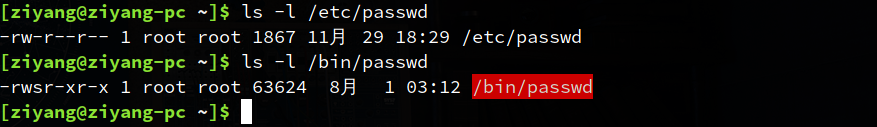
1.1 Linux系统文件和目录权限设置与辨识setuid程序uid差异

1. 设计并实现不同用户对不同类文件的r、w、x权限:

(1) 查看系统文件的权限设置

a) 查看/etc/passwd文件和/etc/bin/passwd文件的权限设置，并分析其权限为什么这么设置；

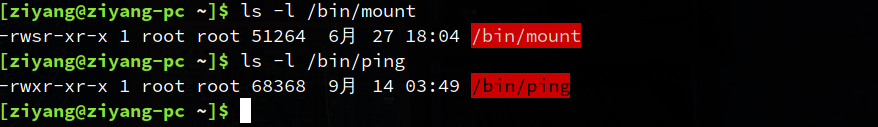
分别运行ls -l /etc/passwd和ls -l /bin/passwd，结果如下：



所有用户的信息都保存在/etc/passwd中，该文件由root创建，每个用户对/etc/passwd都有读权限，但是只有root对其有写权限，其他用户都没有写权限，保证了普通用户无法修改用户信息，保证了系统安全。/bin/passwd用于修改用户的密码，任何用户都可以调用，该文件由root创建，密码保存在/etc/shadow文件中，由于/etc/shadow文件仅仅允许root进行读写，所以/bin/passwd设置密码必须以root身份执行。所以/bin/passwd设置了setuid位，允许普通用户以root身份运行该文件。

b) 找到2个设置了setuid位的可执行程序，该程序的功能，该程序如果不设置setuid位是否能够达到相应的功能，

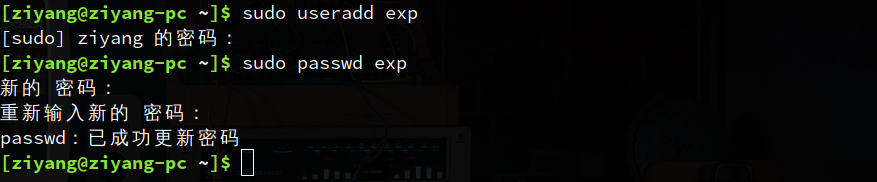
mount和ping程序，查看权限如下：



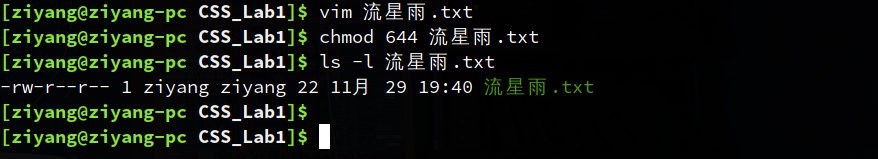
mount程序用于将一个分区或者设备挂载在某空文件夹下，ping命令用于测试网络连接量，由于mount挂载设备，ping建立套接字，这两个行为都需要root权限，普通用户无法执行，所以需要设置setuid位，使运行这两个程序的euid是root。

1. 设置文件或目录权限
2. 用户A具有文本文件”流星雨.txt”，该用户允许别人下载；

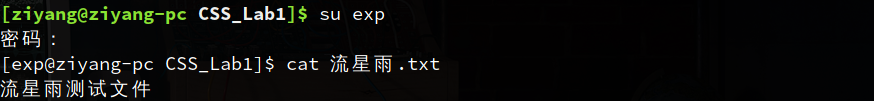
首先创建一个测试用户exp，与当前用户不同组，并设置密码：



允许下载，即需要读权限即可，权限位可设置为644：

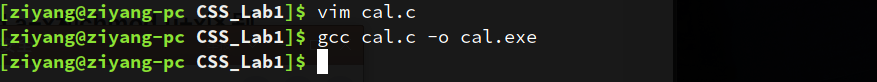


当切换到其他用户时，仍然可以读取该文件：

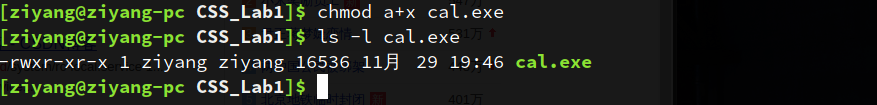


1. 用户A编译了一个可执行文件”cal.exe”，该用户想在系统启动时运行；

首先创建c文件并编译为cal.exe：



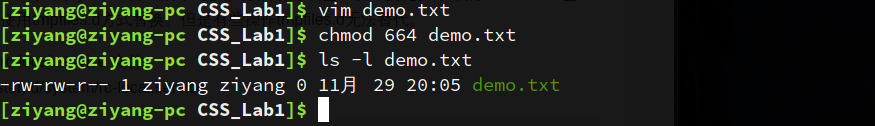
在系统启动时，无法确认是哪个用户，所以需要给所有用户该文件的执行权限：



在高版本Linux中，添加开机启动脚本需要使用systemd，不赘述。

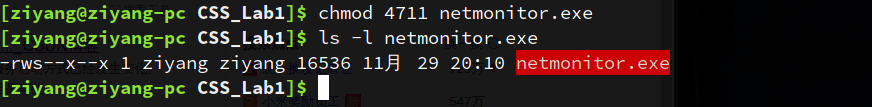
1. 用户A又起草了文件”demo.txt”，想让同组的用户帮其修改文件；

将文件权限设置为664即可，同组的用户即拥有读写权限：



1. 一个root用户拥有的网络服务程序”netmonitor.exe”，需要设置setuid位才能完成其功能。

创建netmonitor.exe后，将权限设置为4711即可：



1. 一些可执行程序运行时需要系统管理员权限，在UNIX中可以利用setuid位实现其功能，但setuid了的程序运行过程中拥有了root权限，因此在完成管理操作后需要切换到普通用户的身份执行后续操作。
2. 设想一种场景，比如提供http网络服务，需要设置setuid位，并为该场景编制相应的代码；
3. 如果用户fork进程后，父进程和子进程中euid、ruid、suid的差别；
4. 利用execl执行setuid程序后，euid、ruid、suid是否有变化；
5. 程序何时需要临时性放弃root权限，何时需要永久性放弃root权限，并在程序中分别实现两种放弃权限方法；
6. execl函数族中有多个函数，比较有环境变量和无环境变量的函数使用的差异。