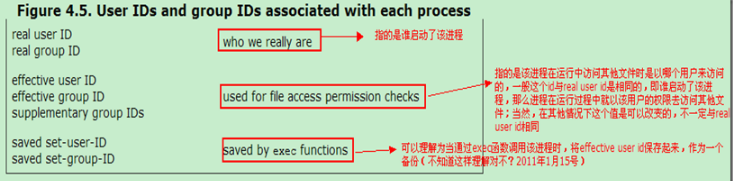
 linux中每个进程有六个或者更多的ID与其关联



real user id一般是从passwd文件中获取的，一般是不会发生改变的，当然也可以改变。可以理解为启动该进程的用户，即哪个用户启动了该进程，那么该进程的real user id就是该用户的id

      effective user id决定了进程访问文件的权限，一般情况下是与real user id是相同的，但可以改变

      saved set-user-id 当该程序被执行时，save set-user-id是effective-user-id的一份copy

      user id和group id情况是类似的，通过介绍set-user-id即可了解set-group-id

      在系统中每个文件都有一个owner和group owner，这一般是通过stat结构中的st\_uid来标记的。在文件的mode里面有一位叫做set-user-id的位，在设置了该位后，程序运行的时候effective-user-id等于该文件的owner id（比较拗口）

如果需要一个进程执行一些只有超级用户才能执行的操作（如修改系统时间），那么必须要让进程的有效用户ID为0。

但是这种做法同样也赋予了进程执行其他操作的权限。处理此类问题的传统的方法：删除有效权限（如将有效用户改为非0，并将0保存在saved set-user-Id 中），并只在需要的时候临时请求这些权限。

Linux能力模型优化了这个问题处理方式：超级用户权限被划分为不同的单元，单元称为能力。进程只在拥有相应的能力的时候才能执行相应的操作。

每个进程拥有 3个相关的能力集——“许可的”、“有效的”和“可继承的”

许可的——一个进程可以使用的能力。若进程从许可集中删除了一些能力，那么将永远无法再重新获取该能力，除非它执行了一个再次授予该能力的程序。

有效的——内核使用这些能力来对进程执行权限检测。只有进程在 许可 集中维护着一个能力，那么进程才能通过从有效 集中删除这个能力来临时禁用该能力，之后再将该能力还原到这个集合中；

可继承的——当进程执行一个程序时，可以将这些权限带入许可集中。