实训9：用户身份及文件权限

实训任务：

1. **任务书中的练习命令**
2. **完成课后练习**

一、实训基础知识

Linux是一个多用户、多任务的操作系统，具有很好的稳定性与安全性，在幕后保障Linux系统安全的则是一系列复杂的配置工作。本章将详细讲解文件的所有者、所属组以及其他人可对文件进行的读（r）、写（w）、执行（x）等操作，以及如何在Linux系统中添加、删除、修改用户账户信息。

我们还可以使用SUID、SGID与SBIT特殊权限更加灵活地设置系统权限，来弥补对文件设置一般操作权限时所带来的不足。隐藏权限能够给系统增加一层隐形的防护层，让黑客最多只能查看关键日志信息，而不能篡改或删除。而文件访问控制列表（Access Control List，ACL）可以进一步让单一用户、用户组对单一文件或目录进行特殊的权限设置，让文件具有能满足工作需求的最小权限。

最后还会讲解如何使用su[命令](https://www.linuxcool.com/)与sudo服务让普通用户具备管理员的权限，不仅能够满足日常工作需求，还可以确保系统的安全性。

**5.1 用户身份与能力**

受到上世纪70年代计算机发展的影响，Linux系统的设计初衷之一就是为了满足多个用户同时工作的需求，因此必须具备很好的安全性，尤其是不能因为一两个服务出错而影响到整台服务器。在第一章学习安装Linux操作系统时，特别要求设置root管理员的密码，这个root管理员就是存在于所有类UNIX系统中的超级用户。它拥有最高的系统所有权，能够管理系统的各项功能，如添加/删除用户、启动/关闭服务进程、开启/禁用硬件设备等。虽然以root管理员的身份工作时不会受到系统的限制，但俗语讲“能力越大，责任就越大”，因此一旦使用这个高能的root管理员权限执行了错误的[命令](https://www.linuxcool.com/)可能会直接毁掉整个系统。使用与否，确实需要好好权衡一下。

在学习时是否要使用root管理员权限来控制整个系统呢？面对这个问题，网络上有很多文章建议以普通用户的身份来操作—这是一个更安全也更“无责任”的回答。今天，就要冒天下之大不韪给出自己的心得—强烈推荐大家在学习时使用root管理员权限！

这种为root管理员正名的决绝态度在网络中应该还是很少见的，我之所以力荐root管理员权限，原因很简单。因为在Linux的学习过程中如果使用普通用户身份进行操作，则在配置服务之后出现错误时很难判断是系统自身的问题还是因为权限不足而导致的；这无疑会给大家的学习过程徒增坎坷。更何况我们的实验环境是使用VMware虚拟机软件搭建的，能够将安装好的系统设置为一次快照，这即便系统彻底崩溃了，您也可以在5秒钟的时间内快速还原出一台全新的系统，而不用担心数据丢失。

推荐每位学生使用root管理员权限来学习Linux系统，等到工作时再根据生产环境决定使用哪个用户权限；这些仅与选择相关，而非技术性问题。

另外，很多图书或培训机构的老师会讲到，Linux系统中的管理员就是root。这其实是错误的，Linux系统的管理员之所以是root，并不是因为它的名字叫root，而是因为该用户的身份号码即UID（User IDentification）的数值为0。在Linux系统中，UID就相当于身份证号码一样具有唯一性，因此可通过用户的UID值来判断用户身份。在RHEL 8系统中，用户身份有下面这些。

管理员UID为0：系统的管理员用户。

系统用户UID为1～999： Linux系统为了避免因某个服务程序出现漏洞而被黑客提权至整台服务器，默认服务程序会有独立的系统用户负责运行，进而有效控制被破坏范围。

普通用户UID从1000开始：是由管理员创建的用于日常工作的用户。

需要注意的是，UID是不能冲突的，而且管理员创建的普通用户的UID默认是从1000开始的（即使前面有闲置的号码）。

为了方便管理属于同一组的用户，Linux系统中还引入了用户组的概念。通过使用用户组号码（GID，Group IDentification），可以把多个用户加入到同一个组中，从而方便为组中的用户统一规划权限或指定任务。假设有一个公司中有多个部门，每个部门中又有很多员工，如果只想让员工访问本部门内的资源，则可以针对部门而非具体的员工来设置权限。例如，通过对技术部门设置权限，使得只有技术部门的员工可以访问公司的数据库信息等。

另外，在Linux系统中创建每个用户时，将自动创建一个与其同名的基本用户组，而且这个基本用户组只有该用户一个人。如果该用户以后被归纳入其他用户组，则这个其他用户组称之为扩展用户组。一个用户只有一个基本用户组，但是可以有多个扩展用户组，从而满足日常的工作需要。

**Tips**

基本组就像是原生家庭，这是随创建账号（出生）就自动完成的；而扩展组则像工作单位，为了完成工作，就要加入到各个不同的群体中，这是需要手动添加的。

**1.  id命令**

id命令用于显示用户详细信息，语法格式为：“id 用户名”。

这是一个要在创建用户前仔细学习的命令，它能够简单轻便的查看到用户的基本信息，例如用户ID、基本组与扩展组GID，便于我们判别某个用户是否已经存在，以及查看到相关信息。

查看一个名称为linuxprobe的用户信息：**练习命令**

[root@linuxprobe ~]# id linuxprobe

uid=1000(linuxprobe) gid=1000(linuxprobe) groups=1000(linuxprobe)

**2.  useradd命令**

useradd命令用于创建新的用户账户，语法格式为：“useradd [参数] 用户名”。

可以使用useradd命令创建用户账户。使用该命令创建用户账户时，默认的用户家目录会被存放在/home目录中，默认的[Shell](https://www.linuxcool.com/)解释器为/bin/bash，而且默认会创建一个与该用户同名的基本用户组。这些默认设置可以根据表5-1中的useradd命令参数自行修改。

表5-1                                       useradd命令中的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -d | 指定用户的家目录（默认为/home/username） |
| -e | 账户的到期时间，格式为YYYY-MM-DD. |
| -u | 指定该用户的默认UID |
| -g | 指定一个初始的用户基本组（必须已存在） |
| -G | 指定一个或多个扩展用户组 |
| -N | 不创建与用户同名的基本用户组 |
| -s | 指定该用户的默认Shell解释器 |

创建一个名称为linuxcool的用户，并使用id命令确认信息：**练习命令**

[root@linuxprobe ~]# useradd linuxcool

[root@linuxprobe ~]# id linuxcool

uid=1001(linuxcool) gid=1001(linuxcool) groups=1001(linuxcool)

下面我们提高难度再创建一个普通用户并指定家目录的路径、用户的UID以及Shell解释器。在下面的命令中，请注意/sbin/nologin，它是终端解释器中的一员，与Bash解释器有着天壤之别。一旦用户的解释器被设置为nologin，则代表该用户不能登录到系统中：**练习命令**

[root@linuxprobe ~]# useradd -d /home/linux -u 8888 -s /sbin/nologin linuxdown

[root@linuxprobe ~]# id linuxdown

uid=8888(linuxdown) gid=8888(linuxdown) groups=8888(linuxdown)

**3.  groupadd命令**

groupadd命令用于创建新的用户组，语法格式为：“groupadd [参数] 群组名”。

为了能够更加高效地指派系统中各个用户的权限，在工作中常常会把几个用户加入到同一个组里面，这样便可以针对一类用户统一安排权限。例如工作中成立一个部门组，有新的同事加入时就把他的账号添加到部门组中，权限自动就跟其他人一模一样了，省去了一系列繁琐的操作。

创建用户组的步骤非常简单，例如使用如下命令创建一个用户组ronny：**练习命令**

[root@linuxprobe ~]# groupadd ronny

**4.  usermod命令**

usermod命令用于修改用户的属性，英文全称为：“user modify”，语法格式为：“ usermod [参数] 用户名”。

前文曾反复强调，Linux系统中的一切都是文件，因此在系统中创建用户也就是修改配置文件的过程。用户的信息保存在/etc/passwd文件中，可以直接用文本编辑器来修改其中的用户参数项目，也可以用usermod命令修改已经创建的用户信息，诸如用户的UID、基本/扩展用户组、默认终端等。usermod命令的参数以及作用如表5-2所示。

表5-2                                            usermod命令中的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -c | 填写用户账户的备注信息 |
| -d -m | 参数-m与参数-d连用，可重新指定用户的家目录并自动把旧的数据转移过去 |
| -e | 账户的到期时间，格式为YYYY-MM-DD |
| -g | 变更所属用户组 |
| -G | 变更扩展用户组 |
| -L | 锁定用户禁止其登录系统 |
| -U | 解锁用户，允许其登录系统 |
| -s | 变更默认终端 |
| -u | 修改用户的UID |

大家不要被这么多参数吓坏了，再来看一下账户linuxprobe的默认信息：**练习命令**

[root@linuxprobe ~]# id linuxprobe

uid=1000(linuxprobe) gid=1000(linuxprobe) groups=1000(linuxprobe)

然后将用户linuxprobe加入到root用户组中，这样扩展组列表中则会出现root用户组的字样，而基本组不会受到影响：**练习命令**

[root@linuxprobe ~]# usermod -G root linuxprobe

[root@linuxprobe ~]# id linuxprobe

uid=1000(linuxprobe) gid=1000(linuxprobe) groups=1000(linuxprobe),**0(root)**

再来试试用-u参数修改linuxprobe用户的UID号码值：**练习命令**

[root@linuxprobe ~]# usermod -u 8888 linuxprobe

[root@linuxprobe ~]# id linuxprobe

uid=**8888**(linuxprobe) gid=1000(linuxprobe) groups=1000(linuxprobe),0(root)

除此之外，同学们最关心的肯定是如果把用户的解释器终端由默认的/bin/bash修改为/sbin/nologin后会有什么样的效果呢？那来试试吧：**练习命令**

[root@linuxprobe ~]# usermod -s /sbin/nologin linuxprobe

[root@linuxprobe ~]# su - linuxprobe

This account is currently not available.

效果很直观吧，将用户的终端设置成/sbin/nologin后马上就不能被登录了（切换身份也不行），但这个用户依然可以被某个服务所调用，管理某个具体的服务。这样的好处是当骇客通过这个服务入侵成功，破坏的范围也仅仅局限于这个特定的服务，不能使用这个用户身份登录到整台服务器上，尽可能的把损失最小化。

**5.  passwd命令**

passwd命令用于修改用户的密码、过期时间等信息，英文全称为：“password”，语法格式为：“ passwd [参数] 用户名”。

普通用户只能使用passwd命令修改自己的系统密码，而root管理员则有权限修改其他所有人的密码。更酷的是，root管理员在Linux系统中修改自己或他人的密码时不需要验证旧密码，这一点特别方便。既然root管理员能够修改其他用户的密码，就表示完全拥有该用户的管理权限。passwd命令中可用的参数以及作用如表5-3所示。

表5-3                                           passwd命令中的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -l | 锁定用户，禁止其登录 |
| -u | 解除锁定，允许用户登录 |
| --stdin | 允许通过标准输入修改用户密码，如echo "NewPassWord" | passwd --stdin Username |
| -d | 使该用户可用空密码登录系统 |
| -e | 强制用户在下次登录时修改密码 |
| -S | 显示用户的密码是否被锁定，以及密码所采用的加密算法名称 |

修改自己的密码只需要输入命令后敲击回车即可：**练习命令**

[root@linuxprobe ~]# passwd

Changing password for user root.

New password: **此处输入密码值**

Retype new password: **再次输入进行确认**

passwd: all authentication tokens updated successfully.

而修改其他人的密码，首先需要检查当前是否为root管理员权限，然后命令后指定要修改那位用户的名称：**练习命令**

[root@linuxprobe ~]# passwd linuxprobe

Changing password for user linuxprobe.

New password:**此处输入密码值**

Retype new password: **再次输入进行确认**

passwd: all authentication tokens updated successfully.

假设您有位同事正在度假，而且假期很长，那么可以使用passwd命令禁止该用户登录系统，等假期结束回归工作岗位时，再使用该命令允许用户登录系统，而不是将其删除。这样既保证了这段时间内系统的安全，也避免了频繁添加、删除用户带来的麻烦：**练习命令**

[root@linuxprobe ~]# passwd -l linuxprobe

Locking password for user linuxprobe.

passwd: Success

[root@linuxprobe ~]# passwd -S linuxprobe

linuxprobe LK 1969-12-31 0 99999 7 -1 (**Password locked.**)

需要解锁时记得也要用管理员的身份，否则普通用户如果有锁定权限的话，系统肯定会乱成一锅粥：**练习命令**

[root@linuxprobe ~]# passwd -u linuxprobe

Unlocking password for user linuxprobe.

passwd: Success

[root@linuxprobe ~]# passwd -S linuxprobe

linuxprobe PS 1969-12-31 0 99999 7 -1 (**Password set, SHA512 crypt.**)

**6.  userdel命令**

userdel命令用于删除已有的用户账户，英文全称为：“user delete”，语法格式为：“ userdel [参数] 用户名”。

如果确认某位用户后续不再会登录到系统中，则可以通过userdel命令删除该用户的所有信息。在执行删除操作时，该用户的家目录默认会保留下来，此时可以使用-r参数将其删除。userdel命令的参数以及作用如表5-4所示。

表5-4                                              userdel命令中的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -f | 强制删除用户 |
| -r | 同时删除用户及用户家目录 |

正常情况下删除一个用户时会建议保留他的家目录数据，避免有什么重要的数据被误删除，所以不用加参数，写清要删除的用户名称就行：**练习命令**

[root@linuxprobe ~]# userdel linuxprobe

[root@linuxprobe ~]# id linuxprobe

id: linuxprobe: no such user

此时该用户虽然被删除，但家目录数据会继续被存放在/home目录中，确认未来无需使用时再手动删除掉它：**练习命令**

[root@linuxprobe ~]# cd /home

[root@linuxprobe home]# ls

linuxprobe linuxcool linuxdown

[root@linuxprobe home]# rm -rf linuxprobe

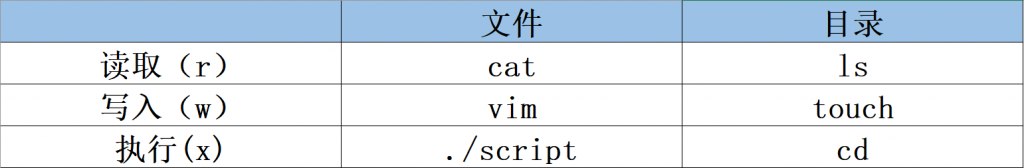
[root@linuxprobe home]# ls

linuxcool linuxdown

**5.2 文件权限与归属**

在Linux系统中，每个文件都有归属的所有者和所有组，并且规定了文件的所有者、所有组以及其他人对文件所拥有的读（r）、写（w）、执行（x）等权限。对于一般文件来说，权限比较容易理解：“可读”表示能够读取文件的实际内容；“可写”表示能够编辑、新增、修改、删除文件的实际内容；“可执行”则表示能够运行一个[脚本](https://www.linuxcool.com/)程序。但是，对于目录文件来说，理解其权限设置来就不那么容易了。很多资深Linux用户其实也没有真正搞明白。对目录文件来说，“可读”表示能够读取目录内的文件列表；“可写”表示能够在目录内新增、删除、重命名文件；而“可执行”则表示能够进入该目录。可以参考表格5-5，帮助同学们理解当文件和目录被设置rwx权限后，能够被执行的命令的区别：

表5-5                                              读写执行权限对于文件与目录可执行命令的区别



文件的读、写、执行权限英文全称分别是read、write、execute，可以简写为r、w、x，亦可分别用数字4、2、1来表示，文件所有者，所属组及其他用户权限之间无关联，如表5-6所示。

表5-6                                              文件权限的字符与数字表示



文件权限的数字法表示基于字符（rwx）的权限计算而来，其目的是简化权限的表示方式。例如，若某个文件的权限为**7**则代表可读、可写、可执行（4+2+1）；若权限为6则代表可读、可写（4+2）。我们来看这样一个例子。现在有这样一个文件，其所有者拥有可读、可写、可执行的权限，其文件所属组拥有可读、可写的权限；而且其他人只有可读的权限。那么，这个文件的权限就是rwxrw-r--，数字法表示即为764。不过大家千万别再将这三个数字相加，计算出7+6+4=17的结果，这是小学的数学加减法，不是Linux系统的权限数字表示法，三者之间没有互通关系。

以rw-r-x-w-权限为例进行讲解，要想转换成数字法，首先要进行各个位之上的数字替代，如图5-1所示。



图5-1 字符与数字权限转换示意图

减号是占位符，代表这里没有权限，数字法用0表示，也就是说rw-转换后是420，r-x转换后是401，-w-转换后是020，三组数字之间每组数字进行相加后得出652便是转换后的数字法权限。

而数字法转回到字符权限相比来说就有些难度了，以652权限为例进行讲解。首先数字6是由4+2得到的，不可能是4+1+1，因为每个权限只能占一位，所以数字5则是4+1得到的，2便是本身，没有权限即是空值0，可以按照表5-6所示的格式填写进去后得到420401020这样一串数字。有了这些信息就好办了，如图5-2所示，把他们都转换回来。

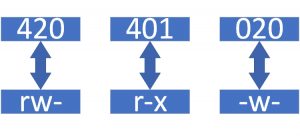


图5-2 数字与字符权限转换示意图

心中牢记——文件的所有者、所有组和其他人权限之间无关联，不要写成rrwwx----的样子，一定要把rwx权限位对应到正确的位置，写成rw-r-x-w-。

Linux系统的文件权限相当复杂，但是用途很广泛，建议大家把它彻底搞清楚之后再学习下一节的内容。现在来练习一下。请各位读者分别计算数字表示法764、652、153、731所对应的字符表示法，然后再把rwxrw-r--、rw--w--wx、rw-r--r--转换成数字表示法。

下面我们利用上文讲解的知识，一起分析图5-3中所示的文件信息。



图5-3 通过ls命令查看到的文件属性信息

在图5-3中，包含了文件的类型、访问权限、所有者（属主）、所属组（属组）、占用的磁盘大小、最后修改时间和文件名称等信息。通过分析可知，该文件的类型为普通文件，所有者权限为可读、可写（rw-），所属组权限为可读（r--），除此以外的其他人也只有可读权限（r--），文件的磁盘占用大小是34298字节，最近一次的修改时间为4月2日的凌晨23分，文件的名称为install.log。

其中排在权限前面的减号（-）是文件类型，新手经常会把它跟无权限混淆。尽管在Linux系统中一切都是文件，但是不同的文件由于作用不同，因此类型也不尽相同，有一点点像Windows系统的后缀名。常见的文件类型包括有：普通文件（-）、目录文件（d）、链接文件（l）、管道文件（p）、块设备文件（b）以及字符设备文件（c）。

普通文件的范围特别广泛，比如纯文本信息、服务配置信息、日志信息以及Shell[脚本](https://www.linuxcool.com/)等等都包含在内，所以几乎在每个目录下都能看到普通文件（-）和目录文件（d）的身影。块设备文件（b）和字符设备文件（c）一般是指硬件设备，比如鼠标、键盘、光驱、硬盘等等都是设备文件，主要集中在/dev/目录中最为常见，不过其实很少会对鼠标键盘进行硬件级别的管理吧~

**5.3 文件的特殊权限**

在复杂多变的生产环境中，单纯设置文件的rwx权限无法满足我们对安全和灵活性的需求，因此便有了SUID、SGID与SBIT的特殊权限位。这是一种对文件权限进行设置的特殊功能，可以与一般权限同时使用，以弥补一般权限不能实现的功能。

下面具体解释这3个特殊权限位的功能以及用法。

**1.  SUID**

SUID是一种对二进制程序进行设置的特殊权限，能够让二进制程序的执行者临时拥有属主的权限（仅对拥有执行权限的二进制程序有效）。例如，所有用户都可以执行passwd命令来修改自己的用户密码，而用户密码保存在/etc/shadow文件中。仔细查看这个文件就会发现它的默认权限是000，也就是说除了root管理员以外，所有用户都没有查看或编辑该文件的权限。但是，在使用passwd命令时如果加上SUID特殊权限位，就可让普通用户临时获得程序所有者的身份，把变更的密码信息写入到shadow文件中。这很像在古装剧中见到的手持尚方宝剑的钦差大臣，他手持的尚方宝剑代表的是皇上的权威，因此可以惩戒贪官，但这并不意味着他永久成为了皇上。因此这只是一种有条件的、临时的特殊权限授权方法。

查看passwd命令属性时发现所有者的权限由rwx变成了rws，其中x改变成s就意味着该文件被赋予了SUID权限。另外有读者会好奇，那么如果原本的权限是rw-呢？如果原先权限位上没有x执行权限，那么被赋予特殊权限后将变成大写的S。**练习命令**

[root@linuxprobe ~]# ls -l /etc/shadow

**----------**. 1 root root 1312 Jul 21 05:08 /etc/shadow

[root@linuxprobe ~]# ls -l /bin/passwd

-rw**s**r-xr-x. 1 root root 34512 Aug 13 2018 /bin/passwd

**Tips**

醒目的红色提醒告诫着用户一定要小心这个权限，因为一旦某个命令文件被设置上了SUID权限，那么就意味着凡是执行的人都可以临时获取到更高的权限，千万不要设置到vim、cat、rm等命令上面！！！

**2.  SGID**

SGID特殊权限有两种应用场景，当对二进制程序进行设置时，能够让执行者临时获取到文件所有组的权限；而对目录进行设置时，则是让目录内新创建的文件自动继承该目录原有用户组的名称。

第一种功能是参考SUID而设计的，不同点在于执行程序的用户获取的不再是文件所有者的临时权限，而是获取到文件所属组的权限。举例来说，在早期的Linux系统中，/dev/kmem是一个字符设备文件，用于存储内核程序要访问的数据，权限为：

**cr--r-----**   1 root system 2,  1 Feb 11 2017  kmem

大家看出问题了吗？除了root管理员或属于system组成员外，所有用户都没有读取该文件的权限。由于在平时我们需要查看系统的进程状态，为了能够获取到进程的状态信息，可在用于查看系统进程状态的ps命令文件上增加SGID特殊权限位。查看ps命令文件的属性信息：

-r-xr-**s**r-x   1 bin system 59346 Feb 11 2017  ps

这样一来，由于ps命令被增加了SGID特殊权限位，所以当用户执行该命令时，也就临时获取到了system用户组的权限，从而顺利地读取到了设备文件。

前文提到，每个文件都有其归属的所有者和所属组，当创建或传送一个文件后，这个文件就会自动归属于执行这个操作的用户（即该用户是文件的所有者）。如果现在需要在一个部门内设置共享目录，让部门内的所有人员都能够读取目录中的内容，那么就可以创建部门共享目录后，在该目录上设置SGID特殊权限位。这样，部门内的任何人员在里面创建的任何文件都会归属于该目录的所属组，而不再是自己的基本用户组。此时，用到的就是SGID的第二个功能，即在某个目录中创建的文件自动继承该目录的用户组（只可以对目录进行设置）。**练习命令**

[root@linuxprobe ~]# cd /tmp

[root@linuxprobe tmp]# mkdir testdir

[root@linuxprobe tmp]# ls -ald testdir

drwxr-xr-x. 2 root root 6 Oct 27 23:44 testdir

[root@linuxprobe tmp]# chmod -R 777 testdir

[root@linuxprobe tmp]# chmod -R g+s testdir

[root@linuxprobe tmp]# ls -ald testdir

**drwxrwsrwx**. 2 root root 6 Oct 27 23:44 testdir

在使用上述命令设置好目录的777权限（确保普通用户可以向其中写入文件），并为该目录设置了SGID特殊权限位后，就可以切换至一个普通用户，然后尝试在该目录中创建文件，并查看新创建的文件是否会继承新创建的文件所在的目录的所属组名称：**练习命令**

[root@linuxprobe tmp]# su - linuxprobe

[linuxprobe@linuxprobe ~]$ cd /tmp/testdir

[linuxprobe@linuxprobe testdir]$ echo "linuxprobe.com" > test

[linuxprobe@linuxprobe testdir]$ ls -al test

-rw-rw-r--. 1 linuxprobe **root** 15 Oct 27 23:47 test

除了上面提到的SGID的这两个功能，再介绍两个与本小节内容相关的命令：chmod和chown。

chmod命令用于设置文件的一般权限及特殊权限，英文全称为：“change mode”，语法格式为：“chmod [参数] 文件名”。

这是一个与日常设置文件权限强相关的命令，例如要把一个文件的权限设置成其所有者可读可写可执行、所属组可读可写、其他人没有任何权限，则相应的字符法表示为rwxrw----，其对应的数字法表示为760。**练习命令**

[root@linuxprobe ~]# ls -l anaconda-ks.cfg

-rw-------. 1 root root 1407 Jul 21 05:09 anaconda-ks.cfg

[root@linuxprobe ~]# chmod 760 anaconda-ks.cfg

[root@linuxprobe ~]# ls -l anaconda-ks.cfg

-rwxrw----. 1 root root 1407 Jul 21 05:09 anaconda-ks.cfg

chown命令用于设置文件的所有者和所有组，英文全称为：“change own”，语法格式为：“chown 所有者:所有组 文件名”。

chmod和chown命令是用于修改文件属性和权限的最常用命令，它们还有一个特别的共性，就是针对目录进行操作时需要加上大写参数-R来表示递归操作，即对目录内所有的文件进行整体操作。

使用“所有者:所有组”的格式轻松把刚刚那个文件的所属信息修改一下，变更后的效果如下：**练习命令**

[root@linuxprobe ~]# chown linuxprobe:linuxprobe anaconda-ks.cfg

[root@linuxprobe ~]# ls -l anaconda-ks.cfg

-rwxrw----. 1 **linuxprobe linuxprobe** 1407 Jul 21 05:09 anaconda-ks.cfg

**3.  SBIT**

现在，大学里的很多老师都要求学生将作业上传到服务器的特定共享目录中，但总是有几个“破坏分子”喜欢删除其他同学的作业，这时就要设置SBIT（Sticky Bit）特殊权限位了（也可以称之为特殊权限位之粘滞位）。SBIT特殊权限位可确保用户只能删除自己的文件，而不能删除其他用户的文件。换句话说，当对某个目录设置了SBIT粘滞位权限后，那么该目录中的文件就只能被其所有者执行删除操作了。

最初不知道是哪位非资深技术人员将Sticky Bit直译成了“粘滞位”，刘遄老师建议将其称为“保护位”，这既好记，又能立刻让人了解它的作用。RHEL 8系统中的/tmp作为一个共享文件的目录，默认已经设置了SBIT特殊权限位，因此除非是该目录的所有者，否则无法删除这里面的文件。

与前面所讲的SUID和SGID权限显示方法不同，当目录被设置SBIT特殊权限位后，文件的其他人权限部分的x执行权限就会被替换成t或者T，原本有x执行权限则会写成t，原本没有x执行权限则会被写成T。

/tmp目录上的SBIT权限默认已经存在，体现为“其他用户”权限字段的权限变为rwt：**练习命令**

[root@linuxprobe ~]# ls -ald /tmp

drwxrwx**rwt**. 17 root root 4096 Oct 28 00:29 /tmp

其实，文件能否被删除并不取决于自身的权限，而是看其所在目录是否有写入权限（其原理会在下个章节讲到）。为了避免现在很多读者不放心，所以下面的命令还是赋予了这个test文件最大的777权限（rwxrwxrwx）：**练习命令**

[root@linuxprobe ~]# cd /tmp

[root@linuxprobe tmp]# echo "Welcome to linuxprobe.com" > test

[root@linuxprobe tmp]# chmod 777 test

[root@linuxprobe tmp]# ls -al test

-rwxrwxrwx. 1 root root 26 Oct 29 14:29 test

随后，切换到一个普通用户身份下，尝试删除这个其他人创建的文件就会发现，即便读、写、执行权限全开，但是由于SBIT特殊权限位的缘故，依然无法删除该文件：**练习命令**

[root@linuxprobe tmp]# su - linuxprobe

[linuxprobe@linuxprobe ~]$ cd /tmp

[linuxprobe@linuxprobe tmp]$ rm -f test

rm: cannot remove 'test': Operation not permitted

工作中对特殊权限善加使用，能够实现很多巧妙的功能，chmod命令设置特殊权限时的参数如表5-7所示。

表5-7                                          SUID、SGID、SBIT特殊权限的设置参数

|  |  |
| --- | --- |
| 参数 | 作用 |
| u+s | 设置SUID权限 |
| u-s | 取消SUID权限 |
| g+s | 设置SGID权限 |
| g-s | 取消SGID权限 |
| o+t | 设置SBIT权限 |
| o-t | 取消SBIT权限 |

切换回root管理员的身份下，在家目录中创建一个名为linux的新目录，随后设置上SBIT权限吧：**练习命令**

[linuxprobe@linuxprobe tmp]$ exit

Logout

[root@linuxprobe tmp]# cd ~

[root@linuxprobe ~]# mkdir linux

[root@linuxprobe ~]# chmod -R o+t linux/

[root@linuxprobe ~]# ls -ld linux/

drwxr-xr-t. 2 root root 6 Feb 11 19:34 linux/

上面的o+t参数是在一般权限已经设置完毕的前提下，又新增了一项特殊权限，那如果我们想一般权限和特殊权限一起设置，有什么高效率的方法呢？

其实SUID、SGID与SBIT也有对应的数字法表示，分别即是4、2、1。也就是说777还不是最大权限，满权限应该是7777，第一个数字代表的是特殊权限位。既然知道了数字表示法是由“特殊权限+一般权限”构成的，那就以上面linux目录的权限为例为同学们梳理下计算方法吧。

在“rwxr-xr-t”权限中，最后一位是t，代表该文件的一般权限为“rwxr-xr-x”，并带有SBIT特殊权限。对于读（r）、写（w）、执行（x）权限的数字计算方法大家应该很熟悉了——即755，而SBIT特殊权限位是1，合并后结果为1755。

再增加点难度，如果权限是“rwsrwSr--”呢？首先不要慌，看到大写S代表原先没有执行权限，因此一般权限为“rwxrw-r--”，数字法计算后结果是764。带有的SUID和SGID特殊权限数字法表示是4和2，心算得出结果是6，合并后结果为6764。这个示例确实难度高一些，读者们可以参考学习图5-4的计算过程，搞明白了再往下看。

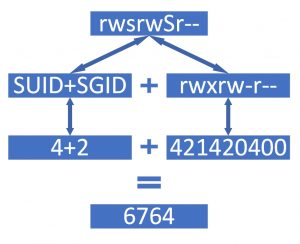


图5-4 权限的字符表示法转数字表示法

将数字法转换成回字符法难度略微高一些，以5537为例为大家讲解。首先特殊权限的5是由4+1组成的，意味着是有SUID和SBIT。SUID和SGID的写法是原先有执行权限则是小写s，如果没有执行权限则是大写S，而SBIT的写法则是原先有执行权限是小写t，没有执行权限是大写T。一般权限的537进行字符转换后应为“r-x-wxrwx”，然后在此基础上增加SUID和SGID特殊权限，合并后结果是“r-s-wxrwt”。可以参考图5-5所示的计算过程来学习理解。

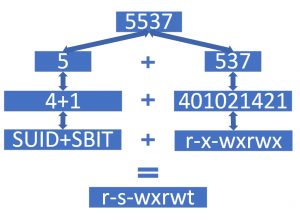


图5-5 权限的数字表示法转字符标识法

Tips

Linux系统中文件的权限位真像北京的房价，寸土寸金，一个占位竟能有这么多含义，工作中一定要小心谨慎。

**出现问题?大胆提问!**

因读者们硬件不同或操作错误都可能导致实验配置出错，请耐心再仔细看看操作步骤吧，不要气馁~

**5.4 文件的隐藏属性**

Linux系统中的文件除了具备一般权限和特殊权限之外，还有一种隐藏权限，即被隐藏起来的权限，默认情况下不能直接被用户发觉。有用户曾经在生产环境和RHCE考试题目中碰到过明明权限充足但却无法删除某个文件的情况，或者仅能在日志文件中追加内容而不能修改或删除内容，这在一定程度上阻止了黑客篡改系统日志的图谋，因此这种“奇怪”的文件也保障了Linux系统的安全性。

既然叫隐藏权限了，那肯定不能用常规的ls命令就让我们看到它的真面目，专用的设置命令是chattr，专用的查看命令是lsattr。

**1.  chattr命令**

chattr命令用于设置文件的隐藏权限，英文全称为：“change attributes”，语法格式为：“chattr [参数] 文件名称”。

如果想要把某个隐藏功能添加到文件上，则需要在命令后面追加“+参数”，如果想要把某个隐藏功能移出文件，则需要追加“-参数”。chattr命令中可供选择的隐藏权限参数非常丰富，具体如表5-8所示。

表5-8                                  chattr命令中的参数及其作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| i | 无法对文件进行修改；若对目录设置了该参数，则仅能修改其中的子文件内容而不能新建或删除文件 |
| a | 仅允许补充（追加）内容，无法覆盖/删除内容（Append Only） |
| S | 文件内容在变更后立即同步到硬盘（sync） |
| s | 彻底从硬盘中删除，不可恢复（用0填充原文件所在硬盘区域） |
| A | 不再修改这个文件或目录的最后访问时间（atime） |
| b | 不再修改文件或目录的存取时间 |
| D | 检查压缩文件中的错误 |
| d | 使用dump命令备份时忽略本文件/目录 |
| c | 默认将文件或目录进行压缩 |
| u | 当删除该文件后依然保留其在硬盘中的数据，方便日后恢复 |
| t | 让文件系统支持尾部合并（tail-merging） |
| x | 可以直接访问压缩文件中的内容 |

为了让读者能够更好地见识隐藏权限的效果，我们先来创建一个普通文件，然后立即尝试删除（这个操作肯定会成功）：**练习命令**

[root@linuxprobe ~]# echo "for Test" > linuxprobe

[root@linuxprobe ~]# rm linuxprobe

rm: remove regular file ‘linuxprobe’? y

实践是检验真理的唯一标准。如果您没有亲眼见证过隐藏权限强大功能的美妙，就一定不会相信原来Linux系统会如此安全。接下来再次新建一个普通文件，并为其设置不允许删除与覆盖（+a参数）权限，然后再尝试将这个文件删除：**练习命令**

[root@linuxprobe ~]# echo "for Test" > linuxprobe

[root@linuxprobe ~]# chattr +a linuxprobe

[root@linuxprobe ~]# rm linuxprobe

rm: remove regular file ‘linuxprobe’? y

rm: cannot remove ‘linuxprobe’: Operation not permitted

可见，上述操作失败了。

**2.  lsattr命令**

lsattr命令用于查看文件的隐藏权限，英文全称为：“list attributes”，语法格式为：“lsattr [参数] 文件名称”。

在Linux系统中，文件的隐藏权限必须使用lsattr命令来查看，平时使用的ls之类的命令则看不出端倪：**练习命令**

[root@linuxprobe ~]# ls -al linuxprobe

-rw-r--r--. 1 root root 9 Feb 12 11:42 linuxprobe

一旦使用lsattr命令后，文件上被赋予的隐藏权限马上就会原形毕露：**练习命令**

[root@linuxprobe ~]# lsattr linuxprobe

-----**a**---------- linuxprobe

此时按照显示的隐藏权限的类型（字母），使用chattr命令将其去掉：**练习命令**

[root@linuxprobe ~]# chattr -a linuxprobe

[root@linuxprobe ~]# lsattr linuxprobe

---------------- linuxprobe

[root@linuxprobe ~]# rm linuxprobe

rm: remove regular file ‘linuxprobe’? y

一般我们会将-a参数设置到日志文件（/var/log/messages）上面，这样在不影响系统正常写入日志的前提下，还可以防止骇客清理自己的作案证据。如果希望彻底的保护起某个文件，不允许任何人修改和删除它的话，不妨加上-i参数试试，效果特别好。

**5.5 文件访问控制列表**

不知道大家是否发现，前文讲解的一般权限、特殊权限、隐藏权限其实有一个共性——权限是针对某一类用户设置的，能够对很多人同时生效。如果希望对某个指定的用户进行单独的权限控制，就需要用到文件的访问控制列表（FACL，File Access Control Lists）了。通俗来讲，基于普通文件或目录设置ACL访问控制其实就是针对指定的用户或用户组设置文件或目录的操作权限，更加精准的派发权限。另外，如果针对某个目录设置了ACL，则目录中的文件会继承其权限；若针对文件设置了ACL，则文件不再继承其所在目录的权限。

为了更直观地看到ACL对文件权限控制的强大效果，我们先切换到普通用户，然后尝试进入root管理员的家目录中。在没有针对普通用户对root管理员的家目录设置ACL之前，其执行结果如下所示：**练习命令**

[root@linuxprobe ~]# su - linuxprobe

[linuxprobe@linuxprobe ~]$ cd /root

-bash: cd: /root: **Permission denied**

[linuxprobe@linuxprobe root]$ exit

**1.  setfacl命令**

setfacl命令用于管理文件的ACL权限规则，英文全称为：“set files ACL”，语法格式为：“ setfacl [参数] 文件名称”。

ACL权限提供的是在所有者、所属组、其他人的读/写/执行权限之外的特殊权限控制，使用setfacl命令可以针对单一用户或用户组、单一文件或目录来进行读/写/执行权限的控制。其中，针对目录文件需要使用-R递归参数；针对普通文件则使用-m参数；如果想要删除某个文件的ACL，则可以使用-b参数。setfacl命令的常用参数如表5-9所示：

表5-9                                              setfacl命令中的参数以及作用

|  |  |
| --- | --- |
| 参数 | 作用 |
| -m | 修改权限 |
| -M | 从文件中读取权限 |
| -x | 删除某个权限 |
| -b | 删除全部权限 |
| -R | 递归子目录 |

例如我们原本是无法进入到/root目录中的，现在给这位普通用户单独设置下权限吧：**练习命令**

[root@linuxprobe ~]# setfacl -Rm u:linuxprobe:rwx /root

随后再切换到这位普通用户的身份下，就能正常进入了呢：**练习命令**

[root@linuxprobe ~]# su - linuxprobe

[linuxprobe@linuxprobe ~]$ cd /root

[linuxprobe@linuxprobe root]$ ls

anaconda-ks.cfg Documents initial-setup-ks.cfg Pictures Templates

Desktop Downloads Music Public Videos

[linuxprobe@linuxprobe root]$ exit

是不是觉得效果很酷呢？但是现在有这样一个小问题—怎么去查看文件上有那些ACL呢？常用的ls命令是看不到ACL表信息的，但是却可以看到文件的权限最后一个点（**.**）变成了加号（**+**）,这就意味着该文件已经设置了ACL了。现在大家是不是感觉学得越多，越不敢说自己精通Linux系统了吧？就这么一个不起眼的点（.），竟然还表示这么一种重要的权限。**练习命令**

[root@linuxprobe ~]# ls -ld /root

dr-xrwx---**+** 14 root root 4096 May 4 2020 /root

**2.  getfacl命令**

getfacl命令用于查看文件的ACL权限规则，英文全称为：“get files ACL”，语法格式为：“ getfacl [参数] 文件名称”。

getfacl命令用于显示文件上设置的ACL信息，格式为“getfacl 文件名称”。Linux系统中的命令就是这么又可爱又好记。想要设置ACL，用的是setfacl命令；要想查看ACL，则用的是getfacl命令。下面使用getfacl命令显示在root管理员家目录上设置的所有ACL信息。**练习命令**

[root@linuxprobe ~]# getfacl /root

ggetfacl: Removing leading '/' from absolute path names

# file: root

# owner: root

# group: root

user::r-x

**user:linuxprobe:rwx**

group::r-x

mask::rwx

other::---

ACL权限还可以针对某个用户组来设置，例如允许某个组的用户都可以读写/etc/fstab个文件：**练习命令**

[root@linuxprobe ~]# setfacl -m g:linuxprobe:rw /etc/fstab

[root@linuxprobe ~]# getfacl /etc/fstab

getfacl: Removing leading '/' from absolute path names

# file: etc/fstab

# owner: root

# group: root

user::rw-

group::r--

**group:linuxprobe:rw-**

mask::rw-

other::r--

设置错了想删除？没问题！清空所有ACL权限用-b参数，指定删除某一条权限就用-x参数：**练习命令**

[root@linuxprobe ~]# setfacl -x g:linuxprobe /etc/fstab

[root@linuxprobe ~]# getfacl /etc/fstab

getfacl: Removing leading '/' from absolute path names

# file: etc/fstab

# owner: root

# group: root

user::rw-

group::r--

mask::r--

other::r--

对于ACL权限的设置都是立即且永久生效的，不需要再编辑什么配置文件，这点特别方便~但是也有个安全隐患，如果我们不小心设置错了权限怎么办？就会覆盖掉文件原始的权限信息，并且永远都找不回来了。

**操作前备份一下，总是好的习惯。**

例如将/home目录上的ACL权限备份一份，要用-R递归参数，这样不仅能够把目录本身的权限备份了，里面的文件权限也都能自动备份，另再外加上第3章节学习过的输出重定向操作就可以实现：**练习命令**

[root@linuxprobe ~]# getfacl -R /home > backup.acl

getfacl: Removing leading '/' from absolute path names

[root@linuxprobe ~]# ls

anaconda-ks.cfg Desktop Downloads Music Public Videos

**backup.acl** Documents initial-setup-ks.cfg Pictures Templates

ACL权限的恢复也很简单，使用的是“--restore”参数，由于备份时已经指定了是对/home目录进行的操作，所以不需要写对应的目录名称，它能够自动找到要恢复的对象：**练习命令**

[root@linuxprobe /]# setfacl --restore backup.acl

**5.6 su命令与sudo服务**

各位读者在实验环境中很少遇到安全问题，并且为了避免因权限因素导致配置服务失败，从而建议使用root管理员来学习本书，但是在生产环境中还是要对安全多一份敬畏之心，不要用root管理员去做所有事情。因为一旦执行了错误的命令，可能会直接导致系统崩溃，这样一来，不但客户指责、领导批评，没准奖金也会鸡飞蛋打。但转头一想，尽管Linux系统为了安全性考虑，使得许多系统命令和服务只能被root管理员来使用，但是这也让普通用户受到了更多的权限束缚，从而导致无法顺利完成特定的工作任务。

su命令可以解决切换用户身份的需求，使得当前用户在不退出登录的情况下，顺畅地切换到其他用户，比如从root管理员切换至普通用户：**练习命令**

[root@linuxprobe ~]# su - linuxprobe

[linuxprobe@linuxprobe ~]$ id

uid=1000(linuxprobe) gid=1000(linuxprobe) groups=1000(linuxprobe) context=unconfined\_u:unconfined\_r:unconfined\_t:s0-s0:c0.c1023

细心的读者一定会发现，上面的su命令与用户名之间有一个减号（-），这意味着完全切换到新的用户，即把环境变量信息也变更为新用户的相应信息，而不是保留原始的信息。强烈建议在切换用户身份时添加这个减号（-）。

另外，当从root管理员切换到普通用户时是不需要密码验证的，而从普通用户切换成root管理员就需要进行密码验证了；这也是一个必要的安全检查：**练习命令**

[linuxprobe@linuxprobe ~]$ su - root

Password: **此处输入管理员密码**

[root@linuxprobe ~]#

尽管像上面这样使用su命令后，普通用户可以完全切换到root管理员身份来完成相应工作，但这将暴露root管理员的密码，从而增大了系统密码被黑客获取的几率；这并不是最安全的方案。

刘遄老师接下来将介绍如何使用sudo命令把特定命令的执行权限赋予给指定用户，这样既可保证普通用户能够完成特定的工作，也可以避免泄露root管理员密码。我们要做的就是合理配置sudo服务，以便兼顾系统的安全性和用户的便捷性。

|  |  |
| --- | --- |
| 参数 | 作用 |
| -h | 列出帮助信息 |
| -l | 列出当前用户可执行的命令 |
| -u 用户名或UID值 | 以指定的用户身份执行命令 |
| -k | 清空密码的有效时间，下次执行sudo时需要再次进行密码验证 |
| -b | 在后台执行指定的命令 |
| -p | 更改询问密码的提示语 |

**Tips**

**授权原则：**在保证普通用户完成相应工作的前提下，尽可能少地赋予额外的权限。

sudo命令用于给普通用户提供额外的权限，语法格式为：“ sudo [参数] 用户名”。

使用sudo命令可以给普通用户提供额外的权限来完成原本只有root管理员才能完成的任务。可以限制用户执行指定的命令、记录用户执行过的每一条命令、集中的管理用户与权限（/etc/sudoers）以及验证密码后一段时间内免验证的方便措施。常见的sudo命令可用参数如表5-10所示：

表5-10                                          sudo命令中的可用参数以及作用

当然，如果担心直接修改配置文件会出现问题，则可以使用sudo命令提供的visudo命令来配置用户权限。

visudo命令用于编辑配置用户sudo权限文件，语法格式为：“ visudo [参数]”。

这是一条会自动调用vi编辑器来配置/etc/sudoers权限文件的命令，能够解决多个用户同时修改权限而导致的冲突问题，不仅如此，visudo命令还可以对配置文件内的参数进行语法检查，在发现参数错误时进行报错提醒，比用户直接修改文件更友好、安全、方便。**练习命令**

**>>> /etc/sudoers: syntax error near line 1 <<<**

What now?

Options are:

(e)dit sudoers file again

e(x)it without saving changes to sudoers file

(Q)uit and save changes to sudoers file (DANGER!)

使用visudo命令配置权限文件时，其操作方法与Vim编辑器中用到的方法完全一致，因此在编写完成后记得在末行模式下保存并退出。配置权限文件时，按照下面的格式将第101行（大约）填写上指定的信息：

**谁可以使用  允许使用的主机=(以谁的身份)  可执行命令的列表**

**谁可以使用：**稍后要为那位用户进行命令授权。

**允许使用的主机：**可以填写ALL代表不限制来源主机，亦可填写如192.168.10.0/24的网段限制来源地址，只有从允许网段登录时才能使用sudo命令。

**以谁的身份：**可以填写ALL代表系统最高权限，也可以是另外一位用户的名字。

**可执行命令的列表：**可以填写ALL代表不限制命令的列表，亦可填写如/usr/bin/cat的文件名称来限制命令列表，多个命令文件之间用逗号（,）间隔。

在Linux系统中配置服务文件时，虽然没有硬性规定，但从经验来讲新增的参数位置不建议太靠上，避免服务一些必要的功能没加载完成时，我们填写的新参数不被执行成功。一般会在配置文件中找一下相似的参数，然后在相邻位置进行新的修改，或者选择文件的中下部位置。**练习命令**

[root@linuxprobe ~]# visudo

99 ## Allow root to run any commands anywhere

100 root ALL=(ALL) ALL

**101 linuxprobe ALL=(ALL) ALL**

在填写完毕后记得要先保存再退出，然后切换至指定的普通用户身份，此时就可以用sudo -l命令查看到所有可执行的命令了（下面的命令中，验证的是该普通用户的密码，而不是root管理员的密码，请读者不要搞混了）：**练习命令**

[root@linuxprobe ~]# su - linuxprobe

[linuxprobe@linuxprobe ~]$ sudo -l

We trust you have received the usual lecture from the local System

Administrator. It usually boils down to these three things:

#1) Respect the privacy of others.

#2) Think before you type.

#3) With great power comes great responsibility.

[sudo] password for linuxprobe: **此处输入linuxprobe用户的密码**

Matching Defaults entries for linuxprobe on localhost:

!visiblepw, always\_set\_home, match\_group\_by\_gid, always\_query\_group\_plugin,

env\_reset, env\_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR LS\_COLORS",

env\_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC\_ADDRESS LC\_CTYPE",

env\_keep+="LC\_COLLATE LC\_IDENTIFICATION LC\_MEASUREMENT LC\_MESSAGES",

env\_keep+="LC\_MONETARY LC\_NAME LC\_NUMERIC LC\_PAPER LC\_TELEPHONE",

env\_keep+="LC\_TIME LC\_ALL LANGUAGE LINGUAS \_XKB\_CHARSET XAUTHORITY",

secure\_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User linuxprobe may run the following commands on localhost:

**(ALL) ALL**

接下来是见证奇迹的时刻！作为一名普通用户，是肯定不能看到root管理员的家目录（/root）中的文件信息的，但是，只需要在想执行的命令前面加上sudo命令就行了：**练习命令**

[linuxprobe@linuxprobe ~]$ ls /root

ls: cannot open directory '/root': Permission denied

[linuxprobe@linuxprobe ~]$ sudo ls /root

anaconda-ks.cfg Documents initial-setup-ks.cfg Pictures Templates

Desktop Downloads Music Public Videos

效果立竿见影！但是考虑到生产环境中不允许某个普通用户拥有整个系统中所有命令的最高执行权（这也不符合前文提到的权限赋予原则，即尽可能少地赋予权限），因此ALL参数就有些不合适了。因此只能赋予普通用户具体的命令以满足工作需求，这也受到了必要的权限约束。如果需要让某个用户只能使用root管理员的身份执行指定的命令，切记一定要给出该命令的绝对路径，否则系统会识别不出来，这时可以先使用whereis命令找出命令所对应的保存路径：**练习命令**

[linuxprobe@linuxprobe ~]$ exit

logout

[root@linuxprobe ~]# whereis cat

cat: **/usr/bin/cat** /usr/share/man/man1/cat.1.gz /usr/share/man/man1p/cat.1p.gz

[root@linuxprobe ~]# whereis reboot

reboot: **/usr/sbin/reboot** /usr/share/man/man2/reboot.2.gz /usr/share/man/man8/reboot.8.gz

然后使用visudo命令继续编辑权限文件，将原先101行所新增的参数作如下修改，多个命令之间用逗号（,）间隔。**练习命令**

[root@linuxprobe ~]# visudo

99 ## Allow root to run any commands anywhere

100 root ALL=(ALL) ALL

101 linuxprobe ALL=(ALL) **/usr/bin/cat,/usr/sbin/reboot**

在编辑好后依然是先保存再退出。再次切换到指定的普通用户，然后尝试正常查看某个系统文件的内容，此时系统提示没有权限（Permission denied）。再使用sudo命令就能顺利地查看文件内容了：**练习命令**

[root@linuxprobe ~]# su - linuxprobe

[linuxprobe@linuxprobe ~]$ cat /etc/shadow

cat: /etc/shadow: Permission denied

[linuxprobe@linuxprobe ~]$ sudo cat /etc/shadow

[sudo] password for linuxprobe: **此处输入linuxprobe用户的密码**

root:$6$tTbuw5DkOPYqq.vI$RMk9FCGHoJOq2qAPRURTQm.Qok2nN3yFn/i4f/falVGgGND9XoiYFbrxDn16WWiziaSJ0/cR06U66ipEoGLPJ.::0:99999:7:::

bin:\*:17784:0:99999:7:::

daemon:\*:17784:0:99999:7:::

adm:\*:17784:0:99999:7:::

lp:\*:17784:0:99999:7:::

sync:\*:17784:0:99999:7:::

shutdown:\*:17784:0:99999:7:::

halt:\*:17784:0:99999:7:::

………………省略部分输出信息………………

[linuxprobe@linuxprobe ~]$ exit

logout

不知大家是否发觉在每次执行sudo命令后都会要求验证一下密码。虽然这个密码就是当前登录用户的密码，但是每次执行sudo命令都要输入一次密码其实也挺麻烦的，这时可以添加NOPASSWD参数，使得用户下次再执行sudo命令时就不用密码验证啦~**练习命令**

[root@linuxprobe ~]# visudo

99 ## Allow root to run any commands anywhere

100 root ALL=(ALL) ALL

101 linuxprobe ALL=(ALL) **NOPASSWD:**/usr/bin/cat,/usr/sbin/reboot

这样，当切换到普通用户后再执行命令时，就不用再频繁地验证密码了，我们在日常工作中也就痛快至极了。**练习命令**

[root@linuxprobe ~]# su - linuxprobe

[linuxprobe@linuxprobe ~]$ reboot

User root is logged in on tty2.

Please retry operation after closing inhibitors and logging out other users.

Alternatively, ignore inhibitors and users with 'systemctl reboot -i'.

[linuxprobe@linuxprobe ~]$ sudo reboot

同学们请细心留意上面的用户身份变换，visudo命令只有root管理员才可以执行，普通用户使用会提示权限不足。

**课后习题：**

1．在Linux系统中，管理员的账号是什么？

2．如何使用Linux系统的命令行来添加或删除用户？

3．若某个文件的所有者具有文件的读/写/执行权限，其余人仅有读权限，那么用数字法表示应该是什么?

4．某文件的字符权限为rwxrw-r--，那么对应的数字法应该为？

5．某链接文件的权限用数字法表示为755，那么相应的字符法表示是什么呢？

6．如果希望用户执行某命令时临时拥有该命令所有者的权限，应该设置什么特殊权限？

7．若对文件设置了隐藏权限+i，则意味着什么？

8．使用访问控制列表（ACL）来限制linuxprobe用户组，使得该组中的所有成员不得在/tmp目录中写入内容。

9．当普通用户使用sudo命令时是否需要验证密码？