

# SCA项目第二阶段报告：从npm锁文件生成SBOM

## 一、项目概述

我们SCA项目的第二阶段成果，核心是做一个实用工具——把js项目里的`package-lock.json`文件，转换成标准化的SBOM软件组件清单。

基于上一阶段对npm的调研，我们已经确认：js项目的`package-lock.json`（也就是“锁文件”）里，包含了项目所有的依赖信息，不管是开发者直接引入的依赖，还是这些依赖又间接依赖的组件，都能在里面找到。

本阶段我们的具体分工如下：

- **毛懿诺** 主要负责“输入解析”相关工作：如果依赖锁文件被打包在zip压缩包里，他会把这个压缩包处理好，从中提取出`package-lock.json`，并给出提取后文件的具体存放地址，方便后续处理；
- **沈高畅** 负责核心的“格式转换”工作：接收提取好的锁文件，把里面的依赖信息（比如组件名称、版本等）整理好，转换成SBOM格式的文件；
- **罗康灵** 该阶段不参与代码开发，主要承担统筹协调工作，同时负责撰写项目报告、对工具进行测试，保证工具能正常使用。

## 二、核心模块与功能

项目包含4个核心Java类文件，各模块功能如下：

### 1. 提取器接口 (`ExTractor.java`)

- **作用**：定义提取目标文件（`package-lock.json`）的标准接口。
- **核心方法**：`Path[] extract(String zipDirPath, String outputDirPath) throws IOException`，用于从指定的ZIP文件目录中提取目标锁文件，并返回提取后的文件路径数组。

### 2. 提取器实现类 (`ExTractorImpl.java`)

- **作用**：实现`ExTractor`接口，负责从ZIP文件中提取`package-lock.json`（或`package_lock.json`）。
- **核心功能**：
  - **查找ZIP文件**：在指定目录（`zipDirPath`）中查找单个ZIP文件（`findSingleZipFile`方法）。
  - **解压ZIP文件**：将找到的ZIP文件解压到临时目录（`extractZipFile`方法）。
  - **查找目标锁文件**：在解压后的目录中递归查找`package-lock.json`或`package_lock.json`，优先选择路径层级最少的文件（`findAllTargetFiles`方法）。
  - **复制文件到输出目录**：将找到的锁文件复制到指定输出目录，并按序号命名（如`package-lock_01.json`），避免重名（`copyAllToOutput`方法）。
  - **日志配置**：通过`setupLogging`方法配置控制台日志，输出操作过程信息（如找到ZIP文件、解压路径等）。

### 3. SBOM构建器 (`SbomBuilder.java`)

- **作用**：将`package-lock.json`的JSON数据转换为符合CycloneDX 1.4格式的SBOM结构。
- **核心功能**：
  - **基础信息设置**：定义SBOM的格式（`bomFormat: "CycloneDX"`）、版本（`specVersion: "1.4"`）等元数据。

- **主组件信息**：从锁文件中提取项目名称、版本、许可证等信息，作为SBOM的主组件（`metadata.component`）。
- **依赖组件解析**：遍历锁文件中的`packages`节点，提取每个依赖的名称、版本、许可证、下载地址（`resolved`）、完整性校验值（`integrity`）等信息，生成`components`数组。
- **去重与标准化**：通过`purl`（Package URL，如`pkg:npm/[name]@[version]`）对依赖组件去重，确保SBOM的唯一性。

#### 4. 主程序类（`Main.java`）

- **作用**：程序入口，处理命令行参数，协调提取锁文件和生成SBOM的流程。
- **核心逻辑**：
  - **参数处理**：接收两个命令行参数（输入路径和输出SBOM路径），若输入为`.json`文件（假定为`package-lock.json`），则直接使用；否则视为ZIP文件，调用`ExtractorImpl`提取锁文件。
  - **SBOM生成**：通过`SbomBuilder`将锁文件解析为CycloneDX格式的SBOM，并写入指定输出文件（默认`sbom.json`）。
  - **异常处理**：捕获流程中的异常并输出错误信息，确保程序稳定退出。

### 三、工作流程

1. **输入处理**：用户提供输入路径（可为`package-lock.json`、ZIP文件或包含ZIP的目录）和输出SBOM路径。
2. **锁文件获取**：
  - 若输入为`*.json`文件，直接作为锁文件使用。
  - 若输入为ZIP相关路径，通过`ExtractorImpl`提取ZIP中的`package-lock.json`。
3. **SBOM生成**：`SbomBuilder`解析锁文件，生成包含主组件和依赖组件信息的CycloneDX格式SBOM。
4. **输出结果**：将生成的SBOM写入指定文件（默认`sbom.json`）。

### 四、补充与完善方向

#### 1. 补充漏洞库匹配

通过将 SBOM 中的组件信息与公开漏洞库匹配，输出组件关联的安全漏洞详情。

#### 2. 完善现有功能

- 目前输入和输出的路径不太符合用户直觉，需修改成合理的交互方式
- 程序的异常处理和日志输出不够完善，用户可能无法有效判断程序终止的原因，需增加异常和日志信息
- 目前只能处理单个锁文件，若一个压缩包中有多个锁文件只能处理第一个，未来可以增加锁文件的处理能力输出多个 `SBOM`
- 目前解压压缩包的临时文件仍储存在C盘，时间长后或处理大型压缩包后可能会对磁盘空间有较大影响，可以在程序退出时删除临时文件以释放空间

### 五、总结

本项目作为SCA项目的第二阶段成果，完成了从npm锁文件到CycloneDX 1.4格式SBOM的自动化转换，通过团队分工协作，完成了“输入-提取-解析-输出”的流程

当前工具已具备基础可用性，但仍存在许多瑕疵，后续将聚焦上述短板优化，实现轻量级SCA工具，满足源组件的合规管理与安全风险识别需求。