

开源治理实践与发展趋势

但吉兵

苏州棱镜七彩信息科技有限公司



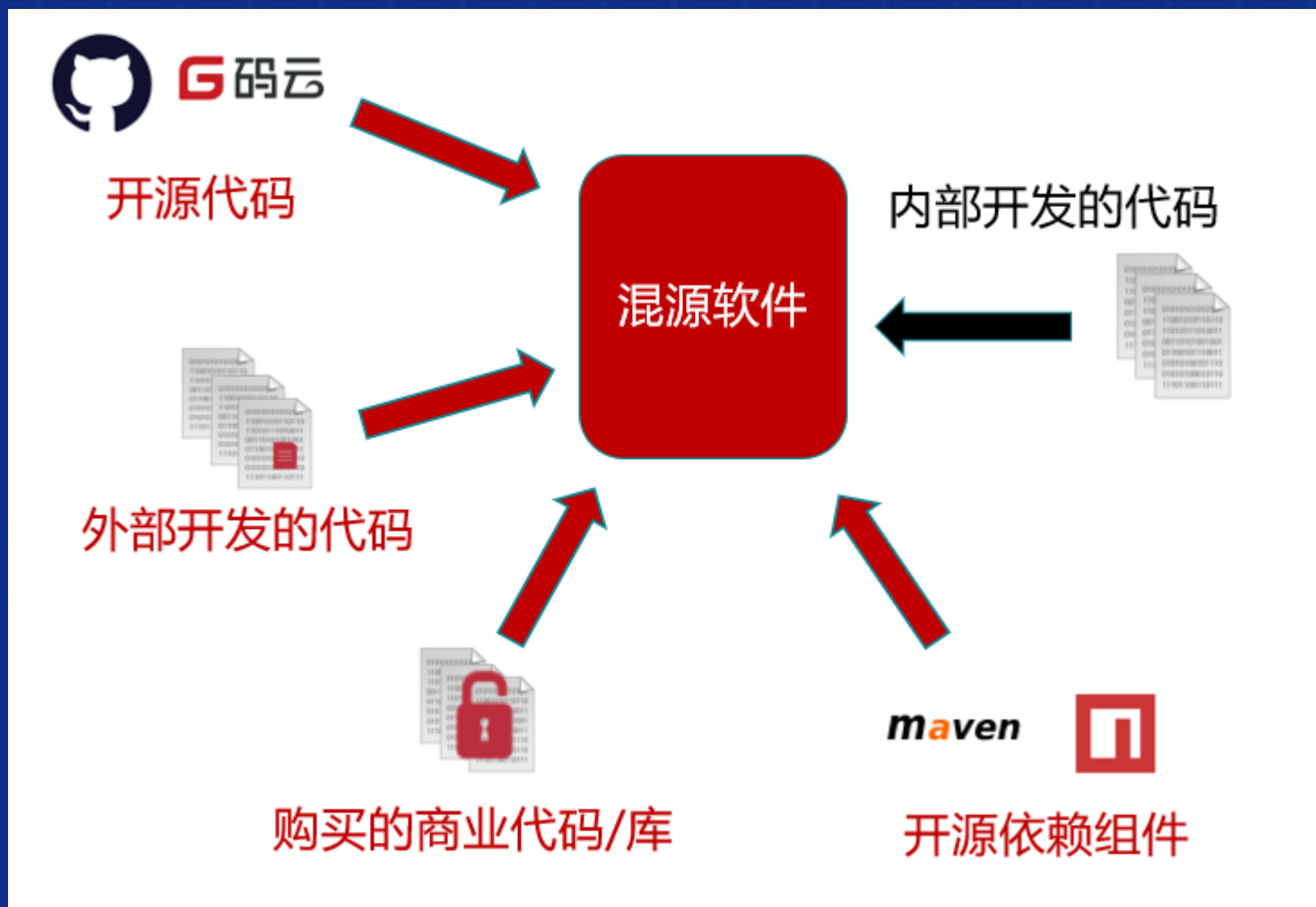


开源治理不可忽视

开源治理为什么不可忽视？

■ 开源治理不可忽视

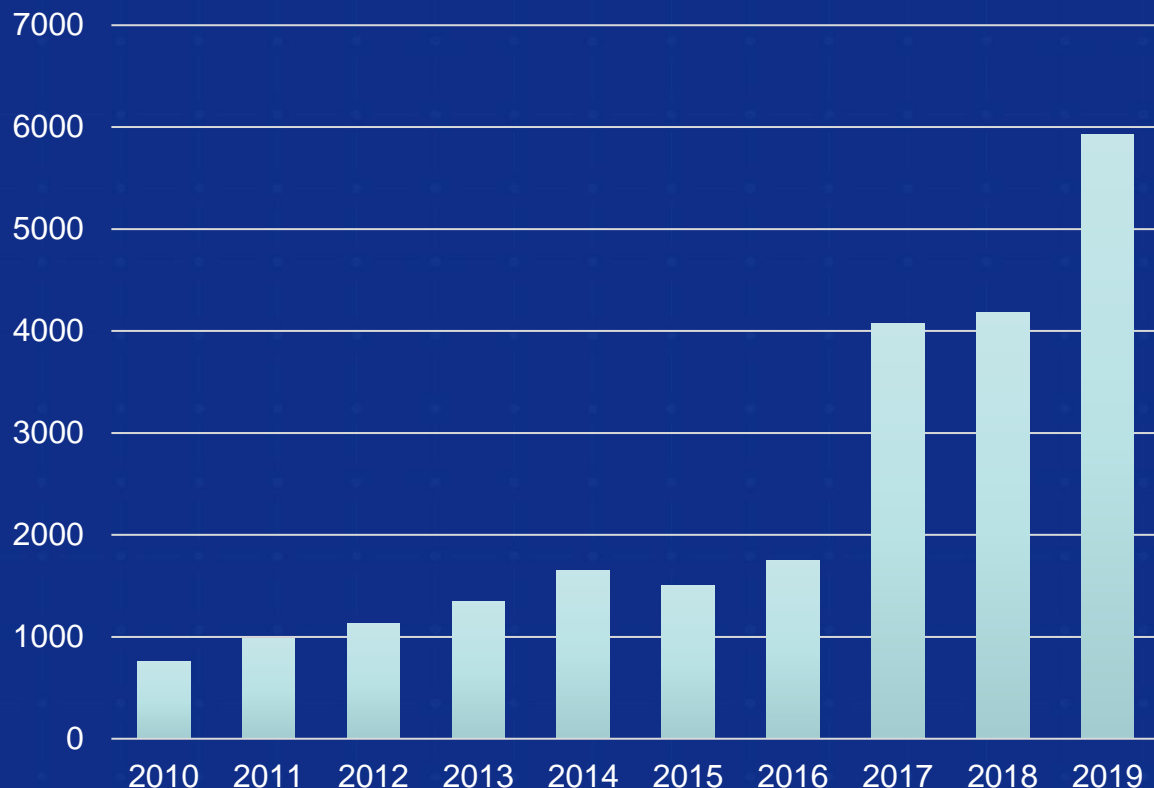
1、用得广泛——混源软件开发已成为主要的开发方式之一





开源治理不可忽视

2、开源漏洞增长迅速



- **增长快速**：2019年同比增长超50%
- **来源广泛且杂乱**：开源软件的漏洞信息散布于各个社区、托管平台等，漏洞信息来源非常复杂
- **官方漏洞库收录有缺失**：部分开源漏洞未被收录于官方漏洞库中
- **漏洞信息难以通知使用者**：开源软件维护者没有直接有效途径通知开源软件使用者最新的漏洞情况



开源治理不可忽视

2、开源漏洞增长迅速

棱镜七彩与CNCERT风险评估联合报告



2019年开源软件风险研究报告

原创 CNCERT软件安全 CNCERT风险评估 5月22日

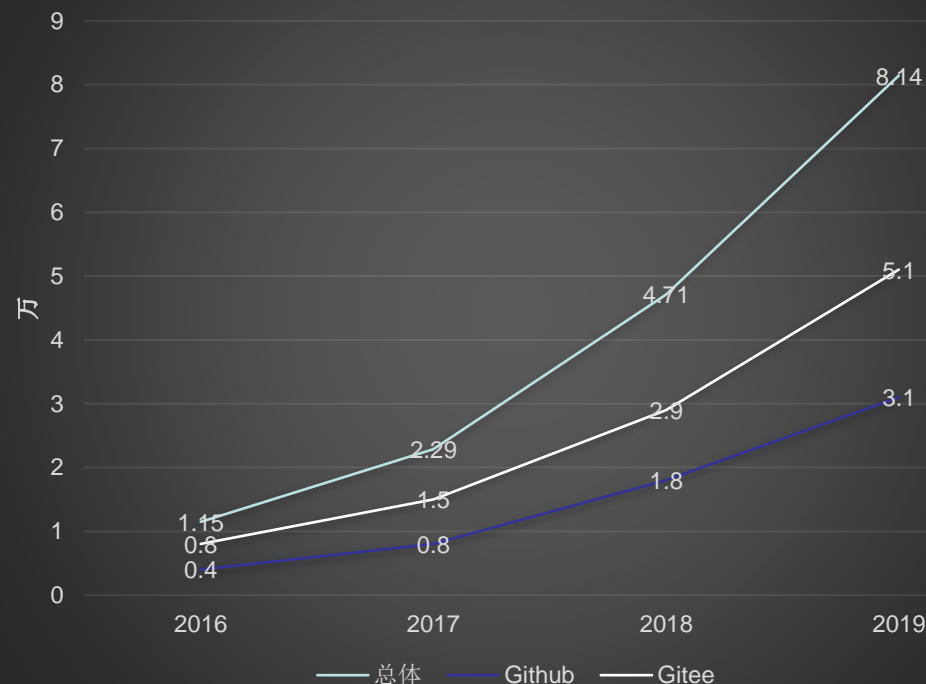


序言

本报告由CNCERT软件安全和苏州棱镜七彩科技有限公司联合发布。

报告一共选取了**48363**个项目进行分析，其中GitHub中选取星数为500星及以上的开源项目**38530**个，Gitee中选取星数为5星及以上的开源项目**9833**个。

组件漏洞数量变化趋势图



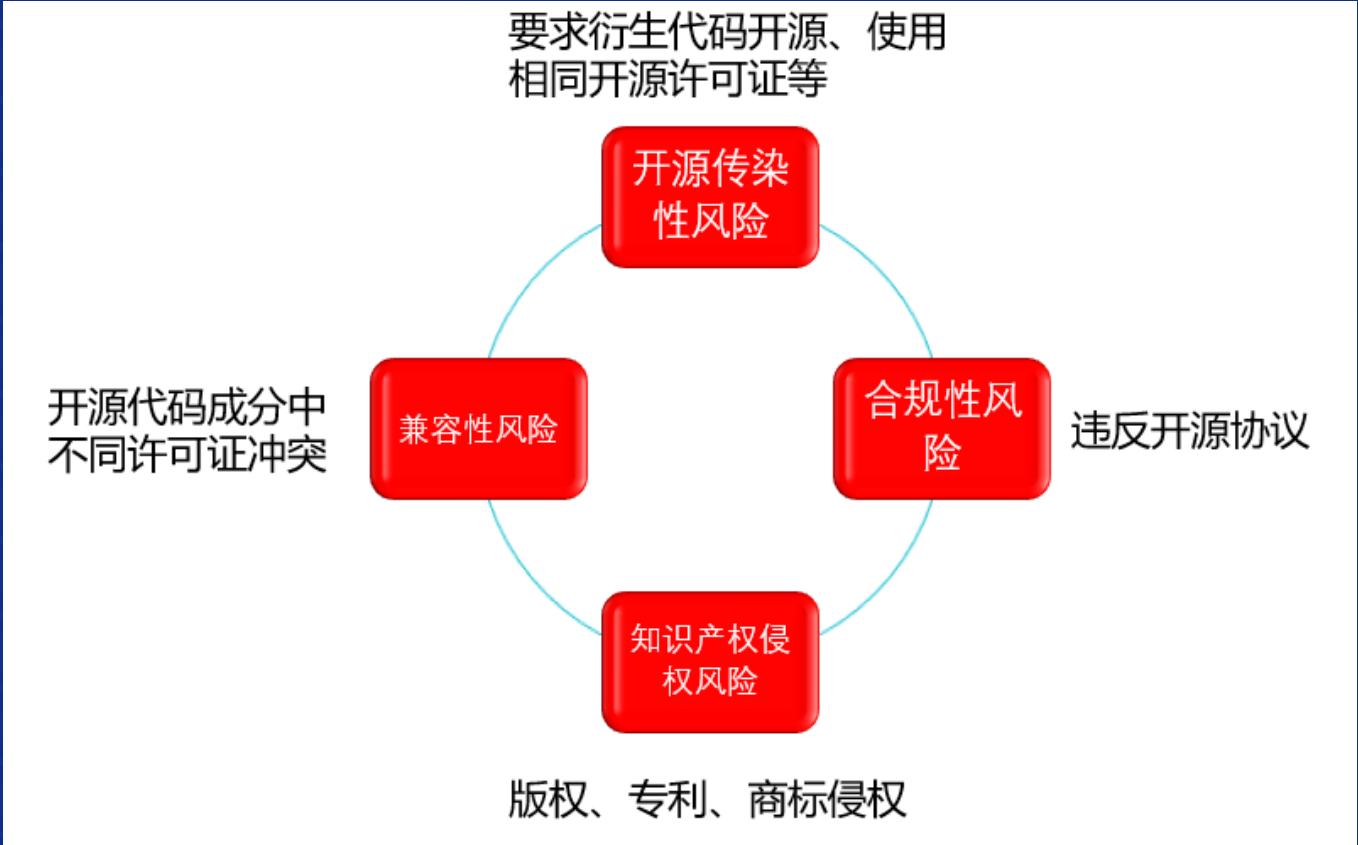
■ ■ ■ 开源治理不可忽视

3、开源协议风险不可忽视

十四五提出完善开源知识产权和法律体系

两公司间关于设计GPL协议的诉讼，该案被称为中国第一个涉及GPL协议的诉讼案件。虽有争议，但一审二审法院一定程度上认可了GPL协议的法律约束力，对于我们对GPL协议的解读和风险认识有重要参考意义。

某公司向Apache基金会对外贡献开源项目时，被发现基于其他开源项目修改而来，被要求整改。





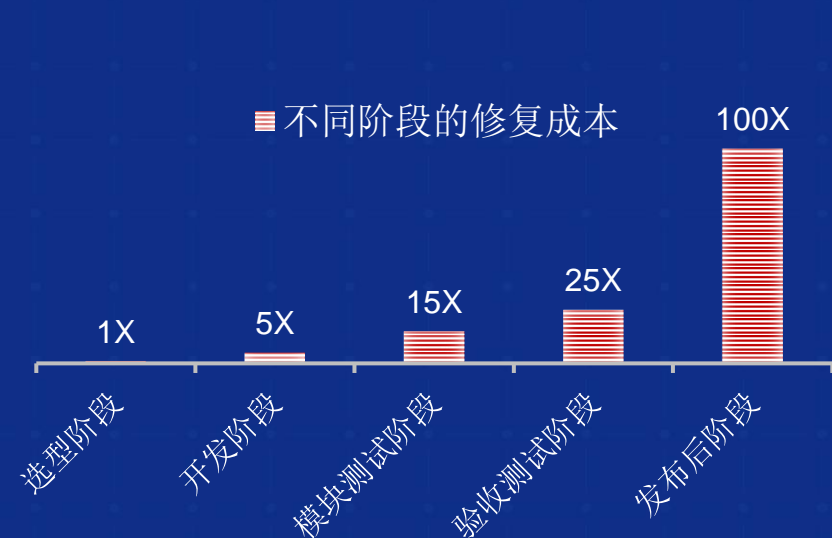
开源治理不可忽视

4、绝大部分使用者没有全面准确掌握自身开源成分

1、发现越晚，修复成本越高

2、无法及时识别受影响的项目和产品

3、管理低效，降低研发效率





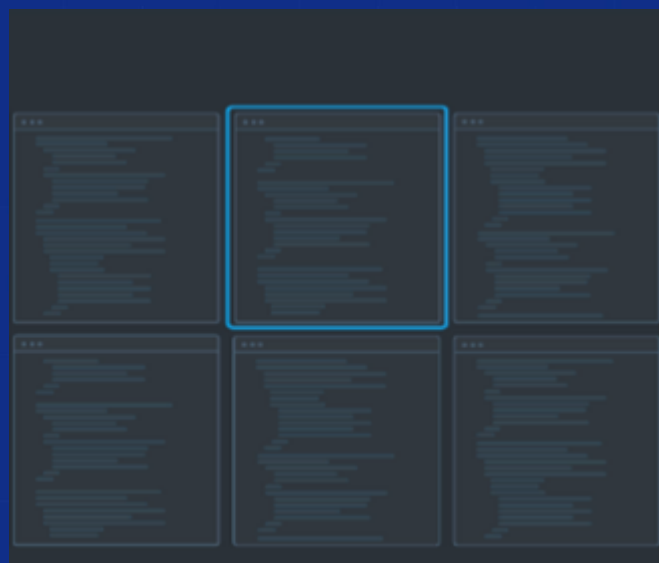
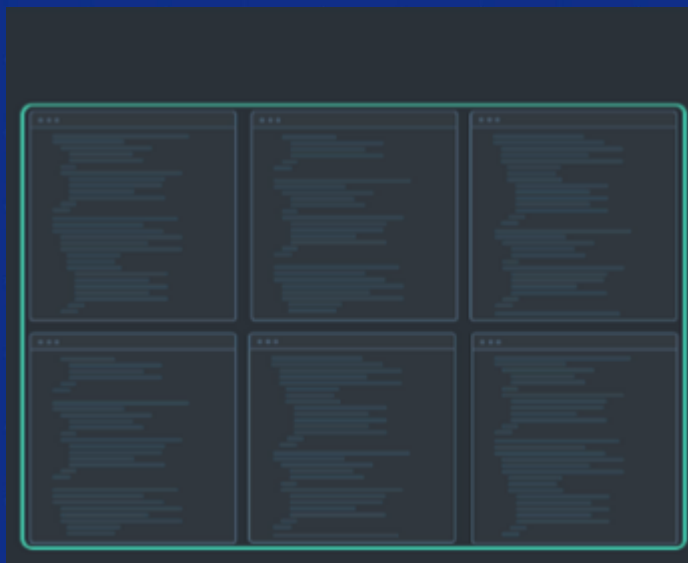
开源治理实践

开源治理如何落地？

开源治理要结合开源软件的使用方式与场景

使用方式：使用开源代码

- 原始引入（引入后不做任何改变）
- 混淆引入（注释、变量名替换等）
- 二次开发引入（增减新代码等）
-



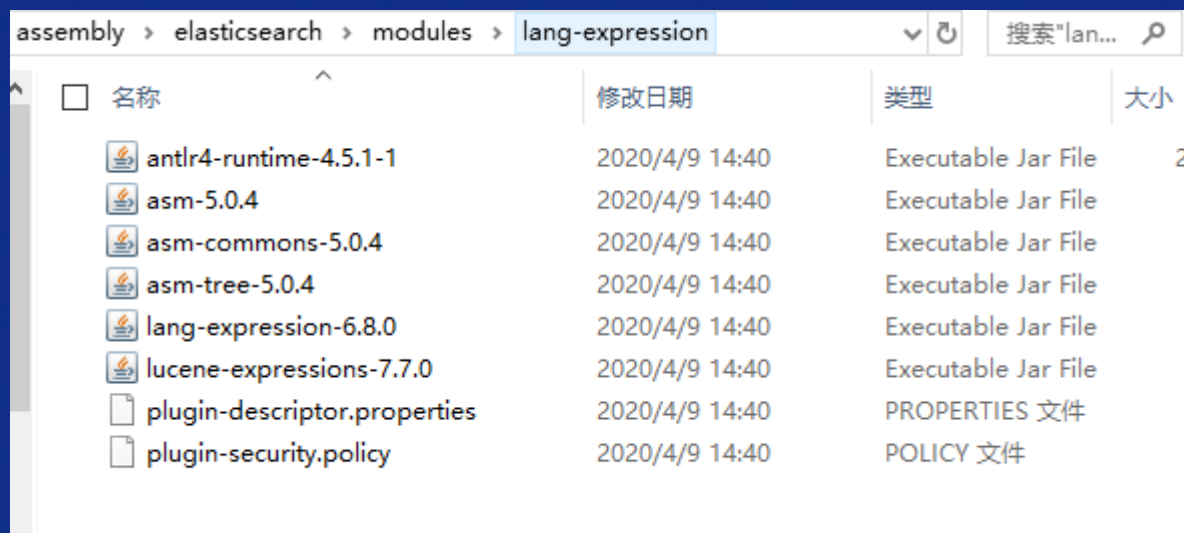


开源治理要结合开源软件的使用方式

使用方式：使用开源依赖组件/制品

- 配置文件引入
- 直接引入制品包

```
- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
</dependency>
- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-aop</artifactId>
</dependency>
```



名称	修改日期	类型	大小
antlr4-runtime-4.5.1-1	2020/4/9 14:40	Executable Jar File	29
asm-5.0.4	2020/4/9 14:40	Executable Jar File	9
asm-commons-5.0.4	2020/4/9 14:40	Executable Jar File	4
asm-tree-5.0.4	2020/4/9 14:40	Executable Jar File	2
lang-expression-6.8.0	2020/4/9 14:40	Executable Jar File	9
lucene-expressions-7.7.0	2020/4/9 14:40	Executable Jar File	7
plugin-descriptor.properties	2020/4/9 14:40	PROPERTIES 文件	
plugin-security.policy	2020/4/9 14:40	POLICY 文件	



开源治理实践

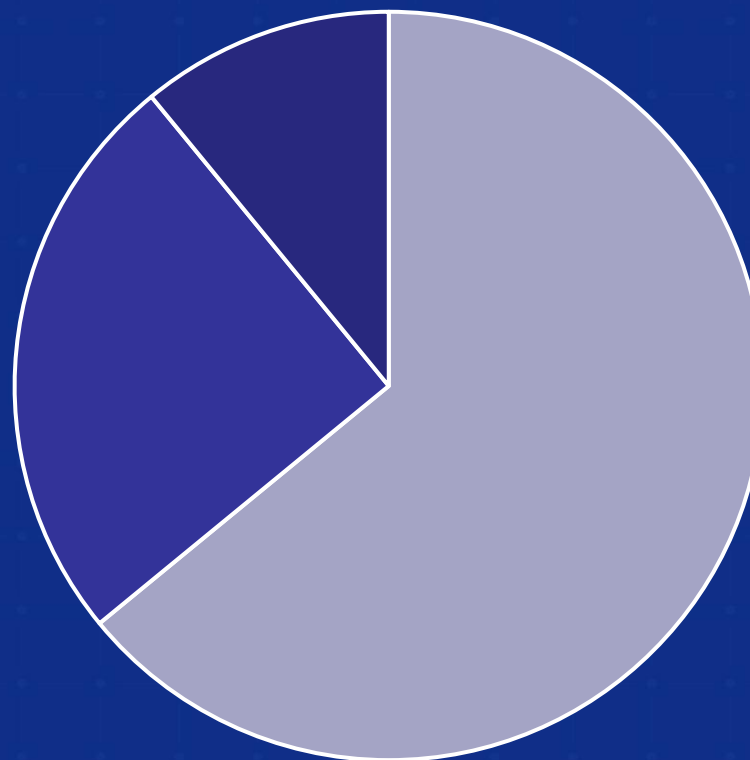
开源治理要结合开源软件的使用场景

使用目的：

- 对外开源
- 内部使用
- 商业使用
- 学习使用

分发方式：

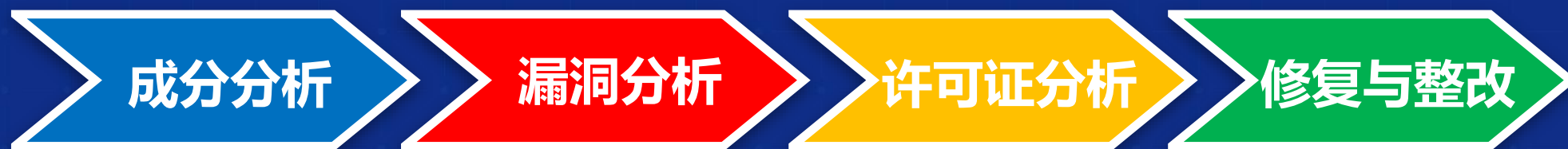
- 软件副本分发
- 云服务



■ 自研代码 ■ 开源代码 ■ 开源组件

开源治理实践

开源治理思路



成分分析是开源软件治理的前提，要全面准确的识别开源成分，才能进行全面的开源治理

基于成分分析结果，对开源成分的安全漏洞进行识别和风险分析，提供修复方案

基于成分分析结果，对开源成分的开源许可证进行识别和风险分析

基于漏洞和许可证的风险分析结果，进行修复和整改

开源治理实践

开源治理思路



棱镜七彩开源治理平台

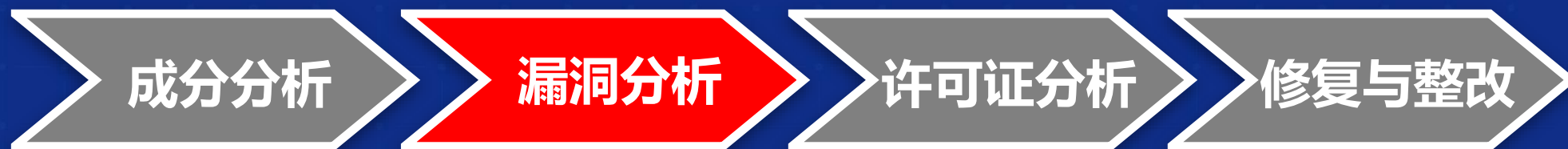
FossCheck
FossEye
FossLicense

- 1、多维度多层次的成分分析能力，全面准确识别出开源成分：
 - 开源代码文件、代码片段引入的识别（完整引入、片段引入、混淆引入）
 - 依赖组件的引入识别（直接引入、间接引入、二进制引入）
- 2、开源成分溯源与确认
 - 溯源来源的开源项目和依赖组件



■ ■ ■ 开源治理实践

开源治理思路



棱镜七彩开源治理平台

FossCheck
FossEye
FossLicense

1、支持多层次的漏洞识别分析：

- 项目级
- 项目版本级
- 依赖组件级
- 代码级
- 函数级



Security
Advisories
Issue Trackers
.....

■ ■ ■ 开源治理实践

开源治理思路



棱镜七彩开源治理平台

FossCheck
FossEye
FossLicense

1、支持多层次的开源许可证识别分析：

- 项目许可证
- 依赖组件许可证
- 文件级许可证

2、兼容性与合规性风险分析

- 不同许可证间的兼容性分析
- 许可证条款符合性风险分析
- 许可证篡改风险分析

3、版权识别与分析

- 开源文件版权识别
- 版权篡改风险分析



■ ■ ■ 开源治理实践

开源治理思路



棱镜七彩开源治理平台

FossCheck

FossEye

FossLicense

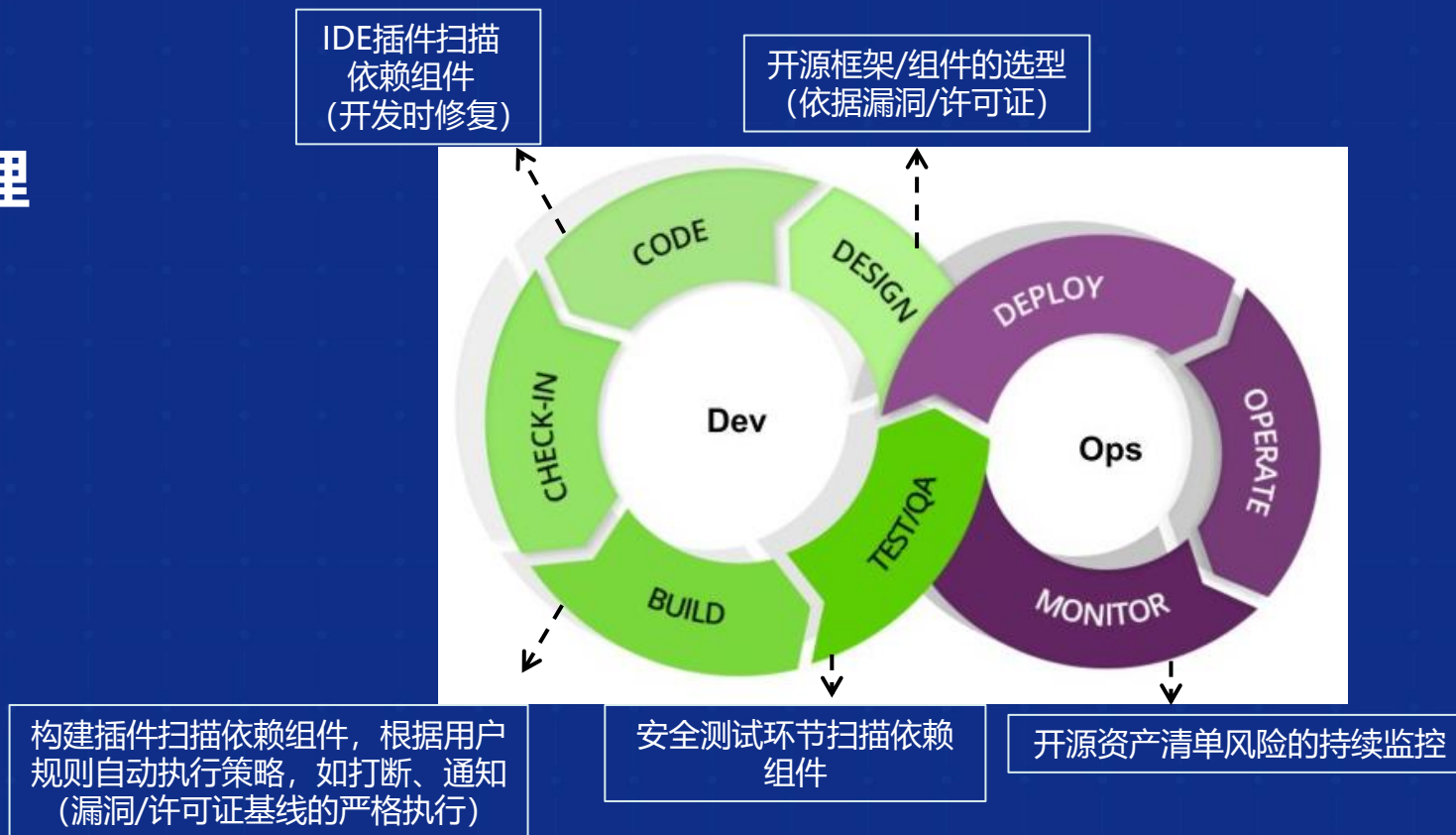
- 1、根据漏洞的修复建议和修复方案进行漏洞修复
 - 推荐与使用版本最接近的安全版本
 - 自动修复依赖组件漏洞版本
 - 使用替换组件修复
 - 从漏洞代码进行修复
- 2、针对存在较大开源许可证风险的成分，进行风险评估与整改
 - 合规风险，整改合规
 - 兼容风险，评估替换
- 3、针对版权篡改风险
 - 恢复原始版权信息
 - 评估替换

开源治理实践

开源治理实践

DevSecOps中的开源治理

- 开源选型
- 流水线集成
- 安全与合规左移
- 自动化执行与管控
- 持续监控
- 资产管理





开源治理发展趋势

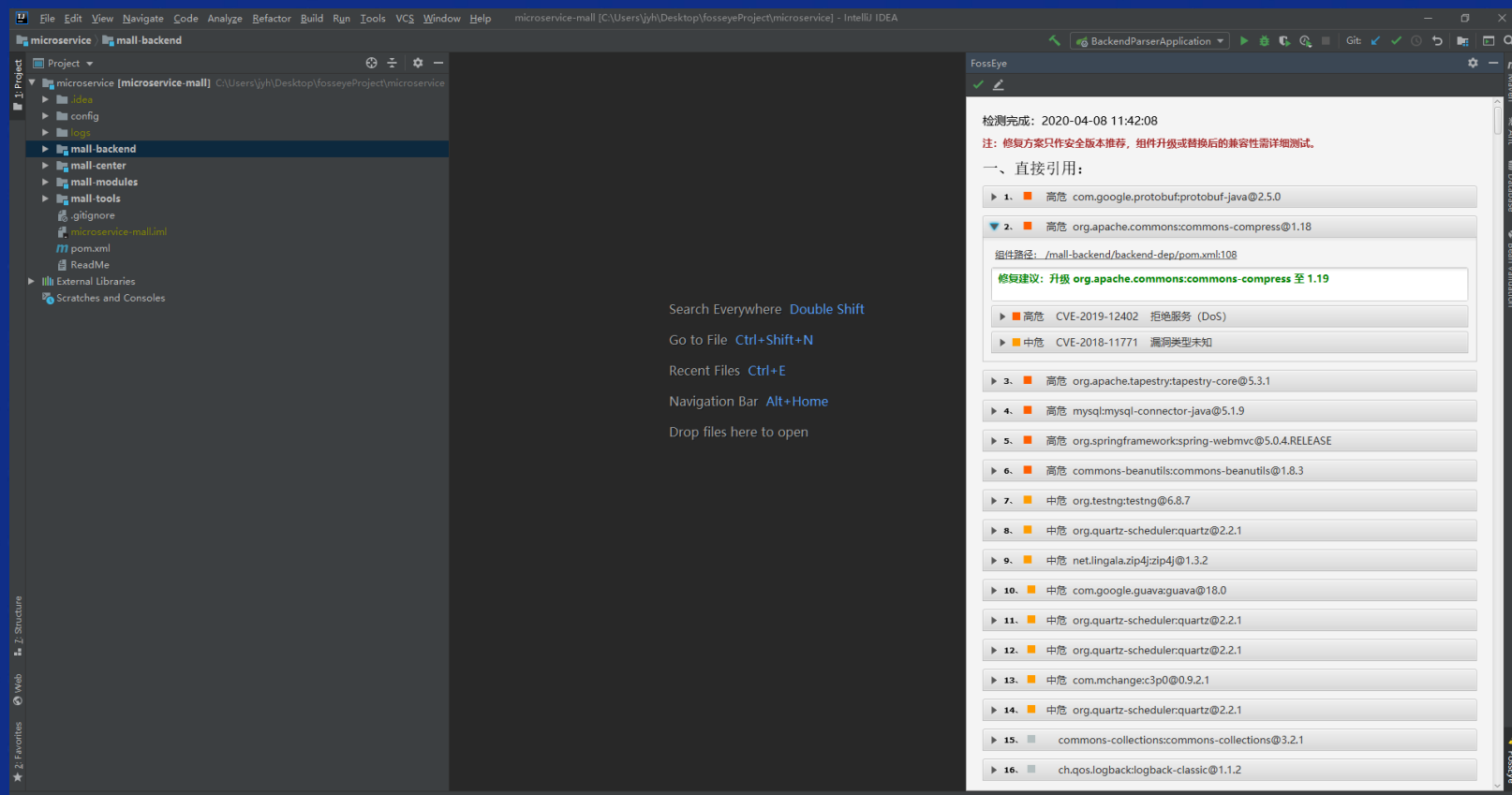
开源治理发展趋势

开源治理实践

开源治理发展趋势

1、从开发者开始进行治疗（左移治理）

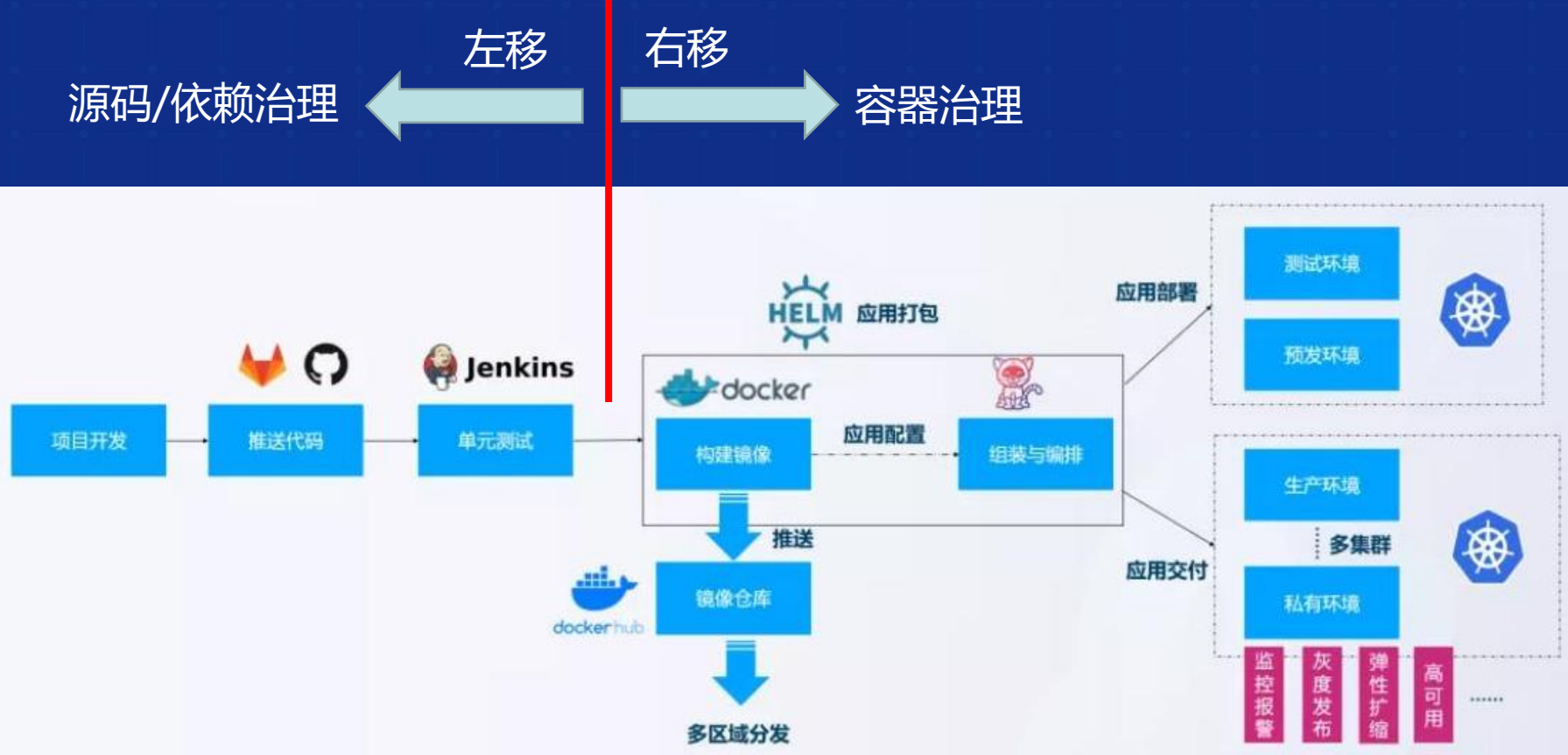
- 快速
- 灵活
- 治理成本低



■ ■ ■ 开源治理实践

开源治理发展趋势

2、云原生的开源治理右移

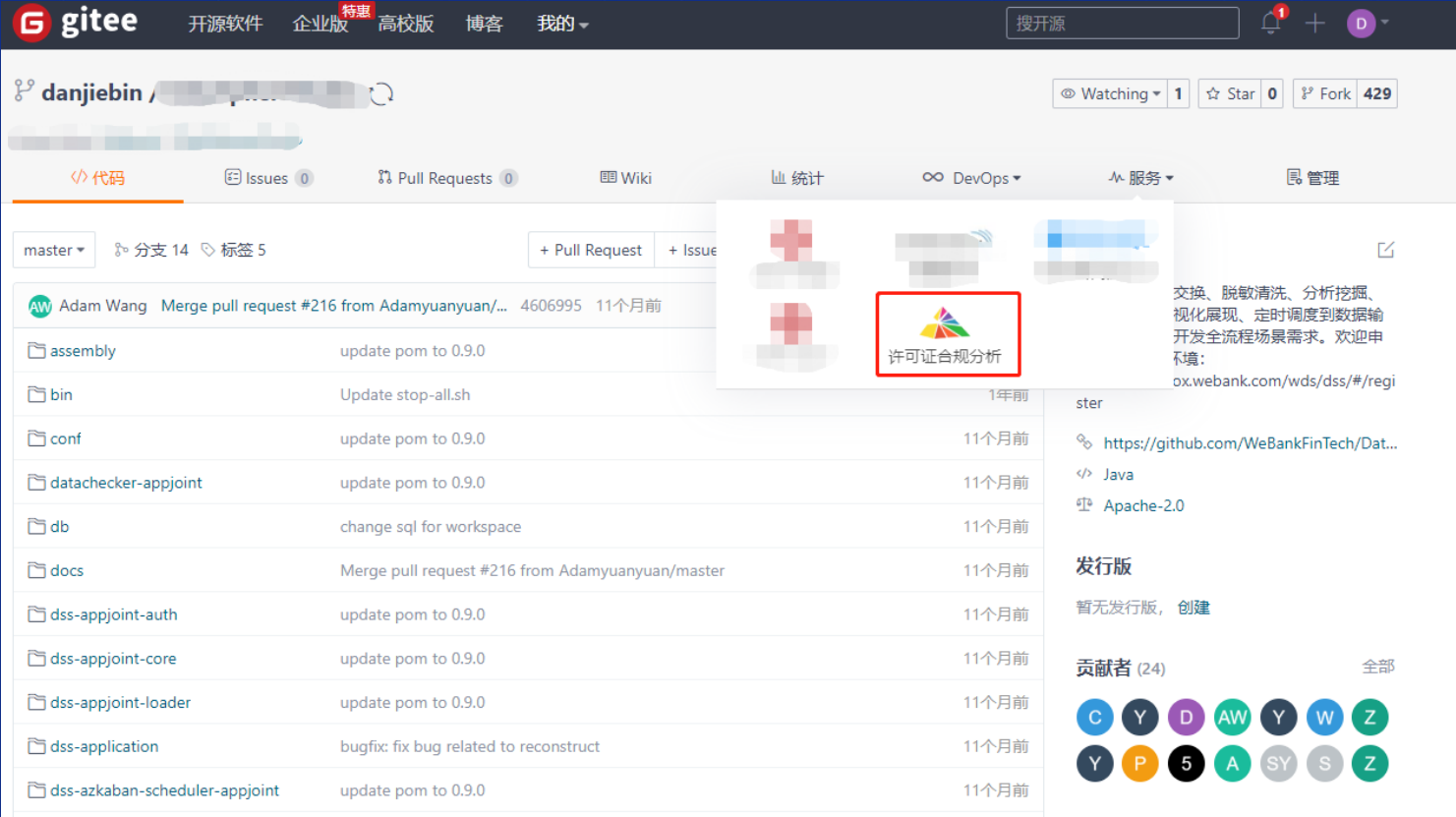


■ ■ ■ 开源治理实践

开源治理发展趋势

3、从开源生态的治理（生态治理左移）

从开源生态的源头治理，改善开源供应链安全与合规态势



THANKS!

