Tencent 腾讯 | 腾讯云

腾讯云

腾讯云代码分析

Tencent Cloud Code Analysis

Version：20221031

目录

01 产品介绍

01 Product description

十年技术锤炼，打造“代码分析+跟踪管理”高性能平台

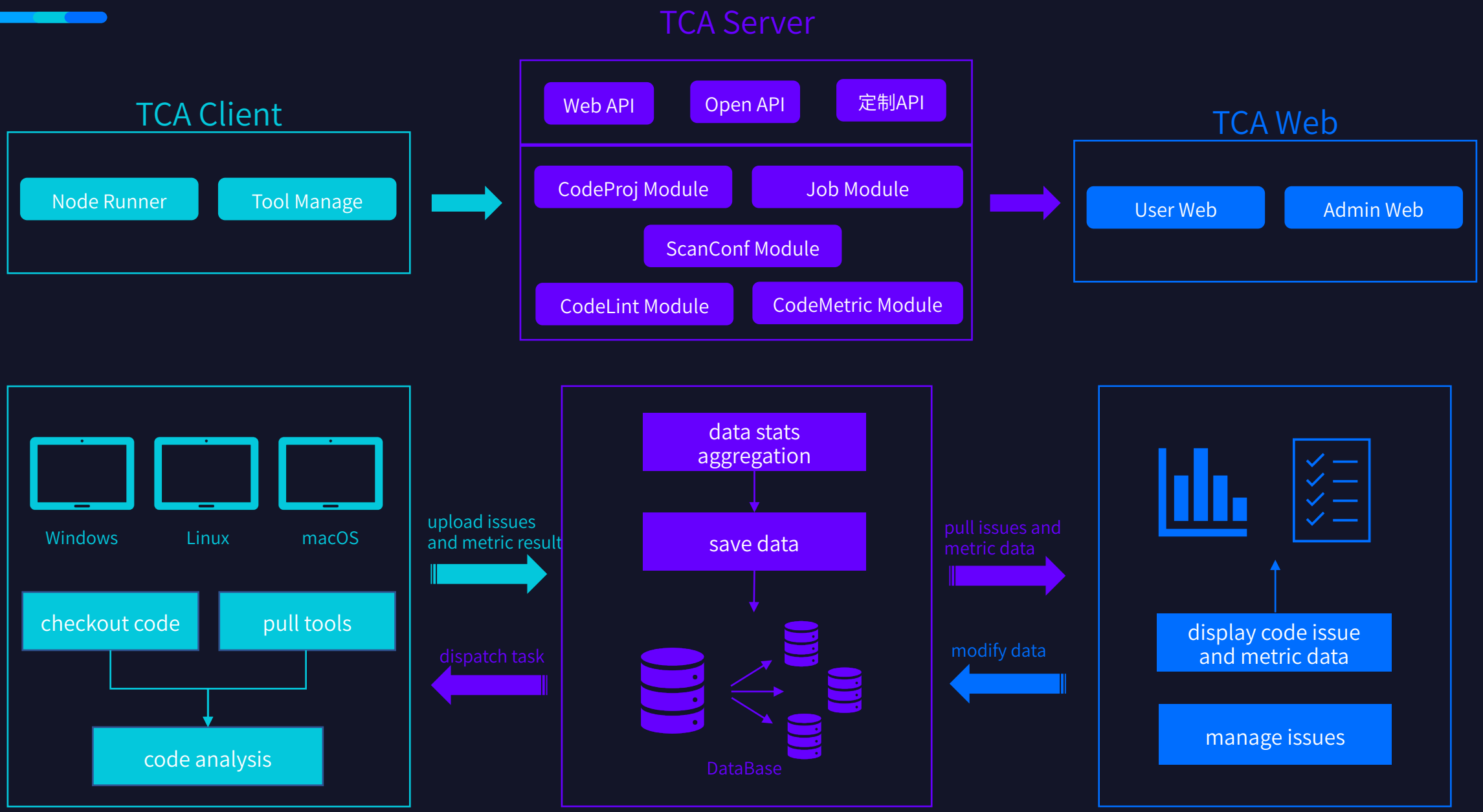
Tencent 腾讯 | 腾讯云

- 腾讯云代码分析立项于 2013 年，是集众多分析工具的云原生、分布式、高性能的代码综合分析跟踪平台。
- 代码分析旨在运用词法分析、语法分析、控制流、数据流分析等的技术，对代码进行综合分析，查找代码中的规范性、结构性、安全漏洞等问题，进而输出代码的全方位质量报告，帮助项目持续监控项目代码质量。
- 腾讯云代码分析帮助项目实现“测试左移”，从而尽早以低成本、高效率发现代码问题，减少修复成本，缩短修复时间。

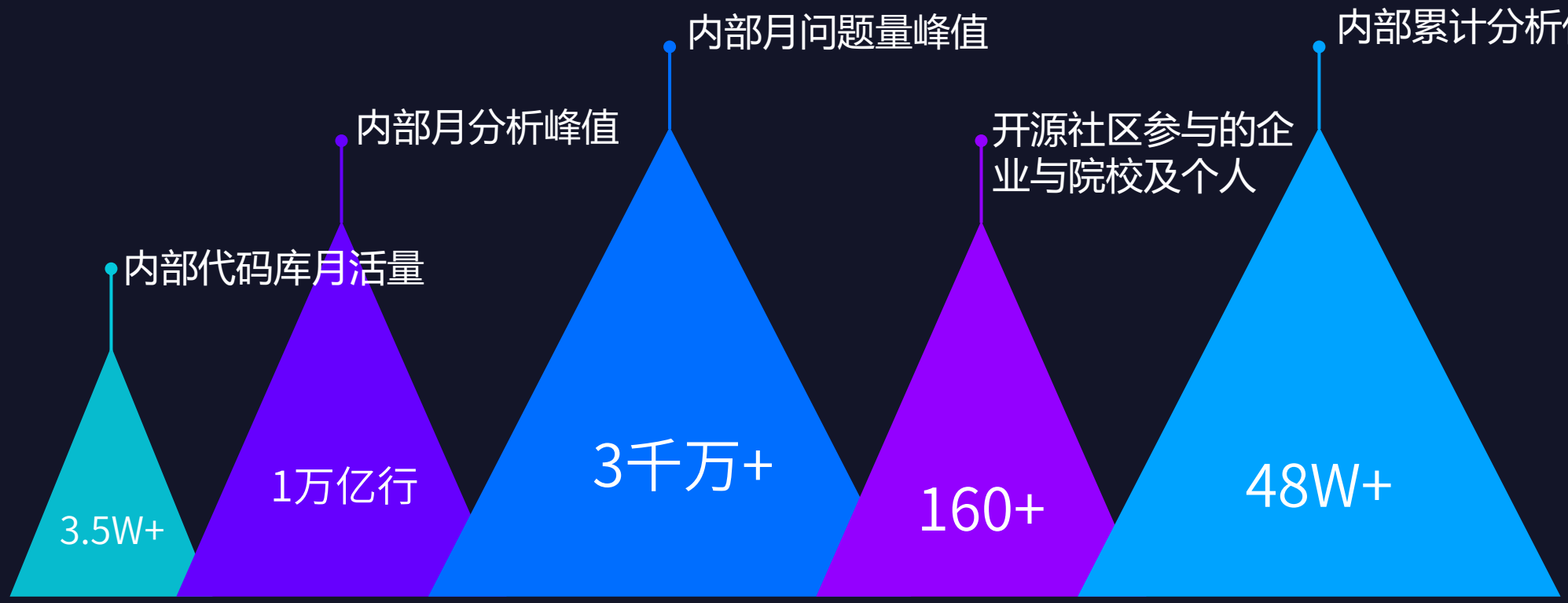
“在实施环节中修复错误，比在设计环节中修复错误的成本高出六倍。测试成本会增加15倍，部署成本会高至100倍。”

“用心关注每行代码迭代，助力传承卓越代码文化”。

分层架构，快速部署，高效分析



内外服务数据



- 腾讯云代码分析自立项以来，在腾讯内部已迭代至 6.0 版本，实践了 48W+ 代码库分析工作，为腾讯云、腾讯会议、手机 QQ、微视、QQ 音乐等明星产品迭代提供了有力支持。
- 对外与 CODING Devops、腾讯云官网、腾讯云移动金融开放平台保持战略合作，助力更多外部企业提升研发质量。
- 在 2021 年底，腾讯云代码分析正式在开源至 GitHub，至今已有 160+ 企业/院校/组织/个人加入开源社区（开源地址：https://github.com/Tencent/CodeAnalysis）。

* 数据统计于 2022 年 10 月

服务客户

Tencent 腾讯 | 腾讯云

中国银联

腾讯云

腾讯会议

腾讯地图

腾讯企点

腾讯电脑管家

腾讯文档

腾讯课堂

QQ

腾讯智慧出行

腾讯视频

企业微信

腾讯乘车码

腾讯医疗

QQ音乐

战略合作：

CODING CLOUD DEVELOPMENT

移动金融开发平台

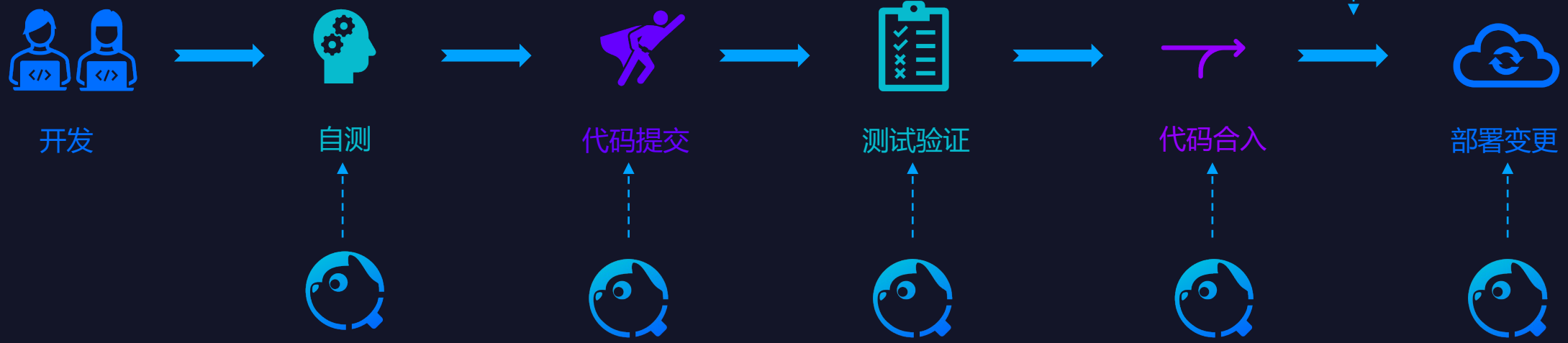
02 应用场景

02 Application scenarios

应用场景

Tencent 腾讯 | 腾讯云

开发全流程可接入，代码缺陷无处可藏。



应用场景

Tencent 腾讯 | 腾讯云

代码安全

注入型漏洞 URL
重定向漏洞

代码质量

数组越界
空指针解引用

代码异味

圈复杂度　重复
代码 过长参数

专项提升

iOS减包
权限调用扫描

Tencent 腾讯 | 腾讯云

代码安全

- 支持对OWASP Top10 中常见的漏洞进行分析，包括SQL注入、XML注入、外部实体注入攻击、敏感信息泄漏、URL重定向漏洞等，并结合CWE中常见漏洞，比如服务端请求伪造漏洞、服务器模板注入漏洞等，进行专项安全漏洞分析，准确识别漏洞所在位置并提供修复建议。

规则名称	规则概要
xss	跨站脚本攻击
ssrf	服务端请求伪造
sql	sql注入
xml	xml注入
reflectioni	反射型注入
ldap_injection	LDAP注入

代码质量

- 数组越界（AOB）和空指针引用（NPD）等这类问题对软件稳定性、代码可靠性影响巨大，但在编码期间很难被检测到。 而普通的代码 走查方式成本高、有效性差，且不易跟踪管理。

- 腾讯云代码分析支持识别潜在漏洞，帮助开发分析和解决代码缺陷，减少代码走查测试成本，提高软件可靠性、健壮性。

func_ret_null

func_ret_null 函数返回值可能为nullpointer，但是调用该函数时指针未经判空便进行使用
在选用func_ret_null_full 时, 检查器会在项目内全局搜索空指针函数的调用情况，否则只会在相关文件内进行检查。

代码示例

以下提供一个或多个func_ret_null代码案例

在下面代码中 `test` 函数中调用 `get_name` 可能返回空指针，在后续使用 `name` 指针前应该判断是否为空指针

```
// name.hpp

char* get_name(int id) {
    char* name = 0;
    if (id == 1) {
        name = "Zeus";
    } else if (id == 2) {
        name = "Hades"
    } else {
        return nullpointer;
    }
    return name;
}

void test(int i) {
    char* name = get_name(i);
```

空指针引用

array_overflow

array_overflow 检查数组越界的情况。不正确的缓存区访问可能损坏内存，导致程序崩溃或读取到权限外的内存。

代码示例

以下提供一个或多个array_overflow案例

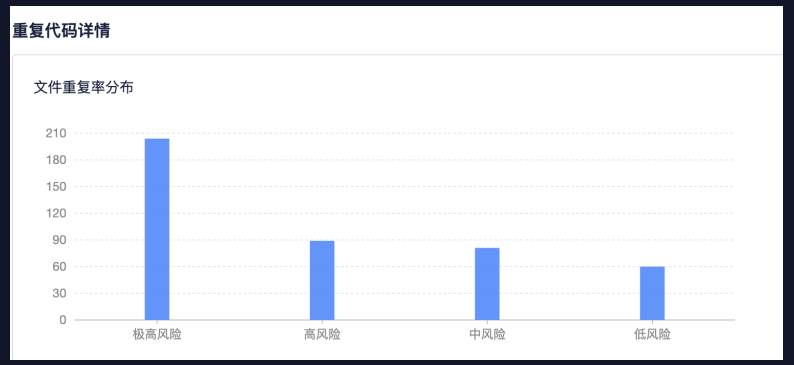
```
void foo() {
    int array[10];
    int i = get();
    // i = 9;
    if (i > 8 && i <= length(array)) {  // Shoud be i < length(array)
        array[i] = 1;  // defect: array[10] overflow
    }
    array[i] = 1;  // defect: array[10] overflow
}


void test(int i) {
    int n= 10;
    char *p = malloc(sizeof(int) * 10);
    int y = n;
    p[y] = 'a'; // defect: writing to buffer[y] overflow
}
```

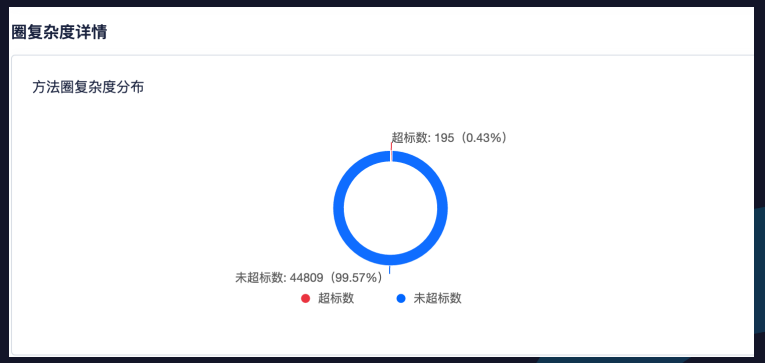
数组越界

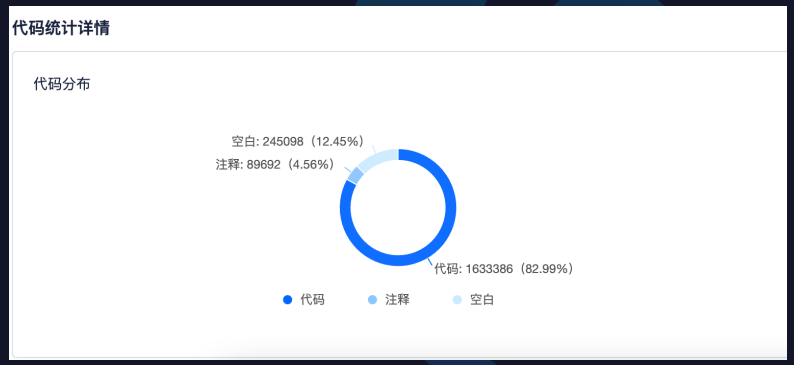
代码异味

- “代码异味是一种表象，它通常对应于系统中更深层次的问题。”如果程序没有用一种好的表达方式来表现，那程序会很难阅读， 难维护，难修改。

- 通过针对圈复杂度、重复代码、过长方法、过长参数列表等多类型代码异味扫描，将代码异味可视化，协助开发者更便捷地重构代码，提升代码的可读性、可维护性。写取悦自己、让他人仰慕的代码。







专项提升

- iOS审核、iOS减包、Android动态权限调用扫描、Android危险权限扫描、Android减包等多类专项扫描，有针对性进行代码问题分析。助力快速通过审核，避免公关危机等。

【Java】Android动态权限调用扫描

- 动态权限API检测 – TwilightManager
- 动态权限API检测 – WallpaperManager
- 动态权限API检测 – TelephonyManager
- ……

【Object-C】减包扫描

- 扫描未使用的变量
- 扫描未使用的方法
- 发现未使用的图片文件
- ……

【Java】Android危险权限扫描

- 文件共享权限扫描
- 文件读写模式扫描
- 通讯录API扫描
- ……

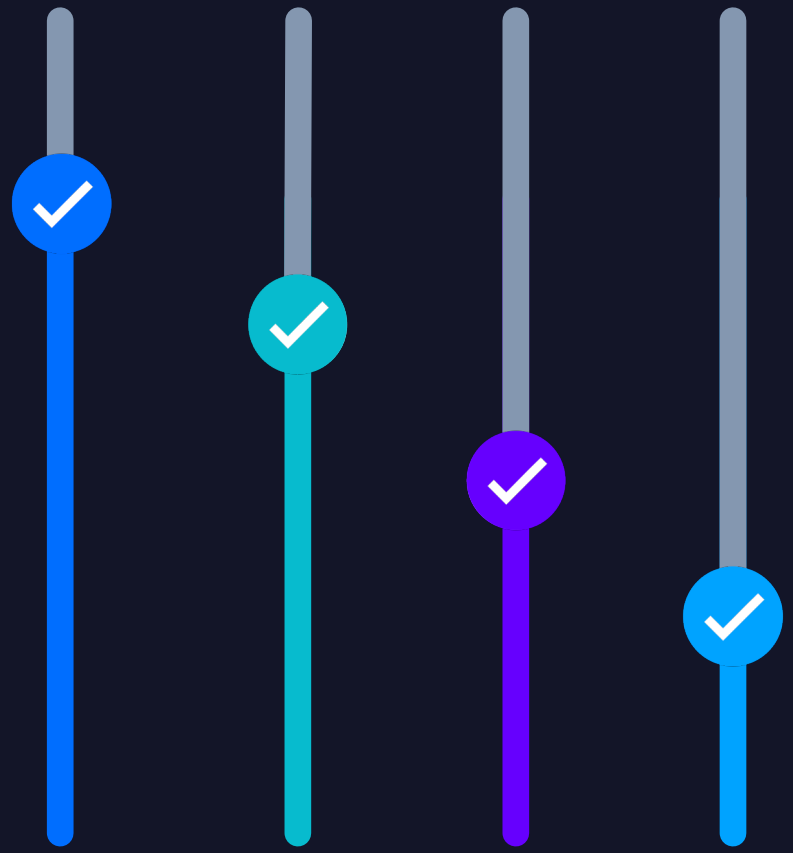
iOS隐私合规检查

- 扫描疑似私有API
- 苹果审核-不得强制安装其他软件
- 苹果审核-不得占位未实现功能
- ……

03 产品特性及优势

03 Product features and benefits

稳定可靠的架构



- 云原生微服务架构，支持资源弹性调度
- 分布式客户端模式，自适应优化分析效率
- 服务分层设计，支持灵活扩展适配
- 数据高效存储，支持大规模并发分析

Tencent 腾讯 | 腾讯云

标准化API接口，灵活融入Devops

- 支持对接业内常见Git/SVN仓库.
- 标准化API全开放，可快速对接主流CI、DevOps平台.
- 支持 GitHub action，Jenkins plugin ，快速接入。

分布式客户端

- 客户端可以作为常驻进程，启动一个分析节点。
- 多个分析节点可以分布式并行执行服务端下发的分析任务，提高扫描效率。
- 分布式的客户端支持的环境含 Linux、Mac、Windows，满足用户高频分析场景。

Tencent 腾讯 | 腾讯云

多工具支持，工具规则可扩展

- 已支持近 200 款工具，1000+分析规则，让各种类型代码都可以轻松扫描。
- 可扩展的用户自定义工具、规则引擎：支持用户自定义工具规则，可以针对自身业务定义业务逻辑规则；可集成自研工具、商业工具，满足项目需要。

多种语言支持

- 覆盖业内主流 33 门语言（部分工具不受语种限制）。
- 支持自动识别语言。

C/C++	C#	Css	Go	Html	JavaScript	T-SQL	Protocol Buffers
Java	Kotlin	Lua	Objective-C	PHP	Python	XML	Rust
Ruby	Scala	Swift	TypeScript	Visual Basic	ABAP	Dart	SQL
Apex	COBOL	Flex	PL/I	PL/SQL	RPG	Shell	WebAssembly

ThinkPHP	Django	Testing	Testify	React	Bottle	Vue	.....

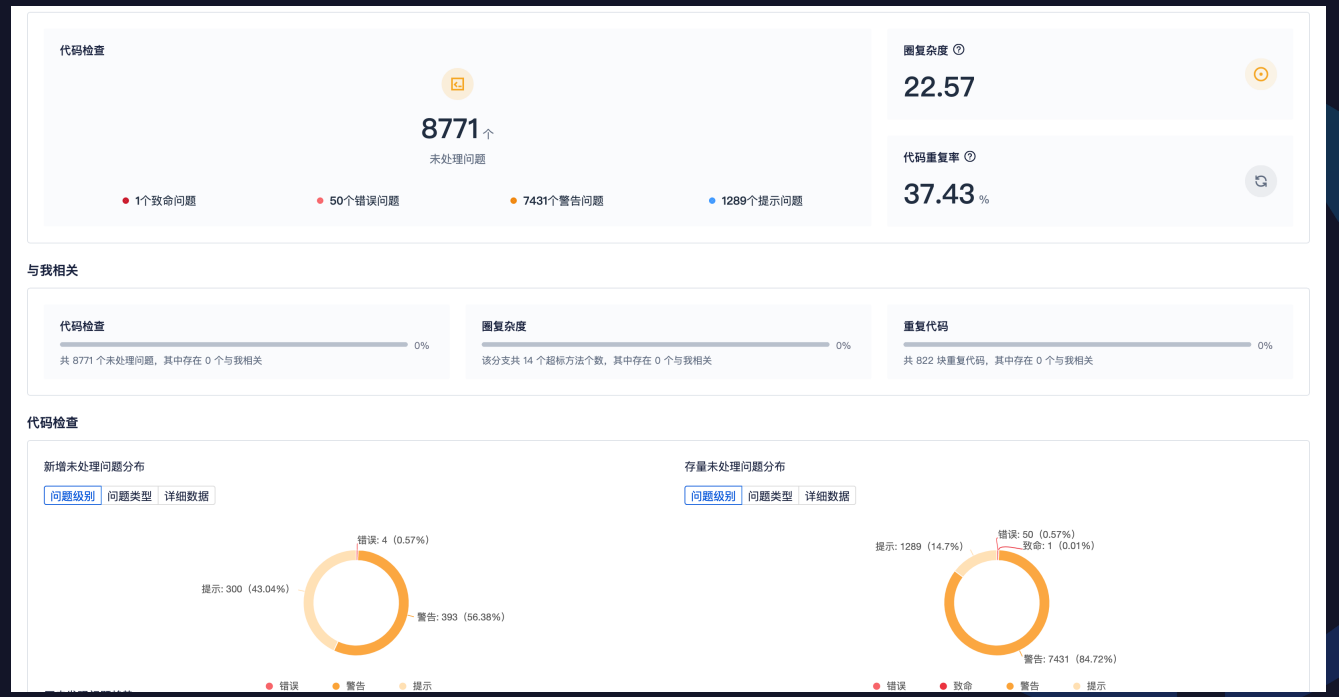
增量全量分析

- 增量全量分析互相配合，满足研发全流程各环节的不同要求，兼顾研发效能和研发质量。
- 增量分析配合高性能工具执行引擎和高效任务调度逻辑， 助力快速获得扫描结果，无需漫长等待，提升研发效能。

Tencent 腾讯 | 腾讯云

全方位质量报告

- 图形化问题分布报告，多维度展示代码库最新健康状态。
- 历史趋势分析视图，轻松监管代码质量趋势。
- 本地离线报告，便捷查阅代码质量概览。

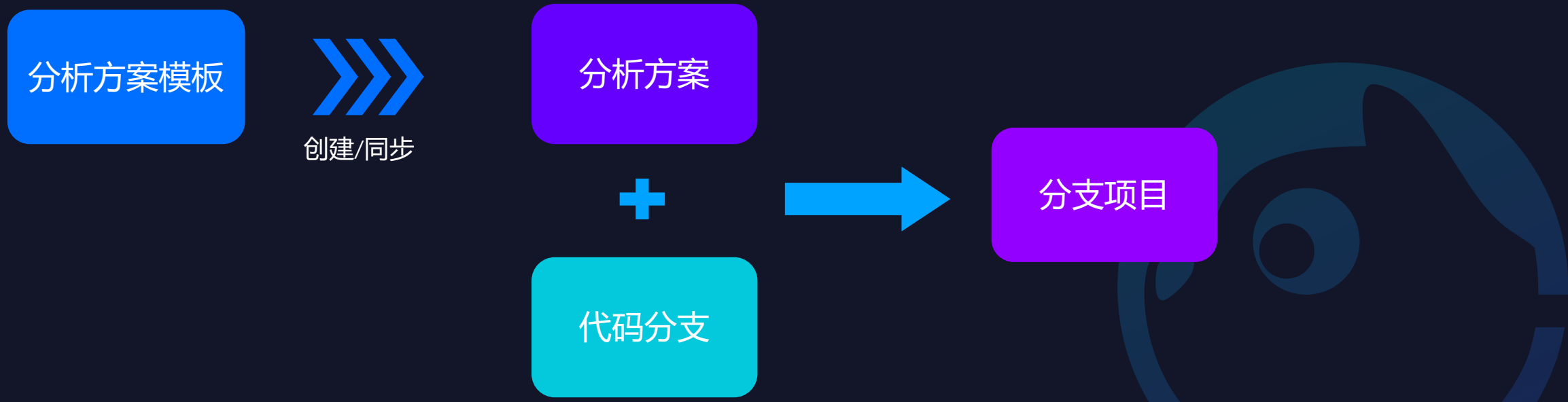


持续跟踪问题

- 问题跟踪和处理：
  - 持续跟踪管理问题：根据最新扫描结果自动关闭已修复问题。
  - 支持主动标记处理问题，如无需处理的问题、误报的问题。
  - everything as code（XaC），支持code.yml设置文件责任人。
- 问题过滤：
  - 支持按照指定分支过滤，轻松配合合流等常见研发场景。
  - 支持使用代码注释过滤问题
  - code.yml过滤文件
  - 可选过滤LFS文件或submodule相关问题。
  - 支持设置质量门禁，为开发过程设置锚点。

分析方案模板

- 结合腾讯多年实践经验总结出安全、基础、规范等不同类型的分析方案模版，开箱即用，减少对规则和功能的学习、配置成本。
- 可根据自身开发情况自定义分析方案模版。灵活配置规则及其参数、度量指标，帮助团队逐步提升代码质量。

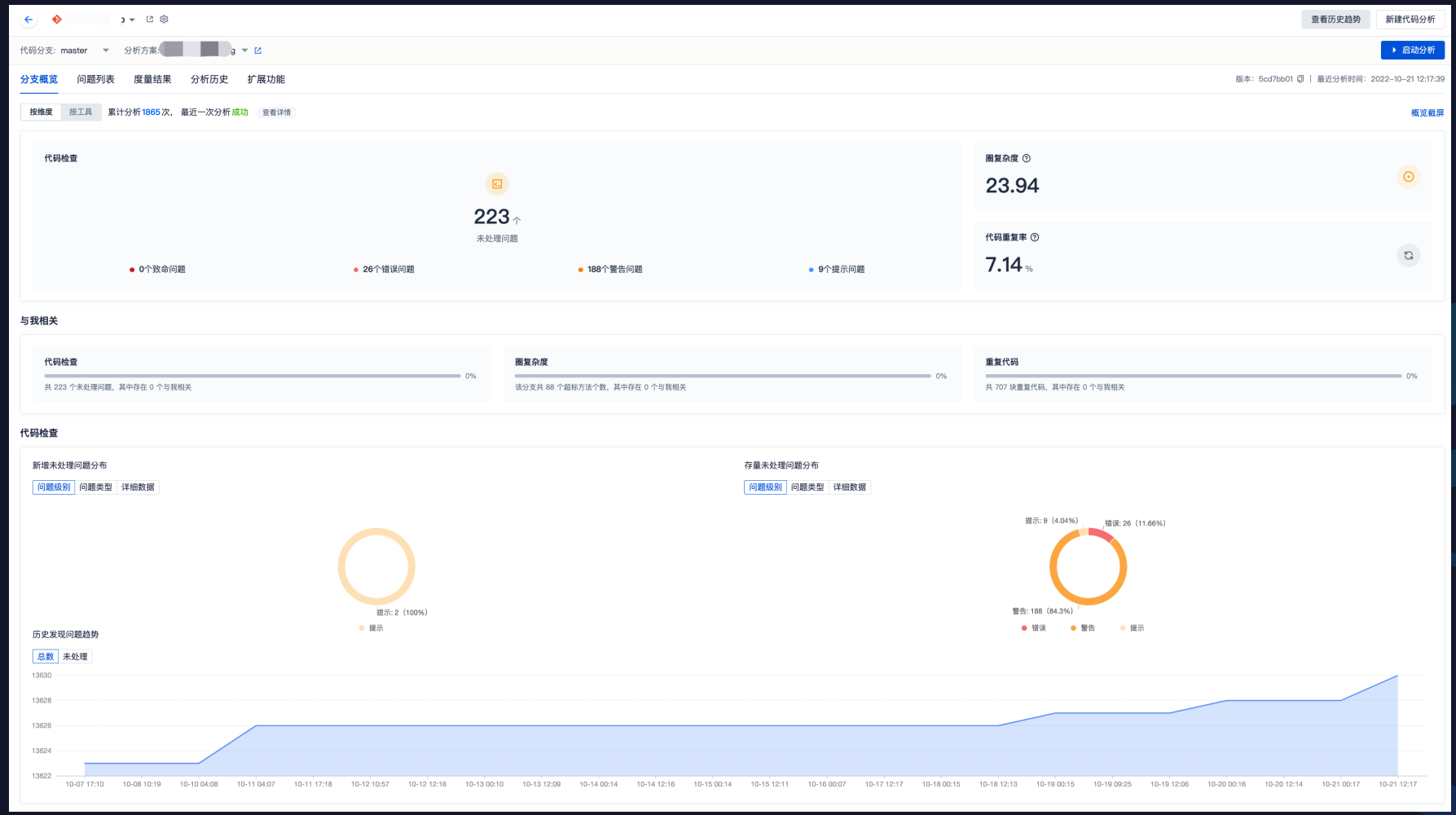


Tencent 腾讯 | 腾讯云

04 页面效果展示

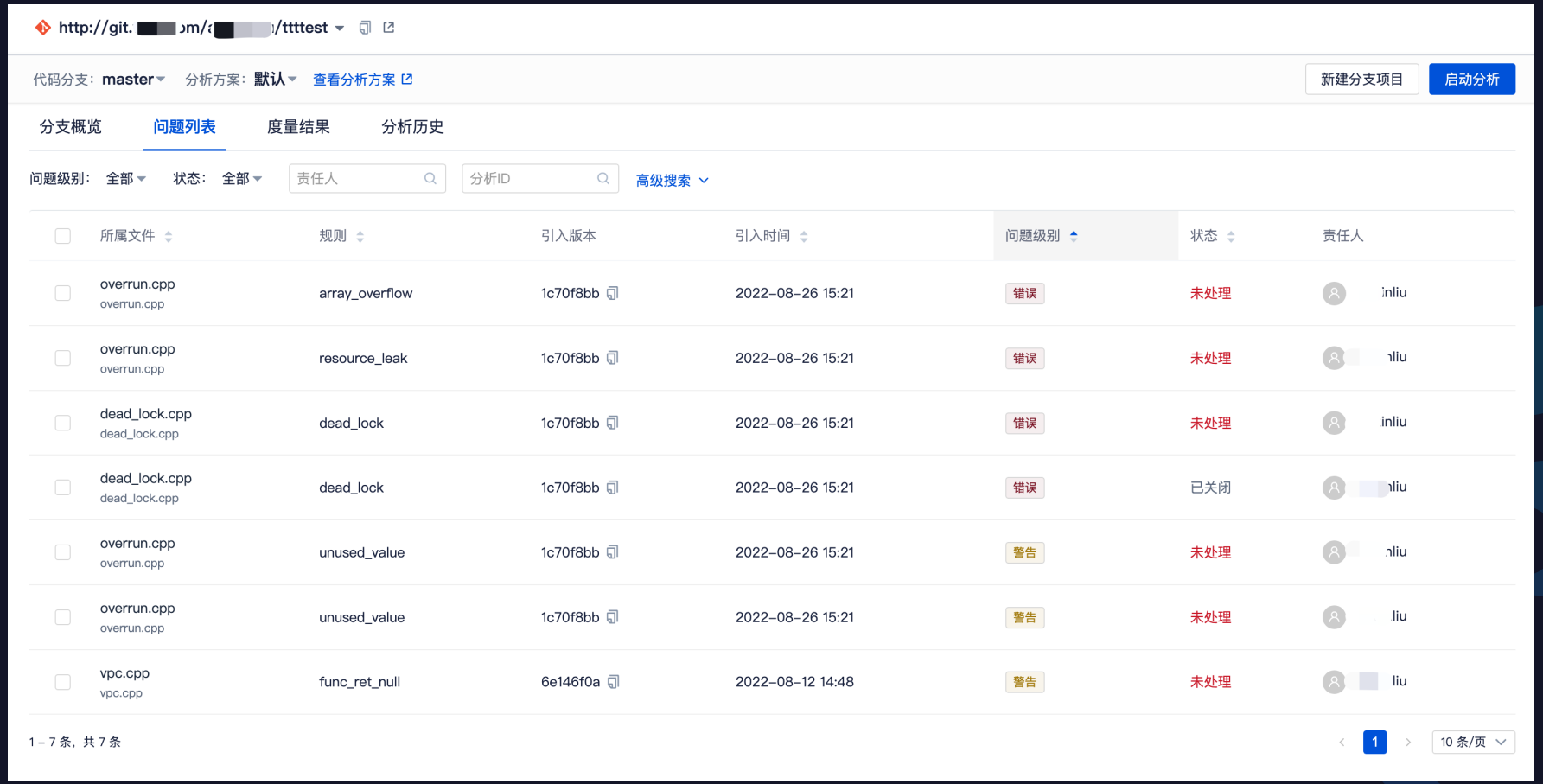
04 UI pages

分析结果概览


查看历史趋势
新建代码分析
代码分支: master
分析方案:
启动分析
分支概览
问题列表
度量结果
分析历史
扩展功能
版本: 5cd7bb01 | 最近分析时间: 2022-10-21 12:17:39
按维度
按工具
累计分析1865次，最近一次分析成功
查看详情
概览截屏
代码检查
223 个
未处理问题
0个致命问题
26个错误问题
188个警告问题
9个提示问题
圈复杂度
23.94
代码重复率
7.14 %
与我相关
代码检查
共 223 个未处理问题，其中存在 0 个与我相关
圈复杂度
该分支共 88 个超标方法个数，其中存在 0 个与我相关
重复代码
共 707 块重复代码，其中存在 0 个与我相关
代码检查
新增未处理问题分布
问题级别
问题类型
详细数据
提示: 2（100%）
提示
存量未处理问题分布
问题级别
问题类型
详细数据
提示: 9（4.04%）
错误: 26（11.66%）
警告: 188（84.3%）
错误
警告
提示
历史发现问题趋势
总数
未处理
13630
13628
13626
13624
13622
10-07 17:10
10-08 10:19
10-10 04:08
10-11 04:07
10-11 17:18
10-12 10:57
10-12 12:16
10-13 00:10
10-13 12:09
10-14 00:14
10-14 12:16
10-15 00:14
10-15 12:11
10-16 00:07
10-17 12:17
10-18 00:15
10-18 12:13
10-19 00:15
10-19 09:25
10-19 12:06
10-20 00:16
10-20 12:14
10-21 00:17
10-21 12:17

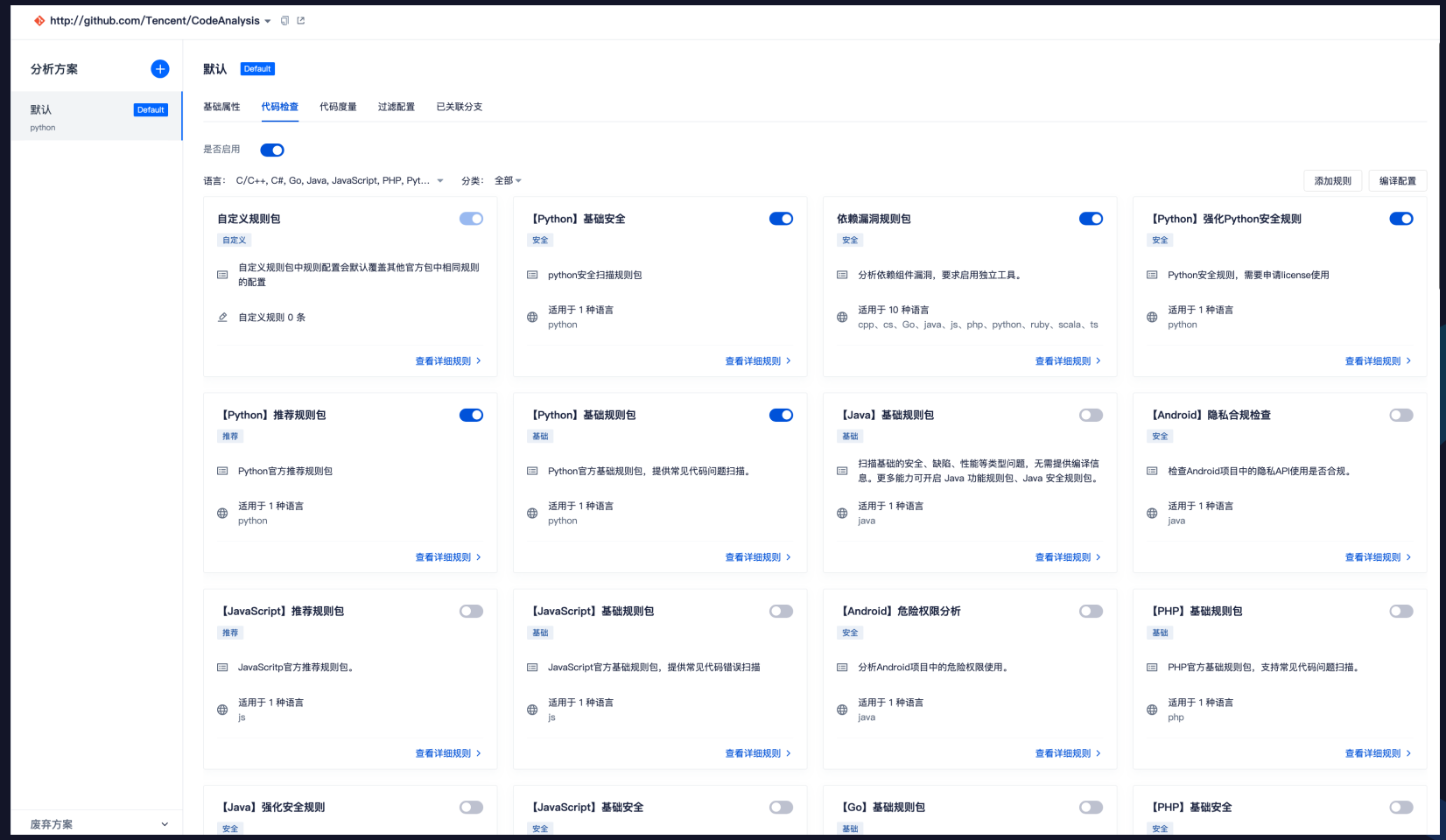
Tencent 腾讯 | 腾讯云

问题列表



Tencent 腾讯 | 腾讯云

代码检查规则配置



05 典型案例展示

04 Typical cases

典型案例展示

强化安全漏洞分析

Java 强化安全规则包

CmdInject

★★★★★

- 检查代码中是否存在命令行注入漏洞。
- 当使用 childprocess 等模块执行命令时，拼接了用户可控的输入，会导致命令执行漏洞。攻击者利用漏洞可以控制目标主机或者容器。
- 修复建议：需要评估 childprocess 等模块执行命令的使用，应限定或校验命令和参数的内容。

```java
void bad(HttpServletRequest req, HttpServletResponse resp){
    String cmd = req.getParameter("cmd");
    Runtime rt = Runtime.getRuntime();
    rt.exec(cmd); // 触发规则
}
```

PathTraversal

★★★★

- 检查代码中是否存在路径穿越漏洞。
- 操作文件时，应该限定文件的路径范围，如果拼接用户输入到文件路径，可能导致路径穿越漏洞。攻击者利用漏洞可以访问到文件系统上的任意文件，这可能导致信息泄漏等问题。
- 修复建议：按业务需求，使用白名单限定后缀范围，校验并限定文件路径范围。

```java
void bad(HttpServletRequest req, HttpServletResponse resp){
    String image = req.getParameter("image");
    File file = new File("resources/images/", image); // 触发规则

    if (!file.exists()) {
        return Response.status(Status.NOT_FOUND).build();
    }

    return Response.ok().entity(new FileInputStream(file)).build();
}
```

SQLInject

★★★★★

- SQLInject 规则用于检查代码中是否存在SQL注入漏洞。
- 错误的拼接用户可控的值到 SQL 语句，可能导致 SQL 注入漏洞。攻击者可以修改 sql 语法来更改查询的目标或结果，泄露数据库敏感信息，也可以使用 SQL 文件操作攻击底层Web服务器。如果使用该 SQL 查询进行授权认证，攻击者还可以用于提权。
- 修复建议：SQL 语句默认使用预编译并绑定变量，使用安全的ORM操作。

```java
void bad(HttpServletRequest req, HttpServletResponse resp){
    String id = req.getParameter("id");
    Connection conn = null;
    Statement statement = null;
    ResultSet rs = null;

    Class.forName("com.mysql.cj.jdbc.Driver");
    conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/sec_sql", "roo
    String sql = "select * from userinfo where id = " + id;
    statement = conn.createStatement();
    statement.executeUpdate(sql); // 触发规则
}
```

SSRF

★★★★

- 检查代码中是否存在服务端请求伪造漏洞 SSRF(Server-side request forgery) 。
- 攻击者在未能取得服务器所有权限时，利用服务器漏洞以服务器的身份发送一条构造好的请求给服务器所在内网。
- 修复建议：限定访问网络资源地址范围，请求网络资源应加密传输。

```java
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebS
import org.springframework.security.config.annotation.web.configuration.WebSecurit

@EnableWebSecurity
@Configuration
public class WebSecurityConfig extends WebSecurityConfigurerAdapter {
  @Override
  protected void configure(HttpSecurity http) throws Exception {
    http
      .csrf(csrf ->
        csrf.disable() // 触发规则
      );
  }
}
```

XSS

★★★★

- 检查代码中是否存在跨站脚本攻击漏洞 XSS(Cross-site scripting) 。
- 如果 WEB 页面在动态展示数据时使用了用户的输入内容，没有对输入的内容过滤或者进行转义，黑客可以通过参数传入恶意代码，当用户浏览该页面时恶意代码会被执行。
- 修复建议：在输出所有用户可控的数据时，对数据做转义或者编码。

```java
void bad(HttpServletRequest req, HttpServletResponse resp){
    String id = request.getParameter("id") != null ? request.getParameter("id") :
    Doc doc = getdetailsById(id);
    byte[] b = doc.getUploaded();
    try {
        response.setContentType("APPLICATION/OCTET-STREAM");
        String disHeader = "Attachment;Filename=" + doc.getName();
        response.setHeader("Content-Disposition", disHeader);
        ServletOutputStream out = response.getOutputStream();
        out.print(b); // 触发规则
    }
}
```

更多详情参考：【Java】强化安全规则包

代码质量缺陷分析

- 检查数组越界的情况。不正确的缓存区访问可能损坏内存，导致程序崩溃或读取到权限外的内存。
- 检查 strcpy, strcat 字符串复制/拼接过程中长度不当导致的溢出， 同样 gets scanf 函数也视为不安全。
- 如果发现多线程中某个全局变量在未持有锁便更新时，则会上报错误。
- 如果发现文件内存在 mtx1 -> mtx2 的上锁顺序时，另存在 mtx2 -> mtx1 的上锁顺序，视为死锁或存在死锁的可能，则会上报错误。 死锁发生时程序将会卡死无法正常执行。
- 在程序申请了资源但并未按时释放时上报错误 目前场景包括：句柄打开时未关闭，指针分配内存后没有及时释放。
- .....

更多详情参考：【C/C++】代码质量缺陷规则包

本地快速扫描

使用场景

本地开发过程中，可以对本地代码目录下的临时代码（未关联scm仓库或未提交到scm仓库的本地代码）进行扫描，对某个目录或某些文件进行快速扫描，产出本地扫描结果。

注：该模式不与代码仓库关联，只对给定的目录或文件进行扫描，不依据版本号做增量分析，也不定位问题责任人。

使用步骤

1. 在页面上创建分析方案模板，获取分析方案模板ID

   该模式不与代码仓库绑定，因此不能直接使用分析方案，需使用分析方案模板，根据模板链接获取模板ID。

2. 初始化扫描需要的工具

   进入客户端client目录，执行 quickinit 指令：

   python3 codepuppy.py quickinit -t TOKEN --scheme-template-id SCHEME_TEMPLATE_ID --org-sid ORG_SID

3. 执行快速扫描

   进入客户端client目录，执行 quickscan 指令：

   python3 codepuppy.py quickscan -t TOKEN --scheme-template-id SCHEME_TEMPLATE_ID --org-sid ORG_SID -s SOURCE_DIR --file FILE

详细指南请参考：快速扫描模式

在Jenkins中使用代码分析

1. 获取代码分析的Jenkins插件
2. 在TCA中创建团队和项目
3. 在Jenkins中上传插件文件，以安装代码分析插件
4. 创建Jenkins任务，配置需要代码库地址和下载凭证
5. 配置Jenkins插件相关参数
    - 在构建任务配置中选择【TCA】插件，配置项目代码路径、TCA团队ID、TCA项目名称、个人token、分支名称、分析语言、方案名称等参数。
6. 启动构建并查看分析结果
    - 点击【Build Now】启动构建任务，在【Console Output】中可以查看控制台执行过程，执行完成后，可在下方看到分析结果的链接。

详细使用说明请参考：Jenkins插件使用说明

结束页

Tencent 腾讯 | 腾讯云

欢迎您的咨询

技术支持：tca@tencent.com

商务合作：tca@tencent.com