
Xcheck
代 码 安 全 检 查

# 目录

# 1 Xcheck在腾讯内部应用的最佳实践

# 产品诞生背景：为DevOps而生的全新自研SAST工具

**开发模式演进**

瀑布式开发 → 敏捷开发 → DevOps开发

**产品定位变化**

安全团队独立使用的审计平台 → 开发团队使用，安全团队维护的CI插件工具

**现有产品难以匹配**

**市面上白盒产品对开发不友好，存在速度慢误报高的通病**

- **速度慢**：扫描速度在几十分钟到数小时，无法适应快速迭代的DevOps开发模式，严重影响流水线自动化效率
- **误报高**：检测报告动辄上百个风险，误报过高，需消耗大量精力去处理，无法作为自动化质量门禁红线

**针对性自研创新，实现内部替换/互补**

coverity　CHECKMARX　FORTIFY　klocwork a Rogue Wave Company

→ XcheckX 代码安全检查

# 腾讯内部落地情况

## 1 已接入的内部DevOps平台

CODING
CLOUD DEVELOPMENT

蓝鲸智云

## 2 分析任务量、接入项目数

**643万**
**次分析任务**

**13.3万**
**接入项目**

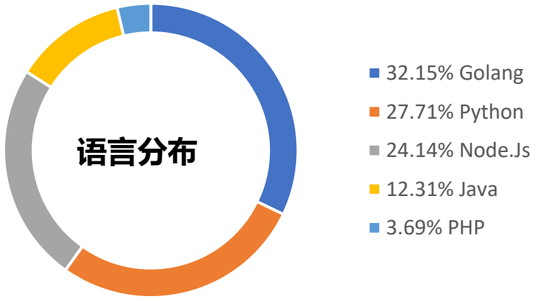
**2020.7~2021.7**

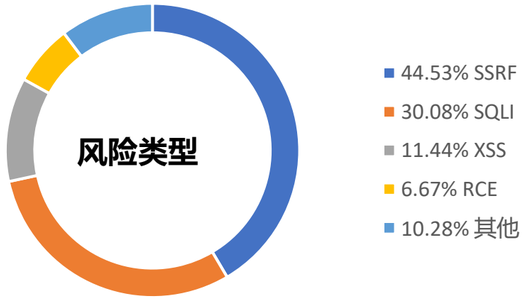
## 3 告警数、修复率、误报数据

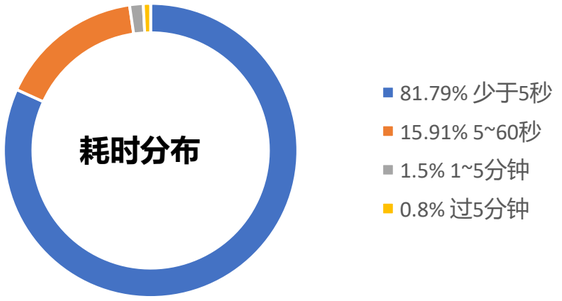
| | 内部项目 |
|---|---|
| 项目数 | 132633 |
| 告警数 | 37678 |
| 已修复 | 10220 |
| 忽略/屏蔽 | 843 |
| 误报率 | 7.62% |

$$\text{误报率} = \frac{\text{忽略} + \text{屏蔽}}{\text{忽略} + \text{屏蔽} + \text{已修复}}$$

## 3 任务语言、风险类别、耗时分布




XcheckX

# 2 Xcheck产品情况介绍

# Xcheck产品定位

## 1 应用安全测试(AST) / Application Security Testing

用于分析和测试应用程序安全漏洞的产品和服务

| # | 分类 | 技术 | 描述 | 代表产品 |
|---|---|---|---|---|
| 1 | **SAST** | **Static AST<br>静态应用安全测试** | **分析应用程序的源码，字节码或二进制代码中是否存在安全漏洞，通常在编码和/或测试阶段进行** | **腾讯Xcheck<br>Fortify/Checkmarx等** |
| 2 | DAST | Dynamic AST<br>动态应用安全测试 | 应用程序处于运行状态，工具发起对应用程序的模拟攻击，分析应用程序反馈，确定是否有安全漏洞 | 腾讯漏洞扫描服务<br>Appscan/AWVS等 |
| 3 | IAST | Interactive AST<br>交互应用安全测试 | 通过代理、VPN或者在服务端部署Agent程序，收集、监控Web应用程序运行时函数执行、数据传输，并与扫描器端进行实时交互，从而发现安全漏洞 | 腾讯IAST<br>OpenRASP等 |
| 4 | SCA | Software composition analysis<br>软件组成分析 | 用于分析应用程序中使用的三方和开源组件的软件成份，发现已知的安全漏洞或其他信息(比如license/敏感信息等) | 腾讯BSCA<br>Black Duck等 |

# Xcheck双检测引擎架构说明

| | 支持语言种类 | 覆盖风险类型 | 适用场景 |
| --- | --- | --- | --- |
| **Xcheck创新引擎<br>（完全自研，独家创新技术路线）** | • 支持7种最常用的后端语言：<br>JAVA、Python、GO、<br>PHP、Node.Js、C、C++<br>• 支持每种语言的常用框架，如有特殊框架需要做适配<br>• 每种语言的都有深度定制的检测算法 | 只支持安全类风险 | DevOps平台流水线插件 |
| **Xcheck传统引擎<br>（对标市面已有产品）** | 支持20+种语言 | 除安全类风险外，<br>还支持代码质量、代码规范及敏感信息等 | 独立审计平台 |

# Xcheck创新引擎详细覆盖范围

## 1 支持Top语言及对应常见Web漏洞

| 语言 | 框架 | 漏洞类型 | 效率 |
| --- | --- | --- | --- |
| **Python** | Django、 Flask<br>Tornado、 Webpy<br>Bottle、 http.server | 命令注入、SQL注入、<br>XSS、XXE、URL跳转、<br>目录穿越、SSRF、<br>反序列化、模板注入 | 千行/秒 |
| **Node.js** | Koa<br>Express | 命令注入、SQL注入、<br>XXE、URL跳转、XPath、<br>目录穿越、SSRF、<br>反序列化、模板注入 | 千行/秒 |
| **Golang** | gin、 Iris<br>net/http、 fastrouter<br>httprouter、 mux<br>go-restful、 fasthttp | 命令注入、SQL注入、<br>XSS、URL跳转、<br>目录穿越、SSRF | 千行/秒 |
| **Java** | Spring<br>HttpServlet<br>websocket | 命令注入、SQL注入、<br>XSS、XXE、URL跳转<br>目录穿越、SSRF、<br>反序列化、SSTI | 千行/秒 |
| **PHP** | Thinkphp<br>Laravel<br>CodeIgniter<br>Yii、 Yaf | 命令注入、SQL注入、<br>XSS、XEE、URL跳转、<br>路径穿越、反序列化、<br>代码执行、变量覆盖、<br>ReDos、phpinfo信息泄露 | 千行/秒 |
| **C/C++** | trpc-c++、 grpc-c++、<br>tars(taf) | 命令注入、SQL注入、<br>URL跳转、目录穿越、<br>SSRF、栈溢出 | 千行/秒 |

## 2 覆盖常见安全类风险

| 常见安全类风险 | | |
| --- | --- | --- |
| 命令注入 | SQL注入 | 模板注入 |
| 反序列化 | 代码执行 | XSS |
| 文件上传 | 文件读取 | 文件删除 |
| XXE | SSRF | URL跳转 |
| XPath注入 | 变量覆盖 | 信息泄露 |

XcheckV

# Xcheck传统引擎详细覆盖范围

代码安全检测 + 代码质量检测 + 代码规范检测 + 敏感信息检测

## 覆盖语言种类及缺陷类型

- 涵盖**20+**种语言
- ABAP/BSP；ActionScript,；Apex；ASP.NET；C# (.NET)；C/C++；Classic ASP (包括 VBScript)；COBOL；ColdFusion CFML；
- HTML；Java(包括 Android) ；JavaScript/AJAX；JSP；MXML (Flex)；Objective C/C++；PHP；PL/SQL；Python；Ruby；Swift；T-SQL；VB.NET；VBScript；Visual Basic；XML；

- 涵盖**2400+**种缺陷类型
- 风险类型参考来源CWE、OWASP、SANS、PCI DSS、STIG、NIST等

# Xcheck接入方式及应用场景

1 主动扫描：通过插件方式嵌入CI/CD，默认触发扫描

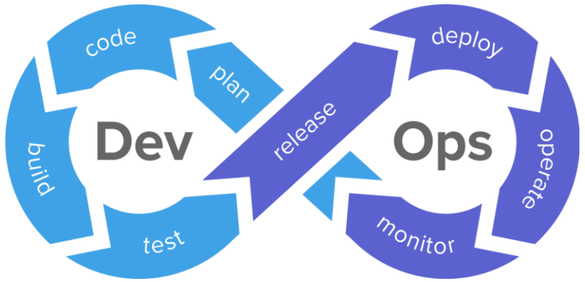
2 被动扫描：针对代码仓库进行全量扫描

3 本地扫描：人工上传代码压缩包进行检测审计

编码 阶段 ✓

构建 阶段 ✓

测试 阶段 ✓




**支持集成的DevOps平台**

GitLab　Jenkins　CODING CLOUD DEVELOPMENT　蓝鲸智云 BLUEKING

?

开放API接口
可集成到其他平台


Web/API

XcheK

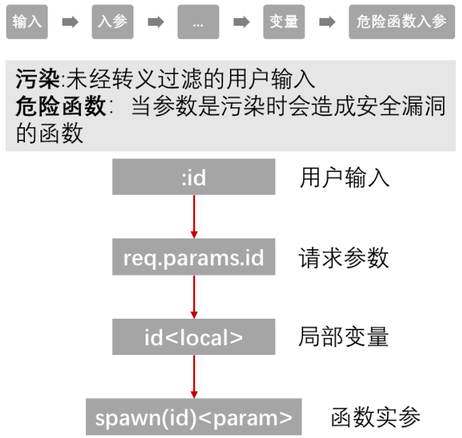
# 3 Xcheck技术创新及产品优势

# 产品核心检测指标对比

| | Xcheck | 市面上其他SAST工具 |
| --- | --- | --- |
| **检测耗时** | **每秒千行级到万行代码**<br>95%的项目1分钟之内扫描完成 | **每秒几十行级到百行代码**<br>动辄几十分钟、数小时 |
| **误报率** | 高置信度分析能力<br>创新引擎误报率**低于10%** | 高误报为市面上白盒工具的普遍痛点<br>误报率在**50%**左右 |
| **自定义规则** | 支持用户便捷快速地自定义规则 | 基本都有<br>但创建和运营较为复杂 |
| **检出能力** | 全面支持语言的特性<br>支持丰富的框架工具库知识<br>支持过程间分析 | 部分特性不支持<br>支持有限的框架<br>过程内分析 |

# 技术创新及产品优势

## 1 检测原理：语义解析+污点追踪




```
const express = require('express');
const cp = require('child_process');
const server = express();

server.get('/abc/:id',(req,res) => {    入参
    const cp = require('child_process');
    res.send(req.params.id)   传递到res.send函数
    var id = req.params.id    传递到局部变量id
    cp.spawn(id)   传递到cp.spawn函数
});
server.listen(3000);
```

## 2 核心技术创新

| 核心技术 | 优势 | 效果 |
| --- | --- | --- |
| **纯自研语义分析算法** | 无需编译<br>准确“理解”代码<br>快速分析 | 精确识别各种语言特性<br>秒级的扫描速度 |
| **精细化的模型设计** | 更精细化的污点传播<br>精确识别风险类型 | 污点不会被放大或消失低误报低漏报 |
| **灵活强大的扩展** | 易编写<br>可调试 | 支持自定义危险函数，批量降低漏报<br>支持自定义过滤函数，批量降低误报 |

## 3 以Java语言为例，独家优势能力介绍

### 1. 类变量识别

精确解析model类，map，JSONObject等结构对象，而不是把他们当成一个整体来处理。例如：将客户端输入反序列化为User类对象时，会细化至User类的变量，所以污点传递至user对象的name变量，而age变量是Integer类型，不受影响

### 2. 识别多态和重载

①能够识别继承关系 ②能够找到正确的重载函数 ③对Map进行精确识别

### 3. Java反射支持

例如当通过Java反射调用IndexService.runCommand方法，并将客户端传入的参数作为方法参数传递时，检查器可以理解这种灵活的语义，污染可以正确传递。

### 4. 防护识别

支持检测各类代码修复手段，代码修复后，漏洞不会再被检查出来，降低误报率。

XcheckX

# 产品具有挖掘0Day漏洞的能力

## 1 扫描外网开源项目和CNVD编号

批量扫描开源项目 → 人工确认漏洞 → 提交漏洞 → 获得编号

| 项目名称 | 漏洞类型 | 危害级别 | 影响版本 | CNVD编号 |
| --- | --- | --- | --- | --- |
| **ThinkAdmin** | 任意命令执行 | 高危 | V4/V5/V6s | CNVD-2020-33163 |
| **Vtiger CRM** | SQL注入 | 中危 | 7.2.0 | CNVD-2020-32231 |
| **信呼OA办公系统** | SQL注入 | 中危 | 2.1.3 | CNVD-2020-45121 |
| **苹果CMS** | 任意文件删除 | 低危 | v10 | CNVD-2020-47656 |
| **ECShop** | SQL注入 | 中危 | 4.1.0 | CNVD-2020-58823 |
| **Jeecg-Boot** | SQL注入 | 中危 | v2.3 | CNVD-2020-59430<br>CNVD-2020-59429 |
| **禅道** | 任意文件下载 | 高危 | v12.4.3 | CNVD-2020-65242 |
| **若依管理系统** | SQL注入 | 中危 | 4.6.1 | CNVD-2021-36506 |
| **MCMS** | SQL注入 | 高危 | 5.1 | CNVD-2021-37317 |

XcheckX

# Thanks!