Linux下我们经常使用的SSH远程连接到系统进行操作，而为了安全，一般使用的是非root用户连接，只有在必要的时候，我们才会使用sudo切换到root帐户下进行操作，以保证系统尽可能的安全和不被损坏。然而，在有的时候我们通过普通用户SSH连接到Linux的时候，使用sudo来执行身份切换的时候，系统会报错（以下是CentOS6.4下的报错）：

对不起，用户 yourusername 不能在 CentOS 上运行 sudo。

或者

Sorry,user yourusername may not run sudo on CentOS

这种情况下，为了让我们的用户能够正常的使用sudo，需要通过修改/etc/sudoers文件。

#添加下面一行到:/etc/sudoers，即可让该用户使用sudoyourusername ALL=(ALL) ALL

这种修改是将所有的权限都赋值给了yourusername，为了sudo这个命令，就这么ALL几下，似乎有点不负责任啊,有木有？

sudoers文件默认是只读的，因此不能直接修改。网上很多方法是使用chmod来修改sudoers的权限后再通过编辑器（vi/vim etc.）来修改sudoers内容，之后再将sudoers文件的权限恢复，来达到修改sudoers的目的。这种方法确实能够修改sudoers，但是却没有办法保证sudoers文件的正确性，如果修改出现错误，将导致sudoers（sudo）不能正常的工作。更正确的做法是直接使用sudoers的专用编辑器：visudo来编辑sudoers文件。

# visudo## visudo默认的编辑器是vi，如果你需要修改你的默认编辑器，可以通过添加EDITOR="" 来修改。# EDITOR="/usr/bin/vim -p -X" visudo## 如果要永久改变默认编辑器，只要将EDITOR设置为环境变量# export EDITOR="/usr/bin/vim -p -X"

 使用visudo来编辑sudoers的好处是：当我们完成sudoers的编辑时，系统会自动检测我们sudoers的格式是否正确，从而保证sudoers文件的正确性。

 其实，sudoers文件是非常简单的，其内容只包含两种，第一种是别名（Aliases，基本变量定义），第二种是用户规范描述（user specifications，指定某用户可以运行的内容）。而这两种内容都是通过扩展巴科斯范式（EBNF）1来进行定义。一旦定义了别名，在用户说明中，就可以使用别名来简化给用户定义运行内容。如果同一个用户在sudoers文件中能够找到多条说明，那么将使用找到的最后一条。

sudoers的别名包含四种：User\_Alias，Runas\_Alias，Host\_Alias和Cmnd\_Alias。

User\_Alias是用户别名，这个和系统中的group有点类似。通过用户别名，在sudoers中，可以将一个或者多个用户用别名来替代。

## 定义一个用户别名User\_Alias OPERATORS = operator1,operator2,remote\_admin

注意：  
1. 等号后面的都是系统中的用户名,用户ID，组，组id等组成，多个用逗号分隔，具体信息可参看文后参考内容sudoers2  
2. 别名只能是大写字母开头的，并由大写字母，下划线和数字组成的字符串

Runas\_Alias和User\_Alias类似，但不同的是Runas\_Alias指明的是用户将要运行命令的身份。Runas\_Alias可以包含Runas\_Alias别名（User\_Alias是包含User\_Alias别名）。

同时要注意的是，上面两个别名在匹配的时候使用的是字符串匹配法则，也就是说，当两个不同用户名的用户，即使使用了同一个用户ID，在实际情况下，也会被认为是不同的。要避免这种情况，可以考虑使用用户的ID来替代用户名。如：

Runas\_Alias ROOT = #0## 所有的用户ID为0的，统归为ROOT别名

Host\_Alias是主机别名，指定规则适用的主机列表。

Host\_Alias SERVERS = 192.168.0.1, 192.168.0.2, server1

注意：

1. Host\_Alias中，只能使用真实的地址，这就意味着127.0.0.1这种地址是永远不会被匹配到的。而localhost要只有在真有主机的主机名为localhost的时候才能被匹配。

2. 如果IP地址没有给定掩码，sudo自动会搜寻本地网卡上对应匹配的掩码地址来作为默认的网络掩码。

Cmnd\_Alias是命令别名，指定用户能够运行的某一类命令的列表。可包含一个，或多个命令（带参数），目录，或者其他的命令别名。

Cmnd\_Alias SHUTDOWN\_CMDS = /sbin/poweroff, /sbin/reboot, /sbin/halt##所有的关机命令Cmnd\_Alias ADMIN\_CMDS = /usr/sbin/passwd, /usr/sbin/useradd, /usr/sbin/userdel, /usr/sbin/usermod, /usr/sbin/visudo## 所有的用户管理命令

注意：

1. 命令列表中如果包含目录（以/结束），那么该目录下（不含子目录）的所有可执行文件都被赋予了执行权限，这和使用通配符\*类似；

2. 命令列表中务必使用全路径名，如果不使用完全限定名，将意味着恶意用户可以创建同名文件并在切换后的用户下运行，这将给系统带来未知的安全风险；

3. 当命令列表中的命令需要带参数的时候，如果出现特殊字符（',',':','=','/'），将需要使用反斜杆（'/'）进行转义。

通过上面四种别名，我们能够定义出特定的用户，主机和命令列表，在实际运行中，我们如果要对这些特定的列表的某些运行时配置项3进行修改，我们可以通过Default来完成。

Default对于前面四种别名的限制格式各有不同，定义如下：

## EBNFDefault\_Entry ::= ('Defaults' |'Default' '@' Host\_List |'Default' ':' User\_List | 'Default' '!' Cmnd\_List | 'Default' '>' Runas\_List ) Parameter\_ListParameter\_List ::= (Parameter '=' Value |Parameter '+=' Value |Parameter '-=' Value |'!'\* Parameter)[ , Parameter\_List]

定义中 \*\_List可以是别名也可以是特定的具体信息（如：用户名）。

别名之后，就是具体的用户规范描述（User Specification）了。

User Specification表述的主要意义就是：**哪些用户在哪些环境里可以作为哪些用户来运行哪些程序**。用EBNF来表示这个定义如下：

User\_Spec ::= User\_List Host\_List '=' (Cmnd\_Spec | Cmnd\_Spec ',' Cmnd\_Spec\_List) /(':' Host\_List '=' Cmnd\_Spec\_List)\*Cmnd\_Spec ::= Runas\_Spec? SELinux\_Spec? Solaris\_Priv\_Spec? Tag\_Spec\* Cmnd Runas\_Spec ::= '(' Runas\_List? (':' Runas\_List)? ')' SELinux\_Spec ::= ('ROLE=role' | 'TYPE=type') Solaris\_Priv\_Spec ::= ('PRIVS=privset' | 'LIMITPRIVS=privset') Tag\_Spec ::= ('NOPASSWD:' | 'PASSWD:' | 'NOEXEC:' | 'EXEC:' |'SETENV:' | 'NOSETENV:' | 'LOG\_INPUT:' | 'NOLOG\_INPUT:' |'LOG\_OUTPUT:' | 'NOLOG\_OUTPUT:')

看起来很复杂，来个栗子也许会好很多：

## 在SERVERS环境中OPERATORS可以以ROOT角色运行SHUTDOWN\_CMDSOPERATORS SERVERS = (ROOT) SHUTDOWN\_CMDS## 在所有的环境中yourusername都可以切换到所有账户下运行所有的命令yourusername ALL = (ALL) ALL## yourusername用户在SERVERS上面执行关机操作yourusername SERVERS = /sbin/halt, /sbin/poweroff## 以root来执行关机操作，但以operator来执行kill命令yourusername SERVERS = (root) SHUTDOWN\_CMDS, (operator) /bin/kill## sudo -u来选择用户## 仅以operator组来执行kill命令yourusername SERVERS = (:operator) /bin/kill## 注意，在sudo的时候需要使用-g配合，但kill还是由yourusername执行## 无需使用密码(不用校验operator身份)，即可以operator身份来执行kill命令yourusername SERVERS = (operator) NOPASSWD: /bin/kill

至于SELinux\_Spec和Solaris\_Priv\_Spec一般不会用到，SELinux\_Spec是在支持SELinux的系统中有效，而Solaris\_Priv\_Spce则是针对Solaris系统的配置。Tag\_Spec相对前两个在Linux系统来说就实用的多，Tag\_Spec共有10个可选tag参数，可以分成5组，每组由互斥的两个tag参数构成，分别是：NOPASSWD和PASSWD，NOEXEC和EXEC，SETENV和NOSETENV，LOG\_INPUT和NOLOG\_INPUT，LOG\_OUTPUT和NOLOG\_OUTPUT。不同组别的tag参数可以同时使用，在使用Tag\_Spec之后，如果没有使用相反的tag参数覆盖，则同行后面的命令将会继承tag参数设置。下面是5组tag的说明：

|  |  |
| --- | --- |
| Tag参数 | 作用描述 |
| NOPASSWD,PASSWD | 标识是否需要身份验证 |
| NOEXEC，EXEC | 是否限制程序进一步运行命令（动态链接有效） |
| SETENV，NOSETENV | 标志如何处理环境变量，建议不用SETENV（默认是NOSETENV），SETENV会禁用通过命令行传递的env\_reset参数 |
| LOG\_INPUT,NOLOG\_INPUT | 输入日志开启标志，LOG\_INPUT开启后，所有输入信息都会被记录，如密码这类敏感信息同样会以明文形式记录下来 |
| LOG\_OUTPUT,NOLOG\_OUTPUT | 输出日志标志，开启后，所有的输出都会被保存在文件中 |

 到此为止，sudoers文件的主体就已经完成了，根据这些东西，你就可以顺利修改sudoers了。

高级部分：

1. 可以通过#includedir，#include两中方式将其他的sudoers文件包含进来；

2. 要注释，用#，当然，第一点中的这个和#后面跟数字的要小心（猜猜看,是神码？）；

3. 在编辑中，为了简便，你可以在主机列表，路径名称和命令列表中使用通配符（不是正则），通配符：\*（匹配0或多个），?（匹配单个字符），[...]（匹配范围内的任一字符），[!...]（说了不是正则了，匹配任一不在指定范围内的字符），/x（转义特殊字符，如：\*，?, [, ]）。例外：""双引号在命令行参数中表示的是命令不允许带任何参数；传递给sudoedit内置命令的命令行参数必须包含路径名称，因此斜杆（/）不会被通配符匹配；

4. 在Alias，Defaulsts，User Specification中设定值的时候可以使用感叹号（!），表示取非(排除)，当然，如果你觉得多几个感叹号能加强效果也是可以的，记住单数表示取反，偶数是双重否定，直接抵消作用即可。同时也要注意，这个感叹号有时候会失效，会带来一些安全风险；

5. 上一点的一点特例:

## 可Runas除了root之外的所有用户User\_Alias SPECIALUSERS = ALL,!root## 这个的作用不同于上面这行规则，sudo匹配的时候是不匹配root## 同时也不会匹配任何其他用户User\_Alias SPECIALUSERS = !root

6. 规则太长，一行写不下？试试行尾加上反斜杆（/），然后换到第二行继续写。

7. MitchellChu本身也是菜鸟一枚，共勉！ （这个最高级:P）

最后回归开篇第一个问题，你觉得这种配置有问题么？

 参考：

1. 扩展巴科斯范式（EBNF）：Extended Backus-Naur Form，Backus-Naur Form,巴科斯范式，扩展巴科斯范式

2. sudoers：Sudoers Manual

3. Sudoers的Default可配置参数请参看2中Sudoers Manual中的SUDOERS OPTIONS

4.其他一些参考：Archlinux SUDO,Ubuntu Sudoers,sudo.ws SUDO manual