ICS・35・080
L・77

团　体　标　准

T/CECC 005—2020

软件代码开源成分与安全检测指南

Guidelines for Open source components analyze and code security audit

2020 年 05 月 13 日 发布　　2020 年 05 月 13 日　实施

中国电子商会

目　次

前　言

本标准按照 GB/T 1.1—2009 给出的规则起草。

请注意本文件的某些内容可能涉及专利，本文件的发布机构不承担识别这些专利的责任。

本标准由中国电子商自主可控技术委员会提出。

本标准由中国电子商会归口。

本标准起草单位：苏州棱镜七彩信息科技有限公司、赛迪智库网络安全研究所、北京中测安华科技有限公司、中国航天科工集团 706 研究所、交通运输信息安全中心有限公司、国网信通产业集团、中金金融认证中心有限公司。

本标准主要起草人：罗峋、但吉兵、易焕腾、覃业博、王超、韦安垒、周鸣爱、魏昂、刘彦钊、田斌、付修峰、曾颖明、李宁、戴明、杜渐、郭璐、李飞、靳鑫、张大健、纪崇廉。

软件代码开源成分与安全检测指南

1 范围

本标准规定了软件代码开源成分及其安全性检测的检测过程及方法，描述了代码中开源成分及其安全性检测的检测条款。

本标准适用于软件开发者和管理者对软件代码中开源成分信息及其安全风险的检测活动。

本标准不涉及源代码安全的传统测试方案如静态扫描、模糊测试、代码审计；不涉及在开源代码（成分）中挖掘新的安全漏洞；不涉及检测标准的评价体系。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅所注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 28452-2012 信息安全技术-应用软件系统通用安全技术要求

GB/T 30279-2013 信息安全技术-安全漏洞等级划分指南。

3 术语和定义

GB/T 25069界定的以及下列术语和定义适用于本文件。

3.1 开源软件 Open source software

开源软件是指那些源代码公开，在符合许可证要求的前提下，可以被自由使用、复制、修改和再发布的一系列软件的集合。

3.2 开源成分 Open source components

软件代码中引用了开源项目的源代码、二进制文件或存在引用依赖关系的代码成分。

3.3 开源许可证 Open source license

开源软件的作者对用户使用其开源软件的法律许可。

3.4 克隆 Clone

在同一个或者多个不同的源代码文件中，通常存在一些完全相同或者非常相似的代码段或代码文件，引入此类代码文件和代码段或其引用的操作即称为克隆。

3.5 知识库 Knowledge database

一种存储特定领域数据的数据库或数据表。

4 概述

4.1 检测目的

软件代码开源成分与安全检测（以下简称“开源检测”）的目的是通过检测分析出软件（含软硬件结合设备）源代码中开源代码成分的构成情况、开源代码成分对应的已知安全漏洞信息、开源代码带来的软件许可证风险信息。从而提高对软件项目开源代码构成成分与安全风险信息的掌握，规避开源安全风险。

4.2 检测场景

开源检测分为自主检测和第三方检测两个场景。自主检测由项目成员自发开展，通过检测提高软件安全性，发现风险和预防安全问题发生。自主检测可参考本指南，流程上可适当裁剪（如签署保密协议、项目背景调研、熟悉代码、检测入场、信息调研等）。第三方检测由第三方检测机构进行，一般具有专业性（部分可能存在强制性），流程上建议完全参照本指南。在进行第三方检测前需进行较多准备工作（如签署保密协议、项目背景调研、熟悉代码、检测入场、信息调研等）。检测工作应于项目验收之前开展。由于第三方检测涉及较多外部因素，进行检测前应提前通知项目成员，降低检测工作延期风险。在项目代码发生重要变更后，应根据情况重新进行检测。

4.3 检测人员

检测人员应依次进行检测准备、检测实施、反馈跟踪工作。检测人员应具备代码安全与软件测试的专业知识，熟悉使用开源成分与安全检测工具；具备客观反映代码问题的工作素质；对项目代码严格保密。

4.4 检测内容

开展检测首先需要制订并遵循检测条款（内容与条款可参见第 6、7、8 章节）。检测条款应包括但不限于成分信息检测条款、漏洞信息检测条款、许可证信息检测条款 3 个方面的具体条款。其中，成分信息检测条款 3 条；漏洞信息检测条款 3 条；许可证信息检测条款 3 条，总计 9 条检测条款。检测时可根据被检测的具体对象及应用场景对上述各条款进行调整制定。

4.5 检测方法

开源检测过程包括工具扫描和人工核实两个步骤，实现对待测条款的逐一检测，所有检测条款均检测完毕并输出检测结果则视为检测完成。检测条款应根据项目实际情况进行调整，以提高检测质量。

由于开源检测需要海量开源数据知识库做支撑，因此需要借助工具检测。同时，工具检测结果存在一定误报现象，需要人工参与审核。开源检测应将人工核实与工具检测相结合进行检测工作，采用专业开源检测工具对代码进行检测，形成检测报告。对于检测工具的检测结果中的误报问题，由检测人员人工进行审核。对于漏报问题应使用不同检测工具进行多次检测，以减少漏报。

检测实施前应制定合理的检测实施计划，检测实施过程中应严格按照检测实施计划进行检测工作，并制定人员分工方案。

5 检测过程

5.1 总体流程

检测过程中应按照检测准备、检测实施、检测报告、结果处置四个阶段依次进行。检测准备阶段应进行签署保密协议、项目背景调研、熟悉代码、检测条款制定、检测环境准备；检测实施阶段应进行检测入场、信息调研、工具检测、人工核实工作；检测报告阶段应进行检测结果的总结、陈述和建议等工作；结果处置阶段由项目开发人员对检测出的问题进行修复，修复完成后应对更改的代码进行再次检测（回归测试），直到问题被彻底解决（对于暂时无法修复的风险问题，可采取其他方法进行风险规避或接受）。开源成分与安全检测流程如图 1 所示。

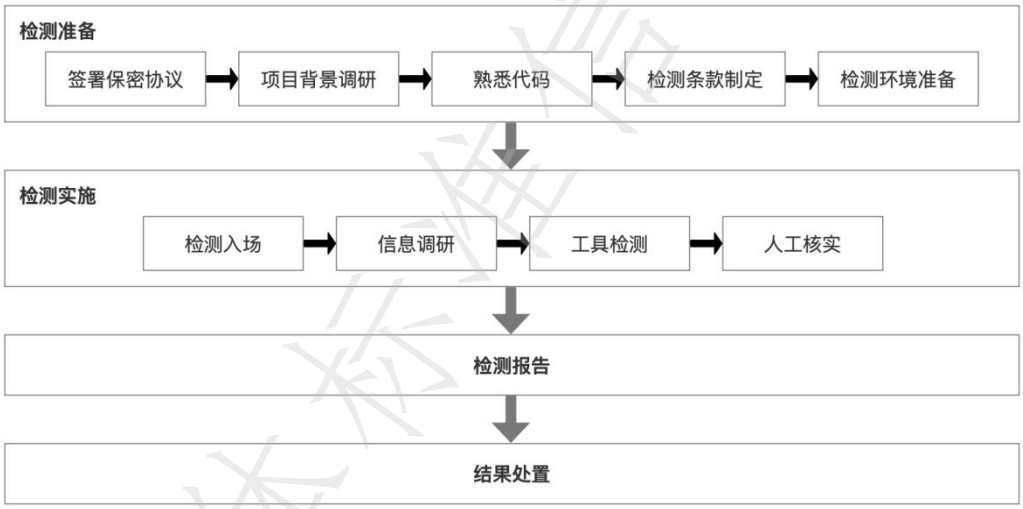

图 1 开源检测流程

5.2 检测准备

5.2.1 签署保密协议

为有效保密检测方提供的项目代码，防止项目代码泄露或公开揭露，签订保密协议。明确责任，约束义务方，保护源代码与知识产权，减少因项目代码泄露为双方造成的损失。

5.2.2 项目背景调研

通过项目成员访谈等方式，了解项目代码的使用场景、目标客户、项目内容、开发规范、已知的开源成分等。从而为误报排查和人工核实提供参考依据。

5.2.3 熟悉代码

通过与项目成员沟通和阅读代码，了解项目基础结构、代码规范、项目架构等。

5.2.4 检测条款制定

通过明确检测目的、背景调研、熟悉代码等工作，定义开源检测关键内容，制定开源检测检测条款（内容与条款可参见第 6、7、8 章节）。

5.2.5 检测环境准备

开源安全检测应在封闭的网络环境（内网环境）开展工作。参与检测工作的服务器、计算机、笔记本以及其他软硬件设备等不能与外部网络互联，以确保软件源代码安全，防范源代码泄露风险。

5.3 检测实施

5.3.1 检测入场

检测人员和项目成员应于检测实施前进行检测入场工作。由检测人员对检测的检测目标、检测方法、检测过程进行介绍。由项目成员对项目状态、项目团队构成、项目背景、项目功能结构等进行介绍。

5.3.2 信息调研

通过项目成员访谈等方式获得项目代码、项目文档、以及其它项目相关信息。了解项目代码业务逻辑。调研了解项目的已知开源成分引用情况。

5.3.3 工具检测

根据代码开源成分与安全检测条款内容，可借助工具实现高效检测。检测工具应包括如下功能：项目基础信息检测、项目开源成分（又称克隆检测）检测、开源项目溯源检测、开源代码漏洞检测、开源许可证侵权检测等。

注：常用的检测工具包括 Black Duck、FOSSID、棱镜七彩开源安全检测平台等，宜选择支持私有云部署或本地部署的检测工具。

5.3.4 人工核实

工具检测完成后，由人工对工具检测的结果一一进行核实，对于工具检测的误报与漏报结果应予以记录。

5.4 检测报告

对项目成员介绍初始检测结果，并由项目成员对检测偏离项进行解释说明，并提供其它相关信息，形成检测报告。检测报告包括检测的总体描述、改进建议、检测结论等内容，并对可能产生的安全风险进行高、中、低分类描述。检测结论给出每条检测条款的检测结果描述。检测报告应明确列明使用的检测工具及其版本信息。

5.5 结果处置

由项目成员对检测出的问题组织召开评审会进行结果处置。对于可以修复的风险问题，对代码变更和相关解释说明进行记录存档。对于变更后的项目代码，应再次进行检测并重新出具检测报告以确保问题得到解决。对于暂时无法修复的风险问题，应有相应的解释说明未修复原因，并可采取其他方法进行风险规避。对于未修复的风险，也可采取持续风险监督，保持风险跟踪。

6 成分信息检测

6.1 项目基本信息检测

6.1.1 检测目的

通过对项目基础信息检测，使开发者更客观全面的认识了解项目，使项目新介入者或第三方检测者更快速的了解项目基本情况，为后续全面的检测工作做好基础和铺垫。

6.1.2 检测内容

项目基本信息检测：即对软件源代码工程中所包含的所有基础信息进行检测。项目基本信息检测文件类型格式（如.c/.java/.cpp/.dll/.txt/.php）及分布情况、文件的数量个数、文件的存储容量、文件所使用的编程语言等。

6.1.3 检测输出结果

项目基本信息检测输出结果宜包括但不限于工程文件中所包含的文件类型格式（如.c/.java/.cpp/.dll/.txt/.php）及分布情况、文件的数量个数、文件的存储容量、文件所使用的编程语言及分布情况、源代码的行数统计等。

6.1.4 检测必要条件

检测工具宜具备文件系统扫描相关技术。

6.1.5 检测方法

通过人工或工具方式，宜从文件类型格式及分布情况、文件的数量个数、文件的存储容量、文件所使用的编程语言及分布情况、源代码的行数等维度，分别对软件工程文件进行分析，并对分析结果进行归纳统计。

6.2 开源代码成分检测

6.2.1 检测目的

通过开源代码成分检测，检测识别软件工程中的开源成分，为软件代码自主比率的测评

提供客观依据。为后续开源代码漏洞检测、开源许可证风险检测提供基础数据，从而降低信息安全风险。

6.2.2 检测内容

开源代码成分检测：即在软件工程的代码文件中，有效检测识别出与开源项目高度相似的文件或代码片段，将其认定为开源代码成分。

6.2.3 检测输出结果

开源代码成分检测输出结果宜包括但不限于：

a）软件工程中所有文件数量；

b）软件工程中被认定为开源成分的文件数量；

c）开源成分占总文件数量的比例情况；

d）与开源项目文本相似的文件信息。包括文件名称，开源项目的文件名称、两文件彼此的相似度并高亮展示相似片段；

e）与开源项目结构相似的文件信息。包括文件名称，开源项目的文件名称、两文件彼此的相似度并高亮展示相似片段；

f）与开源项目代码片段相似的文件信息。包括文件名称，开源项目的文件名称、两文件彼此的相似度并高亮展示相似片段。

6.2.4 检测必要条件

检测工具宜具备开源代码知识库并满足以下要求：

a）收录主流国内外开源社区或网站；

b）开源代码知识库宜经常保持更新，保证源代码库的时效性；

c）宜包含目前主流编程语言。

6.2.5 检测方法

通过工具和人工相结合的方式，将软件工程中的所有文件逐一与开源代码知识库中的文件进行文本比对、文件结构比对、代码片段比对，从而识别检测出与开源项目高度文本相似、结构相似及代码片段相似的文件和代码片段。

检测工具宜运用代码相似度检测技术，对如下情况具备检测识别能力：

a）对开源代码修改函数名、变量名、头文件等；

b）对开源代码，新增、删除部分代码片段；

c）对开源代码，扰乱了代码片段或函数的文本顺序；

d）新增、删除或者修改了大量代码注释；

e）新增、删除换行符、空格符等。

6.3 代码溯源信息检测

6.3.1 检测目的

通过对源代码溯源信息的检测，检测识别软件工程中开源成分的详细信息，为开源代码成分分析、开源代码漏洞分析、开源许可证风险分析、开源成分供应链组织风险分析提供补充数据，从而提高分析质量、效率、全面性。

6.3.2 检测内容

代码溯源信息检测宜对如下信息进行检测：

a）**源代码溯源信息**：即在软件工程的代码文件中，有效检测识别出项目中所包含的所有代码格式为源代码的开源项目的详细信息；

b）**二进制组件溯源信息**：即在软件工程的代码文件中，有效检测识别出项目中所包含的所有代码格式为二进制代码的开源项目的详细信息；

c）**引用依赖溯源信息**：即在软件工程的代码文件中，有效检测识别出项目中所引用依赖的开源项目的详细信息。

6.3.3 检测输出结果

代码溯源信息检测宜包含如下检测输出结果：

a）**源代码溯源信息**：宜包括但不限于溯源项目名称、创建时间、代码托管地址、项目描述、许可证名称、最新版本号、文件匹配数、有效文件数、匹配文件名称、开源社区所属国家、项目主要贡献者及其国籍；

b）**二进制组件溯源信息**：宜包括但不限于二进制组件名称、创建时间、组件描述、开源社区所属国家、项目主要贡献者及其国籍等；

c）**引用依赖溯源信息**：宜包括但不限于项目中引用的开源项目的项目文件名称、项目文件路径、项目版本号、开源社区所属国家、项目主要贡献者及其国籍等。

6.3.4 检测必要条件

检测工具宜具备开源代码知识库并满足以下要求：

a）开源代码知识库数据：包含但不限于每个开源项目的项目名称、创建时间、代码托管地址、项目描述、项目热度、项目版本号、开源社区所属国家、项目主要贡献者及其国籍；

b）开源代码知识库宜具备开源二进制组件信息数据；

c）其它开源代码知识库要求与【6.2.4】所述相同。

6.3.5 检测方法

运用将项目代码与开源项目相匹配的匹配检测技术，与开源代码知识库中的开源项目进行匹配，检测出项目所包含的开源项目和所依赖的开源项目，提取出其中的详细信息。

匹配检测技术宜对如下情况具备检测识别能力：

a）可以文件为粒度进行匹配检测；

b）可以文件结构为粒度进行匹配检测；

c）可以代码片段为粒度进行检测；

d）可以追溯代码中引用依赖的开源项目。

7 漏洞信息检测

7.1 源代码漏洞检测

7.1.1 检测目的

源代码漏洞检测检测目的如下：

a）开源项目代码是软件代码的一个重要组成部分，而开源项目代码漏洞繁多，进行开源项目漏洞检测是保障代码安全的必需环节；

b）通过对软件工程中源代码漏洞信息的检测，与二进制组件漏洞检测、引用依赖漏洞检测互为补充，形成开源安全检测的闭环。

7.1.2 检测内容

源代码漏洞信息：即在软件工程的代码文件中，有效检测识别出项目中所包含的所有代码格式为源代码的开源项目的安全漏洞信息。

7.1.3 检测输出结果

源代码漏洞检测宜具备如下检测输出结果：

a）宜包含软件工程中存在漏洞的开源项目数量、名称、代码托管地址、项目描述、漏洞数量；

b）宜包含但不限于匹配项目漏洞的危害等级、漏洞等级、发布时间、漏洞描述、参考链接、受影响产品、补丁参考链接、解决方案。

7.1.4 检测必要条件

检测工具宜具备开源项目漏洞知识库并满足以下要求：

a）宜从各个国家认可的漏洞网站（如 CNVD/CNNVD/CVE/NVD）收集漏洞数据，对各网站的漏洞数据进行整合处理，形成统一的漏洞数据字典，源代码漏洞知识库宜建立项目-组件-漏洞-版本-文件等关联和映射关系；

b）开源项目漏洞知识库宜包含源代码漏洞数据；

c）开源项目漏洞知识库宜经常更新，保证漏洞库的时效性；

d）宜包含目前主流开发语言的开源项目漏洞；

e）宜包含但不限于项目级、版本级、文件级等多维度的漏洞数据；

f）漏洞的分级分类和描述应符合国家标准。

7.1.5 检测方法

根据漏洞数据字典匹配出开源项目/组件/文件所对应的漏洞信息，并定位到具体的版本，从漏洞数据字典中查询出该漏洞的解决方案。

源代码漏洞检测宜满足如下要求：

a）对漏洞数据进行分析整理，具有项目级、版本级、文件级、代码级等多维度粒度的漏洞；

b）支持在检测出漏洞后，对漏洞提供验证方案和解决方案。

7.2 二进制组件漏洞检测

7.2.1 检测目的

二进制组件漏洞检测检测目的如下：

a）开源项目二进制代码是软件代码的一个重要组成部分，而二进制代码漏洞繁多，进行二进制代码漏洞检测是保障企业代码安全的必需环节；

b）通过对软件工程中二进制漏洞信息的检测，与许可证风险检测、源代码漏洞检测、引用依赖漏洞检测互为补充，形成开源安全检测的闭环。

7.2.2 检测内容

二进制组件漏洞信息：即在软件工程的代码文件中，有效检测识别出项目中所包含的所有代码格式为二进制代码的开源项目的安全漏洞信息。

7.2.3 检测输出结果

二进制组件漏洞检测宜具备如下检测输出结果：

a）宜包含软件工程中存在漏洞的二进制组件数量、名称、代码托管地址、组件描述、漏洞数量；

b）宜包含但不限于匹配组件漏洞的危害等级、漏洞等级、发布时间、漏洞描述、参考链接、受影响产品、补丁参考链接、解决方案。

7.2.4 检测必要条件

检测工具宜具备开源项目漏洞知识库并满足以下要求：

a）宜从国家认可的漏洞网站（如 CNVD/CNNVD/CVE/NVD）获取漏洞数据，对各主流网站的漏洞数据进行整合处理，形成统一的漏洞数据字典；

b）开源项目漏洞知识库宜包含二进制组件漏洞数据；

c）开源项目漏洞知识库宜经常更新，保证漏洞库的时效性；

d）宜包含目前主流开发语言的开源项目漏洞；

e）漏洞的分级分类和描述应符合国家标准。

7.2.5 检测方法

根据漏洞数据字典匹配出二进制组件所对应的漏洞信息，从漏洞数据字典中查询出该漏洞的解决方案。

检测工具宜满足如下要求：

a）对漏洞数据进行分析整理，归纳成完整漏洞信息；

b）支持在检测出漏洞后，对漏洞提供验证方案和解决方案。

7.3 引用依赖漏洞检测

7.3.1 检测目的

引用依赖漏洞检测检测目的如下：

a）开源项目所引用依赖开源项目的代码是软件代码的一个重要组成部分，而引用依赖开源项目的代码漏洞繁多，进行引用依赖开源项目漏洞检测是保障代码安全的必需环节；

b）通过对软件工程中引用依赖开源项目漏洞信息的检测，与源代码漏洞检测、二进制组件漏洞检测互为补充，形成开源安全检测的闭环。

7.3.2 检测内容

引用依赖漏洞信息：即在软件工程的代码文件中，有效检测识别出项目中所引用依赖的开源项目的漏洞信息。

7.3.3 检测输出结果

引用依赖漏洞检测宜包含如下检测输出结果：

a）宜包含软件工程中存在漏洞的开源项目数量、名称、代码托管地址、项目描述、漏洞数量；

b）宜包含但不限于匹配组件漏洞的危害等级、漏洞等级、发布时间、漏洞描述、参考链接、受影响产品、补丁参考链接、解决方案。

7.3.4 检测必要条件

检测工具宜具备开源项目漏洞知识库并满足以下要求：

a）宜从国家认可的漏洞网站（如 CNVD/CNNVD/CVE/NVD）获取漏洞数据，对各主流网站的漏洞数据进行整合处理，形成统一的漏洞数据字典，宜建立项目-组件-漏洞-版本-文件等关联和映射关系，建立开源安全漏洞库；

b）漏洞库宜包含源代码漏洞数据；

c）开源项目漏洞知识库宜经常自动更新，保证漏洞库的时效性；

d）宜包含目前主流开发语言的开源项目漏洞；

e）宜包含但不限于项目级、版本级、文件级多级别的漏洞数据；

f）漏洞的分级分类和描述应符合国家标准。

7.3.5 检测方法

通过代码静态分析技术分析提取项目所依赖的开源项目，根据漏洞数据字典匹配出开源项目/组件所对应的漏洞信息，并定位到具体的版本，从漏洞数据字典中查询出该漏洞的解决方案。

引用依赖漏洞检测宜满足如下要求：

a）对漏洞数据进行分析整理，具有项目级、版本级等多维度粒度的漏洞；

b）支持在检测出漏洞后，对漏洞提供验证方案和解决方案。

8 许可证信息检测

8.1 开源许可证信息检测

8.1.1 检测目的

开源许可证信息检测的目的如下：

a）通过对开源许可证信息的检测，为后续的许可证商用风险检测提供基础数据支撑；

b）通过对开源许可证信息的检测，为后续基于不同许可证信息对比分析的许可证兼容性检测工作提供基础数据；

c）通过对开源许可证信息的检测，使开发者对引入包含对应许可证的开源项目后的基本注意事项形成认识；

d）通过对开源许可证信息的检测，使开发者对相应许可证的义务要求形成认识；

e）通过对开源许可证信息的检测，使开发者对相应许可证的政策要求形成认识。

8.1.2 检测内容

开源许可证信息检测宜对如下信息进行检测：

a）**源代码许可证信息**：即在软件工程的代码文件中，有效检测出文件格式为源代码的开源许可证信息，此开源许可证包括对项目进行许可的项目级许可证、对项目内文件进行许可的文件级许可证、对项目内代码段进行许可的代码级许可证三类；

b）**二进制许可证信息**：即在软件工程的代码文件中，有效检测出格式为二进制代码的开源项目的开源许可证的信息；

c）**引用依赖许可证信息**：即在软件工程的代码文件中，有效检测出项目中所引用依赖的开源项目的开源许可证的信息。

8.1.3 检测输出结果

宜包含但不限于项目所包含开源项目许可证的许可证名称、许可证类型、许可证所在文件路径、条款、原文等信息。

8.1.4 检测必要条件

检测工具宜具备对应开源许可证知识库，该知识库宜满足以下要求：

a）宜包含目前主流许可证的许可证信息；

b）宜包含但不限于许可证名称、许可证类型、许可证所在文件路径、条款、原文等信息；

c）开源许可证库宜经常更新，保证漏洞库的时效性。

8.1.5 检测方法

从开源项目原文件中分析提取出开源项目所包含的许可证，从开源许可证知识库中匹配检测出该许可证所对应的许可证信息。

匹配检测技术宜提取出项目中许可证名称、许可证类型、许可证所在文件路径、条款、原文等许可证相关信息。

8.2 许可证兼容性检测

8.2.1 检测目的

许可证兼容性检测的目的如下：

a）部分许可证与其它部分许可证因其中的条款存在冲突而无法存在于同一个开源项目中，为避免引入的部分开源项目因许可证兼容问题而给项目开发方带来损失，需对许可证兼容性进行检测，在项目开发阶段即发现问题并解决，避免或减少不必要的损失；

b）通过对软件工程中开源许可证兼容性信息的检测，与许可证商用风险检测和开源项目漏洞检测互为补充，形成开源安全检测的闭环；

c）部分许可证兼容性问题会导致许可证商用性问题，通过对软件工程中开源许可证兼容性信息的检测，为许可证商用风险检测提供数据基础。

8.2.2 检测内容

许可证兼容性信息：许可证信息中关于一个许可证与其它许可证是否兼容相关的信息。

8.2.3 检测输出结果

许可证兼容性检测输出结果宜包含但不限于:

a）开源项目中互相冲突的所有许可证的相应冲突条款/描述、冲突许可证名称、冲突许可证描述、冲突信息来源、冲突信息更新时间；

b）软件工程中存在开源许可证兼容性问题的开源项目的项目数量、项目名称、项目托管地址、许可证名称。

8.2.4 检测必要条件

检测工具需具备对应开源许可证知识库，该知识库宜满足以下要求：

a）从广泛的渠道挖掘开源许可证兼容性数据或自主进行许可证兼容性分析，于开源许可证知识库中包含与许可证兼容性相关的许可证信息，宜包含但不限于如下许可证兼容性信息：互相冲突的所有许可证的相应冲突条款/描述、冲突许可证名称、冲突许可证描述、冲突信息来源、冲突信息更新时间；

b）其它开源许可证知识库要求与【8.1.4】相同。

8.2.5 检测方法

从开源项目原文件中分析提取出开源项目所包含的许可证，从开源许可证库中匹配出该许可证所包含的许可证兼容性信息，以及该开源项目的项目基本信息。

8.3 许可证商用风险检测

8.3.1 检测目的

许可证商用风险检测的目的如下：

a）部分许可证仅限于非商业项目用途或用于商业存在一定风险，为避免引入的部分开源项目因许可证商用限制而给项目开发方带来损失，需对许可证商用性进行检测，在项目开

发阶段即发现问题并解决，避免或减少不必要的损失；

b）通过对软件工程中开源许可证商用风险信息的检测，与许可证兼容性检测和开源项目漏洞检测互为补充，形成开源安全检测的闭环。

8.3.2 检测内容

许可证商用风险信息：许可证信息中关于该许可证是否可商用以及商用风险的有关条款与内容信息。

注：商用风险指能为公司带来经济损失、公关问题的风险。

8.3.3 检测输出结果

许可证商用风险检测输出结果宜包含但不限于:

a）软件工程中存在商用风险的开源许可证数量、名称、类型；

b）软件工程中包含存在商用风险的许可证的项目名称、数量、代码托管地址、项目描述、许可证数量、商用风险许可证数量。

8.3.4 检测必要条件

检测工具需具备对应开源许可证知识库，该知识库宜满足以下要求：

a）开源许可证知识库宜包含与许可证商用风险相关的许可证信息；

b）其它开源许可证知识库要求与【8.1.4】相同。

8.3.5 检测方法

从开源项目原文件中分析提取出开源项目所包含的许可证，从开源许可证知识库中匹配出该许可证所包含的与商用风险相关的许可证信息。