**实验3 UNIX系统中权限的管理**

**1140320206 霍峻杰**

**一、实验目的：**

chroot的配置和root 的capability使用

**二、实验1：**

**① 实验要求：**

利用chroot工具来虚拟化管理

1) 实现bash或ps的配置使用；

2)利用chroot实现SSH服务或FTP服务的虚拟化隔离；

3)chroot后如何降低权限，利用实验一中编制的程序检查权限的合理性；

4)在chroot之前没有采用cd xx目录，会对系统有何影响，编制程序分析其影响。

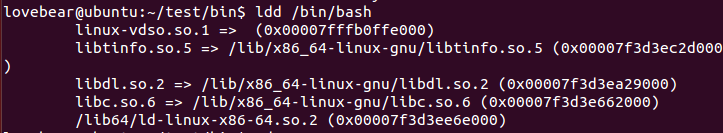
5) 分析chroot的安全性，应用chroot安全原则设置chroot脚本程序实现服务的隔离管理

**② 实验结果：**

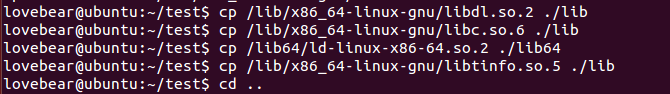
1) 配置bash

这里需要将/bin/bash中的bash文件，和一些lib库的链接文件拷贝到chroot的目的文件中，这里以test文件夹作为chroot的根目录。不知道需要链接哪些文件可以使用ldd命令来查看需要的动态函数库（当然也可以全盘拷贝，但是失去了chroot的意义）。

1. 需要的动态库



2. 拷贝到test目录

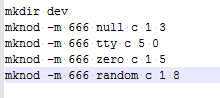


3. 运行测试



2) 利用chroot实现SSH服务或FTP服务的虚拟化隔离：

1. 创建dev文件夹及其设备节点文件

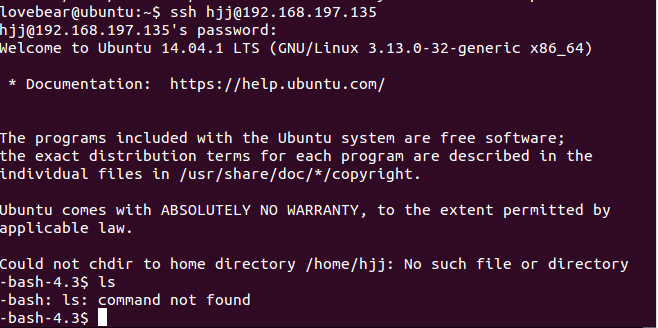


（创建命令被挤掉了，这里的命令是随手记录在notepad++的）

2. 定义使用监狱的用户和监狱目录

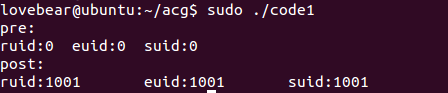


3. 测试，连接到hjj后，不能ls

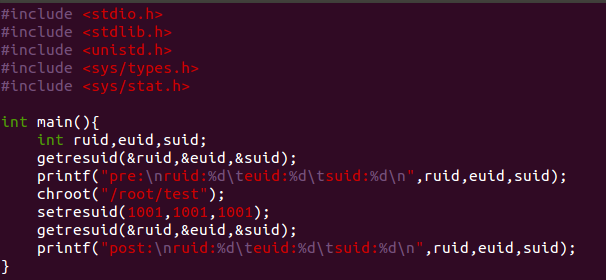


3）chroot后降低权限

1. chroot后会将uid设置为非root，这里测试设置成1001



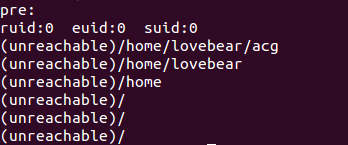
2. 实验代码



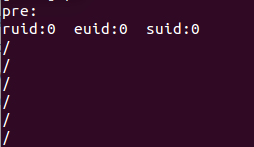
4) 在chroot之前没有采用cd xx目录，会对系统有何影响，编制程序分析其影响：

没有在监狱目录chroot，此时创建的监狱将会拥有当前目录的访问权限，超出了监狱的范围。

1. 测试没有改变目录，此时我所在目录为/home/lovebear/acg,结果打印出上级的目录，说明可以访问所在目录的信息及文件。



2. 在代码中加一行chdir，使chroot前目录已经变为监狱目录，结果表明，最上层为/，而对于其他目录，是透明的。



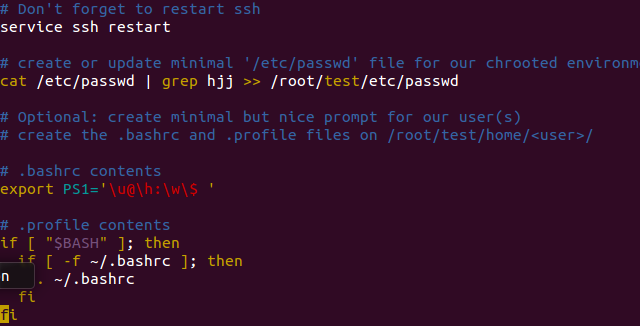
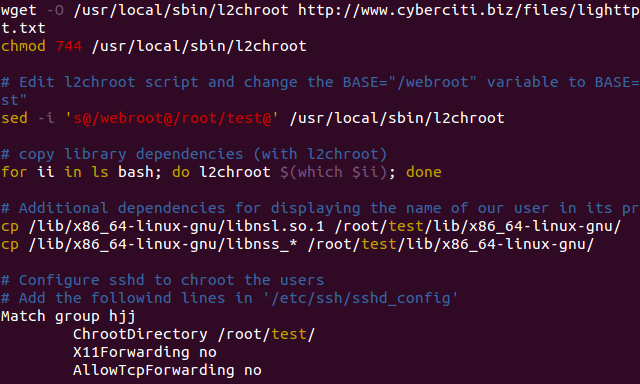
3. 附上代码



5) 分析chroot的安全性，应用chroot安全原则设置chroot脚本程序实现服务的隔离管理

1. 实验步骤和3一样，只是写进了脚本。安全性在于，ssh访问hjj身份时，不提供不在chroot目录下的文件支持，如小问（3）中的ls命令将不被支持。

2. 实验脚本



**三、实验2：**

**① 实验要求：**

1. 修改系统内核，配置capability的能力位，实现几种能力位的设置可验证。

样例：redhat 2.4下的能力

1）函数说明

getcap可以获得程序文件所具有的能力(CAP).

getpcaps可以获得进程所具有的能力(CAP).

setcap可以设置程序文件的能力(CAP).

注:

1)cap\_chown=eip是将chown的能力以cap\_effective(e),cap\_inheritable(i),cap\_permitted(p)三种位图的方式授权给相关的程序文件.

2)如果改变文件名,则能力保留到新文件.

3)用setcap -r /bin/chown可以删除掉文件的能力.

4)重新用setcap授权将覆盖之前的能力.

能力位：CAP\_SYS\_NICE 23(允许提升优先级,设置其它进程的优先级)

{

对于普通用户程序的NICE优先级,不能超过ulimit对它的限制,如下:

nice -n -5 ls

nice: cannot set niceness: Permission denied

而CAP\_SYS\_NICE可以帮助普通用户设置一个想要的一个任意优先级.

setcap cap\_sys\_nice=eip /usr/bin/nice

切换到普通用户,指定优先级,如下:

nice -n -5 ls

log mnt mount.c mounttest pacct psacct psacct.c reboot1 reboot1.c test

[root@localhost zy]# setcap cap\_sys\_nice=eip /home/tttt/test

[root@localhost zy]# getcap /home/tttt/test

/home/tttt/test = cap\_sys\_nice+eip

}

1. 实验要求：

　a)实现３种能力位的授权和查看，并分析授权前和授权后的差异；

b)系统启动时关闭某能力位，对系统的应用和安全性有何影响，以具体能力位为例说明

c)比较能力位和chroot管理权限的差异

**② 实验结果：**

a) 实现３种能力位的授权和查看，并分析授权前和授权后的差异；

1. cap\_dac\_read\_search，忽略所有对读、搜索操作的限制：

Setcap前：



Setcap:



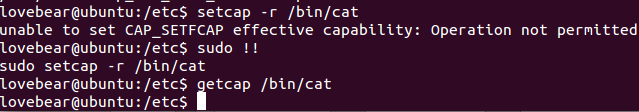
Setcap后：



查看cap位：



取消cap位：



给cat设置cap\_dac\_read\_search位，将不检查打印的文件权限检查，很危险，相当于授予了root的读文件权限。

2. cap\_sys\_time 25，允许普通用户更改系统时钟：

Setcap前：



Setcap:



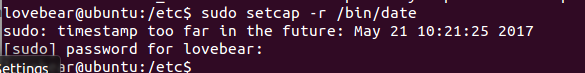
Setcap后：



查看cap：

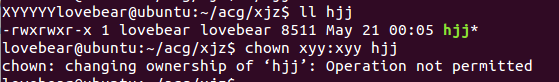


取消cap：



3. cap\_chown，该权限位用于授权普通用户可以通过程序更改任意文件的owner

Setcap前：



Setcap：



Setcap后：



查看cap：



取消cap：



b) 系统启动时关闭某能力位，对系统的应用和安全性有何影响，以具体能力位为例说明。

关闭cap\_dac\_read\_search后，将无法对文件进行root读限制，所有文件将拥有root读权限，普通用户创建一个普通用户权限的文件也可以对一些root文件进行读操作，造成了系统重要信息的泄露。

关闭cap\_sys\_date后，任何用户都可以修改系统的时间，有些软件需要读取主机的时间来确定一些信息，会造成不必要的泄露和损失。

关闭cap\_chown后，任何用户都可以随意操作文件的所述者和组，会导致普通用户相当于拥有了root权限。

c) 比较能力位和chroot管理权限的差异

chroot形成的是一个虚拟环境，在虚拟环境中给予用户有限的权限，安全性中等，因为隶属于root，被攻破会影响到root，也会影响其他进程，并不是一个完全的jail。

而且，在需要root权限的时候，还是直接给root身份，只不过限制了资源，没有限制身份。

能力位是系统对所有资源的授权，将root的特权划分为不同的小部分，资源需要root权限的哪一条，就给资源这一条能力，是对root特权的细粒度分配，分配更加合理，达到最小授权的目的。