操作系统软件包组件引用情况分析及治理边界讨论

sig-compliance

目录

背景与现状

openEuler社区组成-代码仓

openEuler社区自研开源软件

如isula、stratovirt

openEuler社区基于开源衍生软件

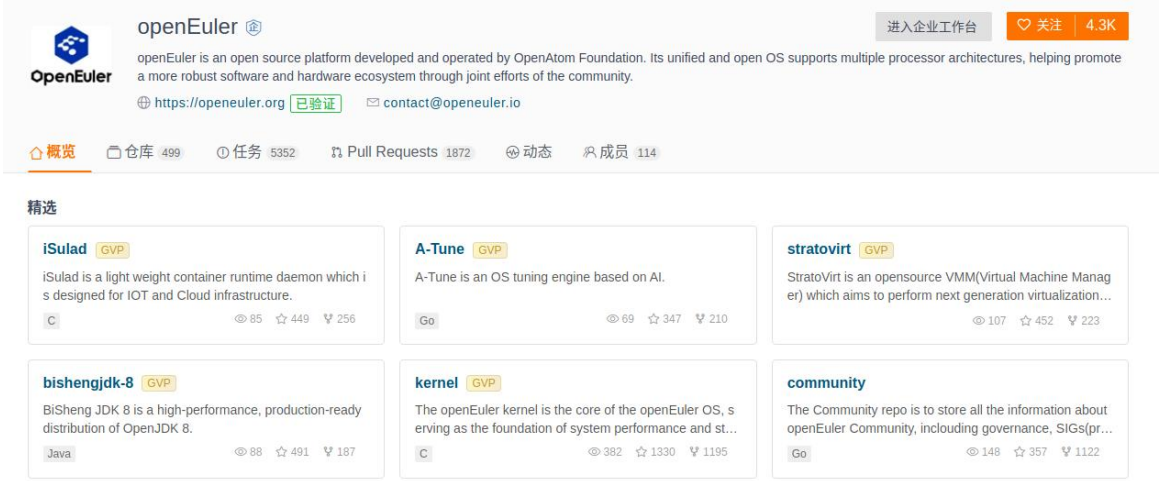
如 bishengjdk

openEuler社区重度维护开源软件

如 kernel、qemu

共计400+

gitee openEuler

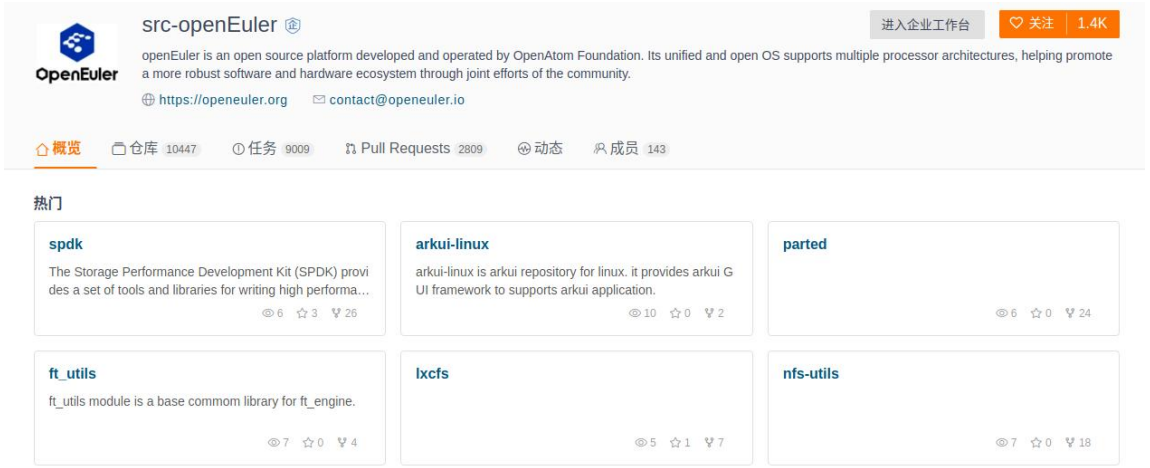


openEuler社区组成-软件包仓

openEuler代码仓为上游的开源软件

其他开源社区或开源项目为上游的开源软件

共计
10000+
gitee src-openEuler



根据选型策略、维护质量等要求，选择其中的部分软件包形成操作系统发行版，俗称“进版本”。

OpenEuler

openEuler软件包成分

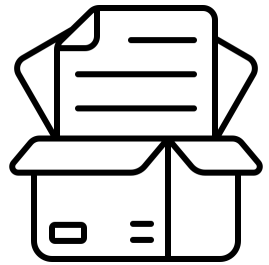
OpenEuler

openEuler自研

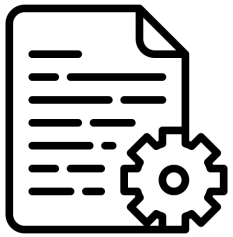



其他上游开源社区或项目

软件源代码



打包文件



配置文件

其他组件

补丁文件

OpenEuler

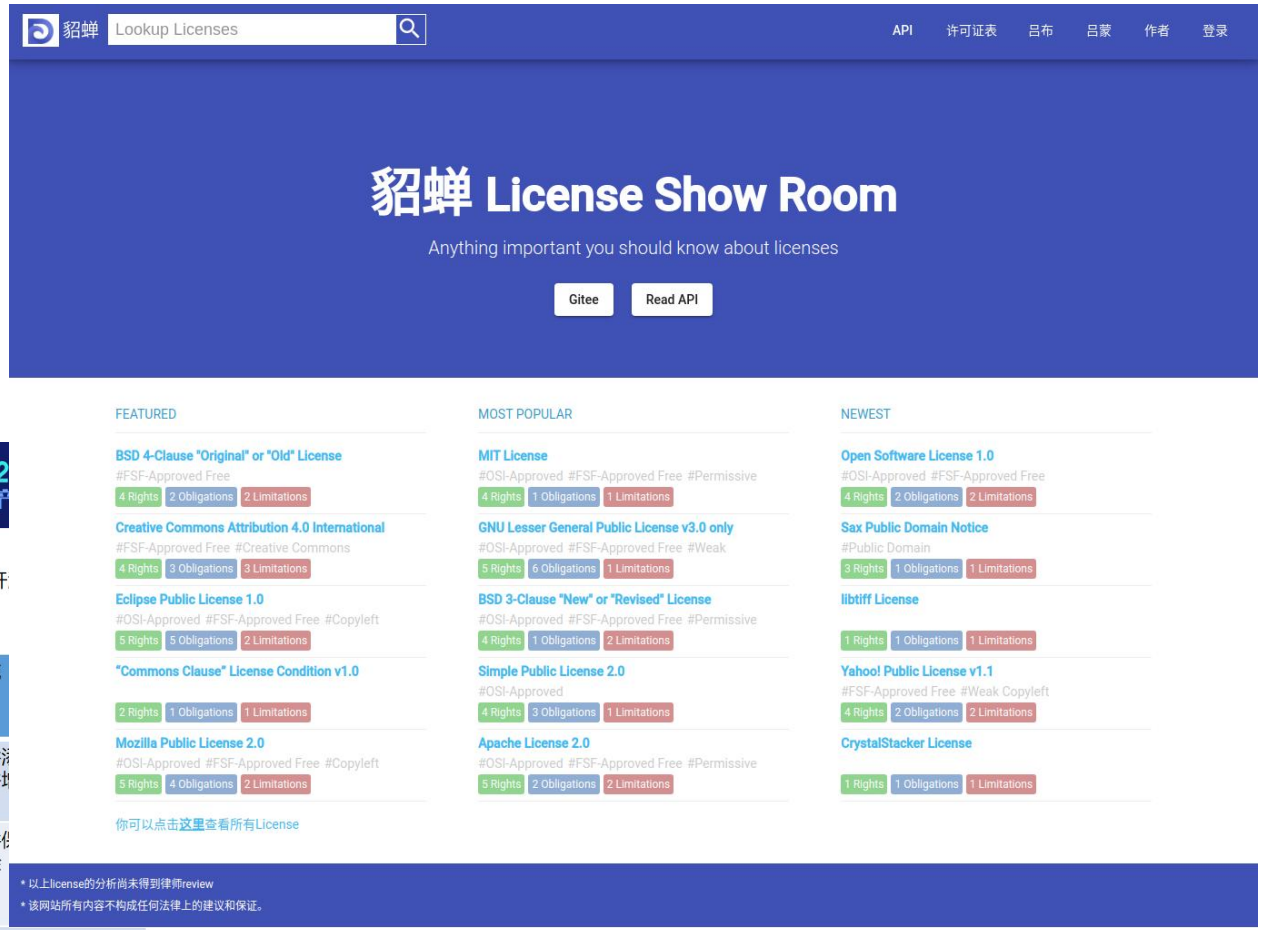
openEuler社区引入

软件包打包

openEuler合规治理成果-基础能力

- License
  - 貂蝉网站提供License基础信息
  - 吕蒙提供文本识别许可证能力
  - 吕布提供开源许可证兼容性判断能力
  - Compliance SIG对社区引入License进行审核判断

- 源代码片段引用治理
  - 对源代码片段引用风险分析较为完善
  - 有相应落地工具实践



源代码片段(Snippet)引用风险分析

2022 开源产

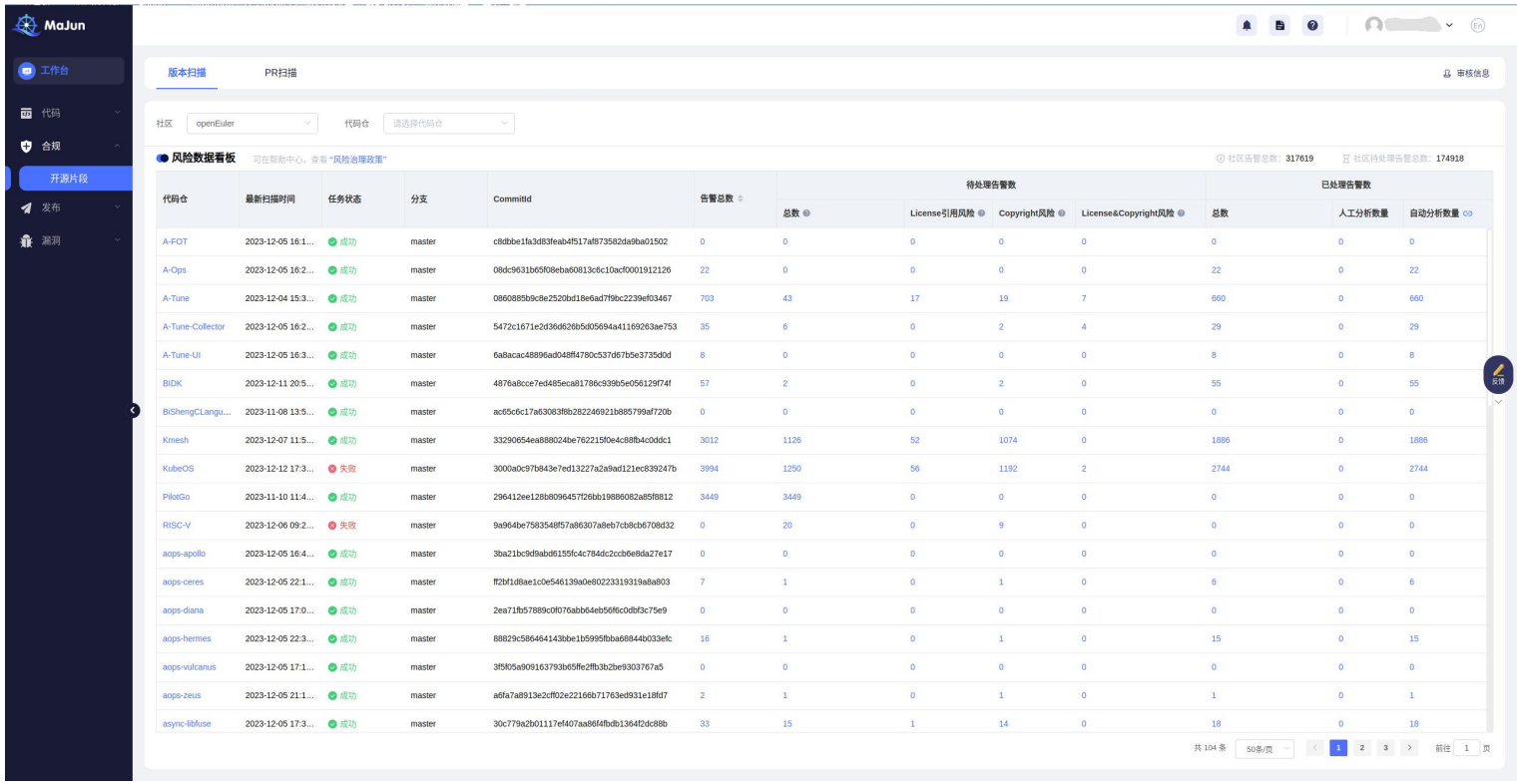
风险分析：**关键点-不要修改被引用片段的License&Copyright声明**

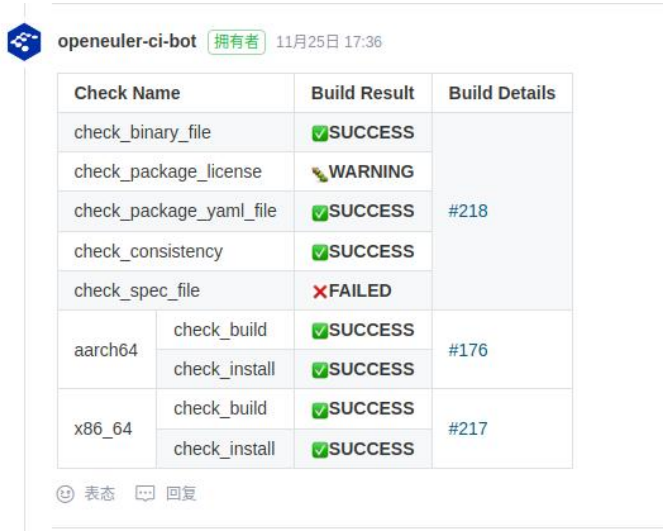
- 我们需要关注4个要素：本地项目的License/Copyright, 本地文件的License/Copyright, 远程项目（被引用的开
  统称远程项目）的License/Copyright, 远程文件的License/Copyright
- 如果文件级的License或Copyright未声明，则它会继承项目级的License/Copyright声明

本地文件L/C声明	远程文件L/C声明	本地项目的声明和远程项目的L/C声明	本地文件“实际的”L/C 声明	远程文件“实际“ L/C 声明	关注的风险	合规措施
空	空	LPL (Local Project License)	LPL, LPC	RPL, RPC	LPL 大概率不等于RPL LPC 一定不等于RPC	本地文件 声明，并
空	有	LPC (Local Project Copyright) RPL (Referenced Project License) RPC (Referenced Project Copyright)	LPL, LPC	RFL, RFC	LPL 大概率不等于RFL LPC 一定不等于RFC	本地文件 不要删除
有	空		LFL，LFC	RPL, RPC	检查：本地文件的L/C声明是否远程项目信息L/C一致。	如有风险：修改本地文件的L/C声明
有	有		LFL, LFC	RFL, RFC	检查： LFL = RFL and LFC = RFC	如有风险：修改本地文件的L/C声明

openEuler合规治理成果-社区治理

- 代码仓
  - 已规范项目许可证、版权声明
  - 已初步具备片段引用治理能力
- 软件包仓
  - 已实现SPEC文件许可证声明与源码声明一致性门禁检查

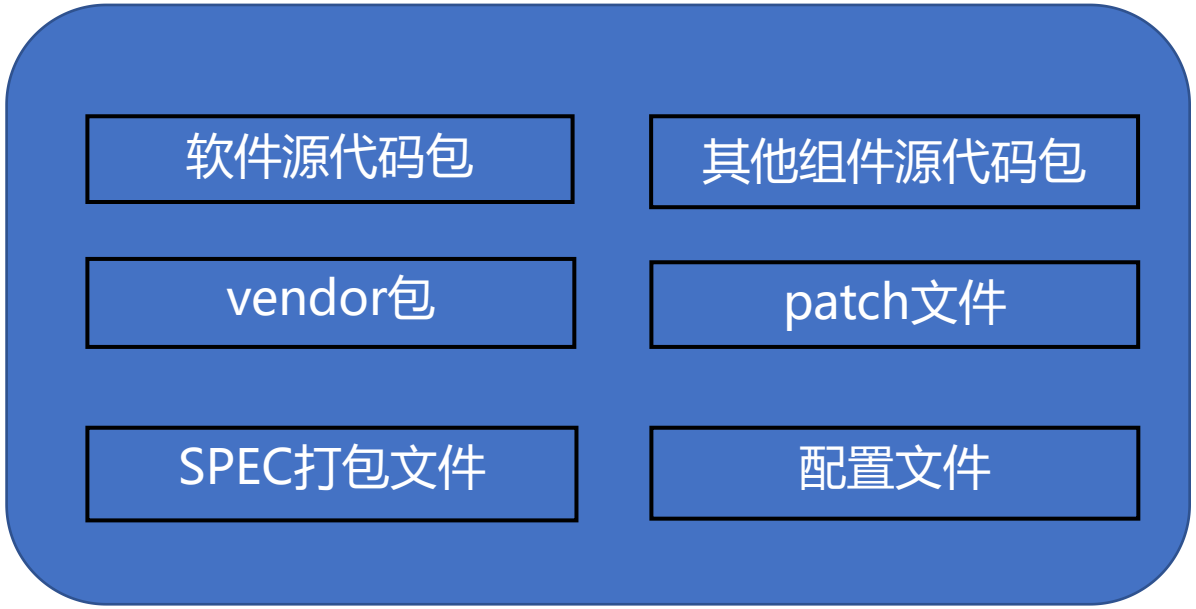




软件包仓仍需进一步更细粒度治理，为此需要梳理软件包组件引用情况，从而确定开源合规治理策略和切入点

软件包组件引用情况分析

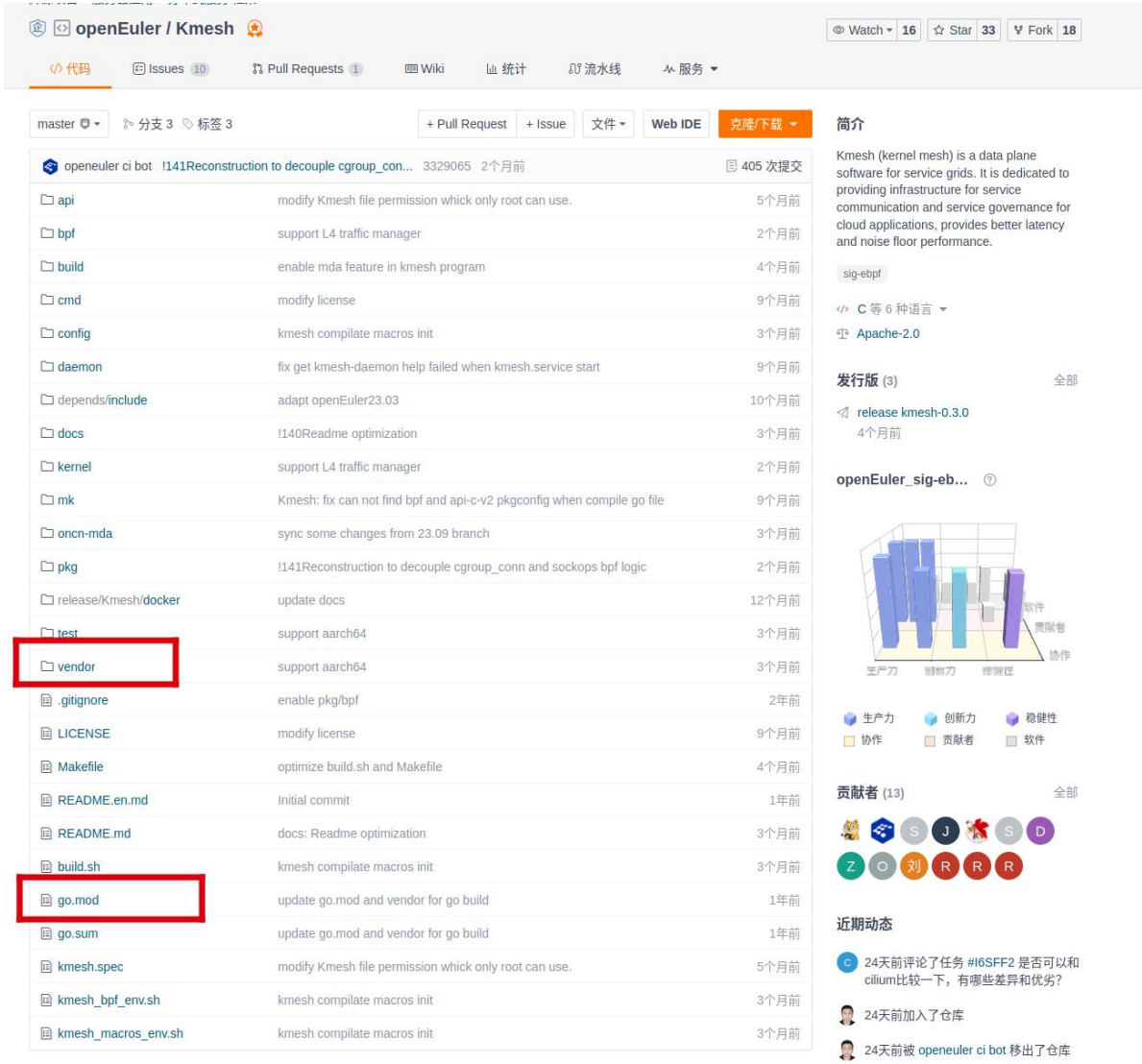
软件包内部成分分析



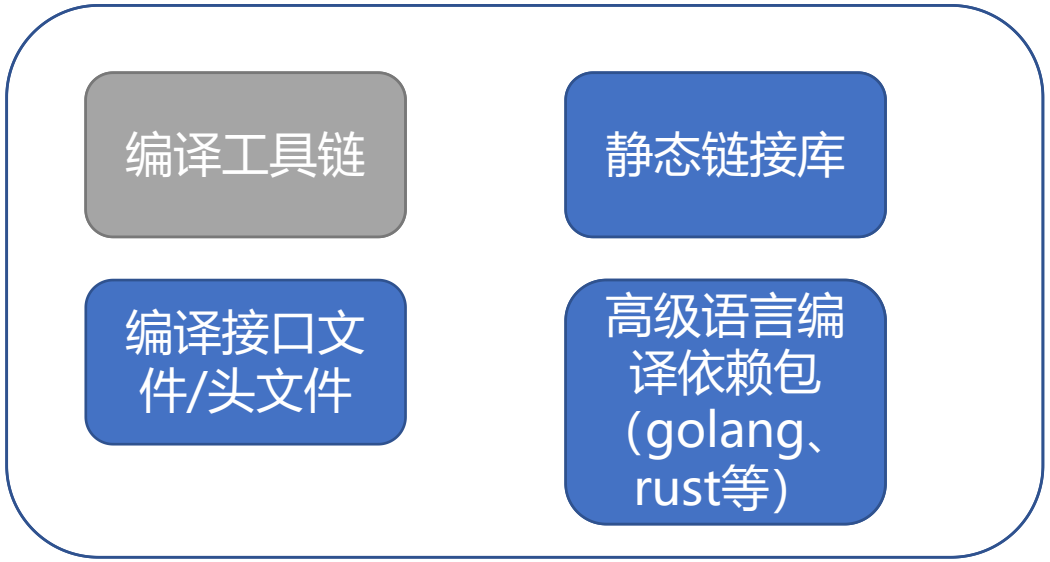
软件包仓每个仓库或操作系统发行版提供的src包解压后，可能会存在上述每类不定项个成分，这些成分满足了功能业务需求的同时，可能会从不同来源、以不同形式引入与目标打包软件版权和开源许可不同的成分，这也为开源合规治理带来了更大的复杂度。

软件源代码包引用情况

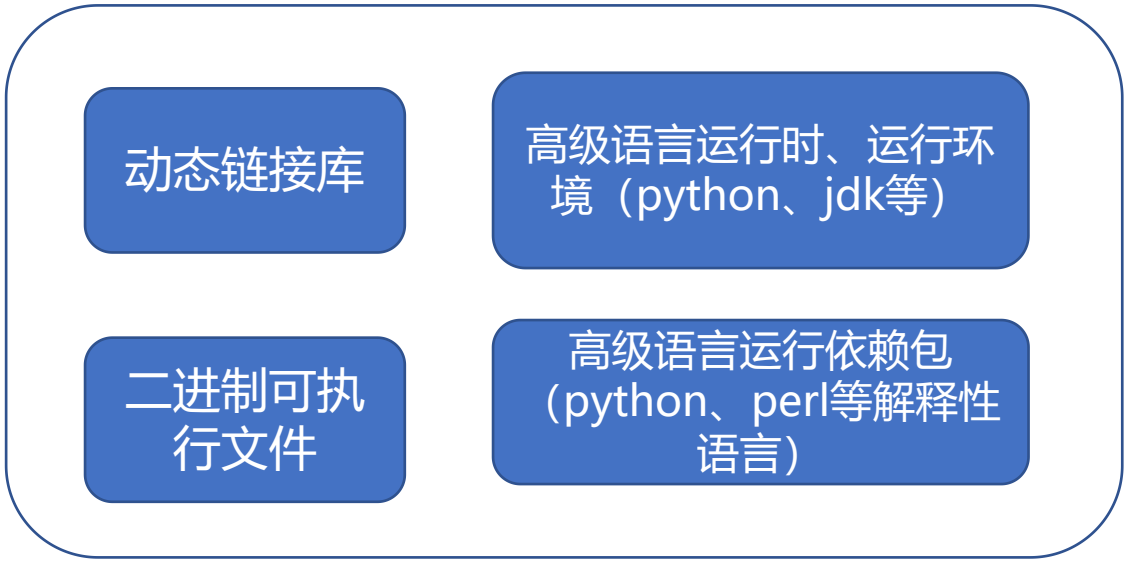
- 片段引用
- 文件引用
- 组件引用-Vendor
- 组件引用-包配置管理文件



SPEC打包文件引用情况



构建依赖



运行依赖

Patch文件引用情况

Patch来源 / Patch引入组件	自研	上游项目 补丁回合	第三方补丁引入
片段引用			
文件引用			
组件引用			

其他组件源代码包引用情况

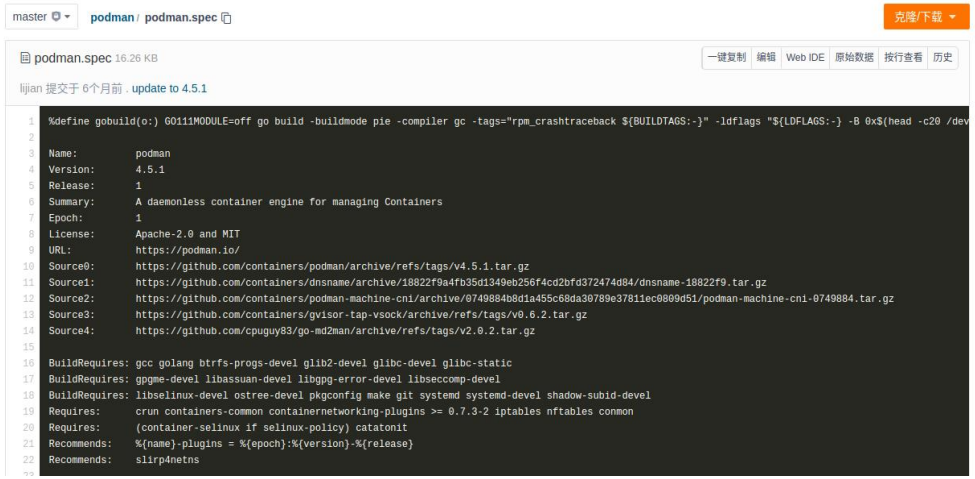
片段引用

文件引用

组件引用-Vendor

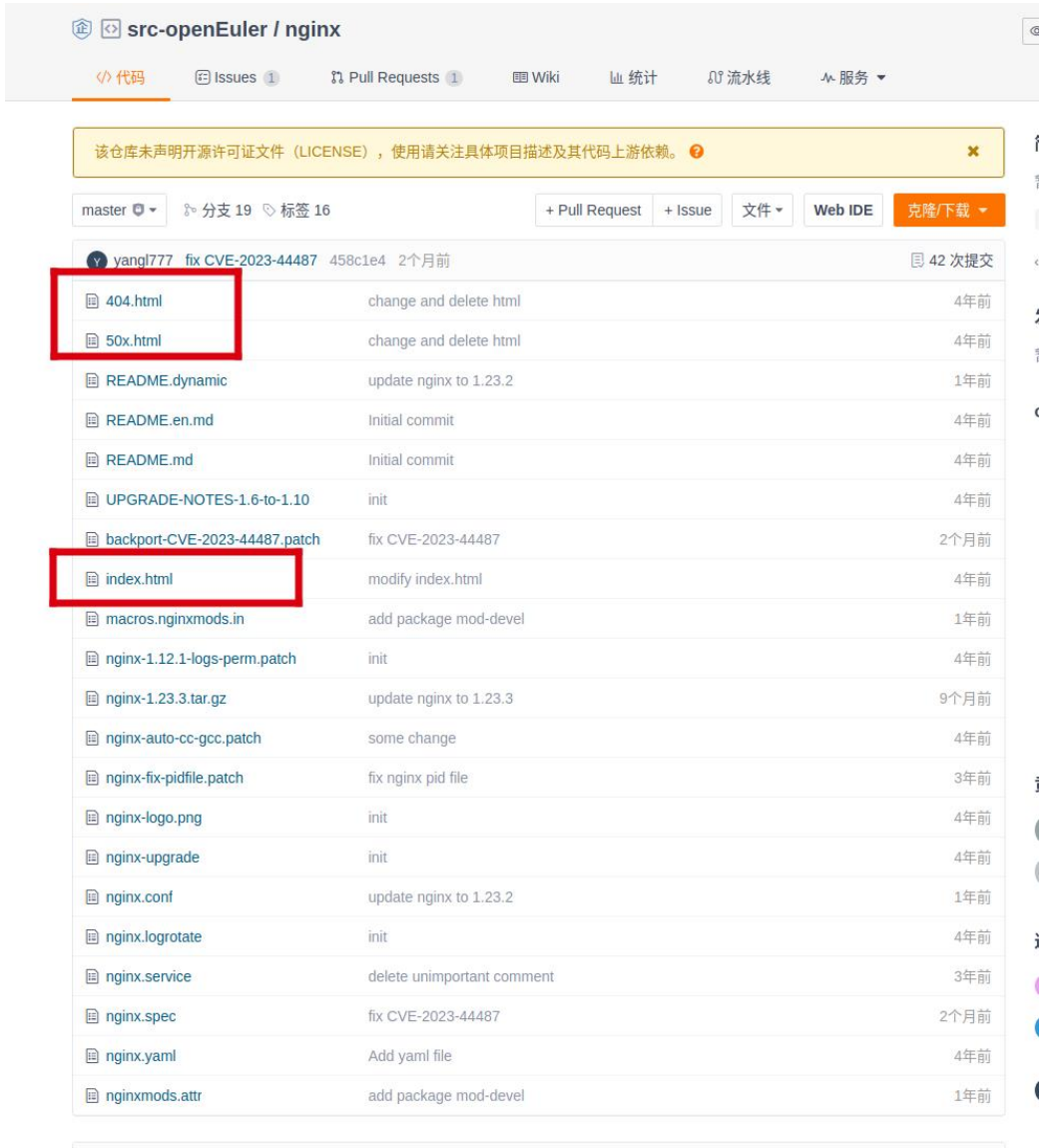
组件引用-包配置管理文件

理论上操作系统软件包中只应当存在一份源代码包，然而开发过程中存在此现象，应当判断该引入组件源代码包作用，最佳实践该组件应当单独作为软件包引入社区。

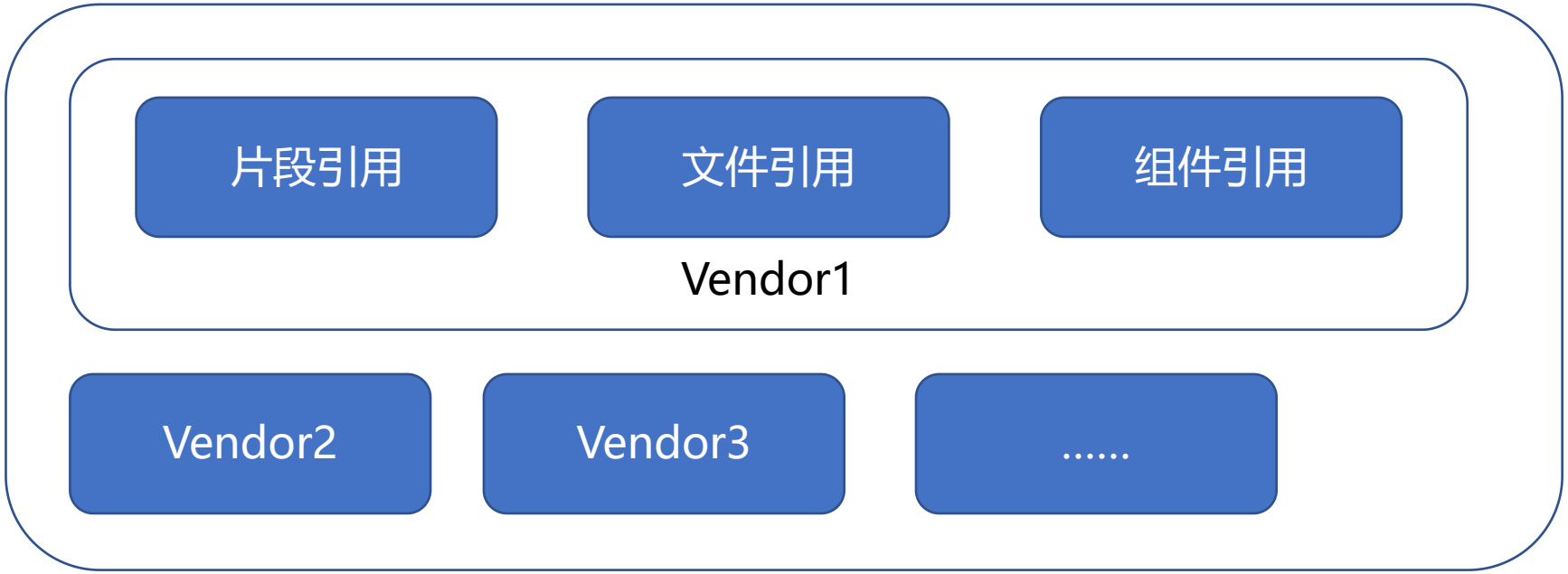


配置文件引用情况

绝大多数情况配置文件不作为版权及开源许可的声索依据，不会引入新的版权或开源许可证问题，但需考虑部分特殊情况可能会涉及商标等问题。



Vendor包引用情况



理论上操作系统软件包中应当将目标软件所需vendor同样打包，转换为软件包依赖，然而开发过程中存在此现象，如golang、rust等语言社区目前无组件引入计划，应用软件打包只能以开发者自行压缩Vendor目录的形式提供。此种方式对安全和开源合规造成较大隐患。

治理边界讨论

抛砖引玉

1.社区对操作系统软件包引入组件的开源合规治理边界?

2.社区软件包开源合规治理与功能业务需求冲突时该如何平衡?
(例如vendor打包问题)

3.开源合规治理中检测到问题由上游引入，该如何处理?

THANKS

开放原子开源基金会 OPENATOM FOUNDATION | OpenEuler