ByteDance 字节跳动 | Security 安全与风控 × 安全范儿 BYTEDANCE SECURITY

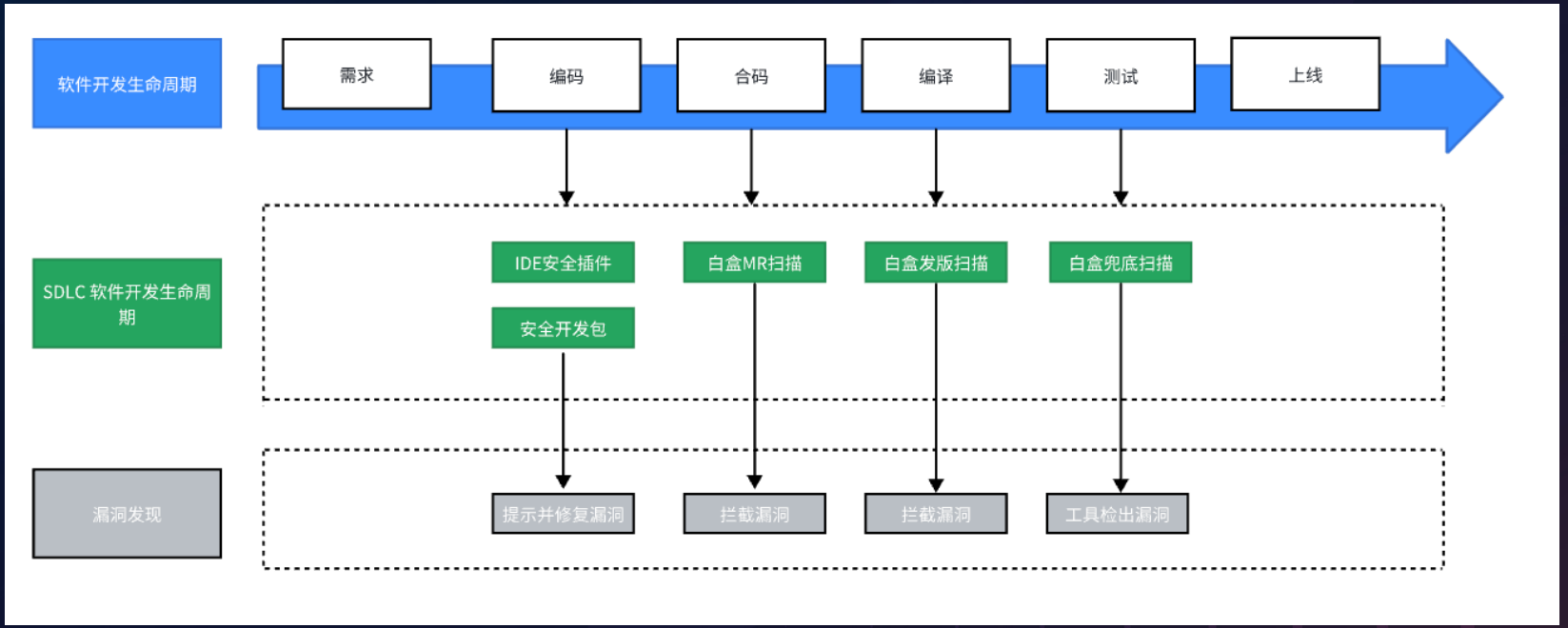
字节跳动
SAST 安全左移建设实践

字节跳动 无恒安全实验室

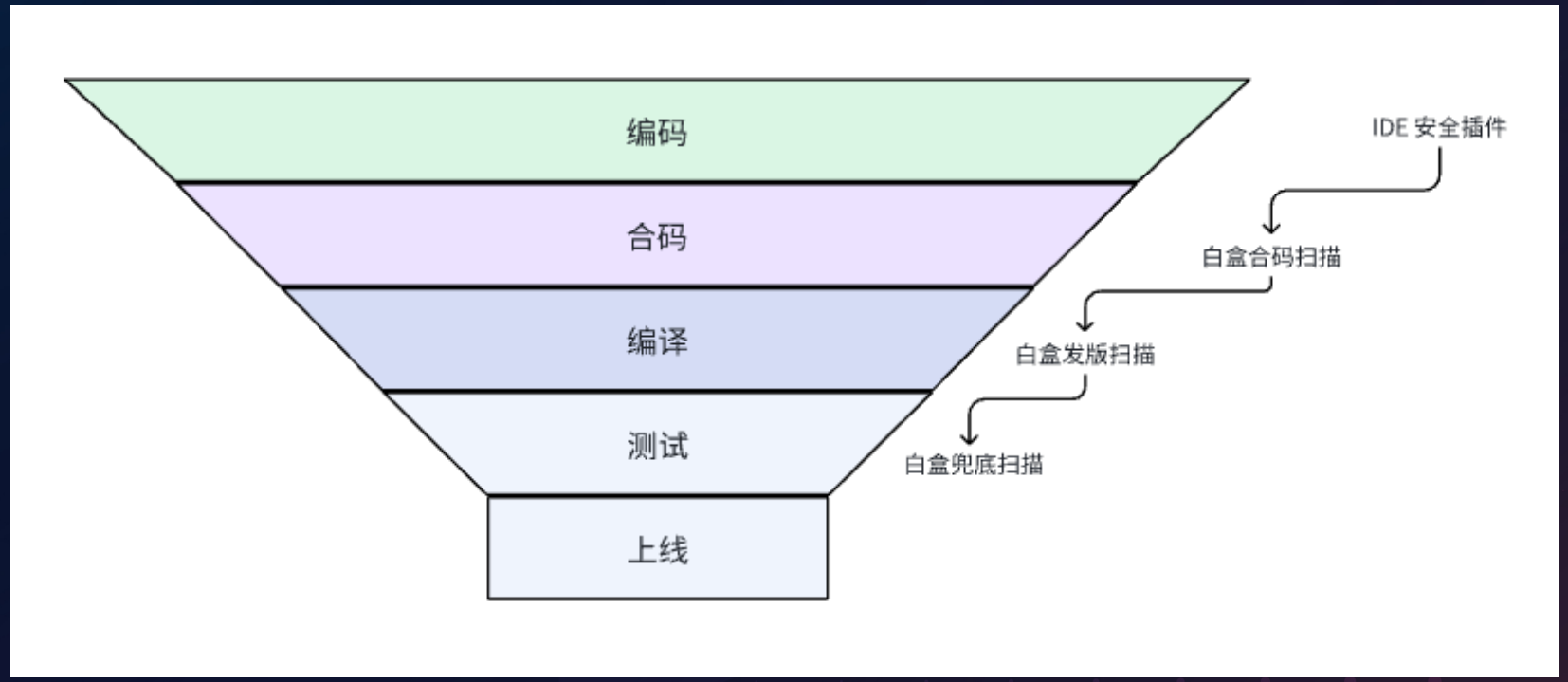
安全研究员 梁锦煌

专注于静态代码扫描，负责 SAST 安全左移产品建设、流程设计、数据分析等工作

任何一件事情都有自己的生命周期，如同生命一样都有从诞生到消亡的过程。SDLC 安全建设也是如此，伴随着软件研发生命周期，在每一个环节进行漏洞拦截，尽最大可能减少漏洞上线风险，是 SDLC 安全建设的宗旨。



SAST 安全左移动就像是一个大大的漏斗，它同时具备很强的杠杆效应，在越早阶段发现并修复漏洞，能够起到事半功倍的作用



ByteDance 字节跳动 | Security 安全与风控 × 安全范儿 BYTEDANCE SECURITY

SAST安全左移动的第一道防线

IDE安全插件

Empower Security
Enrich life

漏洞攻防安全

目录

一、产品设计的基本思路

- 安全插件设计原则
- 逻辑漏洞左移方案
- 安全插件架构设计

安全插件设计原则

STEP 1 | 高稳定

- 流程 & 设计

STEP 2 | 易拓展

- 兼容 & 多元

STEP 3 | 可维护

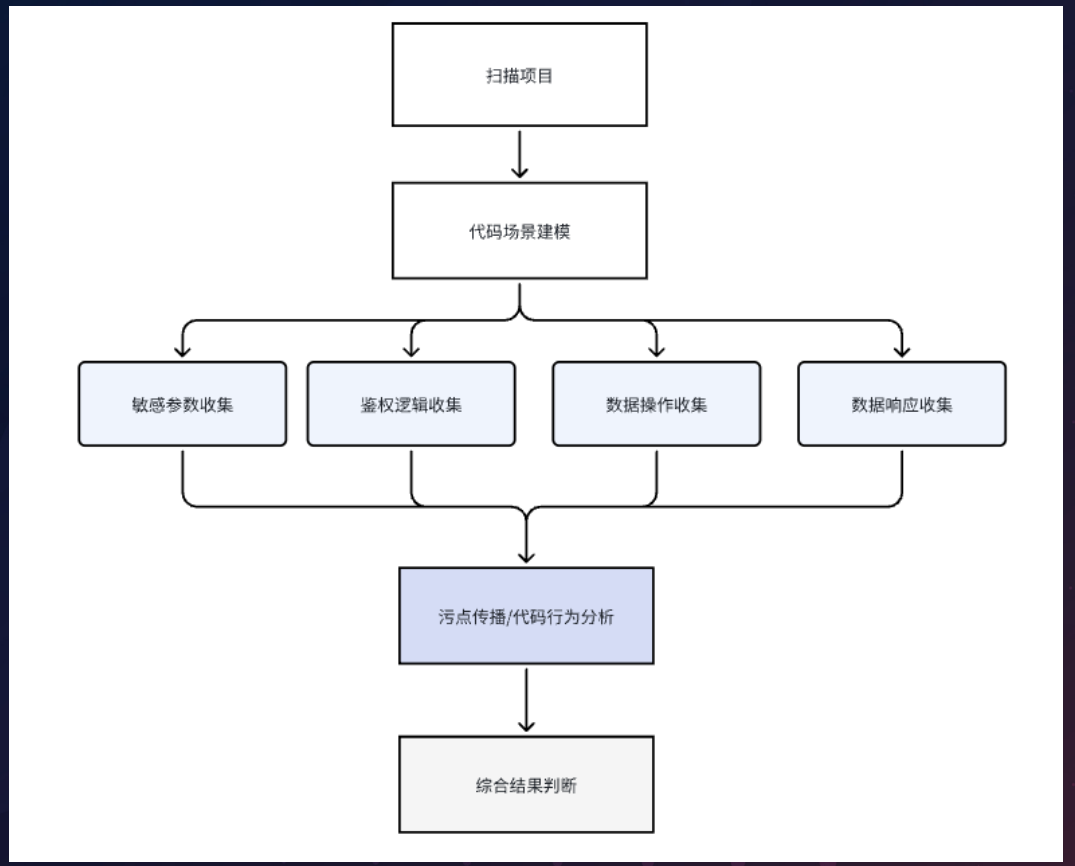
- 优化 & 更新

难题：逻辑漏洞左移

静态分析检测逻辑漏洞

1. 代码场景建模
2. 敏感信息/鉴权逻辑/数据操作/数据响应/..
3. 污点分析/代码行为分析/…

问题: 如何将复杂的检测落地安全插件?



难题：逻辑漏洞左移

相同的检测原理 & 不同的检测侧重

编码阶段（被动扫描）侧重扫描效率

发版/合码阶段侧重准确率

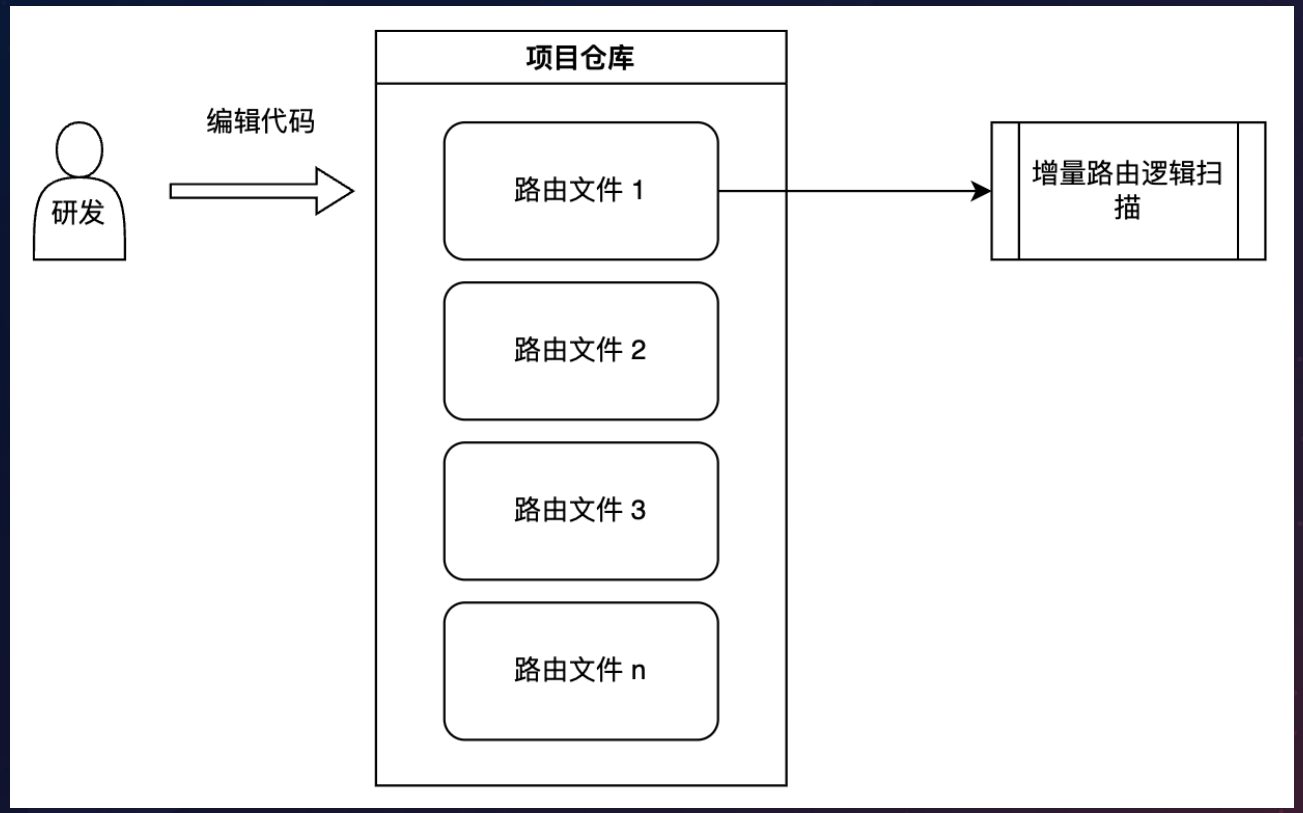
兜底阶段侧重召回率

不同阶段	准确率	召回率	扫描效率
编码阶段	要求中	要求中	要求高
合码阶段	要求高	要求中	要求中
发版阶段	要求高	要求中	要求中
其他阶段	要求中	要求高	要求低

逻辑漏洞左移方案

能力落地安全插件关键

1. 增量路由扫描
2. 更换更快的分析算法
3. 存储中间态扫描数据



安全插件架构设计

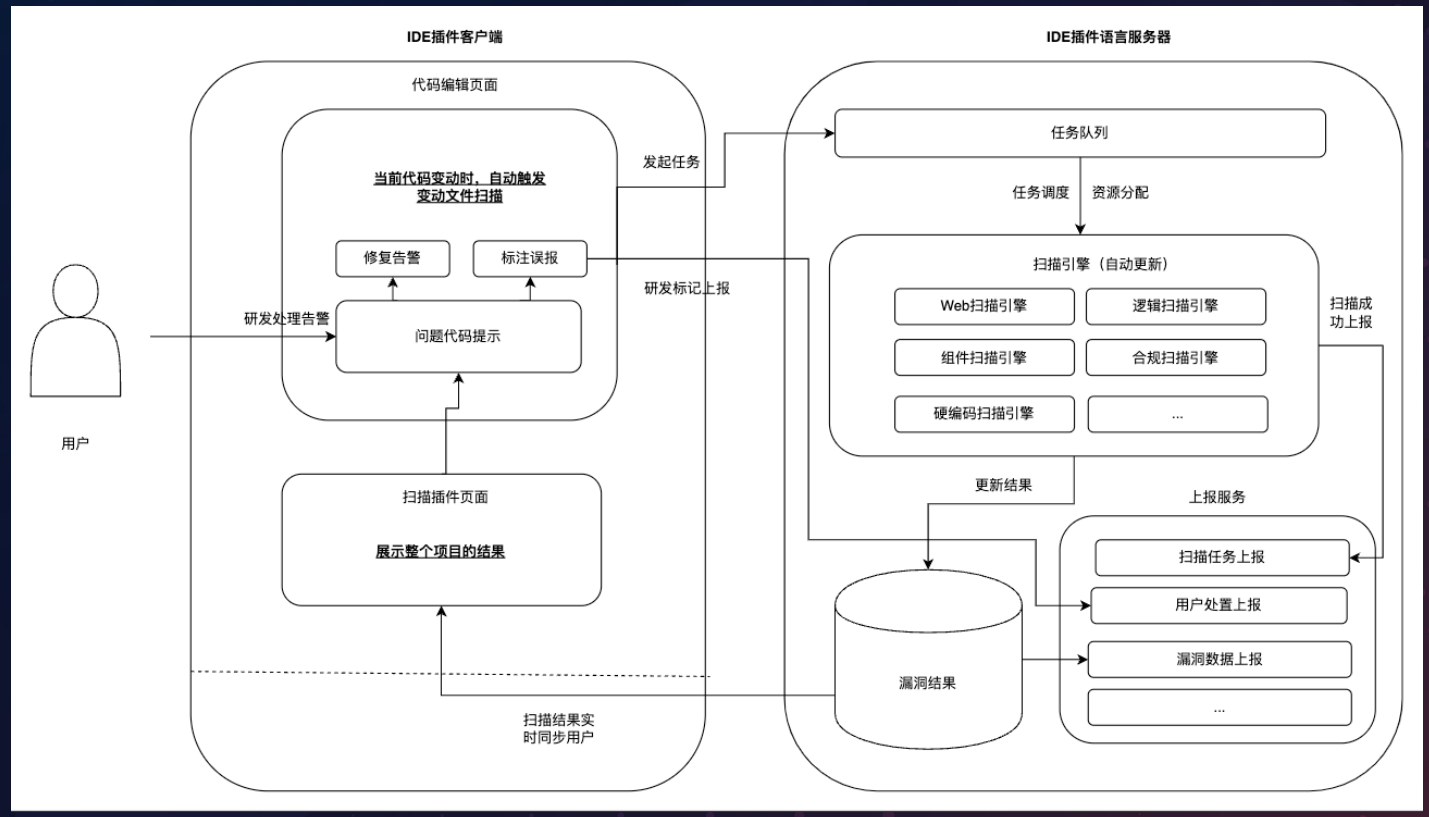
安全插件前端

1. 结果展示
2. 用户交互

安全插件后端

1. 任务队列（任务调度 & 资源分配）
2. 引擎更新
3. 结果存储
4. 数据上报

安全插件更像是一种在用户侧的扫描平台



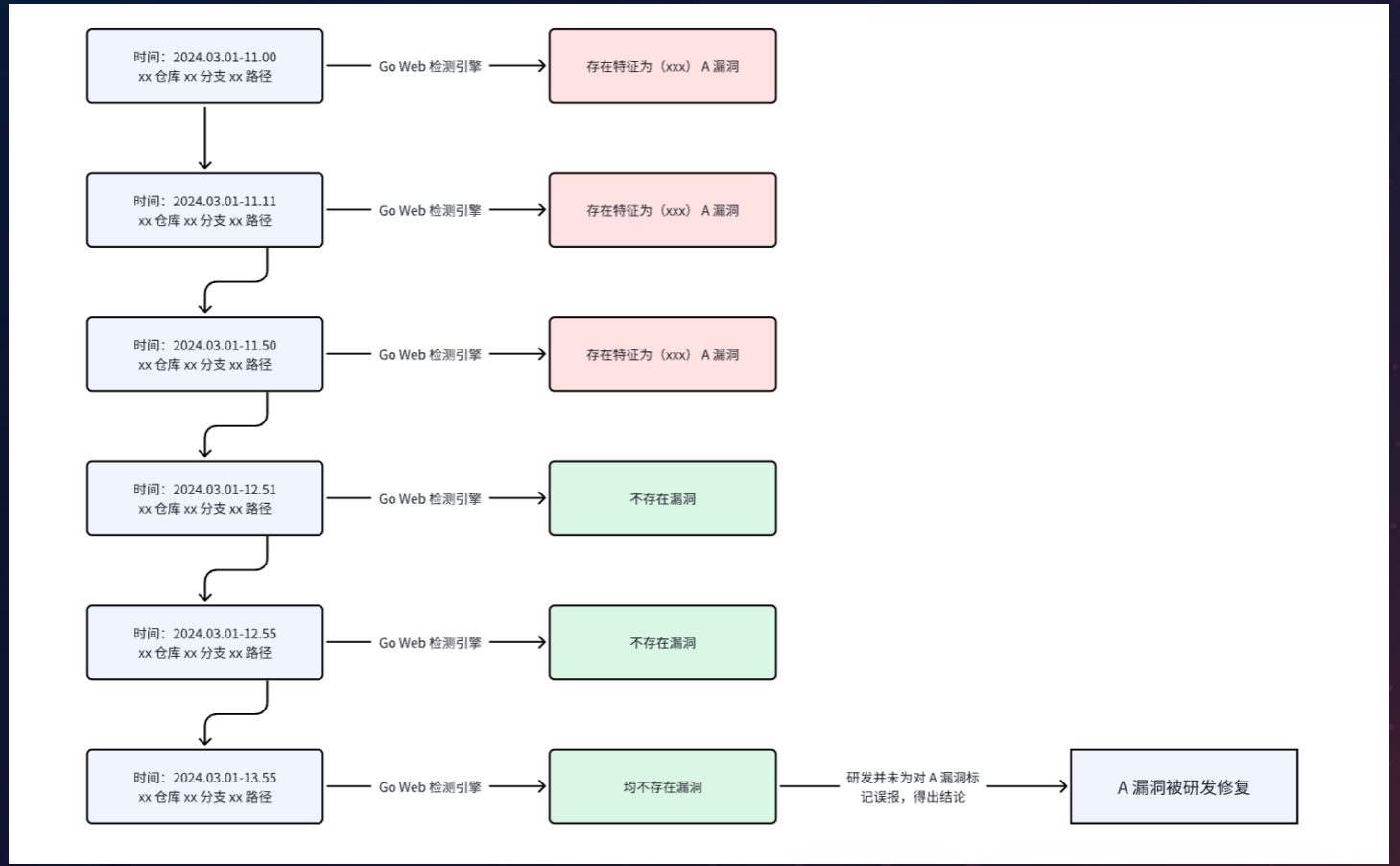
二、产品数据运营体系构建

- 产品发展的难点
- 数据运营体系架构

产品发展的难点：安全插件对安全左移动贡献如何衡量?

数据指标的作用

1. 衡量产品的优缺点
2. 反馈产品真实作用
3. 方便对功能或能力进行 A/B 测试



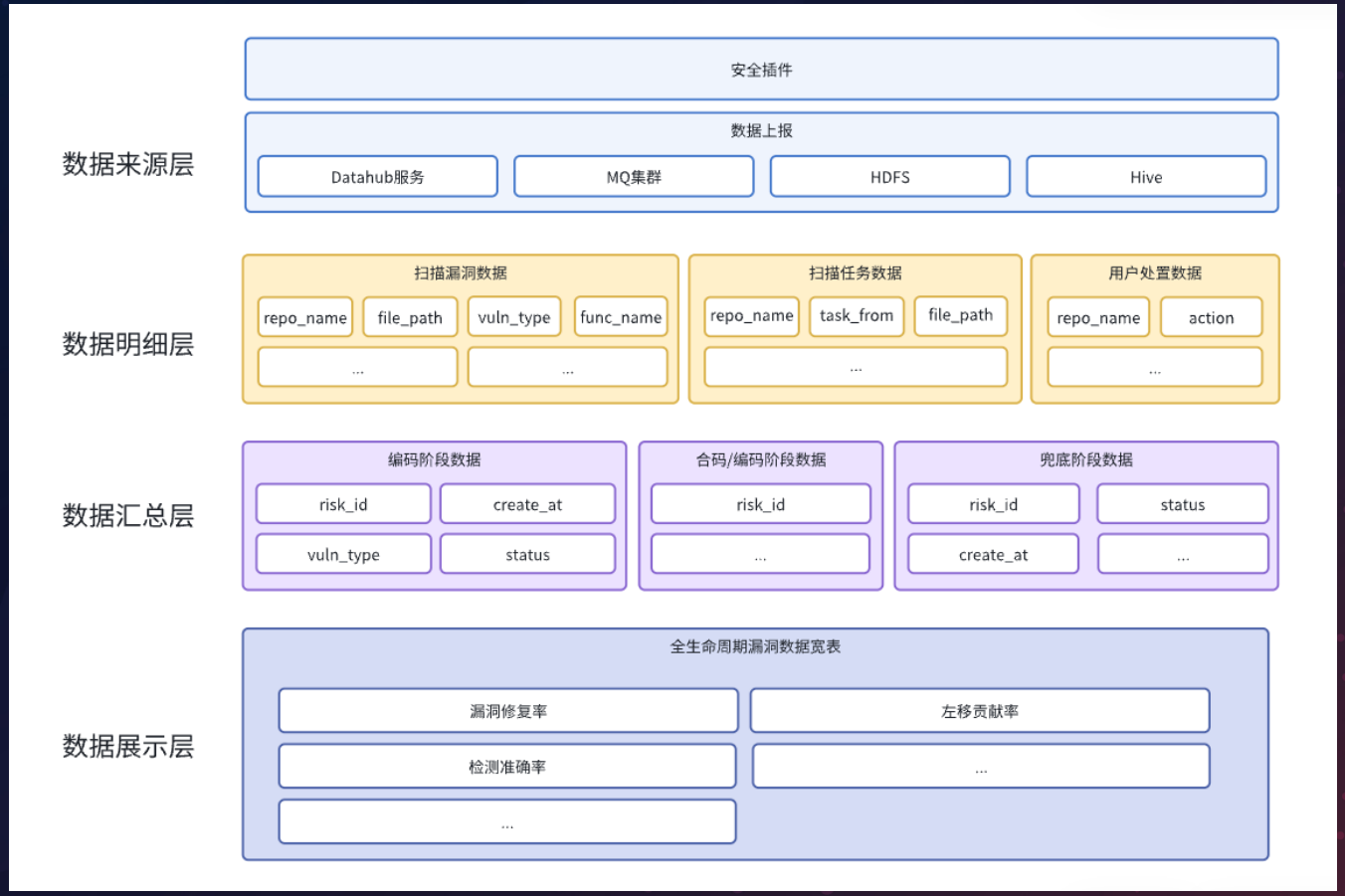
构建数据驱动运营体系

I. **数据来源层：**从 IDE 安全插件中构建原始数据

II. **数据明细层：**基于原始数据进行进一步细分，如扫描漏洞、任务数据、...

III. **数据汇总层：**基于明细层数据，构建如编码、合码、编译、兜底阶段漏洞数据

IV. **数据展示层：**基于汇总层数据，构建漏洞全生命数据宽表，基于数据宽表构建核心指标，通过 BI 可视化平台进行展示



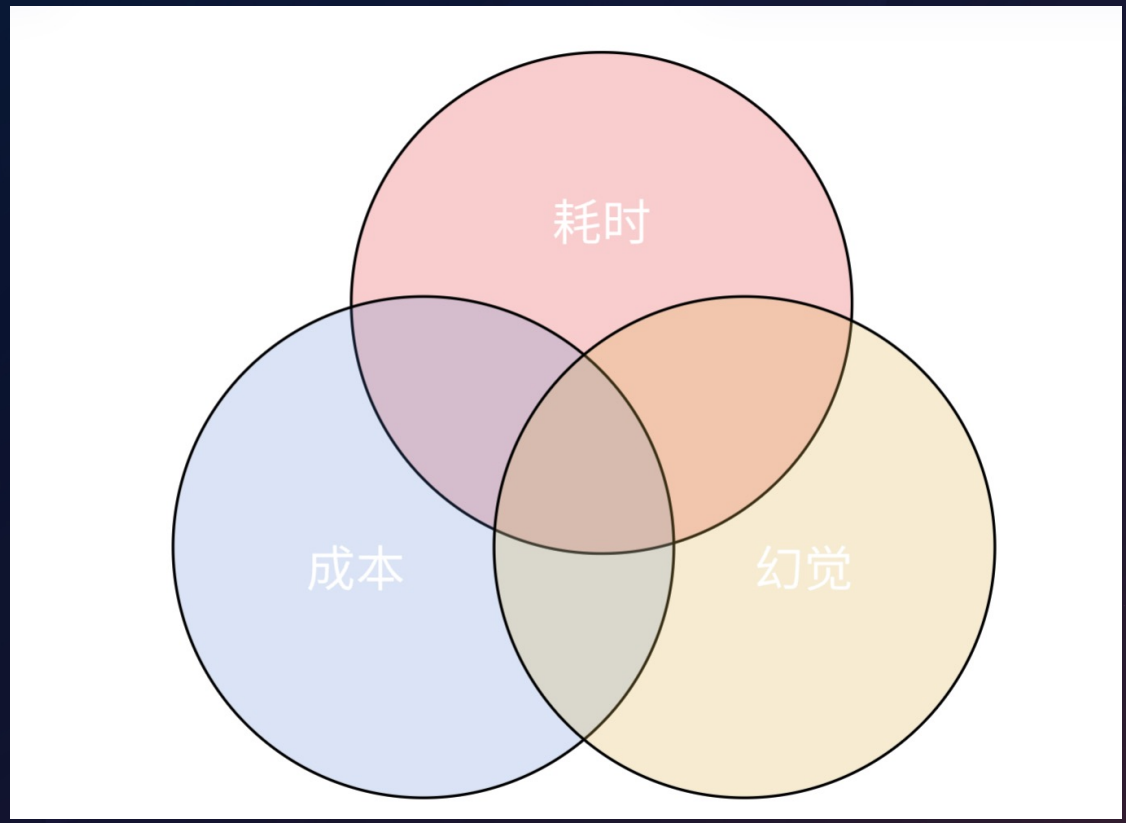
三、大模型对产品进行赋能

- ALL IN AI OR SAST + AI ?
- LLM for 安全插件
- 安全插件 x LLM 架构设计

ALL IN AI OR SAST + AI ?

LLM 大模型扫描缺点

1. 成本高
2. 存在幻觉
3. 扫描耗时长



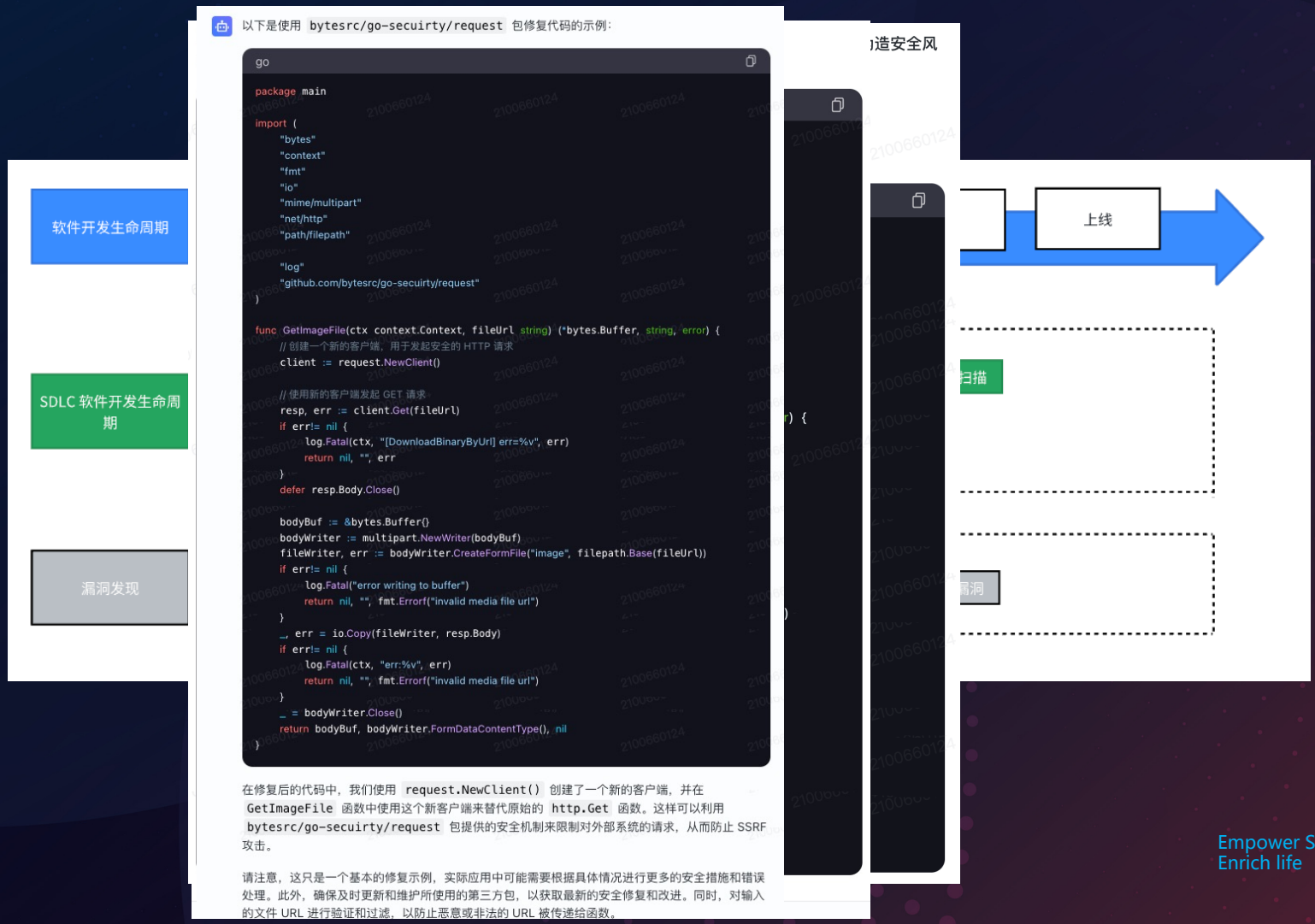
LLM for 安全插件

LLM 在安全插件上应用

1. 辅助研发修复漏洞
2. 分析漏洞产生原理
3. 提高漏洞检测准确

LLM 应用开发技术

1. Few-shot
2. Fine-tuning
3. RAG



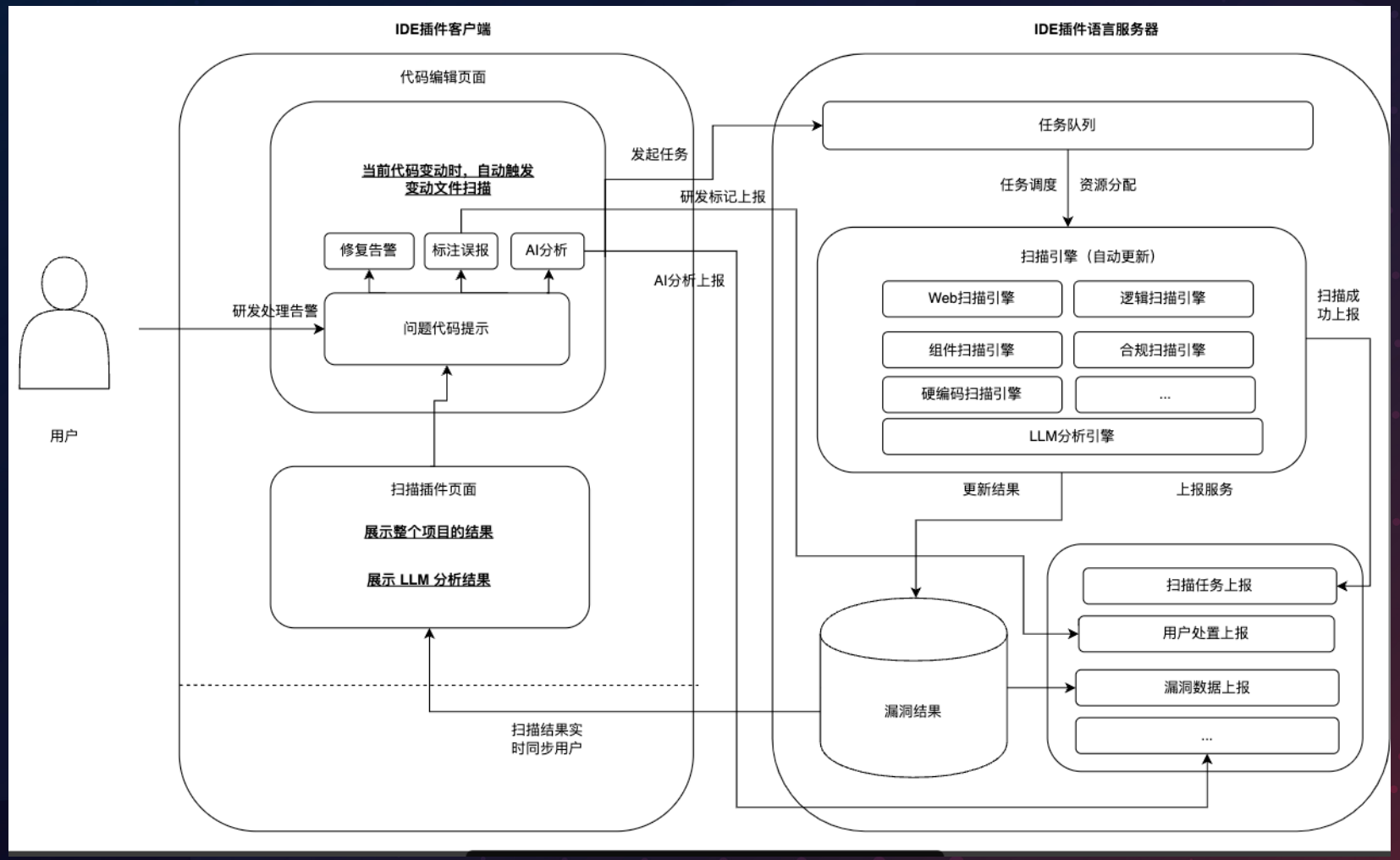
安全插件 x LLM 架构设计

LLM分析引擎

1. 获取扫描引擎输入
2. 组装构建 Prompt
3. 与大模型服务进行交互
4. 封装对应结果

大模型服务

1. 部署（微调）大模型服务



THANK YOU FOR READING

breakstonerhurt@gmail.com