



<https://www.qianxin.com>

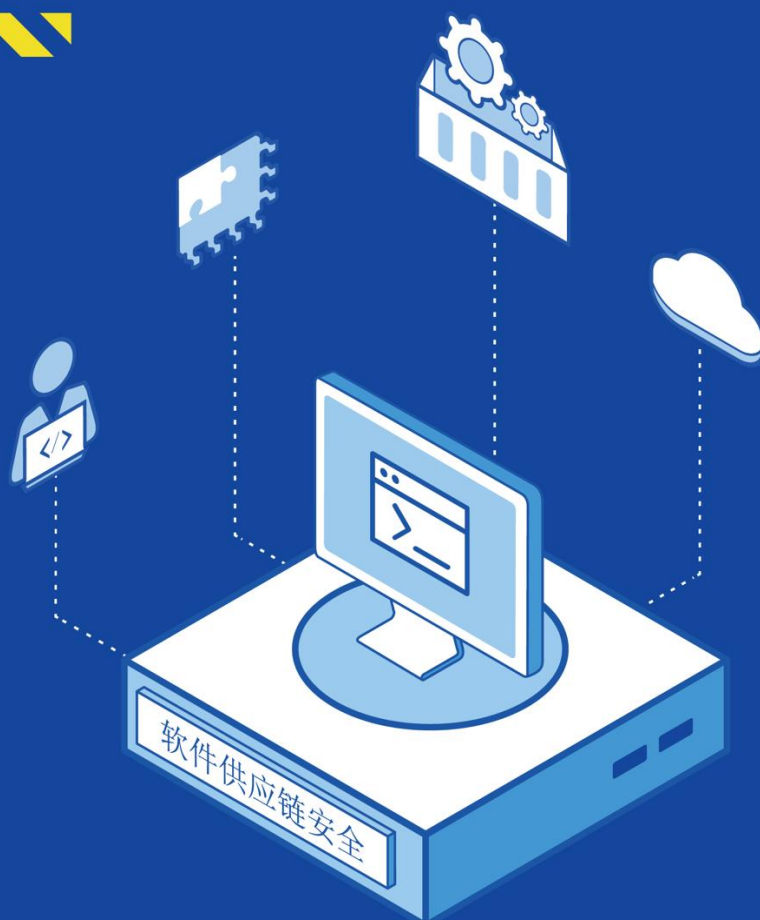
北京2022年冬奥会官方赞助商
Official Sponsor of the Olympic Winter Games Beijing 2022

2022 中国软件供应链 安全分析报告

T H E R E P O R T

发布机构:

奇安信代码安全实验室



目 录

一、概述	1
1、软件供应链安全攻击事件保持持续高发	1
2、开源软件安全风险是当前软件供应链安全的焦点问题	3
二、国内企业自主开发源代码安全状况	5
1、编程语言分布情况	5
2、典型安全缺陷检出情况	6
三、开源软件生态发展与安全状况	7
1、开源软件生态发展状况分析	7
2、开源软件源代码安全状况分析	9
(1) 编程语言分布情况	9
(2) 典型安全缺陷检出情况	10
3、开源软件公开报告漏洞状况分析	10
(1) 大型开源项目漏洞总数及年度增长 TOP20	10
(2) 主流开源软件包生态系统漏洞总数及年度增长 TOP20	12
4、开源软件活跃度状况分析	14
(1) 7 成开源软件项目处于不活跃状态	14



(2) 近 2 万个开源软件一年内更新发布超过 100 个版本	15
5、关键基础开源软件分析.....	16
(1) 主流开源生态关键基础开源软件 TOP50.....	16
(2) 2/3 的关键基础开源软件从未公开披露过漏洞.....	19
(3) 关键基础开源软件的整体运维风险较高.....	20
四、国内企业软件开发中开源软件应用状况.....	20
1、开源软件总体使用情况分析.....	21
(1) 平均每个软件项目使用 127 个开源软件.....	21
(2) 流行开源软件被超过 1/3 的软件项目使用.....	21
2、开源软件漏洞风险分析.....	22
(1) 77%的软件项目存在容易利用的开源软件漏洞.....	22
(2) 平均每个软件项目存在 69 个已知开源软件漏洞.....	23
(3) 影响最广的开源软件漏洞存在于超 3 成的软件项目中	23
(4) 16 年前的开源软件漏洞仍然存在于多个软件项目中	25
3、开源软件许可协议风险分析.....	25
(1) 最流行的开源许可协议在超 3/4 的项目中使用	26
(2) 45.6%的项目使用了含有高风险许可协议的开源软件	26
4、开源软件运维风险分析.....	27
(1) 20 年前的老旧开源软件版本仍在被使用.....	27



(2) 开源软件各版本使用更加混乱	28
五、典型软件供应链安全风险实例分析	29
1、某主流网络接入存储 (NAS) 设备供应链攻击实例分析	29
2、某主流 VPN 路由器供应链攻击实例分析	31
3、三款国产操作系统供应链攻击实例分析	33
4、某国产邮件系统供应链攻击实例分析	35
5、Edge 浏览器供应链攻击实例分析	36
六、总结及建议	38
附录：奇安信代码安全实验室简介	40



一、概述

数字化时代，软件的重要性和软件供应链安全问题的严峻性已成为各方共识。为此，奇安信代码安全实验室去年发布了《2021 中国软件供应链安全分析报告》(<https://h5.qianxin.com/threat/report/detail/132>)，从多个维度分析了软件供应链的安全风险，并提供了详实的统计数据。

本报告是该系列年度分析报告的第二期，继续针对国内企业自主开发的源代码、开源软件生态、国内企业软件开发中开源软件应用等的安全状况，以及典型应用系统供应链安全风险实例进行深入分析，并总结趋势和变化。本年度报告相比于去年的报告新增了以下内容：在开源软件生态发展与安全部分新增了关键基础开源软件分析；在企业软件开发中的开源软件应用部分新增了漏洞利用难度的统计分析和开源许可协议风险分析；在典型软件供应链安全风险实例部分，重点分析了利用开源软件“老漏洞”攻破最新主流产品的实例，通过多个实例验证了由于软件供应链的复杂性，开源软件的“老漏洞”也可以起到“0day 漏洞”的攻击效果。上述报告内容的变化，感兴趣的读者在阅读时可以重点关注。

1、软件供应链安全攻击事件保持持续高发

在过去的一年中，针对软件供应链的安全攻击事件依然呈现出高发态势，造成的危害也非常严重。

2021 年 8 月，台湾芯片设计厂商 Realtek 称，其 WiFi 模块的三款开发包 (SDK) 中存在 4 个严重漏洞。攻击者可利用这些漏洞攻陷目标设备并以最高权限执行任意代码。这些 SDK 用于至少 65 家厂商制造的近 200 款物联网设备中。

2021 年 10 月，攻击者劫持了 NPM 包 `ua-parser-js` 作者的账户约 4 小时，意图安装恶意软件。`ua-parser-js` 每周下载量超过 700 万次，广泛应用于 Facebook、苹果、亚马逊、微软、IBM 等硅谷巨头企业中。

2021 年 11 月，热门 NPM 包 `coa` 和 `rc` 连续遭劫持，并被植入恶意代码，影响全球 React 管道。`coa` 库每周下载量约 900 万，用于 GitHub 上近 500 万个开源库中，`rc` 库每周下载量达 1400 万。

2021 年 12 月，Apache Log4j2 曝出 Log4Shell 漏洞 (CVE-2021-44228)，Apache Log4j2 是 Java 应用最广泛的开源日志组件，广泛应用于政府、企业和公共服务机构的平台、应用和业务系统中，该漏洞覆盖范围广而且利用门槛低，对大量系统和机构造成了严重影响。

2022 年 3 月，俄乌冲突中，NPM 开源包 `Node-ipc` 的作者 RIAEvangelist 作为反战人士，在代码仓库中进行了“供应链投毒”，添加的恶意 `js` 文件能够在包使用者的桌面创建反战标语。`Node-ipc` 使用非常广泛，每周下载量超过 100 万次。

2022 年 4 月，以色列安全公司发现高通和联发科芯片的音频解码器存在三个漏洞 (CVSS 评分为 9.8、7.8 和 5.5)，来源于 11 年前苹果公司开发的开源无损音频编解码器 `Apple Lossless`。利用这些

漏洞进行攻击，可获取受影响移动设备媒体、音频会话的访问权限及摄像头数据流等，漏洞影响范围包括数百万安卓设备。

2022 年 5 月，安全研究人员在 NPM 注册表中发现了一些恶意软件包，与大多数恶意软件相比，其危险性更大，攻击者可以通过后门完全控制被感染的机器。分析人员称，这是一次依赖混淆攻击，其目标非常明确，并且掌握了非常机密的内部信息。德国的著名媒体、物流和工业企业等机构受到攻击。

2、开源软件安全风险是当前软件供应链安全的焦点问题

奇安信代码安全实验室通过数据对比分析发现，与前一年度相比，国内企业自主开发源代码的安全状况有较明显的改善，千行代码缺陷密度和十类典型安全缺陷的总体检出率均有明显下降，这应该得益于软件源代码安全缺陷分析工具的持续应用，以及程序员编写代码时的安全意识提高。但开源软件安全风险仍然居高不下，开源软件的安全风险管控是当前软件供应链安全保障需要解决的核心焦点问题。

1) 国内企业自主开发源代码安全性明显改善

2021 年，国内企业自主开发源代码的整体缺陷密度和高危缺陷密度相比 2020 年均有所下降，其中高危缺陷密度下降较为明显，从 1.08 个/千行降为 0.65 个/千行；十类典型安全缺陷的总体检出率从 2020 年的 77.8% 下降到 59.9%，下降了近 18%。

2) 开源生态保持蓬勃发展，开源软件自身安全问题更加严峻

2020 年底和 2021 年底，主流开源软件包生态系统中开源项目总

量分别为 3814194 个和 4395386 个，一年间增长了 15.2%，开源生态依然保持蓬勃发展的态势。与此同时，开源软件漏洞数量持续增长，2021 年新增的开源软件漏洞达到 6346 个；不活跃的开源项目占比从去年报告的 61.6% 提升至 69.9%。根据“奇安信开源项目检测计划”的实测数据显示，所检测开源软件的总体缺陷密度和高危缺陷密度略高于去年，依然处于较高的水平；十类典型缺陷的总体检出率为 73.5%，远高于去年的 56.3%。总体来看，开源软件自身的安全问题日益严峻。

3) 国内企业软件开发中，开源软件安全风险问题未见改善

2021 年，奇安信代码安全实验室对 3354 个国内企业软件项目中使用开源软件的情况进行了分析，从数据分析情况来看，国内企业使用开源软件时的安全风险问题没有得到改善，开源软件安全风险是当前企业软件开发中亟待解决的首要问题。

在 2021 年分析的 3354 个软件项目中，平均每个项目使用 127 个开源软件；存在已知开源软件漏洞的项目占比 86.4%，平均每个项目存在 69 个已知开源软件漏洞，十几年前的古老开源软件漏洞依然存在于多个项目中；20 年前的老旧开源软件版本仍然在使用，同一开源软件各版本的使用更加混乱，被使用版本最多的开源软件有 235 个版本在使用。整体的安全风险状况与 2020 年相比没有任何改善，风险水平仍处于较高状态。

与此同时，我们观察到 2021 年很多机构和企业开始关注开源软件供应链安全，但仅有少数的机构和企业应用了相关的开源安全治理工具。相信随着开源安全治理工具的推广和持续应用，国内企业软件

开发中的开源软件安全风险问题会逐步得到改善。

二、国内企业自主开发源代码安全状况

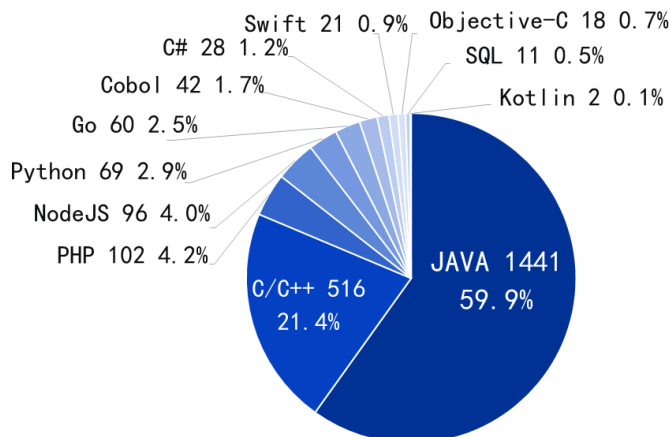
源代码安全是软件供应链安全的基础。2021 年全年，奇安信代码安全实验室对 2406 个国内企业自主开发的软件项目源代码进行了安全缺陷检测，检测的代码总量为 550808816 行，共发现安全缺陷 5427125 个，其中高危缺陷 359055 个，整体缺陷密度为 9.85 个/千行，高危缺陷密度为 0.65 个/千行。与 2020 年相比，整体缺陷密度和高危缺陷密度均有下降，其中高危缺陷密度下降较为明显，从 1.08 个/千行下降至 0.65 个/千行。

1、编程语言分布情况

在 2406 个国内企业自主开发的软件项目中，使用数量排名前 3 的编程语言为 Java、C/C++、PHP，对应的软件项目数量分别为 1441 个、516 个和 102 个，其中 Java 语言项目占比达 60%。相关国内企业在进行软件开发时，Java 语言仍然是首选。编程语言的分布情况如下图所示。



国内企业自主开发的软件项目编程语言总体分布情况



2、典型安全缺陷检出情况

输入验证、路径遍历、跨站脚本、注入、NULL 引用、资源管理、密码管理、API 误用、配置管理、日志伪造等十类典型安全缺陷的总体检出率（检出率指含有某类缺陷的软件项目数占软件项目总数的比例）为 59.9%，低于去年报告的 77.8%。

每类典型缺陷的检出率、排名及排名变化情况如下表所示。除 API 误用类缺陷的检出率从去年报告中的 28.7% 升至 31.5% 外，其他缺陷的检出率均有不同程度的下降。

排名	缺陷类型	检出率	排名变化
1	输入验证	41.8%	0
2	跨站脚本	35.1%	↑ 1
3	API 误用	31.5%	↑ 5
4	NULL 引用	30.2%	↑ 1
5	资源管理	29.9%	↑ 1
6	路径遍历	28.4%	↓ 4
7	注入	26.3%	↓ 3

8	密码管理	22.8%	↓ 1
9	配置管理	18.2%	0
10	日志伪造	12.5%	0

三、开源软件生态发展与安全状况

开源软件是现代软件开发最基础的原材料，其代码自身的安全状况，会直接影响最终软件的安全性。本年度报告从开源软件生态发展状况、开源软件源代码安全状况、开源软件公开报告漏洞状况、开源软件活跃度状况、关键基础开源软件分析等五个方面对 2021 年开源软件生态发展与安全状况进行综合分析。相比于去年的报告，本章节新增了“关键基础开源软件分析”部分，针对使用广泛的关键性、基础性开源软件进行分析，希望能够初步刻画当前软件世界的“底座”及其安全状况。

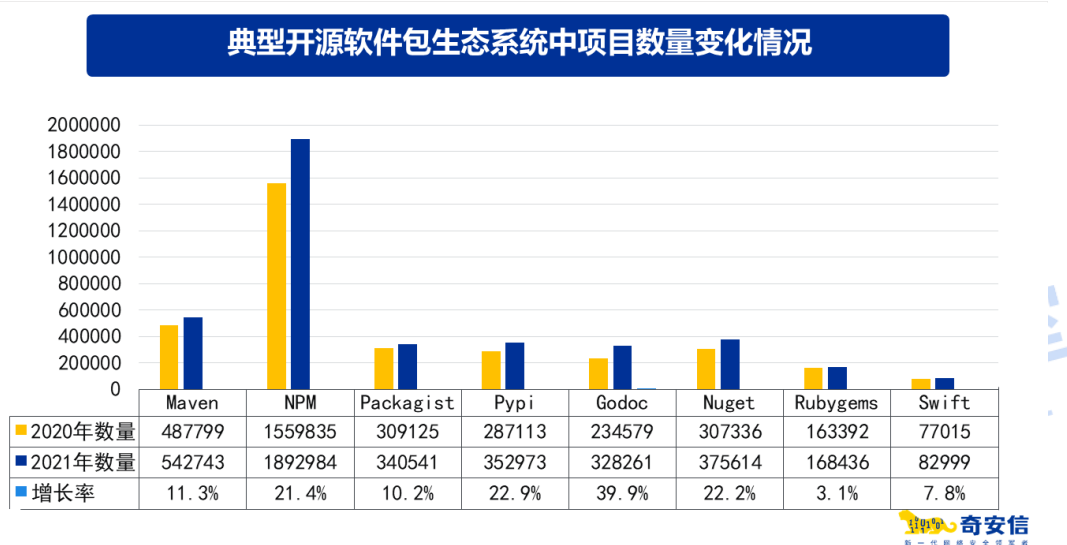
1、开源软件生态发展状况分析

据奇安信代码安全实验室监测和统计，2020 年底和 2021 年底，主流开源软件包生态系统中开源项目总量分别为 3814194 个和 4395386 个，一年间增长了 15.2%；截至 2021 年底，主流开源软件包生态系统中平均每个开源项目有 11.8 个版本，较去年报告的 10.2 个版本有所增加。可以看出，2021 年开源软件生态依然繁荣。

对 Maven、NPM、Packagist、Pypi、Godoc、Nuget、Rubygems、Swift 等八个典型开源软件包生态系统的具体分析如下：

NPM 包生态项目数量最多，Godoc 包生态增速最快。这一状况与

去年报告保持一致。八个典型的开源软件包生态系统中开源项目数量和增长率情况如下图所示，其中开源项目数量最多的是 NPM 包生态系统，截至 2021 年底，其开源项目数量达到了 1892984 个；开源项目数量增速最快的是 Godoc 包生态系统，2021 年一年间的项目总量增速达到了 39.9%。



Maven、NPM、Nuget 三个包生态系统的开源项目开发者仍然是“最勤奋”，开源项目的平均版本数超过 12 个。截至 2021 年底，八个典型的开源软件包生态系统的开源项目数量和版本数量如下表所示。其中，Maven 包生态系统平均每个开源项目有高达 19.9 个版本，各包生态系统的开源项目平均版本数较去年均有不同程度的增长。

序号	包生态系统	2021 年项目数	2021 年版本数	平均版本数
1	Maven	542743	10803609	19.9
2	NPM	1892984	23453934	12.4
3	Packagist	340541	3687777	10.8
4	Pypi	352973	3209147	9.1
5	Godoc	328261	2758135	8.4

6	Nuget	375614	4773760	12.7
7	Rubygems	168436	1169942	6.9
8	Swift	82999	529584	6.4

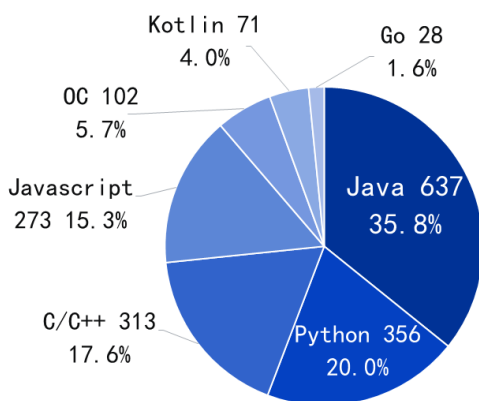
2、开源软件源代码安全状况分析

2021 年全年，“奇安信开源项目检测计划”对 1780 个开源软件项目的源代码进行了安全检测，代码总量为 164414083 行，共发现安全缺陷 2648242 个，其中高危缺陷 162959 个。2021 年检测的 1780 个开源软件项目整体缺陷密度为 16.11 个/千行，高危缺陷密度为 0.99 个/千行，均略高于去年报告的 14.96 个/千行和 0.95 个/千行。

(1) 编程语言分布情况

检测的 1780 个开源项目中，共涉及 7 种编程语言，分别是 Java、C/C++、Python、OC、Go、JavaScript、Kotlin，编程语言的分布情况如下图所示。

开源项目编程语言总体分布情况



(2) 典型安全缺陷检出情况

2021 年检测的 1780 个开源软件项目中，输入验证、路径遍历、跨站脚本、注入、NULL 引用、资源管理、密码管理、API 误用、配置管理、日志伪造等十类典型安全缺陷的总体检出率为 73.5%，远高于去年报告的 56.3%。每类典型缺陷的检出率、排名及排名变化情况如下表所示。

排名	缺陷类型	检出率	排名变化
1	路径遍历	36.7%	↑ 1
2	Null 引用	36.5%	↑ 2
3	资源管理	33.0%	↑ 3
4	输入验证	25.1%	↓ 3
5	密码管理	23.5%	↑ 4
6	日志伪造	22.9%	↑ 2
7	注入	21.4%	↓ 4
8	API 误用	18.2%	↓ 3
9	跨站脚本	15.0%	↓ 2
10	配置管理	10.8%	0

3、开源软件公开报告漏洞状况分析

据奇安信代码安全实验室监测与统计，截至 2021 年底，CVE/NVD、CNNVD、CNVD 等公开漏洞库中共收录开源软件相关漏洞 52716 个，其中有 6346 个漏洞为 2021 年新增漏洞。

(1) 大型开源项目漏洞总数及年度增长 TOP20

截至 2021 年底，历史漏洞总数排名前 20 的大型开源项目信息如

下表所示。历史漏洞数排在前三的项目与 2020 年相同，是 Linux Kernel、Chromium (Google Chrome) 和 Mozilla Firefox。

序号	大型开源项目	主页地址	历史漏洞总数
1	Linux Kernel	https://www.kernel.org/	4653
2	Chromium(Google Chrome)	http://www.chromium.org/	2695
3	Mozilla Firefox	https://www.mozilla.org/en-US/firefox/	2241
4	MySQL	https://dev.mysql.com/	1171
5	Thunderbird	https://www.thunderbird.net/zh-CN/	1119
6	Adobe Flash Player plugin	https://www.adobe.com/products/flashplayer/end-of-life.html	1087
7	Firefox ESR	https://www.mozilla.org/en-US/firefox/enterprise/	863
8	SeaMonkey	https://www.seamonkey-project.org/	706
9	Drupal (core)	https://www.drupal.org/	699
10	PHP	https://www.php.net/	678
11	ImageMagick	https://imagemagick.org/index.php	624
12	GitLab	https://about.gitlab.com/	623
13	Wireshark	https://www.wireshark.org/	623
14	WebKitGTK	http://webkitgtk.org/	591
15	WordPress	https://wordpress.org/	575
16	OpenJDK	https://openjdk.java.net/	518
17	Moodle	https://moodle.org/	442
18	Xen Project (Hypervisor)	https://xenproject.org/	411
19	FFmpeg	https://ffmpeg.org/	392
20	QEMU	https://www.qemu.org/	384

2021 年一年间，公开报告漏洞数量增长排名前 20 的大型开源项目信息如下表所示，Linux Kernel 一年来增加的漏洞最多，达到 419 个。



序号	大型开源项目	主页地址	2021 年漏洞增量
1	Linux Kernel	https://www.kernel.org/	419
2	Chromium (Google Chrome)	http://www.chromium.org/	346
3	TensorFlow	https://www.tensorflow.org/	201
4	MySQL	https://dev.mysql.com/	191
5	GitLab	https://about.gitlab.com/	158
6	Mozilla Firefox	https://www.mozilla.org/en-US/firefox/	123
7	gpac	https://gpac.wp.imt.fr/	116
8	Electron - Cross-platform desktop application shell	https://www.electronjs.org/	83
9	Thunderbird	https://www.thunderbird.net/zh-CN/	79
10	FFmpeg	https://ffmpeg.org/	66
11	Firefox ESR	https://www.mozilla.org/en-US/firefox/enterprise/	62
12	WebKitGTK	http://webkitgtk.org/	60
13	MediaWiki	https://www.mediawiki.org/wiki/MediaWiki	50
14	Oracle VM VirtualBox	https://www.virtualbox.org/	49
15	QEMU	https://www.qemu.org/	41
16	Magento	https://business.adobe.com/	39
17	Nextcloud	https://nextcloud.com/	36
18	Xen Project (Hypervisor)	https://xenproject.org/	34
19	libredwg	https://www.gnu.org/software/	33
20	SWFTTools	http://www.swftools.org/	32

(2) 主流开源软件包生态系统漏洞总数及年度增长 TOP20

截至 2021 年底，主流开源软件包生态系统中历史漏洞总数排名

前 20 的开源软件信息如下表所示。

序号	开源软件	所属包生态系统	历史漏洞总数
1	Apache Tomcat	Maven	183
2	FFmpeg-iOS	Swift	177
3	TYP03 CMS	Packagist	130
4	Magento Core	Packagist	117
5	Ruby on Rails	Rubygems	108
6	Plone	Pypi	90
7	Puppet	Rubygems	86
8	Django	Pypi	86
9	Dolibarr ERP & CRM	Packagist	80
10	Apache Struts	Maven	80
11	Symfony	Packagist	78
12	silverstripe-framework	Packagist	78
13	keycloak	Maven	72
14	Ansible	Pypi、Conda	69
15	jackson-databind	Maven	68
16	Data Mapper for Jackson	Maven	58
17	jackson-mapper-asl	maven	56
18	OpenShift Origin	godoc	55
19	salt	Pypi	46
20	Kubernetes	godoc	46

2021 年一年间，主流开源软件包生态系统中公开报告漏洞数量增长排名前 20 的开源软件信息如下表所示。

序号	开源软件	所属包生态系统	2021 年漏洞增量
1	FFmpeg-iOS	Swift	67
2	Magento Core	Packagist	23
3	Go programming language	Godoc	20



4	Python Pillow	Pypi	20
5	Vaadin	Maven	19
6	Quarkus	Maven	19
7	keycloak	Maven	16
8	TYP03 CMS	Packagist	16
9	salt	Pypi、Conda	14
10	jackson-databind	Maven	13
11	OpenEXR	Conan	13
12	Firefly III	Packagist	12
13	dubbo	Maven	12
14	Synapse homeserver for Matrix.org	Pypi	12
15	Elasticsearch	Maven	12
16	Data Mapper for Jackson	Maven	12
17	jackson-mapper-asl	Maven	12
18	Apache Solr	Maven	11
19	Plone	Pypi	11
20	WebP	Swift	11

4、开源软件活跃度状况分析

活跃度也是衡量开源软件安全性的一个重要维度。不活跃的开源软件，一旦出现安全漏洞，难以得到及时的修复；活跃的开源软件中，如果其版本更新发布的频率过高，同样会增加使用者运维的成本和安全风险。

(1) 7 成开源软件项目处于不活跃状态

我们将超过一年未更新发布过版本的开源软件项目定义为不活

跃项目。2021 年全年，主流开源软件包生态系统中不活跃的开源软件项目数量为 3071132 个，占比达到 69.9%，与去年报告的 61.6%相比，有所增加。

对八个典型的开源软件包生态系统进行分析和比较发现，各包生态系统中不活跃项目的占比较去年报告数据均有小幅升高，其中 NPM 的不活跃项目数量最多，达 1318868 个，Rubygems 的不活跃项目比例最高，占比达到 89.6%。具体数据见下表。

序号	包生态系统	项目总数	不活跃项目数	不活跃项目比例
1	Maven	542743	352390	64.9%
2	NPM	1892984	1318868	69.7%
3	Packagist	340541	240325	70.6%
4	Pypi	352973	217348	61.6%
5	Godoc	328261	222079	67.7%
6	Nuget	375614	255619	68.1%
7	Rubygems	168436	150912	89.6%
8	Swift	82999	70621	85.1%

(2) 近 2 万个开源软件一年内更新发布超过 100 个版本

2021 年全年，主流开源软件包生态系统中，更新发布 100 个以上版本的开源项目有 19265 个，高于去年报告的 13411 个。八个典型的开源软件包生态系统中，一年内更新发布超过 100 个版本的项目数量见下表。

排名	包生态系统	对应的开发语言	一年内发布超过 100 个版本的项目数量
1	NPM	Javascript	9961
2	Maven	Java	4161



3	Nuget	.NET	1624
4	Godoc	Go	1403
5	Pypi	Python	1085
6	Packagist	PHP	883
7	Rubygems	Ruby	51
8	Swift	Swift	16

5、关键基础开源软件分析

现代软件开发的“大厦”可以说构建在开源生态基础之上。在开源生态中，有一些开源软件的地位非常重要，它们被众多其他的开源软件所依赖，被广泛的使用在各种软件系统之中，这些开源软件可以说是现代软件开发的核心基础设施和底座，我们将其称为“关键基础开源软件”。

因为“Log4Shell”漏洞(CVE-2021-44228)而被更多人熟知的 Apache Log4j2 就是关键基础开源软件中的一员，“Log4Shell”漏洞也让我们认识到，关键基础开源软件一旦出现漏洞，其放大效应非常显著，影响范围巨大且难以消除。本节将分析主流开源生态中的关键基础开源软件及其安全状况，希望通过此分析，可以让我们看到还有哪些开源软件可能会成为下一个 Apache Log4j2，也可以启发我们关于开源软件安全更进一步的思考。

(1) 主流开源生态关键基础开源软件 TOP50

关于开源软件重要性的度量，目前并没有被广泛认可的标准，OpenSSF 的“关键性评分 (Criticality Score)”是可参考的一种方

法,但其更像是对软件健康度的度量,不够直观。从安全的视角来看,如果开源软件出现漏洞后,造成的放大效应越大,影响的范围越广,那么这个软件就越重要,这样直接依赖数将是决定一个开源软件重要与否的关键性指标。本报告将使用直接依赖数作为开源软件重要性度量的唯一指标,并且基于经验参考将直接依赖数大于 1000 的开源软件定义为关键基础开源软件。

分析发现, Maven、NPM、Nuget、Pypi、Packagist、Rubygems 等主流开源生态中直接依赖数大于 1000 的开源软件共有 1068 款,直接依赖数排名 TOP50 的软件见下表。从表中可以看出,开源软件 junit:junit 的直接依赖数高达 95614,排名第一,大名鼎鼎的 Apache Log4j2 并没有出现在 TOP50 中,其直接依赖数为 7233,排在第 103 名。换句话说,如果 TOP50 表里的任何一款开源软件曝出严重漏洞,其影响可能都会大过 Apache Log4j2 的“Log4Shell”漏洞(CVE-2021-44228)。

排名	开源软件	所属包生态系统	直接依赖数
1	junit:junit	Maven	95614
2	rake	Rubygems	78524
3	bundler	Rubygems	68683
4	org.scala-lang:scala-library	Maven	68469
5	rspec	Rubygems	58262
6	org.slf4j:slf4j-api	Maven	49376
7	Newtonsoft.Json	Nuget	48964
8	tslib	NPM	42259
9	chalk	NPM	34576



10	commander	NPM	34494
11	lodash	NPM	33964
12	requests	Pypi	32640
13	numpy	Pypi	32498
14	illuminate/support	Packagist	29657
15	guzzlehttp/guzzle	Packagist	26706
16	com. google. guava: guava	Maven	26094
17	express	NPM	24863
18	request	NPM	24479
19	ch. qos. logback: logback-classic	Maven	23898
20	org. mockito: mockito-core	Maven	22417
21	inquirer	NPM	21485
22	react	NPM	20391
23	pandas	Pypi	20223
24	axios	NPM	20146
25	commons-io: commons-io	Maven	19481
26	com. fasterxml. jackson. core: jackson-databind	Maven	19229
27	org. apache. commons: commons-lang3	Maven	19084
28	fs-extra	NPM	18713
29	pytest	Pypi	17708
30	org. clojure: clojure	Maven	17304
31	pry	Rubygems	14197
32	matplotlib	Pypi	14015
33	log4j: log4j	Maven	13749
34	org. projectlombok: lombok	Maven	13735
35	laravel/framework	Packagist	13472
36	scipy	Pypi	13459
37	org. jetbrains. kotlin: kotlin-stdlib-common	Maven	13386



38	org.jetbrains.kotlin:kotlin-stdlib-jdk8	Maven	13349
39	org.slf4j:slf4j-log4j12	Maven	13294
40	yiisoft/yii2	Packagist	13094
41	typescript	NPM	12921
42	rails	Rubygems	12805
43	activesupport	Rubygems	12791
44	minitest	Rubygems	12624
45	react-dom	NPM	12584
46	simplecov	Rubygems	12520
47	moment	NPM	12460
48	org.mockito:mockito-all	Maven	12049
49	com.google.code.gson:gson	Maven	11821
50	org.assertj:assertj-core	Maven	11773

(2) 2/3 的关键基础开源软件从未公开披露过漏洞

公开披露产品漏洞信息，将漏洞进行编号，相关信息收录到主流漏洞库中，有助于漏洞的消控工作，也是目前软件产品厂商的常规做法，目前国际软件大厂在产品漏洞信息的公开披露方面可以说都做的比较优秀，但是对于开源软件来说，目前的情况参差不齐，既有漏洞披露流程很正规的优秀开源项目，也有很多没有漏洞披露流程的开源项目。分析发现，在上述的 1068 款关键基础开源软件中，有 711 款从未公开披露过漏洞，占比高达 66.6%。然而，从未公开披露过漏洞并不代表没有漏洞，对于关键基础开源软件来说，如果没有正规的漏洞披露流程，安全风险很大。

造成这种现象的原因主要有两个，一是有很多漏洞被修补但没有

公开披露，甚至没有被记录，这种情况在开源社区中比较常见，其隐藏的安全风险很大；二是这些开源软件还没有引起足够的关注，针对其的漏洞挖掘研究不多，还是“一片未开垦的荒地”，而对于漏洞挖掘来说，这种“荒地”可能意味着未来的“产粮大户”，随着软件供应链攻击越来越被攻击者青睐，这些关键基础开源软件将是未来防范的重点和难点。

(3) 关键基础开源软件的整体运维风险较高

从“版本更新时间”和“周提交频率”两个维度来分析，近一半的关键基础开源软件活跃度很低。其中，半年内没有发布过新版本的关键基础开源软件有 428 款，占比为 40.1%；一年内周平均提交次数小于 5 的关键基础开源软件数量为 503，占比为 47.1%；一年内周平均提交次数小于 1 的关键基础开源软件数量为 334，占比为 31.3%。

另外，在 TOP50 的关键基础开源软件中，只有 24 款软件明确已获得大厂或者基金会的支持，有 9 款软件的 Github 贡献者数量小于 100，而 npm 生态中的 fs-extra 则主要是由 JP Richardson 个人来维护的。

整体来看，关键基础开源软件的运维状况并不乐观，相关安全风险较大。

四、国内企业软件开发中开源软件应用状况

2021 年，奇安信代码安全实验室对 3354 个国内企业软件项目中

使用开源软件的情况进行了分析，这些软件项目的应用领域涉及政府、金融、能源等重要行业。本年度报告中，新增了开源软件许可协议风险分析以及漏洞的可利用性分析。从数据分析来看，国内企业使用开源软件时的安全风险没有得到改善，开源软件安全风险是当前企业软件开发中亟待解决的首要问题。

1、开源软件总体使用情况分析

(1) 平均每个软件项目使用 127 个开源软件

在被分析的 3354 个国内企业软件项目中，全部使用了开源软件，使用率为 100%。最多的项目使用了 4862 个开源软件，平均每个项目使用 127 个开源软件，平均值与去年报告的 126 相当。使用开源软件最多的 10 个项目情况如下表所示。

项目名称	使用的开源软件数量
项目 1	4862
项目 2	4220
项目 3	3878
项目 4	3838
项目 5	3832
项目 6	3778
项目 7	3536
项目 8	3205
项目 9	3062
项目 10	2997

(2) 流行开源软件被超过 1/3 的软件项目使用

在分析的 3354 个国内企业软件项目中，使用最多的开源软件为

Simple Logging Facade for Java(SLF4J)，被 1213 个项目所使用，占比达 36.2%，远超去年报告中最高的 24.3%。被使用最多的前 10 名开源软件如下表所示。

开源软件名称	使用的项目数	被使用率
Simple Logging Facade for Java (SLF4J)	1213	36.2%
Apache Commons Lang	1187	35.4%
Commons IO	1070	31.9%
log4j	1047	31.2%
Spring Framework	1009	30.1%
Apache Commons Codec	1005	30.0%
Guava: Google Core Libraries for Java	964	28.7%
Apache HttpComponents Client(aka Apache HttpClient)	960	28.6%
jackson-databind	959	28.6%
Jackson JSON processor	959	28.6%

2、开源软件漏洞风险分析

(1) 77%的软件项目存在容易利用的开源软件漏洞

分析发现，在 3354 个国内企业软件项目中，存在已知开源软件漏洞的项目有 2897 个，占比高达 86.4%；存在已知高危开源软件漏洞的项目有 2680 个，占比为 79.9%；存在已知超危开源软件漏洞的项目有 2400 个，占比为 71.6%。

本年度报告中还针对漏洞的利用难度进行了分析，综合漏洞的 POC/EXP 情况以及 CVSS 可利用性指标等因素，我们将漏洞的利用难度分为容易、一般、困难。容易利用的漏洞风险极高，需要被及时修

补。在分析的 3354 个项目中,存在容易利用的漏洞的项目有 2581 个,占比高达 77.0%。

(2) 平均每个软件项目存在 69 个已知开源软件漏洞

在 3354 个国内企业软件项目中,共检出 231368 个已知开源软件漏洞(涉及到 7269 个唯一 CVE 漏洞编号),平均每个软件项目存在 69 个已知开源软件漏洞,略高于去年报告中的 66 个,最多的软件项目存在 1555 个已知开源软件漏洞。存在已知开源软件漏洞数量排名前 10 的项目情况如下表所示。

项目	存在开源软件漏洞数量
项目 1	1555
项目 2	1368
项目 3	1214
项目 4	1209
项目 5	798
项目 6	771
项目 7	770
项目 8	762
项目 9	740
项目 10	718

(3) 影响最广的开源软件漏洞存在于超 3 成的软件项目中

从漏洞的影响度来分析,影响范围最大的开源软件漏洞为 CVE-2022-22965,存在于 31.7%的软件项目中。2021 年影响度排名前 10 的开源软件漏洞如下表所示。



漏洞名称	CVE 编号	影响项目数量	影响度
Spring Framework 远程代码执行漏洞	CVE-2022-22965	1063	31.7%
Vmware Spring Framework 安全漏洞	CVE-2022-22950	1009	30.1%
Vmware Spring Framework 安全特征问题漏洞	CVE-2022-22968	1009	30.1%
Vmware Spring Framework 代码问题漏洞	CVE-2016-1000027	1009	30.1%
FasterXML jackson-databind 缓冲区错误漏洞	CVE-2020-36518	946	28.2%
Apache Commons IO 路径遍历漏洞	CVE-2021-29425	893	26.6%
Google Guava 访问控制错误漏洞	CVE-2020-8908	878	26.2%
Apache Log4j 信任管理问题漏洞	CVE-2020-9488	859	25.6%
jQuery 跨站脚本漏洞	CVE-2020-11022	819	24.4%
jQuery 跨站脚本漏洞	CVE-2020-11023	819	24.4%

容易利用的漏洞更易被用来发起攻击，这类漏洞风险极高。影响度排名前 10 的容易利用的开源软件漏洞情况如下表所示。从表中我们可以看到，Apache Log4j2 的“Log4Shell”漏洞(CVE-2021-44228)排在容易利用的漏洞影响度第 10 名。

容易利用的漏洞名称	CVE 编号	影响项目数量	影响度
Spring Framework 远程代码执行漏洞	CVE-2022-22965	1063	31.7%
Vmware Spring Framework 代码问题漏洞	CVE-2016-1000027	1009	30.1%
jQuery 跨站脚本漏洞	CVE-2020-11022	819	24.4%
jQuery 跨站脚本漏洞	CVE-2020-11023	819	24.4%
Apache HttpClient 安全漏洞	CVE-2020-13956	796	23.7%
OpenSSL 缓冲区错误漏洞	CVE-2021-3711	774	23.1%
jQuery 跨站脚本漏洞	CVE-2019-11358	765	22.8%
FasterXML jackson-databind 代码问题漏洞	CVE-2020-8840	763	22.7%
jQuery 跨站脚本漏洞	CVE-2015-9251	689	20.5%

Apache Log4j 代码问题漏洞	CVE-2021-44228	666	19.9%
---------------------	----------------	-----	-------

(4) 16 年前的开源软件漏洞仍然存在于多个软件项目中

分析发现，与 2020 年结果一样，部分软件项目中仍然存在十几年前公开的古老开源软件漏洞，最古老的漏洞是 2005 年 8 月公开的 CVE-2005-2541，仍然存在于 13 个项目中。部分古老开源软件漏洞的影响情况如下表所示。

漏洞名称	CVE 编号	发布日期	影响项目数量
GNU Tar 安全漏洞	CVE-2005-2541	2005.08.10	13
Apache Tomcat 目录列表拒绝服务漏洞	CVE-2005-3510	2005.11.06	29
Apache Struts 错误响应跨站脚本漏洞	CVE-2005-3745	2005.11.22	1
Apache Tomcat 跨站脚本攻击漏洞	CVE-2005-4838	2005.12.31	30
Apache Software Foundation (ASF) Struts \org.apache.struts.taglib.html.Constants.CANCEL\ 参数安全绕过漏洞	CVE-2006-1546	2006.03.30	17
Apache Software Foundation (ASF) Struts ActionForm 拒绝服务漏洞	CVE-2006-1547	2006.03.30	17
Apache Struts 多个远程漏洞	CVE-2006-1548	2006.03.30	17
GnuPG parse-packet.c 远程缓冲区错误漏洞	CVE-2006-3082	2006.06.19	5
Apache Tomcat 远程目录 \index.jsp\ 信息泄露漏洞	CVE-2006-3835	2006.07.25	9
Direct Web Rendering 安全绕过漏洞	CVE-2007-0184	2007.01.12	1

3、开源软件许可协议风险分析

开源许可协议是开源社区为了维护开源软件作者和贡献者的合

法权利，而定义的一组条款和条件，其他人必须在遵循这些条款和条件的基础上，使用、修改或共享开源软件的源代码、设计和文档。有些开源许可协议是不允许将开源软件修改后做为闭源的商业软件发布和销售的，如 AGPL、GPL 等。企业如未按许可协议的要求使用开源软件，可能会带来知识产权方面的风险。

(1) 最流行的开源许可协议在超 3/4 的项目中使用

在 3354 个被分析的国内企业软件项目中，共发现 488 个开源许可协议。涉及项目数最多的协议是 Apache License 2.0，在 2586 个项目使用，占比为 77.1%。涉及软件项目数排名前 10 的许可协议如下表所示。

开源许可协议	涉及项目数	所占比例
Apache License 2.0	2586	77.1%
MIT License	2428	72.4%
BSD 3-Clause "New" or "Revised" License	1716	51.2%
Eclipse Public License 1.0	922	27.5%
BSD 2-Clause "Simplified" License	764	22.8%
GNU General Public License v2.0 or later	690	20.6%
BSD 4-Clause "Original" or "Old" License	660	19.7%
ISC License	556	16.6%
GNU Lesser General Public License v2.1 only	472	14.1%
Mozilla Public License 1.1	426	12.7%

(2) 45.6%的项目使用了含有高风险许可协议的开源软件

在 3354 个被分析的国内企业软件项目中，存在高风险许可协议

的项目 1530 个，占比 45.6%。所谓高风险许可协议，是指这类许可协议中包含了一些限制性较强的条款，在使用过程中一旦违反这些条款，可能对企业的商业利益和声誉造成严重的损害。这些限制性条款主要包括：“禁止本软件及其衍生品用于商业目的”，“禁止在未支付费用和版税的情况下将该软件用于非公开、非商业用途”，“软件发布时必须公开软件的源代码”等。涉及软件项目数排名前 10 的高风险许可协议如下表所示。

开源许可协议	涉及项目数	所占比例
Eclipse Public License 1.0	919	27.4%
Mozilla Public License 1.1	425	12.7%
GNU General Public License v2.0 or later	248	7.4%
GNU Lesser General Public License v2.1 only	231	6.9%
GNU Affero General Public License v3.0	173	5.2%
GNU General Public License v2.0 only	123	3.7%
GNU General Public License v3.0 only	85	2.5%
GNU Library General Public License v2.1 or later	74	2.2%
GNU General Public License v3.0 or later	62	1.8%
GNU Library General Public License v2 only	56	1.7%

4、开源软件运维风险分析

开源软件运维风险复杂多样，报告主要从老旧开源软件的使用和开源软件多版本的使用角度进行分析。

（1）20 年前的老旧开源软件版本仍在被使用

分析发现，与去年报告数据一致，许多软件项目中仍然在使用十

几年甚至 20 年前发布的开源软件版本，存在很大的运维风险。被使用的老旧开源软件版本中，最老旧的一个是 2002 年 5 月 15 日发布的 JDOM 1.0-FCS，已经有 20 年之久，但仍然被 28 个软件项目所使用。按老旧程度排名前 10 的开源软件如下表所示。

开源软件名称	版本号	版本发布日期	使用它的项目数量
JDOM	1.0-FCS	2002.05.15	28
Apache Xalan	2.5.D1	2003.03.03	1
XML Pull Parsing API	1.1.3.1	2003.06.17	228
JUnit	3.8.1	2004.03.05	100
Log4j	1.2.8	2004.03.05	52
SSLExt	1.2-0	2004.10.04	26
jaxen	1.0-FCS	2005.04.27	13
Mockobjects Core	0.09	2005.04.27	3
Mockobjects Jdk1 4 J2ee1 3	0.09	2005.04.27	3
JLine	0.9.1	2005.05.18	3

(2) 开源软件各版本使用更加混乱

分析发现，各个项目中开源软件使用的版本非常混乱，并非使用的都是最新版本。Spring Data 是被使用版本最多的开源软件，有 235 个版本在被使用，比去年报告中最多的 162 高出很多，说明开源软件版本使用混乱的状况更加严重。按照被使用版本的数量排序，排名前 10 的开源软件情况如下表所示。

开源软件名称	被使用的版本数量
Spring Data	235
Spring Framework	226

Apache Tomcat	206
@types/node	186
jackson-databind	170
electron-to-chromium	168
Hibernate ORM	162
Jetty	158
Spring TestContext Framework	155
Spring Boot	145

五、典型软件供应链安全风险实例分析

1、某主流网络接入存储 (NAS) 设备供应链攻击实例分析

网络接入存储 NAS 基于标准网络协议实现数据传输，为网络中的 Windows/Linux/Mac OS 等各种不同操作系统的计算机提供文件共享和数据备份，在云存储、数据容灾、Web 服务器等中广泛使用。本实例中我们分析的 NAS 系统在市场占有率方面具有领先优势，且预装了磁盘站管理系统。

经分析发现，该磁盘站管理系统中使用了 SQLite、libssh2、Netatalk 等在内的超过 200 款开源软件，并由此引入了 1000 余个已知开源软件漏洞，其中包括超危漏洞 195 个、高危漏洞 461 个。该磁盘站管理系统使用的部分开源软件及漏洞情况如下表所示。

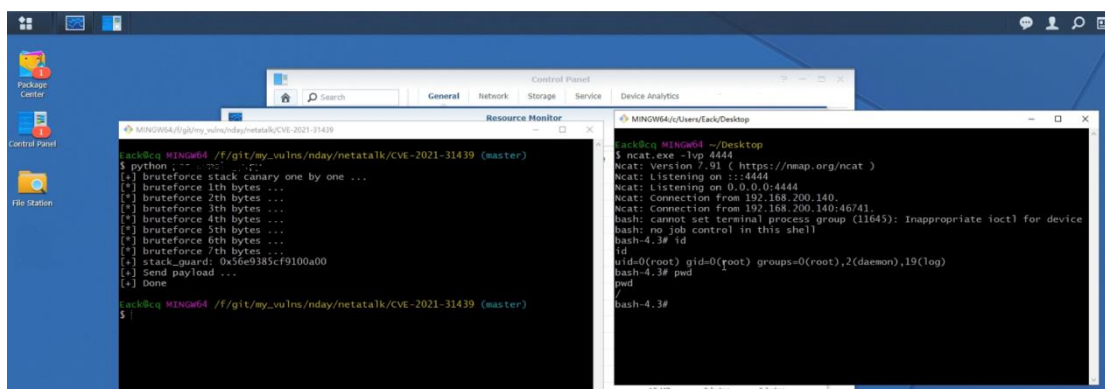
序号	开源软件名称	开源软件版本	漏洞情况
1	SQLite	3.10.2	超危:5, 高危:6, 中危:7
2	libssh2	1.7.0	超危:5, 高危:6
3	libpng	1.6.28	超危:2, 高危:3, 中危:9
4	zlib	1.2.8	超危:2, 高危:3

5	OpenSSL	1.0.2n	超危:1, 高危:4, 中危:12, 低危:3
6	Netatalk	3.1.8	超危:1, 高危:1
7	bzip2	1.0.6	超危:1, 中危:1
8	libTIFF	4.0.8	高危:6, 中危:15
9	c-ares	1.12.0	高危:2, 中危:1
10	pip	8.1.1	高危:2, 中危:1

这其中，Netatalk 是一款开源的 Apple Filing Protocol (AFP) 服务程序，Linux 或 BSD 等系统可使用它提供 Mac 文件服务器文件共享功能，被应用于多家主流 NAS 厂商的设备中。

CVE-2021-31439 是 Netatalk 的一个高危历史漏洞(上表中所示)，Netatalk 组件在处理 DSI 数据包时存在堆溢出问题，未认证的用户通过发送精心构造的数据包，可实现任意代码执行，从而获取设备的控制权。该漏洞影响 Netatalk 组件当前所有版本。

进一步分析验证发现，利用该漏洞，可成功攻击磁盘站管理系统所属的 NAS 设备，如下图所示，成功攻击后可获取反弹 Shell，且对应的权限为 root。



本实例中，软件供应链风险传播链条如下：Netatalk 组件被 NAS 设备的磁盘站管理系统所使用，且该服务默认开启，因此 Netatalk 组件的漏洞影响了该 NAS 设备，导致可针对其成功实施软件供应链攻击。

2、某主流 VPN 路由器供应链攻击实例分析

某国际 TOP 厂商主流千兆 VPN 路由器，广泛应用于各类企业中。对其固件进行分析发现，固件中使用了 tcpdump、mutt、lldpd 等在内的超过 90 款开源软件，并由此引入了 500 余个已知开源软件漏洞，其中包括超危漏洞 162 个、高危漏洞 214 个。该固件中使用的部分开源软件及漏洞情况如下表所示。

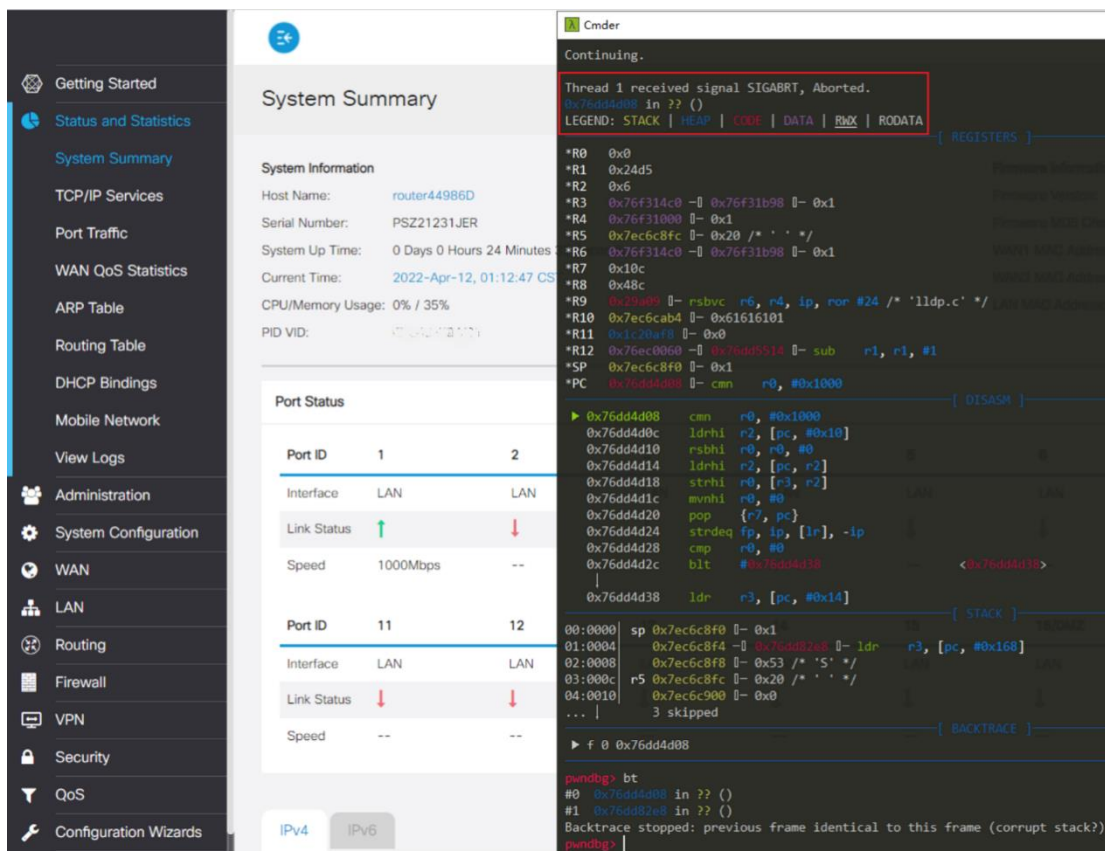
序号	开源软件名称	版本	漏洞情况
1	tcpdump	4.9.2	超危:85, 高危:30, 中危:2, 低危:1
2	mutt	1.5.16	超危:11, 中危:10
3	SQLite	3.21.0	超危:3, 高危:6, 中危:6
4	PCRE	8.35	超危:2, 高危:21, 中危:7
5	zlib	1.2.8	超危:2, 高危:3
6	openldap - LDAP support libraries	2.4.23	超危:1, 高危:17, 中危:13, 低危:1
7	OpenSSL	1.0.2o	超危:1, 高危:4, 中危:11, 低危:3
8	lldpd	0.6.0	超危:1, 高危:3
9	bzip2	1.0.6	超危:1, 中危:1
10	GNU Binutils	2.24	高危:10, 中危:24, 低危:1

这其中，开源软件 lldpd 是 LLDP 行业标准（旨在取代 EDP、CDP 之类的专有链路层协议）的 ISC 许可实现，适用于各种 Unix 系统，并被广泛应用于主流厂商的网络设备（AP、交换机、路由器和摄像头等）中。

CVE-2015-8011 是 lldpd 的一个超危历史漏洞（上表中所示），攻击者可通过发送精心构造的数据包，使得在处理 lldp 数据包中的 Management Address TLV 时发生缓冲区溢出，从而引发代码执行。部分厂商虽然采用了防御性编程，但程序仍会崩溃，造成拒绝服务危害。

值得注意的是，尽管该漏洞是 7 年前被发现的，且 lldpd 官方仓库也发布了相应的更新，但它至今依然存在于部分厂商的网络设备中。

进一步分析验证发现，利用该漏洞可成功攻击此案例中的 VPN 路由器，因设备采用了防御性编程，漏洞可造成拒绝服务的危害。如下图所示，gdb 调试器捕获到了 SIGABRT 信号，同时对应的进程退出，攻击成功。



The screenshot displays a network device's web management interface on the left and a GDB debugger window on the right. The web interface shows system information (Host Name: router44986D, Serial Number: PSZ21231JER, System Up Time: 0 Days 0 Hours 24 Minutes, Current Time: 2022-Apr-12, 01:12:47 CS, CPU/Memory Usage: 0% / 35%, PID VID: 0x76dd4d08) and port status (Port ID 1, 2, 11, 12). The GDB window shows a SIGABRT signal and a backtrace.

System Summary

System Information

Host Name: router44986D
 Serial Number: PSZ21231JER
 System Up Time: 0 Days 0 Hours 24 Minutes
 Current Time: 2022-Apr-12, 01:12:47 CS
 CPU/Memory Usage: 0% / 35%
 PID VID: 0x76dd4d08

Port Status

Port ID	1	2
Interface	LAN	LAN
Link Status	↑	↓
Speed	1000Mbps	--

Port ID 11, 12

Port ID	11	12
Interface	LAN	LAN
Link Status	↓	↓
Speed	--	--

GDB Window

```

Continuing.
Thread 1 received signal SIGABRT, Aborted.
0x76dd4d08 in ?? ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA

[R0] 0x0
[R1] 0x24d5
[R2] 0x6
[R3] 0x76f314c0 - 0x76f31b98 - 0x1
[R4] 0x76f31000 - 0x1
[R5] 0x7ec6c8fc - 0x20 /* ' ' */
[R6] 0x76f314c0 - 0x76f31b98 - 0x1
[R7] 0x10c
[R8] 0x48c
[R9] 0x24d5 - rsbvc r6, r4, ip, nor #24 /* 'lldp.c' */
[R10] 0x7ec6cab4 - 0x61616101
[R11] 0x1c20af8 - 0x0
[R12] 0x7ec6c800 - 0x76dd5514 - sub r1, r1, #1
[SP] 0x7ec6c8f0 - 0x1
[PC] 0x76dd4d08 - cmn r0, #0x1000

0x76dd4d08 cmn r0, #0x1000
0x76dd4d0c ldrhi r2, [pc, #0x10]
0x76dd4d10 rsbhi r0, r0, #0
0x76dd4d14 ldrhi r2, [pc, r2]
0x76dd4d18 strhi r0, [r3, r2]
0x76dd4d1c mvnhi r0, #0
0x76dd4d20 pop {r7, pc}
0x76dd4d24 strdeq fp, ip, [lr], -ip
0x76dd4d28 cmp r0, #0
0x76dd4d2c blt #0x76dd4d38 <0x76dd4d38>
0x76dd4d38 ldr r3, [pc, #0x14]

00:0000 sp 0x7ec6c8f0 - 0x1
01:0004 0x7ec6c8f4 - 0x76dd22e8 - ldr r3, [pc, #0x168]
02:0008 0x7ec6c8f8 - 0x53 /* 'S' */
03:000c r5 0x7ec6c8fc - 0x20 /* ' ' */
04:0010 0x7ec6c900 - 0x0
... | 3 skipped

Backtrace stopped: previous frame identical to this frame (corrupt stack?)
  
```

```

/tmp # ps -aux | grep lldp
9423 root      2152 S      /usr/sbin/lldpd -I eth3*
9429 root      10124 S     /usr/sbin/lldpd -I eth3*
17150 root      3124 S      grep lldp
/tmp # ./gdbserver :12345 --attach 9429 &
Detaching from process 9429
Listening on port 12345
[1]+  Done                  ./gdbserver :12345 --attach 9429
/tmp #
/tmp # ps -aux | grep lldp
17785 root      3120 R      grep lldp
  
```

本实例中，软件供应链风险传播链条如下：开源组件 lldpd 被该

该款主流 VPN 路由器中的固件所使用，进而 lldpd 组件的漏洞影响该路由器，即针对路由器成功实施了软件供应链攻击。

3、三款国产操作系统供应链攻击实例分析

Polkit 是一个开源应用程序框架，通过定义和审核权限规则，实现不同优先级进程间的通讯，使得控制决策集中在统一的框架之中，决定低优先级进程是否有权访问高优先级进程。分析发现，开源软件 Polkit 广泛应用于基于 Linux 内核的国产操作系统中，其漏洞可能引发针对这些操作系统的软件供应链攻击。验证发现，至少 3 款主流国产操作系统会受到 Polkit 历史漏洞的影响。

以某主流国产操作系统为例，它基于 Linux 内核，是一款通用的桌面电脑操作系统，并加入了大量的本地优化功能。

经分析发现，该国产操作系统中使用了 libssh2、libpng、PolKit 等在内的超过 1000 款开源软件，并由此引入了 1000 余个已知开源软件漏洞，其中包括超危漏洞 267 个、高危漏洞 599 个。该操作系统中使用的部分开源软件及漏洞情况如下表所示。

序号	开源软件名称	开源软件版本	漏洞情况
1	libssh2	1.4.3	超危:5, 高危:6
2	libpng	1.6.28	超危:2, 高危:3, 中危:9
3	libproxy	0.4.15	超危:1, 高危:1
4	OpenJPEG	2.3.1	高危:6, 中危:7
5	PolKit	0.105	高危:4, 中危:6, 低危:1
6	urllib3	1.25.8	高危:1, 中危:1
7	python-rsa	4.0	高危:1, 中危:1
8	httplib2	0.14.0	高危:1, 中危:1

9	ReportLab	3.5.34	中危:1
10	paramiko	2.6.0	中危:1

CVE-2021-4034 是 Polkit 的 pkexec 组件中的一个高危历史漏洞（上表中所示），利用此漏洞可在没有参数的情况下越界读写参数，从而提升普通用户的权限，实现本地权限提升攻击，危害极大。

进一步分析验证发现，利用 CVE-2021-4034 可对该国产操作系统进行攻击，如下图所示，漏洞被成功利用后完成了从普通用户“aa”到超级用户“root”的切换，实现了权限提升。

```
aa@aa-virtual-machine:~/CVE-2021-4034$ cat /etc/os-release
NAME="Ubuntu"
VERSION="20.04.1 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04.1 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
aa@aa-virtual-machine:~/CVE-2021-4034$ uname -a
Linux aa-virtual-machine 5.11.0-25-generic #27~20.04.1-Ubuntu SMP Tue Jul 13 17:41:23 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
aa@aa-virtual-machine:~/CVE-2021-4034$ whoami
aa
aa@aa-virtual-machine:~/CVE-2021-4034$ cat /etc/shadow
cat: /etc/shadow: 权限不够
aa@aa-virtual-machine:~/CVE-2021-4034$ ./cve-2021-4034
# whoami
root
# cat /etc/shadow |more
root:!:18978:0:99999:7:::
daemon:!:18978:0:99999:7:::
bin:!:18978:0:99999:7:::
sys:!:18978:0:99999:7:::
sync:!:18978:0:99999:7:::
games:!:18978:0:99999:7:::
man:!:18978:0:99999:7:::
lp:!:18978:0:99999:7:::
mail:!:18978:0:99999:7:::
news:!:18978:0:99999:7:::
uucp:!:18978:0:99999:7:::
proxy:!:18978:0:99999:7:::
www-data:!:18978:0:99999:7:::
backup:!:18978:0:99999:7:::
list:!:18978:0:99999:7:::
irc:!:18978:0:99999:7:::
gnats:!:18978:0:99999:7:::
nobody:!:18978:0:99999:7:::
_apt:!:18978:0:99999:7:::
systemd-timesync:!:18978:0:99999:7:::
systemd-network:!:18978:0:99999:7:::
systemd-resolve:!:18978:0:99999:7:::
tss:!:18978:0:99999:7:::
```

本实例中，软件供应链风险传播链条如下：开源软件 Polkit 被国产操作系统所使用，Polkit 中 pkexec 组件的漏洞影响了这些操作系统，导致可针对其实施权限提升方面的软件供应链攻击。

4、某国产邮件系统供应链攻击实例分析

某国产邮件系统是一款通用邮件管理系统，在国内拥有大量的用户，用户量达数亿级。

经分析发现，该国产邮件系统中使用了 jackson-databind、Spring Framework、Apache Tomcat 等在内的超过 100 款开源软件，并由此引入了 800 余个已知开源软件漏洞，其中包括超危漏洞 56 个、高危漏洞 151 个。该邮件系统中使用的部分开源软件及漏洞情况如下表所示。


序号	开源软件名称	开源软件版本	漏洞情况
1	jackson-databind	2.7.4	超危:25, 高危:38, 中危:4
2	Jetty: Java based HTTP/1.x, HTTP/2, Servlet, WebSocket Server	6.1.14	超危:3, 高危:5, 中危:10, 低危:1
3	Spring Framework	5.1.5.RELEASE	超危:2, 高危:2, 中危:6
4	Apache Tomcat	9.0.27	超危:1, 高危:15, 中危:5
5	MySQL Connector/J	5.1.44	超危:1, 高危:4, 中危:6, 低危:2
6	CXF	3.3.0	超危:1, 高危:3, 中危:4
7	nginx	1.14.2	超危:1, 高危:3, 中危:3
8	Protocol Buffers	2.6.1	高危:2, 中危:1
9	Apache Portable Runtime	1.5.1	高危:2
10	Apache Ant	1.10.5	高危:1, 中危:3

其中，Tomcat 是全世界最著名的基于 Java 语言的轻量级应用服务器，是一款完全开源免费的 Servlet 容器实现，是 Apache 软件基金会（Apache Software Foundation）的 Jakarta 项目中的一个核心项目。Tomcat 由于技术先进、性能稳定、而且免费，成为了流行的

Web 应用服务器。

CVE-2020-1938 是 Tomcat 的 Tomcat AJP Connector 模块中的一个高危历史漏洞（上表中所示），可以读取 Tomcat 上 webapps 目录下的任意文件，如 webapps 配置文件或源代码；或者可以通过配合文件上传漏洞实现 webapps 目录下本地文件的包含，达到命令执行的目的，危害极大。

进一步分析验证发现，利用 CVE-2020-1938 可对该国产邮件系统进行攻击，如下图所示，漏洞被成功利用后完成了下载其中某个 JSP 文件的效果。



```
# Kevin @ KevinMacBook-Pro in ~/work/learn/CVE-2020-1938-tomcat-file_include-file_read on git:mas
$ python Tomcat-ROOT路径下文件读取\ (CVE-2020-1938\).py 192.168.161.138 -p 9909 -f wap.jsp
Getting resource at ajp13://192.168.161.138:9909/asdf
-----
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//OPENWAVE.COM//DTD WML 1.3//EN" "http://www.openwave.com/dtd/wml13.dtd">
<%@ page session="false" contentType="text/vnd.wap.wml" %>
<%
    response.sendRedirect("/asfdf/asfdf/");
%>
<wml>
  <card title="Welcome to My Site">
    <p align="center">
      <anchor title="Link1"><go href="/asfdf/asfdf/">Go</anchor><br/>
    </p>
  </card>
</wml>
```

本实例中，软件供应链风险传播链条如下：开源软件 Tomcat 被某国产邮件系统所使用，Tomcat 中 Tomcat AJP Connector 模块的漏洞影响了该邮件系统，导致可针对其实施文件下载的软件供应链攻击。

5、Edge 浏览器供应链攻击实例分析

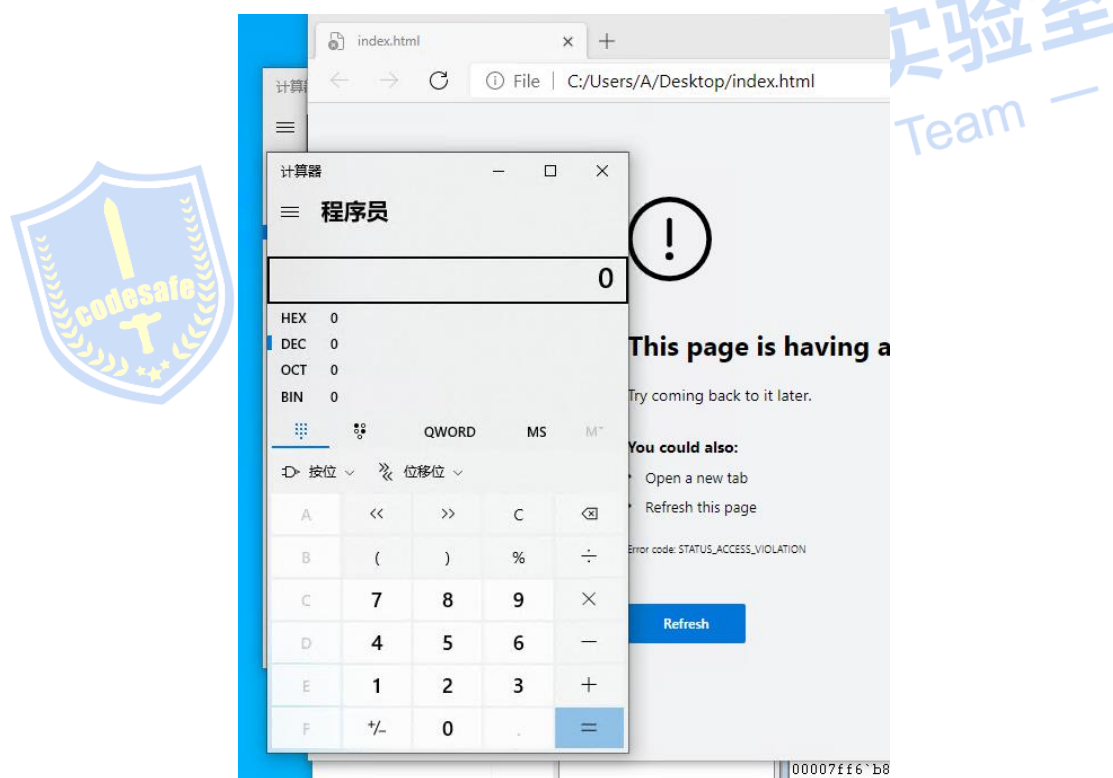
Chromium 内核是 Google 发布的免费开源项目，在许多主流浏览器中都有使用，如微软 Edge 浏览器、Google Chrome 浏览器、Opera 浏览器等。

分析发现，Chromium 内核中使用的开源软件超过 230 款，包括

V8、libpng、libxslt、zlib、SQLite 等，其中 V8 是一款 C++ 语言编写的高性能 JavaScript 和 WebAssembly 开源引擎，用来高效稳定的解析执行 JavaScript 脚本。

CVE-2021-21224 是 V8 的一个高危历史漏洞，可导致远程代码执行。具体而言，V8 的 JIT 模块存在类型混淆问题，攻击者可通过构造 JavaScript 脚本，实现内存的越界读写，造成远程代码执行的危害。

进一步分析验证发现，攻击者可以利用 V8 的 CVE-2021-21224 漏洞，通过“网页钓鱼攻击”的方式在网页中隐藏漏洞利用代码，从而对 Edge 浏览器 (90.0.818.41 之前版本) 进行攻击，如下图所示。



本实例中的软件供应链风险传播链条如下：Edge 浏览器中使用了 Chromium 内核，Chromium 内核又包含开源引擎 V8，因此 V8 引擎的漏洞影响了 Edge 浏览器，导致可对其成功实施软件供应链攻击。

六、总结及建议

过去的一年里，我们看到软件供应链安全的一系列国家标准、行业标准正在制定中，软件供应链安全相关的科研课题和建设项目逐渐增多，部分重要行业监管部门还出台了关于软件供应链安全的具体工作要求，面向软件供应链安全的社区、联盟、论坛等也在逐渐建立和完善中，软件供应链安全的重要性已经逐步成为各方共识。

但另一方面我们也看到，国内大多数机构和企业目前对于软件供应链安全还处于了解和保持关注阶段，尚未付诸真正的行动。从本报告中国内企业软件开发项目的实测数据来看，软件供应链安全相关风险很高，形势严峻且紧迫。鉴于这一现状，我们呼吁：从现在开始就行动起来，不要让软件供应链安全成“灰犀牛”。

软件组成成分的透明性是软件供应链安全保障的基础，我们建议将软件物料清单（SBOM）作为软件供应链安全的抓手首先推进落地，通过软件物料清单（SBOM）的推广应用，牵引软件供应链上下游各个环节的协同，在此基础之上再采取更多举措逐步深化，实现软件供应链安全保障的目标。关于软件物料清单（SBOM）的推进落地，具体建议如下：

1) 从国家与行业监管层面的建议

提高关键基础设施和重要信息系统用户软件供应链安全保障的要求，要求向关键基础设施和重要信息系统用户销售软件产品、提供软件定制开发的厂商和供应商，在交付软件系统的同时，需提供软件

物料清单(SBOM),以保持足够的透明性;组织制定软件物料清单(SBOM)相关的国家标准、行业标准,规范针对软件物料清单(SBOM)的格式和内容要求,促进软件物料清单(SBOM)成为软件产品的标配;建立相应的公共服务平台,提供软件物料清单(SBOM)的检测验证、数据查询、威胁情报等服务。

2) 从软件厂商和供应商层面的建议

在自身软件开发流程中采用开源安全治理工具,持续监测软件开发中所使用的开源软件物料,消减其安全风险;产品正式发布时,应提取和生成产品的软件物料清单(SBOM),并随软件向客户提供;针对自身软件产品中所使用的软件物料,持续监测其安全漏洞等风险情况,并及时为客户提供相应的技术支持服务。

3) 从软件最终用户层面的建议

保持对软件物料透明性的高度关注,在购买软件产品或委托定制开发软件系统时,要求厂商和供应商提软件物料清单(SBOM),并与其签订安全责任协议,要求其对软件物料的安全性负责并提供后续的技术支持服务;对于运行重要业务的软件系统,应使用合适的检测工具验证厂商和供应商所提供的软件物料清单(SBOM)的正确性,确认软件的组成成分及安全风险状况;在软件系统日常运行过程中,应基于软件物料清单(SBOM)持续跟踪软件物料相关的威胁情报,及时采取安全措施,消减相关安全风险。

附录：奇安信代码安全实验室简介

奇安信代码安全实验室是奇安信集团旗下，专注于软件源代码安全分析技术、二进制漏洞挖掘技术与开发的团队。实验室支撑国家级漏洞平台的技术工作，多次向国家信息安全漏洞库（CNNVD）和国家信息安全漏洞共享平台（CNVD）报送原创通用型漏洞信息并获得表彰；帮助微软、谷歌、苹果、Cisco、Juniper、Red Hat、Ubuntu、Oracle、Adobe、VMware、阿里云、飞塔、华为、施耐德、Mikrotik、Netgear、D-Link、Netis、ThinkPHP、以太坊、Facebook、亚马逊、IBM、SAP、NetFlix、Kubernetes、Apache 基金会、腾讯、滴滴等大型厂商和机构的商用产品或开源项目发现了数百个安全缺陷和漏洞，并获得公开致谢。目前，实验室拥有国家信息安全漏洞库（CNNVD）特聘专家一名，多名成员入选微软全球 TOP 安全研究者、Oracle 安全纵深防御计划贡献者等精英榜单。在 Pwn2Own 2017 世界黑客大赛上，实验室成员还曾获得 Master of Pwn 破解大师冠军称号。

基于奇安信代码安全实验室多年的技术积累，奇安信集团在国内率先推出了自主可控的软件代码安全分析系统——奇安信代码卫士和奇安信开源卫士。奇安信代码卫士是一套静态应用程序安全测试系统，可检测 2600 多种源代码安全缺陷，支持 C、C++、C#、Objective-C、Swift、Java、JavaScript、PHP、Python、Cobol、Go 等 20 多种编程语言。奇安信开源卫士是一套集开源软件识别与安全管控于一体的软件成分风险分析系统，通过智能化数据收集引擎在全球范围内广

泛获取开源软件信息和漏洞信息，帮助客户掌握开源软件资产状况，及时获取开源软件漏洞情报，降低由开源软件带来的安全风险，奇安信开源卫士目前可识别 9000 多万个开源软件版本，兼容 NVD、CNNVD、CNVD 等多个漏洞库。奇安信代码卫士和奇安信开源卫士目前已经在数百家大型企业和机构中应用，帮助客户构建自身的代码安全保障体系，消减软件代码安全隐患，并入选国家发改委数字化转型伙伴行动、工信部中小企业数字化赋能专项行动，为中小企业提供软件代码安全检测平台和服务。

