**平时作业1-第2章开源合规与许可证**

**要求：**选取Apache社区中的一个开源项目，选取“开源软件许可合规检测工具”进行扫描。最终生成详细的分析报告。对开源软件许可证合规工具不做具体限制，可从下述工具中，认真挑选一个质量较高，报告全面的工具。

**报告内容要求**：1.介绍选取的开源项目，2. 介绍许可证合规工具的安装过程及遇到的问题，3. 给出工具输出的检测报告；4.针对工具输出报告进行详细分析。内容不超过5页。

**作业命名要求：班级-学号-姓名-作业1，例如，软件1601-20160001-张三-作业1**

**参考工具列表：**

**现有****开源软件许可合规工具**

**1、AboutCode Toolkit**

**「官网」**：AboutCode

*https://www.aboutcode.org/*

**「许可证」**：Apache-2.0

**「摘要」**：AboutCode Toolkit 和 About 文档提供了一种简单的方式来记录你在项目中使用的第三方软件组件的来源、许可、使用和其他重要或有趣的信息。此外，这个工具能够生成归属声明，并识别你的项目中使用的可重新分配的源代码。

**2、AboutCode Manager**

**「官网」**：AboutCode

*https://www.aboutcode.org/*

**「许可证」**：Apache-2.0

**「摘要」**：AboutCode Manager 提供了一个先进的可视化用户界面，帮助你快速评估由ScanCode 识别的许可证和其他通知，并记录你对一个组件的有效许可证的结论。AboutCode Manager是基于 Electron的，是使用 nexB 的 AboutCode 工具的主要桌面/UI 工具。

**3、Apache Rat**

**「官网」**：Apache Rat

*http://creadur.apache.org/rat/*

**「许可证」**：Apache-2.0

**「摘要」**：Apache Rat 是一个发布审计工具，主要针对许可证。它用 Java 编写，通过 Maven 和 Ant 的插件在命令行中运行。Rat 是可扩展的。它是 Apache Creadur 项目的一部分。

**4、Apache Tentacles**

**「官网」**：Apache Tentacles

*http://creadur.apache.org/tentacles/*

**「许可证」**：Apache-2.0

**「摘要」**：Apache Tentacles 通过自动与包含发布工件的版本库进行交互来帮助审查者。Apache Tentacles 简化了审查由大量工件组成的版本库的工作。它是用 Java 编写的，从命令行运行。

**5、Apache Whisker**

**「官网」**：Apache Whisker

*http://creadur.apache.org/whisker/*

**「许可证」**：Apache-2.0

**「摘要」**：Apache Whisker 协助组装的应用程序保持正确的法律文件。Whisker 可以

验证--针对某一个发行版检查其元数据的质量

生成--从元数据中生成法律文件

对于复杂的组合式应用程序特别有用。

**6、Bang**

**「官网」**：Bang

*https://github.com/armijnhemel/binaryanalysis-ng*

**「许可证」**：AGPL-3.0

**「摘要」**：Binary Analysis Next Generation，即 BANG，是一个分析二进制文件的工具。目前，它的主要目标是非常迅速地找出二进制文件的内容，如固件更新，并使从内容中提取的信息可用于进一步分析，如许可证合规性、安全研究或成分分析。它支持大约 130 种不同的文件格式。

**7、Barista**

**「官网」：**Barista Open Source License and Vulnerability Management Tool

*https://optum.github.io/barista/*

**「许可证」**：Apache-2.0

**「摘要」**：开发者关注：Barista 从根本上说是一个扫描工具，用于检测开源组件、许可证和潜在的漏洞。自动创建和维护开源材料清单，包括多层次的依赖性。

自定义业务规则：Barista 管理员确定与检测到的每个许可证相关的职责或任务，并根据部署模式、适用的许可证和检测到的依赖的已知漏洞，来指派项目的批准状态。

云原生架构：Barista 是为云原生部署环境而设计的，允许托管的灵活性和按需扩展。

**8、Bubbly**

**「官网」**：Bubbly

*https://github.com/valocode/bubbly/*

**「许可证」**：MPL 2.0

**「摘要」**：Bubbly 是一个发布准备平台，帮助软件团队自信地发布符合要求的软件。通过报告和分析获得你的发布过程的可见性，以降低风险，提高质量，减少周期时间，并推动持续改进。

**9、CLA Assistant**

**「官网」**：CLA Assistant

*https://github.com/cla-assistant/cla-assistant*

**「许可证」**：Apache-2.0

**「摘要」**：CLA Assistant 通过让贡献者在拉取请求（pull request）中签署贡献者许可协议（CLA）来帮助处理对仓库贡献的法律问题。CLA 可以存储为 GitHub 的 Gist 文件，然后与 CLA Assistant 中的软件库/组织链接。仓库所有者可以审查每个发布版本的 CLA 的签署用户列表。

**10、Cregit**

**「官网」**：Cregit

*https://github.com/cregit/cregit*

**「许可证」**：GPL-3.0

**「摘要」**：Cregit 可以识别源代码的贡献者。一个源文件的 Cregit 版本有两个互动功能。

  1.鼠标移动：你会得到一个添加此令牌（token）的提交信息的摘要。这些信息是：

（1）它的 commitid

（2）它的 git-author（该提交的 Author 字段的值）

（3）它的 git-author-date（提交的 Author Date 字段的值）。

（4）提交的摘要日志

   2.左键点击一个令牌（token）：会打开一个新的窗口，显示（在 GitHub 中）该提交的细节。你可以一直打开这个窗口，它将不断重新加载文件。

**11、Deltacode**

**「官网」**：AboutCode

*https://www.aboutcode.org/*

**「许可证」**：Apache-2.0

**「摘要」**：DeltaCode 允许您轻松地比较一个软件包、组件、代码库或产品的两个版本的 ScanCode 扫描，以便快速识别可能的变化，重点是识别许可证变化。DeltaCode 报告匹配的文件，并提供一个分数和对该分数造成影响的因素列表。你可以使用 DeltaCode 和 ScanCode 来识别和跟踪开源或第三方软件包或组件中的许可证和不同发布版本之间的相关变化。

**12、Eclipse SW360**

**「官网」**：Eclipse SW360

*https://projects.eclipse.org/projects/technology.sw360*

**「许可证」**：EPL-1.0

**「摘要」**：是一个软件目录应用程序，旨在为一个组织所使用的软件组件信息提供一个集中式分享场所。通过为不同的任务提供独立的后台服务和一组访问这些服务的 portlets，它被设计为简洁地集成到与软件工件和项目的管理有关的现有基础设施中。它有与外部系统（如代码扫描工具）互动的连接器。到目前为止，该项目还没有提供下载信息。

**13、Eclipse SW360antenna**

**「官网」**：Eclipse SW360

*https://projects.eclipse.org/projects/technology.sw360.antenna*

**「许可证」**：EPL-2.0

**「摘要」**：Eclipse SW360antenna 是一个可以尽可能地使你的开源许可证合规自动化的流程工具。归根结底，这就是为你的项目：

1.收集所有与合规性有关的数据。

2.处理这些数据并警告是否存在任何与许可证合规性有关的问题，以及

3.生成一套合规性工件（源代码包、披露文件、报告）

**14、Fossology**

**「官网」**：Fossology

*https://www.fossology.org/*

**「许可证」**：GPL-2.0

**「摘要」**：Fossology 是一款用于许可证、版权和出口控制扫描的工具。只需一次点击，你就可以生成一个 SPDX 文件，或一个包含你所有的软件版权声明的 ReadMe 文档。它提供了一个 Web UI 和一个数据库，用于合规工作流程。要扫描，必须将软件包上传到服务器。提供的扫描器有Monk、Nomos 和 Ninka。它对扫描的软件包有版本控制，所以当扫描以前的软件包的较新版本时，只有新改变的文件被重新扫描。

**15、FOSSLight**

**「官网」**：FOSSLight

*https://fosslight.org/*

**「许可证」**：AGPL-3.0 及其它

**「摘要」**：FOSSLight 是一个可以有效地处理开放源码合规过程的综合系统。它提供了。

合规工作流程：它可以处理开放源码的合规工作流程。

合规中心：你可以管理所有关于开放源码合规性的事情，如许可证，漏洞和其他。

可扩展性：它可以使用额外的功能（包括 FOSSLight 扫描仪或其他插件）。

**16、LDBCollector**

**「官网」**：LDBCollector

*https://github.com/maxhbr/LDBcollector*

**「许可证」**：BSD-3-Clause

**「摘要」**：一个小的应用程序，它收集开源软件许可证元数据并将其合并。

**17、License Compatibility Checker**

**「官网」**：license-compatibility-checker

*https://github.com/HansHammel/license-compatibility-checker#readme*

**「许可证」**：MIT

**「摘要」**：根据 SPDX 标准，检查 npm 依赖的 package.json 的许可证兼容性。这个项目宣称是一项正在进行的工作，但已给出了软件包中的许可证的简单比较，并解释了许可证的允许程度（宽松的 Permissive > 弱保护的 Weakly Protective > Strongly Protective 强保护的> Network Protective 网络保护的），并用一个彩色的图显示潜在的不兼容性。

**18、Licensee.js**

**「官网」**：Licensee.js

*https://github.com/jslicense/licensee.js*

**「许可证」**：Apache-2.0

**「摘要」**：Licensee.js 是一个命令行工具，用于根据规则检查 npm 软件包依赖的许可证元数据。它使用 SPDX 许可证表达式和白名单数据来捕获与白名单不同的许可证下的软件包。

**19、Ninka**

**「官网」**：Ninka

*http://ninka.turingmachine.org/*

**「许可证」**：GPL-2.0

**「摘要」**：Ninka 是一个轻量级的源代码许可证识别工具。它是基于句子的，并提供了一种简单的方法来识别源代码文件中的开源许可证。它能够识别几十个不同的许可证（以及它们的变种）。

**20、Opossum Tool**

**「官网」**：Oposssum Tool

*https://github.com/opossum-tool*

**「许可证」**：Apache-2.0

**「摘要」**：一个轻量级的应用程序，用于审计和清点大型代码库的开源许可证合规性。

OpossumUI 的开发目标是建立一个管理和结合不同来源的开源合规数据的工具。虽然现有的软件合规性分析工具可以提供良好的信息，但由于检测率的提高，使用多种此类工具往往会导致大量的数据。即使可以通过自动工具合并结果和过滤噪音，但最后的人工修改往往是必要的。因此，OpossumUI 诞生了。一个轻量级的应用程序，用于审查大型代码库的合规信息。OpossumUI 是一个用来执行以下任务的工具：

1.发现应用中使用的开放源码软件。

2.审查许可证。

3.从源代码扫描中生成报告。

**21、OSS Attribution Builder**

**「官网」**：OSS Attribution Builder

*https://github.com/amzn/oss-attribution-builder*

**「许可证」**：Apache-2.0

**「摘要」**：OSS Attribution Builder 是一个帮助团队为软件产品创建属性文件的网站。

**22、OSS Discovery by OpenLogic**

**「官网」**：OSS Discovery

*http://ossdiscovery.sourceforge.net/*

**「许可证」**：GPL-3.0

**「摘要」**：OSS Discovery 可以找到嵌入在应用程序中并安装在计算机上的开源软件。它是一个扫描工具，可以给出人类可读和机器可读的结果。

**23、OSS Review Toolkit ORT**

**「官网」**：ORT

*https://github.com/heremaps/oss-review-toolkit*

**「许可证」**：Apache-2.0

**「摘要」**：通过检查源代码和依赖关系来验证自由和开源软件许可证的合规性。它的工作原理是分析源代码的依赖性，下载依赖性的源代码，扫描所有源代码的许可证信息，并对结果进行总结。组成 ORT 的不同工具被设计成具有最小命令行界面（用于编程）的库（用于脚本使用）。目前，报告格式有 Excel 表格、NOTICE 文件、静态 HTM L和 Web App。

**24、OSSPolice**

**「官网」**：OSSPolice

*https://github.com/osssanitizer/osspolice*

**「许可证」**：GPL-3.0

**「摘要」**：OSSPolice 是为开发者提供的风险评估服务，可以快速识别其应用程序中潜在的自由软件许可违规行为和已知的 n-day 安全漏洞。

**25、Quartermaster Project QMSTR**

**「官网」**：QMSTR

*https://qmstr.org/*

**「许可证」**：GPL-3.0

**「摘要」**：Quartermaster 是一套命令行工具和构建系统的扩展，它对软件构建进行分析，以创建FOSS 合规性文件并支持合规性决策。Quartermaster 与软件构建过程相邻运行。一个主进程收集有关软件构建的信息。一旦构建完成，主进程会执行一些分析工具，最后是一些报告者。所有的模块都是在主进程的背景下执行的，而不是在构建机器上。主站运送所有模块的依赖关系，而不影响构建客户的文件系统（它在一个容器中运行）。

**26、ScanCode.io and ScanPipe**

**「官网」**：ScanCode.io

*https://scancodeio.readthedocs.io/en/latest/introduction.html#*

**「许可证」**：Apache-2.0

**「摘要」**：ScanCode.io 是一个服务器，用于对软件构成分析（SCA）的过程进行脚本化和自动化，以识别应用程序代码库中的任何开源组件及其许可证合规性数据。ScanCode.io 可用于各种用例，如 Docker 容器和虚拟机组成分析，以及其他应用。

ScanPipe 是一个对开发者友好的框架和应用程序，帮助软件分析师和工程师建立和管理现实生活中的软件组成分析项目，作为脚本管道。

ScanPipe 为执行和组织这些软件组成分析项目所需的基础设施提供了一个统一的框架。

**27、ScanCode Toolkit**

**「官网」**：ScanCode

*https://www.aboutcode.org/*

**「许可证」**：Apache-2.0

**「摘要」**：ScanCode 是一套命令行工具，可以可靠地扫描代码库中的许可证、版权、软件包清单和直接依赖关系以及其他在源代码和二进制代码文件中发现的有趣的来源和许可证信息。ScanCode 提供全面的扫描结果，你可以将其保存为 JSON、HTML、CSV 或 SPDX。作为一个返回 JSON 的命令行应用程序，ScanCode 很容易集成到代码分析管道和 CI/CD 中。

**28、SCANOSS**

**「官网」**：scanoss.com

*https://www.scanoss.com/*

**「许可证」**：GPL-2.0-or-later

**「摘要」**：SCANOSS 是第一个免费和开放源代码的 SCA 平台和开放数据的 OSS 知识库。它在 SPDX 和 CycloneDX 中执行 SBOM 生成，并在代码片段、文件和组件层面检测开源的存在。中心组件是一个基于 OpenAPI 标准的 RESTful API。为不同的语言和与其他工具的整合提供了参考代码。通过 SCANOSS，你可以在任何工具中实现组件、文件和片段的匹配。公共知识库被称为 OSSKB，可以在 oskb.org 上找到。扫描可以安全地和匿名地进行。

**29、SPDX Tools**

**「官网」**：SPDX Tools

*https://spdx.org/tools*

**「许可证」**：Apache-2.0

**「摘要」**：仅需要一次下载，统一的 SPDX 工作组工具能提供翻译、比较和验证功能。该工具是一个Java 命令行工具，有以下功能：

1.TagToSpreadsheet - 将一个标签格式的输入文件转换为电子表格的输出文件

2.TagToRDF - 将一个标签格式的输入文件转换为 RDF 格式的输出文件

3.RdfToTag - 将一个 RDF 格式的输入文件转换为一个标签格式的输出文件

4.RdfToHtml - 将一个 RDF 格式的输入文件转换成一个 HTML 网页输出文件

5.RdfToSpreadsheet - 将一个 RDF 格式的输入文件转换为一个电子表格格式的输出文件

6.SpreadsheetToRDF - 将一个电子表格输入文件转换为 RDF 格式的输出文件

7.SpreadsheetToTag - 将电子表格输入文件转换为标签格式输出文件

8.SPDXViewer - 显示一个 SPDX 文档输入文件（标签/值或 RDF 格式）。

9.CompareMultipleSpdxDocs - 比较多个 SPDX 文档（标签/值或 RDF 格式），并输出到电子表格。

10.CompareSpdxDocs - 比较两个 SPDX 文档（以标签/值或 RDF 格式）。

11.GenerateVerificationCode - 从一个文件目录中生成一个验证码。

**30、SPDX Maven Plugin**

**「官网」**：SPDX Maven Plugin

*https://github.com/spdx/spdx-maven-plugin*

**「许可证」**：Apache-2.0

**「摘要」**：SPDX Maven 插件是 Maven 的一个插件，为 POM 文件中描述的工件生成 SPDX（软件包数据交换 Software Package Data Exchange）文件。

**31、TraceCode toolkit**

**「官网」**：AboutCode

*https://www.aboutcode.org/*

**「许可证」**：Apache-2.0

**「摘要」**：TraceCode Toolkit 帮助你确定哪些组件是为你的产品实际分发或部署的。这是确定你的开源许可义务的基本信息，因为很多都是通过分发或部署才触发的。TraceCode Toolkit 是一个分析构建的跟踪执行的工具，因此你可以了解哪些文件被构建成二进制文件并最终部署在你的分布式软件中。

**32、Tern**

**「官网」**：Tern

*https://github.com/vmware/tern*

**「许可证」**：BSD-2-Clause

**「摘要」**：Tern 是一个用 Python 编写的用于容器的软件包检查工具。Tern 是一个检查工具，用于查找安装在容器镜像中的软件包的元数据。它通过两个步骤完成这一工作。

1.它使用 overlayfs 来挂载容器镜像中的第一个文件系统层

2.然后，它在 chroot 环境下执行 "命令库 "中的脚本，以收集该层中安装的软件包的信息

3.以这些信息为基础，它继续对容器镜像中的其他层进行步骤 1 和 2 的迭代。

4.一旦完成，它就会生成不同格式的报告。默认的报告是对哪些层带来了哪些软件组件的粗略解释。如果提供了一个Docker文件，那么它也会提供Docker文件中的哪些行被用来创建这些层。

**33、Vulnerability Assessment Tool**

**「官网」**：Vulnerability Assessment Tool

*https://github.com/SAP/vulnerability-assessment-tool*

**「许可证」**：Apache-2.0

**「摘要」**：开源漏洞评估工具支持软件开发组织在应用开发过程中安全使用开源组件。该工具分析 Java 和 Python 应用程序，以检测它们是否依赖具有已知漏洞的开源组件，收集关于在特定应用程序上下文中执行脆弱代码的证据（通过静态和动态分析技术的组合），并支持开发人员减轻这种依赖性。因此，它解决了 OWASP 十大安全风险 A9，亦即使用已知漏洞的组件，这往往是数据泄露的根本原因。