静态源代码安全分析工具

SonarQube 测评报告

测评周期：2023 年 9 月 20 日 - 2023 年 9 月 28 日

报告日期：2023 年 9 月 28 日

报告名称	静态源代码安全分析工具 SonarQube 测评报告	版本	v1.1
报告编号	INSBUG-S-202308-005	日期	2023 年 9 月 28 日

供应链安全检测中心

是一家专业的供应链安全检测机构，坐落于国家网络安全人才与创新基地，洞源实验室为该中心旗下的安全实验室，致力于供应链软件和硬件的安全性检测与评估。

实验室拥有具有多年从业经验的安全专家和研究人员，长期关注供应链安全领域的技术发展、安全事件和解决方案。实验室具有成熟的供应链安全检测方法和工具，可以对软件、硬件等关键产品进行全面系统的安全检查，发现潜在的安全风险和漏洞。

一、 测评目的

此次静态源代码安全分析工具产品测评针对国外常见同类产品开展，并基于 OWASP 中国发布的《静态源代码安全扫描工具测评基准 v2.0》开展测评工作，旨在基于该基准中的测评维度评估国外同类产品的产品能力，帮助国内企业、机构或个人作为选用和研究的参考。

二、 测评方法

1. 环境说明

为了保证测评期间工具或产品的封闭性、独立性，或不受云上或在线因素的影响，本次测评期间采用独立的、离线的计算环境进行测评，产品均采用离线部署的版本进行测评。

详细环境配置见下文【测评环境】。

2. 测评对象

被测评的产品包括产品安装包、产品功能以及官方手册或文档，以从真实客户使用的视角评估产品能力，故测评过程中，产品能力的满足情况包括文档的完整性以及功能的完整性和可用性。

本次测评的对象是 SonarQube 产品。

3. 版本选择

本次测评选择 SonarQube Community Edition 的 10.1 版本作为测评对象。

4. 测评依据

本次静态源代码安全分析工具产品测评依据是 OWASP 中国发布的《静态源代码安全扫描工具测评基准 v2.0》，基准测评项包括：

- 部署环境

- 安全扫描
- 漏洞检测
- 源码支持
- 扩展集成
- 产品交互
- 报告输出

5. 测评样本

本次产品测评所有被测产品均采用相同的测试样本进行测试，所有的测试样本均采用开源项目，使用的版本是测评期间该项目的某个版本及其相应的代码量。

为了确保漏洞检测过程中漏洞种类的多样性和漏洞的复杂性，以便更好地验证产品的安全漏洞检测能力，满足可以重复进行漏洞测试的需求，以及避免人工漏洞判断导致的测试主观性，测试样本均采用有明确漏洞类型、漏洞信息的安全漏洞验证开源应用或靶场（包括自建的超过 5 种开发语言、18 种漏洞类型的数百个代码样本库），以用于构建可控的测试环境，从而更全面、严谨地验证工具或产品的检测能力。

Java、PHP 和 C#是当前应用最广泛的编程语言。

- Java 拥有跨平台优势，在服务器端应用开发中使用广泛。
- PHP 是最流行的 Web 应用语言之一，大量开源和业务系统使用 PHP 开发。
- C#在 Windows 系统应用和企业系统开发中应用广泛。

选择这三种语言的测试样本，可以覆盖不同系统环境、业务场景和应用类型,且三种语言均有大量成熟稳定的开源应用，适合作为静态源代码分析工具测评的对象，全面评估工具或产品对各类漏洞的检测效果。

注：

测试样本根据代码量和开发语言从测试样本库中随机挑选，因此相同类型产品的不同批次检测采用的测试样本会有不同。

本次测评选择了四款漏洞测评样本，部分产品可能会针对某些测试样本的漏洞做出定制化的调整以降低误报率和漏报率，因此综合测评结果不代表产品

在实际生产应用中的漏洞检测效果。

6. 漏洞统计

本次测评产品漏洞误报率和漏报率是基于测试样本列表中的漏洞测试样本进行测试。为确保测评数据的准确性和客观性，被测评产品检出的安全漏洞不做人为漏洞分析和准确性判断，故非测试样本官方标识的漏洞不计入误报率和漏报率的统计。

报告采用的漏洞误报率/漏报率的相关概念及计算方式如下：

- 实际漏洞数：测试样本官方标识的漏洞数量。
- 检出漏洞数：产品检测出的官方标识漏洞文件中的漏洞数量。
- 漏洞命中数：产品检测出的漏洞数量命中官方标识漏洞的数量。
- 误报率：（检出漏洞数-漏洞命中数）/检出漏洞数
- 漏报率：（实际漏洞数-漏洞命中数）/实际漏洞数

三、 测评范围

被测产品：SonarQube Community Edition

被测版本：10.1

产品介绍：

SonarQube 是一种自动代码审查工具，可以系统地帮助企业交付高质量的代码。作为 Sonar 解决方案的核心元素 ，SonarQube 可集成到现有的工作流程中并检测代码中的问题，以对项目进行持续的代码检查。该工具可分析 30 多种不同的编程语言 ，并支持集成到 CI 管道和 DevOps 平台中 ，以确保代码符合标准。

四、 测评结果

根据测评详情描述，测评结果分为：满足、部分满足和不满足。

为确保漏洞误报率和误报率的公正性和客观性，测评过程中无人员介入漏

洞分析与判断，故测评结果中漏洞误报率相比实际漏洞误报率或有偏低，详见【漏洞误报率/漏报率】。

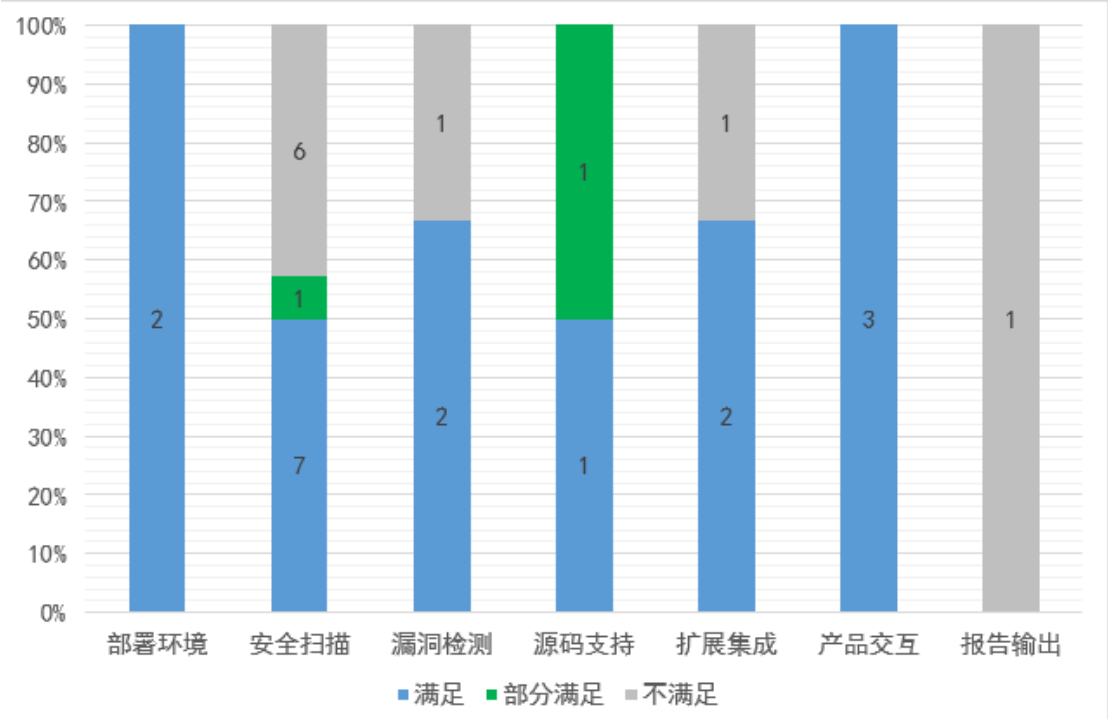

图 1 测评结果总览

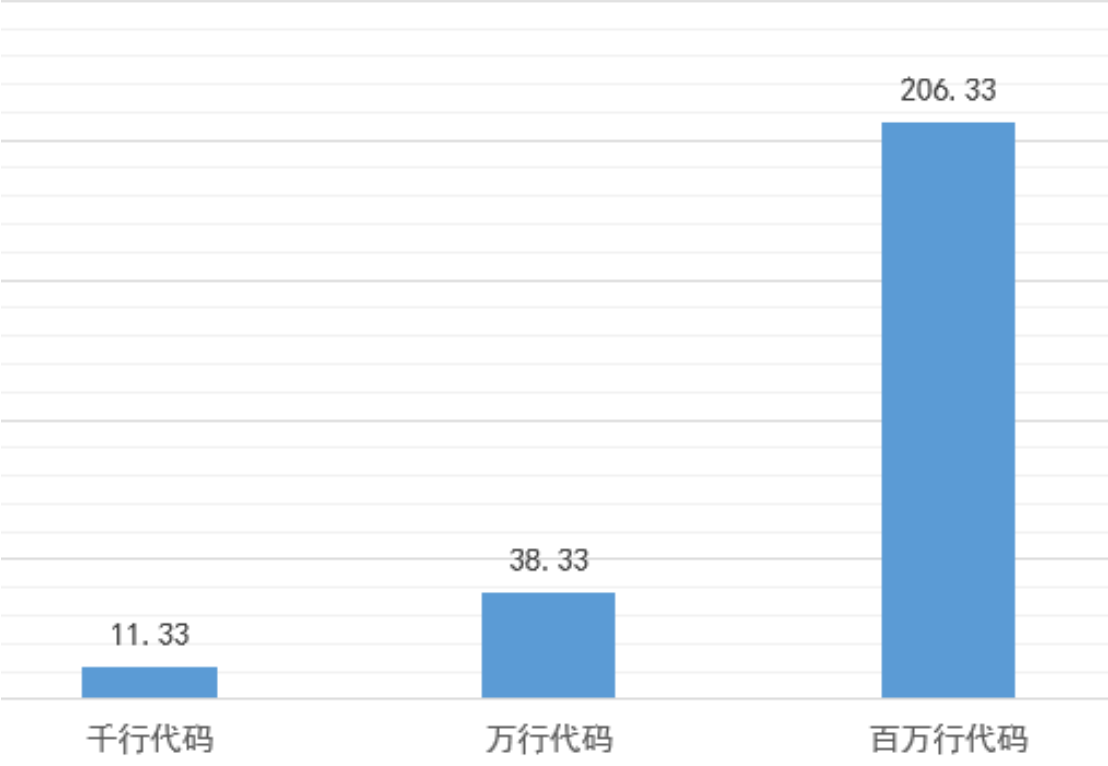

图 2 平均扫描速率（单位：秒）

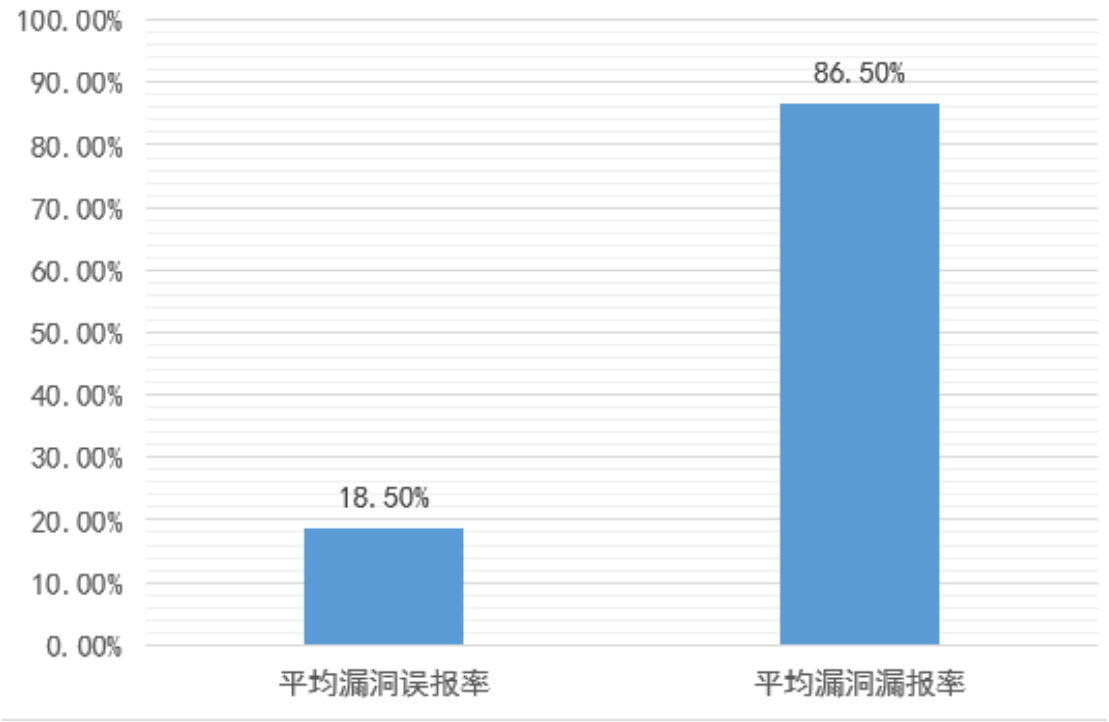

图 3 平均漏洞误报率/漏报率

测评维度	测评项	测评子项	测评结果
部署环境	操作系统支持	-	满足
	容器化支持	-	满足
安全扫描	扫描速度	-	满足 （千行代码平均：11.33s） （万行代码平均：38.33s） （百万行代码平均：206.33s）
	扫描配置	定时扫描	不满足
		扫描进度	不满足
		并发扫描	满足
		增量扫描	不满足
	漏洞误报率	-	满足（平均 18.5%）
	漏洞漏报率	-	不满足（平均 86.5%）
	编译代码支持	-	不满足
	移动应用支持	-	部分满足
	漏洞规则支持	修改漏洞规则能力	不满足
		新增漏洞规则能力	满足
	漏洞标记能力	标记漏洞	满足

		分类漏洞	满足
		归档漏洞	满足
漏洞检测	漏洞类型支持	-	满足
	漏洞信息支持	-	满足
	开发框架支持	-	不满足
源码支持	开发语言支持	-	部分满足
	源码导入方式	-	满足
扩展集成	源代码管理系统集成	-	满足
	缺陷跟踪系统集成	-	不满足
	持续集成系统集成	-	满足
产品交互	图形界面模式	-	满足
	命令行模式	-	满足
	IDE 插件模式	-	满足
报告输出	-	-	不满足

表 1 SonarQube 产品测评结果详情

五、 测评环境

1. 部署环境配置

产品测评采用相同的产品部署环境，以避免由于配置不同导致的产品能力偏差，同时，测评期间采用待测评产品的默认配置与部署，不做额外自定义配置或配置修改。

统一的部署产品环境配置信息如下：

- 处理器：Inter(R) Core(TM) i5-7200U
- 内存：16 GB
- 硬盘：500 GB

2. 测试样本列表

测试样本是产品测评中用于检测产品的扫描速度以及漏洞漏报和误报情况的代码库。

1） 扫描速度测试样本

项目名	版本	代码量	开发语言	项目地址
information-management-system-of-students	2015/5/18	7,920	C#	https://github.com/zhujainxipan/information-management-system-of-students
WebGoat（.Net）	2014/2/23	50,021	C#	https://github.com/jerryhoff/WebGoat.NET
Windows Presentation Foundation (WPF)	v7.0.9	1,938,664	C#	https://github.com/dotnet/wpf
Hackazon 项目 vulninjection 模块	2021/3/12	5,053	PHP	https://github.com/rapid7/hackazon
Hackazon	2021/3/12	450,913	PHP	https://github.com/rapid7/hackazon
WordPress	6.2.2	969,871	PHP	https://github.com/WordPress/WordPress
Hutool 项目 crypto 模块	5.8.20	6,436	Java	https://github.com/dromara/hutool
WebGoat（Java）	v2023.4	82,578	Java	https://github.com/WebGoat/WebGoat
OWASP-Benchmark	1.2beta	1,186,674	Java	https://github.com/OWASP-Benchmark/BenchmarkJava

表 2 扫描速度测试样本

2） 漏洞检测测试样本

项目名	版本	代码量	开发语言	项目地址
WebGoat（.Net）	2014/2/23	50,021	C#	https://github.com/jerryhoff/WebGoat.NET
bWAPP	2021/12/2	33,011	PHP	https://github.com/raesene/bWAPP
WebGoat（Java）	v2023.4	82,578	Java	https://github.com/WebGoat/WebGoat
OWASP-Benchmark	1.2beta	1,186,674	Java	https://github.com/OWASP-Benchmark/BenchmarkJava

表 3 漏洞检测测试样本

六、 产品测试详情

1. 部署环境

1) 操作系统支持

SonarQube 产品支持 Linux、MacOS、Windows 系统。

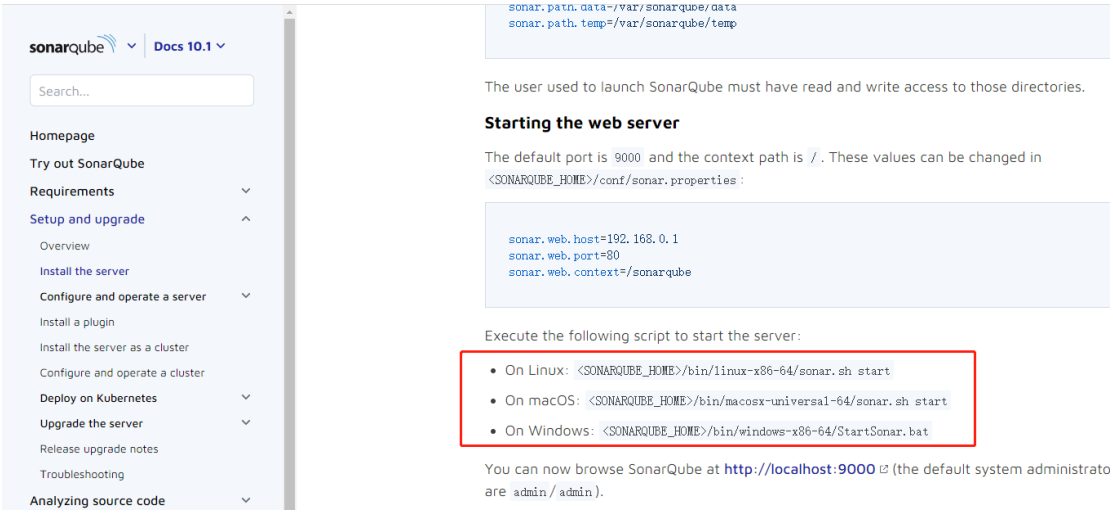

图 4 SonarQube 部署系统支持

2) 容器化支持

SonarQube 支持容器化部署。

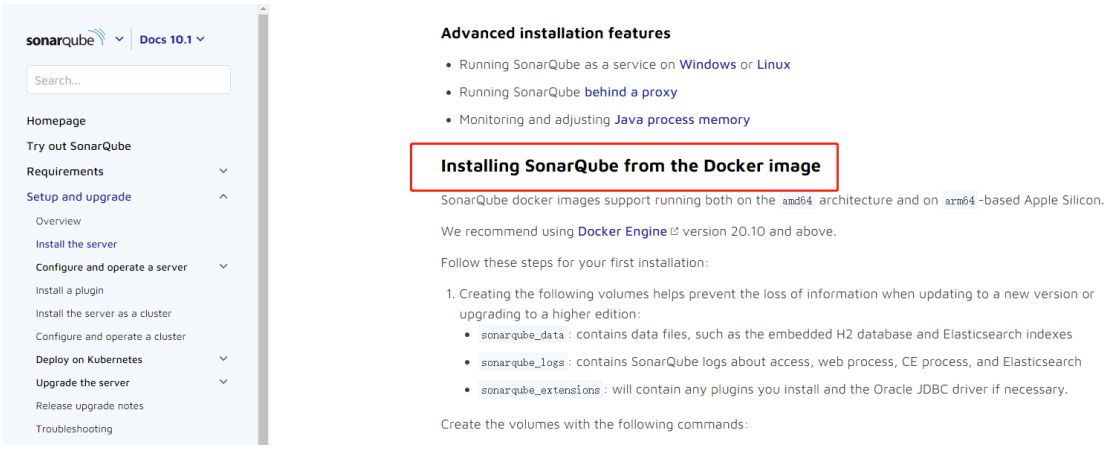

图 5 SonarQube 容器化支持

2. 安全扫描

1) 扫描速度

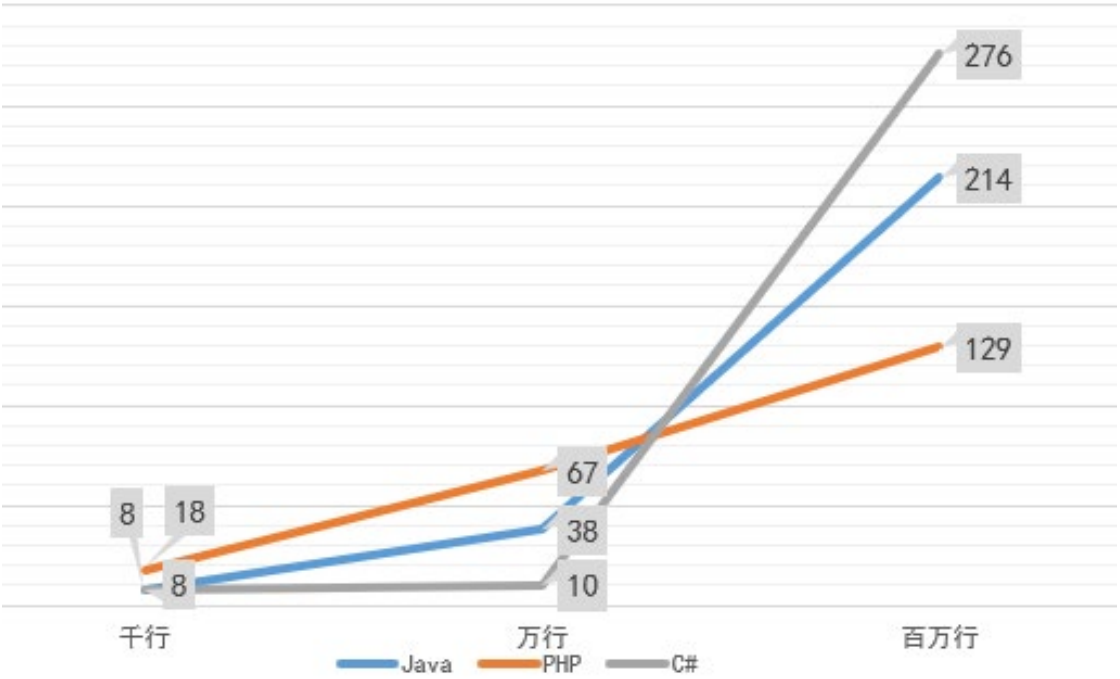

图 6 不同代码量级扫描速度（单位：秒）

测试项目	开发语言	代码量	扫描时长
Hutool 项目 crypto 模块	Java	6,436	8s
Hackazon 项目 vulninjection 模块	PHP	5,053	18s
information-management-system-of-students	C#	7,920	8s

表 4 千行级样本代码扫描速度

测试项目	开发语言	代码量	扫描时长
WebGoat（Java）	Java	82,578	38s
Hackazon	PHP	450,913	67s
WebGoat（.Net）	C#	33,532	10s

表 5 万行级样本代码扫描速度

测试项目	开发语言	代码量	扫描时长
OWASP-Benchmark	Java	1,186,674	214s
WordPress	PHP	969,871	129s
Windows Presentation Foundation (WPF)	C#	1,938,664	276s

表 6 百万行级样本代码扫描速度

2) 扫描配置

（1） 定时扫描

SonarQube 不支持对项目源代码设定周期自动扫描。

（2） 扫描进度

SonarQube 无法呈现实时的代码扫描进度，仅可呈现扫描时长。

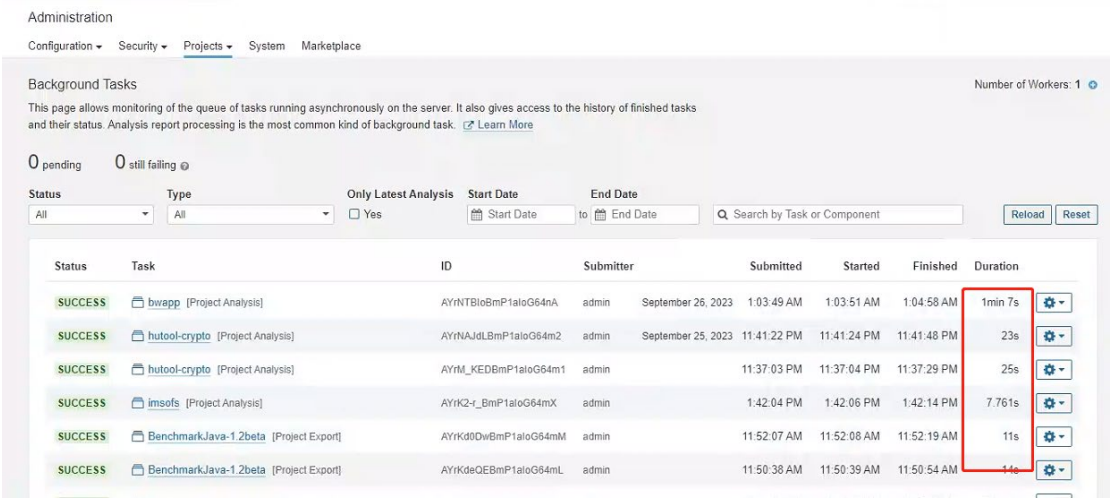

图 7 SonarQube 扫描进度

（3） 并发扫描

SonarQube 支持并发扫描。

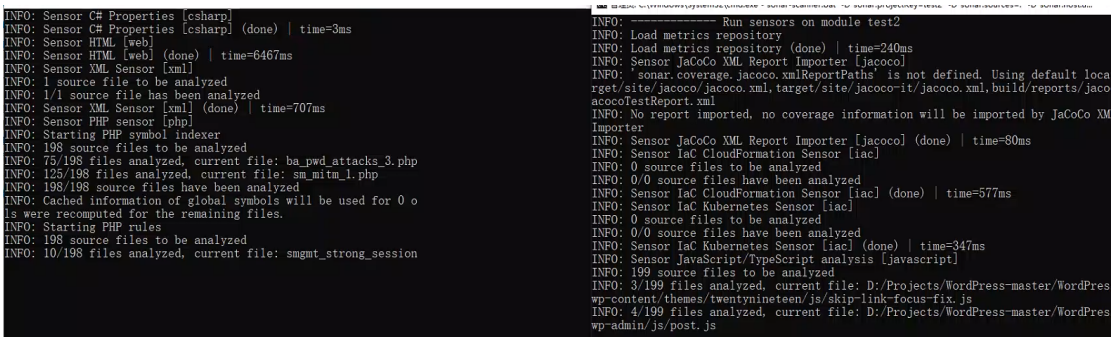

图 8 SonarQube 并发扫描

（4） 增量扫描

SonarQube 不支持增量扫描。

3） 漏洞误报率/漏报率

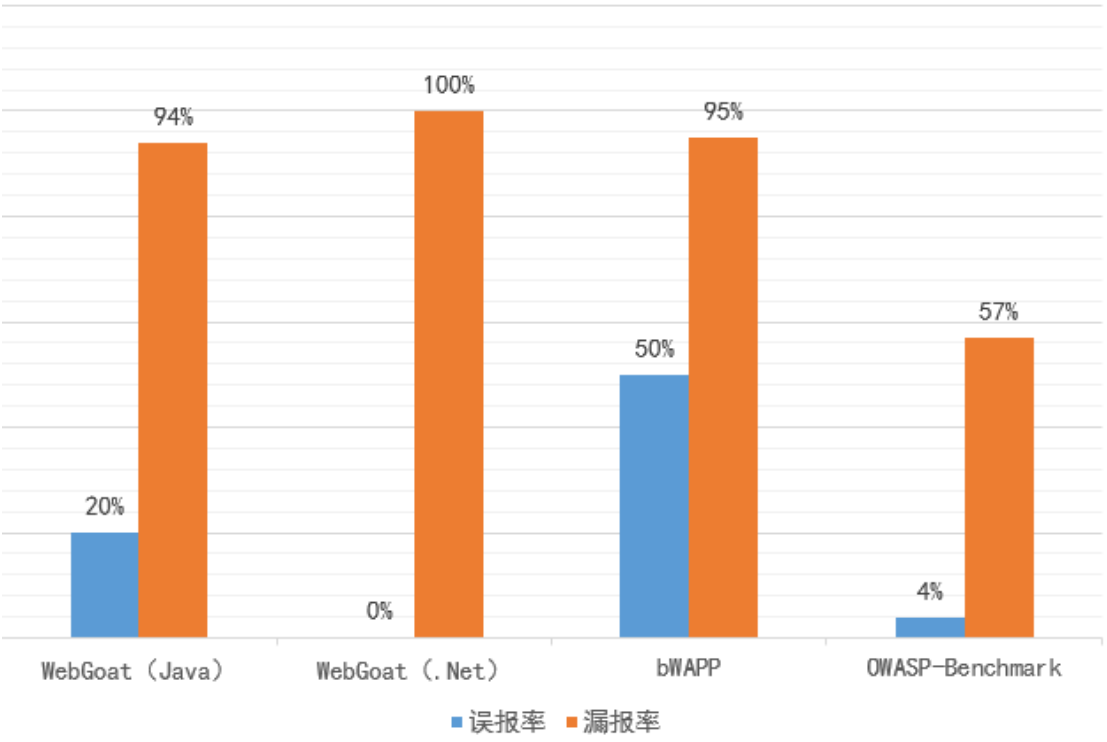

图 9 SonarQube 测试样本漏洞误报率/漏报率统计

序号	漏洞类型	实际漏洞数	检出漏洞数	漏洞命中数	误报率	漏报率
A1	Broken Access Control	9	0	0	0	100%
A2	Crypto	5	0	0	0	100%
A3	Injection	27	5	4	20%	85%
A5	XXE	3	0	0	0	100%
A6	Vulnerable Components	2	0	0	0	100%
A7	Identity & Auth Failure	13	0	0	0	100%
A8	Insecure Deserialization	1	0	0	0	100%
A9	Logging	2	0	0	0	100%

A10	SSRF/CSRF	6	0	0	0	100%
总数		68	5	4	20%	94%

表 7 WebGoat（Java）漏洞测试样本

序号	漏洞类型	实际漏洞数	检出漏洞数	漏洞命中数	误报率	漏报率
A1	Injection	4	0	0	0	100%
A2	XSS	2	0	0	0	100%
A3	Authentication	1	0	0	0	100%
A4	Debugging	1	0	0	0	100%
A5	Encryption	3	0	0	0	100%
A6	.net Exploits	1	0	0	0	100%
总数		12	0	0	0	100%

表 8 WebGoat（.Net）漏洞测试样本

序号	漏洞类型	实际漏洞数	检出漏洞数	漏洞命中数	误报率	漏报率
A1	Injection	21	1	1	0	95%
A2	Broken Auth & Session Mgmt	10	6	2	67%	80%
A3	XSS	15	0	0	0	100%
A4	Insecure Direct Object References	3	0	0	0	100%
A5	Security Misconfiguration	14	0	0	0	100%
A6	Sensitive Data Exposure	5	0	0	0	100%
A7	Missing Functional Level Access Control	9	0	0	0	100%
A8	CSRF	3	0	0	0	100%
A9	Using Known Vulnerable Components	3	1	1	0	67%
A10	Unvalidated Redirects & Forwards	2	0	0	0	100%
总数		85	8	4	50%	95%

表 9 bWAPP 漏洞测试样本

序号	漏洞类型	实际漏洞数	检出漏洞数	漏洞命中数	误报率	漏报率
A1	Command Injection	126	0	0	0	100%
A2	Insecure Cookie Flag	36	0	0	0	100%
A3	LDAP Injection	27	0	0	0	100%
A4	Path Traversal	133	0	0	0	100%
A5	SQL Injection	272	177	177	0	35%
A6	Trust Boundary Violation	83	0	0	0	100%
A7	Weak Encryption Algorithm	130	130	130	0	0
A8	Weak Hashing Algorithm	129	113	89	21%	31%
A9	Weak Randomness	218	218	218	0	0
A10	XPATH Injection	15	0	0	0	100%
A11	XSS (Cross-Site Scripting)	246	0	0	0	100%
总数		1415	638	614	4%	57%

表 10 OWASP-Benchmark 漏洞测试样本

测试项目	实际漏洞数	检出漏洞数	漏洞命中数	误报率	漏报率
WebGoat（Java）	68	5	4	20%	94%
WebGoat（.Net）	12	0	0	0	100%
bWAPP	85	8	4	50%	95%
OWASP-Benchmark	1415	638	614	4%	57%

表 11 漏洞误报率/漏报率结果汇总

4） 编译代码支持

SonarQube 对于 Jar、APK、EXE 等二进制文件无法直接进行分析。

5） 移动应用支持

SonarQube 支持对使用移动应用代码 Java、Kotlin 编写的 Android 应用进

行源代码分析扫描，但不支持 iOS 应用。

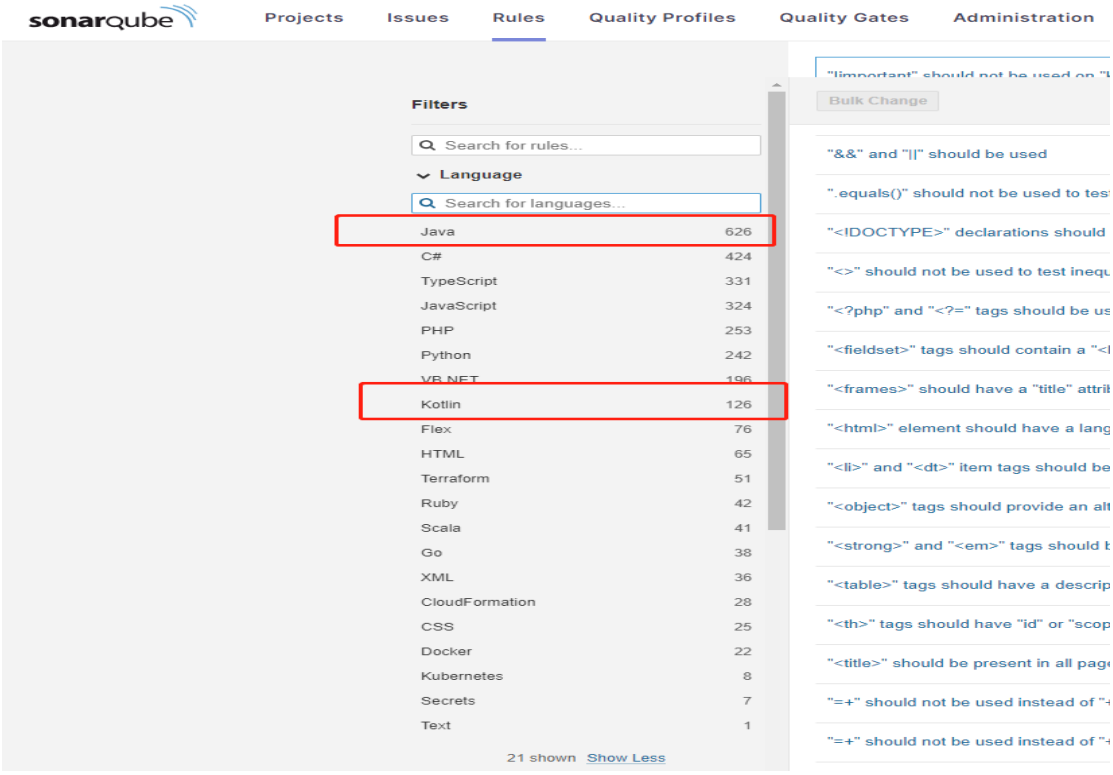

图 10 SonarQube 移动应用支持

6） 漏洞规则支持

（1） 修改漏洞规则能力

SonarQube 不具备修改自带漏洞检测规则的能力。

（2） 新增漏洞规则能力

SonarQube 支持新增漏洞检测规则。

图 11 SonarQube 添加自定义规则

图 12 SonarQube 添加自定义规则

7） 漏洞标记能力

（1） 标记漏洞

SonarQube 支持对检测到的漏洞进行不同的标记，并提供漏洞的详细信息，包括漏洞的严重性、漏洞所在的代码位置、漏洞的类型等。

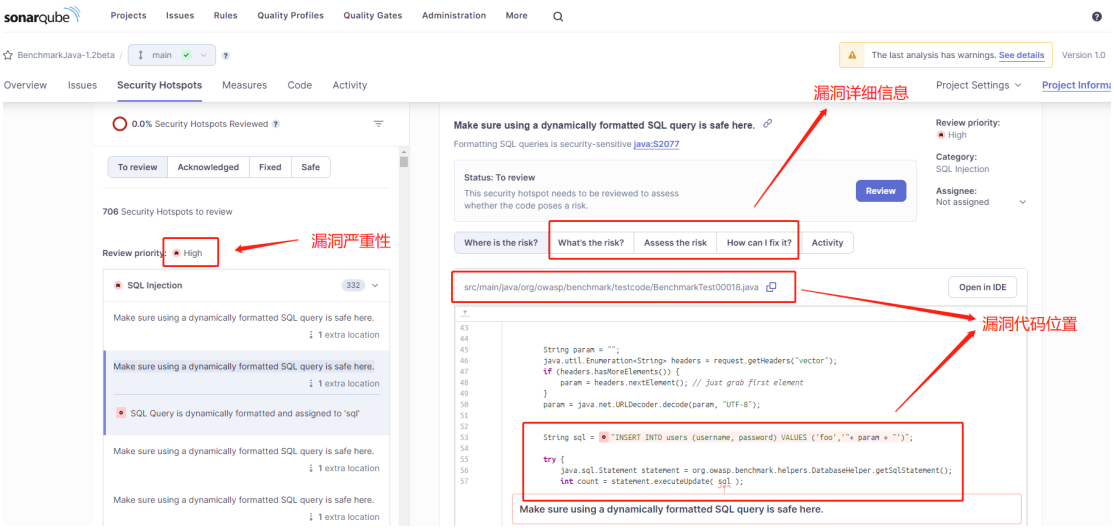

图 13 SonarQube 标记漏洞

（2） 分类漏洞

SonarQube 支持按照等级、类型、所属文件等对漏洞进行分类。

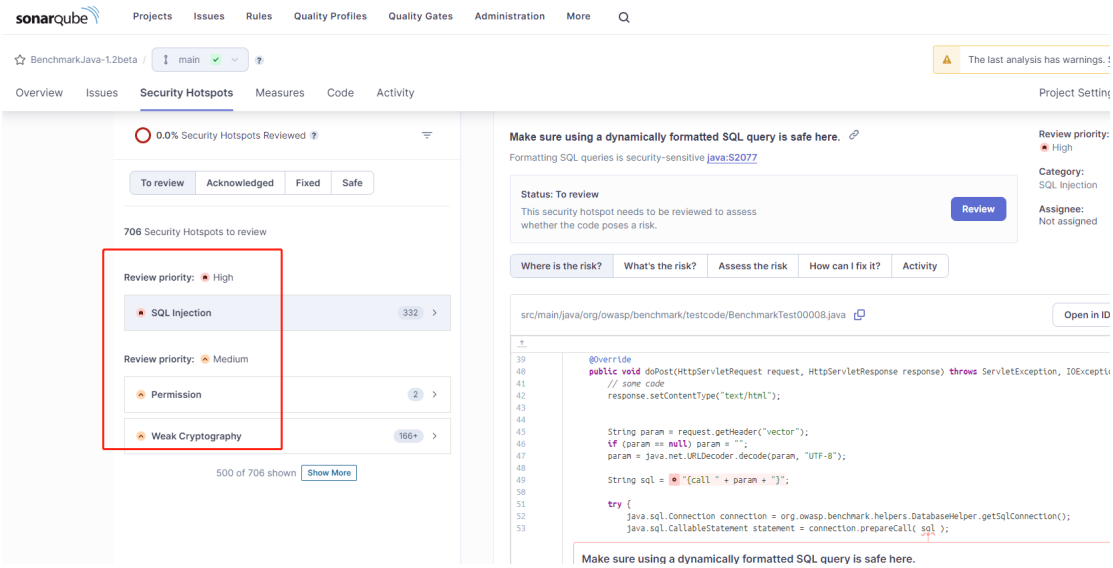

图 14 SonarQube 漏洞分类

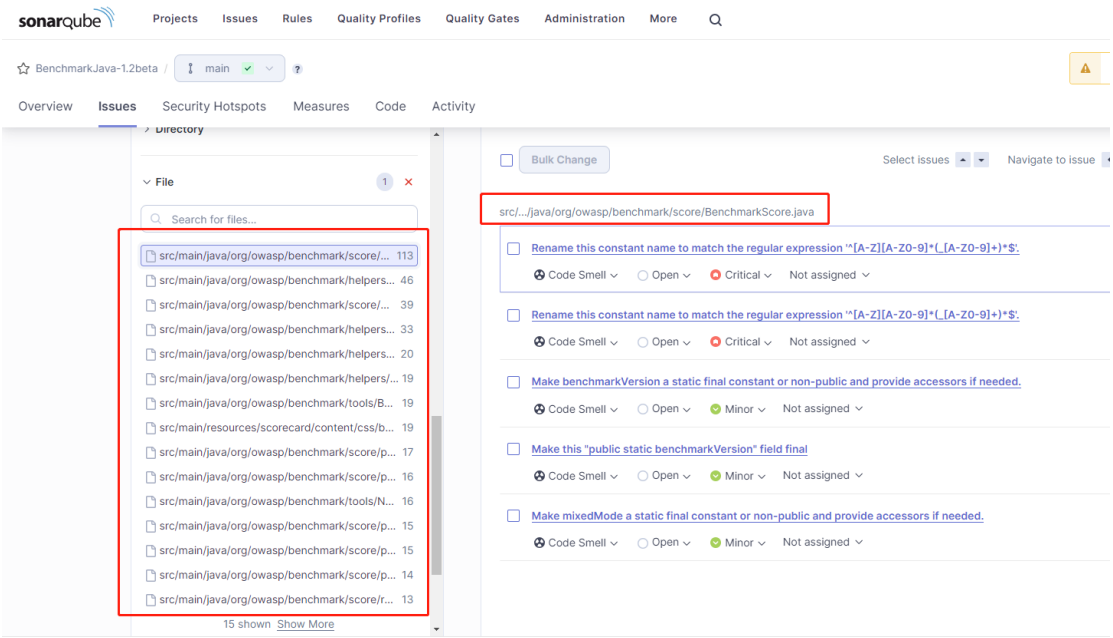

图 15 SonarQube 漏洞分类

（3） 归档漏洞

SonarQube 支持漏洞信息归档，以便使用者跟踪漏洞修复历史和分析漏洞趋势。

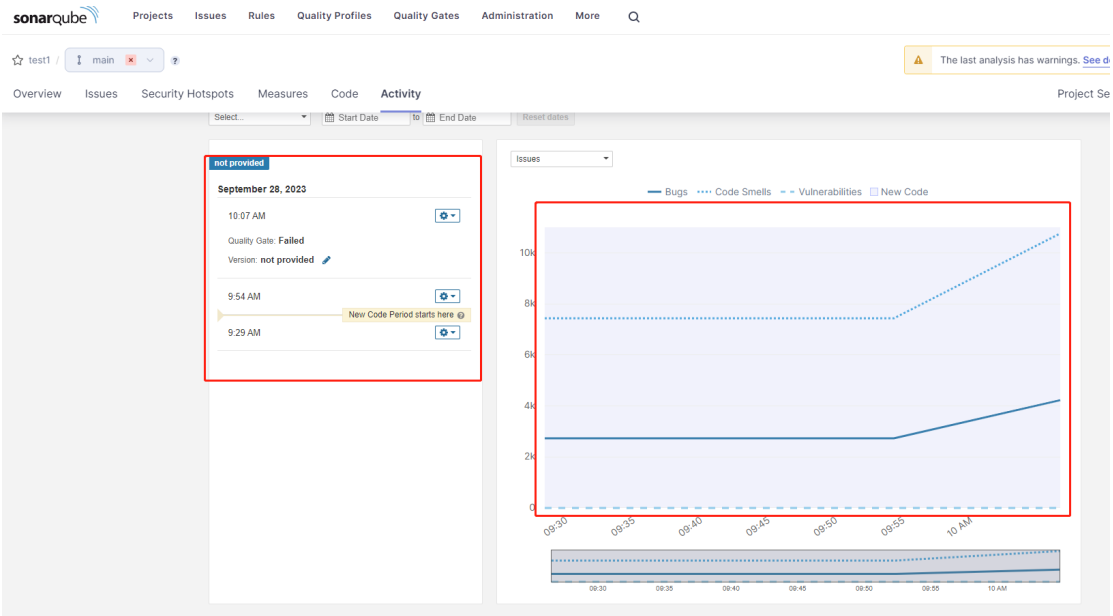

图 16 SonarQube 漏洞归档

3. 漏洞检测

1) 漏洞类型支持

SonarQube 的静态代码分析功能可以检查源代码中的安全漏洞和潜在风险，包括 OWASP Top 10 中列出的漏洞类型。

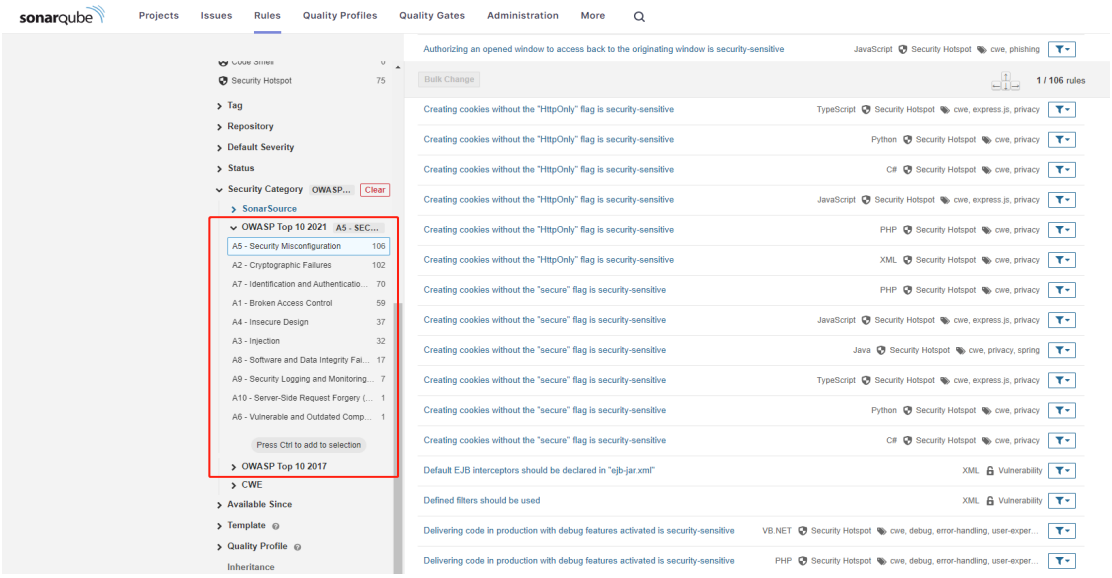

图 17 SonarQube 漏洞类型支持

2) 漏洞信息支持

SonarQube 扫描结果的漏洞信息包括漏洞类型、漏洞描述、漏洞影响、修复建议。

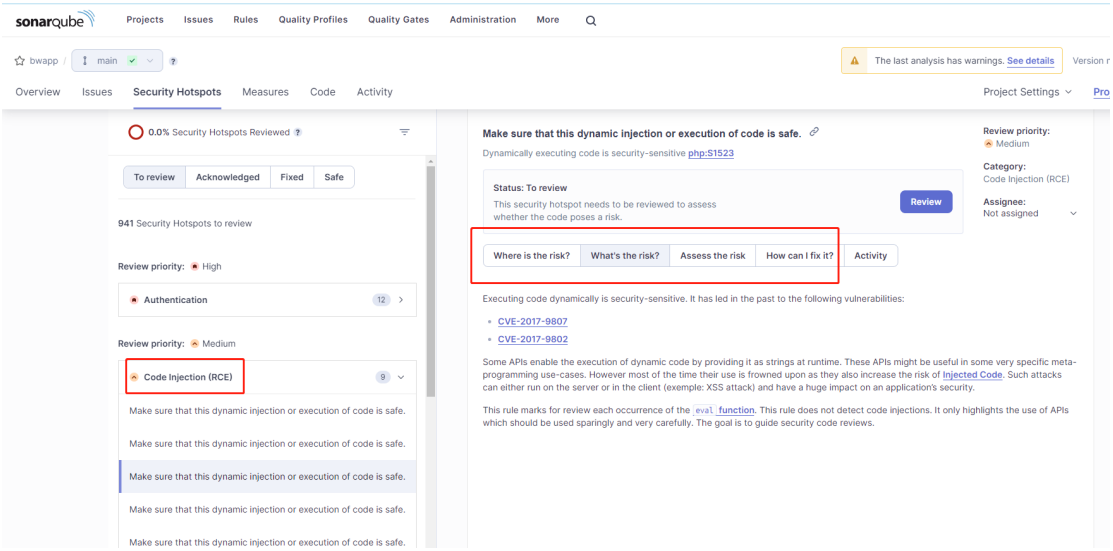

图 18 SonarQube 漏洞详情

3） 开发框架支持

SonarQube 不支持开发框架扫描识别。

4. 源码支持

1） 开发语言支持

SonarQube 支持业界主流的开发语言：Java、Python、Go、JavaScript、PHP、C#、HTML、Ruby 等，但不支持 Swift、Objective-C 语言。

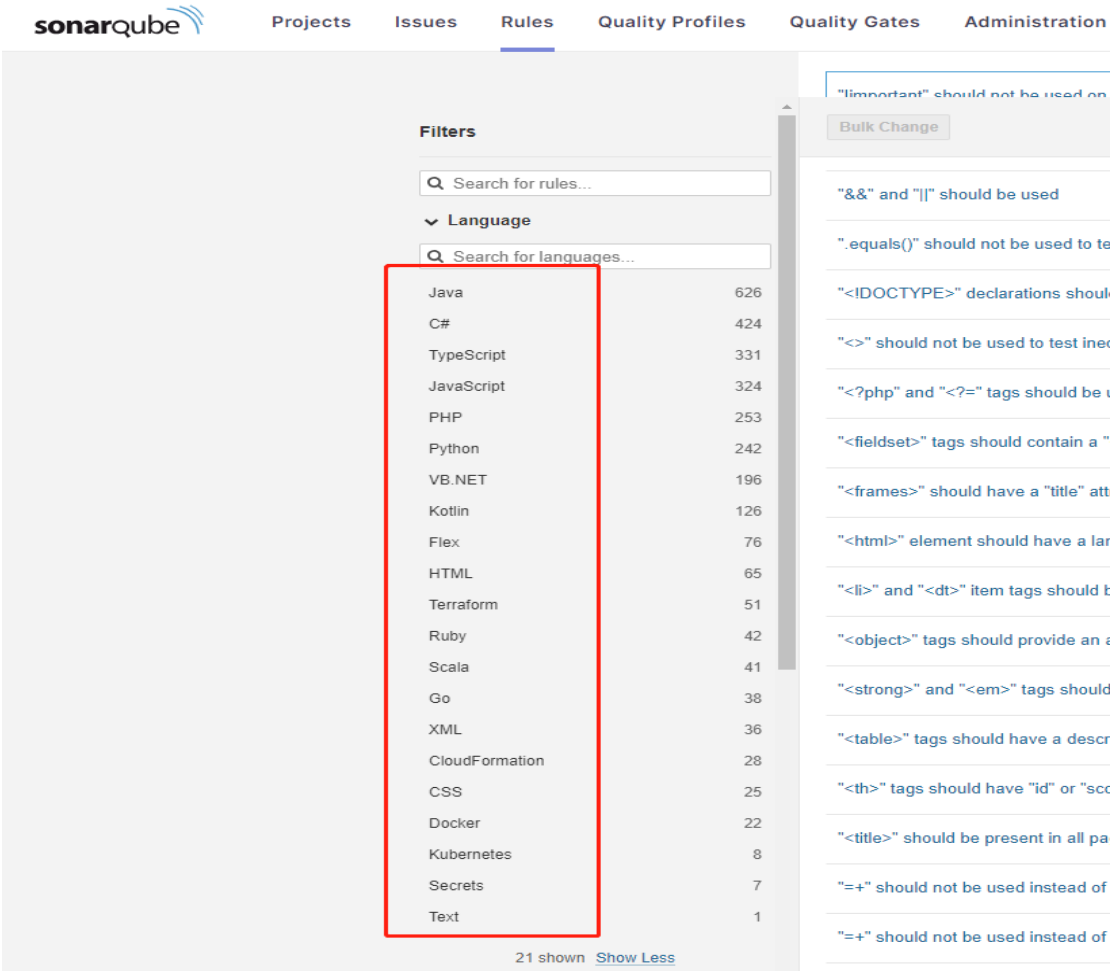

图 19 SonarQube 开发语言支持

2） 源码导入方式

SonarQube 支持源码导入方式：本地上传源码包、代码仓库拉取、源码片段上传。

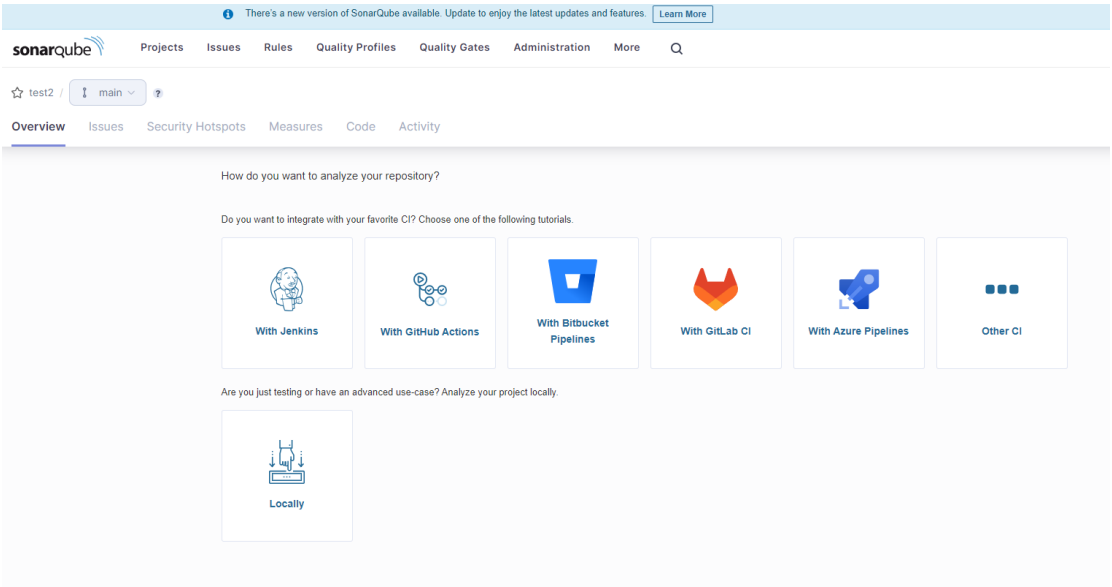

图 20 SonarQube 源码导入方式

5. 扩展集成

1) 源代码管理系统集成

SonarQube 支持与源代码管理系统（或源代码托管平台）的集成。

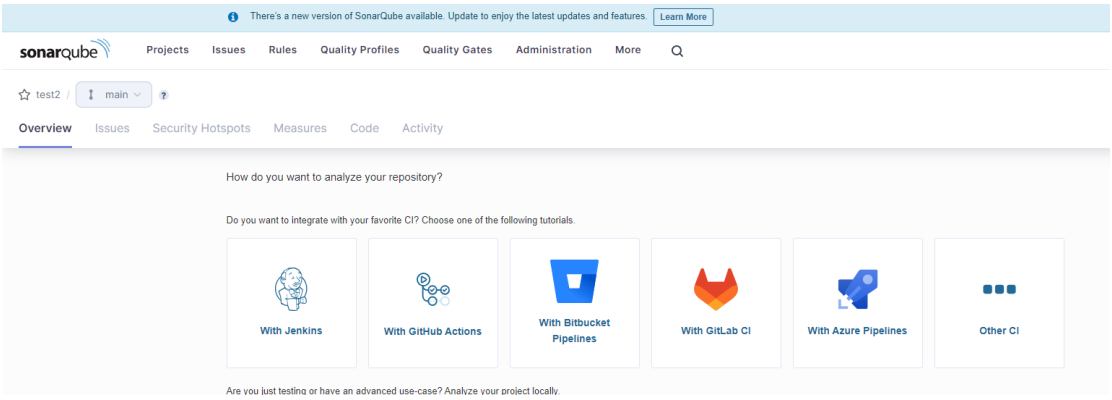

图 21 SonarQube 源代码管理系统集成支持

2) 缺陷跟踪系统集成

SonarQube 不支持与禅道缺陷管理平台集成。

3) 持续集成系统集成

SonarQube 支持与持续集成系统集成。

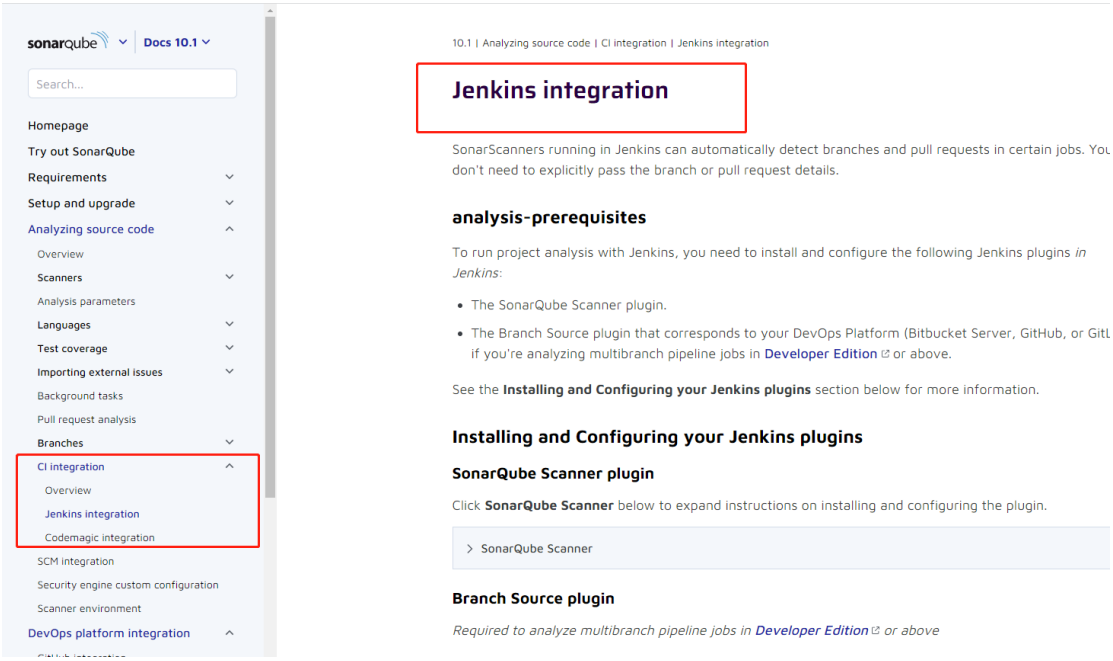

图 22 SonarQube 持续集成系统集成支持

6. 产品交互

1) 图形界面模式

SonarQube 支持浏览器界面完成工具的功能使用。

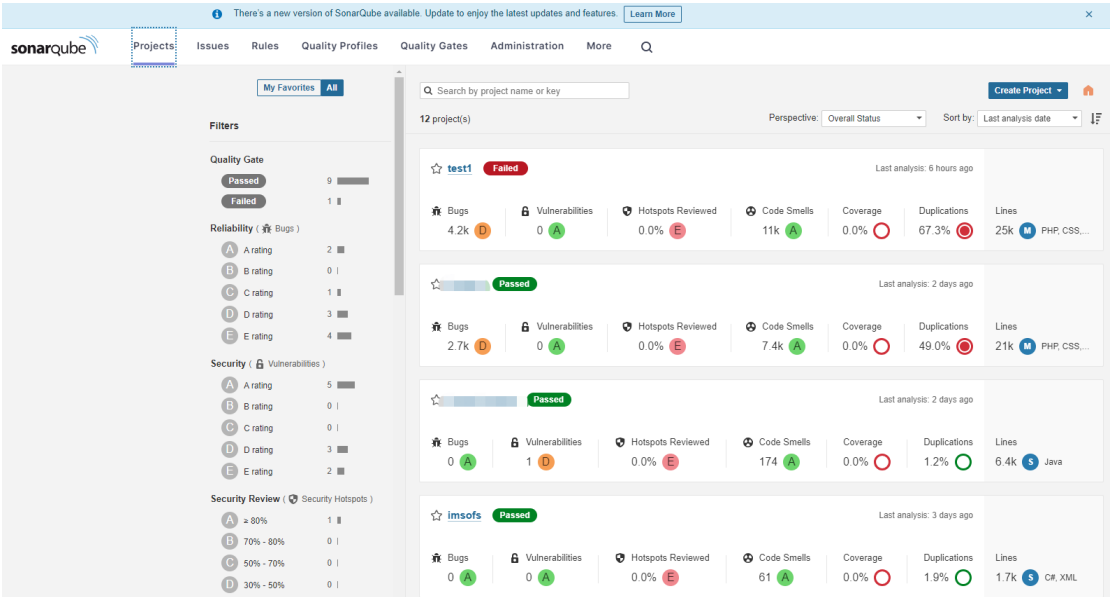

图 23 SonarQube 图形界面模式

2） 命令行模式

SonarQube 支持通过命令行方式完成工具的能使用。

```
D:\>sonar-scanner --help
INFO:
INFO: usage: sonar-scanner [options]
INFO:
INFO: Options:
INFO:  -D,--define <arg>     Define property
INFO:  -h,--help             Display help information
INFO:  -v,--version          Display version information
INFO:  -X,--debug            Produce execution debug output
```

图 24 SonarQube 命令行模式

3） IDE 插件模式

SonarQube 通过 SonarLint 插件可以与 Eclipse、VS Code、Visual Studio、IntelliJ 进行集成。

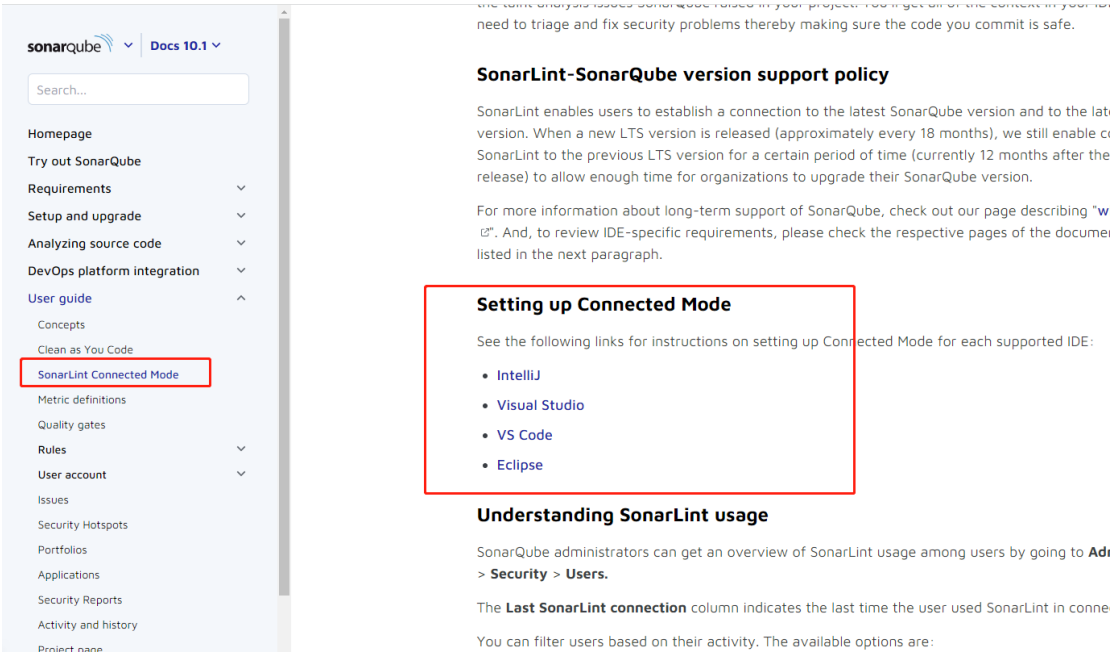

图 25 SonarQube IDE 插件模式

7. 报告输出

所测评的 SonarQube Community Edition 不支持报告输出

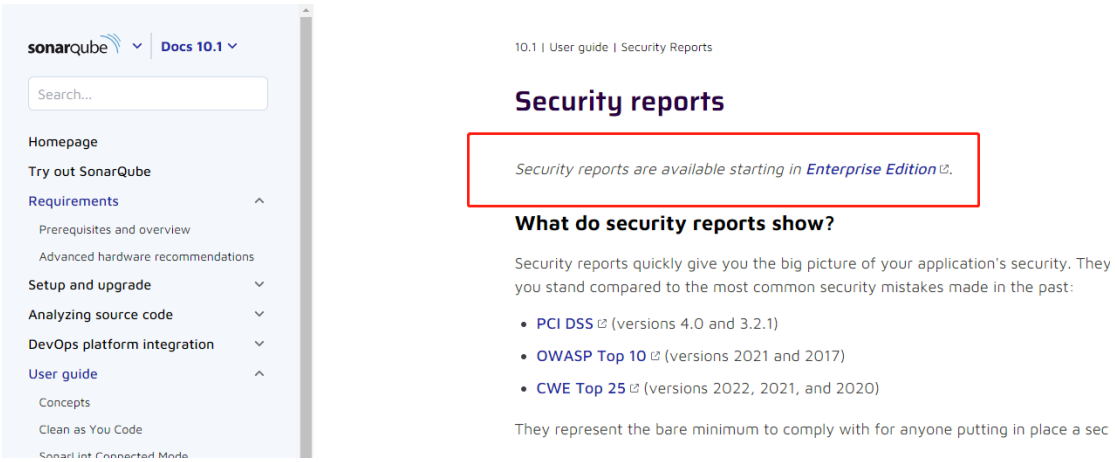

图 26 SonarQube 报告输出功能