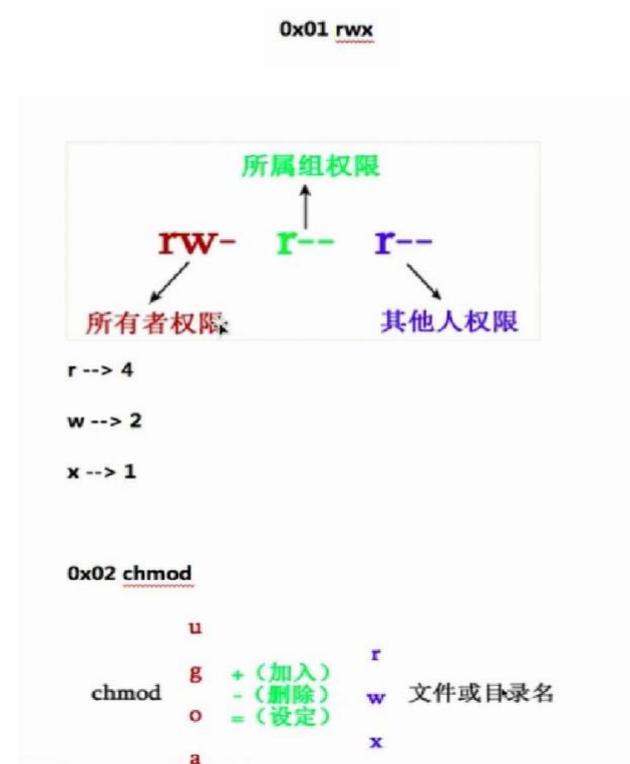


4.1.5 目录结构



4.1.6 文件属性及权限管理



0x03 umask

umask 默认权限确实由 4 个八进制数组成, 第 1 个数代表的是文件所具有的特殊权限, 后 3 位数字是真正的权限值

0x04 chattr(修订) lsattr(显示隐藏属性)

chattr 命令，专门用来修改文件或目录的隐藏属性，只有 root 用户可以使用。该命令的基本格式为：

```
[root@localhost ~]# chattr [+--] [属性] 文件或目录名
```

+表示给文件或目录添加属性，-表示移除文件或目录持有的某些属性，=表示给文件或目录设定一些属性。

表 1 列出了常用的一些属性及功能。

属性选项	功能
i	如果对文件设置 i 属性，那么不允许对文件进行删除、改名，也不能添加和修改数据； 如果对目录设置 i 属性，那么只能修改目录下文件中的数据，但不允许建立和删除文件；
a	如果对文件设置 a 属性，那么只能在文件中增加数据，但是不能删除和修改数据； 如果对目录设置 a 属性，那么只允许在目录中建立和修改文件，但是不允许删除文件；
u	设置此属性的文件或目录，在删除时，其内容会被保存，以保证后期能够恢复，常用来防止意外删除文件或目录。
s	和 u 相反，删除文件或目录时，会被彻底删除（直接从硬盘上删除，然后用 0 填充所占用的区域），不可恢复。

使用 chattr 命令配置文件或目录的隐藏属性后，可以使用 **lsattr 命令** 查看。

lsattr 命令，用于显示文件或目录的隐藏属性，其基本格式如下：

```
[root@localhost ~]# lsattr [选项] 文件或目录名
```

- 常用选项有以下 3 种：
- -a：后面不带文件或目录名，表示显示所有文件和目录（包括隐藏文件和目录）
 - -d：如果目标是目录，只会列出目录本身的隐藏属性，而不会列出所含文件或子目录的隐藏属性信息；
 - -R：和 -d 恰好相反，作用于目录时，会连同子目录的隐藏信息数据也一并显示出来。

0x05sudo 与 su

我们知道，使用 su 命令可以让普通用户切换到 root 身份去执行某些特权命令，但存在一些问题，比如说：

- 仅仅为了一个特权操作就直接赋予普通用户控制系统的完整权限；
- 当多人使用同一台主机时，如果大家都要使用 su 命令切换到 root 身份，那势必就需要 root 的密码，这就导致很多人都知道 root 的密码；

考虑到使用 su 命令可能对系统安装造成的隐患，最常见的解决方法是使用 sudo 命令，此命令也可以让你切换到其他用户的身份去执行命令。

相对于使用 su 命令还需要新切换用户的密码，**sudo 命令的运行只需要知道自己的密码即可**。甚至于，我们可以通过手动修改 sudo 的配置，使其无需任何密码即可运行。

sudo 命令默认只有 root 用户可以使用，该命令的基本格式为：

```
[root@localhost ~]# sudo [-b] [-u 新用户名称] 要执行的命令
```

常用的选项与参数：

- -b：将当前的命令放到背景中让系统自行运行，不对当前的 shell 环境产生影响。
- -u：指定以原欲切换的用户名，若无此选项则代表切换身份为 root。
- -i：此选项的用法为 sudo -i，用于指示当前用户可以用 sudo 执行那些命令。

【例 1】

```
[root@localhost ~]# grep sshd /etc/passwd
sshd:x:74:74:privilege-separated SSH:/var/empty/sshd:/sbin/nologin
[root@localhost ~]# sudo -u sshd touch /tmp/mysshd
[root@localhost ~]# ll /tmp/mysshd
-rw-r--r-- 1 sshd sshd 0 Feb 28 17:42 /tmp/mysshd
```

本例中，无法使用 su - sshd 的方式成功切换到 sshd 用户中，因为此用户的默认 shell 是 /sbin/nologin。这时就体现出 sudo 的优势，我们可以使用 sudo 以 sshd 的身份在 /tmp 目录下新建 mysshd 文件，可以看到，新建的 mysshd 文件的所有者确实是 sshd。

sudo apt-get install package[借助于 root 用户提升权限]



4.1.7 特殊权限

文件的特殊权限 SUID(set uid),SGID(set gid),SBIT(Sticky Bit)
 fhdl/(perm-400-o-pem-02000V-8 【查询系统中是否存在 s 位的程序

- 只有可执行文件才能设定 SetUID 权限，对目录设定 SUID 是无效的。
- 用户要对该文件拥有 x(执行)权限。
- 用户在执行该文件时，会以文件所有者的身份执行。
- SetUID 权限只在文件执行过程中有效，一旦执行完毕，身份的切换也随之消失

SUID 用在文件上，它的作用于是执行文件

SGID 用在目录上最多，继续上一级目录的权限

SBIT 它只对目录有效，对文件却是无效的，它的作用就是防止别人删除对方的资料。

详解：

大家都知道文件和目录的权限最常见的有三个.可读(r)可写(w).可执行(x)...它们的级别分别是 4..2.1.我们有时也会发现有些文件所属主的权限上带有一个 s 的标志位.目录的所属组上也带有 s 标志位.很多人不理解这是为什么.....下面我们举例来看一下....

```
#ls -l/usr/bin/passwd
-rwsr-xr-x1 root root 19876 Jul 17 2006 /usr/bin/passwd
```

我们有一个脚本名为 sum.sh.这个脚本的权限如下：

```
-rwsrwxr-x 1 root root 117 Feb 620:46 sum.sh
```

这个脚本的所属主和所属组为 root.我们当然可以以 root 的用户的身分执行它.我们还可以看见它的所属主的标志位上有个 s.其他人有读取和执行的权限.假如我们现在有个普通用户名为 redhat.现在切换到 redhat.执行此脚本.表面上我们看是执行成功了..其实我们是借助 root 用户的身分来执行它而不是 redhat..这就是 suid 的特性...

详解见文档

4.1.8 账户管理

Linux 用户和用户组

Linux UID 和 GID

Linux /etc/passwd

Linux /etc/shadow

Linux /etc/group

Linux /etc/gshadow

4.1.9 文件安装与卸载

0x01 源码文件

1)./configure[准备、检查、指定工作，加载参数等]

2)make 【编译】

3)make install 【安装】

4)make clear 【清除编译产生的临时文件】

0x02rpm 文件 【*.rpm】

rpm -ivh package

0x03yum 安装

yum install package

sudo apt-get install package 安装
sudo apt-get update 更新源
sudo apt-get install package 安装包

0x05 可执行文件
./package.run[最好是绝对路径]

0x06 卸载
主要通过 rpm 进行卸载，卸载过程如下

3.1.软件的卸载主要是使用rpm来进行的。卸载软件首先要知道软件包在系统中注册的名称。键入命令：

```
1 #查询到当前系统中安装的所有软件包
2 rpm -q -a
3
4 #查询pcre安装的软件包名称
5 rpm -q -a | grep "pcre"
6
7 #查询rpm包安装到哪里
8 rpm -ql pcre-7.8-7.el6.x86_64
```

```
[root@localhost ~]# rpm -q -a | grep "pcre"
pcre-7.8-7.el6.x86_64
[root@localhost ~]# rpm -ql pcre-7.8-7.el6.x86_64
/lib64/libpcre.so.0
/usr/man/pcreprep
/usr/bin/pcretest
/usr/lib64/libpcrecpp.so.0
/usr/lib64/libpcrepp.so.0.0.0
/usr/lib64/libpcreposix.so.0
/usr/lib64/libpcreposix.so.0.0.0
/usr/share/doc/pcre-7.8
/usr/share/doc/pcre-7.8/README
/usr/share/doc/pcre-7.8/COPYING
/usr/share/doc/pcre-7.8/ChangeLog
/usr/share/doc/pcre-7.8/LICENSE
/usr/share/doc/pcre-7.8/NEWS
/usr/share/doc/pcre-7.8/README
/usr/share/man/man1/pcre-config.1.gz
/usr/share/man/man1/pcreprep.1.gz
/usr/share/man/man1/pcretest.1.gz
```

3.2.确定了要卸载的软件名称，就可以开始实际卸载该软件了

```
1 #卸载软件，参数e的作用是使rpm进入卸载模式，对名为[package name]的软件包进行卸载
2 rpm -e [package name]
3
4 #由于系统中各个软件包之间相互有依赖关系，如果因存在依赖关系而不能卸载，rpm将给予提示并停止卸载。你可以使用如下的命令来忽略依赖
5 rpm -e [package name] --nodeps
6
7 #进入安装包卸载
8 make uninstall
```

```
[root@localhost ~]# cd pcre-8.40
[root@localhost pcre-8.40]# make uninstall
{ cd '/usr/local/bin' && rm -f pcretest pcregrep }
{ cd '/usr/local/bin' && rm -f pcre-config }
{ cd '/usr/local/include' && rm -f pcreposix.h pcrepp.h pcre_scanner.h }
/bin/sh ./libtool --mode=uninstall rm -f '/usr/local/lib/libpcre.la'
/bin/sh ./libtool --mode=uninstall rm -f '/usr/local/lib/libpcreposix.la'
/bin/sh ./libtool --mode=uninstall rm -f '/usr/local/lib/libpcrecpp.la'
{ cd '/usr/local/share/man/man1' && rm -f pcre-config.1 pcreprep.1 pcretest.1 }
{ cd '/usr/local/share/man/man3' && rm -f pcre.3 pcre16.3 pcre32.3 pcre_assign_jit_stack.3 pcre_
sec.3 pcre_exec.3 pcre_free_study.3 pcre_free_substring.3 pcre_free_substring_list.3 pcre_fullim
ting.3 pcre_get_substring_list.3 pcre_jit_exec.3 pcre_jit_stack_alloc.3 pcre_jit_stack_free.3 pcr
e_jit_stack_get.3 pcre_jit_stack_get_size.3 pcre_jit_stack_set_size.3 pcre_malloc.3 pcre_malloc_
pcreposix.3 pcrepp_compile.3 pcrepp_sample.3 pcrepp_stack.3 pcrepp_text.3 pcrepp_unicode.3 pcrepp_
3 }
{ cd '/usr/local/include' && rm -f pcre.h pcrepp.h pcre_stringpiece.h }
{ cd '/usr/local/lib/pkgconfig' && rm -f libpcre.pc libpcreposix.pc libpcrecpp.pc }
```