: 或者是一个文件系统的名字，或者是逻辑卷的类型（调页、jfslog、引导、或者系统转储）。还指示文件系统是否是片段的或者压缩的。

最活动的物理卷报告

栏	描述
util	卷的使用情况（忙的时间片）。这些行以递减的顺序按该字段排序。
#rblk	从卷中读入的 512 字节的块的数目。
#wblk	写入卷的 512 字节的块的数目。
KB/sec	总共的卷吞吐量，按千字节每秒。
volume	卷的名称。
description	卷的类型，例如: 120MB disk、355MB SCSI 或 CDROM SCSI。 **注**: 逻辑卷 I/O 请求在物理卷的 I/O 请求之前开始，之后结束。因为这个原因，总共的逻辑卷利用率看起来要比总共的物理卷利用率高一些。

最后，为每个被监控的文件系统级别都会产生详细报告。在缺省情况下，逻辑文件和虚拟内存报告分别限制在 20 个最活跃的文件和段中，可以通过转换数据的总数量来测量。如果指定了 **-v** 标志，所有文件和段的活动都报告。每个被报告的文件、段或者卷都有一个记录。

一些字段报告一个单独的值，其他的一些报告表现许多值的分布情况的统计信息。例如，对所有被监控的读和写的请求，响应时间的统计信息都会被保留。平均的、最小的和最大的响应时间和响应时间的标准差被报告出来。标准差用来显示个别响应时间偏离平均值的程度。大约有三分之二的样本响应时间是在平均值 - 标准偏差和平均值 + 标准偏差以内。如果响应时间的分布散布在较大范围中，标准偏差相比平均响应时间就会很大。在以下列表中描述了四个详细报告:

文件状态详细报告

栏	描述
FILE	文件名称。如果可能的话，给出完整的路径名称。
volume	包含文件的逻辑卷或者文件系统的名称。
inode	在文件系统中的文件的 I 节点数目。
opens	监控时打开文件的次数。
total bytes xfrd	从/到文件的读或者写操作的字节总数目。
reads	对文件的读取调用的数目。

read sizes (bytes)	按字节的读取的传输大小的统计信息（avg/min/max/sdev）。
read times (msec)	按毫秒计的读取响应时间的统计信息（avg/min/max/sdev）。
writes	对文件的写入调用的数目。
write sizes (bytes)	写入传输大小的统计信息。
write times (msec)	写入的响应时间的统计信息。
seeks	子例程调用 **lseek** 的数目。

VM 段状态的详细报告

栏	描述
SEGMENT	内部的段标识。
segtype	段内容的类型。
segment flags	不同的段属性。
volume	对永久段来说，包含相应文件的逻辑卷的名字。
inode	对持久段来说，相应文件的 i 节点数目。
reads	读入段（也就是页）的 4096 字节的页的数目。
read times (msec)	按毫秒计的读取响应时间的统计信息（avg/min/max/sdev）。
read sequences	读取序列的数目。一个序列就是被连续读入的页面的一个字符串。读取序列的数目是顺序访问数量的一个指示符。
read seq. lengths	按页面来描述读取序列长度的统计信息。
writes	从段写的页面数目。
write times (msec)	写入响应时间的统计信息。
write sequences	写入序列的数目。一个序列就是被连续写入的页面的一个字符串。
write seq.lengths	按页面描述的写入序列长度的统计信息。

逻辑卷和物理卷状态的详细报告

栏	描述
VOLUME	卷的名字。
description	卷的描述。（如果讨论一个逻辑卷则描述内容，如果处理一个物理卷则描述类型。）
reads	对卷的读取请求的数目。
read sizes (blks)	以 512 字节的块为单位的读取传输大小的统计信息（avg/min/max/sdev）。
read times (msec)	按毫秒计的读取响应时间的统计信息（avg/min/max/sdev）。
read sequences	读取序列的数目。一个序列就是能连续读入和显示顺序访问数量的 512 字节块的一个字符串。
read seq. lengths	按块描述读取序列长度的统计信息。
writes	对卷的写入请求的数目。
write sizes (blks)	写入传输大小的统计信息。
write times (msec)	写入响应时间的统计信息。
write sequences	写入序列的数目。一个序列就是被连续写入的 512 字节块的一个字符串。
write seq. lengths	按块描述写入序列长度的统计信息。
seeks	读取或者写入请求之前的搜索数目；也可以表达为需要搜索的读取和写入总数的百分比。
seek dist (blks)	以 512 字节块为单位搜索间距统计信息。除了通常的统计信息（avg/min/max/sdev）以外，初始搜索操作（假定从块 0 作为开始位置）的间距也被单独报告。这个搜索间隔有的时候会很大，因此单独报告以避免偏移其他的统计信息。
seek dist (cyls)	（只是固定文件。）以磁盘柱面为单位搜索间距的统计信息。
time to next req	按毫秒描述的时间长度的统计信息（avg/min/max/sdev），这个时间是在对卷的连续读取或者写入的请求之间。该栏显示卷的被访问率。
throughput	总共的卷吞吐量，按千字节每秒。
utilization	卷的时间片忙。在这个报告中的记录以递减的顺序按这个字段排序。

标志

-i *Trace_File*
从指定的 *Trace_File* 中读取 I/O 跟踪数据，而不是从实时的跟踪进程。**filemon** 报告概括了跟踪文件显示的系统和周期的 I/O 活动。
注: 跟踪数据文件通常以循环的方式记录。如果跟踪数据环绕舍入，则跟踪的顺序开始和结束就可能在文件的中间发生。使用 **trcrpt** 命令的原始方式来顺序重写数据，这项工作要在调用 **filemon** 命令之前执行，如下:

```
trcrpt -r file > new.file
```

为了报告能精确，跟踪文件必须包括被 **filemon** 命令所需要的所有 hook。

还必须指定 **-n** 选项。

-n *Gennames_File*
为了脱机跟踪处理而指定一个 *Gennames_File*。该文件通过运行带有 **-f** 选项的 gennames 命令来创建，并且将输出重定向到一个文件中，如下:

```
gennames -f > file
```

也必须指定 **-i** 选项。

-o *File*
写 I/O 活动报告到指定的 *File*，而不是到 **stdout** 文件。

-d
启动 **filemon** 命令，但是一直推迟跟踪直到用户执行 **trcon** 命令。作为缺省值，跟踪是立刻启动的。

-t *n*
设置内核的跟踪缓冲区大小为 *n* 字节。缺省大小为每个 CPU 64 000 字节。如果可以，缓冲区的大小可以通过提供更大的事件容量来增加。（一个典型的事件记录大小是 30 字节。）
注: 内核中的跟踪驱动程序使用双缓冲区，这样事实上就有两个大小分配为 *n* 字节的缓冲区。而且，注意这些缓冲区是插入到存储器中的，所以它们不受页面调度支配。大的缓冲区可能会影响页面调度和其他 I/O 的性能。

-P
在存储器中插入监控进程。**-P** 标志导致 **filemon** 命令的文本和数据页按监控周期的时间插入到存储器中。使用该标志可以保证当运行一个存储受限环境时，实时的 **filemon** 过程不会 page out。

-v
在报告中打印额外的信息。**-v** 标志最重要的影响是被访问的所有的逻辑文件和所有的段都包括在了 I/O 活动报告中，而不是仅仅有最多 20 个活动文件和段。

-O *Levels*
只监控指定的文件系统级别。有效的级别标识是:

lf 逻辑文件级别

vm 虚拟内存级别

lv 逻辑卷级别

pv 物理卷级别

all **lf**、**vm**、**lv** 和 **pv** 的简单表示

vm、**lv** 和 **pv** 级别都是默认的缺省值。

-u
关于将在 **trace** 守护程序的启动之前先打开的文件的报告。进程标识（PID）和文件描述符（FD）被文件名所代替。
注: 既然 PID 和 FD 都是可重用的，那么就有可能看到以相同名字的字段报告的不同文件。

示例

1. 要监控文件系统的虚拟内存、逻辑卷和物理卷级别的物理 I/O 活动，请输入:

   ```
   filemon
   ```

   filemon 命令自动启动系统跟踪并且将它放到后台。在该命令后，输入在这个时刻要运行的应用程序和系统命令，然后输入:

```
trcstop
```

在执行了 **trcstop** 命令后，I/O 活动报告就会显示在标准的输出设备上（但是可能无法滚屏）。虚拟内存的 I/O 报告会被限制在可能导致最多 I/O 的 20 个段。

2. 要按所有的文件系统级别来监控活动，并将报告写入到文件 fmon.out 中，请输入:

```
filemon -o fmon.out -O all
```

filemon 命令自动启动系统跟踪并且将它放到后台。在该命令后，输入在这个时刻要运行的应用程序和系统命令，然后输入:

```
trcstop
```

在执行了 **trcstop** 命令后，I/O 活动报告被写入 fmon.out 文件中。所有的四个级别的文件和 I/O 系统（逻辑文件、虚拟内存、逻辑卷和物理卷级别）都会被监控。逻辑文件和虚拟内存的 I/O 报告限制在导致最多 I/O 的 20 个文件和段（分别地）。

3. 要监控在所有文件系统级别上的活动，并且将一个详细的报告写到文件 fmon.out 中，请输入:

```
filemon -v -o fmon.out -O all
```

filemon 命令自动启动系统跟踪并且将它放到后台。在该命令后，输入在这个时刻要运行的应用程序和系统命令，然后输入:

```
trcstop
```

除了详细的报告是生成在文件 fmon.out 中的以外，本例和前面的一个示例是类似的。主要的区别在于 **filemon** 命令将指出它正在启动跟踪的步骤，并且摘要和详细信息的报告将包括所有导致任何的 I/O（可能有很多）的文件和段，而不是只有最多 20 个。

4. 要报告先前记录的一个跟踪会话捕获的 I/O 活动，请输入:

```
filemon -i trcfile | pg
```

在本示例中，**filemon** 命令从输入文件 trcfile 中读取文件系统跟踪的事件。输入文件必须已经是初始的跟踪格式，作为运行 **trcrpt -r** 命令的一个结果。既然跟踪数据已经在一个文件中被捕捉，**filemon** 命令就不再将它自己放置到后台以使其他的应用程序能够运行。整个文件读取后，一个关于虚拟内存、逻辑卷和物理卷级别的 I/O 活动报告将会被显示在标准输出（这种标准输出，在本例中，是通道 pg）。

5. 要只监控逻辑和物理卷的 I/O 活动，同时使用 **trcon** 和 **trcoff** 命令控制监控的间隔，请输入:

```
filemon -d -o fmon.out -O pv,lv
```

filemon 命令自动启动系统跟踪并且将它放到后台。该命令之后，输入要在这个时刻运行的不被监控的应用程序和系统命令，请输入:

```
trcon
```

在该命令后，输入要在这个时刻运行的被监控的应用程序和系统命令，请输入:

```
trcoff
```

该命令之后，输入要在这个时刻运行的不被监控的应用程序和系统命令，请输入:

```
trcon
```

在该命令后，输入要在这个时刻运行的被监控的应用程序和系统命令，请输入:

```
trcstop
```

在本示例中，**-O** 标志只被用来限制监控逻辑和物理卷。只有那些与逻辑和物理卷相关的跟踪事件才被启用。而且，作为使用 **-d** 标志的一个结果，监控最初是被延缓一直到执行了 **trcon** 命令。通过使用 **trcoff** 和 **trcon** 命令，系统跟踪可以被间断地禁用和重启用，这样就能只监控特殊的间隔。

6. 为了在脱机方式下运行 **filemon**，分别运行 **trace** 和 **gennames** 命令，然后将从那些命令中得到的输出作为 **filemon** 命令的输入，如下:

```
trace -a -T 768000 -L 10000000 -o trace.out -j 000,000,001,002,003,005,006,139,102,10C,106,00A,107,
101,104,10D,15B,12E,130,163,19C,154,3D3,1BA,1BE,1BC,10B,221,1C9,222,228,232,45B
```

运行被监控的应用程序和系统命令，请输入:

```
trcstop
```

然后格式化文件 **trace**:

```
trcrpt -r trace.out > trace.rpt
```

创建文件 **gennames**:

```
gennames -f > gennames.out
```

然后运行 **filemon** 附带着 **-i** 和 **-n** 标志:

```
filemon -i trace.rpt -n gennames.out -O all
```

相关信息

svmon 命令、**trcrpt** 命令、**trcstop** 命令。

lseek 子例程。

《*性能管理*》中的『监视磁盘 I/O』。

Fileplace 命令

用途

显示逻辑或者物理卷中文件块的放置。

语法

fileplace [{ **-l** | **-p** } [**-i**] [**-v**]] *File* | [**-m** *LogicalVolumeName*]

描述

fileplace 命令显示在包含该文件的逻辑或者物理卷中指定文件的位置。

在缺省情况下，**fileplace** 命令将被分配到指定文件的逻辑卷片段范围列出到标准输出。逻辑卷片段列出的顺序与它们在文件中的顺序直接一致。简短的开头将显示出文件大小（按字节）、文件存放处的逻辑卷名字、该卷的块大小（按字节）、分片大小（按字节）和压缩情况，显示文件系统是否经过了压缩。

有时，文件的某些部分可能不能被映射到卷的任何片段。这些区域的大小是整数数目的片段，它们可能暗中被文件系统零填充。**fileplace** 命令显示文件中没有被分配片段的那些区域。

偶尔，**fileplace** 命令也会显示如下情况:

- 统计信息显示文件在卷中分布的程度。

- 文件的间接块地址。
- 对文件的每个物理拷贝，文件在物理（与逻辑相对）卷中的位置。

注:

1. **fileplace** 命令不能显示远程网络文件系统文件的位置。如果指定远程文件，**fileplace** 命令会返回错误消息。然而，如果 **Fileplace** 命令直接在文件服务器上运行，远程文件的位置也能显示出来。
2. **fileplace** 命令直接从磁盘的逻辑卷上读取块的文件列表。当 **fileplace** 命令正在运行时，如果文件被新建、扩展或者截断，文件系统信息可能不再在磁盘上。使用 **sync** 命令可以刷新逻辑卷上的文件信息。
3. 在 JFS2 文件系统中没有间接或者双倍间接块的概念。文件根据范围表示。因此最大范围的大小取决于聚集块的尺寸。在 512 字节的聚集块大小（允许的最小值）下，最大的范围是 512*(2^24-1) 字节的长度（比 8G 稍少一点）。在 4096 字节的聚集块大小（允许的最大值）下，最大的范围是 4096*(2^24-1) 字节的长度（比 64G 略少一点）。

   这些限制只应用于单一的范围；它们在全部文件的大小上没有任何限制影响。

标志

-i
显示文件的间接块（如有的话）。这些间接块会根据其逻辑卷或者物理卷块地址来显示，取决于是否指定了 **-l** 或者 **-p** 标志。

-l
对包含文件的逻辑卷，根据逻辑卷片段来显示文件位置。**-l** 和 **-p** 标志是相互排斥的。

> **注:** 如果不指定 **-l** 或者 **-p** 标志，**-l** 标志就是缺省值。如果两个标志都被指定，则采用 **-p** 标志。

-m *LogicalVolumeName*
显示对逻辑卷的逻辑到物理的映射。

-p
对于包含文件的物理卷来说，根据下面的物理卷显示文件位置。如果对包含文件的逻辑卷进行镜像，则每个镜像拷贝的物理位置都会显示。**-l** 和 **-p**标志是相互排斥的。

-v
显示文件及其位置的更多信息，包括对文件跨卷传播的宽度和卷中文件破碎程度的统计信息。根据逻辑卷或者物理卷的片段数量来表示这种统计，取决于是否指定了 **-l** 或者 **-p** 标志。

文件的空间效率这样计算：非空片段的数目（N）除以赋予该文件的片段范围（R），再乘以 100，或者表示为（*N*/*R*）*100。计算范围的方法是：用最高分配地址减去最低分配地址然后再加 1，或者表示为 *MaxBlk-MinBlk* + 1。例如，文件写入的逻辑块是从 01550 到 01557，所以 *N* 等于 8。*R*（01557 - 01550 +1）也等于 8，该文件的空间效率为 100% 或 8/8 x 100。这个文件的空间效率就是 100%，或者表示为 8/8 * 100。**-v** 标志消息会打印算式（*N*/*R*）+100 的结果。

依照这种计算效率的办法，大于 32KB 的文件会因为其间接块的使用而永远达不到 100% 的效率。

连续效率的定义是 1 减去间隙数目（*nG*）除以可能的间隙数目（*nPG*），或者表示为 1 -（*nG*/*nPG*）。可能的间歇数目等于 *N* 减去 1（*nPG*=*N* - 1）。如果该文件被写到 9 块（大于 32KB），逻辑片段栏就会显示:

```
01550 - 01557
01600
```

该文件存储在可能的 9 个片段中的 2 个。该文件的连续率计算方式为:

```
nG =1
nPG =9-1=8
(1-1/8)*100=87.5%
```

示例

1. 要显示文件在其逻辑卷中的位置，请输入:

   ```
   fileplace data1
   ```

   本示例显示包含文件 data1 的逻辑卷和片段的列表。
2. 要显示一个文件的间接块，请输入:

   ```
   fileplace -i data1
   ```

   除了逻辑卷片段的缺省列表外，被用来在文件系统中存储文件块地址的间接块（如有的话）也会被列举出来。
3. 要显示更多有关文件位置的信息，请输入:

   ```
   fileplace -v data1
   ```

   除了逻辑卷片段的缺省列表外，关于位置效率的统计情况也会显示。
4. 要显示文件在其物理卷中放置的所有信息，请输入:

   ```
   fileplace -piv data1
   ```

   本例将根据下面的物理卷来显示文件列表和间接块，同时显示的还包括位置效率的统计情况。

文件

/dev/hd0、**/dev/hd1**，**.../dev/hd***n* 指定逻辑卷。

相关信息

sync 命令。

《*性能管理*》中的『监视磁盘 I/O』。

《*操作系统与设备管理*》中的『逻辑卷存储器』定义并讨论了逻辑卷存储器。

---

find 命令

用途

用匹配表达式查找文件。

语法

find [**-H** | **-L**] *Path* ... [*Expression*]

描述

find 命令对每个指定的 *Path* 参数递归搜索目录树，搜索与布尔表达式匹配的文件，布尔表达式是使用以下文本中给出的项写的。当 **find** 命令递归下降目录结构时，它不会下降到那些只是象征性链接到当前层次结构的目录。**find** 命令的输出取决于 *Expression* 参数指定的项。

find 命令不支持 4.3 BSD 快速查找语法。

标志

-H 导致为命令行上多次遇到的每个符号链接求值的文件信息和文件类型成为链接引用的文件的信息和类型，而不是链接本身的。如果引用的文件不存在，则文件信息与类型将用于链接本身。不在命令行上的所有符号链接的文件信息将是链接本身的信息。

-L 导致为每个符号链接求值的文件信息和文件类型成为链接引用的文件的信息和类型，而不是链接本身的。

表达式项

这些布尔表达式和变量描述了 **find** 命令的搜索边界，如在 *Path* 和 *Expression* 参数定义的那样。

注: 在下面的定义中，变量 *n* 指定了一个十进制的整数，它能表示为 **+***n*（超过 *n*）、**-***n*（少于 *n*）或者 *n*（恰好等于 *n*）。

\(*Expression* **\)**
如果括号中的表达式为 True，则求值为 true。

-cpio *Device*
将当前的文件写入到在 **cpio** 命令格式中的指定设备。

-depth
始终求值为 True。导致目录层次下降，这样该目录里的所有条目在该目录本身受影响之前被影响。当 **find** 命令和 **cpio** 命令一起被用来传输那些包含在无写权限的目录中的文件时，这是很有用的。

-ea
如果文件具有访问控制信息（ACL）或扩展属性（EA）集，则求值为 True。

-exec *Command*
如果被指定的命令运行或者返回 0 值作为退出状态时，求值为 True。指定命令的末尾必须标有括在引号内的分号、转义的分号或者加号。包含两个字符 {}（花括号）的参数后面必须跟有标在指定命令末尾处的加号。命令参数 {}（花括号）将由当前路径名替换。

-follow
导致符号的和硬链接被跟随:

-fstype *Type*
如果包含文件的文件系统是一个指定的类型，其中 *Type* 变量有一个 **jfs**（日志文件系统）或者 **nfs**（网络文件系统）的值，求值为 True。

-group *Group*
求值为 True，如果文件属于指定的组。如果 *Group* 变量的值是数字，并且没有在 **/etc/group** 文件中出现，它就被解释成组标识。

-nogroup
求值为 True，如果文件属于一个组但是不在 **/etc/group** 数据库中。

-inum *n*
求值为 True，如果文件有一个索引节点与变量 *n* 的值匹配。

-links *n*
求值为 True，如果文件有指定数目的链接。请参考 **ln** 命令中关于链接的描述。

-long
当结合 **-ls** 使用时，打印每个用户／组名称的全部可用字符，而不是截断至前 8 个字符。

-ls
始终求值为 True。导致当前路径名与其关联统计信息一起被打印出来。这些统计信息包括以下内容:

- 索引节点数目
- 按千字节计的大小（1024 字节）
- 保护方式
- 硬链接的数目
- 用户
- 组
- 按字节计的大小
- 修改时间

如果这个文件是特殊文件，则大小字段将包括主要的和次要的设备号。如果该文件是符号链接，则打印该链接到的文件的路径名时前面有 ->（连字符，大于）符号。格式化与 **ls -filds** 命令的相类似，然而格式化在内部执行，不执行 **ls** 命令，因此和 **ls** 命令在输出中的不同可能存在，例如保护方式。

-name *File*
求值为 True，如果 *File* 变量的值与文件名称匹配。可以使用通常的 shell 文件名生成字符（请参阅 **sh** 命令）。当从 shell 使用 **find** 命令时，模式应该括在引号中或者转义字符中。反斜杠符号 （\） 在模式中会被用作一个转义字符。您可以使用通配符（模式匹配），只要它们在引号以内。有关使用通配符的更多信息，请参阅《*操作系统与设备管理*》中的『带有通配符和元字符的模式匹配』。

在诸如 [a-z] 的表达式中，连字符表示依照当前的整理顺序 *通过*。整理顺序可以定义等价的类以供在字符范围中使用。有关整理顺序和等价类的更多信息，请参阅《*AIX 5L V5.3 本地语言支持指南和参考大全*》中的『本地语言支持概述』。

-newer *File*
求值为 True，如果修改当前文件的时间比 *File* 变量指定的更近。

-ok *Command*
和 **-exec** 表达式一样，除了 **find** 命令询问您是否应当启动指定的命令。一个肯定的响应会启动这个命令。指定命令的末尾必须加上括在引号中的分号或 \;（反斜杠转义分号）。

-perm [**-**] *OctalNumber*
求值为 True，如果文件的许可权代码恰好与 *OctalNumber* 参数匹配（请参阅 **chmod** 命令以获取文件许可权的解释）。如果可选的 **-**（连字符）存在，并且至少已设置了这些许可权，则这个表达式的求值为 true。*OctalNumber* 参数可达到 9 个八进制数。

-perm [**-**] *Mode*
方式参数用于表示文件的模式位。它的格式和 **chmod** 描述的符号方式 < 操作数一样，并且如下解释:

最初，假定模板清除了所有文件方式位。Op 符号有以下功能:

+ 在模板中设置相应的方式位

- 清除相应的位

= 设置相应的方式位，不考虑进程文件模式创建掩码的内容。

op 符号 - 不能是方式的首字母。这避免了前导连字符的不明确性。因为最初的方式是所有的位都关闭，所以没有需要使用 - 作为首字母的符号方式。

如果连字符省略了，当文件许可位恰好与结果模板的值匹配时，最初求值为 True。否则，如果方式以连字符作为前缀，则当至少结果模板中的所有的位都在文件许可位中设置时，最初求值为 True。

Mode 参数与 **chmod** 命令的语法是一样的。如果文件恰好有这些许可权，这个表达式求值为 True。如果可选的 **-**（连字符）存在，则当至少设置了这些许可时，这个表达式求值为 True。

-print
始终求值为 True。显示当前路径名。**find** 命令采用 **-print** 表达式，除非存在 **-exec, - ls** 或 **-ok** 表达式。

-prune
始终求值为 True。如果当前路径名是一个目录，停止它的下降。如果指定 **-depth** 标志，则忽略 **-prune** 标志。

-size *n*
如果文件是指定的 *n* 个块长（每块 512 字节），则求值为 True。为了比较，文件大小四舍五入到相比差异最小的块的大小。

-size *n***c**
如果文件恰好是指定的 *n* 个字节长，则求值为 True。将 **c** 添加到变量 *n* 的末尾表示文件的大小是以单独的字节而非块来测量的。

-atime *n*
如果初始化时间减去文件访问时间，除以 86400 秒（废弃任何余数）等于 *n*，则求值为 True。86400 秒为 24 小时。

注: 已更改 **-atime** 的定义来符合 Single UNIX Specification V3。如果文件在 24 小时的 *n*-1 到 *n* 的倍数的时间里被访问，则 **-atime** 的先前行为的求值为 True。在缺省情况下，**find -atime** 按照好像它确实在 UNIX03 前面那样起作用。可以通过将环境变量 **XPG_SUS_ENV** 设置为 `ON` 以及将 **XPG_UNIX98** 设置为 `OFF` 来获取 UNIX03 行为。

可通过将 **XPG_UNIX98** 变量设置为 `ON` 来获得此选项的先前行为。

-ctime *n*
如果文件的索引节点（状态信息）在 24 小时周期的指定数目内被改动，则求值为 True。

-mtime *n*
如果初始化时间减去文件修改时间，除以 86400 秒（废弃任何余数）等于 *n*，则求值为 True。86400 秒为 24 小时。

注: 已更改 **-mtime** 的定义来符合 Single UNIX Specification V3。如果文件在 24 小时的 *n*-1 到 *n* 的倍数的时间里被修改，则 **-mtime** 的先前行为的求值为 True。在缺省情况下，**find -mtime** 按照好像它确实在 UNIX03 前面那样起作用。可以通过将环境变量 **XPG_SUS_ENV** 设置为 `ON` 以及将 **XPG_UNIX98** 设置为 `OFF` 来获取 UNIX03 行为。

可通过将 **XPG_UNIX98** 变量设置为 `ON` 来获得此选项的先前行为。

-amin *Number*
如果在 *Number*-1 到 *Number* 分钟内已访问文件，则求值为 True。例如，如果文件在 1 到 2 分钟内已被访问，则 **-amin 2** 为 true。

-cmin *Number*
如果在指定分钟数内更改文件索引节点（状态信息），则求值为 True。

-mmin *Number*
如果在 *Number*-1 到 *Number* 分钟中已修改文件，则求值为 True

-type *Type*
如果 Type 变量指定了以下值之一，则求值为 True:

b 块特殊文件

c 字符特别文件

d 目录

f 无格式文件

l 符号链接

p FIFO（指定的管道）

s 套接字

-user *User*
如果文件属于指定用户，则求值为 True。如果 *User* 变量的值是数字，并且不作为登录名字在 **/etc/passwd** 文件中出现，它解释为用户标识。

-nouser
如果文件属于不在 **/etc/passwd** 数据库中的用户，求值为 True。

-xdev
始终求值为 True。避免 **find** 命令从一个与 *Path* 参数指定的不同的文件系统中遍历。

以下表达式可通过使用以下优先权降序排列的运算符组合在一起:

1. （*表达式*）一个表达式和运算符的附加组（括号对于 shell 是特定的，并且需要反斜杠转义序列）。
2. **!** *表达式* — 某个表达式取反（‘**!**’是一元“非”运算符）。
3. *表达式* [**-a**] *表达式* — 表达式的串联（AND 运算由两个初始量的并列默示或者明确的以 **-a** 来规定）。
4. *表达式* **-o** *表达式* — 初始量的交替；**-o** 是 OR 运算符。如果第一个表达式是 true，就不会对第二个表达式求值。

注: 当一起使用 **find** 和 **cpio** 命令的时候，当将 `-L` 选项与 **cpio** 命令一起使用时，您必须将 `-follow` 选项与 **cpio** 命令一起使用，反之亦然。不一起使用这两个选项会造成不期望的结果。如果没有表达式，`-print` 就用作缺省表达式。例如，如果给定的表达式不包含任何初始量 `-exec`、`-o k` 或 `-print`，则那些给定的表达式将被替换为（*给定表达式*）`-print`。`-user`、 `-group` 和 `-newer` 每个初始量只计算一次它们各自的参数。使用以参数 `-exec` 或者 `-ok` 指定的命令不会影响到同一个文件中的后续初始量。

退出状态

此命令返回以下出口值:

0 所有的 *Path* 参数都被成功遍历。

>0 发生错误。

示例

1. 要用一个给定的基本文件名称来列出文件系统中的所有文件，请输入:

   ```
   find /  -name .profile -print
   ```

   该命令将搜索整个文件系统并且写出以 **.profile** 命名的所有文件的完整路径名称。/（斜线）指示 **find** 命令搜索根目录及其所有子目录。要节约时间，最好通过指定您认为那些文件可能存在的文件目录的方式来限定搜索。
2. 要列出当前目录树中有特殊许可权代码的那些文件，请输入:

   ```
   find . -perm 0600  -print
   ```

   该命令会列举出那些所有者才有读、写权限的文件的名字。.（点）让 **find** 命令搜索当前目录及其子目录。请参阅 **chmod** 命令以获取许可权代码的说明。
3. 要用特定的许可权代码来搜索几个目录中的文件，请输入:

   ```
   find manual clients proposals -perm -0600  -print
   ```

   该命令将列举出那些有所有者读写权限和其他可能的权限的文件名称。搜索 manual、clients 和proposals 目录和它们的子目录。在前一个示例中，-perm 0600 只选择那些许可权代码完全与 0600 匹配的文件。在此例中，-perm -0600 选择具有允许 0600 和其他高于 0600 级别的访问的许可权代码的文件。这也与许可权代码 0622 和 2744 匹配。
4. 要列出当前目录中在当前 24 小时的时间内更改过的所有文件，请输入:

   ```
   find .  -ctime 1  -print
   ```
5. 要搜索那些有多个链接的常规文件，请输入:

   ```
   find .  -type f  -links +1  -print
   ```

   该命令将列举出那些有超过一个链接（-links+1）的普通文件的名字（-type f）。

   注: 每个目录至少有两个链接: 它父目录的入口链接和它自己的 .（点）条目。**ln** 命令解释多个文件链接。
6. 要找到那些路径名称包含 **find** 的所有可访问的文件，请输入:

   ```
   find .  -name '*find*' -print
   ```
7. 要除去所有那些以 a.out 或者 *.o 命名的文件，这些文件有一周没被访问并且不是使用 **nfs** 安装的，请输入:

   ```
   find / \( -name a.out -o -name '*.o' \) -atime +7 ! -fstype nfs -exec rm {} \;
   ```

   注: 在表达式 **-atime** 中使用的数是 +7。如果您想让这个命令在那些超过一个星期（7 个 24 小时）没被访问的文件上运行，这就是正确的命令行。
8. 要打印在当前目录或者低于当前目录的所有文件的路径名称，这其中不包括名为 SCCS 或者在 SCCS 目录中的目录，请输入:

   ```
   find .  -name SCCS  -prune -o -print
   ```

要打印在当前目录或者低于当前目录的所有文件的路径名，并且包括在名为 **SCCS** 的目录中的文件，请输入:

```
find . -print -name SCCS -prune
```

9. 要搜索那些恰好是 414 个字节长的所有文件，请输入:

   ```
   find . -size 414c -print
   ```

10. 要找到并删除在您的主目录中后缀为 **.c** 的每个文件，请输入:

    ```
    find /u/arnold -name "*.c" -exec rm {} \;
    ```

    每次 **find** 命令识别一个带有后缀名 **.c** 的文件，然后 **rm** 命令删除这个文件。**rm** 命令是为表达式 **-exec** 指定的唯一参数。{}（花括号） 表示当前的路径名称。

11. 在此示例中，`dirlink` 是到目录 `dir` 的符号链接。可通过在命令行引用符号链接 `dirlink` 列出 `dir` 中的文件。要进行此操作，请输入:

    ```
    find -H dirlink -print
    ```

12. 在此示例中，`dirlink` 是到目录 `dir` 的符号链接。要列出 `dirlink` 中遍历 `dir` 下包含任何符号链接的文件层次结构的文件，请输入:

    ```
    find -L dirlink -print
    ```

13. 要确定符号链接 `dirlink` 引用的文件 `dir1` 是否比 `dir2` 新，请输入:

    ```
    find -H dirlink -newer dir2
    ```

    注: 因为使用了 **-H** 标志，时间数据不从 `dirlink` 收集，而从 `dir1`（遍历符号链接时找到此文件）收集。

14. 要以带有扩展用户和组名称的 **ls** 格式生成当前目录中文件的列表，请输入:

    ```
    find . -ls -long
    ```

15. 要列出当前目录中带有 ACL/EA 集的文件，请输入:

    ```
    find . -ea
    ```

文件

/usr/bin/find	包含 **find** 命令。
/bin/find	到 **find** 命令上的符号链接。
/etc/group	包含所有已知组的列表。
/etc/passwd	包含所有已知用户的列表。

相关信息

chmod 命令、**cpio** 命令、**ln** 命令、**sh** 命令。

《*操作系统与设备管理*》中的『备份方法』介绍了归档方法，包括 **cpio** 命令的使用。

《*操作系统与设备管理*》中的『目录』 描述了文件系统中目录的结构和特征。

《*操作系统与设备管理*》中的『文件类型』描述了文件、文件类型、如何命名文件以及如何使用通配符。

《*操作系统与设备管理*》中的『输入和输出重定向』描述了操作系统如何处理输入和输出。

《*操作系统与设备管理*》中的『Shell』描述了 shell、不同类型的 shell，以及 shell 如何影响解释命令的方式。

《*操作系统与设备管理*》中的『文件和目录访问方式』介绍了文件所有权以及用来访问文件和目录的许可权。

finger 命令

用途

显示用户信息。这个命令和 **f** 命令是一样的。

语法

{ **finger** | **f** }[[**-b**][**-h**] [**-l**][**-p**]][[**-i**][**-q**][**-s**][**-w**]]

[**-f**][**-m**][*User*| *User* **@***Host*| **@***Host*]

描述

/usr/bin/finger 命令显示当前登录到主机的用户信息。输出格式随着显示信息的选项而更改。

缺省格式

缺省格式包括以下的条目:

- 登录名字
- 完整的用户名
- 终端名称
- 写入状态（在终端名称前加一个 *（星号）表示禁止写权限）

对主机的每个用户来说，如果已知，缺省信息列表也包括如下条目:

- 空闲时间（如果是一个单独的整数，空闲时间就是几分钟，如果存在一个 “:” （冒号），空闲时间就是几小时几分钟，如果存在一个 “d” ，空闲时间就是几天几小时）。
- 登录时间
- 位置特定的信息

位置特定的信息从 **/etc/passwd** 文件中的 gecos 区去检索。gecos 区可能包括后面跟着一个逗号或者 /（斜线符号）的完整用户名称。Finger 命令将利用位置特定的信息来显示那些跟随在逗号或者斜线符号后面的所有信息。

长格式

无论何时给定用户名列表，**finger** 命令都会采用较长格式。（帐户名称以及用户的名和姓都会被接受。）这种格式是多行的，并且包括所有上面的和如下描述的信息:

- 用户的**主**目录
- 用户的登录 shell
- 在用户主目录中的 **.plan** 文件内容
- 在用户主目录中的 **.project** 文件内容

finger 命令可能也用在查找远程系统中的用户。格式是指定用户为 *User@Host*。如果您省略用户名，**finger** 命令将在远程系统中提供标准格式的列表。

用您偏爱的文本编辑器创建 **.plan** 和 **.project** 文件并且将这些文件放到您的主目录中。当显示 **.plan** 和 **.project** 文件的内容时，**finger** 命令使用 **toascii** 子例程转换常规 ASCII 字符范围外的字符。**finger** 命令在每一个被转换的字符前面显示一个 M-。

当您用 *User* 参数来指定用户的时候，您可以指定用户的名或者姓或者帐户名。当您指定用户时，在指定的主机上，**finger** 命令将只以长格式的形式返回这些用户的信息。

有关 **finger** 命令的其他信息，请参阅《*网络与通信管理*》中的『TCP/IP 的安装』。

标志

-b 给出一个简短、长格式的列表。
-f 禁止在输出中打印报头行（定义了将被显示的字段的第一行）。
-h 禁止以长格式和简短长格式打印 **.project** 文件。
-i 给出空闲时间的一个快速列表。
-l 给出一个长格式的列表。
-m 假定*用户* 参数指定了一个用户标识（用以任意的访问控制），*不是*一个用户的登录名称。
-p 禁止以长格式和简短长格式类型打印 **.plan** 文件。
-q 给出一个快速的列表。
-s 给出一个短格式列表。
-w 给出一个狭窄的、短格式的列表。

参数

@Host 指定远程主机上的所有登录进入的用户。
User 指定一个本地用户标识（用于任意的访问控制）或者本地的用户登录名称，正如在 **/etc/passwd** 文件中指定的一样。
User@Host 在远程主机上指定一个以长格式显示的用户标识。

示例

1. 要得到所有登录到主机 `alcatraz` 上的用户信息，请输入:

   ```
   finger @alcatraz
   ```

   就会显示与以下类似的信息:

   ```
   [alcatraz.austin.ibm.com]
   Login     Name         TTY Idle          When      Site Info
   brown    Bob Brown   console   2d   Mar 15 13:19
   smith    Susan Smith    pts0  11:   Mar 15 13:01
   jones    Joe Jones      tty0    3   Mar 15 13:01
   ```

   用户 `brown` 在控制台登录，用户 `smith` 从伪的电传线路 `pts0` 上登录，用户 `jones` 从 `tty0` 上登录。

2. 要得到关于用户 `brown` 在 `alcatraz` 上的信息，请输入:

   ```
   finger brown@alcatraz
   ```

   就会显示与以下类似的信息:

   ```
   Login name: brown
   Directory: /home/brown    Shell: /home/bin/xinit -L -n Startup
   On since May 8 07:13:49 on console
   No Plan.
   ```

3. 要用简短格式得到用户 `brown` 在本地机上的信息，请输入:

   ```
   finger -q brown
   ```

   就会显示与以下类似的信息:

   ```
   Login           TTY             When
   brown           pts/6           Mon Dec1710:58
   ```

文件

/usr/bin/finger	包含 **finger** 命令。
/etc/utmp	包含当前登录的用户列表。
/etc/passwd	定义用户的帐户、名称和主目录。
/etc/security/passwd	定义用户密码。
/var/adm/lastlog	包含过去的登录时间。
$HOME/.plan	可选的包含一个用户计划的一行描述的文件。
$HOME/.project	可选的包含一个用户的计划任务的文件。

相关信息

hostname 命令，**rwho** 命令。

fingerd 守护程序。

《*网络与通信管理*》中的『显示有关已登录用户的信息的命令』。

《*网络与通信管理*》中的『通信和网络』。

---

fingerd 守护程序

用途

为 **finger** 命令提供服务器功能。

语法

注: **fingerd** 守护程序通常被 **inetd** 守护程序启动。使用系统资源控制器（SRC）命令也能通过命令行控制它。

/usr/sbin/fingerd [**-s**] [**-f**]

描述

/usr/sbin/fingerd 守护程序是一个简单的协议，它给在几个网络站点里的 **finger** 命令提供接口。**finger** 命令返回一个关于当前系统或者一个用户的状态报告。**fingerd** 守护程序侦听在 **/etc/services** 文件和文件 **/etc/inetd.conf** 中列出的端口 79 处的传输控制协议（TCP）请求。

对个别的涉及到 **fingerd** 守护程序的安全性站点，作为缺省值，将不会转发任何的 **finger** 请求到其他的系统。如果它接受到一个 **finger** 转发的请求，**fingerd** 守护程序用 `Finger` 转发服务被禁止的信息来发到 **finger** 命令上。当使用 **-f** 标志运行 **fingerd** 守护程序的时候，系统管理员可以有打开 finger 转发作为缺省值的选项。

fingerd 守护程序的改动可以通过使用系统管理界面程序（SMIT）或者 SRC 或者通过编辑 **/etc/inetd.conf** 文件或者 **etc/services** 文件来实现。不推荐在命令行输入 `fingerd`。当在 **/etc/inetd.conf** 文件未注释时，**fingerd** 守护程序缺省启动。

inetd 守护程序从 **/etc/inetd.conf** 文件和 **/etc/services** 文件中得到它的信息。

在更改了 **/etc/inetd.conf** 或 **/etc/services** 文件后，运行 **refresh -s inetd** 或者 **kill-1***InetdPID* 命令以告知 **inetd** 守护程序的更改到它的配置文件。

fingerd 守护程序应当有一个用户标识附带着至少的可能特权。**nobody** 标识允许最少的许可。赋予 **fingerd** 守护程序 **nobody**的用户标识后，它将允许守护程序能在您的主机上使用。更改 **/etc/services** 文件从而反映您想使用的用户标识。

利用系统资源控制器来操作 fingerd 守护程序

fingerd 守护程序是 **inetd** 守护程序的一个子服务器，它是 SRC 的一个子系统。**fingerd** 守护程序是 **tcpip** SRC 子系统组的一个成员。当这个守护程序没有在文件 **/etc/inetd.conf** 中注释时，它被启用，并且可以通过下面的 SRC 命令来操作:

startsrc	启动一个子系统、子系统组，或是一个子服务器。
stopsrc	终止子系统、子系统组或子服务器。
lssrc	获取一个子系统、子系统组或者一个子服务器的状态。

标志

-s 打开套接字级别调试。

-f 打开这个 **fingerd** 守护程序的 finger 转发服务。

示例

注: 通过使用 SMIT 或者通过编辑 **/etc/inetd.conf** 文件，**fingerd** 守护程序的参数可以被指定。

1. 要启动 **fingerd** 守护程序，请输入:

   ```
   startsrc -t finger
   ```

   这个命令启动 **fingerd** 子服务器。
2. 要停止 **fingerd** 通常的守护程序，请输入:

   ```
   stopsrc -t finger
   ```

   这个命令允许所有的暂挂连接开始和存在连接完成，但是禁止新的连接开始。
3. 要强行中止 **fingerd** 守护程序和所有的 **fingerd** 连接，请输入:

   ```
   stopsrc -t -f finger
   ```

   这个命令立刻中止所有的暂挂连接和存在连接。
4. 要显示一个关于 **fingerd** 守护程序的简短状态报告，请输入:

   ```
   lssrc -t finger
   ```

   这个命令返回守护程序的名字、进程标识和状态（激活的和没有激活的）。

相关信息

finger 命令、**lssrc** 命令、**kill** 命令、**refresh** 命令、**startsrc** 命令、**stopsrc** 命令。

《*网络与通信管理*》中的『TCP/IP 守护程序』。

/etc/inetd.conf 文件格式，**/etc/services** 文件格式。

有关安装基于 Web 的系统管理器安装的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

fish 命令

用途

玩钓鱼纸牌游戏

语法

fish

描述

“去钓鱼”游戏的目标是累积有四张同样面值的卡片的记录。您和电脑（您的对手）轮流从对方手里请求卡片。如果您的对手有一张或者多张所需面值的卡片，他必须将这些卡片放出来。如果没有，他就要提示“去钓鱼！”，于是您可以从公共的卡片里拿一张卡片。如果您拿到了您请求的卡片，您再拿一张。记录过后，这些卡片就被放到桌子上。当没有卡片剩余时游戏结束。拥有最多记录的玩家就赢得了这个游戏。**fish** 命令告诉您获胜方然后退出。

游戏开始前，**fish** 命令提示 `instructions?`。要查看说明，请输入 `Y`（是）。

作为您的第一步，输入一个 `p`，游戏水平变为职业级。缺省的游戏水平是业余级。

玩“去钓鱼”的游戏时，当您的对手提示:

```
you ask me for:
```

如果在提示时只按 Enter 键，您就可以得到在您的对手手上和公用卡片区的卡片数目的信息。

游戏显示:

- 您的当前情况，包括您已经累积的记录。
- “去钓鱼！”当您和您的对手要的卡片都不在对方手里时。
- “去钓鱼！”的提示后卡片被拿出。
- 您的对手向您要的卡片。
- 完全的记录（您的和您的对手的）。
- 当您或者您的对手获取另一个机会时请求的卡片。

示例

下面是一个 fish 屏幕显示的示例:

```
your hand is: A 5 5 7 10 J Q
you ask me for: 5
I say "GO FISH!"
You draw A
I ask you for: 5
Made a book of 5's
I get another guess
I ask you for 6
You say "GO FISH!"
your hand is: A A 7 10 J Q
you ask me for:
```

在游戏完成前退出，请按中断键（Ctrl-C）。

文件

/usr/games 系统游戏的位置。

相关信息

arithmetic 命令、**back** 命令、**bj** 命令、**craps** 命令、**fortune** 命令、**hangman** 命令、**moo** 命令、**number** 命令、**quiz** 命令、**ttt** 命令、**turnoff** 命令、**turnon** 命令和 **wump** 命令。

---

flcopy 命令

用途

从软盘和到软盘的拷贝。

语法

flcopy [**-f** *Device*] [**-h** | **-r**] [**-t** *Number*]

描述

flcopy 命令复制一个软盘（以 **/dev/rfd0** 打开）到一个名为 **floppy** 的在当前目录中创建的文件中，然后打印消息：更换 floppy，完成后点击回车。**flcopy** 命令于是复制 **floppy** 文件到软盘上。您可以指定 **-f**、**-h**、**-r** 或者 **-t** *Number* 标志来更改 **flcopy** 命令的行为。

注: 您不能使用 **flcopy** 命令来将数据从一个软盘复制到另一个不同大小的软盘上。

标志

-f *Device* 允许您指定不同于 **/dev/rfd0** 的驱动器。
-h 导致 **flcopy** 命令在当前的目录中打开 **floppy** 文件并且将它复制到 **/dev/rfd0**。
-r 告诉 **flcopy** 命令在复制软盘到当前目录的 **floppy** 文件后退出。
-t *数目* 只导致被指定了 *Number* 条磁道被复制。复制后的磁道总在软盘的第一个磁道开始。

退出状态

此命令返回以下出口值:

0 成功结束。
>0 发生错误。

示例

1. 要复制 **/dev/rfd1** 到当前目录的 **floppy** 文件，请输入:

```
flcopy -f/dev/rfd1 -r
```

2. 要复制软盘的开始的 100 个磁道，请输入:

```
flcopy -f/dev/rfd1 -t100
```

文件

/usr/sbin/flcopy 包含 **flcopy** 命令。

相关信息

format 或者 **fdformat** 命令。

fd 特殊的文件。

---

flush-secldapclntd 命令

用途

flush-secldapclntd 命令为 **secldapclntd** 守护程序进程刷新高速缓存。

语法

//usr/sbin/flush-secldapclntd

描述

flush-secldapclntd 命令为 **secldapclntd** 守护程序进程清除高速缓存。

示例

1. 要刷新 **secldapclntd** 守护程序的高速缓存，请输入:

   ```
   /usr/sbin/flush-secldapclntd
   ```

文件

/etc/security/ldap/ldap.cfg 包含 **secldapclntd** 守护程序连接到服务器所需要的信息。

相关信息

secldapclntd 守护程序

mksecldap、**stop-secldapclntd**、**start-secldapclntd**、**restart-secldapclntd** 和 **ls-secldapclntd** 命令。

/etc/security/ldap/ldap.cfg 文件。

---

fmt 命令

用途

在发送之前格式化邮件消息。

语法

/usr/bin/fmt [**-***Width*] [*File ...*]

描述

fmt 命令启动一个文本格式化程序来读取输入文件的并置（或者是标准输入，如果没有指定文件），然后在标准输出中产生一个行长度设置为 **-***Width* 值的输入版本。如果没有指定 **-** *Width* 标志的值，将使用 72 个字符的缺省值。作为单词之间的空行和空格，在输入行开头的空格将保持在输出中。

fmt 命令一般被用在格式化邮件消息，这样能在它们被发送之前改善外观。然而，**fmt** 命令也可以用在简单的格式化任务中。例如，在一个如 vi 编辑器的文本编辑程序的可见方式中，命令 **!}fmt** 将格式化一个段落，这样所有的行就都被设置为用 **-***width* 标志指定的宽度值。如果没有用-*Width* 标志指定宽度值，将使用 72 字符作为缺省值。标准文本编辑程序比 **fmt** 命令更适合复杂的格式化操作。

> **注**: 请勿使用 **fmt** 命令，如果消息包含别的文件上的嵌入消息或者预格式化信息。这个命令格式化嵌入消息的标题信息，并且可能更改预格式化信息的格式。

标志

File 指定要格式化的文件的名字
-*宽度* 指定行的长度。*width* 的缺省值是 72 个字符。

示例

1. 要格式化一个您用邮件编辑器创建的消息，请输入:

   ```
   ~| fmt
   ```

   ~| 在消息的左边空白处被输入。在您发出 ~| fmt 命令后，消息被格式化了。单词 (continue) 会显示出来，表示您可以输入更多的信息或者发送这个消息。
2. 要格式化一个文件并且将输出显示在您的屏幕上，请输入:

   ```
   fmt file1
   ```

   在本示例中，文件 file1 被格式化并且显示在您的屏幕上。

文件

/usr/bin/fmt 包含 **fmt** 命令。

相关信息

mail 命令、**nroff** 命令、**vi** 命令。

《*网络与通信管理*》中的『邮件应用程序』。

---

fold 命令

用途

为固定宽度的输出设备折叠长行。

语法

fold [**-b**] [**-s**] [**-w** *Width*] [*File...*]

描述

fold 命令是折叠有限宽度的输出设备的长行的过滤器。作为缺省值，该命令折叠标准输入的内容，阻断那些达到 80 行宽的行。您也可以指定一个或者多个文件作为该命令的输入。

fold 命令在输入行中插入一个换行字符，这样每个输出行就可以尽可能的宽而不超过设定的*Width* 参数值。如果指定了 **-b** 标志，行宽就可以按字节来计数。如果没有指定 **-b** 标志:

- *宽度* 按照被 **LC_CTYPE** 环境变量所决定的列来计数。
- 一个退格字符减少输出行的长度 1。
- 一个制表符跳到下一个列，它的位置是列增加 8。

fold 命令接受在包含制表符的文件中 8 的倍数的 **-w** *宽度*值。当文件包含制表符时，要用其他宽度值，应该在使用 **fold** 命令以前使用 **expand** 命令。

注:

1. **fold** 命令可能影响当前的下划线。
2. **fold** 命令不能在多字节的字符中间插入换行字符，即使使用 **-b** 标志也不行。

标志

-b 按字节计数*宽度*。缺省值是按列计数。

-s 当最右面的空格是在*宽度*限制之内，在空格后阻断该行，如果一个输出行段包含任何空字符。缺省值是阻断行使得每一个输出行段都尽可能宽。

-w *Width* 以变量 *Width* 的值指定最大行宽。缺省值为 80。

退出状态

此命令返回以下出口值:

0 所有输入文件都成功的处理。

>0 发生错误。

示例

要将一个名为 `longlines` 的文件的行折叠成宽度为 72，请输入:

```
fold -w 72 longlines
```

文件

/usr/bin/fold 包含 **fold** 命令。

相关信息

expand 命令、**tab** 命令。

folder 命令

用途

选择和列出文件夹和消息。

语法

folder [**+** *Folder*] [*Message*] [**-all**] [**-nopack** | **-pack**] [**-nofast** | **-fast**] [**-norecurse** | **-recurse**] [**-print** | **-noprint**] [**-header** | **-noheader**] [**-nototal** | **-total**] [**-push** | **-pop**] [**-list** | **-nolist**]

描述

folder 命令设置当前的文件夹和这个文件夹的当前消息，并且列出您的文件夹的信息。作为缺省值，**folder** 命令列出当前文件夹的名字、消息的数目、消息数目的范围和当前的消息。

被 **+***Folder* 标志指定的文件夹成为当前的文件夹。被 *Message* 参数指定的消息成为该文件夹的当前消息。使用 **-pack** 标志能对一个文件夹里的消息重新编号。

标志

-all 显示在您的邮件目录中的关于每个文件夹的信息行。

-fast 只显示文件夹的名字。

+*Folder* 指定要显示的文件夹信息。

-header 显示文件夹信息的柱状图标题。

-help 列出命令语法、可使用的转换（锁定）和版本信息。

注: 对于消息处理程序（MH）来说，标志的名字必须完全清楚的说明。

-list 显示后跟文件夹堆栈内容的当前文件夹。

Message 设定指定的信息作为当前的信息。除非您指定 **+***Folder* 标志，这个命令将为当前的文件夹设置指定的信息。使用以下的参考来设定一个消息:

数目 消息的数目。

cur 或 .（周期）
当前的消息。此为缺省设置。

第一个 文件夹中第一个消息。

最后一个
文件夹中最后一个消息。

下一个 紧跟着当前消息的下一个消息。

新的 创建的新消息。

先前的 在当前消息之前的消息。

-nofast 显示每个文件夹的消息。这个标志是缺省的。

-noheader 禁止文件夹信息的柱状图标题。这个标志是缺省的。

-nolist 禁止显示文件夹堆栈内容。这个标志是缺省的。

-nopack 防止文件夹中的消息重新编号。这个标志是缺省的。

-noprint 防止显示文件夹的信息。如果 **-push**、**-pop** 或 **-list** 标志被指定，**-noprint** 标志是缺省值。

-norecurse 只显示在您的当前文件夹中最高层次的文件夹的信息。关于子文件夹的信息是不显示的。这个标志是缺省的。

-nototal 防止显示在您的邮件目录结构中的所有消息和文件夹。当 **-all** 标志被指定，缺省值是 **-total** 标志；否则，**-nototal** 标志是缺省值。

-pack 对指定文件夹中的消息重新编号。重新编号将清除那些在删除消息后留下的编号间隙。

-pop 将文件夹堆栈最上面的文件夹除去，并设定它为当前的文件夹。**+***Folder* 标志不能用 **-pop** 标志指定。

-print 显示文件夹的消息。如果 **-push**、**-pop** 或者 **-list** 标志被指定，**-noprint** 标志就是缺省值；否则，**-print** 标志是缺省值。

-push 将当前的文件夹移动到文件夹堆栈的最上面并且设定被指定的文件夹为当前文件夹。如果没有文件夹被指定，**-push** 标志将当前的文件夹与文件夹堆栈最上面的文件夹交换。

-recurse 显示在您的当前文件夹中的所有文件夹和子文件夹的信息。

-total 显示在您的邮件目录结构中的所有消息和文件夹。**-total** 标志不显示子文件夹的信息，除非您指定 **-recurse** 标志。如果 **-all** 标志被指定，则 **-total** 标志就是缺省值。

概要文件条目

下面的条目在 *UserMhDirectory*/**.mh_profile** 文件中输入:

Current-Folder: 设定缺省的当前文件夹。

Folder-Protect: 设置新的文件夹目录的保护水平。

Folder-Stack: 指定文件夹堆栈。

lsproc: 指定被用来列出一个文件夹内容的程序。

Path: 指定用户的 MH 目录。

示例

1. 要显示当前文件夹的信息，请输入:

   ```
   folder
   ```

   系统反应与下面的消息类似:

   ```
   inbox+ has 80 messages (1-82); cur = 7; (others).
   ```

   在本示例中，当前文件夹是 inbox。该文件夹包含 80 个消息，消息的范围是从消息 1 到消息 82。当前的消息号是 7。

2. 要显示所有文件夹的信息，请输入:

   ```
   folder -all
   ```

   系统反应与下面的消息类似:

   ```
   Folder  # of messages (range); cur msg  (other files)
   inbox+ has 80 messages (1-82); cur= 7; (others).
   test    has    5  messages  (1-5);    cur=  5;  (others).

           Total=  85  messages  in  2  folders
   ```

   在本例中，有 2 个文件夹，它们一共包含 85 个消息。当前的文件夹是 inbox，它后面跟着一个 +（加号）来指示。

3. 要将文件夹 test 作为当前的文件夹和显示文件夹 test 的信息，请输入:

   ```
   folder  +test
   ```

   系统反应与下面的消息类似:

   ```
   test+ has 5 messages (1-5); cur = 5; (others)
   ```

4. 要使消息 2 成为当前文件夹中的当前消息，请输入:

   ```
   folder 2
   ```

   系统反应与下面的消息类似:

```
test+ has 5 messages (1-5); cur = 2; (others)
```

5. 要创建一个名为 group 的文件夹并且设定它为当前的文件夹，请输入:

   ```
   folder  +group
   ```

   系统反应与下面的消息类似:

   ```
   Create folder "/home/dawn/Mail/group"? _
   ```

   请输入:

   ```
   yes
   ```

   系统反应与下面的消息类似:

   ```
   group+ has no messages.
   ```

6. 要对当前文件夹的消息重新编号，请输入:

   ```
   folder  -pack
   ```

   系统反应与下面的消息类似:

   ```
   inbox+ has 80 messages (1-80); cur= 7; (others).
   ```

   在本示例中，消息被重新编号，这样就可以消除删除消息后留下的消息编号间隙。

文件

$HOME/.mh_profile	包含 MH 用户概要文件。
/usr/bin/folder	包含 **folder** 命令。

相关信息

folders 命令、**mhpath** 命令、**packf** 命令、**refile** 命令。

mh_alias 文件格式，**mh_profile** 文件格式。

《*网络与通信管理*》中的『邮件应用程序』。

---

folders 文件夹

用途

在邮件目录中列出所有的文件夹和消息。

语法

folders [**+***Folder*] [*Message*] [**-all**] [**-pack** | **-nopack**] [**-fast** | **-nofast**] [**-recurse** | **-norecurse**] [**-print** | **-noprint**] [**-header** | **-noheader**] [**-total** | **-nototal**] [**-push** | **-pop**] [**-list** | **-nolist**]

描述

folders 命令列出了在您的邮件目录中的所有文件夹和消息。这个命令等价于被指定了 **-all** 标志的 **folder** 命令。

标志

-all 显示在您的邮件目录中的关于每个文件夹的信息行。

-fast 只显示文件夹的名字。

+*Folder* 指定要显示的文件夹信息。

-header 显示文件夹信息的柱状图标题。这个标志是缺省的。

-help 列出命令语法、可使用的转换（锁定）和版本信息。

注: 对于消息处理程序（MH）来说，标志的名字必须完全清楚的说明。

-list 显示后跟文件夹堆栈内容的当前文件夹。

Message 设定指定的信息作为当前的信息。除非您指定 **+***Folder* 标志，这个命令将为当前的文件夹设置指定的信息。使用以下的参考来设定一个消息:

数目 消息的数目。

cur 或 . (period)
当前的消息。此为缺省设置。

第一个 文件夹中第一个消息。

最后一个
文件夹中最后一个消息。

下一个 紧跟着当前消息的下一个消息。

新的 创建的新消息。

在前的 在当前消息之前的消息。

-nofast 显示每个文件夹的消息。这个标志是缺省的。

-noheader 不显示文件夹信息的柱状图标题。

-nolist 禁止显示文件夹堆栈内容。这个标志是缺省的。

-nopack 防止文件夹中的消息重新编号。这个标志是缺省的。

-noprint 防止显示文件夹的信息。如果 **-push**、**-pop** 或 **-list** 标志被指定，**-noprint** 标志是缺省值。

-norecurse 显示在您的邮件目录中的文件夹的信息。关于子文件夹的信息是不显示的。这个标志是缺省的。

-nototal 防止显示您的邮件目录结构中的所有消息和文件夹。

-pack 对文件夹中的消息重新编号。重新编号将除去那些在消息被删除后留下的编号间隙。

-pop 从文件夹堆栈的最上面除去文件夹并且设定它为当前的文件夹。

-print 显示每个文件夹中的信息数目、每个文件夹的当前消息和当前的文件夹。如果 **-push**，**-pop**，或者 **-list** 标志被指定 **-noprint** 标志就是缺省值；否则，**-print** 标志是缺省值。

-push 将当前的文件夹移动到文件夹堆栈的最上面并且设定被指定的文件夹为当前文件夹。如果没有文件夹被指定，**-push** 标志将当前的文件夹与文件夹堆栈最上面的文件夹交换。

-recurse 显示您的邮件目录结构中的所有文件夹和子文件夹的信息。

-total 显示在您的邮件目录结构中的所有消息和文件夹。**-total** 标志不显示子文件夹的信息，除非您指定 **-recurse** 标志。**-total** 标志是缺省值。

概要文件条目

下面的条目在 *UserMhDirectory***/.mh_profile** 文件中输入:

`Current-Folder:` 设定缺省的当前文件夹。

`Folder-Protect:` 设置新的文件夹目录的保护水平。

`Folder-Stack:` 指定文件夹堆栈。

`lsproc:` 指定被用来列出一个文件夹内容的程序。

`Path:` 指定用户的 MH 目录。

示例

1. 要显示所有文件夹的信息，请输入:

   ```
   folders
   ```

   系统响应与下面的消息类似:

   ```
   Folder  # of messages (range); cur msg  (other files)
   inbox+ has 80 messages (1-82); cur= 7; (others).
   test has 5 messages (1-6);   cur=  5; (others).

      Total= 85 messages in 2 folders.
   ```

   在本例中，有 2 个文件夹，它们一共包含 85 个消息。当前的文件夹是 inbox， 被紧跟着的 +（加号）所指定。

2. 要只列出所有文件夹的名称，请输入:

   ```
   folders    -fast
   ```

   系统反应与下面的消息类似:

   ```
   inbox
   test
   ```

3. 要对所有文件夹中的消息重新编号，请输入:

   ```
   folders    -pack
   ```

   系统反应与下面的消息类似:

   ```
   inbox+ has 80 messages (1-80); cur= 7; (others).
   test has 5 messages (1-5);  cur= 5; (others).
   ```

   在本例中，在文件夹 inbox 中的消息和在文件夹 test 中的消息已经被重新编号，同时消除了在删除消息时留下的消息编号间隙。

文件

$HOME/.mh_profile	包含 MH 用户概要文件。
/usr/bin/folders	包含 **folders** 命令。

相关信息

folder 命令、**mhpath** 命令、**packf** 命令、**refile** 命令。

mh_alias 文件格式，**mh_profile** 文件格式。

《*网络与通信管理*》中的『邮件应用程序』。

---

format 命令

用途

格式化软盘或者可读 / 写介质磁盘。

语法

format [**-d** *Device*] [**-f**] [**-l**]

描述

注: 格式化软盘或者可读 / 写磁盘时将清除上面的任何存在的数据。

format 命令将格式化软盘驱动器里的软盘，它由 *Device* 参数指定。**format** 命令决定设备类型，可能是以下情况中的一种:

- 5.25 英寸低密度软盘（360 KB）包含 40*2 个磁道，每个磁道有 9 个扇区
- 5.25 英寸高容量软盘（1.2 MB）包含 80*2 个磁道，每个磁道有 15 个扇区
- 3.5 英寸低密度软盘（720 KB）包含 80*2 个磁道，每个磁道有 9 个扇区
- 3.5 英寸高容量软盘（1.44 MB）包含 80*2 个磁道，每个磁道有 18 个扇区
- 3.5 英寸高容量软盘（2.88 MB）包含 80*2 个磁道，每个磁道有 36 个扇区

对所有的软盘类型来说，扇区的大小都是 512 个字节。

format 命令格式化被软盘驱动器支持的最高容量的软盘，除非 *Device* 参数指定一个不同的密度。

如果驱动器支持设置缺陷列表头的格式化选项有效位（FOV）为 0，**format** 命令就能格式化一个可读/写的磁盘。格式化一个可读/写磁盘时，在 **-d** 标志之后使用可读/写驱动器的名字（例如 **/dev/romd0**）。获取更多的信息，请参看 **ioctl** 子例程的 **DKFORMAT** 操作，子例程位于 *AIX 5L Version 5.3 Technical Reference: Kernel and Subsystems Volume 2* 中的 scdisk SCSI Device Driver 。

在格式化一个软盘或者可读/写的磁盘之前，**format** 命令会提示验证。这允许您直接地结束这个操作。

标志

-d *Device* 指定用来格式化软盘的设备。如果设备名字以字母 **h** 结尾，则驱动器将以高密度的形式格式化软盘。如果设备名字以字母 **l** 结尾，则驱动器将以低密度的形式格式化软盘。请查阅 **fd** 特别文件以获取关于有效设备类型的信息。这个标志只能与 **format** 命令连用。

注: 如果软盘驱动器支持比软盘制造的最高容量更大的容量，则软盘的容量应该在 **format** 命令的 *Device* 参数（**-d** *Device* 标志）中说明。例如，在一个 4 MB 的软盘驱动器上格式化一个 1 MB的软盘，必须在 **-d** 标志中指定软盘容量，如下:

```
-d /dev/fd0.9 for a 1MB diskette
```

此步的操作失败可能导致读取和写入的错误。

-f 不检查坏磁道而直接格式化软盘，这样会格式化的更快一点。这个标志只能用于软盘，不能用于可读/写的光盘。它也只能用在 **format** 命令中。

-l （小写字母 L）在一个 5.25 英寸、1.2 MB 的软盘驱动器上格式化一个 360 KB 的软盘。在一个 3.5 英寸、1.4 MB的软盘驱动器上格式化一个 720 KB 的软盘。这个标志只能用于软盘，不能用于可读/写的光盘。它也只能用在 **format** 命令中。

注: 一个 360 KB 的软盘驱动器可能不能读取一个在 1.2 MB 驱动器上格式化的 360 KB的软盘。

参数

Device 指定包含要格式化的软盘的设备。缺省值是 **/dev/rfd0** 设备的 0 号驱动器。

示例

1. 在 **/dev/rfd0** 设备上格式化一个软盘，请输入:

   ```
   format   -d   /dev/rfd0
   ```

2. 不检查坏的磁道而直接格式化一个软盘，请输入:

   ```
   format   -f
   ```

3. 在一个位于 **/dev/rfd1** 设备的 5.25 英寸、1.2 MB 的软盘驱动器上格式化一个 360 KB 的软盘，请输入:

   ```
   format   -l   -d   /dev/rfd1
   ```

4. 格式化一个 3.5 英寸、低密度（720 KB）的软盘，请输入:

   ```
   format -d /dev/fd0.9
   ```

5. 格式化一个 3.5 英寸、高密度（1.44 MB）的软盘，请输入:

   ```
   format -d /dev/fd0.18
   ```

6. 在 **/dev/romd0** 设备上格式化一个可读/写的光盘，请输入:

   ```
   format -d /dev/romd0
   ```

文件

/usr/sbin/format	包含 **format** 命令。
/dev/rfd*	指定设备参数。
/dev/fd*	指定设备参数。
/dev/romd*	指定设备参数。
/dev/omd*	指定设备参数。

相关信息

flcopy 命令、**fdformat** 命令。

fd 特殊的文件。

---

fortune 命令

用途

从一个 fortune 数据库中显示一个随机的 fortune。

语法

fortune [**-**] [-**s** | **-l** | **-a** [**-w**]] [*File*]

描述

fortune 命令从 **fortunes.dat** 文件或者被 *File* 参数指定的文件中显示一个 fortune。显示完这个 fortune 后，**fortune** 命令退出。

标志

-	显示用法摘要。
-a	显示一种类型的 fortune。
-l	只显示长 fortune。

-s 只显示短 fortune。
-w 显示完一个 fortune 后，等待以允许用户有时间去读这个 fortune。

文件

/usr/games	系统游戏的位置。
/usr/games/lib/fortune/fortunes.dat	缺省的 **fortune** 数据库的位置。

相关信息

arithmetic 命令、**back** 命令、**bj** 命令、**craps** 命令、**fish** 命令、**hangman** 命令、**moo** 命令、**number** 命令、**quiz** 命令、**ttt** 命令、**turnoff** 命令、**turnon** 命令、**wump** 命令。

---

forw 命令

用途

转发消息

语法

forw [**+** *Folder*] [**-draftfolder +***Folder* | **-nodraftfolder**] [*Message*] [**-draftmessage** *Message*] [**-digest** *Name* [**-issue** *Number*] [**-volume** *Number*]] [**-form** *FormFile*] [**-editor** *Editor* | **-noedit**] [**-whatnowproc** *Program* | **-nowhatnowproc**] [**-filter***File*] [**-annotate** [**-inplace** | **-noinplace**] | **-noannotate**] [**-format** | **-noformat**] [**-help**]

描述

forw 命令为了转发消息启动了接口。根据缺省值，**forw** 命令接口为:

- 打开以编辑 *UserMhDirectory***/draft** 文件
- 提示用户在模板的基础上输入转发的信息，这个模板在 **/etc/mh/mhl.forward** 文件中定义。
- 提示用户输入附加文本，这个附加文本应该同转发信息一同发送的。

为了结束编辑 *UserMhDirectory* **/draft** 文件，按 Ctrl-D 键。**forw** 命令从当前文件夹向 **draft** 文件中加入当前消息。如果您希望加入多于一条的信息，可以用 *Messages* 参数。

> **注**: 在消息头和消息主体之间的一行短划线或者空白行必须保留，因为当传送它的时候这些要用作识别信息。

在退出编辑器时，**forw** 命令启动 `What Now?` 提示。按下回车键可以看见可用的 **whatnow** 子命令列表。这些子命令使您能够继续编辑这个消息，将消息列表，指导消息部署或者结束 **forw** 命令。

forw 命令允许您利用 **-form** 标志来更改转发消息的格式。根据缺省值，这个命令使用位于 *UserMhDirectory***/forwcomps** 文件中的缺省消息格式。如果没有定义您自己的 **forwcomps** 文件就要使用 **/etc/mh/forwcomps** 文件。

利用标志 **-annotate** 用转发信息来评注原始消息。为了确保注释，在退出 **forw** 命令之前要发送已经转发过的注释。

> **注**: **在同一个草稿上的 forw 命令**在多个执行过程中不保留 **-annotate** 标志。

标志

-annotate 用行来注释转发的消息:

```
Forwarded: Date
Forwarded: Addresses
```

利用 **-inplace** 标志来在适当的位置强制注释。这保留同注释的消息的链接。

-digest *Name* 利用摘要设备来为通过 *Name* 变量指定的摘要产生一个新的版本。**forw** 命令扩展了在 **components**文件中的格式化字符串（使用 **repl** 命令采用的同一种格式化字符串机制）并且使用标准的摘要封装算法来构成草稿。草稿写成之后，**forw** 命令写入卷标，为摘要说明条目并且启动编辑器。

除非指定 **-form** 标志，否则 **forw** 命令将使用 *UserMhDirectory***/digestcomps** 文件中的格式。如果这个文件不存在，命令会使用在 **/etc/mh/digestcomps** 文件中指定的缺省值。

-draftfolder +*Folder* 将草稿消息放在指定的文件夹中。如果不指定这个标志，**forw** 命令会根据在消息处理（MH）概要文件中提供的信息中选择一个缺省的草稿文件夹。如果 **+***Folder* 没有指定，则就会假定 `Current-Folder`。您可以在 **$HOME/.mh_profile** 文件中定义一个缺省的草稿文件夹。

注: 如果 **-draftfolder +** *文件夹*后面跟着*消息*参数，它就同指定 **-draftmessage** 标志是一致的。

-draftmessage *Message* 确定草稿消息。如果指定了**-draftfolder** 却没有 **-draftmessage** 标志，则缺省的消息就是 `new`。

-editor *Editor* 为准备消息指定初始的编辑器。

-filter *File* 重新格式化正在转发的消息并且在草稿消息中放置重新格式化的消息。**-filter** 标志接受被 **mhl** 命令采用的格式。

+*Folder* 指定包含您希望转发消息的文件夹。如果没有指定文件夹，就会被认为是当前的文件夹。

-form *FormFile* 在被 *FormFile* 变量指定的格式中显示 **forw** 命令的输出。**forw** 命令在指定的文件中将每一行看成是格式字符串。如果也指定了 **-digest** 标志，则 **forw** 命令采用被 *File* 变量指定的形式作为摘要的形式。如果当 **-digest** 标志被使用的时候没有指定 **-form** 标志则摘要过滤器文件就会变成缺省的形式。

-format 使用 **mhl** 命令和缺省格式文件要重新格式化正在转发的每一条消息并且将重新格式化后的消息存放在草稿消息中。如果 *UserMhDirectory***/mhl.forward** 文件存在，它包含缺省的格式。否则，**/etc/mh/mhl.forward** 文件包含缺省的格式。

-help 列出命令语法、可用的转换（锁定）和版本信息。

注: 对于 MH，这个标志的名称必须拼写完全。

-inplace 强制在适当的位置做出注释以便保留指向注释的消息的链接。

-issue *Number* 指定摘要的期数。缺省的期数比 *UserMhDirectory***/context** 文件中的 `DigestName-issue-list` 条目的当前值大 1。

Message

指定消息。您可以指定几个消息，消息的范围或者单一消息。当指定消息时，采用下面的参考:

Number 消息的数量。

Sequence

用户指定的一组消息。识别的值包括:

all 文件夹中所有的消息。

cur or . (period)
当前的消息。此为缺省设置。

first 文件夹中第一个消息。

last 文件夹中最后一个消息。

new 产生的新消息。

next 紧跟着当前消息的下一个消息。

prev 当前消息的前一个消息。

缺省的消息是在当前文件夹中当前的消息。当您指定几个消息时，转发消息中的头一个变成了当前消息。当您指定文件夹时，那个文件夹就变成了当前的文件夹。

-noannotate
阻止原始消息中的注释。这个标志是缺省的。

-nodraftfolder
将草稿存放在 *UserMhDirectory***/draft** 文件中。

-noedit
禁止初始编辑。

-noformat
防止重新格式化正在转发的消息。这个标志是缺省的。

-noinplace
放置注释在适当的位置。这个标志是缺省的。

-nowhatnowproc
防止 **forw** 命令交互式处理。通过这个标志，没有编辑会发生。

-volume *Number*
指定文摘的卷数。缺省卷数是在 *UserMhDirectory***/context** 文件中 `DigestName-volume-list` 当前的值。

-whatnowproc *Program*
通过转发的任务启动指定的程序来指导您。

注: 如果您为 *Program* 指定了 **whatnow** 命令，**forw** 命令启动一个内部的 **whatnow** 过程而不是一个文件名是 **whatnow** 的程序。

概要文件条目

下面的条目在 *UserMhDirectory***/.mh_profile** 文件中输入:

`Current-Folder:`	设定缺省的当前文件夹。
`Draft-Folder:`	设定缺省的草稿文件夹。
`Editor:`	设定缺省的编辑器。
`fileproc:`	指定用于重新存档消息的程序。
`mhlproc:`	指定用于转发的过滤消息的程序。
`Msg-Protect:`	设定新消息文件的保护级别。
`Path:`	指定 *UserMhDirectory*。
`whatnowproc:`	指定用于提示 `What now?` 问题的程序。

示例

1. 为了将当前的消息转发给另一个人，请输入:

   ```
   forw
   ```

   系统会提示您输入头文件字段中的信息。为了跳过一个字段，可以按下回车键。您必须在 `To:` 字段中输入信息。系统会响应为:

```
---------Enter initial text
```

在转发消息的文本前输入您希望显示的文本，并且按下 Ctrl-D 键。显示转发消息的文本，并以 What now? 提示 在 What now? 提示之后输入 send 以转发消息。

2. 为了从 inbox 文件夹中转发消息 5 ，请输入:

```
forw    +inbox   5
```

文件

/etc/mh/digestcomps	当指定了 **-digest** 标志的时候，定义 MH 缺省消息。
/etc/mh/mhl.forward	包含缺省的 MH 消息过滤器。
UserMhDirectory/**digestcomps**	当指定了 **-digest** 标志的时候，指定用户缺省的消息。(如果存在，它将覆盖 MH 缺省的消息过滤器。)
UserMhDirectory/**forwcomps**	包含用户缺省的消息格式。
UserMhDirectory/**mhl.forward**	包含用户缺省的消息过滤器。(如果存在，它将覆盖 MH 缺省的消息过滤器。)
/usr/bin/forw	包含 **forw** 命令可执行的形式。
$HOME/.mh_profile	包含为单个用户定制 " MH " 的文件。
UserMhDirectory/**draft**	包含为编辑消息创建的草稿。
/etc/mh/forwcomps	定义了通过 **forw** 命令创建的消息组成部分。

相关信息

anno 命令、 **comp** 命令、**dist** 命令、**mhl** 命令、**repl** 命令、**whatnow** 命令。

mh_alias 文件格式，**mh_profile** 文件格式。

《*网络与通信管理*》中的『邮件应用程序』。

---

frcactrl 命令

用途

控制和配置 FRCA。

语法

frcactrl { **load** | **unload** }
frcactrl open *Ip_Address Port* [*Virtual_Host*] *Server_Name Virtual_Root Log_File*
frcactrl close *Ip_Address Port* [*Virtual_Host*]
frcactrl loadfile *Ip_Address Port* [*Virtual_Host*] *Document_Root File ...*
frcactrl stats [reset] [*Interval*]
frcactrl logging *Ip_Address Port* [*Virtual_Host*] { on | off } [*Format*] [*CPU_Id*]
frcactrl { **start** | **stop** } *Ip_Address Port* [*Virtual_Host*]
frcactrl revaltimeout *Ip_Address Port* [*Virtual_Host*] [*Seconds*]
frcactrl pctonintr [*Percentage*]
frcactrl set { option=*value* }
frcactrl get
frcactrl default [*option*]

描述

frcactrl 命令控制和配置 FRCA 的内核扩充。在启动一个想使用 FRCA 的 Web 服务器之前，必须装入内核扩充。

子命令

load 载入没有载入的 FRCA 内核扩充。

unload
卸载已经载入了的 FRCA 内核扩充。

打开 *Ip_Address Port* **[** *Virtual_Host* **]** *Server_Name Virtual_Root Log_File*
在名为 *Server_Name*、IP 地址为 *Ip_Address*、端口为 *Port* 的情况打开和配置一个 FRCA 实例。*Virtual_Root* 参数指定 Web 数据启动的目录。请求将被记录在 *Log_File* 指定的文件中。这个文件名称必须是全限定的。

注: FRCA 只支持一个记录文件。在一个带有 FRCA 的系统中运行超过一个的 Web 服务器时，所有的请求都会被记录到同一个文件中。

close *Ip_Address Port* **[** *Virtual_Host* **]**
关闭与指定 IP 地址和端口相关联的 FRCA 实例。

loadfile *Ip_Address Port* **[** *Virtual_Host* **]** *Document_Root File* **...**
将指定的文件装入到 FRCA 或者网络的高速缓存器中。在这里必须指定 FRCA 实例较早打开所在的 IP 和端口号与根文档和要装载的文件。

stats [reset] [*Interval* **]**
显示 FRCA 的统计信息。可选的 **reset** 子命令清除（置零）这些统计信息。您可以通过使用 *Interval* 参数来按秒指定间隔的持续时间，从而在规则的时间间隔中显示统计信息。

logging *Ip_Address Port* **[** *Virtual_Host* **] { on | off } [** *Format* **] [** *CPU_Id* **]**
将一个 FRCA 实例范围服务的请求记录日志转到指定的 *Ip_Address* 和 *Port* 上。格式可以是 CLF、V-CLF，或者 ECLF 中的一种（通常的记录格式，虚拟主机的 CLF & 扩展的 CLF）。在多处理器机器上，FRCA 记录线程也能通过指定可选的 *CPU_Id* 参数绑定到一个特定的 CPU 上。

start *Ip_Address Port* **[** *Virtual_Host* **]**
将内核获取引擎启用到发送往指定 IP 和端口的服务请求。

stop *Ip_Address Port* **[** *Virtual_Host* **]**
对指定的 IP 和端口禁用内核获取引擎。

revaltimeout *Ip_Address Port* **[** *Virtual_Host* **] [** *Seconds* **]**
对于在指定地址和端口的一个 FRCA 实例，更改重新生效的超时值。超时值必须以秒指定。

pctonintr [*Percentage* **]**
控制能被在中断上下文花费的 CPU 时间百分比。如果这个值太低，则 FRCA 将更多的将请求送到 Web 服务器上，因为它总是在中断上下文执行。任何 >= 100 的值都会导致 FRCA 对 FRCA 高速缓存的每个请求服务。

set {option=*value***}**
将指定的 FRCA 选项指定为该值。当前可用的唯一选项是 **frca_hashsz**，它将 FRCA 散列表中的插槽数设置为指定值。**frca_hashsz** 的缺省值是 12841。如果更改，使用的值必须为质数，因为这将使散列表项更平衡地分布。

get 显示所有可用的 FRCA 选项及它们的当前值。当前仅存在名为 **frca_hashsz** 的选项。

default [*option*]

当使用时不指定选项名称时，将所有选项的值设置为它们的缺省值。如果指定了选项名称，则它仅将指定选项的值设置为缺省值。

示例

1. 下面的示例是关于使用 **open** 子命令的:

```
frcactrl open 9.1.1.1 80 ici imgcache01 /htdocs /logs/frca.log bin
frcactrl open 9.1.1.2 80 ici imgcache02 /htdocs /logs/frca.log bin
```

   在上面的示例中，“ici”是虚拟主机的名字，它被用来访问镜像 imgcache01 或 imgcache02 中的一个。如果 Web 服务器没有被绑定到一个指定的 IP 地址上的话，IP 地址可能就是 0.0.0.0。

2. 为了关闭与 IP 地址 9.1.1.1 和端口 80 相关联的 FRCA 实例，请输入:

```
frcactrl close 9.1.1.1 80
```

3. 为了用 URLs /d 和 /e 装入文件 /a/b/c/d 和 /a/b/c/e 的上下文，请输入:

```
frcactrl loadfile /a/b/c /a/b/c/d e
```

4. 为了显示 FRCA 的统计信息，请输入:

```
frcactrl stats
```

   这将导致 FRCA 统计信息被显示。它们看上去与下面的情况类似:

```
      Total    Deferred       Cache       Cache    Resource
   Requests    Requests        Hits      Misses      Errors
 ------------------------------------------------------------
  1024065396         227  1024065168           1           0
```

5. 这个示例表明在虚拟主机“ici”上如何使用 **start** 子命令:

```
frcactrl start 9.1.1.1 80 ici
```

   注: 虚拟主机参数是可选的。

6. 为了能在虚拟主机“ici”上对端口 80 和 IP 地址 9.1.1.1 上禁用内核获取引擎，请输入:

```
frcactrl stop 9.1.1.1 80 ici
```

7. 下面的示例将在端口 80 和 IP 地址 9.1.1.1 上的 FRCA 示例的重新生效超时值设置为 100 秒。

```
frcactrl revaltimeout 9.1.1.1 80 100
```

8. 为了允许 CPU 将它 98% 的时间用在中断上下文，请输入:

```
frcactrl pctonintr 98
```

9. 要将 **frca_hashsz** 选项的值设置为 24499，请输入:

```
frcactrl set frca_hashsz=24499
```

10. 要将 **frca_hashsz** 的值设置为缺省值，请输入:

```
frcactrl default frca_hashsz
```

文件

/usr/bin/frcactrl

from 命令

用途

决定邮件从谁发来。

语法

from [**-d** *Directory*] [**-s** *Sender*] [*user*]

描述

from 命令在您的邮箱文件中显示消息标题以向您显示这封邮件是谁发送的。如果您指定 *user*，则将检查 *user* 邮箱，而不是您自己的邮箱（只要您对 user 的邮箱有读许可权）。

标志

-d *目录*　　指定系统的邮箱目录。
-s *发送方*　　只显示发送者 发送邮件的消息标题。

参数

user　　指定将检查 *user* 邮箱，而不是您自己的邮箱（只要您对 user 的邮箱有读许可权）。

示例

1. 为了在您的邮箱中显示消息标题，请输入:

   ```
   from
   ```

   发送方的名字和消息数据就会被显示出来。
2. 为了显示一个特定的用户发送邮件的消息标题，请输入:

   ```
   from    -s   dale
   ```

   在这个示例中，只显示了由用户 dale 发送的消息标题。
3. 为了在特定用户的邮箱中显示消息标题，请输入:

   ```
   from   dawn
   ```

   在这个示例中，来自于用户 dawn 邮箱的消息标题显示了出来（如果您有许可来读取 dawn 的邮箱）。
4. 要查看 bob 从 jane 处接收到的所有消息，请输入:

   ```
   from -d /var/spool/mail/bob -s jane root;
   ```

   只要您有许可权（例如 root 用户），这将使您可以看到 bob 从 jane 处接收到的所有消息。

文件

/var/spool/mail/*　　所有用户的系统邮箱。
/usr/bin/from　　用户邮箱文件。

相关信息

mail 命令。

《*网络与通信管理*》中的『邮件应用程序』。

---

fsck 命令

用途

检查文件系统的一致性并且以交互方式修复文件系统。

语法

fsck [**-n**] [**-p**] [**-y**] [**-d***BlockNumber*] [**-f**] [**-i***i-NodeNumber*] [**-o** *Options*] [**-t***File*] [**-V** *VfsName*] [*FileSystem1* - *FileSystem2* ...]

描述

警告: 在出现系统故障之后，总是运行 **fsck** 命令。矫正的动作也许会导致某些数据的丢失。对于每一个一致性的矫正，缺省的操作就是等待操作员输入 yes 或者 no。如果对于已经受到影响的文件系统您没有写的许可，则无论您的实际响应是什么，**fsck** 命令缺省的动作都是 no。

注:

1. 对于一个已经安装好了的文件系统，**fsck** 命令不会做出矫正。
2. **fsck** 命令出于某些原因可以在一个已经安装好了的文件系统中运行，但不是进行修复。但是当文件系统安装完毕之后，也许会返回不准确的错误消息。

fsck 命令检查并以交互方式修复不连贯的文件系统。在安装文件系统之前，应该运行这个命令。您必须能够读设备文件，在这个设备上驻留着文件系统（例如 **/dev/hd0** 设备）。通常，文件系统是连贯的，**fsck** 命令仅仅是报告文件系统中文件的数量、被使用的块和空闲的块。如果文件系统是不连贯的，**fsck** 命令显示关于那些找到的不连贯性的信息并且提示您修复它们的许可。

fsck 命令在修复中是有保留的并且会尽力避免那些可能导致有效数据丢失的动作。在特定的情况下，**fsck** 命令会建议破坏已经损坏的文件。如果您不允许 **fsck** 命令进行必要的修复，则或许会产生一个不连贯的文件系统。安装一个不连贯的文件系统也许会导致系统的崩溃。

如果 JFS2 文件系统有快照，**fsck** 命令将试图保留这些快照。如果此操作失败，则无法保证快照包含来自捕捉到的文件系统的全部先前就存在的映像。**fsck** 命令将删除这些快照和快照逻辑卷。

如果您不用 *FileSystem* 参数指定文件系统，**fsck** 命令将会检查在 **/etc/filesystems** 中列出的所有文件系统，文件的**check**属性将全部设为 True。您可以通过在每一节中加入一行启用这种检查，如下所示:

```
check=true
```

您也可以通过在 **/etc/filesystems** 文件中将文件系统分组来执行多文件系统的检查工作。要做这项工作，在 **/etc/filesystems** 文件中更改检查属性，如下所示:

```
check=Number
```

Number 参数告诉 **fsck** 命令哪一组包含特定的文件系统。使用公共记录设备的文件系统应该被放置在同一个组中。以文件系统在 **/etc/filesystems** 文件中列出的顺序检查它们，一次检查一个。所有 check=true 的文件系统被分到 1 组。**fsck** 命令在其他文件系统之前尝试检查根文件系统，而不管在命令行中或者 **/etc/filesystems** 文件中指定的顺序。

fsck 命令检查以下的不连贯性:

- 分配给多个文件的块或者段。
- 包含交叠块或者段数目的索引节点。
- 包含超出范围的块或者段数目的索引节点。
- 在对文件的目录引用数量和文件链接计数之间的差异。
- 非法分配的块或者段。
- 包含在磁盘映射中标记为空闲的块或者段数量的索引节点。
- 包含被破坏的块或者段数目的索引节点。
- 在索引节点中不是最后磁盘地址的段。检查不会被用于压缩文件系统。
- 包含一个段的超过 32KB 段的文件。检查不会被用于压缩文件系统。
- 尺寸检查:
  - 块的不正确数量。
  - 目录的大小不是 512 字节的整数倍。

  这些检查不被用于压缩的文件系统。
- 目录检查:
  - 目录条目包含一个在索引节点映射中被标记为空闲的索引节点号。
  - 超出范围的索引节点号。
  - 点（.）链接丢失或未指向其本身。
  - 点点（..）链接丢失或未指向父目录。
  - 没有引用的文件或者不可到达的目录。
- 不连贯的磁盘映射。
- 不连贯的索引节点映射。

如果您允许，那些孤立的文件和目录（那些无法到达的）可以通过将它们加到在文件系统根目录下的 **lost+found** 子目录下面使它们重新连接起来。指定的名称是索引节点号。如果您不允许 **fsck** 命令重新配属一个孤立的文件，它会请求破坏这个文件的许可。

除了它的消息之外，**fsck** 命令会通过它的出口值来记录检查和修复的结果。这个出口值可能是以下情况的任意和:

0 所有被检查的文件系统现在都好了。
2 **fsck** 命令在结束检查或修复之前被中断了。
4 **fsck** 命令更改了文件系统；用户必须立即重新启动系统。
8 文件系统包含没有修复的损坏部分。

当系统从磁盘进行引导，引导过程会很明确的运行 **fsck** 命令，以 **//usrx**、**/var**、**/tmp** 文件系统中的 **-f** 和 **-p** 标志指定。如果没有成功的在这些文件系统之一执行 **fsck** 命令，系统不会引导。在这样的系统引导之前，从可删除的介质引导并进行维护工作将会被需要。

如果 **fsck** 命令成功的在 **/**、**/usr**、**/var** 和 **/tmp**中运行， 正常的系统初始化将会继续进行。在正常的系统初始化过程中，**fsck** 命令同 **-f** 和 **-p** 标志一同指定，从 **/ etc/rc** 文件中运行。这个命令序列检查所有的文件系统，在其中 **check** 属性设成了 True（ `check=true`）。如果 **fsck** 命令从 **/etc/rc** 文件中执行，命令不能保证文件系统的连贯性，系统初始化继续进行。任何不连贯文件系统的安装也许会失败。安装失败也许会导致系统不完全的初始化。

注: 在缺省情况下，**//usr**、**/var** 和 **/tmp** 文件系统在 **/etc/filesystem** 节的 **check** 属性被设成了 False（`check=false`）。属性被设成了 False 是由于以下原因:

1. 引导过程在 **//usr** 、**/var**和 **/tmp** 文件系统中明确的运行了**fsck** 命令。
2. 当 **/etc/rc** 文件被执行的时候，**/ /usr** 、 **/var** 和 **/tmp**文件系统就被安装了。**fsck** 命令不会修改一个已经安装了的文件系统。此外，在一个已经安装好了的文件系统上运行的 **fsck** 命令产生不可靠的结果。

您可以使用在基于 Web 的系统管理器 (wsm) 中的文件系统应用程序来更改文件系统的特征。您也可以系统管理界面程序（SMIT）**smit fsck** 快速路径来运行这个命令。

标志

-d *BlockNumber*
搜索指定磁盘块的参考。无论 **fsck** 命令遇到包含特定块的文件，它将会显示索引节点号和所有指向它的路径名称。对于 JFS2 文件系统，引用指定模块的索引节点号将会被显示，但是不是它们的路径名称。

-f
进行快速检查。在正常情况下，通过非正确方式关闭系统来停机仅有的文件系统很可能被影响，这个文件系统就是当系统停止时在安装的那些。 **-f** 标志会提示 **fsck** 命令不要检查没有成功安装的文件系统。**fsck** 命令通过检查文件系统超级块中的 **s_fmod** 标志来决定这件事。

当文件系统没有成功安装的时候，无论何时文件系统被安装和被清除，这个标志都将被设定。如果文件系统被成功的卸载，这不大可能会存在什么问题。因为多数文件系统没有成功安装，不检查这些文件系统能减少检查时间。

-i *i-NodeNumber*
搜索指定索引节点的参考。无论何时 **fsck** 命令遇到一个指向指定索引节点的目录，它都会显示这个参考的完整路径名称。

-n
对 **fsck** 命令所提出的所有问题给出一个`no`的回应；不打开指定的文件系统来写。

-o *Options*
向 **fsck** 命令传递逗号分隔的选项。当前对 JFS 支持以下选项（较新的文件系统废弃了以下选项，可以忽略它们）:

mountable
如果有问题的文件系统可安装（清除），促使 **fsck** 命令成功的退出，返回一个 “0” 值。如果文件系统不可安装，**fsck** 命令退出并返回一个值 “8”。

mytype
如果有问题的文件系统与在 **/etc/filesystems** 文件中或者在命令行中通过 **-V** 标志指定的具有相同的类型，则促使 **fsck** 命令退出并给出一个成功的 “0” 值。否则，返回一个值 “8”。例如，如果**/**（引导文件系统）是一个分类文件系统，则 fsck 命令 `-o mytype -V jfs /` 会退出给出一个 “0” 值。

-p
不显示次要问题的消息但是自动修复问题。这个标志并不是象 **-y** 标志那样授予大规模许可，当系统正常启动的时候对自动进行检查工作有用。无论系统在何时自动运行，您应该将这个标志作为系统启动过程的一部分来使用。如果主要的超级块损坏了，次要的超级块就被验证，并且复制到主要的高级块中。

-t*File*
如果 **fsck** 命令得不到足够的内存来保存它的表的话，在文件系统中作为一个临时文件而不是被检查的文件来指定*File* 参数。如果没有指定 **-t** 标志，则 **fsck** 命令需要一个临时文件，它会提示您给这个临时文件起名字。但是，如果指定了 **-p** 标志，**fsck** 命令是不成功的。如果临时文件不是一个特定的文件，当 **fsck** 命令结束的时候，它就会被删除。

-V *VfsName*
使用为文件系统由 *VFSName* 变量指定的虚拟文件系统的描述，而不是用 **/etc/filesystems** 文件决定描述。如果 **-V** *VfsName* 标志没有在命令行中指定，就会检查 **/etc/filesystems** 文件并且 **vfs=** *匹配节的特性*被认为是正确的文件系统类型。

-y
对所有 **fsck** 命令提出的所有问题假定一个 “yes” 的响应。这个标志使 **fsck** 命令采取它认为必要的行动。仅在损坏严重的文件系统中使用这个标志。

示例

1. 为了检查所有的缺省文件系统，请输入:

   ```
   fsck
   ```

   这个命令检查在 **/etc/filesystems** 文件中所有标记 `check=true` 的文件系统。**fsck** 命令这种形式在对文件系统做出任何更改之前会向您请求许可。
2. 为了利用缺省的文件系统自动修复较次要的问题，请输入:

   ```
   fsck -p
   ```
3. 为了检查一个特定的文件系统，请输入:

   ```
   fsck /dev/hd1
   ```

   这个命令检查位于 **/dev/hd1** 设备上的未安装的文件系统。

文件

/usr/sbin/fsck	包含 **fsck** 命令。
/etc/filesystems	列出已知的文件系统并且定义它们的特征。
/var/spool/mail/*	包含虚拟文件系统类型的描述。
/usr/bin/from	包含当系统启动的时候运行的命令（包括 **fsck** 命令）。

相关信息

dfsck 命令、**fsdb** 命令、**istat** 命令、**mkfs** 命令、**ncheck** 命令、**rc** 命令和 **shutdown** 命令。

filesystems 文件、**filsys.h** 文件。

《*操作系统与设备管理*》中的『文件系统』说明了文件系统类型、管理、结构和维护。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』 。

《*操作系统与设备管理*》中的『系统管理界面工具』说明了 SMIT 结构、主菜单和任务。

---

fsck_cachefs 命令

用途

检查用 CacheFS 高速缓存的数据的完整性。

语法

fsck_cachefs [**-m**] [**-o noclean**] *cache_directory*

描述

fsck 命令的 CacheFS 版本检查高速缓存目录的完整性。能够缺省更正发现的 CacheFs 问题。没有交互式的方式。对于 CacheFS 文件系统 **fsck_cachefs** 命令最有可能的调用是在从 **/etc/rc.nfs** 中条目得到的引导时间。

标志

-m 检查，但是不修复。
-o noclean 即使没有理由怀疑有问题也在高速缓存中进行强制检查。

示例

为了在高速缓存目录中进行强制检查，请输入:

```
fsck_cachefs -o noclean /cache3
```

---

fsdb 命令

用途

调试文件系统。

语法

fsdb filesystem [**-**]

描述

fsdb 命令允许您对 *FileSystem* 参数指定的文件系统进行检查、修改和调试。这个命令向文件系统对象提供访问，例如块、索引节点或者目录。您可以使用 **fsdb** 命令检查并且修复已经被破坏的文件系统。文件系统的关键组成部分可以象征性的引用。这个功能为修改控制块的条目和降序文件系统树简化过程。

为了检查一个文件系统，要通过一个块设备的名称、一个原始设备名称或者一个已经安装的文件系统名称来指定它。在最后一种情况下，**fsdb** 命令通过读 **/etc/filesystems** 文件来决定相关的文件系统名称。安装的文件系统不能被修改。

对于 JFS 文件系统和 JFS2 文件系统，**fsdb** 命令有不同的接口。下面解释如何将 **fsdb** 命令用于 JFS 文件系统。参考 JFS2 子命令可以得到关于 JFS2 子命令的信息。

如果这个指定的文件系统是一个 JFS2 快照，则 **fsdb** 命令启用检查和修改快照高级块、快照映射、块映射 x 树副本和节头。参考 JFS2 快照子命令可以得到有关 JFS2 快照子命令的信息。

fsdb 命令的子命令允许您访问、查看或者更改文件系统中的信息。在子命令中，您输入的任何数字被缺省默认为十进制，除非您在它的前面加前缀 “0” 表示八进制或者加 “0x” 表示十六进制。所有的地址都显示成十六进制。

因为 **fsdb** 命令在一个时间读写一个块，它使用 raw 工作就像使用 block I/O。

标志

- 禁用那些用于检验索引节点和块地址的错误检查例程。**o** 子命令切换这些例程的开关。当这些例程在运行的时候，**fsdb** 命令从高级块中读取关键的文件系统数据。获取的信息允许 **fsdb** 命令成功地访问不同的文件系统对象并且做出不同的错误检查。

子命令

fsdb 子命令是在文件系统中定位和显示，或者修改信息的请求。子命令主要的类别是:

类别	功能
位置	访问文件系统中的信息。
显示	查看文件系统中的信息。
修改	更改文件系统中的信息。

另外，有几个其他的子命令。

位置子命令

有两种类型的位置子命令:

Number[**I** | **M** | **i** | **b**]

OR

d*DirectorySlot*

第一种包括一个数字，后面跟着一个地址说明（可选）。这个地址说明定义了前面的数字如何被解释。有四种地址说明对应四种不同的 *Number* 变量的解释:

I 索引节点映射块数字
M 磁盘映射块数字
i 索引节点号
b 片段数字

依靠地址规范（或者缺少它），这种类型的位置子命令访问信息如下所示:

Number 在 *Number* 变量指定的绝对字节偏移量处访问数据。
*MapBlockNumber***I** 访问通过 *MapBlockNumber* 变量指定的索引节点映射块。
*MapBlockNumber***M** 访问通过 *MapBlockNumber* 变量指定的磁盘映射块。
*InodeNumber***i** 访问通过 *InodeNumber* 变量指定的索引节点。
*FragmentNumber***b** 访问通过 *FragmentNumber* 变量指定的文件系统块。一个片段号包括一个块地址和一个编码长度。一个完整的段地址在长度上是一个 32 位数字。前面 28 位是起始段地址。段的长度在剩余的 4 位被编码；它作为一个少于一个满块的段数目被编码。例如，在一个包含有 1024 字节段的文件系统，地址 0x2000010f 指的是一个块，这个块在 1KB 块数字 0x10f 开始，并且有 2KB 长度。与之相对，在一个包含 512 字节段的文件系统中，地址 0x2000010f 指的是一个块，这个块在 512 字节块 0x10f 开始，并且有 3072 字节的长度（512*6）。

第二种位置子命令是用来访问目录条目的。这个子命令包括一个字符 **d** ，这个字符后面紧跟着一个目录口数字。对于每一个相关的索引节点块，目录数字都以 0 开始。

这种位置子命令访问信息如下所示:

d*DirectorySlot* 访问*目录条目*，这个条目为当前索引节点被 DirectorySlot 变量索引。使用这个子命令，只有分配的目录条目才能被操作。

显示子命令

为了查看与地址规范相关的信息，要使用一个显示子命令，这个子命令要包含在一个与一种显示格式相关联的显示设备当中。

p[*Number*]{ **i** | **d** | **o** | **e** | **c** | **b** | **y** | **M** | **I** | **x** | **s** | **D** }

OR

f[*Number*]{ **i** | **d** | **o** | **e** | **c** | **b** | **y** | **M** | **I** | **x** | **s** | **D** }

显示设备是:

(.,.)**p** 指定一个常规的设备。使用常规的显示子命令来显示与当前地址相对应的数据。如果您在符号 **p** 之后输个数字，**fsdb** 命令显示这个条目的数字。要做一个检查以检测块的边界溢出。如果您输入 “0” 或 “*” （星号），**fsdb** 命令会显示到当前段末的所有条目。

f 指定一个文件设备。使用这个文件显示子命令来显示同当前索引节点关联的数据块。如果您在符号 **f** 后输入一个数字，**fsdb** 命令会显示这个文件的块。块的编号从 0 开始。显示的格式遵循块的数字。如果您输入 **f** 而不输入一个块号，则 **fsdb** 命令会默认显示当前索引节点的块 0。

对于每个设备显示的格式是:

i 作为索引节点显示。
d 作为目录显示。
o 作为八进制的字来显示。
e 作为十进制的字来显示。
c 作为字符来显示。
b 作为八进制字节来显示。
y 作为十六进制字节来显示。
M 作为磁盘映射条目来显示。
I 作为索引节点映射条目来显示。
x 作为十六进制字来显示。
S 作为单间接块来显示。
D 作为双间接块来显示。

选中的显示设备和显示格式在 **fsdb** 命令期间一直起作用，直到明确的更改。如果您指定的地址没有落在一个正确的边界内部，您也许会收到一个错误消息，指出这个不正确的调整。

如果您使用 *Number*、h>, *MapBlockNumber***I** 或 *FragmentNumber***b** 位置子命令来访问索引节点信息的话，您可以单步遍历这些数据，检查每一个字节、字或者双字。通过输入以下子命令中的一种可以选择希望得到的显示模式。

B 在字节模式下开始显示。
D 在双字模式下开始显示。
w 在单字模式下开始显示。

根据这些信息，您可以向前或者向后移动。边界随着显示屏幕而前进，并且在显示的最后一项的地址上留了下来。通过按下 “INTERRUPT” 键，输出可以在任意时间终止。下面的这些符号允许在信息中移动:

Number 向前移动指定数目的当前有效单元。
Number 向后移动指定数目的当前有效单元。

下面的符号允许您存储当前的地址，并且很方便的返回到它。

> 存储当前地址。
< 返回到先前存储过的地址。

您可以用点、制表符和空格作为子命令的定界符，但是它们仅仅对从可能被解释为十六进制位的子命令定界十六进制数字是必要的。按下回车键（进入一个空白行）通过上一次显示的数据类型大小来增加当前的地址。也就是说，这个地址被设在了下一个字节、字、双字、目录条目或者索引节点，这就允许您单步遍历文件系统的区域。

fsdb 命令以一种适合数据类型的格式显示信息。字节、字和双字会作为十六进制地址来显示，这些地址后面跟着在那个地址上数据的十六进制表示，而且在括号中放入了相等的十进制数。**fsdb** 命令相这个地址的末尾加入 **.B** 或者 **.D** 后缀以指出显示的字节或者双字的值。它显示目录作为目录口的偏移量，这些偏移量后面跟着十进制的索引节点号以及条目名称的字符表示。它将索引节点同描述每个元素的标号字段一同显示。环境变量控制数据的格式和时间字段。

修改子命令

通过使用字段规范（对于在索引节点的字段和在目录中的字段），您可以修改同这个地址规范相关联的信息。对于分配新的值，常规的形式是: *助记符运算符新的值*这里助记符参数代表下面列表中所描述的字段之一:

下面的助记符被用作索引节点字段的名称并且指向当前工作的索引节点。

`md`	允许模式
`ln`	链接计数
`uid`	用户数量
`gid`	分组数目
`sz`	文件大小
a*Number*	数据块数量（0 到 8），这里 Number 参数可以是一个位置子命令。
`at`	访问时间
`mt`	修改时间
`maj`	主要设备数
`min`	次要设备数

下面的助记符指的是索引节点和磁盘映射:

`mf`	映射空闲计数
`ms`	映射大小
`mp`	永久分配位图
`mw`	工作分配位图

以下助记符用于目录中字段的名称:

`rl`	目录条目记录的长度
`nl`	目录名称的长度
`nm`	目录名称

Operator 参数有效的值包括:

注: 在试图修改文件系统之前必须卸装它。

= 将 *New-Value* 参数分配给指定的 *Mnemonic* 参数。

=+ 通过指定的 *New-Value* 参数增加了 *Mnemonic* 参数。缺省的*New-Value* 参数是值 1。

=- 通过指定的 *New-Value* 减少*助记符*。缺省的*新值*是值 1。

=″ 将 *New-Value* 参数指定的字符串分配给指定的*助记符*参数。如果当前的显示格式是目录的 **d** 地址规范并且没有指定助记符的话，目录名称被更改。新的目录名称不能长于先前的目录名称。

其他子命令:

其他子命令是:

q 退出。

x*n* 扩展目录 *n* 字节，这里 *n* 加上目录的当前大小在字节上不大于当前目录片段。
! 返回 shell。
O 锁定错误检查。

JFS2 子命令

通过完整的名称或者使用名称的一部分可以输入这些子命令。至少应该输入粗体字部分。

a[lter] <block> <offset> <hex string>	修改磁盘数据。
b[map] [<block number>]	显示块分配映射。
dir[ectory] <inode number> [<fileset>] [R]	显示目录条目。
d[isplay] [<block> [<offset> [<format> [<count>]]]]	显示数据。
dt[ree] {<block number> |<inode number>{a | f } }	显示 d 树节点。
h[elp] [<command>]	提供子命令的帮助。
ia[g] [<IAG number>] [a | <fileset>]	显示 IAG 页面。
im[ap] [a | <fileset>]	显示索引节点分配映射。
i[node] [<inode number>] [a | <fileset>]	显示索引节点。
q[uit]	退出 fsdb 命令。
su[perblock] [p | s]	显示高级块。
x[tree] {<block number> | <inode number>{a | f}}	显示 x 树节点。

a[lter] <block> <offset> <hex string>

其中:

<block>	块号（十进制）
<offset>	在块的范围内的偏移量（十六进制）
<hex string>	十六进制位的字符串

更改磁盘数据。<hex string> 应包含偶数个数字。

b[map] [<block numbers>]

显示块分配的映射。

<block number> 显示描述此块数量的 **dmap** 页面。

子命令:

m	修改当前节点
u	访问上一个级别的 bmap 页面
l	访问左兄弟节点
r	访问右兄弟节点
w	显示 wmap
(.,.)**p**	显示 pmap
s	显示 s 树。
x	退出子命令方式

dir[ectory] <inode number> [<fileset>][R]

<inode number>	目录的索引节点号（十进制）
<fileset>	数量，当前值必须是零
R	递归列出所有的子目录

显示目录条目。

d[isplay] [<block> [<offset> [<format>[<count>]]]]

<block>	块号（十进制）
<offset>	在块的范围内的偏移量（十六进制）
<format>	显示数据的格式（见下面）
<count>	要显示对象的数量（十进制）

在各种不同的格式下显示数据。

格式也许是以下中的一种:

a	美国信息交换标准码（ASCII）	
i	索引节点	结构 dinode
I	索引节点分配的映射	iag_t
s	高级块	结构高级块
x	十六进制	

dt[ree] {<block number> | <inode number>{a | f}}

<block number>	块号包含 d 树页面
<inode number>	目录的索引节点号（十进制）
{a | f}	“a” 指示索引节点号是一个聚集索引节点。 “f” 指示索引节点号是文件集索引节点。

显示 b 树目录的根并且输入一个子命令方式，在这个方式中浏览这个树。

子命令:

m	修改当前的节点
f	步入空闲列表的条目
s	显示指定的插槽入口
[0-9]+	显示指定的 stbl 入口
t	显示格式化过的 stbl

u 访问父节点（不是父目录）
d 访问子节点
x 退出子命令方式

h[elp] [<command>]

<command> 命令名称

打印帮助文本。如果没有参数则列出所有的命令。

ia[g] [<IAG number>] [a | <fileset>]

<IAG number> IAG 数量（十进制）
a 使用聚集节点表
<fileset> 文件集数量（目前必须是零）

显示 iag 信息并且进入子命令模式。

子命令:

e 显示 / 修改索引节点范围映射
m 修改 iag
p 显示 / 修改持久映射
w 显示 / 修改工作映射

im[ap] [a | <fileset>]

a 使用聚集索引节点表
<fileset> 文件集数量（目前必须是零）

显示指定的索引节点映射并且进入子命令方式。

子命令:

e 显示 / 修改索引节点范围映射
m 修改 iag
p 显示 / 修改持久映射

i[node] [<inode number>] [a | <fileset>]

<inode number> 索引节点号（十进制）
a 使用聚集索引节点表

文件集　　文件集的数量（目前必须是零）

显示索引节点信息并且进入子命令方式。

子命令:

m　　修改索引节点
t　　显示 / 修改索引节点的 b 树
e　　显示 / 修改索引节点的 EA

注: **fsdb** 命令了解 **v1** 和 **v2** 扩展属性格式。查看 EA 时的行为取决于被查看的索引节点的格式。

对于 **v1**，在显示索引节点的 EA 后，您可以修改其 pxdTable 或 eaDirectory 条目。指定修改选项，然后指定 pxdTable 或者 eaDirectory 指示符和偏移量到表中。

对于 **v2**，EA 用 **dtree** 子命令格式显示。然后，所有 **dtree** 子命令都可用于 EA 上的进一步操作。

q[uit] 退出 fsdb 命令。

su[perblock] [p | s]

p　　显示首要的高级块
s　　显示次要的高级块

显示高级块数据。

x[tree] {<block number> | <inode number>{a | f} }

<block number>　　块号（十进制）
<inode number>　　索引节点号
{a | f}　　“a” 指示索引节点号是一个聚集索引节点。 “f” 指示索引节点号是文件集索引节点。

显示 x 树的一个索引节点并进入一个子命令方式，在其中浏览 x 树。

子命令:

m　　修改当前节点
u　　访问父节点
d　　访问子节点
n　　访问右兄弟进程
p　　访问左兄弟进程
s　　选择要查看的 xad 条目
x　　退出子命令方式

JFS2 快照子命令

通过完整的名称或者使用名称的一部分可以输入这些子命令。至少应该输入粗体字部分。

a[lter] <block> <offset> <hex string>	修改磁盘数据。
b[map]	显示块映射 x 树的副本。
d[isplay] [<block> [<offset> [<format> [<count>]]]]	显示数据。
h[elp] [<command>]	提供子命令的帮助。
q[uit]	退出 fsdb 命令。
st[able] [<block number>]	显示总结快照表。
s[map] <block number>	显示快照位图。
su[perblock]	显示高级块。

a[lter] <block> <offset> <hex string>

其中:

<block>	块号（十进制）
<offset>	在块的范围内的偏移量（十六进制）
<hex string>	十六进制位的字符串

更改磁盘数据。<hex string> 应包含偶数个数字。

b[map]

显示块映射 x 树的副本。

d[isplay] [<block> [<offset> [<format>[<count>]]]]

<block>	块号（十进制）
<offset>	在块的范围内的偏移量（十六进制）
<format>	显示数据的格式（见下面）
<count>	要显示对象的数量（十进制）

在各种不同的格式下显示数据。

格式也许是以下中的一种:

a	美国信息交换标准码（ASCII）
s	快照段头
t	快照表页面
x	x 树页面

h[elp] [<command>]

<command>	命令名称

提供子命令的帮助。

q[uit] 退出 fsdb 命令。

st[able] [<block number>]
其中:

<block number> 块号（十进制）

显示总结快照表。

s[map] [<block number>]
其中:

<block number> 块号（十进制）

显示快照位图。

su[perblock]

显示高级块。

示例

下面的这些示例表明了在一个 JFS 文件系统中当开始了 **fsdb** 这个命令之后您可以使用的子命令。

1. 要显示索引节点，请输入:
   ```
   386i
   ```
   这个命令在索引节点格式下显示 386 索引节点。现在变成了当前的索引节点。
2. 要将当前索引节点的链接计数更改为值 4，请输入:
   ```
   ln=4
   ```
3. 要将当前索引节点的链接计数增加值 1，请输入:
   ```
   ln=+1
   ```
4. 要显示与当前索引节点关联的文件的某部分，请输入:
   ```
   fc
   ```
   这个命令以 ASCII 码字节的形式显示与当前的索引节点相关的文件的块 0。
5. 要显示目录的条目，请输入:
   ```
   2i.fd
   ```

这个命令将当前的索引节点更改成为根索引节点（索引节点 2 ），然后显示与那个节点相关的第一个块中的目录条目。显示的最后的一个或者更多条目也许有一个索引节点号 0。这些是未用的目录块；这种条目不能在下一个示例操作。

6. 要进入目录树的下一级，请输入:

   ```
   d5i.fc
   ```

   这个命令将当前索引节点更改成与目录字段 5 相关联的索引节点。然后，将这个文件中的第一个块以 ASCII 文本的形式显示出来。fc 目录条目从 0 开始编号。

7. 要显示知道其块号的块，请输入:

   ```
   1b.p0o
   ```

   这个命令以八进制的形式显示了文件系统中的高级块（块 1）。

8. 要更改目录条目的索引节点，请输入:

   ```
   2i.a0b.d7=3
   ```

   这个命令将在根目录（2i）中目录条目 7 的索引节点改为 3。这个示例也说明几个操作如何合在一行中。

9. 要更改目录条目的文件名，请输入:

   ```
   d7.nm="chap1.rec"
   ```

   这个命令将目录条目 7 的名称字段改成了 chap1.rec 。

10. 要显示与当前索引节点关联的文件的给定块，请输入:

    ```
    a2b.p0d
    ```

    这个命令显示了作为目录条目的当前索引节点的块 2。

11. 要显示块 7 中的单独间接块的内容，请输入:

    ```
    7b. p0S
    ```

    这个命令显示了分配给在块 7 中有单独间接块的索引节点的块号。

12. 要显示磁盘映射的第一页，请输入:

    ```
    OM
    ```

13. 要以十六进制显示永久块分配映射的前 10 个字，请输入:

    ```
    mp1.p10x
    ```

    这个命令表明在当前地址的分配位图；例如，在 0M。

文件

/usr/sbin	包含 **fsdb** 命令。
/etc/filesystems	包含关于文件系统的信息。

相关信息

dfsck 命令、**fsck** 命令。

dir 文件、 **filsys.h** 文件

环境杂项设备。

读子例程。

《*操作系统与设备管理*》中的『文件系统』说明了文件系统类型、管理、结构和维护。

《*操作系统与设备管理*》中的『文件』提供了有关如何处理文件的信息。

---

fsplit 命令

用途

将 FORTRAN 源文件分割成单独的例程文件。

语法

fsplit [**-e** *SubprogramUnit*] ... [*File*]

描述

fsplit 命令将文件或者包含 **FORTRAN** 源代码的标准输入作为输入，并且将这些输入分割成 *name.f* 形式的独立例程文件，这里 name 将会是程序单元的名字（例如，函数、子例程、块数据或程序）。

没有命名的块数据子程序的名字有 *blkdtaNNN*.**f** 的形式，这里 *NNN* 是三个数字，这个名字的文件没有已经存在。对于没有命名的主程序名字有 *mainNNN*.**f** 的形式。如果在划分程序单元的时候有错误，或者如果 *name*.**f** 已经存在了，则程序单元会被放在一个形式为 *zzzNNN*.**f** 的文件中，在这里 *zzzNNN*.**f** 不存在。

> **注**: **fsplit** 命令认为子命令的名字在子程序单元第一个没有注释的行中。非标准的源格式能够混淆命令并且产生意想不到的结果。

标志

-e *SubprogramUnit*
只将指定要分割的子程序单元分割成单独的文件。通常每个子程序单元分割成一个文件。

-e 标志只能被用在指定的主程序和块数据子程序。如果找不到通过 **-e** 选项指定的名字，诊断被写到标准错误。

示例

下面的 **fsplit** 命令将子程序 `readit` 和 `doit` 分割成不同的文件。

```
fsplit -e readit -e doit prog.f
```

文件

/usr/bin/fsplit 包含 **fsplit** 命令、

相关信息

asa 或者 **fpr** 命令、 **struct** 命令。

ftp 命令

用途
在本地主机和远程主机之间传送文件。

语法

ftp [**-d**] [**-g**] [**-i**] [**-n**] [**-v**] [**-f**] [**-K**] [**-k** *realm*] [**-q**[**-C**]][*HostName* [*Port*]]

描述
ftp 命令使用“文件传送协议”(FTP）在本地和远程主机或远程主机之间传送文件。

FTP 协议允许在使用不同文件系统的主机之间进行数据传送。尽管协议在传送数据中提供了很高的灵活度，它仍然不会尝试保留特定于某个文件系统的文件属性（如文件保护模式或修改时间）。而且，FTP 协议为文件系统的整体结构做了少许假设，且不提供或不允许诸如循环地复制子目录这样的函数。

请注意: 如果您正在系统之间传送文件，且需要保存文件属性或递归地复制子目录，请使用 **rcp** 命令。

发布子命令
您可以在 `ftp>` 提示符下输入子命令以执行类似这样的任务: 列出远程目录、更改当前的本地和远程目录、在单一请求中传送多个文件、创建和除去目录，以及转到本地 shell 执行 shell 命令。请参考子命令章节取得每个子命令的描述信息。

如果您执行 **ftp** 命令而不为远程主机指定 *HostName* 参数，**ftp** 命令会立即显示 `ftp>` 提示符，等待 **ftp** 子命令。要连接远程主机，请执行 **open** 子命令。当 **ftp** 命令连接到远程主机时，**ftp** 命令在再次显示提示符 `ftp>` 之前会提示输入登录名和密码。如果远程主机中未定义登录名的密码，**ftp** 命令将不成功。

ftp 命令解释器（处理在 `ftp>` 提示符处输入的全部子命令），会提供大多数文件传送程序没有的工具，如:
- 对 **ftp** 子命令处理文件名参数
- 将一组子命令集中成一个单一的子命令宏
- 从 **$HOME/.netrc** 文件中装入宏

这些性能会帮助您简化重复的任务，并允许您在 unattended 方式使用 **ftp** 命令。

命令解释器将按照以下规则处理文件名参数:
- 如果为此参数指定了 -（连字符），则标准输入（stdin） 将用于读取操作，而标准输出用于写入操作。
- 如果未应用前面的检查，且文件名扩展已启用（请参考 **-g** 标志或 **glob** 子命令），则解释器将根据 C shell 的规则扩展文件名。启用了文件名匹配替换以及在期待单一文件名的子命令中使用了模式匹配字符时，结果可能与期待的不一样。

  例如，**append** 和 **put** 子命令将拓展文件名，然后仅使用所生成第一个文件名。其他 **ftp** 子命令，如 **cd**、**delete**, **get**、**mkdir**、**rename** 和 **rmdir** 不会执行文件名拓展，并从字面上接受模式匹配字符。
- 对于 **get**、**put**、**mget** 和 **mput** 子命令，解释器有能力在不同的本地和远程文件名语法样式之间进行翻译和映射（请参考 **case**、**ntrans** 和 **nmap** 子命令），并且在本地文件名不是唯一的情况下有修改它的能力（请参考 **runique** 子命令）。另外，如果远程文件名不是唯一的，则 **ftp** 命令可将指令发送到远程 **ftpd** 服务器，以修改远程的文件名（请参考 **sunique** 子命令）。
- 使用双引号（″ ″） 指定包含空字符的参数。

请注意: **ftp** 命令解释器不支持管道。也无需支持所有多字节字符文件名。

要在交互运行的时候结束 **ftp** 会话，请使用 **quit** 或 **bye** 子命令或在 ftp> 提示符处使用“文件结束（End of File）”(Ctrl-D）键序列。要在文件传送未完成之前结束它，请按中断按键顺序。其缺省“中断键”序列是 Ctrl-C。**stty** 指令能用于重新定义该键系列。

ftp 命令在正常情况下会立即暂停正在发送（从本地主机到远端主机）的传输。**ftp** 命令通过将 FTP ABOR 指令发送到远程 FTP 服务器，来暂停正在接收的传输（从远程主机到本地主机），并废弃所有传入的文件传送包（直到远程服务器停止发送它们为止）。如果远程服务器不支持 ABOR 指令，在远程服务器发送所有请求的文件之前，**ftp** 命令不会显示 ftp> 提示符。另外，如果远程服务器执行未期望的操作时，可能需要结束本地 **ftp** 进程。

安全性和自动登录

如果“标准的”是当前认证方法: : **ftp** 命令也会通过将密码发送到远程主机来处理安全性，并许可自动登录、文件传送和注销操作。

如果您执行 **ftp** 命令并指定远程主机的主机名 (*HostName*)，**ftp** 命令会尝试建立一个对于指定主机的连接。如果 **ftp** 命令连接成功，**ftp** 命令会在当前目录或主目录中搜索本地的 **$HOME/.netrc** 文件。如果文件存在，**ftp** 命令会在文件中搜索启动登录过程的入口，在命令宏定义中搜索远程主机。如果 **$HOME/.netrc** 文件或自动登录项不存在，或是系统已使用 **securetcpip** 命令保护起来，**ftp** 命令会提示用户输入用户名和密码。不论命令行中是否指定 *HostName* 参数，该指令都会显示此提示。

请注意: 排队系统不支持多字节主机名。

如果 **ftp** 命令找到指定主机的 **$HOME/.netrc** 自动登录项，**ftp** 会试图使用该登录项中的信息登录远程主机。**ftp** 命令也会载入任何在登录项中定义的命令宏。在某些情况下（例如，所需的密码在自动登录项中没有列出来），**ftp** 命令会在显示 ftp> 提示符之前提示输入密码。

一旦 **ftp** 命令完成自动登录，如果宏是在自动登录项中定义的，**ftp** 命令就会执行 **init** 宏。如果 **init** 宏不存在或未包含 **quit** 或 **bye** 子命令，**ftp** 命令将显示 ftp> 提示符并等待子命令。

请注意: 在提示符或在 **$HOME/.netrc** 文件中指定的远程主机名必须存在，并拥有在远程主机中定义的密码。否则，**ftp** 指令无效。

如果 Kerberos 5 是当前的认证方法: : **ftp** 命令会按在 IETF 草稿文挡 ″draft-ietf-cat-ftpsec-09.txt″ 定义的那样将此拓展用于 ftp 规范。FTP 安全性扩展将使用 “ Generic Security Service API (GSSAPI）” 安全机制来实现。GSSAPI 提供对于基础的安全性和通信机制来说是独立的服务。GSSAPI 是在 rfc 1508 和 1509 中定义的。

ftp 命令将使用 AUTH 和 ADAT 命令验证 **ftpd** 守护程序。如果两者都支持 Kerberos 认证，则它们会使用本地用户 DCE 凭证验证远程系统上的用户。如果这样失败了，且两个系统中都配置了 “标准的” 认证，则会使用上述的过程。

HostName 参数是传送文件的目的地主机名称。可选的 *Port* 参数指定了端口的标识符，通过此端口进行传输。（**/etc/services** 文件指定了缺省端口。）

标志

-C 允许用户指定: 通过 **send_file** 命令发出的文件必须在网络高速缓冲区（NBC）中经过缓存处理。此标志必须在指定了 -q 标志的情况下使用。仅当文件在无保护的情况下以二进制方式发送时此标志才适用。

-d 将有关 **ftp** 命令操作的调试信息发送给 **syslogd** 守护程序。如果您指定 **-d** 标志，您必须编辑 **/etc/syslog.conf** 文件并添加以下中的一项:

```
user.info FileName
```

OR

```
user.debug FileName
```

请注意: **syslogd** 守护程序调试级别包含信息级别消息。

如果不编辑 **/etc/syslog.conf** 文件，则不会产生消息。变更了 **/etc/syslog.conf** 文件之后，请运行 **refresh -s syslogd** 或 **kill -1** *SyslogdPID* 命令，以通知 **syslogd** 守护程序其配置文件的变更。关于调试级别的更多信息，请参考 **/etc/syslog.conf** 文件。也请参考 **debug** 子命令。

-f 导致转发凭证。如果 Kerberos 5 不是当前认证方法，则此标志将被忽略。

-g 禁用文件名中的元字符拓展。解释元字符可参考为扩展（有时叫做文件名匹配替换）文件名。请参考 **glob** 子命令。

-i 关闭多文件传送中的交互式提示。请参考 **prompt**、**mget**、**mput** 和 **mdelete** 子命令，以取得多文件传送中的提示的描述。

-K 在控制连接和数据连接上禁用在 **sys/socket.h** 文件中定义的 SO_KEEPALIVE 选项。

-k *realm* 如果远程站的域不同于本地系统的域，系统将允许用户指定远程站的域。因此，域和 DCE 单元是同义的。如果 Kerberos 5 不是当前认证方法，则此标志将被忽略。

-n 防止在起始连接中的自动登录。否则， **ftp** 命令会搜索 **$HOME/.netrc** 登录项，该登录项描述了远程主机的登录和初始化过程。请参考 **user** 子命令。

-q 允许用户指定: **send_file** 子例程必须用于在网络上发送文件。仅当文件在无保护的情况下以二进制方式发送时此标志才适用。

-v 显示远程服务器的全部响应，并提供数据传输的统计信息。当 **ftp** 命令的输出是到终端（如控制台或显示）时，此显示方式是缺省方式。

如果 stdin 不是终端，除非用户调用带有 **-v** 标志的 **ftp** 命令，或发送 **verbose** 子命令，否则 **ftp** 详细方式将禁用。

子命令

可以在 `ftp>` 提示符下输入以下 **ftp** 子命令。使用双引号 （" "） 指定包含空格字符的参数。

![*Command* [*Parameters*]] 在本地主机上调用交互的 shell。可通过 shell 命令提供带有一个或多个可选参数的可选命令。

$*Macro* [*Parameters*] 执行指定的宏（早先使用 **macdef** 子命令定义的）。参数不被扩展。

?[*Subcommand*] 显示描述子命令的帮助消息。如果您不指定 *Subcommand* 参数，**ftp** 命令将显示已知子命令的清单。

account [*Password*] 发送远程主机在授权对其资源进行访问之前可能需要的补充密码。如果密码不是命令提供的，将提示用户输入密码。密码不在屏幕上显示。

append *LocalFile* [*RemoteFile*] 将本地文件添加到远程主机文件中。如果远程文件名未指定，则将使用本地文件名，该文件名由 **ntrans** 子命令或 **nmap** 子命令生成的任何设置进行更改。添加文件的时候，**append** 子命令将为 **form**、**mode**, **struct** 和 **type** 子命令使用当前值。

ascii **type ascii** 子命令的同义词。

bell 每个文件传送过程完成之后响一声铃。

binary **type binary** 子命令的同义词。

block **mode block** 子命令的同义词。

bye 结束文件传送会话并退出 **ftp** 命令。与 **quit** 子命令相同。

carriage-control **form carriage-control** 子命令的同义词。

case 为文件名 case 设置一个切换。当 **case** 子命令开启的时候 **ftp** 命令将全部远程文件名从大写改成小写（将它们写入本地目录时）。其缺省值为关闭（这样 **ftp** 命令就会在本地目录中按大写格式写入大写的远程文件名）。

cd *RemoteDirectory* 将远程主机中的工作目录改为指定的目录。

cdup 将远程主机中的工作目录改为当前目录的父目录。

close 结束文件传送会话，但请勿退出 **ftp** 命令。已定义的宏已被擦除。与**disconnect** 子命令相同。

copylocal 切换本地拷贝。**copylocal** 缺省值为关闭。ftp 通过向自身执行 ftp 操作确认您未置零文件（如相同的主机名和相同的路径名）。将 **copylocal** 打开绕过该检查。

cr 在 ASCII 类型文件传送期间接收记录时，从回车和 line-feed 序列中除去回车字符。（**ftp** 命令在文件传送期间用回车和换行中止每一条 ASCII 类型的记录。）

远程主机（其操作系统与正在运行的操作系统不同）的记录可拥有内嵌于记录中的单一换行。要从记录定界符中区分这些内嵌的换行，请将 **cr** 子命令设置为关闭。**cr** 子命令会在开启和关闭中间进行切换。

debug [**0** | **1**] 在调试记录保持开启和关闭之间进行切换。指定 **debug** 或 **debug 1**，以打印发送到远程主机的每一个命令，并保存重新启动的控制文件。重新指定 **debug** 或 **debug 0**，停止调试记录保持。Ctrl-C 键也会保存重新启动的控制文件。

指定 **debug** 子命令将有关 **ftp** 命令操作的调试信息发送给 **syslogd** 守护程序。如果指定了 **debug** 子命令，您必须编辑 **/etc/syslog.conf** 文件并添加以下中的其中一项:

```
user.info FileName
```

OR

```
user.debug FileName
```

请注意: **syslogd** 守护程序调试级别包含信息级别消息。

如果不编辑 **/etc/syslog.conf** 文件，则不会产生消息。变更了 **/etc/syslog.conf** 文件之后，请运行 **refresh -s syslogd** 或 **kill -1** *SyslogdPID* 命令，以通知 **syslogd** 守护程序其配置文件的变更。关于调试级别的更多信息，请参考 **/etc/syslog.conf** 文件。也请参考 **ftp -d** 标志。

delete *RemoteFile* 删除指定的远程文件。

dir [*RemoteDirectory*][*LocalFile*] 将所指定的远程目录的内容清单（*RemoteDirectory*） 写入指定的本地文件（*LocalFile*）。如果 *RemoteDirectory* 参数未指定，**dir** 子命令将列出当前远程目录的内容。如果 *LocalFile* 参数未指定或是一个 -（连字符），则 **dir** 子命令将显示本地终端上的清单。

disconnect 结束文件传送会话，但不退出 **ftp** 命令。已定义的宏已被擦除。与 **close** 子命令相同。

ebcdic **type ebcdic** 子命令的同义词。

exp_cmd 在常规的和试验性的协议命令中切换。其缺省值为关闭。

file **struct file** 子命令的同义词。

form [**carriage-control** | **non-print** | **telnet**] 指定文件传送的格式。**form** 子命令修改 **type** 子命令，以便按指示的格式发送文件传送。有效的参数为 **carriage-control**、**non-print** 和 **telnet**。

carriage-control
将文件传送格式设置为 carriage-control。

non-print
将文件传送格式设置为 non-print。

telnet 将文件传送格式设置为 Telnet。Telnet 是打开对系统的连接的传输控制协议／网间协议（TCP／IP）。

get *RemoteFile* [*LocalFile*] 将远程文件复制到本地主机。如果 *LocalFile* 参数未指定，将在本地使用远程文件名，并由 **case**、**ntrans** 和 **nmap** 子命令**生成的任何设置进行更改。** **ftp** 命令在传送文件时，将使用 **type**、**form**、**mode** 和 **struct** 子命令的当前设置。

glob

切换 **mdelete**、**mget** 和 **mput** 子命令的文件名拓展（文件名匹配替换）。如果文件名匹配替换禁用，这些子命令的文件名参数将不会拓展。启用了文件名匹配替换以及在期待单一文件名的子命令中使用了模式匹配字符时，结果可能与期待的不一样。

例如，**append** 和 **put** 子命令将拓展文件名，然后仅使用所生成第一个文件名。其他 **ftp** 子命令，如 **cd**、**delete**, **get**、**mkdir**、**rename** 和 **rmdir** 不会执行文件名拓展，并从字面上接受模式匹配字符。

mput 子命令的文件名匹配替换在本地执行，其方式与 **csh** 命令的方式一样。对于 **mdelete** 和 **mget** 子命令，每个文件名都是在远程机器上分别地进行扩展的，其清单也不会合并。根据远程主机和 **ftp** 服务器的不同，目录名的拓展可与文件名的拓展不一样。

要预览目录名的扩展请使用 **mls** 子命令。

```
mls RemoteFile
```

要传送文件的整个目录子树，请勿使用 **mget** 或 **mput** 子命令，而按二进制格式传送子树的 **tar** 压缩文档。

hash

切换散列符号 （#） 打印。当 **hash** 子命令是开启的时候 **ftp** 命令会为每个所传送的数据块（1024 字节）显示一个散列符号。

help [*Subcommand*]

显示帮助信息。请参考 **?** 子命令。

image

type image 子命令的同义词。

lcd [*Directory*]

更改本地主机中的工作目录。如果您未指定目录，**ftp** 命令将使用主目录。

local *M*

type local *M* 子命令的同义词。

ls [*RemoteDirectory*] [*LocalFile*]

请将远程目录缩写的文件清单写入本地文件。如果 *RemoteDirectory* 参数未指定，**ftp** 命令将列出当前远程目录。如果 *LocalFile* 参数未指定或是一个 -（连字符），则 **ftp** 命令将显示本地终端上的清单。

macdef *Macro*

定义子命令宏。随后直到空行的行（两个连续的换行）将作为宏的文本保存。能为所有宏定义多达 16 个宏，包含 4096 个字符。在重新定义或执行 **close** 子命令前，应将宏保持为已定义的宏。

$（美元符号）和 \（反斜杠） 是 **ftp** 宏中的特殊字符。跟随一个或多个数字的 $ 符号将被调用行中对应的宏参数所替换（请参考 **$** 子命令）。$ 符号后紧随字母 i 表示将要循环该宏，$i 字符组合将被每一传递中连续的字符替换。

第一个参数用于第一个传递，第二个参数用于第二个传递，依此类推。\ 符号会防止下一个字符的特殊处理。请使用 \ 符号关闭 $ 和 \.（反斜杠句点）符号的特殊意义。

mdelete *RemoteFiles*

扩展在远程主机上 *RemoteFiles* 参数所指定的文件，并删除远程文件。

mdir [*RemoteDirectories* *LocalFile*]

扩展在远程主机上 *RemoteDirectories* 参数所指定的目录，并将这些目录的内容清单写入在 *LocalFile* 参数中定义的文件。如果 *RemoteDirectories* 参数包含模式匹配字符，**mdir** 子命令将提示输入本地文件（如果没指定）。如果 *RemoteDirectories* 参数是用空格隔开的远程目录清单，则清单中最后一个参数必须是一个本地文件名或 -（连字符）。

如果 *LocalFile* 参数是 -（连字符），**mdir** 子命令将显示本地终端上的清单。如果交互式提示启用（请参考 **prompt** 子命令），**ftp** 命令将提示用户验证最后一个参数是本地文件且不是远程目录。

mget *RemoteFiles*

请扩展在远程主机上 *RemoteFiles* 参数，并将指示的远程文件拷贝到本地主机的当前目录中。请参考 **glob** 子命令，取得更多有关文件名扩展的信息。远程文件名在本地使用，并由 **case**、**ntrans** 和 **nmap** 子命令所生成的设置进行变更。在传送文件时，**ftp** 命令将使用 **form**、**mode**、**struct** 和 **type** 子命令的当前设置。

mkdir [*RemoteDirectory*]

创建在 *RemoteDirectory* 参数（远程主机中）中指定的目录。

mls [*RemoteDirectories LocalFile*] 扩展在远程主机上在 *RemoteDirectories* 参数中指定的目录，并将所指示的远程目录的缩写文件清单写入本地文件。如果 *RemoteDirectories* 参数包含模式匹配字符，**mls** 子命令将提示输入本地文件（如果没指定）。如果 *RemoteDirectories* 参数是用空格隔开的远程目录清单，则清单中最后一个参数应是一个本地文件名或 -（连字符）。

如果 *LocalFile* 参数是 -（连字符），**mls** 子命令将显示本地终端上的清单。如果交互式提示启用（请参考 **prompt** 子命令），**ftp** 命令将提示用户验证最后一个参数是本地文件且不是远程目录。

mode [**stream** | **block**] 设置文件传送方式。如果参数未提供，则缺省值为 **stream**。

block 请将文件传送方式设置为 block。

stream 将文件传送方式设置为 stream。

modtime 显示远程机器中所指定文件的最后修改时间。如果 **ftp** 命令未连接到执行前的主机，则 **modtime** 子命令将带错误消息而终止。**ftp** 命令忽略第一个参数以外的参数。如果 *FileName* 参数未指定，**ftp** 命令将提示输入文件名。如果未给出文件名，**ftp** 命令会将用法消息发送给标准输出，并终止该子命令。

如果远程主机中有 *FileName* 参数指定的名称，且该名称指定一个文件，则 **ftp** 命令将包含文件最后修改时间的消息发送给标准输出，并终止该子命令。如果 *FileName* 指定一个目录，则 **ftp** 命令会将错误消息发送给标准输出，并终止该子命令。

注: **modtime** 子命令在允许的时候会解释元字符。

mput [*LocalFiles*] 扩展在主机的 *LocalFiles* 参数中指定的文件，并将指示的本地文件复制给远程主机。请参考 **glob** 子命令，取得更多有关文件名扩展的信息。本地文件名在远程主机中使用，并由 **ntrans** 和 **nmap** 子命令所生成的设置进行变更。 **ftp** 命令会在传送文件时，使用 **type**、**form**、**mode** 和 **struct** 子命令的当前设置。

nlist [*RemoteDirectory*][*LocalFile*] 将所指定的远程目录的内容清单（*RemoteDirectory*） 写入指定的本地文件（*LocalFile*）。如果 *RemoteDirectory* 参数未指定，**nlist** 子命令将列出当前远程目录的内容。如果 *LocalFile* 参数未指定或是一个 -（连字符），则 **nlist** 子命令将显示本地终端上的清单。

nmap [*InPattern OutPattern*] 打开或关闭文件名映射机制。如果未指定任何参数，文件名映射将关闭。如果参数已指定，在没有指定目标文件名的时候，将为 **mget** 和 **mput** 子命令以及 **get** 和 **put** 子命令映射源文件名。此子命令在本地和远程主机使用不同的文件命名约定或惯例时很有用。映射以下由 *InPattern* 和 *OutPattern* 参数设置的模式。

InPattern 参数为传入的文件名指定了模板，该文件名有可能已根据 **case** 和 **ntrans** 设置处理过。可将从 `$1` 到 `$9` 的模板变量包含进 *InPattern* 参数。除 $（美元符号）和 \$（反斜杠，美元符号）以外，*InPattern* 参数中的所有字符，都是按字面意义处理的，并用作 *InPattern* 变量之间的定界符。例如，如果 *InPattern* 参数是 `$1.$2` 且远程文件名是 `mydata.dat`，则 `$1` 的值是 `mydata`，`$2` 的值是 `dat`。

此 *OutPattern* 参数确定结果文件名。从 `$1` 到 `$9` 的变量将被它们从 *InPattern* 参数获得的值替换，而变量 `$0` 将由原文件名替换。另外，如果 *Sequence1* 不为空，序列 [*Sequence1*,*Sequence2*] 将由 *Sequence1* 替换；否则它将由 *Sequence2* 的值替换。例如，子命令:

```
nmap $1.$2.$3 [$1,$2].[$2,file]
```

会从 `myfile.data` 或 `myfile.data.old` 中产生 `myfile.data`，从 `myfile` 中产生 `myfile.file`，从 `.myfile` 中产生 `myfile.myfile`。请使用 \（反斜杠）符号防止 *OutPattern* 参数中 $（美元符号）、[（左括号）、]（右括号）和 ,（逗号）的特殊意义。

non-print **form non-print** 子命令的同义词。

ntrans [*InCharacters* [*OutCharacters*]] 打开或关闭文件名字符翻译机制。如果未指定任何参数，字符翻译将关闭。如果参数已指定，在没有指定目标文件名的时候，将为 **mget** 和 **mput** 子命令以及 **get** 和 **put** 子命令翻译源文件名中的字符。

此子命令在本地和远程主机使用不同的文件命名约定或惯例时很有用。字符翻译遵从 *InCharacters* 和 *OutCharacters* 参数设置的模式。匹配 *InCharacters* 参数中字符的源文件名中的字符将由 *OutCharacters* 参数中对应的字符替换。

如果 *InCharacters* 参数指定的字符串比 *OutCharacters* 参数指定的字符串长，而且其在 *OutCharacters* 参数中无对应的字符，则 *InCharacters* 参数中的字符将被删除。

open *HostName* [*Port*] 在由 *HostName* 参数所指定主机中建立对 FTP 服务器的连接。如果可选的端口号已指定 **ftp** 命令将尝试连接该端口中的服务器。如果已设置自动登录功能（即，该 **-n** 标志没有在命令行中指定），则 **ftp** 命令就会试图让用户登录到 FTP 服务器。

您还必须拥有其中带有正确信息的 **$HOME/.netrc** 文件和正确的许可集。**.netrc** 文件应在您的主目录中。

passive 切换文件传送的被动方式。在被动方式关闭的情况下调用文件传送命令（如 **get** 、**mget** 和 **put**，或 **mput**）时，**ftp** 服务器就会打开对客户机的数据连接。在被动方式下，客户机在发送和接收数据的时候打开对主机的数据连接。

private 请将保护级别设置为 “private”。在此级别，数据的保护既完整而又机密。

prompt 切换交互式提示。如果交互式提示启用（缺省值）**ftp** 命令在检索、发送或删除文件（在 **mget**、**mput** 和 **mdelete**）之前会提示进行验证操作。否则，**ftp** 命令会根据所有指定的文件来执行。

protect 此指令会返回保护的当前级别。

proxy [*Subcommand*] 在辅助控制连接中执行 **ftp** 命令。此子命令允许 **ftp** 子命令同时连接两个远程 FTP 服务器，以便在两个服务器之间传送文件。第一个 **proxy** 子命令应是建立次要控制连接的 **open** 子命令。请输入 **proxy ?** 子命令，查看其他在二级连接中可执行的 **ftp** 子命令。

以下子命令在由 **proxy** 子命令开头时的表现会不一样:

- **open** 子命令在自动登录过程中不定义新的宏。
- **close** 子命令不会擦除现存的宏定义。
- **get** 和 **mget** 子命令从主连接中的主机上将文件传送到二级连接的主机上。
- **put**、 **mput** 和 **append** 子命令从第二连接中的主机上将文件传送到主连接的主机上。
- **restart** 子命令可由 **proxy** 命令进行处理。
- **status** 子命令显示出准确的信息。

文件传送要求第二连接上的 FTP 服务器必须支持 PASV（passive）指令。

put *LocalFile* [*RemoteFile*] 将本地文件存储到远程主机中。如果您未指定 *RemoteFile* 参数，**ftp** 命令将使用本地文件名命名远程文件，而远程文件名由 **ntrans** 和 **nmap** 子命令所生成的设置加以更改。 **ftp** 命令会在传送文件时，将使用 **type**、**form**、**mode** 和 **struct** 子命令的当前设置。

pwd 显示远程主机中的当前目录名。

quit 关闭连接并退出 **ftp** 命令。与 **bye** 子命令相同。

quote *String* 逐字地将由 *String* 参数指定的字符串发送到远程主机。执行 **remotehelp** 或 **quote help** 子命令，显示 *String* 参数的有效值清单。

注: 涉及数据传输的 “Quoting” 命令会产生不可预见的结果。

record **struct record** 子命令的同义词。

recv *RemoteFile* [*LocalFile*] 将远程文件复制到本地主机。与 **get** 子命令相同。

reinitialize 通过刷新所有 I/O 并允许完成传输操作来对 FTP 会话进行重新初始化。按照用户已开始 FTP 会话但未登录到远程主机的情况复位所有缺省值。

remotehelp [*Subcommand*] 请求从远程 FTP 服务器获得帮助。

rename *FromName ToName* 重新命名远程主机上的文件。

reset 清除回复队列。此子命令将使命令分析重新同步。

restart get | put | append 请在生成上一个检查点的位置重新启动文件传送。要想顺利运行，该子命令应与异常终止子命令相同，包括结构、类型和格式。有效参数为 **get**、**put** 和 **append**。

rmdir *RemoteDirectory* 除去由远程主机中的 *RemoteDirectory* 参数所指定的远程目录。

runique (ReceiveUnique) 在子命令 **get** 和 **mget** 期间，切换为本地目的文件创建唯一文件名称的功能。如果此功能为“关闭”状态（预设值），**ftp** 命令会覆盖本地文件。否则，如果本地文件名与为本地目的文件指定的名称相同，**ftp** 将使用.1 修改为本地目的文件指定的名称。如果本地文件已使用了新名称，**ftp** 会将后缀 .2 添加给所指定的名称。如果本地文件已使用了此第二个名称，则 **ftp** 命令会在找到唯一的文件名或未找到唯一文件名但到达 .99 之前继续增加后缀。如果 **ftp** 命令找不到唯一的文件名，**ftp** 会报告错误，传输过程也不会发生。请注意 **runique** 子命令不会影响从 shell 命令产生的本地文件名。

safe 将保护级别设置为 “safe”。在此级别，数据的完整性会得到保护。

send *LocalFile* [*RemoteFile*] 将本地文件存储到远程主机中。其方式与 **put** 子命令相同。

sendport 切换对于 FTP PORT 指令的使用。根据预设，**ftp** 命令在为每个数据传输建立连接的时候会使用 PORT 指令。当禁用 PORT 指令时，**ftp** 不会在数据传输中使用 PORT 命令。PORT 指令在处理某些 FTP 服务器时很有用，那些服务器在未正确指示指令已被接受时会忽略 PORT 指令。

site *Args* 使用 **chmod** 命令显示或设置空闲超时周期、显示或设置文件创建 umask 或更改文件的许可权。*Args* 参数可能的值是 **umask** 和 **chmod**。

size *RemoteFile* 以字节的方式显示 *RemoteFile* 参数所指定的远程文件的大小。

status 显示 **ftp** 命令的当前状态以及子命令的状态。

stream **mode stream** 子命令的同义词。

struct [**file** | **record**] 设置数据传送结构类型。有效的参数为 **file** 和 **record**。

file 将数据传送结构类型设置为 file。

record 将数据传送结构类型设置为 record。

sunique (Send/Store Unique) 在子命令 **put** 和 **mput** 期间，切换为远程目的文件创建唯一文件名称的功能。如果此功能为“关闭”状态（预设值），**ftp** 命令会覆盖远程文件。否则，如果远程文件名与为远程目的文件指定的名称相同，远程 FTP 服务器会修改远程目的文件的名称。请注意远程服务器必须支持 STOU 指令。

system 显示在远程机器上运行的操作系统类型。

telnet **form telnet** 子命令的同义词。

tenex **type tenex** 子命令的同义词。

trace 切换包跟踪。

type [**ascii** | **binary** | **ebcdic** | **image** | **local** *M* | **tenex**] 设置文件传送类型。有效参数是 **ascii**、**binary**、**ebcdic**、**image**、**local** *M* 和 **tenex**。如果参数未指定，则当前类型被显示。缺省类型为 **ascii**; **binary** 类型可能会比 **ascii** 更有效。

ascii 将文件传送类型设置为网络 ASCII。此类型为缺省值。文件传送使用二进制映像传送可能会更有效。请参考 **binary** 参数，以取得更多信息。

binary 将文件传送类型设置为二进制映像。此类型可能比 ASCII 传送更有效。

ebcdic 将文件传送类型设置为 EBCDIC。

image 将文件传送类型设置为二进制映像。此类型可能比 ASCII 传送更有效。

local *M*
将文件传送类型设置为本地。*M* 参数定义每计算机字位的十进制数。此参数没有缺省值。

tenex 将文件传送类型设置为 TENEX 机器需要的类型。

user *User* [*Password*] [*Account*] 识别远程 FTP 服务器的本地用户（*User*）。如果未指定 *Password* 或 *Account* 参数，而远程服务器需要它，则 **ftp** 命令会提示输入本地的密码和帐户。如果 *Account* 参数是必须的，则 **ftp** 命令会在远程登录过程完成后将它发送到远程服务器。

注: 除非通过在命令行上指定 **-n** 标志来禁用自动登录，否则 **ftp** 命令将 *User*、*Password* 和 *Account* 参数自动地发送给远程服务器用于初始连接。您亦需主目录中的 **.netrc** 文件以便发布自动登录。

verbose 切换详细方式。如果详细方式启用（缺省值），**ftp** 命令将显示远程 FTP 服务器的全部响应。另外，**ftp** 会在传输完成后显示所有文件传送的统计信息。

示例

1. 要调用 **ftp** 命令，请登录 canopus 系统，显示本地帮助信息，显示远程帮助信息，显示状态，切换 **bell**、**prompt**、**runique**、**trace** 和 **verbose** 子命令，然后再退出，请输入:

```
$ ftp canopus
Connected to canopus.austin.century.com.
220 canopus.austin.century.com FTP server (Version 4.1 Sat Nov 23 12:52:09 CST 1991) ready.
Name (canopus:eric): dee
331 Password required for dee.
Password:
230 User dee logged in.
ftp> help
Commands may be abbreviated. Commands are:
!          delete          mdelete          proxy          runique
$          debug           mdir             sendport       send
account    dir             mget             put            size
append     disconnect      mkdir            pwd            status
ascii      form            mls              quit           struct
bell       get             mode             quote          sunique
binary     glob            modtime          recv           system
bye        hash            mput             remotehelp     tenex
case       help            nmap             rstatus        trace
cd         image           nlist            rhelp          type
cdup       lcd             ntrans           rename         user
close      ls              open             reset          verbose
cr         macdef          prompt           rmdir          ?
clear      private         protect          safe
ftp> remotehelp
214-The following commands are recognized(* =>'s unimplemented).
 USER   PORT   RETR   MSND*  ALLO   DELE   SITE*  XMKD   CDUP
 PASS   PASV   STOR   MSOM*  REST*  CWD    STAT*  RMD    XCUP
 ACCT*  TYPE   APPE   MSAM*  RNFR   XCWD   HELP   XRMD   STOU
 REIN*  STRU   MLFL*  MRSQ*  RNTO   LIST   NOOP   PWD
 QUIT   MODE   MAIL*  MRCP*  ABOR   NLST   MKD    XPWD
 AUTH   ADAT   PROT   PBSZ   MIC    ENC    CCC
214 Direct comments to ftp-bugs@canopus.austin.century.com.
ftp> status
Connected to canopus.austin.century.com.
No proxy connection.
Mode: stream; Type: ascii; Form: non-print; Structure: file
Verbose: on; Bell: off; Prompting: on; Globbing: on
Store unique: off; Receive unique: off
Case: off; CR stripping: on
Ntrans: off
Nmap: off
Hash mark printing: off; Use of PORT cmds: on
ftp> bell
Bell mode on.
ftp> prompt
```

```
Interactive mode off.
ftp> runique
Receive unique on.
ftp> trace
Packet tracing on.
ftp> verbose
Verbose mode off.
ftp> quit
$
```

2. 要调用 **ftp** 命令，请登录 canopus 系统，打印工作目录，更改工作目录，将文件传送类型设定为 ASCII 向远程主机发送本地文件，将工作目录改为父目录，然后退出，请输入:

```
$ ftp canopus
Connected to canopus.austin.century.com.
220 canopus.austin.century.com FTP server (Version 4.1 Sat Nov 23 12:52:09 CST 1991) ready.
Name (canopus:eric): dee
331 Password required for dee.
Password:
230 User dee logged in.
ftp> pwd
257 "/home/dee" is current directory.
ftp> cd desktop
250 CWD command successful.
ftp> type ascii
200 Type set to A.
ftp> send typescript
200 PORT command successful.
150 Opening data connection for typescript (128.114.4.99,1412).
226 Transfer complete.
ftp> cdup
250 CWD command successful.
ftp> bye
221 Goodbye.
$
```

3. 要调用具有自动登录功能的 **ftp** 命令（使用 **.netrc** 文件），请打开与 canopus 系统的会话，登入，将工作目录改为父目录，显示工作目录，列出当前目录的内容，删除文件，将当前目录的内容清单写入本地文件，关闭会话，然后退出，请输入:

```
$ ftp canopus
Connected to canopus.austin.century.com.
220 canopus.austin.century.com FTP server (Version 4.1 Sat Nov 23 12:52:09 CST 1991) ready.
331 Password required for dee.
230 User dee logged in.
ftp> cdup
250 CWD command successful.
ftp> pwd
257 "/home" is current directory.
ftp> dir
200 PORT command successful.
150 Opening data connection for /usr/bin/ls (128.114.4.99,1407)
(0 bytes).
total 104
drwxr-xr-x   2 system        32 Feb 23 17:55 bin
Drwxr-xr-x  26 rios        4000 May 30 17:18 bin1
drwxr-xr-x   2 system        32 Feb 23 17:55 books
drwxrwxrwx  18 rios        1152 Jun  5 13:41 dee
-r--r--r--   1 system      9452 May 17 12:21 filesystems
drwxr-xr-x   2 system        32 Feb 23 17:55 jim
drwxr-xr-x   5 system        80 Feb 23 17:55 krs
drwxrwxrwx   2 rios       16432 Feb 23 17:36 lost+found
-rwxr-xr-x   1 rios        3651 May 24 16:45 oldmail
```

```
drwxr-xr-x   2 system       256 Feb 23 17:55 pubserv
drwxrwxrwx   2 system       144 Feb 23 17:55 rein989
drwxr-xr-x   2 system       112 Feb 23 17:55 reinstall
226 Transfer complete.
ftp> delete oldmail
250 DELE command successful.
ftp> mdir /home/dee/bin binlist
output to local-file: binlist? y
200 PORT command successful.
150 Opening data connection for /usr/bin/ls (128.114.4.99,1408) (0 bytes).
226 Transfer complete.
ftp> close
221 Goodbye.
ftp> quit
$
```

文件

/usr/samples/tcpip/netrc	包含 **.netrc** 样本文件。
/etc/syslog.conf	包含 syslogd 守护程序的配置信息。

相关信息

csh 命令、**kill** 命令、 **rcp** 命令、**refresh** 命令、**rlogin** 命令、**rsh** 命令、 **stty** 命令、**telnet** 命令和 **tftp** 命令。

ftpd 守护程序、**syslogd** 守护程序。

.netrc 文件格式。

《*网络与通信管理*》中的『使用 ftp 和 rcp 命令的文件传输』。

《*网络与通信管理*》中的『通信和网络』。

《*网络与通信管理*》中的『认证和安全 rcmd』。

《*性能管理*》中的『网络选项可调参数』。

---

ftpd 守护程序

用途

为因特网 FTP 协议提供服务器功能。

语法

注: **ftpd** 守护程序一般由 inetd 守护程序启动。也可以用 SRC 命令从命令行来控制。

/usr/sbin/ftpd [**-d**] [**-f**] [**-ff**] [**-k**] [**-l**] [**-U**] [**-t** *TimeOut*] [**-t** *MaxTimeOut*] [**-s**] [**-u** *OctalVal*] [**-q** [**-C**]]

描述

/usr/sbin/ftpd 守护程序是 DARPA 因特网文件传输协议（FTP）服务器进程。**ftpd** 守护程序使用传输控制协议（TCP）侦听 **/etc/services** 文件中 **ftp** 命令服务规范指定的端口。

ftpd 守护程序的修改可以通过系统管理界面程序（SMIT）或者系统资源控制器（SRC）来完成，通过编辑 **/etc/inetd.conf** 或者 **/etc/services**文件来实现。不建议在命令行中输入 `ftpd`。**ftpd** 守护程序如果在 **/etc/inetd.conf** 文件中未注释，缺省情况下它将启动。

inetd 守护程序从 /etc/inetd.conf 文件和 /etc/services 文件中获取信息。

如果您更改 /etc/inetd.conf 或者 /etc/ services 文件，运行 **refresh -s inetd** 或者 **kill -1** *InetdPID* 命令以通知 **inetd** 守护程序关于配置文件的改动。

ftpd 守护程序根据 csh 命令的约定来扩展文件名。 此命令可以使用如下的元字符串: *（星号），? （问号），[ ]（左右方括号），{ } （左右花括号） 以及 ~ （代字号）。

ftpaccess.ctl 文件

搜索 **/etc/ftpaccess.ctl** 文件，寻找以 **allow:**、**deny:**、**readonly:**、**writeonly:**、**readwrite:**、**useronly:**、**grouponly:**、**herald:** 和 / 或 **motd:** 开始的行。其余的行忽略。如果文件不存在，则所有主机均允许 ftp 访问。**allow:** 和 **deny:** 行用于限制对主机的访问。 **readonly:**, **writeonly:** 和 **readwrite:** 行用于限制 ftp 的读（获取）写（存放）。**useronly:** 和 **grouponly:** 行用于定义匿名用户。 **herald:** 和 **motd:** 行用于登录前后的多行消息。

/etc/ftpaccess.ctl 中所有行的语法形式:

```
keyword: value, value, ...
```

每个关键字可以指定一个或者多个值。同样的关键字可以出现在很多行里。/etc/ftpaccess.ctl 文件中的每一行最多包含 1024 个字符，超过的将被忽略。

allow: 和 **deny:** 行的语法是:

```
allow: host, host, ...
deny: host, host, ...
```

如果指定 **allow:** 行，则仅允许所有 **allow:** 行中都列出的主机访问 ftp。其他的主机将拒绝访问 ftp。如果没有 **allow:** 行，则除 **deny:** 行中指定的主机外，所有主机都将能够访问 ftp。主机可以通过主机名或者 IP 地址来指定。

readonly:、**writeonly**: 和 **readwrite**: 行的语法是:

```
readonly: dirname, dirname, ...
writeonly: dirname, dirname, ...
readwrite: dirname, dirname, ...
```

readonly: 行列出了只读目录， **writeonly:** 行列出了只写目录。仅允许写入的目录中拒绝读访问，只读目录中拒绝写访问。除非指定了 **readwrite:** 行，否则所有其他目录都被授予访问权。如果指定了 **readwrite:** 行，则只有在 **readwrite:** 行和 / 或 **readonly:** 行中列出的目录可以进行读访问，同样，只有在 **readwrite:** 行和 / 或 **writeonly:** 行中列出的目录可以进行写访问。同时，这些行的值可以是 “ALL” 或者 “NONE”。

useronly:、**puseronly:**、**grouponly:** 和 **pgrouponly:** 行的语法为:

```
useronly: username, username, ...
puseronly: username, username, ...
grouponly: groupname, groupname, ...pgrouponly: groupname, groupname, ...
```

用户名在 **/etc/passwd** 中，组名在 **/etc/group** 中。**useronly:** 和 **puseronly:** 行定义匿名用户。**grouponly:** 和 **pgrouponly:** 行定义一组匿名用户。这些匿名用户类似于在 ftp 活动中的匿名用户，他们被限制在自己的主目录下。**由于他们没有密码保护**，**useronly:** 和 **grouponly:** 行定义的匿名用户类似于 anonymous 用户。**puseronly:** 和 **pgrouponly:** 行定义有密码保护的匿名用户。

注: 必须为 **puseronly:** 和 **pgrouponly:** 用户创建密码并禁用登录。

herald: 和 **motd:** 行的语法如下:

```
herald: path
motd: on|off
```

path 是包含将在登录前显示的多行 herald 的文件的全路径名称。当 **motd:** 行的值为 'on' 时, **$HOME/motd** 文件包含登录后将显示的多行消息。如果用户是定义好的匿名用户, **/etc/motd** 文件包含登录后将显示的多行消息。(请注意 /etc/motd 是在匿名用户的登陆根目录下)。 **motd:** 行的缺省值是 off。

如果当前的认证方法是标准操作系统的认证方法:

在 **ftpd** 守护程序可以为客户进程传输文件之前, 它必须认证客户进程。**ftpd** 守护程序根据如下规则认证客户进程:

- 用户必须在密码数据库中拥有密码, **/etc/security/passwd**。(如果用户密码非空, 客户机进程必须提供此密码。)
- 用户名不能出现在 **/etc/ftpusers** 文件中。
- 用户的登录 shell 程序必须出现在 **/etc/security/login.cfg** 文件的 shell 属性中。
- 如果用户名为 `anonymous`、`ftp` 或 **/etc/ftpaccess.ctl** 文件中定义的匿名用户, 则匿名 FTP 帐户必须在密码文件中进行定义。在这种情况下, 客户机进程可以使用任何密码登陆。一般约定, 密码为客户主机的名称。 **ftpd** 守护程序采用了特殊的办法来限制客户进程对匿名帐号的访问。

如果 Kerberos 5 是当前的认证方法:

ftpd 守护程序允许在以下条件均满足时进行访问:

- ftp 客户机的本地用户拥有当前的 DCE 凭证。
- 本地系统和远程系统均支持 **AUTH** 命令。
- 远程系统接受了 DCE 凭证, 并认为以此足以访问远程帐户。请参阅 **kvalid_user** 函数以获取更多信息。

文件传送协议子树方针

当处理某个匿名 FTP 用户时, 服务器在 FTP 用户帐户的主目录中执行 **chroot** 命令。为获得较高的安全性, 请在构造 FTP 子树时执行以下规则:

~ftp	创建 root 用户的主目录, 方式为 r-xr-xr-x (555)。
~ftp/bin	创建 root 用户的目录, 并且对其他人是不可写的。 **ls** 程序必须在此目录中以支持列表命令。此程序应该有 111 方式。
~ftp/etc	创建 root 用户的目录, 并且对其他人是不可写的。
~ftp/pub	在 777 方式下创建此目录, 并被 FTP 所有。用户应该将匿名帐户可访问的文件放在此目录下。

注: shell 脚本 **/usr/samples/tcpip/anon.ftp** 使用以上规则为您设置匿名 FTP 帐户。

当处理 /etc/ftpaccess.ctl 中定义的匿名 FTP 用户时, 服务器在用户帐户的主目录下执行 **chroot** 命令。为获得更好的安全性, 当构造用户子树时请执行以下规则:

~user 创建 root 用户的主目录, 方式为 `r-xr-xr-x` (555)。

~user/bin

创建 root 用户的目录, 并对其他人不可写。**ls** 程序必须在此目录中以支持列表命令。此程序应该有 111 方式。

~user/etc

创建 root 用户的目录，并对其他人不可写。

~user/pub

以 777 方式创建此目录，并被用户所有。用户应该将可以通过匿名帐户访问的文件放在此目录中。

注: shell 脚本 **/usr/samples/tcpip/anon.users.ftp** 使用以上规则为您设置匿名 FTP 帐户。

服务器必须以 root 用户运行，来创建有特权端口号的套接字。服务器维护一个登录用户的有效的用户标识当在套接字上绑定地址时回复给 root 用户。

支持文件传送协议请求

ftpd 守护程序目前支持以下 FTP 请求:

ABOR 终止前一次命令。
ACCT 指定帐户（被忽略）。
ADAT 指定认证 / 安全数据。
ALLO 分配存储器（空）。
APPE 添加到文件。
AUTH 指定认证 / 安全机制。
CCC 指定清除命令通道。
CDUP 转到当前工作目录的父目录。
CWD 更改工作目录。
DELE 删除文件。
ENC 指定私有保护命令。
HELP 给出帮助信息。

LIST 在目录中给出列表文件（此 FTP 请求和 ls -lA 命令是同样的）。
MKD 创建目录。
MDTM 显示文件最后的修改时间。
MIC 指定完整性保护命令。
MODE 指定数据传输方式。
NLST 给出目录下文件的名称列表（此 FTP 请求和 ls 命令是同样的）。
NOOP 无操作。
PASS 指定密码。
PASV 为服务器间的传输做准备。
PBSZ 指定保护缓冲区的大小。
PORT 指定数据连接端口。
PROT 指定数据通道保护级别。
PWD 显示当前工作目录。
QUIT 终止会话。
RETR 检索文件。
RMD 除去目录。
RNFR 指定重命名源文件名。
RNTO 指定重命名目标文件名。
SITE SITE 请求支持以下非标准的或者 UNIX 专用的命令:

UMASK
更改 umask（ **SITE UMASK 002** ）。

IDLE 设置闲置时间（ **SITE IDLE 60** ）。

CHMOD
更改文件的方式（ **SITE CHMOD 755** *文件名*）。

HELP 给出帮助信息（ **SITE HELP** ）。

SIZE 返回当前文件的大小。
STAT 返回服务器的状态。
STOR 存储文件。
STOU 使用唯一的文件名存储文件。
STRU 指定作为文件结构传输的数据结构。
SYST 显示服务器系统的操作系统类型。
TYPE 用 Type 参数指定数据传输类型。
USER 指定用户名。
XCUP 更改当前工作目录的父目录（通常不使用）。
XCWD 更改当前目录（通常不使用）。
XMKD 创建目录（通常不使用）。
XPWD 打印当前工作目录（通常不使用）。
XRMD 除去目录（通常不使用）。

其余在因特网 RFC 959 中定义的 FTP 请求被识别，但是不执行。**MDTM** 和 **SIZE** 请求在 RFC 959 中没有指定，但它们已经被列入下次的 FTP RFC 更新计划。

如果在数据传输过程中收到 **STAT** 请求而且前面有 Telnet **IP** 信号和 **SYNCH** 信号，则返回传输状态。

ftpd 守护程序应该通过使用系统管理界面程序（SMIT）或者修改 /etc/inet.conf 文件来被控制。不建议在命令行中输入 `ftpd` 。

使用系统资源控制器来操作 fptd 守护程序

ftpd 守护程序是 inetd 守护程序的子服务器，而 inetd 守护程序则是系统资源控制器的子系统（SRC）。**ftpd** 守护程序是 tcpip SRC 子系统组的成员。此守护程序在 /etc/inetd.conf 文件中缺省启用，它可以通过以下 SRC 命令来进行操作:

startsrc 启动一个子系统、子系统组，或是一个子服务器。
stopsrc 终止子系统、子系统组或子服务器。
lssrc 获取一个子系统，子系统组，或是一个子服务器的状态。

标志

-C 允许用户指定：通过 **send_file** 命令发出的文件必须在网络高速缓冲区（NBC）中经过缓存处理。此标志必须在指定了 -q 标志的情况下使用。仅当文件在无保护的情况下以二进制方式发送时此标志才适用。

-d 向 syslogd 守护程序发送关于 ftpd 守护程序操作的调试信息。如果指定了 **-d** 标志，必须编辑 **/etc/syslog.conf** 文件，然后增加如下条目:

```
daemon.debug FileName
```

注: **syslogd** 守护程序的调试级别包括信息级别消息。

如果不编辑 **/etc/syslog.conf** 文件，则没有消息产生。在修改完 **/etc/syslog.conf** 文件后，运行 **refresh -s syslogd** 命令，或者 **kill -1** *SyslogdPID* 命令，以通知 **syslogd** 守护程序其配置文件的更改。要获取更多关于调试级别的信息，请参考 **/etc/syslog.conf** 文件。

-f 当客户机请求服务器连接回特定端口时，禁用特权端口的检查。在缺省情况下，**ftpd** 不允许客户机请求连接到特权端口，这是出于安全方面的考虑。

-ff 当客户机请求服务器连接回特定客户机端口时，禁用特权端口和匹配用于控制连接的那个的 IP 地址的检查。使用此标志后，客户机可以请求服务器向备用的主机或者接口发送数据。在缺省情况下，**ftpd** 出于安全考虑，不允许此操作。

-k 在数据传输套接字上，设置在 /sys/socket.h 文件中定义的 **SO_KEEPALIVE** 选项，以启用在事件 TCP/IP 暂挂中的数据传输到超时。空闲的间隔时间依 system-wide 的取值而定，此参数通过 no 命令中的 tcp_keepidle 和 tcp_keepintvl 选项指定。如果没有此标志，**ftpd** 数据传输将不会超时。

-l 向 **syslogd** 守护程序发送关于 ftpd 守护程序的日志信息。如果指定 **-l** 标志，必须编辑 **/etc/syslog.conf** 文件并增加如下条目:

```
daemon.info FileName
```

如果不编辑 **/etc/syslog.conf** 文件，则不会产生消息。当修改完 **/etc/syslog.conf** 文件，运行 **refresh -s syslogd** 命令或者 **kill -1** *SyslogdPID* 命令以通知 **syslogd** 守护程序其配置文件的修改。关于调试级别的更多信息，请参考 **/etc/syslog.conf** 文件。

-q 允许用户指定: **send_file** 子例程必须用于在网络上发送文件。仅当文件在无保护的情况下以二进制方式发送时此标志才适用。

-t*TimeOut* 经过指定的秒数后退出已经停止的会话，秒数通过 *TimeOut* 变量来指定。缺省值是 15 分钟（900 秒）。超时同时应用于数据连接和控制连接。

-t *MaxTimeOut* 经过指定的最大秒数后退出已经停止的会话，最大秒数通过 *MaxTimeOut* 变量来指定。缺省值是 2 小时（7200 秒）。

-s 打开 socket-level 调试。

-u *OctalVal* 设置 **ftpd** 守护程序的 umask。*OctalVal* 变量必须指定为八进制的值，用来定义 umask。缺省的 umask 是八进制的 027，它导致了 rw-r**- 的文件许可。**

-U 在传输中保持文件未锁定。如果此标志通过 **/usr/sbin/ftpd 来指定**，则文件在传输过程中仍可打开。

安全性

ftpd 守护程序是服务名称为 *ftp* 的支持 PAM 的应用程序。对认证使用 PAM 的系统范围配置是通过在 **/etc/security/login.cfg** 的 **usw** 节中，将 **auth_type** 属性值修改为 PAM_AUTH 作为 root 用户来设置的。

启用 PAM 时使用的认证机制取决于 **/etc/pam.conf** 中 **ftp** 服务的配置。**ftpd** 守护程序需要 **/etc/pam.conf** 条目用于 **auth**、**accoun** 和 **session** 模块类型。以下列出了 **/etc/pam.conf** 中对 **ftp** 服务的建议的配置:

```
#
# AIX ftp configuration
#
ftp auth      required     /usr/lib/security/pam_aix

ftp account   required     /usr/lib/security/pam_aix

ftp session   required     /usr/lib/security/pam_aix
```

示例

注: 通过使用 SMIT 或编辑 **/etc/inetd.conf** 文件可以指定 **ftpd** 守护程序的参数。

1. 启动 **ftpd** 守护程序，请输入:

   ```
   startsrc -t ftp
   ```

   startsrc 命令带 **-t** 标志启动 **ftpd** 子服务器。您必须使用 **-t** 标志来指定子服务器。否则，命令不会正确执行。

2. 要停止 **ftpd** 守护程序，一般输入:

   ```
   stopsrc -t ftp
   ```

   stopsrc 命令带 **-t** 标志停止 **ftpd** 子服务器。 **stopsrc** 命令允许启动所有暂挂的连接，并完成所有现存的连接，但不允许启动新的连接。您必须使用 **-t** 标志来指定子服务器。否则，命令不会正确执行。

3. 要强制中止 **ftpd** 守护程序和所有 **ftpd** 连接，请输入:

   ```
   stopsrc -t -f ftp
   ```

stopsrc 命令和 **-t -f** 标志迫使 **ftpd** 子服务器停止。它立刻中止所有暂挂的连接和现存的连接。

4. 要显示关于 **ftpd** 守护程序的简短状态报告，请输入:

```
lssrc -t ftp
```

lssrc 命令和 **-t** 标志返回守护程序的名称、进程标识和状态（活动或者中止）。您必须使用 **-t** 标志来指定子服务器。否则，命令不会正确执行。

文件

/etc/locks/ftpd	包含内部锁和进程标识存储器。
/etc/group	包含组的密码。
/etc/passwd	包含用户的密码。
/etc/security/login.cfg	包含登录和用户认证的配置信息。
/etc/security/passwd	包含加密后的密码。
/etc/syslog.conf	包含 syslogd 守护程序的配置信息。
/usr/samples/tcpip/anon.ftp	包含示例 shell 脚本，通过它可以设置匿名 FTP 帐户。此文件也包含使用说明书。

相关信息

ftp 命令、 **lssrc** 命令、 **kill** 命令、 **no** 命令、 **rcp** 命令、 **refresh** 命令、 **rlogin** 命令、 **rsh** 命令、 **startsrc** 命令、 **stopsrc** 命令以及 **telnet** 命令。

inetd 守护程序、 **syslogd** 守护程序。

kvalid_user 函数。

/etc/ftpusers 文件格式、 **/etc/inetd.conf** 文件格式、 **/etc/services** $HOME/.k5login 文件格式。

《*网络与通信管理*》中的『TCP/IP 守护程序』。

《*网络与通信管理*》中的『认证和安全 rcmd』。

《*性能管理*》中的『网络选项可调参数』。

---

fuser 指令

用途

使用文件或文件结构识别进程。

语法

fuser [**-c** | **-d** | **-f**] [**-k** | **-K** { *SignalNumber* | *SignalName* }] [**-u**] [**-x**] [**-V**]*File* ...

描述

此 **fuser** 命令列出了本地进程的进程号，那些本地进程使用 *File* 参数指定的本地或远程文件。对于阻塞特别设备，此命令列出了使用该设备上任何文件的进程。

每个进程号后面都跟随一个字母，该字母指示进程如何使用文件。

c 将此文件作为当前目录使用。

e 将此文件作为程序的可执行对象使用。
r 将此文件作为根目录使用。
s 将此文件作为共享库（或其他可装载对象）使用。

进程号被写入标准输出（在进程号之间有空格的行中）。一个换行符被写入标准错误（在每个文件操作数的最后一个输出之后）。其他所有输出被写入标准错误。

此 **fuser** 命令不会检测有 mmap 区域的进程，其中相关的文件描述符已从此被关闭。

标志

-c 包含 *File* 的文件系统中关于任何打开的文件的报告。
-d 暗示使用了 **-c** 和 **-x** 标志。关于任何与文件系统（自父目录删除的）无链接的打开文件的报告。当与 **-V** 标志一起使用时，它也会报告被删除文件的节点号和大小。
-f 仅对文件的打开实例报告。
-K *SignalNumber* | *SignalName* 将指定信号发送到每个本地进程。仅有 root 用户能终止另一用户的进程。信号可以指定为信号名称（如 -9）或 KILL（用于 SIGKILL 信号）。*SignalName* 的有效值是 **kill -l** 命令所显示的那些值。
-k 将 **SIGKILL** 信号发送到每个本地进程。仅有 root 用户能终止另一用户的进程。
注: **fuser -k** 或 **-K** 可能无法检测和杀死程序开始运行后立即创建的新进程。
-u 为进程号后圆括号中的本地进程提供登录名。
-V 提供详细输出。
-x 与 **-c** 或 **-f** 连用，报告除标准 fuser 输出以外的可执行的和可载入的对象。

示例

1. 要列出使用 **/etc/passwd** 文件的本地进程的进程号，请输入:

   ```
   fuser /etc/passwd
   ```

2. 要列出使用 **/etc/filesystems** 文件的进程的进程号和用户登录名，请输入:

   ```
   fuser -u /etc/filesystems
   ```

3. 要终止使用给定文件系统的所有进程，请输入:

   ```
   fuser -k -x -u -c /dev/hd1
   ```

   或者

   ```
   fuser -kxuc /home
   ```

   任一命令都列出了进程号和用户名，然后终止每个正在使用 **/dev/hd1 (/home)** 文件系统的进程。仅有 root 用户能终止属于另一用户的进程。如果您正在试图卸下 **/dev/hd1** 文件系统，而一个正在访问 **/dev/hd1** 文件系统的进程不允许这样，您可能希望使用此命令。

4. 要列出正在使用已从给定文件系统删除的文件的全部进程，请输入:

   ```
   fuser -d /usr
   ```

文件

/dev/kmem 用于系统映像。
/dev/mem 也用于系统映像。

相关信息

kill 命令、**killall** 命令、**mount** 命令和 **ps** 命令。

关于用户的标识和认证、自主访问控制、可信计算库和审计的更多信息，请参阅《*安全性*》。

---

fwtmp 命令

用途

通过从标准输入中按照 wtmp 格式读取二进制记录，处理连接时的记帐记录，再将它们转换为格式化的 ASCII 记录。ASCII 版本在需要编辑坏记录时有用。

语法

/usr/sbin/acct/fwtmp [**-i**] [**-c**] [**-X**]

描述

fwtmp 命令通过从标准输入中以 **wtmp** 格式读取二进制记录并将它们转换为格式化的 ASCII 记录来操作记帐记录。

标志

-i 接受 utmp 格式的 ASCII 记录作为输入。
-c 将输出转化为 **utmp** 格式的二进制记录。
-ic 将 ASCII utmp 格式的输入记录转化为二进制输出记录。
-X 打印每个用户名的全部可用字符，而不是截断为前 8 个字符。

安全性

访问控制: 这些命令应该只对 adm 组内成员授权执行 （x） 访问。

示例

1. 要将 **wtmp** 格式的二进制记录转换为名称为 dummy.file 的 ASCII 记录，请输入:

   ```
   /usr/sbin/acct/fwtmp < /var/adm/wtmp > dummy.file
   ```

   二进制 wtmp 文件的内容重定向到 dummy ASCII 文件

2. 将 ASCII dummy.file 转换为名为 /var/adm/wtmp 的 **wtmp** 格式的二进制文件，请输入带有 -ic 开关的 fwtmp 命令:

   ```
   /usr/sbin/acct/fwtmp  -ic < dummy.file  > /var/adm/wtmp
   ```

   dummy ASCII 文件重定向到二进制 **wtmp** 文件。

文件

/usr/sbin/acct/fwtmp 包含 **fwtmp** 命令。
/var/adm/wtmp 包含日期更改记录，新旧日期均在记录中。
/usr/include/utmp.h 包含历史记录，历史记录中有原因、日期和时间。

相关信息

acctcon1 或者 **acctcon2** 命令， **acctmerg** 命令， **acctwtmp** 命令， **runacct** 命令， **wtmpfix** 命令。

《*操作系统与设备管理*》中的『设置记帐系统』描述了建立记帐系统必须采取的步骤。

请参阅《*操作系统与设备管理*》中的『记帐命令』，以获得可自动运行或者从键盘输入的记帐命令的列表，并了解有关日报告和月报告的准备，以及记帐文件。

---

fxfer 命令

用途

在本地系统和主机间传输文件，主机通过 HCON 连接。

语法

重新启动中断了的文件传输

fxfer -R [**-n** *SessionName*]

从主机下载文件

fxfer [**-n** *SessionName*] [**-a** | **-r**] [**-d**] [**-c** | **-C**] [**-J**] [**-f** *FileName*] [**-F**] [**-H** *HostType*][**-I** *InputField*] [**-q**] [**-t** [[**-l**] [**-s**] [**-b**]] | **-T** [[**-l**] [**-s**] [**-b**]]]

[**-v**] [**-x** *HostLogin*] [**-e**] [**-X** *CodeSet*] *SourceFile DestFile*

向主机上载文件

fxfer [**-n** *SessionName*] [**-a** | **-r**] [**-u**] [**-c** | **-C**] [**-J**] [**-f** *FileName*] [**-H** *HostType*] [**-q**] [**-t** [[**-l**] [**-s**]] | **-T** [[**-l**] [**-s**]]] [**-l**] [**-s**] [**-v**] [**-x** *HostLogin*] [**-X** *CodeSet*] [**-F** | **-V** | **-U**] [**-B** *BlockSize*] [**-L** *LoglRecLength*] [**-I** *InputField*] [**-S** *NumberUnits* [**,***IncreaseUnits* | **,***IncreaseUnits*,*UnitType* | **,,***UnitType*]] [**-M** *Volume*] [**-N** *Unit*] [**-k**] *SourceFile DestFile*

显示帮助屏幕

fxfer -h

描述

fxfer 命令在本地系统和大型机主机之间传输文件，大型机主机通过主机连接程序（HCON）进行连接。文件可以从本地系统传输到主机（上载），或者从主机传到本地系统（下载）。 **fxfer** 命令将以 SourceFile 参数命名的文件传给以 DestFile 参数命名的文件。当 HCON 会话需要特定的会话概要文件或者现存的会话时，传输过程开始。

主机操作系统可以是 VM/CMS、MVS/TSO、CICS/VS（CICS/MVS® 或 CICS/VSE®）、VSE/ESA™ 或 VSE/SP，并安装了相应版本的 3270 文件传输程序（**IND$FILE** 或相当于它的程序）。主机的文件传送程序版本由会话概要文件中的 File Transfer Program 取值来确定。**fxfer 命令支持文本或二进制数据的传输。文件可以主机发出，也可以传给主机，可以包含 ASCII 或者 EBCDIC 译文，也可以不包含。**

安全机制避免了未经授权的访问、已有文件的损坏和数据遗失。如果非 HCON 用户发出 **fxfer** 命令，此命令失败。如果 **fxfer** 命令在结束前被中断，则传输状态保存在 RESTART 文件中。

如果 fxfer 命令和 **-h** 标志一起发出，则显示帮助屏幕。如果此命令和 **-R** 标志一起发出，则搜索 **$HOME** 目录下的重新启动文件。如果存在重新启动文件，则显示重新启动菜单，启用文件传输的重新开始。如果 **-h** 和 **-R** 标志没有指定，此命令试着执行指定的文件传输。

fxfer 命令信息包括:

- 标记
- 主机文件特征标志
- 示例
- 文件

此命令需要:

- 一个或多个适配器，用于连接大型机主机。
- 一个要安装在主机上的大型机操作系统，这些操作系统包括:
  - VM/SP CMS
  - VM/XA CMS
  - MVS/SP™ TSO/E
  - MVS/XA™ TSO/E
  - CICS/VS（对于 CICS/MVS 或 CICS/VSE）
  - VSE/ESA
- 大型机主机支持的文件传送程序（ **IND$FILE** 或者其等同物）被安装在大型机上。

使用 fxfer 命令的会话概要文件

fxfer 命令和 HCON 会话互相通信，并可能需要一个特定的会话概要文件。会话概要文件定义了:

- 和主机的通信路径
- 主机类型
- 缺省文件传输方向（下载或者上传）
- 恢复时间
- 文件传送等待周期

当 **fxfer** 命令执行自动登录时，概要文件也可定义:

- 主机登录标识符
- AUTOLOG 节点标识符
- AUTOLOG 跟踪开关状态
- AUTOLOG 超时值

用户一般在调用 **fxfer** 命令时指定会话概要文件。当此命令从已有会话的子 shell 运行时发生异常。在这种情况下，如果用户没有指定会话概要文件，**fxfer** 命令使用已有的会话。如果正确的会话没有运行，**fxfer** 命令就调用新的会话。

fxfer 命令搜索 HCON 会话，如下:

- 当不用 **-n** *SessionName* 标志时，
  - 如果 **fxfer** 命令从已有会话的子 shell 发出，命令使用和此子 shell 相关的会话（在 **$SNAME** 环境变量中定义）。
  - 如果不是从仿真器会话的子 shell 发出，**fxfer** 命令则发出错误消息并且终止。

- 当和 **-n** *SessionName* 标志一起发出时，文件传输在指定的会话上执行。如果不存在指定的会话，命令在会话概要文件中搜索此会话。如果找不到指定的会话概要文件，**fxfer** 命令发出错误消息并且终止。如果存在指定的概要文件，**fxfer** 命令尝试自动登录主机，通过使用会话概要文件中定义的 AUTOLOG 值，此值使用 **-x** 标志来定义，或者通过提示用户输入必要的登录信息。

中断和重新启动的文件传送

fxfer 命令可以在结束前通过运算符或者不可恢复的通信错误来中断。如果中断了，此命令在 RESTART 文件中保存传输状态。传输可以从开始处重新开始，而不会丢失数据。

如果在传输中断后开始传输其他新的文件，**fxfer** 命令则发出 RESTART 文件已被创建的信号，并且显示如下选项:

- 重新启动中断文件的传输。
- 保存 RESTART 文件并退出文件传输程序。
- 删除 RESTART 文件并退出文件传输程序。
- 删除 RESTART 文件并继续当前的传输。

fxfer 命令和 **-r** 标志配合使用也可以重新启动中断了的文件传输。

如果在以自动登录开始的文件传输过程中主机通信丢失或者断开，可以通过重新连接登录到主机上以恢复文件传输。恢复时间由会话概要文件中的 File Transfer Recovery Time 取值确定。一旦主机重新建立连接，文件传输从开始处继续。如果通信不能重新建立，文件传输程序生成一个 RESTART 文件。

如果显式文件传送和主机之间的通信丢失，用户必须重新启动仿真器会话，并且在重新启动文件传输前登录回主机。

源文件和目标文件

fxfer 命令 *SourceFile* 和 *DestFile* 参数是必需的。 *SourceFile* 参数为文件传输指定了源文件。 *DestFile* 参数为文件传输指定目标文件。本地系统文件名使用普通格式。主机文件名遵从主机命名约定，它是以下格式之一:

主机类型	文件名格式
VM/CMS	*″ 文件名 文件类型文件方式 ″* **注**: ″″（双引号）对于所有 VM/CMS 文件名来说都是不可少的，这样才能保证文件的正确传输。

主机类型	文件名格式
MVS/TSO	″[’]*DataSetName* [**(***MemberName***)**] [**/***Password*][’]″

此处:

DataSetName
指示物理顺序数据集或者分割数据集。

（*MemberName*）
指示已有分割数据集目录下的成员名。**()**（圆括号）括起 *MemberName*，是必需的。

/*Password*
如果给 MVS/TSO 数据集指定了密码保护，此参数必须有。密码之前的 **/** （斜杠）是必需的。

注:

1. ″ ″ （双引号）对于所有的 MVS/TSO 文件名是必须要的。这样才能保证文件正确传输。
2. 当为 MVS/TSO 文件名指定完整路径名时，使用 ’（单引号）在 ″ （双引号）之间。请勿在单引号和双引号之间加空格，也请勿在引号和文件名之间加空格。

CICS/VS	″*FileName*″
VSE/ESA	″*FileName FileType*″

注:

1. 必须对所有 CICS/VS、VSE/ESA 和 VSE/SP 文件名使用 ″ ″（双引号）来确保正确的文件传输。
2. CICS/VS、VSE/ESA 和 VSE/SP 文件名约定允许文件名长达 8 个字符。
3. 在 DBCS 环境下，HCON 不支持 VSE 主机。

标志

注: 双字节字符集（DBCS）支持日本英语、日本片假名、韩国语或者繁体汉语，注意事项有:

- 如果指定了 DBCS **-I** 或者 **-s** 标志，以下翻译标志之一（ **-t -t** 或者 **-J**）也必须指定，否则 DBCS 标志就被忽略了。
- **-M**、**-N** 和 **-k** 标志只在 MVS/TSO 主机中使用。
- **-e** 标志只有在下载时对 CICS® 有效。
- **-b** 标志只有在下载时有效。

-a 将 *SourceFile* 指定的文件添加到 *DestFile* 指定的文件中去，前提是存在目标文件。如果 *DestFile* 指定的文件不存在，此标记被忽略并创建目标文件。

注: 当上传文件至 CICS/VS 主机时， **-a** 标志无效。对于 VSE/ESA，**-a** 标志只有在向 CICS 临时存储器（FILE=TS）上传时有效。

-b 在使用 **-t** 、 **-T**、**-c** 或者 **-C** 标志时，在每条记录的尾部保留空格。 **-b** 标志只在 DBCS 环境下被支持。

-c 在 DBCS 环境下，如果文件传输是上传，**-c** 标志将文件的 LF（换行）代码转化为 CRLF（回车换行）代码。对于下载文件传输，**-c** 标志将文件中的 CRLF 代码转化为 LF 代码。

-C 在 DBCS 环境中，**-C** 标志在上传文件时禁止 PC-DOS 文件中 EOF（文件结束）代码的传输。对于下载文件传送，**-C** 标志在 PC-DOS 文件的尾部附加一个 EOF 代码 x’1A。

-d 下载文件，通过从主机传到本地系统。如果此标志和 **-u** 标志都没有指定，则由会话概要文件中的 File Transfer Direction 特征量确定传输方向。

注: 当从 VSE/ESA 主机的文件传输（FILE=HTF）下载已翻译文件时，将从主机系统删除该文件，除非指定 **-I** "KEEP" 标志。

-e 在文件传送结束后删除临时存储队列。下载时，仅与 CICS 主机一起使用此标志。**-e** 标志仅被 DBCS 环境支持。

-f *FileName* 将文件传送过程中的诊断输出（或文件传送状态）放入文件中，文件通过 *FileName* 变量来指定。

如果 **-f** 标志在异步传输中没有指定，则消息存放在 **$HOME/hconerrors** 文件中。如果 **-f** 标志在同步传输中没有指定，则消息发送给标准输出。

指定文件传送参数或者文件名或者文件传输故障的错误信息被定向到标准输出（如果是本地系统屏幕），也可能定向到 **$HOME/hconerrors** 文件（如果标准输出不是本地系统屏幕）。

-h 为 **fxfer** 命令显示帮助屏幕。此屏幕归纳了所有可用的命令标志和命令操作。当指定了此标志时，则忽略所有其他的标志，并且不传输文件。

注:

1. 如果使用了 **-h** 标志，则忽略所有其他的标志。不传输文件。
2. 如果没有从已有的 HCON 会话的子 shell 初始化 **fxfer** 命令，则 **-h** 标志或者 **-n** 标志是必须要的。

-H *HostType* 指定主机的类型。 *HostType* 变量可以是以下的值:

CMS VM/SP CMS 或 VM/XA CMS

TSO MVS/SP TSO 或 MVS/XA TSO

CICS CICS/VS（The CICS 主机类型包括 CICS/VSE、CICS/MVS、CICS/ESA® 和 CICS/MVS/ESA。）

VSE VSE/ESA（在 DBCS 环境中不受支持。）

如果省略 **-H** 标志，则使用在会话概要文件中由 Host Type 特征量指定的值。用户必须指定正确的主机操作系统。

注:

1. 如果指定了 **CICS** 或者 **VSE** 值，系统返回错误，则用备用值重新执行此命令。CICS 和 VSE **IND$FILE** 程序在功能上是可以互换的，它们的开头有一个 6 字节大小的差异，这导致了两个版本在操作上不兼容。目标主机可能正使用程序的备用版本。
2. 为了将文件传输到 MVS/TSO 主机，可能需要在初始化文件传送之前离开会话管理器方式。

-I *InputField* 指定主机文件传送选项，选项直接放在 **IND$FILE** 命令中。也可以给 **IND$FILE** 命令增加注释，注释放在 **)**（右括号）之后。 *InputField* 变量指定的值在放在引号中，如下:

```
-I "FILE=TS) This is a comment"
```

注: **-I** 字段在 DBCS环境下不支持。

-J 允许在 EBCDIC 和 ASCII 之间的数据转换和标准化 SI/SO 字符。转化依赖于传输的方向:

上载 将文件中的 1 字节的字符转换为 EBCDIC 代码。对于 DBCS 国家或地区，扩展代码译为合适的 DBCS 代码。SO/SI 字符插入到 DBCS 字段中，字段中包含 DBCS 字符。如果文件包含控制代码 0x1E或者 0x1F它们就分别用 SO 和 SI 字符来替换。

下载 将 EBCDIC 代码译为文件中的 1 字节的字符；对于 DBCD，DBCS 代码译为扩展代码。从 DBCS 字段删除 SO/SI 字符。

注: **-J** 字段仅在 DBCS 环境中支持。

-k 在完成文件传输时发布数据集中未使用的记录。只在 MVS/TSO 环境使用此标志。**-k**标志仅在 DBCS 环境中支持。

-I 指定在 DBCS 环境下的主机语言。此选项必须和一个翻译标志配合使用（ **-t** 、**-t** 或者 **-J**）。如果 **-t** 、**-t** 或者 **-J** 都省略了， **-I** 标志也被忽略。如果没有指定 **-I** 标志，则使用会话概要文件中定义的主机语言。如果指定了 **-I** 标志，则使用的主机语言是会话概要文件中所定义语言的备用语言。例如，如果在会话概要文件中的语言特征量是 JPK （日本片假名），则用于文件传输的主机语言就是日本英语。 **-I** 标志只在 DBCS 环境下支持。

-M *Volume* 为数据集分配指定主机磁盘的卷序列号。只在 MVS/TSO 环境能使用此标志。 **-M** 标志只在 DBCS 环境下支持。

-n *SessionName* 指定之前定义的会话名称，会话的特征量控制着文件传输。会话名时从 a 到 z 之间的一个单字符。大写字母当小写处理。

-n *SessionName* 标志是必须要的，除非当用户从已有会话的子 shell 中初始化 **fxfer** 命令。在这种情况下，如果没有使用 **-n** 标志，**fxfer** 命令不履行已有的会话。

注:

1. 指定的会话必须通过使用基于 Web 的系统管理器的 **smit hcon** fast path 命令或者 **mkhcons** 命令被提前定义了。
2. 如果没有从已有的 HCON 会话的子 shell 初始化 **fxfer** 命令，则 **-h** 标志或者 **-n** 标志是必须要的。

-N *Unit* 为数据集分配指定主机磁盘的单元类型。只在 MVS/TSO 环境使用此标志。**-N** 标志只在 DBCS 环境下支持。

-q 作为后台进程异步运行文件传输。如果文件传输还没有结束，则将当前的传输请求放入队列中。如果 **-q** 标志没有指定，则文件传输操作是同步的。如果 **-f** 标志没有指定，诊断输出和状态则存放在 **$HOME/hconerrors** 文件中。

注: 系统限制了在 Interprocess Communication（IPC）消息队列中允许的字节数目。结果就是，在任何时刻，文件传输队列中的最大数目大约是 580。

-r 在主机上指定已有文件的代替物（上载），或者在本地系统上指定已有文件的代替物（下载）。对于下载，当传输成功时完成替代。这样做可以确保已有文件不会丢失或者损坏，如果由于某种原因传输过程没有完成。

如果指定了 **-r** 标志，并且文件不存在，则在文件传输过程中创建此文件。如果 **-r** 标志*没有*指定，并且存在目标文件，则产生错误消息。

对于上载，**-r** 标志在使用主机文件传输程序低于 PTF UR20455 for MVS/TSO 或 PTF UR90118 for VM/CMS 的版本时必须指定。对于 VSE 和 CICS，**-r** 标志将被忽略。

注: 主机文件传输程序一般缺省替代文件。如果不是，添加 `-I "replace"` 至 **fxfer** 命令中以指定替代。

注: 在替换主机上的文件时，必须指定逻辑记录长度（ **-L** 标记）和记录格式（**-F** 或者 **-V** 标记），此长度和格式必须和已有文件的长度和格式相同。如果不这么做，可能导致数据毁坏。此项不适用于 VSE/ESA。

-r 重新启动前一次文件传输（被用户中断或者恢复努力失败），通过使用保存在一个 RESTART 文件中的信息: **$HOME/x_fxfer.r** 文件或者 **$HOME/i_fxfer.r** 文件。如果文件传输不是从已有会话的子 shell 调用，**-n** *SessionName*标志必须被包含，以指定要使用的会话。如果 **-R** 标志和任何其他传输标志一起指定，则忽略这些标志，并且显示 RESTART 文件传输菜单。

注: 使用了 **-R** 标志后，所有其他的标志都被忽略，除了 **-n** *SessionName*。显示 RESTART 文件传输菜单。

-s 指定 DBCS 环境下的 SO/SI 处理。**-s** 标志必须和以下一个解释标记一起使用（ **-t**、**-t** 或者 **-J**）。如果省略了 **-t** 、**-t** 或者 **-J** 、**-s** 标志将被忽略。当指定了 **-s** 标志，以下用于文件传输的功能将执行:

上载 SO/SI 字符未插入 DBCS 字段。

下载 SO/SI 字符用 DBCS 字段中的控制字符（0x1E/0x1F）来替代。

-s 只在 DBCS 环境下支持。

-t 为文件执行 ASCII-EBCDIC 转换。在下载时，**fxfer** 命令将 EBCDIC 转为 ASCII。上载时，**fxfer** 命令将 ASCII 转为 EBCDIC。通过会话概要文件中的 Language 特征量来指定语言。 **-t** 标志假设文件是文本文件。换行字符是行的定界符。

当在 DBCS 环境下使用 **-t** 标志，并且和其他 DBCS 支持的标志一起使用时， **-t** 标志的行为发生如下变化:

上载 将 JISCII（日本语）或者 ASCII（韩国语，繁体汉语）转化为 EBCDIC。在 DBCS 字段中插入 SO/SI 字符。

下载 将 EBCDIC 转化为 JISCII（日本语）或者 ASCII（韩国语，繁体汉语）。从 DBCS 字段删除 SO/SI 字符。

-t 为磁盘操作系统文件进行 ASCII-EBCDIC 转换。字符序列、作为行定界符的 CRLF 和磁盘操作系统的 EOF（文件结尾）字符插入下载文件的尾部。从 EBCDIC 转为 ASCII 转化使用的语言在会话概要文件中通过 Language 特征量来指定。**-T** 标志用于转化磁盘操作系统文件。

注: 如果 **-T** ， **-t** 或者 **-J** 标志都没有指定，文件传输不进行转化，以二进制格式进行传输。

-u 上传文件，通过将文件从本地系统传到主机上。如果这个标志和 **-d** 标志都没有指定，文件传输的方向由会话概要文件中的 File Transfer Direction 特征量决定。

-v 将当前文件传输的状态定期写到屏幕上，或者写到 **-f** 标志指定的状态文件中。状态包括从文件传输开始传输数据后传输了的字节数目和所花的时间。

-x *HostLogin* 使用 *HostLogin* 变量指定的登录标识以登录主机。用户被提示输入密码。

HostLogin 字符串包含主机登录标识、AUTOLOG 节点标识符和其他可选的 AUTOLOG 值。字符串不能包含空格并且必须包含 AUTOLOG 节点标识符。AUTOLOG 字符串格式如下:

```
UserID,AutologNodeID[,Trace,Time  .  .  .]
```

如果没有指定 **-x** 标志，关于 *HostLogin* 字符串的信息从会话概要文件处取得，如下:

- 如果主机登录标识在会话概要文件中设置了，则提示输入密码。剩下的参数从概要文件处检索。
- 如果在概要文件中没有设置主机登录标识，则提示输入主机登录字符串和密码。
- 您输入的值往往重设概要文件参数。例如，如果在概要文件中设置了 AUTOLOG 时间，但是您在提示符下输入了另外一个值，则使用输入值。

如果在主机登录字符串中省略了特定的参数，这些参数将从概要文件中检索，如果这些参数在概要文件中有定义。例如，如果在概要文件中设置了 AUTOLOG Node ID AUTOLOG Trace 和 AUTOLOG Time 参数，则在提示符下只需要输入 host login ID。

文件传输过程登录主机，并使用会话概要文件建立一个仿真器会话，此会话概要文件由 **-n** 标志指定。一旦进程成功登录，文件传输开始。

会话概要文件中的 File Transfer Wait Period 参数决定了登录会话的持续时间。使用此参数，主机登录会话被维持，以进行后续的文件传输。重新登录就不必要了。

-X *CodeSet*　指定 ASCII-EBCDIC 转化要使用的备用的代码集。如果省略 **-X** 标志，则使用系统语言环境下指定的代码集。支持以下代码集:

缺省值　使用当前系统的 ASCII 代码页。

IBM-932
在 DBCS 环境中使用 IBM 代码页 932 进行翻译。

ISO8859-1
使用 ISO 8859-1 拉丁字母 1 代码页。

ISO8859-7
使用 ISO 8859-7 希腊字母表。

ISO8859-9
使用 ISO 8859-9 土耳其字母表。

IBM-eucJP
在日语环境中使用 IBM 扩展 UNIX 代码进行翻译。

IBM-eucKR
在韩国语环境中使用 IBM 扩展 UNIX 代码进行翻译。

IBM-eucTW
在繁体中文环境中使用 IBM 扩展 UNIX 代码进行翻译。

主机文件特征标志

以下标志指定了主机文件特征，它们只可以用于上传文件（**-F** 标志除外，它可在从 VSE 主机中下载时使用）:

-B *BlockSize*　指定主机数据集的块大小。**-B** 标志只可以用于 MVS/TSO 环境下的顺序数据集。 *BlockSize* 变量不能超出单个磁道的容量。**-B** 标志在文件正被附加时忽略。如果块大小为 0 则出错。

-F　指定固定长度的记录。此为缺省值，如果 **-V** ， **-t** ， **-T** ， **-c** 和 **-C** 标志都没有指定。如果文件正被附加，则忽略 **-F** 标志。

在 CICS 或 VSE 主机上，以下翻译标志之一（ **-t** 或 **-T** ）或者 CRLF 标志之一（ **-c** 或 **-C** ）必须和 **-F** 标志一起指定，这是因为 CICS 和 VSE 主机文件传输程序不支持固定的记录长度。**-F** 标志和翻译标志结合起来会导致传输程序用空格来填充记录到逻辑记录长度的结尾。缺省值为 80。

注: 当从 VSE 主机上下载时，使用 **-F** 标志可以避免从转化文件中删除尾部的空格。

-L *LoglRecLength*　指定主机文件的逻辑记录字节长度。对于新文件，缺省值为 80。对于变长度的记录，*LoglRecLength* 是记录的最大大小。如果文件正在被附加，则忽略 **-L** 标志。*LoglRecLength* 值为 0 时则出错。

因为 MVS™ 的开销，MVS/TSO 主机上变量长度记录中存储的实际字节数比 *LoglRecLength* 变量指定的值小四个字节。

CICS 和 VSE 主机文件传输程序不支持逻辑记录长度。对于到或自 CICS 或 VSE 主机的传输，**-L** 标志必须与 **-F** 标志配合使用。**-F** 和 **-L** 标志配合使用会导致传输程序在逻辑记录长度的尾部填充空格。缺省值为 80。

注: 如果记录长度大于缺省记录长度 80，则 **-L** 标志是必需的。

-S *NumberUnits* [**,***IncreaseUnits* | **,***IncreaseUnits***,***UnitType* | **,,***UnitType*]

在 TSO 中，指定给新的顺序数据集分配的空间大小。对于较大的 MVS 文件，则使用主机上允许的最大块大小，以确保填充整个磁盘磁道。**-S** 标志仅在 MVS/TSO 主机上可用。

以下变量可以和 **-S** 标志配合使用。如果使用了，它们必须按照给定顺序来指定，且要用逗号分隔开。如果省略了另一个变量前的变量，逗号必须作为占位符被包含。在 **-S** 标志和 *NumberUnits* 变量间需要一个空格。但在字符串变量中不能有空格。

NumberUnits
指定初始要添加的空间单元数。0 或者负值不能指定给 *NumberUnits* 变量。

IncreaseUnits
当前一次分配的空间已满时，指定给数据集增加的空间单元数（可选）。

UnitType
定义了空间的单位，对于磁道可能是 `T`，对于柱面可能是 `C`，或者定义了一个数字，以指定写到数据集中纪录的平均块大小（以字节计）。如果没有指定 *UnitType* 变量，缺省值为 **-B** 标志指定的值。如果没有指定 **-B** *BlockSize* 标志，缺省值为 80。

以下是能与 **-S** 标志结合的可能的变量组合:

-S *NumberUnits*,*IncreaseUnits*,*UnitType*

-S *NumberUnits*,*IncreaseUnits*

-S *NumberUnits*

-S *NumberUnits*,,*UnitType*

-U 指定未定义长度的记录。**-U** 标志只能在 MVS/TSO 环境中使用。如果文件正在被附加则忽略 **-U** 标志。

-V 指定可变长度的记录。此为缺省值，如果没有指定 **-F** 标志，并且指定了 **-t**、**-t**、**-c** 或者 **-C** 标志。如果文件正在被附加，则忽略 **-V** 标志。

由于变量记录长度为缺省值，CICS 或 VSE 主机文件传输程序不支持 **-V** 标志。

示例

以下示例为会话 `a` 采用的会话概要文件为:

```
Session   type         DFT
Communication   device         3270c0
Language         English   (U.S.A.)
Host   type         CMS
File   transfer   direction         up
File   transfer   wait   period         10
File   transfer   recovery   time         30
```

此处:

- host type 是 VM/CMS。
- 通过 DFT 3270 连接设备进行连接。
- 文件传输的缺省方向为上传（使用会话概要文件 `a` 来下载文件，用户必须指定 **-d** 标志和 **fxfer** 命令）。
- 文件传输过程保持 10 分钟登录状态。
- 如果传输中断，传输进程在 30 分钟内尝试恢复连接，然后将信息保存到 RESTART 文件中，以便后续传输。
- 解释语言为 U.S.A. ASCII-EBCDIC。

1. 要上载 `samplefile` 文件（当前目录下）至主机，并且使用 U.S.A. 转化表将它转化为 EBCDIC 输入:

   ```
   fxfer   -n   a   -t   samplefile   "test   file   a"
   ```

   – **-n** 通知 **fxfer** 命令使用 `a` 会话传输文件。

– **-t** 指示 fxfer 命令使用 new-line 字符进行转化。

转化后的数据放在主机上的 a 测试文件中。因为主机文件名包含空格，文件名需要用引号括起来。

2. 要将 file2 文件上传到 VM/CMS 主机下的 test file b ，请输入:

```
fxfer -urv -L 132 -V -H CMS file2 "test file b"
```

– **-u** 指示 **fxfer** 命令上传此文件。
– **-H** 表明主机类型为 VM/CMS。如果目标文件存在，它将被传输文件替换（因为指定了 **-r** 标志）。
– **-v** 让 **fxfer** 显示传输的字节数和用时。终端上显示了状态或者诊断输出。
– 如果主机文件不存在，主机文件的最大逻辑记录长度设置为 132 字节（ **-L** 标志）。
– 主机文件记录的格式是变量（ **-V** 标志）。没有进行转化。

3. 要从仿真器会话 a 的子 shell 程序将本地系统 **/etc/motd** 文件上传到 CICS motdfile 主机文件，同时进行翻译并填充空格，请输入:

```
fxfer -utFH CICS -I ")This is a comment" /etc/motd "motdfile"
```

– **-u** 通知命令上传文件。
– **-t** 从 ASCII 到 EBCDIC进行转化。
– **-F** 使传输程序在上传文件填充空格到 80 列（缺省记录长度）。要修改缺省列，使用 **-L** 标志和不同的记录长度（列）。
– **-H** 指定主机为 CICS 类型。
– **-I** 指定 *InputField* 的值被添加到 **IND$FILE** 命令中去。

在这个示例中，“This is a comment”是主机注释字段。

使用 **fxfer** 命令上传或者下载文件时，到或者从当前环境外的 TSO 环境必须有其他环境的授权。需要使用单引号（'），然后双引号（"）来修饰文件（或数据集）。

4. 例如，上传文件 newfile 到 TSO 环境，其中完整的限定名称是 sys4.parmlib.samplefile ，请输入:

```
fxfer -urtvH TSO 'newfile' "sys4.parmlib.samplefile"
```

– **-u** 通知命令上传文件.
– 如果 sys4.parmlib.samplefile 文件存在，它将被替换（ **-r** 标志）成newfile文件的转化后内容（ **-t** 标志）。
– **-v** 通知 **fxfer** 命令每隔几秒将文件传输状态写到本地屏幕上。
– **-H** 通知 **fxfer** 命令主机是 MVS/TSO 主机。

注: 此示例假定 **fxfer** 命令是从已有会话的子 shell 发出（使用 **e789** 命令建立会话）。

5. 从 MVS/TSO 主机下载文件 spfuser.test 至本地系统输入:

```
fxfer -n a -d -r -H TSO spfuser.test samplefile1
```

– **-n** 通知 **fxfer** 命令使用会话 a 传输文件。如果会话 a 还没有创建，命令试图自动登录。因为没有指定主机登陆标识，**fxfer** 命令检查会话概要文件寻找登录标识。如果还没有指定登录标识，则提示用户输入登录标识和密码。
– **-d** 重设缺省的文件传输方向，缺省为上载。
– 如果 samplefile1 文件已经存在，它将被从主机上下载的文件替换掉（**-r** 标志）。
– **-H** 通知 **fxfer** 命令主机为 MVS/TSO 主机而不是 VM/CMS（在会话概要文件中的缺省值）。

传输文件放在 samplefile1 文件中，在本地系统里。文件传输是同步执行的。

6. 下载 VM/CMS 主机test file a 并将它附加在本地系统中的 mydir/samplefile 文件后，使用会话概要文件 a 并自动登录，请输入:

```
fxfer   -n   a   -dat   -q   -f   status.out
-x   laura,vm1,trace   "test   file   a"   mydir/samplefile
```

– **-n** 通知 **fxfer** 命令使用会话概要文件 a 来传输文件。

– **-x** 提供主机登录标识。 **fxfer** 命令首先检查会话是否已经建立在本地系统上了。如果是，此命令在已有会话的基础上传输文件。如果会话 a 还没有建立。**fxfer** 命令通过使用主机登录标识 laura 和自动登录脚本 vm1 来自动登录，并跟踪登录活动。提示用户输入密码。此命令传输文件。

– **-dat** 通知 **fxfer** 命令下载文件（**-d** 标志），使用 U.S.A. 转化表（在会话概要文件中定义）将数据从 EBCDIC 转化为 ASCII（**-t** 标志），并附加（ **-a** 标志）转化后的文件到 mydir/samplefile 文件中，此文件在本地系统里。如果 mydir/samplefile 文件不存在，**fxfer** 命令忽略 **-a** 标志并创建文件。

– 状态和诊断输出放在 status.out 文件中，文件位于当前本地目录（ **-f** 标志）。

– **-q** 通知 **fxfer** 命令异步传输文件。

当用户输入密码，提示符返回并且在后台执行文件传输。

要在同一个文件传输进程中增加另一个要执行的文件传输，请输入:

```
fxfer   -n   a   -daq   -f   status.out   "test   file   b"
mydir/samplefile
```

– **-n** 通知 **fxfer** 命令使用 a 会话来传输文件。因为 a 会话已经在前一个命令中创建了，**fxfer** 命令不需要再次登录主机了。

– **-d** 通知此命令从主机上下载文件。

– **-a** 通知此命令将 test file b 主机文件附加到 mydir/samplefile 文件后，此文件在本地系统中。

– **-q** 通知 **fxfer** 命令异步传输文件。

fxfer 命令继续将状态信息发送到 status.out 文件，此文件在本地系统中，（**-f** 文件）。

注:

a. 如果 **fxfer** 命令的文本超出了屏幕的边界，文本自动折到下一行。按下 Enter 键来折行文本将出错。

b. 当在队列中有异步传输时启动同步文件传输将出错。

c. 只要会话仍在运行，并且 **dfxfer** 进程仍然登录在主机上，就不会提示用户要求输入登录标识或者密码。进程保持登录状态的持续时间由会话概要文件中 File Transfer Wait Period 决定。

7. 要从仿真器子 shell 重新启动中断了的文件传输，请输入:

```
fxfer   -R
```

-r 通知 **fxfer** 命令使用保存在某一 RESTART 文件中的信息执行文件传输。此 RESTART 文件是 **$HOME/x_fxfer.r** 显式重新启动文件或者 **$HOME/i_fxfer.r** 隐式重新启动文件。如果 **-r** 标志和其他文件传输标志一同指定，其他标志都被忽略。显示此 RESTART 文件传输菜单。使用此菜单，通知 **fxfer** 命令传输中断的文件。

8. 通过命令行而不是从仿真器子 shell 来重新启动文件传输，请输入:

```
fxfer   -R   -n   a
```

-n 标志通知 **fxfer** 命令使用 a 会话执行重新启动后的传输。

文件

/usr/bin/fxfer	包含 **fxfer** 命令。
/usr/bin/dfxfer	包含 **dfxfer** 进程。
$HOME/i_fxfer.r	包含关于自动登录队列的重新启动信息。**fxfer** 命令创建的临时文件。

$HOME/x_fxfer.r	包含关于手动登录队列的重新启动信息。**fxfer** 命令创建的临时文件。
$HOME/hconerrors	包含 HCON 诊断输出和文件传输状态。HCON 命令创建的临时文件。
/usr/lib/libfxfer.a	为有规划的文件传输包含库。

相关信息

SMIT 命令。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

---

gated 守护程序

用途

为 RIP、RIPng、EGP、BGP、BGP4+、HELLO、IS-IS、ICMP、ICMPv6 和 SNMP 协议提供网关路由功能。

注: 从命令行用 SRC 命令来控制 **gated** 守护程序。每次系统启动时用 **rc.tcpip** 文件启动守护程序。

语法

/usr/sbin/gated [**-c**] [**-C**] [**-n**] [**-N**] [**-t** *TraceOptions*] [**-f** *ConfigFile*] [*TraceFile*]

描述

/usr/sbin/gated 守护程序用于处理多路由协议并替换 **routed** 和任何使用（HELLO） 路由协议的路由守护程序。**/usr/sbin/gated** 守护程序目前处理路由信息协议（RIP）、下一代路由信息协议（RIPng）、外部网关协议（EGP）、边界网关协议（BGP） 和 BGP4+、防御通信网络本地网协议 （HELLO） 以及开放最短路径优先（OSPF）、中间系统到中间系统（IS-IS）以及因特网控制消息协议（ICMP）/ 路由器发现路由协议。此外，**gated** 守护程序支持简单网络管理协议（SNMP）。**gated** 进程可以被配置以执行所有这些协议或这些协议中的任何组合。**gated**守护程序的缺省配置文件为 **/etc/gated.conf** 文件。**gated**守护程序将其进程标识存储在 **/etc/gated.pid** 文件中。

注: 当 **gated** 和 **routed** 守护程序在同一主机上一起运行时，可能会发生不可预测的结果。

如果在命令行指定了跟踪文件，或没有指定跟踪标志的话，则**gated**守护程序从终端中分离出来并在后台运行。如果指定了跟踪标志，但却没有指定跟踪文件，则 **gated** 假定跟踪目标为一个**标准错误**并仍在前台运行。

注: IS-IS 路由协议不能在 64 位的内核上运行。

信号

gated 服务器在您使用 **kill** 命令向它发出信号时执行以下操作:

SIGHUP 重新阅读配置。

SIGHUP会使 gated 重新阅读配置文件。而 **gated** 守护程序首先会对所有分配的策略结构进行清理。所有 BGP 和 EGP 同位体都被标志为删除，而配置文件则被重新分解。

如果重新分解成功，则不再存在于配置中的 BGP 和 EGP 同位体都被关闭，新的同位体被启动。**gated** 守护程序试图决定，对现有的同位体所做的更改是否需要关机并重新启动程序。

注: 当 OSPF（开放最短路径优先）被启用时，重新配置功能则处于禁用状态。

SIGTIN 当前状态的快照。

所有 gated 任务、定时器、协议和表格的当前状态都被写入**/var/tmp/gated_dump**中。

派生出一个子进程以转储表信息即可实现上述操作。这样，就不会影响 **gated** 守护程序的路由功能。

SIGTERM 适当的关机。

当收到 **SIGTERM** 信号时，**gated** 守护程序试图进行适时的关机。所有任务和协议都要被关闭。其中绝大多数都将立即被终止，而 EGP 同位体例外，因为它要等待确认。这时，有必要重复 **SIGTERM** 一次或两次，如果该过程时间太长的话。

当收到 **SIGTERM** 信号时，所有协议路由都将从内核的路由表中清除掉。界面路由、带 **RTF_STATIC** 设置的路由（来自支持的 **route** 命令）和指定 **retain** 的静态路由仍将被保留。终止带完整的外部路由的 **gated** 守护程序，请用 **SIGKILL** 或 **SIGQUIT** 信号（这些信号可以产生核心转储）。

SIGUSR1 切换跟踪。

一旦收到 **SIGUSR1** 信号，**gated** 守护程序将关闭跟踪文件。其后的 **SIGUSR1** 信号将使其重新打开。这允许文件被规则地移动。

注: **SIGUSR1** 信号在没有指定跟踪文件或正在执行跟踪到 **stderr** 的情况下不可能使用。

SIGUSR2 检查界面的变化。

一旦收到 **SIGUSR2** 信号，**gated** 守护程序会重新扫描内核界面清单，看看有没有什么变化。

gated 和 snmpd 守护程序

gated 守护程序被内部配置成一个 **snmpd** 守护程序的 SNMP 多路复用（SMUX）协议同位体或代理进程。有关更多信息，请参考《*网络与通信管理*》中的『SNMP 守护程序处理』。

使用系统资源控制器来操作 gated 守护程序。

gated 守护程序可以通过系统资源控制器（SRC）来控制。**gated** 守护程序是 SRC **tcpip** 系统组的成员。缺省情况下，该守护程序是禁用的，且可以由以下 SRC 命令来操作:

startsrc 启动一个子系统，子系统组，或是一个子服务器。
stopsrc 终止子系统、子系统组或子服务器。
refresh 使子系统或子系统组重新读取适当的配置文件。
lssrc 获取一个子系统，子系统组，或是一个子服务器的状态。

注: 从 **startsrc** 命令进行初始启动时，直到所有 **gated** 初始化都完成之后 **gated** 守护程序才开始响应其他 SRC 命令。一个非常大的 **/etc/gated.conf** 文件会需要一分钟或更多的时间来完成分析。

标志

-c 为查找语法错误指定配置文件解析，在发生那些错误后 **gated** 守护程序退出。如果没有错误发生，则 **gated** 守护程序会将一个转储文件放入 **/var/tmp/gated_dump** 文件中。**-c** 标志表示**-tgeneral**、**kernel**、**nostamp** 标志。如果指定了**-c**标志，则**gated**守护程序会忽略配置文件中的所有**跟踪选项**和**跟踪文件**子句。

-C 指定只为语法错误分析配置文件。如果发现错误，**gated** 守护程序将退出且状态为 **1**，如果没有发现任何错误，则其退出状态为 **0**。**-C**标志表示 **-tnostamp** 标志。

-f *ConfigFile* 指定备用配置文件。缺省情况，**gated** 守护程序使用 **/etc/gated.conf** 文件。

-n 指定 **gated** 守护程序不会对内核路由表进行修改。这用于对带有实际路由数据的 **gated** 配置文件进行测试。

-N 指定 **gated** 守护程序不守护。通常来说，如果没有指定对 **stderr** 的追踪，同时父进程标识不是 1 的话，则**gated**守护程序起守护作用。该标志允许使用一种类似于 **/etc/inittab** 的方法调用一个进程标识不是 1 的**gated**守护程序。

-t*TraceOptions* 指定哪些跟踪选项是在系统启动时被起用。当在没有*TraceOptions*变量时使用，该标志就启动一般跟踪选项。用逗号将每个跟踪选项隔开。请勿在标志和第一个跟踪选项之间插入空格。

必须使用 **-t** 标志，这种标志用于跟踪那些在对 **/etc/gated.conf** 文件分析之前发生的事件，如决定界面配置并从内核中读取路由。

而 **gated.conf** 文件则描述了可用的跟踪选项。

示例

1. 启动 **gated** 守护程序，请输入类似以下的命令:

   ```
   startsrc -s gated -a "-tall /var/tmp/gated.log"
   ```

   该命令启动 **gated** 守护程序并将消息记入日志。信息被发送至 **/var/tmp/gated.log** 文件。
2. 正常终止 **gated** 守护程序，请输入:

   ```
   stopsrc -s gated
   ```

   该命令会终止守护程序。**-s**标志指出跟随的子系统将被停止。
3. 从 **gated** 守护程序中获取短状态，请输入:

   ```
   lssrc -s gated
   ```

   这个命令返回了守护程序的名字，进程标识和状态（活动或非活动）。

文件

/etc/gated.pid	包含 **gated** 进程标识。
/var/tmp/gated_dump	指定存储器转储文件。
/var/tmp/gated.log	指定记录错误信息的日志文件。

相关信息

kill、**gdc** 命令、**ospf_monitor** 命令及 **ripquery** 命令。

routed 守护程序。

gated.conf 文件格式。

《*网络与通信管理*》中的『如何配置 gated 守护程序』。

《*网络与通信管理*》中的『TCP/IP 路由』、『TCP/IP 协议』和『TCP/IP 守护程序』。

---

gdc 命令

用途

为 **gated** 提供一个可操作的用户界面。

语法

gdc [**-q**] [**-n**] [**-c** *coresize*] [**-f** *filesize*] [**-m** *datasize*] [**-s** *stacksize*] [**-t** *seconds*] *Subcommands*

描述

gdc 命令为**gated** 路由守护程序的操作提供了一个面向用户的界面。它为以下内容提供支持:

- 启动和停止守护程序
- 当守护程序运行时，发送信号对其进行操作
- 维护并对配置文件进行语法检查
- 用于状态转储和核心转储的生成和清除。

gdc 命令可以非常可靠地决定 **gated** 的运行状态，并能在错误发生时产生可靠的退出状态。该点使它在用于操作 **gated** 的 shell 脚本中时有利。用 **gdc** 执行的命令，以及可选地，由这些命令的执行而产生的错误消息将通过同一 **syslogd** 设备记录日志，这些设备都是 **gated** 自身使用的，用于对在守护程序上执行的操作进行审计跟踪。

标志

-n 在不更改内核转发表的情况下运行。这对测试非常有用，而且当将其作为一个不转发的路由服务器运行时也很有用。

-q 安静地运行。使用这种标志通常被打印到标准输出中的信息消息被禁止，同时，使用 **syslogd** 记录错误消息，而不是将其打印到标准错误输出中。当从 shell 脚中运行 **gdc** 时，这非常方便。

-t *seconds* 指定以秒计 **gdc** 等待 **gated** 完成某些操作的时间，特别是在终止和启动时。将该数值的缺省值设为 10 秒。

-c *coresize* 为一个用 **gdc** 启动的 **gated** 产生的核心转储设置最大大小。这在一些系统中是非常有用的，这些系统的缺省最大核心转储大小太小，无法使 **gated** 按错误产生完全核心转储。

-f *filesize* 设定用 **gdc** 启动的 **gated** 会产生的最大文件大小。这在一些系统中非常有用。在这些系统中，缺省的最大文件转储大小太小，使得 **gated** 无法在请求时生成一个完全的状态转储。

-m *datasize* 设置一个 **gated** 数据段的最大大小，该 gated 由 **gdc** 启动。这对一些系统非常有用。在这些系统中，缺省的数据段大小太小，使得 **gated** 无法运行。

-s *stacksize* 设置一个 **gated** 堆栈的最大大小，该 gated 由 **gdc** 启动。这对一些系统非常有用。在这些系统中，缺省的最大堆栈大小太小，使得 **gated** 无法运行。

子命令

以下子命令可以使信号被送至 **gated**，用于各种目的:

COREDUMP 向 **gated** 发送一个异常终止信号，使它终止并产生核心转储。

dump 向 **gated** 发信号将其当前状态转储到文件 **/var/tmp/gated_dump** 中。

interface 向 **gated** 发信号以重新检查界面配置。**gated** 通常在任何事件中周期性地重复该工作，但此功能可被用于促使守护程序在发现有变更发生时，立即检查其界面状态。

KILL 使 **gated** 不正常地终止。

reconfig 向 **gated** 发信号使其重新阅读其配置文件，从而按适当的情况重新配置其当前状态。

term 向 **gated** 发信号使其在正常地关闭所有正在运行的路由协议后终止。即使一些协议还没有完全关闭，执行第二次该命令会使 **gated** 终止。

toggletrace 使跟踪处于暂挂状态，而且如果 **gated** 目前正在跟踪一个文件的话，关闭跟踪文件。如果 **gated** 跟踪目前处于暂挂状态，则该子命令会使跟踪文件重新打开，启动跟踪。这对移动跟踪文件非常有用。

以下子命令可以进行与配置文件相关的操作。

checkconf 检查 **/etc/gated.conf** 是否有语法错误。在更改配置文件之后和发送一个 **reconfig** 信号到当前正在运行的 **gated** 之前这样做会非常有用，能确保配置中没有错误，这些错误可以使运行的 **gated** 在重新配置时被终止。当使用该命令时，**gdc** 发送一个信息消息，来提示是否有分析错误。如果存在，它会将错误输出保存在文件中以备检查。

checknew 该任务类似于 **checkconf**，其不同之处在于它检查的是**新**配置文件 **/etc/gated.conf+**。

newconf 将 **/etc/gated.conf+** 文件作为 **/etc/gated.conf** 移动到位，如上面描述的那样，保留文件的旧版本。给出此命令时，**gdc** 将拒绝做任何事情，如果**新**配置文件不存在或看起来可疑时。

backout 将配置文件向**更新的**方向旋转，实际上就是将**旧的**配置文件改名为 **/etc/gated.conf**。如果 **/etc/gated.conf-** 不存在或长度为 0，或者此操作将删除长度不为 0 的现有文件 **/etc/gated.conf+**，此命令将拒绝执行该操作。

BACKOUT 执行**逆序恢复**操作，即便是 **/etc/gated.conf+** 存在且长度不为 0。

modeconf 将所有配置文件方式设置为 664、所有者为 root 用户和组为 system。

createconf 如果 **/etc/gated.conf+** 不存在，创建一个长度为 0 的文件。该文件的方式设为 664、所有者 root 用户和组系统。

以下子命令为启动和停止 **gated** 以及决定其运行状态提供支持:

running 决定 **gated** 目前是否正在运行。这可以通过检查 **gated** 是否锁定了包含其 pid 的文件，该文件中的 pid 是否是敏感的以及是否有带该 pid 的进程正在运行来完成。如果 **gated** 正在运行，退出且退出状态为 0，否则退出状态为非 0。

start 启动 **gated**。如果 **gated** 已经处于运行状态，则命令则返回一个错误。否则它将执行 **gated** 二进制文件并一直等到延迟间隔时间（缺省设置为 10 秒，否则在 **-t** 选项中设置），直到新启动的进程在进程标识文件中获取一个锁为止。如果在执行二进制文件时检测到错误，或在指定的等待时间内，在进程标识文件中无法获取锁的话，则返回非零退出状态。

stop 如果可能的话，适当地停止 **gated**；如果不行的话，则非正常地停止。如果 **gated** 目前没有处于运行状态，则命令返回一个错误（非零退出状态）。否则，它将向 **gated** 发送一个终止信号，并等候最多延迟间隔时间（缺省值为 10 秒，否则在**-t** 选项中指定）以使进程退出。如果 **gated** 无法在延迟间隔时间内退出，则它会再次发送一个终止信号。如果在第二个延迟间隔时期结束后仍无法退出，则会用 kill 信号第三次向其发送信号。这将迫使其立即终止，除非其中什么坏了。当命令检测到 **gated** 已经终止时，命令以 0 退出状态终止，否则以非零退出状态终止。

restart 如果 **gated** 正在运行，则可以通过用于上面的 **stop** 命令相同的过程将其终止。当前一个 **gated** 终止时，或如果它在命令执行前仍未运行，那么一个新的 **gated** 进程将通过使用上面 **start** 命令中所描述的过程执行。如果这个过程中的任何一步失败的话，就会返回一个非零退出状态。

对于通过执行上面的一些命令而创建的文件，可以用以下子命令将其除去:

rmcore 除去任何已有的 **gated** 核心转储文件。

rmdump 除去任何已有的 **gated** 状态转储文件。

rmparse 当执行一个 **checkconf** 或 **checknew** 命令，并且在被检查的配置文件中发现语法错误时，除去生成的分析错误文件。

以下子命令可以显示 **gated** 的版本信息:

version 显示 **gated** 的版本信息。当该命令被执行时，**gated** 不能已经处于运行状态。没有**gdc** 选项用于此命令。

缺省情况 **gated** 从一个通常命名为 **/etc/gated.conf** 的文件中获取其配置。**gdc** 程序也保留配置文件的几个其他版本，尤其是以以下名称命名的版本:

/etc/gated.conf+	新配置文件。当要求 **gdc** 安装一个新配置文件时，该文件被重命名为 **/etc/gated.conf**。
/etc/gated.conf-	旧的配置文件。当 **gdc** 被要求安装一个新的配置文件时，原来的 **/etc/gated.conf** 被重命名为这个名字。
/etc/gated.conf—	真正的旧配置文件。**gdc** 在该名字下保留原有的旧配置文件。

文件

/usr/sbin/gated	**gated** 二进制文件。
/etc/gated.conf	当前的 **gated** 配置文件。
/etc/gated.conf+	更新的配置文件
/etc/gated.conf-	旧一些的配置文件
/etc/gated.conf—	更旧的配置文件
/etc/gated.pid	**gated** 存储其进程标识的地方。
/var/tmp/gated_dump	**gated** 的状态转储文件。
/var/tmp/gated.log	储存配置文件分析错误的位置。

相关信息

gated 守护程序和 **syslogd** 守护程序。

---

gencat 命令

用途

生成并修改一个消息编目。

语法

gencat *CatalogFileSourceFile ...*

描述

gencat 命令生成一个消息编目文件（通常为 ***.cat**），该文件是从消息文本开始文件（通常为 ***.msg**）中生成。**gencat** 命令将*SourceFile* 参数指定的消息文本开始文件合并到一个由*CatalogFile* 参数指定的格式化消息编目中。输入消息到一个源文件后，用 **gencat** 命令对源文件进行处理，生成一个消息编目。如果还没有编目文件的话，**gencat** 命令就生成一个编目文件。如果编目文件已经存在，**gencat** 命令就会在编目文件中包括新消息。

您可以指定任何数量的消息文本开始文件。**gencat** 命令会按指定的顺序，一个接一个地处理多个源文件。每一个后继的源文件都会修改编目。如果设定与消息数量有冲突的话，则*源文件*参数中定义的新消息文本会替换当前*编目文件*参数中所包含的旧的消息文本。消息数量必须在 1 到 **NL_MSGMAX** 的范围内。设置数量必须在 1 到 **NL_SETMAX** 的范围内。

gencat 命令不接受符号消息标识。如果想使用符号消息标识的话，您必须运行 **mkcatdefs** 命令。

> **注**: 如果 **-** 字符被指定为*编目文件*参数，使用标准输出。当**-** 字符被指定为*源文件*参数时，使用标准输入。

退出状态

此命令返回以下出口值:

0 成功结束。
>0 发生错误。

示例

从源文件 test.msg 中生成一个 test.cat 编目，请输入:

```
gencat test.cat test.msg
```

test.msg 文件不包含符号标识。

文件

/usr/bin/gencat 包含 **gencat** 命令。

相关信息

dspcat 命令、**dspmsg** 命令、**mkcatdefs** 命令和 **runcat** 命令。

catclose 子例程、**catgets** 子例程和 **catopen** 子例程。

更多关于消息设备的信息，请参阅《*AIX 5L V5.3 本地语言支持指南和参考大全*》中的『消息设备总览』。

---

gencopy 命令

用途

允许复制各种打包格式（installp、RPM、ISMP）的软件产品。

语法

从介质将软件复制到目标位置

gencopy -d *Media* [**-t** *TargetLocation*] [**-D**] [**-b** *bffcreateFlags*] [**-U**] [**-X**] **-f** *File* | *CopyList...* | all

将介质上的软件产品和软件包列出

gencopy -L -d *Media* [**-D**]

描述

gencopy 命令是 **bffcreate** 命令的包装器。它决定必须复制哪些映像并调用适当的命令。对一些 RPM、ISMP 或其他类型的映像，在这些映像中，所要求的文件列表尚属未知，将子目录中的所有文件复制到目标位置。

标志

-b *bffcreateFlags* 指定以下有效的标志: **l**、**q**、**v**、**w** 和 **S**。
-d *Media* 指定安装映像所在的设备或目录。介质可以是设备（**/dev/cd0**、**/dev/rmt0**）或目录。
-D 指定调试方式。此标志用于调试该脚本。它将生成大量输出，不应将其用于正常操作。

-f *File* 指定一个带有要复制到目标位置的映像列表的文件。**installp**、RPM 和 ISMP 映像应该分别带有前缀 **I:**、**R:** 和 **J:**。为临时修订包加上前缀 **E:**。

-L 列出介质上的安装软件包。该列表用冒号隔开，它包含以下信息:

file_name:package_name: 文件集 :V.R.M.F: 类型 : 平台 : 描述

bos.sysmgt:bos.sysmgt:bos.sysmgt.nim.client:4.3.4.0:I:R: 网络安装管理器 – 客户工具

bos.sysmgt:bos.sysmgt:bos.sysmgt.smit:4.3.4.0:I:R: 系统管理界面工具（SMIT）

-t *TargetLocation* 指定存储安装映像文件的目录。如果没有指定 **-t** 标志，则文件将被保存在 **/usr/sys/inst.images** 目录中。

-U 如果需要，将目标资源库的目录结构升级到当前标准。当前标准要求根据程序包类型和体系结构将映像组织到子目录中。例如，installp 映像位于 *SaveDir***/installp/ppc** 目录中。当从包含该结构的源中复制时，要求目标符合。指定 **-U** 标志允许 **gencopy** 命令在资源库中创建相应的子目录结构并将现有映像移动到相应的位置。该标志应该只需要使用一次，除非随后执行无效手动复制。

-X 如果需要空间的话，文件系统会自动扩展。

示例

将所有映像从光盘 （**/dev/cd0**） 复制到 **LPP_SOURCE**（**/export/lpp_source/500**）应用，请输入:

```
gencopy -d  /dev/cd0 -t  /export/lpp_source/500 all
```

文件

/usr/sbin/gencopy
/usr/sys/inst.data/sys_bundles
/usr/sys/inst.data/user_bundles

相关信息

bffcreate 命令。

---

gencore 命令

用途

为正在运行的进程生成核心文件。

语法

gencore *ProcessID FileName*

描述

gencore 命令生成进程核心文件，该进程由*进程标识* 指定，同时不用终止进程。所生成的核心文件中有进程存储映像，这些映像可以与 **dbx** 命令共用以调试。所生成的核心文件将按照*文件名*参数所指定的名称命名。

参数

文件名 为 **gencore** 命令所生成的核心文件命名。

进程标识 指定进程的进程标识，从这个进程中，**gencore** 将会生成核心文件。

退出状态

0 核心文件已经成功生成。

>0 发生错误。可能只生成了部分核心文件。

示例

1. 为进程标识为 1095 的进程生成一个名为“core.1095”的核心文件，请输入:

   ```
   gencore 1095 core.1095
   ```

   核心文件就会在不终止进程的情况下生成。

文件

/usr/bin/gencore	包含 **gencore** 命令。

相关信息

dbx 命令和 **kill** 命令。

---

genfilt 命令

用途

添加过滤规则。

语法

genfilt -v 4|6 [**-n** *fid*] [**-a D|P|I|L|E|H|S**] **-s** *s_addr* **-m** *s_mask* **[-d** *d_addr*] [**-M** *d_mask*] [**-g Y|N**] [**-c** *protocol*] [**-o** *s_opr*] [**-p** *s_port*] [**-O** *d_opr*] [**-P** *d_port*] [**-r R|L|B**] [**-w I|O|B**] [**-I Y|N**] [**-f Y|N|O|H**] [**-t** *tid*] [**-i** *interface*] [**-D** *description*] [**-e** *expiration_time*] [**-x** *quoted_pattern*] [**-X** *pattern_filename*] [**-C** *antivirus_filename*]

描述

用 **genfilt** 命令向过滤规则表添加过滤规则。由此命令生成的过滤规则叫做手动过滤规则。可以使用 **genfilt** 命令、IPsec 系统管理界面程序（IP V4 或 IP V6）或“虚拟专用网”子菜单中的基于 Web 的系统管理器配置 IPsec 过滤规则。

标志

-a *Action*

允许以下 *Action* 值:

- **D** （拒绝）块流量。
- **P** （许可）允许流量。
- **I** 使之成为 IF 过滤规则。
- **L** 使之成为 ELSE 过滤规则。
- **E** 使之成为 ENDIF 过滤规则。
- **H** 使之成为 SHUN_HOST 过滤规则。
- **S** 使之成为 SHUN_PORT 过滤规则。

所有 IF 规则必须以关联 ENDIF 规则结束。可嵌套这些条件规则，但必须保持正确的嵌套和范围，否则 **mkfilt** 命令将不能正确装入规则。

-C *antivirus_filename*

指定抗病毒文件名。**-C** 标志意味着 ClamAV 病毒库的一些版本（http://www.clamav.net）。

-c *protocol*

有效值为: **udp**、**icmp**、**icmpv6**、**tcp**、**tcp/ack**、**ospf**、**ipip**、**esp**、**ah** 和 **all**。**值 all 表示过滤规则应用于所有协议。协议也可以通过数字来指定（1 到 252）。缺省值为 all。**

-D *description*

过滤器规则的简短描述文本。这是静态过滤规则的一个可选标志，但不适用于动态过滤规则。

-d *d_addr*

指定目标地址。可以是 IP 地址或主机名。如果指定了主机名，使用名称服务器返回的该主机的第一个 IP 地址。这个值连同目的地子网掩码将与 IP 包的目标地址进行比较。

-e *expiration_time*

指定到期时间。到期时间是规则应保持活动的时间量（以秒计）。*expiration_time* 不从数据库中除去过滤规则。*expiration_time* 与处理网络流量时过滤规则处于活动状态的时间量相关。如果未指定 *expiration_time*，则过滤规则的生存时间是无限的。如果指定 *expiration_time* 与 SHUN_PORT（**-a S**）或 SHUN_HOST（**-a H**）过滤规则一起使用，则它就是一旦满足过滤规则参数就拒绝或避开远程端口或远程主机的时间量。如果无论避免规则如何而指定该 *expiration_time*，则它就是一旦将过滤规则装入内核并启动处理网络流量，过滤规则保持活动的时间量。

-f

指定分段控制。分段控制标志指定规则应用于所有包（**Y**）、只应用于分段头和未分段包（**H**）、只应用于分段和分段头（**O**）或只应用于未分段包（ **N**）。缺省值为 **Y**。

-g

是否应用到源路由中？必须指定为 **Y**（是）或 **N**（否）。如果指定 **Y**，过滤规则可以用于使用源路由的 IP 包。缺省值为是（**Y**）。这个域仅用于许可规则。

-i *interface*

指定应用过滤规则的 IP 接口名。名称的示例为: **all**、**tr0**、**en0**、**lo0** 和 **pp0**。缺省值为 **all**。

-l

指定记录控制。必须指定为 **Y**（是）或 **N**（否）。如果指定为 **Y**，则与过滤规则匹配的包将被包含在过滤日志中。缺省值为 **N**（否）。

-M

指定目标子网掩码。这用于 IP 包目标地址和过滤规则目标地址的对比中。

-m

指定源子网掩码。这用于 IP 包源地址和过滤规则源地址的对比中。

-n

指定过滤规则标识。新规则将添加在您指定的过滤规则 “前面”。IP V4 的标识必须大于 1，因为第一个过滤规则是由系统生成的，而且不可移动。如果不用这个标志，新规则就添加到过滤规则表的末尾。

-O

指定目标端口或 ICMP 代码操作。这就是用于将包的目标端口 /ICMP 码和目标端口或 ICMP 码（**-P** 标志）进行比较的操作。有效值为: **lt**、**le**、**gt**、**ge**、**eq**、**neq** 和 **any**。缺省值为 **any**。当 **-c** 标志为 **ospf** 时，该值必须为 **any**。

-o

指定源端口或 ICMP 类型操作。这就是要用于将包的源端口 / ICMP 类型和过滤规则中所指定的源端口或 ICMP 类型（**-p** 标志）进行比较的操作。有效值为: **lt**、**le**、**gt**、**ge**、**eq**、**neq** 和 **any**。缺省值为 **any**。当 **-c** 标志为 **ospf** 时，该值必须为 **any**。

-p

指定源端口或 ICMP 类型。这是将要用于和 IP 包的源端口（或 ICMP 类型）进行比较的值/类型。

-P

指定目标端口 / ICMP 代码。这是将要用于和 IP 包的目标端口（或 ICMP 码）进行比较的值/码。

-r

路由。它指定规则是应用于转发包（**R**）、发往或来自本地主机的包（**L**）或同时用于两者（**B**）。缺省值为 **B**。

-s *s_addr* 指定源地址。可以是 IP 地址或主机名。如果指定了主机名，使用名称服务器返回的该主机的第一个 IP 地址。这个值连同源子网掩码将与 IP 包的源地址进行比较。

-t 指定与该过滤规则相关的通道的标识。所有与过滤规则匹配的包都要通过指定的通道。如果没有指定此标志，规则只用于非通道流量。

-v 指定过滤规则的 IP 版本。有效值为 **4** 和 **6**。

-w *Direction* 指定规则是应用于输入包（I）、输出包（O）或同时用于两者（B）。缺省值为 B。使用带 **-x**、**-X** 或 **-C** 模式选项的（O）输出方向无效。指定（B）带模式选项的两种方向是有效的，但对于包将只检查输入包。

-X *pattern_filename* 指定模式文件名。如果多个模式与该过滤规则相关联，则必须使用模式文件名。模式文件名的格式必须是每行一个模式。模式是不加引号的字符串。当激活过滤规则时，该文件被读取一次。关于更多信息，请参阅 **mkfilt** 命令。

-x *pattern* 指定加引号的字符串或模式。指定的这个字符串将被解释为 ASCII 字符串，除非在它前面有一个 0x，这种情况下，它将被解释为十六进制字符串。**-x** *pattern* 与网络流量相比较。

---

geninstall 命令

用途

通用安装程序，用于安装多种打包格式的软件产品。例如，installp、RPM、SI 和 ISMP。

语法

geninstall -d *Media* [**-l** *installpFlags*] [**-E** | **-T**] [**-t** *ResponseFileLocation*] [**-e** *LogFile*] [**-p**] [**-F**] [**-Y**] [**-Z**] [**-D**] { **-f** *File* | *Install_List*] | all}

或

geninstall -u [**-e** *LogFile*] [**-E** | **-t**] [**-t** *ResponseFileLocation*] [**-D**] {**-f** *File* | *Uninstall_List...*}

或

geninstall -L -d *Media* [**-e** *LogFile*] [**-D**]

描述

接受当前所有的 **installp** 标志，并将它们传到 **installp** 上去。一些标志（如，**-L**）会过载来表示列出介质上所有的产品。对 ISMP 打包产品没有意义的标志被忽略。这允许程序（如 NIM）继续一直向 **geninstall** 发送 **installp** 标志， 但仅使用有意义的标志。

geninstall 命令提供了查看对 **/etc/check_config.files** 中列出的配置文件所作修改的简便方法。当在 **geninstall** 安装或更新操作期间更改了这些文件时，新旧文件之间的不同之处记录将在 **/var/adm/ras/config.diff** 中。如果 **/etc/check_config.files** 请求保存旧文件，则可以在 **/var/adm/config** 目录中找到旧文件。

可编辑 **/etc/check_config.files** 文件并可以使用它来指定已更改的旧配置文件是应保存（由 s 表示）还是删除（由 d 表示），格式如下:

```
d /etc/inittab
```

一个 **geninstall** 命令的安装活动摘要保存在 **/var/adm/sw/geninstall.summary** 中。该文件含有由 **installp** 安装的文件集列表（这些表用冒号分隔），以及用 ISMP 安装的组件。这主要用来提供无提示安装的摘要信息。

注: 请参阅 **/usr/lpp/bos** 目录中的 **README.ISMP** 文件以了解更多关于打包 ISMP 安装及使用响应文件的信息。

标志

-d *Device or Directory*
指定包含有要安装映像的设备或目录。

-D
指定调试方式。此标志用于调试该脚本。它将生成大量输出，不应将其用于正常操作。

-e *LogFile*
启用事件记录日志。**-e** 标志使用户能够将 **geninstall** 命令输出的某些部分附加到由 *LogFile* 变量指定的文件中去。*LogFile* 必须指定一个已存在的可写文件，并且该文件驻留的文件系统必须有足够空间来存储日志。日志文件不折行。

-E
在缺省位置（包含产品安装文件的目录）创建的 ISMP 响应文件记录。该选项要求交互式、完全地运行 ISMP 安装或卸载程序。生成的响应文件将用于为将来对同一产品的安装或卸载提供相同选项。创建响应文件记录也导致安装或卸载产品。

-f *File*
指定一个带有要复制到目标位置的映像列表的文件。**installp**、RPM 和 ISMP 映像应该分别带有前缀 **I:**、**R:** 和 **J:**。为临时修订包加上前缀 **E:**。

-F
允许用户对已经安装的包重新安装，也可安装比当前已安装版本更旧的包。

-I *installpFlags*
指定 **installp** 所使用的标志，当调用 **installp** 命令时。**installp** 进行安装期间所使用的标志是 **a**、**b**、**c**、**D**、**e**、**E**、**F**、**g**、**I**、**J**、**M**、**N**、**O**、**p**、**Q**、**q**、**S**、**t**、**v**、**V**、**w** 和 **X** 等标志。**installp** 进行安装时不使用的标志是 **C**、**i**、**r**、**z**、**A** 和 **l** 等标志。**installp** 命令应该直接调用以执行这些功能。**-u**、**-d**、**-L** 和 **-f** 标志应无 **-I** 标志时使用。

-L
对介质上的内容进行列表。输出格式与 **installp -Lc** 格式相同， 对于用 ISMP 和 RPM 格式化的产品，末尾有附加字段。

-p
通过运行指定操作的所有预安装检测程序来执行一个预览操作。

-t *ResponseFileLocation*
允许指定响应文件或响应文件模板的备用位置。缺省位置是包含产品安装文件的目录。该标志可以用于在不同位置创建响应文件记录或模板。*ResponseFileLocation* 可以是文件或目录名。如果 *ResponseFileLocation* 是目录，它必须已经存在。如果 *ResponseFileLocation* 不是现有目录，则将假定为指定了文件名。

-t
在缺省位置（包含产品安装文件的目录）创建的 ISMP 响应文件模板。生成的模板用在将来采用您所选的选项来安装或卸载同一产品时创建响应文件。创建响应文件模板将不会引起安装或卸载产品。

-u
执行对指定软件的卸载。对 ISMP 产品，调用在供应商数据库中列出、前缀为 "J:"的卸载程序。

-Y
同意软件安装需要的软件许可证协议。该标志可接受为 **-I** 选项的 **installp** 标志。

-Z
通知 **geninstall** 在无提示模式下调用安装。

示例

安装在驱动器 cd0 中的 CD 介质的所有产品，请输入:

```
geninstall -d /dev/cd0 all
```

如果 ISMP 映像在介质上存在， 则显示一个图形化界面。**installp**、SI 或 RPM 映像的安装不会进行提示，除非 **installp** 映像分布在多张 CD 上。

文件

/usr/sbin/geninstall
/usr/sys/inst.data/sys_bundles
/usr/sys/inst.data/user_bundles

相关信息

installp 命令，**install_wizard** 命令。

genkex 命令

用途

genkex 命令会抽取目前已载入系统的内核扩展列表，并在列表中显示每一个内核扩展的地址、大小和路径名称。

语法

genkex [-**dh**]

描述

对于那些载入系统的内核扩展，内核将保留一份链接起来的列表。在该列表中，会有一些被称为“载入程序入口”的数据结构。载入程序入口中包含有扩展的名称、起始地址和大小。该信息会由 **genkex** 命令收集并写成报告。

标记

-d 除了文本部分的地址和大小，还显示数据部分的地址和大小。
-h 显示用法语句。

示例

生成一份已载入内核扩展清单，请输入:

```
genkex
```

相关信息

genkld 命令和 **genld** 命令。

《*性能管理*》中的『监控与调整命令和子例程』。

genkld 命令

用途

genkld 命令可以抽取目前已载入系统的共享对象列表，同时在列表上显示每一个对象的地址、大小及路径名称。

语法

genkld [-**dh**]

描述

对于那些已载入系统的共享对象，内核程序会保留一个链接起来的列表。在该列表中，会有一些被称为“载入程序入口”的数据结构。一个载入程序入口包括对象名称、其起始地址以及大小。此类信息将由 **genkld** 命令收集并报告。

标记

-d 除了文本部分的地址和大小，还显示数据部分的地址和大小。
-h 显示用法语句。

示例

获取一份已载入的共享对象清单，请输入:

```
genkld
```

相关信息

genkex 命令和 **genld** 文件。

《*性能管理*》中的『监控与调整命令和子例程』。

---

genld 命令

用途

genld 命令收集所有当前运行于系统中的进程列表，同时可选地针对每一个进程报告已载入的对象列表。

语法

genld [**-h** | **-l** [**-d**]] [**-a** *Area*]

描述

对于每一个目前正在运行的进程来说，**genld** 命令会打印包含进程 ID 和名称的报告，然后是为该进程装载的对象列表（可选）。对象的地址和路径名称都会被显示出来。库的成员显示在方括号中。例如，`/usr/lib/libc.a[shr.o]` 指 `shr.o` 是 **libc.a** 库的装入成员。

注: 仅允许 root 用户和安全组成员带 **-l** 标志运行此命令。

标志

-a *Area* 仅列出使用由 *Area* 参数指定的共享库区域的进程。
-d 除了文本部分的地址和大小，还显示数据部分的地址和大小。没有 **-l** 标志的情况下，该选项不起作用。
-h 显示用法语句。
-l 为每一个运行在系统中的进程报告已载入对象列表。

示例

获取每一个运行进程的已载入对象列表，请输入:

```
genld -l
```

相关信息

genkex 命令和 **genkld** 命令。

《*性能管理*》中的『监控与调整命令和子例程』。

gennames 命令

用途
收集所有以离线方式运行 **filemon** 和 **netpmon** 命令所必需的信息。

语法
gennames[**-f**]

描述
gennames 命令收集以离线方式运行 **filemon** 和 **netpmon** 命令所需的名称到地址的映射信息。收集的信息包括:

- 所有已载入的内核扩展列表，类似于 **genkex** 命令所生成的报告。
- 所有已载入的共享库列表，类似于 **genkld** 命令所生成的报告。
- 所有已载入进程的列表，类似于 **genld** 命令所生成的报告。
- 为 **/unix** 和所有内核扩展及库收集 **stripnm -z** 命令的输出。

标志
-f 为物理和逻辑卷收集设备信息。它还会打印出脱机 **filemon** 所用的虚拟文件系统信息。

示例
收集以离线方式运行 **filemon** 命令所需的信息，请输入:

```
gennames -f > gen.out
```

相关信息
filemon 命令、**gensyms** 命令、**genkex** 命令、**genld** 命令、**netpmon** 命令和 **stripnm** 命令。

《*性能管理*》中的『监控与调整命令和子例程』。

gensyms 命令

用途
收集所有以离线方式运行 **curt**、**splat** 和 **tprof** 命令所必需的信息。

语法
gensyms [**-ofhs**] [**-k** *kernel*] [**-i** *file*] [**-b** *binary*[,*binary*[,...]]] [**-S** *path*]

描述
gensyms 命令收集以离线方式运行 **curt**、**splat** 和 **tprof** 命令所需的名称到地址的映射信息。收集的信息包括:

- 所有已载入的内核扩展列表
- 所有已载入的共享库列表

- 所有已载入的进程列表
- 为 **/unix** 收集所有内核扩展、库、与进程对应的所有对象文件、 **stripnm** 命令输出。

标记

-b *binary*	指定要为其查找符号的二进制的可选列表。
-f	禁止显示源文件名。
-h	显示帮助消息。
-i *file*	读取来自指定文件的符号。
-k *kernel*	指定 kernel 映像名（缺省为 **/unix**）。
-o	显示偏移量，而不是地址
-s	仅为 **-k** 和 **-b** 标志提供的文件查找标志。
-S *path*	指定搜索路径列表；该列表用于查找二进制。

示例

收集以离线方式运行 tprof 命令所需的信息以及用户程序测试概要，请输入:

```
gensyms > test.syms
```

相关信息

curt 命令、**gennames** 命令、**splat** 命令、**stripnm** 命令和 **tprof** 命令。

《*性能管理*》中的『监控与调整命令和子例程』。

---

gentun 命令

用途

在通道数据库中生成通道定义。

语法

gentun -s *src_host_IP_address* **-d** *dst_host_IP_address* **-v 4|6** [**-t** *tun_type*] [**-m** *pkt_mode*] [**-t** *IBM*] [**-t** *manual*] [**-m** *tunnel*] [**-m** *transport*] [**-f** *fw_address*] [**-x** dst_mask]] [**-e** [*src_esp_algo*]] [**-a** [*src_ah_algo*]] [**-p** *src_policy*] [**-A** [*dst_ah_algo]*] [**-P** *dst_policy*] [**-k** *src_esp_key*] [**-h** *src_ah_key*] [**-K** *dst_esp_key*] [**-H** *dst_ah_key*] [**-n** *src_esp_spi*] [**-u** *src_ah_spi*] [**-N** *dst_esp_spi*] [**-U** *dst_ah_spi*] [**-b** *src_enc_mac_algo*] [**-c** *src_enc_mac_key*] [**-B** *dst_enc_mac_algo*] [**-C** *dst_enc_mac_key*] [**-g**] [**-z**] [**-E**]

描述

gentun 命令在本地主机和通道伙伴主机之间创建通道定义。关联的自动生成的通道过滤规则可以用该命令可选择地生成。

标志

-a	认证算法由源主机用于 IP 数据包认证。**-a** 的有效值取决于主机上安装的是哪种认证算法。可以通过签发 **ipsecstat -A** 命令来显示所有认证算法的列表。**手动**通道的缺省值为 HMAC_MD5。
-A	（只用于**手动**通道）认证算法由目标主机用于 IP 数据包认证。**-A** 的有效值取决于主机上安装的是哪种认证算法。可以通过签发 **ipsecstat -A** 命令来显示所有认证算法的列表。如果不用这个标志，则用 **-a** 标志所使用的值。

-b （只用于**手动**通道）源主机 ESP 认证算法（仅限于新头格式）。**-b** 的有效值取决于主机上安装的是哪种认证算法。可以通过签发 **ipsecstat -A** 命令来显示所有认证算法的列表。

-B （只用于**手动**通道）目标主机 ESP 认证算法（仅限于新头格式）。**-B** 的有效值取决于主机上安装的是哪种认证算法。可以通过签发 **ipsecstat -A** 命令来显示所有认证算法的列表。如果不用这个标志，则被设置为与 **-b** 标志相同的值。

-c （只用于**手动**通道）源 ESP 认证密钥（仅限于新头格式）。这必须是一个以“Ox”为开始的十六进制字符串。如果不用这个标志，系统将为您生成一个标志。

-C （只用于**手动**通道）目标主机 ESP 认证密钥（仅限于新头格式）。这必须是一个以“Ox”为开始的十六进制字符串。如果不用这个标志，则被设置为与 **-c** 标志相同的值。

-d 目标主机 IP 地址。在主机对主机的模式中，这就是通道所用的目标主机接口的 IP 地址。在主机 - 防火墙 - 主机的模式中，这就是防火墙后面的目标主机的 IP 地址。主机名仍有效，且将使用名称服务器返回的主机名的第一个 IP 地址。

-e 加密算法，由源主机用于 IP 数据包加密。**-e** 的有效值取决于主机上安装的是哪些加密算法。可以通过签发 **ipsecstat -E** 命令来显示所有加密算法的列表。

-E （只用于**手动**通道）加密算法，由目标主机用于对 IP 数据包进行加密。**-E** 的有效值取决于主机上安装的是哪些加密算法。可以通过签发 **ipsecstat -E** 命令来显示所有加密算法。如果不用这个标志，则用 **-e** 标志所使用的值。

-f 在源主机和目标主机之间的防火墙的 IP 地址。在主机和防火墙之间将建立通道。因此，防火墙主机上必须加上相应的通道定义。主机名也可以用于该标志中，且将使用名称服务器返回的第一个 IP 地址。

-g 系统自动生成的过滤规则标志。如果不用这个标志，命令会自动为通道生成两个过滤规则。自动生成的过滤规则允许在通道两个端点之间的 IP 流量流过通道。如果指定了 **-g** 标志，命令将仅创建通道定义，且用户必须添加用户定义的过滤规则以使通道工作。

-h 这是**手动**通道的 AH 密钥字符串。输入必须是以“0x”为开始的十六进制字符串。如果不用这个标志，系统将使用随机数发生器为您生成密钥。

-H （只用于**手动**通道）目标主机 AH 的密钥字符串。输入必须是以“0x”为开始的十六进制字符串。如果不用这个标志，系统将使用随机数发生器为您生成密钥。

-k 这是**手动**通道的 ESP 密钥字符串。用于源主机创建通道。输入必须是以“0x”为开始的十六进制字符串。如果不用这个标志，系统将使用随机数发生器为您生成密钥。

-K （只用于**手动**通道）目标主机 ESP 密钥字符串。所输入的必须是以“0x”为开始的十六进制字符串。如果不用这个标志，系统将使用随机数发生器为您生成密钥。

-l 密钥使用期限，以分计。

对**手动**通道来说，该值标明了在通道到期之前的可操作时间。

对**手动**通道有效的值为 0 - 44640。值 0 标明**手动**通道永远也不会到期。**手动**通道的缺省值为 480。

-m 安全数据包方式。该值必须被指定为**通道**或**传送**。缺省值是**传送**。通道方式将封装整个 IP 数据包，而传送模式只封装 IP 数据包的数据部分。当生成主机 - 防火墙 - 主机方式（主机在防火墙后面）的通道时，此标志必须使用**通道**的值。

如果已指定 **-f** 标志的话，-m 标志必须强制使用缺省值（**通道**）。

-n （只用于**手动**通道）源 ESP 的安全参数索引。这是一个数值，和目标 IP 地址一起标识使用 ESP 的数据包所使用的安全关联。如果不用这个标志，系统将为您生成一个 SPI。

-N （只用于**手动**通道）目标主机 ESP 的安全参数索引。如果在 **-P** 标志中指定的策略包含 ESP 的话，对于**手动**通道必须输入此项。该标志不适用于 **IBM** 通道。

-p 源策略，识别该主机是如何使用 IP 数据包认证和 / 或加密的。如果指定为 **ea**，IP 数据包会先加密，然后认证。如果指定为 **ae**，则先认证，后加密。如果单独指定 **e** 或 **a**，则 IP 数据包只被加密或只认证。该标志的缺省值取决于是否提供 **-e** 和 **-a** 标志。**ea** 只有在 **-e** 和 **-a** 标志两者提供或都不提供时才成为缺省策略。否则，策略将反映支持的是 **-e** 和 **-a** 标志中的哪一个。

-P （只用于**手动**通道）目标主机策略，标识目标主机是如何使用 IP 数据包认证和 / 或加密的。如果指定为 **ea**，则 IP 数据包先加密，后认证。如果指定为 **ae**，则先认证，后加密。如果只指定 **e** 或 **a** 则对 IP 数据包只加密或只认证。**ea** 只有在 **-E** 和 **-A** 标志两者提供或都不提供时才成为缺省策略。否则，策略将反映是支持 **-E** 和 **-A** 标志中的哪一个。

-s 源主机 IP 地址，通道所用的本地主机接口的 IP 地址。主机名仍有效且将使用名称服务器返回的主机名的第一个 IP 地址。

-t 通道类型。必须指定为**手动**。

当使用**手动**通道时，需要手动执行初始通道密钥和任一后续密钥的更新。一旦手动安装了密钥，所有通道操作都使用该相同的密钥，直到手动更改密钥为止。

当您想构建带非 IBM 的 IP 安全主机或任何 IP V6 端点（此处的端点支持 IP 通道的新建 IP 安全封装格式的 RFC 1825-1829 或 IETF 草图）的通道时，应该选择**手动**通道值。

-u （只用于**手动**通道）源 AH 的安全参数索引。使用 SPI 和目标 IP 地址决定 AH 使用的安全关联。如果不用这个标志，则使用 -n SPI 的值。

-U （只用于**手动**通道）目标主机 AH 的安全参数索引。如果不用这个标志，则使用 -N spi 的值。

-v 为其创建通道的 IP 版本。对于 IP V4 通道，使用值 **4**。对于 IP V6 通道，使用值 **6**。

-x 防火墙后用于安全网络的网络掩码。目标主机是安全网络的成员。**-d** 与 **-x** 的组合使源主机可以通过源防火墙通道（必须以通道方式）与安全网络中的多个主机进行通信。

只有使用 **-f** 标志时，该标志才有效。

-y （只用于**手动**通道）阻止重播标志。只有 ESP 或 AH 头使用新头格式时，阻止重播才有效（见 **-z** 标志）。**-y** 标志的有效值为 Y（是）和 N（否）。如果该标志的值为 Y，所有用于该通道的封装（AH、ESP、发送和接收）将都使用重播域。缺省值为 N。

-z （只用于**手动**通道）新头格式标志。新头格式在 ESP 和 AH 头中为阻止重播保留了一个字段，也允许 ESP 认证。仅当重播标志（**-y**）设置为 Y 时，才使用重播字段。**-z** 标志的有效值为 Y（是）和 N（否）。不用 **-z** 标志时的缺省值取决于您为通道所选的算法。缺省值为 N，除非用于 **-a** 或 **-A** 标志的是算法不是 KEYED_MD5，或如果用的是 **-b** 或 **-B** 标志。

相关信息

chtun 命令、**exptun** 命令、**imptun** 命令、**lstun** 命令、**mktun** 命令和 **rmtun** 命令。

---

genxlt 命令

用途

生成一个代码设置转换表供 **lconv** 库使用。

语法

genxlt [OutputFile]

描述

本 **genxlt** 命令从标准输入中读取一个源代码设置转换表文件并将编译版本写入由 *OutputFile* 参数指定的文件。如果未指定一个值给 *OutputFile* 参数，则使用标准输出。本源代码设置转换表文件包含 **genxlt** 命令生成编译版本使用和遵循的伪指令。

代码设置转换表源文件的格式为:

- 开始的非空格字符是 #（磅符号）的行被视为注释行。
- 空行和只包含空格字符的行被视为注释行。
- 非注释行具有以下格式:

```
%token <blank>   # <tab> and <space>
%token <hex>     # <zero>, <one>, <two>, <three>, <four>,
                 # <five>, <six>, <seven>, <eight>, <nine>,
                 # <a>, <b>, <c>, <d>, <e>, <f>,
                 #  <A>, <B>, <C>, <D>, <E>, <F>,
%token <any>     # any character but '\n'
```

```
line    :  offset  blank  value  blank  comment  '\n'
        | 'SUB'  blank  value  blank  comment  '\n'
        ;

blank    : <blank>
        | blank <blank>
        ;

offset   :  '0x'  <hex>
        |  offset  <hex>
        ;


value    :  offset
        |  'invalid'
        |  'invalid'
        |  'substitution'
        ;


comment  :  '#'  <any>
        |  comment  <any>
        ;
```

偏移是‘SUB’的行被用来指定缺省置换字符。

如果表被设置为“置换”，使用本表的 **iconv** 转换器利用 SUB 值作为偏移量。

如果值被设定为“无效”，则使用本表的 **iconv** 转换器为偏移返回错误。

如果偏移在源代码设置转换表文件中多次被找到，则最后的条目被用在转换表的编译。

偏移及值必须在 0x00 到 0xff（包含 0x00 和 0xff）的范围之间。

以下是一个代码设置转换表的摘录:

```
SUB     0x1a     substitute character
0x80    0xc7     C  cedilla
0x81    0xfc     u  diaeresis
0x82    0xe9     e  acute
0x83    0xe2     a  circumflex
0x84    0xe4     a  diaeresis
0x85    0x40     a  grave
0x9F            substitution
0xff            invalid
```

如果成功，**genxlt** 命令以值 0 退出。如果输出文件打不开，**genxlt** 命令不成功并以值 1 退出。如果输入流中检测出语法错误，**genxlt** 命令将以值 2 立即退出，并将发生语法错误的行号写到标准错误。

由 **genxlt** 命令生成的文件名必须遵循以下的名称约定，为了使 **iconv** 子系统识别其作为一个转换文件:

```
fromcode: "IBM-850"
tocode: "ISO8859-1"
conversion table file: "IBM-850_ISO8859-1"
```

转换表名称由 tocode 文件名和 fromcode 文件名合并而成，以下划线连接。

示例

生成一个非英文、用户定义的代码设置转换表，请输入:

```
cp  /usr/lib/nls/loc/iconvTable/ISO8859-1_IBM-850_src  $HOME
vi  $HOME/ISO8859-1_IBM-850_src
genxlt  <  $HOME/ISO8859-1_IBM-850_src  >  cs1_cs2
```

相关信息

iconv 命令。

iconv_open 子例程、**iconv** 子例程及 **iconv_close** 子例程提供在一个程序中使用转换服务的方法。

本地语言支持及《*AIX 5L V5.3 本地语言支持指南和参考大全*》中的『编程的转换器概述』。

---

get 命令

用途

创建指定版本的 SCCS 文件。

语法

获取 SCCS 文件的只读版本

get [**-g**] [**-m**] [**-n**] [**-p**] [**-s**] [**-c** *Cutoff*] [**-i***List*] [**-r** *SID*] [**-t**] [**-x** *List*] [**-w** *String*] [**-l** [**p**]] [**-L**] *File ...*

获取 SCCS 文件的可编辑版本

get [**-e**] [**-k**] [**-b**] [**-s**] [**-c** *Cutoff*] [**-i** *List*] [**-r** *SID*] [**-t**] [**-x** *List*] [**-l** [**p**]] [**-L**] *File ...*

描述

get 命令读取“源代码控制系统（SCCS）”文件的指定版本，并根据指定的标志创建 ASCII 文本文件。然后 **get** 命令将每一个文本文件写入与原 SCCS 文件同名但不含 **s.** 前缀的文件（**g 文件**）。

标志与文件可按任意顺序指定，所有标志应用于所有指定的文件。如果为 *File* 参数指定一个目录，则 **get** 命令在该目录下以 **s.** 前缀开头的所有文件上执行请求的操作。如果指定 **-**（减号）给 *File* 参数，则 **get** 命令读取标准输入并将每行解释为 SCCS 文件名。**get** 命令继续读取输入，直到它读到文件结束符。

如果有效用户在包含 SCCS 文件的目录下具有写权限，而实际用户却没有，则当使用 **-e** 标志时只可以指定一个文件。

注: **get** 命令支持用 **w** 标志指定的文件名和字符串数据的“多字节字符集（MBCS）”。

获取只读文件版本

get 命令创建文件的只读版本和可编辑版本。当应用程序不要求更改文件内容时则应使用文件的只读版本。可编译源代码文件的只读版本。文本文件可从只读版本显示或打印出来。

在使用标识关键字时可编辑版本与只读版本之间的区别是很重要的。*标识关键字*是当 **get** 命令检索只读文件时扩展到一些文本值的符号。在可编辑版本中，不扩展关键字。标识关键字可出现在 SCCS 文件的任何位置。有关标识关键字的进一步信息，请参阅 **prs** 命令。

SCCS 文件

除了具有 **s.** 前缀的文件（**s 文件**），**get** 命令还创建多个辅助文件: **g 文件**、**l 文件**、**p 文件** 和 **z 文件**。这些文件由它们的标记（连字号前面的字母）标识。**get** 程序通过将 SCCS 文件名中前面的 **s.** 替换为适当的标记来命名辅助文件，**g 文件** 例外，它是通过除去 **s.** 前缀来命名。例如，名为 **s.sample** 的文件，辅助文件名应为 **sample**、**l.sample**、**p.sample** 和 **z.sample**。

这些文件服务于以下用途:

s 文件 包含原文件文本以及所有对文件所作的更改（变化量）。它还包括如下信息: 谁能更改文件内容，谁进行了更改，什么时候进行的这些更改以及所作更改的本质。因为是只读文件，所以不能直接编辑该文件。然而，它包含 SCCS 命令构建您可编辑的 **g 文件**所需的信息。

g 文件 ASCII 文本文件，它包含用 **-r** 标志指定的 SCCS 文件版本（或标志缺省时为最新的干线版本）。该文件可以直接编辑。当完成了所有更改并要对文件进行新的更改时，则在此文件上运行 **delta** 命令。**get** 命令在当前目录创建 **g 文件**。

它每次运行时，**get** 命令会创建一个 g 文件，除非指定了 **-g** 标志或 **-p** 标志。实际用户拥有它（而非有效用户）。如果不指定 **-k** 或 **-e** 标志，则文件为只读文件。如果指定了 **-k** 或 **-e** 标志，则所有者具有 **g 文件**的写权限。要创建 **g 文件**，必须在当前目录下具有写权限。

l 文件 当指定了 **-l** 标志时，**get** 命令创建 **l 文件**。**l 文件**是只读文件。它包含显示生成 **g 文件**时应用的变化量的表。要创建 **l 文件**，必须在当前目录下具有写权限。**l 文件**中的行具有以下格式:

- 如应用了变化量，则为空格字符，否则为星号。
- 如应用了变化量或不应用并忽略变化量时，为空格字符。如不应用且不忽略变化量，则出现星号。
- 一个代码表明变化量应用与否的特殊原因:

  空格 通常包含或排除

  I 使用 **-i** 标志包含

  X 使用 **-x** 标志排除

  C 使用 **-c** 标志切断

- SID。
- 文件创建的日期与时间。
- 变化量创建者的登录名。

注释与“修改请求（MR）”数据跟在后续行之后，缩进一个水平制表符。每个条目以空行结束。例如，要用 **-c** 标志切断变化量，则 l 文件中的条目可能为:

```
**C 1.3 85/03/13 12:44:16 pat
```

且初始变化量的条目可能为:

```
1.1 85/02/27 15:42:20 pat
创建日期和时间 85/02/27 15:42:20 由 pat 创建
```

p 文件 当指定了 **-e** 或 **-k** 标志时，**get** 创建 **p 文件**。**p 文件**将 **get -e** 命令生成的信息传递到 **delta** 命令。**p 文件**还在运行 **delta** 命令或者将联合编辑键字母（**j**）设置在 SCCS 文件中之前防止对相同的 SID 执行后续的 **get -e** 命令。**j** 键字母允许在同一 SID 上运行多个 **get** 命令。**p 文件**创建于包含 SCCS *File* 的目录中。要在 SCCS 目录中创建 **p 文件**，必须具有该目录下的写权限。除其所有者之外，**p 文件**的许可权代码对所有人都是只读的，且它属有效用户所有。**p 文件**不应直接由其所有者编辑。**p 文件**包含:

- 当前 SID
- 要创建的新变化量的 SID
- 用户名
- **get** 命令的日期与时间
- **-i** 标志，如果存在
- **-x** 标志，如果存在

p 文件包含具有文件的每个暂挂变化量的前导信息的项。任意两行不能有相同的新变化量 SID。

z 文件 **z 文件**是相对同时更新的锁定机制。**z 文件**包含创建它的 **get** 命令的二进制进程编号。该文件创建在包含 SCCS 文件的目录里，仅当 **get** 命令运行时存在。

当使用 **get** 命令时，它显示正在访问的 SID 以及从 SCCS 文件所创建的行数。如果指定 **-e** 标志，则所要作的变动的 SID 出现在访问 SID 之后和创建行数之前。如果指定不止一个文件、目录或标准输入，则 **get** 命令在处理每个文件之前显示文件名。如果指定 **-i** 标志，则 **get** 命令列出在 `Included` 字之下包含的变化量。如果指定 **-x** 标志，则 **get** 命令列出在 `Excluded` 字下排除的变化量。

下表举例说明了 **get** 命令如何决定它所检索文件的 SID 以及暂挂 SID。“指定的 SID”列显示了用 **-r** 标志指定 SID 的不同方法。第一列还说明可以存在的各种情况，包括 **-b** 标志是否与 **get -e** 命令一起使用。“检索的 SID”列表示构成 **g 文件**的文件的 SID。“要创建的变化量的 SID”列表示当应用 **delta** 命令时将会创建的版本的 SID。

SID 确定

指定的 SID	检索的 SID	要创建的变化量的 SID
无 [1] **-b Used?** 无 **其他条件** R 缺省为 mR [2]	mR.mL	mR.(mL+1)
无 [1] **-b Used?** yes **其他条件** R 缺省为 mR	mR.mL	mR.mL.(mB+1).1
R **-b Used?** 无 **其他条件** R>mR	mR.mL	R.1 [3]
R **-b Used?** 无 **其他条件** R=mR	mR.mL	mR.(mL+1)
R **-b Used?** yes **其他条件** R>mR	mR.mL	mR.mL.(mB+1).1
R **-b Used?** yes **其他条件** R=mR	mR.mL	mR.mL.(mB+1).1

SID 确定

指定的 SID	检索的 SID	要创建的变化量的 SID
R **-b Used?** N/A **其他条件** R<mR 和 R 不存在	hR.mL [4]	hR.mL.(mB+1) .1
R **-b Used?** N/A **其他条件** 在大于等于 R 的发行版中存在干线后续作业	R.mL	R.mL.(mB+1).1
R.L. **-b Used?** 无 **其他条件** 无干线后续作业	R.L.	R.(L+1)
R.L. **-b Used?** yes **其他条件** 无干线后续作业	R.L.	R.L(mB+1).1
R.L. **-b Used?** N/A **其他条件** 在大于等于 R 的发行版中的干线后续作业	R.L.	R.L.(mB+1).1
R.L.B. **-b Used?** 无 **其他条件** 无分支后续作业	R.L.B.mS	R.L.B.(mS+1)
R.L.B. **-b Used?** yes **其他条件** 无分支后续作业	R.L.B.mS	R.L.(mB+1).1
R.L.B.S. **-b Used?** 无 **其他条件** 无分支后续作业	R.L.B.S.	R.L.B.(S+1)

SID 确定

指定的 SID	检索的 SID	要创建的变化量的 SID
R.L.B.S. **-b Used?** yes **其他条件** 无分支后续作业	R.L.B.S.	R.L.(mB+1).1
R.L.B.S. **-b Used?** N/A **其他条件** 分支后续作业	R.L.B.S.	R.L.(mB+1).1
注: 在 “SID 确定” 表中，字母 R、L、B、S 表示 SID 的发行版、级别、分支和序列组件。字母 *m* 表示最大值。		

[1] 仅当**-d**（缺省 SID）标志在文件之中不存在时应用（请参阅 **admin** 命令）。

[2] mR 表示现有最高发行版本。

[3] 在新的发行版中第一个变化量的强制创建。

[4] hR 是比指定的不存在的发行版本 R 低的现有的最高发行版本。

标识关键字

通过在每次标识关键字出现时替换它们的值将标识信息插入到从 SCCS 文件检索的文本之中。以下关键字可用于存储在 SCCS 文件中的文本:

关键字	值
%M%	模块名称: 文件中的 **m** 标志的值，或者，如果不存在，则为除去带 **s.** 的 SCCS文件的名称。
%I%	检索文本的 SCCS 标识（SID）（*%R%.%L%* 或 *%R%.%L%.%B%.%S%*）。
%R%	发行版。
%L %	级别。
%B%	分支。
%S%	序列。
%D%	当前日期，格式为 *YY/MM/DD*。
%H%	当前日期，格式为 *MM/DD/YY*。
%T%	当前时间，格式为 *HH:MM:SS*。
%E%	创建最新应用的变化量的日期，格式为 *YY/MM/DD*。
%G%	创建最新应用的变化量的日期，格式为 *MM/DD/YY*。
%Y%	模块类型: SCCS 文件中 **t** 标志的值。
%F%	SCCS 文件名。
%P%	SCCS 绝对路径名。
%Q%	文件中 **-q** 标志的值。
%C%	当前行号。此关键字用于标识程序输出的消息，例如这本不应该发生错误消息。**%C%** 不会用于每行来提供序列号。
%Z%	四字符字符串 `@(#)`，可通过 *what* 识别。
%W%	构造 *what* 字符串的速记符号: `%W% = %Z%%M%<tab>%I%`
%A%	另一个构造 *what* 字符串的速记符号: `%A% = %Z%%Y% %M% %I%%Z%`

标志

-b 指定要创建的 delta 在新分支中应该有一个 SID。新 SID 根据 “SID 确定” 表中给出的规则编号。可仅与 **-e** 标志一起使用 **-b** 标志。仅当想从叶增量（无后续作业的变化量）分支时才需要它。尝试在非叶增量上创建变化量会自动产生分支，即使未设置 **b** 头标志。如果在 SCCS 文件中不指定 **b** 头标志，**get** 命令将忽略 **-b** 标志，因为该文件不允许分支。

-c *Cutoff* 指定切断的日期和时间，格式为 *YY[MM[DD[HH[MM[SS]]]]]*。**get** 命令不包含在 g 文件中指定切断后对 SCCS 文件所创建的变化量。*Cutoff* 变量中任何未指定项的值缺省为其最大许可值。这样，一个只指定了年份的切断日期与时间将指定该年的最后的一个月份、日、小时、分钟和秒。任意数量的非数字字符可分隔 *Cutoff* 变量日期和时间的两位项。这就允许用若干种方法指定日期和时间，如下所示:

```
-c85/9/2,9:00:00
-c"85/9/2 9:00:00"
"-c85/9/2 9:00:00"
```

-e 表示正在创建的 g 文件将由用户通过应用 **get** 命令来编辑。之后由 **delta** 命令记录变化量。**get -e** 命令创建 p 文件以防止其他用户发出另一个 **get -e** 命令以及在 **delta** 命令运行之前于同一个 SID 上编辑第二个 g 文件。文件所有者可以通过使用带 **-fj** 标志的 **admin** 命令来允许在同一个 SID 上的联合编辑从而覆盖此限制。其他用户可在得到许可的条件下，通过不带 **-e** 标志的 **get** 命令获取只读副本。**get -e** 命令执行用顶、底和 SCCS 文件中授权用户列表来指定的 SCCS 文件保护。请参阅 **admin** 命令。

注: 如果意外破坏了用 **get -e** 命令创建的 g 文件，可以通过 **get -k** 命令重新创建该文件。

-g 禁止 **g 文件**的实际创建。首先用 **-g** 标志创建 **l 文件**或验证特殊 SID 的存在。请勿将其与 **-e** 标志一起使用。

-i *List* 指定要包含在 **g 文件**的创建之中的变化量列表。SID 列表格式由逗号分隔的各 SID 的组合以及由连字符分隔的两个 SID 表示的 SID 范围构成。可以用如下任一命令行来指定同一个 SID。

```
get -e -i1.4,1.5,1.6 s.file
get -e -i1.4-1.6 s.file
```

可以通过上表所示的 “指定的 SID” 列中的任一形式来指定变化量的 SCCS 标识。**get** 命令解释部分的 SID，如 “检索的 SID” 列中所示。

-k 禁止在 **g 文件**中用它们的值来替换标识关键字。**-k** 标志由 **-e** 标志来暗示。如果意外破坏了用 **get -e** 命令创建的 **g 文件**，可以通过重新发出 **get** 命令和 **-k** 标志来重新创建文件而不用 **-e** 标志。

-l[**p**] 将变化量摘要写入到 l 文件中。如果指定 **-lp**，则变化量摘要写到标准输出，且 **get** 命令不创建 **l 文件**。用该标志来决定将哪些变化量用于创建当前正在使用的 g 文件。有关 **l 文件**的格式，请参阅 **sccsfile** 文件。另见 **-L** 标志。

-L 将变化量摘要写到标准输出。指定 **-L** 标志与使用 **-lp** 标志相同。

-m 在 **g 文件**每行文本之前写入将行插入到 SCCS 文件之中的变化量的 SID。格式为:

```
SID 制表符文本行
```

-n 在 **g 文件**文本的每行之前写入 **%M%** 关键字的值。格式是 **%M%** 的值后跟水平制表符，后跟文本行。当同时使用 **-m** 和 **-n** 标志时，格式为:

```
%M% 值 制表符 SID 制表符 文本行
```

-p 将从 SCCS 文件创建的文本写到标准输出并且不创建 g 文件。所有通常发送到标准输出的信息输出被发送到标准错误，除非指定 **-s** 标志和 **-p** 标志。这时，通常发送到标准输出的输出在任何地方都不会出现。

-r *SID* 指定要创建的 SCCS 文件版本的 SCCS 标识字符串（SID）。SID 确定表显示创建的文件的版本并将暂挂变化量的 SID 显示为指定 SID 的函数。

-s 禁止通常要写入到标准输出的所有输出。错误消息（写到标准错误输出），不受影响。

-t 访问给定发行版本中创建的最新的变化量或为给定发行版本和级别创建的最新的变化量。

-w *String* 用 *String* 的值替换 g 文件中不用于编辑的 **%W%** 关键字。

-x *List* 排除 **g 文件**创建中指定的变化量列表。有关 SID 列表格式，请参阅 **-i** 标志。

退出状态

此命令返回以下出口值:

0 成功完成。
>0 发生错误。

示例

以下描述与示例说明了只读与可编辑版本文件之间的区别。

1. 要打印文件中的当前日期与 SID，请将以下符号置于文件之中;

   ```
   %H% %I%
   ```

   %H% 是当前日期的符号，**%I%** 是 SID 符号。当 **get** 命令检索可编辑文件时，它将符号留在文件之中，不执行文本值替换。

2. 以下 **get** 命令的示例构建了最高的 SID 版本，因为该示例未指定文件的版本:

   ```
   $ ls
   s.test.c
   $ get s.test.c
   3.5
   59 line
   $ ls
   s.test.c test.c
   ```

3. 在下两个示例中，**-r** 标志指定了要获取的版本:

   ```
   $ get -r1.3 s.test.c
   1.3
   67 lines

   $ get -r1.3.1.4 s.test.c
   1.3.1.4
   50 lines
   ```

4. 如果仅指定了 SID 的发行版号，则 **get** 命令在发行版号之内查找最高级别的文件。

   ```
   $ get -r2 s.test.c
   2.7
   21 lines
   ```

5. 如果指定的 SID 大于现有最大 SID，则 **get** 命令将获取该现有最大 SID。如果指定的 SID 小于现有最小 SID，则 SCCS 写入一个出错消息。在以下示例中，发行版 7 是现有的最高发行版:

   ```
   $ get -r9 s.test.c
   7.6
   400 lines
   ```

6. **-t** 标志在给定的发行版或级别中获取最高版本。最高版本是最近创建的变化量，与其位置无关。在下一个示例中，发行版 3 中现有的最高变化量为 3.5，而最新创建的变化量为 3.2.1.5。

   ```
   $ get -t -r3 s.test.c
   3.2.1.5
   46 lines
   ```

7. 上述示例使用 **get** 命令获取只读文件。要创建可编辑并可用于创建新变化量的文件副本，请使用带 **-e** 标志的 **get** 命令。用 **unget** 来撤销 **get -e** 命令的作用并废弃创建变化量之前对文件所作的任何更改。以下示例显示如何使用 **-e** 标志:

   ```
   $ ls
   s.test.c
   $ get -e s.test.c
   1.3
   ```

```
new delta 1.4
67 lines
$ ls
p.test.c s.test.c test.c
```

工作文件为 test.c。如果编辑该文件并用 **delta** 命令保存更改，则 SCCS 会创建具有 SID 1.4 的新变化量。文件 p.test.c 为临时文件，由 SCCS 用于跟踪文件版本。

在上述示例中，还可以用 **-r** 标志获取特定的版本。假设发行版 1 是现有的最高发行版，而变化量 1.3 早已存在并且是发行版本中最高的变化量，则以下三种 **get** 命令的用法是等效的:

```
$ get -e s.test.c
$ get -e -r1 s.test.c
$ get -e -r1.3 s.test.c
```

8. 要开始使用新的（更高值）发行版号，请用 **-r** 标志获取该文件，并指定一个比当前最高发行版号更高的发行版号。在下一个示例中，发行版 2 还不存在:

```
$ get -e -r2 s.test.c
1.3
new delta 2.1
67 lines
```

注意 **get** 命令表示当 **delta** 命令存储对 SCCS 文件的更改时将要创建的新变化量的版本。

9. 要创建一个分支变化量，请使用 **-r** 标志并指定出现分支处的发行版和级别。在下一个示例中，变化量 1.3 和 1.4 已经存在。

```
$ get -e -r1.3 s.test.c
1.3
new delta 1.3.1.1
67 lines
```

用相同方法在分支上创建变化量。

要编辑一个文件，用 **get -e** 命令获取文件版本并用 **delta** 命令保存更改。可存在 SCCS 文件的多个不同可编辑版本，只要它们在不同目录之中。如果试图（用 **get** 命令）而不用 **delta** 命令将可编辑文件版本的副本放入一个目录下，则 SCCS 写入一个错误消息。

要不止一次获取同一个可编辑文件版本，在 SCCS 文件中用 **admin** 命令设置 **j** 头标志。通过使用 **-f** 标志设置 **j** 选项。可从不同目录下多次获取同一个 SID，为每个 **get** 命令创建一个独立的文件。尽管文件起源于单个 SID，SCCS给每一个文件唯一的新 SID。

10. 在以下示例中，**pwd** 命令显示了当前目录。然后用 **admin** 命令设置 **j** 选项:

注: 在本例中您必须在两个目录下都有写访问权才能发出命令。

```
$ pwd
/home/marty/sccs
$ admin -fj s.test.c
```

11. 然后用 **get** 命令来检索文件的最新版本:

注: 在本例中您必须在两个目录下都有写访问权才能发出命令。

```
$ get -e s.test.c
1.1
new delta 1.2
5 lines
```

12. 更改为 /home/new 目录，并再次发出 **get** 命令。

注: 在本例中您必须在两个目录下都有写访问权才能发出命令。

```
$ cd /home/new
$ get -e /home/marty/sccs/s.test.c
1.2
new delta 1.1.1.1
5 lines
```

注意 SCCS 从 1.1 的单个原始文件版本创建两个变化量，1.2 和 1.1.1.1。请参照文件 **p.test.c**。它显示当前使用中的每个版本的独立条目。**p.test.c** 文件保留在目录中直到您用 **delta** 命令或 **unget** 命令来处理两个文件版本。

文件

/usr/bin/get 包含 **get** 命令。

相关信息

admin 命令、**delta** 命令、**prs** 命令和 **sact** 命令、**sccshelp** 命令、**unget** 命令、**what** 命令。

AIX 5L Version 5.3 Files Reference 中的 **sccsfile** 文件格式。

在《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『SCCS 命令列表』。

在《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『源代码控制系统（SCCS）概述』。

---

getconf 命令

用途

将系统配置变量值写入标准输出。

语法

getconf [**-v** *specification*] [*SystemwideConfiguration* | *PathConfiguration PathName*] [*DeviceVariable DeviceName*]

getconf -a

描述

用 *SystemwideConfiguration* 参数调用 **getconf** 命令，将 *SystemwideConfiguration* 参数所指定的变量值写到标准输出。

用 *PathConfiguration* 和 *PathName* 参数调用 **getconf** 命令，写入 *PathConfiguration* 参数为 *PathName* 参数指定路径指定的变量值，到标准输出。

getconf 命令，用 **-a** 标志调用，并写入全部系统配置变量值到标准输出。

getconf 命令，用 *DeviceVariable* 和 *DeviceName* 参数调用，写磁盘设备名或位置的值到标准输出，设备路径由 *DeviceName* 参数指定。

如果在系统中定义指定的变量且其值描述为可从 **confstr** 子例程得到，则指定变量的值按以下格式写入:

“%s\n”，< 值 >

否则，如果在系统中定义指定变量，其值按以下格式写入:

“%d\n”，< 值 >

如果指定变量是有效的但在系统中未定义，则将以下内容写入标准输出:

“undefined\n”

如果变量名无效或出现错误，就会有一个诊断消息写入标准错误。

标记

-a *specification*　显示了指定规格及版本，其配置变量等待确定。如果该标志未被指定，返回值将响应一个实现缺省值 XBS5 的相应的编辑环境。

-v　将全部系统配置变量值写入标准输出。

参数

PathName　为 *PathConfiguration* 参数指定路径名。

SystemwideConfiguration　指定一个系统配置变量。

PathConfiguration　指定一个系统路径配置变量。

DeviceName　指定一个设备路径名。

DeviceVariable　指定一个设备变量。

当列入以下的表格中的第一列符号被用作 **system_var** 操作数时，**getconf** 将产生与用第二列的值调用 **confstr** 时相同的值:

注: **_CS_AIX_ARCHITECTURE** 和 **_CS_AIX_BOOTDEV** 变量，用做 **confstr** 参数时，只对 root 用户可用。

system_var	confstr 名称值
BOOT_DEVICE	_CS_AIX_BOOTDEV
MACHINE_ARCHITECHTURE	_CS_AIX_ARCHITECTURE
MODEL_CODE	_CS_AIX_MODEL_CODE
PATH	_CS_PATH
XBS5_ILP32_OFF32_CFLAGS	_CS_XBS5_ILP32_OFF32_CFLAGS
XBS5_ILP32_OFF32_LDFLAGS	_CS_XBS5_ILP32_OFF32_LDFLAGS
XBS5_ILP32_OFF32_LIBS	_CS_XBS5_ILP32_OFF32_LIBS
XBS5_ILP32_OFF32_LINTFLAGS	_CS_XBS5_ILP32_OFF32_LINTFLAGS
XBS5_ILP32_OFFBIG_CFLAGS	_CS_XBS5_ILP32_OFFBIG_CFLAGS
XBS5_ILP32_OFFBIG_LDFLAGS	_CS_XBS5_ILP32_OFFBIG_LDFLAGS
XBS5_ILP32_OFFBIG_LIBS	_CS_XBS5_ILP32_OFFBIG_LIBS
XBS5_ILP32_OFFBIG_LINTFLAGS	_CS_XBS5_ILPBIG_OFF32_LINTFLAGS
XBS5_LP64_OFF64_CFLAGS	_CS_XBS5_LP64_OFF64_CFLAGS
XBS5_LP64_OFF64_LDFLAGS	_CS_XBS5_LP64_OFF64_LDFLAGS
XBS5_LP64_OFF64_LIBS	_CS_XBS5_LP64_OFF64_LIBS
XBS5_LP64_OFF64_LINTFLAGS	_CS_XBS5_LP64_OFF64_LINTFLAGS
XBS5_LPBIG_OFFBIG_CFLAGS	_CS_XBS5_LPBIG_OFFBIG_CFLAGS

system_var	confstr 名称值
XBS5_LPBIG_OFFBIG_LDFLAGS	_CS_XBS5_LPBIG_OFFBIG_LDFLAGS
XBS5_LPBIG_OFFBIG_LIBS	_CS_XBS5_LPBIG_OFFBIG_LIBS
XBS5_LPBIG_OFFBIG_LINTFLAGS	_CS_XBS5_LPBIG_OFFBIG_LINTFLAGS

环境变量

以下的环境变量影响 **getconf** 的执行:

LANG　为取消设置或空的国际化变量值提供一个缺省值。如果 LANG 被取消设置或空，则将使用来自依赖实现的 l 缺省语言环境的相应值。如果任何一个国际化变量包含一个无效设置，实用程序将象未定义变量义变量来运行。

LC_CALL　如果设置为一个非空字符串值，则所有其他国际化变量值会被覆盖。

LC_CTYPE　确定将文本数据字节序列解释为字符的语言环境（例如，在参数中，单字节字符与多字节字符相对）。

LC_MESSAGES　确定用以影响写入标准错误的诊断消息格式和内容的语言环境。

NLSPATH　决定处理 LC_MESSAGES 消息目录的位置。

系统范围配置变量

SystemwideConfiguration 参数指定系统配置变量，其值在整个系统中都可用。有两种系统配置变量:

- 系统范围配置变量
- 系统标准配置变量

系统范围配置变量

系统范围配置变量包含系统各部分中的最小值。以下的列表定义系统范围配置变量，这些变量与 **getconf** 命令一起使用:

_CS_PATH　**PATH** 环境变量值，用以查找命令。

ARG_MAX　用于一个 **exec** 子例程的参数的最大长度（以字节计），包含环境数据。

BC_BASE_MAX　**bc** 命令使用的 **obase** 变量允许的最大值。

BC_DIM_MAX　**bc** 命令所允许的一个数组中的最大元素数。

BC_SCALE_MAX　**bc** 命令使用的 **scale** 变量允许的最大值。

BC_STRING_MAX　**bc** 命令可接受的字符串常量的最大长度。

CHARCLASS_NAME_MAX　字符类名中的最大字节数。

CHAR_BIT　类型 **character** 的位数。

CHAR_MAX　类型 **character** 的最大值。

CHAR_MIN　类型 **character** 的最小值。

CHILD_MAX　每个真实用户标识同时打开进程的最大数量。

CLK_TCK　由 **time** 子例程返回的每秒钟时钟计数。

COLL_WEIGHTS_MAX　配给在一个语言环境定义文件的 **LC_COLLATE** 语言环境节中条目的最大权重。

CS_PATH　**PATH** 环境变量值，用以查找命令。

EXPR_NEST_MAX　圆括号内可嵌套的最大表达式数，用于 **expr** 命令。

INT_MAX　类型的最大值，类型为 **int**。

INT_MIN　类型的最小值，类型为 **int**。

LINE_MAX　当实用程序被描述成为处理文本文件时，命令输入行（标准输入或者是其他文件）的最大长度（以字节计）。长度包含换行字符的空间。

LONG_BIT　类型中的位数，类型为 **long int**。

LONG_MAX　类型的最大值，类型为 **long int**。

LONG_MIN　类型的最小值，类型为 **long int**。

MB_LEN_MAX　在所有支持的语言环境中一个字符的最大字节数。

NGROUPS_MAX 每个进程中同时补充的最大组标识数。
NL_ARGMAX 在 **printf** 和 **scanf** 子例程调用中的数字最大值。
NL_LANGMAX 在一个 LANG 名中最大字节数。
NL_MSGMAX 最大消息数。
NL_NMAX 一个从 N 到 1 整序映射中最大字节数。
NL_SETMAX 最大设置数。
NL_TEXTMAX 一个消息字符串中最大字节数。
NZERO 缺省进程优先级。
OPEN_MAX 在进程中可同时打开的最大文件数。
PATH 用以查找命令的以冒号相隔的路径前缀的序列。
RE_DUP_MAX 当使用间隔符号参数时允许正则表达式重复出现次数的最大值，正如 **ed** 命令使用的 *m* 和 *n* 参数。
SCHAR_MAX 类型的最大值，类型为 **signed char**。
SCHAR_MIN 类型的最小值，类型为 **signed char**。
SHRT_MAX 类型的最大值，类型为 **short**。
SHRT_MIN 类型的最小值，类型为 **short**。
SSIZE_MAX 一个类型为 **ssize_t** 的对象的最大值。
STREAM_MAX 一个进程可同时打开的流数。
TMP_MAX 由 **tmpnam** 子例程生成的唯一路径名的最小数量。一个应用程序可靠调用 **tmpnam** 子例程的最多次数。
TZNAME_MAX 一个时区名支持的最大字节数（非 **TZ** 环境变量的长度）。
UCHAR_MAX 类型的最大值，类型为 **unsigned char**。
UINT_MAX 类型的最大值，类型为 **unsigned int**。
ULONG_MAX 类型的最大值，类型为 **unsigned long int**。
USHRT_MAX 类型的最大值，类型为 **unsigned short int**。
WORD_BIT 字或类型 **int** 中的位数。
KERNEL_BITMODE 内核的位方式，32 位或 64 位.
REAL_MEMORY 真实内存大小。
HARDWARE_BITMODE 计算机硬件的位方式，32 位或 64 位。
MP_CAPABLE 计算机微处理器容量。

系统标准配置变量

系统标准配置变量包含由一个特殊系统标准所要求的*最小*值。**_POSIX_**、**POSIX2_** 和 **_XOPEN_** 前缀显示变量包含分别由 POSIX 1003.1、POSIX 1003.2 和 X/Open 系统标准要求的系统特性最小值。系统标准是系统满足的用来支持特定系统标准的全系统最小值。实际配置值可能超出这些标准。用于 **getconf** 命令的这些系统标准配置变量的定义如下:

_POSIX_ARG_MAX 用于 **exec** 子例程的参数的最大长度（以字节为长度），包含环境数据。
_POSIX_CHILD_MAX 每个真实用户标识同时打开进程的最大数量。
_POSIX_JOB_CONTROL 如果系统支持作业控制，则值为 1。
_POSIX_LINK_MAX 到单个文件的最大链接数。
_POSIX_MAX_CANON 在终端规范输入队列中的最大字节数。
_POSIX_MAX_INPUT 在终端输入队列中允许的最大字节数。
_POSIX_NAME_MAX 文件名中的最大字节数（不包含终止空字符）。
_POSIX_NGROUPS_MAX 每个进程中同时补充的最大组标识数。
_POSIX_OPEN_MAX 在进程中可同时打开的最大文件数。
_POSIX_PATH_MAX 路径名中的最大字节数。
_POSIX_PIPE_BUF 写入管道时保证成为原子的最大字节数。
_POSIX_SAVED_IDS 值为 1 。每个进程具有保存的 set-user-ID 和保存的 set-group-ID。
_POSIX_SSIZE_MAX 可存进一个类型为 **ssize_t** 对象的最大值。
_POSIX_STREAM_MAX 一个进程可同时打开的流数。
_POSIX_TZNAME_MAX 一个时区名支持的最大字节数（非 **TZ** 环境变量的长度）。
_POSIX_VERSION 操作系统遵守的 POSIX 1 标准（C语言绑定）的版本。

_XOPEN_CRYPT	如果系统支持 X/Open 加密功能组则值为 1。
_XOPEN_ENH_I18N	如果系统支持 X/Open 增强国际化功能组则值为 1。
_XOPEN_SHM	如果系统支持 X/Open 共享内存功能组则值为 1。
_XOPEN_VERSION	操作系统所遵守的 X/Open 可移植性指南的版本。
_XOPEN_XCU_VERSION	操作系统所遵守的 X/Open 命令和实用程序规范的版本。
_XOPEN_XPG2	如果系统支持 1987 年 1 月卷 2 的 X/Open 可移植性指南，XVS 系统调用和库，则值为 1，否则为未定义。
_XOPEN_XPG3	如果系统支持 1992 年 2 月的 X/Open 规范，系统接口和头第三版，则缺省值为 1，否则为未定义。
_XOPEN_XPG4	如果系统支持 1992 年 7 月的 X/Open CAE 规范，系统接口和头第 4 版，则值为 1，否则为未定义。
POSIX2_BC_BASE_MAX	允许的最大值，用于 **obase** 变量并使用 **bc** 命令。
POSIX2_BC_DIM_MAX	**bc** 命令允许数组中的最大元素数。
POSIX2_BC_SCALE_MAX	允许的最大值，用于 **scale** 变量，执行 **bc** 命令。
POSIX2_BC_STRING_MAX	可被 **bc** 命令接受的字符串常量的最大长度。
POSIX2_CHAR_TERM	系统支持至少一个终端类型则值为 1；否则值为 -1。
POSIX2_COLL_WEIGHTS_MAX	配给一个条目的最大权重数目，条目在一个语言环境定义文件中的 **LC_COLLATE** 语言环境变量中。
POSIX2_C_BIND	如果系统支持 POSIX 2 中的 C 语言绑定选项，则值为 1；否则，值为-1。
POSIX2_C_DEV	如果系统支持 POSIX 2 中的 C 语言开发实用程序则值为 1；否则，值为 -1。
POSIX2_C_VERSION	操作系统所遵守的 POSIX 2 标准（C 语言绑定）的版本。
POSIX2_EXPR_NEST_MAX	圆括号内可嵌套的最大表达式数，用于 **expr** 命令。
POSIX2_FORT_DEV	如果系统支持 POSIX 2 中的 FORTRAN 开发实用程序选项则值为 1；否则，值为 -1。
POSIX2_FORT_RUN	如果系统支持 POSIX 2 中的 FORTRAN 运行时实用程序选项则值为 1；否则，值为 -1。
POSIX2_LINE_MAX	当命令被描述成处理文本文件时，一个命令输入行（标准输入或其他文件）的最大字节长度。长度包含换行字符空间。
POSIX2_LOCALEDEF	如果系统支持由 **localedef** 命令创建语言环境，则值为 1；否则值未定义。
POSIX2_RE_DUP_MAX	当使用间隔计数参数时正则表达式所允许重复出现的最大值，例如带 *m* 和 *n* 参数使用 **ed** 命令。
POSIX2_SW_DEV	系统支持软件开发实用程序选项则值为 1；否则，值为 -1。
POSIX2_UPE	如果系统支持 POSIX 2 中用户可移植实用程序选项，则值为 1；否则，值为 -1。
POSIX2_VERSION	系统支持的 POSIX 2 标准的最新版本的批准日期。这个日期是一个六位数字，前四位数字表示年份后两位数字表示月份。POSIX 2 标准的不同版本由 IEEE 标准委员会定期批准，批准日期用于区分不同版本。

系统路径配置变量

PathConfiguration 参数指定了系统路径配置变量，该变量值包含在系统中的路径和路径结构信息。以下的列表定义了这些变量:

_POSIX_CHOWN_RESTRICTED	**chown**() 子例程以适当的特权限制在进程，并且将文件的组标识只更改为进程的有效组标识或辅助组标识之一。如果 *PathName* 参数引用目录，其返回值应用于除在目录中已存在的或可被创建的目录外的所有文件。
_POSIX_NO_TRUNC	路径名长于变量 *NAME_MAX* 指定的限定值就会产生错误。如果 *PathName* 参数引用目录，其返回值应用于目录中的文件名。
_POSIX_VDISABLE	终端特殊字符，定义在 **termios.h** 文件中，可使用该字符值禁用。
LINK_MAX	链接一个文件的最大链接数。如果 *PathName* 参数引用目录，则返回值应用于该目录。
MAX_CANON	在终端规范输入行中的最大字节数。

MAX_INPUT	在终端输入队列有可用空间的最大字节数。
NAME_MAX	文件名中的最大字节数（不包含终止空字符）。如果 *PathName* 参数引用目录，返回值应用于目录内的文件名。
PATH_MAX	路径名中的最大字节数，包含终止空字符。如果 *PathName* 参数引用目录，返回值为当指定目录是工作目录时的相对路径名的最大长度。
PIPE_BUF	当写入管道时保证成为原子的最大字节数。如果这个 *PathName* 参数指向先进先出或是管道，其返回值应用于引用对象。如果 *PathName* 参数引用目录，返回值应用于任何已存在的或可在目录内创建的先进先出。
DISK_PARTITION	磁盘的物理分区大小。 **注**：对于 DISK_PARTITION 路径配置变量，这个 *PathName* 参数必须指定为之查询信息的磁盘的完整路径。
DISK_SIZE	磁盘大小以兆字节计。 **注**：对于 **DISK_SIZE** 路径配置变量，这个 *PathName* 参数必须指定为之查询信息的磁盘的完整路径。

设备变量

DeviceVariable 参数显示了这个 *DeviceName* 参数是一个设备路径，例如 **/dev/hdisk0**。给定一个磁盘的路径，则这个 **getconf** 命令显示这个磁盘的设备名或位置。

DISK_DEVNAME	设备名或设备位置。

退出状态

此命令返回以下出口值:

0 这个指定变量是有效的并且其当前状态的信息已被成功写入。
>0 发生错误。

1. 显示变量 **ARG_MAX** 的值，请输入:

```
getconf ARG_MAX
```

2. 显示所有系统配值变量值，输入:

```
 getconf -a
```

3. 显示变量 **NAME_MAX** 的值，用于 **/usr** 目录，请输入:

```
getconf NAME_MAX /usr
```

4. 以下 shell 命令显示怎样处理未指定结果:

```
if value=$(getconf PATH_MAX /usr)
then
     if [ "$value" = "undefined" ]
     then
               echo
                    The value of PATH_MAX in /usr is undefined.
     else

                    The value of PATH_MAX in /usr is $value.
     fi
else
               echo
                    Error in the getconf command.
fi
```

示例

1. 显示变量 **ARG_MAX** 的值，请输入:

```
getconf ARG_MAX
```

2. 显示变量 **NAME_MAX** 的值，用于 **/usr** 目录，请输入:

```
getconf NAME_MAX /usr
```

3. 以下 shell 命令显示怎样处理未指定结果:

```
if value=$(getconf PATH_MAX /usr)
then     if [ "$value" = "undefined" ]
     then
             echo
                   The value of PATH_MAX in /usr is undefined.
     else
             echo
                   The value of PATH_MAX in /usr is $value.
     fi
else
     echo Error in the getconf command.
fi
```

4. 如果命令:

```
getconf _XBS5_ILP32_OFF32
```

   没有写入 -1\n 或 undefined\n 到标准输出，则表单命令:

```
getconf -v XBS5_ILP32_OFF32 ...
```

   将确定相应于 XBS5_ILP32_OFF32 编译环境的配置变量值。编译环境指定于 **c89**，扩展描述。

5. 如果命令:

```
getconf _XBS5_ILP32_OFFBIG
```

   没有写入 -1\n 或 undefined\n 到标准输出，则表单命令:

```
getconf -v XBS5_ILP32_OFFBIG ...
```

   将确定相应于 XBS5_ILP32_OFFBIG 编译环境的配置变量值，编译环境值指定于 **c89**，扩展描述。

6. 如果命令:

```
getconf _XBS5_LP64_OFF64
```

   没有写入 -1\n 或 undefined\n 到标准输出，则表单命令:

```
getconf -v XBS5_LP64_OFF64 ...
```

   将确定相应于 XBS5_LP64_OFF64 编译环境的配置变量值，编译环境值指定于 c89，扩展描述。

7. 如果命令:

```
getconf _XBS5_LPBIG_OFFBIG
```

   没有写入 -1\n 或 undefined\n 到标准输出，则如下形式的命令:

```
getconf -v _XBS5_LPBIG_OFFBIG
```

   将确定相应于在 c89，扩展描述中指定的 XBS5_LPBIG_OFFBIG 编译环境的配值变量值。

8. 确定磁盘 hdisk0 大小，若是 root 用户，则输入以下:

```
getconf DISK_SIZE /dev/hdisk0
```

9. 确定实际内存大小，输入以下:

```
getconf REAL_MEMORY
```

10. 确定是否机器硬件是 32 位或 64 位，则输入以下:

```
getconf HARDWARE_BITMODE
```

11. 确定是否内核是 32 位或 64 位，则输入以下:

```
getconf KERNEL_BITMODE
```

12. 确定磁盘 hdisk0 的设备名或位置，则输入以下:

```
getconf DEVICE_NAME /dev/hdisk0
```

文件

/usr/bin/getconf	包含一个 **getconf** 命令。
/usr/include/limits.h	定义系统配置变量。
/usr/include/unistd.h	定义系统配置变量。

相关信息

confstr 子例程，**pathconf** 子例程，**sysconf** 子例程。

《*操作系统与设备管理*》中的『命令』。

---

getdev 命令

用途

列出符合指定标准的设备。

语法

getdev [**-a**] [**-e**] [*Criteria*] [*DeviceList*]

描述

列出匹配给定标准的设备。标准以表达式的形式给出。**getdev** 命令可以检查系统上所有的设备或指定列表中的设备。

标志

-a	指定设备必须符合由该命令生成的列表中包括的所有标准。如果没有定义标准，则 **-a** 标志没有什么影响。
-e	指定*设备列表*中所列的设备不能包括在由 **getdev** 命令生成的列表中。没有*-e* 标志，只生成*设备列表*中所列的设备。如果没有指定设备，该标志被忽略。

参数

Criteria 在设备被包括在所生成的列表中之前，定义设备必须符合的标准。*标准*可以被指定为一个表达式或一个表达式列表，由 **getdev** 命令生成的列表中所包括的设备必须符合这些表达式。如果没有提供标准，所有设备都将被列入列表中。

设备必须至少满足此列表中的一个标准。然而，可以使用选项 **-a** 来指定应该执行“逻辑与”运算。然后，只有那些符合列表中列出的所有标准的设备才能被包括进来。

Criteria 参数中指定的标准可以有四种可能的表达式类型。

属性=*值*
: 引出所有带成员的设备。这些设备已经被定义了*属性*，其属性与*值*相等。

属性!=*值*
: 引出所有带成员的设备。这些设备已经被定义了*属性*，其属性与*值*不等。

属性:* 引出所有定义了*属性*的、带成员的设备。

属性!:* 引出所有未定义*属性*的、带成员的设备。

以下是有效的设备属性:

别名 设备名称。

描述 描述设备。

类型 描述设备类型的标记。
type 属性的值的有效组可通过执行以下命令获得。**odmget PdDv | grep -w class | awk '{print $3}' | sed 's/"//g' | sort | uniq**

状态 设备的当前状态。
状态的可能值的列表是:
1.Defined
2.Available
3.Stopped
4.Diagnose
status 的值是不区分大小写的。

DeviceList 指定空格分隔的设备列表以便于*标准*检查。

退出状态

0 成功完成命令。

> 1 发生错误。

示例

1. 要显示所有设备，请输入:
   ```
   getdev
   ```
2. 要列出类型为“lvtype”的设备，请输入:
   ```
   getdev type=logical_volume
   ```
3. 要列出类型不是“logical_volume”的设备，请输入:
   ```
   getdev type!=logical_volume
   ```
4. 要列出类型为“logical_volume”或其设备别名为“sys0”的设备，请输入:
   ```
   getdev type=logical_volume alias=sys0
   ```

其输出看似如下:

```
hd1
hd2
hd3
hd4
...
sys0
```

5. 要列出类型为 “logical_volume” 且其设备别名为 “lv01” 的设备，请输入:

   ```
   getdev -a type=logical_volume alias=lv01
   ```

6. 要显示定义了**status**属性的设备，请输入:

   ```
   getdev status:*
   ```

7. 要显示 **desc** 属性尚未定义的设备，请输入:

   ```
   getdev desc!:*
   ```

文件

/usr/sbin/getdev	包含 **getdev** 命令

相关信息

getdgrp 命令、**lsdev** 命令。

---

getdgrp 命令

用途

罗列与指定标准相匹配的设备类。

语法

getdgrp [**-a**] [**-e**] [**-l**][*Criteria*] [*DeviceClassList*]

描述

列出包含与给定条件匹配的设备的设备类。该条件以表达式的形式给出。

标志

-a	显示设备必须符合所有包含在由此命令生成的报告中的设备类标准。如果没有定义标准，则 **-a** 标志无效。
-e	显示应该将参数列表中指定的设备类排除在由此命令生成的报告之外。如果没有指定设备，则 **-e** 标志无效。
-l	显示选项 **-e** 和列表 **dgroup** 中的所有设备类，即使不包含有效的设备成员，也应该罗列出来。如果命令行指定了*Criteria*，则此选项无效。

参数

Criteria 可以将设备所属的设备类包含在生成的列表之前，必须定义它所匹配的标准。可以将*标准*指定为一个表达式或一个表达式列表，设备类如要包含在由 **getdgrp** 生成的列表中，则其设备必须符合这些表达式。如果没有给出标准，则此列表包含所有的设备类。

设备必须至少满足此列表中的一个标准。然而，可以使用选项 **-a** 来指定应该执行“逻辑与”运算。因此，只包含那些包含符合列表当中所有标准的设备的类。

*标准*参数中指定的标准可以有四种可能的表达式类型。

属性=*值*
将其成员已定义 *属性*，并且等于 *值*的所有设类备提出。

属性!=*值*
将其成员已定义*属性*，并且不等于*值*的所有设备类提出。

属性:* 将其成员已定义*属性*的所有设备类提出。

属性!:* 将其成员未定义*属性*的所有设备类提出。

以下是有效的设备属性:

别名 设备名称。

描述 描述设备。

类型 描述设备类型的标记。

状态 设备的当前状态。
状态的可能值的列表是:
1.Defined
2.Available
3.Stopped
4.Diagnose
status 的值是不区分大小写的。

DeviceClassList 在定制的设备配置数据库中或者预定义的设备配置数据库中指定设备类名。

退出状态

0 成功完成命令。

1 命令语法不正确，使用了无效选项，或出现内部错误。

2 定制的设备对象类或预定义设备对象类不能打开阅读。

示例

1. 要显示所有设备类，请输入:

```
getdgrp
```

输出类似以下格式:

```
adapter aio
bus
cdrom
disk
diskette
gxme
if
keyboard
```

```
lft
logical_volume
lvm
memory
mouse
planar
processor pty
pwrmgt
rcm
bin
tape
tcpip
tty
```

2. 要列出其设备类型为“logical_volume”的设备类，请输入:

```
getdgrp type=logical_volume
```

输出类似于以下内容:

```
logical_volume
```

3. 要列出其设备类型为“lvtype”或其设备别名为“sys0”的设备类，请输入:

```
getdgrp type=logical_volume alias=sys0
```

输出类似于以下内容:

```
logical_volume
sys
```

4. 要列出其状态属性已定义的设备类，请输入:

```
getdgrp status=defined
```

输出类似于以下内容:

```
logical_volume
posix_aio
rcm
```

5. 要显示设备类中定义了 **status** 属性的且属于“processor”设备类的设备，请输入:

```
getdgrp status:* processor
```

输出类似于以下内容:

```
processor
```

6. 要显示其中设备没有定义 **status** 属性的设备类，请输入:

```
getdgrp status!:* processor
```

文件

/usr/sbin/getdgrp	包含 **getdgrp** 命令

相关信息

getdev 命令、**lsdev** 命令。

getea 命令

用途

从文件中获取指定的扩展属性。

语法

getea [**-n** *Name*] [**-l**] [**-e** *RegExp*] [**-s**] *FileName*

描述

getea 命令从文件中读取指定的扩展属性。如果指定了 **-n** *Name* 参数，则仅获取与 *Name* 匹配的扩展属性。

注: 为防止命名冲突，JFS2 保留 8 字符的前缀 (0xf8)SYSTEM(0xF8)，用于系统定义的扩展属性。请避免将此前缀用于命名用户定义的扩展属性。

如果指定了 **-e** *RegExp* 参数，则仅获取与正则表达式 *RegExp* 匹配的扩展属性。如果 **-n** 或 **-e** 标志都未指定，则获取所有扩展属性。

该命令没有用来获取 ACL。**aclget** 命令用于获取 ACL。

标志

-e *RegExp*	指定正则表达式以获取所有匹配的扩展属性。值以字符格式显示。
-l	指定从符号链接本身而不是从它指向的文件获取扩展属性。
-n *Name*	指定要获取的特定扩展属性的名称。值以字符格式显示。
-s	仅显示扩展属性的名称而不显示值。
FileName	指定读取扩展属性的文件。

退出状态

0	成功完成。
正整数	发生错误。

示例

1. 要检索文件 design.html 的所有指定扩展属性，请输入:

   ```
   getea design.html
   ```

2. 要获取文件 design.html 的指定扩展属性 Approver，请输入:

   ```
   getea -n Approver design.html
   ```

3. 要只获取文件 design.html 的所有指定扩展属性的名称，请输入:

   ```
   getea -s design.html
   ```

4. 要获取符号链接 design.html 的所有指定扩展属性，请输入:

   ```
   getea -l design.html
   ```

位置

/usr/sbin

相关信息

《*AIX 5L V5.3 命令参考大全，卷 1*》中的 **chfs** 和 **crfs** 命令。

《*AIX 5L V5.3 命令参考大全，卷 5*》中的 **setea** 命令。

---

getopt 命令

用途

分析命令行标志和参数。

语法

getopt*Format Tokens*

描述

getopt 命令对一列使用指定预期标志和参数的格式的记号进行分析。标志是一个单一的 ASCII 字母，当其后跟有冒号时，预期会有一个参数，可能或可能不用一个或多个制表符或空格将此参数和标志分开。参数中可以包括多字节字符，但是不能作为标志字母。

当 **getopt** 命令读取完所有记号，或者当它遇到特殊标记 —（双连字符）时即完成处理。然后，**getopt** 命令输出处理过的标志，—（双连字号）和任何其余标记。

如果标记不能与标志相匹配，**getopt** 命令将会对标准错误写出一条消息。

示例

可以在程序框架的 shell 脚本中使用 **getopt** 命令来分析选项，如以下示例所示:

```
#!/usr/bin/bsh
# parse command line into arguments
set -- `getopt a:bc $*`
# check result of parsing
if [ $? != 0 ]
then
        exit 1
fi
while [ $1 != -- ]
do
        case $1 in
        -a)     # set up the -a flag
                AFLG=1
                AARG=$2
                shift;;
        -b)     # set up the -b flag
                BFLG=1;;
        -c)     # set up the -c flag
                CFLG=1;;
        esac
        shift   # next flag
done
shift   # skip --
# now do the work
.
.
.
```

注: 在 C shell 中，请使用以下命令来运行 **getopt** 命令:

```
set argv=`getopt OptionString $*`
```

在下面的每个示例中，**getopt** 命令应该以同样的方式处理标志和参数:

- `-a ARG -b -c`
- `-a ARG -bc`
- `-aARG -b -c`
- `-b -c -a ARG`

文件

/usr/bin/getopt 包含 **getopt** 命令。

相关信息

bsh 命令、**csh** 命令。

getopt 子例程。

《*操作系统与设备管理*》中的『Shell』。

---

getopts 命令

用途

处理命令行参数，并校验有效选项。

语法

getopts *OptionString Name* [*Argument* ...]

描述

getopts 命令是 Korn/POSIX shell 的内置命令，用来从参数列表检索选项以及选项参数。选项由一个**+**（加号）或者是由一个**-**（减号）后跟一个字符开始。一个既不是以**+**，也不是以**-**开始的选项结束*选项字符串*。每次调用 **getopts** 命令时，它将下一个选项的值放置在*名称* 内，并将下一个要处理的参数的索引置于 shell 变量 **OPTIND** 中。一旦调用了 shell， **OPTIND** 将初始化为 1。当选项以 **+**开头，则**+** 将预先设为*名称*中的值。

如果*选项字符串*中的字符后面带有“**:**”（冒号），则预期此选项将带有参数。当选项需要选项参数时，**getopts** 命令就将其置于变量 **OPTARG** 中。

当查找到 *选项字符串*所不包含的选项字符，或者查找到的选项没有所需的选项参数时:

- 如果*选项字符串*不以**:**（冒号）开头，
  - *名称* 将会被设置为 **?** （问号）字符，
  - **OPTARG**. 将被取消设置，并且
  - 诊断消息将被写入到标准错误中。

这种情况被认为是在将参数传递给所调用的应用程序的过程中所检测到的错误，而不是在处理 **getopts** 命令的过程中所发生的错误；如上所述，写入诊断消息，但退出状态将变为零。

- 如果*选项字符串*以 **:**（冒号）开头，

– *名称* 将被设为 **?** （问号）字符，这是对未知的选项来说的，或者为缺少的所需选项设为**:**（冒号）字符，
– **OPTARG** 将被设置为已查找到的选项字符，并且
– 标准错误中将不写入任何输出。

以下任何字符都可以识别选项结尾：特殊选项**- -**，查找到不以**-**，或者**+**为开头的参数，或者遇到错误。

当遇到选项结尾时:

- **getopts** 命令将退出运行，并且返回值大于零，
- **OPTARG** 将被设置为第一个非选项参数索引，在这种情况下，如果第一个 **- -** 参数之前未出现其他非选项参数，将认为它是选项参数，或者如果没有非选项参数，设置为值 $#+1，
- *名称*将被设置为**?** （问号）字符，

参数

OptionString 包含 **getopts** 命令识别的选项字符串。如果字符后带有冒号，则预期选项将带有参数，应该以单独参数的形式提供此参数。可以用空格将选项与参数分隔开。如果选项字符是未知的或者选项参数丢失，则*选项字符串*中的第一个字符将决定 **getopts** 命令的行为。

注：应用程序不应该将问号和冒号字符作为选项字符。使用其他非字母数字的字符会产生不明的结果。

Name 由 **getopts** 命令对查找到的选项字符设置。

Argument ... 一个或多个被空格分隔的字符串，由 **getopts** 命令校验是否是合法选项。如果省略参数 ，就使用位置参数。有关位置参数的更多信息，请参阅 Korn Shell 中的『Korn shell 或 POSIX shell 中的参数替换』。

注： 一般来说，不必将参数指定为 **getopts** 命令的一部分，但在进行脚本调试时可能会有所帮助。

退出状态

此命令返回以下出口值:

0 查找到由*选项字符串*指定的或未指定的选项。
>0 遇到选项结束或发生错误。

示例

1. 以下 **getopts** 命令规定 a、 b 和c 为有效选项，并且选项 a 和 c 带有参数:

```
getopts a:bc: OPT
```

2. 以下 **getopts** 命令指定 a、b 以及 c 为有效选项， 并且选项 a 和 b 带有参数，而且 **getopts** 在命令行遇到未定义的选项时，它将 OPT 的值设置为 **?**:

```
getopts :a:b:c OPT
```

3. 以下脚本分析和显示其参数:

```
aflag=
bflag=

while getopts ab: name
do
            case $name in
            a)     aflag=1;;
            b)     bflag=1
                          bval="$OPTARG";;
            ?)     printf"Usage: %s: [-a] [-b value] args\n" $0
```

```
                        exit 2;;
            esac
done

if [ ! -z "$aflag" ]; then
            printf "Option -a specified\ n"
fi

if [ ! -z "$bflag" ]; then
            printf'Option -b "%s" specified\ n' "$bval"
fi

shift $(($OPTIND -1))
printf "Remaining arguments are: %s\n" "$*"
```

相关信息

《*操作系统与设备管理*》中的『Korn shell 或 POSIX shell 命令』。

---

gettable 命令

用途

从主机获取网络信息中心格式主机表。

语法

/usr/sbin/gettable [**-v**] *Host* [*OutFile*]

描述

使用 **/usr/sbin/gettable** 命令从由*主机*参数标明的服务器中获取 NIC 标准主机表。如果检索到，这些表被放置在由 *OutFile* 参数标明的文件中。

gettable 命令打开到服务规范中所标明的端口的传输控制协议（TCP）连接以获取*主机*参数。紧接着请求所有的名称，并且将生成的信息放置在输出文件当中。

gettable 命令最好与 **htable** 命令一同使用，因为后者可以将 NIC 标准文件格式转化为网络库查找例程所使用的格式。

标志

-v 只获取版本号而不是完整的主机表，并且将其输出到 *OutFile* 中，或者在缺省的情况下，输出到命名为 **hosts.ver** 的文件中.

参数

Host 指定提供主机表信息的服务器。
OutFile 指定想要存放主机表信息的文件。如果使用 **gettable** 命令而不带**-v** 标志，则缺省的文件名是 **hosts.txt**。

相关信息

htable 命令。

《*网络与通信管理*》中的『传输控制协议（TCP）』和『TCP/IP 协议』。

---

gettrc 命令

用途

管理跟踪文件的收集。

语法

gettrc [**-c**] [**-C** *dirname*] [**-m**] [**-M** *dirname*] [**-s**] [**-S** *dirname*]

描述

gettrc 命令是与 **snap** 命令结合使用的脚本。它管理系统跟踪文件、轻量级内存跟踪（LMT）文件和组件跟踪（CT）文件的收集。

标志

-c	收集组件跟踪文件。
-C *dirname*	从 *dirname* 指定的目录中收集组件跟踪文件。
-m	收集内存跟踪文件。
-M *dirname*	从 *dirname* 指定的目录中收集轻量级内存跟踪文件。
-s	收集系统跟踪文件。
-S *dirname*	从 *dirname* 指定的目录中收集系统跟踪文件。

退出状态

0	命令成功完成。
>0	发生错误。

示例

1. 要将 **gettrc** 与 **snap** 命令结合使用，以检索不同类型的跟踪文件，请输入:

   ```
   snap "gettrc -c -C dirname -m -M dirname -s -S dirname"
   ```

   这将返回系统跟踪文件、LMT 文件和 CT 文件，包括那些在 *dirname* 所指定的目录中列出的文件。

位置

/usr/lib/ras/snapscripts/gettrc

文件

/usr/lib/ras/cpufmt
/etc/trcfmt

相关信息

snap 命令。

---

getty 命令

用途

设置端口特征。

语法

getty [[**-r** | **-u** | **-U**] [**-d**] [**-H** *HeraldString*] [**-M** *motdFile*] [**-N**]] *PortName*

描述

getty 命令设置和管理终端线路和端口。**getty** 命令由 **init** 命令来运行。**getty** 命令与终端状态管理员程序相链接。终端状态管理员程序提供了终端控制和登录的复合功能。

注: **getty** 命令不在命令行输入。

当作为 **getty** 命令调用时，终端状态管理员程序提供了通常的端口管理功能，这包括 ：

双向使用	允许使用终端线路用来启动和接受连接。
线路速度	设置发送和接收的波特率。
奇偶性	将奇偶性设置为偶数、奇数或无。
延迟	设置回车、制表符、换行以及换页延迟。
字符集映射	为大小写、制表符和换行控制设置字符集映射。
登录器程序	指定用于使用户登录到系统的程序。如果设置了此属性，则禁用安全注意密钥（SAK）的处理功能。如果没有设置此属性，则其缺省值为 **/usr/sbin/login** 。**logger**属性包含在对象数据管理员（ODM）数据库中。
字符和行擦除	设置用于字符和行擦除的击键。
回应方式	设置本地或远程回应。

当调用 **getty** 命令时，会发生以下步骤:

1. 根据 ODM 数据库中的**所有者**和**保护**属性来设置端口保护。如果没有指定这些属性，则其缺省值为 root 用户和 622。
2. 打开由*端口名*参数指定的端口。如果在端口可以进行载波检测，则直到载波出现或其他的进程已丢失此端口的载波，打开才完成。
3. 指定的端口可能会被锁定。如果 **getty** 命令带 **-u** 或 **-r** 标志运行，它会试图锁定端口。如果锁定了端口，此命令将一直等待端口可用为止，继而退出。如果指定了 **-r** 标志，**getty** 命令将等待从端口接收一个字节的数据，然后继续。
4. 根据所指定端口的配置信息设置终端属性。此时能否启用安全注意密钥处理就在于系统的配置如何。
5. 载波消息被写入到指定的端口。
6. 从指定端口读取登录名。如果发生帧错误或中断，则 **getty** 命令将使用下一组配置的终端属性重复第四和第五步。这是循环调制解调器波特率最常使用的方法。但是可以通过在 ODM 数据库中输入以逗号隔开的值来循环任何 ODM 字段（除了 `logmodes`和 `runmodes`）。

7. 按照 runmodes 参数和登录名称复位终端方式。如果登录名称由新行终止，则 **getty** 命令打开回车符到新行的映射。如果所有的字母字符都是大写，如果可能的话，提示用户使用小写字符登录，并打开从小写到大写的映射。
8. 如果程序由登录器参数指定，则会执行它且禁用安全注意密钥处理。否则，“终端状态管理器”程序执行标准系统登录。

注: 如果用户在登录时输入了 Sequence Attention Key 序列，则用户登录的是可信的 shell（如果系统经配置，端口安全可信，允许用户从可信路径登录）。

标志

-d	提供调试信息。
-H *HeraldString*	指定备用预告消息以在端口写入提示的登录名。消息字符串必须是一个字而且不能包含任何空格。该字符串将优先于在 **/etc/security/login.cfg** 文件中定义的预告消息。如果没有字符串以该选项或在 **login.cfg** 文件中指定，则使用来自于消息编目的缺省预告。
-M *motdFile*	指定到每日文件的备用消息的路径。如果没有指定，在缺省情况下该值为 **/etc/motd**。
-N	使 **getty** 绕过在 **/etc/utmp** 文件中任何进程标识的检查。这允许不同于最低登录 shell 的进程来**执行 getty**。
-r	将端口设为可共享（双向）使用。如果锁定不成功，则 **getty** 命令一直等待至锁可用为止，然后退出。如果锁定成功， **getty** 命令将在锁定端口后等待从端口接收一字节的数据。
-u	将端口设为可共享（双向）使用。如果锁定不成功，**getty** 命令将一直等待锁定为止，然后退出。
-U	除了 **getty** 不会等待锁可用之外，与 **-u** 标志相同。这使得端口可用，而无需顾虑锁。

安全性

访问控制: 应该如可信计算库中的程序一样安装本程序，任何用户和到根的 **setuid** 都可执行。

示例

如果想要能够登录 tty0， 请将下一行添加到 **/etc/inittab** 文件中:

```
tty0:2:respawn: /usr/sbin/getty /dev/tty0
```

此命令初始化端口/dev/tty0 并设置端口特征。

文件

/usr/sbin/getty	包含 **getty** 命令。
/etc/locks	包含阻止多重使用通信设备和多重调用远程系统的锁文件。
/usr/sbin/login	**login** 命令。
/etc/security/login.cfg	包含端口登录配置。
/etc/motd	包含登录后显示的日消息。
/usr/bin/setmaps	**setmaps** 命令。
/etc/utmp	包含有关用户登录系统的信息。

相关信息

login 命令、**setgroups** 命令、**shell** 命令、**su** 命令、**telinit or init** 命令、**tsm** 命令。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『程序员 – 对象数据管理器（ODM）概述』。

glbd 守护程序

用途

管理全局位置代理数据库。

语法

/etc/ncs/glbd [**-create** { **-first** [**-family** *FamilyName*] | **-from** *HostName* }] [**-change_family** *FamilyName*] [**-listen** *FamilyList*] [**-version**]

描述

glbd 守护程序用于管理全局位置代理（GLB）数据库。GLB 数据库是“网络计算系统（NCS）”的一部分，它帮助客户机定位网络或因特网上的服务器。GLB 数据库可以存储运行进程时所在的服务器的位置（特别的，网络地址和端口号）。**glbd** 守护程序负责维护此数据库并提供对它的访问。

有两个版本的 GLB 守护程序，即 **glbd** 和 **nrglbd**。

您可以复制 GLB 数据库来增强它的可用性。这种数据库副本可存在于多个主机上，在每个这样的主机上都将运行有 **glbd** 守护程序以维护数据库副本的一致性。（在因特网上，每个网络中至少要运行一个 **glbd** 守护程序。）每个 GLB 副本都会保留所有其他 GLB 副本的列表。**drm_admin** 工具负责管理 GLB 数据库的复制和副本列表的复制。

目前 **glbd** 可支持 DARPA IP 和域 DDS 网络协议。GLB 副本允许从 IP 和 DDS 客户机访问其数据库。但是，当 GLB 副本进行相互通信以维护 GLB 数据库的副本时，它们只能使用一个协议系列。您可以选择 GLB 使用的协议系列。在因特网中，所有的路由节点都必须支持该系列。

glbd 守护程序可以用两种方式之中的任意一种启动:

- 通过系统资源控制器（推荐使用此方法）。方法是在命令行中输入以下内容:

  ```
  startsrc -s glbd
  ```

- 由具有 root 用户权限的人员在命令行中输入以下内容:

  ```
  /etc/ncs/glbd &
  ```

在启动 **glbd** 守护程序之前，必须在系统上已配置并运行了 TCP/IP。另外，在启动 **glbd** 守护程序之前还必须启动并运行 **llbd** 守护程序。

标志

-create
创建 GLB 副本。此选项可创建 GLB 数据库并启动代理进程。它必须连同 -first 或 -from 使用。

-first
创建网络或因特网上的首个 GLB 副本（即第一个实例）。此选项只能连同 **-create** 选项使用。

-family *FamilyName*
指定首个 GLB 副本将用来在副本列表中标识它自己的地址系列。此选项只能连同 **-first** 选项使用。随后创建的任何副本都必须使用此系列同该副本通信。目前*FamilyName* 可能是 **dds** 或 **ip**。如果未使用此选项，在副本列表上将按照此副本的 DDS 地址来标识它。

-from *主机名*
创建其他的 GLB 副本。此选项可以连同 **-create** 选项使用。在*HostName* 指明的主机上必须已有 GLB 副本存在。新副本的数据库和副本列表将从 *HostName* 上的列表得到初始化。*HostName* 会在它的副本列表中添加关于新副本的项并将此项传播给其他的 GLB 副本。

HostName 以 family:host 的形式指定，其中，host 可以按照主机名称或主机的网络地址来指定。例如，`ip:jeeves`、`ip:bertie` 和 `ip:#192.5.5.5` 都是可接受的主机名。

新副本使用与 *HostName* 相同的地址系列在副本列表中标识自己。例如，如果 *HostName* 为 IP 地址，则新副本也将按照其 IP 地址列在副本列表中。

-change_family *FamilyName*
更改各个 GLB 副本的地址系列。请仅在网络重新配置要求您作这样的更改时才使用此选项。*FamilyName* 目前可以是 **dds** 或 **ip**。

-listen *FamilyList*
限制 GLB 侦听的地址系列。此选项仅在创建对 GLB 的访问被限制为网络或因特网中的主机子集的特殊配置时才使用。

FamilyList 是 GLB 将要侦听的地址系列的列表。列表中的名称要用空格分隔。可能的系列名称包括 **dds** 和 **ip**。

GLB 将总是侦听将其列在副本列表所使用的系列的请求，即使此系列未在 *FamilyList* 中指定。

如果在启动 **glbd** 时未带 **-listen** 选项，GLB 将侦听 NCS 和本地主机所支持的所有地址系列。在 Apollo 系统中，此系列集始终包含 **dds**，还可能包含 **ip**。在其他大多数系统中，**ip** 通常是唯一的系列。

-version
显示 **glbd** 所属的 NCS 版本，但不启动该守护程序。

文件

/etc/ncs/glb_log
包含来自 **glbd** 的诊断输出。

/etc/rc.ncs
包含启动 NCS 守护程序的命令。

示例

1. 创建并首次启动网络或因特网上的首个 GLB 副本:

   ```
   /etc/ncs/glbd -create -first -family ip &
   ```

2. 首次启动 GLB 的后续副本并从主机 jeeves 对其数据库初始化:

   ```
   /etc/ncs/glbd -create -from ip:jeeves &
   ```

3. 重新启动现有的 GLB 副本:

   ```
   /etc/ncs/glbd &
   ```

相关信息

drm_admin 命令、**lb_admin** 命令和 **startsrc** 命令。

llbd 守护程序。

---

gprof 命令

用途

显示调用图概要分析数据

语法

/usr/ccs/bin/gprof [**-b**] [**-c** [*filename*]] [**-e** *Name*] [**-E** *Name*] [**-f** *Name*] [**-g** *filename*] [**-i** *filename*] [**-p** *filename*] [**-F** *Name*] [**-L** *PathName*] [**-s**] [**-x** [*filename*]] [**-z**] [**a.out** [**gmon.out** ...]]

描述

gprof 命令可生成 C、Pascal、FORTRAN 或 COBOL 程序的执行概要文件。被调用的例程的结果合并到各个调用程序的概要文件中。**gprof** 命令有助于标识程序如何消耗 CPU 资源量。要了解程序的哪些功能（例程）正在使用 CPU，您可以用 **gprof** 命令对该程序进行概要分析。

概要数据可从调用图概要文件（缺省状态下为 **gmon.out**）中获取，这些程序是由使用 **-pg** 选项的 **cc** 命令编译的程序创建的。**-pg** 选项链接在为概要分析而编译的各种版本的库例程中，且它读取指定的对象文件（缺省情况下为 **a.out**）中的符号表，并将该符号表与调用图概要文件相关联。如果指定了多个概要文件，**gprof** 命令的输出显示给定概要文件中的概要信息的总结。

-pg 选项可使编译器在为程序中每个重新编译的函数生成的对象代码中插入对 **mcount** 子例程的调用。在程序执行期间，每当父函数调用子函数时，子函数都会调用 **mcount** 子例程来使用于该父-子函数对的独特计数器加 1。在不是使用 **-pg** 选项重新编译的程序中不会插入 **mcount** 子例程，因此也就不会保持是谁调用它们的记录。

注: 来自 C++ 对象文件的符号在使用之前将其名称解码。

GPROF 环境变量可以用来为概要分析设置不同选项。该环境变量的语法定义如下:

```
GPROF = profile:<profile-type>,scale:<scaling-factor>,file:<file-type>,filename:<filename>
```

此处:

- <profile-type> 描述需要执行什么类型的概要分析，可以是进程也可以是线程。输入 “process” 表示概要分析详细程度处于进程级别，输入 “thread” 表示概要分析详细程度处于线程级别。

- <scaling-factor> 描述需要分配多少内存给调用图表概要文件，在缺省情况下，进程级别概要分析的比例因子为 2 而线程级别概要分析的比例因子为 8。比例因子为 2 表示每个进程或线程分配了进程大小的 1/2 内存，比例因子为 8 表示每个进程或线程分配了进程大小的 1/8 内存。该内存是存储调用图表信息的缓冲区。
- <file-type> 描述需要什么类型的 **gmon.out** 文件，值 multithread 表示每个进程需要一个 gmon.out 文件，值 multithread 表示每个线程需要一个 **gmon.out** 文件。如果使用 -pg 选项概要分析应用程序且该应用程序进行了派生，则指定 multi 为父进程生成一个 **gmon.out** 文件，再为子进程生成另一个。生成的 **gmon.out** 文件的命名约定如下:
  - 对于 multi 文件类型: <prefix>-processname-pid.out
  - 对于多线程文件类型: <prefix>-processname-pid-Pthread<threadid>.out

  <prefix> 的缺省值为 **gmon**。用户可以使用 GPROF 环境变量的 filename 参数定义自己的前缀。
- <filename> 描述生成的 **gmon.out** 文件需要使用的前缀。在缺省情况下，该前缀为 **gmon**。

注: 指定 `profile:thread` 生成格式 **gmon.out** 文件，该文件只能用 AIX 5.3 gprof 读取。如果用户想要旧格式的 **gmon.out** 文件并仍旧想指定 **profile:thread**，则您必须指定 **file:multithread**，这将为每个线程生成旧格式的 **gmon.out** 文件。因此，如果您的应用程序具有两个线程，则将生成两个 **gmon.out** 文件，一个线程一个，并使用如前所述的命名约定。您无法通过在 AIX 5.2 或更早版本中使用 **-pg** 编译应用程序并在 AIX 5.3 上运行该程序来启用线程级别概要分析。要启用线程级别概要分析，您必须在 AIX 5.3 中用 **-pg** 编译该应用程序。

gprof 命令将得到三项结果:

1. 首先会产生类似于 **prof** 命令提供的平面概要文件。此列表按照递减的时间顺序给出了程序中每个函数的总执行时间和调用计数。这些时间随后沿调用图边界传播。由此可以找到执行周期，从而使处于同一周期中的调用共享该周期的时间。
2. 第二个列表显示按照各自存在的时间排序的函数，包括其调用图表后代的时间。在每个功能项的下方列出了其（直接）调用图的子代，并附有如何将它们的时间传播到此功能的指示。在此函数的上方类似地显示了如何将此函数及其后代的时间传播给它的（直接）调用图父代。
3. 同时还显示周期，包括整个周期项和该周期的成员列表以及这些成员在该周期中占用的时间和调用次数。

注: 如果对 **gprof** 的输入包含线程级别概要分析数据（AIX 5.3 格式 **gmon.out** 文件），则 **gprof** 命令为每个线程生成以上三项，以累积报告开始，随后是每个线程的报告（以线程标识升序排列）。

grpof 命令还可以用来分析远程机器上程序的执行概要文件。这可以通过运行 **gprof** 命令并带有 **-c** 选项（该选项在调用图概要文件上，调用图概要文件缺省情况下为 **gmon.out**）以生成一个文件（缺省情况下为 **gprof.remote**），然后在远程机器上处理该文件来实现。如果使用除 **gmon.out** 以外的调用图概要文件，则应该在 **-c** *Filename* 和可执行文件名称后指定调用图概要文件名称。如果 **GPROF** 环境变量的 **file** 属性被设置为 **multi**; 创建了多个 gmon.out 文件，执行程序分叉时针对每个 PID 有一个 gmon.out 文件，则必须指定 *Filename*。可以在远程机器上使用 **-x** 选项来处理 **gprof.remote**（在缺省情况下）文件以生成概要文件报告。

使用 fork 和 exec 子例程时的概要分析

如果程序在多个并发的进程中运行了 **fork** 或 **exec** 子例程时，使用 **gprof** 命令进行概要分析会存在一些问题。概要是每个进程的环境属性，因此，如果对其进行概要分析的进程又派生出新进程，则此子进程也会被列入分析的范畴。然而，这样一来两个进程都向运行父进程的目录中的 **gmon.out** 文件写入，从而导致它们其中的写入内容被覆盖。因此，对于多进程概要分析，推荐您使用 **tprof** 命令。在 AIX 5.3 中，您可以使用 **file:mutli** 避免破坏父进程的 **gmon.out** 文件，**file:multi** 使用 AIX 5.3 命名约定生成 **gmon.out** 文件，因此子进程 **gmon.out** 文件不会具有与父进程相同的名称，这可以避免覆盖。

对于 AIX 5.3 以前的版本：如果您必须使用 **gprof** 命令，绕开该问题的一种方法是调用 **chdir** 子例程来更改子进程的当前目录。这样一来，当子进程退出时，其 **gmon.out** 文件就会被写入到新的目录。以下示例演示了这种方法:

```
cd /u/test   # current directory containing forker.c program
pg forker.c
main()
{
int i, pid;
static char path[]="/u/test2";
pid=fork();             /* fork a child process */
if(pid==0) {            /* Ok, this is the child process */
   chdir (path);        /* create new home directory so
                           gmon.out isn't clobbered! */
   for (i=0; i<30000; i++) sub2();  /* 30000 calls to sub2
                                       in child profile */
   }
else                   /* Parent process... leave gmon.out
                          in current directory */
   for (i=0;i<1000; i++) sub1(pid);   /* 1000 calls to sub1
                                         in parent profile */
}
int sub1(pid)   /* silly little function #1, called
                   by parent 1000 times */
int pid;
{
int i;
printf("I'm the parent, child pid is %i.\n",pid);
}
int sub2()     /* silly little function #2, called
                  by child 30,000 times */
{
printf("I'm the child.\n");
}
cc -pg forker.c -o forker   # compile the program
mkdir /u/test2              # create a directory for childi
                              to write gmon.out in
forker >/dev/null           # Throw away forker's many,
                              useless output lines
gprof forker   >parent.out  # Parent process's gmon.out is
                              in current directory
gprof forker ../test2/gmon.out >child.out
                            # Child's gmon.out is in test2
                              directory
```

此时，如果将 test 目录中的 parent.out 和 child.out 这两个 **gprof** 命令的输出列表进行比较，您可以看到 sub1 子例程在父进程中调用了 1,000 次，在子进程中调用了 0 次，而 sub2 子例程在子进程中调用了 30,000 次，在父进程中调用了 0 次。

运行 **exec** 子例程的进程不继承概要分析。然而，如果由 **exec** 子例程执行的程序是使用 **-pg** 选项编译的，则此程序应进行概要分析。就拿前面的 forker.c 示例来说，如果同时对父进程和由 **exec** 子例程程序运行的程序进行概要分析，则其中一个会覆盖另外一个的 **gmon.out** 文件，除非对其中之一中使用 **chdir** 子例程。

在没有源代码的情况下进行概要分析

如果没有程序的源代码，您可以在不重新编译的情况下使用 **gprof** 命令进行概要分析。然而，您必须能通过相应的编译程序命令（例如，用于 C 语言的 **cc** 命令）重新链接程序模块。如果不作重新编译，您将无法获得调用频率计数，尽管平面概要文件在没有它们的情况下仍然有用。作为额外的补偿，您的程序会以与平常大致相同的速度运行。下面说明了如何进行概要分析:

```
cc -c dhry.c         # Create dhry.o without call counting code.
cc -pg dhry.o -L/lib -L/usr/lib -o dhryfast
                     # Re-link (and avoid -pg libraries).
dhryfast             # Create gmon.out without call counts.
gprof >dhryfast.out  # You get an error message about no call counts
                     #  -- ignore it.
```

在不调用计数的情况下运行时，一些快速执行的功能根本不会显示在列表中（但您知道必须要调用它们）。尽管不可见，但这个结果对 **gprof** 命令而言是正常的。**gprof** 命令仅列出了那些至少调用一次或每个时钟周期至少注册一次的功能。虽然快速执行函数也运行，但快速执行功能通常不会接收任何时钟周期。由于暂挂了调用计数功能，因此这些小函数不会被列出。（如果在 **cc -pg** 命令行中省略了 **-L** 选项，您可以获得运行时例程的调用计数。）

使用更少的实内存

由于 **-pg** 选项专用相当于程序文本大小一半的固定实时内存缓冲区，因此使用 **gprof** 命令进行概要分析时可能导致程序过度分页。过度分页不会影响概要分析所生成的数据，原因是被分析的程序在等待 I/O 时不生成时钟周期（它仅在使用 CPU 时才这样做）。如果因为过度分页而导致的时间延迟是不可接受的，建议您使用 **tprof** 命令。

标志

-b 禁止打印概要文件中每个字段的描述。

-c *Filename* 创建包含远程概要文件信息处理所需信息的文件。请勿将 **-c** 标志与其他标志一起使用。

-E *Name* 禁止打印例程 *Name* 及其子代的图表概要文件项，此标志类似于 **-e** 标志，但它在总时间和百分比时间的计算中排除了由例程 *Name* 及其子代所用的时间。（**-E** *MonitorCount* **-E** *MonitorCleanup* 为缺省值。）

-e *Name* 禁止打印例程 *Name* 及其所有子代的图表概要文件项（除非它们有未被限制的其他祖先）。可以给定多个 **-e** 标志。一个 **-e** 标志只能指定一个例程。

-F *Name* 打印例程 *Name* 及其子代的图表概要文件项，它类似于 **-f** 标志，但它在总时间和百分比时间计算中仅使用所打印的例程的时间。可以指定多个 **-F** 标志。一个 **-F** 标志只能指定一个例程。**-F** 标志覆盖 **-E** 标志。

-f *Name* 打印指定的例程 *Name* 及其子代的图表概要文件项。可以指定多个 **-f** 标志。一个 **-f** 标志只能指定一个例程。

-g *Filename* 向指定的输出 *filename* 写入调用图表信息。它也禁止概要文件信息，除非使用了 **-p** 标志。

-i *Ffilename* 向指定的输出 *filename* 写入例程索引表。如果未使用该标志，则索引表或者位于标准输出的底部，或者位于用 **-p** 和 **-g** 标志指定的 filename 的底部。

-L *PathName* 使用备用的路径名定位共享对象。

-p *Filename* 向指定的输出 filename 写入平面概要文件信息。它也禁止调用图表信息，除非使用了 **-g** 标志。

-s 生成表示所有指定的概要文件中的概要分析信息的总结的 **gmon.sum** 概要文件。这个总结性的概要文件可以提供给 **gprof** 命令的后继执行（使用 **-s** 标志），从而可将数次运行 **a.out** 文件的概要分析数据累积起来。

-x *Filename* 从 *Filename*（以 **-c** 选项创建的文件）获取信息来生成概要文件报告。如果不指定 *Filename*，**gprof** 命令将搜索缺省的 **gprof.remote** 文件。

-z 显示使用次数为零的例程（按照调用计数和累积时间的指示）。

示例

1. 要获取概要分析输出，请输入:

   ```
   gprof
   ```

2. 要获取先前运行的命令的概要分析输出（可能已移走），可输入:

   ```
   gprof -L/home/score/lib runfile runfile.gmon
   ```

本示例使用给定的 **runfile.gmon** 文件作为样本数据和 **runfile** 文件作为本地符号，检查 **/u/score/lib** 文件中的可加载对象。

3. 要对样本程序 **dhry.c** 进行概要分析:
   a. 使用 **cc -pg** 命令重新编译应用程序，如下所示:

      ```
      cc -pg dhry.c -o dhry # Re-compile to produce gprof output.
      ```

   b. 运行重新编译的程序。在当前的工作目录（而不是该程序的可执行文件所驻留的目录）中创建名为 gmon.out 的文件。

      ```
      dhry    # Execute program to generate ./gmon.out file.
      ```

   c. 在包含 gmon.out 文件的目录中运行 **gprof** 命令以产生 CALL-GRAPH 和 FLAT PROFILE 报告。

      ```
      gprof >gprof.out     # Name the report whatever you like
      vi gprof.out         # Read flat profile first.
      ```

   d. 要生成线程级别概要分析详细程度，请如下导出 GPROF 环境变量并运行该应用程序，请执行以下操作:

      ```
      export GPROF=profile:thread
      dhry   # Execute program to generate ./gmon.out file which has thread level granularity
      ```

   e. 要使用 mygmon 前缀生成每进程 **gmon.out** 文件，请执行以下操作:

      ```
      export GPROF=file:multi,filename:mygom
      dhry  # Execute program to generate ./gmon-dhry-2468.out
      ```

   f. 要生成每线程 **gmon.out** 文件，且比例因子为 10，文件名前缀为 tgmon，请执行以下操作:

      ```
      export GPROF=profile:thread,file:multithread,scale:10,filename:tgmon
      dhry # Execute program to generate ./tgmon-dhry-2468-Pthread215.out
      ```

   g. 要从 gmon-dhry-2468.out 中仅查看图形概要文件报告，请执行以下操作:

      ```
      gprof -p fprofile.out ./dhry ./gmon-dhry-2468.out
      ```

   h. 要从 gmon-dhry-2468.out 中仅查看平面概要文件报告，请执行以下操作:

      ```
      gprof -g callgraph.out ./dhry ./gmon-dhry-2468.out
      ```

4. 要使用 **gprof** 命令的远程处理功能:
   a. 以 **cc -pg** 命令重新编译应用程序:

      ```
      cc -pg thread.c -o thread  -lpthread
      ```

   b. 启用线程级概要分析详细程度并使用不同的 **gmon.out** 名称:

      ```
      export GPROF=profile:thread,filename:mygmon
      ```

   c. 运行重新编译的程序。在当前工作目录（不是程序可执行文件驻留的目录）中创建名为 **mygmon.out** 的文件:

      ```
      thread    # Execute program to generate mygmon.out file.
      ```

   d. 使用 **-c** 标志生成 **my.remote** 文件，接下来该文件可以被带到远程机器上用于处理:

      ```
      gprof -c my.remote thread mygmon.out
      ```

   e. 在远程机器上，使用 **-x** 标志从 **my.remote** 文件抽取信息:

      ```
      gprof -x my.remote
      ```

在 **gprof** 命令的通篇描述中，大部分示例都使用了 C 程序 **dhry.c**。但是，只要将 C 编译器和 **cc** 替换成相应的编译器名称，并将 *function* 一词替换成 *subroutine*，则关于这些示例的讨论也同样适用于 FORTRAN、Pascal 或 COBOL 模块。例如，以下命令即显示了如何对名为 matrix.f 的 FORTRAN 程序进行概要分析:

```
xlf -pg matrix.f -o matrix # 对 matrix.f 程序进行 FORTRAN 样式的编译
matrix                   # Execute with gprof profiling,
                         #   generating gmon.out file
gprof > matrix.out       # Generate profile reports in
                         #   matrix.out from gmon.out
vi matrix.out            # Read flat profile first.
```

文件

a.out	名称列表和文本空间
gmon.out	动态调用图和概要文件
gmon.sum	动态调用图和概要文件总结
gprof.remote	远程概要分析文件
/usr/ucb/gprof	包含 **gprof** 命令。
/usr/ccs/bin/gprof	包含 **gprof** 命令

相关信息

cc 命令、**prof** 命令。

exit 子例程、**monitor** 子例程、**profil** 子例程。

《*性能管理*》中的『监控与调整命令和子例程』。

《*操作系统与设备管理*》中的『命令』。

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『子例程概述』。

---

grap 命令

用途

对图表进行排版以供 **pic** 命令处理。

语法

grap [**-l**] [**-t** *Name*] [**—**] [*File ...*]

描述

grap 命令用于处理 grap 语言输入文件并生成 **pic** 命令的输入。grap 语言是一种图表排版语言。以下是其典型的命令行语句:

```
grap 文件 | pic | troff | Typesetter
```

图表在 **.G1** 和 **.G2 troff** 命令请求之间。这些请求所封装的数据会使用自动提供的选项得到缩放并被绘制成图表。同时也提供了修改框架、添加标签、替代缺省选项、更改绘图样式、定义坐标范围和转换以及从文件包含数据的命令。另外，**grap** 命令还提供了与 **pic** 命令相同的循环、坐标以及宏处理。

Grap 语言文件包含 grap 程序。grap 程序是用以下形式编写的:

```
.G1
grap Statement
grap Statement
grap Statement
.G2
```

参数

File 指定将通过 **grap** 命令处理并输入到 **pic** 命令的 grap 语言文件（grap 程序）。

grap 语句摘要

以下对可用于创建 grap 程序的 grap 语句进行了总结:

frame 定义图表周边的框架。其语法为:

```
frame [ht Expression] [wid Expression] [[Side] LineDescription]
```

有关属性的定义如下:

- *Side*: top, bot, left, right
- *LineDescription*: solid, invis, dotted [Expression], dashed [Expression]

高度值缺省为 2 英寸，宽度值缺省为 3 英寸，侧边缺省为实线。如果省略了 side，则 *linedesc* 将应用于整个框架。

label 在图表的指定一侧放置标签。其语法为:

```
label Side StringList ... Shift
```

有关属性的定义如下:

- *Shift*: left, right, up, or down *expression*
- *StringList*: str ... rjust, ljust, above, below [size (+)Expression] ...
- *String*: "..."

coord 定义替换的坐标系统。其语法为:

```
coord [Name] [x Expression,Expression] [y  Expression,Expression] [[log x] [log y] [log log]]
```

ticks 将 tick 标记放在框架的一侧。其语法为:

```
ticks side [[in] [out] [Expression]] [Shift] [TickLocations]
```

有关属性的定义如下:

- *Shift*: left, right, up, down Expression
- *TickLocations*: at [*Name*] *Expression* [*String*], *Expression* [*String*], ... from [*Name*] *Expression* to *Expression* [by [*Operation*] *Expression*] *String*

如果未指定任何 tick，则它们会自动提供; ticks off 可禁止自动 tick。

grid 沿着（也就是垂直于）指定的一侧生成网格线。其语法为:

```
grid Side [LineDescription] [Shift] [TickLocations]
```

在标注网格时会使用与标记相同的机制。

plot 在某处放置文本。其语法为:

```
StartList at Point plot Expression [Start] at Point
```

有关属性的定义如下:

- *StringList*: str ... rjust, ljust, above, below [size +)Expression] ...
- *Point*: [Name] Expression Expression

line 在两个点之间绘制线条或箭头。其语法为:

```
{line | arrow} from Point to Point [LineDescription]
```

linedesc 的属性定义如下:

- *Point*: [*Name*] *Expression Expression*
- *LineDescription*: `solid, invis, dotted [Expression], dashed Expression]`

circle 绘制圆。其语法为:

```
circle at Point [radius Expression]
```

半径单位为英寸; 缺省大小为 “小”。

draw 定义线条序列。其语法为:

```
draw [Name] at Point[LineDescription]
```

next 继续序列。其语法为:

```
next [Name] at Point [LineDescription]
```

new 开始新序列。其语法为:

```
new [Name] at Point [LineDescription]
```

numberlist 按照给定的一组数字创建线条。这些数字被视作 *x*、*y*1、*y* 2 等坐标点; 在绘图时将在单个的 *x* 坐标值描点。其语法为:

```
number x, y1, y2 ...
```

for 创建循环。其语法为:

```
for Variable {from | =} Expression to Expression [by [arithmetic or multiplicative operator] Expression] do X Anything X
```

X 为任何未在字符串中出现的单个字符。如果 X 为左括号 ″{″, 则在字符串中可能包含后跟右括号 ″}″ 的成对括号。当变量的取值是在从第一个表达式到第二个表达式的范围时, 会重复 Anything 文本。

if 创建条件评估。其语法为:

```
if 表达式 then X Anything X [else X Anything X]
```

define 提供与 “优先级中断控制器 （PIC）” 相同的宏处理程序。其语法为:

```
define 宏名称 X Anything X
```

copy 复制文件; 包含该文件的当前内容。其语法为:

```
copy 文件名
```

copy-thru 通过宏复制文件。

```
copy 文件名 thru 宏名称
```

所有数字或引用的字符串都被视作参数。复制操作将一直进行到文件结束或下一个 .G2。until String 子句是可选的。在某一行的开始域为字符串时, 该子句可以使复制操作停止。

以下语句可通过宏复制后继的行:

```
copy thru 宏名称
```

在所有情况下, 您都可以按内联而不是按名称来指定宏:

```
copy thru x MacroBody x
```

sh 将文本传递给 UNIX shell。其语法为:

```
sh x Anything x
```

变量 Anything 会经过扫描, 以确定它是否为宏。**pid** 宏是内置的。它是由进程标识号组成的字符串; 您可以使用它来生成唯一的文件名。

pic 将文本传递给 pic，并移除 **pic**。变量和宏将被忽略。以句点开始的行（不是数字）会假设为**troff** 命令而按字面传递。

graph 定义名为 *Picname* 的新图表，并且重新设置所有的坐标系统。其语法为:

```
graph Picname [pic-text]
```

当 **graph** 命令用在 grap 程序中时，**graph** 命令必须跟在 .G1 之后。使用图表文本，您可以定位与以前的图表相关的这个图表。方法是像以下示例那样引用它们的框架:

```
graph First
 ...
graph Second with .Frames.w at First.Frame.e + [0.1,0]
```

图表文本中的宏和表达式将被忽略。图表名必须符合 **pic** 语法以大写字母开头。

print 在 grap 处理其输入时写入标准错误。此语句在调试时可能非常有用。其语法为:

```
print [Expression | String]
```

grap 语言约定

grap 语言使用以下约定:

- #（井号符）表示开始一个注释。注释在一行的结束处自动中止。
- 持续有多行的语句必须在每个新行的开头放置一个 \ （反斜杠字符）。
- 在一行中出现的多个语句必须用分号分隔开。
- grap 语言将忽略空白行。
- 预定义的字符串包括: bullet、plus、box、star、dot、times、htick、vtick、square 以及 delta。
- grap 中可用的内置函数包括: log（以 10 为底）、exp（以 10 为底）、int、sin、cos、atan2、sqrt、min、max 和 rand。

标志

-l 禁止 **grap** 命令查找 **/usr/lib/dwb/grap.defines** 宏定义库文件。

-t*Name* 将 *Name* 变量的值指定为 **grap** 命令的输出设备。缺省值为 **-Tibm3816**。

- - （一对破折号）指示标志的结束。

File

/usr/lib/dwb/grap.defines 包含标准绘图字符的定义。

相关信息

pic 命令。

---

greek 命令

用途

将 Teletype Model 37 工作站的英文输出转换成其他工作站输出。

语法

greek [**-t** *Name*]

描述

greek 命令可重新翻译（包括逆向翻译和半行翻译）Teletype Model 37 的字符集以便在其他的工作站中显示。如果可能，它会通过叠印处理来模拟特殊字符。**greek** 命令读取标准输入并写入到标准输出中。

标志

-t*Name*	使用指定的工作站名称。如果省略了 **-T** 标志，**greek** 命令会试图使用在 **$TERM** 环境变量中指定的工作站。*Name* 变量的值可以是下述的任何一个:
300	DASI 300
300-12	间距为 12 的 DASI 300
300s	DASI 300s
300s-12	间距为 12 的 DASI 300s
450	DASI 450
450-12	间距为 12 的 DASI 450
2621	Hewlett-Packard 2621、2640 和 2645
2640	Hewlett-Packard 2621、2640 和 2645
2645	Hewlett-Packard 2621、2640 和 2645
4014	Tektronix 4014
hp	Hewlett-Packard 2621、2640 和 2645
tek	Tektronix 4014。

环境变量

$TERM 指定工作站名称。

相关信息

eqn 命令、**hp** 命令、**mm** 命令、**neqn** 命令、**nroff** 命令、**troff** 命令。

---

grep 命令

用途

在文件中搜索模式。

语法

grep [**-E** | **-F**] [**-i**] [**-h**] [**-H**] [**-L**] [**-r** | **-r**] [**-s**] [**-v**] [**-w**] [**-x**] [**-y**] [[[**-b**] [**-n**]] | [**-c** | **-l** | **-q**]] [**-p** [*Separator*]] { [**-e** *PatternList* ...] [**-f** *PatternFile* ...] | *PatternList* ... } [*File* ...]

描述

grep 命令用于搜索由 *Pattern* 参数指定的模式，并将每个匹配的行写入标准输出中。这些模式是具有限定的正则表达式，它们使用 **ed** 或 **egrep** 命令样式。**grep** 命令使用压缩的不确定算法。

如果在 *File* 参数中指定了多个名称，**grep** 命令将显示包含匹配行的文件的名称。对 shell 有特殊含义的字符（$, *, [, |, ^, (,), \ ）出现在 *Pattern* 参数中时必须带双引号。如果 *Pattern* 参数不是简单字符串，通常必须

用单引号将整个模式括起来。在诸如 [a-z] 之类的表达式中，-（减号）cml 可根据当前正在整理的序列来指定一个范围。整理顺序可以定义等价的类以供在字符范围中使用。如果未指定任何文件，**grep** 会假定为标准输入。

注:

1. 请勿对特殊文件运行 **grep** 命令，这样做可能产生不可预计的结果。
2. 输入行不应包含空字符。
3. 输入文件应该以换行符结束。
4. 换行符不会与正则表达式匹配。
5. 虽然一些标志可以同时被指定，但其中的某些标志会覆盖其他标志。例如，**-l** 选项将优先于所有其他标志。另外，如果您同时指定了 **-E** 和 **-F** 标志，则后指定的那个会有优先权。

标志

-b 在每行之前添加找到该行时所在的块编号。使用这个标志有助于通过上下文来找到磁盘块号码。**-b** 标志不能用于来自标准输入和管道的输入。

-c 仅显示匹配行的计数。

-E 将每个指定模式视作扩展的正则表达式（ERE）。ERE 的空值将匹配所有的行。

注: 带有 **-E** 标志的 **grep** 命令等价于 **egrep** 命令，只不过它们的错误和使用信息不同以及 **-s** 标志的作用不同。

-e *PatternList* 指定一个或多个搜索模式。其作用相当于一个简单模式，但在模式以 -（减号）开始的情况下，这将非常有用。模式之间应该用换行符分隔。连续使用两个换行符或者在引号后加上换行符 （"\n） 可以指定空模式。除非同时指定了 **-E** 或 **-F** 标志，否则每个模式都将被视作基本正则表达式（BRE）。**grep** 可接受多个 **-e** 和 **-f** 标志。在匹配行时，所有指定的模式都将被使用，但评估的顺序没有指定。

-F 将每个指定的模式视作字符串而不是正则表达式。空字符串可匹配所有的行。

注: 带有 **-F** 标志的 **grep** 命令等价于 **fgrep** 命令，只不过它们的错误和使用信息不同以及 **-s** 标志具有不同的作用。

-f *PatternFile* 指定包含搜索模式的文件。模式之间应该用换行符加以分隔，空行将被认为是空模式。每种模式都将被视作基本的正则表达式（BRE），除非同时指定了 **-E** 或 **-F** 标志。

-h 禁止在匹配行后附加包含此行的文件的名称。当指定多个文件时，将禁止文件名。

-H 如果指定了 **-r** 或 **-R** 选项并且在命令行上指定了引用文件类型目录的符号链接，则 **grep** 将搜索符号链接所引用的目录文件以及文件层次结构中在它以下的所有文件。

-i 在进行比较时忽略字母的大小写。

-l 仅列出（一次）包含匹配行的文件的名称。文件名之间用换行符加以分隔。如果搜索到标准输入，将返回（标准输入）的路径名。**-l** 标志同 **-c** 和 **-n** 标志的任意组合一起使用时，其作用类似于仅使用了 **-l** 标志。

-L 如果指定了 **-r** 或 **-R** 选项，并且引用文件类型目录的符号链接在命令行上指定或在文件层次结构转移过程中遇到，则 **grep** 将搜索符号链接所引用的目录文件以及文件层次结构中在它以下的所有文件。如果同时指定了 **-H** 和 **-L**，则命令行上最近指定的选项将生效。

-n 在每一行之前放置文件中相关的行号。每个文件的起始行号为 1，在处理每个文件时，行计数器都将被复位。

-p[*Separator*] 显示包含匹配行的整个段落。段落之间将按照 *Separator* 参数指定的段落分隔符加以分隔，这些分隔符是与搜索模式有着相同格式的模式。包含段落分隔符的行将仅用作分隔符，它们不会被包含在输出中。缺省的段落分隔符是空白行。

-q 禁止所有写入到标准输出的操作，不管是否为匹配行。如果选择了输入行，则以零状态退出。**-q** 标志同 **-c** 和 **-l**、**-n** 标志的任意组合一起使用时，其作用类似于仅使用了 **-q** 标志。

-r 递归地搜索目录。在缺省情况下，按照到目录的链接。

-r 递归地搜索目录。在缺省情况下，不按照到目录的链接。
-s 禁止通常因为文件不存在或不可读取而写入的错误信息。其他的错误信息并未被禁止。
-v 显示所有与指定模式不匹配的行。
-w 执行单词搜索。
-x 显示与指定模式精确匹配而不含其他字符的行。
-y 当进行比较时忽略字符的大小写。
PatternList 指定将在搜索中使用的一个或多个模式。这些模式将被视作如同是使用 **-e** 标志指定的。
File 指定将对其进行模式搜索的文件的名称。如果未给出 *File* 变量，将使用标准输入。

退出状态

此命令返回以下出口值:

0 找到匹配项。
1 未找到匹配项。
>1 发现语法错误，或者文件不可访问（即使找到了匹配项）。

示例

1. 若使用包含以下模式匹配字符的模式: *, ^, ?, [,], \(, \), \{ 和 \}, 请输入:

   ```
   grep  "^[a-zA-Z]"  pgm.s
   ```

   这将显示 pgm.s 中第一个字符为字母的所有行。
2. 若显示所有与模式不匹配的行，请输入:

   ```
   grep   -v  "^#" pgm.s
   ```

   这将显示 pgm.s 中首字母不是 #（井字符）的所有行。
3. 若显示文件 file1 中与abc 或 xyz 字符串匹配的所有行，请输入:

   ```
   grep -E  "abc|xyz"  file1
   ```
4. 若在名为 test2 的文件中搜索 $（美元符号），请输入:

   ```
   grep \\$ test2
   ```

   为了强制 shell 将 \$（单反斜杠和美元符号）传递给 **grep** 命令，必须要使用 \\（双反斜杠）。\（单反斜杠）字符可通知 **grep** 命令将其后的字符（本例中为 $）视作原义字符而不是表达式字符。如果使用 **fgrep** 命令，则可以不必使用反斜杠之类的转义字符。
5. 通过 **/tmp** 递归地搜索以找到含有字 IBM 的文件，而不是通过指向目录的链接进行递归搜索，请输入:

   ```
   grep –R IBM /tmp
   ```

   或

   ```
   grep –r -H IBM /tmp
   ```
6. 要通过 **/tmp** 递归地搜索以找到含有词 IBM 的文件，同时也通过链接进行递归搜索，请输入:

   ```
   grep –r IBM /tmp
   ```

   或

   ```
   grep -R -L IBM /tmp
   ```

文件

/usr/bin/grep 包含 **grep** 命令。

相关信息

ed 命令、**egrep** 命令、**fgrep** 命令、**sed** 命令。

《*操作系统与设备管理*》中的『文件』。

《*操作系统与设备管理*》中的『输入和输出重定向』。

《*AIX 5L V5.3 本地语言支持指南和参考大全*》中的『本地语言支持概述』。

《*操作系统与设备管理*》中的『Shell』。

---

groups 命令

用途

显示组成员身份。

语法

groups [*User...*]

描述

缺省情况下，**groups** 命令可将当前进程的组成员身份写入标准输出中。如果命令参数指定了多个用户，则每个用户的组成员身份都将被显示。

如果未在用户数据库中找到给定的用户，**groups** 命令将在发出警告信息后继续它对参数列表中的下一个用户的操作。

安全性

访问控制: 该程序应该作为“信任的计算库”中的正常用户程序安装。

示例

若显示在参数列表中列出的用户的组成员身份，请输入:

```
$ groups sys root lp adm
sys : sys
root : system bin sys security cron audit lp
lp : lp printq
adm : adm
```

文件

/usr/bin/groups	包含 **groups** 命令
/usr/ucb/groups	到 **groups** 命令的符号链接
/etc/group	组文件; 包含组标识
/etc/ogroup	此组文件的前一个版本

/etc/passwd 密码文件；包含用户标识
/etc/opasswd 此密码文件的前一个版本

相关信息

getty 命令、**login** 命令、**setgroups** 命令、**su** 命令、**tsm** 命令。

---

grpck 命令

用途

验证组定义的正确性。本文档描述了 AIX **grpck** 命令和 System V **grpck** 命令。

语法

grpck { **-n** | **-p** | **-t** | **-y** } { **ALL** | *Group* ... }

描述

grpck 命令通过检查**所有**组或由 *Group* 参数所指定的组的定义，来验证用户数据库文件中的组定义是否正确。如果指定了多个组，则组与组之间必须有空格。

注: 此命令会将其消息写入到**标准错误**中。

您必须选择一个标志以指示是否让系统设法修复错误的属性。以下属性将被检查:

name 检查组名称的唯一性和组成。组名称必须是等于或少于 8 个字节的唯一字符串。它不能以 +（加号）、:（冒号）、-（减号）或 ~（鼻音符） 打头。另外，它在字符串中不能包含冒号 (:)，并且不能是关键字 **ALL** 或 **default**。系统不能修复此类错误。
groupID 检查组标识的唯一性和组成。组标识不能为空，而且只能由十进制数组成。系统不能自行修复此类错误。
users 检查在组数据库文件中列出的用户是否存在。如果指示让系统修复错误，它将删除所有未能在用户数据库文件中找到的用户。
adms 检查在组数据库文件中作为组管理员列出的用户是否存在。如果指示让系统修复错误，它将删除所有未能在用户数据库文件中找到的管理员。

admin 检查 **/etc/security/group** 文件中的每个组的有效管理属性。没有可用的系统修复。

sysck 命令通常都会调用 **grpck** 命令作为验证受信任系统的安装的一部分。另外，root 用户或安全组的成员也可以使用此命令。

grpck 命令将检查数据库管理安全文件（**/etc/passwd.nm.idx**、**/etc/passwd.id.idx**、**/etc/security/passwd.idx** 以及 **/etc/security/lastlog.idx**）是否为最新的或是否比相应的系统安全文件新。请注意，**/etc/security/lastlog.idx** 不如 **/etc/security/lastlog** 新是正常的。如果数据库管理安全文件已过时，会显示警告信息以指示 root 用户该运行 **mkpasswd** 命令了。

标志

-n 报告错误，但不会修复它们。
-p 修复错误，但不进行报告。
-t 报告错误并询问是否修复。
-y 修复错误并作出报告。

安全性

访问控制: 此命令应该将执行（x）访问的权限授予 root 用户和安全组的成员。此命令的 **setuid** 应设置成 root 用户，并具有**信任的计算库**属性。

访问的文件:

方式	文件
r	**/etc/passwd**
r	**/etc/security/user**
rw	**/etc/security/group**
rw	**/etc/group**

审计的事件:

事件	信息
GROUP_User	用户、组、属性 ! 错误、状态
GROUP_Adms	用户、组、属性 ! 错误、状态

示例

1. 验证所有的组成员和管理员是否在用户数据库中存在，并要求报告任何错误信息（不修复），可输入:

   ```
   grpck  -n ALL
   ```

2. 验证所有的组成员和管理员是否在用户数据库中存在，并要求修复错误（不报告），可输入:

   ```
   grpck  -p ALL
   ```

3. 验证为 install 组定义的组名称和组标识的唯一性，可输入:

   ```
   grpck  -n install
   ```

   或者

   ```
   grpck  -t install
   ```

   或者

   ```
   grpck  -y install
   ```

   grpck 命令不更正组的名称和标识错误。因此，**-n**、**-t** 以及 **-y** 标志将仅报告组名称和组标识的错误但不更正它们。

文件

/usr/sbin/grpck	包含 **grpck** 命令。
/etc/passwd	包含用户的基本属性。
/etc/security/user	包含用户的扩展属性。
/etc/group	包含组的基本属性。
/etc/security/group	包含组的扩展属性。

相关信息

pwdck 命令、**sysck** 命令、**usrck** 命令。

关于用户的标识和认证、自主访问控制、可信计算库和审计的更多信息，请参阅《*安全性*》。

System V grpck 命令

语法

/usr/sysv/bin/grpck

描述

/usr/sysv/bin/grpck 命令通过检查“所有”组的定义来验证用户数据库文件中组定义的正确性。此 **/usr/sysv/bin/grpck** 命令是 **/usr/sbin/** 中现有 **grpck** 命令的 System V 版本。此命令用 **-n** 和 **ALL** 选项调用 **/usr/sbin/grpck** 命令。

退出状态

0 成功完成。

>0 发生错误。

示例

1. 验证所有的组成员和管理员是否在用户数据库中存在，并要求报告任何错误信息（不修复），可输入:

   ```
   /usr/sysv/bin/grpck
   ```

文件

/usr/sysv/bin/grpck
包含 **grpck** 命令的 System V 版本。

相关信息

/usr/sbin/grpck 命令。

---

grpsvcsctrl 命令

用途

启动组服务子系统。

语法

grpsvcsctrl { **–a** | **–s** | **–k** | **–d** | **–c** | **–u** | **–t** | **–o** | **–h** }

描述

grpsvcsctrl 命令用于启动组服务子系统。此控制脚本可控制组服务所需的子系统操作。这些子系统由“系统资源控制器（SRC）”控制，并属于名为 **grpsvcs** 的子系统组。每个子系统都关联着相应的守护程序。从视图的操作点，服务组的子系统组按如下组织:

子系统	组服务
子系统组	**grpsvcs**
SRC 子系统	**grpsvcs** — 与 **hagsd** 守护程序相关联。节点上的子系统名为 **grpsvcs**。每个节点上的 **grpsvcs** 子系统都与此节点所属的集群相关联。
守护程序	**hagsd** — 提供大部分的组服务功能。

grpsvcsctrl 脚本通常不从命令行运行。它通常在集群安装期间由启动命令调用。

grpsvcsctrl 脚本提供了大量控制功能来操作组服务子系统:

- 添加、启动、停止、删除和清理组服务子系统
- 打开或关闭跟踪程序

在执行上述任何功能之前，该脚本都要首先获得当前集群的名称。

添加子系统: 当指定了 **-a** 标志时，该控制脚本会使用 **mkssys** 命令向 SRC 添加组服务子系统。控制脚本运行如下:

1. 确保 **grpsvcs** 子系统已停止。
2. 从全局对象数据管理器（ODM）获取用于该集群的 **grpsvcs** 子系统的端口号并确保此端口号已在 **/etc/services** 文件中设置。有效的端口号的范围为 10000 到 10100（包含 1000 和 10100）。
3. 在 **/etc/services** 文件中输入的服务名称应该为 **grpsvcs.***cluster_name*。
4. 从 SRC 中删除 **grpsvcs** 子系统（如果它仍留在此位置）。
5. 向 SRC 添加 **grpsvcs** 子系统。在 **mkssys** 命令中将相应的集群名称配置为守护程序的参数。

启动子系统: 当指定 **-s** 标志时，此控制脚本使用 **startsrc** 命令启动组服务子系统 **grpsvcs**。

停止子系统: 当指定了 **-k** 标志时，此控制脚本使用 **stopsrc** 命令停止组服务子系统 **grpsvcs**。

删除子系统: 当指定了 **-d** 标志时，该控制脚本使用 **rmssys** 命令从 SRC 中删除组服务子系统。控制脚本运行如下:

1. 确保 **grpsvcs** 子系统已停止。
2. 使用 **rmssys** 命令从 SRC 中删除 **grpsvcs** 子系统。
3. 从 **/etc/services 文件**中删除有关的端口号。

清理子系统: 当指定了 **-c** 标志时，该控制脚本将停止所有系统分区的组服务子系统并从 SRC 中将它们删除。控制脚本运行如下:

1. 使用 **stopsrc -g grpsvcs** 命令停止所有分区中的子系统组的子系统实例。
2. 使用 **rmssys** 命令从 SRC 中删除所有分区的子系统组的所有子系统实例。

打开跟踪功能: 当指定了 **-t** 标志时，该控制脚本使用 **traceson** 命令打开 **hagsd** 守护程序的跟踪功能。

禁用跟踪功能: 当指定了 **-o** 标志时，该控制脚本会使用 **tracesoff** 命令关闭 **hagsd** 守护程序的跟踪功能（使其恢复为缺省值）。

记录日志: 在组服务守护程序运行时，它们将在 **/var/ha/log** 目录下的一个日志文件中写入一些信息项，从而来提供有关其操作和错误的信息。

每个守护程序都按照预先设定的行数来限制日志的大小。缺省值为 5000 行。当限幅一到，后台程序附加字符串 **.bak** 到当前的日志文件名后，然后开始一个新的日志。如果 **.bak** 版本已存在，旧版本删除后当前日志才能更名。

标志

-a 添加子系统。

-s 启动子系统。

-k 停止子系统。

-d 删除子系统。

-c 清理子系统（即，从所有系统分区中将它们删除）。

-u 从所有分区中删除组服务子系统。

-t 打开子系统跟踪程序。

-o 关闭子系统跟踪程序。

-h 将脚本的用法声明写入到标准输出中。

安全性

您必须以有效的 **root** 用户标识来运行该命令。

退出状态

0 指示命令已成功完成。

1 显示有错误产生。

限制

此脚本仅在 HACMP 环境中有效。

标准输出

当 **-h** 标志被指定时， 该命令的用法说明写到标准输出中去。

标准错误

必要时此命令将错误信息写到标准错误中。

示例

1. 向 SRC 添加组服务子系统，可输入:

   ```
   grpsvcsctrl -a
   ```

2. 若启动组服务子系统，可输入:

   ```
   grpsvcsctrl -s
   ```

3. 若停止组服务子系统，可输入:

   ```
   grpsvcsctrl -k
   ```

4. 若从 SRC 中删除组服务子系统，可输入:

   ```
   grpsvcsctrl -d
   ```

5. 若清理组服务子系统，可输入:

   ```
   grpsvcsctrl -c
   ```

6. 若启用组服务守护程序 **hagsd** 的跟踪功能，可输入:

   ```
   grpsvcsctrl -t
   ```

7. 若关闭组服务守护程序 **hagsd** 的跟踪功能，可输入:

   ```
   grpsvcsctrl -o
   ```

位置

/usr/sbin/rsct/bin/grpsvcsctrl 包含 **grpsvcsctrl** 脚本

文件

/var/ha/log/grpsvcs_nodenum_instnum.cluster_name
包含节点上的 **hagsd** 守护程序的日志

文件名称包含以下变量:

nodenum
是运行守护程序的节点的编号

instnum
是守护程序的实例编号

cluster_name
是运行守护程序的集群的名称

相关信息

命令: **lssrc**、**mkssys**、**rmssys**、**startsrc**、**stopsrc**

守护程序: **hagsd**

---

gssd 守护程序

用途

GSS 操作的服务内核请求。

语法

/usr/sbin/gssd

描述

某些 NFS 安全方法（如 Kerberos 5）在称为 “常规安全服务” （GSS）的更加常规的机制下提供。在 AIX 中，IBM Network Authentication Service（NAS）文件集中的库提供 GSS 服务。扩展包中提供 NAS。**gssd** 守护程序使这些 GSS 服务可用于 NFS 服务器内核代码。如果 **gssd** 守护程序没有运行，则通过使用 GSS 安全方法（如 Kerberos 5）的 NFS 访问文件的努力将失败。**gssd** 守护程序使用 RPC 程序号 400234 注册。

使用以下系统资源控制器（SRC）命令启动和停止 **gssd** 守护程序:

```
startsrc -s gssd
stopsrc -s gssd
```

文件

/etc/nfs/hostkey 用以下格式指定 **keytab** 文件位置和主机主体:

```
到 keytab 文件的路径
主机主体
```

/etc/nfs/princmap 用以下格式指定到主机主体的映射:

```
principal1 alias1 alias2 alias3
principal2 alias1
```

别名可以是 IP 地址或主机名; 主体必须匹配 Kerberos 维护的主机密钥。

---

ha_star 命令

用途

处理高可用性事件。

语法

ha_star [**-C**]

描述

ha_star 命令是一般的高可用性处理的命令。当固件报告了可预见的 CPU 故障时，操作系统会自动通过 **/etc/rc.ha_star** 来调用此命令。

如果不带任何标志调用 **ha_star**，则它仅处理新事件。如果 **ha_star** 没有找到任何新事件，它将退出。

ha_star 在运行时会处理所有的新事件，即使这些事件抵达时 **ha_star** 正在处理现已存在的事件。在任何给定的时候，都只能运行一个 **ha_star** 的实例。如果启动了 **ha_star** 的第二个实例，它将退出。

当报告了高可用性事件时，操作系统将调用 **ha_star**。这种事件处理可能失败或被取消（例如，通过信号）。终止或取消的事件被保存在内核存储器中。当导致终止的原因得到更正后，可以重新尝试执行该事件处理。此时要求系统管理员手动调用 **ha_star**。

ha_star 命令可生成错误或故障错误日志项。

按事件类型所作的描述

当检测到可预见的处理器故障时，操作系统将调用 **ha_star** 命令来释放对 CPU 的分配。由于某些线程会继续绑定在要释放其分配的 CPU 上，因此释放 CPU 分配的操作可能失败。有时，系统管理员可以修复将导致释放分配失败的状况。例如，他们也许能确定并停止那些含有绑定在最后一个逻辑 CPU 上的线程的应用程序。

-C 标志可指示将要继续的高可用性事件属于释放 CPU 分配事件。

标志

-C 指定重新启动的事件是释放 CPU 分配事件。

文件

/usr/sbin/ha_star 包含 **ha_star** 命令。

相关信息

《*AIX 5L V5.3 通用编程概念: 编写并调试程序*》中的『动态处理器释放』。

《*操作系统与设备管理*》中的『启用动态处理器释放』。

---

ha.vsd 命令

用途

查询和控制可恢复虚拟共享磁盘子系统的 rvsd 守护程序的活动。

语法

ha.vsd {**adapter_recovery [on | off] | debug [off] | mksrc | query | quorum** *n* **| qsrc | refresh [noquorum] | reset | reset_quorum | rmsrc | start | stop | trace [off]**}

描述

使用此命令来显示关于可恢复虚拟共享磁盘子系统的信息、更改定额所需的节点数以及更改子系统的状态。

标记

–a 指定所有虚拟共享磁盘。

–v *vsd_name_list*
指定一个或多个虚拟共享磁盘名，用逗号分开。

–n *node_list* 指定一个或多个节点号，用逗号分开。

参数

adapter_recovery [on | off]
启用或禁用通信适配器恢复。缺省值是 **on**。

必须重新启动可恢复虚拟共享磁盘子系统使此操作数生效。

debug [off] 指定 **debug** 将可恢复虚拟共享磁盘子系统的标准输出和标准错误重定向到控制台，从而使可恢复虚拟共享磁盘子系统在因错误而退出时不会重新运行。（您可以使用 **lscons** 命令来确定当前的控制台。）

必须重新启动可恢复虚拟共享磁盘子系统使此操作数生效。

一旦打开调试并且重新启动了可恢复虚拟共享磁盘，就应该发出 **ha.vsd trace** 来打开跟踪。

请在 IBM 服务代表的指导下使用此操作数。

注: 在引导节点时的缺省值是将标准输出和标准错误路由到控制台上。如果关闭调试，标准输出和标准错误将路由到 **/dev/null**，并且将丢失所有进一步的跟踪消息。通过发出 **ha.vsd qsrc** 可以确定是否已打开调试。如果已打开调试，则返回值将是:

```
action = "2"
```

mksrc 使用 **mkssys** 来创建可恢复的虚拟共享磁盘子系统。

query 详细显示可恢复的虚拟共享磁盘子系统的当前状态。

quorum *n* 设置定额值，该值是在激活虚拟共享磁盘之前必须加入组中的总节点数。通常将定额定义成节点中的大多数，这些节点被定义成 RSCT 对等域中的虚拟共享磁盘节点，但是此命令允许您覆盖该定义。

发出此命令时，可恢复虚拟共享磁盘子系统必须处于活动状态。这不是持久的更改。

qsrc 显示可恢复虚拟共享磁盘守护程序的系统资源控制器（SRC）配置。

refresh [noquorum]

使用 **refresh** 命令对所有正在运行的可恢复虚拟共享磁盘子系统异步启动刷新协议。在进行刷新之前将复位定额，除非指定了 **noquorum**。请使用 **ha.vsd query**检查是否完成。以下各项在设备驱动程序中刷新:

1. 已添加或删除的节点
2. 已添加或删除的虚拟共享磁盘
3. 虚拟共享磁盘的已更改属性 size_in_MB

reset 停止并重新启动可恢复虚拟共享磁盘子系统。

reset_quorum

复位缺省定额。

rmsrc 使用 **rmssys** 除去可恢复虚拟共享磁盘子系统。

start 启动可恢复虚拟共享磁盘子系统。

stop 停止可恢复虚拟共享磁盘子系统。

trace [off] 请求或停止跟踪可恢复虚拟共享磁盘子系统。发出此命令时，可恢复虚拟共享磁盘子系统必须处于活动状态。

只有在 **debug** 操作数已用于将标准输出和标准错误发送到控制台，并且重新启动了可恢复虚拟共享磁盘子系统后，此操作数才有意义。

安全性

您必须有 **root** 权限才能运行该命令。

退出状态

0 标明命令已成功完成。

nonzero 显示有错误产生。

限制

必须从对等域中的联机节点上发出此命令。要使对等域联机，请使用 **startrpdomain** 命令。要使特定节点在现有对等域中联机，请使用 **startrpnode** 命令。有关创建和管理 RSCT 对等域的更多信息，请参考 *RSCT Administration Guide*。

示例

1. 要停止可恢复虚拟共享磁盘子系统然后重新启动它，请输入:

   ```
   ha.vsd reset
   ```

   系统返回如下消息:

正在等待 rvsd 子系统退出。rvsd 子系统已成功退出。正在启动 rvsd 子系统。rvsd 子系统启动了 PID=xxx。

2. 要将定额更改成 RSCT 对等域中的五个节点，请输入:

```
ha.vsd quorum 5
```

系统返回如下消息:

```
定额已从 8 更改成 5。
```

3. 要查询 rvsd 子系统，请输入:

```
ha.vsd query
```

系统显示类似于以下的消息:

```
Subsystem         Group            PID     Status
      rvsd              rvsd             18320   active
      rvsd(vsd): quorum= 9/4, active=1, state=idle, isolation=member,
                 NoNodes=10, lastProtocol=nodes_failing,
                 adapter_recovery=on, adapter_status=up,
                 RefreshProtocol has never been issued from this node,
                 Running function level 4.1.0.0.
```

其中:

quorum 是在激活虚拟共享磁盘之前必须加入组中的总节点数或服务器节点数。在上述系统输出中，quorum 9/4 表示节点总数为（9），服务器节点数为（4）。

active 表示要加入的组的激活状态:

0: 组是不活动的（不满足定额）。

1: 组是活动的，并且已激活共享磁盘。

state 表示当前正在运行的协议。

isolation 表示组的成员资格状态

isolated: 尚未提议组 “join”。

proposed: 已提议组 “join”。

member: 用户是组的成员（提供者）。

NoNodes 表示已加入组的节点数。

lastProtocol 表示在组中运行的上一个协议。

adapter_recovery

表示通信适配器的恢复支持:

on: 已启用适配器恢复。

off: 已禁用适配器恢复。

adapter_status

表示通信适配器的状态:

up: 适配器已启动。

down: 适配器已关闭。

unknown: 适配器状态未知。

RefreshProtocol ...

表示是否已从该节点发出刷新协议。如果发出，则会显示成功或出错的日期和时间。

Running function level

表示子系统正在运行的功能级别，采用版本，发行版，修改版，修订级别的格式（vrmf）。（较低级别的子系统共存可能会限制用户以下降的功能级别运行。）

位置

/opt/rsct/vsd/bin/ha.vsd

相关信息

命令：**ha_vsd**

---

ha_vsd 命令

用途

启动和重新启动可恢复虚拟共享磁盘子系统。这包括配置虚拟共享磁盘以及激活可恢复性子系统。

语法

ha_vsd [**reset**]

描述

在安装可恢复虚拟共享磁盘软件后，使用此命令来启动它，或者通过 **reset** 选项来停止然后重新启动此程序。

标记

–a 指定所有虚拟共享磁盘。

–v *vsd_name_list*
指定一个或多个虚拟共享磁盘名，用逗号分开。

–n *node_list* 指定一个或多个节点号，用逗号分开。

参数

reset 停止并重新启动可恢复虚拟共享磁盘子系统。

安全性

您必须有 **root** 权限才能运行该命令。

退出状态

0 标明命令已成功完成。

1 显示有错误产生。

限制

必须从对等域中的联机节点上发出此命令。要使对等域联机，请使用 **startrpdomain** 命令。要使特定节点在现有对等域中联机，请使用 **startrpnode** 命令。有关创建和管理 RSCT 对等域的更多信息，请参考 *RSCT Administration Guide*。

示例

要停止可恢复虚拟共享磁盘子系统然后重新启动它，请输入:

```
ha_vsd reset
```

位置

/opt/rsct/vsd/bin/ha_vsd

相关信息

命令: **ha.vsd**

---

haemd 守护程序

用途

观察由资源监视器更新的资源变量实例，产生并报道客户程序事件。

语法

haemd

描述

haemd（事件管理器）守护程序查看资源监视器更新的资源变量实例，并生成及向客户机程序报告事件。

一个 **haemd** 守护程序实例在集群的每一个节点上执行。**haemd** 守护程序受系统资源控制器（SCR）的控制。

由于受 SRC 的控制，该守护程序不能直接从命令行启动。它通常由 **emsvcsctrl** 命令启动。如果必须直接启动或停止该守护程序，请使用 **emsvcsctrl** 命令。

当 SRC 创建 **haemd** 守护程序时，实际启动的程序为 **haemd_HACMP**。**haemd_HACMP** 程序将首先收集该守护程序所需的信息，然后才运行 **haemd** 程序。换句话说，在 SRC 创建的进程中，**haemd_HACMP** 程序将被 **haemd** 程序替换。

有关该事件管理器守护程序的更多信息，请参阅 **emsvcsctrl** 命令。

具体实现

此守护程序是 AIX Reliable Scalable Cluster Technology（RSCT）文件集的一部分。

位置

/usr/sbin/rsct/bin/haemd **haemd** 守护程序所在的位置

相关信息

命令: **emsvcsctrl** 和 **haemd_HACMP**

haemd_HACMP 命令

用途
事件管理器守护程序的启动程序。

语法
haemd_HACMP [**-d** *trace_arg*]

描述
haemd_HACMP 命令是 **haemd** 守护程序的启动程序。当 **emsvcsctrl** 命令在系统资源控制器（SRC）中配置事件管理子系统时，**haemd_HACMP** 被指定为要启动的程序。

该程序只能由 SRC 调用。要启动事件管理子系统，请使用 **emsvcsctrl** 命令。

标志
-d *trace_arg*
只有在 IBM 技术支持中心指导下，才能使用该标志。可供使用的跟踪参数除 **reg** 和 **dinsts** 以外，其余的参数和 **haemtrcon** 命令相同。要使用此标志，必须使用带 **-a** 标志的 **chssys** 命令更改 SRC 中的 **emsvcs** 子系统定义。随后必须停止该守护程序并重新启动它。

限制
此命令仅在 HACMP 环境中有效。

具体实现
此命令是 Reliable Scalable Cluster Technology（RSCT）文件集的一部分。

位置
/usr/sbin/rsct/bin/haemd_HACMP
haemd_HACMP 程序所在的位置

相关信息
命令：**emsvcsctrl**、**haemd** 和 **haemtrcon**

haemqvar 命令

用途
查询资源变量。

语法
haemqvar [**-H** *domain* | **-S** *domain*] [**-c** | **-d** | **-i**] [**-f** *file*] [**-h**] [*class var rsrcID* [*"*]]

描述

haemqvar 命令是用来在事件管理子系统中查询资源变量信息的命令。缺省状态下，该命令将当前 SP 域（也就是由 SP_NAME 环境变量定义的当前的 SP 系统分区）中所有资源变量的定义写入标准输出。如果 SP_NAME 没有设置，则就使用缺省的系统分区。可以使用 -S 标志来指定另一个 SP 域（系统分区）。要在 HACMP 域中查询变量，请使用 -H 标志。对于 SP 域来说，域标志自变量就是系统分区名。对于 HACMP 域来说，域标志实参是 HACMP 集群名。-H 标志被指定后，必须在 HACMP/ES 集群的一个节点上执行该命令。

以下信息为每个资源变量定义报告:

- 变量名称
- 值类型
- 数据类型
- SBS 格式（当数据类型为结构字节字符串时）
- 原始值
- 类
- 定位器
- 变量描述
- 资源标识及对该标识的描述
- 缺省表达式（如果定义）及对该表达式的描述

鉴于该命令的缺省行为将生成大量的输出，需要将标准输出重定向到一个文件。

如果指定 -d 标志，则向标准输出中只写入资源变量名和简短描述，一行写一对名称和描述。

如果指定 -c 标志，则向标准输出写入所有资源变量实例的当前值，每个实例值占一行。输出行包括资源变量实例的位置（节点编号）、资源变量名、该实例的资源标识以及该资源变量实例的值。如果该资源变量是一个结构字节字符串（SBS）数据类型，则报告每个 SBS 字段的值。

-i 标志与 -c 标志报告的信息除了一点以外，其余全部相同。不同之处就是前者报告的变量实例值是最后得出的值，而不是当前值。因此，-i 标志可以用来确定存在的资源变量实例。

对于 -c 标志和 -i 标志来说，如果它们在获取资源变量实例的信息的过程中遇到错误，则输出行将包括错误消息、符号错误代码、发生错误的位置（如果能够确定的话）、资源变量名以及资源标识。

在返回特定的资源变量的信息时，需要指定操作数 class、var 和 rsrcID。这些操作数在指定附加的资源变量时可以反复使用。而且，为匹配多个资源变量，var 和 rsrcID 操作数还可以作为通配符使用。注: 空字符串操作数或星号外必须加引号。

如果操作数 class 不是一个空字符串，则指定的类中的所有的变量，经 var 和 rsrcID 变量进行进一步的限制后，成为查询的目标。如果操作数 class 是一个空字符串，则指定的类中的所有的变量，经 var 和 rsrcID 自变量进行进一步的限制后，成为查询的目标。可以以两种方式之一将 var 自变量作为通配符来使用:

1. 指定变量名为空字符串
2. 略去任何组件后面的名称

当资源变量名按照前一种方式使用通配符时，所有的资源变量，经 class 和 rsrcID 变量进行进一步的限制后，成为查询的目标。当资源变量名按照后一种方式使用通配符时，所有的高位（最左的）组件与 var 自变量相匹配的变量，经 class 和 rsrcID 变量进行进一步的限制后，成为查询的目标。

所有的资源变量实例，或不指定 -c 或者 -i 标志，由 class 和 var自变量指定的而与 rsrcID 自变量相匹配的变量的定义是查询的目标。

如果 -c 标志和 -i 标志都没有被指定，则 rsrcID 变量就是由分号隔开的资源标识元素名称列表。如果 -c 标志或 -i 标志中的一个被指定，则 rsrcID 变量就是由分号隔开的名称 / 值对列表。一个名称 / 值对包含一个资源标识元素名称，后接等号，再接资源标识元素值。元素值可能包含单一值、一组值、由逗号隔开的多个单一值的列表或者由逗号隔开的组值的列表。一组值采取 a-b 的格式，而且只在类型整数（类型信息可以从变量定义中获取）的资源标识元素中有效。资源标识中不能有空格。

资源标识元素使用通配符时，将其值指定为星号字符。只有被定义为包含元素，且其所包含的元素已经在 rsrcID 变量中进行指定的变量才是查询的对象。如果资源标识的任何元素中包含星号字符，而不是名称 / 值对（或者是一个名称，在查询定义时），凡是被定义为至少包含剩余的指定元素的变量就是查询的对象。如果资源标识只包含星号字符，则整个资源标识都被通配；所有的资源变量的所有实例，经 class 和 var 变量进行进一步的限制后，成为查询的目标。

注：如果 rsrcID 变量中包含分号或者星号，必须在其外面加引号。

操作数 class、var 和 rsrcID 可以被放入文件中，每组操作数占一行，而不必被指定为命令自变量。使用 -f 标志来指定命令的文件名。如果使用 -f 标志，则命令中的所有操作数都可以省略。在文件中，空字符串由两个相邻双引号表示。当资源标识完全由通配符表示时，可以是单一的星号（*）或者双引号中的星号（″*″）。每行中，自变量由空格或者制表符分隔。

下面是在 rsrcID 自变量中使用通配符的示例。在这些示例中，假设 class 和 var 自变量为空字符串。如果 class 和 var 自变量中的一个，或者它们两个同时都不是空字符串，则查询的对象将因此而有所限制。在前三个示例中，所有其资源标识被定义为包含且只包含元素 NodeNum、VG 和 LV 的变量都是相匹配的。

1. 在本示例中，只有一个实例是匹配的:

   ```
   NodeNum=5;VG=rootvg;LV=hd4
   ```

2. 本示例中，每个节点的一个实例是匹配的:

   ```
   NodeNum=*;VG=rootvg;LV=hd4
   ```

3. 在本示例中，所有的相匹配的资源变量的实例都是相匹配的:

   ```
   NodeNum=*;VG=*;LV=*
   ```

4. 在本示例中，凡是其资源标识被定义为只包含元素 NodeNub 的变量是相匹配的。相匹配的实例与节点 9 相关联:

   ```
   NodeNum=9
   ```

5. 在本示例中，同组的变量是相匹配的，但是每个变量的所有的实例都是相匹配的:

   ```
   NodeNum=*
   ```

6. 在本示例中，凡是其资源标识被定义为包含元素 NodeNum 和 VG 以及零单元或者更多的附加单元的变量都是相匹配的。相匹配的各实例与节点 9 相联:

   ```
   NodeNum=9;VG=*;*
   ```

7. 在本示例中，凡是其资源标识被定义为包含元素 NodeNum、零或者更多的附加元素的变量都是相匹配的。变量的所有实例都是相匹配的:

   ```
   NodeNum=*;*
   ```

假设可以为查询灵活地指定资源变量，那么没有匹配的资源变量实例或者资源变量定义也是可以的。如果没有出现任何上述或者下述形式的匹配错误消息的报告。

如果变量 class、var 或者 rsrcID 的规范出现了错误，则输出行将包含错误消息、符号错误代码以及指定的类名称、资源变量名和资源标识。

标志

-H *domain*
在由 *domain* 指定的 HACMP 域中查询资源变量。

-S *domain*
在由 *domain* 指定的 SP 域中查询资源变量。

-c 查询当前的资源变量值。

-d 查询资源变量定义，但是将同时生成简易格式的输出。

-i 查询资源变量实例。

-f *file* 查询在 *file* 中指定的资源变量。

-h 显示用法声明。

参数

class 指定资源变量类名称或者空字符串。

var 指定资源变量名称或者空字符串。

rsrcID 指定资源标识或星号。

安全性

您必须具有对 SDR 的 root 用户权限和写入权限才能运行此命令。

您应该在控制工作站上运行这个命令。在运行该命令前，您必须将 SP_NAME 这个环境变量设为正确的系统分区名。

退出状态

0 标明命令已成功完成。

1 标明有错误产生。产生了标明错误原因的一个或多个错误消息。

限制

此命令只在 PSSP 环境下有效。

标准输出

当命令执行成功时，写入以下信息:

```
Reading Event Management data for partition syspar_name

CDB=new_EMCDB_file_name Version=EMCDB_version_string
```

标准错误

必要时此命令将错误信息写到标准错误中。

示例

1. 要获取当前集群中所有的资源变量的定义，并将输出放入文件，请输入:

   ```
   haemqvar -H HAcluster > vardefs.out
   ```

2. 要在名为 HAcluster 的 HACMP 集群中获取所有资源变量（其资源标识包含元素 VG）的简短格式列表，请输入:

```
haemqvar -H HAcluster -d "" "" "VG;*"
```

3. 要获取其资源标识只包含元素 VG 和 NodeNum 的资源变量，请输入:

```
haemqvar -H HAcluster -d "" "" "VG;NodeNum"
```

位置

/usr/sbin/rsct/bin/haemqvar **haemqvar** 命令的位置

文件

/usr/sbin/rsct/install/config/haemloadlist
包含事件管理子系统的配置数据的缺省值

相关信息

命令: **haemcfg**、**SDRCreateObjects**、**SDRDeleteObjects**

文件: **haemloadlist**

有关事件管理配置数据库的系统数据库（SDR）的类和属性的信息，请参考*《RSCT 事件管理编程指南和参考》*

---

haemtrcoff 命令

用途

关闭事件管理器守护程序的跟踪.

语法

haemtrcoff -s *subsys_name* **-a** *trace_list*

描述

用 haemtrcoff 命令为事件管理器守护程序的指定活动关闭跟踪。将跟踪输出放在系统分区的事件管理跟踪日志中。

标志

-s *subsys_name*
指定事件管理子系统名。在一个节点上，这是 emsvcs。必须指定该参数。

-a *trace_list*
指定跟踪参数列表。每个参数指定活动类型，跟踪将针对该类型而关闭。至少必须指定一个参数。如果指定了一个以上的参数，则参数之间用逗号隔开。列表不包括空格。

参数

以下跟踪参数可以被指定:

init 停止跟踪事件管理器守护程序的初始化。

config 停止从配置文件转储信息。

insts 停止跟踪由守护程序处理的资源变量实例。

rmctrl 停止跟踪资源监视器控制。

cci 停止跟踪客户通信（内部）接口。

emp 停止跟踪事件管理器协议。

obsv 停止跟踪资源变量观察。

evgn 停止跟踪事件的发生与通知。

reg 停止跟踪事件的注册和注销。

pci 停止跟踪对等的通信（内部）接口。

msgs 停止跟踪守护程序收到以及发出的所有消息。

query 停止跟踪由守护程序处理的查询。

gsi 停止跟踪组服务（内部）接口。

eval 停止跟踪表达式求值。

rdi 停止跟踪可靠的守护程序（内部）接口。

sched 停止跟踪内部调度程序。

shm 停止跟踪共享存储器管理活动。

all 停止跟踪所有活动。

all_but_msgs
停止跟踪除消息外所有的活动。消息活动由 msgs 参数定义。

安全性

您必须要有 root 特权和对 SDR 的写权限才能运行此命令。

您应该在控制工作站上运行这个命令。在运行该命令前，您必须将 SP_NAME 这个环境变量设为正确的系统分区名。

退出状态

0 指示命令已成功完成。

1 显示有错误产生。如果存在一个或多个错误消息，则显示错误原因。

限制

在正常运行期间，别使用该命令。只有在 IBM 技术支持中心指导下，才能使用该命令。它提供了一些调试信息，并可能降低运行在系统分区中的事件管理子系统等的性能。

标准输出

当命令执行成功时，写入以下信息消息:

```
Reading Event Management data for partition syspar_name

CDB=new_EMCDB_file_name Version=EMCDB_version_string
```

标准错误

必要时此命令将错误信息写到标准错误中。

示例

1. 在一个集群节点上为事件管理子系统关闭所有跟踪，登录节点，请输入:

   ```
   haemtrcoff -s emsvcs -a all
   ```

2. 在一个集群节点上为事件管理子系统关闭所有初始化和配置的跟踪，登录该节点，请输入:

   ```
   haemtrcoff -s emsvcs -a init,config
   ```

位置

/usr/sbin/rsct/bin/haemtrcoff **haemtrcoff** 命令的位置

文件

/var/ha/log/em.trace.cluster_name
包括名为*cluster_name* 的集群上**haemd** 守护程序的跟踪日志。

/var/ha/log/em.msgtrace.cluster_name
包括来自名为 *cluster_name* 的集群上事件管理器守护程序的消息跟踪输出

相关信息

命令: **haemtrcon**

守护程序: **haemd**

脚本: **emsvcsctrl**

---

haemtrcon 命令

用途

为事件管理器守护程序打开跟踪.

语法

haemtrcon -s *subsys_name* **-a** *trace_list*

描述

haemtrcon 命令用来为事件管理器守护程序的指定活动打开跟踪。跟踪输出置于系统分区的事件管理跟踪日志中。使用时，**regs**、**dinsts**、**iolists** 和 **olists** 参数执行一次跟踪。指定的信息放在跟踪日志中，但不再进行进一步跟踪。

标志

-s *cluster_name*
指定事件管理子系统名. 在一个节点上，*cluster_name* 是 **emsvcs**。必须指定这个标志和参数。

-a *trace_list*
指定跟踪参数列表。每个参数指定活动类型，跟踪为该活动类型打开。至少要指定一个参数。如果两个以上的参数被指定，则参数之间要用逗号隔开。列表不包括空格。

参数

以下跟踪参数可以被指定:

init 跟踪事件管理器守护程序的初始化。

config 从配置文件中转储信息。

insts 跟踪由守护程序处理的资源变量实例。

rmctrl 跟踪资源监视器控制。

cci 跟踪客户机通信（内部）接口。

emp 跟踪事件管理程序协议。

obsv 跟踪资源变量观察。

evgn 跟踪事件的发生和事件的通知。

reg 跟踪事件的注册与注销。

pci 跟踪对等的通信（内部）接口。

msgs 跟踪到来的以及由守护程序发出的所有消息。

query 跟踪由守护程序处理的查询。

gsi 跟踪组服务（内部）接口。

eval 跟踪表达式求值。

rdi 跟踪可靠的守护程序（内部）接口。

sched 跟踪内部调度程序。

shm 跟踪共享存储管理活动。

all 跟踪所有的活动。

all_but_msgs
停止跟踪除消息外所有的活动。消息事件由 msgs 变量定义。

regs 跟踪当前注册的事件。

dinsts 跟踪守护程序知道的所有资源变量实例。

iolists 跟踪直接观察列表

olists 跟踪观察列表

限制

在正常运行期间，别使用该命令。只有在 IBM 技术支持中心指导下，才能使用该命令。它提供了一些调试信息，以及可能丢失事件管理子系统或在该系统分区中正在运行的其他子系统的一些性能。

具体实现

此命令是 Reliable Scalable Cluster Technology（RSCT）文件集的一部分。

示例

1. 在一个集群节点上为事件管理子系统打开所有跟踪，登录节点，输入:

   ```
   haemtrcon -s emsvcs -a all
   ```

2. 在一个集群节点上为事件管理子系统打开所有初始化和配置的跟踪，登录该节点，输入:

```
haemtrcon -s emsvcs -a init,config
```

位置

/usr/sbin/rsct/bin/haemtrcon **haemtrcon** 命令的位置

相关信息

命令: **haemtrcoff**

守护程序: **haemd**

脚本: **emsvcsctrl**

---

haemunlkrm 命令

用途

解锁和启动一个资源监视器。

语法

haemunlkrm **-s** *subsys_name* **-a** *resmon_name*

描述

如果事件管理守护程序在两个小时间隔内经过三次尝试后仍不能成功启动一个资源监视器，或在两个小时间隔内 *n* 次成功连接到某个资源监视器的实例，则“锁定”资源监视器，且不再尝试启动或连接到它的任一个实例上。在 HACMP/ES 集群中，*n* 是 **3**。一旦失败的原因确定下来，且问题已得到纠正，**haemunlkrm** 命令可用来解锁资源监视器并且尝试启动它或连接到一个资源监视器实例。

通过 **lssrc** 命令来显示事件管理器守护程序的状态，其状态表示是否锁定了某个资源监视器。

标志

-s *subsys_name*
: 指定事件管理子系统名。在一个节点上，*subsys_name* 是 **emsvcs**。必须指定这个标志和参数。

-a *resmon_name*
: 指定要解锁和启动的资源监视器的名字。

参数

以下跟踪参数可以被指定:

init 跟踪事件管理器守护程序的初始化。

config 从配置文件中转储信息。

insts 跟踪由守护程序处理的资源变量实例。

rmctrl 跟踪资源监视器控制。

cci 跟踪客户机通信（内部）接口。

emp 跟踪事件管理程序协议。

obsv 跟踪资源变量观察程序。

evgn 跟踪事件的发生和通知。

reg 跟踪事件的注册与注销。

pci 跟踪对等的通信（内部）接口。

msgs 跟踪到来的以及由守护程序发出的所有消息。

query 跟踪由守护程序处理的查询。

gsi 跟踪组服务接口程序（内部）。

eval 跟踪表达式求值。

rdi 跟踪可靠的守护程序（内部）接口。

sched 跟踪内部调度程序。

shm 跟踪共享存储管理活动。

all 跟踪所有的活动。

all_but_msgs
停止跟踪除消息外所有的活动。消息活动由 msgs 参数定义。

regs 跟踪当前注册的事件。

dinsts 跟踪守护程序知道的所有资源变量实例。

iolists 跟踪直接观察列表

olists 跟踪观察列表

安全性

您必须要有 root 特权和对 SDR 的写权限才能运行此命令。

您应该在控制工作站上运行这个命令。在运行该命令前，您必须将 SP_NAME 这个环境变量设为正确的系统分区名。

退出状态

0 指示命令已成功完成。

1 显示有错误产生。如果存在一个或多个错误消息，则显示错误原因。

限制

在正常运行期间，别使用该命令。只有在 IBM 技术支持中心指导下，才能使用该命令。它提供了一些调试信息，以及可能丢失事件管理子系统或在该系统分区中正在运行的其他子系统的一些性能。

标准输出

当命令执行成功时，写入以下信息性消息:

```
Reading Event Management data for partition syspar_name

CDB=new_EMCDB_file_name Version=EMCDB_version_string
```

标准错误

必要时此命令将错误信息写到标准错误中。

示例

1. 本示例应用于在一个节点上解锁一个资源监视器。

   如果 **lssrc** 命令的输出显示程序资源监视器 **IBM.PSSP.harmpd** 被锁住，纠正阻塞资源监视器被启动的条件并输入:

   ```
   haemunlkrm -s emsvcs -a IBM.PSSP.harmpd
   ```

位置

/usr/sbin/rsct/bin/haemunlkrm

haemunlkrm 命令的位置

文件

/var/ha/log/em.trace.*cluster_name*

包含名为 *cluster_name* 的集群上 hamed 守护程序的跟踪日志。

/var/ha/log/em.msgtrace.cluster_name

包含来自名为 *cluster_name* 的集群上事件管理器守护程序的消息跟踪输出。

相关信息

命令: **haemtrcoff**

守护程序: **haemd**

脚本: **emsvcsctrl**

---

hagsd 守护程序

用途

观察由资源监视器更新的资源变量实例，产生并报道客户程序事件。

语法

hagsd [**-a**] [**-s**] [**-k**] [**-d**] [**-c**] [**-u**] [**-t**] [**-o**] [**-r**] [**-h**] *daemon_name*

描述

hagsd 守护程序是组服务子系统的一部分， 它提供了一个通用功能来协调和监视在集群节点上运行的应用程序状态的变化。该守护程序提供子系统的大多数服务。*daemon_name* 指定守护程序用以命名日志文件并识别其在 AIX 错误日志中的消息的名称。

一个 **hagsd** 守护程序实例在每个集群节点上执行。**hagsd** 守护程序受系统资源控制器（SCR）的控制。

由于守护程序在 SRC 控制下，最好不要从命令行直接启动它。在正常情况下守护程序由 **grpsvcsctrl** 命令来调用，该命令由集群启动进程来依次调用。如果一定要直接启动或停止守护程序，则使用 **startsrc** 或 **stopsrc** 命令。

标志

–a 添加子系统。

–s 启动子系统。

–k 停止子系统。

–d 删除子系统。

–c 清除子系统，也就是说， 从所有系统分区中删除它们。

–u 解除子系统在所有系统分区中的配置。

–t 打开子系统跟踪。

–o 关闭子系统跟踪。

–r 刷新子系统。

–h 显示用法信息。

参数

deamon_name
指定守护程序用以命名日志文件并识别其在 AIX 错误日志中的消息的名称。

安全性

您必须有 **root** 权限才能运行该脚本。

退出状态

0 指示命令已成功完成。

1 显示有错误产生。

限制

此命令只在 PSSP 环境下有效。

标准输出

当 **-h** 标志被指定时， 该命令的用法说明写到标准输出中去。

标准错误

必要时此命令将错误信息写到标准错误中。

示例

1. 在当前系统分区中，将组服务子系统添加到 SRC 上， 设置 SP_NAME 环境变量为正确的系统分区名，输入:

   ```
   hagsctrl -a
   ```

2. 在当前系统分区中，启动组服务子系统， 设置 SP_NAME 环境变量到正确的系统分区名，输入:

   ```
   hagsctrl -s
   ```

3. 在当前系统分区中，停止组服务子系统，设置 SP_NAME 环境变量为正确的系统分区名，输入:

   ```
   hagsctrl -k
   ```

4. 在当前的系统分区中，从 SRC 上删除组服务子系统， 设置 SP_NAME 环境变量为正确的系统分区名，输入:

   ```
   hagsctrl -d
   ```

5. 在所有系统分区中，清除组服务子系统，输入:

   ```
   hagsctrl -c
   ```

6. 在控制工作站上，从所有系统分区，解除组服务子系统的配置，输入:

   ```
   hagsctrl -u
   ```

7. 在当前系统分区中，打开服务组 daemon 的跟踪程序， 设置 SP_NAME 环境变量为正确的系统分区名，输入:

   ```
   hagsctrl -t
   ```

8. 在当前系统分区中，关闭服务组 daemon 的跟踪程序， 设置环境变量 SP_NAME 为正确的系统分区名，输入:

   ```
   hagsctrl -o
   ```

位置

/usr/sbin/rsct/bin/hagsd 包含 **hagsd** 守护程序

文件

/var/ha/log/hags_nodenum_instnum**.**syspar_name
包含节点上 **hagsd** 守护程序的日志

/var/ha/log/hags.syspar_name_nodenum_instnum**.**syspar_name
包含控制工作站上的每一 **hagsd** 守护程序的日志。

文件名包含以下变量:

- *nodenum* 是有守护程序运行的节点号。
- *instnum* 守护程序的实例数目。
- *syspar_name* 是有守护程序运行的系统分区的名称。

相关信息

命令: **grpsvcsctrl**

---

hagsns 命令

用途

获得分组服务名称服务器信息。

语法

hagsns [**–h** *host*] [**–c**] **–g** *group_name*

hagsns [**–h** *host*] [**–c**] **–s** *subsystem_name*

hagsns [**–h** *host*] [**–c**] **–p** *subsystem_pid*

描述

使用 **hagsns** 命令来查询组服务名称服务器的状态.

标志

–c 将输出强制为 “仅英语”。如果没有指定 **–c** 标志，则对输出使用守护程序的语言环境。

–g *group_name*
指定一组子系统来为之获取状态。如果 *group_name* 变量没有包含在子系统对象类中，命令就不会成功。

–h *host*
指定为之获得名称服务器状态的主机。

–p *subsystem_pid*
指定 *subsystem_pid* 的一个特例以为之获得名称服务器的状态。

–s *subsystem_name*
指定一个可为之获得状态的子系统。*subsystem_name* 变量可能是实际的子系统名，也可能是该子系统的同义名。如果 *subsystem_name* 变量没有包含在子系统对象类中，命令就不会成功。

参数

daemon_name
指定守护程序用以命名日志文件和识别其在 AIX 错误日志中的消息的名称。

安全性

您必须有 **root** 权限才能运行该命令。

退出状态

0 显示命令成功地完成。

非零值 显示有错误产生。

限制

此命令只在 PSSP 环境下有效。

标准输出

当 **-h** 标志被指定时， 该命令的用法说明写到标准输出中去。

标准错误

必要时此命令将错误信息写到标准错误中。

示例

如果想从组服务子系统获得域信息，输入:

```
hagsns -c -s cthags
```

或

```
hagsns -s cthags
```

输出表示如下:

```
HA GS NameServer Status
NodeID=1.16, pid=14460, domainID=6.14, NS established,CodeLevel=GSLevel(DRL=8)
NS state=kCertain, protocolInProgress=kNoProtocol,outstandingBroadcast=KNoBcast
Process started on Jun 19 18:34:20, (10d 20:19:22) ago, HB connection took (19:14:9).
Initial NS certainty on Jun 20 13:48:45, (10d 1:4:57) ago, taking (0:0:15).
Our current epoch of Jun 23 13:05:19 started on (7d 1:48:23), ago.
Number of UP nodes: 12
List of UP nodes: 0 1 5 6 7 8 9 11 17 19 23 26
```

在本例中， **domainID=6.14** 意思是节点 6 是名称服务器（NS）节点。域标识由一个节点号和一个别名号组成。别名号是一个整数，每当启动组服务守护程序时它就加 1。**NS 已建立**意味着已建立名称服务器。

位置

/usr/sbin/rsct/bin/hagsns 包含 **hagsns** 命令

文件

/var/ha/log/hags_*nodenum_instnum*.*syspar_name*
包含节点上 **hagsd** 守护程序的日志

/var/ha/log/hags.*syspar_name_nodenum_instnum*.*syspar_name*
包含控制工作站上的每一个 **hagsd** 守护程序的日志。

文件名包含以下变量:

- *nodenum* 是有守护程序运行的节点号。
- *instnum* 是守护程序的实例号。
- *syspar_name* 是有守护程序运行的系统分区的名称。

相关信息

命令: **hagsvote**、**lssrc**、**nlssrc**

---

hagsvote 命令

用途

G 获得分组服务组表决信息

语法

hagsvote [**–h** *host*] [**–l**] [**–a** *argument*] [**–c**] **–g** *group_name*

hagsvote [**–h** *host*] [**–l**] [**–a** *argument*] [**–c**] **–s** *subsystem_name*

hagsvote [**–h** *host*] [**–l**] [**–a** *argument*] [**–c**] **–p** *subsystem_name*

描述

为分组服务使用 **hagsvote** 命令来查询表决协议的状态。

标志

–a 指定分组服务的组名。该组名不同于 **–g** 标志的组名。在这种情况下，从客户机的第一个加入协议的请求来创建这个组。

–c 请求分组服务表决信息的规范输出。输出用英文显示，与所安装的语言环境无关。如果没有指定 **–c**，守护程序的语言环境就会被用于输出。

–g *group_name*
指定一组子系统来为之获取状态。如果 *group_name* 变量没有包含在子系统对象类中，命令就不会成功。

–h *host*
指定所要获取状态的主机名。

–l 请求以『长』格式输出详细信息。

–p *subsystem_pid*
指定为其获取表决的 *subsystem_pid* 变量的一个特定实例。

–s *subsystem_name*
指定要表决的子系统。*subsystem_name* 变量可能是实际的子系统名，也可能是其同义名。如果 *subsystem_name* 变量没有包含在子系统对象类中，命令就不会完成。

参数

daemon_name
指定守护程序用以命名日志文件和识别其在 AIX 错误日志中的消息的名字。

安全性

必须有 **root 用户**特权才能运行此命令。

退出状态

0 指示命令已成功完成。

非零 显示有错误产生。

限制

此命令只在 PSSP 环境下有效。

标准输出

必要时此命令将错误信息写到标准错误中。

标准错误

必要时此命令将错误信息写到标准错误中。

示例

1. 要查看以长格式为 **SourceGroup** 分组而选取的协议的状态信息，请输入:

   ```
   hagsvote -ls cthags -a theSourceGroup (locale-dependent)
   ```

   输出表示如下:

```
Number of groups: 4
Group name [theSourceGroup] GL node [26] voting data:
GL in phase [1] of n-phase protocol of type [Join].
Local voting data:
Number of providers: 1
Number of providers not yet voted: 1 (vote not submitted).
Given vote: [No vote value] Default vote: [No vote value]
ProviderID Voted? Failed? Conditional?
[101/26] No No Yes
Global voting data:
Number providers not yet voted: 1
Given vote: [No vote value] Default vote: [No vote value]
Nodes that have voted: []
Nodes that have not voted: [26]
```

输出的第一行表示分组总数为 4。第二行提供了组名和组的引导符节点（在本例中是节点 26）。其余行给出以下表决数据:

- 组引导符位于 n 阶段协议的第一阶段。
- 协议为联合协议。
- 对于本地节点，有一个供应商，还没有表决的供应商的数目为 1。
- 不给定缺省表决值，也不给定表决值。
- 在 “ProviderID Voted? Failed? Conditional?” 行下， “[101/16] No No Yes,” 意味着提供程序标识为 101/26，尚未表决，未失败，但等待表决（因此它是有条件的）。

输出然后就会显示全局表决状态:

- 没有表决的供应商数目仍为 1。
- 没有给定表决值，也没有给定缺省表决值。
- 已表决的节点为零。
- 没有表决的节点为节点 26。

2. 在下面的示例里，输出的每一行的意思除了节点 26 是组引导符节点外都与第一个示例相同。

```
hagsvote -ls cthags -theSourceGroup -c（canonical form）
```

输出表示如下:

```
Number of groups: 4
Group Name: theSourceGroup
GL Node: 26 (I am GL)
Current phase number of an n-phase protocol: 1
Protocol name: [Join]
Local voting data:
Number of local providers: 1
Number of local providers not yet voted: 1 (vote not submitted)
Given vote: [No vote value] Default vote: [No vote value]Global voting data:
Number of nodes in group: 1
Number of global providers not yet voted: 1
Given vote: [No vote value] Default vote: [No vote value]
Nodes that have voted: []
Nodes that have not voted: [26]
```

位置

/usr/sbin/rsct/bin/hagsvote 包含 **hagsvote** 命令

文件

/var/ha/log/hags_nodenum_instnum**.**syspar_name
包含节点上 **hagsd** 守护程序的日志

/var/ha/log/hags.syspar_name_nodenum_instnum**.**syspar_name
包含控制工作站上的每一个 **hagsd** 守护程序的日志。

文件名包含以下变量:

- *nodenum* 是有守护程序运行的节点号。
- *instnum* 是守护程序的实例号。
- *syspar_name* 是有守护程序运行的系统分区的名字。

相关信息

命令:**hagsns**、**lssrc**、**nlssrc**

---

halt 或 fasthalt 命令

用途

停止处理器。

语法

{ **halt** | **fasthalt** } [**-l**] [**-n**] [**-p**] [**-q**] [**-y**]

描述

halt 命令将数据写到磁盘上并停止处理器运行。机器不重新启动。只有 root 用户可以使用此命令。如果别的用户登录进系统，就不能使用此命令。如果没有别的用户登录， 就可以使用 **halt** 命令。如果您不想立即重新启动机器，可以使用 **halt** 命令。当显示 `....Halt completed....` 消息时，您可以关闭电源。

halt 命令使用 **syslogd** 命令记录关机，且将一个关机记录放在 **/var/adm/wtmp** 登录记帐文件中。系统也将一个条目写入错误记录里，声明系统已关机。

fasthalt 命令通过调用 **halt** 命令使系统停止。**halt** 命令提供 BSD 兼容性。

标志

-l 不在记帐文件中记录停机。**-l** 标志并不阻碍记帐文件更新。**-n** 与 **-q** 标志暗示 **-l** 标志。

-n 停止前要防止 **sync**。

-p 没有关闭电源而使系统停机。
注:如果与无需永久停机的标志一起使用，-p 标志不会产生影响。如果别的操作数请求延迟开机和重新引导，电源仍然会关闭。

-q 引起快速停机。
注: 运行带有 **-q** 标志的 **halt** 命令不会发出 **sync**，因此系统会立即停机。

-y 用拨号操作使系统停机。

示例

1. 要使系统停机而不在记帐文件中记录停机，请输入:

   ```
   halt -l
   ```

2. 要使系统迅速停机，请输入:

   ```
   halt -q
   ```

3. 要通过拨号使系统停机，请输入:

   ```
   halt -y
   ```

文件

/usr/bin/from	指定系统启动脚本。
/var/adm/wtmp	指定登录记帐文件。

相关信息

fastboot 命令、**fsck** 命令、**rc** 命令、**shutdown** 命令、**sync** 命令。

syslogd 守护程序。

---

hangman 命令

用途

启动 hangman 猜字游戏。

语法

hangman [*File*]

描述

hangman 命令从一本标准字典里选取一个至少 7 个字母的单词。*File* 参数指定一本备用字典。通过一次猜一个字母来猜单词。您可以犯 7 次错误。

当您启动 hangman，游戏就会显示:

```
guesses: word: ....... errors: 0/7guess:
```

guesses 显示了您猜时所用的字母。您所猜的每一个字母列在guess 后面。word: .... 显示了谜语单词中的字母个数。在本例中有 7 个 .，因此单词有 7 个字母。当您猜对时， 游戏就用正确的字母代替了适当的 .。errors: 0/7 显示了错误猜测的个数。您在 guess: 提示符处输入您所猜的字母。例如:

```
guesses: word: .......... errors: 0/7
guess: q
guesses: q word: .......... errors: 1/7
guess: a
guesses: aq word: .a....a... errors: 1/7
guess: b
guesses: abq word: .a....a... errors 2/7
guess: j
guesses: abjq word: .a....a... errors: 3/7
guess: s
guesses: abjqs word: .a....a..s errors: 3/7
guess: z
```

```
guesses: abjqsz word: .a....a..s errors: 4/7
guess: y
guesses: abjqsyz word: .a....a..s errors: 5/7
guess: k
guesses: abjkqsyz word: .a....a..s errors: 6/7
guess: x
the answer was calculates, you blew it
```

退出游戏，请按中断（Ctrl-C）或文件结束（Ctrl-D）按键顺序。

文件

/usr/games 系统游戏的位置。

相关信息

arithmetic 命令、**back** 命令、**bj** 命令、**craps** 命令、**fish** 命令、**fortune** 命令、**moo** 命令、**number** 命令、**quiz** 命令、**ttt** 命令、**turnoff** 命令、**turnon** 命令和 **wump** 命令。

---

hatsoptions 命令

用途

在一个节点或一个控制工作站上控制拓扑服务选项。

语法

hatsoptions [**-s**] [**-d**]

描述

在此命令被执行之前，环境变量 HB_SERVER_SOCKET 必须被设置到被拓扑结构服务子系统所用的 UNIX 域套接字的位置。可使用下面的语句:

```
export HB_SERVER_SOCKET=/var/ha/soc/hats/server_socket.partition name
```

作为选择，变量 HA_SYSPAR_NAME 可被设置为分区名。

为使此命令成功执行，必须运行拓扑服务守护程序。

hatsoptions 可被用来控制拓扑服务中的许多选项。选项 **-s** 指示拓扑服务守护程序拒绝那些明显延迟的消息。这可在一个有时在网络或发送器与接收器节点间消息发生延迟的大系统配置中使用。仅当所有节点和控制工作站上的计时时钟都同步时才使用此选项。否则当发送器的计时时钟落后于接收器的计时时钟时，报文可能被错误地废弃。

选项 **-d** 指示拓扑服务守护程序不得拒绝那些明显延迟的报文。这是缺省值。

标志

–s 指示拓扑服务守护程序拒绝那些明显延迟的报文。

–d 指示拓扑服务守护程序不得拒绝那些明显延迟的报文。(此为缺省设置)。

安全性

必须有 **root 用户**特权才能运行此命令。

退出状态

0 指示命令已成功完成。

1 指示发生错误。

环境变量

HB_SERVER_SOCKET

此环境变量应在命令执行前进行设置。它必须被设置为被用来连接拓扑服务守护程序的拓扑服务客户机所用的 UNIX 域套接字的位置。此环境变量必须设置为 **/var/ha/soc/hats/server_socket.***partition name*。

HA_SYSPAR_NAME

如果 HB_SERVER_SOCKET 没有进行设置，则 HA_SYSPAR_NAME 必须被设置为分区名。

限制

此命令仅在对等域中有效。

标准输出

当 **-h** 标志被指定时， 该命令的用法说明写到标准输出中去。所有详细消息都写入标准输出中。

标准错误

必要时此命令将错误信息写到标准错误中。

示例

要为指示本地节点上的拓扑服务守护程序开始废弃明显延迟的报文，输入:

```
export HA_SYSPAR_NAME=partition1

/usr/sbin/rsct/bin/hatsoptions -s
```

位置

/usr/sbin/rsct/bin/hatsoptions

包含 **hatsoptions** 命令

文件

/var/ha/soc/hats/server_socket.*partition name*

相关信息

命令: **hatsctrl**、**hats**、**lssrc**、**startsrc**、**stopsrc**、**syspar_ctrl**。

hash 命令

用途
记住或报告命令路径名。

语法

将命令路径名添加到路径名列表:
hash [*Command* ...]

清除路径名列表:
hash -r

描述
hash 命令通过添加路径名到列表或清除列表内容影响当前 shell 记住命令路径名的方式。

当不指定参数或标志时，**hash** 命令向标准输出报告路径名列表的内容。此报告包含由先前 **hash** 命令调用找到的当前 shell 环境中命令的路径名。此显示还包含通过正常命令搜索进程调用并找到的那些命令。

> 注: **hash** 命令不报告 shell 内置命令。

可以使用 **-r** 标志清除命令路径名列表的内容。还可以通过复位 **PATH** 环境变量的值从列表清除路径名。用最简单的格式，这可以通过输入以下命令完成:

```
PATH="$PATH"
```

如果使用了 *Command* 参数，则 **hash** 命令搜索指定命令的路径名并将该路径添加到列表。当指定命令时，请勿使用 /（斜杠）。

因为 **hash** 命令影响当前 shell 环境，所以它作为 Korn shell 或 POSIX shell常规内置命令提供。如果 **hash** 命令在独立的命令执行环境中调用，如以下示例所示，则它不会影响调用者环境的命令搜索进程:

```
nohup hash -r
find . -type f | xargs hash
```

使用 **hash** 命令等同于使用 **alias -t** 命令。

标志

-r 清除路径名列表的内容。

参数

Command 指定添加到路径名列表的 *Command*。

退出状态
返回以下出口值:

0 成功完成。

>0 发生错误。

示例

1. 要找到 **wc** 命令的路径名并将其添加到路径名列表，请输入:

   ```
   hash wc
   ```

2. 要清除路径名列表的内容，请输入:

   ```
   hash -r
   ```

文件

/usr/bin/ksh	包含 Korn shell **hash** 内置命令。
/usr/bin/hash	包含 **hash** 命令。

相关信息

alias 命令、**bsh** 命令、**ksh** 命令。

---

head 命令

用途

显示文件的头几行。

语法

head [**-** *Count* | **-c** *Count* | **-n** *Number*] [*File* ...]

描述

head 命令将每一个指定文件或标准输入的指定数量的行或字节写入标准输出。如果不为 **head** 命令指定任何标志， 缺省显示前 10 行。*File* 参数指定了输入文件名。输入文件必须是文本文件。当指定多个文件时每一文件的开始应与以下一致。

```
==> filename <==
```

要显示一组短文件并对每一个文件进行识别，请输入:

```
example% head -9999 filename1 filename2...
```

标志

-*Count*	从每一个要显示的指定文件的开头指定行数。*Count* 变量必须是一个正的十进制整数。此标志等价于 **-n** *Number* 标志，但如果考虑可移植性，就不应该使用。
-c *End*	指定要显示的字节数。*Number* 变量必须是一个正的十进制整数。
-n *Number*	从每一个要显示的指定文件的开头指定行数。*Number* 变量必须是一个正的十进制整数。此标志等价于 **-** *Count* 标志。

退出状态

此命令返回以下出口值:

0 成功完成。
>0 发生错误。

示例

要显示 Test 文件的前 5 行，请输入:

```
head -5 Test
```

或者

```
head -n 5 Test
```

相关信息

tail 命令。

《*操作系统与设备管理*》中的『文件』。

《*操作系统与设备管理*》中的『输入和输出重定向』。

---

help 命令

用途

为新用户提供信息。

语法

帮助

描述

help 命令为新用户显示一页信息。信息可用于以下主题:

- 连接或显示文件。
- 交互式行编辑。
- 发送和接收邮件。
- 读系统消息。
- 更改密码文件信息。
- 识别系统当前用户。
- 为系统其他用户发送报文。
- 显示目录内容。
- 在源代码控制系统中查看信息。
- 设置终端方式。

示例

要获取帮助，请在命令行输入 help。

相关信息

cat 命令、**ex** 命令、**finger** 命令、**ls** 命令、**mail** 命令、**passwd** 命令、**sccshelp** 命令、**tset** 命令、**who** 命令、**write** 命令。

---

host 命令

用途

将一个主机名解析到一个因特网地址或将一个因特网地址解析到一个主机名。

语法

host [**-n** [**-a**] [**-c** *Class*] [**-d**] [**-r**] [**-t** *Type*] [**-v**] [**-w**] [**-z**]] *Hostname* | *Address* [*Server*]

hostnew [**-a**] [**-c** *Class*] [**-d**] [**-r**] [**-t** *Type*] [**-v**] [**-w**] [**-z**] *Hostname* | *Address* [*Server*]

描述

/usr/bin/host 命令返回一个主机的因特网地址（当 *Hostname* 参数被指定时），或返回主机名（当 *Address* 参数被指定时）。**host** 命令可能还显示与 *HostName* 参数关联的任何别名，这取决于名称解析服务的配置。名称解析服务的示例包含 **local**、**nis** 和 **bind**。

如果本地主机使用域名协议，则在搜索本地 **/etc/hosts** 文件之前先查询本地或远程名称服务器数据库。

host 命令也可返回在 DNS（域名系统）找到的其他名称记录。**-z** 标志指定这种方式。其他标志允许查询定制。

标志

-a 等价于使用 “**-v -t ***”

-c *Class* 当搜索非因特网数据时要指定要查找的类。有效类为:

IN 因特网类

CHAOS
Chaos类

HESIOD
MIT Althena Hesiod 类

ANY 通配符（以上任意一个）

***** 通配符（以上任意一个）

-d 打开调试方式。

-n 等价于发布 **/usr/bin/hostnew** 命令。**hostnew** 命令是 **host** 命令的 5.2 版。

-r 禁用递归处理。

-t *Type* 指定要查询的记录类型。有效类型为:

A 主机的因特网地址

CNAME
别名的规范名称

HINFO 主机 CPU 与操作系统类型

KEY 安全密钥记录

MINFO 邮箱或邮件列表信息

MX 邮件交换器

NS 指定范围的名称服务器

PTR 如果查询的是一个因特网地址则为主机名; 否则, 为其他信息的指针

SIG 签名记录

SOA 域的"授权开始"信息

TXT 文本信息

UINFO 用户信息

WKS 所支持的众所周知的服务。

-v 详细方式。

-w 永远等待 DNS 服务器的一个回答。

-z 使用显示资源记录信息的新输出。

参数

Address 说明用来解析主机名的主机的因特网地址。*Address* 参数必须是一个有效的因特网地址, 以加点的十进制格式表示。

HostName 说明要用来解析因特网地址的主机名称。*HostName* 参数可以是一个唯一的主机名, 也可以是一个众所周知的主机名 (诸如 *nameserver*、*printserver*、*timeserver* , 如果这些存在的话)。

Server 指定要查询的名称服务器。

示例

1. 要显示主机名为 mephisto 的主机地址, 请输入:

   ```
   host mephisto
   ```

   就会显示与以下类似的信息:

   ```
   mephisto is 192.100.13.5, Aliases: engr, sarah
   ```

2. 要显示地址为 192.100.13.1 的主机名, 请输入:

   ```
   host 192.100.13.1
   ```

   就会显示与以下类似的信息:

   ```
   mercutio is 192.100.13.1
   ```

3. 要显示名为 test.ibm.com 的域的 MX 记录, 请输入:

   ```
   host -n -t mx test.ibm.com
   ```

   或者

   ```
   hostnew -t mx test.ibm.com
   ```

就会显示与以下类似的信息:

```
test.ibm.com mail is handled (pri=10) by test1.tt.ibm.com  test.ibm.com mail is handled (pri=10) by
 test2.aix.ibm.com
```

文件

/etc/hosts 包含因特网协议（IP）名和在本地网络上的主机地址。

相关信息

hostname 命令

named 守护程序。

《*网络与通信管理*》中的『通信和网络』。

---

hostent 命令

用途

在系统配置数据库中直接操作地址映射条目。

语法

要添加地址到主机名的映射

hostent -a *IPAddress* **-h** *"HostName..."*

要删除一个地址到主机名的映射

hostent -d *IPAddress*

要删除所有地址到主机名的映射

hostent -X

要更改地址到主机名的映射

hostent -c *IPAddress* **-h** *"HostName..."* [**-i** *NewIPAddress*]

要在 Colon 格式下显示地址或主机名

hostent -s { *IPAddress* | “*HostName*” } [**-Z**]

要在 Colon 格式下显示所有地址到主机名的映射

hostent -S [**-Z**]

描述

在系统配置数据库中，**hostent** 低级命令添加、删除或更改地址映射条目。使用数据库中的条目映射因特网协议（IP）地址（本地或远程）到其等价的主机名。

hostent 命令可以在 **/etc/hosts** 文件中显示一个或所有地址到主机名的映射条目。某个指定的本地或远程主机的因特网协议（IP）地址或许与一个或多个主机名相关联。用点式十进制格式表示 IP 地址。用最长不超过 255 个字符的字符串表示主机名，并不使用空字符。每个条目必须包含在一行中。多重*主机名*（或别名）可以被指定。

注: 有效主机名或主机别名必须包含至少一个字母字符。如果您选择指定以 x 开头，后跟任意一个十六进制的数字（0-f）的主机名或别名，则主机名或别名必须还包含至少一个另外的不能用一个十六进制数字表示的字母。系统将以 x 开头，后跟一个十六进制数字的形式解释成某个地址的十六进制表示，除非在主机名或别名中存在至少一个不是十六进制数字的字母。由此，xdeer 将是一个有效的主机名，而 xdee 则不是。

在基于 Web 的系统管理器（wsm）下，您可以使用系统应用程序来更改系统特征。您也可以使用系统管理界面程序（SMIT）**smit hostent** 快速路径来运行该命令。

标志

注: 标志 **-a**、**-d**、**-c**、及 **-s** 不能一起使用。

-a *IPAddress*	对于数据库中指定的因特网协议地址，添加 IP 地址到主机名的映射条目。用 **-h** 标志指定主机名。
-c *IPAddress*	在数据库中更改 IP 地址到主机名称的映射条目，该地址与 *IPAddress* 变量指定的地址一致。用 **-h** 标志指定更改的主机名。如果要将当前的 IP 地址更改为一个新地址（*IPAddress*），则使用 **-i** 标志。
-d *IPAddress*	在数据库中删除 IP 地址到主机名的映射条目，该地址与由 *IPAddress* 变量指定的地址一致。
-h*"HostName..."*	指定主机名列表。列表中的条目由空格分开。**-h***"HostName..."* 标志应该与 **-a** 标志搭配使用。**-c** 标志或许也需要 **-h** "*HostName...*" 标志。
-i *NewIPAddress*	指定新的 IP 地址。如果一个现有 IP 地址要被 *NewIPAddress* 变量替代，则需要此标志与 **-c** 标志一起使用。
-S	显示数据库中的所有条目。
-s "*HostName*"	显示 IP 地址到主机名的映射条目，其主机名与由 " *HostName*" 参数指定的主机名匹配。
-s *IPAddress*	显示 IP 地址到主机名的映射条目，其与由 *IP 地址*变量指定的条目匹配。
-X	删除数据库中所有的 IP 地址到主机名的映射条目。
-Z	在 colon 格式下生成查询输出。当从 SMIT 可用性界面调用 **hostent** 命令时，使用该标志。

注: 此 **hostent** 命令可以认出以下地址: .08、.008、.09 及 .009。首位为零的地址被解释为八进制，同时八进制数字不能包括 8s 或 9s。

示例

1. 要在将地址与一系列主机名相关联的数据库中添加条目，请按照以下格式输入命令:

   ```
   hostent  -a 192.100.201.7  -h "alpha bravo charlie"
   ```

   在示例 1 中，IP 地址 192.100.201.7 被指定为主机地址，该地址具有一个主要的主机名 alpha 及同义名 bravo 和 charlie。

   注: 如果您想使用 .08、.008、.09、或.009 来添加地址，您将获得一个错误消息，提示 "IPAdress地址 已经存在，" 尽管地址并不实际存在于 **/etc/hosts** 文件中。

2. 要在与主机名匹配的数据库中显示条目，请按照以下格式输入命令:

   ```
   hostent  -s alpha
   ```

在示例 2 中，条目将显示与主机名 alpha 匹配。

3. 要更改一个条目的 IP 地址为新的 IP 地址，请输入:

```
hostent  -c 192.100.201.7  -i 192.100.201.8
```

在示例 3 中，原有的 IP 地址为 192.100.201.7，更改后的新地址为 192.100.201.8。

文件

/etc/hosts 包括网络的主机名和地址。

相关信息

hostname 命令

《*网络与通信管理*》中的『TCP/IP 名称解析』。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

---

hostid 命令

用途

设置或显示当前本地主机的标识。

语法

/usr/sbin/hostid [*HexNumber* | *InternetAddress* | *HostName*]

描述

/usr/sbin/hostid 命令以一个十六进制数字，显示当前本地主机的标识（或者是唯一的主机名，或者是数值参数）。该数值应在所有主机中具有唯一性，同时通常被设置为由*因特网地址*或*主机名*参数指定的主机地址。root 用户可以通过为 *HexNumber*、*InternetAddress* 或 *HostName* 参数指定一个十六进制的数设置 **hostid** 命令。**/etc/rc.net** 文件设置主机标识符为主机名。

参数

HexNumber	指定一个唯一的十六进制数字来表示当前本地主机。
InternetAddress	指定一个因特网地址来表示当前本地主机。
HostName	指定一个符号名称来映射唯一的主机。

示例

1. 要使用 **hostid** 命令，为本地因特网地址设置本地主机标识符时，请按照以下格式输入命令:

```
hostid   192.9.200.3
0xc009c803
```

hostid 命令将因特网地址 192.9.200.3 转变为一个十六进制的表示 0xc009c803，然后将本地主机（您连接到网络的工作站）设置为此地址。

2. 要显示本地主机的标识符，请输入:

```
hostid
0xc009c803
```

hostid 命令以一个十六进制数字显示主机的标识符。

相关信息

hostname 命令

gethostid 子例程、**sethostid** 子例程。

rc.net 文件格式。

《*网络与通信管理*》中的『TCP/IP 寻址』。

---

hostmibd 守护程序

用途

启动 **hostmibd** dpi2 子代理守护程序作为后台进程。

语法

hostmibd [**-f** *File*] [**-d** [*Level*]] [**-h** *Hostname*] [**-c** *Community*]

描述

hostmibd 命令启动 **hostmibd** dpi2 子代理。这个命令只能由具有根特权的用户或系统组的成员发出。

hostmibd 守护程序遵守由 RFC 1592 所定义的简单网络管理协议分布式协议接口第 2.0 版。它承担着 dpi2 子代理与 dpi2 代理之间的通信工作，通信渠道是在 RFC1592 的 3.1 部分中所定义的 dpiPortForTCP.0 (1.3.6.1.4.1.2.2.1.1.1.0)。

管理信息库（MIB）是由 RFC 1155 定义的。正被 hostmibd 管理中的这个特殊的 MIB 变量是由 RFC 1514 定义的。目前 MIB 变量的管理由 **hostmibd** 按以下的 4 个子树进行:

- hrSystem (1.3.6.1.2.1.25.1)
- hrStorage (1.3.6.1.2.1.25.2)
- hrDevice (1.3.6.1.2.1.25.3)
- hrSWInstalled (1.3.6.1.2.1.25.6)

hostmibd 守护程序的执行通常是在系统启动时调用了 **/etc/rc.tcpip**shell 脚本。

hostmibd 守护程序应使用系统资源控制器（SRC）来进行控制。输入 **hostmibd** 到命令行中是不被推荐使用的。

使用以下 SRC 命令来操作 **hostmibd** 守护程序:

startsrc
启动一个子系统，子系统组，或是一个子服务器。

stopsrc
停止一个子系统，子系统组，或是一个子服务器。

refresh
促使子系统或子系统组重读相应的配置文件。

lssrc 获取一个子系统，子系统组，或是一个子服务器的状态。如果用户发出了长状态格式的 **lssrc** 命令，但其不是 root 用户，则不会显示公用名。

标志

-c *Community* 使用指定的公用名。如果 **-c** 标志没指定，则缺省公用名为 “public “。

-d *Level* 指定跟踪/调试级别。级别分为:

- 0 = 最小级别
- 8 = DPI 级别 1
- 16 = DPI 级别 2
- 32 = 内置级别 1
- 64 = 内置级别 2
- 128 = 内置级别 3

为多重跟踪级别添加数字。缺省级别为 56，如果当 **-d** 标志被指定，而 *Level* 没被指定时。如果 **-d** 标志没被指定，则缺省级别为 0。

-f *File* 指定一个非缺省配置文件。如果 **-f** 标志没被指定，缺省配置文件为 **/etc/hostmibd.conf**。请参阅 **/etc/hostmibd.conf** 文件了解此文件格式的信息。

-h *Host* 向指定的主机发送请求。如果 **-h** 标志没指定，缺省目标主机为 “loopback” （127.0.0.1）。

示例

1. 要启动 **hostmibd** 守护程序，请仿照以下内容输入命令:

   ```
   startsrc -s hostmibd -a "-f /tmp/hostmibd.conf"
   ```

   这个命令启动了 **hostmibd** 守护程序并且从 **/tmp/hostmibd.conf** 读取了配置文件.
2. 要停止 **hostmibd** 守护程序，通常输入:

   ```
   stopsrc -s hostmibd
   ```

   这个命令停止了 **hostmibd** 守护程序。**-s** 标志指定了随后被停止的子系统。
3. 要从 **hostmbid** 获取短状态，请输入:

   ```
   lssrc -s hostmbid
   ```

   该命令会返回守护程序的名称、过程标识及其状态（处于活动状态还是非活动状态）。
4. 要从 **hostmibd** 守护程序获取长状态，请输入:

   ```
   lssrc -ls hostmibd
   ```

   如果是 root 用户，则这个状态报告的长表单就会将配置参数列入 **/etc/hostmibd.conf**。

文件

/etc/hostmibd.conf 为 **hostmibd** 命令定义配置参数。

/etc/mib.defs 定义简单网络管理协议代理和管理器能够识别和处理的管理信息库（MIB）变量。

相关信息

snmpdv3 守护程序，**snmpmibd** 守护程序。

---

hostname 命令

用途

设置或显示当前主机系统的名称。

语法

/usr/bin/hostname [*HostName*] [**-s**]

描述

/usr/bin/hostname 命令显示当前主机系统的名称。只有得到 root 用户权限的用户才能设置主机名。**mkdev** 命令和 **chdev** 命令同样可以永久设置主机名。当您正在第一次定义 TCP/IP 实例时，使用 **mkdev** 命令。

您可以在基于 Web 的系统管理器（wsm）下使用系统应用程序，以更改系统特征。您也可以使用系统管理界面程序（SMIT）**smit mkhostname** 快速路径来运行该命令。

标志

-s 从打印名称中获取任何域信息。

参数

HostName 设置主机的主名称

注: 您必须有 root 用户权限，才能使用*主机名*参数。

相关信息

chdev 命令、**mkdev** 命令。

gethostname 子例程、**sethostname** 子例程。

有关安装基于 Web 的系统管理器的信息，请参阅《*AIX 5L V5.3 基于 web 的系统管理器管理指南*》中的『第二章: 安装与系统需求』。

《*网络与通信管理*》中的『TCP/IP 名称解析』。

---

hosts2ldif 命令

用途

从一个主机文件中创建一个 LDAP 数据交换格式（LDIF）文件。

语法

hosts2ldif [**-i** *InputFile*] [**-o** *OutputFile*] [**-s** *SearchBase*]

描述

/usr/sbin/hosts2ldif 命令从 **/etc/hosts** 或别的类似于 **/etc/hosts** 的文件创建一个 LDAP 数据交换格式（LDIF）文件。没有标志，**/etc/hosts** 文件被用来创建 **/tmp/hosts.ldif** LDIF 文件，用 cn=hosts 作为基础域名。

使用这个命令创建的 LDIF 文件遵照 SecureWay® Directory Schema，可用来建立 **ldap** 机制。可支持 **ldap** 机制，推荐使用的是 **nis_ldap** 机制而不是 **ldap** 机制。

标志

-i *InputFile* 指定用于输入的主机文件。
-o *OutputFile* 指定用于输出的 LDIF 文件。
-s *SearchBase* 指定 LDAP 服务器上的主机表的基础域名。

示例

1. 要从 **/etc/hosts** 文件创建 **/home/ldifhosts**，请输入:

   ```
   hosts2ldif -o /home/ldifhosts
   ```

2. 要从 **/home/hosts.bak** 文件创建 **/tmp/hosts.ldif**，请输入:

   ```
   hosts2ldif -i /home/hosts.bak
   ```

3. 要使用 cn=hosttab 作为基础域名从 **/etc/hosts** 文件创建 **/home/ldifhosts**，请输入:

   ```
   hosts2ldif -o /home/ldifhosts -s cn=hosttab
   ```

文件

/etc/hosts 包含在本地网络上的主机的因特网协议（IP）名称和地址。

相关信息

《*网络与通信管理*》中的『TCP/IP 名称解析』。

---

hp 命令

用途

为 HP2640- 和 HP2621- 系列终端处理特定功能。

语法

hp [**-e**] [**-m** ...]

描述

hp 命令读取标准输入（输出通常来自 **nroff** 命令），并写到通常是 Hewlett-Packard 2640 和 2621 系列终端屏幕的标准输出。

如果您的终端具有显示增强功能，您可以显示下标字符和上标字符。使用数学符号功能，您可以正常地显示希腊字符和其他特殊字符（有两个例外）。**hp** 命令用右箭头来近似地表示逻辑运算符 NOT，并仅能显示完整符号的上半部。

叠印字符是后跟一个退格符和其他字符的字符。如果重写字符或退格符后输入的字符是一个加了下划线的字符，叠印字符会显示出加下划线或者以反转视频显现（取决于终端增强性能）。

> **注:** 某些序列的控制字符（逆向换行符和退格符）可以使文本从显示器上消失。由 **tbl** 命令生成的带有竖线的表或许会丢失包含竖线底部的文本行。首先通过 **col** 命令传递输入，然后通过 **hp** 命令，您或许能够避免这些问题。

标志

-e 显示加下划线的叠印字符、半亮上标字符和加下划线的半亮下标字符。否则，所有的叠印字符、下标字符和上标字符则显现为反转视频（dark-on-light）。仅当您的显示器具有显示增强功能时使用该标志。

-m 对于文本中的任意数量的连续空白行，仅产生一个空白行。

相关信息

col 命令、 **eqn** 命令、 **greek** 命令、 **nroff** 命令、 **tbl** 命令。

---

hplj 命令

用途

为 HP LaserJet 系列打印机后处理 **troff** 命令输出。

语法

hplj [**-F** *Directory*] [**-quietly**] [**-landscape**] [*File ...*]

描述

hplj 命令处理 **troff** 命令的输出为到 Hewlett-Packard LaserJet 系列打印机的输出。

如果给定一个或多个文件作为选项，则 **hplj** 命令负责处理这些文件。如果文件没指定，则此命令会作为一个解释标准输入的过滤器。参数 *File* 指定了被 **hplj** 命令处理的作为输出到 HP LaserJet 系列打印机上的文件。

> **注:** **hplj** 命令可以使用 K 编码磁带或文本平衡式编码磁带，如果它们已被装入打印机。（文本-平衡式，HP 部件号 C2053A #C07，将替代 K 编码磁带。）缺省字体文件假设其中一个编码磁带已安装。如果没有 K 编码磁带，可使用下载的位图字体来代替。要完成这个工作，可运行存放在字体目录下的 **no_cart** shell 脚本，这个为 HP 打印机专用的目录是**/usr/lib/font/devhplj**。

不正确的输出发生于如果您的字体文件假设已安装了编码磁带而实际上没有安装。不正确的输出还发生在如果除了 K 编码磁带或文本-平衡式外，还安装了其他的编码磁带或软件字体。

hplj 命令依赖于文件名以 **.out** 为结尾的 **/usr/lib/font/devhplj** 下的文件。除非文件被适当设置，否则这个命令不会产生合理的输出。更多信息，请参阅 **troff** 字体文件格式文档。

标志

-F*Directory* 确定要在其中寻找字体文件的指定目录。作为缺省，**hplj** 命令在 **/usr/lib/font/devhplj** 目录中查找字体文件。

-quietly 禁止所有非致命错误消息。

-landscape 以横向格式打印指定文件。页面设置为横向，因此对于正常阅读，页面的宽度比长度要大。缺省情况下，**hplj** 命令以纵向形式打印。

注: 横式方式只在 HP Jet II 打印机上使用 Courier 字体时才可用。因此，**troff** 文档必须采用 Courier 字体格式。为完成该格式，将以下几行文字插入 **troff** 输入文件的开始:

```
.fp 1 C
.fp 2 C
.fp 3 CB
```

Courier 字体被装入 1 号和 2 号字体位置，Courier 粗体装入 3 号字体位置。

示例

1. 要使用 **lp** 命令打印一个名为 foo 的**troff** 文件到名为 hp 的打印机上，请输入:

   ```
   troff -mm -Thplj foo | hplj | lp -dhp -o -dp
   ```

2. 要使用 **qprt** 命令打印一个名为 boo 的 **troff** 文件到名为 hp 的打印机上，请输入:

   ```
   troff -mm -Thplj boo | hplj | qprt -dp -Php
   ```

注: 两个示例中的 **-dp** 标志均以 “经过” （未修改）方式向打印设备发送打印机数据。

File

/usr/lib/font/devhpl/*.out 包含字体文件。

相关信息

troff 命令格式化文本以便在排版设备上打印。

troff 字体文件格式为 troff 命令指定描述文件。

---

hpmcount 命令

用途

测量应用程序性能。

语法

hpmcount [**-a**] [**-d**] [**-H**] [**-k**] [**-o** *file*] [**-s** *set*] *command*

hpmcount [**-h**]

描述

hpmcount 命令提供由 *command* 命名的应用程序的最大执行时间、硬件性能计数器信息、派生硬件度量和资源利用率统计信息（从 **getrusage()** 系统调用获取）。

要监视的事件类型以及关联的硬件性能计数器是通过以下方式指定的：指定 **-s** 选项；在 **HPM_EVENT_SET** 环境变量中指定事件组名、事件集编号或逗号分隔的事件集编号列表；或者在 **libHPM_events** 输入文件（优先于 **HPM_EVENT_SET**）中指定计数器／事件对（POWER3/PowerPC 604 RISC 微处理器）或事件组名称（POWER4 及更高型号）。

有效事件集编号从 1 到依赖于处理器类型的上限，处理器类型可使用 **pmlist** 命令列出。可指定逗号分隔的事件集列表，而不是事件集编号，在此情况下，将选择计数器多路复用方式。要选择所有事件集，请将编号值设置为 0。

通过指定 **-k** 和 **-H** 选项，可以将系统和管理程序（针对支持管理程序方式的处理器）活动包含到计数中。

标志

-a	聚集关于 POE 运行的计数器。
-d	对于计数器多路复用方式添加详细的事件集计数。
-H	添加代表该进程的管理程序活动。
-h	显示帮助消息。
-k	添加代表该进程的系统活动。
-o *file*	输出文件名
-s *set*	列出预定义的事件集或者逗号分隔的事件集列表（1 到 *N*，或者 0 表示选择全部。请参阅 **pmlist** 命令。）当使用逗号分隔的事件集列表时，将选择计数器多路复用方式。

参数

command	指定为其进行性能测量的已执行程序。

环境变量

以下环境变量直接影响 **hpmcount** 命令的执行（有其他的 **MP_*** 环境变量影响并行程序的执行）。

HPM_EVENT_SET	选择一个事件集。该值可以是一个从 1 到 6（在 POWER3 系统上）、从 1 到 4（在 PowerPC 604 RISC 微处理器 系统上）或从 1 到一个依赖于处理器的上限（在 POWER4 及后续版本的系统上）的整数。此环境变量还用来在 POWER4 及后续版本的系统上选择一个事件组名。可指定逗号分隔的事件集列表。在此情况下，将选择计数器多路复用方式。
HPM_DIV_WEIGHT	提供用来在 POWER4 系统上计算加权翻转的权重（一个大于 1 的整数）。
MP_CHILD	在指定聚集计数来补足输出结果文件名（*myID*）、同步结果整理以及更接近地识别详细／调试诊断消息的并行环境中使用。
MP_PROCS	程序任务的数量。

HPM_AGGREGATE_OUTPUT
聚集在 POE 应用程序上的计数（强制命令行实参 **-a**）。使用此标志，将为所有任务生成一个单一文件性能文件。此项仅处理 POE 或 Load Leveller，它要求在系统上提供并行文件系统（如 GPFS）。

HPM_LOG_DIR
当设置此标志时，**hpmcount** 将在提供的目录中写入一个带有性能数据的 **hpm_log.***id* 文件。这是对常规输出的一个附加内容。

MP_PARTITION
在 POE 应用程序上，*id* 是 **MP_PARTITION** 提供的一个 POE 标识。否则，它将是 *pid*。还命名内部键锁和数据文件。

HPM__MX_DURATION
在以计数器多路复用方式进行计数时，此标志指定每个时间片的持续时间。它以毫秒为单位表示，并且必须在 10 毫秒到 30 秒的范围之内。如果未设置此标志，则用于时间片持续时间的缺省值为 100 毫秒。

另外，以下环境变量（由用户提供）指定用于派生度量值计算的内存等待时间、高速缓存等待时间和 TLB 不命中等待时间的估算值。这些环境变量在顺序上不比最终在文件 **HPM_flags.env**（如果存在）中提供的评估值优先。

- **HPM_MEM_LATENCY**
- **HPM_L3_LATENCY**
- **HPM_L35_LATENCY**
- **HPM_AVG_L3_LATENCY**
- **HPM_AVG_L2_LATENCY**
- **HPM_L2_LATENCY**
- **HPM_L25_LATENCY**
- **HPM_L275_LATENCY**
- **HPM_L1_LATENCY**（读取但不使用此变量）
- **HPM_TLB_LATENCY**

退出状态

0 成功完成。
>0 发生错误。

示例

1. 要运行 **ls** 命令并写入来自硬件计数器的有关事件集 5 中的事件的信息，请输入:

```
hpmcount -s 5 ls
```

2. 要运行 **ls** 命令，并写入来自硬件计数器（使用计数器多路复用方式）的有关事件集 5、2 和 9 中的事件的信息，请输入:

```
hpmcount -s 5,2,9 ls
```

具体实现

hpmcount 命令使用 PMAPI 线程级 API。

hpmcount *command* 参数不作为带有选项的应用程序名的命令行解析。相反，必须创建一个包含该命令行的 shell 脚本。

位置

/usr/bin/perf/pmapi/hpmcount

标准输入

不使用。

标准输出

除非在命令行上指定 **-o** *file* 选项，否则性能监视结果将写入 **stdout**。

标准错误

仅用于诊断消息。

文件

使用以下输入文件（如果存在）。

libHPM_events

用户提供的事件集文件。此文件在顺序上不比以 **-s** 选项指定的命令行优先。Power3/PowerPC 604 RISC 微处理器 计数器 / 事件对的格式为 *counternumber eventname*。例如:

```
0 PM_LD_MISS_L2HIT
1 PM_TAG_BURSTRD_L2MISS
2 PM_TAG_ST_MISS_L2
3 PM_FPU0_DENORM
4 PM_LSU_IDLE
5 PM_LQ_FULL
6 PM_FPU_FMA
7 PM_FPU_IDLE
```

还可以指定逗号分隔的事件列表。这将开启计数器多路复用方式:

```
0 PM_CYC,PM_FPU_FIN,PM_IC_MISS
1 PM_LD_CMPL,PM_INST_CMPL,PM_DC_MISS
2 PM_INST_CMPL,PM_FPU_WT,PM_INST_CMPL
3 PM_LD_MISS_DC_XU,PM_CYC,PM_CYC
```

对于 POWER4 事件组名，格式为 *event_group_name*。例如:

```
pm_hpmcount1
```

还可以指定逗号分隔的事件列表。这将开启计数器多路复用方式:

```
pm_hpmcount1,pm_hpmcount2,pm_basic
```

HPM_flags.env

包含用于派生度量值计算的环境变量 / 值对的文件。例如:

```
HPM_L2_LATENCY 12
HPM_EVENT_SET  5
```

./.hpm_lockfile_mp_partition_

锁文件。为 **hpmcount** 命令的内部使用保留此文件。

./.hpm_datafile_mp_partition_

累积结果文件。为 **hpmcount** 命令的内部使用保留此文件。

使用以下输出文件。

file_*myID*.*pid*	以 **-o** 选项为 **hpmcount** 输出结果指定文件，其中 *myID* 取自 **MP_CHILD** 环境变量，缺省值为 0000。
HPM_LOG_DIR/hpm_log.MP_PARTITION 或 **HPM_LOG_DIR/hpm_log.***pid*	为关于 POE 运行的聚集计数器指定的日志文件。
./.hpm_lockfile_*mp_partition*	锁文件。为 **hpmcount** 命令的内部使用保留此文件。
./.hpm_datafile_*mp_partition*	累积结果文件。为 **hpmcount** 命令的内部使用保留此文件。

相关信息

『hpmstat 命令』，**pmlist** 命令。

getrusage 子例程和 **pm_initialize** 子例程。

AIX 5L Version 5.3 Performance Tools Guide and Reference 中的 Performance Monitor API Programming。

---

hpmstat 命令

用途

提供系统范围的硬件性能计数器信息。

语法

hpmstat [**-d**] [**-H**] [**-k**] [**-o** *file*] [**-r**] [**-s** *set*] [**-T**] [**-U**] [**-u**] *interval count*

hpmstat [**-h**]

描述

hpmstat 命令提供最大执行时间、硬件性能计数器信息和派生硬件度量值。只有具有 root 权限的用户可以使用该命令。

当不带命令行选项指定该命令时，**hpmstat** 针对缺省的事件集 1 计数 1 秒钟内用户、内核以及管理程序（针对处理器支持管理程序方式）活动的缺省的 1 次迭代。然后，它会将原始计数器值和派生的度量值写至标准输出。在缺省情况下，禁用 **runlatch** 以便可以在空闲周期中执行时执行计数。

当指定 **-U** 选项时，*interval* 是毫秒数，迭代 *count* 为无穷大，不计算派生的度量值，也不将度量值写至标准输出。如果指定了计数器多路复用方式，则忽略此选项。

当指定 **-T** 选项时，输出信息在时间戳记（秒加毫秒）之后，计时信息以时间戳记而不是以秒计的时间书写。

使用事件集 **-s** 选项或通过在 **HPM_EVENT_SET** 环境变量中指定事件组名或事件集号指定要监视的事件类型和关联的硬件性能计数器。或者在 **libHPM_events** 输入文件中指定计数器 / 事件对（POWER3 / PowerPC 604 RISC 微处理器）或事件组名（POWER4 及后续版本）（顺序较 **HPM_EVENT_SET** 优先）。

可指定逗号分隔的事件集列表，而不是事件集编号，在此情况下，将选择计数器多路复用方式。要选择所有事件集，请将事件集编号值设置为 0。

有效事件集编号从 1 到依赖于处理器类型的上限，处理器类型可使用 **pmlist** 命令列出。

标志

-d 对于计数器多路复用方式添加详细的事件集计数。
-H 只计数管理程序活动。
-h 显示帮助消息。
-k 只计数系统活动。
-o *file* 输出文件名。
-r 在空闲周期中执行时启用 **runlatch** 并禁用计数器。
-s *set* 列出预定义的事件集或者逗号分隔的事件集列表（1 到 *N*，或者 0 表示选择全部。请参阅 **pmlist** 命令。）当使用逗号分隔的事件集列表时，将选择计数器多路复用方式。
-T 书写时间戳记而不是以秒计的时间。
-U 以毫秒设置计数时间间隔。如果指定了计数器多路复用方式，则忽略此选项。
-u 只计数用户活动。

参数

interval 以秒或毫秒显示计数时间间隔，缺省值为 1。
count 显示要计数的迭代数。缺省值为 1，时间间隔以秒计，当指定选项 **-U** 时该值为无穷大。

环境变量

以下环境变量直接影响 **hpmstat** 命令的执行（有其他的 **MP_*** 环境变量影响并行程序的执行）。

HPM_EVENT_SET 选择一个事件集。该值可以是一个从 1 到 6（在 POWER3 系统上）、从 1 到 4（在 PowerPC 604 RISC 微处理器 系统上）或从 1 到一个依赖于处理器的上限（在 POWER4 及后续版本的系统上）的整数。此环境变量还用来在 POWER4 及后续版本的系统上选择一个事件组名。
HPM_DIV_WEIGHT 提供用来在 POWER4 系统上计算加权翻转的权重（一个大于 1 的整数）。
HPM__MX_DURATION 在以计数器多路复用方式进行计数时，此标志指定每个时间片的持续时间。它以毫秒为单位表示，并且必须在 10 毫秒到 30 秒的范围之内。如果未设置此标志，则用于时间片持续时间的缺省值为 100 毫秒。

另外，以下环境变量（由用户提供）指定用于派生度量值计算的内存等待时间、高速缓存等待时间和 TLB 不命中等待时间的估算值。这些环境变量在顺序上不比最终在文件 **HPM_flags.env**（如果存在）中提供的评估值优先。

- **HPM_MEM_LATENCY**
- **HPM_L3_LATENCY**
- **HPM_L35_LATENCY**
- **HPM_AVG_L3_LATENCY**
- **HPM_AVG_L2_LATENCY**
- **HPM_L2_LATENCY**
- **HPM_L25_LATENCY**

- **HPM_L275_LATENCY**
- **HPM_L1_LATENCY**（读取但不使用此变量）
- **HPM_TLB_LATENCY**

退出状态

0	成功完成。
>0	发生错误。

示例

要写针对系统、用户和管理程序活动，在超过 1 秒的时间间隔里关于来自硬件计数器的事件集 2 中的事件的信息，请输入:

```
hpmstat -s 2
```

具体实现

hpmstat 命令使用 PMAPI 系统级 API。因为如果线程级 API 正在使用，系统级 API 将报告无效的数据，所以在线程级 API 进行调用的同时不允许进行系统级 API 调用。允许线程环境将锁定系统级 API，直到收回对最后一次环境的分配才会释放锁定。为此，如果由 **libhpm** 或 **hpmcount** 测量的程序在 **hpmstat** 活动期间是活动的，则 **hpmstat** 计数器将不准确。

位置

/usr/bin/perf/pmapi/hpmstat

标准输入

不使用。

标准输出

除非在命令行上指定 **-o** *file* 选项，否则性能监视结果将写入 **stdout**。

标准错误

仅用于诊断消息。

文件

使用以下输入文件（如果存在）。

libHPM_events

用户提供的事件集文件。此文件在顺序上不比以 **-s** 选项指定的命令行优先。Power3/PowerPC 604 RISC 微处理器 计数器／事件对的格式为 *counternumber eventname*。例如:

```
0 PM_LD_MISS_L2HIT
1 PM_TAG_BURSTRD_L2MISS
2 PM_TAG_ST_MISS_L2
3 PM_FPU0_DENORM
4 PM_LSU_IDLE
5 PM_LQ_FULL
6 PM_FPU_FMA
7 PM_FPU_IDLE
```

对于 POWER4 事件组名，格式为 *event_group_name*。例如:

```
pm_hpmcount1
```

HPM_flags.env

包含用于派生度量值计算的环境变量／值对的文件。例如:

```
HPM_L2_LATENCY 12
HPM_EVENT_SET  5
```

使用以下输出文件。

file

以 **-o** 选项指定的用于 **hpmstat** 输出结果的文件。

相关信息

第 636 页的『hpmcount 命令』，**pmlist** 命令。

pm_initialize 子例程。

AIX 5L Version 5.3 Performance Tools Guide and Reference 中的 Performance Monitor API Programming。

hps_dump 命令

用途

转储网络终端加速器（NTX）适配器内存内容到主机文件。该命令仅应用于 AIX 4.2.1 或其后版本。

语法

hps_dump [**-f** *Name*] [**-d** *Device*]

描述

hps_dump 命令使用装入程序接口从适配器卡将所有的内存内容上载到文件中。由此会产生一个系统快照，用于以后的分析和调试。文件的前 1024 字节包含以下内容:

80	识别字符串，包含版本。
80	从主机系统转储的时间和日期。
80	注释。
268	从主机适配器得到的日志表。
32	系统地址表。

8　　转储的起始和结束地址范围。
476　　填充到全部 1024 字节。

标志

-f *Name*　　指定转储名称。使用该选项来覆盖缺省文件名 **./hpscore**。
-d *Device*　　指定适配器的裸设备文件名。使用这个选项可以覆盖缺省设备名 **/dev/rhp0**。

退出状态

此命令返回以下出口值:

0　　成功完成。
>0　　发生错误。

安全性

访问控制: 您必须拥有 root 用户权限才能运行这个命令。

审计事件: N/A

示例

1. 要获取缺省适配器内存到当前目录下的文件 **hpscore** 的转储，请输入:

   ```
   hps_dump
   ```

2. 要获取缺省适配器内存到缺省适配器的当前目录下的文件 **hpsdebug** 的转储，请输入:

   ```
   hps_dump -f hpsdebug
   ```

3. 要获取适配器内存 **/dev/rhp1** 到缺省适配器的当前目录下的文件 **hpsdebug** 的转储，请输入:

   ```
   hps_dump -f hpsdebug -d /dev/rhp1
   ```

文件

/usr/bin/hps_dump　　包括 **hps_dump** 命令。
/dev/rhp0　　缺省 NTX 裸设备文件名。

相关信息

/dev/rhp 文件。

---

htable 命令

用途

将主机文件转换为网络库例程所使用的格式。

语法

/usr/sbin/htable [**-c** *connected-nets*] [**-l** *local-nets*] *input-file*

注: 请勿将空格放在逗号的任何一边。

描述

htable 命令将主机文件由 RFC 810 指定的格式转换为网络库例程使用的格式。这个转换创建了三个文件: **/etc/hosts** 文件, **/etc/networks** 文件和 **/etc/gateways** 文件。

当不使用 **named** 守护程序时, **gethostbyname** 子例程使用 **hosts** 文件将主机名映射到地址。**getnetent** 子例程使用 **networks** 文件将网络名映射成数字。

gateways 文件可以被 **routed** 守护程序用于识别被动因特网网关。

如果任何本地 **hosts**, **networks**, 或 **gateways** 文件(分别为 **localhosts**, **localnetworks**, 或 **localgateways**)存在于当前目录中, 则那个文件的内容将被准备作为输出文件。在这些文件中, **htable** 程序只解释 **gateways** 文件。准备的内容允许站点维护一般不出现在主数据库中的本地条目。

标志

-c *connected-nets*
如果网络路由守护程序使用 **gateways** 文件, 指定一个主机直接连接到的网络的列表。用逗号分隔网络, 并使用网络名或标准因特网的点符号(例如, `-c arpanet,128.32,LocalEthernet`)。**htable** 命令只包含直接被连接到指定网络之一的网关或是可以从另一个在已连接网络上的网关能够访问到的网关。

-l *local-nets*
指定一个被 **htable** 命令当作本地的网络列表。只能从 **localhosts** 文件中得到有关在本地网络上的主机的信息。用逗号分隔网络, 并使用网络名或标准因特网的点符号(例如, `-l 128.32,local-ether-net`)。从主数据库得到的本地主机条目可以省略, 以便 **localhosts** 文件可以覆盖输入文件(命令行中指定的文件)中的条目。

文件

/*CurrentDirectory*/**localgateways** 包含本地网关信息。
/*CurrentDirectory*/**localhosts** 包含本地主机名信息。
/*CurrentDirectory*/**localnetworks** 包含本地网络信息。

相关信息

gettable 命令。

named 守护程序, **routed** 守护程序。

gateways 文件格式, **hosts** 文件格式, **gateways** 文件格式。

《*网络与通信管理*》中的『TCP/IP 路由网关』。

hty_load 命令

用途

显示或下载网络终端加速器(NTX)适配器配置。

语法

hty_load [**-d** *Device*] [**-f** *ConfigFileName*]

描述

hty_load 命令显示或下载适配器配置。如果不带任何标志发出此命令，则系统为 **/dev/rhp0** 设备文件显示当前的适配器配置。给定 *Device* 参数，则**hty_load** 命令将一个配置文件装入 tty 驱动程序。tty 驱动程序使用此文件对主机表达服务（HPS）和适配器都进行配置。

通常，从 **/etc/rc.ntx** 文件中调用 **hty_load** 命令。

配置文件

hty_load 命令使用单个配置文件对适配器进行配置。每个条目在单独的行上。条目由换行字符进行分隔。条目中的字段由制表符或空格符进行分隔。配置文件中的条目有以下字段:

```
MinorNumber Cluster NumberOfPorts
```

这些字段有以下值:

MinorNumber 指定卡的次要设备号。
Cluster 此字段总为 1。

NumberOfPorts 指定 hty 设备数。这个数字依赖于您所使用的适配器的型号。可用通道数对于 2MB 的板是从 1 到 256，对于 8MB 的板是从 1 到 2048。

配置文件也支持注释。注释行以一个 #（磅符号）号开始。在注释符右侧的所有内容均被忽略。注释行以换行符结束。

标志

-d *Device* 指定适配器的裸设备文件名。使用这个选项可以覆盖缺省设备名 **/dev/rhp0**。
-f *ConfigFileName* 指定驱动程序配置文件名。缺省的配置文件是 **/etc/hty_config** 文件。

退出状态

此命令返回以下出口值:

0 成功完成。
>0 发生错误。

安全性

访问控制: 您必须拥有 root 用户权限才能运行这个命令。

审计事件: N/A

示例

要装入系统配置并使用缺省驱动程序配置文件，请输入:

```
hty_load -d /dev/rhp0
```

文件

/usr/bin/hty_load	包含 **hty_load** 命令。
/etc/rc.ntx	调用 **hty_load** 命令。
/etc/hty_config	缺省 NTX 驱动程序配置文件名。
/dev/rhp0	缺省 NTX 裸设备文件名。

相关信息

/dev/rhp 文件。

---

hyphen 命令

用途

查找带连字符的单词。

语法

hyphen [*File* ...]

描述

hyphen 命令读取一个或多个英文文件，从中查找所有以带连字符的单词结尾的行并将这些单词写入标准输出。参数 *File* 指定了 **hyphen** 命令要读取的英文文件。缺省为标准输入。如果没有指定文件，或 -（连字符）被指定为最后的文件名， **hyphen** 命令就读取标准输入。**hyphen** 可以用作过滤器。

> **注:** **hyphen** 命令无法读取斜体或带下划线的带连字符的单词。有时， **hyphen** 命令会给出一些不必要的输出。

示例

要检查由文本格式化程序在文件中执行的连字符连接，请输入:

```
mm [Flag...] [File...] | hyphen
```

相关信息

mm 命令、**troff** 命令。

附录. 声明

本信息是为在美国提供的产品和服务编写的。

IBM 可能在其他国家或地区不提供本文档中讨论的产品、服务或功能特性。有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代表咨询。任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务，则由用户自行负责。

IBM 公司可能已拥有或正在申请与本文档描述的内容有关的各项专利。提供本文档并未授予用户使用这些专利的任何许可。您可以用书面方式将许可查询寄往:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区: International Business Machines Corporation “按现状”提供本出版物，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的有关非侵权、适销和适用于某种特定用途的保证。某些国家或地区在某些交易中不允许免除明示或暗含的保证。因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。此处的信息将定期更改；这些更改将编入本资料的新版本中。IBM 可以随时对本出版物中描述的产品和／或程序进行改进和／或更改，而不另行通知。

本程序的被许可方如果要了解有关程序的信息以达到如下目的: （i）允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换，以及（ii）允许对已经交换的信息进行相互使用，请与以下地址联系:

IBM Corporation
Dept. LRAS/Bldg. 003
11400 Burnet Road
Austin, TX 78758-3498
U.S.A.

只要遵守适当的条件和条款，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际软件许可协议或任何同等协议中的条款提供。

有关双字节（DBCS）信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中获取。IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。那些 Web 站点中的资料不是 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

本信息包含日常商业运作所用到的数据和报表示例。为了尽可能完整地演示它们，示例中包括了个人、公司、品牌和产品的名称。所有这些名称都是虚构的，如果和实际的商务企业所用的名称和地址雷同，则纯属配合。

版权许可:

本信息包括源语言形式的样本应用程序，这些样本说明不同操作平台上的编程方法。如果是为了开发、使用、营销或分发与编写样本程序的操作平台的应用程序编程接口一致的应用程序，您可以任何形式复制、修改和分发这些样本程序，而无须向 IBM 付费。这些示例并未在所有条件下作全面测试。因此，IBM 不能保证或暗示这些程序的可靠性、可维护性或功能。如果是为了开发、使用、营销或分发符合 IBM 应用程序编程接口的应用程序，则您可以任何形式复制、修改和分发这些样本程序，而无须向 IBM 付费。

此处包含的任何性能数据都是在受控环境中测得的。因此，在其他操作环境中获得的数据可能会有明显的不同。有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。此外，有些测量是通过推算而估计的。实际结果可能会有差异。本文档的用户应当验证其特定环境的适用数据。

商标

以下术语为 International Business Machines Corporation 在美国和 / 或其他国家或地区的商标:

AIX
AIX 5L
CICS
CICS/ESA
CICS/MVS
CICS/VSE
DPI HACMP
IBM
LoadLeveler
Micro Channel
MVS
MVS/SP
MVS/XA
OS/400
POWER
PowerPC
RACF
RS/6000
SecureWay

VSE/ESA

WebExplorer

XT

UNIX 是 The Open Group 在美国和／或其他国家或地区的注册商标。

Java 和所有基于 Java 的商标和徽标是 Sun Microsystems,Inc. 在美国和／或其他国家或地区的注册商标。

其他公司、产品和服务名称可能是其他公司的商标或服务标记。

索引

[B]

[C]

[D]

[H]

[J]

[K]

[L]

[M]

[N]

[X]

[Y]

[Z]

A

D

E

F

G

H

I

L

M

N

P

R

S

T

W

[特别字符]

中国印刷

S151-0036-03