公 开

2015 年第三季度开源软件源代码
安全漏洞分析报告

国家互联网应急中心

实验室

2015 年 10 月

目录

1 概述

近年来，随着软件技术飞速发展，开源软件已在全球范围内得到了广泛应用。数据显示，从 2012 年起，已有超过 80%的商业软件使用开源软件。开源软件的代码一旦存在安全问题，必将造成广泛、严重的影响。为了解当前开源软件的安全情况，实验室本季度对 28 款广泛使用的知名开源软件进行了源代码安全测试。结合漏洞扫描工具和人工审计的结果 ,形成了本漏洞分析报告。本次测试在代码层面共发现高危安全漏洞 3133 个。与上季度的结果相比，这些开源软件存在的安全问题依然严重。

2 被测开源软件

表 1 列出了本次被测的 28 个开源软件项目的概况，涵盖了 java，php，C#三种编程语言。这些软件项目都是国际、国内知名的，拥有广泛用户的软件项目，其中不乏由知名软件公司开发的软件。由于这些软件大多具有巨大的用户群体，软件中的安全漏洞很可能会造成严重的后果。

表 1 被测开源软件项目概览

项目名称	版本号	主要编程语言	功能说明	代码行数(L)
RoadFlow	2.0	C#	集成工作流引擎的 ASP.NET MVC 快速开发平台	301,859
G4Studio	4.0.2	java	JavaEE 开源快速开发平台	61,213
Magnolia	5.4.2	java	基于 Java 的开源内容管理系统	6,223
Springside	4.2.3	Java web	SpringSide 以 Spring Framework 为核心，提供一个 Pragmatic 的企业应用 KickStart 与 Full-Stack 的开源构件库	22,201
哎嘛	1.30	java	基于 GPL 授权协议的 OSC 开源中国第三方客户端	85,825

Flight	1.2.13	php	Flight 是个快速,简单,可扩展的 PHP 框架，允许用户快速简单的创建 RESTful web 应用	1624
Laravel	4.2	php	开源框架	8,399
Yii	1.1.16	php	开源框架	700,942
CodeIgniter	3.0.0	php	开源框架	104,632
Kohana	3.3.4	php	开源框架	1,019
Hadoop	2.7.1	java	分布式系统基础架构	1,611,332
Tengine	2.1.1	C#	Web 开源服务器	166,011
JFinal OA	2.0	java	JFinal 和 dwz 开发的基于中小企业的 OA 系统	20,488
IKAnalyzer	3.0	java	是一个开源的,基于 java 语言开发的轻量级的中文分词工具包	17,327
JFlow	2.3.4	java	工作流程引擎开源软件	514,993
EntireJ	2.0	java	EntireJ 是一个 RAD 快速应用开发环境	33,674
ThinkAndroid	1.0.0	java	ThinkAndroid 是一个免费的开源的、简易的、遵循 Apache2 开源协议发布的 Android 开发框架	17,575
Android Annotations	3.3.2	java	Android Annotations 是一个开源的框架，用于加速 Android 应用的开发	35,111
xUtils	2.16.4	java	Android 工具包	12,937
Android Volley	3.0.0	java	基于 Android Volley 的网络请求工具。	3,748
LoonAndroid		java	Android 开发框架 LoonAndroid	8,399
Wicket	6.24.0	java	Wicket 是一个基于 Java 的 Web 开发框架	207,274
android-async-http	1.4.9	java	android-async-http 是 Android 上的一个异步 HTTP 客户端开发包。	7,617
Tapestry	5.40	java	Tapestry 是一个开源的基于 servlet 的应用程序框架，它使用组件对象模型来创建动态的，交互的 web 应用	226,811

cakephp	3.0	php	Php 开源框架	45,879
Symfony	3.8	Php	Php 开源框架	216,016
Zend Framework	1.12.16	Php	Php 开源框架	1,225,390
Zend Framework	2.4.8	php	Php 开源框架	163,307

3 测试内容

3.1 安全漏洞种类

本次测试涵盖各类常见安全漏洞。根据安全漏洞形成的原因、被利用的可能性、造成的危害程度和解决的难度等因素进行综合考虑，可以将常见的安全漏洞分为八类：

1) 输入验证与表示（Input Validation and Representation）

   输入验证与表示问题通常是由特殊字符、编码和数字表示所引起的，这类问题的发生是由于对输入的信任所造成的。这些问题包括 :缓冲区溢出、跨站脚本、SQL 注入、命令注入等。

2) API 误用（API Abuse)

   API 是调用者与被调用者之间的一个约定，大多数的 API 误用是由于调用者没有理解约定的目的所造成的。当使用 API 不当时，也会引发安全问题。

3) 安全特性（Security Features）

   该类别主要包含认证、访问控制、机密性、密码使用和特权管理等方面的漏洞。

4) 时间和状态（Time and State）

分布式计算与时间和状态有关。线程和进程之间的交互及执行任务的时间顺序往往由共享的状态决定，如信号量、变量、文件系统等。与分布式计算相关的漏洞包括竞态条件、阻塞误用等。

5) 错误和异常处理缺陷（Errors）

这类漏洞与错误和异常处理有关，最常见的一种漏洞是没有恰当的处理错误（或者没有处理错误）从而导致程序运行意外终止，另一种漏洞是产生的错误给潜在的攻击者提供了过多信息。

6) 代码质量问题（Code Quality）

低劣的代码质量会导致不可预测的行为。对于攻击者而言，低劣的代码使他们可以以意想不到的方式威胁系统。常见的该类别漏洞包括死代码、空指针解引用、资源泄漏等。

7) 封装和隐藏缺陷（Encapsulation）

合理的封装意味着区分校验过和未经检验的数据，区分不同用户的数据，或区分用户能看到和不能看到的数据等。常见的漏洞包括隐藏域、信息泄漏、跨站请求伪造等。

8) 代码运行环境的缺陷（Environment）

该类漏洞是源代码之外的问题，例如运行环境配置问题、敏感信息管理问题等，它们对产品的安全仍然是至关重要的。

前七类漏洞与源代码中的安全缺陷相关，它们可以成为恶意攻击的目标，一旦被利用会造成信息泄露、权限提升等严重后果。最后一类漏洞描述实际代码之外的安全问题，它们容易造成软件的运行异常、数据丢失等严重问题。

3.2 安全漏洞级别

我们将源代码的安全问题分为四种级别：极高危（Critical）、高危（High）、中等（Medium）和低（Low）。衡量级别的标准包括两个维度，可信程度（confidence）和严重程度（severity）。可信程度是指发现的问题是否准确的可能性，比如将每个strcpy()调用都标记成缓冲区溢出漏洞的可信程度很低。严重程度是指假设测试技术真实可信的情况下检出问题的严重性，比如缓冲区溢出（buffer overflow）通常是比空指针引用（null pointer dereference）更严重的安全问题。将这两个因素综合起来可以准确的为安全问题划分级别，如图 1 所示。

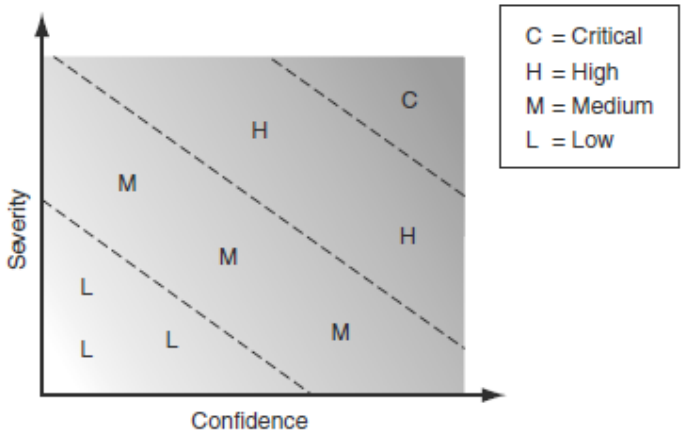


图 1 漏洞级别与严重程度、可信程度的关系

4 开源软件项目的安全漏洞情况

4.1 安全漏洞情况概览

由于极高危、高危级别的安全漏洞危害程度较高，更能反映出软件中迫切需要解决的安全问题，本部分仅讨论被测项目中的这两种级别的漏洞情况。表 2 展示了被测项目中存在的极高危以及高危安全漏洞的情况，图 2 用柱状图展示了该表中的各项分析结果。从中可以看出，大多数软件项目都存在不同程度的安全问题。

此次检测总计发现极高危漏洞 566 个, 高危漏洞 2567 个。依据此次代码测试结果，PHP 依然是漏洞多发语言。作为常用的 PHP 开源框架，Zend Framework 1.0 版本包含漏洞高达 828 个，其中极高危为 301，高危漏洞为 527 个。而在 Zend Framework 2.0 版本中漏洞总数量仅为 44 处，相比之下安全情况有了明显改善。同样作为国内使用率较高的 PHP 开源框架，YII 和 CodeIgniter 也包含了相当数量的漏洞。

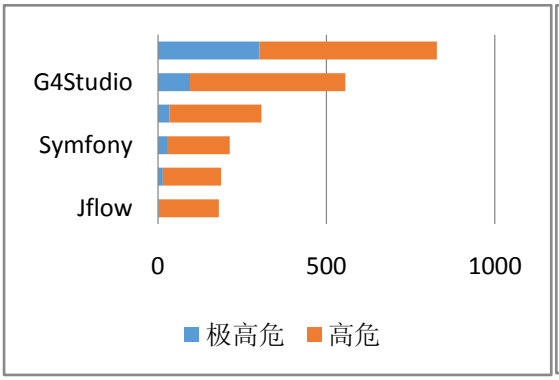
本次共测试 3 款 JAVA WEB 开源框架，包括 Wicket，Tapestry 和 Springside，其中 Wicket 的漏洞数最多，为 307 个，其次为 Tapestry，包含漏洞共 141 个；这两款软件的漏洞数量明显高于本次检测的其它同类软件。结合漏洞密度进行分析，JFinal OA 是基 java 开源框架 JFinal 和 dwz 开发的中小企业 OA 系统，该软件代码量不大，同时漏洞数量较多（共 187 个），总体来说，其安全性是本次测试的所有软件中最差的。此外，开源软件 G4Studio 的漏洞密度同样较高，由于该软件是国内面向中小企业的 JAVA 开发平台，其自身的漏洞将严重影响基于它所开发出产品的安全性。

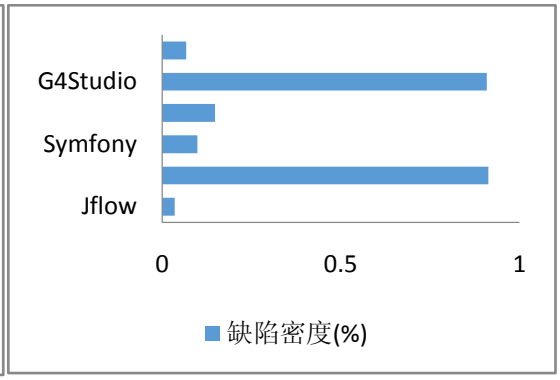
此外，有两款软件 Tengine 、Kohana 未检测出极高危、高危安全漏洞，表现出相对较好的安全性。与上季度相比，这批开源软件系统的总体安全情况有明显改善，但安全问题依然比较严重。漏洞数量排名靠前的项目处于极其容易被攻击者利用的状态，实际使用者急需通过安装补丁或者更新版本的方式进行修复和升级。

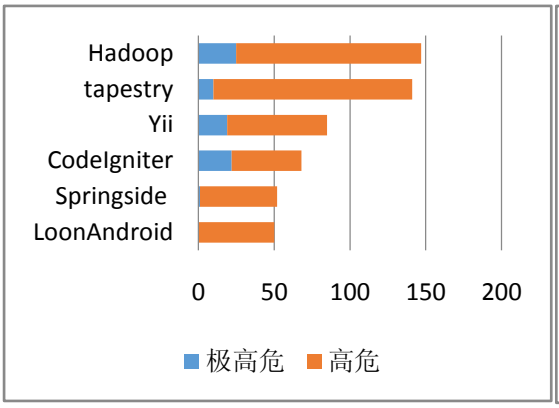
表 2. 开源软件项目中安全漏洞概览

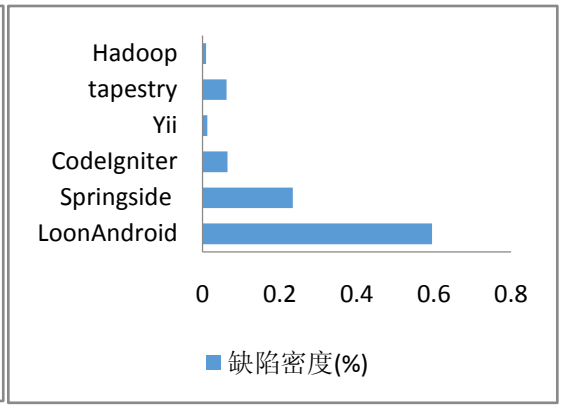
项目名称	极高危(C)	高危(H)	总和(H+C)	漏洞密度%((H+C)/L)
RoadFlow	0	6	6	0.001987683
G4Studio	95	461	556	0.908303792
Magnolia	0	1	1	0.01606942
Springside	1	51	52	0.234223684

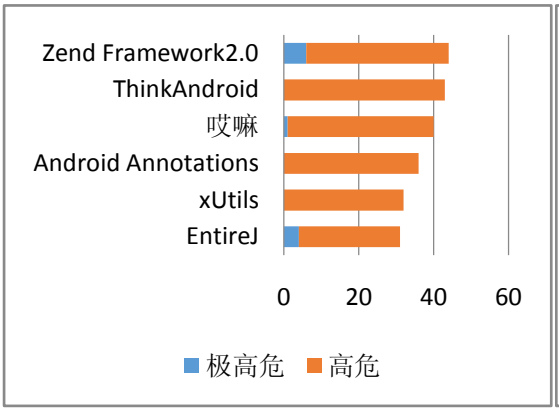
哎嘛	1	39	40	0.046606467
Flight	0	1	1	0.061576355
Laravel	5	16	21	0.250029765
Yii	19	66	85	0.012126538
CodeIgniter	22	46	68	0.064989678
Kohana	0	0	0	0
Hadoop	25	122	147	0.009122887
Tengine	0	0	0	0
JFinal OA	14	173	187	0.912729403
IK-Analyzer	0	1	1	0.00577134
Jflow	2	179	181	0.035146109
EntireJ	4	27	31	0.092059155
ThinkAndroid	0	43	43	0.244665718
Android Annotations	0	36	36	0.10253197
xUtils	0	32	32	0.247352555
Android Volley	0	14	14	0.373532551
LoonAndroid	0	50	50	0.595308965
Wicket	34	273	307	0.148113126
android-async-http	0	20	20	0.262570566
tapestry	10	131	141	0.062166297
cakephp	0	28	28	0.061030101
Symfony	27	186	213	0.098603807
Zend Framework1.0	301	527	828	0.067570325
Zend Framework2.0	6	38	44	0.026943119

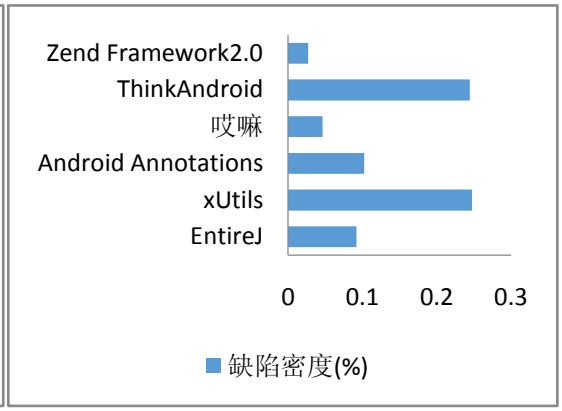

G4Studio
Symfony
Jflow
0
500
1000
极高危
高危

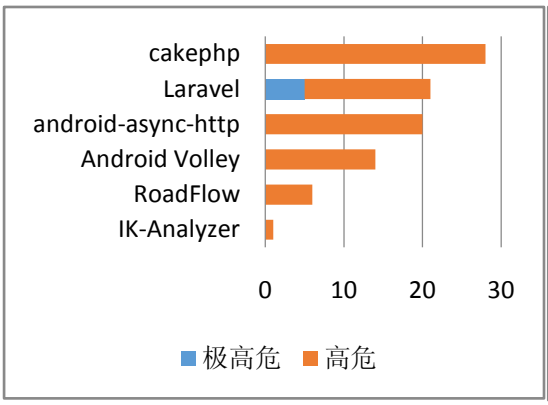

G4Studio
Symfony
Jflow
0
0.5
1
缺陷密度(%)

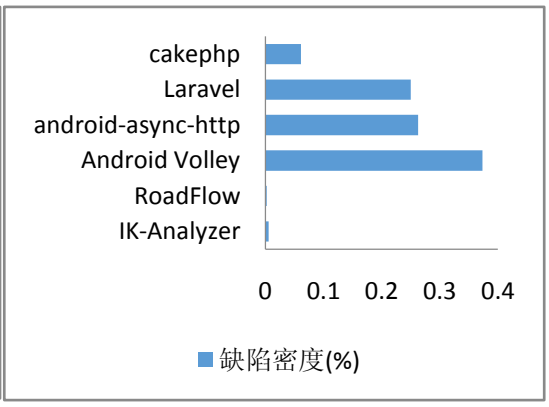

Hadoop
tapestry
Yii
CodeIgniter
Springside
LoonAndroid
0
50
100
150
200
极高危
高危


Hadoop
tapestry
Yii
CodeIgniter
Springside
LoonAndroid
0
0.2
0.4
0.6
0.8
缺陷密度(%)


Zend Framework2.0
ThinkAndroid
哎嘛
Android Annotations
xUtils
EntireJ
0
20
40
60
极高危
高危


Zend Framework2.0
ThinkAndroid
哎嘛
Android Annotations
xUtils
EntireJ
0
0.1
0.2
0.3
缺陷密度(%)


cakephp
Laravel
android-async-http
Android Volley
RoadFlow
IK-Analyzer
0
10
20
30
极高危
高危


cakephp
Laravel
android-async-http
Android Volley
RoadFlow
IK-Analyzer
0
0.1
0.2
0.3
0.4
缺陷密度(%)

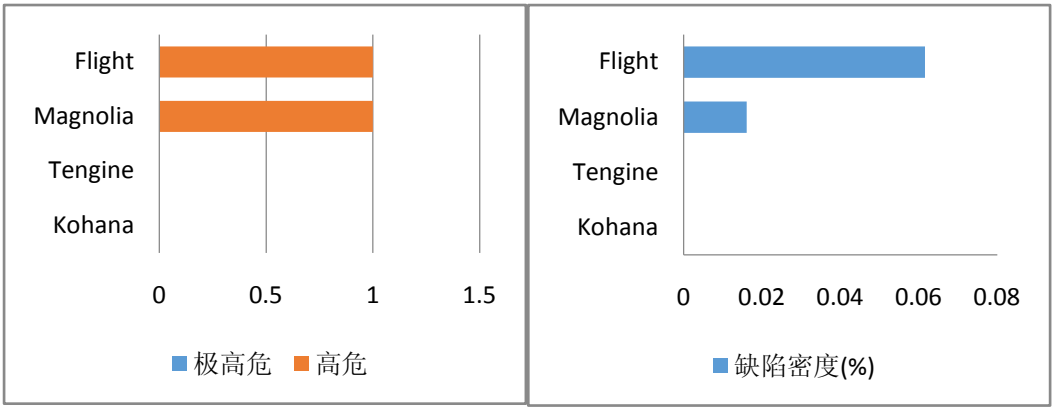


图 2 开源软件项目中安全漏洞分析图

4.2 高危安全漏洞分布情况

此次测试中发现的高危漏洞不仅数量众多，覆盖的种类也较为繁杂。图 3 展示了被测项目中高危以上级别漏洞大类的分布情况。数据表明，大多数漏洞为“输入验证与表示”类漏洞，该类漏洞易被攻击者利用，通过绕过用户输入验证从而对 Web 应用系统进行攻击，产生严重危害。“安全特性”类漏洞也占据了较大比例，攻击者可利用该类漏洞破解加密算法，导致用户隐私信息泄露等严重安全问题。此外，“代码质量问题”类漏洞也出现较多，产生的主要原因是开发人员安全意识不足，此类漏洞会导致内存溢出、资源耗尽等安全隐患，严重情况下会导致系统运行异常、甚至系统崩溃。

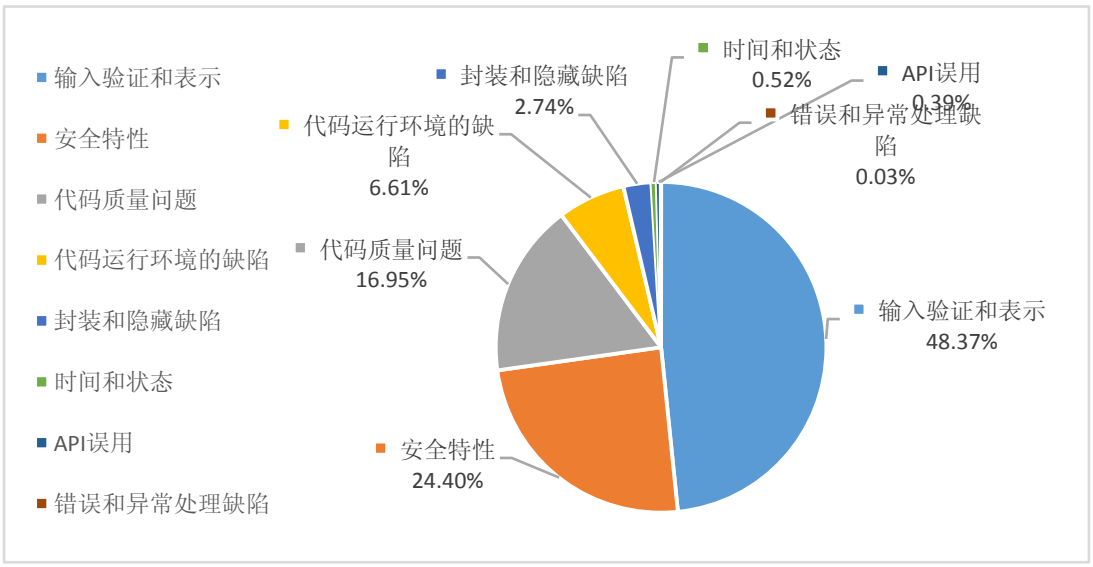


图 3 被测项目中高危以上安全漏洞的分布情况（按大类划分）

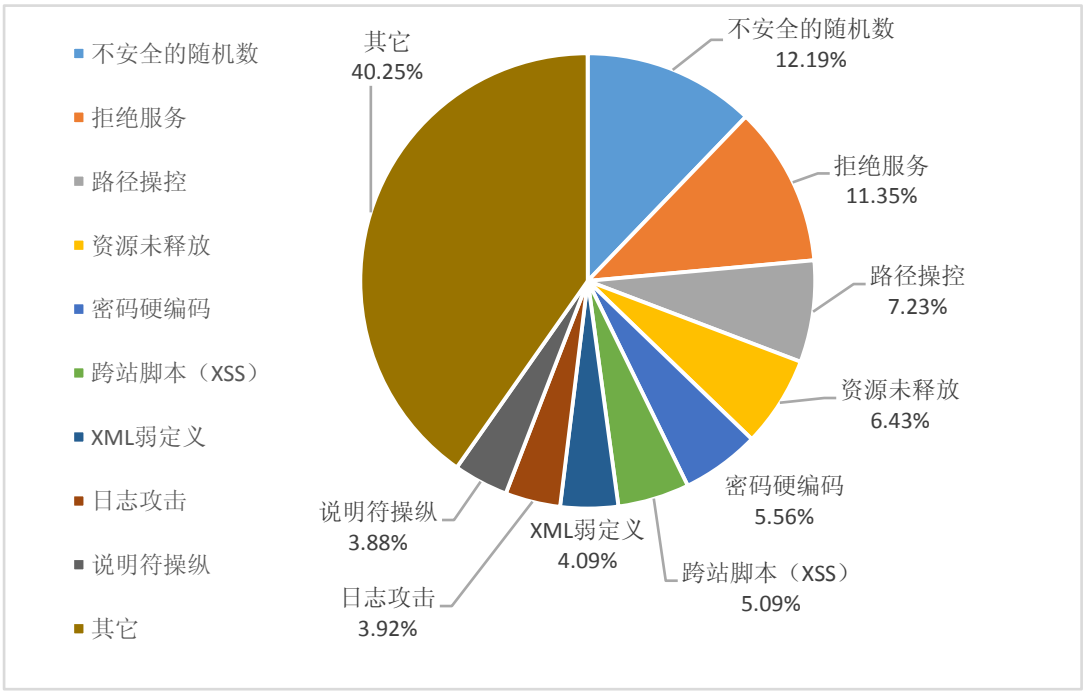


图 4 被测项目中高危以上安全漏洞的分布情况（按具体漏洞划分）

图 4 进一步展示了被测项目中的各种具体的高危以上级别安全漏洞的分布情况。在被测的 30 个项目中，出现次数超过 100 次的高危以上漏洞共有 9 种。可以看出，出现最多的前 5 种漏洞依次是 :不安全的随机数(12.19% ,364 个)、拒绝服务(11.35%， 339 个) 、路径操控(7.23% ,216 个) 、资源未释放(6.43% ,192 个)和密码硬编码(5.56%，166 个 ）。下面对这 5 种漏洞进行简要说明，并给出防范建议。

1) 不安全的随机数（属于安全特性缺陷）

危害：标准的伪随机数值生成器不能抵挡各种加密攻击。

防范：使用密码学的伪随机数生成器，使输出结果较难预测。

2) 拒绝服务（属于输入验证与表示缺陷）

危害：攻击者可以造成程序崩溃或使合法用户无法正常使用。

防范：排除代码中只需要使用少量请求就可以使得攻击者让应用程序过载的漏洞。

3) 路径操控（属于输入验证与表示缺陷）

危害：允许用户输入访问文件系统的路径，可以使攻击者访问或修改受保护的系统资源。

防范：对用户输入进行限制，阻止攻击者访问或修改受保护的系统资源。

4) 未释放资源（属于代码质量缺陷）

危害：当程序没有释放某个资源时，可能只会导致一般的软件可靠性问题，但如果攻击者能够故意触发资源泄漏，该攻击者就有可能通过耗尽资源池的方式发起拒绝服务攻击。

防范：每个资源的打开操作都要有对应的关闭操作。

5) 密码硬编码（属于安全特性缺陷）

危害：程序员将密码嵌在代码逻辑当中，不仅带来逻辑上的难于理解和不好维护，也可能造成敏感信息泄露。

防范：密码应以加密方式保存在数据库或项目配置文件中。

4.3 安全漏洞总体分布情况

本部分进一步展示被测项目中，所有级别安全漏洞的总体分布情况。图 5 展示了被测项目中安全漏洞大类的分布情况。与高危以上级别的漏洞分布情况相比，“代码质量问题”、“错误和异常处理”和“封装和隐藏缺陷”比重明显增加。这三种类型的漏洞相对来说威胁较低，容易被开发人员忽视；这些漏洞虽然不易直接产生重大危害，但可能导致系统运行不稳定、系统重要信息泄露等安全隐患，一旦被攻击者利用也会造成严重后果。

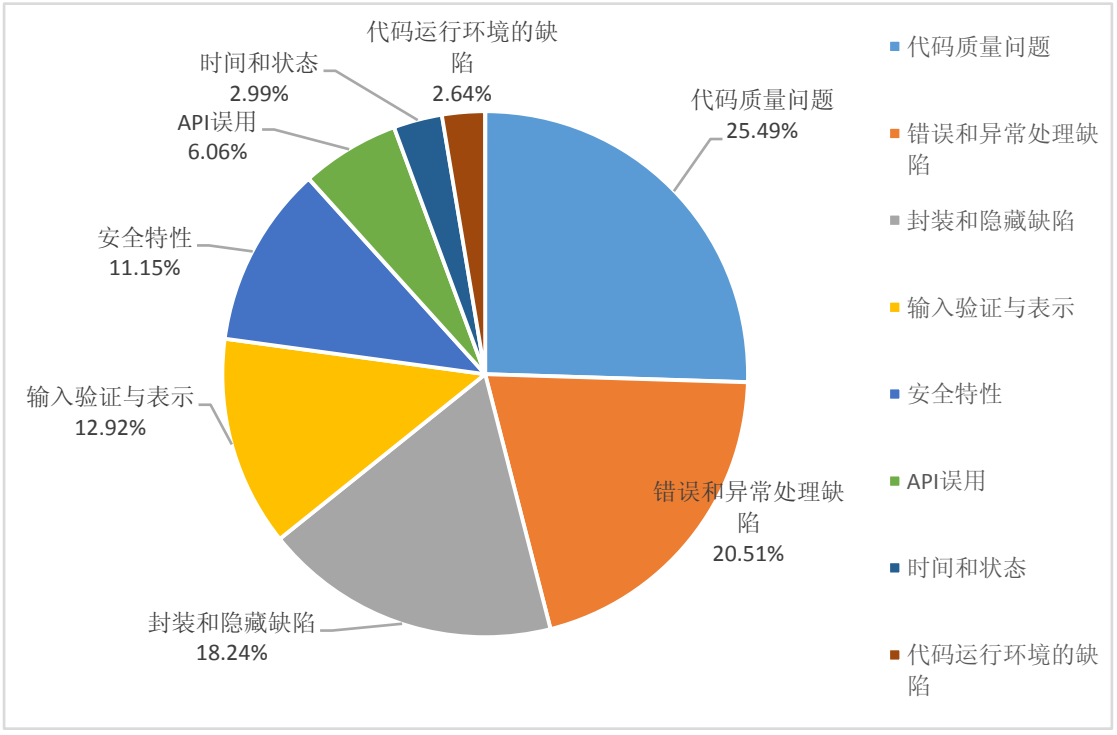


图 5 被测项目中的全部安全漏洞的分布情况（按大类划分）

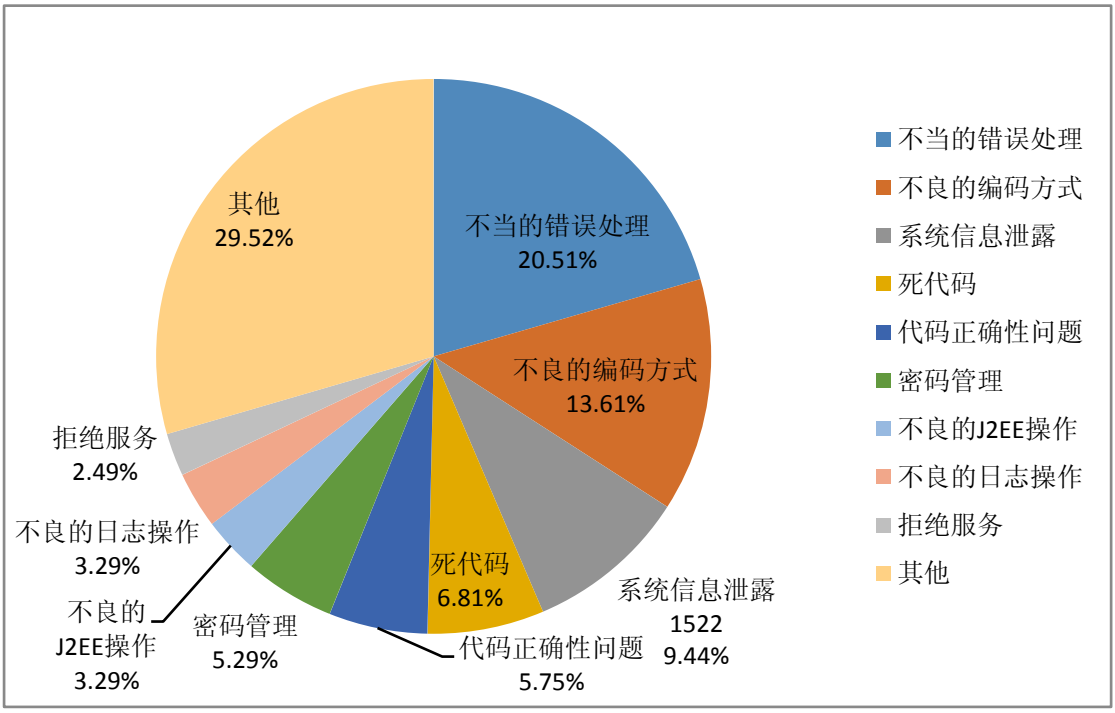


图 6　被测项目中的全部安全漏洞的分布情况（按具体安全漏洞划分）

图 6 进一步展示了被测项目中的各种具体的安全漏洞的分布情况。在被测的 30 个项目中，出现次数超过 400 次的漏洞共有 9 种。出现最多的 5 种漏洞依次是：不当的错误处理（20.51%，3307 个），不良的编码方式（13.61%，2194 个），系统信息泄漏（9.44%，1522 个），死代码（6.81%，1098 个），代码正确性问题（5.75%，928 个)。下面对这排名前 5 种漏洞进行简要说明，并给出防范建议。

1）不当的错误处理（属于错误和异常处理缺陷）

危害：对于提供了错误或异常处理 API 的程序语言，如果对于错误不做处理或处理不当，会造成输出大量错误信息、程序崩溃等后果。

防范：遵守错误处理规范，错误处理要全面，尤其对于处理过程中出现的新错误也要进行嵌套式处理。

2）不良的编码方式（属于代码质量缺陷）

危害：代码中可能存在不好的编码方式（如变量赋值后并不使用，而变成死存储；一个代码块中不包含任何指令等），除了造成存储浪费以外，很可能是代码逻辑出现错误。

防范：检查代码编写是否存在错误；如果逻辑上不存在错误，建议删除冗余代码并修改代码样式，提高可读性。

3）系统信息泄露（属于封装和隐藏缺陷）

危害：由于对系统敏感信息处理不当，会暴露系统数据或调试信息，这会帮助攻击者获悉系统脆弱点，从而进行攻击尝试。

防范：区分敏感和不敏感信息，对于敏感信息禁止输出。错误信息应由自定义页面输出，避免错误页面中包含系统信息。

4）死代码（属于代码质量缺陷）

危害：在程序操作过程中存在永远不可能被执行到的代码，最终可能会因为系统中的未用代码过多而导致程序出现性能问题。此外，考虑是否代码逻辑出现错误。

防范：检查代码编写是否存在错误；处理程序中无效代码，提高代码使用率。

5）代码正确性问题（属于代码质量缺陷）

危害：错误代码会导致不可预测的行为。对于攻击者而言，错误代码使他们可以通过意想不到的方式威胁系统。

防范：提高代码质量，降低代码问题对系统的影响。

5 关于本报告的说明

一、本报告仅从代码角度进行漏洞分析。在实际系统中，由于软件实际部署环境、安全设备等的限制，部分漏洞可能无法通过渗透测试得到验证。

二、本报告中的漏洞仅适用于表 1 中列出的特定软件版本。当软件版本有任何更新、修改和优化时，本报告不再适用。