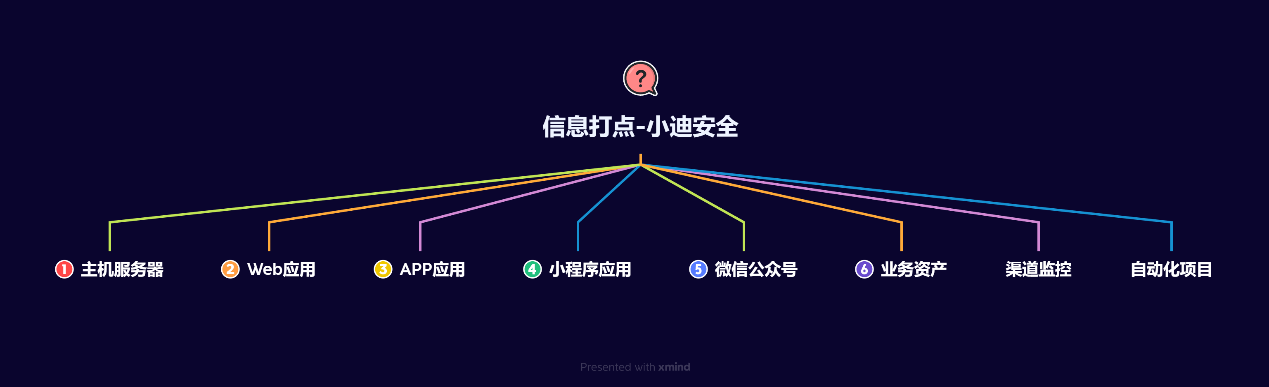
信息收集-Web应用&JS架构&URL提取&数据匹配&Fuzz接口&WebPack分析&自动化



#知识点：  
1、信息收集-服务器系统-操作系统&IP资产  
2、信息收集-服务器系统-端口扫描&服务定性  
---------------------------------------------------  
1、信息收集-Web应用-机构产权&域名相关性  
2、信息收集-Web应用-DNS&证书&枚举子域名  
---------------------------------------------------  
1、信息收集-Web应用-架构分析&指纹识别  
2、信息收集-Web应用-架构分析&WAF&蜜罐  
3、信息收集-Web应用-架构分析&框架组件识别  
---------------------------------------------------  
1、信息收集-Web应用-源码获取-已知指纹&未知指纹  
2、信息收集-Web应用-源码获取-泄漏问题&发现指纹  
---------------------------------------------------  
1、信息收集-Web应用-JS提取分析-人工&插件\*项目  
2、信息收集-Web应用-JS提取分析-URL&配置&逻辑  
  
#章节点：(待补充)  
#具体点：(待补充)

|  |  |  |
| --- | --- | --- |
| **标签** | **名称** | **地址** |
| FUZZ测试 | ffuf | <https://github.com/ffuf/ffuf> |
| 匹配插件 | Hae | <https://github.com/gh0stkey/HaE> |
| JS提取 | JSFinder | <https://github.com/Threezh1/JSFinder> |
| JS提取爬虫 | URLFinder | <https://github.com/pingc0y/URLFinder> |
| WebPack分析 | Packer-Fuzzer | <https://github.com/rtcatc/Packer-Fuzzer> |
| JS匹配插件 | BurpAPIFinder | <https://github.com/shuanx/BurpAPIFinder> |
| JS提取 | LinkFinder | <https://github.com/GerbenJavado/LinkFinder> |
| WebPack分析 | jjjjjjjjjjjjjs | <https://github.com/ttstormxx/jjjjjjjjjjjjjs> |
| JS提取爬虫 | FindSomething | <https://github.com/momosecurity/FindSomething> |
| JS匹配插件 | Unexpected\_information | <https://github.com/ScriptKid-Beta/Unexpected_information> |
| FUZZ字典 | 字典集合 | [https://wordlists.assetnote.io](https://wordlists.assetnote.io/) |

演示案例：

* Web应用-架构分析-WAF&蜜罐识别
* Web应用-架构分析-框架组件指纹识别
* Web应用-源码获取-已知指纹&未知指纹
* Web应用-源码获取-泄漏问题&发现指纹
* Web应用-JS前端分析-人工&提取项目
* Web应用-JS前端分析-插件&流量匹配

#操作系统  
1、Web大小写  
2、端口服务特征  
3、TTL值判断返回  
  
#IP资产  
归属地查询  
归属云厂商  
IP反查机构  
IP反查域名  
IP-C段查询  
这里可以转换成Web上，机构上，从而获取更多信息  
  
#端口资产  
1、网络资产引擎：直接使用网络测绘引擎(Fofa、Hunter、Quake、00信安等搜索IP；此类网络资产测绘引擎都是每隔一段时间会对全网的网络资产去做一个轮询，那可能每个引擎的轮询周期、扫描精准度都不同，故建议可使用不同的搜索引擎以获取更多资产。  
地址导航：https://dh.aabyss.cn/  
参考：https://mp.weixin.qq.com/s/FRgPQKJDj2xRCduwPfZrTw  
  
2、在线端口扫描：百度或google直接搜索在线端口扫描就会有一些网站，同理很多功能都可以直接搜索在线xxx；例如在线正则提取解析、在线编码转换等。  
  
3、本地离线工具：推荐使用Nmap、Masscan、Fscan、KScan，其中Nmap最为准确，但最慢；Masscan最快，误报相对更高。  
端口扫描：https://xz.aliyun.com/t/15753  
演示：Yakit Nmap TscanPlus FScan Tanggo等  
考虑：1、防火墙 2、内网环境  
可能出现案例：数据库端口开放，但进行端口扫描，发现数据库端口没有开放（排除防火墙）\*注意：扫描中选择扫描协议是绕过安全组防火墙设置的一种手法，具体成功需看出网入网配置  
  
#应用服务  
见上图端口协议对应服务应用  
  
#角色定性判定  
1、网站服务器  
2、数据库服务器  
3、邮件系统服务器  
4、文件存储服务器  
5、网络通信服务器  
6、安全系统服务器

主动信息收集：  
通过直接经过目标服务器网络流量的信息收集方式。  
  
被动信息收集：  
不与目标系统直接交互的情况下获取信息收集方式。  
  
域名：  
1、备案信息  
通过域名查备案信息，备案信息获取更多域名  
2、企业产权  
通过企业产权查询Web,APP,小程序等版权资产  
3、域名相关性  
Whois信息：例如域名所有人、域名注册商、邮箱等。  
通过域名注册接口获取后缀  
查询域名注册邮箱  
通过域名查询备案号  
通过备案号查询域名  
反查注册邮箱  
反查注册人  
通过注册人查询到的域名在查询邮箱  
通过上一步邮箱去查询域名  
查询以上获取出的域名的子域名  
  
子域名：  
在后续测试中，还要注意对子域名进行筛选整理，太多的垃圾子域名和没用的子域名，主要看你的收集的子域名方法决定。  
1、DNS数据  
以DNS解析历史记录查询域名资产  
2、证书查询  
以SSL证书解析查询域名资产  
3、网络空间  
多网络空间综合型获取的记录  
4、威胁情报  
各类接口的集成的记录  
5、枚举解析  
结果主要以字典决定  
https://github.com/knownsec/ksubdomain  
https://github.com/shmilylty/OneForAll  
6、JS提取子域名  
后续会讲到

#Web架构  
开源CMS：  
Discuz、WordPress、PageAdmin、蝉知等  
  
前端技术：  
HTML5、Jquery、Bootstrap、Vue、NodeJS等  
  
开发语言：  
PHP、JAVA、Ruby、Python、C#、JS、Go等  
  
框架组件：  
SpringMVC、Thinkphp、Yii、Tornado、Vue等  
  
Web服务器：  
Apache、Nginx、IIS、Lighttpd等  
  
应用服务器：  
Tomcat、Jboss、Weblogic、Websphere等  
  
数据库类型：  
端口扫描，组合判断，应用功能  
Mysql、SqlServer、Oracle、Redis、MongoDB等  
  
操作系统信息：  
Linux、Windows、Mac等  
  
应用服务信息：  
FTP、SSH、RDP、SMB、SMTP、LDAP、Rsync等  
  
CDN信息：  
帝联、Cloudflare、网宿、七牛云、阿里云等  
  
WAF信息：  
创宇盾、宝塔、ModSecurity、玄武盾、OpenRASP等。  
  
蜜罐信息：  
HFish、TeaPot、T-Pot、Glastopf等  
  
其他组件信息：  
FastJson、Shiro、Log4j、Solr、XStream等  
  
#指纹识别：  
1、平台：  
https://www.yunsee.cn/ http://finger.tidesec.net/  
https://fp.shuziguanxing.com/  
  
2、项目：  
https://github.com/AliasIO/wappalyzer  
https://github.com/HackAllSec/hfinger  
https://github.com/Tuhinshubhra/CMSeeK  
https://github.com/lemonlove7/EHole\_magic  
https://github.com/emo-crab/observer\_ward   
3、使用网络空间的ico识别  
借助目标的ic图标进行识别  
  
#WAF识别：  
拦截页面，identywaf项目内置  
https://github.com/stamparm/identYwaf  
https://github.com/EnableSecurity/wafw00f  
  
#蜜罐识别：  
1、项目识别：  
https://github.com/graynjo/Heimdallr  
https://github.com/360quake/quake\_rs  
quake.exe init apikey值  
quake.exe honeypot 目标  
2、人工识别：  
\*端口多而有规律性（针对多服务蜜罐）  
\*Web访问协议就下载（针对多服务蜜罐）  
\*设备指纹分析（见上图，针对蜜罐特征）  
  
#框架组件识别：  
后续会讲到更多识别方法及项目插件  
例子：  
FastJson：https://forum.butian.net/share/1679  
Shiro：https://mp.weixin.qq.com/s/j1nDnb0Ub5bk2-Tq5tt2pg

参考：  
https://www.secpulse.com/archives/124398.html  
https://mp.weixin.qq.com/s/QgLDdaefXlZtvlSiFQShZw  
https://mp.weixin.qq.com/s/zhjxW2mUEgq6dFAp3kBcgQ  
  
#源码泄漏原因：  
1、从源码本身的特性入口  
2、从管理员不好的习惯入口  
3、从管理员习惯的开发入口  
4、从管理员不好的配置入口  
5、从管理员不好的意识入口  
6、从管理员资源信息搜集入口  
源码泄漏大概集合：  
Webpack打包泄漏  
composer.json  
GitHub源码泄漏  
git源码泄露  
svn源码泄露  
网站备份压缩文件  
DS\_Store文件泄露  
hg源码泄漏  
SWP 文件泄露  
CVS泄露  
Bzr泄露  
WEB-INF/web.xml 泄露  
  
思路点：  
1、使用指纹识别平台获取到目标是什么名字的源码程序  
通过搜索引擎获取到源码  
  
2、使用指纹识别平台不能获取目标信息  
后端：借助svn git composer.json ds\_store 备份扫描等配置泄漏安全  
前端：webpack打包 谷歌插件获取源码  
如果没有这些问题 还想获取源码怎么办 见下面方法  
  
3、借助资源平台去搜索（github gitee oschina等源码）  
-特定文件  
-特征联系方式  
-github监控（ 除了源码，漏洞更新，工具更新等都有用处）  
目标使用的源码是开源还是闭源，是否公开  
  
#演示案例：  
1、已知指纹识别获取源码途径  
  
2、GIT泄漏-未知指纹识别获取源码途径  
3、SVN泄漏-未知指纹识别获取源码途径  
4、DS\_Store泄漏-未知指纹识别获取源码途径  
GIT：https://github.com/lijiejie/GitHack  
SVN：https://github.com/callmefeifei/SvnHack  
DS\_Store：https://github.com/lijiejie/ds\_store\_exp  
通用：https://github.com/0xHJK/dumpall  
  
5、WebPack打包-未知指纹识别获取源码途径  
https://github.com/NothingCw/SourceDetector-dist  
  
6、composer.json-未知指纹识别获取源码途径  
7、源码文件备份泄漏-未知指纹识别获取源码途径  
目录文件扫描工具获取存在，直接访问  
 8、Github资源搜索-未知指纹识别获取源码途径  
\*拓展：配合后期监控保证第一时间通知  
资源搜索语法：  
in:name test #仓库标题搜索含有关键字   
in:descripton test #仓库描述搜索含有关键字   
in:readme test #Readme文件搜素含有关键字   
stars:>3000 test #stars数量大于3000的搜索关键字   
stars:1000..3000 test #stars数量大于1000小于3000的搜索关键字 forks:>1000 test #forks数量大于1000的搜索关键字   
forks:1000..3000 test #forks数量大于1000小于3000的搜索关键字 size:>=5000 test #指定仓库大于5000k(5M)的搜索关键字 pushed:>2019-02-12 test #发布时间大于2019-02-12的搜索关键字 created:>2019-02-12 test #创建时间大于2019-02-12的搜索关键字 user:test #用户名搜素   
license:apache-2.0 test #明确仓库的 LICENSE 搜索关键字 language:java test #在java语言的代码中搜索关键字   
user:test in:name test #组合搜索,用户名test的标题含有test的  
关键字配合谷歌搜索：  
site:Github.com smtp   
site:Github.com smtp @qq.com   
site:Github.com smtp @126.com   
site:Github.com smtp @163.com   
site:Github.com smtp @sina.com.cn   
site:Github.com smtp password   
site:Github.com String password smtp  
  
9、另类目标源码获取  
https://www.huzhan.com/  
https://www.goofish.com/  
从目标转到寻找源码系统上，从源码系统上找应用目标  
  
#源码泄漏实战文章  
渗透测试实战---某资金盘信息收集(.idea目录泄露)  
https://mp.weixin.qq.com/s/7cSrDZci\_drE6wTGvQOOHw  
SRC挖掘-js.map泄露到接管云上域控  
https://mp.weixin.qq.com/s/QZsNjbTiaqC1qwzARVAiGQ

在日常渗透测试中，从JavaScript(JS)文件中提取信息是一项关键的步骤，以往有些案例就是通过JS文件中发现的敏感信息从而拿下重要的系统。在JS开发的WEB应用和PHP，JAVA,NET等区别在于即没有源代码，也可以通过浏览器的查看源代码获取部分源代码逻辑。从而获取URL，获取JS敏感信息，获取代码传参等，所以相当于JS开发的WEB应用大部分属于白盒测试（默认有大量源码参考），一般会在JS中寻找更多的URL地址，在JS代码逻辑（加密算法，APIkey配置，验证逻辑等）进行后期安全测试。  
  
0、JS信息收集  
 接口URL提取  
 后端地址提取  
 敏感信息提取  
 配置信息提取  
 其他信息提取（环境，注释，用途等）  
  
1、JS安全问题  
 源码泄漏，代码审计，JS逆向分析  
 未授权访问=JS里面分析更多的URL访问确定接口路径  
 敏感Key泄漏=JS文件中配置接口信息（云应用，短信，邮件，数据库等）  
 API接口安全=（代码中加密提交参数传递，更多的URL路径）  
  
2、流行的Js框架有那些？  
 Vue NodeJS jQuery Angular等  
  
3、如何判定JS开发应用？  
 插件wappalyzer  
 源程序代码简短  
 引入多个js文件  
 一般有/static/js/app.js等顺序的js文件  
 一般cookie中有connect.sid  
   
4、如何获取更多的JS文件？  
 手工-浏览器搜索  
 半自动-Burpsuite插件  
 工具化-各类提取&FUZZ项目  
   
5、如何快速获取价值信息？  
 src=  
 path=  
 method:"get"  
 http.get("  
 method:"post"  
 http.post("  
 $.ajax  
 http://service.httppost  
 http://service.httpget  
  
案例：  
0、人工JS中提取URL信息  
1、从JS中提取到URL信息  
2、从JS中提取到敏感信息  
3、从JS中提取到配置信息  
4、从JS中Fuzz提取到更多  
5、WebPack打包器信息获取  
  
#项目插件工具：  
https://github.com/ffuf/ffuf  
用Go编写的快速 Web 模糊测试程序。  
  
https://github.com/gh0stkey/HaE  
HaE是一款网络安全（数据安全）领域下的框架式项目，采用了乐高积木式模块化设计理念，巧妙地融入了人工智能大模型辅助技术，实现对HTTP消息（包含WebSocket）精细化的标记和提取。  
  
https://github.com/Threezh1/JSFinder  
一款用作快速在网站的js文件中提取URL，子域名的工具  
  
https://github.com/pingc0y/URLFinder  
URLFinder是一款快速、全面、易用的页面信息提取工具  
用于分析页面中的js与url,查找隐藏在其中的敏感信息或未授权api接口  
  
https://github.com/rtcatc/Packer-Fuzzer  
针对Webpack等前端打包工具所构造的网站进行快速、高效安全检测的扫描工具  
  
https://github.com/shuanx/BurpAPIFinder  
攻防演练过程中，我们通常会用浏览器访问一些资产，但很多未授权/敏感信息/越权隐匿在已访问接口过html、JS文件等，通过该BurpAPIFinder插件我们可以：  
1、发现通过某接口可以进行未授权/越权获取到所有的账号密码、私钥、凭证  
2、发现通过某接口可以枚举用户信息、密码修改、用户创建接口  
3、发现登陆后台网址  
4、发现在html、JS中泄漏账号密码或者云主机的Access Key和SecretKey  
5、自动提取js、html中路径进行访问，也支持自定义父路径访问 ...  
  
https://github.com/GerbenJavado/LinkFinder  
功能类似于JSFinder，但JSFinder好久没更新了。  
  
https://github.com/ttstormxx/jjjjjjjjjjjjjs  
针对webpack站点，爬取网站JS文件，分析获取接口列表，自动结合指纹识别和fuzz获取正确api根，可指定api根地址（针对前后端分离项目，可指定后端接口地址），根据有效api根组合爬取到的接口进行自动化请求，发现未授权/敏感信息泄露，回显api响应，定位敏感信息、敏感文件、敏感接口。支持批量模式。支持需认证接口自动尝试bypass。  
  
https://github.com/momosecurity/FindSomething  
该工具是用于快速在网页的html源码或js代码中提取一些有趣的信息的浏览器插件，包括请求的资源、接口的url，请求的ip和域名，泄漏的证件号、手机号、邮箱等信息。  
  
https://github.com/ScriptKid-Beta/Unexpected\_information  
BurpSuite插件用来标记请求包中的一些敏感信息、JS接口和一些特殊字段，  
防止我们疏忽了一些数据包，使用它可能会有意外的收获信息。

涉及资源：[资源下载地址](https://docs.qq.com/doc/DQ3Z6RkNpaUtMcEFr)