58同城

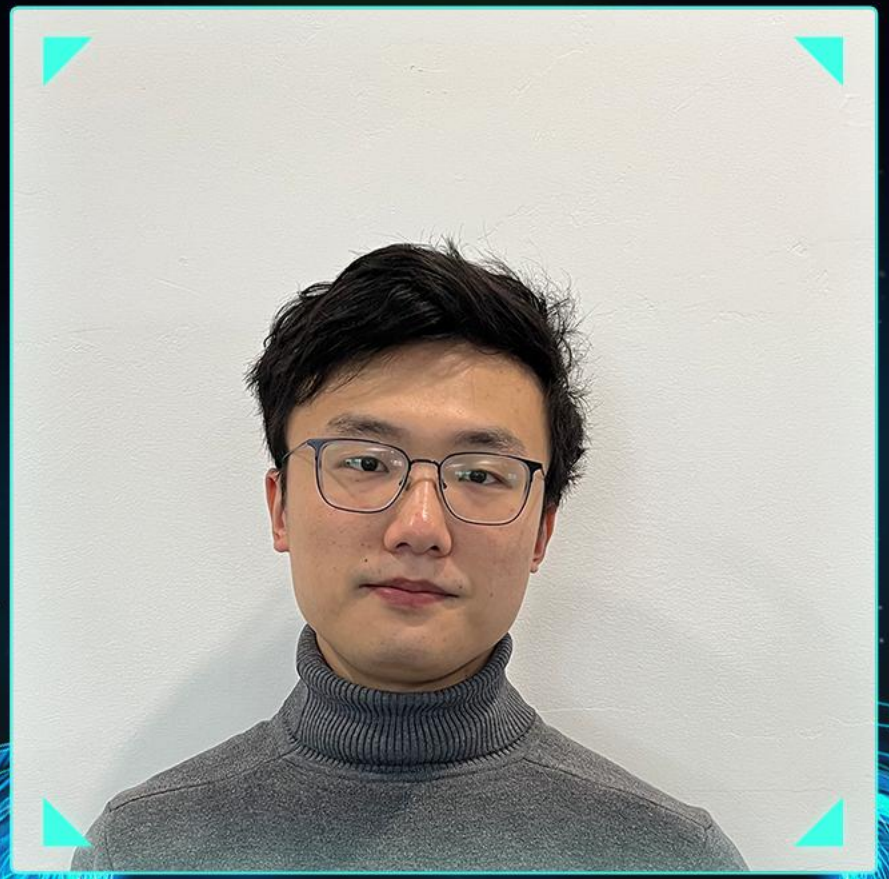
2021

58同城 第2届 安全技术沙龙

业务风控建设 & 应用安全实践

58安全应急响应中心 Security Response Center | 58安全 58 CYBERSECURITY

DATE - 2021.12.17 线上直播



» 方 钊 «

58集团资深安全工程师

分享主题

58集团代码分析技术实践

议题介绍

代码分析技术是甲方应用安全里非常重要的一项安全问题发现手段，但是在传统应用安全建设中，技术的落地往往存在一些如：白盒误报运营成本高、SCA无法证明漏洞可利用性、单项产品能力存在瓶颈等落地难题。本次议题主要介绍58集团在代码分析技术上的一些优秀实践及架构设计

01

代码分析安全应用

58 DevOps现状

58集团自建了DevOps流程，日均构建、发布上千次，在快速迭代的背景下，需要在应用上线前发现安全问题，收敛整个应用安全风险

2021年通过白盒/SCA上线前发现漏洞占比29.3%，本次分享会着重分析一下甲方业务场景下的代码分析落地的一些问题

02

甲方代码分析困境

2- 1

静态分析落地难题

静态分析误报

OWASP对SAST和DAST类产品在OWASP Benchmark的测试结果——检出率最高100%

误报率最高52%

Analyzer	Benchmark version	TPR	FPR	Y
ShiftLeft	1.2	1.0	0.25	0.75
FBwFindSecBugs	1.2	0.97	0.58	0.39
SonarQube	1.2	0.50	0.17	0.33
SAST-04	1.1	0.61	0.29	0.33
SAST-06	1.1	0.85	0.52	0.33
SAST-02	1.1	0.56	0.26	0.31
SAST-03	1.1	0.46	0.214	0.25
SAST-05	1.1	0.48	0.29	0.19
SAST-01	1.1	0.29	0.12	0.17
PMD	1.2	0.00	0.00	0.00

白盒误报

莱斯定理

莱斯定理：递归可枚举语言的所有非平凡性质都是不可判定的

在程序分析中,不存在既无误报又无漏报的分析程序

在白盒分析的场景中，一般来说更倾向更全面 即接受可接受的误报，换取程序的安全及健壮

停机问题

```
void Evil() {
    if (Halt(Evil))
        while(1);
    else
        return;
}
```

白盒误报

近似求解

定义数据抽象：+: 正　-: 负　0: 零　T: 未知

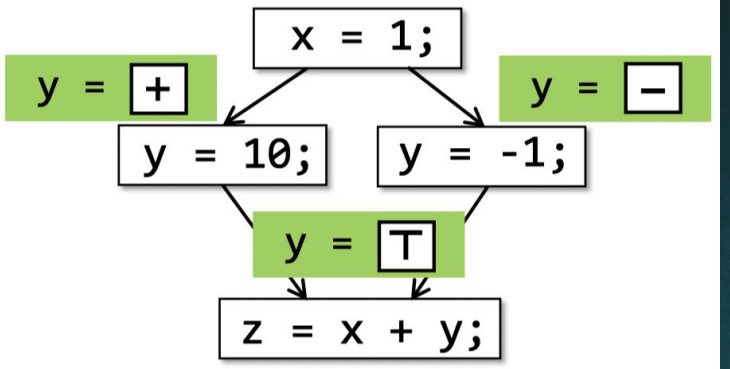
⊥: 未定义

运算函数抽象

+ + + = +	+ / + = +
− + − = −	− / − = +
0 + 0 = 0	T / 0 = ⊥
+ + − = T	+ / − = −

过近似导致精度丢失案例

```
x = 1;
if(input)
  y = 10;
else
  y = -1;
z = x + y;
```



近似前 y = -1 or 10 z=11或0

近似后 y=T z=T

白盒里使用的其他近似方法：指针分析、污点分析规则、数据流、控制流图抽象都可能导致过近似的问题增加误报

2- 2

软件成分可利用性难题

软件成分漏洞可利用性证明

基于组件版本匹配安全漏洞，无法证明漏洞可利用性，每一次的修复都在消耗研发成本

VULNERABILITY	AFFECTS	TYPE	PUBLISHED
M Cryptographic Issues	org.apache.directory.server:apacheds-kerberos-codec [0,]	Maven	06 May, 2021
M Cross-site Scripting (XSS)	org.webjars:highcharts [0,]	Maven	06 May, 2021
M Cross-site Scripting (XSS)	org.webjars.npm:highcharts [0,]	Maven	06 May, 2021
M Insecure Permissions	com.vaadin:flow-server [3.0.0,6.0.6),[2.0.9,2.5.3)	Maven	06 May, 2021
M Regular Expression Denial of Service (ReDoS)	org.webjars:is-svg [0,]	Maven	05 May, 2021
M Regular Expression Denial of Service (ReDoS)	org.webjars.npm:is-svg [0,]	Maven	05 May, 2021
H Regular Expression Denial of Service (ReDoS)	com.vaadin:vaadin-compatibility-server [8.0.0, 8.13.0)	Maven	05 May, 2021
M Improper Output Neutralization for Logs	org.apache.unomi:unomi-wab [,1.5.5)	Maven	04 May, 2021
M Improper Output Neutralization for Logs	org.apache.unomi:unomi-plugins-base [,1.5.5)	Maven	04 May, 2021
M Improper Output Neutralization for Logs	org.apache.unomi:unomi [,1.5.5)	Maven	04 May, 2021
M Cross-site Scripting (XSS)	com.box:box-android-sdk [0,]	Maven	04 May, 2021
H Arbitrary Code Execution	org.javadelight:delight-nashorn-sandbox [,0.2.0)	Maven	04 May, 2021
M Prototype Pollution	org.webjars:handlebars [0,]	Maven	04 May, 2021

白盒误报

FastJson反序列化

满足污点分析模型的Fastjson数据流也不一定能利用

限制条件：

- JDK版本
- 反序列化利用链依赖组件版本
- Fastjson组件版本

Fastjson是否能利用不仅仅依赖于是否使用，诸多运行时组件依赖也影响漏洞可利用性

```java
@RestController
public class FastjsonController {

    @RequestMapping("/fastjsonExploit")
    public String request(@RequestParam String json, @RequestHeader Map<String, String> headers) {
        headers.forEach((key, value) -> {
            System.out.printf("Header '%s' = %s%n", key, value);
        });

        try {
            Object object=JSON.parseObject(json);
            return object.toString();
        } catch (Exception e){
            return e.toString();
        }
    }
}
```

长期推修不可利用的漏洞将极大的浪费研发资源，同时也将降低研发对安全能力的预期和信任

2- 3

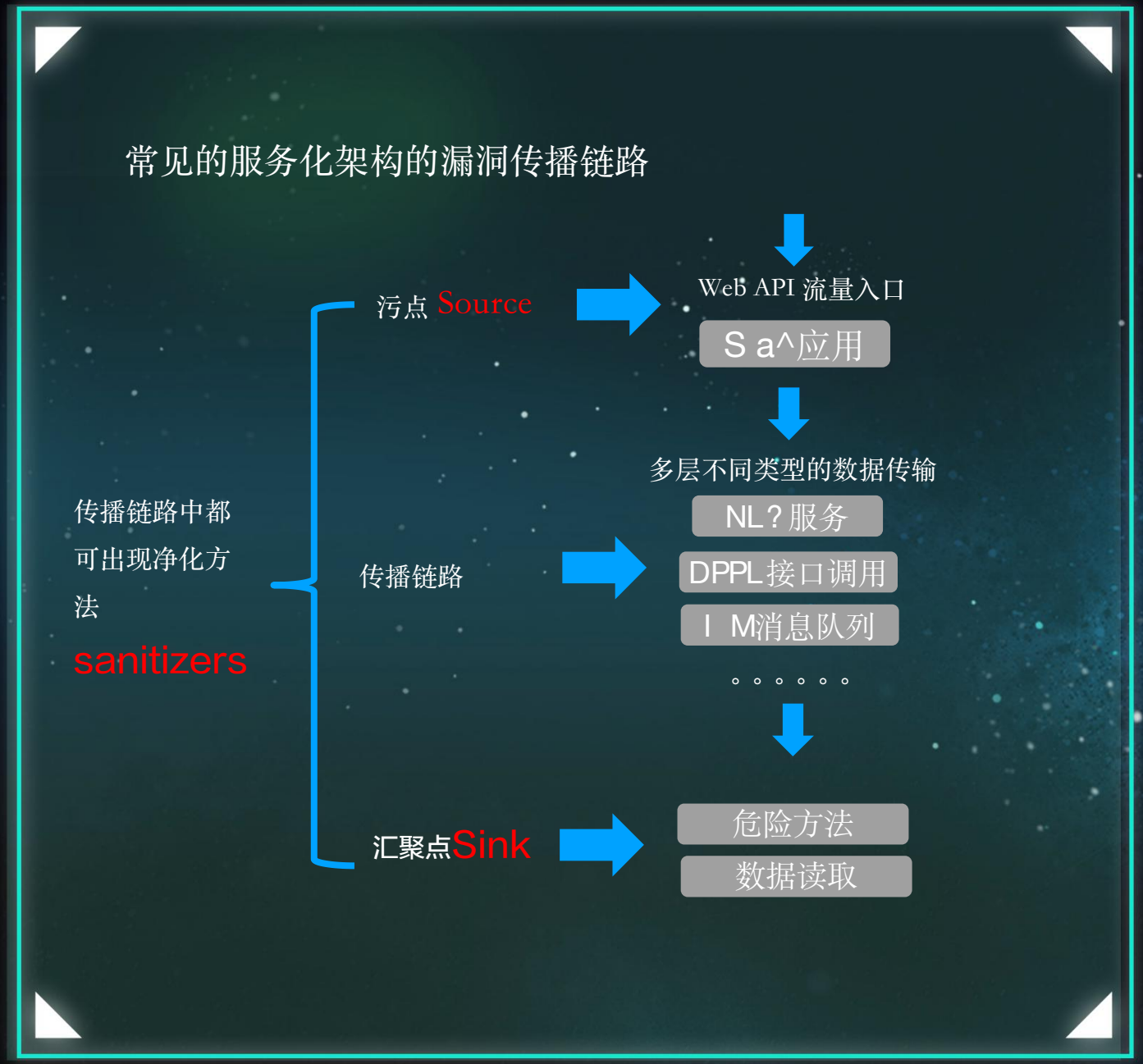
跨应用安全漏洞分析难题

跨应用漏洞

服务化架构数据流

服务化架构下，漏洞的污点传播途径往往不在集中在单个应用里，给漏洞发现带来了巨大的挑战

黑盒扫描的应用并非漏洞触发应用的情况也时常出现，对漏洞治理也造成了很大挑战



2- 5

研发修复漏洞难题

研发漏洞修复难题

Q1 不理解漏洞，更不理解为啥要修复

Q2 理解漏洞危害，想修复但是不知道怎么修

Q3 知道怎么修复，但是修复成本很高

03

困难的本质是什么

03 代码分析应用困境信任与理解

核心问题在于研发与安全人员相互理解与信任

安全能力可靠易用

安全能力低成本、体验良好

管理上拉齐风险认知

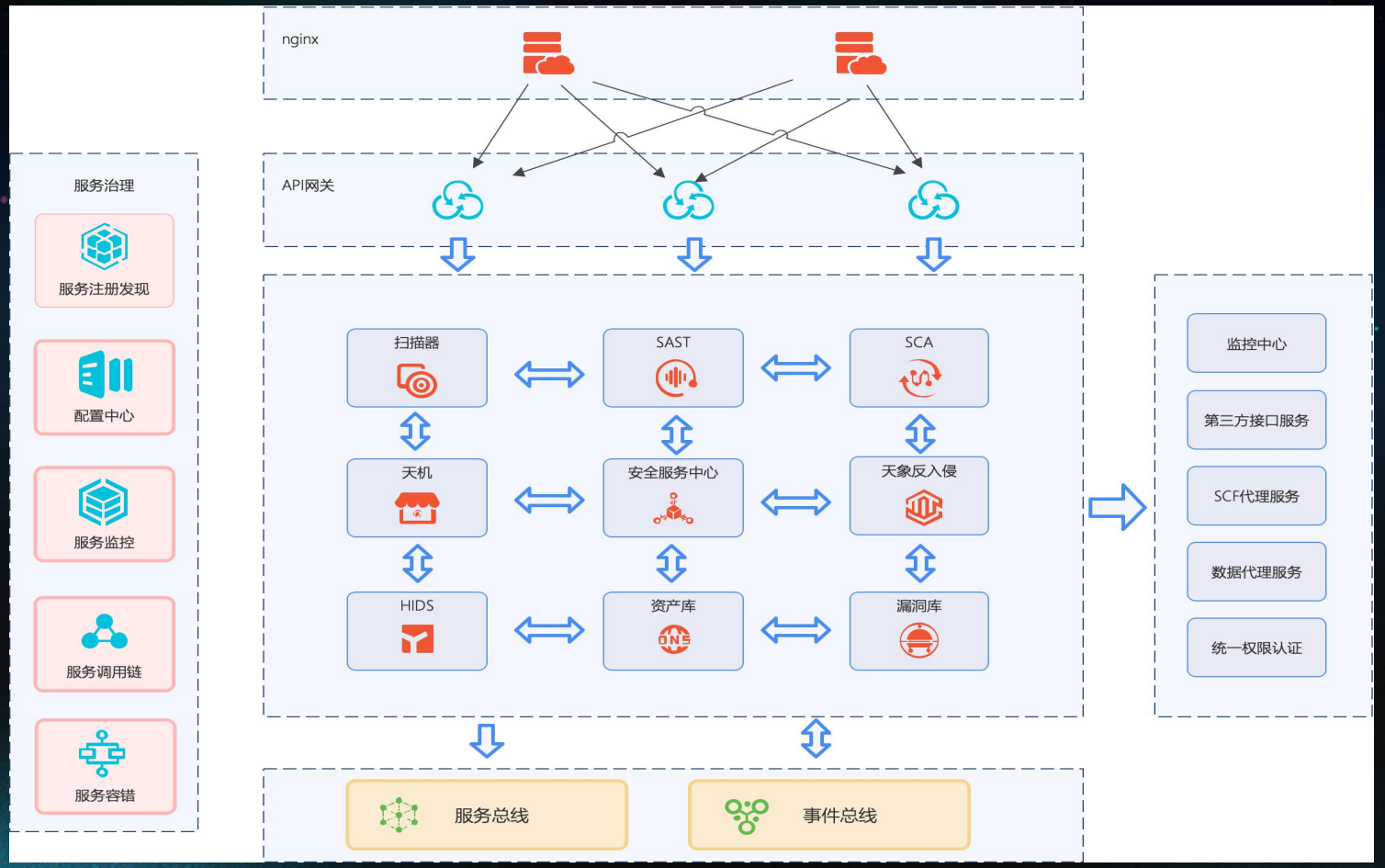
管理上拉齐安全技术认知

安全技术的落地需要积累业务部门对安全部门的信任

04

58应用代码分析实践

04 58安全实践 服务化架构

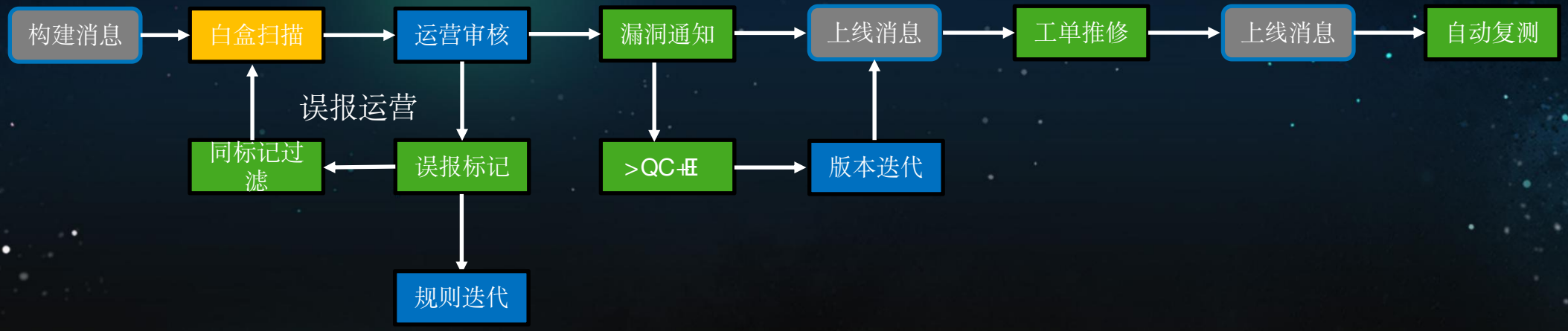


04 58安全实践 白盒DevOps融合

消息驱动的自动流程　消息驱动的人工流程　?E? @消息　安全产品能力

核心思想： 消息驱动、服务化编排、产品能力解耦、能力原子化服务化

能力解耦：白盒只负责扫描应用，?E? @流程信息、服务编排由服务总线负责



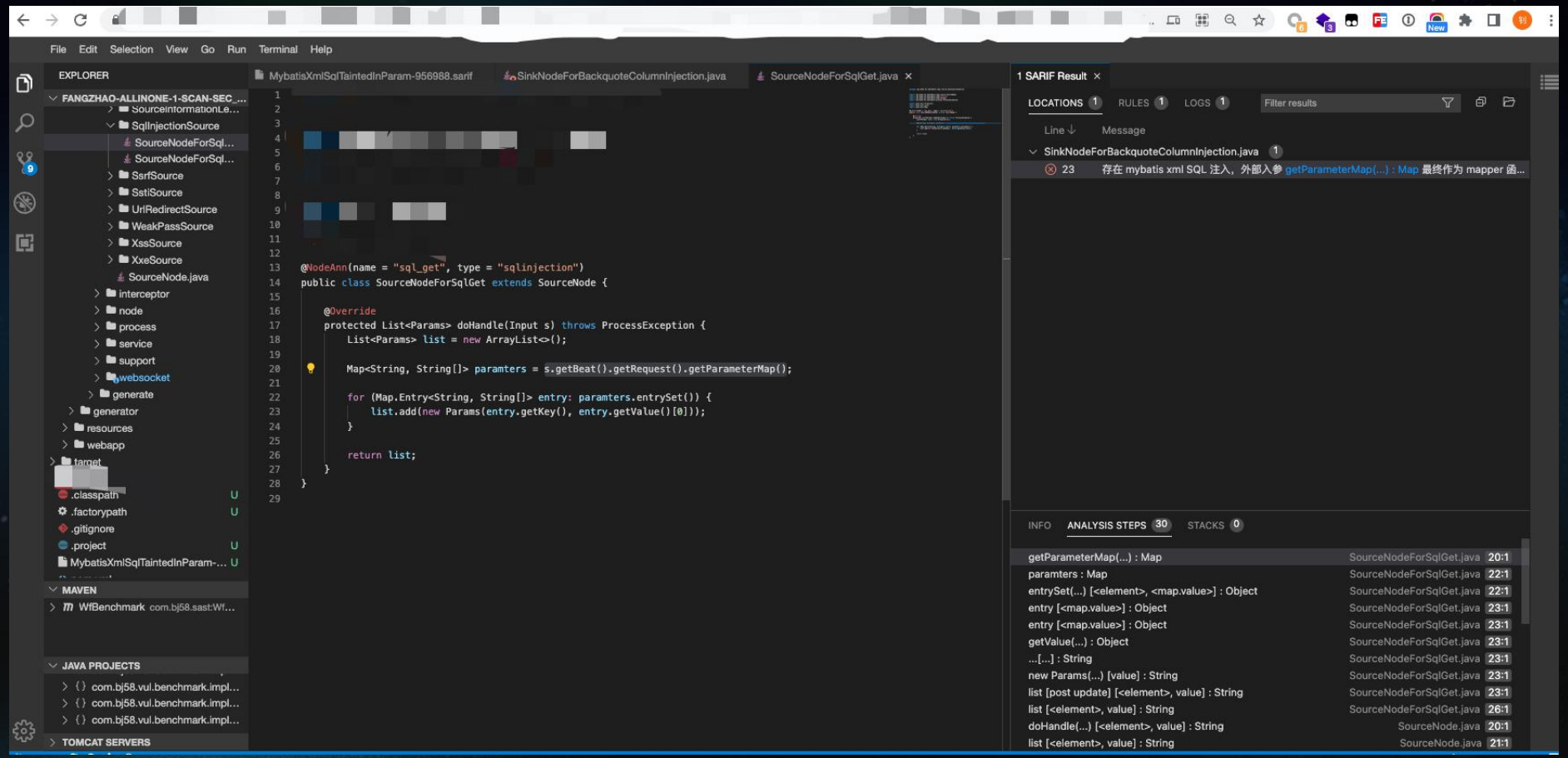
漏洞CodeFlow唯一值

主要代码位置行d] od去重，O=OP自动判断误报，处理后自动判断误报的漏洞有-，S ' 个，需要人工审计的漏洞只有5，，，'

	规则ID	规则简略描述	成分标记结果	成分标记结果附加信息	白盒分析结果	扫描规则结果id	漏洞数量	操作
-	66	MybatisXmlSqlTaintedInParam 规则	无规则	该 sarif SCA 无规则。	成功	25804958	5	

结果id	主要位置行哈希	结果标记	hash标	快照标	标记理由	标记人	标记时间	绑定服务
994497	43350c65fcf48304:1	误报	误报	误报	从 cookie 中获取键值，然后再通过键值去 Redis 读取数据，此时 payload 已不可控。	sast	2021-11-05 11:53:02	
994498	f0eec54ee64de70b:1	误报	误报	误报	从 cookie 中获取键值，然后再通过键值去 Redis 读取数据，此时 payload 已不可控。	sast	2021-11-05 11:53:02	
994499	23a1021585710c58:1	误报	误报	误报	从 cookie 中获取键值，然后再通过键值去 Redis 读取数据，此时 payload 已不可控。	sast	2021-11-05 11:53:02	
994500	720f1f03240118eb:1	误报	误报	误报	从 cookie 中获取键值，然后再通过键值去 Redis 读取数据，此时 payload 已不可控。	sast	2021-11-05 11:53:02	
994501	f1ce6d35c0739bb3:1	误报	误报	误报	从 cookie 中获取键值，然后再通过键值去 Redis 读取数据，此时 payload 已不可控。	sast	2021-11-05 11:53:02	

04 58安全实践 VSCode Online在线修复



安全SDK修复方案

提供标准化的漏洞修复方案，通过f] r] 的OL特性针对不同业务场景使用不同的安全方法

Security SDK 使用帮助文档

SDK介绍

Security SDK是由 TEG-安全平台部-基础安全部 研发供全集团使用的Java安全漏洞修复SDK，目前Java安全类库提供了 以下一些安全漏洞自助修复方法

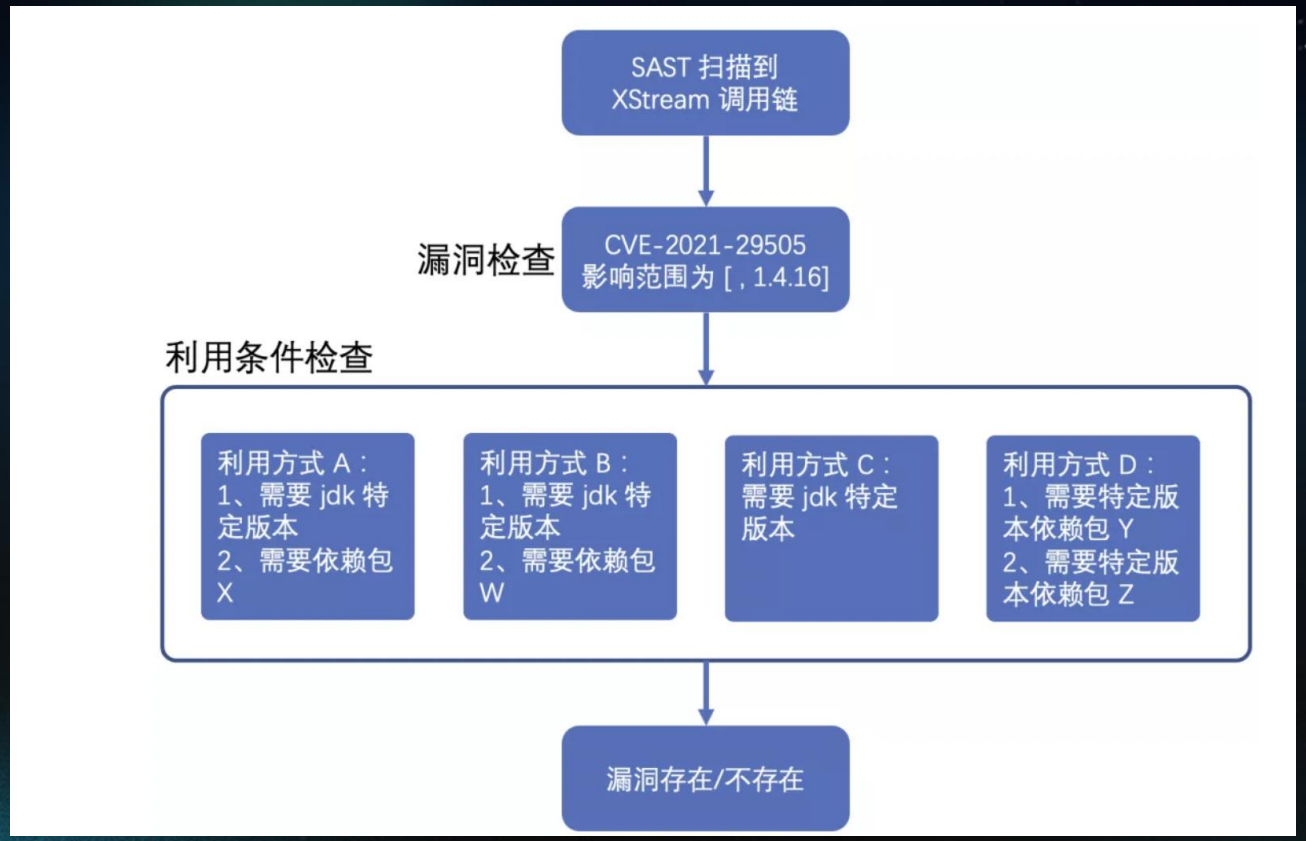
- SQLI（Sql Query Injection-Sql注入）
- XSS（Cross Site Scripting-跨站脚本执行）
- SSRF（Server Side Request Forgery-跨站脚本伪造）
- CommandInjection（CommandInjection-命令注入）
- XXE（XML External Entity-XML外部实体注入）
- UnsafeDeserialization（UnsafeDeserialization-不安全的反序列化）
- PathInjection（PathInjection-文件目录注入）

04 安全实践 自助扫描

研发同学可进行自助扫描，在编码阶段就能发现白盒相关漏洞

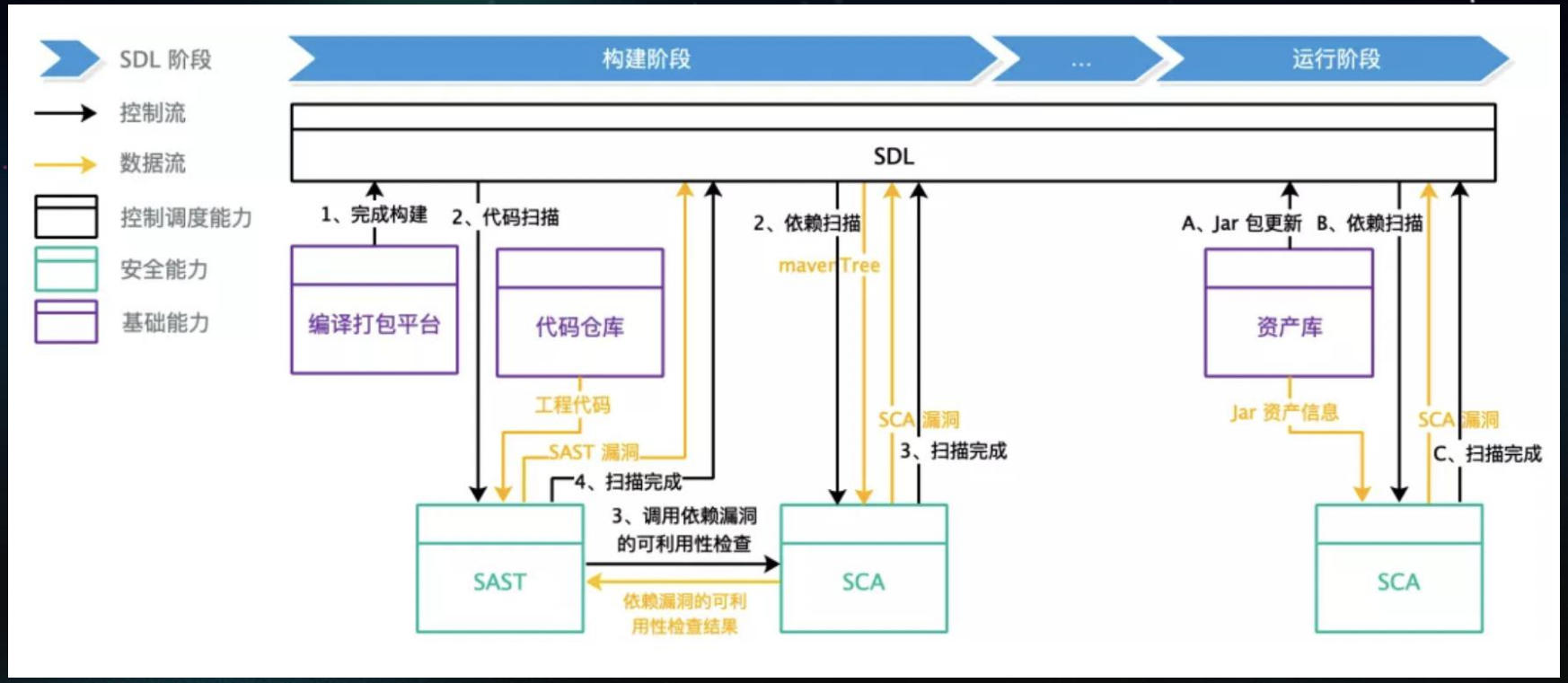
SCA漏洞可利用性证明

以Topa]i 为例，存在污点分析调用链时，版本及依赖需要满足条件才能利用



SCA漏洞可利用性证明

消息驱动、数据模型打通可支持产品联动



SCA漏洞可利用性验证

O? =漏洞分为推荐修复和必须修复，推荐修复代表使用了包含漏洞的组件，必须修复代表使用了包含漏洞的组件，并存在可调用的数据流

源码漏洞列表

漏洞ID	漏洞类型	危险等级	人工审核	漏洞状态	查看详情
1028866	sql注入	高危	未审核	新增	查看详情
1028867	URL跳转	低危	已确认	新增	查看详情
1028868	SSRF跨站请求伪造	中危	误报	历史遗留	查看详情

< 1 2 3 4 ··· 20 >

依赖包漏洞列表

漏洞ID	漏洞类型	危险等级	修复要求	漏洞状态	查看详情
2028866	FastJson不安全版本	高危	必须修复	新增	查看详情
2028867	Xstream不安全版本	高危	必须修复	新增	查看详情
2028868	Log4J不安全版本	中危	推荐修复	历史遗留	查看详情

白盒URI库

消息驱动的自动流程　消息驱动的人工流程　?E? @消息　安全产品能力

每次构建迭代更新流量库版本QNENL？数据，进行QNH去重和黑白盒漏洞相互验证

应用地图

应用地图探索

- 将全量发布的Web应用入口设为Source，Rpc调用Client接口设为Sink，绘制Web应用内的数据流图
- 将RPC的服务端接口设置为Source，调用的Http接口和Rpc接口设为Sink，绘制Rpc应用内的数据流图
- 根据应用的发布消息、服务注册/发现/治理等信息串联Rpc的Server和Client信息，构成应用地图的节点和边
- 当出现某个漏洞Sink点的时候，由数据流图回溯是否存在Web Source，并且查看数据链路中是否存在sanitizers净化
- 通过图数据库查询数据链路及漏洞通路，提供基础服务

05

未来展望

白盒扫描未来规划

- 支持更多的语言类型
- 根据已经打标的结果训练
- 扫描器原子能力开放业务使用
- 跨应用、跨语言白盒扫描

SCA未来计划

- 支持更多的语言类型
- 支持镜像检查
- 支持license合规扫描
- 相似组件检查

应用地图未来计划

支持更多的语言类型

接入动态数据提升精度

自动化漏洞发现

敏感数据查询链路

其他计划

- 火线IAST共建
- RASP应用内防御

THANKS