ICS 03.080.01

CCS A 11

团　　体　　标　　准

T/ JDFA01-2024

开源软件安全风险评价标准

Open source software security risk assessment standards

2024-9-10 发布　　**2024-9-10 实施**

江苏省数字金融协会　发 布

目 次

前　言

根据中国人民银行等单位联合发布的《关于规范金融业开源技术应用与发展的意见》，标准编制组经广泛调查研究，认真总结实践经验，参考国内外的相关标准，并在广泛征求意见的基础上，制定本标准。

本标准按照 GB/T 1.1—2020《标准化工作导则 第 1 部分：标准化文件的结构和起草规则》的规定起草。

本标准由中国人民银行江苏省分行提出。

本标准由江苏省数字金融协会归口。

本标准主编单位：中科南京软件技术研究院

本标准参编单位：江苏省软件产品检测中心

江苏省知识产权保护中心

江苏省专利信息服务中心

南京理工大学

南京信息工程大学

南京银行

苏商银行

江苏省联合征信有限公司

江苏省数字化协会

江苏省生产力促进中心产业大数据与软件开发服务中心

联通数字科技有限公司

江苏省国信数字科技有限公司

江苏省软件产业股份有限公司

江苏移动信息系统集成有限公司

中国移动紫金（江苏）创新研究院有限公司

南京数字经济科技学会

南京市企业征信服务有限公司

南京联合产权（科技）交易所

南京金盾公共安全技术研究院有限公司

南京扬子江数字科技发展有限公司

南京可信数据服务有限公司

南京大数据检测技术有限公司

江苏风云科技服务有限公司

金盾检测技术股份有限公司

苏州市软件评测中心有限公司

北京中科微澜科技有限公司

南京睿鲸数字科技有限公司

本标准主要起草人员：刘斌、吴敬征、李千目、吴奕、王亚利、刘琦、张晖、吴真炜、徐小锋、任坚斌、王治平、陈俊、何满怀、王品亮、田健、曾飞、张继栋、施琦、王宏图、包岩、孙凯、吴伟、洪刚、胡成亚、施志晖、王玮、陈志军、戴晔、王雷、黄道芹、付蓉、孔桂兰、何平、杨加钰

开源软件安全风险评价标准

1 范围

标准规范了开源软件安全风险评价流程及指标体系等。

本标准适用于开源软件的安全风险评价。

2 规范性引用文件

下列文件对本文件的应用是必不可少的。凡是注日期的引用文件，仅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。

—《信息安全技术 软件产品开源代码安全评价方法（征求意见稿）》

—《软件供应链安全治理实践指南白皮书（2023）》

—《加快开源软件发展三年行动计划（2023-2025 年）》

—《“十四五”软件和信息技术服务业发展规划》

—《信息安全技术 关键信息基础设施安全保护要求》（GB/T 39204-2022）

—《关键信息基础设施安全保护条例》

—《网络安全审查办法》

—《信息安全技术 术语》（GB/T 25069-2022）

—《ICT 供应链安全风险管理指南》（GB/T 36637-2018）

—《开源软件安全使用规范》（T/CFAS 0001-2019）

3 认定评价原则

开源软件安全风险评价应遵循以下原则：

a） 全面性：评价过程应该覆盖开源软件的各个方面，包括源代码、依赖关系、社区支持、文档质量等。全面性的评估有助于全面了解软件的安全性。

b） 透明性：评价的过程和结果是透明的，能够为利益相关方提供清晰的了解。这包括评估方法、标准、指标的明确说明，以及评估报告的及时发布。

c） 实证性：评价应该基于实际的证据和数据，而非仅仅依赖猜测或主观判断。这可以通过对源代码的溯源、漏洞管理工具的使用等手段来实现。

d） 持续性：安全风险评价是一个持续的过程，而不是一次性的活动。由于软件和威胁环境的不断变化，定期的评估可以及时发现和应对新的安全风险。

e） 合规性：评价过程应该符合相关的法规和标准，例如开源软件的许可证合规性、隐私法规等。合规性的考量有助于确保评价过程的合法性和可信度。

f） 用户中心：评价以最终用户的安全利益为中心，考虑他们的使用场景和需求。这有助于确保评价结果与实际应用场景相匹配。

4 术语与定义

下列术语和定义适用于本文件。

4.1

开源软件 open source software

允许用户查看、修改和重新分发源代码的软件。其许可证通常符合开源定义的标准。

4.2

开源许可证 open Source License

一种具有法律效力的、允许用户自由使用、修改、复制或分发软件代码的授权条款格式合同或协议。

4.3

开源社区 open source community

以开源代码的贡献者为主体，在开源代码贡献过程中形成的具有特定文化、组织结构、运行机制的共同体。

4.4

代码标准合规性 Code Standards Compliance

估开源软件的代码是否符合制定的编码标准，以确保代码的一致性、可读性和可维护性。

4.5

开源软件安全风险评价 Open Source Software Security Risk Assessment

通过系统性的方法评估开源软件中存在的潜在安全风险，包括漏洞、威胁和合规性问题。

4.6

许可证冲突　License conflict

开源许可证冲突是当一个软件项目或者组件中存在多个许可证，处于上层控制域的许可证（如项目许可）和上层控制域的许可证（如代码文件声明的许可）不相同时，若上层许可的各项规定均严格于下层许可，则不存在冲突，反之存在许可证冲突。

5　开源软件安全风险评价流程

开源软件安全风险评价流程为：

A）安全漏洞分析：收集大型开放安全漏洞库、开源社区安全公告、威胁情报等开源软件的漏洞信息，并对收集到的漏洞进行威胁分析、影响范围分析等。

B）代码合规性检测：使用基于图匹配的代码标准合规检测等方法对开源软件的许可证进行检查，确保其符合组织内部政策和法规；使用代码审查工具检测代码中是否存在违反合规性规定的部分。

C）开源许可证合规性分析：使用基于大规模数据库和文本分类算法，获取不同许可证的条款信息来构建精确到条款的开源项目许可证冲突列表，构建开源许可证分析器、许可证嗅探器和计算引擎三部分来进行开源项目许可证冲突检测。通过分析开源软件所使用的许可证，输出准确可靠、清晰易懂、多维度多层次的许可证使用与条款级冲突检测结果，并评估开源软件是否符合公司或项目的许可证合规性标准。

D）软件可维护性分析：评估开源软件项目的活跃度，包括最近的更新频率、提交者活动等，以判断项目是否仍在积极维护。分析开源社区的健康状况，包括社区规模、反馈响应速度等，以了解用户和开发者参与度。分析开源项目的问题跟踪系统，对于 git 仓库，通过获取 gitlog 的 commit 记录来获取更新频率和提交者相关活动。对于 github 仓库可以获取 push 事件、commit comment 事件等 event 事件了解已解决的问题数量和速度，以判断维护团队的响应效率。

E） PoC 判定：分析可能存在的 PoC，了解攻击者如何利用软件漏洞进行攻击。基于 PoC 的分析，评估潜在风险和漏洞的危害程度。

F）APT 攻击跟踪：收集关于高级持续威胁（APT）的情报，包括攻击模式、

使用的工具、攻击技术等。分析开源软件是否与已知的 APT 攻击有关，了解软件是否容易受到特定威胁。获取其最新的 APT 攻击情报，包括攻击模式、使用的工具、攻击技术等，积极参与安全社区，借助专业社区的力量共同收集和分析 APT 攻击相关情报，形成更全面的数据集。通过将外部提供的 APT 攻击工具映射到开源软件的使用情况，深入了解软件是否使用了与已知 APT 攻击相关的工具。对已知 APT 攻击中常见的攻击技术进行分析，验证开源软件是否易受到这些技术的威胁，为软件加固提供指导。

G）漏洞的响应和修复策略：确定修复优先级、选择适当的修复方法和工具，以及规划修复的时间和资源。根据漏洞的严重性和影响范围，可以决定优先修复高风险的漏洞。修复方法可以包括安装补丁、更新软件版本、配置安全设置等。

H）设置权重计算安全风险：制定不同安全风险因素的权重，以反映其在整体风险中的相对重要性。根据安全漏洞、合规性、可维护性等方面的评估结果，计算每个因素的得分。将各个方面的得分整合，计算整体的安全风险得分，用于优先级排序和决策制定。

该评价流程以系统性和综合性的方式评估开源软件的安全风险等级，确保在使用和集成开源软件时能够充分考虑潜在的安全问题。

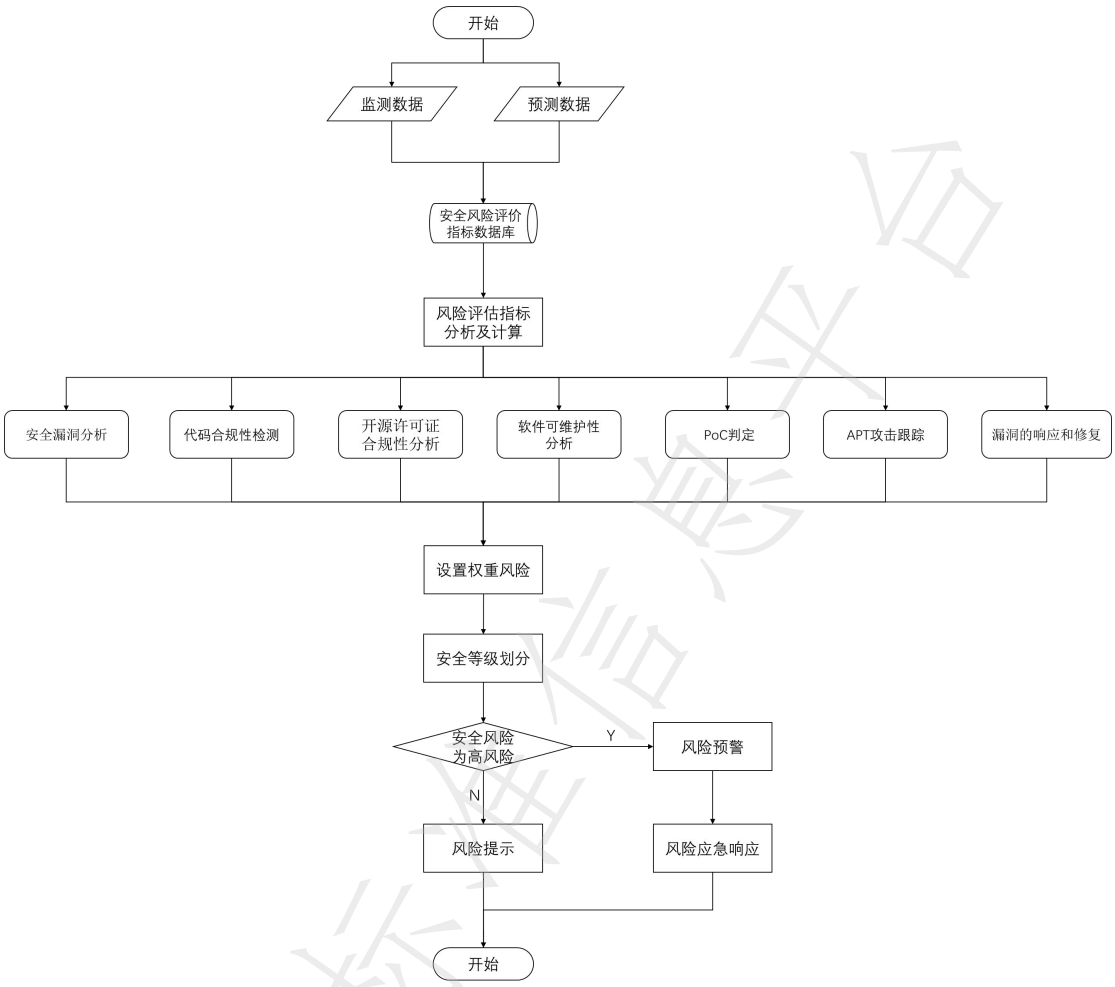

图 1 开源软件安全风险评价流程示意图

6 风险评估和分类

1) 安全漏洞分析：

a) 收集大型开放安全漏洞库、开源社区安全公告、威胁情报等开源软件的漏洞信息；

b) 对收集到的漏洞进行威胁分析，包括攻击方式、危害程度等；

c) 进行影响范围分析，评估漏洞对系统、数据的潜在影响；

d) 判定漏洞的严重性，建立漏洞等级体系。

2) 代码合规性检测：

a) 使用基于图匹配的代码标准合规检测方法对开源软件的许可证进行检查；

b) 确保许可证符合组织内部政策和法规；

c) 使用代码审查工具检测代码中是否存在违反合规性规定的部分；

d) 形成代码合规性报告，包含违规部分的详细信息。

3) 开源许可证合规性分析：

a) 使用大规模数据库和文本分类算法获取不同许可证的条款信息；

b) 构建精确到条款的开源项目许可证冲突列表；

c) 分析开源软件所使用的许可证，输出准确、清晰的许可证使用与条款级冲突检测结果；

d) 评估开源软件是否符合公司或项目的许可证合规性标准。

4) 软件可维护性分析：

a) 评估开源软件项目的活跃度，包括最近的更新频率、提交者活动等；

b) 分析开源社区的健康状况，包括社区规模、反馈响应速度等；

c) 分析开源项目的问题跟踪系统，了解已解决的问题数量和速度；

d) 判断维护团队的响应效率，形成维护性评估结果。

5) PoC 判定：

a) 分析可能存在的 PoC，了解攻击者如何利用软件漏洞进行攻击；

b) 基于 PoC 的分析，评估潜在风险和漏洞的危害程度；

c) 判断 PoC 是否公开，是否存在可用的防护措施。

6) APT 攻击跟踪：

a) 收集关于高级持续威胁（APT）的情报，包括攻击模式、使用的工具、攻击技术等；

b) 分析开源软件是否与已知的 APT 攻击有关；

c) 了解软件是否容易受到特定威胁；

d) 形成 APT 攻击跟踪报告，提供防范建议。

7) 漏洞的响应和修复策略：

a) 识别漏洞：使用漏洞扫描工具或监测系统，及时发现开源软件项目中的潜在漏洞；

b) 评估漏洞严重性：对发现的漏洞进行评估，确定其对系统安全的影响程度，以便有针对性地处理；

c) 寻找已有解决方案：查看开源社区、漏洞数据库等资源，了解是否已经存在关于发现漏洞的解决方案或补丁。

d) 联系维护团队：向开源项目的维护团队报告漏洞，提供详细的漏洞信息和可能的修复建议，包括安装补丁、更新软件版本、配置安全设置等。

e) 跟踪漏洞修复进度：在开源项目的问题跟踪系统中跟踪漏洞修复的进度，了解维护团队的响应速度和解决效率。

f) 定期审查：建立定期的漏洞审查机制，对开源软件项目进行定期评估，确保漏洞响应和修复策略的持续有效性。

8) 设置权重计算安全风险：

a) 制定不同安全风险因素的权重，考虑漏洞、合规性、可维护性等方面的相对重要性；

b) 根据评估结果，计算每个因素的得分；

c) 整合各个方面的得分，计算整体的安全风险得分；

d) 用于优先级排序和决策制定，形成安全风险权重计算。

7 更新和维护

（一）版本确认与官方发布：

a) 开源社区互动： 在版本确认过程中，积极参与相关开源社区，关注官方发布通道，确保及时获取最新版本信息。例如，加入相关开源项目的邮件列表，定期查看和参与讨论，这有助于了解即将发布的版本、解决的问题以及社区的最新动态。关注项目在社交媒体平台上的官方账号，及时获取项目的最新消息、版本发布和重要更新。积极参与社交媒体上的讨论，获取其他用户的反馈和经验。使用RSS

工具订阅项目的更新源，以确保第一时间收到最新的博客文章、新闻和发布通知，及时了解项目的动态。。

b) 验证数字签名：在确认软件版本时，通过验证官方发布的数字签名确保最大限度的完整性和真实性，以防止在下载过程中遭遇潜在的篡改。确保了解项目使用的数字签名算法，以及它们的安全性和可靠性。常见的签名算法包括 RSA、DSA 和 ECDSA。从官方渠道获取开发团队的公钥，确保公钥的完整性。最好是通过多个途径获取公钥，例如通过 HTTPS 访问项目网站、使用 PGP 密钥服务器等。使用专业的签名验证工具，如 GnuPG（GNU Privacy Guard）或类似工具，对下载的软件进行数字签名的验证。验证签名时，了解项目的公开信任模型。了解开发团队如何建立信任链，以确保所使用的公钥是合法和可信的。

（二） 获取与维护活动安全评估：

a) 下载来源验证：对于所有对开源软件的获取过程，确保从官方来源或受信任的镜像站点下载软件，减少中间人攻击的风险。在无法直接从官方获取时，选择使用被广泛认可和信任的镜像站点。确保这些站点是由软件项目或相关社区官方认可的，并定期检查它们的可信度。载软件包之前，检查它是否附带了数字签名。使用开发者的公钥验证签名，以确保软件的完整性和真实性。数字签名提供了对软件包是否被篡改的额外验证层。

b) 使用安全通信协议：在下载、更新或修改软件的过程中，使用安全的通信协议，如 HTTPS，以保护数据的机密性和完整性。针对软件下载和更新的服务器，确保其 SSL/TLS 配置是最新且安全的。使用最新的加密算法，禁用不安全的协议版本，以保护通信的安全性。对于使用镜像站点的情况，验证这些站点是否同样通过 HTTPS 提供服务。避免从未加密的镜像站点下载软件，以减轻中间人攻击和数据篡改的风险。如果在组织内使用代理服务器，确保代理服务器本身也使用安全的通信协议。防止代理服务器成为潜在的漏洞，确

保其不会篡改或劫持下载的软件。定期检查用于加密通信的SSL/TLS证书的有效性。使用过期或被吊销的证书会增加安全风险，因此要及时更新证书。

（三） 遗留系统变更的安全计划与评审：

a) 全面变更分析：在进行任何继承系统变更之前，进行全面的软件安全计划，包括变更类型、模块影响和潜在新的安全风险的分析。对系统的不同模块进行深入分析，明确每个变更对模块的影响。这包括对模块之间的依赖关系、数据流、接口等进行全面的了解，以便及时发现潜在的问题并采取相应的安全措施。通过采用先进的漏洞分析工具和技术，对变更引入的潜在新的安全风险进行详尽的识别。这包括对代码的静态分析、动态测试以及模糊测试等手段，以保证系统在变更后不会引入新的漏洞和风险。

b) 安全评审流程：制定吸引人的安全评审流程，确保所有变更计划均经过专业安全评审，以识别潜在的安全隐患。

i. 安全评审团队组建

建立专业的安全评审团队，包括安全专家、架构师、开发人员和测试人员等。确保评审团队具备全面的技术知识和经验，能够全面审查变更计划的安全性。

ii. 评审文档和会议

要求变更计划提交详细的评审文档，包括变更的目的、实现方式、可能影响等方面的信息。召开定期的评审会议，确保所有关键人员都有机会提出问题和建议，从而更好地发现潜在的安全隐患。

iii. 自动化工具支持

结合自动化安全工具，对变更计划进行自动化审查，以提高评审的效率和全面性。这包括对代码的静态分析、依赖扫描、漏洞检测等工具的使用，以辅助人工评审。

iv. 持续监控与反馈

建立变更后的持续监控机制，及时捕捉潜在的安全问题。同时，建立反馈机

制，确保评审结果能够及时传达给开发团队，以便及时修复和改进。

（四） 持续更新与维护计划：

a) 制定定期的更新与维护计划：

i. 周期性漏洞扫描与评估：

设定周期性漏洞扫描计划，通过自动化工具对系统进行漏洞扫描，并对扫描结果进行详细评估，以及确定漏洞的严重程度和紧急性。

ii. 漏洞报告管理：

建立漏洞报告管理系统，记录每一次漏洞扫描的结果，明确漏洞的描述、影响范围和可能的攻击路径

iii. 制定紧急修复计划：

针对高危漏洞，建立紧急修复计划，确保在最短时间内部署相应的修复措施，以防范潜在的安全风险

b) 应对安全威胁和技术变化的更新计划：建立技术监测团队，定期追踪安全领域的新兴威胁、漏洞和攻击技术，并进行趋势分析，以确保系统始终能够抵御最新的安全威胁。为团队成员提供定期的安全培训，使其保持对新技术和威胁的敏感性，并能够及时适应变化的安全环境。引入自动化工具，监测已安装的开源软件，并根据最新的安全公告自动应用适用的安全补丁。确保系统随时处于最新的安全状态。

c) 安全补丁的及时应用：制定安全补丁测试流程，确保在应用补丁之前对其进行充分的测试，以防止不稳定性和不良影响。设计完善的灾备计划和回滚策略，以应对在安全补丁应用过程中可能发生的问题，确保系统的可用性和稳定性。设定定期的安全审查计划，检查已应用的安全补丁是否有效，并对系统进行进一步的安全评估，以保持系统的整体安全性。

（五） 漏洞响应机制：

a) 快速响应团队成立：设立专门的漏洞响应团队，确保漏洞的快速确认、定级和通知相关团队，以及制定合理的修复计划。

b) 漏洞确认：利用自动化工具和实时监测系统迅速检测潜在漏洞，对漏洞进行定级，根据潜在风险严重性划分为紧急、重要、一般等级。设立有效的通信渠道，确保漏洞信息的及时传达、确定紧急响应计划，确保高危漏洞得到及时处理。

c) 漏洞修复测试：在应用修复之前，进行充分的漏洞修复测试，以确保修复计划的有效性且不会引入新的问题，测修复对系统性能的影响，以保证修复后系统的高效运行。

（六） 技术变化适应性：

a) 跟踪行业和技术的变化，定期进行深入的行业趋势研究，包括新技术的涌现、安全漏洞的曝光以及最佳实践的变化。通过跟踪最新行业动向，确保团队对技术变化有敏感性。。建立严格的安全标准，并持续评估系统中使用的开源软件，确保其符合最新的安全标准。通过定期的安全审查，及时发现并处理潜在的安全风险。

b) 制定技术升级计划：根据评估结果，制定系统中开源软件的技术升级计划，确保软件在新的技术环境中仍然安全可靠。根据评估结果确定开源软件的升级优先级和时间表。确保升级过程中不影响系统的正常运行，并充分测试新版本的兼容性。制定开源软件的技术生命周期管理策略，提前规划软件更新和替换。通过提前预测软件版本的停止支持日期，避免因为使用过时版本而面临的安全风险。

（七） 安全计划执行监控：

a) 建立监控机制：建立监控系统，监测安全计划的执行，包括安全补丁的应用、系统变更的合规性，以及漏洞修复的实际效果。。引入自动审计工具，监控系统变更的合规性，及时发现并处理违规操作。制定紧急响应计划，对于监测到的异常情况能够迅速采取措施进行应对。

b) 问题追踪与修订：在监控过程中发现的问题及时纳入修订计划，确保安全计划的不断完善和适应变化。对于紧急问题，建立快速响应机制，确保及时制定修订计划。

8 法律和合规性考虑

（一）开源软件在业务关键核心功能中的重要程度

1. 业务依赖性：考虑开源软件在业务运作中的关键程度，以确定其对整体业务的依赖程度。
2. 核心功能涉及的开源组件：评估核心业务功能中所使用的开源组件，重点关注对业务稳定性和可用性有重要影响的组件。
3. 对业务流程的贡献：分析开源软件对业务流程的贡献，确保其能够顺利集成并提供必要的支持。

（二）开源软件安全风险可能带来的危害程度

1. 敏感数据和隐私保护：考虑开源软件在处理敏感数据时的安全性，以及对用户隐私的保护程度。
2. 业务中断和数据泄露风险：评估开源软件的漏洞可能导致的业务中断和数据泄露风险，特别关注对业务连续性的潜在影响。
3. 合规性风险：分析开源软件在满足法规和合规性方面的能力，强调其在保护用户数据和遵守隐私法规方面的合规性。

（三）对法规和合规性的影响

1. 数据隐私法规要求：确保开源软件符合相关数据隐私法规，如 GDPR 等，以防范法律责任。
2. 知识产权合规性：评估开源软件的知识产权合规性，确保使用的开源组件不侵犯任何知识产权。
3. 安全标准遵循：强调开源软件在安全标准（如 OWASP Top 10）方面的符合性，以确保应对当前安全威胁。

（四）法规要求的指导和建议

1. 隐私保护策略：制定和实施严格的隐私保护策略，包括用户数据的加密、访问控制和数据处理透明度。
2. 定期合规性审查：定期进行开源软件合规性审查，确保其满足最新的法规和合规性要求。
3. 法务咨询与合同管理：与法务团队合作，确保使用开源软件的合同明确规定了法规和合规性方面的责任和义务。
4. 持续监测与响应：建立持续监测机制，及时应对新出现的法规和合规性要求，确保开源软件的持续合规性。
5. 教育培训：对开发人员和相关团队进行法规和合规性培训，提高对合规性要求的认识。

附录 A

（规范性附录）

开源软件安全风险评价指标体系权重评分表

表 A.1 给出了开源软件安全风险评价指标体系权重评分表示例。

表 A.1 开源软件安全风险评价指标体系权重评分表示例

一级指标	二级指标	权重 x 50%	权重 x 100%	权重（分数）	评价标准
安全漏洞分析	漏洞信息收集	0.1	0.2	20	信息收集及时、全面，有助于风险预防。
	威胁分析	0.15	0.3	30	对攻击方式和危害程度的准确评估。
	影响范围分析	0.1	0.2	20	对漏洞潜在影响的评估是否全面。
	漏洞严重性判定	0.15	0.3	30	建立漏洞等级体系，判定严重性。
代码合规性检测	许可证检查	0.1	0.2	20	对许可证合规性的全面检查。
	许可证符合政策和法规	0.1	0.2	20	许可证符合内部政策和法规。
	代码审查和合规性报告	0.15	0.3	30	通过代码审查工具检测违规部分。
	形成详细的合规性报告	0.15	0.3	30	提供详细的违规部分信息。
开源许可证合规性分析	许可证信息获取	0.1	0.2	20	通过数据库和算法获取许可证信息。
	许可证合规性检测	0.1	0.2	20	分析许可证使用与条款级冲突检测。
	准确输出冲突检测结果	0.15	0.3	30	输出精确、清晰的检测结果。
	评估合规性标准	0.15	0.3	30	评估软件是否符合公司或项目标准。
软件可维护性分析	活跃度评估	0.1	0.2	20	评估开源软件项目的活跃度。
	健康状况分析	0.1	0.2	20	分析开源社区的规模和响应速度。
	问题跟踪分析	0.15	0.3	30	了解已解决问题的数量和速度。
	维护团队响应效率	0.15	0.3	30	判断维护团队对问题的响应效率。
PoC 判定	攻击方式分析	0.1	0.2	20	分析攻击者如何利用漏洞进

					行攻击。
	危害程度评估	0.15	0.3	30	评估漏洞的潜在危害程度。
	公开 PoC 检查	0.1	0.2	20	判断 PoC 是否公开，是否存在防护措施。
APT 攻击跟踪	情报收集	0.1	0.2	20	收集关于高级持续威胁的情报。
	软件关联 APT 攻击	0.1	0.2	20	分析软件是否与已知的 APT 攻击有关。
	软件易受威胁分析	0.15	0.3	30	了解软件是否容易受到特定威胁。
	APT 攻击跟踪报告	0.15	0.3	30	提供防范建议的 APT 攻击跟踪报告。

附录 B

（规范性附录）

开源软件安全风险评价相关指标的计算方法

B.1 安全漏洞统计

开源软件中，每个漏洞都有不同的危害程度，可以根据漏洞的影响因素给予相应的分数，然后对所有漏洞的分数进行加权求和计算软件中漏洞的危害程度总分。

$$危害程度总分 = \sum_{i=1}^{n} (V_i * W_i)$$

其中，V_i表示软件的第 i 个漏洞危险等级，W_i表示第 i 个漏洞的危害程度权重。

B.2可维护性分析计算

可维护性分析旨在评估开源软件项目的维护状况，包括最近的更新频率、提交者活动等。

1、活跃度得分

活跃度得分反映了项目的最近更新频率和提交者活动。活跃度得分计算公式如下所示：

$$AS = \frac{RUC}{TPD} + CA$$

其中 RUC 表示最近一个固定时间段内的项目更新次数；TPD 表示项目的总存在时间（以月为单位）；CA 是贡献者的活跃度，通过计算最近一段时间内提交的贡献者数量来衡量。

2、维护团队响应效率得分

维护团队响应效率得分反映了维护团队对问题的响应速度。具体的计算方式

如下所示：

$$MA = \frac{IRT}{TIR} * RTE$$

其中 IRT 表示在规定时间内得到响应的问题数量；TIR 是总问题数量；RTE 是响应时间的效率，通过计算平均响应时间来衡量。

B.3开源软件安全风险计算

在开源软件安全风险评估中，设安全风险得分为S_r，则基于相关六个维度定义一个综合的开源软件安全风险计算数学公式：

$$S_r = \sum_{i=1}^{n} (W_i * \frac{P_{i,impact} * P_{i,likelihood}}{M_i})$$

其中：

N 是六个维度的总数；W_i是第 i 个维度的权重；$P_{i,impact}$是第 i 个维度影响评估；$P_{i,likelihood}$是第 i 个维度的有问题发生的可能性；M_i是第 i 个维度最大可能值。

附录 C

（规范性附录）

相关国家标准及指南

表 C.1 相关国家标准

序号	标准号	标准名称
1	GB/T 36475—2018	《软件产品分类》
2	GB/T 30279—2020	《信息安全技术 网络安全漏洞分类分级指南》
3	GB/T 39204-2022	《信息安全技术 关键信息基础设施安全保护要求》
4	GB/T 25069-2022	《信息安全技术 术语》
5	GB/T 36637-2018	《ICT 供应链安全风险管理指南》
6	T/CFAS 0001-2019	《开源软件安全使用规范》

附录 D

（资料性附录）

开源软件安全风险评估流程示例

下面以金融行业某一开源软件为例，说明开源软件安全风险评估流程。

步骤一：读取开源软件的监测数据和预测数据，并将数据存储到安全风险评价指标数据库中。

监测数据包括但不限于：

a. 系统性能数据：这包括 CPU 使用率、内存占用、磁盘 I/O、网络带宽等系统性能指标，这些数据可以用于监控系统的运行状态和性能瓶颈。

b. 用户行为数据：如果用户与开源软件有交互，那么用户的操作、点击、浏览等行为数据也可以被收集和分析，以了解用户的使用习惯和偏好。

c. 日志数据：软件运行过程中产生的日志数据包含了丰富的信息，如错误日志、访问日志、操作日志等，可以用于诊断和解决问题。

d. 安全数据：包括安全事件、漏洞信息、入侵检测等安全相关的数据，这些数据可以帮助发现潜在的安全威胁和漏洞。

预测数据包括但不限于：

a. 趋势预测数据：基于历史数据和当前数据，利用机器学习、数据挖掘等技术对未来的趋势进行预测，如用户增长趋势、系统负载趋势等。

b. 风险评估数据：通过对系统性能、用户行为、安全数据等进行分析，评估系统的风险水平，如预测可能出现的故障、安全漏洞等。

c. 需求预测数据：如果开源软件用于产品或服务的开发，那么预测用户的需求和期望也是非常重要的。通过收集和分析用户反馈、市场数据等信息，可以预测用户对产品的需求和期望，从而指导产品的设计和开发。

注：开源软件的监测数据和预测数据可能因具体的应用场景和用途而有所不同。此外，由于开源软件通常具有高度的灵活性和可定制性，因此用户可以根据自己的需求和偏好来定义和收集特定的监测和预测数据。

步骤二：选择安全风险评估指标

对于金融软件，选择安全漏洞分析、代码合规性检测、开源许可证合规性分析、软件可维护性分析作为安全风险评估指标。评估指标的评估要求默认全选。金融软件的评估指标如表 D.1 所示。

表 D.1　金融软件的评估指标

评估指标	评估要求
安全漏洞分析	a) 收集大型开放安全漏洞库、开源社区安全公告、威胁情报等开源软件的漏洞信息； b) 对收集到的漏洞进行威胁分析，包括攻击方式、危害程度等； c) 进行影响范围分析，评估漏洞对系统、数据的潜在影响； d) 判定漏洞的严重性，建立漏洞等级体系。
代码合规性检测	a) 使用基于图匹配的代码标准合规检测方法对开源软件的许可证进行检查； b) 确保许可证符合组织内部政策和法规； c) 使用代码审查工具检测代码中是否存在违反合规性规定的部分； d) 形成代码合规性报告，包含违规部分的详细信息。
开源许可证合规性分析	a) 使用大规模数据库和文本分类算法获取不同许可证的条款信息； b) 构建精确到条款的开源项目许可证冲突列表； c) 分析开源软件所使用的许可证，输出准确、清晰的许可证使用与条款级冲突检测结果； d) 评估开源软件是否符合公司或项目的许可证合规性

	标准。
软件可维护性分析	a) 使用大规模数据库和文本分类算法获取不同许可证的条款信息； b) 构建精确到条款的开源项目许可证冲突列表； c) 分析开源软件所使用的许可证，输出准确、清晰的许可证使用与条款级冲突检测结果； d) 评估开源软件是否符合公司或项目的许可证合规性标准。

步骤三：设置评估指标的权重

金融行业开源软件测评过程中对评估属性和测评指标进行分级定义和分级运算，每一级有相应的权重体系，自下而上计算得出开源软件的量化评分值。实际测评时，将可量化的指标进行量化，不可量化的定性指标则采用专家评估法进行确定。另外，针对某些特殊指标可以设置“一票否决”权，如开源许可证中存在不满足应用需求的限制条件等。本文件测评模型中的评估属性和测评指标可结合具体开源软件特点进行合理增删。

一级评估属性的权重反映了金融机构对开源软件的关注重点，金融机构重点关注的评估属性具有更大权重值。通过专家评分的方式，调研开源软件各项一级评估属性的相对重要性，并用求平均和归一化的方法确定各一级评估属性的权重。

设置评估指标的权重为安全漏洞分析 25%、代码合规性检测 25%、软件可维护性分析 25%、开源许可证合规性分析 25%。

步骤四：执行安全风险评估，计算得到评估得分。

步骤 4.1：计算评估指标的指标通过率

对于评估指标的评估要求，逐一评估是否通过要求，计算得到每个评估指标的指标通过率。指标通过率计算公式见公式（1）。

$$\text{指标通过率} = \frac{\text{通过测评的指标数}}{\text{总测评指标数}} \times 100\% \qquad (1)$$

对于金融软件，得到评估指标的指标通过率，见表 D.2。

表 D.2　金融软件的评估结果

评估指标	评估要求	评估结果	指标通过率
安全漏洞分析	漏洞信息收集	通过	100%
	威胁分析	通过	
	影响范围分析	通过	
	漏洞严重性判定	通过	
代码合规性检测	许可证检查	通过	50%
	许可证符合政策和法规	通过	
	代码审查和合规性报告	不通过	
	形成详细的合规性报告	不通过	
开源许可证合规性分析	许可证信息获取	通过	100%
	许可证合规性检测	通过	
	准确输出冲突检测结果	通过	
	评估合规性标准	通过	
软件可维护性分析	活跃度评估	通过	75%
	健康状况分析	通过	
	问题跟踪分析	通过	
	维护团队响应效率	不通过	

步骤 4.2：计算测评结果。

金融软件的测评结果由第一阶段计算得到的指标通过率得分乘以相应的权重后相加得出，即

$$V = \sum_{j=1}^{m} (T_j \times W_j) \qquad (2)$$

式中：

V 表示开源软件的测评结果；

T_j表示第 j 项评估指标的评估值；

W_j表示第 j 项评估指标的权重；

m 表示评估指标的数量。

通过公式（2）计算得到金融软件的测评结果，即：

$$V = 1 \times 0.25 + 0.5 \times 0.25 + 0.75 \times 0.25 + 1 \times 0.25 = 0.8125$$

步骤五：开源软件安全风险等级评价。

针对开源软件进行评价时，建议划分为 A、B、C、D 四个评价等级，将开源软件的测评结果进行归一化处理后的通用划分方法见表 3，具体评分区间可根据实际需求灵活调整。金融机构应根据实际需求定期展开等级评价，以动态评估开源软件服务能力，等级评价方法见表 D.3。

表 D.3　金融软件的等级评价方法

等级	评分区间
A	0.9≤V≤1
B	0.8≤V＜0.9
C	0.6≤V＜0.8
D	0≤V＜0.6

将金融软件的计算得到对应到表 D.3 中，可得该金融软件的安全风险等级属于 B 级。

参考文献

[1] 《软件产品分类》（GB/T 36475—2018）

[2] 《信息安全技术 网络安全漏洞分类分级指南》（GB/T 30279—2020）

[3] 悬镜科技. 软件供应链安全治理与运营白皮书.2022 年

[4] 国家标准. 信息安全技术 软件供应链安全要求.

[5] GB/T 36637-2018. 信息安全技术 ICT 供应链安全风险管理指南.

[6] Synopsys. 开源安全和风险分析报告.2023 年

[7] OWASP Foundation. OWASP API 安全 TOP10, 2019

[8] Salt Labs. State of API Security Q1 2023. 2023

[9] 中国信息通信研究院. 软件物料清单实践指南. 2023 年.

[10] 中国联通研究院. CU-DevSecOps 实践白皮书.

[11] 《信息安全技术 软件产品开源代码安全评价方法（征求意见稿）》

[12] 《软件供应链安全治理实践指南白皮书（2023）》

[13] 《加快开源软件发展三年行动计划（2023-2025 年）》

[14] 《“十四五”软件和信息技术服务业发展规划》

[15] 《信息安全技术 关键信息基础设施安全保护要求》（GB/T 39204-2022）

[16] 《关键信息基础设施安全保护条例》

[17] 《网络安全审查办法》

[18] 《信息安全技术 术语》（GB/T 25069-2022）

[19] 《ICT 供应链安全风险管理指南》（GB/T 36637-2018）

[20] 《开源软件安全使用规范》（T/CFAS 0001-2019）