su和sudo的区别与使用

**一.   使用 su 命令临时切换用户身份**

**1、 su 的适用条件和威力**

su命令就是切换用户 的工具，怎么理解呢？比如我们以普通用户beinan登录的，但要添加用户任务，执行useradd ，beinan用户没有这个权限，而这个权限恰恰由root所拥有。解决办法无法有两个，一是退出beinan用户，重新以root用户登录，但这种办法 并不是最好的；二是我们没有必要退出beinan用户，可以用su来切换到root下进行添加用户的工作，等任务完成后再退出root。我们可以看到当然 通过su 切换是一种比较好的办法；

通过su可以在用户之 间切换，如果超级权限用户root向普通或虚拟用户切换不需要密码，什么是权力？这就是！而普通用户切换到其它任何用户都需要密码验证；

**2、 su 的用法：**

su [OPTION选项参数] [用户]

-, -l, --login 登录并改变到所切换的用户环境；

-c, --commmand=COMMAND 执行一个命令，然后退出所切换到的用户环境；

至于更详细的，请参看 man su ；

**3、 su 的范例：**

su 在不加任何参数，默认为切换到root用户，但没有转到root用户家目录下，也就是说这时虽然是切换为root用户了，但并没有改变root登录环境； 用户默认的登录环境，可以在/etc/passwd 中查得到，包括家目录，SHELL定义等；

|  |
| --- |
| [beinan@localhost ~]&#8194;$ su Password: [root@localhost beinan]# pwd /home/beinan |

su 加参数 - ，表示默认切换到root用户，并且改变到root用户的环境；

|  |
| --- |
| [beinan@localhost ~]&#8194;$ pwd /home/beinan [beinan@localhost ~]&#8194;$ su - Password: [root@localhost ~]# pwd /root |

su 参数 - 用户名

[beinan@localhost ~] $ su - root 注：这个和su - 是一样的功能；

Password:

[root@localhost ~]# pwd

/root

[beinan@localhost ~] $ su - linuxsir 注：这是切换到 linuxsir用户

Password: 注：在这里输入密码；

[linuxsir@localhost ~] $ pwd 注：查看用户当前所处的位置；

/home/linuxsir

[linuxsir@localhost ~] $ id 注：查看用户的UID和GID信息，主要是看是否切换过来了；

uid=505(linuxsir) gid=502(linuxsir) groups=0(root),500(beinan),502(linuxsir)

[linuxsir@localhost ~] $

[beinan@localhost ~] $ su - -c ls 注：这是su的参数组合，表示切换到root用户，并且改变到root环境，然后列出root家目录的文件，然后退出root用户；

Password: 注：在这里输入root的密码；

anaconda-ks.cfg Desktop install.log install.log.syslog testgroup testgroupbeinan testgrouproot

[beinan@localhost ~] $ pwd 注：查看当前用户所处的位置；

/home/beinan

[beinan@localhost ~] $ id 注：查看当前用户信息；

uid=500(beinan) gid=500(beinan) groups=500(beinan)

**4、 su的优缺点；**

su 的确为管理带来方便，通过切换到root下，能完成所有系统管理工具，只要把root的密码交给任何一个普通用户，他都能切换到root来完成所有的系统 管理工作；但通过su切换到root后，也有不安全因素；比如系统有10个用户，而且都参与管理。如果这10个用户都涉及到超级权限的运用，做为管理员如 果想让其它用户通过su来切换到超级权限的root，必须把root权限密码都告诉这10个用户；如果这10个用户都有root权限，通过root权限可 以做任何事，这在一定程度上就对系统的安全造成了威协；想想Windows吧，简直就是恶梦；“没有不安全的系统，只有不安全的人”，我们绝对不能保证这 10个用户都能按正常操作流程来管理系统，其中任何一人对系统操作的重大失误，都可能导致系统崩溃或数据损失；所以su 工具在多人参与的系统管理中，并不是最好的选择，su只适用于一两个人参与管理的系统，毕竟su并不能让普通用户受限的使用；超级用户root密码应该掌 握在少数用户手中，这绝对是真理！所以集权而治的存在还是有一定道理的；

**二. sudo 授权许可使用的su，也是受限制的su**

**1. sudo 的适用条件**

由于su 对切换到超级权限用户root后，权限的无限制性，所以su并不能担任多个管理员所管理的系统。如果用su 来切换到超级用户来管理系统，也不能明确哪些工作是由哪个管理员进行的操作。特别是对于服务器的管理有多人参与管理时，最好是针对每个管理员的技术特长和 管理范围，并且有针对性的下放给权限，并且约定其使用哪些工具来完成与其相关的工作，这时我们就有必要用到 sudo。

通过sudo，我们能 把某些超级权限有针对性的下放，并且不需要普通用户知道root密码，所以sudo 相对于权限无限制性的su来说，还是比较安全的，所以sudo 也能被称为受限制的su ；另外sudo 是需要授权许可的，所以也被称为授权许可的su；

sudo 执行命令的流程是当前用户切换到root（或其它指定切换到的用户），然后以root（或其它指定的切换到的用户）身份执行命令，执行完成后，直接退回到 当前用户；而这些的前提是要通过sudo的配置文件/etc/sudoers来进行授权；

**2、 从编写 sudo 配置文件/etc/sudoers开始**

sudo的配置文件是 /etc/sudoers ，我们可以用他的专用编辑工具visodu ，此工具的好处是在添加规则不太准确时，保存退出时会提示给我们错误信息；配置好后，可以用切换到您授权的用户下，通过sudo -l 来查看哪些命令是可以执行或禁止的；

/etc/sudoers 文件中每行算一个规则，前面带有#号可以当作是说明的内容，并不执行；如果规则很长，一行列不下时，可以用\号来续行，这样看来一个规则也可以拥有多个 行；

/etc/sudoers 的规则可分为两类；一类是别名定义，另一类是授权规则；别名定义并不是必须的，但授权规则是必须的；

**3、 /etc/sudoers 配置文件中别名规则**

别名规则定义格式如 下：

Alias\_Type NAME = item1, item2, ...

或

Alias\_Type NAME = item1, item2, item3 : NAME = item4, item5

别名类型 （Alias\_Type）：别名类型包括如下四种

Host\_Alias 定义主机别名；

User\_Alias 用户别名，别名成员可以是用户，用户组（前面要加%号）

Runas\_Alias 用来定义runas别名，这个别名指定的是“目的用户”，即sudo 允许切换至的用户；

Cmnd\_Alias 定义命令别名；

NAME 就是别名了，NMAE的命名是包含大写字母、下划线以及数字，但必须以一个大写字母开头，比如SYNADM、SYN\_ADM或SYNAD0是合法 的，sYNAMDA或1SYNAD是不合法的；

item 按中文翻译是项目，在这里我们可以译成成员，如果一个别名下有多个成员，成员与成员之间，通过半角,号分隔；成员在必须是有效并事实存在的。什么是有效的 呢？比如主机名，可以通过w查看用户的主机名（或ip地址），如果您只是本地机操作，只通过hostname 命令就能查看；用户名当然是在系统中存在的，在/etc/paswd中必须存在；对于定义命令别名，成员也必须在系统中事实存在的文件名（需要绝对路 径）；

item成员受别名类 型 Host\_Alias、User\_Alias、Runas\_Alias、Cmnd\_Alias 制约，定义什么类型的别名，就要有什么类型的成员相配。我们用Host\_Alias定义主机别名时，成员必须是与主机相关相关联，比如是主机名（包括远程 登录的主机名）、ip地址（单个或整段）、掩码等；当用户登录时，可以通过w命令来查看登录用户主机信息；用User\_Alias和 Runas\_Alias定义时，必须要用系统用户做为成员；用Cmnd\_Alias 定义执行命令的别名时，必须是系统存在的文件，文件名可以用通配符表示，配置Cmnd\_Alias时命令需要绝对路径；其中 Runas\_Alias 和User\_Alias 有点相似，但与User\_Alias 绝对不是同一个概念，Runas\_Alias 定义的是某个系统用户可以sudo 切换身份到Runas\_Alias 下的成员；我们在授权规则中以实例进行解说；别名规则是每行算一个规则，如果一个别名规则一行容不下时，可以通过\来续行；同一类型别名的定义，一次也可 以定义几个别名，他们中间用:号分隔，

Host\_Alias HT01=localhost,st05,st04,10,0,0,4,255.255.255.0,192.168.1.0/24 注：定义主机别名HT01，通过=号列出成员

Host\_Alias HT02=st09,st10 注：主机别名HT02，有两个成员；

Host\_Alias HT01=localhost,st05,st04,10,0,0,4,255.255.255.0,192.168.1.0/24:HT02=st09,st10 注：上面的两条对主机的定义，可以通过一条来实现，别名之间用:号分割；

注：我们通过 Host\_Alias 定义主机别名时，项目可以是主机名、可以是单个ip（整段ip地址也可以），也可以是网络掩码；如果是主机名，必须是多台机器的网络中，而且这些机器得能 通过主机名相互通信访问才有效。那什么才算是通过主机名相互通信或访问呢？比如 ping 主机名，或通过远程访问主机名来访问。在我们局域网中，如果让计算机通过主机名访问通信，必须设置/etc/hosts， /etc/resolv.conf ，还要有DNS做解析，否则相互之间无法通过主机名访问；在设置主机别名时，如果项目是中某个项目是主机名的话，可以通过hostname 命令来查看本地主机的主机名，通过w命令查来看登录主机是来源，通过来源来确认其它客户机的主机名或ip地址；对于主机别名的定义，看上去有点复杂，其实 是很简单。

如果您不明白 Host\_Alias 是怎么回事，也可以不用设置主机别名，在定义授权规则时通过ALL来匹配所有可能出现的主机情况。如果您把主机方面的知识弄的更明白，的确需要多多学习。

User\_Alias SYSAD=beinan,linuxsir,bnnnb,lanhaitun 注：定义用户别名，下有四个成员；要在系统中确实在存在的；

User\_Alias NETAD=beinan,bnnb 注：定义用户别名NETAD ，我想让这个别名下的用户来管理网络，所以取了NETAD的别名；

User\_Alias WEBMASTER=linuxsir 注：定义用户别名WEBMASTER，我想用这个别名下的用户来管理网站；

User\_Alias SYSAD=beinan,linuxsir,bnnnb,lanhaitun:NETAD=beinan,bnnb:WEBMASTER=linuxsir 注：上面三行的别名定义，可以通过这一行来实现，请看前面的说明，是不是符合？

Cmnd\_Alias USERMAG=/usr/sbin/adduser,/usr/sbin/userdel,/usr/bin/passwd [A-Za-z]\*,/bin/chown,/bin/chmod

注意：命令别名下的成 员必须是文件或目录的绝对路径；

|  |
| --- |
| Cmnd\_Alias DISKMAG=/sbin/fdisk,/sbin/parted Cmnd\_Alias NETMAG=/sbin/ifconfig,/etc/init.d/network Cmnd\_Alias KILL = /usr/bin/kill Cmnd\_Alias PWMAG = /usr/sbin/reboot,/usr/sbin/halt Cmnd\_Alias SHELLS = /usr/bin/sh, /usr/bin/csh, /usr/bin/ksh, \  /usr/local/bin/tcsh, /usr/bin/rsh, \  /usr/local/bin/zsh |

注：这行定义命令别名 有点长，可以通过 \ 号断行；

Cmnd\_Alias SU = /usr/bin/su,/bin,/sbin,/usr/sbin,/usr/bin

在上面的例子中，有 KILL和PWMAG的命令别名定义，我们可以合并为一行来写，也就是等价行；

Cmnd\_Alias KILL = /usr/bin/kill:PWMAG = /usr/sbin/reboot,/usr/sbin/halt 注：这一行就代表了KILL和PWMAG命令别名，把KILL和PWMAG的别名定义合并在一行写也是可以的；

Runas\_Alias OP = root, operator

Runas\_Alias DBADM=mysql:OP = root, operator 注：这行是上面两行的等价行；至于怎么理解Runas\_Alias ，我们必须得通过授权规则的实例来理解；

**4、 /etc/sudoers中的授权规则**

授权规则是分配权限的 执行规则，我们前面所讲到的定义别名主要是为了更方便的授权引用别名；如果系统中只有几个用户，其实下放权限比较有限的话，可以不用定义别名，而是针对系 统用户直接直接授权，所以在授权规则中别名并不是必须的；

授权规则并不是无章可 寻，我们只说基础一点的，比较简单的写法，如果您想详细了解授权规则写法的，请参看man sudoers

授权用户 主机=命令动作

这三个要素缺一不可， 但在动作之前也可以指定切换到特定用户下，在这里指定切换的用户要用( )号括起来，如果不需要密码直接运行命令的，应该加NOPASSWD:参数，但这些可以省略；举例说明；

**实例 一：**

beinan ALL=/bin/chown,/bin/chmod

如果我们在/etc /sudoers 中添加这一行，表示beinan 可以在任何可能出现的主机名的系统中，可以切换到root用户下执行 /bin/chown 和/bin/chmod 命令，通过sudo -l 来查看beinan 在这台主机上允许和禁止运行的命令；

值得注意的是，在这里 省略了指定切换到哪个用户下执行/bin/shown 和/bin/chmod命令；在省略的情况下默认为是切换到root用户下执行；同时也省略了是不是需要beinan用户输入验证密码，如果省略了，默认 为是需要验证密码。

为了更详细的说明这 些，我们可以构造一个更复杂一点的公式；

授权用户 主机=[(切换到哪些用户或用户组)] [是否需要密码验证] 命令1,[(切换到哪些用户或用户组)] [是否需要密码验证] [命令2],[(切换到哪些用户或用户组)] [是否需要密码验证] [命令3]....

**注 解：**

凡是[ ]中的内容，是可以省略；命令与命令之间用,号分隔；通过本文的例子，可以对照着看哪些是省略了，哪些地方需要有空格；

在[(切换到哪些用户 或用户组)] ，如果省略，则默认为root用户；如果是ALL ，则代表能切换到所有用户；注意要切换到的目的用户必须用()号括起来，比如(ALL)、(beinan)

**实例 二：**

beinan ALL=(root) /bin/chown, /bin/chmod

如果我们把第一个实例 中的那行去掉，换成这行；表示的是beinan 可以在任何可能出现的主机名的主机中，可以切换到root下执行 /bin/chown ，可以切换到任何用户招执行/bin/chmod 命令，通过sudo -l 来查看beinan 在这台主机上允许和禁止运行的命令；

**实例 三：**

beinan ALL=(root) NOPASSWD: /bin/chown,/bin/chmod

如果换成这个例子呢？ 表示的是beinan 可以在任何可能出现的主机名的主机中，可以切换到root下执行 /bin/chown ，不需要输入beinan用户的密码；并且可以切换到任何用户下执行/bin/chmod 命令，但执行chmod时需要beinan输入自己的密码；通过sudo -l 来查看beinan 在这台主机上允许和禁止运行的命令；

关于一个命令动作是不 是需要密码，我们可以发现在系统在默认的情况下是需要用户密码的，除非特加指出不需要用户需要输入自己密码，所以要在执行动作之前加入NOPASSWD: 参数；

有可能有的弟兄对系统 管理的命令不太懂，不知道其用法，这样就影响了他对 sudoers定义的理解，下面我们再举一个最简单，最有说服务力的例子；

**实例 四：**

比如我们想用 beinan普通用户通过more /etc/shadow文件的内容时，可能会出现下面的情况；

[beinan@localhost ~] $ more /etc/shadow/etc/shadow: 权限不够

这时我们可以用 sudo more /etc/shadow 来读取文件的内容；就就需要在/etc/soduers中给beinan授权

于是我们就可以先su 到root用户下通过visudo 来改/etc/sudoers ；（比如我们是以beinan用户登录系统的）

[beinan@localhost ~] $ su

Password: 注：在这里输入root密码

下面运行 visodu；

[root@localhost beinan]# visudo 注：运行visudo 来改 /etc/sudoers

加入如下一行，退出保 存；退出保存，在这里要会用vi，visudo也是用的vi编辑器；至于vi的用法不多说了；beinan ALL=/bin/more 表示beinan可以切换到root下执行more 来查看文件；

退回到beinan用 户下，用exit命令；

[root@localhost beinan]# exit

exit

[beinan@localhost ~] $

查看beinan的通 过sudo能执行哪些命令？

[beinan@localhost ~] $ sudo -l

Password: 注：在这里输入beinan用户的密码

User beinan may run the following commands on this host: 注：在这里清晰的说明在本台主机上，beinan用户可以以root权限运行more ；在root权限下的more ，可以查看任何文本文件的内容的；

(root) /bin/more

最后，我们看看是不是 beinan用户有能力看到/etc/shadow文件的内容；

[beinan@localhost ~] $ sudo more /etc/shadow

beinan 不但能看到 /etc/shadow文件的内容，还能看到只有root权限下才能看到的其它文件的内容，比如；

[beinan@localhost ~] $ sudo more /etc/gshadow

对于beinan用户 查看和读取所有系统文件中，我只想把/etc/shadow 的内容可以让他查看；可以加入下面的一行；

beinan ALL=/bin/more /etc/shadow

题外话：有的弟兄会 说，我通过su 切换到root用户就能看到所有想看的内容了，哈哈，对啊。但咱们现在不是在讲述sudo的用法吗？如果主机上有多个用户并且不知道root用户的密码， 但又想查看某些他们看不到的文件，这时就需要管理员授权了；这就是sudo的好处；

**实例 五：练习用户组在/etc/sudoers中写法；**

如果用户组出现在 /etc/sudoers 中，前面要加%号，比如%beinan ，中间不能有空格；%beinan ALL=/usr/sbin/\*,/sbin/\*

如果我们在 /etc/sudoers 中加上如上一行，表示beinan用户组下的所有成员，在所有可能的出现的主机名下，都能切换到root用户下运行 /usr/sbin和/sbin目录下的所有命令；

**实例 六：练习取消某类程序的执行：**

取消程序某类程序的执 行，要在命令动作前面加上!号； 在本例中也出现了通配符的\*的用法；

beinan ALL=/usr/sbin/\*,/sbin/\*,!/usr/sbin/fdisk 注：把这行规则加入到/etc/sudoers中；但您得有beinan这个用户组，并且beinan也是这个组中的才行；

本规则表示 beinan用户在所有可能存在的主机名的主机上运行/usr/sbin和/sbin下所有的程序，但fdisk 程序除外；

[beinan@localhost ~] $ sudo -l

Password: 注：在这里输入beinan用户的密码；

|  |
| --- |
| User beinan may run the following commands on this host: (root) /usr/sbin/\* (root) /sbin/\* (root) !/sbin/fdisk [beinan@localhost ~]&#8194;$ sudo /sbin/fdisk -l Sorry, user beinan is not allowed to execute '/sbin/fdisk -l' as root on localhost. |

注：不能切换到 root用户下运行fdisk 程序；

**实例 七：别名的运用的实践；**

假如我们就一台主机 localhost，能通过hostname 来查看，我们在这里就不定义主机别名了，用ALL来匹配所有可能出现的主机名；并且有beinan、linuxsir、lanhaitun 用户；主要是通过小例子能更好理解；sudo虽然简单好用，但能把说的明白的确是件难事；最好的办法是多看例子和man soduers ；

User\_Alias SYSADER=beinan,linuxsir,%beinan

|  |
| --- |
| User\_Alias DISKADER=lanhaitun Runas\_Alias OP=root Cmnd\_Alias SYDCMD=/bin/chown,/bin/chmod,/usr/sbin/adduser,/usr/bin/passwd [A-Za-z]\*,!/usr/bin/passwd root  Cmnd\_Alias DSKCMD=/sbin/parted,/sbin/fdisk 注：定义命令别名DSKCMD，下有成员parted和fdisk ； SYSADER ALL= SYDCMD,DSKCMD DISKADER ALL=(OP) DSKCMD |

注解：

第一行：定义用户别名 SYSADER 下有成员 beinan、linuxsir和beinan用户组下的成员，用户组前面必须加%号；

第二行：定义用户别名 DISKADER ，成员有lanhaitun

第三行：定义 Runas用户，也就是目标用户的别名为OP，下有成员root

第四行：定义 SYSCMD命令别名，成员之间用,号分隔，最后的!/usr/bin/passwd root 表示不能通过passwd 来更改root密码；

第五行：定义命令别名 DSKCMD，下有成员parted和fdisk ；

第六行：表示授权 SYSADER下的所有成员，在所有可能存在的主机名的主机下运行或禁止 SYDCMD和DSKCMD下定义的命令。更为明确遥说，beinan、linuxsir和beinan用户组下的成员能以root身份运行 chown 、chmod 、adduser、passwd，但不能更改root的密码；也可以以root身份运行 parted和fdisk ，本条规则的等价规则是；

beinan,linuxsir,%beinan ALL=/bin/chown,/bin/chmod,/usr/sbin/adduser,/usr/bin/passwd [A-Za-z]\*,!/usr/bin/passwd root,/sbin/parted,/sbin/fdisk

第七行：表示授权 DISKADER 下的所有成员，能以OP的身份，来运行 DSKCMD ，不需要密码；更为明确的说 lanhaitun 能以root身份运行 parted和fdisk 命令；其等价规则是：

lanhaitun ALL=(root) /sbin/parted,/sbin/fdisk

可能有的弟兄会说我想 不输入用户的密码就能切换到root并运行SYDCMD和DSKCMD 下的命令，那应该把把NOPASSWD:加在哪里为好？理解下面的例子吧，能明白的；

SYSADER ALL= NOPASSWD: SYDCMD, NOPASSWD: DSKCMD

5、/etc /sudoers中其它的未尽事项；

在授权规则中，还有 NOEXEC:和EXEC的用法，自己查man sudoers 了解；还有关于在规则中通配符的用法，也是需要了解的。这些内容不多说了，毕竟只是一个入门性的文档。soduers配置文件要多简单就有多简单，要多难 就有多难，就看自己的应用了。

6、sudo的用法；

我们在前面讲的 /etc/sudoers 的规则写法，最终的目的是让用户通过sudo读取配置文件中的规则来实现匹配和授权，以便替换身份来进行命令操作，进而完成在其权限下不可完成的任务；

我们只说最简单的用 法；更为详细的请参考man sudo

sudo [参数选项] 命令

-l 列出用户在主机上可用的和被禁止的命令；一般配置好/etc/sudoers后，要用这个命令来查看和测试是不是配置正确的；

-v 验证用户的时间戳；如果用户运行sudo 后，输入用户的密码后，在短时间内可以不用输入口令来直接进行sudo 操作；用-v 可以跟踪最新的时间戳；

-u 指定以以某个用户执行特定操作；

-k 删除时间戳，下一个sudo 命令要求用求提供密码；

举列：

首先我们通过 visudo 来改/etc/sudoers 文件，加入下面一行；

beinan,linuxsir,%beinan ALL=/bin/chown,/bin/chmod,/usr/sbin/adduser,/usr/bin/passwd [A-Za-z]\*,!/usr/bin/passwd root,/sbin/parted,/sbin/fdisk

然后列出beinan 用户在主机上通过sudo 可以切换用户所能用的命令或被禁止用的命令；

[beinan@localhost ~] $ sudo -l 注：列出用户在主机上能通过切换用户的可用的或被禁止的命令；

Password: 注：在这里输入您的用户密码；

User beinan may run the following commands on this host:

(root) /bin/chown 注：可以切换到root下用chown命令；

(root) /bin/chmod 注：可以切换到root下用chmod命令；

(root) /usr/sbin/adduser 注：可以切换到root下用adduser命令；

(root) /usr/bin/passwd [A-Za-z]\* 注：可以切换到root下用 passwd 命令；

(root) !/usr/bin/passwd root 注：可以切换到root下，但不能执行passwd root 来更改root密码；

(root) /sbin/parted 注：可以切换到 root下执行parted ；

(root) /sbin/fdisk 注：可以切换到root下执行 fdisk ；

通过上面的sudo -l 列出可用命令后，我想通过chown 命令来改变/opt目录的属主为beinan ；

[beinan@localhost ~] $ ls -ld /opt 注：查看/opt的属主；

drwxr-xr-x 26 root root 4096 10月 27 10:09 /opt 注：得到的答案是归属root用户和root用户组；

[beinan@localhost ~] $ sudo chown beinan:beinan /opt 注：通过chown 来改变属主为beinan用户和beinan用户组；

[beinan@localhost ~] $ ls -ld /opt 注：查看/opt属主是不是已经改变了；

drwxr-xr-x 26 beinan beinan 4096 10月 27 10:09 /opt

我们通过上面的例子发 现beinan用户能切换到root后执行改变用户口令的passwd命令；但上面的sudo -l 输出又明文写着不能更改root的口令；也就是说除了root的口令，beinan用户不能更改外，其它用户的口令都能更改。下面我们来测试；

对于一个普通用户来 说，除了更改自身的口令以外，他不能更改其它用户的口令。但如果换到root身份执行命令，则可以更改其它用户的口令；

比如在系统中有 linuxsir这个用户, 我们想尝试更改这个用户的口令，

[beinan@localhost ~] $ passwd linuxsir 注：不通过sudo 直接运行passwd 来更改linuxsir用户的口令；

passwd: Only root can specify a user name. 注：失败，提示仅能通过 root来更改；

[beinan@localhost ~] $ sudo passwd linuxsir 注：我们通过/etc/sudoers 的定义，让beinan切换到root下执行 passwd 命令来改变linuxsir的口令；

Changing password for user linuxsir.

New UNIX password: 注：输入新口令；

Retype new UNIX password: 注：再输入一次；

passwd: all authentication tokens updated successfully. 注：改变成功；