

Linux培训系列

第一讲

将给您介绍 bash（标准的 Linux shell），为您展示如何充分利用如 ls、cp 和 mv 这样的标准 Linux 命令，并向您讲解 Linux 的权限和所有权模型以及更丰富的内容。

内容基础，语言简短简洁

红联Linux论坛是致力于Linux技术讨论的站点，目前网站收录的文章及教程基本能满足不同水平的朋友学习。

红联Linux门户：www.linux110.com

红联Linux论坛：www.linuxdiyf.com/bbs

下载:Linux电子书籍：

<http://www.linux286.com/linux/linuxdzsj.htm>

目录

介绍 **bash**

- 介绍 *bash - shell*
- 您在运行 *bash* 吗？
- 关于 *bash*
- 使用 “*cd*”
- 路径
- 绝对路径
- 相对路径
- 使用 “*..*”
- 使用 “*..*”（续）
- 相对路径示例
- 理解 “*.*”
- cd* 和主目录
- 其他用户的主目录

使用 **Linux** 命令

- 使用 *Linux* 命令 - 介绍 “*ls*”
- 长目录清单
- 长目录清单（续）
- 查看目录
- 递归和索引节点清单
- 理解索引节点，第 1 部分
- 理解索引节点，第 2 部分
- 理解索引节点，第 3 部分
- mkdir*
- mkdir -p*
- touch*
- echo* 和重定向
- echo* 和重定向
- cat* 和 *cp*
- mv*

创建链接和删除文件

- 创建链接和删除文件 - 硬链接
- 硬链接（续）
- 符号链接
- 符号链接（续）
- 符号链接深入
- rm*
- rmdir*
- rm* 和 目录

介绍通配符

- 介绍通配符
- 介绍通配符（续）
- 理解不匹配
- 理解不匹配（续）
- 通配符语法：***
- 通配符语法：*?*
- 通配符语法：*[]*
- 通配符语法：*[!]*

[通配符告诫说明](#)

[通配符告诫说明（续）](#)

[单引号与双引号的对比](#)

海量Linux技术文章

[海量Linux技术文章](#)

介绍 bash

介绍 bash - shell

发布时间 :2007-01-26 20:56:33

如果您使用过 Linux 系统，那么您知道当登录时，将会看到像这样的提示符：

\$

您所看到的特殊的提示符可能看起来很不一样。它可能包含系统的主机名、当前的工作目录名，或者两者都有。但是不管这个特殊的提示符看起来像什么，有一件事是肯定的。打印出这个提示符的程序叫“shell”，极有可能您的特殊的 shell 是一个叫 bash 的程序。

您在运行 bash 吗？

发布时间 :2007-01-26 20:57:26

您可以通过输入下面的命令来检查您是否正在运行 bash：

```
$ echo $SHELL  
/bin/bash
```

如果上面的命令行报错或者不会类似地响应我们的示例，那么您可能正在运行一个不同于 bash 的 shell。

关于 bash

发布时间 :2007-01-26 20:57:58

Bash 是 “ Bourne-again shell ” 的首字母缩写，它是大多数 Linux 系统缺省的 shell。shell 的任务是执行您的命令，使您能够与 Linux 系统进行交互。当您输完命令，您可以通知 shell 执行 exit 或 logout 命令，在此您将返回到登录提示符。

顺便提一下，您还可以通过在 bash 提示符下按 control-D 来注销。

使用 “ cd ”

发布时间 :2007-01-26 20:58:33

您可能已经发现，目不转睛地盯着 bash 提示符可不是世界上最让人感到有劲的事。那么，让我们来开始用 bash 来浏览我们的文件系统。在提示符下，输入下面的命令（不包括 \$）：

```
$ cd /
```

我们只告诉 bash 您想在 /（也称为根目录）中工作；系统上的所有目录形成一棵树，/ 被认为是这棵树的顶部，或者是根。cd 设置当前您正在工作的目录，也称为“当前工作目录”。

路径

发布时间 :2007-01-26 20:59:04

要看 bash 的当前工作目录，您可以输入：

```
$ pwd  
/
```

在上面的示例中，cd 的 / 参数叫做路径。它告诉 cd 我们要转到什么地方。特别是，/ 参数是一个绝对路径，意味着它指定了相对于文件系统树的根的位置。

绝对路径

发布时间 :2007-01-26 20:59:46

这里有几个其它的绝对路径：

```
/dev
/usr
/usr/bin
/usr/local/bin
```

您可以看到，所有绝对路径有一个共同点就是，它们都以 / 开头。通过路径 /usr/local/bin，我们告诉 cd 进入 / 目录，接着进入这个目录之下的 usr 目录，然后再进入 local 和 bin。绝对路径总是通过是否以 / 开头来判断。

相对路径

发布时间 :2007-01-26 21:00:17

另一种路径叫相对路径。在 Bash 中，cd 以及其它命令总是解释那些相对于当前目录的路径。相对路径绝不会以 / 开头。这样，如果我们在 /usr 中：

```
$ cd /usr
```

那么，我们可以使用相对路径来转到 /usr/local/bin 目录：

```
$ cd local/bin  
$ pwd  
/usr/local/bin
```

使用 “ .. ”

发布时间 :2007-01-26 21:00:48

相对路径还可以包含一个或多个 .. 目录。.. 目录是指向父目录的专门目录。那么，继续前面的示例：

```
$ pwd
/usr/local/bin
$ cd ..
$ pwd
/usr/local
```

您可以看到，现在我们的当前目录是 /usr/local。我们能够“后退”到相对于我们所在的当前目录的一个目录。

使用“..”（续）

发布时间 :2007-01-26 21:01:23

此外，我们还可以将 .. 添加到一个现有的相对路径中，使我们可以进入与我们已在目录并排的目录，例如：

```
$ pwd
/usr/local
$ cd ../share
$ pwd
/usr/share
```

相对路径示例

发布时间 :2007-01-26 21:02:02

相对路径可以变得相当复杂。这里有几个示例，所有的都没有显示出结果的目标路径。请试着推断一下，输入这些命令后，您最终将会转到什么地方：

```
$ cd /bin
$ cd ../usr/share/zoneinfo
```

```
$ cd /usr/X11R6/bin
$ cd ../lib/X11
```

```
$ cd /usr/bin
$ cd ../bin/../bin
```

现在，试验一次，看看您的推断是否正确。

理解 “ . ”

发布时间 :2007-01-26 21:02:34

在我们结束 cd 的介绍之前，我们还需要讨论一些更多的内容。首先，还有另一个叫 . 的专门的目录。它表示 “当前目录”。然而该目录不为 cd 命令使用，它通常用来执行一些当前目录中的程序，如下所示：

```
$ ./myprog
```

在上面的示例中，驻留在当前工作目录中的 myprog 可执行文件将被执行。

cd 和主目录

发布时间 :2007-01-26 21:03:08

如果我们想要转到主目录，我们可以输入：

```
$ cd
```

没有参数，cd 将转到主目录，对于超级用户来说是 /root，对于一般用户来说通常是 /home/username。但是，如果我们想要指定一个主目录中的文件，将会怎样呢？可能我们想要将一个文件参数传给 myprog 命令。如果该文件在主目录中，我们可以输入：

```
$ ./myprog /home/drobbins/myfile.txt
```

但是，使用像这样的绝对路径并不总是很方便。幸好，我们可以使用 ~（代字符）字符来完成同样的事：

```
$ ./myprog ~/myfile.txt
```

其他用户的主目录

发布时间 :2007-01-26 21:03:39

Bash 将把单独的 ~ 扩展为指向主目录，然而您还可以用它来指向其他用户的主目录。例如，如果我们想要引用 fred 的主目录中的名为 fredsfiler.txt 的文件，可以输入：

```
$ ./myprog ~fred/fredsfiler.txt
```

使用 Linux 命令

使用 Linux 命令 - 介绍 “ls”

发布时间 :2007-01-26 21:04:38

现在，我们将快速地看一看 ls 命令。很可能，您已经很熟悉 ls，并且知道只输入 ls 本身将列出当前工作目录的内容：

```
$ cd /usr
$ ls
X11R6  doc      i686-pc-linux-gnu lib  man      sbin  ssl
bin    gentoo-x86 include      libexec portage  share tmp
distfiles i686-linux info          local  portage.old src
```

通过指定 -a 选项，您可以看到目录中的所有文件，包括隐藏文件 — 那些以 . 开头的文件。您可以在下面的示例中看到，ls -a 将显示 . 和 .. 专门的目录链接：

```
$ ls -a
.  bin    gentoo-x86    include libexec portage  share tmp
.. distfiles i686-linux    info    local  portage.old src
X11R6 doc      i686-pc-linux-gnu lib  man    sbin    ssl
```

长目录清单

发布时间 :2007-01-26 21:05:32

您还可以在 `ls` 命令行中指定一个或多个文件或目录。如果您指定一个文件，`ls` 将只显示那个文件。如果您指定一个目录，`ls` 将显示这个目录的内容。当您需要在目录清单中查看权限、所有权、修改时间和大小信息时，`-l` 选项很管用。

长目录清单（续）

发布时间 :2007-01-26 21:06:30

在下面的示例中，我们使用 -l 选项来显示我的 /usr 目录的完整的清单。

```
$ ls -l /usr
drwxr-xr-x  7 root  root    168 Nov 24 14:02 X11R6
drwxr-xr-x  2 root  root   14576 Dec 27 08:56 bin
drwxr-xr-x  2 root  root   8856 Dec 26 12:47 distfiles
lrwxrwxrwx  1 root  root      9 Dec 22 20:57 doc -> share/doc
drwxr-xr-x 62 root  root   1856 Dec 27 15:54 gentoo-x86
drwxr-xr-x  4 root  root   152 Dec 12 23:10 i686-linux
drwxr-xr-x  4 root  root    96 Nov 24 13:17 i686-pc-linux-gnu
drwxr-xr-x 54 root  root   5992 Dec 24 22:30 include
lrwxrwxrwx  1 root  root    10 Dec 22 20:57 info -> share/info
drwxr-xr-x 28 root  root  13552 Dec 26 00:31 lib
drwxr-xr-x  3 root  root    72 Nov 25 00:34 libexec
drwxr-xr-x  8 root  root   240 Dec 22 20:57 local
lrwxrwxrwx  1 root  root      9 Dec 22 20:57 man -> share/man
lrwxrwxrwx  1 root  root    11 Dec  8 07:59 portage -> gentoo-x86/
drwxr-xr-x 60 root  root   1864 Dec  8 07:55 portage.old
drwxr-xr-x  3 root  root   3096 Dec 22 20:57 sbin
drwxr-xr-x 46 root  root   1144 Dec 24 15:32 share
drwxr-xr-x  8 root  root    328 Dec 26 00:07 src
drwxr-xr-x  6 root  root    176 Nov 24 14:25 ssl
lrwxrwxrwx  1 root  root    10 Dec 22 20:57 tmp -> ../var/tmp
```

清单中，第一栏显示了每一项的权限信息。我将逐位说明该信息怎样被解释。接下来的一栏列出了每个文件系统对象的链接数，这里我们一带而过，稍后我们再回过头来讨论。第三和第四栏分别列出所有者和组。第五栏列出了对象的大小。第六栏是对象的“最近修改时间”或“mtime”（建立时间）。最后一栏是对象的名称。如果文件是符号链接，您将看到后面跟有 -> 以及符号链接所指向的路径。

查看目录

发布时间 :2007-01-26 21:07:00

有时，您想要查看目录本身，而不是目录内部。对于这种情况，您可以指定 -d 选项，它将告诉 ls 查看所有目录，否则在通常情况下，ls 要查看目录内部：

```
$ ls -dl /usr /usr/bin /usr/X11R6/bin ../share
drwxr-xr-x  4 root  root    96 Dec 18 18:17 ../share
drwxr-xr-x 17 root  root   576 Dec 24 09:03 /usr
drwxr-xr-x  2 root  root  3192 Dec 26 12:52 /usr/X11R6/bin
drwxr-xr-x  2 root  root 14576 Dec 27 08:56 /usr/bin
```

递归和索引节点清单

发布时间 :2007-01-26 21:07:36

您可以使用 `-d` 来查看目录本身，而您还可以用 `-R` 来完成相反的工作 — 不仅只查看一个目录内部，而且要递归地查看该目录内所有的目录内部！我们将不会有对应该选项的任何示例输出（因为它一般占很大的篇幅），但是为了感觉一下它是怎样工作的，您可以试几个 `ls -R` 和 `ls -RI` 命令。

最后，`ls` 的 `-i` 选项可以用来在清单中显示文件系统对象的索引节点号：

```
$ ls -i /usr
1409 X11R6      314258 i686-linux      43090 libexec      13394 sbin
1417 bin        1513 i686-pc-linux-gnu  5120 local          13408 share
8316 distfiles  1517 include          776 man             23779 src
 43 doc         1386 info             93892 portage        36737 ssl
70744 gentoo-x86 1585 lib              5132 portage.old     784 tmp
```

理解索引节点，第 1 部分

发布时间 :2007-01-26 21:08:37

文件系统的每个对象都分配到一个独一无二的索引，叫做索引节点号。这可能看起来微不足道，但是理解索引节点对于理解许多文件系统操作来说很重要。例如，请考虑出现在每个目录中的 `.` 和 `..` 链接。为了完全理解 `..` 目录实际上是什么，我们将先来看一看 `/usr/local` 的索引节点号：

```
$ ls -id /usr/local
5120 /usr/local
```

`/usr/local` 目录有一个 5120 索引节点号。现在，我们来看一看 `/usr/local/bin/..` 的索引节点号：

```
$ ls -id /usr/local/bin/..
5120 /usr/local/bin/..
```

理解索引节点，第 2 部分

发布时间 :2007-01-26 21:09:20

您可以看到，/usr/local/bin/.. 具有和 /usr/local 相同的索引节点号！这就是我们抓住的问题的实质。过去，我们认为 /usr/local 是这个目录本身。现在，我们发现索引节点 5120 实际上是这个目录，并且我们发现了指向该索引节点的两个目录条目（叫做“链接”）。/usr/local 和 /usr/local/bin/..都链接到索引节点 5120。虽然索引节点 5120 只在磁盘中的一地方存在，但是多个目录条目都链接到它上面。

理解索引节点，第 3 部分

发布时间 :2007-01-26 21:09:55

事实上，通过使用 `ls -dl` 命令，我们可以看到索引节点 5120 被引用的总次数：

```
$ ls -dl /usr/local
drwxr-xr-x  8 root  root    240 Dec 22 20:57 /usr/local
```

如果我们看一看从左起的第二栏，我们可以看到目录 `/usr/local`（索引节点 5120）被引用了 8 次。在我的系统中，引用该索引节点的不同路径有这些：

```
/usr/local
/usr/local/.
/usr/local/bin/..
/usr/local/games/..
/usr/local/lib/..
/usr/local/sbin/..
/usr/local/share/..
/usr/local/src/..
```

mkdir

发布时间 :2007-01-26 21:10:28

我们来快速地看一看 mkdir 命令，它可以用来创建新目录。下面的示例创建了三个新目录：tic、tac 和 toe，都在 /tmp 下：

```
$ cd /tmp
$ mkdir tic tac toe
```

缺省情况下，mkdir 不会为您创建父目录；邻接的上一元素的完整路径必须存在。因此，如果您想要创建目录 won/der/ful，您将需要发出三个单独的 mkdir 命令：

```
$ mkdir won/der/ful
mkdir: cannot create directory `won/der/ful': No such file or directory
$ mkdir won
$ mkdir won/der
$ mkdir won/der/ful
```

mkdir -p

发布时间 :2007-01-26 21:11:02

然而，mkdir 有一个很方便的 -p 选项，该选项告诉 mkdir 创建所有缺少的父目录，如下所示：

```
$ mkdir -p easy/as/pie
```

总之，非常简单。要学习更多关于 mkdir 命令的知识，请输入 man mkdir 来阅读手册页。除 cd（它内置在 bash 中）之外，这几乎适用于这里所涉及的所有命令（比如 man ls）。

touch

发布时间 :2007-01-26 21:11:36

现在，我们将要快速地看一看 cp 和 mv 命令，这些命令用来复制、重命名以及移动文件和目录。为了开始该概述，我们将首先用 touch 命令在 /tmp 中创建一个文件：

```
$ cd /tmp  
$ touch copyme
```

如果文件存在，touch 命令将更新文件的“mtime”（请回想 ls -l 输出中的第六栏）。如果文件不存在，那么将创建一个新的空文件。现在您应该有一个大小为零的 /tmp/copyme 文件。

echo 和重定向

发布时间 :2007-01-26 21:12:06

既然文件存在，我们来把一些数据添加到文件中。我们可以使用 echo 命令来完成，它带有自己参数，并且把这些参数打印到标准输出。首先，单独的 echo 命令是这样的：

```
$ echo "firstfile"
firstfile
```

echo 和重定向

发布时间 :2007-01-26 21:12:42

带有输出重定向的同样的 echo 命令为：

```
$ echo "firstfile" > copyme
```

大于符号告诉 shell 将 echo 的输出写到名为 copyme 的文件中。如果该文件不存在，将创建这个文件；如果该文件存在，将覆盖这个文件。通过输入 ls -l，我们可以看到 copyme 文件为 10 个字节长，因为它包括 firstfile 这个词和换行符：

```
$ ls -l copyme
-rw-r--r--  1 root   root      10 Dec 28 14:13 copyme
```

cat 和 cp

发布时间 :2007-01-26 21:13:18

为了在终端显示文件的内容，要使用 cat 命令：

```
$ cat copyme  
firstfile
```

现在，我们可以使用 cp 命令的基本调用来由原始的 copyme 文件创建 copiedme 文件：

```
$ cp copyme copiedme
```

通过观察，我们发现它们确实是相互独立的文件；它们的索引节点号不同：

```
$ ls -li copyme copiedme  
648284 copiedme 650704 copyme
```

mv

发布时间 :2007-01-26 21:13:59

现在，我们来用“mv”命令将“copiedme”重命名为“movedme”。其索引节点号将仍然是同一个；但是，指向该索引节点的文件名将改变。

```
$ mv copiedme movedme
$ ls -li movedme
648284 movedme
```

只要目标文件和源文件驻留在同一文件系统上，被移动的文件索引节点号就将仍然不变。

创建链接和删除文件

创建链接和删除文件 - 硬链接

发布时间 :2007-01-26 21:14:45

当谈及目录条目和索引节点之间关系时，我们提到了链接这个术语。Linux 实际有两种链接。到此为止我们所讨论的这种链接叫硬链接。一个给定的索引节点可以有任意数目的硬链接，该索引节点一直存在于文件系统，直到所有的硬链接消失。可以使用 `ln` 命令来创建新的硬链接：

```
$ cd /tmp
$ touch firstlink
$ ln firstlink secondlink
$ ls -li firstlink secondlink
15782 firstlink 15782 secondlink
```

硬链接（续）

发布时间 :2007-01-26 21:15:34

您可以看到，硬链接工作于索引节点级别，指向特殊的文件。在 Linux 系统上，硬链接有几个局限性。第一，您只能给文件建立硬链接，而不能给目录建立硬链接。的确如此；即便 `.` 和 `..` 是系统给目录创建的硬链接，也不允许您（“root”用户也不行）创建任何您自己的硬链接。

硬链接的第二个局限性是它们不能跨文件系统。这意味着，如果您的 `/` 和 `/usr` 存在于不同的文件系统，您不能创建从 `/usr/bin/bash` 到 `/bin/bash` 的链接。

符号链接

发布时间 :2007-01-26 21:16:10

实际上，符号链接（symbolic link，或“symlinks”）比硬链接更常用到。符号链接是一种专门的文件类型，在这种文件类型中，链接通过名称引用另一个文件，而不是直接引用索引节点。符号链接不阻止文件被删除；如果目标文件消失，那么符号链接仅仅是不可用，或“被破坏”。

符号链接（续）

发布时间 :2007-01-26 21:16:46

通过将 -s 选项传给 ln，可以创建符号链接。

```
$ ln -s secondlink thirdlink
$ ls -l firstlink secondlink thirdlink
-rw-rw-r-- 2 agriffis agriffis    0 Dec 31 19:08 firstlink
-rw-rw-r-- 2 agriffis agriffis    0 Dec 31 19:08 secondlink
lrwxrwxrwx 1 agriffis agriffis   10 Dec 31 19:39 thirdlink -> secondlink
```

在 ls -l 输出中，可以用三种方式区分符号链接和一般文件。第一，请注意第一栏包含一个 l 字符的输出表明是符号链接。第二，符号链接的大小是目标文件（本例是 secondlink）的字符数。第三，输出的最后一栏显示目标文件名。

符号链接深入

发布时间 :2007-01-26 21:18:45

符号链接通常比硬链接更灵活。您可以给任何类型的文件系统对象（包括目录）创建符号链接。又因为符号链接的实现是基于路径的（而不是索引节点），所以创建指向另一个文件系统上的对象的符号链接是完全可行的。但是，这一事实也使符号链接理解起来很复杂。

请考虑我们想要在 /tmp 中创建一个指向 /usr/local/bin 的链接的情况。我们应该输入：

```
$ ln -s /usr/local/bin bin1
$ ls -l bin1
lrwxrwxrwx  1 root  root      14 Jan  1 15:42 bin1 -> /usr/local/bin
```

或者还可以输入：

```
$ ln -s ../usr/local/bin bin2
$ ls -l bin2
lrwxrwxrwx  1 root  root      16 Jan  1 15:43 bin2 -> ../usr/local/bin
```

您可以看到，两个符号链接都指向同一目录。但是，如果我们的第二个符号链接在任何时刻被移动到另一个目录，由于相对路径的缘故，它将遭到“破坏”。

```
$ ls -l bin2
lrwxrwxrwx  1 root  root      16 Jan  1 15:43 bin2 -> ../usr/local/bin
$ mkdir mynewdir
$ mv bin2 mynewdir
$ cd mynewdir
$ cd bin2
bash: cd: bin2: No such file or directory
```

因为 /tmp/usr/local/bin 这个目录不存在，我们不能再把目录转到 bin2；换句话说，bin2 现在被破坏了。

由于这个原因，有时避免用相对路径信息来创建符号链接是个好主意。但是，在许多情况下，相对的符号链接很管用。请考虑一个示例，在这个示例中您想要给 /usr/bin 中的一个程序创建一个别名：

```
# ls -l /usr/bin/keychain
-rwxr-xr-x  1 root  root     10150 Dec 12 20:09 /usr/bin/keychain
```

作为 root 用户，您可能想要给 “keychain” 创建一个别名，比如 “kc”。在这个示例中，我们有 root 访问权，由 bash 提示符改变为 “#” 可以证明。我们之所以需要 root 访问权是因为一般用户不能在 /usr/bin 中创建文件。作为 root 用户，我们可以像下面这样给 keychain 创建一个别名：

```
# cd /usr/bin
# ln -s /usr/bin/keychain kc
```

当这个解决方法起作用时，如果我们想要把两个文件都移到 /usr/local/bin 时，它将会出现问题。

```
# mv /usr/bin/keychain /usr/bin/kc /usr/local/bin
```

因为在符号链接中，我们使用了绝对路径，而我们的 kc 符号链接仍然指向 /usr/bin/keychain，它已不存在了——另一个被破坏的符号链接。符号链接中的相对路径和绝对路径都各具优点，您应该使用适合于您的特殊应用的路径类型。一般情况下，相对路径或绝对路径都能工作得很好。在这种情况下，下面的示例将起作用：

```
# cd /usr/bin
# ln -s keychain kc
# ls -l kc
lrwxrwxrwx  1 root  root      8 Jan  5 12:40 kc -> keychain
```

rm

发布时间 :2007-01-26 21:19:18

既然我们知道怎样使用 cp、mv 和 ln，现在我们该学习怎样把对象从文件系统中删除了。通常，这用 rm 命令来完成。要删除文件，只需在命令行中指定它们：

```
$ cd /tmp
$ touch file1 file2
$ ls -l file1 file2
-rw-r--r--  1 root  root      0 Jan  1 16:41 file1
-rw-r--r--  1 root  root      0 Jan  1 16:41 file2
$ rm file1 file2
$ ls -l file1 file2
ls: file1: No such file or directory
ls: file2: No such file or directory
```

rmdir

发布时间 :2007-01-26 21:19:50

要删除目录，您有两种选择。您可以删除目录中所有的对象，然后使用 rmdir 来删除目录本身：

```
$ mkdir mydir
$ touch mydir/file1
$ rm mydir/file1
$ rmdir mydir
```

rm 和 目录

发布时间 :2007-01-26 21:20:24

或者，您可以使用 `rm` 命令的 `recursive force` 选项来告诉 `rm` 删除您指定的目录以及目录中包含的所有对象：

```
$ rm -rf mydir
```

一般情况下，`rm -rf` 是删除目录树的首选方法。在使用 `rm -rf` 时要十分小心，因为它的功能可以被很好地利用，也可能会因使用不当造成恶果。

介绍通配符

介绍通配符

发布时间 :2007-01-26 21:21:27

在您日常的 Linux 使用中，有很多时候您可能需要一次对多个文件系统对象执行单一操作（比如 rm）。在这些情况下，在命令行中输入许多文件通常让人感到厌烦：

```
$ rm file1 file2 file3 file4 file5 file6 file7 file8
```

介绍通配符（续）

发布时间 :2007-01-26 21:22:03

为了解决这个问题，您可以利用 Linux 内置的通配符支持。这种支持也叫做“globbing”（由于历史原因），允许您通过使用通配符模式一次指定多个文件。Bash 和其它 Linux 命令将通过在磁盘上查找并找到任何与之匹配的文件来解释这种模式。因此，如果在当前工作目录中，您有从 file1 到 file8 的文件，那么您可以输入下面的命令来删除这些文件：

```
$ rm file[1-8]
```

或者，如果您只想要删除文件名以 file 开头的文件，您可以输入：

```
$ rm file*
```

理解不匹配

发布时间 :2007-01-26 21:22:38

或者，如果您想要列出 /etc 中以 g 开头的所有文件系统对象，您可以输入：

```
$ ls -d /etc/g*  
/etc/gconf /etc/ggi /etc/gimp /etc/gnome /etc/gnome-vfs-mime-magic /etc/gpm /etc/group /etc/group-
```

现在，如果您指定了没有任何文件系统对象与之匹配的模式，会怎么样呢？在下面的示例中，我们试图列出 /usr/bin 中以 asdf 开头并且以 jkl 结尾的所有文件：

```
$ ls -d /usr/bin/asdf*jkl  
ls: /usr/bin/asdf*jkl: No such file or directory
```

理解不匹配（续）

发布时间 :2007-01-26 21:23:10

这里是对所发生情况的说明。通常，当我们指定一种模式时，该模式与底层系统上的一个或多个文件匹配，bash 以空格隔开的所有匹配对象的列表来替换该模式。但是，当模式不能找到匹配对象时，bash 将不理睬参数、通配符等等，保留原样。因此，当 “ls” 不能找到文件 /usr/bin/asdf**ijkl* 时，它会报错。此处的有效的规则是：glob 模式只在与文件系统中的对象匹配时才可以进行扩展。

通配符语法： *

发布时间 :2007-01-26 21:23:44

既然我们理解了 globbing 如何工作，我们来复习一下通配符语法。您可以使用几个用于通配符扩展的专门的字符；这里有一个：

*

* 将与零个或多个字符匹配。这就是说“什么都可以”。例子：

/etc/g* 与 /etc 中以 g 开头的所有文件匹配。

/tmp/my*1 与 /tmp 中以 my 开头，并且以 1 结尾的所有文件匹配。

通配符语法： ?

发布时间 :2007-01-26 21:24:22

?

? 与任何单个字符匹配。例子：

myfile? 与文件名为 myfile 后跟单个字符的任何文件匹配。
/tmp/notes?txt 将与 /tmp/notes.txt 和 /tmp/notes_txt 都匹配，如果它们存在。

通配符语法： []

发布时间 :2007-01-26 21:25:09

[]

该通配符与 ? 相似，但允许指定得更确切。要使用该通配符，把您想要匹配的所有字符放在 [] 内。结果的表达式将与 [] 中任一字符相匹配。您也可以使用 - 来指定范围，甚至还可以组合范围。例子：

myfile[12] 将与 myfile1 和 myfile2 匹配。只要当前目录中至少有一个这样的文件存在，该通配符就可以进行扩展。

[Cc]hange[Ll]og 将与 Changelog、ChangeLog、changeLog 以及 changelog 匹配。您可以看到，与大写形式的变形匹配时，使用括弧通配符很有用。

ls /etc/[0-9]* 将列出 /etc 中以数字开头的所有文件。

ls /tmp/[A-Za-z]* 将列出 /tmp 中以大写字母或小写字母开头的所有文件。

通配符语法： [!]

发布时间 :2007-01-26 21:25:41

[!]

除了不与括弧中的任何字符匹配外，[!] 构造与 [] 构造类似，只要不是列在 [! 和] 之间的字符，它将与任何字符匹配。例子：

rm myfile[!9] 将删除除 myfile9 之外的名为 myfile 加一个字符的所有文件。

通配符告诫说明

发布时间 :2007-01-26 21:26:16

这里有一些使用通配符时应该注意的告诫说明。由于 bash 对与通配符相关的字符（?、[、]、*）进行特别处理，因此您将包含这些字符的参数输入到命令中时，需要特别小心。例如，如果您想要创建一个包含字符串 [fo]* 的文件，下面这个命令可能不会执行您想要做的事：

```
$ echo [fo]* > /tmp/mynewfile.txt
```

如果 [fo]* 这个模式与当前工作目录中的任何文件匹配，那么您将在 /tmp/mynewfile.txt 内发现那些文件的名称，而不是您所期望的文字 [fo]*。解决方法是什么呢？嗯，一种方法是用单引号把这些字符括起来，这将告诉 bash 单纯地执行，而不会对其进行通配符扩展：

```
$ echo '[fo]*' > /tmp/mynewfile.txt
```

通配符告诫说明（续）

发布时间 :2007-01-26 21:26:59

采用这种方法，您的新文件将包含所期望的文字的 [fo]*。另一种方法是，您可以使用反斜杠，告诉 bash [、] 和 * 应该被当成文字处理，而不是被当成通配符处理：

```
$ echo \[fo\]* > /tmp/mynewfile.txt
```

两种方法都能同样地起作用。既然我们谈到反斜杠扩展，那么现在是时候提一提了，为了指定文字 \，您可以将它放入单引号中，或者也可以输入 \\（它将被扩展为 \）。

单引号与双引号的对比

发布时间 :2007-01-26 21:27:41

请注意双引号的作用和单引号很接近，而双引号还允许 bash 做一些有限的扩展。因此，当您确实想要把文字文本传给命令时，单引号是最好的选择。要获取关于通配符扩展更多的信息，请输入 `man 7 glob`。要获取关于 bash 中引号作用的更多信息，请输入 `man 8 glob`，并阅读题为 QUOTING 的章节。

海量Linux技术文章

海量 Linux 技术文章

发布时间 :2006-11-24 16:50:29

下面是linux技术文章快速入口。需要联网：

[Linux 技术交流](#)

<http://www.linuxdiyf.com/bbs/forum-3-1.html>

[Linux 应用](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=1>

[Linux 安装及学习指导](#)

<http://www.linuxdiyf.com/bbs/forum-45-1.html>

[Linux 系统安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=11>

[Linux 学习指导](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=12>

[Linux 软件安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=13>

[shell](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=3>

[Linux 壁纸](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=4>

[红旗](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=5>

[Redhat](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=6>

[SuSE](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=7>

Linux 认证

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=9>

Linux下载分享{酷件、书籍、视频分享 }

<http://www.linuxdiyf.com/bbs/forum-6-1.html>

服务器应用

<http://www.linuxdiyf.com/bbs/forum-7-1.html>

数据库应用

<http://www.linuxdiyf.com/bbs/forum-8-1.html>

Linux 编程与内核

<http://www.linuxdiyf.com/bbs/forum-9-1.html>

UniX 技术文章

<http://www.linuxdiyf.com/bbs/forum-32-1.html>

Linux 业界声音、新闻

<http://www.linuxdiyf.com/bbs/forum-11-1.html>

Linux 人才招聘信息

<http://www.linuxdiyf.com/bbs/forum-46-1.html>

网络转载，感谢原创作者！

制作：红联Linux论坛

祝您阅读愉快！

Linux培训系列

第二讲

将向您演示如何使用正则表达式在文件中搜索文本模式。接着，我们将向您介绍文件系统层次结构标准（Filesystem Hierarchy Standard，或者称为 FHS），并向您演示如何在您的系统上定位文件。然后，我们将通过在后台运行 Linux 进程、列出进程清单、从终端上拆离进程以及更多内容，向您演示如何完全控制 Linux 进程。最后，我们将向您简要介绍 shell 管道、重定向和文本处理命令。

红联Linux论坛是致力于Linux技术讨论的站点，目前网站收录的文章及教程基本能满足不同水平的朋友学习。

红联Linux门户：www.linux110.com

红联Linux论坛：www.linuxdiyf.com/bbs

下载:Linux电子书籍：

<http://www.linux286.com/linux/linuxdzsj.htm>

目录

正则表达式

- 正则表达式
- 与 glob 的比较
- 简单子串
- 理解简单子串
- 元字符
- 使用 []
- 使用 [^]
- 区别语法
- “ * ” 元字符
- 行的开始和结束
- 完整行正则表达式

FHS 与查找文件

- FHS 与查找文件 - 文件系统层次结构标准
- 两个独立的 FHS 类别
- /usr 中的辅助层次结构
- 查找文件
- PATH
- 修改 PATH
- 关于 “ which ” 的一切
- “ which -a ”
- whereis
- find
- find 和通配符
- 在 find 中忽略大小写
- find 和正则表达式
- find 和类型
- find 和 mtime
- daystart 选项
- size 选项
- 处理找到的文件
- 定位
- 使用 updatedb
- slocate

进程控制

- 进程控制 - 启动 xeyes
- 停止进程
- fg 和 bg
- 使用 “ & ”
- 多个后台进程
- 介绍信号
- SIGTERM 和 SIGINT
- 功能强大的 kill
- nohup
- 使用 ps 来列出进程
- 查看森林（层次结构）和树
- “ u ” 和 “ l ” ps 选项
- 使用 “ top ”

nice

renice

文本处理

文本处理 - 再述重定向

管道示例

解压缩管道

更长的管道

马上开始文本处理

cat、*sort* 和 *uniq*

wc、*head* 和 *tail*

tac、*expand* 和 *unexpand*

cut、*nl* 和 *pr*

tr、*sed* 和 *awk*

od、*split* 和 *fmt*

paste、*join* 和 *tee*

快要结束了！重定向

使用 *>>*;

Linux文章汇集

海量Linux技术文章

正则表达式

正则表达式

发布时间 :2007-01-26 22:02:44

正则表达式（也称为“regex”或“regexp”）是一种用来描述文本模式的特殊语法。在Linux系统上，正则表达式通常被用来查找文本的模式，以及对文本流执行“搜索-替换”操作以及其它功能。

与 glob 的比较

发布时间 :2007-01-26 22:03:34

当我们看到正则表达式时，您可能发现正则表达式的语法看起来与我们上一篇教程，但是，不要让它欺骗您；它们的类似性只是表面的。虽然正则表达式和文件名匹配替换模式可能看上去相类似，但是它们是根本不同的两种类型。

简单子串

发布时间 :2007-01-26 22:04:08

记住那个警告，让我们看一下最基本的正则表达式，简单子串。为了这样做，我们要使用 grep，它是一个扫描文件内容来查找适合特定正则表达式的命令。grep 打印与正则表达式匹配的每一行，并忽略与之不匹配的每一行：

```
$ grep bash /etc/passwd
operator:x:11:0:operator:/root:/bin/bash
root:x:0:0::/root:/bin/bash
ftp:x:40:1::/home/ftp:/bin/bash
```

在上面的命令中，grep 的第一个参数是一个正则表达式；第二个参数是一个文件名。grep 读取 /etc/passwd 中的每一行并对它应用简单子串正则表达式 bash 来查找匹配项。如果找到一个匹配项，那么 grep 打印出整行；否则，忽略该行。

理解简单子串

发布时间 :2007-01-26 22:04:41

一般来说，如果您正在搜索一个子串，那么您可以不提供任何“特殊”字符，而只是逐字地指定文本。只有在子串包含 +、.、*、[、] 或 \（在这样的情况下，这些字符需要用引号括起来并在它们的前面使用反斜杠）才需要做特殊的事情。下面是简单子串正则表达式几个其它示例：

/tmp（扫描查找文字串 /tmp）

“ \[box\] ”（扫描查找文字串 [box]）

“ *funny* ”（扫描查找文字串 *funny*）

“ ld\.so ”（扫描查找文字串 ld.so）

元字符

发布时间 :2007-01-26 22:05:15

使用正则表达式，可以利用元字符来执行比我们至今已研究过的示例复杂得多的搜索。这些元字符中的一个是一个点（.），它与任何单个字符匹配：

```
$ grep dev.hda /etc/fstab
/dev/hda3    /          reiserfs    noatime,ro 1 1
/dev/hda1    /boot      reiserfs    noauto,noatime,notail 1 2
/dev/hda2    swap       swap        sw 0 0
#/dev/hda4   /mnt/extra reiserfs    noatime,rw 1 1
```

在本示例中，文字文本 dev.hda 没有出现在 /etc/fstab 中的任何一行中。但是，grep 扫描这些行时没有查找文字 dev.hda 字符串，而是查找 dev.hda 模式。请记住 . 将与任何单个字符相匹配。正如您看到的，. 元字符在功能上等价于 glob 扩展中的 * 元字符的工作原理。

使用 []

发布时间 :2007-01-26 22:05:52

如果我们希望与比 . 更具体一点地来匹配字符，那么我们可以使用 [和]（方括号）来指定要匹配的字符子集：

```
$ grep dev.hda[12] /etc/fstab
/dev/hda1    /boot      reiserfs    noauto,noatime,notail 1 2
/dev/hda2    swap       swap        sw 0 0
```

正如您看到的，这个特殊语法的作用与“glob”文件名扩展中的 [] 相同。同样，这是学习正则表达式的难点之一——这个语法与“glob”文件名扩展语法类似，但又不尽相同，它经常给学习正则表达式的人带来困惑。

使用 [^]

发布时间 :2007-01-26 22:06:21

通过使 [后面紧跟一个 ^，您可以使方括号中的意思相反。在本例中，方括号将与未列在方括号内的任意字符匹配。同样，请注意我们在正则表达式中使用 [^]，而在 glob 中使用 [!]

```
$ grep dev.hda[^12] /etc/fstab
/dev/hda3    /          reiserfs    noatime,ro 1 1
#/dev/hda4   /mnt/extra reiserfs     noatime,rw 1 1
```

区别语法

发布时间 :2007-01-26 22:06:54

注意下面一点很重要：方括号内部的语法根本不同于正则表达式其它部分中的语法。例如，如果在方括号内放置一个 `.`，那么它允许方括号与文字 `.` 匹配，就象上面示例中的 1 和 2。比较起来，除非有 `\` 作为前缀，否则方括号外面的文字 `.` 被解释为一个元字符。通过输入如下命令，我们可以利用这一事实来打印 `/etc/fstab` 中包含文字串 `dev.hda` 的所有行的列表：

```
$ grep dev[.]hda /etc/fstab
```

或者，我们也可以输入：

```
$ grep "dev\\.hda" /etc/fstab
```

这两个正则表达式都不可能与您 `/etc/fstab` 文件中的任何行相匹配。

“ * ” 元字符

发布时间 :2007-01-26 22:07:25

某些元字符本身不匹配任何字符，但却修改前一个字符的含义。一个这样的元字符是 *（星号），它用来与前一个字符的零次或者多次重复出现相匹配。这里是一些示例：

ab*c（与 abbbbc 匹配但不与 abqc 匹配）
ab*c（与 abc 匹配但不与 abbqbbc 匹配）
ab*c（与 ac 匹配但不与 cba 匹配）
b[cq]*e（与 bqe 匹配但不与 eb 匹配）
b[cq]*e（与 bccqqe 匹配但不与 bccc 匹配）
b[cq]*e（与 bqqcce 匹配但不与 cqe 匹配）
b[cq]*e（与 bbbeee 匹配）
.*（与任何字符串匹配）
foo.*（与以 foo 开始的任何字符串相匹配）

ac 行与正则表达式 ab*c 相匹配，因为星号也允许前面的表达式（b）出现零次。请注意解释 * 正则表达式元字符所用的方法与解释 * glob 字符的方法根本不同。

行的开始和结束

发布时间 :2007-01-26 22:07:56

我们在这里要详细描述的最后几个元字符是 ^ 和 \$ 元字符，它们用来分别与行的开始和结束相匹配。通过在正则表达式开始处使用一个 ^ ，您可以将您的模式“锚定”在行的开始。在下面的示例中，我们使用 ^# 正则表达式来与以 # 字符开始的任何行相匹配：

```
$ grep ^# /etc/fstab
# /etc/fstab: static file system information.
#
```

完整行正则表达式

发布时间 :2007-01-26 22:08:27

可以组合 `^` 和 `$` 来与完整的行相匹配。例如，下面的正则表达式将与以 `#` 字符开始并以 `.` 字符结束的行相匹配，在中间可以有任意多个其它字符：

```
$ grep '^#.*$' /etc/fstab  
# /etc/fstab: static file system information.
```

在上面的示例中，我们用单引号将我们的正则表达式括起来以阻止 shell 解释 `$`。在不使用单引号的情况下，`grep` 甚至没有机会查看 `$`，`$` 就从我们的正则表达式上消失了。

FHS 与查找文件

FHS 与查找文件 - 文件系统层次结构标准

发布时间 :2007-01-26 22:09:45

Filesystem Hierarchy Standard 是指定 Linux 系统上目录布局的文档。FHS 被设计来提供一个通用布局以简化与分布无关的软件开发。FHS 指定下列目录（直接来自 FHS 规范）：

- /（根目录）
- /boot（引导装入程序的静态文件）
- /dev（设备文件）
- /etc（主机特定的系统配置）
- /lib（基本共享库和核心模块）
- /mnt（临时挂装文件系统的挂装点）
- /opt（附加的应用程序软件包）
- /sbin（基本系统二进制文件）
- /tmp（临时文件）
- /usr（辅助层次结构）
- /var（可变数据）

两个独立的 FHS 类别

发布时间 :2007-01-26 22:10:21

FHS 的布局规范基于存在两个独立的文件类别：可共享与不可共享以及可变与静态这一思想。可共享数据能在主机之间被共享；不可共享数据特定于给定主机（例如配置文件）。可变数据可以被修改；静态数据不可以被修改（除了在系统安装和维护阶段）。

下面的表格概述了四种可能的组合，并列出了与那些类别相符的目录示例。这个表还是直接取自 FHS 规范：

+-----+-----+-----+			
	可共享	不可共享	
+-----+-----+-----+			
静态	/usr	/etc	
	/opt	/boot	
+-----+-----+-----+			
可变	/var/mail	/var/run	
	/var/spool/news	/var/lock	
+-----+-----+-----+			

/usr 中的辅助层次结构

发布时间 :2007-01-26 22:11:04

在 /usr 下，您会发现一个看上去与根文件系统非常相似的辅助层次结构。当机器打开并运行时，/usr 的存在并不重要，所以能在网络上共享它（“可共享”），或者从 CD-ROM 上挂装它（“静态”）。大多数 Linux 设置不利用 /usr 的共享，但是理解根目录中主层次结构和 /usr 中辅助层次结构之间的区别的用处是有价值的。

这就是我们要说的有关 Filesystem Hierarchy Standard 的所有内容。该文档本身非常具有可读性，所以您应该去看一下。我们承诺如果您读了它，那么您将对 Linux 文件系统理解得更多。

查找文件

发布时间 :2007-01-26 22:11:36

Linux 系统通常包含数十万个文件。可能您非常精明能干，从未丢失它们中的任何一个，但是更可能的是，您偶尔在查找一个文件时需要帮助。Linux 上有几个不同的工具用于查找文件。下面的演示将向您介绍它们，并帮助您选择适合您的工作的工具。

PATH

发布时间 :2007-01-26 22:12:14

当您在命令行上运行程序时，bash 实际上搜索目录列表来查找您所请求的程序。例如，当您输入 ls，bash 实质上不知道 ls 程序位于 /usr/bin。但是，bash 引用一个名为 PATH 的环境变量，它是一个用冒号分隔的目录列表。我们可以检查 PATH 的值：

```
$ echo $PATH  
/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin:/usr/X11R6/bin
```

给定了 PATH 的值（您的可以不同），bash 将首先检查 /usr/local/bin，然后是 /usr/bin 以搜索 ls 程序。ls 最有可能被保存在 /usr/bin 内，所以 bash 在那里停止。

修改 PATH

发布时间 :2007-01-26 22:12:46

您可以通过在命令行上为 PATH 指派元素来扩充它：

```
$ PATH=$PATH:~/bin
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin:/usr/X11R6/bin:/home/agriffis/bin
```

您也可以除去 PATH 上的元素，尽管这不是那么容易，因为您不能引用现有的 \$PATH。最好的办法是简单输入您想要的新 PATH：

```
$ PATH=/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:~/bin
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/home/agriffis/bin
```

关于 “ which ” 的一切

发布时间 :2007-01-26 22:13:17

通过使用 which，您能查看 PATH 中是否有给定程序。例如，我们通过下面的命令发现 Linux 系统没有（普通的）sense：

```
$ which sense
which: no sense in (/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin:/usr/X11R6/bin)
```

在本示例中，我们成功定位 ls：

```
$ which ls
/usr/bin/ls
```

“ which -a ”

发布时间 :2007-01-26 22:13:47

最后，您应该知道 -a 标志，它使 which 向您显示您的 PATH 中给定程序的所有实例：

```
$ which -a ls
/usr/bin/ls
/bin/ls
```

whereis

发布时间 :2007-01-26 22:14:22

如果您不只对程序位置感兴趣，而且想要找到更多信息，那么可以尝试 whereis 程序：

```
$ whereis ls  
ls: /bin/ls /usr/bin/ls /usr/share/man/man1/ls.1.gz
```

这里我们看到 ls 出现在两个常见二进制位置 /bin 和 /usr/bin 中。另外，我们被告知手册页定位在 /usr/share/man。如果您要输入 man ls，那么这就是您将看到的手册页。

whereis 程序还具有搜索源代码、指定备用搜索路径和搜索不寻常项的能力。

find

发布时间 :2007-01-26 22:14:51

find 命令是您工具箱中的另一个工具。使用 find，您不会受限于程序；通过使用多种搜索标准，您能搜索您想要的任何文件。例如，要搜索 /usr/share/doc 目录下名为 README 的文件：

```
$ find /usr/share/doc -name README
/usr/share/doc/ion-20010523/README
/usr/share/doc/bind-9.1.3-r6/dhcp-dynamic-dns-examples/README
/usr/share/doc/sane-1.0.5/README
```

find 和通配符

发布时间 :2007-01-26 22:15:23

您可以在 -name 的参数中使用 “ glob ” 通配符，前提条件是您用双引号引用了它们或用反斜杠进行了转义（这样它们就能被完整地传递到 find 而不是被 bash 扩展）。例如，我们可能想要搜索带有扩展名的 README 文件：

```
$ find /usr/share/doc -name README\  
/usr/share/doc/iproute2-2.4.7/README.gz  
/usr/share/doc/iproute2-2.4.7/README.iproute2+tc.gz  
/usr/share/doc/iproute2-2.4.7/README.decnet.gz  
/usr/share/doc/iproute2-2.4.7/examples/diffserv/README.gz  
/usr/share/doc/pilot-link-0.9.6-r2/README.gz  
/usr/share/doc/gnome-pilot-conduits-0.8/README.gz  
/usr/share/doc/gimp-1.2.2/README.i18n.gz  
/usr/share/doc/gimp-1.2.2/README.win32.gz  
/usr/share/doc/gimp-1.2.2/README.gz  
/usr/share/doc/gimp-1.2.2/README.perl.gz  
[578 additional lines snipped]
```

在 find 中忽略大小写

发布时间 :2007-01-26 22:15:54

当然，您可能想要在搜索中忽略大小写：

```
$ find /usr/share/doc -name '[Rr][Ee][Aa][Dd][Mm][Ee]*'
```

或者，更简单：

```
$ find /usr/share/doc -iname readme\*
```

正如您看到的，您能使用 -iname 来进行不区分大小写的搜索。

find 和正则表达式

发布时间 :2007-01-26 22:16:24

如果您熟悉正则表达式，那么使用 `-regex` 选项将把输出限制成匹配某一模式的文件名。与 `-iname` 选项类似，它有一个相应的 `-iregex` 选项，该选项忽略模式中的大小写。例如：

```
$ find /etc -iregex '.*xt.*'  
/etc/X11/xkb/types/extra  
/etc/X11/xkb/semantics/xttest  
/etc/X11/xkb/compat/xttest  
/etc/X11/app-defaults/XTerm  
/etc/X11/app-defaults/XTerm-color
```

请注意：不象许多程序，`find` 要求指定的正则表达式与整个路径匹配，而不只是该路径的一部分。为此，指定前导和尾随的 `.*` 是必要的；只使用 `xt` 是不够的。

find 和类型

发布时间 :2007-01-26 22:16:55

-type 选项允许您查找某一类型的文件系统对象。可能的 -type 参数是 b（块设备）、c（字符设备）、d（目录）、p（命名管道）、f（常规文件）、l（符号链接）和 s（套接字）。例如，要在 /usr/bin 中搜索包含字符串 vim 的符号链接：

```
$ find /usr/bin -name '*vim*' -type l
/usr/bin/rvim
/usr/bin/vimdiff
/usr/bin/gvimdiff
```

find 和 mtime

发布时间 :2007-01-26 22:17:33

-mtime 选项允许您根据最近一次的修改时间来选择文件。mtime 的参数以 24 小时为单位，当输入时带加号（表示“之后”）或者减号（表示“之前”）时，它最有用。例如，考虑如下情形：

```
$ ls -l ?  
-rw----- 1 root  root      0 Jan  7 18:00 a  
-rw----- 1 root  root      0 Jan  6 18:00 b  
-rw----- 1 root  root      0 Jan  5 18:00 c  
-rw----- 1 root  root      0 Jan  4 18:00 d  
$ date  
Mon Jan  7 18:14:52 EST 2002
```

您可以搜索在过去的 24 小时之内创建的文件：

```
$ find . -name \? -mtime -1  
./a
```

或者您可以搜索在当前 24 小时周期之前创建的文件：

```
$ find . -name \? -mtime +0  
./b  
./c  
./d
```

-daystart 选项

发布时间 :2007-01-26 22:18:03

如果您另外指定了 -daystart 选项，那么时间周期以今天的开始时为开始，而不是 24 小时之前。例如，这是昨天和前天创建的一组文件：

```
$ find . -name \? -daystart -mtime +0 -mtime -3
./b
./c
$ ls -l b c
-rw----- 1 root  root    0 Jan 6 18:00 b
-rw----- 1 root  root    0 Jan 5 18:00 c
```

-size 选项

发布时间 :2007-01-26 22:18:34

-size 选项允许您根据文件的大小来查找它们。缺省情况下，-size 的参数是 512 个字节的块，但是添加后缀可以使操作更简便。可用的后缀是 b（512 字节的块）、c（字节）、k（千字节）和 w（2 字节的字）。另外，您可以在前放置加号（“大于”）或者减号（“小于”）。

例如，要在 /usr/bin 中查找小于 50 个字节的常规文件：

```
$ find /usr/bin -type f -size -50c
/usr/bin/krdb
/usr/bin/run-nautilus
/usr/bin/sgmlwhich
/usr/bin/muttbug
```

处理找到的文件

发布时间 :2007-01-26 22:19:06

您可能在想如何处理所有这些找到的文件！不用担心，通过使用 `-exec` 选项，`find` 具有对它找到的文件进行操作的能力。这个选项接受命令行作为它的参数来执行，它以；中断，并用文件名来替换任何出现的 `{}`。下面这个示例可以帮助您完全理解它：

```
$ find /usr/bin -type f -size -50c -exec ls -l '{}' ';'
-rwxr-xr-x  1 root  root    27 Oct 28 07:13 /usr/bin/kbdb
-rwxr-xr-x  1 root  root    35 Nov 28 18:26 /usr/bin/run-nautilus
-rwxr-xr-x  1 root  root    25 Oct 21 17:51 /usr/bin/sgmlwhich
-rwxr-xr-x  1 root  root    26 Sep 26 08:00 /usr/bin/muttbug
```

正如您看到的，`find` 是一个功能非常强大的命令。在 UNIX 和 Linux 开发的几年中，它获得了发展。`find` 中还有许多其它有用的选项。您可以在 `find` 手册页中学习它们。

定位

发布时间 :2007-01-26 22:19:35

我们已经学习了 which、whereis 和 find。您可能已经注意到执行 find 要花一些时间，因为它需要读取它正在搜索的每个目录。事实表明 locate 命令可以通过依靠外部数据库来加速操作。

locate 命令与路径名的任何部分相匹配，而不只是文件本身。例如：

```
$ locate bin/l  
/var/ftp/bin/l  
/bin/l  
/sbin/lsmo  
/sbin/lspci  
/usr/bin/l  
/usr/bin/l  
/usr/sbin/l
```

使用 updatedb

发布时间 :2007-01-26 22:20:11

大多数 Linux 系统包含一个周期性的进程来更新这个数据库。如果您的系统在运行上述命令时返回如下错误，那么您需要运行 updatedb 来生成搜索数据库：

```
$ locate bin/l  
locate: /var/spool/locate/locatedb: No such file or directory  
$ su  
Password:  
# updatedb
```

运行 updatedb 命令可能要花很长时间。如果您硬盘的噪音很大，那么将听到许多吵闹声，因为这正在为整个文件系统建立索引。

slocate

发布时间 :2007-01-26 22:20:41

在 Linux 的许多分发版（distribution）中，locate 命令已经被 slocate 所替代。通常有一个至“locate”的符号链接，这样您不需要记住拥有的是哪一个。slocate 代表“安全定位（secure locate）”。它将许可权信息存储在数据库中，这样普通用户不能以别的方式窥探他们不能读取的目录。slocate 的用法信息在本质上与 locate 的信息相同，尽管输出可能不同（取决于正在运行命令的用户）。

进程控制

进程控制 - 启动 xeyes

发布时间 :2007-01-26 22:21:26

为了学习进程控制，我们首先需要启动一个进程：

```
$ xeyes -center red
```

您将注意到弹出一个 xeyes 窗口，红色眼球跟随您的鼠标在屏幕上移动。您可能还会注意到在终端上没有新的提示符。

停止进程

发布时间 :2007-01-26 22:22:01

为了恢复提示符，您可以输入 Control-C（通常写为 Ctrl-C 或 ^C）：

```
^C
$
```

您获得了一个新的 bash 提示符，但 xeyes 窗口消失了。事实上，整个进程已被杀死。如果不通过 Control-C 来杀死它，我们还可以使用 Control-Z 来使它只是停止：

```
$ xeyes -center red
^Z
[1]+  Stopped                  xeyes -center red
$
```

这次您获得了一个新的 bash 提示符，并且 xeyes 窗口依然保留着。但是，如果您试图稍微移动这个窗口，那么将注意到眼球在某一位置被冻结了。如果 xeyes 窗口被另一个窗口遮盖，然后又出现，那么您将看到它根本不会重绘眼睛。进程没有做任何操作。事实上，它是“被停止了”。

fg 和 bg

发布时间 :2007-01-26 22:22:39

为了使进程 “重新活动” 并再次运行，我们能用 bash 内置的 fg 使它在前台运行：

```
$ fg
```

```
xeyes和xeyes
```

```
^Z
[1]+  Stopped                  xeyes -center red
$
```

现在用 bash 内置的 bg 来继续在后台运行它：

```
$ bg
[1]+ xeyes -center red &
$
```

好极了！现在 xeyes 进程在后台运行，并且出现了新的正在工作的 bash 提示符。

使用 “ & ”

发布时间 :2007-01-26 22:23:11

如果我们一开始想要在后台启动 xeyes（而不是使用 Control-Z 和 bg），那么我们只须在 xeyes 命令行的最后添加一个 “&”：

```
$ xeyes -center blue &  
[2] 16224
```

多个后台进程

发布时间 :2007-01-26 22:23:47

现在红色和蓝色 xeyes 都在后台运行。我们可以用 bash 内置的 jobs 列出这些作业：

```
$ jobs -l
[1]- 16217 Running          xeyes -center red &
[2]+ 16224 Running          xeyes -center blue &
```

左列中的号码是当作业被启动时，bash 指定给它们的作业号码。作业 2 有一个 +（加号），这表示它是“当前作业”，这意味着输入 fg 将把它带到前台。您也可以通过指定作业号码将指定的作业带到前台，换句话说，fg 1 使红色 xeyes 成为前台任务。下一列是包含在列表中的进程标识或者是 pid，由于 jobs 的 -l 选项可得到该列表。最后，这两个作业当前都是“Running”，并且它们的命令行都列在右边。

介绍信号

发布时间 :2007-01-26 22:24:22

为了杀死、停止或者继续运行进程，Linux 使用了一种称为“信号”的特殊通讯方式。通过将某一信号发送给进程，您可以使它中断、停止或执行其它操作。这就是当您输入 Control-C、Control-Z 或使用 bg 或 fg 内置命令时真正执行的操作 — 您正使用 bash 将一个特殊信号发送给进程。还可以通过使用 kill 命令并在命令行上指定 pid（进程标识）来发送这些信号：

```
$ kill -s SIGSTOP 16224
$ jobs -l
[1]- 16217 Running          xeyes -center red &
[2]+ 16224 Stopped (signal) xeyes -center blue
```

正如您看到的，kill 不一定“杀死”进程，尽管它能这样做。通过使用“-s”选项，kill 能将任何信号发送给进程。当分别将 SIGINT、SIGSTOP 或 SIGCONT 信号发送给进程时，Linux 就杀死、停止或继续运行这些进程。您还可以将其它信号发送给进程；这些信号中的一些可能是用与应用程序相关的方法来解释的。

SIGTERM 和 SIGINT

发布时间 :2007-01-26 22:24:52

如果您想杀死进程，那么有几种选择。缺省情况下，kill 发送 SIGTERM，它与 Control-C 著名的 SIGINT 不同，但是通常具有相同的结果：

```
$ kill 16217
$ jobs -l
[1]- 16217 Terminated          xeyes -center red
[2]+ 16224 Stopped (signal)     xeyes -center blue
```

功能强大的 kill

发布时间 :2007-01-26 22:25:28

进程可以自己选择或者由于被停止或由于某种原因“被阻塞”而忽略 SIGTERM 和 SIGINT。在这些情况下可能需要使用功能强大的 SIGKILL 信号。进程不能忽略 SIGKILL：

```
$ kill 16224
$ jobs -l
[2]+ 16224 Stopped (signal)      xeyes -center blue
$ kill -s SIGKILL
$ jobs -l
[2]+ 16224 Interrupt           xeyes -center blue
```

nohup

发布时间 :2007-01-26 22:26:00

您启动作业的终端被称为这个作业的控制终端。当您注销时，一些 shell（缺省情况下不是 bash）将向这些后台作业传送 SIGHUP 信号，从而导致这些进程退出。为了保护进程以免产生这种行为，当您启动进程时，请使用 nohup：

```
$ nohup make &  
$ exit
```

使用 ps 来列出进程

发布时间 :2007-01-26 22:26:36

我们较早使用的 jobs 命令只列出了从您 bash 会话上启动的进程。为了查看您系统上所有的进程，请使用同时带有 a 和 x 选项的 ps：

```
$ ps ax
PID TTY    STAT  TIME COMMAND
  1 ?      S     0:04 init [3]
  2 ?      SW    0:11 [keventd]
  3 ?      SWN   0:13 [ksoftirqd_CPU0]
  4 ?      SW    2:33 [kswapd]
  5 ?      SW    0:00 [bdflush]
```

我们只列出了开始的几个进程，因为通常它是一个非常长的列表。这提供给您整台机器正在执行的进程的一个快照，但您可能要筛选掉一些信息。如果您要省略 ax，那么将只看到您拥有的并具有控制终端的进程。命令 ps x 将显示您所有的进程，甚至那些没有控制终端的进程。如果您要使用 ps a，那么可以获取附加在终端上的每人的进程列表。

查看森林（层次结构）和树

发布时间 :2007-01-26 22:27:06

您也可以列出有关每个进程的不同信息的列表。使用 --forest 选项可以很容易地查看进程的层次结构，它将向您显示系统上的各种进程是如何相互关联的。当一个进程启动一个新进程时，那个新进程被称为“子”进程。在 --forest 列表中，父进程出现在左侧，而子进程作为分支出现在右侧：

```
$ ps x --forest
  PID TTY          STAT TIME COMMAND
   927 pts/1    S      0:00 bash
   6690 pts/1    S      0:00 \_ bash
  26909 pts/1    R      0:00 \_ ps x --forest
 19930 pts/4    S      0:01 bash
 25740 pts/4    S      0:04 \_ vi processes.txt
```

“ u ” 和 “ l ” ps 选项

发布时间 :2007-01-26 22:27:41

“ u ” 或 “ l ” 选项也可以被添加到 “ a ” 和 “ x ” 的任何组合中以包含关于每个进程的更多信息：

```
$ ps au
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
agriffis	403	0.0	0.0	2484	72	tty1	S	2001	0:00	-bash
chouser	404	0.0	0.0	2508	92	tty2	S	2001	0:00	-bash
root	408	0.0	0.0	1308	248	tty6	S	2001	0:00	/sbin/agetty 3
agriffis	434	0.0	0.0	1008	4	tty1	S	2001	0:00	/bin/sh /usr/X
chouser	927	0.0	0.0	2540	96	pts/1	S	2001	0:00	bash

```
$ ps al
```

F	UID	PID	PPID	PRI	NI	VSZ	RSS	WCHAN	STAT	TTY	TIME	COMMAND
100	1001	403	1	9	0	2484	72	wait4	S	tty1	0:00	-bash
100	1000	404	1	9	0	2508	92	wait4	S	tty2	0:00	-bash
000	0	408	1	9	0	1308	248	read_c	S	tty6	0:00	/sbin/ag
000	1001	434	403	9	0	1008	4	wait4	S	tty1	0:00	/bin/sh
000	1000	927	652	9	0	2540	96	wait4	S	pts/1	0:00	bash

使用 “ top ”

发布时间 :2007-01-26 22:28:12

如果您正在连续多次运行 ps，并尝试观察其中的变化，那么您可能想要用 top。top 显示了持续更新的进程列表，以及一些有用的摘要信息：

```
$ top
10:02pm up 19 days, 6:24, 8 users, load average: 0.04, 0.05, 0.00
75 processes: 74 sleeping, 1 running, 0 zombie, 0 stopped
CPU states: 1.3% user, 2.5% system, 0.0% nice, 96.0% idle
Mem: 256020K av, 226580K used, 29440K free,    0K shrd,   3804K buff
Swap: 136544K av,  80256K used, 56288K free      101760K cached
```

PID	USER	PRI	NI	SIZE	RSS	SHARE	STAT	LIB	%CPU	%MEM	TIME	COMMAND
628	root	16	0	213M	31M	2304	S	0	1.9	12.5	91:43	X
26934	chouser	17	0	1272	1272	1076	R	0	1.1	0.4	0:00	top
652	chouser	11	0	12016	8840	1604	S	0	0.5	3.4	3:52	gnome-termin
641	chouser	9	0	2936	2808	1416	S	0	0.1	1.0	2:13	sawfish

nice

发布时间 :2007-01-26 22:28:52

每个进程都有一个优先级设置，Linux 用它来确定：该进程相对于与系统上其它进程的运行速度。通过使用 nice 命令来启动进程，您能设置进程的优先级：

```
$ nice -n 10 oggenc /tmp/song.wav
```

因为优先级设置称为 nice，所以很容易记住一个更大的值对于其它进程是非常友好的，从而允许它们获取对 CPU 的优先访问权。缺省情况下，用 0 设置来启动进程，所以上面的 10 设置意味着 oggenc 将欣然放弃对 CPU 的访问权，而把它交给其它进程。一般来说，这意味着 oggenc 将允许其它进程以它们正常的速度运行，而不管 oggenc 突然多么迫切地需要访问 CPU。您可以在上面的 ps 和 top 列表的 NI 列下看到这些 nice 值（niceness）的级别。

renice

发布时间 :2007-01-26 22:29:24

只有在您启动进程时，nice 命令才改变它的优先级。如果您想要更改正在运行的进程 nice 值设置，那么使用 renice：

```
$ ps l 641
 F  UID  PID  PPID PRI  NI   VSZ  RSS WCHAN  STAT TTY      TIME COMMAND
000 1000  641    1   9   0 5876 2808 do_sel S    ?        2:14 sawfish
$ renice 10 641
641: old priority 0, new priority 10
$ ps l 641
 F  UID  PID  PPID PRI  NI   VSZ  RSS WCHAN  STAT TTY      TIME COMMAND
000 1000  641    1   9  10 5876 2808 do_sel S    ?        2:14 sawfish
```

文本处理

文本处理 - 再述重定向

发布时间 :2007-01-26 22:30:38

可以使用 “>” 操作符将命令的输出重定向到一个文件，如下所示：

```
$ echo "firstfile" > copyme
```

除了将输出重定向到一个文件之外，我们也可以利用 shell 强大的名为管道的特性。通过使用管道，我们能将一个命令的输出传递给另一个命令的输入。考虑下面示例：

```
$ echo "hi there" | wc
  1   2   9
```

| 字符用来将左侧命令的输出连接到右侧命令的输入。在上面的示例中，echo 命令打印出后面跟有换行符的字符串 hi there。那个输出通常出现在终端上，但是管道将它重定向到 wc 命令，该命令显示它输入中的行、字和字符的数量。

管道示例

发布时间 :2007-01-26 22:31:12

这里是另一个简单示例：

```
$ ls -s | sort -n
```

在本例中，ls -s 通常打印终端上当前目录的列表，并在每个文件前面打印它的大小。但是我们已通过管道将输出传递给 sort -n，它根据文件大小对输出排序。这是在您的主目录中查找大型文件的一个实际有用的方法！

下列的示例更复杂，但是它们演示了通过可以使用管道实现的强大功能。我们将抛弃我们还未讨论的一些命令，但不要让它使您放慢脚步。请集中精力理解管道的工作原理，这样您才能将它们应用到日常 Linux 任务中。

解压缩管道

发布时间 :2007-01-26 22:31:47

通常为了解压缩并解包文件，您可以执行以下操作：

```
$ bzip2 -d linux-2.4.16.tar.bz2  
$ tar xvf linux-2.4.16.tar
```

这个方法的缺点是需要您的磁盘上创建一个中间的未压缩文件。由于 tar 具备从其输入上直接读的能力（而不是指定一个文件），所以我们可以使用管道来产生相同的最终结果：

```
$ bzip2 -dc linux-2.4.16.tar.bz2 | tar xvf -
```

哇！我们压缩的 tarball 已经被解压缩了，而且不需要中间文件。

更长的管道

发布时间 :2007-01-26 22:32:17

这是另一个管道示例：

```
$ cat myfile.txt | sort | uniq | wc -l
```

我们使用 `cat` 将 `myfile.txt` 的内容传递给 `sort` 命令。当 `sort` 命令接收到这个输入时，它对所有的输入行排序，以便它们按字母次序排列，然后它将输出发送到 `uniq`。`uniq` 除去任何重复行，将筛选后的输出发送到 `wc -l`。我们在前面就已经看到 `wc` 命令了，但它没有带命令行选项。当给定 `-l` 选项时，它只打印它输入中的行数，而不包括字和字符。用您喜爱的文本编辑器尝试创建两个测试文件，然后使用这个管道来查看您获得了什么样的结果。

马上开始文本处理

发布时间 :2007-01-26 22:32:51

现在我们着手快速查看标准 Linux 文本处理命令。因为我们正在本教程中讨论许多内容，所以没有足够的篇幅来提供每个命令的示例。因此，鼓励您阅读每个命令的手册页（例如，通过输入 `man echo`），并花一些时间使用每个命令来学习它们及其选项的工作原理。通常，这些命令将任何文本处理的结果打印到终端，而不是修改任何指定文件。

在快速查看了标准 Linux 文本处理命令之后，我们将详细讨论输出和输入重定向。那么，是的，隧道的尽头就是光明。:)

`echo`

`echo` 将它的参数打印到终端。如果您想要嵌入反斜杠转义序列，那么使用 `-e` 选项；例如 `echo -e "foo\nfoo"` 将打印 `foo`，然后打印一个换行，接着再打印 `foo`。使用 `-n` 选项告知 `echo` 省略缺省情况下附加到输出的最后一个换行。

cat、sort 和 uniq

发布时间 :2007-01-26 22:33:23

cat

cat 将指定为参数的文件内容打印到终端。作为管道的第一个命令，这是很方便的，例如，cat foo.txt | blah。

sort

sort 按字母次序打印在命令行上指定的文件内容。当然，sort 也接受用管道传送的输入。输入 man sort 来熟悉控制排序行为的各种选项。

uniq

uniq 获取已排序的文件或数据流（通过管道）并除去重复行。

wc、head 和 tail

发布时间 :2007-01-26 22:33:52

WC

wc 打印出指定文件或输入流（来自管道）中的行、字和字节的数量。输入 man wc 来学习如何精调显示的内容。

head

head 打印出文件或流的前十行。使用 -n 选项来指定应显示的行数。

tail

打印出文件或流的最后十行。使用 -n 选项来指定应显示的行数。

tac、expand 和 unexpand

发布时间 :2007-01-26 22:34:23

tac

tac 与 cat 类似，但它以逆向顺序打印所有行，换句话说，先打印最后一行。

expand

expand 将输入制表符转换为空格。使用 -t 选项来指定制表符停止位。

unexpand

unexpand 将输入空格转换为制表符。使用 -t 选项来指定制表符停止位。

cut、nl 和 pr

发布时间 :2007-01-26 22:34:54

cut

cut 从输入文件或流的每个行上抽取出由字符限定的字段。

nl

nl 将行号添加到输入的每个行上。这对于打印输出很有用。

pr

pr 将文件分解为多个页面的输出；通常用于打印。

tr、sed 和 awk

发布时间 :2007-01-26 22:35:33

tr

tr 是字符转换工具；它用来将输入流中的某些字符映射成输出流中的某些其它字符。

sed

sed 是一个功能强大的面向流的文本编辑器。

awk

awk 是一种方便的面向行的文本处理语言。

od、split 和 fmt

发布时间 :2007-01-26 22:36:09

od

od 将输入流转换为八进制或十六进制的“转储”格式。

split

split 将较大的文件拆分成许多较小、更易处理的块。

fmt

fmt 对段落重新格式化以便在其边缘处进行换行。这个能力被构建到大多数文本编辑器中，但是应知道它仍是一个好工具。

paste、join 和 tee

发布时间 :2007-01-26 22:36:41

paste

paste 获取两个或更多文件作为输入，连接输入文件上的每个后续行，并输出结果行。它对于创建文本的表或列是很有用的。

join

join 与 paste 类似，但它在每个输入行中使用一个字段（缺省情况下是第一个字段）来匹配一在单行上合并的字段。

tee

tee 将它的输入打印到文件和屏幕。当您想创建某些日志记录，但还想在屏幕上看时，这很有用。

快要结束了！重定向

发布时间 :2007-01-26 22:37:19

与在 bash 命令行上使用 > 类似，您也可以使用 < 来将文件重定向到一个命令。对于许多命令，您能简单地在命令行上指定文件名，但是，一些命令只对标准输入起作用。

Bash 和其它 shell 支持“本地文件（herefile）”的概念。这允许您在命令调用后面几行中为命令指定输入，同时用一个标记值来使它终止。下面是一个示例：

```
$ sort <<END
apple
cranberry
banana
END
apple
banana
cranberry
```

在我们的示例中，我们输入字 apple、cranberry 和 banana，后跟“END”来表示输入的结束。然后 sort 程序以字母次序返回我们的字。

使用 >>

发布时间 :2007-01-26 22:37:53

您也许认为 >> 在某些方面与 << 相类似，但事实上不是。它只意味着将输出附加到文件上，而不是象 > 那样进行覆盖。例如：

```
$ echo Hi > myfile
$ echo there. > myfile
$ cat myfile
there.
```

哇！我们丢失了“Hi”部分！我们的本意是这样：

```
$ echo Hi > myfile
$ echo there. >> myfile
$ cat myfile
Hi
there.
```

这样要好得多！

Linux文章汇集

海量 Linux 技术文章

发布时间 :2006-11-24 16:50:29

下面是linux技术文章快速入口。需要联网：

[Linux 技术交流](#)

<http://www.linuxdiyf.com/bbs/forum-3-1.html>

[Linux 应用](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=1>

[Linux 安装及学习指导](#)

<http://www.linuxdiyf.com/bbs/forum-45-1.html>

[Linux 系统安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=11>

[Linux 学习指导](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=12>

[Linux 软件安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=13>

[shell](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=3>

[Linux 壁纸](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=4>

[红旗](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=5>

[Redhat](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=6>

[SuSE](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=7>

Linux 认证

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=9>

Linux下载分享{酷件、书籍、视频分享 }

<http://www.linuxdiyf.com/bbs/forum-6-1.html>

服务器应用

<http://www.linuxdiyf.com/bbs/forum-7-1.html>

数据库应用

<http://www.linuxdiyf.com/bbs/forum-8-1.html>

Linux 编程与内核

<http://www.linuxdiyf.com/bbs/forum-9-1.html>

UniX 技术文章

<http://www.linuxdiyf.com/bbs/forum-32-1.html>

Linux 业界声音、新闻

<http://www.linuxdiyf.com/bbs/forum-11-1.html>

Linux 人才招聘信息

<http://www.linuxdiyf.com/bbs/forum-46-1.html>

网络转载，感谢原创作者！

制作：红联Linux论坛

祝您阅读愉快！

Linux培训系列

第三讲

讲述不同的主题（包括：系统和因特网文档、Linux 权限模式、用户帐户管理以及登录环境调节），我们将使您的基本的 Linux 管理技能方面的知识趋于完善。

内容基础，语言简短简洁

红联Linux论坛是致力于Linux技术讨论的站点，目前网站收录的文章及教程基本能满足不同水平的朋友学习。

红联Linux门户：www.linux110.com

红联Linux论坛：www.linuxdiyf.com/bbs

下载:Linux电子书籍：

<http://www.linux286.com/linux/linuxdzsj.htm>

目录

系统和网络文档

- 系统和网络文档 - Linux 系统文档的类型
- 手册页
- 手册页 (续)
- 手册页章节
- 多个手册页
- 查找正确的手册页
- 所有关于 apropos 的内容
- MANPATH
- GNU 信息
- GNU 信息 (续)
- /usr/share/doc
- Linux 文档计划
- LDP 概述
- 邮件列表
- 更多的关于邮件列表的内容
- 新闻组
- 供应商和第三方 Web 站点

Linux 权限模型

- Linux 权限模型 - 一个用户、一个组
- 理解“ls -l”
- 三个三元组
- 我是谁？
- 我在哪一组？
- 改变用户和组所有权
- 递归的所有权改变
- 介绍 chmod
- 用户 / 组 / 其他粒度
- 重新设置权限
- 数字模式
- 数字权限语法
- umask
- umask (续)
- 介绍 suid 和 sgid
- 介绍 suid 和 sgid (续)
- suid
- suid/sgid 告诫说明
- 改变 suid 和 sgid
- 权限和目录
- 目录和 sgid
- 目录和删除
- 难以理解的第一位

Linux 帐户管理

- Linux 帐户管理 - 介绍 /etc/passwd
- /etc/passwd 技巧和窍门
- /etc/shadow
- /etc/group
- 组提示

[手工地添加用户和组](#)

[编辑 /etc/passwd](#)

[编辑 /etc/shadow](#)

[设置密码](#)

[编辑 /etc/group](#)

[创建主目录](#)

[帐户管理实用程序](#)

[更多的命令](#)

调节用户环境

[调节用户环境 - 介绍 “ fortune ”](#)

[.bash_profile](#)

[登录 shell](#)

[理解 --login](#)

[交互性测试](#)

[/etc/profile 和 /etc/skel](#)

[export](#)

[标记要导出的变量](#)

[导出和设置 -x](#)

[用 “ set ” 设置变量](#)

[取消设置与 FOO= 的比较](#)

[导出以改变命令行为](#)

[使用 “ env ”](#)

汇集海量Linux技术文章

[海量Linux技术文章](#)

系统和网络文档

系统和网络文档 - Linux 系统文档的类型

发布时间 :2007-01-26 23:08:06

从本质上说，Linux 系统中有三种文档资源：手册页、信息页和 /usr/share/doc 中的应用程序随附的文档。在本章中，我们将揭示浏览这其中的每一种资源的方法，而不用“突破常规”地查找更多信息。

手册页

发布时间 :2007-01-26 23:08:37

手册页（manual pages，或“man pages”）是 UNIX 和 Linux 的参考文档的典型形式。理想的情况是，您可以在手册页中查找任何命令、配置文件或库例程的信息。实际上，由于 Linux 是自由软件，一些手册页没有编写，或者显得过时了。虽然如此，当您需要帮助的时候，手册页仍是您的首选。

要访问手册页，只需输入 man，后面跟上您要查询的主题。页面阅读器（pager）将被启动，那么当您完成阅读时，您需要按 q。例如，为了查找关于 ls 命令的信息，您要输入：

```
$ man ls
```

手册页（续）

发布时间 :2007-01-26 23:09:13

了解手册页的布局对于快速地转到您所需要的信息很有帮助。一般来说，您将在手册页中找到下面这些章节：

NAME 命令的名称和单行描述

SYNOPSIS 怎样使用命令

DEscr ptION 命令功能的深入讨论

EXAMPLES 怎样使用命令的建议

SEE ALSO 相关主题（通常是手册页）

手册页章节

发布时间 :2007-01-26 23:09:51

构成手册页的这些文件存储在 `/usr/share/man` 中（或者有些旧一点的系统存储在 `/usr/man` 中）。在该目录内，您将发现手册页被组织成下面这些章节：

- man1 用户程序
- man2 系统调用
- man3 库函数
- man4 特殊文件
- man5 文件格式
- man6 游戏
- man7 其它

多个手册页

发布时间 :2007-01-26 23:10:35

有些主题在多个章节中存在。为了说明这一点，我们来使用 `whatis` 命令，它将显示一个主题所有可用的手册页：

```
$ whatis printf
printf      (1) - format and print data
printf      (3) - formatted output conversion
```

在这种情况下，`man printf` 将第 1 节（“用户程序”）中的页面作为缺省手册页。如果我们正在写 C 程序，我们可能对第 3 节（“库函数”）中的页面更感兴趣。您可以通过在命令行中指定章节来打开某一章节中的手册页，因此要打开 `printf(3)`，我们将输入：

```
$ man 3 printf
```

查找正确的手册页

发布时间 :2007-01-26 23:11:08

有时，对于给定的主题很难找到正确的手册页。在这种情况下，您可以试着使用 `man -k` 来搜索手册页的 NAME 这一节。请注意这是子串搜索，因此运行像 `man -k ls` 这样的命令将给出一大堆输出！下面是使用更具体的查询的一个示例：

```
$ man -k whatis
apropos      (1) - search the whatis database for strings
makewhatis   (8) - Create the whatis database
whatis       (1) - search the whatis database for complete words
```

所有关于 apropos 的内容

发布时间 :2007-01-26 23:11:44

啊，前一屏的这个示例引出了两点更多的内容！第一，apropos 命令正好等价于 man -k。（事实上，我要让您知道一些小窍门。当您运行 man -k 时，它实际在幕后运行 apropos。）第二点是 makewhatis 命令，它扫描您的 Linux 系统上的所有手册页，并且为 whatis 和 apropos 构建数据库。通常这由 root 用户定期运行，从而使数据库保持更新：

```
# makewhatis
```

要获取关于“man”及其参数的更多信息，您应该从它本身的手册页开始：

```
$ man man
```

MANPATH

发布时间 :2007-01-26 23:12:14

缺省情况下，man 程序将在 /usr/share/man、/usr/local/man、/usr/X11R6/man 以及还可能在 /opt/man 中查找手册页。有时，您可能发现您需要给该搜索路径添加一个附加项。如果是这样，只需在文本编辑器中编辑 /etc/man.conf，添加一行类似这样的内容：

MANPATH /opt/man

从这一点向前，将找到 /opt/man/man* 目录中的所有手册页。请记住您将需要重新运行 makewhatis，从而将这些新手册页添加到 whatis 数据库中。

GNU 信息

发布时间 :2007-01-26 23:12:46

手册页的一个缺点是它们不支持超文本，因此您不能容易地从一个地方跳到另一个地方。GNU 的工作者们意识到了这个缺点，因此他们发明了另一种文档格式：“信息”页。许多 GNU 程序带有信息页形式的扩展文档。您可以用“info”命令开始阅读信息页：

```
$ info
```

以这种方式调用 info 将在系统上生成可用页面的索引。您可以用箭头键在上面来回移动，使用 enter 键进入链接（用星号表明），按 q 退出。这些键是基于 Emacs 的，因此如果您对这种编辑器很熟悉，那么您应该能够很容易地浏览。

GNU 信息（续）

发布时间 :2007-01-26 23:13:18

您也可以在命令行中指定信息页：

```
$ info diff
```

为了获取关于使用 info 阅读器更多的信息，请试着阅读它的信息页。您应该能够自己学会使用我已经提到的几个键进行浏览：

```
$ info info
```

/usr/share/doc

发布时间 :2007-01-26 23:13:54

Linux 系统上还有最后一种帮助资源。许多程序还带有其它格式的附加文档：文本、PDF、Postscript、HTML，这里仅举出几种。在 /usr/share/doc 中（或者旧一些的系统上的 /usr/doc）看一看。您将发现一个很长的目录列表，其中每个目录都带有您系统上的某个应用程序。搜索该文档通常可以发现一些在手册页或信息页中找不到的精品，比如教程或附加的技术文档。快速检查将发现这里有大量有用的阅读材料：

```
$ cd /usr/share/doc
$ find . -type f | wc -l
7582
```

哎呀！今晚您的家庭作业就是阅读这些文档的一半（3791）。等着明天测验哦。

Linux 文档计划

发布时间 :2007-01-26 23:14:25

除系统文档之外，因特网上有很多优秀的 Linux 参考资料。“Linux 文档计划”（Linux Documentation Project）是一群志愿者将完整的免费 Linux 文档系列放在一起的行动。该计划的存在是为了将 Linux 文档的不同片段放在容易搜索和使用的地方。

LDP 概述

发布时间 :2007-01-26 23:15:07

LDP 由下面这些方面组成：

指南 — 更长、更深入的书籍，比如 The Linux Programmer's Guide

HOWTO — 特定主题的帮助，比如 DSL HOWTO

FAQ — 带有回答的“常见问题”（Frequently Asked Questions），比如 Brief Linux FAQ

手册页 — 个别命令的帮助（它们与您在 Linux 上使用 man 命令时所得到的手册页相同）

邮件列表

发布时间 :2007-01-26 23:15:39

邮件列表很可能提供了 Linux 开发者相互合作的最重要的因素。项目一般由相隔遥远的贡献者制定，他们甚至可能在地球的两端。对于一个项目，邮件列表为每个开发者提供一种方法，使他们可与所有其他开发者联系，还可以通过电子邮件举行小组讨论。最有名的开发邮件列表之一是“Linux Kernel Mailing List”，在<http://www.tux.org/lkml/> 中有描述。

更多的关于邮件列表的内容

发布时间 :2007-01-26 23:16:21

除了开发之外，邮件列表还可以提供提问以及从博学的开发者，或者甚至从其他用户那里得到回答的方法。例如，单独的分发包经常给新来者提供邮件列表。您可以检查您的分包包的 Web 站点，以获取所提供的关于邮件列表的信息。

如果您花时间阅读过前一屏上链接的 LKML FAQ，您可能已经注意到邮件列表订户通常对于被重复问到的问题不太友好。在写问题之前，搜索一下所给邮件列表的归档文件总是很明智的。这很可能也将节省您的时间！

新闻组

发布时间 :2007-01-26 23:17:10

因特网“新闻组”类似于邮件列表，但是它基于叫做 NNTP（“Network News Transfer Protocol”，网络新闻传输协议）的协议，而不使用电子邮件。为了参加新闻组，您需要使用 NNTP 客户端程序，比如 slrn 或 pan。其主要的优点是您可以只参加您想参加的讨论，而不会总有邮件发到您的信箱中。

最有影响力的新闻组的讨论是 comp.os.linux。您可以在 LDP 站点 <http://www.linuxdoc.org/linux/#ng> 上浏览该列表。

和邮件列表一样，新闻组讨论经常被归档。一个很受欢迎的新闻组归档站点是 Deja News。

供应商和第三方 Web 站点

发布时间 :2007-01-26 23:17:50

各种 Linux 分销商的 Web 站点经常提供更新的文档、安装说明、硬件兼容性 / 不兼容性声明以及其它支持，如知识库搜索工具。例如：

Redhat Linux
Debian Linux
Gentoo Linux
SuSE Linux
Caldera
Turbolinux

Linux 权限模型

Linux 权限模型 - 一个用户、一个组

发布时间 :2007-01-26 23:18:54

在这一章，我们将来看一看 Linux 权限和所有权模型。我们已经看到每个文件属于一个用户和一个组。这正是 Linux 中权限模型的核心。您可以在 `ls -l` 清单中查看用户和组：

```
$ ls -l /bin/bash
-rwxr-xr-x  1 root  wheel   430540 Dec 23 18:27 /bin/bash
```

在这个特殊的示例中，`/bin/bash` 可执行文件属于 `root` 用户，并且在 `wheel` 组中。Linux 权限模型通过允许给每个文件系统对象设置三种独立的权限级别来工作 — 它们为文件的所有者、文件的组以及所有其他用户。

理解“ls -l”

发布时间 :2007-01-26 23:19:28

我们来看一看我们的 ls -l 输出，检查一下这个清单的第一栏：

```
$ ls -l /bin/bash
-rwxr-xr-x  1 root  wheel   430540 Dec 23 18:27 /bin/bash
```

第一个字段 -rwxr-xr-x 包含该特殊文件的权限的符号表示。该字段中的首字符（-）指定该文件的类型，本例中它是一个常规文件。其它可能的首字符还有：

- “d” 目录
- “l” 符号链接
- “c” 字符专门设备文件
- “b” 块专门设备文件
- “p” 先进先出
- “s” 套接字

三个三元组

发布时间 :2007-01-26 23:20:01

```
$ ls -l /bin/bash
-rwxr-xr-x  1 root   wheel   430540 Dec 23 18:27 /bin/bash
```

该字段的其余部分由三个三元组字符组成。第一个三元字符组代表文件所有者的权限，第二个代表文件的组的权限，第三个代表所有其他用户的权限：

"rwx"

"r-x"

"r-x"

上面，r 表示允许读（查看文件中的数据），w 表示允许写（修改文件以及删除），x 表示允许“执行”（运行程序）。将所有这些信息放在一起，我们可以发现每个人都能够读该文件的内容和执行该文件，但是只允许文件所有者（root 用户）可以以任何方式修改该文件。因此，虽然一般用户可以复制该文件，但是只允许 root 用户更新或删除它。

我是谁？

发布时间 :2007-01-26 23:20:35

在我们看怎样改变文件的用户所有权和组所有权之前，我们首先来看一看怎样得知您当前的用户标识和组成员资格。除非最近您使用过 su 命令，否则您当前的用户标识是您用来登录系统的用户标识。但是，如果您经常使用 su，您可能不记得您当前有效的用户标识。要查看用户标识，输入 whoami：

```
# whoami
root
# su drobbins
$ whoami
drobbins
```

我在哪一组？

发布时间 :2007-01-26 23:21:08

要看看您属于哪一组，使用 group 命令：

```
$ groups
drobbins wheel audio
```

在上面的示例中，我是 drobbins、wheel 和 audio 组的成员。如果您想看看其他用户在什么组，指定他们的用户名作为参数：

```
$ groups root daemon
root : root bin daemon sys adm disk wheel floppy dialout tape video
daemon : daemon bin adm
```

改变用户和组所有权

发布时间 :2007-01-26 23:21:42

为了改变文件或其它文件系统对象的所有者或组，分别使用 `chown` 或 `chgrp`。这两个命令都要一个用户名或组名作参数，后面跟上一个或多个文件名。

```
# chown root /etc/passwd  
# chgrp wheel /etc/passwd
```

您还可以用 `chown` 命令的另一种形式同时设置所有者和组：

```
# chown root.wheel /etc/passwd
```

除非您是超级用户，否则您不可以使用 `chown`，然而任何人都可以使用 `chgrp` 来将文件的组所有权改为他们所属的组。

递归的所有权改变

发布时间 :2007-01-26 23:22:11

chown 和 chgrp 都有一个 -R 选项，该选项可以用来告诉它们递归地将所有权和组改变应用到整个目录树中。
例如：

```
# chown -R drobbins /home/drobbins
```

介绍 chmod

发布时间 :2007-01-26 23:22:40

chown 和 chgrp 可以用来改变文件系统对象的所有者和组，而另一个程序 — 叫做 chmod — 用来改变我们可以在 ls -l 清单中看到的 rwx 权限。chmod 带有两个或多个参数：“mode”，描述怎样改变权限，后面跟将会受到影响的文件或文件列表：

```
$ chmod +x scr_ptfile.sh
```

在上面的示例中，我们的“mode”是+x。您可能会猜到，+x 模式告诉 chmod，使该特殊文件对于用户、组以及其它任何人都是可执行的。

如果我们想要除去一个文件的所有执行权限，我们应该这样做：

```
$ chmod -x scr_ptfile.sh
```

用户 / 组 / 其他粒度

发布时间 :2007-01-26 23:23:08

到此，我们的 `chmod` 示例已经影响到了所有三个三元组 — 用户、组和所有其他用户。通常，一次只修改一个或两个三元组很方便。要这样做，只需要在 `+` 或 `-` 符号之前，给您想要修改的特定的三元组指定符号字符。对于“用户”三元组使用 `u`，对于“组”三元组使用 `g`，对于“其他 / 每个人”使用 `o`：

```
$ chmod go-w scr_ptfile.sh
```

我们刚除去了组和所有其他用户的写权限，而保留“所有者”权限不动。

重新设置权限

发布时间 :2007-01-26 23:23:36

除了交替打开和关闭权限位以外，我们还可以一起重新设置它们。通过使用 = 操作符，我们可以告诉 chmod 我们要指定权限和取消别的权限：

```
$ chmod =rx scr ptfile.sh
```

上面，我们只设置了所有的“ read ”和“ execute ”位，没有设置所有的“ write ”位。如果您仅仅想重新设置特定的三元组，您可以像下面这样，在 = 之前指定该三元组的符号名：

```
$ chmod u=rx scr ptfile.sh
```

数字模式

发布时间 :2007-01-26 23:24:09

直到现在为止，我们使用了叫做“符号”的模式来用 `chmod` 指定权限的改变。然而，指定权限还有一种普遍使用的方法 — 使用 4 位八进制数。使用叫做数字权限语法的语法，每一位代表一个权限三元组。例如，在 1777 中，777 设置本章我们所讨论的“owner”、“group”和“other”标志。1 用来设置专门的权限位，我们将在本章的结束部分讲到。这个图表说明了怎样解释第二到四位（777）：

模式 数字

rw	x	7
rw	-	6
r	-x	5
r	--	4
-wx		3
-w	-	2
--x		1
---		0

数字权限语法

发布时间 :2007-01-26 23:24:40

当您需要给一个文件指定所有权限时，数字权限语法特别有用，比如在下面的示例中：

```
$ chmod 0755 scr_ptfile.sh
$ ls -l scr_ptfile.sh
-rwxr-xr-x 1 drobbins drobbins 0 Jan 9 17:44 scr_ptfile.sh
```

在该示例中，我们使用了 0755 模式，它展开为一个完整的权限设置“-rwxr-xr-x”。

umask

发布时间 :2007-01-26 23:25:11

当进程创建了新文件时，它指定新文件应该具有的权限。通常，所请求的模式是 0666（每个人可读和可写），它比我们希望的具有更多的权限。幸运的是，不管什么时候创建了新文件，Linux 将参考叫做“umask”的东西。系统用 umask 值来将初始指定的权限降低为更合理、更安全的权限。您可以通过在命令行中输入 umask 来查看您当前的 umask 设置：

```
$ umask  
0022
```

Linux 系统上，umask 的缺省值一般为 0022，它允许其他人读您的新文件（如果他们可以得到它们），但是不能进行修改。

umask（续）

发布时间 :2007-01-26 23:25:50

为了在缺省的情况下使新文件更安全，您可以改变 umask 设置：

```
$ umask 0077
```

umask 将确保组和其他用户对于新创建的文件绝对没有任何权限。那么，umask 怎样工作呢？与文件的“常规”权限不同，umask 指定应该关闭哪一个权限。我们来参阅一下我们的“模式到数字”映射表，从而使我们可以理解 0077 的 umask 的意思是什么：

模式 数字

rwX	7
rw-	6
r-x	5
r--	4
-wx	3
-w-	2
--x	1
---	0

使用该表，0077 的最后三位扩展为 ---rwxrwx。现在，请记住 umask 告诉系统禁用哪个权限。根据推断，我们可以看到将关闭所有“组”和“其他”权限，而“用户”权限将保留不动。

介绍 suid 和 sgid

发布时间 :2007-01-26 23:26:24

当您最初登录时，将启动一个新的 shell 进程。您已经知道，但是您可能还不知道这个新的 shell 进程（通常是 bash）使用您的用户标识运行。照这样，bash 程序可以访问所有属于您的文件和目录。事实上，作为用户，我们完全依靠其它程序来代表我们执行操作。因为您启动的程序继承了您的用户标识，因此它们不能访问任何不允许您访问的文件系统对象：

介绍 suid 和 sgid（续）

发布时间 :2007-01-26 23:26:56

例如，一般用户不能直接修改 passwd 文件，因为“write”标志已经对除“root 用户”以外的每个用户关闭：

```
$ ls -l /etc/passwd
-rw-r--r--  1 root  wheel    1355 Nov  1 21:16 /etc/passwd
```

但是，一般用户确实需要他们在需要改变其密码的任何时候，能够修改 /etc/passwd（至少间接地）。但是，如果用户不能修改该文件，究竟怎样完成这个工作呢？

suid

发布时间 :2007-01-26 23:27:27

幸好，Linux 权限模型有两个专门的位，叫做“suid”和“sgid”。当设置了一个可执行程序的“suid”这一位时，它将代表可执行文件的所有者运行，而不是代表启动程序的人运行。

现在，回到 /etc/passwd 问题。如果看一看 passwd 可执行文件，我们可以看到它属于 root 用户：

```
$ ls -l /usr/bin/passwd  
-rwsr-xr-x 1 root wheel 17588 Sep 24 00:53 /usr/bin/passwd
```

您还将注意到，这里有一个 s 取替了用户权限三元组中的一个 x。这表明，对于这个特殊程序，设置了 suid 和可执行位。由于这个原因，当 passwd 运行时，它将代表 root 用户执行（具有完全超级用户访问权），而不是代表运行它的用户运行。又因为 passwd 以 root 用户访问权运行，所以能够修改 /etc/passwd 文件，而没有什么问题。

suid/sgid 告诫说明

发布时间 :2007-01-26 23:27:56

我们看到了 suid 怎样工作，sgid 以同样的方式工作。它允许程序继承程序的组所有权，而不是当前用户的程序所有权。

这里有一些关于 suid 和 sgid 的其它的但是很重要的信息。首先，suid 和 sgid 占据与 ls -l 清单中 x 位相同的空间。如果还设置了 x 位，则相应的位表示为 s（小写）。但是，如果没有设置 x 位，它将表示为 S（大写）。

另一个很重要的提示：在许多环境中，suid 和 sgid 很管用，但是不恰当地使用这些位可能使系统的安全遭到破坏。最好尽可能地少用“suid”程序。passwd 命令是为数不多的必须使用“suid”的命令之一。

改变 suid 和 sgid

发布时间 :2007-01-26 23:28:25

设置和除去 suid 与 sgid 位相当简单。这里，我们设置 suid 位：

```
# chmod u+s /usr/bin/myapp
```

此处，我们从一个目录除去 sgid 位。我们将看到 sgid 位怎样影响下面几屏中的目录：

```
# chmod g-s /home/drobbins
```

权限和目录

发布时间 :2007-01-26 23:28:52

到此为止，我们从常规文件的角度来看权限。当从目录的角度看权限时，情况有一点不同。目录使用同样的权限标志，但是它们被解释为表示略微不同的含义。

对于一个目录，如果设置了“ read ”标志，您可以列出目录的内容；“ write ”表示您可以在目录中创建文件，“ execute ”表示您可以进入该目录并访问内部的任何子目录。没有“ execute ”标志，目录内的文件系统对象是不可访问的。没有“ read ”标志，目录内的文件系统对象是不可查看的，但是只要有人知道磁盘上对象的完整路径，就仍然可以访问目录内的对象。

目录和 sgid

发布时间 :2007-01-26 23:29:24

如果启用了目录的“sgid”标志，在目录内创建的任何文件系统对象将继承目录的组。当您需要创建一个属于同一组的一组人使用的目录树时，这种特殊的功能很管用。只需要这样做：

```
# mkdir /home/groupspace  
# chgrp mygroup /home/groupspace  
# chmod g+s /home/groupspace
```

现在，mygroup 组中的所有用户都可以在 /home/groupspace 内创建文件或目录，同样，他们也将自动地分配到 mygroup 的组所有权。根据用户的 umask 设置，新文件系统对象对于 mygroup 组的其他成员来说，可以或不可以是可读、可写或可执行的。

目录和删除

发布时间 :2007-01-26 23:29:51

缺省情况下，Linux 目录以一种不是在所有情况下都很理想的方式表现。一般来说，只要对一个目录有写访问权，任何人都可以重命名或删除该目录中的文件。对于个别用户使用的目录，这种行为是很合理的。

但是，对于很多用户使用的目录来说，尤其是 /tmp 和 /var/tmp，这种行为可能会产生麻烦。因为任何人都可以写这些目录，任何人都可以删除或重命名任何其他人的文件 — 即使是不属于他们的！显然，当任何其他用户在任何时候都可以输入 “rm -rf /tmp/*” 并损坏每个人的文件时，很难把 /tmp 用于任何有意义的文件。

所幸，Linux 有叫做“粘滞位”（sticky bit）的东西。当给 /tmp 设置了粘滞位（用 `chmod +t`），唯一能够删除或重命名 /tmp 中文件的是该目录的所有者（通常是 root 用户）、文件的所有者或 root 用户。事实上，所有 Linux 分发包都缺省地启用了 /tmp 的粘滞位，而您还可以发现粘滞位在其它情况下也很管用。

难以理解的第一位

发布时间 :2007-01-26 23:30:33

总结本章，我们最后来看一看数字模式的难以理解的第一位数。您可以看到，这个第一位数用来设置 sticky、suid 和 sgid 位：

suid sgid sticky 模式数字

on on on 7

on on off 6

on off on 5

on off off 4

off on on 3

off on off 2

off off on 1

off off off 0

这里有一个怎样用 4 位数字模式来设置一个目录的权限的示例，该目录将由一个工作组使用：

```
# chmod 1775 /home/groupfiles
```

作为家庭作业，请想一想 1755 数字模式权限设置的含义。

Linux 帐户管理 - 介绍 /etc/passwd

发布时间 :2007-01-26 23:31:34

在这一章，我们来看一看 Linux 帐户管理机制。我将以介绍 /etc/passwd 文件开始，该文件定义了 Linux 系统上存在的所有用户。您可以通过输入 “ less /etc/passwd ” 来查看您自己的 /etc/passwd 文件。

/etc/passwd 中的每一行定义一个用户帐户。这里有一个来自于我的 /etc/passwd 文件的示例行：

```
drobbins:x:1000:1000:Daniel Robbins:/home/drobbins:/bin/bash
```

您可以看到，这一行中有相当多的信息。实际上，每个 /etc/passwd 行由多个字段组成，每个字段用 : 隔开。

第一个字段定义了用户名（drobbins），第二个字段包含一个 x。在旧式的 Linux 系统上，该字段将包含一个用来认证的加密密码，但是事实上现在所有的 Linux 系统将这个密码信息存储在另一个文件中。

第三个字段（1000）定义了与该特殊用户相关联的数字用户标识，第四个字段（1000）将用户与一个特殊组关联起来；在下面几屏中，我们将看到定义组 1000 的地方。

第五个字段包含该帐户的文本描述 — 在本例中，是用户的名称。第六个字段定义该用户的主目录，第七个字段指定用户缺省的 shell — 当用户登录时，将自动启动的 shell。

/etc/passwd 技巧和窍门

发布时间 :2007-01-26 23:32:10

您可能已经注意到，/etc/passwd 中定义的用户帐户比实际登录您系统的用户帐户多得多。这是因为不同的 Linux 组件使用用户帐户来加强安全性。通常，这些系统帐户有一个小于 100 的用户标识（“uid”），这其中的很多系统帐户将像 /bin/false 这样的程序列为缺省的 shell。因为 /bin/false 程序什么也不做，而是返回一个错误码退出，这有效地阻止这些帐户被用作登录帐户 — 他们只供内部使用。

/etc/shadow

发布时间 :2007-01-26 23:32:42

这样，用户帐户本身在 /etc/passwd 中定义。Linux 系统包含一个 /etc/passwd 的同伴文件，叫做 /etc/shadow。该文件不像 /etc/passwd，只有对于 root 用户来说是可读的，并且包含加密的密码信息。我们来看一看 /etc/shadow 的一个样本行：

```
drobbins:$1$1234567890123456789012345678901:11664:0:-1:-1:-1:-1:0
```

每一行给一个特殊帐户定义密码信息，同样的，每个字段用：隔开。第一个字段定义与这个 shadow 条目相关的特殊用户帐户。第二个字段包含一个加密的密码。其余的字段在下表中描述：

字段 3 自 1/1/1970 起，密码被修改的天数

字段 4 密码将被允许修改之前的天数（0 表示“可在任何时间修改”）

字段 5 系统将强制用户修改为新密码之前的天数（1 表示“永远都不能修改”）

字段 6 密码过期之前，用户将被警告过期的天数（-1 表示“没有警告”）

字段 7 密码过期之后，系统自动禁用帐户的天数（-1 表示“永远不会禁用”）

字段 8 该帐户被禁用的天数（-1 表示“该帐户被启用”）

字段 9 保留供将来使用

/etc/group

发布时间 :2007-01-26 23:33:25

接下来，我们来看一看 /etc/group 文件，它定义了 Linux 系统上所有的组。这里有一个样本行：

```
drobbins:x:1000:
```

/etc/group 字段格式如下。第一个字段定义组名称，第二个字段是不再使用的密码字段（现在只是保留为 x），第三个字段定义了这个特殊组的数字组标识，第四个字段（上面的示例为空）定义是该组成员的所有用户。

您将回想起样本 /etc/passwd 行引用的组标识为 1000。即使 /etc/group 的第四个字段没有列出 drobbins 用户名，这将起到把 drobbins 用户放到 drobbins 组中的作用。

组提示

发布时间 :2007-01-26 23:33:56

关于用户和组相关联的一点提示 — 在一些系统上，您将发现每个新的登录帐户与同名组（通常是标识号一样）相关联。在其它系统上，所有登录帐户将属于单个用户组。在您管理的系统上，您使用的方法取决于您自己。为每个用户创建匹配组的好处是，通过将可信的朋友放在自己的个人组中，使用户能够更容易地控制对自己文件的访问权。

手工地添加用户和组

发布时间 :2007-01-26 23:34:55

现在，我将为您展示怎样创建您自己的用户和组帐户。学习怎样完成这些工作的最好方法是，手工地将用户新添加到系统中。为了开始学习，首先确保您的 EDITOR 环境变量设置为您喜欢的文本编辑器：

```
# echo $EDITOR  
vim
```

如果不是，您可以通过输入这样的命令来设置 EDITOR：

```
# export EDITOR=/usr/bin/emacs
```

现在，输入：

```
# vipw
```

现在您应该发现自己在所喜欢的文本编辑器中，编辑器内 /etc/passwd 文件被装入，显示在屏幕上。当修改系统 passwd 和 group 文件时，使用 vipw 和 vigr 命令非常重要。它们采用了额外的预防措施来确保您关键的 passwd 和 group 文件被恰当地锁定，使它们不会破坏。

编辑 /etc/passwd

发布时间 :2007-01-26 23:35:24

既然您已经打开了 /etc/passwd 文件，则接着添加下面的代码行：

```
testuser:x:3000:3000:LPI tutorial test user:/home/testuser:/bin/false
```

我们刚刚添加了一个 UID 为 3000 的“testuser”用户。我们将他添加到 GID 为 3000 的组中，该组还未创建。另一种做法是，如果愿意，我们还可以给这个用户分配 users 组的 GID。这个新用户有一条注释为：LPI tutorial test user；该用户的主目录设置为 /home/testuser，出于安全的目的，该用户的 shell 设置为 /bin/false。如果我们正创建一个非测试帐户，那么我们可以将 shell 设置为 /bin/bash。OK，接着保存您所做的更改，然后退出。

编辑 /etc/shadow

发布时间 :2007-01-26 23:35:55

现在，我们需要在 /etc/shadow 中给这个特殊用户添加一个条目。要这样做，输入 vipw -s。您将会看到您喜欢的编辑器，它现在包含 /etc/shadow 文件。现在，接着复制一个现有用户帐户行（也就是有一个密码，并且长于标准系统帐户条目）：

```
drobbins:$1$1234567890123456789012345678901:11664:0:-1:-1:-1:-1:0
```

现在，将所复制的代码行中的用户名改为您的新用户的名称，确保所有的字段（特别是密码的期限设置）设置为您喜欢的模式：

```
testuser:$1$1234567890123456789012345678901:11664:0:-1:-1:-1:-1:0
```

保存，然后退出。

设置密码

发布时间 :2007-01-26 23:36:24

您将回到提示符。现在，是给您的新用户设置密码的时候了：

```
# passwd testuser
Enter new UNIX password: (enter a password for testuser)
Retype new UNIX password: (enter testuser's new password again)
```

编辑 /etc/group

发布时间 :2007-01-26 23:36:56

既然 /etc/passwd 和 /etc/shadow 设置好了，现在该恰当配置 /etc/group 了。要这么做，输入：

```
# vigr
```

您的 /etc/group 文件将出现在您面前，准备好进行编辑。现在，如果您选择给您的特殊的测试用户分配 users 缺省组，那么您不需要将任何组添加到 /etc/groups 中。但是，如果您选择给该用户创建新的组，接着添加下面的行：

```
testuser:x:3000:
```

保存，然后退出。

创建主目录

发布时间 :2007-01-26 23:37:30

我们基本上已经完工。输入下面的命令来创建 testuser 的主目录。

```
# cd /home
# mkdir testuser
# chown testuser.testuser testuser
# chmod o-rwx testuser
```

我们用户的主目录现在已经到位，并且帐户已准备好可用。好的，基本就绪。如果您想使用该帐户，您将需要使用 vipw 来将 testuser 的缺省 shell 改为 /bin/bash，使用户可以登录。

帐户管理实用程序

发布时间 :2007-01-26 23:38:04

既然您知道怎样手工添加新帐户和组，我将要评论一下 Linux 下可用的各种省时的帐户管理实用程序。由于版面的限制，我将不深究描述这些命令的众多细节。请记住，通过查看命令的手册页，您总能够获得关于命令的更多信息。如果您计划参加 LPIC 101 考试，我建议您花些时间来让您自己熟悉一下下面每一条命令。

newgrp

缺省情况下，用户创建的任何文件都被分配到 /etc/passwd 中所指定的用户的组。如果用户属于其他组，他或她可以输入 newgrp thisgroup 来将当前缺省组的成员资格设置为组 thisgroup。然后，所创建的任何新文件将继承该组的成员资格。

chage

chage 命令用来查看和改变存储在 /etc/shadow 中的密码期限设置。

gpasswd

一个一般目的的组管理工具

groupadd/groupdel/groupmod

用来在 /etc/group 中添加 / 删除 / 修改组

更多的命令

发布时间 :2007-01-26 23:38:32

useradd/userdel/usermod

用来在 /etc/passwd 中添加 / 删除 / 修改用户。这些命令还完成其它各种便利功能。要获取更多的信息，请参阅手册页。

pwconv/grpconv

用来将 passwd 和 group 文件转换为“新式”的 shadow 密码。事实上，所有 Linux 系统已经使用 shadow 密码，因此您应该不会需要使用这些命令。

pwunconv/grpunconv

用来将 passwd、shadow 和 group 文件转换成“旧式”的非 shadow 密码。您应该不会需要使用这些命令。

调节用户环境

调节用户环境 - 介绍 “ fortune ”

发布时间 :2007-01-26 23:39:13

您的 shell 有很多可设置为适合您的个人爱好的有用的选项。但是，到目前为止，除了每次重新输入以外，我们还没有讨论到每次您登录时，自动设置这些设置的任何方法。在本章中，我们将看一看通过修改启动文件来调节您的登录环境。

首先，当您初次登录时，我们来添加一条友好的消息。要看示例消息，运行 fortune：

```
$ fortune
No amount of careful planning will ever replace dumb luck.
```

.bash_profile

发布时间 :2007-01-27 11:05:17

现在，我们来设置 fortune，使每次您登录时，它能运行。使用您喜欢的文本编辑器来编辑您的主目录中名为 .bash_profile 的文件。如果该文件还不存在，则接着创建它。在顶部插入一行：

```
fortune
```

试着注销，然后再回来。除非您正在运行一个像 xdm、gdm 或 kdm 这样的显示管理器，否则当您登录时，您应该会很愉快地看到：

```
mycroft.flatmonk.org login: chouser
Password:
Freedom from incrustations of grime is contiguous to rectitude.
$
```

登录 shell

发布时间 :2007-01-27 11:05:54

当 bash 启动，它将遍历您主目录中的 .bash_profile 文件，就象在 bash 提示符下输入命令一样运行每一行。这叫做 “source” 文件。

根据 bash 启动的方式，bash 的动作有些不同。如果它作为 “登录” shell 被启动，它将像上面那样动作 — 首先 source 系统范围的 /etc/profile，然后是您个人的 ~/.bash_profile。

告诉 bash 作为登录 shell 运行有两种方式。一种方式是当您初次登录时使用 — bash 由一个名为 -bash 的进程启动。您可以在您的进程清单中看到这些：

```
: $ ps u
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
chouser   404  0.0  0.0  2508  156 tty2    S      2001   0:00 -bash
```

您很可能看到长得多的清单，但是在您的 shell 名之前，至少有一个带有短划线的 COMMAND，如上面示例中的 -bash。shell 用这个短划线来确定它是否正作为 “登录” shell 运行。

理解 --login

发布时间 :2007-01-27 11:06:29

告诉 bash 作为“登录”shell 运行的第二种方法是用 --login 命令行选项。终端仿真器（如 xterm）有时使用这个选项来使它们的 bash 会话表现得像初始登录会话。

当您登录以后，将运行 shell 的更多副本。除非它们以 --login 启动或进程名中有短划线，否则这些会话将不是“登录”shell。但是，如果它们给出提示符，那么它们叫“交互式”shell。如果 bash 作为“交互式”shell 启动，而不是作为“登录”shell 启动，它将忽略 /etc/profile 和 ~/.bash_profile，而将 source ~/.bashrc。

```
interactive login profile rc
yes yes source ignore
yes no ignore source
no yes source ignore
no no ignore ignore
```

交互性测试

发布时间 :2007-01-27 11:07:03

有时 bash source 您的 ~/.bashrc，即使它不是真正的交互式 shell，比如当使用像 rsh 和 scp 这样的命令时。将这些牢记在心很重要，因为像前面我们用 fortune 命令所做的一样，打印出文本可能真的会打乱这些非交互式的 bash 会话。在从启动文件打印出文本之前，使用 PS1 变量来检测当前的 shell 是否是交互式 shell 是一个好办法：

```
if [ -n "$PS1" ]; then
fortune
fi
```

/etc/profile 和 /etc/skel

发布时间 :2007-01-27 11:07:33

作为系统管理员，您掌管着 /etc/profile。因为当初次登录时，每个人都 source 它，所以使它保持工作状态很重要。它也是提供给新用户的强大工具，该工具使新用户一登录进他们的新帐户，一切就正确运行。

但是，有一些您可能希望新用户作为缺省值的设置，而且允许容易地修改它们。这是 /etc/skel 目录的用途所在。当您用 useradd 命令来创建一个新用户帐户时，它将所有的文件从 /etc/skel 复制到用户的新的主目录中。这意味着您可以将有帮助的 .bash_profile 和 .bashrc 文件放在 /etc/skel 中，使新用户有一个好的开始。

export

发布时间 :2007-01-27 11:08:10

可以给 bash 中的变量作上标记，使它们在任何 bash 启动的新的 shell 中设置相同；这被称为做上标记以便导出。在您的 shell 会话中，您可以列出 bash 所有的当前标记为要导出的变量：

```
$ export
declare -x EDITOR="vim"
declare -x HOME="/home/chouser"
declare -x MAIL="/var/spool/mail/chouser"
declare -x PAGER="/usr/bin/less"
declare -x PATH="/bin:/usr/bin:/usr/local/bin:/home/chouser/bin"
declare -x PWD="/home/chouser"
declare -x TERM="xterm"
declare -x USER="chouser"
```

标记要导出的变量

发布时间 :2007-01-27 11:08:46

如果变量没有标记为导出，任何它启动的新的 shell 将不会设置该变量。但是，您可以通过将变量传给内置的 export 来将其标记为导出：

```
$ FOO=foo
$ BAR=bar
$ export BAR
$ echo $FOO $BAR
foo bar
$ bash
$ echo $FOO $BAR
bar
```

在本示例中，一起设置了变量 FOO 和 BAR，但是只有 BAR 被标记为导出。当启动了新的 bash，它丢掉 FOO 的值。如果您退出这个新的 bash，您可以看到最初的 bash 仍然有 FOO 和 BAR 的值：

```
$ exit
$ echo $FOO $BAR
foo bar
```

导出和设置 -x

发布时间 :2007-01-27 11:09:19

由于这种行为，可以在 ~/.bash_profile 或 /etc/profile 中设置变量和标记为导出，然后再也不需要重新设置。但是，有一些不能导出的选项，因此为了设置得一致，必须将它们放在您的 ~/.bashrc 和 环境配置文件中。这些选项用内置的 set 来调整：

```
$ set -x
```

-x 选项使 bash 打印出它要运行的每个命令：

```
$ echo $FOO
+ echo foo
foo
```

这对于理解没有预料到的引用行为或类似的莫名其妙的现象非常有用。要关闭 -x 选项，设置 set +x。请参阅 bash 手册页来获取内置的 set 的所有选项的信息。

用 “ set ” 设置变量

发布时间 :2007-01-27 11:09:50

内置的 set 还可以用来设置变量，但这样使用时，它是可选的。bash 命令 set FOO=foo 表示的意思正好和 FOO=foo 相同。取消设置变量用内置的 unset 来完成：

```
$ FOO=bar
$ echo $FOO
bar
$ unset FOO
$ echo $FOO
```

取消设置与 FOO= 的比较

发布时间 :2007-01-27 11:10:26

这与将变量设置为什么也不设不相同，虽然有时很难区别。一种区别的方法是使用不带参数的内置的 set 来列出所有当前变量：

```
$ FOO=bar
$ set | grep ^FOO
FOO=bar
$ FOO=
$ set | grep ^FOO
FOO=
$ unset FOO
$ set | grep ^FOO
```

除了 set 列出所有变量而不仅仅是那些标记为要导出的变量外，像这样不带参数使用 set 与使用内置的 export 类似。

导出以改变命令行为

发布时间 :2007-01-27 11:10:57

通常，可以通过设置环境变量来改变命令的行为。正和新的 bash 会话一样，从您的 bash 提示符启动的其它程序将只能看见标记为导出的变量。例如，命令 man 检查变量 PAGER，看一看用什么程序来每次一页地遍历文本。

```
$ PAGER=less
$ export PAGER
$ man man
```

将 PAGER 设置为 less，您将每次看到一页，按空格键移到下一页。如果您将 PAGER 改为 cat，将立刻显示所有的文本，没有停顿。

```
$ PAGER=cat
$ man man
```

使用 “ env ”

发布时间 :2007-01-27 11:11:30

不幸的是，如果您忘记将 PAGER 设置回 less，man（像其它命令一样）将继续没有停顿地显示所有的文本。如果您仅一次想将 PAGER 设为 cat，您可以使用 env 命令：

```
$ PAGER=less
$ env PAGER=cat man man
$ echo $PAGER
less
```

这一次，PAGER 值为 cat，被导出到 man，但在 bash 会话中，PAGER 变量本身仍然未改变。

汇集海量Linux技术文章

海量 Linux 技术文章

发布时间 :2006-11-24 16:50:29

下面是linux技术文章快速入口。需要联网：

[Linux 技术交流](#)

<http://www.linuxdiyf.com/bbs/forum-3-1.html>

[Linux 应用](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=1>

[Linux 安装及学习指导](#)

<http://www.linuxdiyf.com/bbs/forum-45-1.html>

[Linux 系统安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=11>

[Linux 学习指导](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=12>

[Linux 软件安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=13>

[shell](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=3>

[Linux 壁纸](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=4>

[红旗](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=5>

[Redhat](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=6>

[SuSE](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=7>

Linux 认证

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=9>

Linux下载分享{酷件、书籍、视频分享 }

<http://www.linuxdiyf.com/bbs/forum-6-1.html>

服务器应用

<http://www.linuxdiyf.com/bbs/forum-7-1.html>

数据库应用

<http://www.linuxdiyf.com/bbs/forum-8-1.html>

Linux 编程与内核

<http://www.linuxdiyf.com/bbs/forum-9-1.html>

UniX 技术文章

<http://www.linuxdiyf.com/bbs/forum-32-1.html>

Linux 业界声音、新闻

<http://www.linuxdiyf.com/bbs/forum-11-1.html>

Linux 人才招聘信息

<http://www.linuxdiyf.com/bbs/forum-46-1.html>

网络转载，感谢原创作者！

制作：红联Linux论坛

祝您阅读愉快！

Linux培训系列

第四讲

将通过讨论多个主题（包括 Linux 文件系统、Linux 引导过程、运行级别、文件系统限额和系统日志），来巩固您在重要的 Linux 管理技能方面的知识。

内容基础，语言简短简洁

红联Linux论坛是致力于Linux技术讨论的站点，目前网站收录的文章及教程基本能满足不同水平的朋友学习。

红联Linux门户：www.linux110.com

红联Linux论坛：www.linuxdiyf.com/bbs

下载:Linux电子书籍：

<http://www.linux286.com/linux/linuxdzsj.htm>

目录

Linux 文件系统

- Linux 文件系统 - 块设备
- 完整的磁盘和分区
- 介绍 fdisk
- 使用 fdisk
- 使用 fdisk (续)
- fdisk 以及更多内容
- 创建文件系统
- 挂装文件系统
- 创建和使用交换分区
- 查看挂装的文件系统
- 挂装选项
- 介绍 fstab
- 样本 fstab
- 样本 fstab (续)
- 卸装文件系统
- 介绍 fsck
- fsck 的问题
- ext2 文件系统
- ext3 文件系统
- ReiserFS 文件系统
- XFS 文件系统
- JFS 文件系统
- VFAT

引导系统

- 引导系统
- MBR
- 内核引导过程
- /sbin/init
- 深入研究：LILO
- 使用 LILO
- 一个重要的 LILO 问题
- 深入研究：GRUB
- 使用 GRUB
- dmesg
- /var/log/messages
- 单用户方式
- 使用单用户方式
- 更改运行级别
- 恰当地关机
- 立即关机
- 缺省运行级别

运行级别

- 运行级别 - 单用户方式
- 单用户方式 (续)
- 运行级别
- telinit
- 运行级别规则

[“ now ” 和停机](#)

[配置 init](#)

文件系统限额

[文件系统限额](#)

[内核支持](#)

[文件系统支持](#)

[配置限额](#)

[配置限额（续）](#)

[quota 命令](#)

[查看限额](#)

[edquota](#)

[edquota（续）](#)

[理解 edquota](#)

[进行更改](#)

[复制限额](#)

[组限制](#)

[repquota 命令](#)

[repquota 选项](#)

[监控限额](#)

[修改宽限时间](#)

[在引导时检查限额](#)

系统日志

[系统日志 - 介绍 syslogd](#)

[读取日志](#)

[跟踪日志文件](#)

[查找日志](#)

[日志概述](#)

[syslog.conf](#)

[syslog.conf](#)

[重新装入和附加信息](#)

[安全性注释](#)

[logrotate](#)

[高级主题 — klogd](#)

[高级主题 — 替代日志记录器](#)

Linux资源汇集

[海量Linux技术文章](#)

Linux 文件系统

Linux 文件系统 - 块设备

发布时间 :2007-01-27 11:22:39

在本节中，我们将仔细研究 Linux 文件系统，以便您熟悉管理员需要知道的所有具体的详细资料。我将从介绍“块设备”开始。最为人们熟知的块设备可能是表示 Linux 系统中第一个 IDE 驱动器的块设备：

`/dev/hda`

如果您的系统使用 SCSI 驱动器，那么您的第一个硬盘驱动器将是：

`/dev/sda`

以上的块设备表示磁盘的抽象接口。用户程序可以使用这些块设备与磁盘进行交互而无需担心驱动器是 IDE、SCSI 或其它东西。程序只需将磁盘上的存储信息视为一串大小为 512 字节、连续的、可随机访问的块来进行寻址。

完整的磁盘和分区

发布时间 :2007-01-27 11:23:08

在 Linux 下，我们可以使用特殊的“mkfs”命令并将特殊块设备指定为命令行参数来创建文件系统。

然而，尽管理论上可以使用一个象 /dev/hda 或 /dev/sda 那样的“完整的磁盘”的块设备（表示整个磁盘）来包含单个文件系统，但实际上从未采取过这种方法。而是将整个磁盘块设备分成更小、更可管理的、名为“分区”的块设备。可以使用名为 fdisk 的工具创建分区，该工具用来创建和编辑存储在磁盘上的分区表。分区表对如何分割整个磁盘进行了精确定义。

介绍 fdisk

发布时间 :2007-01-27 11:23:45

为了研究磁盘的分区表，我们可以运行 fdisk，其中将表示整个磁盘的块设备指定为参数：

```
# fdisk /dev/hda
```

```
# fdisk /dev/sda
```

请注意：如果磁盘的任何分区包含正在使用的文件系统或包含重要的数据，则不应该保存或作出任何对磁盘分区的更改。那样做会导致磁盘上的数据丢失。

使用 fdisk

发布时间 :2007-01-27 11:24:28

在 fdisk 中，您将看到与下面相似的提示符：

Command (m for help):

输入 p 以显示磁盘的当前分区配置：

Command (m for help): p

Disk /dev/hda: 240 heads, 63 sectors, 2184 cylinders
Units = cylinders of 15120 * 512 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1		1	14	105808+	83	Linux
/dev/hda2		15	49	264600	82	Linux swap
/dev/hda3		50	70	158760	83	Linux
/dev/hda4		71	2184	15981840	5	Extended
/dev/hda5		71	209	1050808+	83	Linux
/dev/hda6		210	348	1050808+	83	Linux
/dev/hda7		349	626	2101648+	83	Linux
/dev/hda8		627	904	2101648+	83	Linux
/dev/hda9		905	2184	9676768+	83	Linux

Command (m for help):

我们将这个特殊磁盘配置成包含七个 Linux 文件系统（以“Linux”列出）和一个交换分区（以“Linux swap”列出）。请注意左边相应的分区块设备名称，从 /dev/hda1 开始直到 /dev/hda9。在 PC 的早期，分区软件最多只允许四个分区（称为“主”分区）。由于这限制过多，因此产生了一个名为扩展分区的变通方法。扩展分区和主分区非常相似，并且占据四个主分区限制中的一个。然而，扩展分区可以拥有任意数目的所谓逻辑分区，这有效地解决了四个分区的限制。

使用 fdisk (续)

发布时间 :2007-01-27 11:25:08

Command (m for help): p

Disk /dev/hda: 240 heads, 63 sectors, 2184 cylinders
Units = cylinders of 15120 * 512 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1		1	14	105808+	83	Linux
/dev/hda2		15	49	264600	82	Linux swap
/dev/hda3		50	70	158760	83	Linux
/dev/hda4		71	2184	15981840	5	Extended
/dev/hda5		71	209	1050808+	83	Linux
/dev/hda6		210	348	1050808+	83	Linux
/dev/hda7		349	626	2101648+	83	Linux
/dev/hda8		627	904	2101648+	83	Linux
/dev/hda9		905	2184	9676768+	83	Linux

Command (m for help):

在我们的示例中，hda1 到 hda3 是主分区。hda4 是包含逻辑分区 hda5 到 hda9 的扩展分区。因此，在本示例中，您实际上永远不可能使用 /dev/hda4 直接存储任何文件系统 — 它只能作为分区 hda5 到 hda9 的容器。另外，请注意每个分区都有一个“标识 (Id)”，这也称做“分区类型”。无论您何时创建新的分区，您都必须确保正确设置了分区类型。‘83’是包含 Linux 文件系统的分区的正确类型，‘82’是用于 Linux 交换分区的正确类型。您可以使用 fdisk 中的 t 选项设置分区类型。Linux 内核在引导期间用分区类型设置来自动检测磁盘上的文件系统和交换设备。

fdisk 以及更多内容

发布时间 :2007-01-27 11:25:41

关于 fdisk 的内容还有很多，由于篇幅所限无法在这里全都讨论，这其中还包括新分区的创建（用 n 命令）以及将更改写入磁盘（用 w 命令）。请记住您可以输入 m 来获得帮助。如果您初次接触 fdisk，我建议您通过在没有数据丢失危险的空闲磁盘上创建一些分区来熟悉该程序的用法。一旦创建分区并将它们写入磁盘，您的新分区块设备就准备好可以使用了。我们马上将用这些新的块设备来存储新的 Linux 文件系统。

创建文件系统

发布时间 :2007-01-27 11:26:22

在新的块设备可用来存储文件以前，我们需要在上面创建新的文件系统。我们通过使用 mkfs 命令做到这一点—— 我们根据要创建的文件系统的类型来使用特殊的 mkfs。在本示例中，我们使用 mke2fs 在 /dev/hda6（一个空的、未使用的分区块设备）上创建 ext2 文件系统：

```
# mke2fs /dev/hdc6
mke2fs 1.25 (20-Sep-2001)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
1537088 inodes, 3072423 blocks
153621 blocks (5.00%) reserved for the super user
First data block=0
94 block groups
32768 blocks per group, 32768 fragments per group
16352 inodes per group
Superblock backups stored on blocks:
32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208
```

```
Writing inode tables: done
Writing superblocks and filesystem accounting information: done
```

This filesystem will be automatically checked every 22 mounts or 180 days, whichever comes first. Use tune2fs -c or -i to override.

通过上述命令，mke2fs 在 /dev/hda6 上创建了空白的 ext2 文件系统。

挂装文件系统

发布时间 :2007-01-27 11:26:54

创建文件系统之后，我们就可以使用 mount 命令挂装它：

```
# mount /dev/hdc6 /mnt
```

要挂装一个文件系统，需要将分区块设备指定为第一个参数，将“挂装点”指定为第二个参数。新的文件系统将在挂装点处“嫁接”。这样做的另一个效果是“隐藏”位于父文件系统上 /mnt 目录中的任何文件。以后卸载该文件系统时，这些文件将重新出现。执行挂装命令后，所有添加到 /mnt 的文件将存储在新的 ext2 文件系统中。

创建和使用交换分区

发布时间 :2007-01-27 11:27:29

如果我们刚刚创建了一个分区作为交换设备使用，我们要使用 `mkswap` 命令初始化该分区，并将这个分区块设备作为参数指定：

```
# mkswap /dev/hdc6
```

不同于常规文件系统，我们不挂装交换分区。而是用 `swapon` 命令启用交换分区：

```
# swapon /dev/hdc6
```

通常，Linux 系统的启动脚本将负责自动启用交换分区。因此，`swapon` 命令通常只在您要立即添加刚创建的交换分区时才需要。要查看当前启用的交换设备，请输入 `cat /proc/swaps`。

查看挂装的文件系统

发布时间 :2007-01-27 11:28:03

要查看挂装了什么文件系统，只需输入 mount 本身即可：

```
# mount
/dev/ide/host0/bus1/target0/lun0/part7 on / type xfs (rw,noatime,nodiratime)
proc on /proc type proc (rw)
none on /dev type devfs (rw)
tmpfs on /dev/shm type tmpfs (rw)
/dev/hdc6 on /mnt type ext2 (rw)
```

您也可以通过输入 `cat /proc/mounts` 查看类似信息。因为我的 Linux 系统使用 devfs，所以 mount 输出的第一行列出了较长的“根（root）”分区块设备路径。“根”文件系统将在引导期间由内核自动挂装。对 /dev 使用新的 devfs 设备管理文件系统的系统所用的正式的分区和磁盘块设备名称比 Linux 过去一直使用的更长。例如，/dev/ide/host0/bus1/target0/lun0/part7 是 /dev/hdc7 的正式名称，而 /dev/hdc7 本身只是指向正式块设备的符号链接。通过查看 /dev/.devfsd 文件是否存在，您可以确定文件系统是否在使用 devfs；如果该文件存在，则 devfs 是活动的。

挂装选项

发布时间 :2007-01-27 11:28:45

可以通过指定挂装选项来定制待挂装文件系统的各种属性。例如，您可以使用“ro”选项将文件系统挂装为“只读”：

```
# mount /dev/hdc6 /mnt -o ro
```

/dev/hdc6 挂装为只读后，就不能修改 /mnt 中的任何文件 — 只能读取。如果您的文件系统已经挂装为“读 / 写”方式，而您想把它切换为只读方式，您可以使用“remount”选项以避免再次卸装和重新挂装该文件系统：

```
# mount /mnt -o remount,ro
```

请注意：不需要指定分区块设备，因为已经挂装了文件系统而且 mount 知道 /mnt 与 /dev/hdc6 相关联。要使文件系统再次为可写，我们可以将它重新挂装为读 / 写方式：

```
# mount /mnt -o remount,rw
```

请注意：如果有任何进程打开了 /mnt 中的任何文件或目录，那么这些重新挂装命令将不会成功地完成。要熟悉 Linux 下所有可用的挂装选项，请输入 man mount。

介绍 fstab

发布时间 :2007-01-27 11:29:17

到目前为止，我们已经了解了如何手工挂装文件系统。一般而言，如果您要经常挂装一个文件系统，那么手工挂装往往有些麻烦。而且对于基本的文件系统（如一个单独的 /var 文件系统），手工挂装是不可能的。这些文件系统需要在引导期间自动挂装，只要向 /etc/fstab 文件添加合适的项，我们就可以告诉系统做这些事情。即使您不希望在引导期间自动挂装该文件系统，一个 /etc/fstab 项也可以使挂装变得比较容易，正如我们马上会看到的那样。

样本 fstab

发布时间 :2007-01-27 11:29:56

让我们考察一个样本 /etc/fstab 文件：

```
# <fs>      <mountpoint>  <type>      <opts>      <dump/pass>

/dev/hda1    /boot ext2 noauto,noatime 1 2
/dev/hdc7    /  xfs noatime,osyncisdsync,nodiratime 0 1
/dev/hdc5    none      swap sw 0 0
/dev/cdrom /mnt/cdrom iso9660 noauto,ro,user 0 0
# /proc should always be enabled
proc /proc      proc defaults 0 0
```

如上所示，/etc/fstab 中的每个未注释行指定一个分区块设备、一个挂装点、一个文件系统类型、挂装该文件系统时要用到的文件系统选项以及两个数字字段。第一个数字字段用来向 dump 备份命令指明应该备份的文件系统。当然，如果您不打算在系统上使用 dump 命令，那么忽略这个字段不会有问题。后一个字段由 fsck 文件系统完整性检查程序使用，并告诉该程序在引导时检查文件系统所应采用的顺序。我们将在下几屏再次接触 fsck。

观察 /dev/hda1 行；您会发现 /dev/hda1 是一个应该挂装在 /boot 挂装点的 ext2 文件系统。现在，观察 <opts> 列中 /dev/hda1 的挂装选项。noauto 选项告诉系统在引导期间不要自动挂装 /dev/hda1；如果没有这个选项，系统引导时会自动将 /dev/hda1 挂装到 /boot。

还要注意 noatime 选项，它关闭了磁盘上 atime（最近访问时间）信息的记录。该信息通常是不需要的，关闭 atime 更新对文件系统性能有积极作用。您还可以使用 nodiratime 挂装选项来关闭目录 atime 更新。

样本 fstab (续)

发布时间 :2007-01-27 11:30:41

```
# <fs>      <mountpoint>  <type>  <opts>      <dump/pass>

/dev/hda1    /boot          ext2 noauto,noatime 1 1
/dev/hdc7    /              xfs  noatime,osyncisdsync,nodiratime 0 0
/dev/hdc5    none           swap sw  0 0
/dev/cdrom   /mnt/cdrom     iso9660 noauto,ro,user  0 0
# /proc should always be enabled
proc         /proc          proc defaults  0 0
```

现在，观察 /proc 行并注意 defaults 选项。当希望只用标准挂装选项挂装文件系统时，请使用 defaults。因为 /etc/fstab 有多个字段，所以我们不能简单地让这个选项字段留作空白。

还要注意 /dev/hdc5 的 /etc/fstab 行。该行将 /dev/hdc5 定义为交换设备。因为交换设备不象文件系统那样被挂装，所以在挂装点字段指定 none。由于这个 /etc/fstab 项，系统启动时将自动启用 /dev/hdc5 交换设备。

如果有如上所示用于 /dev/cdrom 的 /etc/fstab 项，挂装 CD-ROM 驱动器会变得更加简单。我们不再输入：

```
# mount -t iso9660 /dev/cdrom /mnt/cdrom -o ro
```

我们现在可以输入：

```
# mount /dev/cdrom
```

事实上，使用 /etc/fstab 允许我们利用 user 选项。user 挂装选项告诉系统允许任何用户挂装这个特殊的文件系统。这对于可移动媒体设备（象 CD-ROM 驱动器）很方便。如果没有这个 fstab 挂装选项，则只有 root 用户能够使用 CD-ROM 驱动器。

卸装文件系统

发布时间 :2007-01-27 11:31:17

通常情况下，所有挂装的文件系统在系统重新引导或关机时都会自动卸装。当卸装文件系统时，内存中所有位于高速缓存中的文件系统数据都被刷新到磁盘中。

然而，也可以手工卸装文件系统。在可以卸装文件系统以前，您首先必须确保正在运行的进程在当前讨论的文件系统上没有打开的文件。然后，使用 `umount` 命令，将设备名称或者挂装点指定为参数：

```
# umount /mnt
```

或

```
# umount /dev/hdc6
```

卸装之后，`/mnt` 中被以前挂装的文件系统所“覆盖”的所有文件将重新出现。

介绍 fsck

发布时间 :2007-01-27 11:31:49

如果您的系统因某种原因崩溃或锁定，那么系统将无法彻底卸装您的文件系统。如果发生了这种情况，您的文件系统就处于不一致（不可预测）状态。当系统重新引导时，fsck 程序将检测出文件系统没有完全卸装，并希望对 /etc/fstab 中列出的文件系统的一致性检查。

有一点很重要 — 对于由 fsck 检查的文件系统，必须在 /etc/fstab 的 “ pass ” 字段（最后一个字段）中有非零数字。通常情况下，应将根文件系统的 passno 设置为 1，这指定应首先检查它。所有其它应在启动期间检查的文件系统的 passno 应为 2 或更高。

有时候，您会发现 fsck 在重新引导后不能完全修复一个部分损坏的文件系统。在这些情况下，您所能做的就是将系统降为单用户模式，然后手工运行 fsck，并将这个分区块设备作为参数提供。在 fsck 进行文件系统修复时，它可能会问您是否修复特殊的文件系统缺陷。通常情况下，您应该对所有这些问题回答 y (yes) 并允许 fsck 执行它的工作。

fsck 的问题

发布时间 :2007-01-27 11:32:27

fsck 扫描的问题之一是需要花较长的时间才能完成，因为它要扫描整个文件系统的元数据（内部数据结构）以确保其一致性。对于一些非常大的文件系统，用超过一小时的时间进行一次彻底的 fsck 并不罕见。

为了解决这个问题，人们设计了一种新的文件系统，名为日志记录文件系统（journaling filesystem）。日志记录文件系统记录一个近期对文件系统元数据所做更改的磁盘日志。如果发生崩溃，该文件系统驱动程序检查日志。因为日志含有磁盘上近期更改的精确记录，所以只需检查这部分文件系统元数据来找出错误。多亏这一重要的设计差异，对一个日志记录系统进行一致性检查通常只需大约几秒钟（不管文件有多大）。出于这个原因，日志记录文件系统正在 Linux 社区流行起来。

现在，让我们研究可用于 Linux 的各种文件系统。

ext2 文件系统

发布时间 :2007-01-27 11:33:38

做为参考，目前已到了ext4，现多用ext3。

ext2 文件系统多年来已经成为标准的 Linux 文件系统。对于大多数应用程序，它通常都有很好的性能，但它没有任何日志记录能力。这使得它不适合非常大的文件系统，因为执行 fscks 要花太多的时间。另外，由于每个 ext2 文件系统只能拥有固定数目的索引节点（inode），所以 ext2 有些内置的限制。可以这么说，通常认为 ext2 是一个非常健壮和有效的非日志记录文件系统。

内核：2.0+

日志记录：无

mkfs 命令：mke2fs

mkfs 示例：mke2fs /dev/hdc7

相关命令：debugfs、tune2fs 和 chatter

与性能相关的挂装选项：noatime 和 nodiratime

ext3 文件系统

发布时间 :2007-01-27 11:34:15

ext3 文件系统使用与 ext2 相同的磁盘格式，但增加了日志记录能力。事实上，在所有的 Linux 文件系统中，ext3 具有最广泛的日志记录支持，它不仅支持元数据日志记录，还支持有序日志记录（缺省）和完全的“元数据+数据”日志记录。这些“特殊”日志记录方式有助于确保数据完整性，而不象其它日志记录实现仅仅缩短 fsck 的运行时间。出于这个原因，如果数据完整性是绝对最重要的，那么 ext3 是可用的最佳文件系统。然而，这些数据完整性功能确实会在某种程度上影响性能。另外，因为 ext3 使用与 ext2 相同的磁盘格式，所以和它的非日志记录表亲（ext2）一样，它也受到同样的可伸缩性限制。如果您在寻找一个出色的、通用的同时又非常健壮的日志记录文件系统，那么 ext3 是很好的选择。

内核：2.4.16+

日志记录：元数据、有序数据写入和完全的“元数据+数据”

mkfs 命令：mke2fs -j

mkfs 示例：mke2fs -j /dev/hdc7

相关命令：debugfs、tune2fs 和 chattr

与性能相关的挂装选项：noatime 和 nodiratime

其它挂装选项：

data=writeback（禁用日志记录）

data=ordered（缺省值，将元数据日志记录和数据与元数据一起写到磁盘）

data=journal（用于数据和元数据完整性的完全数据日志记录。写操作性能降低一半。）

ReiserFS 文件系统

发布时间 :2007-01-27 11:34:47

ReiserFS 是一个相对较新的文件系统，它的设计目标是提供非常好的小文件性能、非常好的通用性能以及是非常可伸缩的。ReiserFS 使用元数据日志以避免长时间的 fsck，但日志实现可能使最近修改的数据在系统锁定时被毁坏。通常情况下，ReiserFS 有非常好的性能，但在装入特定种类的文件系统时可能出现某种性能反常的情况。另外，ReiserFS 的 fsck 工具还不成熟，因此从毁坏的文件系统恢复数据可能会有困难。这些问题中的很多是由于 ReiserFS 相对较新、仍在发展过程中。许多人因 ReiserFS 的速度和可伸缩性而喜爱它。

内核：2.4.0+（推荐 2.4.16+）

日志记录：元数据

mkfs 命令：mkreiserfs

mkfs 示例：mkreiserfs /dev/hdc7

与性能相关的挂装选项：noatime、nodiratime 和 notail

XFS 文件系统

发布时间 :2007-01-27 11:35:26

XFS 文件系统是正在由 SGI 移植到 Linux 的企业级日志记录文件系统。现成的内核中还没有 XFS，不过在 <http://oss.sgi.com/projects/xfv> 找到有关 XFS 的更多信息。

JFS 文件系统

发布时间 :2007-01-27 11:35:59

JFS 是一个由 IBM 移植到 Linux 的高性能日志记录文件系统。JFS 由 IBM 企业服务器使用，并且是为高性能应用程序而设计的。现成的内核中还没有 JFS。您可以在 JFS 项目网站学习更多有关 JFS 的知识。

VFAT

发布时间 :2007-01-27 11:36:27

VFAT 文件系统实际不是那种您可以选择用于存储 Linux 文件的文件系统。相反，它是一种与 DOS 兼容的文件系统驱动程序，允许您挂装基于 DOS 和 Windows FAT 的文件系统并与之交换数据。VFAT 文件系统驱动程序在标准 Linux 内核中存在。

引导系统

引导系统

发布时间 :2007-01-27 11:37:58

本节介绍 Linux 引导过程。我们将介绍引导装入程序的概念、如何在引导时设置内核选项以及如何检查引导日志以找出错误。

MBR

发布时间 :2007-01-27 11:38:41

不管安装的是哪个分发版（distribution），所有机器的引导过程都很相似。考虑下列硬盘示例：

```
+-----+
|  MBR  |
+-----+
| Partition 1: |
| Linux root (/) |
| containing |
| kernel and |
| system. |
+-----+
| Partition 2: |
| Linux swap |
+-----+
| Partition 3: |
| Windows 3.0 |
| (last booted |
| in 1992) |
+-----+
```

首先，计算机的 BIOS 读取硬盘上的头几个扇区。这些扇区包含一个非常小的程序，名为“主引导记录（Master Boot Record）”，或简称为“MBR”。MBR 已经存储了 Linux 内核在硬盘上的位置（在上面的示例中是分区 1），因此它将内核装入内存，然后启动它。

内核引导过程

发布时间 :2007-01-27 11:39:23

接下来您看到的東西（尽管它可能一闪而过）是与下面类似的一行：

```
Linux version 2.4.16 ( root@time.flatmonk.org) (gcc version 2.95.3 20010315 (release)) #1 Sat Jan 12  
19:23:04 EST 2002
```

这就是内核开始运行时所打印的第一行信息。首先是内核版本，接着是构建该内核的用户标识（通常是 root 用户），然后是构建它的编译器，最后是构建时的时间戳记。

那一行之后是来自内核的关于系统硬件的大量输出：处理器、PCI 总线、磁盘控制器、磁盘、串口、软驱、USB 设备、网络适配器、声卡和其它可能的设备将依次报告它们的状态。

/sbin/init

发布时间 :2007-01-27 11:40:01

当内核装入结束时，它启动一个名为 init 的程序。该程序直到系统关机才停止运行。如您所见，始终给它分配进程标识 1：

```
$ ps --pid 1
PID TTY          TIME CMD
 1 ?        00:00:04 init.system
```

init 程序通过运行一系列脚本来引导分发版的余下部分。这些脚本通常位于 /etc/rc.d/init.d 或 /etc/init.d 中，它们执行的服务有设置系统主机名、检查文件系统是否有错误、挂装附加的文件系统、启用联网以及启动打印服务等。当脚本运行结束时，init 启动名为 getty 的程序，该程序会显示登录提示符，然后您就可以开始使用了！

深入研究： LILO

发布时间 :2007-01-27 11:40:31

既然我们已经快速了解了引导过程，现在让我们更仔细地研究第一个部分：MBR 和装入内核。维护 MBR 是“引导装入程序”的职责。基于 x86 的 Linux 的两个最流行的引导装入程序是“LILO”（Linux LOader 的缩写）和“GRUB”（GRand Unified Bootloader 的缩写）。

LILO 是二者中出现得较早、较常用的引导装入程序。引导时会出现简短的“LILO boot:”提示符，说明 LILO 存在于您的系统上。请注意，您可能要在引导时按住 shift 键才能看到提示符，因为系统常常被配置为不间断地快速运行，从而会很快闪过这个画面。

LILO 提示符没有太多华而不实的东西，不过如果您按一下 <tab> 键，您会看到一个可引导的可选内核（或其它操作系统）列表。通常列表中只有一项。您可以输入其中一项的名称然后按 <enter> 来引导它。或者您可以简单地按一下 <enter>，在缺省情况下将引导列表中的第一项。

使用 LILO

发布时间 :2007-01-27 11:41:09

有时候您希望在引导期间将一个选项传递到内核。一些较常用的选项是：root= 用来指定替代的引导文件系统，init= 用来指定替代的 init 程序（如指定 init=/bin/sh 来挽救被错误配置的系统），mem= 指定系统中的内存数量（例如，在 Linux 只自动识别了 128 M 的情况下使用 mem=512M）。您可以在 LILO 引导提示符下向内核传递下列选项：

LILO boot: linux root=/dev/hdb2 init=/bin/sh mem=512M

如果您需要经常指定命令行选项，您应该考虑把它们添加到 /etc/lilo.conf 文件。

一个重要的 LILO 问题

发布时间 :2007-01-27 11:41:48

在转向讨论 GRUB 以前，还要讨论一个重要的 LILO 问题。只要您对 /etc/lilo.conf 做了更改，或安装了新的内核，您都必须运行 lilo。lilo 程序会重写 MBR 以反映您所做的更改，包括记录内核的绝对磁盘位置。这里的示例利用 -v 标记进行详细信息描述。

```
# lilo -v
LILO version 21.4-4, Copyright (C) 1992-1998 Werner Almesberger
'liba32' extensions Copyright (C) 1999,2000 John Coffman
```

```
Reading boot sector from /dev/hda
Merging with /boot/boot.b
Mapping message file /boot/message
Boot image: /boot/vmlinuz-2.2.16-22
Added linux *
/boot/boot.0300 exists - no backup copy made.
Writing boot sector.
```

深入研究： GRUB

发布时间 :2007-01-27 11:42:38

GRUB 引导装入程序被认为是继 LILO 之后的下一代引导装入程序。对用户而言，最显而易见的是它用一个菜单界面取代了 LILO 的原始提示符。对于系统管理员而言，改变更为显著。GRUB 比 LILO 支持更多的操作系统，它在引导菜单中提供了基于密码的安全性，并且更易于管理。

通常用 grub-install 命令安装 GRUB。安装完毕后，可以编辑文件 /boot/grub/menu.lst 来管理 GRUB 菜单。这两个任务都超出了本文的讨论范围；在试图安装或管理 GRUB 之前，您应该阅读 GRUB 信息页。

使用 GRUB

发布时间 :2007-01-27 11:43:21

要向内核传递参数，您可以在引导菜单上按 e。这使您有机会编辑（再次按 e）要装入的内核名称或传递给它的参数。当您结束编辑后，按 <enter> 键，然后按 b 键以用您所做的更改来引导。

LILO 与 GRUB 之间值得一提的显著区别是：GRUB 不需要在每次更改配置或安装新内核后重新安装其引导装入程序。这是因为 GRUB 理解 Linux 文件系统，而 LILO 只是存储要装入内核的绝对磁盘位置。当系统管理员安装新内核后忘了输入 lilo 时，GRUB 的这一简单事实可以缓解他们的挫折感。

dmesg

发布时间 :2007-01-27 11:44:07

来自内核和 init 脚本的引导消息通常在屏幕上停留的时间很短，您可能会注意到一个错误消息，但在您能看清楚它以前，它已经消失了。在这种情况下，您可以在系统引导之后到两个地方去查看以找出错误（并且有希望找到修复它的方法）。

如果错误出现在内核正在装入或检测硬件设备的时候，您可以用 dmesg 命令获得一个内核日志的副本。

```
# dmesg | head -1
```

```
Linux version 2.4.16 ( root@time.flatmonk.org) (gcc version 2.95.3 20010315 (release)) #1 Sat Jan 12  
19:23:04 EST 2002
```

嘿，我们认得那一行！这就是内核在装入时打印的第一行信息。的确，如果您将 dmesg 的输出传送到分页程序上，您会看到内核在引导时打印的所有消息以及内核在那段时间已打印到控制台的所有消息。

/var/log/messages

发布时间 :2007-01-27 11:44:39

第二个查看信息的地方在 /var/log/messages 文件中。该文件由 syslog 守护程序记录，syslog 守护程序接受来自库、守护程序和内核的输入。消息文件中的每一行都被打上时间戳记。这个文件是查找引导期间的 init 脚本阶段出现的错误的好位置。例如，要查看来自名称服务器的最后几条消息：

```
# grep named /var/log/messages | tail -3
Jan 12 20:17:41 time /usr/sbin/named[350]: listening on IPv4 interface lo, 127.0.0.1#53
Jan 12 20:17:41 time /usr/sbin/named[350]: listening on IPv4 interface eth0, 10.0.0.1#53
Jan 12 20:17:41 time /usr/sbin/named[350]: running
```

单用户方式

发布时间 :2007-01-27 11:45:10

我们知道，可以在内核引导时向它传递参数。最常使用的参数之一是 s，它使系统以“单用户”方式启动。这一方式通常只挂装根文件系统，启动 init 脚本的最小子集，然后启动 shell 而不是提供登录提示符。另外，没有配置联网，因此外部因素没有机会影响您的工作。

使用单用户方式

发布时间 :2007-01-27 11:45:38

那么在这样的状态下我们能够“做什么”呢？要回答这个问题，我们必须认识到 Linux 和 Windows 之间的巨大差异。Windows 被设计成通常在同一时间内只能由坐在控制台前的一个人使用。它实际上一直处于“单用户”方式。相反，Linux 更多地用于为网络应用程序服务，或为网络上的远程用户提供 shell 或 X 会话。当您希望执行维护操作（如：从备份中恢复、创建或修改文件系统、从 CD 升级系统等等）时，您不希望有这些额外的变数。在这些情况下，您应该使用单用户方式。

更改运行级别

发布时间 :2007-01-27 11:46:17

事实上，要进入单用户方式不必重新引导。init 程序管理系统的当前方式（或称为“运行级别（runlevel）”）。Linux 系统的标准运行级别按如下分类并定义：

- 0：停止计算机
- 1 或 s：单用户方式
- 2：多用户，无网络
- 3：多用户，文本控制台
- 4：多用户，图形控制台
- 5：同 4
- 6：重新引导计算机

这些运行级别因分发版而异，所以请确保参考您的分发版的文档。要转至单用户方式，可以使用 telinit 命令，它指示 init 更改运行级别：

```
# telinit 1
```

从上表中您可以看到您还可以用这种方式关闭或重新引导系统。telinit 0 将停机；telinit 6 将重新引导计算机。当您发出 telinit 命令更改运行级别时，init 脚本的一个子集将运行以关闭或启动系统服务。

恰当地关机

发布时间 :2007-01-27 11:46:49

然而，如果还有用户在使用系统，那么此时关机是相当粗鲁的（用户可能会非常生气）。shutdown 命令提供了一种方法，以一种合理对待用户的方式来更改运行级别。类似于 kill 命令那种可以向一个进程发送多种信号的能力，shutdown 可以用来停机、重新引导或转至单用户方式。例如，要在 5 分钟内转至单用户方式：

```
# shutdown 5  
Broadcast message from root (pts/2) (Tue Jan 15 19:40:02 2002):  
The system is going DOWN to maintenance mode in 5 minutes!
```

如果此时您按 control-c 组合键，您可以取消切换至单用户方式前的延时等待。上面的消息将在系统的所有终端上出现，因此用户有合理数量的时间保存他们的工作并注销。（有些人可能会争论 5 分钟是不是“合理”的）。

立即关机

发布时间 :2007-01-27 11:47:21

如果您是唯一使用系统的人，您可以用“now”代替以分钟为单位的参数。例如，要立即重新引导系统：

```
# shutdown -r now
```

在此情况下，您没有机会按 control-c 组合键；因为系统已经在进行关机了。最后，-h 选项使系统停机：

```
# shutdown -h 1
Broadcast message from root (pts/2) (Tue Jan 15 19:50:58 2002):
The system is going DOWN for system halt in 1 minute!
```

缺省运行级别

发布时间 :2007-01-27 11:47:51

此时您可能已得出结论：init 程序在 Linux 系统上是非常重要的。您可以编辑文件 /etc/inittab 来配置 init，这在 inittab(5) 手册页中有描述。我们只讨论这个文件中的关键一行。

```
# grep ^id: /etc/inittab  
id:3:initdefault:
```

在我的系统上，运行级别 3 是缺省运行级别。如果您更喜欢您的系统立即引导至图形登录界面（通常为运行级别 4 或 5），那么更改这个值会有用。要那样做的话，只需简单地编辑文件并更改那一行的值。但要小心！如果您将它改为某个无效值，您可能不得不使用我们在前面提到的 `init=/bin/sh` 技巧。

运行级别

运行级别 - 单用户方式

发布时间 :2007-01-27 11:48:41

还记得关于引导装入程序一节所讲的吗？在引导内核时向它传递参数是可能的。最常用的参数之一是 s，它使系统以“单用户”方式启动。这一方式通常只挂装根文件系统，启动 init 脚本的最小子集，然后启动 shell 而不是提供登录提示符。另外，它不配置联网，因此外部因素没有机会影响您的工作。

单用户方式（续）

发布时间 :2007-01-27 11:49:13

那么在这样的状态下我们能够“做什么”呢？要回答这个问题，我们必须认识到 Linux 和 Windows 之间的巨大差异。Windows 被设计成通常在同一时间内只能由坐在控制台前的一个人使用。它实际上一直处于“单用户”方式。相反，Linux 更多地用于为网络应用程序服务，或为网络上的远程用户提供 shell 或 X 会话。当您希望执行维护操作（如：从备份中恢复、创建或修改文件系统、从 CD 升级系统等等）时，您不希望有这些额外的变数。在这些情况下，您应该使用单用户方式。

运行级别

发布时间 :2007-01-27 11:49:46

事实上，要进入单用户方式不必重新引导。init 程序管理系统的当前方式（或称为“运行级别”）。Linux 系统的标准运行级别按如下分类并定义：

- 0：停止计算机
- 1 或 s：单用户方式
- 2：多用户，无网络
- 3：多用户，文本控制台
- 4：多用户，图形控制台
- 5：同 4
- 6：重新引导计算机

这些运行级别因分发版而异，所以请确保参考您的分发版的文档。

telinit

发布时间 :2007-01-27 11:50:19

要转至单用户方式，可以使用 telinit 命令，它指示 init 更改运行级别：

```
# telinit 1
```

从上表中您可以看到您还可以用这种方式关机或重新引导系统。telinit 0 将停机；telinit 6 将重新引导计算机。当您使用 telinit 命令更改运行级别时，init 脚本的一个子集将运行以关闭或启动系统服务。

运行级别规则

发布时间 :2007-01-27 11:50:55

然而，如果还有用户在使用系统，那么此时关机是相当粗鲁的（用户可能会非常生气）。shutdown 命令提供了一种方法，以一种合理对待用户的方式来更改运行级别。类似于 kill 命令那种可以向一个进程发送多种信号的能力，shutdown 以用来停机、重新引导或转至单用户方式。例如，要在 5 分钟内转至单用户方式：

```
# shutdown 5
Broadcast message from root (pts/2) (Tue Jan 15 19:40:02 2002):
The system is going DOWN to maintenance mode in 5 minutes!
```

如果此时您按 control-c 组合键，您可以取消切换至单用户方式前的延时等待。上面的消息将在系统的所有终端上出现，因此用户有合理数量的时间保存他们的工作并注销。（有些人可能会争论 5 分钟是不是“合理”的）。

“ now ” 和停机

发布时间 :2007-01-27 11:51:34

如果您是唯一使用系统的人，您可以用 now 代替以分钟为单位的参数。例如，要立即重新引导系统：

```
# shutdown -r now
```

在此情况下，您没有机会按 control-c 组合键；因为系统已经在进行关机了。最后，-h 选项使系统停机：

```
# shutdown -h 1
Broadcast message from root (pts/2) (Tue Jan 15 19:50:58 2002):
The system is going DOWN for system halt in 1 minute!
```

配置 init

发布时间 :2007-01-27 11:52:10

此时您可能已得出结论：init 程序在 Linux 系统上是非常重要的。您可以编辑文件 /etc/inittab 来配置 init，这在 inittab(5) 手册页中有描述。我们只讨论这个文件中的关键一行。

```
# grep ^id: /etc/inittab  
id:3:initdefault:
```

在我的系统上，运行级别 3 是缺省运行级别。如果您更喜欢您的系统立即引导至图形登录（通常为运行级别 4 或 5），那么更改这个值会有用。要那样做的话，只需简单地编辑文件并在那一行更改那个值。但要小心！如果您将它改为某个无效值，您可能不得不使用我们在前面提到的 `init=/bin/sh` 技巧。

文件系统限额

文件系统限额

发布时间 :2007-01-27 11:52:48

限额（quota）是 Linux 的一个特性，它让您跟踪用户或组的磁盘使用情况。它们可用于防止任何单个用户或组不公平地使用文件系统的一部分或将它填满。限额只能由 root 用户启用和管理。在本节中，我将介绍如何在 Linux 系统上设置限额并有效地管理它们。

内核支持

发布时间 :2007-01-27 11:53:28

限额是文件系统的一个特性；因此，它们需要内核支持。首先需要做的是验证您的内核支持限额。您可以用 `grep` 做到这一点：

```
# cd /usr/src/linux
# grep -i quota .config
CONFIG_QUOTA=y
CONFIG_XFS_QUOTA=y
```

如果难以用该命令返回的信息作出结论（如未设置 `CONFIG_QUOTA`），那么您应该重新构建内核以包含限额支持。这一过程并不困难，但超出本教程这一节的讨论范围之外。如果您不熟悉构建和安装新内核的步骤，您应该考虑参考这个教程。

文件系统支持

发布时间 :2007-01-27 17:27:53

在深入研究限额管理以前，请注意 2.4.x 内核为止的 Linux 系列上的限额支持尚不完整。限额目前在 ext2 和 ext3 文件系统中还存在问题，而 ReiserFs 似乎根本不支持限额。本教程的示例使用的是 XFS，它似乎可以较好地支持 quota。

配置限额

发布时间 :2007-01-27 17:28:29

要开始在系统配置限额，您应该编辑 `/etc/fstab` 文件以挂装受启用限额影响的文件系统。在我们的示例中，我们使用挂装时启用用户和组限额的 XFS 文件系统。

```
# grep quota /etc/fstab
/usr/users /mnt/hdc1 xfs  usrquota,grpquota,noauto 0 0
# mount /usr/users
```

配置限额（续）

发布时间 :2007-01-27 17:29:11

请注意：usrquota 和 grpquota 选项不一定在文件系统上启用限额。要确保启用了限额，您可以使用 quotaon 命令：

```
# quotaon /usr/users
```

如果您想以后禁用限额，则有一个相应的 quotaoff 命令：

```
# quotaoff /usr/users
```

但在目前，如果您正在尝试本教程中的几个示例，请确保启用了限额。

quota 命令

发布时间 :2007-01-27 17:29:45

quota 命令显示了当前挂装的所有文件系统的用户磁盘使用情况和限制。-v 选项包括启用了限额的文件系统列表，但当前没有给用户分配存储空间。

```
# quota -v
```

Disk quotas for user root (uid 0):

Filesystem	blocks	quota	limit	grace	files	quota	limit	grace
/dev/hdc1	0	0	0	3	0	0		

第一列 blocks 显示 root 用户当前在每个列出的文件系统上正在使用多少磁盘空间。接下来的 quota 和 limit 列显示当前的磁盘空间限制。我们稍后将解释 quota 和 limit 之间的区别以及 grace 列的含义。files 列显示 root 用户在特定文件系统上拥有多少文件。其后的 quota 和 limit 列则显示对这些文件的限制。

查看限额

发布时间 :2007-01-27 17:30:18

任何用户都可以使用 quota 命令查看自己的限额报告，如前一个示例所示。但是只有 root 用户可以查看其他用户和组的限额。例如，假设我们在 /usr/users 上挂装了一个文件系统 /dev/hdc1，并且有两个用户：jane 和 john。首先，让我们看看 jane 的磁盘使用情况和限制。

```
# quota -v jane
```

Disk quotas for user jane (uid 1003):

Filesystem	blocks	quota	limit	grace	files	quota	limit	grace
/dev/hdc1	4100	0	0	6	0	0		

在这一示例中，我们看到 jane 的 quota 被设置为零，这表示没有限制。

edquota

发布时间 :2007-01-27 17:30:52

现在假设我们希望给用户 jane 一个限额。我们用 edquota 命令实现。在我们开始编辑限额之前，让我们看看我们在 /usr/users 上还有多少可用空间：

```
# df /usr/users
```

Filesystem	1k-blocks	Used	Available	Use%	Mounted on
/dev/hdc1	610048	4276	605772	1%	/usr/users

这不是一个特别大的文件系统，仅仅 600M 左右。给 jane 一个限额似乎是明智的，这样她所用的空间就不会超过她应得的空间。当您运行 edquota 时，会为您在命令行指定的每个用户或组创建一个临时文件。

edquota（续）

发布时间 :2007-01-27 17:31:25

edquota 命令为您提供一个编辑器，它使您能通过这个临时文件添加和 / 或修改 quota。

```
# edquota jane
```

Disk quotas for user jane (uid 1003):

Filesystem	blocks	soft	hard	inodes	soft	hard
/dev/hdc1	4100	0	0	6	0	0

与上面 quota 命令的输出相似，这个临时文件中的 blocks 和 inodes 列显示 jane 当前正在使用的磁盘空间和文件数目。您不能修改 blocks 或 inodes 的数量；任何这样的尝试都将立即被系统废弃。soft 和 hard 列显示 jane 的限额，我们可以看到当前对它没有限制（零表明没有限额）。

理解 edquota

发布时间 :2007-01-27 17:31:56

soft 限制是在文件系统上分配给 jane 的磁盘空间最大使用量（也就是她的限额）。如果 jane 使用的磁盘空间数量超过在 soft 限制中分配给她的空间数量，将通过电子邮件就她的违规行为提出警告。hard 限制表明对磁盘使用的绝对限制，用户不能超过该限制。如果 jane 试图使用的磁盘空间多于 hard 限制中指定的磁盘空间，她将得到 “ Disk quota exceeded ” 错误信息并且不能完成该操作。

进行更改

发布时间 :2007-01-27 17:32:33

那么我们在这里更改 jane 的 soft 和 hard 限制，然后保存该文件：

Disk quotas for user jane (uid 1003):

Filesystem	blocks	soft	hard	inodes	soft	hard
/dev/hdc1	4100	10000	11500	6	2000	2500

运行 quota 命令，我们可以检查我们所做的修改：

```
# quota jane
```

Disk quotas for user jane (uid 1003):

Filesystem	blocks	quota	limit	grace	files	quota	limit	grace
/dev/hdc1	4100	10000	11500		6	2000	2500	

复制限额

发布时间 :2007-01-27 17:33:09

您会记得在这个文件系统上我们还有另一个用户 john。如果我们希望给 john 的限额与 jane 的限额相同，我们可以在 edquota 命令中使用 -p 选项，它会将 jane 的限额作为原型用于命令行上所有随后的用户。这是一个为用户组设置限额的简便方法。

```
# edquota -p jane john
# quota john
```

Disk quotas for user john (uid 1003):

Filesystem	blocks	quota	limit	grace	files	quota	limit	grace
/dev/hdc1	0	10000	11500		1	2000	2500	

组限制

发布时间 :2007-01-27 17:33:49

使用 edquota , 我们还可以根据文件的组所有权来限制磁盘空间分配。例如 , 要编辑 users 组的限额 :

```
# edquota -g users Disk quotas for group users (gid 100): Filesystem blocks soft hard inodes soft hard
/dev/hdc1 4100 500000 510000 7 100000 125000
```

然后要查看修改的 users 组的限额 :

```
# quota -g users Disk quotas for group users (gid 100): Filesystem blocks quota limit grace files quota limit
grace /dev/hdc1 4100 500000 510000 7 100000 125000
```

repquota 命令

发布时间 :2007-01-27 17:34:26

如果您的文件系统有很多用户，那么使用限额命令查看每个用户的限额会比较麻烦。repquota 命令将文件系统的限额以一个清楚易懂的报告进行总结。例如，要查看 /usr/users 上所有用户和组的限额：

```
# repquota -ug /usr/users
```

```
*** Report for user quotas on device /dev/hdc1
```

```
Block grace time: 7days; Inode grace time: 7days
```

		Block limits				File limits			
User		used	soft	hard	grace	used	soft	hard	grace
root	--	0	0	0	3	0	0		
john	--	0	10000	11500		1	2000	2500	
jane	--	4100	10000	11500		6	2000	2500	

```
*** Report for group quotas on device /dev/hdc1
```

```
Block grace time: 7days; Inode grace time: 7days
```

		Block limits				File limits			
Group		used	soft	hard	grace	used	soft	hard	grace
root	--	0	0	0	3	0	0		
users	--	4100	500000	510000		7	100000	125000	

repquota 选项

发布时间 :2007-01-27 17:34:59

repquota 有其它两个选项值得一提。repquota -a 将报告所有启用了限额并且当前已挂装的读-写文件系统。repquota -n 将不把 uid 和 gid 解析为名称。这可以加快大型列表的输出。

监控限额

发布时间 :2007-01-27 17:35:32

如果您是系统管理员，您会希望有一种方法可以监控限额以确保没有用户超过它们。实现它的一个简单方法是使用 `warnquota`。`warnquota` 命令会向超过 `soft` 限制的用户发送电子邮件。通常将 `warnquota` 作为 `cron` 作业运行。

当用户超过 `soft` 限制时，`quota` 命令输出中的 `grace` 列将指出宽限时间 — 在该文件系统强制执行 `soft` 限制之前有多少时间。

Disk quotas for user jane (uid 1003):

Filesystem	blocks	quota	limit	grace	files	quota	limit	grace
/dev/hdc1	10800*	10000	11500	7days	7	2000	2500	

缺省情况下，`blocks` 和 `inodes` 的宽限时间是 7 天。

修改宽限时间

发布时间 :2007-01-27 17:36:06

您可以用 equota 修改文件系统的宽限时间：

```
# edquota -t
```

这为您提供一个临时文件编辑器，它看起来与下面相似：

```
Grace period before enforcing soft limits for users:
Time units may be: days, hours, minutes, or seconds
Filesystem      Block grace period   Inode grace period
/dev/hdc1       7days                7days
```

该文件中的文本简单明了，无需解释。请确保给您的用户足够的时间来接收警告电子邮件并找出一些要删除的文件！

在引导时检查限额

发布时间 :2007-01-27 17:36:40

您可能还希望在引导时检查限额。要实现这一点，您可以使用一个脚本运行 quotacheck 命令；在 Quota Mini HOWTO 有一个脚本示例。quotacheck 命令还具有修复损坏的限额文件的能力；请阅读 quotacheck(8) 手册页来熟悉这一点。

还要记住我前面提到的关于 quotaon 和 quotaoff 的内容。您应该将 quotaon 合并到引导脚本中以启用限额。要在所有支持限额的文件系统上启用 quota，请使用 -a 选项：

```
# quotaon -a
```

系统日志

系统日志 - 介绍 syslogd

发布时间 :2007-01-27 17:37:31

syslog 守护程序为记录来自运行于系统之上的程序的消息提供了一种成熟的客户机-服务器机制。syslog 接收来自守护程序或程序的消息，根据优先级和类型将该消息分类，然后根据由管理员可配置的规则将它写入日志。结果是一个健壮而统一的管理日志的方法。

读取日志

发布时间 :2007-01-27 17:38:06

让我们跳过一些内容来直接研究一个 syslog 记录的日志文件的内容。之后我们再回过头研究 syslog 配置。FHS（请参阅本教程系列的第 2 部分）要求将日志文件放在 /var/log 中。我们在这里使用 tail 命令来显示 “messages” 文件中的最后 10 行：

```
# cd /var/log
# tail messages
Jan 12 20:17:39 bilbo init: Entering runlevel: 3
Jan 12 20:17:40 bilbo /usr/sbin/named[337]: starting BIND 9.1.3
Jan 12 20:17:40 bilbo /usr/sbin/named[337]: using 1 CPU
Jan 12 20:17:41 bilbo /usr/sbin/named[350]: loading configuration from '/etc/bind/named.conf'
Jan 12 20:17:41 bilbo /usr/sbin/named[350]: no IPv6 interfaces found
Jan 12 20:17:41 bilbo /usr/sbin/named[350]: listening on IPv4 interface lo, 127.0.0.1#53
Jan 12 20:17:41 bilbo /usr/sbin/named[350]: listening on IPv4 interface eth0, 10.0.0.1#53
Jan 12 20:17:41 bilbo /usr/sbin/named[350]: running
Jan 12 20:41:58 bilbo gnome-name-server[11288]: starting
Jan 12 20:41:58 bilbo gnome-name-server[11288]: name server starting
```

我们希望您能在文本处理的纷乱事件中记得 tail 命令显示文件的最后几行。在本示例中，我们可以看到最近在这个系统上启动了名为 bilbo 的名称服务器 named。如果我们在部署 IPv6，我们可能会注意到 named 找不到 IPv6 接口，这表明可能出现了问题。另外，我们可以看到最近可能有用户已经启动了 GNOME，这可以由 gnome-name-server 的存在看出。

跟踪日志文件

发布时间 :2007-01-27 17:38:38

有经验的系统管理员可能会使用 `tail -f` 以便当日志文件的输出出现时进行跟踪：

```
# tail -f /var/log/messages
```

例如，以调试理论上的 IPv6 问题为例，在停止和启动 `named` 时在一个终端运行上述命令会立即显示来自该守护程序的消息。这在调试时是一个有用的技术。有些管理员甚至喜欢在终端上一直运行 `tail -f messages`，这样他们可以随时关注系统事件。

查找日志

发布时间 :2007-01-27 17:39:14

另一种有用的技术是使用 grep 实用程序搜索日志文件，这在本教程系列的第 2 部分有描述。在上述示例中，我们可以使用 grep 找到 “ named ” 行为发生改变的地方：

```
# grep named /var/log/messages
```

日志概述

发布时间 :2007-01-27 17:39:43

以下内容概括了通常位于 /var/log 并由 syslog 维护的日志文件：

messages：来自一般系统程序和守护程序的信息性与错误消息

secure：认证消息与错误，为了额外的安全性而与“message”分隔

maillog：与邮件相关的消息与错误

cron：与 cron 相关的消息与错误

spooler：UUCP 和与新闻相关的消息与错误

syslog.conf

发布时间 :2007-01-27 17:41:26

事实上，现在是研究 syslog 配置文件 /etc/syslog.conf 的时候了。（注：如果您没有 syslog.conf，为了获得信息请继续阅读，不过您可以使用另一个 syslog 守护程序）。通过浏览那个文件，我们发现以上提到的每个常见日志文件都有项，可能还有其它项。该文件的格式为 facility.priority action，这些字段的定义如下：

facility

指定产生消息的子系统。facility 有效的关键字有 auth、authpriv、cron、daemon、kern、lpr、mail、news、syslog、user、uucp 以及 local0 到 local7。

priority

指定消息的最低严重性，即此优先级和高于此优先级的消息将由这个规则匹配。priority 的有效关键字有 debug、info、notice、warning、err、crit、alert 和 emerg。

action

action 字段可以是文件名、tty（如 /dev/console）、以 @ 为前缀的远程机器、以逗号分隔的用户列表，或是 * 以表明向所有登录用户发送消息。最常见的操作是一个简单的文件名。

syslog.conf

发布时间 :2007-01-27 17:41:44

事实上，现在是研究 syslog 配置文件 /etc/syslog.conf 的时候了。（注：如果您没有 syslog.conf，为了获得信息请继续阅读，不过您可以使用另一个 syslog 守护程序）。通过浏览那个文件，我们发现以上提到的每个常见日志文件都有项，可能还有其它项。该文件的格式为 facility.priority action，这些字段的定义如下：

facility

指定产生消息的子系统。facility 有效的关键字有 auth、authpriv、cron、daemon、kern、lpr、mail、news、syslog、user、uucp 以及 local0 到 local7。

priority

指定消息的最低严重性，即此优先级和高于此优先级的消息将由这个规则匹配。priority 的有效关键字有 debug、info、notice、warning、err、crit、alert 和 emerg。

action

action 字段可以是文件名、tty（如 /dev/console）、以 @ 为前缀的远程机器、以逗号分隔的用户列表，或是 * 以表明向所有登录用户发送消息。最常见的操作是一个简单的文件名。

重新装入和附加信息

发布时间 :2007-01-27 17:42:14

希望这个配置文件的概述有助于您体验 syslog 系统的长处。您应该在进行更改之前阅读 `syslog.conf(5)` 手册页来获得更多的信息。另外，`syslogd(8)` 手册页提供了更详细的信息。

请注意：在对该配置文件的更改生效前，您需要向 syslog 守护程序通知所做的更改。向它发送 SIGHUP 是个正确的办法，您可以用 `killall` 命令轻松地做到这一点：

```
# killall -HUP syslogd
```

安全性注释

发布时间 :2007-01-27 17:42:43

您应该清楚如果 syslogd 写的日志文件还不存在的话，程序将创建它们。无论您当前的 umask 如何设置，该文件将被创建为可被所有用户读取。如果您关心安全性，那么您应该用 chmod 命令将该文件设置为仅 root 用户可读写。此外，可以用适当的许可权配置 logrotate 程序（在下面描述）以创建新的日志文件。syslog 守护程序始终会保留现有日志文件的当前属性，因此一旦创建了文件，您就不需要担心它。

logrotate

发布时间 :2007-01-27 17:43:17

/var/log 中的日志文件将随时间而变大，并有可能填满文件系统。建议利用 logrotate 这样的程序来管理日志的自动归档。logrotate 程序通常作为日常 cron 作业运行，并且可以配置为对日志文件执行循环、压缩、除去或发送电子邮件等操作。

例如，logrotate 的缺省配置会每周循环日志，保留 4 周的备份日志（通过在文件名后附加序号），并且压缩备份日志以节省空间。另外，还可以将该程序配置成将 SIGHUP 发送到 syslogd，这样守护程序将注意到现在为空的日志文件，并会适当地给它们附加信息。

有关 logrotate 的更多信息，请参阅 logrotate(8) 手册页，它包含该程序的描述以及配置文件的语法。

高级主题 — klogd

发布时间 :2007-01-27 17:43:47

在结束 syslog 的讨论之前，我想为渴望了解更多知识的读者提及几个高级主题。在您尝试理解与 syslog 相关的主题时，这些提示会使您免去一些麻烦。

首先，syslog 守护程序实际上是 sysklogd 软件包的一部分，这个软件包有另一个守护程序，名为 klogd。klogd 的任务是接收来自内核的信息和错误消息，然后将它们传递到 syslogd 进行分类和日志记录。klogd 接收到的消息和您可以使用 dmesg 命令获得的消息完全相同。区别是 dmesg 打印内核中环形缓冲区（ring buffer）的当前内容，而 klogd 将消息传递到 syslogd 使它们不会因环形缓冲区被覆盖而丢失。

高级主题 — 替代日志记录器

发布时间 :2007-01-27 17:44:17

其次，标准 `sysklogd` 包有替代选项。这些替代选项试图比 `sysklogd` 更有效、更易于配置，而且功能尽可能更丰富。`Syslog-ng` 和 `Metalog` 似乎是较受欢迎的替代项中的几个；如果您觉得 `syslogd` 达不到您所需的功能级别，您可以研究它们。

最后，您可以在脚本中使用 `logger` 命令记录消息。请参阅 `logger(1)` 手册页以获得更多信息。

Linux资源汇集

海量 Linux 技术文章

发布时间 :2006-11-24 16:50:29

下面是linux技术文章快速入口。需要联网：

[Linux 技术交流](#)

<http://www.linuxdiyf.com/bbs/forum-3-1.html>

[Linux 应用](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=1>

[Linux 安装及学习指导](#)

<http://www.linuxdiyf.com/bbs/forum-45-1.html>

[Linux 系统安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=11>

[Linux 学习指导](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=12>

[Linux 软件安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=13>

[shell](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=3>

[Linux 壁纸](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=4>

[红旗](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=5>

[Redhat](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=6>

[SuSE](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=7>

Linux 认证

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=9>

Linux下载分享{酷件、书籍、视频分享 }

<http://www.linuxdiyf.com/bbs/forum-6-1.html>

服务器应用

<http://www.linuxdiyf.com/bbs/forum-7-1.html>

数据库应用

<http://www.linuxdiyf.com/bbs/forum-8-1.html>

Linux 编程与内核

<http://www.linuxdiyf.com/bbs/forum-9-1.html>

UniX 技术文章

<http://www.linuxdiyf.com/bbs/forum-32-1.html>

Linux 业界声音、新闻

<http://www.linuxdiyf.com/bbs/forum-11-1.html>

Linux 人才招聘信息

<http://www.linuxdiyf.com/bbs/forum-46-1.html>

网络转载，感谢原创作者！

制作：红联Linux论坛

祝您阅读愉快！

Linux培训系列

第五讲

欢迎学习“编译源代码和管理软件包”，在本教程中，我们将向您演示如何从源代码编译程序、如何管理共享库以及如何使用 Red Hat 和 Debian 软件包管理系统。

内容基础，语言简短简洁

红联Linux论坛是致力于Linux技术讨论的站点，目前网站收录的文章及教程基本能满足不同水平的朋友学习。

红联Linux门户：www.linux110.com

红联Linux论坛：www.linuxdiyf.com/bbs

下载:Linux电子书籍：

<http://www.linux286.com/linux/linuxdzsj.htm>

目录

共享库

共享库

静态可执行程序与动态可执行程序比较

动态链接相关性

动态装入器

ld.so.conf

ld.so.cache

ldconfig 技巧

LD_LIBRARY_PATH

从源代码编译应用程序

从源代码编译应用程序 - 介绍

下载

解包

列出压缩文档

解包 bzip2 压缩的压缩文档

bzip2 管道

bzip2 管道 (续)

检查源代码

配置

使用配置

--prefix 选项

使用 --prefix

FHS 怎么样?

该配置了

该配置了 (续)

config.cache

配置脚本和制作文件

制作文件介绍

调用 make

安装

make install

安装之后

好, 完成了!

可能出现的问题

遗漏一些库

其它问题

其它问题 (续)

软件包管理概念

软件包管理概念

软件包管理的缺点

rpm , Red Hat 软件包管理器

rpm , Red Hat 软件包管理器

安装 rpm

重新安装一个 rpm

强制安装一个 rpm

用 --nodeps 安装或删除

更新软件包

用 rpm -q 查询

用 `rpm -ql` 列出文件
用 `rpm -qp` 查询软件包
查询所有已安装的软件包
查找文件的所有者
显示相关性
验证软件包的完整性
验证已安装的软件包
配置 `rpm`

Debian 软件包管理

Debian 软件包管理 - 介绍 `apt-get`
模拟安装
软件包资源列表：`apt-setup`
从 `apt-get` 到 `dselect`
启动 `dselect`
使用 `dselect` 的 `Select` 方式
软件包状态
安装和配置 (`dpkg-reconfigure`)
获取已安装软件包的状态
文件与其 `.deb` 之间的链接
查找要安装的软件包

Linux海量文章

海量Linux技术文章

共享库

共享库

发布时间 :2007-01-27 19:50:34

Linux 系统上有两类根本不同的 Linux 可执行程序。第一类是静态链接的可执行程序。静态可执行程序包含执行所需的所有函数 — 换句话说，它们是“完整的”。因为这一原因，静态可执行程序不依赖任何外部库就可以运行。

第二类是动态链接的可执行程序。我们将在下页讨论这一内容。

静态可执行程序与动态可执行程序比较

发布时间 :2007-01-27 19:51:11

我们可以用 ldd 命令来确定某一特定可执行程序是否为静态链接的：

```
# ldd /sbin/sln
not a dynamic executable
```

“ not a dynamic executable ” 是 ldd 说明 sln 是静态链接的一种方式。现在，让我们比较 sln 与其非静态同类 ln 的大小：

```
# ls -l /bin/ln /sbin/sln
-rwxr-xr-x 1 root root 23000 Jan 14 00:36 /bin/ln
-rwxr-xr-x 1 root root 381072 Jan 14 00:31 /sbin/sln
```

如您所见，sln 的大小超过 ln 十倍。ln 比 sln 小这么多是因为它是动态可执行程序。动态可执行程序是不完整的程序，它依靠外部共享库来提供运行所需的许多函数。

动态链接相关性

发布时间 :2007-01-27 19:51:45

要查看 ln 依赖的所有共享库的列表，可以使用 ldd 命令：

```
# ldd /bin/ln
libc.so.6 => /lib/libc.so.6 (0x40021000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

如您所见，ln 依赖外部共享库 libc.so.6 和 ld-linux.so.2。通常，动态链接的程序比其静态链接的等价程序小得多。不过，静态链接的程序可以在某些低级维护任务中发挥作用。例如，sln 是修改位于 /lib 中的不同库符号链接的极佳工具。但通常您会发现几乎所有 Linux 系统上的可执行程序都是某种动态链接的变体。

动态装入器

发布时间 :2007-01-27 19:52:19

那么，如果动态可执行程序不包含运行所需的所有函数，Linux 的哪部分负责将这些程序和所有必需的共享库一起装入，以使它们能正确执行呢？答案是动态装入器（dynamic loader），它实际上是您在 `ld` 的 `ldd` 清单中看到的作为共享库相关性列出的 `ld-linux.so.2` 库。动态装入器负责装入动态链接的可执行程序运行所需的共享库。现在，让我们迅速查看一下动态装入器如何在系统上找到适当的共享库。

ld.so.conf

发布时间 :2007-01-27 19:52:54

动态装入器找到共享库要依靠两个文件 — /etc/ld.so.conf 和 /etc/ld.so.cache。如果您对 /etc/ld.so.conf 文件进行 cat 操作，您可能会看到一个与下面类似的清单：

```
$ cat /etc/ld.so.conf
/usr/X11R6/lib
/usr/lib/gcc-lib/i686-pc-linux-gnu/2.95.3
/usr/lib/mozilla
/usr/lib/qt-x11-2.3.1/lib
/usr/local/lib
```

ld.so.conf 文件包含一个所有目录（/lib 和 /usr/lib 除外，它们会自动包含在其中）的清单，动态装入器将在其中查找共享库。

ld.so.cache

发布时间 :2007-01-27 19:53:29

但是在动态装入器能“看到”这一信息之前，必须将它转换到 ld.so.cache 文件中。可以通过运行 ldconfig 命令做到这一点：

```
# ldconfig
```

当 ldconfig 操作结束时，您会有一个最新的 /etc/ld.so.cache 文件，它反映您对 /etc/ld.so.conf 所做的更改。从这一刻起，动态装入器在寻找共享库时会查看您在 /etc/ld.so.conf 中指定的所有新目录。

ldconfig 技巧

发布时间 :2007-01-27 19:54:03

要查看 ldconfig 可以 “看到” 的所有共享库，请输入：

```
# ldconfig -p | less
```

还有另一个方便的技巧可以用来配置共享库路径。有时候您希望告诉动态装入器在尝试任何 /etc/ld.so.conf 路径以前先尝试使用特定目录中的共享库。在您运行的较旧的应用程序不能与当前安装的库版本一起工作的情况下，这会比较方便。

LD_LIBRARY_PATH

发布时间 :2007-01-27 19:54:39

要指示动态装入器首先检查某个目录，请将 LD_LIBRARY_PATH 变量设置成您希望搜索的目录。多个路径之间用逗号分隔；例如：

```
# export LD_LIBRARY_PATH="/usr/lib/old:/opt/lib"
```

导出 LD_LIBRARY_PATH 后，如有可能，所有从当前 shell 启动的可执行程序都将使用 /usr/lib/old 或 /opt/lib 中的库，如果仍不能满足一些共享库相关性要求，则转回到 /etc/ld.so.conf 中指定的库。

我们已经完成了对 Linux 共享库的介绍。要了解有关共享库的更多内容，请输入 man ldconfig 和 man ld.so。

从源代码编译应用程序

从源代码编译应用程序 - 介绍

发布时间 :2007-01-27 19:55:28

假设您发现了一个特定的应用程序并想将它安装在系统上。您可能需要运行这个程序的最新版本，但还没有这个最新版本的打包格式（如 rpm）。或许您只能得到这个特定应用程序的源代码格式，或者您需要启用缺省情况下在 rpm 中未启用的该程序的某些特性。

无论什么原因，不管您必须还是仅仅因为您想从源代码编译该程序，本节将向您演示如何做。

下载

发布时间 :2007-01-27 19:55:58

首先，要找到并下载您要编译的源代码。它们可能在以 .tar.gz、tar.Z、tar.bz2 或 .tgz 扩展名结尾的单个压缩文档中。继续进行，用您喜爱的浏览器或 ftp 程序下载这个压缩文档。如果这个程序碰巧有一个网页，那么最好访问该网页以熟悉所有可能有用的安装文档。

您正在安装的程序可能依赖于许多当前在系统上已安装或未安装的其它程序。如果您确切知道该程序依赖的其它程序或库当前没有安装在系统上，则您需要先安装这些软件包（从象 rpm 那样的二进制软件包安装或同样从它们的源代码进行编译）。这样，您将为成功安装原始源代码文件做好准备。

解包

发布时间 :2007-01-27 19:56:27

解包源压缩文档相对较简单。如果压缩文档名称以 .tar.gz、.tar.Z 或 .tgz 结尾，您应该可以通过输入以下内容来解包：

```
$ tar xzvf archivername.tar.gz
```

（x 用于解压缩，z 用于 gzip 解压，v 用于显示详细信息 — 打印解压缩的文件名，而 f 意味着文件名将接着在命令行上出现）。

几乎所有的“源代码 tar 包（tarball）”都将创建一个包含程序所有源代码的主目录。这样，当您解包这个压缩文档时，您的当前工作目录不会被大量的文件搞得乱七八糟 — 相反，所有的文件被整齐地组织在单个目录中，不会妨碍工作。

列出压缩文档

发布时间 :2007-01-27 19:57:13

时不时您会遇到这样的压缩文档，在对它解压缩时，会在您的当前工作目录中创建数量巨大的文件。尽管大多数 tar 包不是这样创建的，但这样的情况确有发生。如果您希望验证您的 tar 包被正确地组装在一起并会创建一个主目录来容纳源代码，您可以输入以下内容来查看其内容：

```
$ tar tzvf archivename.tar.gz | more
```

（t 用于显示压缩文档的文本清单。不进行解压缩）。

如果压缩文档清单的左边没有列出公共目录，则您要创建一个新的目录，将 tar 包移至该目录下，进入该目录，这时再解压缩这个 tar 包。不这样做的话，您会面对一团乱麻！

解包 bzip2 压缩的压缩文档

发布时间 :2007-01-27 19:57:47

您的压缩文档有可能是 .tar.bz2 格式。具有这种扩展名的压缩文档是用 bzip2 进行压缩的。bzip2 的压缩效果通常比 gzip 好得多。它唯一的不足之处是压缩和解压缩的速度较慢，并且在运行时，bzip2 比 gzip 消耗更多的内存。对于现代计算机，这不成问题，因此，可以预料 bzip2 会随着时间的推移变得越来越流行。

因为 bzip2 日益受到欢迎，所以许多 Linux 分发版（distribution）都带有经过补丁程序修正的 tar 版本，这样传递一个 y 或 i 选项将通知 tar：压缩文档是 bzip2 格式，需要用 bzip2 程序自动解压缩。要查看您是否带有用补丁修正过的 tar 版本，可以试着输入：

```
$ tar tyvf archive.tar.bz2 | more
```

或

```
$ tar tivf archive.tar.bz2 | more
```

即使两个命令都不起作用（而且 tar 提示参数无效），仍有办法 — 请继续阅读。

bzip2 管道

发布时间 :2007-01-27 19:58:24

哦，您的 tar 版本不能识别这些方便的 bzip2 快捷方式 — 该怎么办呢？所幸有一个简单的方法可以解压缩 bzip2 tar 包的内容，并且这个方法几乎可以在所有的 UNIX 系统上工作，即使正被讨论的系统碰巧有一个非 GNU 版本的 tar。要查看 bzip2 文件的内容，我们可以创建一个管道：

```
$ cat archive.tar.bz2 | bzip2 -d | tar tvf - | most
```

接下来的这个管道实际上将解压缩 archive.tar.bz2 的内容：

```
$ cat archive.tar.bz2 | bzip2 -d | tar xvf -
```

bzip2 管道（续）

发布时间 :2007-01-27 19:58:57

在前两个示例中，我们创建了一个标准的 UNIX 管道从压缩文档查看并解压缩文件。因为用了 f - 选项来调用 tar，因此它从标准输入（stdin）读取 tar 数据，而不是试图从磁盘上的文件读取数据。

如果尝试用管道方法解压缩您的压缩文档的内容，而系统提示找不到 bzip2，则系统可能没有安装 bzip2。可以从 <http://sourceware.cygnum.com/bzip2> 下载 bzip2 的源代码。安装 bzip2 源代码之后（通过遵照本教程），您才能首先解包并安装您希望安装的应用程序

检查源代码

发布时间 :2007-01-27 19:59:28

解包源代码之后，您可以进入解包的目录并检查其中的内容。最好是能找到所有与安装有关的文档。通常，这一信息可以在位于主源代码目录的 README 或 INSTALL 文件中找到。另外，可以查找 README.platform 和 INSTALL.platform 文件，这里的 platform 是您的特定操作系统名称。

配置

发布时间 :2007-01-27 19:59:55

现在，许多源代码在主源代码目录中包含配置脚本。这个脚本（通常由开发人员使用 GNU autoconf 程序生成）特别设计用来设置源代码以使它们能在您的系统上正确编译。这个配置脚本在运行时会探测您的系统以确定系统性能，然后创建制作文件（Makefile），其中包含在您的系统上构建和安装源代码的指令。

这个配置脚本几乎总是被命名为“configure”。如果您在主源代码目录中找到配置脚本，那么它可以供您使用。如果您没有发现配置脚本，那么您的源代码可能带有一个设计用来跨越不同系统工作的标准制作文件 — 这意味着您可以略过以下配置步骤，在我们开始讨论“make”处继续学习本教程。

使用配置

发布时间 :2007-01-27 20:00:28

在运行配置脚本之前，最好是先熟悉它。输入 `./configure --help`，您可以查看您的程序能够使用的所有不同配置选项。您所看到的选项，特别是在 `--help` 打印输出的顶部列出的那些项，都是几乎可以在每个配置脚本中找到的标准选项。在结尾部分列出的选项通常与您正尝试编译的特定软件包相关。查看它们并注意那些您希望启用或禁用的选项。

--prefix 选项

发布时间 :2007-01-27 20:00:57

大多数基于 GNU autoconf 的配置脚本都有 --prefix 选项来允许您控制程序的安装位置。缺省情况下，大多数源代码安装时都用 /usr/local 前缀。这意味着二进制文件最终在 /usr/local/bin 中，手册页则在 /usr/local/man 中，等等。这通常就是您所期望的； /usr/local 通常用于存储您自己编译的程序。

使用 --prefix

发布时间 :2007-01-27 20:01:27

如果您想把源代码安装在别的地方，假设安装在 `/usr` 中，则您要向配置脚本传递 `--prefix=/usr` 选项。同样地，您也可以用 `--prefix=/opt` 选项告诉配置脚本将源代码安装到 `/opt` 目录。

FHS 怎么样？

发布时间 :2007-01-27 20:02:01

有时候，一个特定程序可能缺省地将它的一些文件安装到磁盘上的非标准位置。特别地，一个源代码压缩文档可能有许多没有遵守 Linux 文件系统层次结构标准（FHS）的安装路径。幸运的是，配置脚本不仅允许更改安装前缀，而且允许我们更改各个系统部件（如手册页）的安装位置。

因为大多数源代码压缩文档还不符合 FHS，所以这种能力非常有用。为了使您的源代码包符合 FHS，您几乎总要向配置命令行添加 `--mandir=/usr/share/man` 和 `--infodir=/usr/share/info`。

该配置了

发布时间 :2007-01-27 20:02:36

一旦已经检查了各种配置选项并确定了要使用的选项，就可以运行配置脚本了。请注意您可能不需要在运行配置脚本时包含所有命令行选项 — 在大多数情形下，缺省值可以发挥作用（但可能不完全是您想要的）。

该配置了（续）

发布时间 :2007-01-27 20:03:07

要运行配置脚本，请输入：

```
$ ./configure <options>
```

这看起来象：

```
$ ./configure
```

或

```
$ ./configure --prefix=/usr --mandir=/usr/share/man --infodir=/usr/share/info --with-threads
```

您需要的选项取决于您正在配置的特定软件包。当您运行配置脚本时，它需要一两分钟来探测您系统上可用的特定功能部件或工具，并随着它工作的进行打印出各种配置检查的结果。

config.cache

发布时间 :2007-01-27 20:03:41

配置过程完成后，配置脚本将它所有的配置数据存储在一个名为 config.cache 的文件中。这个文件和配置脚本本身都在同一目录中。如果您在更新系统配置后需要再次运行 ./configure，请确保您先执行 rm config.cache 命令；否则配置脚本将只使用旧的设置而不重新检查系统。

配置脚本和制作文件

发布时间 :2007-01-27 20:04:13

配置脚本完成之后，就是将源代码编译为可运行程序的时候了。可以用名为 make 的程序来执行这一步骤。如果您的软件包含配置脚本，那么当您运行它时，配置脚本会创建一些为您的系统特别定制的制作文件。这些文件会告诉 make 程序如何构建源代码以及如何安装所产生的二进制文件、手册页和支持文件。

制作文件介绍

发布时间 :2007-01-27 20:04:43

Makefile 通常被称为 makefile 或 Makefile。每个含有源文件的目录通常都有一个制作文件，另外主源代码目录中还有一个制作文件。autoconf 生成的制作文件含有确定如何构建某些目标（如您希望安装的程序）的指示信息（正式名称是规则）。make 指出所有规则运行的顺序。

调用 make

发布时间 :2007-01-27 20:05:16

调用 make 比较简单；只需在当前目录中输入“make”即可。make 程序随即将在当前目录中查找并解释名为 makefile 或 Makefile 的文件。如果您只输入“make”本身，那么它将构建缺省目标。开发人员通常设置制作文件以使缺省目标编译所有源代码：

```
$ make
```

有些 makefile 没有缺省目标，因此您要指定一个以使编译得以开始：

```
$ make all
```

输入这些命令的其中之一后，您的计算机将用几分钟的时间将您的程序编译成目标代码。假设它不出差错地完成编译，您就可以准备将已编译程序安装到系统上。

安装

发布时间 :2007-01-27 20:05:46

编译好程序之后，还有一个更重要的步骤：安装。尽管程序已编译，但它仍未做好使用准备。您需要将它的所有组件从源代码目录复制到您的文件系统上适当的、“活动的”位置。例如，您需要将所有的二进制文件复制到 `/usr/local/bin`，将所有的手册页安装到 `/usr/local/man`，等等。

在安装软件以前，您要成为 root 用户。实现这一点通常有两种方式，在另一台终端以 root 用户登录，或者输入“su”，那时就会提示您输入 root 用户的密码。输入密码以后，您就将一直拥有 root 用户的特权，直到您输入“exit”或按 control-D 键组合从当前 shell 会话退出为止。如果您已经是 root 用户，那就可以进行下一步行动了！

make install

发布时间 :2007-01-27 20:06:16

安装源代码比较简单。只需在主源代码目录中输入：

```
# make install
```

输入“make install”告诉 make 要满足“install”目标；这个目标通常用来将所有新近创建的源文件复制到磁盘上的正确位置，以便可以使用您的程序。如果您没有指定 --prefix 选项，则很有可能有相当多的文件和目录将被复制到 /usr/local 目录。根据程序大小的不同，安装目标需要的时间可能从几秒到几分钟不等。

除了简单地复制文件以外，make install 还会确保安装的文件有正确的所有权和许可权。make install 成功完成后，程序就安装完毕并且可以使用（或几乎可以使用）。

安装之后

发布时间 :2007-01-27 20:06:50

现在您的程序已经安装完毕，那么下一步是什么？当然是运行它啦！如果您不熟悉如何使用刚刚安装的程序，则可以通过输入以下命令来阅读这个程序的手册页：

```
$ man programname
```

程序可能需要另外的配置步骤。例如，如果您安装了 Web 服务器，则您需要将它配置成在系统引导时自动启动。在应用程序运行以前，您可能还需要在 /etc 中定制一个配置文件。

好，完成了！

发布时间 :2007-01-27 20:07:21

既然您已经从源代码完整地安装了一个特定的软件包，现在可以运行它了！要启动这个程序，请输入：

\$ programname

祝贺您！

可能出现的问题

发布时间 :2007-01-27 20:07:49

configure 或 make (甚至可能是 make install) 很有可能会因某种错误代码而异常终止。以下几页将帮助您改正一些常见问题。

遗漏一些库

发布时间 :2007-01-27 20:08:21

您可能时常会有 configure 完全失败的经历，这往往是因为某个库没有安装。为了能使构建过程继续进行，您需要暂停当前的程序配置，搜寻源代码或二进制软件包以找到程序所需的库。当安装正确的库后，configure 或 make 应该能顺利运行并成功地完成。

其它问题

发布时间 :2007-01-27 20:08:51

有时候，您会碰到某种不知如何修复的错误。随着您的 UNIX / Linux 经验的增长，您将能够诊断越来越多在 configure 和 make 过程中遇到的、看似神秘的错误情况。

有时，错误的出现是因为安装的库太旧（甚至也可能是因为太新了！）。还有些时候，碰到的问题实际上是开发人员的过错，他们可能没有预计到他们的程序会在您这样的系统上运行 — 或者他们只是打字有误

其它问题（续）

发布时间 :2007-01-27 20:09:21

对于这样的问题，您要清楚能在哪里获得帮助。如果这是您首次尝试从源代码编译程序，那么最好还是选择另一个更简单的程序来编译。当您编译了较简单的程序后，您就会有修正最初遇到的问题所必需的经验。随着您继续学习更多有关 UNIX 如何工作的知识，您将更接近这样的境界：“微调”制作文件和源代码就可以让那些看上去甚至有些古怪的代码顺利地编译。

软件包管理概念

软件包管理概念

发布时间 :2007-01-27 20:10:09

除了从源代码构建应用程序以外，还有另一种在 Linux 系统上安装软件的方法。所有的 Linux 分发版都使用某种形式的软件包管理来安装、更新和卸载软件包。与直接从源代码安装相比，软件包管理有明显优势：

- 易于安装和卸载
- 易于更新已安装的软件包
- 保护配置文件
- 轻松跟踪已安装文件

软件包管理的缺点

发布时间 :2007-01-27 20:10:43

在讲解如何使用最流行的软件包管理工具以前，我得承认有些 Linux 用户不喜欢软件包管理。他们可能会提出软件包管理的以下缺点：

- 为特定系统构建的二进制文件性能更好
- 解决软件包相关性比较麻烦
- 软件包数据库的破坏会导致系统不可维护
- 创建软件包比较困难

这些说法确有其事，但 Linux 用户的一般看法是软件包管理的优势大于劣势。另外，上面列出的每个不利因素都有一个相应的反证：可以为不同的系统，构建多个优化的软件包；可以增强软件包管理器来自动解决相关性；可以基于其它文件重建数据库；而以后更新或除去这个软件包时的方便性可以弥补最初创建它时所做的努力。

rpm , Red Hat 软件包管理器

rpm , Red Hat 软件包管理器

发布时间 :2007-01-27 20:12:26

rpm , Red Hat 软件包管理器 ((R)ed Hat (P)ackage (M)anager)

rpm 入门

对 Linux 分发版而言，1995 年 Red Hat 引入 rpm 是一个巨大的进步。它不仅使 Red Hat Linux 上的软件包管理成为可能，而且因为它的 GPL 许可证，rpm 已经成为开放源代码打包的事实标准。

尽管有 GUI 和基于 Web 的工具提供更友好的界面，rpm 程序在缺省情况下是一个命令行界面。在这一节中，我们将介绍最常用的命令行操作，并且将 Xsnow 程序作为示例使用。如果您愿意按照下面的步骤执行，可以在下面下载 rpm，它应该可以用于大多数基于 rpm 的分发版。

xsnow-1.41-1.i386.rpm

注：如果这一节中“rpm”一词的不同使用让您有些困惑的话，请记住“rpm”通常指程序，而“一个rpm”或“那个rpm”则通常指一个rpm软件包。

安装 rpm

发布时间 :2007-01-27 20:13:01

首先让我们使用 rpm -i 来安装 Xsnow rpm :

```
# rpm -i xsnow-1.41-1.i386.rpm
```

如果这个命令没有产生输出，那么它就起作用了！您应该能够运行 Xsnow 在您的 X 桌面上享受一场暴风雪。我们本身是希望在安装一个 rpm 时有某些可视化反馈的，所以我们可以加上 -h（用 # 号表示进展）和 -v（详细信息）选项：

```
# rpm -ivh xsnow-1.41-1.i386.rpm
```

```
xsnow          #####
```

重新安装一个 rpm

发布时间 :2007-01-27 20:13:36

如果您直接按照步骤执行，您可能会在前一个示例中看到来自 rpm 的下列消息：

```
# rpm -ivh xsnow-1.41-1.i386.rpm
package xsnow-1.41-1 is already installed
```

在有些情况下，您可能希望重新安装一个 rpm，例如，您可能不小心删除了二进制文件 /usr/X11R6/bin/xsnow。在那种情况下，您应该首先用 `rpm -e` 除去那个 rpm，然后重新安装它。请注意：下面的示例中来自 rpm 的资料信息不妨碍从系统除去软件包。

```
# rpm -e xsnow
removal of /usr/X11R6/bin/xsnow failed: No such file or directory
```

```
# rpm -ivh xsnow-1.41-1.i386.rpm
xsnow          #####
```

强制安装一个 rpm

发布时间 :2007-01-27 20:14:10

有时除去一个 rpm 是不现实的，尤其是系统上有别的程序依赖于它的时候。例如，您可能已经安装了一个 “x-amusements ” rpm，它将 Xsnow 列为相关程序，因此用 rpm -e 除去 Xsnow 是不允许的：

```
# rpm -e xsnow
error: removing these packages would break dependencies:
      /usr/X11R6/bin/xsnow is needed by x-amusements-1.0-1
```

在这种情况下，您可以用 --force 选项重新安装 Xsnow：

```
# rpm -ivh --force xsnow-1.41-1.i386.rpm
xsnow      #####
```

用 --nodeps 安装或除去

发布时间 :2007-01-27 20:14:46

上页中使用 --force 的一个替代方法是用 --nodeps 选项除去 rpm。这一选项取消了 rpm 的内部相关性检查，所以在大多数情况下不推荐使用。然而，它偶尔会有用处：

```
# rpm -e --nodeps xsnow
```

```
# rpm -ivh xsnow-1.41-1.i386.rpm
xsnow      #####
```

在安装 rpm 时，您也可以使用 --nodeps。再次重申上面所说的，不推荐使用 --nodeps，但它有时是必需的：

```
# rpm -ivh --nodeps xsnow-1.41-1.i386.rpm
xsnow      #####
```

更新软件包

发布时间 :2007-01-27 20:15:19

最终可能将有 Xsnow 版本 1.42 的 rpm，它可以在 Xsnow 作者的网站上获得。这时，您将要更新现有的 Xsnow 安装。如果您要使用 rpm -ivh --force，似乎会奏效，但 rpm 的内部数据库会列出两个版本都已安装。相反，您应该用 rpm -U 来更新您的安装：

```
# rpm -Uvh xsnow-1.42-1.i386.rpm
xsnow          #####
```

这里有一个小窍门：我们几乎不使用 rpm -i，因为如果系统上不存在 rpm，则 rpm -U 会安装一个。如果您在命令行指定多个软件包，其中有些软件包当前已安装而有些则没有，那么这一选项会特别有用。

```
# rpm -Uvh xsnow-1.42-1.i386.rpm xfishtank-2.1tp-1.i386.rpm
xsnow          #####
xfishtank      #####
```

用 rpm -q 查询

发布时间 :2007-01-27 20:16:02

您可能已经在前几个示例中注意到：安装 rpm 需要完整的文件名，但除去 rpm 只需要文件的名称即可。这是因为 rpm 维护一个当前已安装软件包的内部数据库，所以您可以通过名称来引用安装的软件包。例如，让我们查询 rpm 安装的是什么版本的 Xsnow：

```
# rpm -q xsnow
xsnow-1.41-1
```

事实上，rpm 知道的有关已安装软件包的信息远不止是名称和版本。我们可以用 rpm -qi 查询有关 Xsnow rpm 的更多信息：

```
# rpm -qi xsnow
Name       : xsnow                Relocations: (not relocateable)
Version    : 1.41                Vendor: Dan E. Anderson http://www.dan.drydog.com/
Release    : 1                  Build Date: Thu 10 May 2001 01:12:26 AM EDT
Install date: Sat 02 Feb 2002 01:00:43 PM EST   Build Host: danx.drydog.com
Group      : Amusements/Graphics   Source RPM: xsnow-1.41-1.src.rpm
Size       : 91877                License: Copyright 1984, 1988, 1990, 1993-1995, 2000 by Rick Jansen.
Allows packaging & necessary changes for Unix/Linux distributions.
Packager   : Dan E. Anderson http://dan.drydog.com/
URL        : http://www.euronet.nl/~rja/Xsnow/
Summary    : An X Window System based dose of Christmas cheer.
Description:
The Xsnow toy provides a continual gentle snowfall, trees, and Santa
Claus flying his sleigh around the screen on the root window.
Xsnow is only for the X Window System, though; consoles just get coal.
```

用 rpm -ql 列出文件

发布时间 :2007-01-27 20:16:35

由 rpm 维护的数据库包含很多信息。我们已经看到它记录了已安装软件包的版本，以及它们的相关信息。还可以用 rpm -ql 列出给定的已安装软件包所拥有的文件：

```
# rpm -ql xsnow
/etc/X11/applnk/Games/xsnow.desktop
/usr/X11R6/bin/xsnow
/usr/X11R6/man/man1/xsnow.1x.gz
```

结合 -c 选项或 -d 选项，您可以将输出分别限制为配置文件或文档文件。这一类型的查询对有较长文件列表的较大 rpm 更有用，但我们仍然可以用 Xsnow rpm 来演示：

```
# rpm -qlc xsnow
/etc/X11/applnk/Games/xsnow.desktop
```

```
# rpm -qld xsnow
/usr/X11R6/man/man1/xsnow.1x.gz
```

用 rpm -qp 查询软件包

发布时间 :2007-01-27 20:17:11

如果您在安装软件包之前已经有了用 rpm -qi 获得的信息，您也许就能够更好地决定是否安装这个软件包。实际上，使用 rpm -qp 允许您查询一个 rpm 文件而不是查询数据库。到目前为止我们见到的所有查询命令都可以用于 rpm 文件和已安装的软件包。下面再次列出所有的示例，这次使用的是 -p 选项：

```
# rpm -qp xsnow-1.41-1.i386.rpm
xsnow-1.41-1
```

```
# rpm -qpi xsnow-1.41-1.i386.rpm
[与上页中的 rpm -qi 有相同的输出]
```

```
# rpm -qpl xsnow-1.41-1.i386.rpm
/etc/X11/applnk/Games/xsnow.desktop
/usr/X11R6/bin/xsnow
/usr/X11R6/man/man1/xsnow.1x.gz
```

```
# rpm -qplc xsnow-1.41-1.i386.rpm
/etc/X11/applnk/Games/xsnow.desktop
```

```
# rpm -qpId xsnow-1.41-1.i386.rpm
/usr/X11R6/man/man1/xsnow.1x.gz
```

查询所有已安装的软件包

发布时间 :2007-01-27 20:17:47

您可以加上 -a 选项来查询安装在系统上的所有软件包。如果您用管道命令将输出通过 sort 传入一个分页程序，则可一览系统上安装的软件包。例如：

```
# rpm -qa | sort | less  
[省略输出]
```

以下是在我们的一个系统上安装的 rpm 数目：

```
# rpm -qa | wc -l  
287
```

而以下是所有这些 rpm 中的文件总数：

```
# rpm -qal | wc -l  
45706
```

这里有一个简单窍门：使用 rpm -qa 可以简化多个系统的管理。如果您在一台机器上将排序的输出重定向至一个文件，然后在另一台机器上做同样操作，您可以用 diff 程序来观察二者的区别。

查找文件的所有者

发布时间 :2007-01-27 20:18:31

有时找出什么 rpm 拥有给定的文件是非常有用的。理论上，您可以用与下面类似的 shell 构造了解哪个 rpm 拥有 /usr/X11R6/bin/xsnow（假装您不记得了）：

```
# rpm -qa | while read p; do rpm -ql $p | grep -q '^/usr/X11R6/bin/xsnow$' && echo $p; done
xsnow-1.41-1
```

因为输入这些要很长的时间，而运行的时间更长（在我们的一台奔腾机上要 1 分 50 秒），rpm 开发人员很聪明地把这种功能包含进了 rpm。您可以用 rpm -qf 查询一个特定文件的所有者：

```
# rpm -qf /usr/X11R6/bin/xsnow
xsnow-1.41-1
```

即使在奔腾机上，这也只要 0.3 秒的时间运行。而且，与复杂的 shell 构造相比，连打字高手也会喜欢 rpm -qf 的简单。

显示相关性

发布时间 :2007-01-27 20:19:10

除非您使用如 `--nodeps` 那样的选项，rpm 通常不允许您安装或除去那些会破坏相关性的软件包。例如，您不能在缺少 X 库的系统上安装 Xsnow。安装 Xsnow 之后，您不能在缺少 Xsnow（而且可能还有半数您已安装的软件包）的情况下除去 X 库。

这就是 rpm 的强项，这一点有时甚至令人头疼。它意味着只要您安装了 rpm，它就应该正常工作。您无需做额外的工作，因为 rpm 已经验证了系统上存在相关性。

有时候，在您解决相关问题时，用 `-R` 选项查询一个软件包有助于了解这个软件包在系统上被期望具有的所有情况。例如，Xsnow 软件包依赖 C 库、math 库、X 库和 rpm 的特定版本：

```
# rpm -qpR xsnow-1.41-1.i386.rpm
rpmlib(PayloadFilesHavePrefix) <= 4.0-1
ld-linux.so.2
libX11.so.6
libXext.so.6
libXpm.so.4
libc.so.6
libm.so.6
libc.so.6(GLIBC_2.0)
libc.so.6(GLIBC_2.1.3)
rpmlib(CompressedFileNames) <= 3.0.4-1
```

您也可以省略 `-p` 来向已安装的数据库查询同样的信息：

```
# rpm -qR xsnow
```

验证软件包的完整性

发布时间 :2007-01-27 20:19:51

当您从 Web 或 ftp 站点下载 rpm 时，出于安全性考虑，您可能希望在安装该软件包之前，先验证它的完整性。所有的 rpm 都已使用 MD5 校验和“签名”。另外，有些作者使用 PGP 或 GPG 签名进一步加强他们的软件包的安全性。要查看软件包的签名，您可以使用 --checksig 选项：

```
# rpm --checksig xsnow-1.41-1.i386.rpm
xsnow-1.41-1.i386.rpm: md5 GPG NOT OK
```

稍等片刻！根据那个输出，GPG 签名为 NOT OK。让我们加一些详细说明看看出了什么问题：

```
# rpm --checksig -v xsnow-1.41-1.i386.rpm
xsnow-1.41-1.i386.rpm:
MD5 sum OK: 8ebe63b1dbe86ccd9eaf736a7aa56fd8
gpg: Signature made Thu 10 May 2001 01:16:27 AM EDT using DSA key ID B1F6E46C
gpg: Can't check signature: public key not found
```

那么，问题出在我们不能获取作者的公钥。在我们从软件包作者的网站获得公钥后，（如来自 rpm -qi 的输出所示），签名检出如下：

```
# gpg --import dan.asc
gpg: key B1F6E46C: public key imported
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: Total number processed: 1
gpg:             imported: 1
```

```
# rpm --checksig xsnow-1.41-1.i386.rpm
xsnow-1.41-1.i386.rpm: md5 gpg OK
```

验证已安装的软件包

发布时间 :2007-01-27 20:20:31

与检查 rpm 的完整性相似，您也可以用 rpm -V 检查已安装文件的完整性。这一步骤确保文件从 rpm 安装以后没有再被修改。

```
# rpm -V xsnow
```

通常这一命令没有输出，这表明文件正常。让我们把事情弄得更有趣味，然后再试一次：

```
# rm /usr/X11R6/man/man1/xsnow.1x.gz
```

```
# cp /bin/sh /usr/X11R6/bin/xsnow
```

```
# rpm -V xsnow
S.5...T  /usr/X11R6/bin/xsnow
missing  /usr/X11R6/man/man1/xsnow.1x.gz
```

这个输出向我们显示 Xsnow 二进制文件的 MD5 校验和、文件大小和 mtime 测试失效。而且手册页全部丢失！让我们修复这个损坏的安装：

```
# rpm -e xsnow
removal of /usr/X11R6/man/man1/xsnow.1x.gz failed: No such file or directory
```

```
# rpm -ivh xsnow-1.41-1.i386.rpm
xsnow          #####
```

配置 rpm

发布时间 :2007-01-27 20:21:01

rpm 很少需要配置。它运行良好。在较旧的 rpm 版本中，您可以在 /etc/rpmrc 中做更改来影响运行时操作。在最近的版本中，这个文件被移到 /usr/lib/rpm/rpmrc，并且不打算让系统管理员编辑它。它通常只列出各个平台的标志和兼容性信息（例如，i386 与所有其它 x86 体系结构兼容）。

如果您希望配置 rpm，可以通过编辑 /etc/rpm/macros 实现。因为这很少有必要，我们将让您在 rpm 捆绑的文档中阅读它。您可以用下列命令找到合适的文档：

```
# rpm -qld rpm | grep macros
```

Debian 软件包管理

Debian 软件包管理 - 介绍 apt-get

发布时间 :2007-01-27 20:21:52

Debian 软件包管理系统由几个不同的工具组成。命令行工具 apt-get 是安装新软件包的最简单方法。例如，要安装程序 Xsnow，可以 root 用户的身份执行以下操作：

```
# apt-get install xsnow
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
  xsnow
0 packages upgraded, 1 newly installed, 0 to remove and 10 not upgraded.
Need to get 17.8kB of archives. After unpacking 42.0kB will be used.
Get:1 http://ftp-mirror.internap.com stable/non-free xsnow 1.40-6 [17.8kB]
Fetched 17.8kB in 0s (18.4kB/s)
Selecting previously deselected package xsnow.
(Reading database ... 5702 files and directories currently installed.)
Unpacking xsnow (from .../archives/xsnow_1.40-6_i386.deb) ...
Setting up xsnow (1.40-6) ...
```

浏览这个输出，您可以看到即将安装 Xsnow，然后从 Web 上获取它，解包，最后设置。

模拟安装

发布时间 :2007-01-27 20:22:30

如果 apt-get 发觉您要安装的软件包依赖其它一些软件包，它会自动获取并安装这些软件包。在上一个示例中，只安装了 Xsnow，这是因为它的所有相关性都已满足。

然而，有时候 apt-get 需要获取的软件包列表会很大，所以一般最好在安装前先查看将要安装的文件。-s 选项正好做这一工作。例如，在我们的一个系统上，如果我们试图安装图形电子邮件程序 balsa：

```
# apt-get -s install balsa
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  esound esound-common gdk-imlib1 gnome-bin gnome-libs-data imlib-base libart2
  libaudiofile0 libesd0 libglib1.2 libgnome32 libgnomesupport0 libgnomeui32
  libgnorba27 libgnorbagtk0 libgtk1.2 libjpeg62 liborbit0 libpng2 libproplist0
  libtiff3g libungif3g zlib1g
The following NEW packages will be installed:
  balsa esound esound-common gdk-imlib1 gnome-bin gnome-libs-data imlib-base
  libart2 libaudiofile0 libesd0 libglib1.2 libgnome32 libgnomesupport0
  libgnomeui32 libgnorba27 libgnorbagtk0 libgtk1.2 libjpeg62 liborbit0 libpng2
  libproplist0 libtiff3g libungif3g zlib1g
0 packages upgraded, 24 newly installed, 0 to remove and 10 not upgraded.
```

然后它会列出软件包将安装和配置（或设置）的顺序。

软件包资源列表：**apt-setup**

发布时间：2007-01-27 20:23:03

因为 apt-get 会自动为您获取软件包，所以它必须知道在哪里能找到那些还没有安装的软件包。这一信息在 /etc/apt/sources.list 中。尽管您可以手工编辑这个文件（请参阅 sources.list 手册页），但您会发现使用一个交互工具会更方便：

```
# apt-setup
```

这个工具遍历可以找到 Debian 软件包的位置，如 CDROM、Web 站点和 ftp 站点。当您完成后，它将更改写到您的 /etc/apt/sources.list 文件，这样 apt-get 可以在您请求这些软件包时找到它们。

从 apt-get 到 dselect

发布时间 :2007-01-27 20:23:36

apt-get 工具有许多命令行选项，您可以在 apt-get 手册页查到。缺省项通常工作得很好，如果您发现您在频繁地使用同一个选项，您可以向 /etc/apt/apt.conf 文件添加一个设置。配置文件的这一语法在 apt.conf 手册页中有描述。

除了我们一直使用的 install 命令以外，apt-get 还有许多其它命令。其中之一是 apt-get dselect-upgrade，它遵守为 Debian 系统上的每个软件包所设置的状态（status）。

启动 dselect

发布时间 :2007-01-27 20:24:16

每个软件包的状态都存储在文件 /var/lib/dpkg/status 中，但用另一个交互工具更新它效果最好。

```
# dselect
```

Debian GNU/Linux `dselect' package handling frontend.

- * 0. [A]ccess Choose the access method to use.
- 1. [U]pdate Update list of available packages, if possible.
- 2. [S]elect Request which packages you want on your system.
- 3. [I]nstall Install and upgrade wanted packages.
- 4. [C]onfig Configure any packages that are unconfigured.
- 5. [R]emove Remove unwanted software.
- 6. [Q]uit Quit dselect.

Move around with ^P and ^N, cursor keys, initial letters, or digits;
Press <enter> to confirm selection. ^L redraws screen.

Version 1.6.15 (i386). Copyright (C) 1994-1996 Ian Jackson. This is
free software; see the GNU General Public License version 2 or later for
copying conditions. There is NO warranty. See dselect --license for details.

使用 dselect 的 Select 方式

发布时间 :2007-01-27 20:24:51

选择 Select 选项，您就可以查看和更改软件包的状态。它随即会显示一页帮助信息。读完后就按空格键。现在您可以看到与下面类似的软件包列表：

EIOM	Pri	Section	Package	Inst.ver	Avail.ver	Descr	ption
			All packages				
			Newly available packages				
			New Important packages				
			New Important packages in section admin				
n*	Imp	admin	at	<none>	3.1.8-10	Delayed job execution and	
n*	Imp	admin	cron	<none>	3.0pl1-57.3	management of regular bac	
n*	Imp	admin	logrotate	<none>	3.2-11	Log rotation utility	
			New Important packages in section doc				
n*	Imp	doc	info	<none>	4.0-4	Standalone GNU Info docum	
n*	Imp	doc	manpages	<none>	1.29-2	Man pages about using a L	
			New Important packages in section editors				
n*	Imp	editors	ed	<none>	0.2-18.1	The classic unix line edi	
n*	Imp	editors	nvi	<none>	1.79-16a.1	4.4BSD re-implementation	
			New Important packages in section interpreters				
n*	Imp	interpre	perl-5.005	<none>	5.005.03-7.	Larry Wall's Practical Ex	
			New Important packages in section libs				
n*	Imp	libs	libident	<none>	0.22-2	simple RFC1413 client lib	
n*	Imp	libs	libopenldap-	<none>	1.2.12-1	OpenLDAP runtime files fo	
n*	Imp	libs	libopenldap1	<none>	1.2.12-1	OpenLDAP libraries.	
n*	Imp	libs	libpcre2	<none>	2.08-1	Philip Hazel's Perl Compa	

软件包状态

发布时间 :2007-01-27 20:25:20

每个软件包的状态都可以在略带神秘的标题 EIOM 下看到。我们关心的列在 M 字符下，其中每个软件包都用下列标记之一来进行标记：

要更改标记，只要按住代表您想要的代码的键（等号、破折号或下划线）即可，但如果您想把标记改为 *（星号），则必须按 +（加号）。

当您完成操作后，用一个大写的 Q 来保存您的更改并退出 Select 屏幕。任何时候，如果您在 dselect 中需要帮助，可以输入？（问号）。输入空格则从帮助屏幕返回。

安装和配置（ dpkg-reconfigure ）

发布时间 :2007-01-27 20:26:04

除非您运行象 `apt-get dselect-upgrade` 这样的命令，否则 Debian 不会基于状态设置来安装或除去软件包。这个命令实际上立刻为您完成了几个步骤 — 安装、除去和配置。安装和除去步骤不需要停下来询问您任何问题。然而，配置步骤为了按您的希望来设置软件包，可能会问任意多个问题。

要运行这些步骤还有其它方法。例如，您可以从主 `dselect` 菜单单独选择每一步骤。

有些软件包使用名为 `debconf` 的系统用于其配置步骤。那些使用 `debconf` 的软件包可以用各种方式询问其配置问题，询问方式可以是：在文本终端中、通过图形界面或通过 Web 页面等。要配置一个这样的软件包，可以使用 `dpkg-reconfigure` 命令。您甚至可以用它来确保所有的 `debconf` 软件包都已完全配置。

```
# dpkg-reconfigure --all
debconf: package "3c5x9utils" is not installed or does not use debconf
debconf: package "3dchess" is not installed or does not use debconf
debconf: package "9menu" is not installed or does not use debconf
debconf: package "9wm" is not installed or does not use debconf
debconf: package "a2ps" is not installed or does not use debconf
debconf: package "a2ps-perl-ja" is not installed or does not use debconf
debconf: package "aalib-bin" is not installed or does not use debconf
```

这会产生一个很长的、未使用 `debconf` 的软件包列表，但它也会找到使用 `debconf` 的软件包，并且以易于使用的格式让您回答每个软件包提出的问题。

获取已安装软件包的状态

发布时间 :2007-01-27 20:26:41

目前我们所评论 Debian 软件包管理工具最适于处理有较长软件包列表的多步操作。但它们不包括软件包管理的某些具体操作。对于这一类工作，您可以使用 dpkg。

例如，要获得软件包的完整状态和描述，可以使用 -s 选项：

```
# dpkg -s xsnow
Package: xsnow
Status: install ok installed
Priority: optional
Section: non-free/x11
Installed-Size: 41
Maintainer: Martin Schulze <joey@debian.org>
Version: 1.40-6
Depends: libc6, xlib6g (>= 3.3-5)
Description: Brings Christmas to your desktop
Xsnow is the X-windows application that will let it snow on the
root window, in between and on windows. Santa and his reindeer
will complete your festive-season feeling.
```

文件与其 .deb 之间的链接

发布时间 :2007-01-27 20:27:20

因为 .deb 软件包含有文件，所以您可能会想到应该有什么办法列出软件包内的文件。哦，没错；只要用 -L 选项：

```
# dpkg -L xsnow
/.
/usr
/usr/doc
/usr/doc/xsnow
/usr/doc/xsnow/copyright
/usr/doc/xsnow/readme.gz
/usr/doc/xsnow/changelog.Debian.gz
/usr/X11R6
/usr/X11R6/bin
/usr/X11R6/bin/xsnow
/usr/X11R6/man
/usr/X11R6/man/man6
/usr/X11R6/man/man6/xsnow.6.gz
```

要用其它方法，并且要找出哪个软件包含有一个特定的文件，可以使用 -S 选项：

```
# dpkg -S /usr/doc/xsnow/copyright
xsnow: /usr/doc/xsnow/copyright
```

软件包的名称刚好在冒号的左边。

查找要安装的软件包

发布时间 :2007-01-27 20:27:53

通常，apt-get 已经知道您可能需要的所有 Debian 软件包。如果它不知道，您或许能在这些 Debian 软件包列表中（或 Web 上的其它地方）找到该软件包。

如果您已经找到并下载了一个 .deb 文件，您可以用 -i 选项安装它：

```
# dpkg -d /tmp/dl/xsnow_1.40-6_i386.deb
```

如果您找不到希望看到的 .deb 文件，但却发现了 .rpm 文件或一些其它类型的软件包，您或许可以使用 alien。
alien 程序可以将软件包从各种格式转换成 .deb。

Linux海量文章

海量 Linux 技术文章

发布时间 :2006-11-24 16:50:29

下面是linux技术文章快速入口。需要联网：

[Linux 技术交流](#)

<http://www.linuxdiyf.com/bbs/forum-3-1.html>

[Linux 应用](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=1>

[Linux 安装及学习指导](#)

<http://www.linuxdiyf.com/bbs/forum-45-1.html>

[Linux 系统安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=11>

[Linux 学习指导](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=12>

[Linux 软件安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=13>

[shell](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=3>

[Linux 壁纸](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=4>

[红旗](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=5>

[Redhat](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=6>

[SuSE](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=7>

Linux 认证

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=9>

Linux下载分享{酷件、书籍、视频分享 }

<http://www.linuxdiyf.com/bbs/forum-6-1.html>

服务器应用

<http://www.linuxdiyf.com/bbs/forum-7-1.html>

数据库应用

<http://www.linuxdiyf.com/bbs/forum-8-1.html>

Linux 编程与内核

<http://www.linuxdiyf.com/bbs/forum-9-1.html>

UniX 技术文章

<http://www.linuxdiyf.com/bbs/forum-32-1.html>

Linux 业界声音、新闻

<http://www.linuxdiyf.com/bbs/forum-11-1.html>

Linux 人才招聘信息

<http://www.linuxdiyf.com/bbs/forum-46-1.html>

网络转载，感谢原创作者！

制作：红联Linux论坛

祝您阅读愉快！

Linux培训系列

第六讲

将向您演示如何从源代码编译 Linux 内核。在演示过程中，我们将讨论各种重要的内核配置选项，更深入地介绍有关内核中 PCI 和 USB 支持的信息。在本系列教程（共 8 篇；本文是第 6 篇）结束时，您将具备成为 Linux 系统管理员所必需的知识。

红联Linux论坛是致力于Linux技术讨论的站点，目前网站收录的文章及教程基本能满足不同水平的朋友学习。

红联Linux门户：www.linux110.com

红联Linux论坛：www.linuxdiyf.com/bbs

下载Linux电子书籍：

<http://www.linux286.com/linux/linuxdzsj.htm>

目录

介绍内核

- 介绍内核
- 与硬件进行相互操作
- CPU 抽象
- 抽象 IO
- 以网络为中心
- 联网的优点
- 引导回顾
- 介绍 ... 模块！
- 模块所在位置
- 模块 — 并非用于所有进程！

查找并下载源代码

- 查找并下载源代码 - 内核版本历史
- 使用哪些内核源代码
- 从源代码获得内核
- 解包内核

配置内核

- 配置内核
- 新的配置方法
- 配置技巧
- 代码成熟级别选项
- 模块以及与 CPU 相关的选项
- 常规和并行端口选项
- RAID 和 LVM
- 联网和相关设备
- IDE 支持
- SCSI 支持
- 各种字符设备
- 文件系统和控制台驱动程序

编译和安装内核

- 编译和安装内核 - make dep
- make bzImage
- 编译模块

引导配置

- 引导配置 - LILO 简介
- 配置 LILO
- LILO 代码
- 调整 LILO
- 最终的 lilo.conf
- LILO 配置的原因

PCI 设备

- PCI 设备
- 检查 PCI 设备
- PCI 设备资源

Linux USB

- Linux USB
- 启用 USB
- UHCI、OHCI 和 EHCI — 天啊！

最后几步

挂装 usbdevfs

海量Linux文章

海量Linux技术文章

介绍内核

介绍内核

发布时间 :2007-01-27 21:43:29

通常，“Linux”一词指的是完整的Linux分发版（distribution）和使分发版工作的所有协作运行程序。然而，您或许会惊奇地发现从技术角度讲Linux是并且只是一个内核。尽管一般所指的“Linux”的其它部分（如shell和编译器）是完整的操作环境的基本组成部分，但从技术角度讲，这些部分与Linux（内核）是分开的。尽管如此，人们仍用“Linux”来指“基于Linux的分发版”。不过，至少每个人都认同Linux内核是所有“Linux操作系统”的心脏。

与硬件进行相互操作

发布时间 :2007-01-27 21:43:57

Linux 内核的主要作用是直接与系统中的硬件进行相互操作。内核在原始硬件与应用程序之间提供了一个抽象层。例如，按此方式，程序本身无需知道特定主板芯片组或磁盘控制器的详细信息 — 而是可以在向磁盘读写文件的更高层次上进行操作。

CPU 抽象

发布时间 :2007-01-27 21:44:27

Linux 内核还在系统的处理器（或多处理器）之上提供一个抽象层次 — 使多个程序看起来象是同时运行。内核负责使每个进程公平、分时共享处理器的计算资源。

若正在运行 Linux，那么正在使用的内核不是支持 UP（单处理器）的，就是支持 SMP（对称多处理器）的。如果您正好有一块 SMP 主板，但使用的是 UP 内核，那么 Linux 将不会“看见”额外的处理器！修正这一问题需要编译用于您硬件的特殊 SMP 内核。目前，SMP 内核也可运行在单处理器系统上，但性能略有下降。

抽象 IO

发布时间 :2007-01-27 21:44:55

内核还处理大量需要的任务：抽象所有文件输入输出格式。设想一下，如果每个程序都必须直接与特定的磁盘硬件打交道，会发生什么 — 如果您更换磁盘控制器，所有的程序都会停止运行！幸运的是，Linux 内核遵循 UNIX 模型：提供所有程序都可以使用的简单数据存储和访问抽象。这样的话，您喜欢的数据库就不需要考虑数据是存储在 IDE 硬盘上、还是存储在 SCSI RAID 阵列上或是存储在挂装于网络的文件系统上。

以网络为中心

发布时间 :2007-01-27 21:45:25

Linux 赢得声誉的主要因素之一是其健壮的联网功能，尤其是 TCP/IP 支持。如果您猜想 TCP/IP 栈在 Linux 内核中，那么您猜对了！内核为在网络上发送数据的程序提供了符合标准的高层次接口。在幕后，Linux 内核直接与特定的以太网卡或 pppd 守护程序进行相互操作，并处理低层次的因特网通信详细信息。请注意：本系列的下一篇文章（第 7 部分）将讨论 TCP/IP 和联网。

联网的优点

发布时间 :2007-01-27 21:45:55

Linux 的优点之一是内核中有大量可用的可选特性，特别是与联网有关的特性。例如，可以将内核配置为允许整个内部网络经过 Linux 调制解调器访问因特网 — 这称为 IP 伪装（IP Masquerading）或 IP NAT。

此外，可以将 Linux 内核配置为导出或挂装基于网络的 NFS 文件系统，这是考虑到允许 LAN 上其它 UNIX 机器可以轻松地与 Linux 系统共享数据。内核中有许多好东西，一旦开始研究 Linux 内核的众多配置选项，您就会了解。

引导回顾

发布时间 :2007-01-27 21:46:24

现在是快速复习 Linux 引导过程的良好时机。启动基于 Linux 的系统时，内核映象（以单个二进制文件形式存储）由引导装入程序（如 LILO 或 GRUB）从磁盘装入内存。此时，内核接管系统。内核首先做的事情之一是检测并初始化所有它找到且配置成支持的硬件。硬件正确初始化之后，内核就准备启动常规用户空间程序（也称为“进程”）。

由内核运行的第一个进程是 /sbin/init。它依次启动 /etc/inittab 中指定的其它进程。几秒钟内，Linux 系统就启动并运行起来，准备好供您使用。虽然您从不直接与内核打交道，但 Linux 内核始终运行于所有常规进程“之上”，并提供各种程序和库运行所必需的虚拟与抽象。

介绍 ... 模块！

发布时间 :2007-01-27 21:46:52

所有较新的 Linux 内核都支持内核模块。内核模块是真正美妙的事物 — 它们是内核的组成部分，驻留在磁盘上相对较小的二进制文件中。每当内核需要一个特定模块的功能时，就会从磁盘装入这个特定模块并自动将它与自身结合，这样便动态扩展了内核的能力。

如果装入的内核模块的特性在几分钟内未被使用，则内核会自动将它与内核其余部分分离并从内存卸装它 — 这被称为自动清除（autocleaning）。若没有内核模块，则需要确保运行的内核（作为单个二进制文件存在于磁盘上）完全包含所有可能需要的功能。若没有模块，则需要构建一个全新内核，以将新的重要功能添加到其中。

通常，用户构建一个包含所有基本功能的内核映象，然后再构建一组模块，这些模块对应着用户未来可能需要的功能。以后要使用时，则按照需要，将适当的模块装入内核。这也有助于节约 RAM，因为模块只有从磁盘装入后才使用 RAM。当从内核除去模块后，其使用的内存就被释放并用于其它用途。

模块所在位置

发布时间 :2007-01-27 21:47:30

内核模块通常位于 /lib/modules/x.y.z (其中 x.y.z 是模块所兼容的内核版本) ；每个模块在其名称末尾都有 “.o” ，表明它是包含机器指令的二进制文件。正如您猜想的那样，每个单独的模块代表内核功能的一个特殊组

D 擊瞿？榭贍芴崙 FAT 文件系统支持，而另一个模块则可能支持一块特殊的 ISA 以太网卡。

模块 — 并非用于所有进程！

发布时间 :2007-01-27 21:48:02

值得一提的是：不能将所有东西都放入模块中。因为模块存储于磁盘上，可引导的内核映象需要拥有对磁盘控制器、驱动器和根文件系统的内编译（compiled-in）支持。如果没有将这些基本组件编译到内核映象中 — 也就是说，如果试图将它们作为模块编译 — 那么内核将没有从磁盘装入这些模块所必需的能力，这会产生一个很令人讨厌的“先有鸡还是先有蛋”的问题，结果是一个不能引导系统的内核！

查找并下载源代码

查找并下载源代码 - 内核版本历史

发布时间 :2007-01-27 21:49:45

目前最新的内核是2.6.20。也是较稳定的2.6内核系列。

可用的内核是 2.4.18。2.4.18 内核是 2.4 稳定内核系列的一部分。这一系列的内核发行版打算用于生产系统。

还有几个 2.5 系列内核可供使用，但不应当在生产系统上使用它们。“2.5”中的“5”是奇数，表示这些内核本质上是实验性的，并且打算供内核开发人员使用。当“2.5”内核准备好用于生产使用时，就会出现“2.6”（第二个数为偶数）系列。

使用哪些内核源代码

发布时间 :2007-01-27 21:50:13

若只想编译当前已安装内核的新版本（例如，要启用 SMP 支持），则最好的方法是安装分发版的内核源代码包。这样做之后，您应该在 /usr/src/linux 中发现一组新文件。

然而，有时候您想安装新内核。通常，最好的方法是只安装分发版内核源代码包的新版本或更新版本。该包将包含内核源代码，这些内核源代码经过补丁程序的修正，并且经过调整以便能最优地运行在 Linux 系统上。

从源代码获得内核

发布时间 :2007-01-27 21:50:44

如果您具有冒险精神，则可以从 <http://www.kernel.org/pub/linux/kernel> 抓取一个“主线”内核源代码 tar 包。在这个目录中，您可以找到由 Linus 或 Marcelo 发布的正式内核源代码。这些源代码可能没有分发版内核源代码包中的所有特性，因此，在知道自己正在做什么以前 ... 或在有“额外的”机器和大量空闲时间以前，通常最好不要使用主线内核:)

在 kernel.org，您会发现内核源代码根据内核版本（v2.2、v2.4 等）被组织到几个不同的目录中。在每个目录中，您会发现文件被标为“linux-x.y.z.tar.gz”和“linux-x.y.z.tar.bz2”。这些文件是 Linux 内核源代码 tar 包。您还会看到标为“patch-x.y.z.gz”和“patch-x.y.z.bz2”的文件。这些文件是补丁程序，可以用来更新完整内核源代码的较早版本。如果要编译新的内核发行版，则需下载这些“linux”文件之一。

解包内核

发布时间 :2007-01-27 21:51:16

如果您从 kernel.org 下载了新内核，则现在可以将它解包。要这么做，用 cd 命令进入 /usr/src。如果在那里有一个现成的“linux”目录，则将它改成“linux.old”（以 root 用户身份执行 mv linux linux.old）。

现在，可以解压缩新内核了。若仍处在 /usr/src 中，则输入 tar xzvf /path/to/my/kernel-x.y.z.tar.gz 或 cat /path/to/my/kernel-x.y.z.tar.bz2 | bzip2 -d | tar xvf -，这取决于源代码是用 gzip 还是用 bzip2 压缩的。输入这一命令后，新的内核源代码将被解压缩到新的“linux”目录中。注意 — 完整的内核源代码通常占据 50 MB 以上的磁盘空间！

配置内核

配置内核

发布时间 :2007-01-27 21:51:50

编译内核以前，需要配置它。可以用配置来精确控制在新内核中启用（以及禁用）哪些内核特性。还可以控制将哪些部分编译到内核二进制映像（将在引导时装入）中，而将哪些部分编译到“按需装入”的内核模块文件中。

老式的配置内核的方法是非常令人痛苦的，包括进入 /usr/src/linux 并输入 make config。尽管 make config 仍然有效，但请不要尝试用这种方法配置内核 — 除非您喜欢在命令行回答数百个（是的，数百个！）yes/no 问题。

新的配置方法

发布时间 :2007-01-27 21:52:23

不用再输入“make config”，新的方法是输入“make menuconfig”或“make xconfig”来配置内核。如果输入“make menuconfig”，则会得到一个好看的基于控制台的彩色菜单系统，可以用它配置内核。如果输入“make xconfig”，则会得到一个非常好看的基于 X 的 GUI，可以用它配置各种内核选项。

使用“make menuconfig”时，左边有“<>”的选项可以被编译为模块。当选项被突出显示时，按空格键在取消该选项（“<>”）、选择将该选项编译到内核映象中（“<*”）和选择编译为模块（“<M>”）三者间切换。也可以按“y”键启用选项，按“n”禁用该选项或按“m”键以在可能的情况下将它编译为模块。幸运的是，大多数内核配置选项都有详细的帮助信息，可以通过输入 h 查看这些信息。

配置技巧

发布时间 :2007-01-27 21:52:54

遗憾的是，内核配置选项太多了，根本没有足够篇幅在这里全部介绍它们（不过，如果有兴趣，可以查看 [options\(4\)](#) 手册页以获取更完整的选项清单）。

在下列几页中，将对执行“`make menuconfig`”或“`make xconfig`”命令时出现的重要类别进行概述，并同时指出基本的或重要的内核配置选项。

代码成熟级别选项

发布时间 :2007-01-27 21:53:22

现在，我们来看看不同的内核配置选项类别。下面简略概述了每个类别。我鼓励您通过在 `/usr/src/linux` 中输入 “`make menuconfig`” 或 “`make xconfig`” 来加深对这些类别的理解。

Code maturity level options：该配置类别包含单一选项：“Prompt for development and/or incomplete code/drivers”。如果启用这个选项，那么许多被认为还处在实验阶段的选项（如 ReiserFS、devfs 和其它选项）将在其它类别菜单下可见。如果不选中这个选项，可见的选项将只是那些被认为是“稳定的”选项。通常，启用这个选项是个好主意，这样可以看到内核必须提供的所有功能。

模块以及与 CPU 相关的选项

发布时间 :2007-01-27 21:53:51

Loadable module support：该配置类别下是三个与内核对模块的支持相关的选项。通常，应该启用所有这三项。

Processor type and features：这一部分包含各种特定于 CPU 的配置选项。“Symmetric multiprocessing support option”特别重要，如果系统有一个以上的 CPU，则应启用该选项。否则，只能利用系统中的第一个 CPU。通常应该启用“MTRR Support”选项，因为它可以在现代系统上的 X 中产生更好的性能。

常规和并行端口选项

发布时间 :2007-01-27 21:54:18

General setup：在这一节中，通常应启用联网和 PCI 支持选项，还应启用“Kernel support for ELF binaries”（将它构建到内核中，而不是构建为模块）。建议启用 a.out 和 MISC 二进制选项，不过将它们构建为内核模块通常更有意义。还要确保启用“System V IPC”和“Sysctl support”。请参阅内置帮助，以获取有关这些选项的更多信息。

Parallel port support 选项：拥有并行端口设备（包括打印机）的人都会对 Parallel port support 节感兴趣。请注意：为了获得完整的打印机支持，除了启用此处适当的并行端口支持以外，还必须启用“Character devices”节下的“Parallel printer support”。

RAID 和 LVM

发布时间 :2007-01-27 21:54:46

Multi-device support (RAID and LVM)：这包含与 Linux 软件 RAID 和逻辑卷管理有关的选项。软件 RAID 允许以冗余方式使用磁盘以提高可用性。您可以在 developerWorks 软件 RAID 系列中找到有关软件 RAID 的更多信息（请参阅本教程最后一节“参考资料”，获取相关链接）。

联网和相关设备

发布时间 :2007-01-27 21:55:13

Networking options：您当然会猜到这包含与联网有关的选项！如果计划将 Linux 系统连接至一个典型网络，则应确保启用“Packet socket”、“Unix domain sockets”和“TCP/IP networking”。您可能会对各种其它选项感兴趣，其中包括“Network packet filtering”，它允许您使用 iptables 命令设置自己的有状态防火墙。有关这一操作的信息，请参阅 developerWorks 教程 Linux 2.4 stateful firewall design。

Network device support：使 Linux 联网正常工作的第二个要求是将对特别的联网硬件的支持编译进来。应选择对您希望内核支持的网卡（或多块网卡）的支持。需要的选项最有可能位于“Ethernet (10 or 100Mbit)”子类别下。

IDE 支持

发布时间 :2007-01-27 21:55:43

ATA/IDE/MFM/RLL support：对使用 IDE 驱动器、CD-ROM、DVD-ROM 和其它外围设备的系统而言，这一节包含重要的选项。如果系统有 IDE 磁盘，则请确保启用“Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support”、“Include IDE/ATA-2 DISK support”以及适合特殊主板的芯片组（构建到内核中，而不是构建为模块——这样系统才能引导！）。如果有 IDE CD-ROM，则请确保还启用了“Include IDE/ATAPI CD-ROM support”。注：若没有特定芯片组支持，IDE 仍可工作，但可能无法利用特别的主板的所有性能增强功能。

还请注意：对几乎所有的系统，都极力推荐“Enable PCI DMA by default if available”选项。若没有缺省地启用 DMA（直接内存存取），则 IDE 外围设备将以 PIO 方式运行，并且性能可能比正常情况下慢 15 倍！通过以 root 用户身份在 shell 提示符上输入 `hdparm -d 1 /dev/hdx`，可以验证是否在特别的磁盘上启用了 DMA，其中 `/dev/hdx` 是与要启用 DMA 的磁盘对应的块操作特殊设备。

SCSI 支持

发布时间 :2007-01-27 21:56:12

SCSI support：这一节包含与 SCSI 磁盘和外围设备有关的所有选项。如果有基于 SCSI 的系统，请务必根据需要启用“SCSI support”、“SCSI disk support”、“SCSI CD-ROM support”和“SCSI tape support”。如果正从 SCSI 磁盘引导，则请确保将“SCSI support”和“SCSI disk support”都内编译到内核中，而不是选择将它们编译成可装入的模块。为使 SCSI 正确地工作，还需要执行额外的步骤：进入“SCSI low-level drivers”子类别，确保启用了特别的 SCSI 卡的支持并将其配置为直接编译到内核中。

各种字符设备

发布时间 :2007-01-27 21:56:38

Character devices：这一节包含各种内核驱动程序的大杂烩。请确保启用“Virtual terminal”和“Support for console on virtual terminal”，内核引导后出现的标准的基于文本的控制台需要它们。很有可能还需要启用“Unix98 PTY support”。如果您希望使用并行打印机，请记住还要启用“Parallel printer support”。其它所有选项通常是可选的。推荐采用“Enhanced real-time clock support”；通常需要“/dev/agpgart (AGP support)”和“Direct Rendering Manager”以利用 X 下丰富的 Linux 3D 加速功能（特别当您有 Voodoo3+、ATI Rage 128、ATI Radeon 或 Matrox 等显卡时）。要想使 X 在加速方式中工作，除了简单地启用这些选项外还需要额外的配置步骤。

文件系统和控制台驱动程序

发布时间 :2007-01-27 21:57:08

File systems：正象您猜想的那样，这包含与文件系统驱动程序相关的选项。要确保将用于“/”（根目录）的文件系统编译到内核中。这一文件系统通常是 ext2，但也可能是 ext3、JFS、XFS 或 ReiserFS。请确保还启用了“/proc file system support”选项，因为大多数分发版都需要它。除非计划使用“/dev file system support”（在这种情况下则应将“/dev/pts”选项置为禁用），否则，通常还应启用“/dev/pts file system support for Unix98 PTYs”。

Console drivers：大多数人通常会启用“VGA text console”（x86 系统上通常需要）以及可选的“Video mode selection support”。也有可能使用“Frame-buffer support”，它将使文本控制台以图形显示，而不再是文本屏幕。这些驱动程序中的一些会对 X 有负面影响，因此最好是坚持使用 VGA 文本控制台，至少在开始阶段是如此。

编译和安装内核

编译和安装内核 - make dep

发布时间 :2007-01-27 21:57:53

一旦配置了内核，就到了编译它的时候了。但能够编译它之前，需要生成相关性信息。在 /usr/src/linux 中可通过输入 make dep 做到这一点。

make bzImage

发布时间 :2007-01-27 21:58:19

现在是编译实际二进制内核映象的时候了。输入 make bzImage。几分钟后，将完成编译，并且您将在 /usr/src/linux/arch/i386/boot（用于 x86 PC 内核）中找到 bzImage 文件。很快您将看到如何安装新内核映象，不过现在该讨论模块了。

编译模块

发布时间 :2007-01-27 21:58:48

既然已生成了 bzImage，那么是编译模块的时候了。即使在配置内核时没有启用任何模块，也不要省略这一步——养成编译 bzImage 后立即编译模块的习惯很有益处。而且，如果确实无模块启用编译——这个步骤会一闪而过。输入 `make modules && make modules_install`。这将编译模块并随后将其安装至 `/usr/lib/<kernelversion>` 中。

祝贺您！内核现已完全编译，而且模块全都被编译并安装。现在是重新配置 LILO 的时候了，这样您就可以引导新内核。

引导配置

引导配置 - LILO 简介

发布时间 :2007-01-27 21:59:30

终于到了重新配置 LILO 的时候了，这样它就可以装入新内核。LILO 是最流行的 Linux 引导装入程序，而且所有流行的 Linux 分发版都使用它。首先要做的是查看 `/etc/lilo.conf` 文件。该文件包含类似于 “`image=vmlinuz`” 的一行内容。这一行告诉 LILO 在哪里寻找内核。

配置 LILO

发布时间 :2007-01-27 22:00:04

要配置 LILO 以引导新内核，有两种选择。第一种是覆盖现有的内核 — 如果没有某种紧急引导方法（如带有这一特别的内核的引导磁盘），这样做就有些冒险。

较安全的选择是配置 LILO，以便它既可以引导新内核也可以引导旧内核。LILO 可以配置为在缺省情况下引导新内核，而当遇到问题时仍可选择较旧的内核。这是推荐的选择，我们会向您演示如何执行。

LILO 代码

发布时间 :2007-01-27 22:00:36

lilo.conf 可能看起来如下：

```
boot=/dev/hda
delay=20
vga=normal
root=/dev/hda1
read-only
```

```
image=/vmlinuz
label=linux
```

要向 lilo.conf 添加新的引导项，请执行以下步骤。首先，将 /usr/src/linux/arch/i386/boot/bzImage 复制到 root 分区上的一个文件，如 /vmlinuz2。复制该文件后，再复制 lilo.conf 的最后三行内容，并将它们添加到该文件的末尾 ... 我们几乎完成了 ...

调整 LILO

发布时间 :2007-01-27 22:01:09

现在，lilo.conf 看起来如下：

```
boot=/dev/hda
delay=20
vga=normal
root=/dev/hda1
read-only
```

```
image=/vmlinuz
label=linux
```

```
image=/vmlinuz
label=linux
```

现在，将第一个“image=”行改为读作image=/vmlinuz2。接下来，将第二个“label”行改为读作label=oldlinux。此外，请确保在靠近文件顶部有一行“delay=20”——若没有，则添加一行。如果有，则要确保数字至少是20。

最终的 lilo.conf

发布时间 :2007-01-27 22:01:39

最终的 lilo.conf 文件看起来如下：

```
boot=/dev/hda
delay=20
vga=normal
root=/dev/hda1
read-only

image=/vmlinuz2
label=linux

image=/vmlinuz
label=oldlinux
```

完成上述操作后，需要以 root 身份运行“lilo”。这非常重要！如果不这么做，引导过程将无法进行。运行“lilo”可以给它更新引导映射的机会。

LILO 配置的原因

发布时间 :2007-01-27 22:02:05

现在对所做的更改加以说明。我们对 lilo.conf 文件进行了设置以允许引导两个不同的内核。该文件允许引导位于 /vmlinuz 的原始内核。它也允许引导位于 /vmlinuz2 的新内核。缺省情况下，它会尝试引导新内核（因为新内核的 image/label 行在配置文件中首先出现）。

如果出于某种原因需要引导旧内核，只需重新引导计算机并按住 shift 键。LILO 会检测到这一操作，并且允许输入希望引导的映象的标号。要引导旧内核，需要输入 oldlinux，然后按 Enter 键。要查看可能存在的标号的列表，需要按 TAB 键。

PCI 设备

PCI 设备

发布时间 :2007-01-27 22:02:40

这一节将较仔细地研究在 Linux 下处理 PCI 设备的更详细的信息。在 Linux 下启用对 PCI 设备的支持非常简单。只需确保在 “ General Setup ” 内核配置类别下启用 “ PCI support ”。还推荐采用 “ PCI device name database ” 选项，因为它允许查看 Linux 可以看到的 PCI 设备的实际英文名（而不只是它们的正式 PCI 设备标识号码）。确保启用以上选项后，Linux 就做好了支持 PCI 的准备。

唯一的额外步骤是：针对安装在系统中的卡的类型，启用特定的驱动程序。例如，若要安装 SoundBlaster Live! 卡，则需启用 “ Creative SBLive! ” 支持（位于 “ Sound ” 类别下），而要安装 3Com 3c905c Fast Ethernet 卡的话，则需启用 “ Network device support/Ethernet (10 or 100Mbit) ” 类别 / 子类别下的 “ 3c590/3c900 series (592/595/597) "Vortex/Boomerang" support ”。

检查 PCI 设备

发布时间 :2007-01-27 22:03:12

要查看有关已安装的 PCI 设备的信息，可以输入 `cat /proc/pci` 以查看精简的（且略带神秘的）信息 — 或输入 `lspci -v` 获取更详细和更易于理解的输出信息。“`lspci`”是 `pciutils` 包的一部分，它的源代码可以从 <http://atrey.karlin.mff.cuni.cz/~mj/pciutils.html> 获得。通常，使用与特别的分发版一起提供的 `pciutils` 版本就足够了。当输入 `lspci -v` 时，可能会看到许多以前从不知道其存在的 PCI 设备。通常，这样的设备是众多内置于计算机主板的、基于 PCI 的外围设备之一。可以在计算机 BIOS 中禁用（如果设备当前为不可见，则也可启用）这些设备，通常在“Integrated peripherals”节下进行设置。通常可以在系统引导期间按 `Delete` 键或 `F2` 键来访问计算机的 BIOS。

`pciutils` 包还包含名为“`setpci`”的程序，可以用这个程序更改各种 PCI 设备设置（包括 PCI 设备等待时间）。要了解有关 PCI 设备等待时间以及它对系统所产生的影响的更多信息，请参阅 developerWorks 文章 [Linux hardware stability guide, Part 2](#)。

PCI 设备资源

发布时间 :2007-01-27 22:03:46

为了能正常工作，系统中的 PCI 设备需要利用系统的各种特定硬件资源，如中断等。在有数据等待处理时，许多 PCI 设备利用硬件中断向处理器发送信号。要了解哪些中断正在被各种硬件设备使用，可以通过输入 `cat /proc/interrupts` 查看 `/proc/interrupts` 文件。该命令的输出看起来如下：

```

      CPU0
0:  3493317      XT-PIC timer
1:   86405      XT-PIC keyboard
2:    0      XT-PIC cascade
5:    0      XT-PIC eth0
8:    2      XT-PIC rtc
9:   62653      XT-PIC usb-uhci, usb-uhci, eth1
10: 1550399      XT-PIC Audigy
12:  413422      XT-PIC PS/2 Mouse
14:   85418      XT-PIC ide0
15:    4      XT-PIC ide1
NMI:    0
ERR:    0
```

第一列显示的是 IRQ 号码；第二列显示的是内核对于这一特别的 IRQ 处理了多少中断；最后一列则显示与这个 IRQ 相关联的硬件设备（或多个设备）的“短名”。如您所见，如有必要，多个设备能够共享同一个 IRQ。

还可以通过输入 `cat /proc/ioports` 查看硬件设备正在使用的 IO 端口。

Linux USB

Linux USB

发布时间 :2007-01-27 22:04:19

配置内核时，您可能注意到“USB support”节，这一节包含属于USB（也称为通用串行总线）的选项。USB是将外围设备连接到PC的相对较新的方法。目前，有USB鼠标、键盘、游戏控制器、打印机、调制解调器和更多其它设备。因为Linux USB的支持相对较新，许多Linux用户从未在其Linux系统上使用过USB设备，或可能还不完全熟悉Linux USB支持是如何工作的。以下几页将迅速地介绍Linux USB以帮助您入门。

启用 USB

发布时间 :2007-01-27 22:04:47

要启用 Linux USB 支持，首先进入“USB support”节并启用“Support for USB”选项。尽管这个步骤相当直观明了，但接下来的 Linux USB 设置步骤则会让人感到糊涂。特别地，现在需要选择用于系统的正确 USB 主控制器驱动程序。您的选项是“EHCI”、“UHCI”、“UHCI (alternate driver)”和“OHCI”。这是许多人对 Linux 的 USB 开始感到困惑的地方。

UHCI、OHCI 和 EHCI — 天啊！

发布时间 :2007-01-27 22:05:19

要理解“EHCI”及其同类是什么，首先要知道每块支持插入USB设备的主板或PCI卡都需要有USB主控制器芯片组。这个特别的芯片组与插入系统的USB设备进行相互操作，并负责处理允许USB设备与系统其它部分通信所必需的所有低层次细节。

Linux USB 驱动程序有三种不同的USB主控制器选项是因为在主板和PCI卡上有三种不同类型的USB芯片。“EHCI”驱动程序设计成为实现新的高速USB 2.0协议的芯片提供支持。“OHCI”驱动程序用来为非PC系统上的（以及带有SiS和ALi芯片组的PC主板上的）USB芯片提供支持。“UHCI”驱动程序用来为大多数其它PC主板（包括Intel和Via）上的USB实现提供支持。只需选择与希望启用的USB支持的类型对应的“?HCI”驱动程序即可。如有疑问，为保险起见，可以启用“ECHI”、“UHCI”（两者中任选一种，它们之间没有明显的区别）和“OHCI”。

最后几步

发布时间 :2007-01-27 22:05:49

启用了“USB support”和适当的“?HCI”USB 主控制器驱动程序后，使 USB 启动并运行只需再进行几个步骤。应该启用“Preliminary USB device filesystem”，然后确保启用所有特定于将与 Linux 一起使用的实际 USB 外围设备的驱动程序。例如，为了启用对 USB 游戏控制器的支持，我启用了“USB Human Interface Device (full HID) support”。我还启用了主“Input core support”节下的“Input core support”和“Joystick support”。

挂装 usbdevfs

发布时间 :2007-01-27 22:06:21

一旦用新的已启用 USB 的内核重新引导后，应输入以下命令将 USB 设备文件系统挂装到 /proc/bus/usb：

```
# mount -t usbdevfs none /proc/bus/usb
```

为了在系统引导时自动挂装 USB 设备文件系统，请将下面一行添加到 /etc/fstab 中的 /proc 挂装行之后：

```
none          /proc/bus/usb      usbdevfs      defaults 0 0
```

有关 USB 的更多信息，请访问随后在“参考资料”中列出的 USB 站点。

海量Linux文章

海量 Linux 技术文章

发布时间 :2006-11-24 16:50:29

下面是linux技术文章快速入口。需要联网：

[Linux 技术交流](#)

<http://www.linuxdiyf.com/bbs/forum-3-1.html>

[Linux 应用](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=1>

[Linux 安装及学习指导](#)

<http://www.linuxdiyf.com/bbs/forum-45-1.html>

[Linux 系统安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=11>

[Linux 学习指导](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=12>

[Linux 软件安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=13>

[shell](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=3>

[Linux 壁纸](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=4>

[红旗](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=5>

[Redhat](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=6>

[SuSE](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=7>

Linux 认证

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=9>

Linux下载分享{酷件、书籍、视频分享 }

<http://www.linuxdiyf.com/bbs/forum-6-1.html>

服务器应用

<http://www.linuxdiyf.com/bbs/forum-7-1.html>

数据库应用

<http://www.linuxdiyf.com/bbs/forum-8-1.html>

Linux 编程与内核

<http://www.linuxdiyf.com/bbs/forum-9-1.html>

UniX 技术文章

<http://www.linuxdiyf.com/bbs/forum-32-1.html>

Linux 业界声音、新闻

<http://www.linuxdiyf.com/bbs/forum-11-1.html>

Linux 人才招聘信息

<http://www.linuxdiyf.com/bbs/forum-46-1.html>

网络转载，感谢原创作者！

制作：红联Linux论坛

祝您阅读愉快！

Linux培训系列

第七讲

将介绍 TCP/IP 和以太网 Linux 联网的基本原理，说明如何使用 inetd 和 xinetd 超级服务器，提供保护 Linux 系统的重要技巧，还将说明如何设置和使用 Linux 打印服务器。

内容基础，语言简短简洁

红联Linux论坛是致力于Linux技术讨论的站点，目前网站收录的文章及教程基本能满足不同水平的朋友学习。

红联Linux门户：www.linux110.com

红联Linux论坛：www.linuxdiyf.com/bbs

下载:Linux电子书籍：

<http://www.linux286.com/linux/linuxdzsj.htm>

目录

TCP/IP 联网

TCP/IP 联网

仅有以太网还不够

解决方案：以太网上的 TCP/IP

IP 地址简介

将 IP 地址与以太网接口关联

使用 ifconfig -a

TCP/IP 在运行了

名称解析限制

使用 DNS

连接至外部世界

因特网服务

因特网服务 - inetd 简介

配置 inetd : /etc/services

配置 inetd ; /etc/inetd.conf

禁用服务

使用初始脚本停止 / 启动 inetd

用手工方式停止 / 启动 inetd

TCP 封装器简介

用 TCP 封装器进行日志记录

用 TCP 封装器限制对本地用户的访问

用 TCP 封装器将访问限制为已知主机

xinetd : 扩展的 inetd

xinetd 配置

安全性概述

安全性概述

日志文件的文件权限

root 用户其它文件的文件权限

用户文件的文件权限

查找 SUID/SGID 程序

用 ulimit 设置用户限制

用 ulimit 设置 CPU 时间限制

关闭未使用的网络服务（超级服务器）

关闭未使用的网络服务（独立服务器）

测试更改

拒绝登录以进行维护

iptables (ipchains) 简介

iptables 和 Linux 信息包过滤器

入侵检测 — 系统日志 (syslog)

入侵检测 — tripwire

入侵检测 — portsentry

常规指南：保持软件为最新

打印

打印

安装打印假脱机程序守护程序 (lpd)

基本打印机设置 (/etc/printcap)

创建假脱机文件目录

使用打印假脱机程序客户机

打印至远程 LPD 服务器

打印至远程的 MS Windows 或 Samba 服务器

Magicfilter

调整 printcap 以指向 Magicfilter

Linux海量文章

海量Linux技术文章

TCP/IP 联网

TCP/IP 联网

发布时间 :2007-01-28 11:42:17

简介

设置一个由大量 Linux 机器组成的基于以太网的局域网（LAN）是常见且相对简单的任务。通常，需要做的就是确保 Linux 系统都安装了某种以太网卡。然后，使用以太网电缆将机器连接到中央以太网集线器或交换机。若所有系统都把对相应的以太网卡的支持（以及 TCP/IP 支持）编译到内核中，则就技术而言，这些系统已经具备了在新的以太网 LAN 上通信的一切条件。

仅有以太网还不太够

发布时间 :2007-01-28 11:42:49

尽管有了让 LAN 工作所需的所有硬件和内核支持，仍不会有多大用处。绝大多数 Linux 应用程序与服务并不使用原始的以太网信息包或帧交换信息。相反，它们使用称为 TCP/IP 的高级协议。毫无疑问，您一定听说过 TCP/IP — 它是一组大体上形成因特网基础的协议（因此得名：传输控制协议 / 网际协议）。

解决方案：以太网上的 TCP/IP

发布时间 :2007-01-28 11:43:17

于是，解决方案就是配置新的以太网 LAN 以使它可以交换 TCP/IP 流量。要理解解决方案是如何工作的，首先需要知道一点有关以太网的知识。特别地，以太网 LAN 上每台机器中的以太网卡都有唯一的硬件地址。网卡在生产时就被分配了硬件地址，硬件地址看起来与下面相似：

00:01:02:CB:57:3C

IP 地址简介

发布时间 :2007-01-28 11:43:49

这些硬件地址被用做以太网 LAN 上单个系统的唯一地址。使用硬件地址的话，一台机器可以做一些事情，例如，可以向另一台机器发送以太网帧。这一方法存在的问题是基于 TCP/IP 的通信使用另一寻址方案，即称为 IP 地址的寻址方案。IP 地址看起来如下：

192.168.1.1

将 IP 地址与以太网接口关联

发布时间 :2007-01-28 11:44:20

为了使以太网 LAN 使用 TCP/IP，需要将每台机器的以太网卡（因而也就是它的硬件地址）与一个 IP 地址关联。幸运的是，在 Linux 下有一个将 IP 地址与以太网接口关联的简便方法。事实上，如果当前正在通过 Linux 使用以太网，那么分发版的系统初始化脚本中很可能有类似如下的命令：

```
ifconfig eth0 192.168.1.1 broadcast 192.168.1.255 netmask 255.255.255.0
```

以上命令中，ifconfig 命令被用来关联 eth0（也就是 eth0 的硬件地址）和 192.168.1.1 IP 地址。另外，还指定了各种其它与 IP 相关的信息，包括广播地址（192.168.1.255）和网络掩码（255.255.255.0）。当命令完成时，eth0 接口将被启用并具有关联的 IP 地址。

使用 ifconfig -a

发布时间 :2007-01-28 11:44:57

可以通过输入 ifconfig -a 查看当前正在运行的所有网络设备，命令执行结果的输出与下面相似：

```
eth0    Link encap:Ethernet  HWaddr 00:01:02:CB:57:3C
        inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:100
        RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
        Interrupt:5 Base address:0xc400

lo      Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        UP LOOPBACK RUNNING  MTU:16436  Metric:1
        RX packets:1065 errors:0 dropped:0 overruns:0 frame:0
        TX packets:1065 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
        RX bytes:262542 (256.3 Kb)  TX bytes:262542 (256.3 Kb)
```

从上面的输出可以看到一个已配置的 eth0 接口和 lo (localhost) 接口。lo 接口是特殊的虚拟接口，它经过配置以使您在即便没有网络的情况下也可以在本地运行 TCP/IP 应用程序。

TCP/IP 在运行了

发布时间 :2007-01-28 11:45:25

当所有网络接口都设置好并与相应的 IP 地址关联后，以太网网络也可以用来传送 TCP/IP 流量。LAN 上的系统现在可以用 IP 地址相互寻址，象 ping、telnet 和 ssh 这样的命令将在机器间正常工作。

名称解析限制

发布时间 :2007-01-28 11:45:55

然而，尽管能够输入 ping 192.168.1.1 这样的命令，但还不能通过名称引用机器。例如，不能输入 ping mybox。要做到这一点，需要在每台 Linux 机器上设置名为 /etc/hosts 的文件。在该文件中，要指定一个 IP 地址，以及与每个 IP 地址关联的名称（或多个名称）。因此，如果我有带三个节点的网络，那么我的 /etc/hosts 文件看起来可能与下面相似：

```
127.0.0.1    localhost
192.168.1.1  mybox mybox.gentoo.org
192.168.1.2  testbox testbox.gentoo.org
192.168.1.3  mailbox mailbox.gentoo.org
```

请注意：/etc/hosts 包含“localhost”到 127.0.0.1 IP 地址的强制映射。我还指定了 LAN 上所有系统的主机名称，包括短名称（“mybox”）和全限定名称（“mybox.gentoo.org”）。将这个 /etc/hosts 文件复制到每个系统后，我就能够通过名称而不仅是 IP 地址来引用系统了。ping mybox 现在将可以执行了！

使用 DNS

发布时间 :2007-01-28 11:46:26

虽然这种方法可用于小型 LAN，但不便于在拥有许多系统的大型 LAN 上使用。对于这样的配置，通常最好是将所有的 IP 至主机名映射信息存储在一台机器上，然后在这台机器上设置所谓的“DNS 服务器”（域名服务服务器）。然后，可以配置每台机器来联系这台特别的机器以接收最新的 IP 至名称映射。要做到这一点，可以在每台机器上创建一个与下面相似的 /etc/resolv.conf 文件：

```
domain gentoo.org
nameserver 192.168.1.1
nameserver 192.168.1.2
```

在上面的 /etc/resolv.conf 中，我告诉系统所有非限定主机名（如与“testbox.gentoo.org”相对的“testbox”）都应视为本地主机名。我还指定一台运行在 192.168.1.1 上的 DNS 服务器，以及一台运行在 192.168.1.2 上的备份服务器。实际上，几乎所有与网络连接的 Linux PC 都已经在其 resolv.conf 文件中指定了名称服务器，即使它们不在 LAN 上，也是如此。这是因为它们在其因特网服务供应商处被配置为使用 DNS 服务器，以便将主机名映射为 IP 地址（这样，那个系统上的用户就可以浏览 Web 并访问象 ibm.com 这样的著名站点，而无需通过 IP 地址来引用它们！）。

连接至外部世界

发布时间 :2007-01-28 11:46:56

说到与因特网连接，该如何配置我们简单的 3 系统 LAN 以使它与“外部世界”的系统连接呢？通常，我们会购买某种路由器将我们的以太网网络与 DSL 或有线电视调制解调器、T1 或电话线连接。可以用 IP 地址配置这个路由器以使它能够与我们 LAN 上的系统通信。我们可以依次将 LAN 上每个系统都配置为将这个路由器作为其缺省路由或网关使用。这样做的意义在于：对不在我们 LAN 上的系统寻址的任何网络数据将被路由至我们的路由器，路由器再负责将数据转发至我们 LAN 之外的远程系统。通常，分发版的系统初始化脚本会为您处理缺省路由的设置。这些脚本执行该操作所用的命令看起来可能与下面相似：

```
route add -net default gw 192.168.1.80 netmask 0.0.0.0 metric 1
```

在上述 route 命令中，缺省路由设置为 192.168.1.80 — 路由器的 IP 地址。要查看系统上所有配置的路由，可以输入 route -n。目标为“0.0.0.0”的路由是缺省路由。

因特网服务

因特网服务 - inetd 简介

发布时间 :2007-01-28 11:47:47

单个 Linux 系统可以提供数十、甚至数百个网络服务。例如，使用 telnet 程序，您可以访问远程系统上的 telnet 服务。同样地，使用 ftp 程序，您可与远程系统上的 ftp 服务连接。

为了提供这些服务，远程系统运行每个服务器的实例（例如 /usr/sbin/in.telnetd 和 /usr/sbin/in.ftpd）以接受连接或者运行 inetd。inetd 程序接受每个进入的连接，然后根据其连接的类型启动处理该连接的适当的服务。出于这个原因，inetd 也被称为“因特网超级服务器”。

在典型的安装了 Linux 的系统上，inetd 处理大多数进入的连接。只有少数程序（如 sshd 和 lpd）处理它们自己的网络通信而无需依靠 inetd 接受进入的连接。

配置 inetd : /etc/services

发布时间 :2007-01-28 11:48:25

上页提到 inetd 根据类型对进入的连接进行分类。每个进入的连接都在 TCP/IP 头中包含一些标识字段。我们最感兴趣的字段是源地址、目标地址协议和端口号。进入连接由 inetd 根据端口号和协议（通常是 TCP 或 UDP，请查看 /etc/protocols 以获得完整的 inetd 可以提供的服务列表）进行分类。

每行都有如下格式：

```
service-name port-number/protocol-name aliases # comment
```

例如，让我们研究最上面的几项：

```
# grep ^[^#] /etc/services | head -5
tcpmux    1/tcp          # TCP port service multiplexer
echo      7/tcp
echo      7/udp
discard   9/tcp  sink null
discard   9/udp  sink null
```

通常，/etc/services 已经包含所有有用的服务名称和端口。如果您希望添加自己的端口，可以查询已分配端口号列表。

配置 inetd ; /etc/inetd.conf

发布时间 :2007-01-28 11:48:59

inetd 的实际配置是在 /etc/inetd.conf 中完成的，配置格式如下：

service-name socket-type protocol wait-flag user server-program

因为服务是在 inetd.conf 中由服务名称而不是端口指定的，所以，为了符合由 inetd 处理的条件，必须将服务列在 /etc/services 中。

让我们看看 /etc/inetd.conf 一些常见的行。例如，telnet 和 ftp 服务：

```
# grep ^telnet /etc/inetd.conf
telnet stream tcp  nowait root  /usr/sbin/in.telnetd
```

```
# grep ^ftp /etc/inetd.conf
ftp  stream tcp  nowait root  /usr/sbin/in.ftpd -l -a
```

对这两个服务的配置为：使用 TCP 协议，并以 root 用户的身份运行服务器（in.telnetd 或 in.ftpd）。有关 /etc/inetd.conf 中字段的完整说明，请参阅 inetd(8) 手册页。

禁用服务

发布时间 :2007-01-28 11:49:31

在 inetd 中禁用服务很简单：只要在 /etc/inetd.conf 注释掉该服务所在的行即可。您可能不希望完全除去该行，以便以后需要时可以引用它。例如，有些系统管理员出于安全性的原因宁愿禁用 telnet（因为连接完全是明文）：

```
# vi /etc/inetd.conf  
[comment out undesired line]
```

```
# grep ^.telnet /etc/inetd.conf  
#telnet stream tcp    nowait root    /usr/sbin/in.telnetd
```

使用初始脚本停止 / 启动 inetd

发布时间 :2007-01-28 11:50:06

我们在上页对 /etc/inetd.conf 所做的更改将在重新启动 inetd 程序后才生效。大多数分发版在 /etc/init.d 或 /etc/rc.d/init.d 中有初始脚本：

```
# /etc/rc.d/init.d/inet stop
Stopping INET services:          [ OK ]
```

```
# /etc/rc.d/init.d/inet start
Starting INET services:         [ OK ]
```

事实上，通常可以使用 “ restart ” 作为快捷方式：

```
# /etc/rc.d/init.d/inet restart
Stopping INET services:         [ OK ]
Starting INET services:        [ OK ]
```

用手工方式停止 / 启动 inetd

发布时间 :2007-01-28 11:50:44

如果上页中的助手脚本不起作用，老式方法甚至更简单。可以使用 killall 命令停止 inetd：

```
# killall inetd
```

可以在命令行调用 inetd 来简单地再次启动它。它会自动在后台运行：

```
# /usr/sbin/inetd
```

有一个快捷方式无需停止 inetd 就可命令它重新读取配置文件：只要向它发送 HUP 信号：

```
# killall -HUP inetd
```

此刻应该不能 telnet 或 ftp 到这个系统，因为 telnet 和 ftp 被禁用。尝试用 telnet localhost 进行检查。如果需要 telnet 或 ftp 访问，所需做的全部就是重新启用它！

以下是我所遇到的情况：

```
# telnet localhost
telnet: Unable to connect to remote host: Connection refused
```

TCP 封装器简介

发布时间 :2007-01-28 11:51:14

tcp_wrappers 包提供了一个名为 tcpd 的很小的守护程序，该程序由 inetd 而不是实际的服务守护程序调用。tcpd 程序将每个进入连接的源地址编入日志，并可以过滤它们而只允许来自可信系统的连接。

要使用 tcpd，可以按下列方式将它插入到 inetd 中：

```
ftp    stream tcp    nowait root    /usr/sbin/tcpd  in.ftpd -l -a
telnet stream tcp    nowait root    /usr/sbin/tcpd  in.telnetd
```

用 TCP 封装器进行日志记录

发布时间 :2007-01-28 11:51:45

缺省情况下，连接不受限制但会被日志记录下来。例如，我们可以重新启动 inetd 以使在前页做的更改生效。然后一些快速调查应该显示已记录的连接：

```
# telnet localhost
login: (press <ctrl-d> to abort)
```

```
# tail -1 /var/log/secure
Feb 12 23:33:05 firewall in.telnetd[440]: connect from 127.0.0.1
```

tcpd 将记录 telnet 连接尝试，因此看起来有些东西在工作了。因为 tcpd 提供一致的连接日志记录服务，这就免除了单个服务守护程序每次自己将连接编入日志的需要。事实上，在接受连接的工作方面，它与 inetd 相似，因为那使每个守护程序不需要接受自己的连接。Linux (UNIX) 的简单程度真是不可思议！

用 TCP 封装器限制对本地用户的访问

发布时间 :2007-01-28 11:52:16

tcpd 程序配置为使用两个文件：/etc/hosts.allow 和 /etc/hosts.deny。这两个文件中行的格式为：

```
daemon_list : client_list [ : shell_command ]
```

按以下顺序授权或拒绝访问。搜索在第一次匹配时停止：

当与 /etc/hosts.allow 中的项匹配时，则授权访问

当与 /etc/hosts.deny 中的项匹配时，则拒绝访问

若没有匹配项，则授权访问

例如，若只允许对内部网络进行 telnet 访问，可通过在 /etc/hosts.deny 中设置策略（拒绝除 localhost 以外的其它来源的所有连接）着手：

```
in.telnetd: ALL EXCEPT LOCAL
```

用 TCP 封装器将访问限制为已知主机

发布时间 :2007-01-28 11:52:49

无需重新装入 inetd，因为每当 telnet 端口有连接时，就会调用 tcpd。因此我们可以立即尝试：

```
# telnet box.yourdomain.com
Trying 10.0.0.1...
Connected to box.yourdomain.com.
Escape character is '^]'.
Connection closed by foreign host.
```

哦！被拒绝了！（这是人生中为数不多的几次经历：拒绝表示成功。）要重新启用来自自己网络的访问，可在 /etc/hosts.allow 中插入例外：

```
in.telnetd: .yourdomain.com
```

此刻我们就能够再次成功地用 telnet 访问系统了。而这仅仅触及了 tcp_wrappers 的能力的表面。在 tcpd(8) 和 hosts_access(5) 手册页中还有有关 tcp_wrappers 的更多信息。

xinetd : 扩展的 inetd

发布时间 :2007-01-28 11:53:19

尽管 inetd 是经典的因特网超级服务器，但最近对它进行了多次改写以试图添加特性和更多的安全性。xinetd 程序在许多新近的分发版（包括 Red Hat 和 Debian）中取代了 inetd。部分扩展的特性是：

- 访问控制（内置 TCP 封装器）
- 详尽的日志记录（连接持续时间和失败的连接等等）
- 来自另一个主机的服务重定向
- IPv6 支持
- 通过代码片段而不是一个汇总文件进行配置

xinetd 配置

发布时间 :2007-01-28 11:54:01

xinetd 的配置文件是 /etc/xinetd.conf。最常见的情况下，那个文件仅包含为其余服务设置缺省配置参数的几行：

```
# cat /etc/xinetd.conf
defaults
{
    instances      = 60
    log_type       = SYSLOG authpriv
    log_on_success = HOST PID
    log_on_failure = HOST RECORD
}
includedir /etc/xinetd.d
```

文件的最后一行指示 xinetd 从 /etc/xinetd.d 目录的文件代码片段读取额外的配置信息。我们快速地看看 telnet 代码片段：

```
# cat /etc/xinetd.d/telnet
service telnet
{
    flags          = REUSE
    socket_type    = stream
    wait          = no
    user           = root
    server         = /usr/sbin/in.telnetd
    log_on_failure += USERID
}
```

如您所见，配置 xinetd 并不困难，而且它比 inetd 更直观。您可以在 xinetd(8)、xinetd.conf(5) 和 xinetd.log(5) 手册页中获得有关 xinetd 的更多信息。

在 Web 上也有关于 inetd、tcp_wrappers 和 xinetd 的大量信息。

安全性概述

安全性概述

发布时间 :2007-01-28 11:54:38

维护一个完全安全的系统是不可能的。然而，只要勤奋，则有可能使 Linux 机器足够安全，并让大多数偶尔出现的骇客、脚本小子（script-kiddies）以及其它的“坏家伙”止步而去骚扰其他人。请记住：仅仅遵循本教程不会产生一个安全的系统。相反，我们希望您接触到主要主题的多个方面，并向您提供一些有关如何入门的有用示例。

Linux 系统安全性可分为两个部分：内部安全性和外部安全性。内部安全性指预防用户无意或恶意地破坏系统。外部安全性指防止未授权用户获得对系统的访问。

本章将首先介绍内部安全性，然后介绍外部安全性，最后介绍一些常规指导原则和技巧。

日志文件的文件权限

发布时间 :2007-01-28 11:55:09

内部安全性可以是很大的任务，这要看您对用户的信任程度。这里介绍的指导原则是设计用来防止偶然用户访问敏感信息和防止不公平地使用系统资源。

至于文件权限，您可能希望修改以下三种情况的权限：

首先，/var/log 中的日志文件不需要是所有人都可以读取的。没有理由让非 root 用户窥视日志。

root 用户其它文件的文件权限

发布时间 :2007-01-28 11:55:39

其次，root 用户的点文件对于普通用户应是不可读的。检查 root 用户主目录中的文件（ls -la）以确保它们受到适当的保护。甚至可以使整个目录仅对 root 用户可读：

```
# cd
# pwd
/root
# chmod 700 .
```

用户文件的文件权限

发布时间 :2007-01-28 11:56:12

最后，用户文件在缺省情况下通常被创建为所有人可读。那可能不是用户所期望的，而且它当然不是最好的策略。应该使用与下面类似的命令在 /etc/profile 中设置缺省的 umask：

```
if [ "$UID" = 0 ]; then
    # root user; set world-readable by default so that
    # installed files can be read by normal users.
    umask 022
else
    # make user files secure unless they explicitly open them
    # for reading by other users
    umask 077
fi
```

应该查询 umask(2) 和 bash(1) 手册页以获取有关设置 umask 的更多信息。请注意：umask(2) 手册页涉及 C 函数，但它所包含的信息也适用于 bash 命令。

查找 SUID/SGID 程序

发布时间 :2007-01-28 11:56:55

寻求 root 访问权的恶意用户总是会在系统上寻找设置了 SUID 或 SGID 位的程序。

应该仔细考虑每个程序以确定是否需要将其 SUID 或 SGID 位打开。系统上有些 SUID/SGID 程序可能是根本不需要的。

要搜索具有这样性质的程序，可使用 find 命令。例如，可以在 /usr 目录中启动对 SUID/SGID 程序的搜索：

```
# cd /usr
# find . -type f -perm +6000 -xdev -exec ls {} \;
-rwsr-sr-x  1 root  root    593972 11-09 12:47 ./bin/gpg
-r-xr-sr-x  1 root  man      38460 01-27 22:13 ./bin/man
-rwsr-xr-x  1 root  root     15576 09-29 22:51 ./bin/rcp
-rwsr-xr-x  1 root  root      8256 09-29 22:51 ./bin/rsh
-rwsr-xr-x  1 root  root     29520 01-17 19:42 ./bin/chfn
-rwsr-xr-x  1 root  root     27500 01-17 19:42 ./bin/chsh
-rwsr-xr-x  1 lp    root      8812 01-15 23:21 ./bin/lppasswd
-rwsr-x---  1 root  cron     10476 01-15 22:16 ./bin/crontab
```

在这个清单中，我已经发现了需要更仔细检查的候选对象：lppasswd 是 CUPS 打印软件分发版的一部分。因为没有在系统上提供打印服务，所以我会考虑除去 CUPS，那也会除去 lppasswd 程序。lppasswd 中可能没有危及安全性的错误，但为什么要在不使用的程序上冒险呢？同样地，应该关闭所有不使用的服务。您总是可以在需要时再启用它们。

用 ulimit 设置用户限制

发布时间 :2007-01-28 11:57:29

bash 中的 ulimit 命令提供了限制特定用户的资源使用情况的方法。一旦限制降低，则在进程的生命期内无法提高该限制。此外，该限制会被所有子进程继承。结果是：可以在 /etc/profile 中调用 ulimit，而限制将以不能撤销的方式应用于所有用户（假设用户正在运行 bash 或另一个 shell，该 shell 在登录时运行 /etc/profile）。

要检索当前限制，可使用 ulimit -a：

```
# ulimit -a
core file size      (blocks, -c) 0
data seg size       (kbytes, -d) unlimited
file size           (blocks, -f) unlimited
max locked memory   (kbytes, -l) unlimited
max memory size     (kbytes, -m) unlimited
open files          (-n) 1024
pipe size           (512 bytes, -p) 8
stack size          (kbytes, -s) unlimited
cpu time            (seconds, -t) unlimited
max user processes  (-u) 3071
virtual memory      (kbytes, -v) unlimited
```

以一种能实际提高系统安全性而不会对合法用户造成麻烦的方式设置这些限制是相当复杂的，所以调整这些设置时要小心。

用 ulimit 设置 CPU 时间限制

发布时间 :2007-01-28 11:58:01

作为 ulimit 的一个示例，我们尝试将一个进程的 CPU 时间设置为 1 秒钟，然后用一个忙循环使它超时。一定要确保启动新的 bash 进程（象我们在下面做的那样），以在其中进行尝试；否则将被注销！

```
# time bash
# ulimit -t 1
# while true; do true; done
Killed
```

```
real  0m28.941s
user   0m1.990s
sys    0m0.017s
```

在上面的示例中，“user”时间加上“sys”时间等于该进程所用的全部 CPU 时间。当 bash 进程到达 2 秒标记时，Linux 断定它超过 1 秒的限制，因此该进程被杀掉。酷吧？

注：一秒钟只是示例而已。不要对您的用户这样做！即使几小时也是不对的，因为 X 真地很消耗时间（我当前的会话已用掉了 69+ 小时的 CPU 时间）。在实际的实现中，您可能要对某些项而不是 CPU 时间执行 ulimit。

关闭未使用的网络服务（超级服务器）

发布时间 :2007-01-28 11:58:55

关闭未使用的网络服务一直是提高入侵预防能力的好方法。例如，如果正在运行因特网超级服务器（如本教程前面描述的 `inetd` 或 `xinetd`），那么 `in.rshd`、`in.rlogind` 和 `in.telnetd` 通常都在缺省情况下启用。这些网络服务几乎都已被更安全的替代项（如 `ssh`）所取代。

要在 `inetd` 中禁用服务，只需在 `/etc/inetd.conf` 中在适当的行前面加上“#”将其注释掉；然后重新启动 `inetd` 即可。（这在本教程前面已有描述，若需要复习，可返回几页快速浏览。）

要在 `xinetd` 中禁用服务，可以执行与 `/etc/xinetd.d` 中适当的代码片段相似的工作。例如，要禁用 `telnet`，可以将 `/etc/xinetd.d/telnet` 文件的整个内容注释掉，或简单地删除该文件。重新启动 `xinetd` 以完成此过程。

如果正在结合 `inetd` 使用 `tcpd`，或如果正在使用 `xinetd`，还可以选择限制与可信的主机进行的进入连接。对于 `tcpd`，可参阅本教程的前几章。对于 `xinetd`，可在 `xinetd.conf(5)` 手册页中搜索“`only_from`”。

关闭未使用的网络服务（独立服务器）

发布时间 :2007-01-28 11:59:32

有些服务器并不由 inetd 或 xinetd 启动，但却作为“独立”服务器始终运行着。这样的服务器通常是 atd、lpd、sshd、nfsd 和其它服务器。事实上，inetd 和 xinetd 本身都是独立服务器，如果在它们各自的配置文件中注释掉所有的服务，就选择了将它们完全关闭。

独立服务器通常在系统引导或更改运行级别时由 init 系统启动。

要使 init 系统不再启动服务器，在每个运行级别目录中找到指向该服务器启动脚本的符号链接，然后删除它。运行级别目录的名称通常为 /etc/rc3.d 或 /etc/rc.d/rc3.d（针对运行级别 3）。还需要检查其它运行级别。

除去服务的运行级别符号链接后，仍需要关闭当前运行的服务器。最好用服务的初始化脚本完成这一操作，通常可以在 /etc/init.d 或 /etc/rc.d/init.d 中找到这一脚本。例如，要关闭 sshd：

```
# /etc/init.d/sshd stop
* Stopping sshd... [ ok ]
```

测试更改

发布时间 :2007-01-28 12:00:09

在修改 inetd 或 xinetd 配置以禁用或限制服务，或用服务器初始化脚本关闭该服务器后，应该对所做的更改加以测试。可以使用 telnet 客户机通过指定服务名称或号码来测试 tcp 端口。例如，要验证 rlogin 已被禁用：

```
# grep ^login /etc/services
login      513/tcp
# telnet localhost 513
Trying 127.0.0.1...
telnet: Unable to connect to remote host: Connection refused
```

除了标准 telnet 客户机以外，还应考虑使用实用程序以测试系统“开放程度”的可能性。我们推荐使用 netcat 和 nmap。

ncat 是“网络瑞士军刀”：它是使用 TCP 或 UDP 协议、跨越网络连接读写数据的简单 UNIX 实用程序。nmap 是用于网络探测或安全性审计的实用程序。具体而言，nmap 扫描端口以确定哪个端口打开了。

可以在本教程最后一章（参考资料）中找到指向这些实用程序的链接。

拒绝登录以进行维护

发布时间 :2007-01-28 12:00:44

除了以上方法外，还有通过创建 /etc/nologin 文件来拒绝登录的普通方法。通常这一方法用于短期维护操作。仍然可以允许以 root 用户身份登录，但将拒绝以其他用户身份登录。例如：

```
# cat > /etc/nologin
```

```
=====
```

```
System is currently undergoing maintenance
until 2:00. Please come back later.
```

```
=====
```

```
# telnet localhost
```

```
login: agriffis
```

```
Password:
```

```
=====
```

```
System is currently undergoing maintenance
until 2:00. Please come back later.
```

```
=====
```

```
Login incorrect
```

完成维护后，一定要删除这个文件，否则在您想起以前，没人能登录！我可没这么做过，对，我没有。

iptables (ipchains) 简介

发布时间 :2007-01-28 12:01:16

iptables 和 ipchains 命令用于在运行的 Linux 内核中调整 and 检查网络信息包过滤器规则。ipchains 命令用于 2.2.x 版本内核，尽管它仍可用于 2.4.x 内核，但已被 iptables 取代。

可设置信息包过滤器规则进行防火墙和路由器的活动。可以对 iptables 命令加上 -L 选项来检查当前的规则：

```
# iptables -L
Chain INPUT (policy ACCEPT)
target    prot opt source                destination
```

```
Chain FORWARD (policy ACCEPT)
target    prot opt source                destination
```

```
Chain OUTPUT (policy ACCEPT)
target    prot opt source                destination
```

这是一个非常开放的系统示例，没有启用路由或防火墙。

iptables 和 Linux 信息包过滤器

发布时间 :2007-01-28 12:01:50

有效地使用 Linux 信息包过滤器需要对 TCP/IP 联网及其如何在 Linux 内核中实现有扎实的理解。netfilter 主页（请参阅本教程最后一章的参考资料，以获得链接）是学习更多知识的好去处。

在能自如地构建自己的规则集以前，有许多脚本可以让您入门，只要您信任它们的作者即可。最完整的脚本之一是 gShield（请参阅参考资料）。您可以调整其注释良好且相当简单的配置文件以设置信息包过滤器规则最常规的格式。

入侵检测 — 系统日志 (syslog)

发布时间 :2007-01-28 12:02:23

入侵检测通常被那些相信自己安置的入侵预防设备的系统管理员所忽略。不幸的是，这意味着一旦黑客找到可以入侵的细微漏洞，在注意到他们的存在以前，系统可能很长一段时间都处于他们的控制之下。

入侵检测最基本的形式是注意系统日志。这些文件通常出现在 /var/log 目录中，不过实际的文件名会因分发版和配置而有所不同。

```
# less /var/log/messages
Feb 17 21:21:38 [kernel] Vendor: SONY    Model: CD-RW CRX140E  Rev: 1.0n
Feb 17 21:21:39 [kernel] eth0: generic NE2100 found at 0xe800, Version 0x031243,
DMA 3 (autodetected), IRQ 11 (autodetected).
Feb 17 21:21:39 [kernel] ne.c:v1.10 9/23/94 Donald Becker ( becker@scyld.com)
Feb 17 21:22:11 [kernel] NVRM: AGPGART: VIA MVP3 chipset
Feb 17 21:22:11 [kernel] NVRM: AGPGART: allocated 16 pages
Feb 17 22:20:05 [PAM_pwd] authentication failure; (uid=1000)
-> root for su service
Feb 17 22:20:06 [su] pam_authenticate: Authentication failure
Feb 17 22:20:06 [su] - pts/3 chouser-root
```

要理解所有这些消息可能需要进行一些实践，但大多数重要消息都相当清楚。例如，在日志的末尾，我们可以看到用户 “chouser” 试图使用 su 成为 root 用户，但失败了。

入侵检测 — tripwire

发布时间 :2007-01-28 12:02:52

有许多可用的包可以对整个文件系统进行“快照”，然后将它与较早的快照比较以了解什么发生了更改。若能明确地定义哪些文件作为系统正常操作的一部分应该发生更改，则这些包能很快提醒黑客的存在及其活动。

Tripwire 是最流行的入侵检测包之一（请参阅本教程末尾的参考资料以获取链接）。安装 tripwire 后，必须定制它的配置文件以使它知道哪些文件应该更改而哪些不应更改。还需要告诉它如何向您发送有关发生什么更改的报告，以及它应隔多久运行一次（通常每天一次）。

入侵检测 — portsentry

发布时间 :2007-01-28 12:03:25

PortSentry 包来自 Psionic Technologies，它实际上有点介于入侵预防与检测之间。该包监控网络连接，并且如果它看到任何它认为“可疑”的与系统连接的尝试，它会把这一事件编入日志然后阻止它再次发生。该包也可以在本教程末尾的参考资料中找到。

当安装了该包并运行它时，将能够在 syslog 中看到所有尝试的连接，并看到 PortSentry 如何对它们做出反应：

```
# tail /var/log/messages
Oct 15 00:21:24 mycroft portsentry[603]: attackalert:
    SYN/Normal scan from host: 302.174.40.34/302.174.40.34 to TCP port: 111
Oct 15 00:21:24 mycroft portsentry[603]: attackalert:
    Host 302.174.40.34 has been blocked via wrappers with string:
    "ALL: 302.174.40.34"
Oct 15 00:21:24 mycroft portsentry[603]: attackalert:
    Host 302.174.40.34 has been blocked via dropped route using command:
    "/sbin/route add -host 302.174.40.34 reject"
Oct 15 00:21:24 mycroft portsentry[603]: attackalert:
    SYN/Normal scan from host: 302.174.40.34/302.174.40.34 to TCP port: 111
Oct 15 00:21:24 mycroft portsentry[603]: attackalert:
    Host: 302.174.40.34/302.174.40.34 is already blocked Ignoring
Oct 15 00:33:59 mycroft portsentry[603]: attackalert:
    SYN/Normal scan from host: 302.106.103.19/302.106.103.19 to TCP port: 111
Oct 15 00:33:59 mycroft portsentry[603]: attackalert:
    Host 302.106.103.19 has been blocked via wrappers with string:
    "ALL: 302.106.103.19"
Oct 15 00:33:59 mycroft portsentry[603]: attackalert:
    Host 302.106.103.19 has been blocked via dropped route using command:
    "/sbin/route add -host 302.106.103.19 reject"
```

常规指南：保持软件为最新

发布时间 :2007-01-28 12:03:54

因为所有软件都可能存在安全性漏洞，所以重要的是：只要获得包的安全性修正包就立刻安装。这是安全专家最常提出的一条建议，也是管理员新手们最常忽略的一条建议。不要吃过苦头才吸取教训 — 机器因为您忽视了使补丁程序保持最新而被人通过存在数年之久的后门侵入。

对于开放源码和封闭源码哪个更安全的争论非常激烈。迄今最好的结论是：管理正确时，两者都足够安全，这里的“管理”包括保持安全性补丁程序为最新！

有几个网站可以帮助保持软件为最新，并有助于提防已知的威胁。包括特别注意安全性的 CERT 和 SecurityFocus 的 BugTraq 列表，以及通常的软件更新站点（象 freshmeat.net）和分发版的主页。我们还将参考资料中重复这些 URL，不过安全性真的非常重要 — 如果还不熟悉这些站点的话，建议您现在就花几分钟访问头两个站点。

打印

打印

发布时间 :2007-01-28 12:04:37

这一章将介绍 Linux 上的经典 UNIX 打印系统（有时被称为 Berkeley LPD）的设置和使用。其它可用于 Linux 的系统则不在这里介绍；请参阅本教程末尾的参考资料一章以获取有关这些系统的信息。

物理上安装打印机超出了本教程的范围。当打印机正确连接后，则要安装打印假脱机程序守护程序，以使网络上的机器（包括运行假脱机程序的机器）能够将打印作业发送给打印机。

安装打印假脱机程序守护程序（lpd）

发布时间 :2007-01-28 12:05:11

最好的 LPD 打印假脱机程序之一是 LPRng。其安装方法取决于分发版；请参阅 LPI 102 系列第 1 部分以获取有关在 Red Hat 或 Debian 中安装软件包的详细信息。

安装打印假脱机程序守护程序（正式名称为行式打印机守护程序）以后，就可以从命令行运行。以普通用户身份登录，然后试着运行以下命令：

```
$ /usr/sbin/lpd --help
--X option form illegal
usage: lpd [-FV] [-D dbg] [-L log]
Options
-D dbg    - set debug level and flags
           Example: -D10,remote=5
           set debug level to 10, remote flag = 5
-F        - run in foreground, log to STDERR
           Example: -D10,remote=5
-L logfile - append log information to logfile
-V        - show version info
```

既然已安装了守护程序，则应确保将它设置为自动运行。

基本打印机设置（ /etc/printcap ）

发布时间 :2007-01-28 12:05:48

打印假脱机程序守护程序起着一种管道的作用。它接受来自各个打印客户机的打印作业，然后将这些作业传递到适当的打印机。当打印机忙时，这些作业就“假脱机”，等待打印机会。

当在本地打印机上打印时，该“管道”的两端都由配置文件 /etc/printcap（有时位于 /etc/lprng/printcap）描述。printcap（printer capabilities 的缩写）中的每一项描述一个打印假脱机文件：

```
$ more /etc/printcap
lp|Generic dot-matrix printer entry:\
    :lp=/dev/lp0:\
    :sd=/var/spool/lpd/lp:\
    :pl#66:\
    :pw#80:\
    :pc#150:\
    :mx#0:\
    :sh:
```

请注意：项的最后一行没有尾随的反斜杠（\）。

您的分发版可能有其它项，并且可能更复杂，但它们都大致有这样的形式。首先是项的名称 lp，随后是对这个假脱机文件较长的描述。关键字 / 值对 lp=/dev/lp0 指定将要打印假脱机文件中打印作业的 Linux 设备，而 sd 关键字则给出打印作业前存放它们的目录。

余下的关键字 / 值对则提供有关连接到 /dev/lp0 的打印机类型的详细信息。printcap 手册页对它们做了描述，稍后我们将介绍其中的一部分。

创建假脱机文件目录

发布时间 :2007-01-28 12:06:25

如果创建一个打印假脱机项，则需要确保假脱机文件的目录存在并且具有正确的权限。如果希望打印机守护程序（通常以用户 lp 的身份运行）能访问假脱机文件目录，则必须以 root 用户的身份运行以下命令：

```
# mkdir -p /var/spool/lpd/lp
# chown lp /var/spool/lpd/lp
# chmod 700 /var/spool/lpd/lp
# checkpc -f
# /etc/init.d/lprng restart
```

LPRng 包含一个用于检查 printcap 的有用工具。它甚至会为您设置假脱机文件目录（如果您忘了以手工方式这么做的话）：

```
# checkpc -f
```

最后，重新启动 lpd。为了使更改生效，每次更改 printcap 时都需要这么做。您可能需要使用 lpd 而不是 lprng：

```
# /etc/init.d/lprng restart
```

较老的 Berkeley 打印系统不包含 checkpc 工具，所以您必须亲自在各台打印机上打印测试页，以确保 printcap 和打印假脱机文件目录是正确的。

使用打印假脱机程序客户机

发布时间 :2007-01-28 12:07:05

打印假脱机程序本身带有几个客户机以便与服务器守护程序通信。使用最多的可能是 `lpr`，它仅仅将文件发送至服务器以在假脱机文件中排队然后打印。要尝试该程序，首先找到或制作一个小的样本文本文件。然后输入命令：

```
$ lpr sample.txt
```

若该命令起作用，则在屏幕上应该看不到响应，但打印机应该开始运行，而且应很快就能打印出该样本文本的硬拷贝。如果该命令执行的输出看起来不太正确，不必担心；稍后我们将设置过滤器，它应能确保所有种类的文件格式都能正确地打印。

可以用 `lpq` 命令检查打印假脱机文件队列中的打印作业列表。选项 `-P` 指定要显示的队列名称；如果不使用该选项，则 `lpq` 将使用缺省打印假脱机文件（就象 `lpr` 在前面所做的那样）：

```
$ lpq -Plp
Printer: lp@localhost 'Generic dot-matrix printer entry'
Queue: 1 printable job
Server: pid 1671 active
Unspooler: pid 1672 active
Rank  Owner/ID          Class Job Files      Size Time
active chouser@localhost+670  A   670 sample.txt      8 21:57:30
```

如果要停止打印作业，可以使用 `lprm` 命令。若一个作业花的时间过长，或者用户不小心发送了多份相同文件，则可能要执行该命令。只要从上面列出的 `lpq` 命令复制作业标识即可：

```
$ lprm chouser@localhost+670
Printer lp@localhost
checking perms 'chouser@localhost+670'
dequeued 'chouser@localhost+670'
```

可以使用交互式工具 `lpc` 对打印假脱机文件进行许多其它操作。请参阅 `lpc` 手册页以获取详细信息。

打印至远程 LPD 服务器

发布时间 :2007-01-28 12:07:46

即使本地机器上没有打印机，仍可以使用 lpd 跨越网络将打印作业发送至与别的机器相连的打印机。在客户机器上，可以向 /etc/printcap 添加一条看似本地打印机而实际上将打印作业路由至服务器机器的打印假脱机文件项。该项看起来应与下面相似：

```
farawaylp|Remote printer entry:\
:rm=faraway:\
:rp=lp:\
:sd=/var/spool/lpd/farawaylp:\
:mx#0:\
:sh:
```

这里我们希望执行打印作业的机器名称是 faraway，而那台机器上打印机的名称是 lp。假脱机文件目录 /var/spool/lpd/farawaylp 是打印作业在能够被发送至远程打印假脱机程序以前在本地保存的位置，而且在打印作业能发送到打印机以前，可能还要在远程打印假脱机程序处再次对它们进行假脱机处理。同样地，将需要创建这个假脱机文件目录并设置其权限：

```
# mkdir -p /var/spool/lpd/farawaylp
# chown lp /var/spool/lpd/farawaylp
# chmod 700 /var/spool/lpd/farawaylp
# checkpc -f
# /etc/init.d/lprng restart
```

在本地，我们将这个远程打印机命名为 farawaylp，因此我们可以将打印作业发送至 farawaylp：

```
$ lpr -Pfarawaylp sample.txt
```

打印至远程的 MS Windows 或 Samba 服务器

发布时间 :2007-01-28 12:08:33

感谢 Samba，打印至远程 Microsoft Windows 打印服务器只稍稍复杂一点。首先，添加本地 printcap 项：

```
smb|Remote windows printer:\
:if=/usr/bin/smbprint:\
:lp=/dev/null:\
:sd=/var/spool/lpd/smb:\
:mx#0:
```

这里新的关键字是 if，即输入过滤器。将它指向 smbprint 脚本将使打印作业被发送至 Windows 服务器而不是 lp 设备。我们仍必须列出打印守护程序用于锁定而使用的设备（此例中是 /dev/null）。但实际上将没有打印作业被发送到那里。

不要忘记创建假脱机文件目录

```
# mkdir -p /var/spool/lpd/smb
# chown lp /var/spool/lpd/smb
# chmod 700 /var/spool/lpd/smb
# checkpc -f
# /etc/init.d/lprng restart
```

在您喜爱的编辑器中，在上面命名的假脱机文件目录中创建一个 .config 文件。在本例中，该文件为 /var/spool/lpd/smb/.config：

```
server="WindowsServerName"
service="PrinterName"
password=""
user=""
```

调整这些值以指向希望使用的 Windows 机器和打印机名称，然后就可以使用以下命令：

```
$ lpr -Psmc sample.txt
```

smbprint 脚本应该与 Samba 一起提供，但该脚本并不包含在所有分发版中。如果在系统上找不到这个脚本，可以从 Samba HOWTO 获得。

Magicfilter

发布时间 :2007-01-28 12:09:03

迄今我们只尝试了打印文本文件，这还不是特别令人兴奋。通常，任何一台打印机只能打印一种格式的图形文件 — 然而我们希望打印的格式有几十种：Postscript、gif、jpeg 等等。名为 Magicfilter 的程序起着输入过滤器的作用，很象 smbprint 的所为。Magicfilter 并不转换文件格式，而是提供标识您正在尝试打印的文档类型的框架，然后通过适当的转换工具运行该文档：转换工具必须单独安装。到目前为止，最重要的转换工具是 Ghostscript，它可以将文件从 Postscript 格式转换成许多打印机的本机格式。

调整 printcap 以指向 Magicfilter

发布时间 :2007-01-28 12:09:45

安装这些工具后，只需再调整 printcap 一次即可。添加 if 关键字以指向与打印机配合的 Magicfilter：

```
lp|The EPSON Stylus Color 777 sitting under my desk:\
:if=/usr/share/magicfilter/StylusColor-777@720dpi-filter:\
:gqfilter:\
:lp=/dev/usb/lp0:\
:sd=/var/spool/lpd/lp:\
:pl#66:\
:pw#80:\
:pc#150:\
:mx#0:\
:sh:
```

在 /usr/share/magicfilter 中有用于许多不同打印机和打印机设置的过滤器，因此要确保使用的是适合您打印机的过滤器。每个过滤器都是一个文本文件，并且打印机的全称常常位于顶部。当您不清楚过滤器的文件名是什么时，这会对您有所帮助。

我还向这个 printcap 项添加了 gqfilter 标志，这样，即使打印作业来自远程打印机，也可使用输入过滤器。这种方法只适用于 LPRng。

因为较早的时候就设置了 /var/spool/lpd/lp 打印假脱机文件目录，所以我只需要检查 printcap 语法，然后重新启动服务器：

```
# checkpc -f
# /etc/init.d/lprng restart
```

现在您能够打印各种文档，包括 Postscript 文件。换句话说，现在可以从您喜爱的 Web 浏览器的菜单中选择“Print”来进行工作了。

Linux海量文章

海量 Linux 技术文章

发布时间 :2006-11-24 16:50:29

下面是linux技术文章快速入口。需要联网：

[Linux 技术交流](#)

<http://www.linuxdiyf.com/bbs/forum-3-1.html>

[Linux 应用](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=1>

[Linux 安装及学习指导](#)

<http://www.linuxdiyf.com/bbs/forum-45-1.html>

[Linux 系统安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=11>

[Linux 学习指导](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=12>

[Linux 软件安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=13>

[shell](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=3>

[Linux 壁纸](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=4>

[红旗](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=5>

[Redhat](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=6>

[SuSE](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=7>

Linux 认证

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=9>

Linux下载分享{酷件、书籍、视频分享 }

<http://www.linuxdiyf.com/bbs/forum-6-1.html>

服务器应用

<http://www.linuxdiyf.com/bbs/forum-7-1.html>

数据库应用

<http://www.linuxdiyf.com/bbs/forum-8-1.html>

Linux 编程与内核

<http://www.linuxdiyf.com/bbs/forum-9-1.html>

UniX 技术文章

<http://www.linuxdiyf.com/bbs/forum-32-1.html>

Linux 业界声音、新闻

<http://www.linuxdiyf.com/bbs/forum-11-1.html>

Linux 人才招聘信息

<http://www.linuxdiyf.com/bbs/forum-46-1.html>

网络转载，感谢原创作者！

制作：红联Linux论坛

祝您阅读愉快！

Linux培训系列

第八讲

将介绍安全 shell (ssh) 和相关工具，并演示如何使用和配置网络文件系统 (NFS) 版本 3 服务器和客户机。Linux培训系列结束。您将具备成为 Linux 系统管理员所必需的知识。

内容基础，语言简短简洁

红联Linux论坛是致力于Linux技术讨论的站点，目前网站收录的文章及教程基本能满足不同水平的朋友学习。

红联Linux门户：www.linux110.com

红联Linux论坛：www.linuxdiyf.com/bbs

下载:Linux电子书籍：

<http://www.linux286.com/linux/linuxdzsj.htm>

目录

安全 shell

- 安全 shell - 交互式登录
- 安全 shell
- 使用 ssh
- 启动 sshd
- 安全复制 (secure copy)
- 安全 shell 认证选项

NFS

- NFS
- NFS 基础
- NFS 的属性
- Linux 下的 NFS 版本 3
- 保护 NFS

设置 NFS

- 设置 NFS
- 准备好 /etc/exports
- 解决导出限制
- /etc/exports 文件
- 另一个 /etc/exports 文件
- 启动 NFS 3 服务器
- 更改导出选项
- 配置 NFS 客户机
- 启动 NFS 客户机服务
- 挂装导出的 NFS 文件系统
- 挂装导出文件*内部的*目录

Linux文章汇集

- 海量Linux技术文章

安全 shell

安全 shell - 交互式登录

发布时间 :2007-01-28 13:12:36

回顾以往，如果希望建立网络上的交互式登录通话，则要使用 telnet 或 rsh。然而，随着联网越来越普及，这些工具变得越来越不适宜，因为它们极不安全。

telnet 客户机与服务器之间传递的数据是未经加密的，因而可以被任何正在监听网络的人读取。不仅如此，认证（向服务器发送密码）是以明文形式执行的，这使得捕获网络数据以即时获取密码对于某些人成了小事一桩。事实上，使用网络嗅探器，某些人可以重建您的整个 telnet 会话，并能看到您在屏幕上看到的一切。

很明显，这些在设计时假定网络是安全和不可嗅探的工具已不应当今的分布式和公共网络。

安全 shell

发布时间 :2007-01-28 13:13:07

这就需要更好的解决方案，而该解决方案就是一个名为安全 shell（或 ssh）的工具。该工具最流行的现代版可以从 openssh 软件包获得，而该软件包几乎存在于每个 Linux 分发版中，更不用说许多其它的系统了。

ssh 与其不安全的“表亲”的显著区别在于：ssh 使用强加密对客户机和服务器之间的所有通信进行加密。通过这样做，监控客户机和服务器之间的通信就变得困难（甚至不可能）。用这样的方式，ssh 提供的服务正如宣传的那样——它是安全的 shell。事实上，ssh 具有极好的“全能”安全性——即使认证，也会利用加密和各种密钥交换策略，来确保用户的密码不会轻易被任何监控着网络上传输的数据的人截取。

在这个因特网普及化的时代，ssh 是使用 Linux 系统时增强网络安全性的有价值的工具。大多数了解安全性的网络管理员都不赞成——甚至根本不允许——在他们的系统上使用 telnet 和 rsh，因为 ssh 是非常有能力和安全的替代工具。

使用 ssh

发布时间 :2007-01-28 13:13:38

通常，大多数分发版的 openssh 软件包无需任何手工配置就可以使用。安装 openssh 后，将得到两个二进制文件。其中一个当然就是 ssh — 可以用来连接至任何运行着 sshd（安全 shell 服务器）的系统的安全 shell 客户机。要使用 ssh，通常要输入与下面相似的命令来启动会话：

```
$ ssh drobbins@otherbox
```

在上面，我指示 ssh 以“drobbins”用户帐户登录远程机器。和使用 telnet 一样，会提示您输入密码；密码输入后，就会向您提供新的远程系统上的登录会话。

启动 sshd

发布时间 :2007-01-28 13:14:11

如果允许 ssh 连接至您的机器，则需要启动 sshd 服务器。要启动 sshd 服务器，通常要使用与 openssh 包一起提供的 rc 脚本，输入如下内容：

```
# /etc/init.d/sshd start
```

或

```
# /etc/rc.d/init.d/sshd start
```

如有必要，可以通过修改 /etc/ssh/sshd_config 文件来调整 sshd 的配置选项。有关各种可用选项的更多信息，可输入 man sshd。

安全复制（ secure copy ）

发布时间 :2007-01-28 13:14:46

openssh 包本身还带有一个名为 scp（代表“secure copy”）的方便工具。可以使用这个命令在网络上各种系统之间安全地复制文件。例如，如果我希望将 ~/foo.txt 复制到我在远程机器的主目录，可以输入：

```
$ scp ~/foo.txt drobbins@remotebox
```

提示输入我在远程机器上的密码后，将执行复制。或者，如果我希望将远程机器 /tmp 目录下名为 bar.txt 的文件复制到我本地系统的当前工作目录，我会输入：

```
$ scp drobbins@remotebox:/tmp/bar.txt .
```

安全 shell 认证选项

发布时间 :2007-01-28 13:15:20

openssh 还有许多其它认证方法。使用得当的话，它们允许您与远程系统认证时无需每次连接都输入密码或密码短语。要学习有关如何做到这一点的更多知识，请参阅 developerWorks openssh 密钥管理文章（在本教程最后一章“参考资料”中列出）。

NFS

NFS

发布时间 :2007-01-28 13:15:57

网络文件系统（Network File System (NFS)）是一种允许透明文件共享的技术，这种共享出现在通过局域网（也就是 LAN）连接的 UNIX 和 Linux 系统之间。NFS 已出现了很长时间；它在 Linux 和 UNIX 世界里广为人知而且被广泛使用。特别地，NFS 常用于在网络上多台机器之间共享主目录，当用户登录至 LAN 上的一台机器（*任何一台*机器）时，这为他或她提供了一致的环境。由于 NFS，挂装远程文件系统树结构并将其完全集成到系统的本地文件系统成为可能。NFS 的透明性和成熟使它成为在 Linux 下进行网络文件共享的有用和流行的选择。

NFS 基础

发布时间 :2007-01-28 13:16:25

要使用 NFS 共享文件，首先需要设置 NFS 服务器。这个 NFS 服务器随后可以“导出”文件系统。当文件系统导出后，就意味着 LAN 上的其它系统可以访问它。然后，任何同样设置为 NFS 客户机的获授权的系统都可以用标准“mount”命令挂装这个导出的文件系统。挂装完成后，远程文件系统以与本地挂装的文件系统（象/mnt/cdrom）挂装后相同的方式“嫁接”。正从 NFS 服务器而不是磁盘读取所有的文件数据这一事实对于任何标准 Linux 应用程序都完全不是问题。一切正常。

NFS 的属性

发布时间 :2007-01-28 13:16:58

共享的 NFS 文件系统有许多有趣的属性。第一个“极好的属性”是 NFS 的无状态设计的结果。因为客户机对 NFS 服务器的访问本质上就是无状态的，所以 NFS 服务器重新引导而不会导致客户机应用程序崩溃或失败是有可能的。所有对远程 NFS 文件的访问将只是“暂停”，直到服务器恢复为在线为止。同样，因为 NFS 的无状态设计，NFS 服务器可以处理大量客户机，除了在网络上传送实际文件数据的开销以外，不会有任何其它开销。换句话说，NFS 性能取决于正在网络上传送的 NFS 数据的数量，而不是碰巧正在请求上述数据的客户机数量。

Linux 下的 NFS 版本 3

发布时间 :2007-01-28 13:17:50

注：不知现在发展版本到多少了，对这个感兴趣请先在网上搜索。

设置 NFS 时，强烈推荐使用 NFS 版本 3 而不是版本 2。版本 2 有一些严重的文件锁定问题，而且通常因中断某些应用程序而声名狼藉。相反，NFS 版本 3 非常出色而且健壮，并且能胜任它的工作。既然 Linux 2.2.18+ 支持 NFS 3 客户机和服务器，那么没有任何理由再考虑使用 NFS 2 了。

保护 NFS

发布时间 :2007-01-28 13:18:15

值得一提的是：NFS 版本 2 和 3 都有一些非常明显的安全性限制。它们被设计成在特殊的环境（安全、可信的 LAN）中使用。特别地，NFS 2 和 3 被设计成在只有管理员才被允许对机器进行“root”访问的 LAN 上使用。由于 NFS 2 和 NFS 3 的设计，如果恶意用户可以对您 LAN 上的机器进行“root”访问，则他或她将能够绕过 NFS 安全性，而且极有可能能够访问甚至修改 NFS 服务器上的文件，而这些用户通常是不能访问这些文件的。出于这个原因，不应该随便地部署 NFS。如果您打算在 LAN 上使用 NFS，很好 — 但请首先建立防火墙。要确保 LAN 之外的人不能访问 NFS 服务器。然后，确保内部 LAN 是相对安全的，并确保您完全清楚所有加入 LAN 的主机。一旦 LAN 的安全性经过彻底复查和（如果必要）改进，您就已经为安全地使用 NFS 做好了准备（有关这一点的更多信息，请参阅本教程系列的第 7 部分）。

设置 NFS

设置 NFS

发布时间 :2007-01-28 13:19:01

在 Linux 下设置 NFS

使用 NFS 3 的第一步是设置 NFS 3 服务器。选择将为 LAN 上其它机器提供文件服务的系统。在这台机器上，我们将需要在内核中启用 NFS 服务器支持。应该使用 2.2.18+ 内核（推荐 2.4+）以利用 NFS 3，它比 NFS 2 稳定得多。如果正在编译自己的定制内核，则进入 `/usr/src/linux` 目录并运行 `make menuconfig`。然后，选择“File systems”节，接着选择“Network File Systems”节，然后确保启用以下选项：

```
<*> NFS file system support
[*] Provide NFSv3 client support
<*> NFS server support
[*] Provide NFSv3 server support
```

准备好 /etc/exports

发布时间 :2007-01-28 13:19:31

接下来，编译并安装新内核，然后重新引导。系统现在将具有内置的 NFS 3 服务器和客户机支持。

既然我们的 NFS 服务器已在内核中支持 NFS，那么该是设置 /etc/exports 文件的时候了。/etc/exports 文件将描述可用于导出的本地文件系统，并描述哪些主机将能够访问这些文件系统，以及是将这些文件导出为读 / 写还是只读。还允许我们指定控制 NFS 行为的其它选项。

但在查看 /etc/exports 文件的格式以前，恰好有一个重大的实现警告！Linux 内核中的 NFS 实现只允许每个文件系统有一个本地目录的导出。这意味着：如果 /usr 和 /home 都在同一 ext3 文件系统上（例如，使用 /dev/hda6），那么在 /etc/exports 中不可能既有 /usr 导出行又有 /home 导出行。如果您试着添加这两行，则当重读 /etc/exports 文件（如果在 NFS 服务器启动并运行后输入 `exportfs -ra`，就会发生）时，您将看到如下错误：

```
sidekick:/home: Invalid argument
```

解决导出限制

发布时间 :2007-01-28 13:20:01

下面介绍如何解决这一问题。如果 /home 和 /usr 在同一底层本地文件系统上，则不能将两者都导出。因此只导出 /。NFS 客户机将能够毫无问题地通过 NFS 挂装 /home 和 /usr，而 NFS 服务器的 /etc/exports 文件现在是“合法的”，每个底层本地文件系统只包含一个导出行。既然您理解了 Linux NFS 的这一实现，我们就准备好查看 /etc/exports 的格式。

/etc/exports 文件

发布时间 :2007-01-28 13:20:34

理解 /etc/exports 格式的最好方法可能是查看一个快速示例。以下是我在 NFS 服务器上使用的一个简单的 /etc/exports 文件：

```
# /etc/exports: NFS file systems being exported. See exports(5).  
/ 192.168.1.9(rw,no_root_squash)  
/mnt/backup 192.168.1.9(rw,no_root_squash)
```

如您所见，我的 /etc/exports 文件的第一行是一条注释。在第二行，我选择根（“/”）文件系统用于导出。请注意：尽管这会导出“/”下的所有东西，但不会导出任何其它本地文件系统。例如，如果我的 NFS 服务器有一台挂装在 /mnt/cdrom 上的 CD-ROM，则 CDROM 的内容将是不可用的，除非在 /etc/exports 中显式地将其导出。现在，请注意我的 /etc/exports 文件中的第三行。在这一行，我导出 /mnt/backup；正如您可能猜到的那样，/mnt/backup 在与 / 不同的文件系统上，并且包含我系统的备份。每一行都有“192.168.1.9(rw,no_root_squash)”。该信息告诉 nfsd 只有 IP 地址为 192.168.1.9 的 NFS 客户机才可用使用这些导出文件。该信息还告诉 nfsd 使这些文件系统对于 NFS 客户机系统是可写和可读的，并指示 NFS 服务器允许远程 NFS 客户机允许超级用户帐户以获得对文件系统真正的“root”访问。

另一个 /etc/exports 文件

发布时间 :2007-01-28 13:21:06

下面是一个 /etc/exports，它导出的文件系统与前一页的相同，只不过它将使我的导出文件对 LAN 上所有的机器（从 192.168.1.1 到 192.168.1.254）都可用：

```
# /etc/exports: NFS file systems being exported. See exports(5).  
/ 192.168.1.1/24(rw,no_root_squash)  
/mnt/backup 192.168.1.1/24(rw,no_root_squash)
```

在这个样本 /etc/exports 文件中，我用主机掩码 /24 屏蔽掉我指定的 IP 地址中的最后八位。IP 地址说明和 “(” 之间不能有空格，这一点也很重要，否则 NFS 将错误地解释您的信息。而且，与您猜想的一样，除了 “rw” 和 “no_root_squash” 以外，还可以指定其它选项；请输入 “man exports” 以获得完整列表。

启动 NFS 3 服务器

发布时间 :2007-01-28 13:21:39

一旦 /etc/exports 配置完毕，就可以准备启动 NFS 服务器了。大多数分发版都有可用来启动 NFS 的 “ nfs ” 初始化脚本 — 请输入 /etc/init.d/nfs start 或 /etc/rc.d/init.d/nfs start 以使用它。一旦启动了 NFS，输入 rpcinfo 应该显示与下面相似的输出：

```
# rpcinfo -p
program vers proto  port
100000  2  tcp   111  portmapper
100000  2  udp   111  portmapper
100024  1  udp  32802  status
100024  1  tcp  46049  status
100011  1  udp   998  rquotad
100011  2  udp   998  rquotad
100003  2  udp   2049  nfs
100003  3  udp   2049  nfs
100003  2  tcp   2049  nfs
100003  3  tcp   2049  nfs
100021  1  udp  32804  nlockmgr
100021  3  udp  32804  nlockmgr
100021  4  udp  32804  nlockmgr
100021  1  tcp  48026  nlockmgr
100021  3  tcp  48026  nlockmgr
100021  4  tcp  48026  nlockmgr
100005  1  udp  32805  mountd
100005  1  tcp  39293  mountd
100005  2  udp  32805  mountd
100005  2  tcp  39293  mountd
100005  3  udp  32805  mountd
100005  3  tcp  39293  mountd
```

更改导出选项

发布时间 :2007-01-28 13:22:08

如果曾在 NFS 守护程序运行时更改了 /etc/exports 文件，只需输入 `exportfs -ra` 来应用更改。既然 NFS 服务器已经启动并运行，则可以准备好配置 NFS 客户机以使它们能挂装导出的文件系统。

配置 NFS 客户机

发布时间 :2007-01-28 13:22:42

只需确保启用了以下选项，NFS 3 客户机的内核配置与 NFS 服务器的内核配置基本类似：

<*> NFS file system support

[*] Provide NFSv3 client support

启动 NFS 客户机服务

发布时间 :2007-01-28 13:23:19

要启动适当的 NFS 客户机守护程序，通常可以使用名为 “nfslock” 或 “nfsmount” 的系统初始化脚本。通常，该脚本将启动 rpc.statd，它就是 NFS 3 客户机需要的一切 — rpc.statd 允许文件锁定以正确地工作。设置了所有的客户机服务后，在本地机器上运行 rpcinfo 将显示如下所示的输出：

```
# rpcinfo
program vers proto  port
100000  2  tcp   111  portmapper
100000  2  udp   111  portmapper
100024  1  udp  32768  status
100024  1  tcp  32768  status
```

也可以通过输入 rpcinfo -p myhost 从远程系统执行这一检查，如下所示：

```
# rpcinfo -p sidekick
program vers proto  port
100000  2  tcp   111  portmapper
100000  2  udp   111  portmapper
100024  1  udp  32768  status
100024  1  tcp  32768  status
```

挂装导出的 NFS 文件系统

发布时间 :2007-01-28 13:23:53

正确设置客户机和服务器后（并假设 NFS 服务器配置成允许客户机连接），就可以着手在客户机上挂装导出的 NFS 文件系统了。在本示例中，inventor 是 NFS 服务器，而 sidekick（IP 地址是 192.168.1.9）是 NFS 客户机。inventor 的 /etc/exports 文件包含与下面相似的一行，这一行允许来自 192.168.1 网络上任何机器的连接：

```
/ 192.168.1.1/24(rw,no_root_squash)
```

现在，以 root 用户身份登录至 sidekick 后，可以输入：

```
# mount inventor:/mnt/nfs
```

inventor 的根文件系统现在就被挂装在 sidekick 上的 /mnt/nfs；现在可以输入 `cd /mnt/nfs`，然后查看 inventor 的文件。请再次注意：如果 inventor 的 /home 树结构在另一个文件系统上，则 /mnt/nfs/home 将不会包含任何东西 — 访问那些数据需要另一个 mount（以及 inventor 的 /etc/exports 文件中的另一项）。

挂装导出文件*内部的* 目录

发布时间 :2007-01-28 13:24:26

请注意：inventor 的 / 192.168.1.1/24(rw,no_root_squash) 行还允许挂装 / 内部的目录。例如，如果 inventor 的 /usr 和 / 在同一个物理文件系统上，而您只对侧在 sidekick 上挂装 inventor 的 /usr 感兴趣，则可以输入：

```
# mount inventor:/usr /mnt/usr
```

inventor 的 /usr 树结构现在以 NFS 方式挂装至已经存在的 /mnt/usr 目录。再次强调：inventor 的 /etc/exports 文件无需显式地导出 /usr；它“免费”包含在“/”导出行中。

Linux文章汇集

海量 Linux 技术文章

发布时间 :2006-11-24 16:50:29

下面是linux技术文章快速入口。需要联网：

[Linux 技术交流](#)

<http://www.linuxdiyf.com/bbs/forum-3-1.html>

[Linux 应用](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=1>

[Linux 安装及学习指导](#)

<http://www.linuxdiyf.com/bbs/forum-45-1.html>

[Linux 系统安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=11>

[Linux 学习指导](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=12>

[Linux 软件安装](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=45&filter=type&typeid=13>

[shell](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=3>

[Linux 壁纸](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=4>

[红旗](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=5>

[Redhat](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=6>

[SuSE](#)

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=7>

Linux 认证

<http://www.linuxdiyf.com/bbs/forumdisplay.php?fid=3&filter=type&typeid=9>

Linux下载分享{酷件、书籍、视频分享 }

<http://www.linuxdiyf.com/bbs/forum-6-1.html>

服务器应用

<http://www.linuxdiyf.com/bbs/forum-7-1.html>

数据库应用

<http://www.linuxdiyf.com/bbs/forum-8-1.html>

Linux 编程与内核

<http://www.linuxdiyf.com/bbs/forum-9-1.html>

UniX 技术文章

<http://www.linuxdiyf.com/bbs/forum-32-1.html>

Linux 业界声音、新闻

<http://www.linuxdiyf.com/bbs/forum-11-1.html>

Linux 人才招聘信息

<http://www.linuxdiyf.com/bbs/forum-46-1.html>

网络转载，感谢原创作者！

制作：红联Linux论坛

祝您阅读愉快！