API接口深度发现的动态爬虫实现(1)

原创 扫到漏洞的 李姐姐的扫描器 2025年04月30日 19:38 北京

背景

目前，各大SRC接收的漏洞中，API接口漏洞占比极高。例如：

```
1 越权、未授权访问、敏感接口暴露、信息泄漏、并发、SSRF等
```

在躺着自动化捡漏洞这个领域，**早些年，得资产者得天下**。 白帽子手上掌握的目标资产越全，越容易捡到漏洞。主要是：

```
1 域名、IP、端口、指纹、业务特性、框架、字典等
```

如今，**资产中最重要的部分，已经变成了API接口**。

同样的Web目标入口：

- 1) 静态爬虫扫描器：获取API接口能力接近为**0**
- 2) 动态爬虫扫描器： 若无法带身份扫描，获取API接口接近**5%**
- 3) 动态爬虫扫描器 + 支持认证： 如果不具备深度发现API接口的功能，接近发现**30%**
- 4) Burp Suite + 人工导航： 接近发现**40%**，很多接口，测试同学的账号找不到功能入口，不具备身份权限

  因此，现在白帽子挖不到漏洞，一个重要的门槛，是没有所需权限的测试帐号，比如商家端、企业类型的帐号。在获取测试账号这一环节会浪费较多时间

- 5) 动态爬虫+Burp+Fuzz+人工： 幸运的情况下，能接近**50%**

上面的说法显得夸张，但实际情况是，**绝大部分web应用，暴露到前端功能界面，Javascript源代码中的，往往都只是一个API子集**。 后面API还多着呢，看不见，测不到，才是更为普遍的情况。

总结几个主要的观点：

1) API接口的自动化发现已经成为黑盒Web漏扫的决定性因素之一

2) 接口发现通常需要依赖动态爬虫，需要JS解释器

3) 需要解决身份认证的问题，没有认证信息，多数接口请求不会成功

4) 需要支持复杂的交互，智能的交互，触发API调用

5) 需要能够fuzz，发现没有在JS中引入的更多接口

6) 需要获取到接口的参数名和参数值，用于漏洞测试

Crawlergo vs Rad vs BurpSuite

笔者对白帽子常用的几个爬虫工具进行了评估，分别是：

1) **crawlergo 0.4.4（开源）**

2) **Rad 1.0（不开源）**

3) **Burp Suite Professional 2025.1.4（不开源）**

不可否认，这3个工具都非常优秀。 然而，在API深度发现方面，它们都存在一定的不足。

Crawlergo的问题

Crawlergo是一个非常优秀的开源项目，能看出来作者分析解决了很多细节问题。这是一份极具参考价值的源代码。作者写了十分详尽的文章介绍实现细节，值得阅读

https://www.anquanke.com/post/id/178339

现在的版本存在以下问题：

1) 表单填充的逻辑较为简单，作者没有进一步完善

2) 自动填表在Element UI等情况下无法工作，示例代码

```
<input type="text" class="el-select__input" autocomplete="off" role="combobox"
```

上述元素是一个combo 下拉框输入，crawlergo只看到是一个text input。

3) 会遗漏API接口，测试中发现modal框中触发的交互请求，是没有捕获到API接口的

4）**作者提到了拦截HTTP请求返回204，锁定导航。 实测大部分情况有效，一部分情况下未能锁定，因为，导航时甚至都没有新的Request产生**

5）大量无效的请求和误报，前端打包后产生的大量无效的URL，被请求后写入结果

6）工具没有提供交互登录支持（rad burp有）

总结： 交互深度不够、无效结果多，接口发现不够

Rad的问题

Rad是长亭发布的爬虫工具，运行效率很高，并且支持 --wait-login 交互登录。

它的问题是结果不够稳定，多次运行，结果不一致，一会儿多，一会儿少（站点本身是稳定的）。

释放的leakless.exe文件在PC上被会识别为木马。

由于没有代码，没有准确定位到具体问题。但整体API接口发现率同样不足。

总结： 结果不确定、接口发现不够

Burp Suite的问题

当前Burp Suite是支持动态爬虫的。

但是，离谱的是它并不支持直接固定和保存认证身份。

用户交互得到的身份认证凭据，应该被扫描器视为一个高成本、高价值的数据。

显然，Burp这块设计是存在不足的。

如下图所示

只有2种模式： 1) 录入账号密码 2) 录制登录行为序列并重放

问题是，**重放一般是在大型网站是无法成功的，现在很多扫码登录、验证码单次登录的**。重放登录只对一些小网站有点效果罢了。

看文档，提到能**配置cookie jar，测试发现，对headless chrome无效**。cookie jar的规则并不能被初始化到动态爬虫中，这里出现巨坑

因为无法修改Burp的chromium启动参数，我尝试去已启动的chromium 进程目录下手工注入登录会话，结果测试也失败了。Burp不会复用这个父进程的user dir。

总结：内部的cookie jar无法预注入到chromium爬虫，爬虫效果待验证。

验证方案为：

通过本地127.0.0.1反向代理目标网站，默认注入Cookie，burp爬虫自带身份爬取目标网站。

总结

本篇笔者先介绍测试结论：目前常见的爬虫工具中，对API接口深度发现存在一定的不足。这个爬虫的复杂度在于：

1） 登录、获取身份、维持身份

2） 爆破、fuzz接口

3）自动填表、自动触发API请求（覆盖率不足）

4）其他通用爬虫的基础难题： 去重、效率等

未完待续

1) **crawlergo 0.4.4（开源）：**

https://github.com/Qianlitp/crawlergo

2) **Rad 1.0（不开源）：**

https://github.com/chaitin/rad

3) **Burp Suite Professional 2025.1.4（不开源）：**

https://portswigger.net/burp

修改于2025年05月01日