面向开源供应链的包管理工具建模及实现

计理强[1]，佟祥正[1]，张世新[2]，燕季薇[2]，晏荣杰[1]，严俊[1,2]

联系方式：yrj@ios.ac.cn

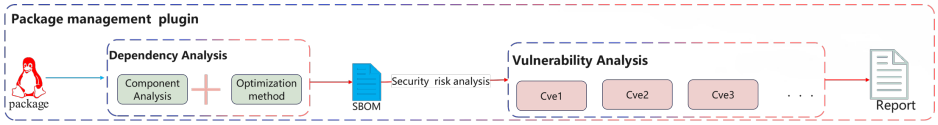
[1] 中国科学院软件研究所 基础软件与系统重点实验室（计算机科学国家重点实验室）
[2] 中国科学院软件研究所 软件工程技术研究开发中心

研究背景

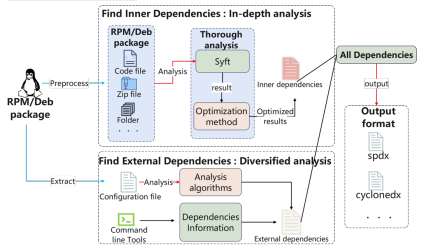
当前开源供应链的包管理工具无法有效分析软件包的安全漏洞，为这些软件包进行成分分析提供软件物料清单 (SBOM) 是最有效的手段，我们聚焦于Linux生态的RPM和Deb软件包，开发一套分析系统，对软件包进行依赖和漏洞分析，可以提高软件包的安全性。

系统简介

整个系统分为依赖分析和漏洞分析两个模块，输入的软件包通过依赖分析输出多种格式的软件物料清单，该清单作为输入参与后续的漏洞分析，最终生成软件包的全局依赖图和安全漏洞报告

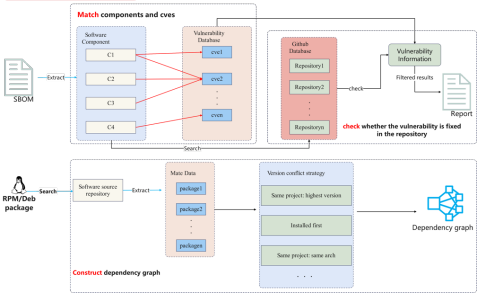


依赖分析



- 通过软件包预处理，实现深层次的分析，对初步的分析结果进行优化，得到内部依赖
- 解析软件包的配置文件并使用命令工具获取外部依赖
- 最终汇总两种依赖，支持导出多种格式的物料清单

漏洞分析



- 通过物料清单解析出所有依赖，在漏洞数据库中匹配对应的CVE，额外在Github仓库中检验漏洞是否解决
- 获取软件包元数据，使用版本冲突解决策略，构建全局的依赖图

工具效果

Project	Benchmark	syft	OPenSCA	Dependency-check	Murphy	Ours
	Depend	Depend	Depend	Depend	Depend	Depend
ears	6	5	0	4	0	5
fat-jars	1	1	1	1	0	1
jars-in-jar	5	4	1	3	0	4
original-jars	3	2	1	3	0	3
recompiled-jars	2	2	1	1	0	2
wars	5	3	5	3	0	4
exclusion-support	2	2	2	0	2	2
interpolated-variables	2	2	2	0	2	2
project-aggregation	2	1	2	0	2	2
project-inheritance	2	2	2	0	2	2
scopes	3	2	3	0	2	3
version-range	3	3	1	0	3	3
Sum	36	29	21	15	13	33

与主流依赖分析工具的准确率比较

Tools	ACC
Syft	80. 56%
Dependency-check	41. 67%
Murphy	36. 11%
OPenSCA	58. 33%
Ours	**91. 67%**

我们在efda的12个java项目中进行了测试，Benchmark列给出项目中依赖的基准数据，Depend列出依赖的软件包个数，ACC为准确率；其中syft, OPenSCA, Dependency-check, Murphy是目前主流的开源依赖分析工具。

总结

我们聚焦于Linux生态的RPM和Deb软件包，使用深层扫描的机制，聚合内部和外部依赖信息构建物料清单，在漏洞分析阶段使用软件仓库验证机制提高漏洞检测准确率，提出版本冲突解决机制并构建更准确的全局依赖图。实验表明，我们的方法可以有效分析软件中的依赖，大大提高依赖分析的准确率。