计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200400077

混源软件项目中的开源组件影响分析

赵 亮
军事科学院系统工程研究院系统总体研究所 北京 100101

摘 要 文中研究了混源软件的代码结构特征,按照功能已知性、代码有用性、代码安全性和知识产权属性等4个标准将混源代码空间进行了划分,展示了混源代码独特的代码空间特征;从正反两个方面分析了开源组件对混源项目的进度、质量、成本和知识产权等方面产生的正、反两方面的影响,将许可证根据传染性强弱分为3种类型;通过安全攸关领域开源组件应用的案例调研,展示了开源应用的基本情况,分析了开源组件实际应用中存在的问题。基于以上研究,认为从管理上应该建立开源组件的全生命周期管理机制,加大基于开源的创新,鼓励融入和回馈开源社区;从技术层面,应在项目初期即做好组件的选择,加强产品研发过程管理,并紧跟开源社区做好项目的演化,从而更好地利用开源,促进软件混源项目管理。

关键词:混源软件;开源组件;许可证
中图法分类号 TP311

Analysis of Impact of Open Source Components in Mixed Source Software Projects

ZHAO Liang
General System Institute, Department of System Engineering, Military Academy of Sciences, Beijing 100101, China

Abstract This paper studies the code structure characteristics of mixed source software, according to four standards, which include function knowledge, code usefulness, code security and intellectual property rights. This shows the unique code space of mixed source code. This paper analyzes the positive and negative effects of open source components on the progress, quality, cost and intellectual property rights of mixed source projects. The licenses are divided into three types according to their infectivity. Through the case study of open source component in the safety critical field software project, the basic situation of open source application is shown, and the problems existing in the practice are analyzed. Based on the above research, this paper brings forward that the whole life cycle management mechanism of open source components, and the innovation based on open source should be strengthened, and encourage integration and feedback to the open source community, from the technical view, we should make careful component selection, strengthen product development process management in the early stage of the project, and closely follow the open source community to product evolution. These can help to make better use of open source component and promote software mixed source project management.

Keywords Mixed source software, Open source software component, License

1 引言

基于开源组件的混源软件研发是当今软件项目研发的一种重要形式。Garnter报告显示,99%的企业在其业务系统中使用了开源软件[1]。数据显示,基于开源软件的软件研发方式,能够增加29%的创新,使质量提高26%,安全性提高29%,成本降低33%[2]。与此同时,任何一项技术都是双刃剑,在带来有利影响的同时,也有不利的影响。对混源软件而言,其因独特的代码结构特征对项目风险管理提出了新的要求,如由于原开发团队缺位、开发力量不能按需调用、开源项目本身的生命周期不可预期等因素带来的混源软件研发问题。同时,在安全攸关的领域,如电力、金融和国防科技等,未经严格审核的开源软件的广泛应用可能造成潜在的风险越来越大。客观分析开源组件对混源软件研发和软件质量的影响,规避基于开源应用带来的风险,成为现代软件研发必须要面对和解决的关键问题[3-5]。

2 混源软件结构分析

2.1 混源软件定义及特征

定义1(混源软件) 指混合多种来源、多种形式的代码构建而成的一种软件形式。

混源软件的特点包括:1)异构异质。混源软件的组件由来自不同地区、不同背景和不同能力的人员按照不同标准和流程研发,代码结构和质量存在巨大差别。2)状态与行为空间爆炸。各独立代码有各自的状态与行为空间,再加上代码之间的交互带来的放大效应,使得混源软件整体状态与行为空间爆炸。3)多源异步持续演化。不同来源和不同地区的软件组件仍然在按照各自独立的路径和不同的节奏进行持续演化。

2.2 混源软件代码空间

我们可以从功能已知性、代码有用性、代码安全性和知识产权属性4个方面分析混源代码独特的结构特征。

定义2(功能已知性) 指混源项目研发团队是否完全理

本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
通信作者:赵亮(liangzhao@tsinghua.edu.cn)

解和掌握第三方软件代码的所有功能。

定义 3(代码有用性) 指混源项目集成的代码中所包含的功能是否都服务于混源项目的功能。

定义 4(代码安全性) 指软件代码中是否包含安全缺陷或漏洞。

功能已知性产生的主要原因在于,即使第三方软件提供了完整的源代码,但是由于代码量巨大、系统结构复杂,对于使用方而言,很难在短时间内完全掌握所有代码的功能。另外,对于大多数项目而言,其使用的仅仅是第三方代码的一部分功能,集成方在一定时间内没有兴趣掌握所有的源代码功能。代码有用性产生的主要原因在于,第三方代码中包含了对本项目有用的内容,同时也引入了直观上对于本项目没有直接功能价值的代码,但是由于可能存在的定义、数据和引用上的依赖,这些代码不能简单地从原始项目中删除。基于功能已知性和功能有用性,整个混源代码空间可以分为 4 个区域:区域 I,功能已知且有用的代码;区域 II,功能已知但无用的代码;区域 III,功能未知且无用的代码;区域 IV,功能未知但有用的代码。

从代码安全性的角度看,混源代码包含了不安全的代码(八边形内)和安全的代码(八边形外),从代码的知识产权看,混源代码还包含了来自第三方知识产权的代码(星形部分)和自有知识产权代码(星形部分外)。

这样,混源软件的代码空间划分如图 1 所示的 3 个层次 4 个区域(注:为了表示的方便,在图 1 中形成了第三方知识产权代码和不安全代码的包含关系,而实际项目中,不安全的代码可能潜藏在混源软件的各个部分)。

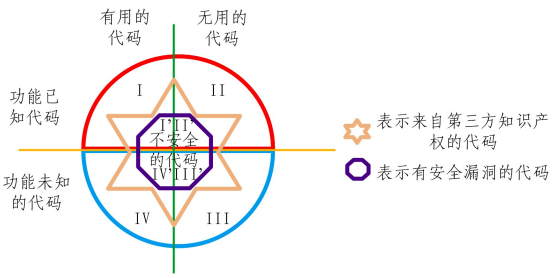

图 1 混源代码组成示意图

Fig. 1 Components of mixed-source code

这种特殊的代码空间显示了混源代码的代码结构特征。

3 开源组件的风险影响

本节从正、反两个方面和短期、长期两个角度,集中讨论开源组件对混源项目进度、质量、成本和知识产权等几个方面的影响。

3.1 开源组件对进度的影响

从实践的角度看,优秀的开源项目和代码具有良好的可读性,并且提供了相对丰富的文档,社区比较活跃,学习曲线峰度大,节约了项目研发和集成的时间。但是如果需要修改漏洞和深层定制,由于原始开发团队不可缺位,开发力量不能按需调用,也会造成进度方面的风险。尤其是选择使用的开源项目社区活跃度下降时,风险愈发加大。

3.2 开源组件对质量的影响

对于开源软件的质量有两种观点,一种观点认为开源软件经过了最广大用户的检验,代码质量有保障;另一种观点认为,开源软件没有遵循严格的软件工程方法,代码质量堪忧。从 Coverity 近年发布的开源代码质量报告看,开源软件的平均缺陷密度与商用软件的缺陷密度基本不相上下[6]。但是,开源软件中的软件漏洞不可避免,如 openssl 的 Heartbleed 漏洞,几乎影响了全球范围的网站安全。因此,需要及时发现和修补漏洞。对某些安全关键领域,一旦开源软件中被种植了恶意代码,后果会非常严重。另外,使用已经停止演化的开源项目,质量风险会进一步加大,影响系统的稳定性、健壮性与可维护性[7]。

3.3 开源组件对成本的影响

短期看,因为开源软件本身是免费使用和分发的,在项目中采用开源组件没有采购成本,而且由于研发周期缩短,进一步减小了研发成本。从长期维护的角度看,开源组件的维护由社区完成,亦降低了混源项目的维护成本。但是如果在开源组件的使用中违反了开源许可证协议,陷入许可证符合性、专利和商标纠纷,则可能带来知识产权方面的成本。

3.4 开源组件对知识产权的影响

知识产权是混源软件与传统自研项目的最大差别。由于混源代码中使用了第三方知识产权的内容,根本上改变了整个混源项目的知识产权结构,掺杂的许可证、专利和商标问题需要谨慎处理。因为开源组件导致的知识产权问题已经形成了上亿美元的诉讼,对原始研发厂商和开源软件团队都造成了巨大的影响[6]。根据开源许可证对开源软件复制、修改、分发等操作要求的不同,可以将许可证分为 3 类。

P1:强传染性许可证。该许可证要求软件发布时必须公布源码、保留完整的原作者版权信息、不可以使用其他许可证发布、始终保护该软件及其衍生品不被商业化等,如 GPL,AGPL。

P2:弱传染性许可证。该许可证要求软件发布时可以与闭源软件混合,但须保留完整的原作者版权信息,并且保持原部分继续以原许可证发布,其他部分可以使用其他许可协议发布,如 LGPL,EPL,MPL 等。

P3:开放型许可证。该类许可证鼓励代码共享,要求在后续衍生品中保持原作者版权信息的完整性,并持续传递,允许在许可条件中增加自己的许可,但不允许修改原始许可,代码可以开源也可以商用,如 Apache,BSD,MIT 等。

4 案例分析

4.1 调研情况

为研究混源项目中开源组件的影响,本文调研了 7 个单位的 27 个项目中开源组件的使用情况。这 7 个单位分别为航天、航空、船舶和电子科技领域的研发机构。从性质上说,这些单位都属于强监管单位,对项目质量和安全性有较高要求。我们的主要调研工作在于了解各单位近 5 年来在项目中使用开源软件的情况。项目类型范围包括:计划任务、基金计划、合作项目以及自研项目等。对采集到的信息,进一步统计了以下信息:1)开源项目的版本信息,包括应用版本和当前最新版本信息;2)对应项目的开源许可证信息;3)NVD 数据库查询开源组件的已知漏洞信息,即到 NVD 网站输入开源组件名称,统计显示的与之相关的漏洞信息数量(截至 2020 年 04 月 10 日)[8]。

开源组件影响分析结果如表 1 所列。

表1　开源组件影响分析

Table 1　Analysis of open source components

项目		正面因素	负面因素
短期影响	进度	缩短开发时间,提高开发效率,尽快完成交付	原开发力量不能按需调用,对于混源代码的学习和掌握需要一定的时间
	质量	通过测试和广泛应用的开源组件,软件质量有保障	质量差、过时的开源组件会降低软件质量
	成本	采购成本为0,二次研发成本低	学习成本;开源许可证对开源组件的使用、修改和分发有明确要求,极端情况下会产生新的成本
	知识产权	基于开源组件可以迅速构造自有知识产权的产品	传染性强的开源组件要求衍生产品按照许可证要求发布软件,共享知识产权
长期影响	进度	成熟的开源组件,社区有专门的维护力量,可以伴随升级	由于原开发团队缺位、文档缺失、功能定制需要较长时间,版本演化路径不可控
	质量	随着社区开源组件的演化,提高软件应对风险的能力	开源组件停止演化,会影响软件
	成本	伴随社区的升级获得更新的功能,减少软件维护需要的人力成本	专利、知识产权的纠纷带来更多的成本投入,不可控制
	知识产权	基于开放型许可证项目,可以迅速构造自身产品;伴随开源组件的推广,提高产品的影响力	产品门槛降低,同领域商业竞争激烈;知识产权掺杂,存在产生纠纷的可能性

4.2　调研分析

4.2.1　基本数据

1)开源组件应用广泛。从调研结果看,被调研单位几乎在所有的新研软件项目中都采用或引入了开源组件。开源组件无论是作为工具、基础环境或者研发的初始平台,均已广泛应用在强监管领域的各条生产线和各个研究领域。

2)开源组件引入效果鲜明。引入开源显著缩短了项目研发周期,降低了成本,提高了研发效率,并为创新性提供了主要的技术途径。开源组件提供了各单位开展研究的基础平台和实现创新突破的关键路径,使原来“做不了”的变成“可以做”,如基于 chromium 内核研制国产平台上的浏览器。

3)基于开源组件的产品品牌。基于开源组件研发的项目大都形成了研制单位的自研产品,并进行了二次分发。

4.2.2　存在的问题

1)开源组件引进的随意性。这些单位本来都属于强监管领域,对输入、输出和研发过程有严格的要求,但是在开源组件引入中,并没有建立一套完整的流程和管理机制,存在一定的决策随意性。各项目组在引进开源组件时基本没有考虑许可证协议的要求和应用领域的特征,在使用中也没有遵循相关的约定,存在比较严重的许可证符合性违规风险。

2)开源组件应用的简单性。开源组件引进后,作为基础环境、运行环境和工具环境等使用的居多,对开源组件本身进行二次开发和改进的比较少。开源组件起到了商用软件的替代作用,但并没有形成具有核心竞争力的技术和好用易用的产品。

3)开源组件演化的封闭性。从引用版本看,只有极少数项目使用了对应开源软件的最新版本(与项目立项时间有关)。大部分项目在开源组件引进之后实现了闭源管理。这一方面使得开源组件演化的新成果没有及时进入到自研产品中,形成了与开源社区的技术代差;另一方面,闭源演化使得开源组件失去了维护和更新,问题得不到及时修正,导致很多已经暴漏或公布的漏洞没有及时打补丁。这种漏洞的数量非常惊人(如表2所列),如果产品应用于核心关键领域,极易被攻击。

表2　开源组件信息列表

Table 2　List of open source components

NO.	软件名称	所用版本	最新版本	已知漏洞	开源协议	等级
1	chromium	V37	V68	1580	BSD	P3
2	Tomcat	V8	V9.0	268	Apache	P3
3	Eclipse	V3.6.1	V7.2	88	Eclipse	P2
4	openssh-client	V7.4	V8.2	117	OpenBSD	P3
5	docker	V1.10.3	V3.0	83	Apache2.0	P3
6	seaweedfs	V1.1	V1.33	—	Apache	P3
7	dpdk	V18.02	V19.02	4	GPL	P1
8	snort	V3.0beta	V3.0beta	47	GPL	P1
9	rabbitmq	V3.7.9	V3.7.14	25	MPL	P2
10	ceph	V12.2.10	V13.2.0	32	GPL	P1
11	hadoop	V2.8.4	V3.2.0	44	Apache	P3
12	zookeeper	V3.4.6	V3.5.4beta	28	Apache	P3
13	spark	V2.2.1	V2.4.2	108	Apache	P3
14	hive	V2.3.3	V3.1.1	25207	Apache	P3
15	hbase	V1.3.2	V2.1.4	22	Apache	P3
16	ambari	V2.6.1	V2.7.3	18	Apache	P3
17	elastic stack	V5.6.9	V7.0	33	elastic	P2
18	openjdk	V8	V11	155	GPL	P1
19	libvirt	V4.3.0	V5.3.0	86	GPL	P1
20	qemu	V2.12	V4.0.0	357	GPL	P1
21	openstack	Juno	Train	270	Apache	P3
22	Spring	V4.2.0	V5.1.7	133	Apache	P3
23	hibernate	V4.3.5	V5.4.2	11	LGPL	P2
24	Struts	V2.2.1	V2.5.20	86	Apache	P3
25	MySQL	V5.1.31	V8.0	1215	GPL	P1
26	linux	V2.6	V5.4	7890	GPL	P1
27	QT	V5.12	V5.12	489	GPL	P1
28	FASTDB	V2.49	V3.75	—	FASTDB	P3
29	ZeroMQ	V4.2.1	V4.3.2	6	LGPL	P2

4)混源项目知识产权“双标”意识严重。大部分开源组件被集成后,形成了自己的品牌标识。厂商注重了对自己的知识产权保护,但是对于是否遵守了开源许可证的知识产权要求,意识却比较淡漠,典型的双重标准。目前其产品运行的相对安全性主要建立在应用领域封闭性的基础上。随着应用范围的扩展和接入技术的发展,许可证符合性风险及其他知识产权风险可能会逐渐暴露。

5　混源软件的风险管理

5.1　管理层面

(1)建立开源组件全生命周期管理机制。项目单位应设

(下转第583页)