链滴

怎么自定义安全检测规则

作者：nanolikeyou
原文链接：https://ld246.com/article/1539054715466
来源网站：链滴
许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

对于应用软件的分析常见的分类方式有以下两种:一种按照程序代码的文本形式分为二进制和源代码分，这种分类法更多是从技术实现的角度出发;另一种按照是否运行程序代码，将程序分析方法分为动态静态分析，这种分类方式更容易说清漏报和误报的发生根源.动态分析是通过真实或模拟环境执行程序现漏洞（黑盒），而静态分析不执行程序，通过对源代码或二进制文件等进行漏洞检测（白盒）。由程序漏洞大多使用隐蔽的数据或函数.所以动态检测的方式很难发现大部分的程序漏洞。通过静态分析代码或二进制文件来发现程序的威胁效果是最好的。目前静态代码分析主要有以下几种方式：

1. 缺陷模式匹配:这种方式的原理是建立一个缺陷模式库，将需要检测的源代码与缺陷模式库进行匹配来发现代码中存在的安全问题。这种方式的优点是简单易实现，缺点则是误报率高，且需要缺陷模式足够强大。

2. 类型推断:这种检测方式是通过对代码中运算对象类型进行分析，保证每条语句都针对正确的类型进执行。这种方式需要制定一套类型推理的规则。类型推断技术能够检测出代码中的类型错误，检测代质量方面的一些问题，但对于安全问题的检测不够适用。

3. 模型检查:这种检测方式将代码抽象成一个自动机系统，每条语句抽象为其中的一个状态，通过分析个状态机，分析出代码中的问题。这种方式能够检验出程序并发等时序特性，对安全问题的检测也不适用。

4. 数据流分析:这种方式是从分析程序的语法出发，通过分析控制流图，得到程序中所有变量的定值信，包括变量的值、赋值、 传递等信息。这种方式需要首先对程序进行完善的语法解析，工作难度较大但实现后，能够有效的分析出安全问题的传播， 降低静态分析的误报率，因此对于安全分析意义较为大。

那么以上的类型如何落地呢？当前的我们的工作场景主要是通过fuzzing、反复黑盒测试发现应用系中发现的安全问题，为了提高效率，增大漏洞检测覆盖面，通过优化商业安全工具或者自研静态代码计工具来提升1、4方面的技术效果进行提升，可以从技术上验证实现了对此类接口的二次开发，容许行自定义规则。当前的首要工作需要逐步根据casestudy和基于经验的一些漏洞收集分析规则，作为续工作的重要储备，最好有案例或者错误代码，团队审议如何编制加入这些自定义检测规则。

暂时将规则大致为CSRF，XSS，SQL注入，命令执行，JSON劫持，信息泄露，水平权限，服务器访控制，源代码泄露，逻辑设计，权限控制，上传漏洞，Git仓库潜在风险，安全配置不当，URL重定向CRLF漏洞，ssrf，暴力破解、运行时暴露敏感信息明文，降低误报率。合入试运行的时候，采用命名-TEST开头的标志