安世加沙龙第三十四期

云原生DevSecOps建设实践

欢聚集团 李志鹏 2022.05

内容提要

1. 云原生定义及容器云架构简介

2. DevSecOps体系介绍

3. DevSecOps体系及工具链建设实践

安世加

云原生定义

CNCF Cloud Native Definition v1.0

Approved by TOC: 2018-06-11

日本語版 (Japanese) | 한국어 (Korean) | Deutsch (German) | Español (Spanish) | (Hebrew) עברית | 中文版本 (Chinese) | (Arabic) العربية
Français (French) | Polski (Polish) | Português Brasileiro (Portuguese-BR) | Português de Portugal (Portuguese-PT) | Русский (Russian) | Bahasa Indonesia (Indonesian) | Türkçe (Turkish) | Български (Bulgarian)

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

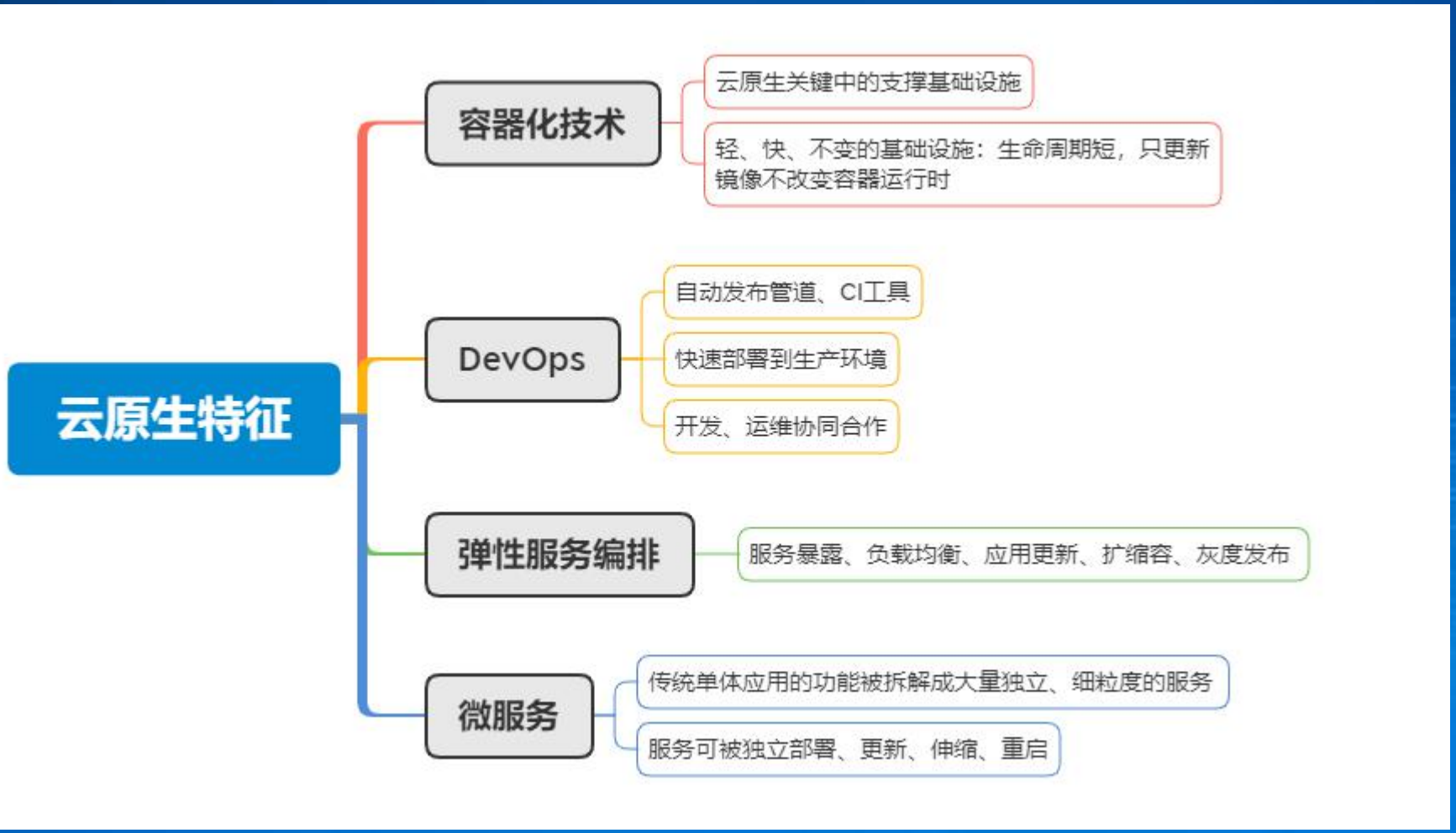
The Cloud Native Computing Foundation seeks to drive adoption of this paradigm by fostering and sustaining an ecosystem of open source, vendor-neutral projects. We democratize state-of-the-art patterns to make these innovations accessible for everyone.

云原生是一种行为方式和设计理念，究其本质，凡是能够提高云上资源利用率和应用交付效率的行为或方式都是云原生的。

关键词：容器化、服务网格、微服务、不变的基础设施

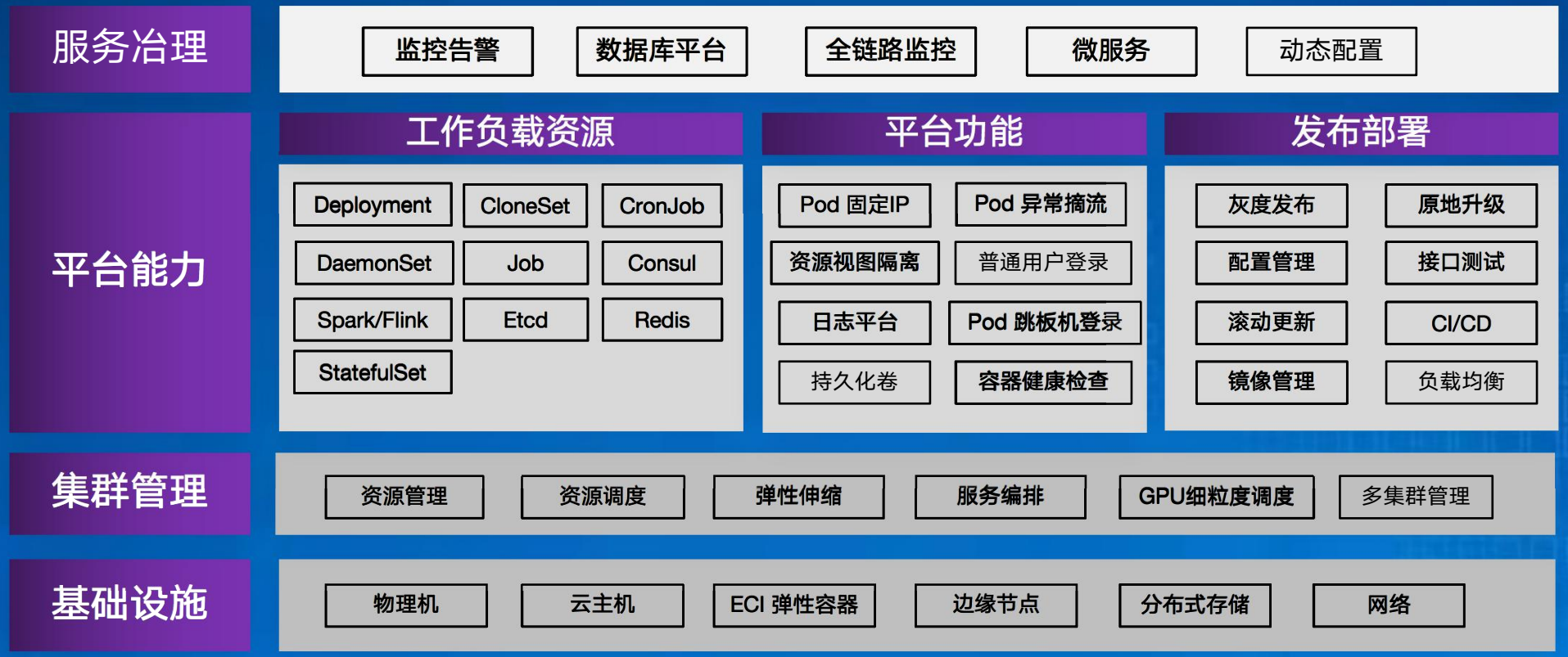
安世加

云原生特征


云原生特征
容器化技术
云原生关键中的支撑基础设施
轻、快、不变的基础设施：生命周期短，只更新镜像不改变容器运行时
DevOps
自动发布管道、CI工具
快速部署到生产环境
开发、运维协同合作
弹性服务编排
服务暴露、负载均衡、应用更新、扩缩容、灰度发布
微服务
传统单体应用的功能被拆解成大量独立、细粒度的服务
服务可被独立部署、更新、伸缩、重启

安世加

容器云架构


服务治理
监控告警
数据库平台
全链路监控
微服务
动态配置
平台能力
工作负载资源
Deployment
CloneSet
CronJob
DaemonSet
Job
Consul
Spark/Flink
Etcd
Redis
StatefulSet
平台功能
Pod 固定IP
Pod 异常摘流
资源视图隔离
普通用户登录
日志平台
Pod 跳板机登录
持久化卷
容器健康检查
发布部署
灰度发布
原地升级
配置管理
接口测试
滚动更新
CI/CD
镜像管理
负载均衡
集群管理
资源管理
资源调度
弹性伸缩
服务编排
GPU细粒度调度
多集群管理
基础设施
物理机
云主机
ECI 弹性容器
边缘节点
分布式存储
网络

业务容器化收益

提升效率，解放生产力

- 资源交付快，资源申请分钟级可用
- 实例自动弹性伸缩。资源弹性伸缩不需要人工介入
- 管理成本低，业务方不用关注基础设施管理，只需要专注业务实现

成本节约

- 相比较物理、云主机部署，可节省30-50%的成本
- 机器综合资源利用率高

可靠性好

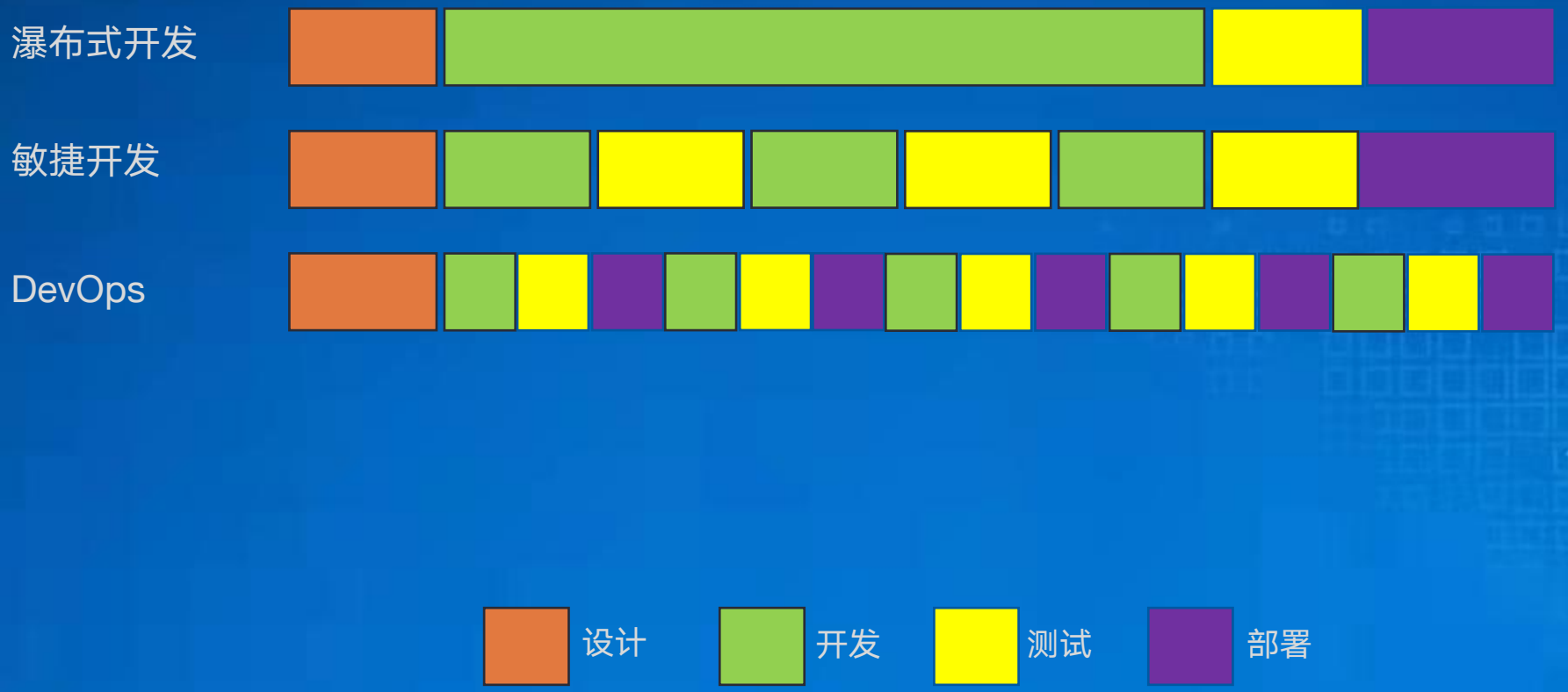
- 服务健康检查
- 故障自动切换

隔离性好

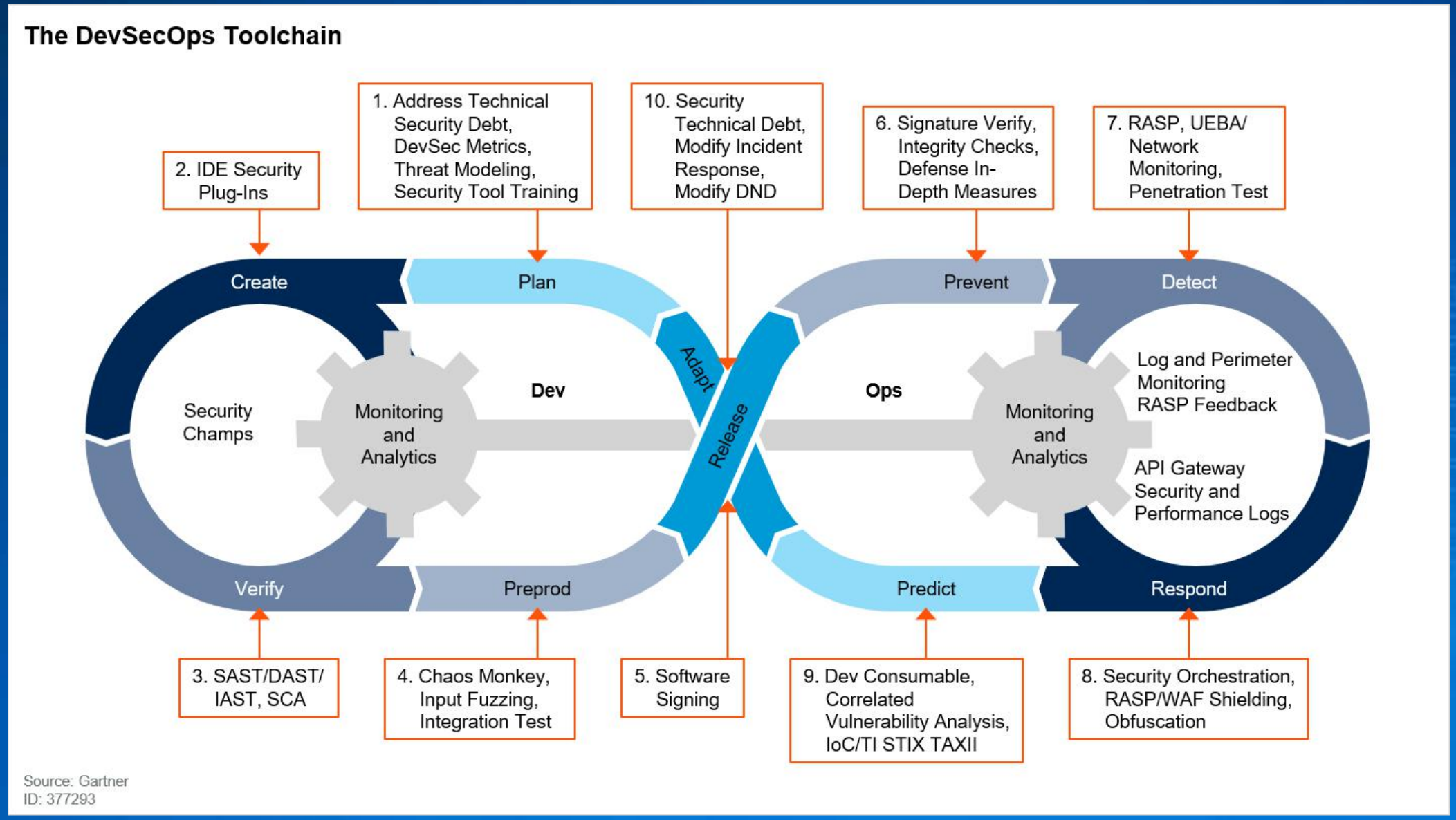
- 应用程序之间的故障相互不影响。不会因为某一个服务异常，而导致整台服务器受影响

安世加

瀑布开发 vs 敏捷开发 vs DevOps



DevSecOps方法论



DevSecOps建设思路

- 对标Gartner toolchain，查漏补缺
- 结合运维CI/CD流程系统，设置卡点，嵌入安全环节
- 自研 + 开源工具 + 二次开发

安世加

DevSecOps体系

关键工具链技术

- *AST检测工具（白盒、黑盒、灰盒）
- 软件成分分析（SCA）
- 容器安全检测工具（镜像、运行时）
- 程序运行时保护（RASP）

安世加

*AST技术特点对比

特性	SAST（白盒）	DAST（黑盒）	IAST（灰盒）
误报率	高	低	低
检出率	高	低	高
语言支持	区分不同语言	不区分语言	区分不同语言
使用成本	高，误报多，人工介入排查场景多	较低	低
漏洞验证难度	高，较难验证	较低	低
使用风险	无	主动扫描流量可能影响生产环境	特定场景可能存在脏数据影响

安世加

工具链建设介绍：SAST

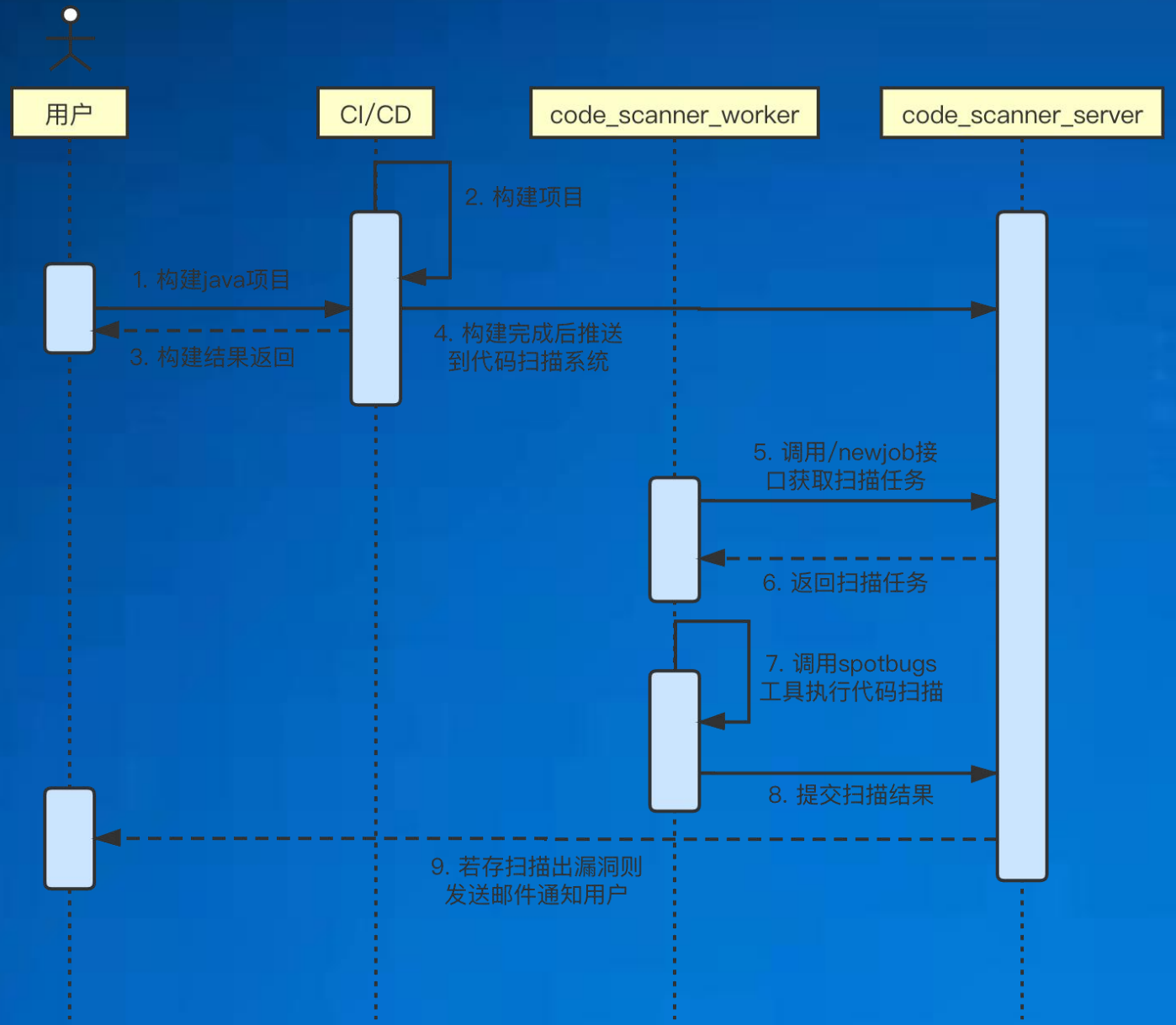
SAST工具：FindSecBugs

- OWASP开源工具SpotBugs的插件
- 针对java语言的静态代码扫描
- 支持128种漏洞代码模式检测，包括注入、XSS等OWASP Top 10主要漏洞
- 二次开发打通构建流程

FindSecBugs

安世加

工具链建设介绍：SAST



code_scanner_server

- 提供服务接口
- CI/CD平台的项目构建通知
- 任务下发/结果入库/邮件通知等功能

code_scanner_worker

- 调用spotbugs对构建任务进行扫描

工具链建设介绍：SAST

存在问题	解决/优化方案
扫描结果过多，误报率较高	提供用户标识误报/忽略的交互机制，引导用户优先关注处理高风险结果；根据扫描结果优化白名单
开发人员看不懂扫描结果，无法判断是否应处理漏洞结果	针对高频问题整理简明wiki；对开发人员定期提供安全培训

安世加

工具链建设介绍：IAST

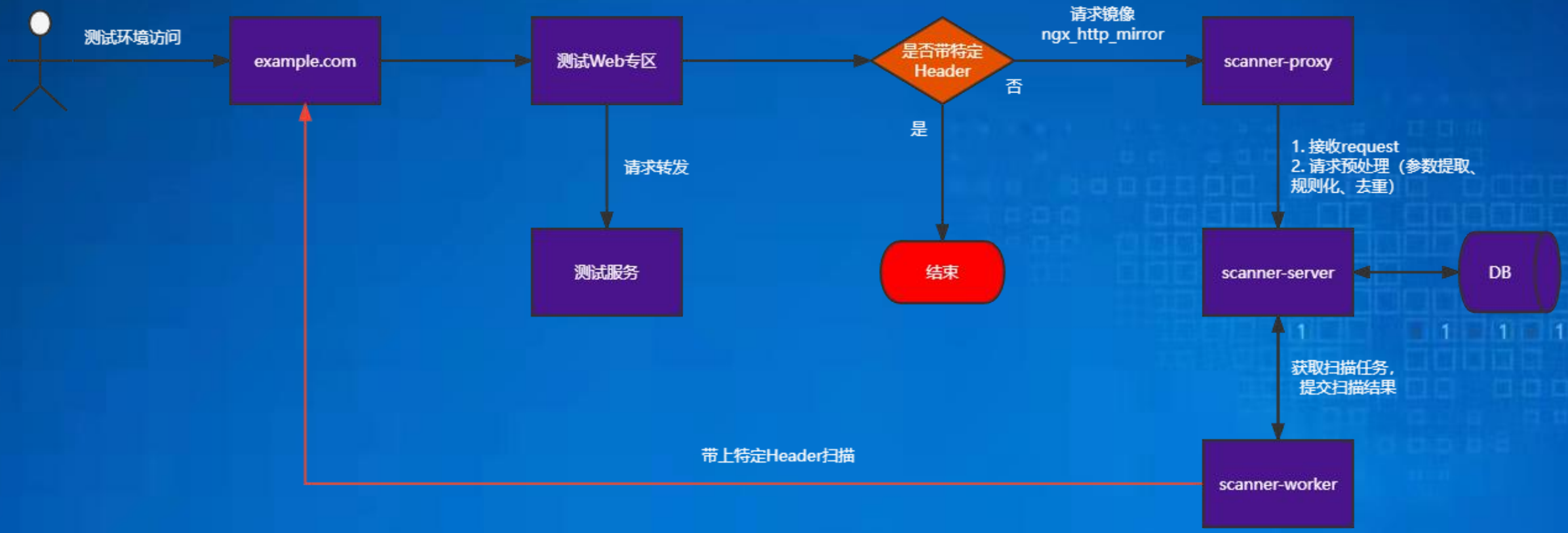
IAST实施方案

- 基于nginx mirror流量镜像的非侵入式、无agent式部署方式
- 扫描集成xray、sqlmap、自定义扫描等开源/自研工具
- 打通CI/CD发布环境，扫描开关灵活配置

安世加

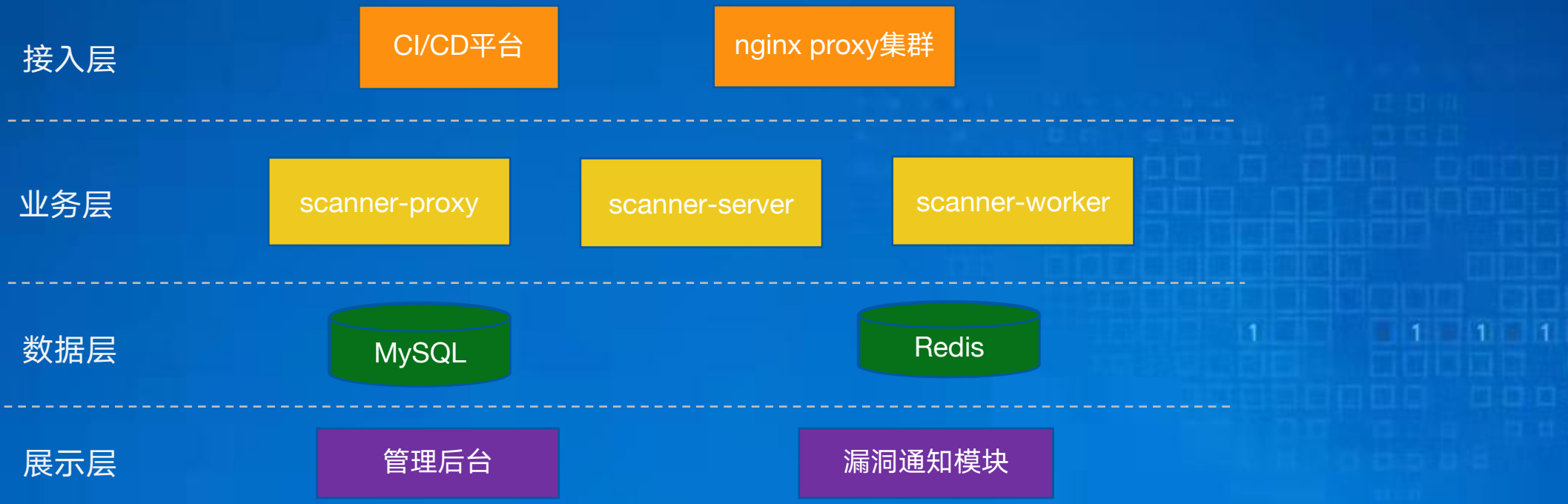
工具链建设介绍：IAST

扫描方案



工具链建设介绍：IAST

系统架构



安世加

工具链建设介绍：SCA

SCA简介

- SCA（Software Composition Analysis）软件成分分析，通俗的理解就是通过分析软件包含的信息和特征来实现对软件的识别、管理、追踪的技术。
- 2019年，Gartner在报告中把SCA纳入AST技术领域范围，从而形成了包含SAST、DAST、IAST和SCA的应用软件安全测试技术体系。
- SCA通常在开发过程中对应用程序进行分析，以检测开源软件组件是否带有已知的漏洞。

工具链建设介绍：SCA

SCA工具选型

Why Dependency-Check?

- OWASP出品，开源免费，适合二次开发。
- 支持面广，可集成性强，方便与Ant、Maven、Gradle、Jenkins等构建工具集成。

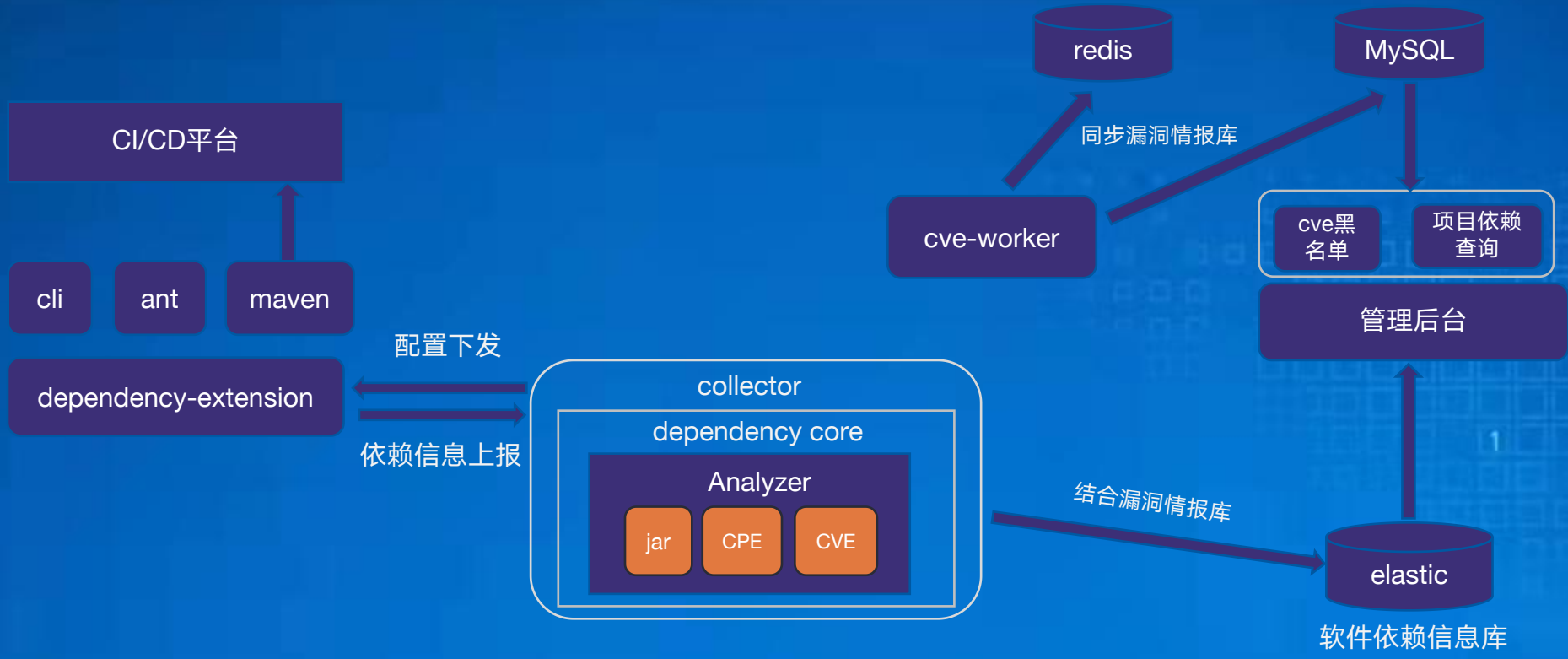
Tools Listing

Name	Owner	Licence	Platforms
Black Duck Hub	Synopsys	Commercial	Cross Platform
Clarity	Insignary	Commercial	Cross Platform / SaaS
ClearlyDefined	Open Source Initiative	Open Source	SaaS
DejaCode	nexB	Commercial	SaaS
Dependency-Check	OWASP	Open Source	Cross Platform
Dependency-Track	OWASP	Open Source	Cross Platform
Dependabot	Dependabot	Commercial / Freemium	SaaS
DepShield	Sonatype	Open Source	Cross Platform / SaaS
DotNET Retire	Retire.NET Project	Open Source	Cross Platform
FlexNet Code Insight	Flexera	Commercial	Cross Platform
FOSSA	FOSSA	Commercial / Freemium	SaaS
FOSSology	Linux Foundation	Open Source	Cross Platform
Grafeas	Grafeas	Open Source	Cross Platform
Greenkeeper	Greenkeeper	Open Source	SaaS
OSS Review Toolkit	HERE	Open Source	Cross Platform
Ion Channel SA	Ion Channel	Commercial	SaaS

安世加

工具链建设介绍：SCA

SCA系统架构



工具链建设介绍：SCA

SCA功能特点

- 插件轻量化，只负责依赖信息收集
- 透明嵌入构建流程，对用户无感知
- 支持信息收集、阻断构建功能
- 依赖库信息由elasticsearch管理，搜索查询方便

安世加

工具链建设介绍：SCA

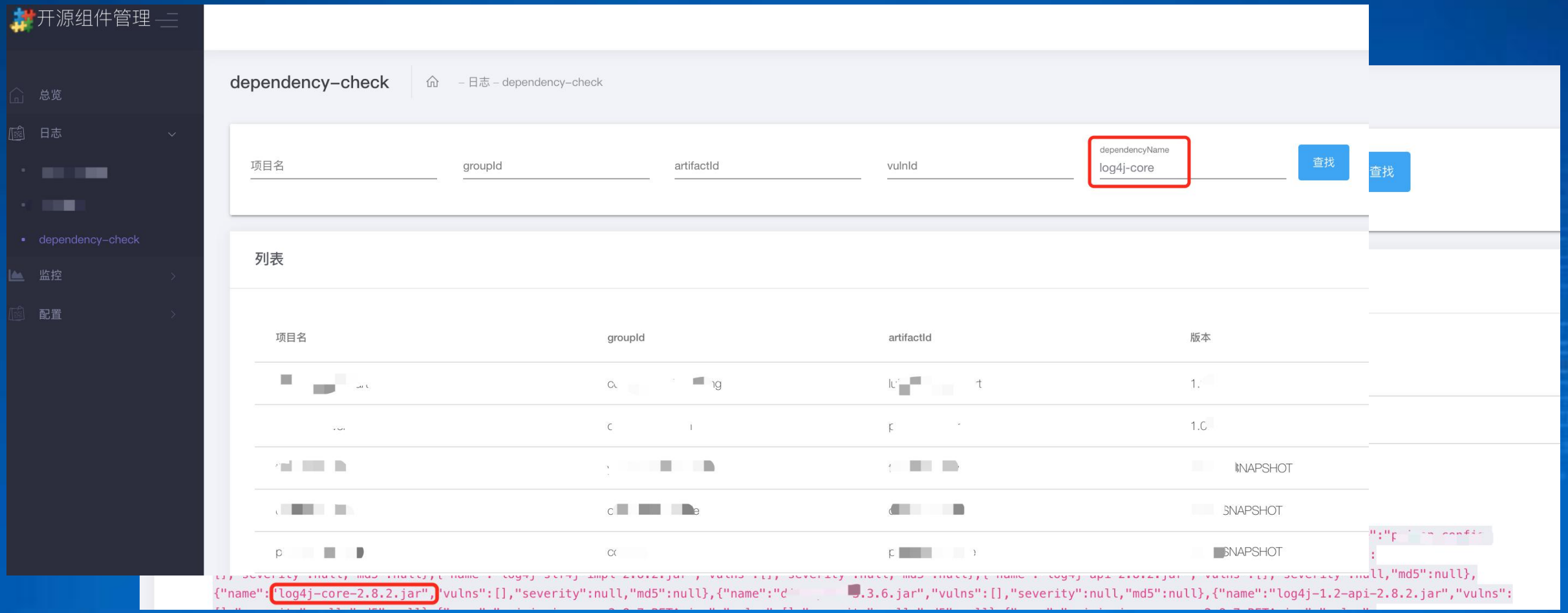
SCA应急响应案例介绍：Log4j2 RCE漏洞响应

1	定位漏洞依赖库信息	根据漏洞情报确定漏洞影响的包名及版本号
2	定位使用了漏洞库的项目	从SCA系统导出依赖了log4j漏洞版本的项目列表
3	推动项目负责人修复漏洞	根据项目信息，推动负责人缓解影响或者升级版本

安世加

工具链建设介绍：SCA

SCA应急响应案例介绍：Log4j2 RCE漏洞响应



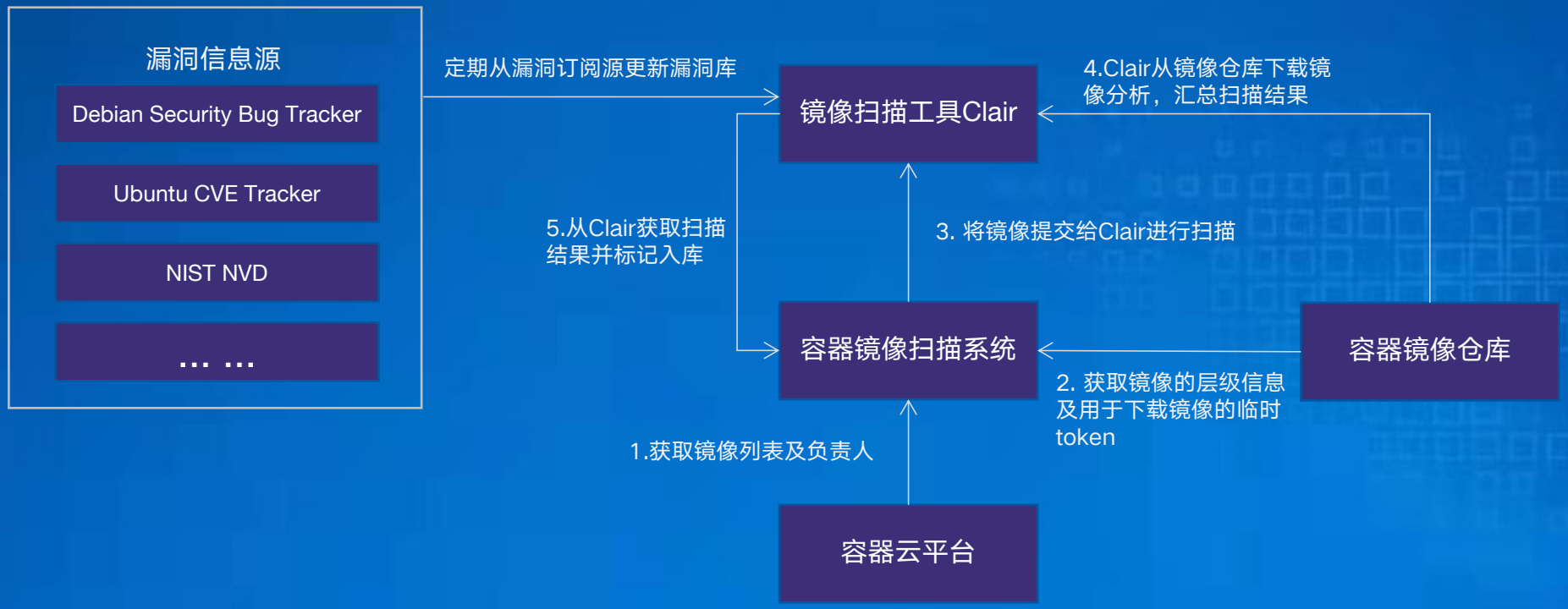
工具链建设介绍：容器镜像扫描

项目背景

- 微服务架构的兴起，容器化部署已经成为时下最流行的生产方式。
- 经过两年来的推进，公司约80%以上的服务部署已迁移至自建k8s容器云平台上。
- 容器是基于镜像的，镜像的安全性决定了容器运行环境的安全性。
- 前期的目标是从镜像安全入手，构建一个针对容器镜像的静态扫描系统，能够发现高危的Nday漏洞

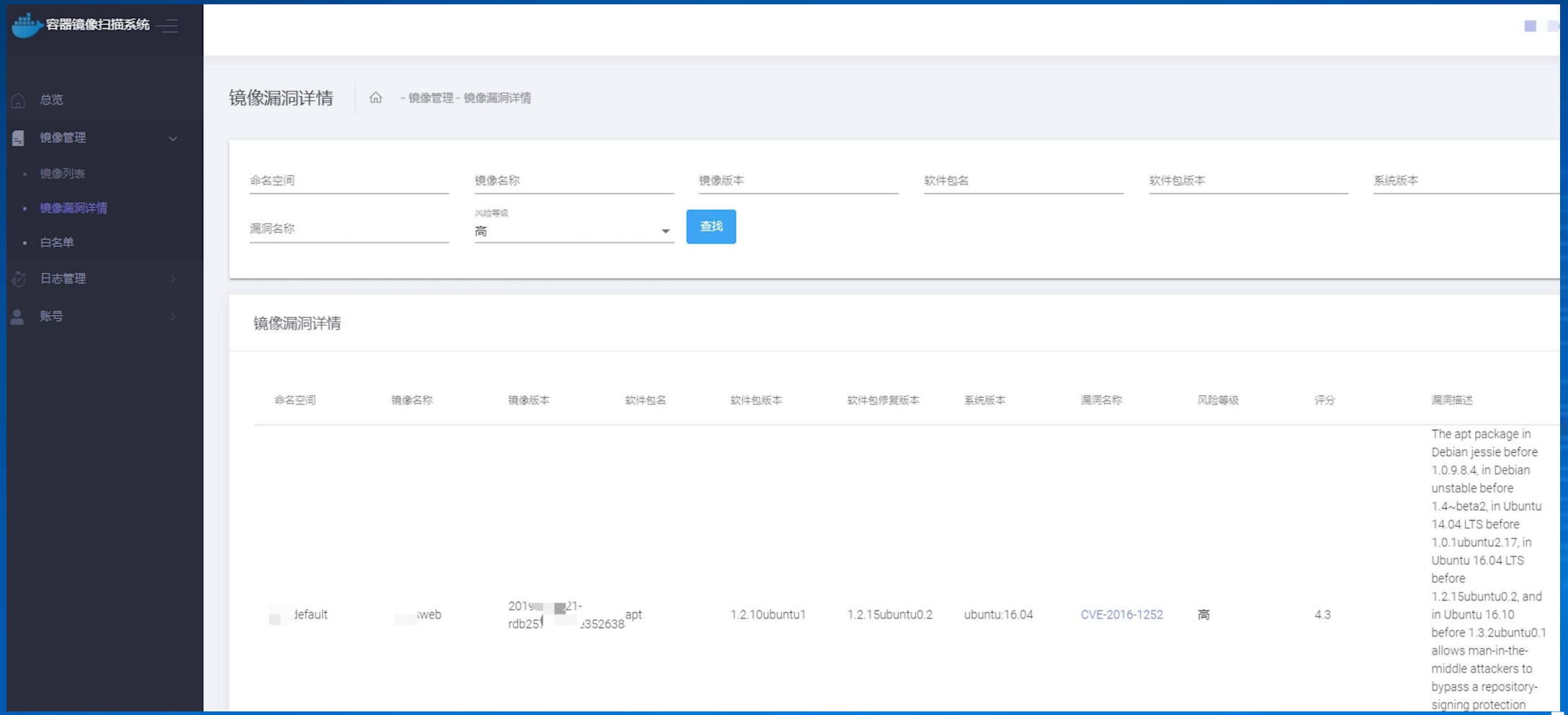
工具链建设介绍：容器镜像扫描

系统架构



工具链建设介绍：容器镜像扫描

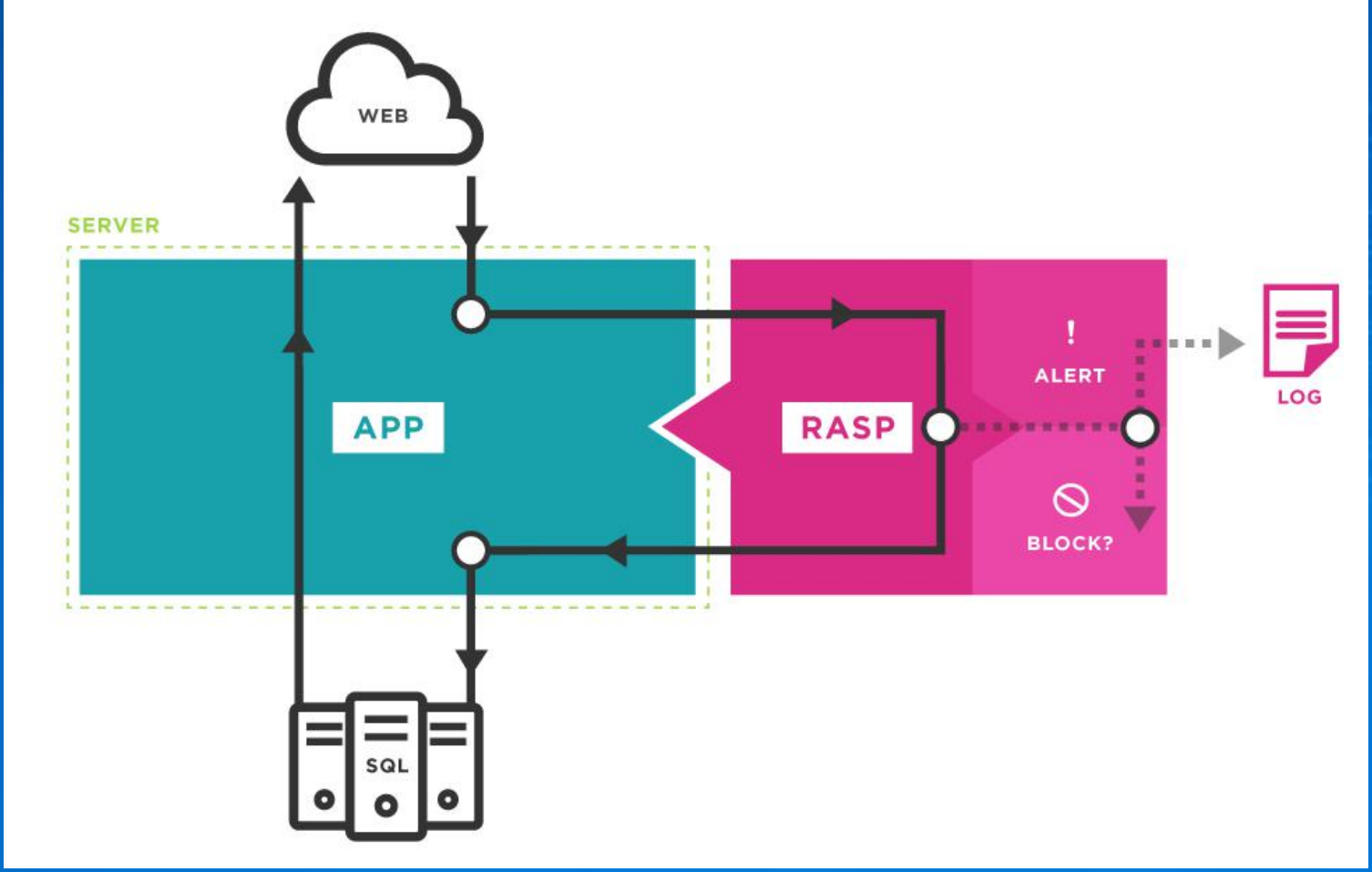
系统展示



工具链建设介绍：RASP

技术简介

RASP（Runtime application self-protection）即运行时应用自我保护，RSAP 将自身注入到应用程序中，与应用程序融为一体，实时监测、阻断攻击，使程序自身拥有自保护的能力。并且应用程序无需在编码时进行任何的修改，只需进行简单的配置即可。



工具链建设介绍：RASP

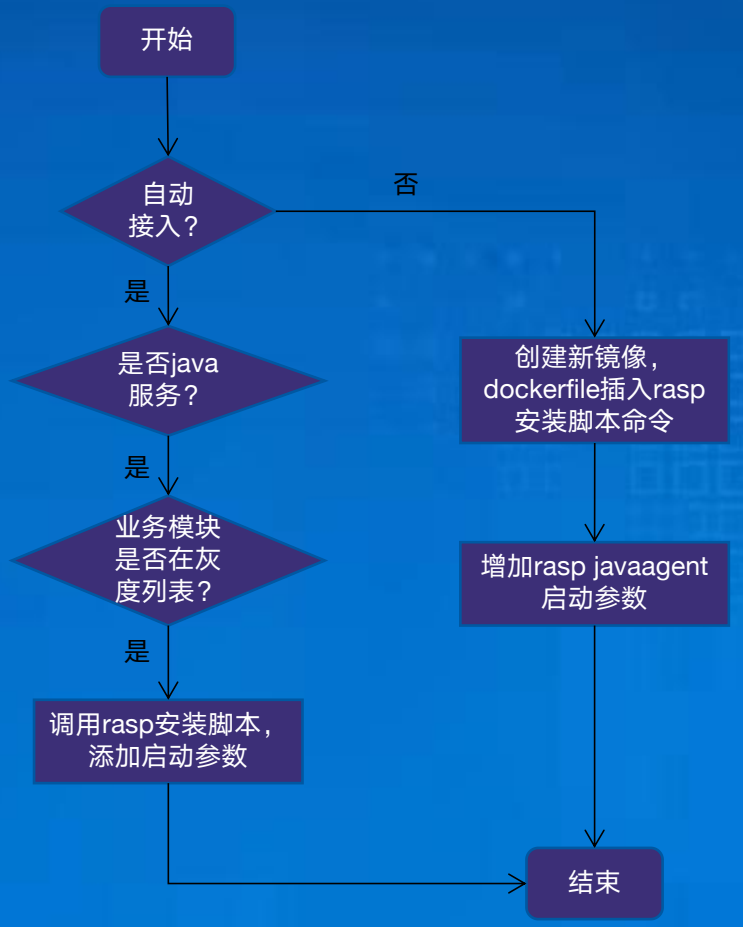
RASP vs WAF

	RASP	WAF
部署	服务器上单独部署，嵌入在应用程序内部，应用代码无感知（java程序启动时加上–javaagent rasp.jar参数即可）	外部边界入口统一部署
	开发语言强相关，但防护插件可共用	支持透明(串联)、旁路、反向代理三种方式
	更了解应用程序上下文	容易形成单点故障，影响面大
性能	由于只在关键点检测，不是所有请求都匹配所有规则	正则匹配的规则越多，性能越低
	openrasp对业务服务器CPU性能影响在2-4%	对业务服务器CPU没有影响
准确性	只在可能发生攻击的关键函数调用处检查是否有恶意内容输入，并监视整个数据逻辑流	基于相对原始的模式匹配对于输入内容进行检测
协议	不限定协议	只能处理能解析的协议（如HTTP）
语言支持	理论上不限制语言，但每种语言需要开发单独的探针（openrasp目前支持Java和PHP）	不受程序设计语言限制
可视性	可以向开发人员提供详细的攻击路径和调用链信息，降低代码漏洞修复的难度	不能获取应用程序内部的信息

工具链建设介绍：RASP

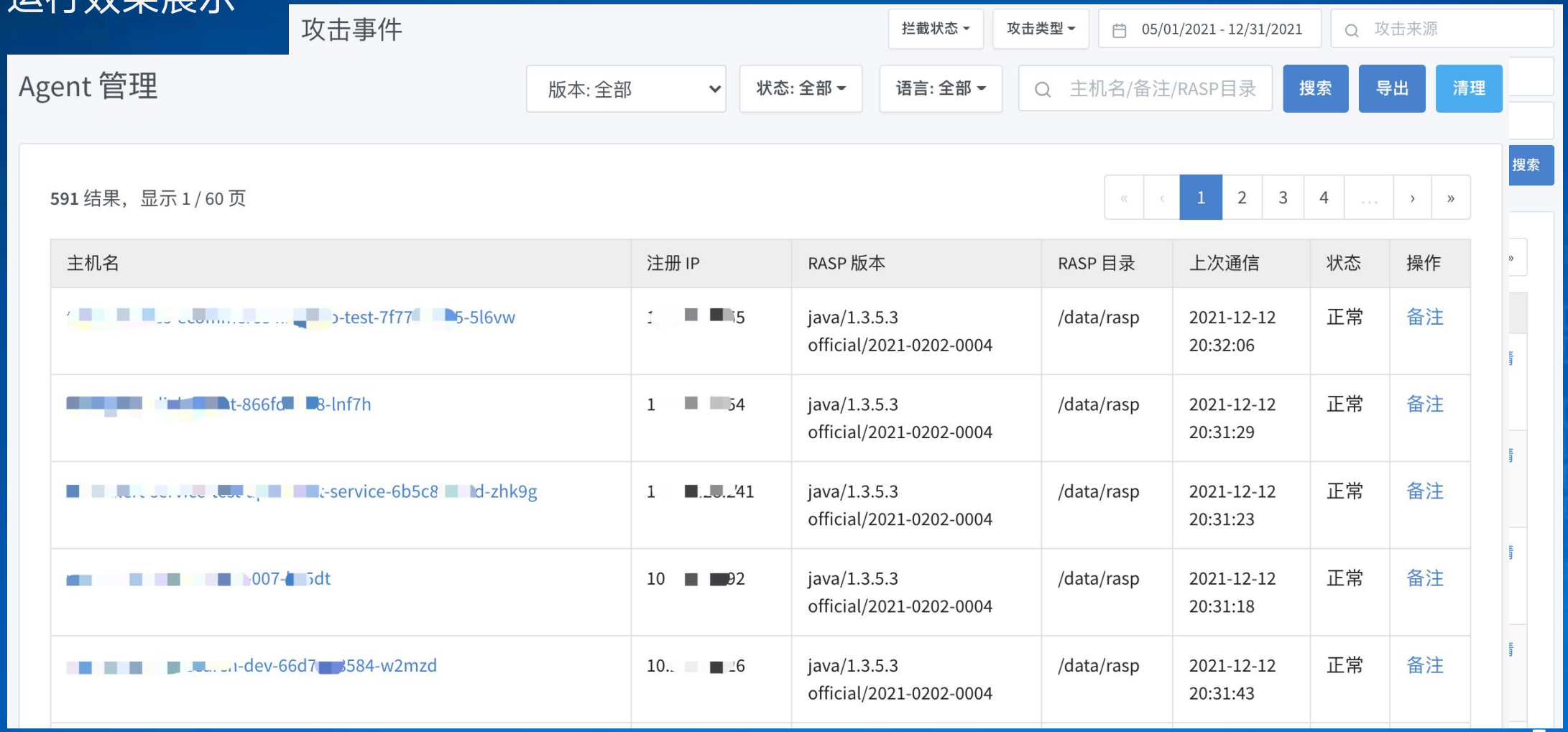
接入流程

- 支持自动接入/手动接入两种模式
- 测试环境默认开启



工具链建设介绍：RASP

运行效果展示



几点思考

- 人人有责：安全是团队所有成员的责任，需要贯穿整个业务生命周期的每个环节
- 安全没有银弹：没有一劳永逸的万能工具，只有因地制宜的落地方案
- 拥抱变化：技术的快速迭代驱使安全从业人员技能革新

安世加

关注我们

安世加专注于网络安全行业，通过互联网平台、线下沙龙、峰会、人才招聘等多种形式，致力于创建亚太地区最好的甲乙双方交流、学习的平台。为培养安全人才，提升行业整体素质，助推安全生态圈的健康发展而努力。

安世加