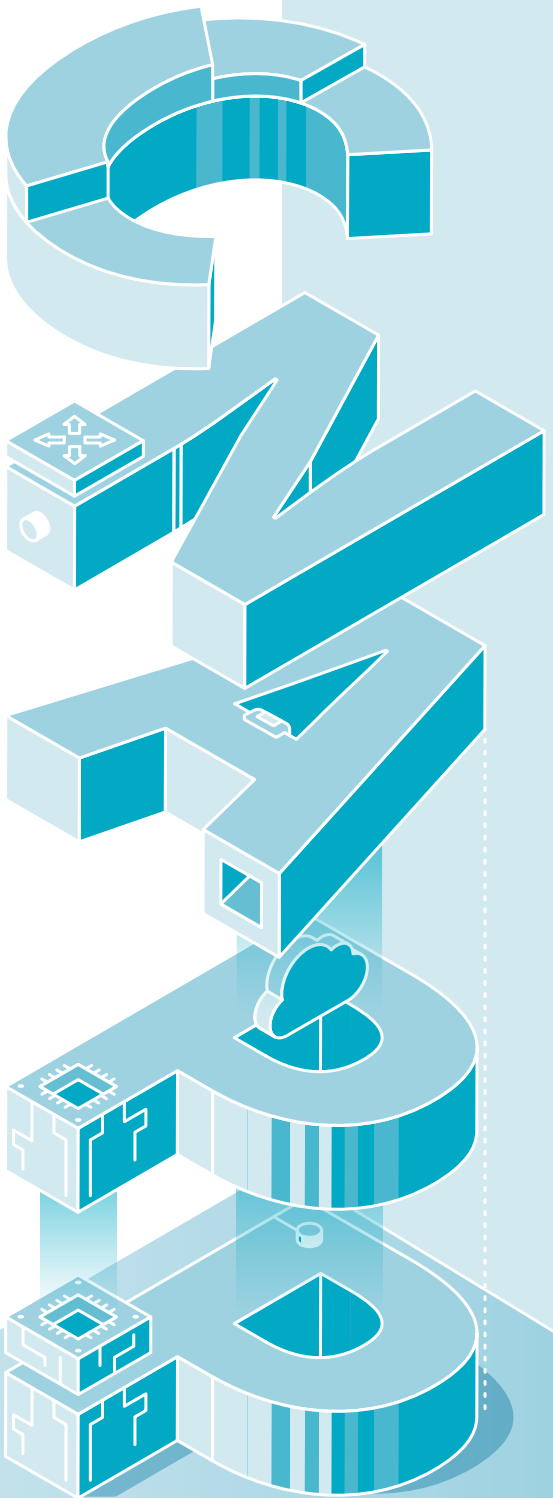
云原生应用保护平台建设指南

（2024年12月）



主编单位

青藤云安全

中国信通院云大所

浙江大学

编写说明

2024 年是网络强国战略提出 10 周年，也是完成“十四五”规划目标任务的关键年。在此背景下，光明网网络安全频道携手中国信息通信研究院云计算与大数据研究所、《信息安全研究》，在浙江大学网络空间安全学院、《中国金融电脑》、青藤云安全等单位支持下，联合推出《安全洞察 · 大咖说》云原生安全专题访谈活动，邀请来自政府、金融、通信、交通、能源、制造、互联网等行业相关负责人，分享数字化转型持续深化路径、新技术应用安全风险防范策略等内容；同时，聚焦前沿趋势及行业实践调研，撰写发布《云原生应用保护平台（CNAPP）建设指南（2024）》，旨在积极发挥行业示范引领作用，降低重复试错成本，为各行业用户提升安全防护策略提供借鉴参考。

参编单位：

青藤云安全

中国信通院云大所

浙江大学

天翼云科技有限公司

北京车和家信息技术有限公司

光明网网络安全频道

《信息安全研究》杂志社

参编人员（排名不分先后）：

胡　俊、李　漫、尹贺杰、杨冬富、杜　岚、仇保琪、申文博、李爱民、宋志明、张　乐、凌　杰、赵　东、李　超、崔　岩、李政葳、刘　昊、潘　静

前言

随着云计算技术的飞速发展，云原生应用已成为推动企业数字化转型的核心力量。然而，随着企业架构向云原生的迁移，传统的安全防护措施面临着前所未有的挑战。云原生环境的动态性、分布式特征以及对微服务架构的依赖，使得安全威胁更加隐蔽和复杂。为了应对这些挑战，云原生应用保护平台（CNAPP）应运而生，它提供了一种全新的安全解决方案，旨在为云原生应用的整个生命周期提供全面的保护。

本报告全面介绍了 CNAPP 的概念定义、价值和技术架构，结合国内外市场的发展现状与技术进展，分析了企业在 CNAPP 实施过程中面临的挑战和机遇。指南深入解析了开发安全、基础设施安全和运行时安全等核心技术，提供了企业 CNAPP 建设的框架内涵、原则和方法。此外，结合运营商、政务、金融、制造业等行业的实践案例，该指南为各类企业提供了 CNAPP 建设的最佳路径和经验借鉴，并展望了未来发展趋势。该指南为企业构建云原生安全体系提供了宝贵的参考，帮助其应对云环境下的复杂安全挑战。

通过本报告的深入分析和实践指导，企业能够构建起一个强大的云原生安全体系，有效应对云环境下的复杂安全挑战，确保业务的连续性和数据的安全性。希望本报告能为企业的云原生安全建设提供宝贵的参考和指导。

CONTENTS

目录

01 概述

（一）概念定义

随着云原生技术在越来越多的企业中获得应用，企业的业务灵活性和效率显著提升，然而，与传统本地的数字化架构相比，云原生技术面临着更多复杂的安全威胁。云环境的动态性、多租户特性以及容器和微服务等新兴架构增加了潜在的攻击面，传统的安全策略和工具无法充分应对这些新威胁。因此，企业必须采取更加全面和集成的安全解决方案，以保障云原生架构的安全性。在这一背景下，云原生应用保护平台（CNAPP）应运而生。

云原生应用保护平台（CNAPP）的全称是 Cloud-Native Application Protection Platform，这一概念最早由 Gartner 公司在 2021 年提出。Gartner 在其报告中引入了 CNAPP 的概念，旨在通过集成一系列云原生安全能力来提供从开发到运行的全生命周期的安全保护，包括云安全配置管理、云工作负载保护、容器安全等功能。CNAPP 的目标是通过统一的视角和自动化手段来应对现代云原生应用的复杂性和安全挑战。Gartner 对云原生应用保护平台（CNAPP）的定义是：CNAPP 是一组集成的安全与合规能力，旨在保护云原生应用的整个生命周期，从开发到部署及运行时安全。它通过整合云安全配置管理（CSPM）、云工作负载保护平台（CWPP）、身份和访问管理（IAM）、DevSecOps 集成以及运行时安全等功能，提供对云环境的统一可视化和控制。CNAPP 帮助企业摆脱使用多个分散安全工具的复杂性，以更全面的方式应对云原生架构中的独特安全挑战。

（二）价值阐述

云原生应用保护平台（CNAPP）为企业提供了从开发到运营的全面安全解决方案，简化了安全管理，提升了合规性，降低了安全风险和运维成本，成为企业应对云原生架构安全挑战的核心工具。

首先在安全性方面，云原生应用保护平台（CNAPP）为企业提供全生命周期的安全防护，特别是针对云原生架构中的现代化安全威胁。它在开发阶段落实安全左移的理念，确保在应用设计和开发之初就引入必要的安全措施，同时覆盖应用在生产环境中的持续监控与防护，减少安全盲区。同时，CNAPP 支持企业将安全集成到开发流程中，推动 DevSecOps 的实践。通过在开发阶段引入自动化的安全扫描和风险检测工具，CNAPP 能帮助开发人员及早发现和修复安全问题，实现安全左移，减少后期修复漏洞的成本和风险。此外，CNAPP 针对云原生架构的独特挑战，如容器逃逸、镜像漏洞和跨租户攻击，提供定制化的防护措施，

通过集成多层次的安全功能（如身份和访问管理、微服务通信加密、容器镜像扫描等），应对现代化的安全威胁，确保应用的安全性和合规性。

其次在管理效率方面，CNAPP 整合了云安全配置管理（CSPM）、云工作负载保护（CWPP）、容器和 Kubernetes 安全等功能，通过统一的平台提供全局的安全可视化视图，便于企业实时监控和响应安全威胁，提升了安全运营的效率。传统的安全工具通常需要大量手动操作和管理，增加了企业的运维负担。CNAPP 的自动化和整合性大幅减少了复杂的运维流程，简化了安全策略的实施与调整。此外，CNAPP 提供了集中化的管理平台，减少了多工具协作中的信息孤岛现象，从而提升整体安全运营效率并降低运维成本。

最后在合规方面，CNAPP 利用自动化技术帮助企业持续监测其云环境的安全配置，确保符合行业标准和法律法规的合规性要求。通过自动识别错误配置和潜在风险，CNAPP 可以减少人工干预，降低错误发生的可能性，确保云环境始终保持在合规状态下运行。

02 发展现状

（一）国内外市场发展现状

云原生安全关注度提升，CNAPP 建设价值逐步凸显。目前越来越多公司对云原生应用程序所暴露的风险尤为重视，通过部署 CNAPP 产品来保护云原生应用程序，并应对不断扩大的攻击面。通过对 CNAPP 输出的检测结果运营，可防止运行时环境中的威胁，减轻云基础设施中的错误配置，并在整个开发过程中将安全嵌入。

根据 Frost&Sullivan 最近对 2360 多名首席信息安全官和 C-level 级别的领导者进行的客户研究，大多数的组织（31%）都部署云安全技术来防止漏洞的产生，并检测和应对云上各类威胁（30%）。许多公司还投身于云安全解决方案，以应对未知威胁（24%）和监管合规性（12%）。这表明全球企业都对云安全的认识有了显著提高。

48% 的受访组织目前使用 CWPP，而 41% 的组织计划在未来两年内部署类似产品，只有 10% 的组织表示，他们不打算在未来几年增加解决方案。这些发现与采用其他云安全解决方案相一致，包括 CSPM、SaaS 安全态势管理（SSPM）、CIEM 和 CNAPP。

2023 年，全球 CNAPP 市场收入为 38.784 亿美元，同比增长 31.3%。Frost&Sullivan 预计接下来的五年，这一势头将以 22.8% 的复合年增长率持续下去，由于对整体云原生安全解决方案的需求不断增加，2028 年的收入将达到 108.188 亿美元。

Gartner 对 CNAPP 市场也做出相应预测，具体预测内容如表 1 所示。

产品类型	2023年底市场规模	2024年预测增长
CSPM	14亿	26.3%
应用安全测试软件	18亿	27%
CWPP	39亿	27.9%
漏洞管理工具	22亿	14.8%
WEB应用和API保护	17亿	16.2%

表1 Gartner CNAPP市场预测

Gartner 认为，到 2029 年，60% 未在其云架构中部署统一 CNAPP 解决方案的企业将缺乏对云攻击面的广泛可见性，因此无法实现其预期的零信任目标；超过 80% 的企业将采用集中式平台工程和运营方法来促进 DevOps 自助服务和扩展，而 2023 年这一比例不到 30%；35% 的企业应用程序将在容器中运行，比 2023 年的不到 15% 有所增加。

（二）技术发展现状

CNAPP 旨在通过一体化平台提升云原生应用全生命周期的安全性。CNAPP 旨在提升云原生应用的可观测性和安全性，整合多云环境中的资产信息，并重点保护 IaaS 和 PaaS 公有云环境及其工作负载和应用。现阶段，CNAPP 正处于基础能力扩展阶段，厂商们纷纷从各自专注的领域（如 CSPM、CWPP、DevSecOps）转向 CNAPP 的构建，通过引入更多功能提升产品覆盖范围。CNAPP 的核心功能包括云平台与 Kubernetes 的集成、风险分析与优先排序、实时或快照的运行时状态分析以及攻击路径分析。

CNAPP 工作负载运行部署方式遭遇瓶颈。然而，目前在 CNAPP 的工作负载运行时监测中，关于是否采用代理的争论仍然存在。随着 eBPF 技术的兴起，代理对工作负载的可见性得到了显著提升，而 DevOps 技术的发展则使得代理的部署更加方便快捷。基于代理的解决方案能够实时检测和缓解威胁，帮助安全团队更快速地响应事件。此外，代理还能获取细粒度的数据，提供对工作负载的深度可见性，使安全运营人员能够获得更深入的洞察。然而，在处理边缘工作负载（如无服务器架构和 PaaS）时，这类方案面临资源覆盖不足和高昂运营成本的挑战。

为了解决用户的痛点，厂商们积极探索无需代理的监测方案，使用户无需安装任何 Agent 即可全面了解云工作负载的状态。这种方案通过云 API 对底层实例的硬盘进行快照，由安全厂商对快照数据进行扫描，既实现了对工作负载的全面监控，又不影响业务的连续性。此外，它还提供了统一的安全态势视图，简化了跨平台的管理。然而，该方法也存在一定的局限性。首先，部分容器的生命周期较短，即使快照扫描速度很快，仍可能无法实时捕捉所有容器的活动。其次，依赖云 API 进行数据收集在深度洞察安全数据方面有所不足。最后，云上工作负载产生的存储数据量庞大，可能带来高额的云存储成本。

CNAPP 注重敏捷开发，重视不可变基础设施。如今，CNAPP 运行在容器中，构建于离散的代码功能之上，这些功能以松散耦合的微服务形式操作，通常通过 API 相互通信。在支持频繁更新的 DevOps 风格持续集成（CI）/ 持续交付（CD）管道中开发，使工作负载及其微服务更加短暂。它们还使用自定义代码和开源代码以及来自开源或私有源代码库的库。CNAPP 部署在编程化的云基础设施上，使应用程序与基础设施的依赖关系隔离，并利用云的共享基础设施，根据业务需求以弹性方式进行扩展和收缩。CNAPP 优先考虑不可变性，生产环境中的变化很少或没有（所有生产环境中的变化都通过开发管道驱动）。

总的来说，CNAPP 在未来还需要不断深化威胁检测、响应速度和数据分析方面的能力，弥补“有代理”与“无代理”方案的不足，实现更高效的安全监测与防护。此外，用户的体验也不容忽视，需要持续优化用户界面和操作流程，提供更加直观和友好的使用体验，降低企业在部署和使用 CNAPP 时的技术门槛。

（三）现有的挑战和机遇

1. CNAPP面临来自组织架构、产品融合、多云环境等诸多挑战

1 **安全团队在CNAPP产品的责任划分**	众所周知 CNAPP 所能覆盖的资产范围比较广泛，而如今很多组织都会有多个安全团队共同承担云原生应用程序的安全责任，包括云安全、数据安全、应用安全、应急响应、安全架构等不同领域的团队。每个团队都拥有解决部分云风险的责任，但在产品运营和产品规划上，团队之间如果缺少紧密地协作，与明确的职责划分，那么当出现安全风险时会影响风险地快速处置以及安全治理。
2 **原有产品与CNAPP结合**	许多组织已部署了各类安全产品，以解决从代码、到云上的安全性与合规性。比如运行时保护（CWP），端点检测和响应（EDR），云安全态势管理（CSPM）等各类解决安全产品。 而随着 CNAPP 的发展，原有的产品可能已不能满足当前安全方面完整的需求，CNAPP 产品部署的同时需要考虑与原有产品的结合或者是兼容性，以免带来预期之外的影响。 并且也要考虑避免部署完整 CNAPP 所带来的运营量，告警策略需严格把控，防止大量告警使高风险问题被海量告警淹没，造成信息资产损失。

3

部署架构及多云场景

目前 CNAPP 产品还存在对过多云环境的兼容问题，与不同的云平台的配置集成以及使用体验，都是需要设计优化的地方。如果同一类型的告警需要针对不同云单独开启或配置，那么会大大增加安全运营人员的治理成本。而对于多云环境的挑战不仅仅在于产品的使用上，不同平台的 API 或是 SDK，都有各种的不一致，CNAPP 数据的结构化和数据上下文也是需要非常投入精力的，这对于全面理解和解决风险的场景与优先级至关重要，。

而一些 CNAPP 产品仅提供 SaaS 服务，另一些则设计为在客户环境中运行。SaaS 服务部署的产品则要额外考虑资产的数据安全，是否会由于部署 CNAPP 产品产生更多的暴露面，最优解决方案应当采用分布式云架构，有云管理的控制平面和用户控制的分布式检测（例如，本地扫描容器或快照，无需上传到 SaaS 服务）。一些解决方案不提供数据扫描位置选择，强制用户在公共云环境中扫描，增加了计算成本并抑制 CNAPP 的采用。

2. AI能力提升为CNAPP发展带来新的机遇

云原生应用程序保护平台（CNAPP）的发展进入了一个技术变革的新阶段。随着人工智能（AI）技术的不断成熟，CNAPP 与 AI 的融合呈现较大的发展机遇。

AI 的集成能够极大地增强 CNAPP 的能力和运营效率。首先，AI 可以提供强大的自动化能力。通过机器学习算法，AI 可以自动识别安全漏洞、异常行为和潜在威胁，极大地减少了人工干预和操作的需求。此外，AI 可以快速处理和分析大量数据，从而即时发现并响应安全事件。这不仅提高了响应速度，还增加了检测的准确性。

其次，AI 的风险预测能力能够提供前瞻性保护。通过对历史安全数据进行分析挖掘，AI 能够预测潜在的攻击向量和新兴威胁，从而提前采取防御措施，增强安全态势感知。此类预防性保护措施有助于降低攻击成功的概率，保护企业的数字资产和业务连续性。

并且 AI 能力的嵌入，可以通过多维度数据整合，帮助企业打破数据孤岛，形成全局的一体化风险视图，从而实现更高效的风险管理。通过 AI 驱动的威胁情报分析和自动化漏报处理，使 CNAPP 可以显著减少误报数量，并优先处理高危漏洞，提高漏洞修复的效率和准确性。

综上所述，AI 与 CNAPP 的融合将极大地提升云原生应用程序的安全保护能力。通过强化自动化、风险预测、数据集成和漏洞管理，AI 不仅解决了现有的痛点，也为 CNAPP 的发展开辟了新的空间和机遇。

03 核心技术和能力解析

（一）开发安全

1. 源码安全

云原生应用开发过程中往往面临着诸多安全挑战，这些挑战包括源代码的机密性、完整性和可用性保护需求，源码安全相关的技术和重要性阐述如下所示。

（1）源码安全的重要性

核心资产保护：源代码是软件开发公司的核心资产，包含了技术细节、创意设计、商业计划等敏感信息，其安全性直接关系到公司的知识产权和商业利益。

避免经济损失：源代码泄露可能导致被其他公司恶意抄袭，开发类似产品，从而给公司带来巨大的经济损失。

信誉与信任：源码泄露还可能引发安全问题，损害公司声誉，失去用户信任。

（2）源码安全核心技术解析

① 数据加密技术

数据加密技术包括透明加密和全磁盘加密两种，透明加密指的是不影响开发人员正常工作的前提下，对源代码进行加密，确保其在存储和传输过程中的安全性全磁盘加密指的是对整个开发环境或存储设备进行加密，防止物理介质丢失导致的源码泄露。

其中，透明加密技术，又称自动加密技术或无感加密，是一种在数据保存或传输过程中自动对数据进行加密，而在用户访问时自动解密的技术。透明加密技术通常与操作系统、文件系统或应用程序紧密结合，其核心思想是将加密和解密过程隐藏在操作系统内部，使得用户无感知。具体来说，透明加密软件会在操作系统内核中嵌入一个虚拟加密层，这个层会将用户的文件系统访问请求拦截下来，然后对文件进行加密或解密操作。加密过程：当数据被写入磁盘、通过网络发送或通过特定应用程序处理时，加密引擎会自动识别并加密这些数据。解密过程：当用户需要访问这些数据时，加密引擎会再次介入，自动解密数据以供用户使用。

全磁盘加密技术，又称全卷加密或全驱动器加密，是一种在操作系统级别对磁盘上的所有数据（包括操作系统本身、用户文件、系统文件等）进行加密的技术。其目的是防止未授权的物理访问（如盗窃硬盘、非法访问等）导致的数据泄露。即使磁盘被非法获取，未经授权的访问者也无法读取加密后的数据。全磁盘加密技术通常通过以下步骤实现：首先，进行加密密钥的生成与管理，由系统生成一个或多个加密密钥，这些密钥用于加密和解密磁盘上的数据。密钥也可以存储在专用的硬件安全模块（HSM）中，或者通过其他安全方式进行管理和保护。加密过程中，当磁盘写入数据时，加密引擎会自动拦截这些数据，并使用加密密钥对数据进行加密。加密后的数据以密文形式存储在磁盘上。在解密过程，当系统需要读取磁盘上的数据时，加密引擎会自动识别并解密这些数据。解密后的数据以明文形式提供给系统或应用程序使用。

② 访问控制技术

访问控制技术包括细粒度权限管理和身份认证与授权，其中细粒度权限管理指的是根据员工的角色和职责，设置不同的访问权限，确保只有授权人员才能访问源代码。身份认证与授权指的是通过强身份认证机制，如多因素认证，确保只有合法用户才能访问源码仓库。

具体而言，细粒度权限管理（Fine-Grained Access Control, FGAC）是指对系统资源或功能的访问权限进行细致划分的授权方式。它允许对单个数据项或操作进行权限控制，而不是像粗粒度权限管理那样，仅对整个资源集或操作集进行统一控制。其目的在于提高数据安全性，防止未经授权的访问和滥用，同时确保合法用户能够高效地使用所需资源。细粒度权限管理技术通常基于以下原则进行工作：一是角色与权限定义：即定义不同的用户角色，并为每个角色分配相应的权限。权限可以具体到某个数据项、某个操作或某个时间段等。二是访问控制策略，即制定详细的访问控制策略，明确哪些用户或角色可以访问哪些资源或执行哪些操作。策略可以基于用户的身份、角色、上下文信息（如时间、地点、设备类型等）进行动态调整。

权限验证与授权是指当用户尝试访问资源或执行操作时，系统会进行权限验证。根据用户的身份、角色和访问控制策略，判断用户是否具有相应的权限。如果用户具有相应权限，则允许访问或执行操作；否则，拒绝访问并可能记录日志或触发警报。身份认证是确认用户身份的过程，它要求用户提供身份证明（如用户名和密码、生物特征等），系统通过验证这些证明来确定用户是否有权访问系统或资源。常用方式包括：用户名和密码、多因素认证、单点登录（SSO）、OAuth/OpenID Connect（第三方身份认证服务）等。

③ 安全审计技术

安全审计主要指的是审计日志，即记录所有对源代码的访问和操作行为，形成审计日志，便于后续

追溯和审查。审计日志是一种记录系统或应用活动详细信息的日志，它记录了用户操作、系统事件、错误信息等关键数据，用于后续的安全分析、问题排查和合规性检查。审计日志通常包含以下内容：一是用户活动，如登录、注销、权限变更、文件操作等；二是系统事件，如服务启动、停止、异常、错误等。三是时间戳：记录事件发生的确切时间，以便进行时间线分析和追溯。四是来源 IP：记录操作或事件的发起者 IP 地址，有助于追踪攻击者。五是详细操作信息：如操作的具体内容、参数、结果等，以便进行深入的安全分析。

④ 代码审查与安全扫描

代码审查与安全扫描是源码安全的核心部分，主要包括静态代码分析，指的是在代码编写阶段，通过工具自动检查代码中的安全漏洞和潜在错误。静态代码分析能够在不执行代码的情况下，通过检查源代码的语法、结构、数据流等特性，来发现潜在问题、错误和安全漏洞的技术。不仅能够在早期发现问题，在编译和运行代码之前就发现潜在的问题，如语法错误、逻辑错误、安全漏洞等，减少后期调试和修复的成本。而且提高代码质量：帮助开发人员遵循最佳实践和代码规范，如代码风格、命名规则、注释等，提高代码的可读性和可维护性。还能够增强安全性，识别和修复代码中的安全漏洞和潜在的安全风险，如 SQL 注入、跨站脚本攻击（XSS）、缓冲区溢出等，提高应用程序的安全性。静态代码分析的工作原理主要包括以下几个步骤：

词法分析：将源代码分解为一系列的标记（token），如变量名、关键字、运算符等，建立代码的基本语法结构。

语法分析：将词法分析器生成的标记按照语法规则进行解析，建立抽象语法树（AST），表示代码的结构和关系。

数据流分析：通过分析代码中的数据流和变量的使用情况，来检测未初始化的变量、空指针引用、不可达代码等问题。

规则检查：基于预定义的规则或模式，检查代码中的常见问题、最佳实践和安全漏洞。

生成报告：将发现的问题、潜在影响以及修复建议生成详细的报告，供开发人员参考。

⑤ 物理与网络安全措施

源码安全中的物理安全措施技术是一系列旨在通过物理手段保护源代码不被非法访问、窃取或破坏的技术和策略。这些措施主要关注于硬件、网络环境和物理空间的安全管理。物理隔离是最直接且有效的防止源代码泄露的手段之一。企业应将开发环境与外部网络物理隔离，使用专用的开发网络，并禁止外部设备（如 U 盘、移动硬盘）连接到开发环境。这样可以从物理层面切断潜在的泄露途径，确保源代

码在封闭、可控的环境中运行和存储。同时，为了防止源代码因意外情况（如硬件故障、自然灾害）而丢失，企业应建立完善的备份与恢复策略。这包括定期备份源代码、将备份数据存储在安全的物理位置、以及制定详细的恢复计划等。在发生意外情况时，企业能够迅速恢复源代码，减少损失。网络安全措施则是通过使用专用的 VPN 设备和 https 安全隧道等技术，与源码仓库进行对接，从而进行数据交互，能够确保远程访问源代码时的数据传输安全性。

2. 制品安全

在云原生环境中，制品通常指的是应用程序的构建块，如容器镜像、软件包等。这些制品在开发、构建、部署和运行的各个阶段都可能面临安全风险。因此，确保制品的安全性是保障云原生应用整体安全性的关键。

企业能够通过整合多种安全功能，为制品安全提供了全面的保护。这些功能包括但不限于：

（1）容器镜像扫描

越来越多的企业采用容器化应用来加速软件开发和部署。然而，容器镜像作为创建容器的模板，其安全性直接关系到整个应用的安全性。因此，对容器镜像进行安全扫描，及时发现并修复其中的安全漏洞，成为保障云原生应用安全的重要手段。

容器镜像扫描技术主要基于以下原理进行：

文件系统遍历：扫描工具会遍历容器镜像中的所有文件系统层，逐个检查其中的软件包（Package）和文件。

漏洞库匹配：扫描工具会维护一个包含已知安全漏洞的数据库（如 CVE 漏洞库），并将镜像中的软件包版本与漏洞库中的信息进行匹配，以发现是否存在已知的安全漏洞。

恶意文件检测：除了漏洞检测外，扫描工具还会通过一些文件识别的扫描库（如 Yara）来识别镜像中的恶意文件。此外，某些商业安全软件能够结合木马库、病毒库来识别镜像中的风险文件。

容器镜像扫描的过程通常包括以下几个步骤：

镜像上传：用户将需要扫描的容器镜像上传到镜像仓库或扫描平台。

扫描触发：扫描可以是由用户手动触发，也可以设置为定时任务自动触发。

扫描执行：扫描工具开始执行扫描任务，遍历镜像文件系统，进行漏洞库匹配和恶意文件检测。

结果生成：扫描完成后，扫描工具会生成详细的扫描报告，列出镜像中存在的安全漏洞和恶意文件信息。

结果处理：用户根据扫描报告中的信息，对镜像进行修复或选择是否继续部署。

容器镜像扫描技术具有以下特点和优势：

高效性：扫描工具能够快速遍历镜像文件系统，并与漏洞库进行匹配，提高扫描效率。

全面性：扫描工具能够覆盖镜像中的所有软件包和文件，确保漏洞和恶意文件的全面检测。

准确性：通过维护最新的漏洞库和恶意文件识别库，扫描工具能够提供准确的扫描结果。

灵活性：扫描工具可以根据用户需求进行定制化配置，如设置不同的扫描规则、报告生成方式等。

（2）软件供应链安全

软件供应链包括软件产品的开发、编译、构建、部署、运营等全过程，涉及到众多的参与方和组件。软件供应链安全旨在确保软件产品在整个生命周期内都是可信的、安全的。这包括对软件开发过程中的代码审查、漏洞扫描、组件来源管理等，以及对软件部署和运营过程中的安全加固、风险评估和监控等措施。软件供应链安全是保障企业信息安全和业务连续性的重要环节。近年来，由于软件供应链安全问题导致的安全事件频发，给企业和个人带来了巨大的经济损失和声誉损失。同时，软件供应链安全也是国家信息安全的重要组成部分，对于保障国家安全和社会稳定具有重要意义。

软件供应链安全面临的挑战多种多样，主要包括：一是恶意代码注入：攻击者通过在软件供应链中注入恶意代码，实现对软件的篡改和破坏，从而达到窃取数据、破坏系统等目的。二是组件漏洞：软件组件中存在的漏洞往往会被攻击者利用，导致软件产品的安全性受到威胁。三是非法软件组件：软件产品中可能存在非法的软件组件，这些组件可能携带恶意代码或者存在版权问题。四是供应链管理不规范：软件供应链管理不规范可能导致参与方之间的信息不透明，使得攻击者有机可乘。

为了应对上述挑战，软件供应链安全技术不断发展，主要包括以下几个方面：

1

代码审查与漏洞扫描

代码审查与漏洞扫描主要包括：静态应用程序安全测试（SAST），即不运行被测程序本身，仅通过分析或检查源程序的语法、结构、过程、接口等来检查程序的正确性。动态应用程序安全测试（DAST），即通过模拟攻击来测试应用程序的安全性，能够发现运行时漏洞。交互式应用程序安全测试（IAST），即结合 SAST 和 DAST 的优点，在应用程序运行时提供详细的漏洞信息和上下文。

2 组件来源管理	组件来源管理指的一是对软件组件进行严格的来源审查，确保组件的合法性和安全性。二是使用可信的组件库和版本控制系统，避免使用存在已知漏洞的组件。
3 安全加固与风险评估	安全加固与风险评估一是指对软件进行安全加固，如加固代码、限制权限等，而是指定期进行风险评估，识别潜在的安全威胁，并采取相应的应对措施。
4 监控与应急响应	监控和应急响应首先要建立完善的监控体系，对软件供应链中的各个环节进行实时监控。其次，是需要制定应急响应计划，一旦发生安全事件，能够迅速响应并处理。
5 自动化安全工具	利用自动化安全工具进行软件供应链安全防御，如自动化代码审查工具、自动化漏洞扫描工具等。这些工具能够显著提高安全检测的效率和准确性，降低人为错误的风险。

（二）云原生基础设施安全

1. 云基础设施权限管理

云基础设施权限管理（Cloud Infrastructure Entitlement Management, CIEM）是确保云环境中访问权限和身份验证得到有效控制的关键组件，主要用于管理和控制云平台的使用和访问权限。通过对云平台进行身份权限管理，检测用户账号是否存在过度授权、密码过期等问题，帮助管理人员及时发现并解决授权管理方面的问题，提高云平台的安全性和可靠性。

大多数 CIEM 解决方案都提供了一个集中的仪表板，用于跟踪和控制分散在云上的资源、服务和管理帐户的访问权限。领先的 CIEM 解决方案提供了人工智能分析和评估工具，可以智能地识别和排序与配置错误、影子管理帐户以及人员、应用程序和机器身份的过度授权相关的风险。这有助于云安全团队优先处理补救措施，同时制定积极主动、知情的分阶段降低风险的方法。

（1）CIEM 的核心功能

权限管理

CIEM 能够检测、自动调整并持续监视云环境中所有标识的权限，确保它们符合最小特权原则。

跨云权限发现：提供关键云平台的精细和标准化指标，帮助组织了解哪些权限被授予了哪些身份。

权限优化：通过实施严格的访问控制，优化权限配置，减少不必要的权限授予。

异常检测

CIEM 能够使用机器学习和分析技术来检测帐户权限中的异常事件，如特权累积、休眠和不必要的权限。

异常的实时警报和响应：当检测到潜在的安全威胁时，CIEM 能够向安全和 DevOps 团队发送警报，并提供详细的取证报告。

合规性支持

CIEM 解决方案持续监视访问活动，以识别过时的标识与范围合适的活动权限，从而帮助组织遵守用户权限的合规性法规和标准。

可视化与监控

CIEM 提供权限可视化功能，将数据点集中到简洁、可操作的见解中，使安全和 DevOps 团队能够有效地监控云安全状况和用户对云资源的访问。

（2）CIEM 的主要作用

整合安全策略

CIEM 作为 CNAPP 的一部分，确保云基础设施的访问权限管理与其他安全组件（如 CWPP、CSPM）紧密集成，形成强大的防御体系。

最小特权原则

CIEM 在云环境中实施最小特权原则，确保用户只能访问其完成工作所必需的资源。这有助于减少潜在的安全风险，防止恶意用户或内部威胁者利用过多权限进行非法活动。

2. 基础设施即代码（IaC）

IaC 是一种将基础设施、工具和服务以及对这些基础设施的管理作为软件系统来处理的方法。它采用软件工程实践，以可重复的、可靠的方式来设计、改变和部署软件环境。通过代码来管理和配置基础设施，可以极大地提高基础设施的自动化程度，减少人为错误，提高部署效率。

（1）IaC 带来的安全优势

提高自动化程度：IaC 通过自动化工具来管理和配置基础设施，减少了手动操作，从而降低了人为错误的风险。

增强一致性：通过代码定义基础设施，可以确保每次部署的环境都是一致的，消除了配置漂移的问题。

提高可见性：IaC 使得基础设施的配置和管理过程更加透明，便于监控和审计。

促进团队协作：IaC 使得开发人员和运维人员可以更加紧密地协作，共同管理基础设施。

（2）IaC 面临的安全挑战

代码安全性：IaC 的配置文件本身也是代码，因此可能存在代码漏洞或配置不当的问题。如果攻击者能够修改这些配置文件，就可能对基础设施造成破坏。

权限管理：在 IaC 环境中，需要严格管理对基础设施配置文件的访问权限，以防止未授权访问或修改。

版本控制：IaC 配置文件应该纳入版本控制系统，以便跟踪和管理变更历史。但是，这也可能引入新的安全风险，如代码泄露或版本回退导致的安全问题。

（3）IaC 安全的最佳实践

使用安全的编码实践：在编写 IaC 配置文件时，应遵循安全的编码实践，如避免硬编码敏感信息、使用参数化查询等。

实施严格的权限管理：对 IaC 配置文件的访问权限进行严格控制，确保只有授权人员才能访问和修改这些文件。

加强版本控制：将 IaC 配置文件纳入版本控制系统，并定期进行代码审查和安全扫描，以发现和修复潜在的安全问题。

使用安全的部署流程：在部署 IaC 配置文件时，应使用安全的部署流程，如使用加密通道传输配置文件、在部署前进行安全验证等。

持续监控和审计：对 IaC 环境进行持续监控和审计，及时发现并响应潜在的安全威胁。

（4）常见的 IaC 安全扫描

① YAML安全扫描

YAML 安全检测是针对 YAML 格式的配置文件进行的安全评估过程，旨在发现潜在的安全漏洞、配置错误或不当的安全设置。在云原生和 Kubernetes 等环境中，YAML 文件被广泛用于定义和配置资源，因此 YAML 文件的安全性至关重要。YAML 安全检测的内容包括：

语法检查：验证 YAML 文件的语法是否正确，避免因语法错误导致的配置问题。

配置审查：检查 YAML 文件中的配置项，确保其符合安全最佳实践。例如，检查是否启用了不必要的服务、是否配置了弱密码等。

敏感信息检测：搜索 YAML 文件中的敏感信息，如密钥、密码等，并评估其保护措施是否到位。

权限管理：检查 YAML 文件中关于权限的配置，确保资源的访问权限得到合理控制。

漏洞扫描：利用自动化工具对 YAML 文件进行漏洞扫描，发现潜在的安全漏洞。

② Dockerfile安全扫描

Dockerfile 是 Docker 镜像的构建脚本，包含了构建镜像所需的所有指令。通过安全检测，可以识别 Dockerfile 中潜在的安全问题，如使用过期的软件包、不安全的默认密码、不恰当的权限设置等，从而避免这些问题在镜像构建过程中被引入，影响镜像及容器的安全性。Dockerfile 安全检测的内容包括：

基础镜像检查：1. 验证基础镜像的安全性，确保使用的镜像版本没有已知的安全漏洞。2. 避免使用标签为“latest”的基础镜像，因为这可能导致在不同时间构建时引入不同版本的漏洞。

软件包安装与管理：1. 检查安装的软件包是否已过时，确保使用的是最新版本或已修补安全漏洞的版本。2. 尽量避免在 Dockerfile 中使用 apt-get update 和 apt-get install 等命令直接安装软件包，因为这可能会增加镜像的复杂性和攻击面。可以考虑使用多阶段构建来减少最终镜像中的不必要文件和依赖。

权限与用户管理：1. 确保在 Dockerfile 中设置了适当的用户权限，避免以 root 用户运行容器内的应用程序。2. 使用非特权用户运行应用程序可以减少潜在的安全风险。

敏感信息保护：1. 避免在 Dockerfile 中硬编码敏感信息，如密码、密钥等。2. 使用环境变量或 Docker Secrets 等机制来保护敏感信息。3. 健康检查与监控：4. 在 Dockerfile 中添加 HEALTHCHECK 指令，以监测容器内应用程序的健康状态。

其他安全检查：1. 避免在 Dockerfile 中暴露不必要的端口。2. 使用绝对路径设置 WORKDIR 指令。3. 清理不必要的文件和依赖，以减小镜像体积并减少潜在的安全风险。

3. 云安全态势管理

- **编排工具及组件安全**

（1）Kubernetes 基础概念

Kubernetes 是一个开源的编排管理平台，用于支持容器化工作负载和应用的自动化部署、扩展和管理，且拥有庞大且快速发展的生态系统。

（2）Kubernetes 安全原则

Kubernetes 主体最小权限

Kubernetes 服务账户、用户和组与 kube-apiserver 连接，用来管理 Kubernetes 对象。通过 RBAC（Role-Based Access Control）管理机制，不同的用户或服务账户可能有不同的权限来操作 Kubernetes 对象。RBAC 是一种基于授予用户或组的角色来管理资源访问的模式。RBAC 的核心要素包括三个：一是主体，即请求访问 Kubernetes API 的服务账户、用户或组；二是资源，即需要由主体访问的 Kubernetes 对象。三是行为，即主体对资源的不同类型的访问行为，例如创建、更新、列表、删除。Kubernetes RBAC 定义了主体和他们对 Kubernetes 生态系统中不同资源的访问类型。

通过 RBAC 模式和最小权限原则，为用户或系统组件提供执行其工作职责所需的最小权限等级或许可。通过限制不必要的权限，有助于减少潜在的安全风险。

Kubernetes工作负载最小权限

在通常情况下，会有一个与 Kubernetes 工作负载相关的服务账户。因此，Pod 内的进程可以使用服务账户令牌与 kube-apiserver 通信。DevOps 过程中应该谨慎地给服务账户授予必要的权限，以达到最小权限的目的。

除了访问 kube-apiserver 来操作 Kubernetes 对象，Pod 中的进程还可以访问工作节点和集群中其他 Pod/ 微服务的资源。

访问系统资源的最小权限

在容器或 Pod 内运行的微服务只不过是工作节点上的一个进程，被隔离在自己的命名空间内。一个 Pod 或容器可以根据配置访问工作节点上不同类型的资源，该访问权限是由安全上下文控制的。配置 Pod/ 容器的安全上下文会在开发人员的 任务列表中（在安全设计人员和审查人员的帮助下），而 Pod 安全策略是限制 Pod/ 容器在集群级别访问系统资源的另一种方式，会在 DevOps 的待办事项列表中。

访问网络资源的最小权限

在默认情况下，同一 Kubernetes 集群内的任何 Pod 都可以与其他 Pod 进行连接，如果 Kubernetes 集群外没有配置代理规则或防火墙规则，那 Pod 可能会访问互联网资源。Kubernetes 的开放性模糊了容器服务的安全边界，但是不能因此忽视网络资源的安全性，比如容器或 Pod 可以访问的其他微服务提供的 API。

可以通过限制网络访问范围、使用隔离的网络环境、实施访问控制、加密敏感通信以及定期审查和更新权限等措施，可以确保容器应用只能访问其执行任务所需的最少网络资源。同时，也需要关注实施过程中的挑战，并采取相应的解决方案来确保安全性和性能之间的平衡。

访问应用资源的最小权限

如果工作负载所访问的应用程序支持多个具有不同权限级别的用户，最好检查一下授予工作负载的用户权限是否有必要。例如，一个负责审计的用户不需要任何写入的权限，应用程序开发人员在设计应用程序时应牢记这一点，有助于确保工作负载在访问应用资源时拥有最少的权限。

（3）Kubernetes 安全控制技术

认证和授权在保护应用程序的安全性方面发挥着重要作用，认证和授权经常互换使用，但也存在很大不同。认证是为了验证用户的身份，一旦身份被验证，授权就被用来检查该用户是否有权限执行所需的行动。认证通过使用一些用户知道的信息来验证其身份，最简单的验证方式就是通过用户名和密码进行验证。一旦应用程序验证了用户的身份，就会检查该用户可以访问资源的访问控制列表。此时，将用户的访问控制列表与请求属性进行比较，就可以允许或拒绝某个请求。

1）Kubernetes 认证

Kubernetes 中的所有请求都来自外部用户、服务账户或 Kubernetes 组件。如果请求的来源是未知的，那它将被视为一个匿名的请求。根据组件的配置，匿名请求可被认证模块允许或拒绝。v1.6 以上版本默认允许匿名访问，以支持 RBAC 和 ABAC 授权模式的匿名和未认证用户的匿名访问。在 API Server 配置中，可以通过 --anonymous-auth=false 命令来明确禁用匿名访问。

Kubernetes 使用一种或多种认证策略，下面逐一进行介绍。

客户端证书	使用 X509 证书机构（CA）证书是 Kubernetes 中最常见的认证策略，通过 --client-ca-file=<path> 传到服务器，kube-apiserver 使用 <path>（CA 证书），通过 TLS 双向认证验证客户端证书是否由它签发。证书中的通用名称属性通常被用作请求的用户名，而组织属性则被用来识别用户组。
静态令牌	静态令牌是在开发和调试环境中一种常用的认证模式，但不适用于生产集群环境，因此不做详细说明。
基本认证	基本认证是静态令牌的一种变体，多年来一直作为 Web 服务的一种认证方法。基本认证不能在生产集群中使用，因此也不做详细说明。
Bootstrap令牌	Bootstrap 令牌是对静态令牌的一种改进。Bootstrap 令牌是 Kubernetes 中使用的默认认证方式，它们被动态地管理，作为密钥存储于 kube-system。
服务账户令牌	服务账户认证功能可以自动启用，用来验证不记名令牌。签名密钥用 -service-account-key-file 指定，如下所示，如果这个值未被指定，则使用 Kubernetes API Server 的私钥。 服务账户由 kube-apiserver 创建，与 Pod 相关联。如果没有指定服务账户，默认的服务账户将与 Pod 关联。

Webhook令牌	在 Webhook 模式下，Kubernetes 会调用集群外的 REST API 来确定用户的身份。认证的 Webhook 模式可以通过启用–authorization-webhook-config-file=<path> 来传递给 API Server。
认证代理	kube-apiserver 可以被配置为使用 X-Remote 请求来识别用户，可以通过向 API Server 添加以下参数来启用这种方法。
用户模拟	集群管理员和开发人员可以使用用户模拟来调试新用户的认证和授权策略。要使用用户模拟策略，一个用户必须被授予模拟策略的权限。API Server 使用以下方式来模拟一个用户。 Impersonate-User：充当用户名。。 Impersonate-Group：作为组名。可以多次使用来设置多个组，可选。 Impersonate-Extra-(extra name)：用于将额外字段与用户关联的动态 header，可选。 一旦 API Server 收到了模拟信息，API Server 就会验证该用户是否经过认证并有模拟的权限，如果有的话，kubectl 就可以使用 --as 和 --as-group 命令来模拟用户，一旦认证模块验证了用户的身份，就会解析该请求，以检查该用户是否被允许访问或修改该请求。

2）Kubernetes 授权

授权决定了请求是被允许还是被拒绝。一旦请求的来源被识别，主动授权模 块就会根据用户的授权策略来评估请求的属性。对照用户的授权策略、评估请求的属性，以允许或拒绝请求。每个请求依次通过授权模块，如果任何模块提供了允许或拒绝的决定，整个过程就会被自动接受或拒绝。

- **网络配置及安全策略**

集群网络是 Kubernetes 的核心概念，其中必须考虑容器、Pod、服务和外部服务之间的通信。默认情况下，几乎没有网络策略通过隔离资源来防止集群失陷时的横向移动或权限提升。网络配置及安全策略是限制网络攻击者在集群内移动和提升权限的有效方法。

网络策略控制Pod、命名空间和外部IP地址之间的流量。默认情况下，Pod或命名空间没有网络策略，导致 Pod 网络内的入口和出口流量不受限制。通过适用于 Pod 或 Pod 命名空间的网络策略，可以对 Pod 进行隔离。一旦根据网络策略选中了一个 Pod，它就会拒绝适用对象进行的任何不被允许的连接。

要创建网络策略，需要支持 NetworkPolicy API 的网络插件，使用 podSelector 或 namespaceSelector 选项来选择 Pod。Pod 的默认策略是拒绝所有入口和出口流 量，并确保将任何未选择的 Pod 隔离开来。对于允许的连接，可采取其他策略放松限制。对于外部 IP 地址，可以使用 ipBlock 在入口和出口策略中进行设置，但不同的 CNI 插件、云提供商或服务实现可能会影响 NetworkPolicy 的处理顺序和集群内地址的重写。

网络策略还可以与防火墙和其他外部工具结合使用，以创建网络分段。将网 络分割成独立的子网络或安全区有助于将面向公众的应用与敏感的内部资源隔离开。在 Kubernetes 中，网络分段可用于分离应用程序或资源类型，以限制攻击面。

- **云安全配置管理**

云安全配置管理是 CNAPP 的核心功能之一，它涵盖了云资源的配置审计、合规性评估、风险检测和自动化修复等多个方面。

1 **配置审计**	能够持续扫描云环境中的所有资源配置，包括虚拟机、容器、数据库、网络设置等。通过与云平台的 API 集成，实时获取最新的配置信息，并进行全面的审计。审计结果将用于识别潜在的配置错误、不合规项和安全隐患。
2 **合规性评估**	支持多种云安全合规性标准（如 PCI DSS、HIPAA、GDPR 等），并根据这些标准对云资源配置进行合规性评估。提供预定义的合规性模板和报告，帮助用户快速了解云环境的合规状况。对于不合规的资源配置，将提供修复建议或自动化修复措施。
3 **风险检测**	通过高级分析技术，结合上下文和关联信息，对云环境中的潜在风险进行检测。能够识别配置错误、漏洞、未授权访问等安全威胁，并评估其可能的影响范围和严重程度。检测结果将用于指导后续的修复和加固工作。

4 自动化修复	提供自动化的修复功能，能够针对检测到的安全威胁和配置错误进行快速修复。自动化修复可以减少人工干预，提高修复效率和准确性。对于无法自动修复的问题，CNAPP 将提供详细的修复指导和建议。
5 策略管理	允许用户定义和管理云安全策略，包括访问控制策略、安全组策略、防火墙规则等。用户可以根据业务需求和安全要求，灵活地调整和优化这些策略。自动应用这些策略到云环境中，确保所有资源配置都符合安全要求。
6 变更管理	支持对云资源配置的变更进行管理和审计。能够跟踪和记录配置变更的历史记录，包括变更时间、变更内容、变更人员等信息。通过变更管理，用户可以了解云环境的配置变化情况，并及时发现潜在的安全风险。

（三）运行时安全

1. 云工作负载安全

云工作负载安全是 CNAPP 里重要组成部分，云上的工作负载是一套支撑 IT 业务系统运行的相关功能或一些原子能力，诸如服务器、VM、容器、等。通常情况下企业在云上使用最多的工作负载环境就是 VM 和容器，当然近些年也兴起了一些 serverless 类型的工作负载。云工作负载安全主要以主机、容器、和 serverless 防护进行展开，从云的服务类型恰好对应 IaaS、PaaS、SaaS ，从用户的角度考虑用户的安全职责也略有不同，非自建部分不负责主要安全指责，而是由云厂商承担。

（1）主机安全（IaaS 模式）

在 IaaS 云的模式下，用户需要负责主机和虚拟的安全，主机安全是一套关键的保护措施，适用于物理服务器、虚拟机、云主机以及各种终端设备，确保它们免受恶意软件、未授权访问和其他安全威胁的侵害。它在保护组织的关键资产和数据、确保业务连续性和数据完整性方面发挥着至关重要的作用。通过实施主机安全措施，组织能够有效防止数据泄露、抵御恶意软件攻击、增强合规性、降低业务风险、提高系统稳定性，并快速响应安全威胁，为维护网络安全提供了坚实的基础。主机工作负载保护主要包含基线检查、风险发现、入侵检测、资产管理等能力。

1 基线检查	应以等保标准，行业标准等建立和维护操作系统、应用程序和配置的标准安全基线，确保所有系统遵循一致的安全标准，可以快速识别配置偏差和潜在风险。
2 风险发现	应使用自动化工具定期扫描系统以发现高危漏洞、安全补丁、弱密码、账号风险，应用风险等问题，主动识别风险帮助及时修复漏洞，减少被攻击的机会。
3 入侵检测	应对主机的暴力破解、异常登录、反弹 shell、web 后门、远程命令执行、本地提权等恶意攻击行为进行检测，发现实时攻击行为并采取控制措施。
4 资产管理	应维护一个准确的主机资产清单，包括基本资产、系统层资产、应用层资产、业务层资产、各类资产进行详细统计，清晰的资产清单有助于更有效地管理安全补丁和合规性，减少未管理资产带来的风险，同时对供应链风险和 0 day 风险排查起到帮助。 基本资产包括硬件、CPU、内存、磁盘、网卡 IP、操作系统等信息。主机系统层资产包括包括进程、端口、账号、启动项、计划任务、环境变量、内核模块等信息。主机应用层资产包括安装包、jar 包、应用、数据库、中间件等信息。对主机业务层资产包括 web 站点、web 框架、web 应用、web 服务等信息。

（2）容器安全 (PaaS 模式)

容器安全是针对容器化技术特别设计的保护措施，适用于使用 Docker、Kubernetes 等容器技术的各种场景，包括持续集成和持续部署（CI/CD）流程、微服务架构、云原生应用开发和 DevOps 实践。容器安全的目标是确保容器镜像的安全性、容器运行时的安全性以及资产管理和风险发现的能力。

1

资产管理

应维护一个准确的容器资产清单，包括基本镜像资产、集群资产、应用层资产、业务层资产等各类资产进行详细统计，清晰的资产清单有助于更有效地管理安全补丁和合规性，减少未管理资产带来的风险，同时对供应链风险和 0 day 风险排查起到助。

镜像资产管理应能够清点仓库镜像、和节点镜像，具体包括 Repository、Tag 、创建时间镜像大小、关联镜像、以及关联到镜像的风险信息。集群资产管理应能够支持自动获取集群中的资源对象，具体包括：集群、Ingress、Service、Controller、Pod 、endpoint、Secret、PV、PVC 等。应用层资产应能够支持自动识别容器的应用资产，如 Oracle、tomcat、nginx、SSH、rsyslog、redis 等，应用信息应至少包括应用名、应用类型、应用版本、运行用户、二进制路径等。业务层资产应能够支持自动获取容器中的 web 站点，nginx，tomcat 等，应支持国产 web 资产如 tongweb 等，资产信息应包括域名、服务类型、启动用户、主目录、所有者权限、虚拟目录信息等。

2

基线扫描

应以等保标准，行业标准等建立和维护操作系统、应用程序和配置的标准安全基线，确保所有系统遵循一致的安全标准，可以快速识别配置偏差和潜在风险。

3

入侵检测

应支持全面的容器运行时入侵检测，如容器反弹 shell、webshell、暴力破解、病毒检测、可疑进程、webrce、黑客工具、隧道攻击、网络行为、容器逃逸、K8S API 可疑调用等行为进行实时检测。应支持安全事件响应能力，包括重启容器，暂停容器，删除容器等操作。

4

镜像安全

镜像作为容器环境中最重要的不可变基础资产，应支持对运行的镜像、节点中镜像、CI 过程中镜像、仓库中镜像进行统一扫描和风险识别。应支持镜像安全补丁检查、镜像语言框架漏洞检查、镜像恶意文件检查、镜像敏感信息检查、自定义检查、可信镜像检查、开源 license 检查、不合规软件应用检查、基础镜像识别、超大镜像检查、镜像风险分析、镜像层文件视角分析、恶意镜像启动阻断能力。

（3）serverless 安全

Serverless 提供了一种“无服务器”的计算模式，允许开发人员构建和运行应用程序和服务，而无需管理基础设施或服务器端。这种模式简化了 CI/CD、服务器配置维护更新、IT 资源容量的规划和伸缩等工作，使研发人员专注于业务逻辑的编写，而运维人员则转向确保服务水平协议（SLA）的实现。在 Serverless 架构下，用户需要对代码、身份权限、应用安全负责，而不需要考虑基础设施的安全。

① 身份权限管理

确保只有授权用户和系统能够访问函数或其他资源。通过实施多因素认证（MFA）、临时凭证服务和基于角色的访问控制（RBAC）或基于属性的访问控制（ABAC），强化身份验证和授权机制。同时，定期审查权限设置，确保它们符合最小权限原则，并通过安全审计功能监控权限使用情况，以维护配置安全。

② 代码安全

代码层面的安全至关重要，首先，对开发人员进行安全编码培训，确保他们了解如何编写避免安全漏洞的代码。其次，通过代码审查和自动化的静态代码分析工具（SAST）来识别代码中的安全问题，同时使用动态代码分析工具（DAST）在运行时检测漏洞。管理好项目依赖项，定期扫描并更新有安全漏洞的库。在设计阶段采用安全设计原则，如最小权限原则，并确保配置文件和环境变量不包含硬编码的敏感信息。将安全测试，如渗透测试和模糊测试，纳入软件开发生命周期。自动化安全流程并集成到 CI/CD 中，持续评估代码质量和安全性，包括对供应链中第三方服务和工具的安全性评估，限制对源代码的访问，并实施代码签名以保证软件包的完整性。

③ 应用防护

在无服务器（Serverless）架构中，应用防护是关键环节，尤其需要针对注入攻击和 DDoS 攻击等常见威胁采取有效措施。

防止注入攻击，通过实施严格的输入验证和使用预编译的 SQL 语句，以及对所有用户输入进行适当的编码和转义，可以显著降低 SQL 注入和 XSS 攻击的风险。

2. Web应用和API保护

WAAP 是 Web 应用程序和 API 保护的缩写，这是一种旨在保护 Web 应用程序和 API 免受各种日益复杂的网络攻击的安全技术。

一个优秀的基于云的 WAAP 建设方案，可在减少工作量和开销的同时，保护应用程序和 API 免受各种网络和应用程序层的威胁。从爬虫程序监测和抵御，到 DDoS 防护和自我调整建议，实现自动化和简单化，一个多维自适应安全引擎可将威胁情报与每个 Web 和 API 请求的数据 / 元数据相关联，从而提供基于威胁的检测。

（1）Web 应用保护

Owasp Web top 10 风险防御

应具备 Owasp Web top 10 防御能力，能够自动防御包括 SQL 注入和 XSS 在内的常见安全威胁，这不仅减少了对传统规则引擎的依赖，也降低了误报率，提高了安全防护的准确性。

内存马防护

应具备内存马防护能力专门针对攻击者植入的恶意脚本，如 WebShell，能够及时检测并阻止其执行，从而保护服务器不受持续威胁。

热补丁

应具备补丁能力允许在应用程序运行时快速应用安全补丁，无需重启服务，这大大缩短了漏洞修复的时间窗口，减少了系统暴露的风险。

防DDos和bot攻击

实现网络层 DDoS 攻击防护，并在几秒内抵御应用程序层攻击，在应用的登录页面和表单提交处添加 CAPTCHA，以区分人类用户和自动化 bot。

（2）API 防护

API 安全是针对应用程序编程接口（API）的保护措施，适用于所有依赖 API 进行数据交换和通信的场景。API 安全的核心价值在于确保数据的完整性、保密性和可用性，同时防止诸如数据泄露、服务滥用、未授权访问和各种网络攻击等安全风险。通过实施 API 安全措施，组织能够提高服务的可靠性和用户的信任度，满足合规性要求，并降低因安全漏洞导致的财务损失和声誉损害。此外，API 安全还有助于实现细粒度的访问控制、监控和分析 API 使用情况，从而优化性能和用户体验。

自动发现API 服务

自动化工具识别和映射网络应用程序及 API，确保所有面向网络的服务都被识别并纳入保护范围，减少人工管理的需要，确保 API 的安全防护全面覆盖，避免遗漏。

Owasp API top 10 风险保护

实现对 API 的 Owasp API top 10 的保护，实现对，API 对象级别授权失效，API 认证失效，API 对象属性级别授权失效，API 未受限制的资源消耗，API 的不安全使用等风险进行发现以及防护。

应用调用链路跟踪

宜具备应用调用链路跟踪能力通过监控应用的内部调用和数据流，为安全团队提供了深入分析和快速定位问题的能力，提高了事件响应的速度和准确性。

3. 网络微隔离

微隔离是一种高效的网络安全策略，适用于数据中心虚拟化、云服务、容器化部署、微服务架构和大型网络分段等多种场景。它通过创建网络中的细微隔离边界，显著减少攻击面，有效防止攻击者在网络内部的横向移动，同时提高网络流量的可见性，使安全团队能够快速监测、识别并响应潜在威胁。

（1）流量识别

网络拓扑动态展示	能够直观实时展示云环境的网络拓扑结构，帮助用户掌握云内部的细微变化、流量信息和威胁信息，应支持通过雷达图展示主机与主机之间，容器与容器之间，controller 与 controller 之间的访问关系，应至少包括访问者、被访问者、访问协议、访问端口、访问次数等，增强的可见性使得网络安全团队能够实时监控和理解网络内部的通信模式，及时发现不正常的访问行为或潜在的攻击，从而快速响应并采取措施。

(2) 流量隔离

1 **细粒度的访问控制**	应支持主机环境下的主机隔离策略同时支持 kubernetes 集群环境下的微隔离策略设置，策略的设置应支持 pod、controller、label、image、IP 等维度。细粒度的访问控制限制了潜在攻击者在网络内部的横向移动能力，减少了安全事件的影响范围。同时，它也支持更精确的安全管理策略，提高了对不同网络环境的适应性和灵活性。
2 **自动化的策略管理**	宜支持实现基于主机，容器网络拓扑关系，自动生成网络策略的能力，同时应确保在多云或混合云环境中，安全策略的一致性，无论工作负载部署在何处。
3 **快速隔离能力**	应支持实现对失陷的工作负载的快速隔离能力，禁止攻击的外溢和扩散。

04 企业CNAPP建设路径

（一）框架内涵

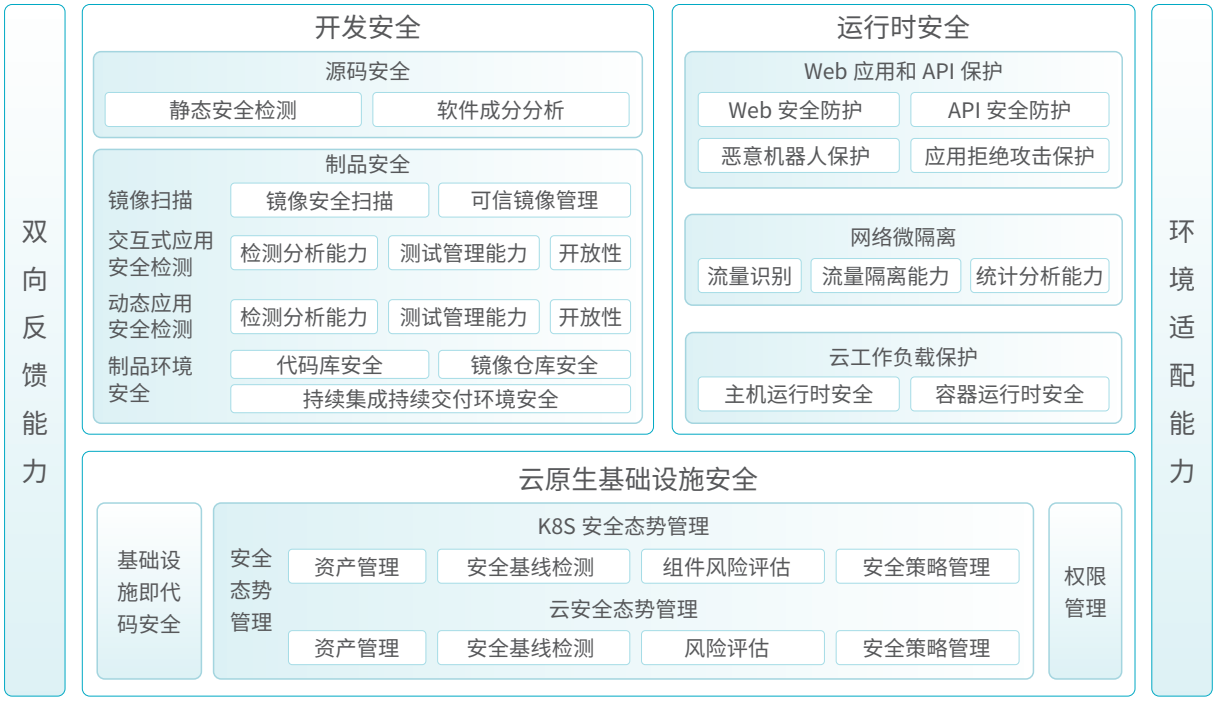


图1 CNAPP建设框架

CNAPP 建设应覆盖三大安全方向并具备两大能力。其中三大方向包含基础设施安全、制品安全以及运行时安全。基础设施安全的主要内容包括基础设施及代码（IaC）安全; 以资产管理、安全极限检测、组件风险评估以及安全策略管理为核心的云（原生）安全态势管理；权限管理三大部分。制品安全需要包含的功能有涉及静态安全检测和软件成分分析的代码安全；包含镜像安全扫描；具备检测分析能力、测试管理能力以及足够的开放性的交互式应用安全检测；涉及代码库安全、镜像仓库安全以及持续集成持续交付环境安全的制品环境安全；具备检测分析能力、测试管理能力以及足够的开放性的动态应用安全检测。运行时安全应包含的内容有 Web 应用和 API 保护，涉及 Web 安全防护、API 安全防护、恶意机器人保护以及应用拒绝攻击保护；具备流量识别、流量隔离及统计分析能力的网络微隔离功能；保障主机及运行时安全的云工作负载保护。

除上述三大安全方向外 CNAPP 还应具备两大能力，即双向反馈能力与环境适配能力。前者能够显著提高安全人员再利用 CNAPP 处理云原生应用安全问题是的使用体验，后者则能保证 CNAPP 与本地开发环境的无缝集成。

（二）建设原则

1. 无缝集成

无缝集成指的是在 CNAPP 建设过程中，应力求做到无缝集成，即能够将 CNAPP 涉及的多个方案、工具等进行有效集成，并能够与本地开发环境紧密配合。具体而言，要做到无缝集成可以从集成的广度及深度两个指标入手进行衡量。在广度方面，一个有效集成的 CNAPP 应具有覆盖制品安全、基础设施安全、运行时安全三大方向的能力，通过将多种安全方案与工具做到在一个安全平台内能够对整个云原生安全应用的各个组成部分进行高可视性的监控并在发生安全事件时以整体为对象帮助安全人员及时进行补救。在深度层面，无缝集成的 CNAPP 在所覆盖的每一个安全方面上都应能有效触及到全部细分功能，并自动调度对应的方案与工具进行处理。例如在运行时安全中应能够同时触及 Web 应用和 API 保护、运工作负载保护及网络微隔离，并针对每个细分问题进行针对型的方案调度。例如在运工作负载保护功能中针对主机运行时安全及容器运行时安全分别进行针对性处理。

2. 以应用为中心

CNAPP 的“以应用为中心”原则强调对云原生应用全生命周期的安全保护。它专注于应用及其运行环境，包括代码、配置和数据的安全。CNAPP 通过自动化集成到 CI/CD 流程中，实现安全扫描和合规性检查，同时利用行为分析技术监控应用行为，快速识别并响应安全威胁。此外，CNAPP 提供端到端的可见性和监控，确保安全团队能够全面了解应用状态。它还支持跨云环境的一致性安全策略，帮助企业满足合规性要求，同时优化用户和开发者的体验。这一原则确保了云原生应用的安全性，同时提高了开发和运维的效率。

3. 安全左移

在 CNAPP 的建设过程中应遵循安全左移原则。与常规应用不同，云原生应用因其架构更加灵活与动态，因此所面临的安全挑战也更为复杂，出现安全问题可能造成的损失也会更严峻。因此在云原生应用领域安全不应再作为一种事后的补充，而应该作为一个内置的过程嵌入到整个开发运营过程当中。因此在建设 CNAPP 时不但要保证安全团队使用的便利性，还要保证其能够同时被开发团队接受，并快速与开发环境相融合。具体而言建设得当的 CNAPP 应该能够在开发过程中就为开发人员提供漏洞检测、漏洞修复等功能，帮助开发人员将安全性在应用开发的早期就融入到基础设施当中。

4. 自动化

在 CNAPP 的建设过程中，自动化原则分两方面。其一是为了减少开发人员和安全人员的额外精力投入，还应尽量保证整个过程的自动化水平。例如优质的 CNAPP 产品在安全工具、安全策略的集成过程中应尽量减少人工介入的需求，做到自动检测，自动整合，自动优化。另一方面是在 CNAPP 的后续使用过程中，各类功能也都应该尽可能实现自动化以减少安全人员的工作量，特别是对于重复性的低难度工作，可以根据先前的处理经验及安全人员设置的策略进行自动处理。更进一步还可以将大模型与 CNAPP 重合，降低 CNAPP 使用门槛，提高使用体验并能够对更复杂的安全问题进行自动化处理。

5. 简单化

在 CNAPP 的建设过程中，为了最大化投入产出比，实现降本增效，应遵循简单化原则，包括简化复杂的安全基础设施和安全概念、为跨团队协同提供更简洁的方式以及降低开发、安全团队的复杂性。例如利用 CNAPP 的高度集成、高可视性等优点，将原先需要安全人员投入精力去判断相关性的安全事件、安全问题进行精简呈现，减少所涉及到的安全工具数量；利用其覆盖云原生应用全生命周期的优势连接开发团队和安全团队使之在统一平台上进行合作，简化跨团队协作的成本；通过自动化来减少不必要的工作量，精简开发、安全团队的人员复杂性，确保每一个人都发挥最大的价值。

（三）建设方法

CNAPP应具备的基础能力	
开发安全	源码安全： · 支持代码静态安全检测 · 支持常见的软件成分分析 · 易于集成到CI/CD流程中
	制品安全： · 支持镜像安全扫描、可信镜像管理 · 对制品环境（镜像仓库和代码库）实施访问控制、用户行为审计 · 支持对镜像仓库的漏洞扫描能力 · 易于集成到CI/CD中，以帮助自动化漏洞扫描和报告
云工作负载保护	风险管理： · 运行时对所有容器镜像、镜像仓库、容器主机和虚拟机实例进行漏洞扫描 · 灵活的安全报告输出 · 整合外部漏洞信息源，具有统一的漏洞清单，包含漏洞优先级
	容器/K8S态势管理： · 统一资产清单，包括Kubernetes和云资源 · 为所有资源创建简单灵活的策略设置和漏洞修复，具备运行时修复和IaC修复能力
	运行时安全与事件响应： · 部署简单性。完整的CNAPP将结合Agent和Agentless以创建统一解决方案。 · 可在策略中执行镜像配置规则和简单实施基础镜像补救措施 · 对工作负载上的恶意活动进行实时检测，具有持续更新的规则和集成威胁信息源能力 · 强大的调查和取证能力，如证据获取、自动化调查/取证 · 基于AI增强和提升检测能力 · 同时涵盖云工作负载和Kubernetes编排的服务和工作负载，提供统一的安全和风险看板。

云安全态势管理	态势管理（云/IaaS）： · 支持策略即代码的自定义策略（最好支持Open Policy Agent [OPA]等标准），以及与行业框架和合规监管要求一致的现成策略库，以检测和配置任何云中的设置。 · 能够实时检测跨云的工作负载和编排服务中出现的配置漂移。 · 攻击路径分析，能支持在多个事件之间进行相关性分析（例如，横向移动）。 · 在云环境中对所有策略和配置状态进行详细和灵活的报告，最好具备行业框架、合规报告以及自定义选项。
	云漏洞管理： · 通过将运行时上下文与静态检查（配置错误、已知漏洞）相结合，对安全风险进行排序。
	权限/授权管理： · 身份和访问管理分析能力，能够评估权限使用情况，并能够按照最严格的身份和访问策略落地执行。 · 识别和分析风险配置的用户属性和设置，比如缺乏多因素认证 (MFA)、公开或过度宽松的云访问密钥、缺乏访问密钥轮换等情况。
云检测与响应	· 实时检测恶意活动和行为，不断更新规则并集成威胁源 · 灵活的规则语言用于自定义规则 · 支持跨越云、容器和Kubernetes多个不同平台 · 强大的调查和取证能力，如证据捕获、自动化调查/取证操作 · 事件丰富化和转发，能够与第三方安全工具和平台对接（如SIEM） · 检测和分析Kubernetes网络事件 · 具备主机、托管容器服务工作负载的检测和响应能力 · 随着ChatGPT等生成式AI工具的兴起，许多安全解决方案开始利用这项技术来帮助用户。例如允许用户使用自然语言提示来获取丰富上下文相关告警。

表2 CNAPP基础能力表

CNAPP 的建设路径依赖于企业的建设阶段，企业处在不同的云原生安全建设阶段，其需要构建的 CNAPP 能力也不尽相同。

1. 建设初期：了解CNAPP基础能力，树立一体化安全意识

在云原生应用保护平台（CNAPP）的建设初期，企业往往面临着一系列挑战。尽管企业可能已经开始了云原生安全的相关建设，但这些措施往往是孤立的，缺乏一个统一的防护框架。这是因为在云原生安全技术的早期发展阶段，还没有形成成熟的体系化防护框架，同时，由于资源和技术基础的限制，安全技术供应商往往只能专注于云原生安全的某个细分领域，开发出早期可落地的产品。

随着云原生技术被应用于更多核心业务，这种分散的安全防护体系显然无法满足日益增长的安全需求。安全防护体系的一个显著特点是木桶效应，即整体防护效果往往受限于最弱的环节。此外，云环境下的安全孤岛和整体复杂性的增加，缺乏端到端的可观测性，为网络攻击提供了可利用的盲点，同时也增加了企业协同运维管理的难度。

因此，企业需要从云原生整体的技术栈出发，从网络安全完整的攻击链出发，构建一个覆盖开发到运行时流程的一体化云原生安全防护体系。在这个阶段，企业用户需要对自身的云原生安全建设进行深入剖析，明确关键安全需求，并选择合适的方式开展云原生一体化建设。

在 CNAPP 建设的初期，企业尚未完全明确自身的关键需求，直接采购或构建现有的 CNAPP 产品可能并不现实。因此，企业用户应该采取以下策略：

1. 明确需求和痛点：企业应根据现有经验，明确自身在云原生安全过程中遇到的最大痛点问题，简化当前的安全工具，尝试集成成熟且有效的安全能力。

2. 减少供应商数量：考虑减少当前的供应商数量，以降低中后期建设 CNAPP 时的集成难度和更换供应商的额外成本。

3. 开展云原生安全意识培训：企业应加强员工的云原生安全意识培训，建立一体化的工作流程，强调跨部门协作的重要性，特别是加强安全人员、开发人员、运维人员的沟通，为建设 CNAPP 提供良好的软性环境。

4. 构建端到端的可观测性：企业应投资于构建端到端的可观测性，以便更好地监控和分析云原生环境中的安全事件，及时发现并响应潜在的安全威胁。

5. 采用模块化和可扩展的架构：在 CNAPP 的建设过程中，企业应采用模块化和可扩展的架构，以便在未来能够轻松地集成新的安全技术和功能。

通过这些策略，企业可以在 CNAPP 建设初期打下坚实的基础，为后续的云原生安全防护体系建设提供支持，确保企业能够在云原生环境中实现高效、可靠的安全防护。

2. 建设中期：完善CNAPP全面能力，构建完整CNAPP方案

在建设中期，企业已经明确自身需求，充分了解 CNAPP 基本建设原则和路径，并且内部安全工具已去繁化简，且已构建相关人员一体化安全防护意识。在该阶段，企业应根据 CNAPP 关键能力列表和框架，集成现有工具或平台，形成一体化的云原生应用防护。如表 2 所示，CNAPP 所需具备的基础能力包括五个方面，分别是开发安全、云工作负载保护（CWPP）、云安全态势管理（CSPM），以及云检测与响应，。

在开发安全方面，企业应对提供源码安全检测和软件成分分析，易于融入 CI/CD 流程，确保代码安全。同时，我们支持镜像安全扫描、可信镜像管理，以及对制品环境的访问控制和行为审计，实现自动化漏洞扫描和报告，保障制品安全。

在云工作负载保护（CWPP）方面，企业应当实现风险管理、云原生安全态势管理（KSPM），以及运行时安全与事件响应。其中，风险管理的基本要求是企业建设的 CNAPP 方案可以实现对所有镜像及镜像仓库、容器主机和虚拟机实例进行漏洞扫描，同时具备灵活的安全报告输出形式，支持 word、pdf、excel 等多种形式的报告。另外类似漏洞扫描的风险管理方式应该易于集成到 CI/CD 流程中，帮助生成自动化的漏洞扫描和报告。其次，风险管理应该具备完善、全面的漏洞情报源，具备统一的漏洞清单。云原生安全态势管理的基本要求首先是需要提供云原生资产的统一资产清单，这里的资产包括 kubernetes 集群资产和云上资源，同时应当为所有资源创建简单灵活的策略设置和漏洞修复，具备运行时修复和 IaC 修复能力。运行时安全与事件响应要求企业建设的 CNAPP 方案可以在策略中执行镜像配置规则和简单实施基础镜像补救措施，同时对工作负载上的恶意活动进行实时检测，具有持续更新的规则和集成威胁信息源能力。并且在安全事件发生后具备强大的调查和强大的调查和取证能力，如证据获取、自动化调查 / 取证。此外，企业应该尝试使用 AI 技术增强和提升攻击检测的能力。建设中期阶段的 CNAPP 应该能够结合 Agent 和 Agentless 通过创建统一解决方案的方式简单快速地部署到本地开发流程当中。此外，应为威胁检测、漏洞管理等全方位功能提供云工作负载和 Kubernetes 编排的服务的统一的操作界面，最大程度上增加安全问题的可视性，确保安全人员在处理问题时不必在多个控制面板之间来回切换以提高使用体验。

在云安全态势管理（CSPM）方面，企业应当能通过 CNAPP 实现态势管理（云 /IaaS）、云漏洞管理及权限 / 授权管理功能。具体而言在这一阶段态势管理要求企业所构建的 CNAPP 能够检测和配置任何云中的设置，这需要支持策略即代码的自定义策略，例如 Open Policy Agent [OPA]，以及与行业框架和合规监管要求一致的现成策略库。此外 CNAPP 还应保证能够实时检测跨云的工作负载和编排服务中出现的配置漂移以及在受到攻击后循序进行综合多个事件的高级攻击路径分析，并基于其结果生成一份对所有策略和配置状态进行详细和灵活的报告，应支持 word、pdf、excel 等常见形式并能够允许安全团队根据需要对报告进行自定义调整。在云漏洞管理方面，本阶段内 CNAPP 应具备通过将运行时上下文与静态检查相结合，从而为安全漏洞进行排序，便于安全人员快速识别关键问题。最后在权限 / 授权管理功能上 CNAPP 应具备识别和分析风险配置的用户属性和设置的能力，并根据权限使用情况，并能够按照最严格的身份和访问策略落地执行。

在云检测与响应方面，建设中期的 CNAPP 应具备绝大部分核心能力。包括能够跨云、容器和 Kubernetes 等多个不同的平台进行实时的恶意活动和恶意行为检测，并不断更新规则集成威胁源，让安全团队具备更强的及时反应能力及总结备案能力；具备强大的调查和取证能力，能够完成证据捕获、自动化调查及取证操作，并将获取到的证据及时存档并与第三方安全工具和平台快速对接，构建起全方位保护机制；能够检测和分析 Kubernetes 网络事件；具备主机、托管容器服务工作负载的检测和响应能力；以及支持灵活的规则语言允许安全人员根据不同云原生应用的特点进行自定义规则。

3. 建设后期：优化CNAPP进阶能力，追求技术先进性

CNAPP应具备的进阶能力
· 支持serverless应用架构 · 利用生成式人工智能在云库存资产上进行自然语言搜索和查询 · 提供具备机器学习能力的多层保护，作为基于规则检测的补充 · 能够在多个事件/来源之间进行跨领域数据关联 · 支持所有主要云、Linux和Kubernetes供应商 · 从内部实时查看关键虚拟机和容器的工作负载，包括工作负载检测/响应 · 开发过程中进行API发现和扫描以确保正确配置，并在运行时进行监控 · 在常规工作负载监控（例如，查看事件日志、网络日志和DNS查找）的基础上扩展云检测和响应（CDR）功能
CNAPP可具备的额外能力
· 应用程序运行时自我保护（RASP） · 无服务器功能检测和监控 · 应用层可观察性/监控 · 支持基于VMware的基础架构（基于内部部署和公共云） · 支持其他云和容器环境，如Red Hat OpenShift和SUSE的Rancher · 运行时的web应用程序和API保护（WAAP） · 针对未知漏洞的传统动态扫描和静态分析和API扫描 · SCA之外的开发管道/软件供应链安全性

表3 CNAPP进阶能力表

在云原生应用保护平台（CNAPP）的建设进入后期阶段时，企业应该采取更为细致和深入的策略来最大化 CNAPP 的价值。这主要可以通过两个方面来实现：一是对现有 CNAPP 进行优化迭代，二是根据企业自身的特定需求部署 CNAPP 的进阶和可选功能。这样的策略将有助于企业全方位地构建起云原生安全防护体系。

首先，企业应该基于中期时安全团队和开发团队的实践反馈，对 CNAPP 的各项功能进行评估。这包括判断这些功能是否真正满足产品的安全需求，以及是否存在资源配置过剩的情况。通过与团队成员的深入讨论，企业可以有序地调整功能配置，优化资源使用效率，并通过不断的迭代来实现一个更加贴合企业实际需求的定制化 CNAPP。这种持续的优化过程是至关重要的，因为它确保了 CNAPP 能够随着企业需求的变化而进化。

其次，企业应该保持对新技术的敏感性，并在发现有价值的技术时及时更新 CNAPP 平台。这种更新不仅是为了保持技术的先进性，也是为了确保 CNAPP 能够适应不断变化的云原生环境和安全威胁。通过这种方式，企业能够确保其 CNAPP 平台始终处于行业前沿，从而更好地保护其云原生应用。

除了基本的功能优化和迭代，企业还可以考虑为 CNAPP 集成更多的可选功能，以满足特定的业务需求。这些可选功能可能包括但不限于：

1. 支持 Serverless 应用架构：随着 Serverless 架构的兴起，企业可能需要 CNAPP 能够支持这种无服务器的计算模型，以覆盖更广泛的云原生产品。

2. 集成生成式人工智能技术：通过将生成式 AI 技术与 CNAPP 结合，企业可以实现更高级的安全功能，如自然语言处理（NLP）支持的检索、自动化的漏洞修复，甚至是利用垂直领域大模型来自动识别和处理更复杂、更高级的安全漏洞。

3. 自动化安全策略管理：企业可以利用 CNAPP 来自动化安全策略的部署和管理，减少人为错误，提高响应速度。

4. 高级威胁检测和响应：通过集成更高级的威胁检测和响应机制，CNAPP 可以帮助企业更快地识别和应对潜在的安全威胁。

5. 合规性自动化：对于需要遵守特定行业标准和法规的企业，CNAPP 可以帮助自动化合规性检查和报告，确保企业的操作符合相关要求。

通过这些进阶和可选功能的集成，企业不仅能够提升 CNAPP 的功能性，还能够提高其在云原生安全防护中的整体效益。这样的全方位策略将有助于企业在不断变化的云原生环境中保持领先地位，确保其应用和数据的安全。

05 行业实践

（一）运营商

1. 需求分析

某大型运营商研究院是该集团公司产品开发和业务研究的主要科研机构，是该运营商研发和创新体系的重要组成部分。研究院拥有雄厚的科研基础设施，强大的科研开发团队和高水平的研发成果。

近年来，随着大数据、云计算、容器化、微服务、平台战略等新技术和新概念的层出不穷和快速发展，在业务支撑、架构能力、平台扩展性等方面对旧有的烟囱式建设的业务支撑系统提出了巨大的挑战。该研究院在集团提出以容器为基础平台战略的同时，针对容器技术的安全防护展开研究。

研究发现，一是容器技术不同于传统虚拟化技术，其自身隔离性较弱，它们共享宿主机的操作系统，容器之间可以相互访问和影响；二是在应用容器技术的过程中，存在多个安全薄弱点，包括镜像、镜像仓库、容器集群等; 三是无法有效识别容器内部行为，包括正常的行为，例如网络连接，以及恶意的行为，例如容器的提权、逃逸等；基于此，亟需建立整体的容器安全体系，加强纵深防御。

2. 整体方案设计

随技术路线从虚拟化转向容器化，基于云原生容器技术重构了基础设施，建立了开发平台、容器云等基于云原生为底座的基础平台。云原生技术使得大型复杂软件的应用拆分，各应用之间松耦合，从而降低了系统复杂度，还做到了独立发布部署、独立扩展和跨语言编程等，这些变化也驱动着安全工作模式及重心的转变。

图2 某运营商云原生安全架构图

适应云原生架构的整体方案设计

所以在安全方案设计上，也要贴合云原生技术架构，以满足业务容器化后，传统安全能力不再适用的问题，例如：业务容器会动态漂移、容器内部行为及日志等默认不会记录、基于 IP 的访问控制方式不再适用等。基于此整体的方案需满足以下几个方面：一是满足容器技术高密度、高动态、快速迭代、敏捷等多种特性；二是能够将安全防护覆盖云原生应用的整个生命周期，从需求分析、软件开发、软件测试、软件发布一直延伸到软件运维；三是相关能力能够与现有安全体系兼容联动，统一管理及运营。

3. 实施技术路线

（1）研发侧安全提升

首先，镜像是容器运行的基础，包含业务运行需要的所有环境、文件和配置等信息，但基础镜像当前基于公网的开源仓库，且代码也引入许多开源组件，无法保障业务镜像的安全性。且研发更侧重于业务的功能实现，安全重视程度不足，基于此建立了镜像安全防护能力，保障研发侧安全。

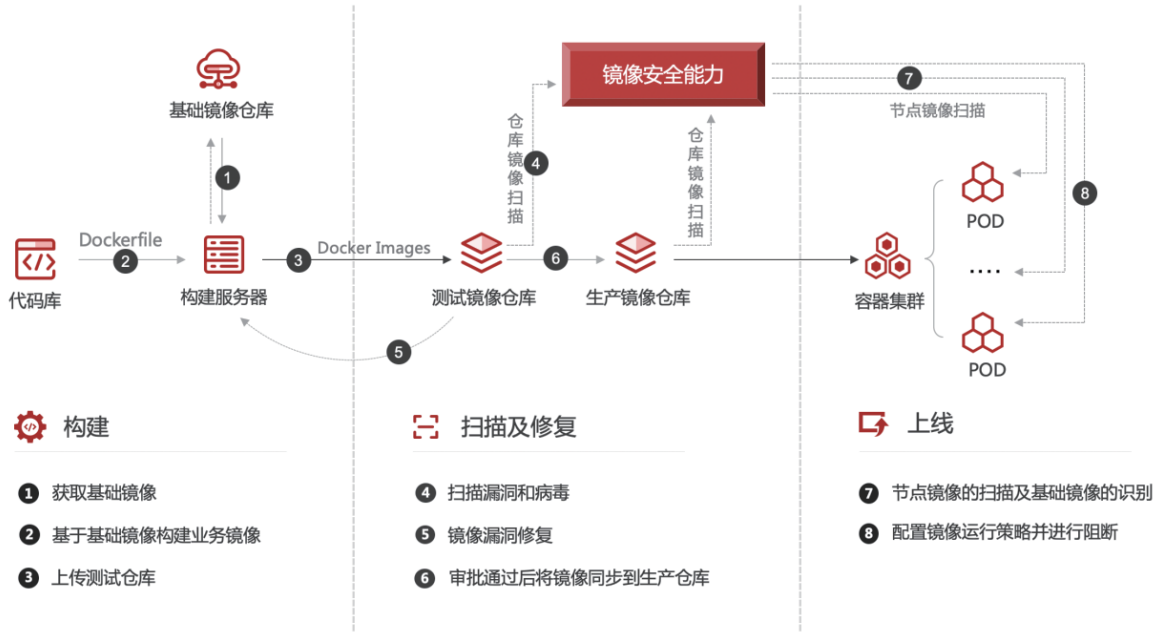


图3 某运营商镜像安全能力

镜像安全建立了对研发侧交付制品检测及阻断的能力，检测能力可识别镜像中可能存在的软件漏洞、软件许可、恶意文件、敏感信息等多个方面的安全风险。阻断能力可防止风险镜像流入线上业务。在能力落地前，研发流程为从公网拉取基础镜像，通过开发构建后，经过测试验证后，满足业务功能需求则直接上线。而能力融入后，在业务镜像构建完成后，以及上线前，又嵌入了安全检测的步骤。极大的加强了线上的安全性。

（2）安全薄弱点能力提升

在推进以容器为基础平台的战略过程中，发现容器平台由于自身的技术实现，大多的传统安全能力已经不再适用，例如入侵检测无法侵入容器内部、容器平台漏洞及配置合规性无法验证、基于 IP 的访问控制手段无法定位具体业务等。需要通过全方位的容器安全能力提升补足短板。

图4 某运营商云原生安全薄弱点监控

基于此对安全薄弱点进行全方位安全能力提升，建立了 IAC 配置安全检测、容器的入侵检测、微服务的漏洞发现、容器集群安全等相关能力，填补了应用容器技术后，多个安全建设空白。对容器涉及到的全生态，业务的全生命周期建立了安全防护。且落地只需在容器集群内运行安全终端，即可实现一键防护，形成整体的安全防护体系。

（3）安全管理提升

不单单是安全问题，在管理方面也有待提升，这包括由于容器技术的实现机制，导致业务容器在由于迭代更新、编排调度等消逝后，其在运行过程中存在的行为事件将不可查。对溯源、排障等多个方面都带来极大影响；且在应用容器技术后，对于云外及云内流量可以感知并进行检测，但对于容器集群内的网络流量还暂未形成清晰台账，内部的业务连接关系无法感知；以及由于细化到命名空间级的权限配置较为复杂，需要一定的技术积累，对人员要求较高，如何使得各业务线管理自己的容器也需要新的手段。

因此，建立了一个整体的安全管理平台，并搜集容器的所有事件行为，对容器资产进行管理。在记录容器的所有行为事件的同时，还可自定义时间留存信息。网络的事件也会进行拓扑绘制，并可进行数据导出，外加安全能力的补足，形成了整体的解决方案。在使用上也可基于功能以及资产两个维度进行权限划分，适应多租户场景。

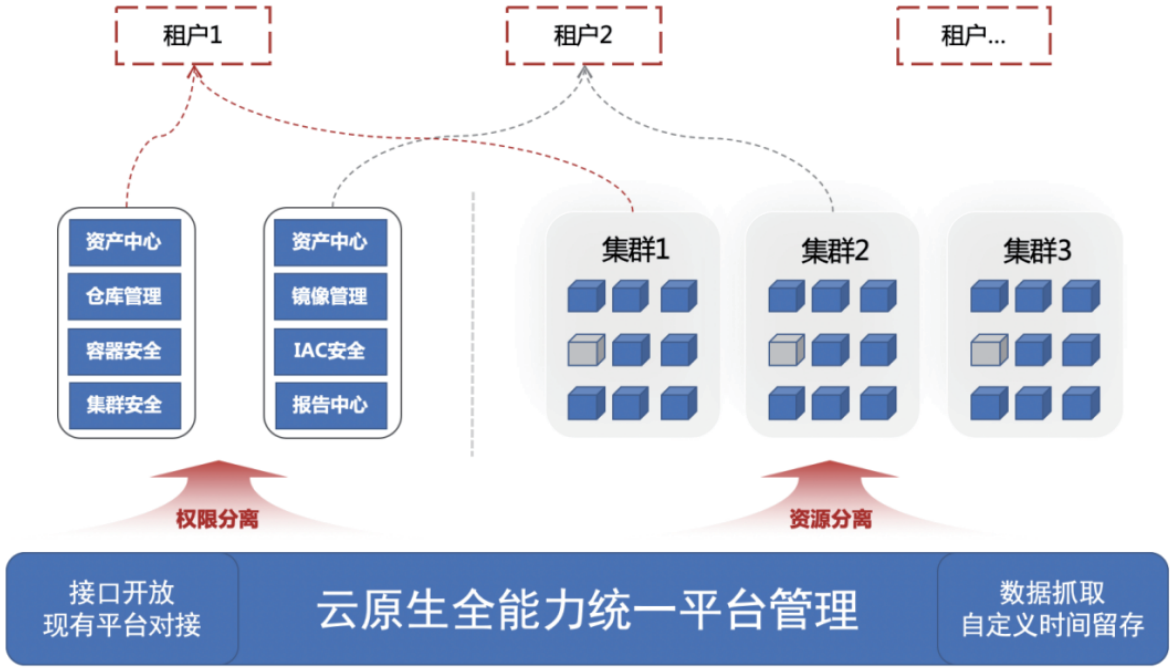


图5 某运营商统一云原生安全管理平台

4. 价值体现

通过整体解决方案的落地，首先补足了该运营商研究院在容器方面安全的空白，建立起自构建镜像 - 镜像存储 / 分发 - 容器运行 - 提供服务 - 业务消逝 - 基础平台的全生命周期的风险感知，以及安全防护能力。其次，将安全能力与开发平台进行深度结合后，形成 DevSecOps，使得安全风险由事后感知修复，向事前评估整改转变。最后，通过一个平台，对所有的能力形成统一管理，并建立了报告中心、资产中心、漏洞管理、响应中心等一系列安全运营能力，使得安全人员可感知、可处置、可溯源，同时加强了人员效率。

（二）保险业

1. 需求分析

在当今数字化时代，应用安全至关重要，它不仅是企业业务的基石，更是防止数据泄露和系统被攻击的关键。近年来，攻防对抗日趋激烈，互联网应用系统侧的沦陷是较为常见的突破口。太平洋保险因业务需要，部分业务系统部署使用了外部成品软件，这些外部软件存在的已知和未知安全漏洞依赖外部厂商修复，修复时效往往难以达到公司要求。此外，部分高危成品软件可能存在的 0Day 漏洞，在黑客手中掌握可静默攻击受影响应用，甲方单位对此类漏洞面临无补丁或无修复方案的问题，安全防护挑战大。因此，太平洋保险引入了应用安全运行时防护技术 RASP（Runtime application self-protection），将防护引擎插桩到应用内部，能够感知应用上下文拦截从应用程序到系统的所有调用，实时检测和阻断安全攻击，作为抵御 0Day 漏洞高发或无法修复、难以防御且需开放使用应用的防护补偿。

2. 整体解决方案设计

① 提升“两高一弱”组件漏洞防护能力

“两高一弱”涵盖了高危端口（服务）和高危漏洞，对于 Fastjson、Shiro 等使用广泛、版本碎片化的第三方组件库和开发框架组件，往往多个版本存在高危漏洞，由于涉及开发代码层面的组件引用，存在修复难度大，时效长，防护难度高的特点。本次形成了在漏洞修复前，集高危组件版本识别、应用运行时免疫防护一体的方案。

② 建立应用运行时模型

太平洋保险针对应用实施 RASP 方案，制定专属安全规则。规则基于应用的日常行为建立基线，形成黑白灰名单。黑名单包括确定的攻击行为实施拦截，白名单涵盖正常业务操作进行排除，对于 SQL 注入以及一些应用反序列化等较为难以简单归类的行为，将其归类为灰名单实施告警后人工介入分析研判，同时 RASP 工具提供了丰富的上下文及堆栈，能够辅助专业安全人员或特定的安全大模型进行准确的研判。这种动态监测方法不仅提高了威胁检测的准确性，还能及时发现异常行为，防止潜在的安全事件发生，进一步强化应用安全防护。

③ 实现应用检测与响应

通过实现应用的检测与响应（Application Detection & Response，ADR）理念，能够对应用进行深入分析和实时监控，快速识别并响应安全事件，建立应用运行时监测模型，应对不断变化的安全 0day 威胁，保持业务的连续性和数据的完整性。

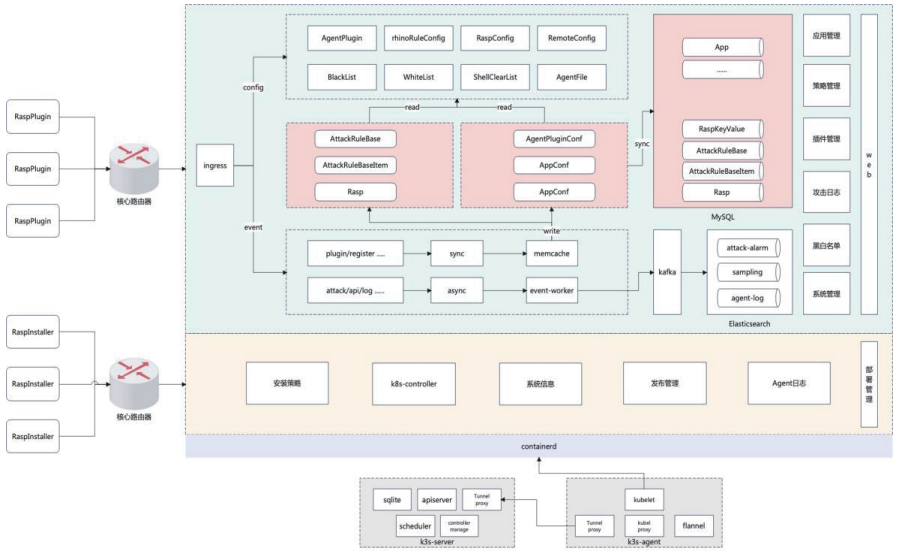


图6 太平洋保险RASP技术架构图

3. 实施技术路线

1

利用RASP技术进行高危组件及应急漏洞防护补偿

通过采用 RASP（Runtime Application Self-Protection）运行时应用自我保护技术，对高危组件及应急漏洞进行收敛与防御。RASP 插桩技术与真实业务流量结合，通过应用行为发现潜在攻击。此外，通过监控还可以发现漏洞导致的异常行为，深度洞察发现漏洞利用。RASP 的动态安全防护机制，随应用变化能够不断学习和适应新的威胁模式，这种预防性措施能提前布防，增强面对真实攻击的防御能力。

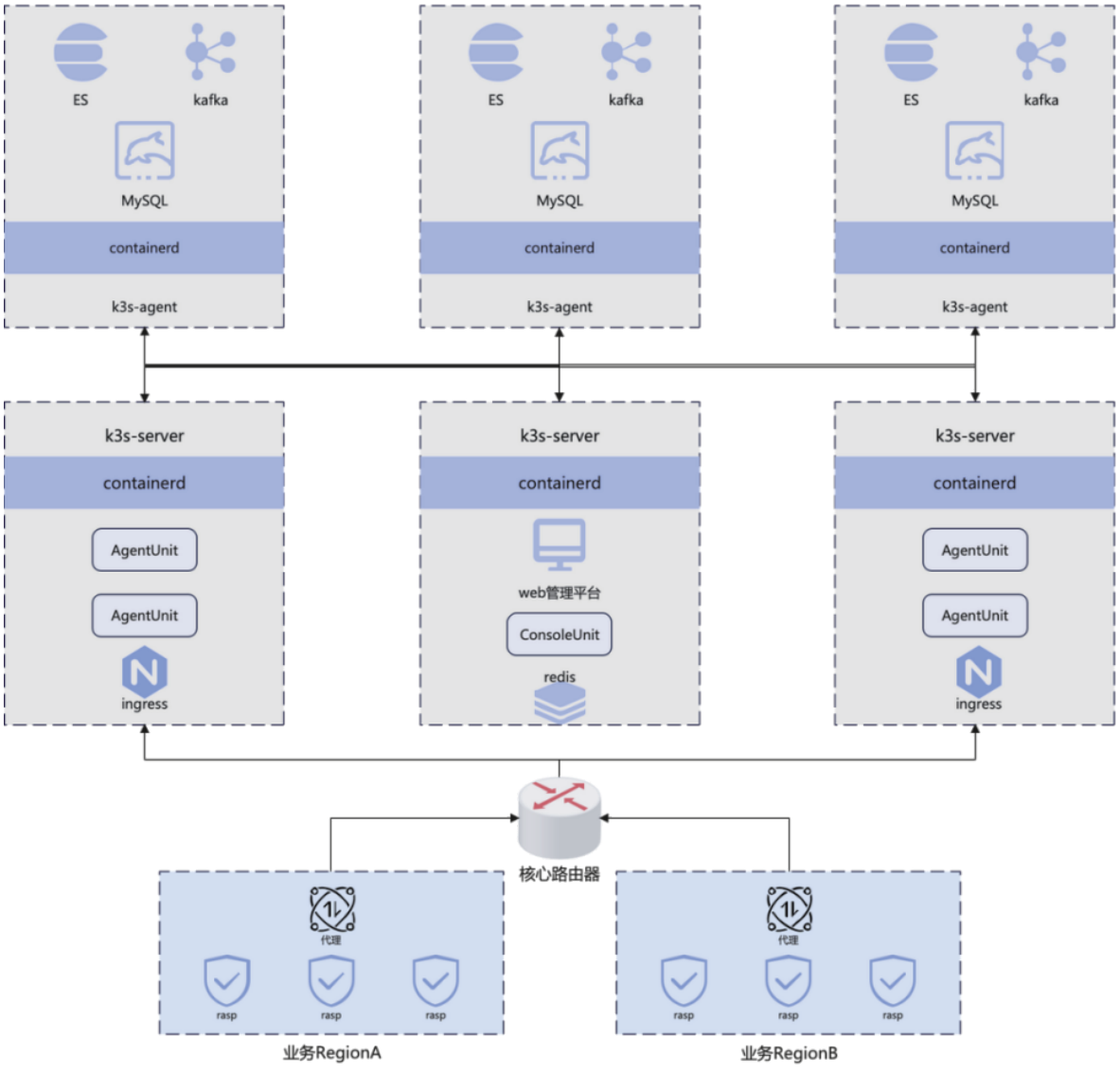


图7 太平洋保险RASP部署架构图

2 **ASP与WAF的防御相结合**	在应用节点部署 RASP 后，可形成与 WAF 协同工作、联合封禁和溯源。当 WAF 检测到可疑流量时，可以迅速通知 RASP 进行更深入的应用层分析；反之，RASP 发现的异常行为也可以反馈给 WAF，用于优化流量过滤规则。这种无缝配合不仅提高了威胁检测的准确性，还大大缩短了响应时间。
3 **RASP工具的0day攻击感知和内存马发现**	RASP 技术能够在应用运行时，实时监控内存中的行为，通过异常行为的感知，及时发现并清除内存马。此外，RASP 还能够感知其他异常攻击行为，识别潜在的“0day 漏洞”，在官方尚未发布补丁的情况下，通过 RASP 工具提前发现并防御这些漏洞。

4. 价值体现

1 **RASP工具整体部署情况**	目前公司部署 RASP 的应用已接近 150 个，部署方式囊括传统云主机、Docker 容器化、K8S 集群。防护策略细分了高危组件应用拦截、专项应用拦截、通用应用拦截策略。上述应用的约 1000 个节点部署 RASP 后，对 WebLogic、Tomcat、SpringBoot、Jetty、宝兰德、东方通等国内外中间件插桩适配性运行稳定，防护有效。
2 **检测和防护效果实例**	在内部安全攻防演练阶段，RASP 工具捕捉到了一项潜在攻击警告。通过 RASP 工具内置的智能化分析引擎，对攻击行为进行了即时的剖析与研判，最终确认这是一次针对系统上传功能的漏洞利用尝试。利用 RASP 从应用行为层面切入的机制，直接拦截了试图利用该上传漏洞的恶意操作，有效遏制了安全风险蔓延。 针对另一起 SQL 注入针对内部某保险系统的攻击告警，经过对 SQL 语句的研判及业务核实，确认该 SQL 语句虽属正常业务但执行方式却与典型的 SQL 注入攻击模式高度相似，对此正常 SQL 请求加白后，SQL 注入的阻断策略仍然有效。

3

经验总结

因为 RASP 工具与应用紧密结合，因此不能盲目随意注入安装，不同应用环境下的 RASP 部署也不尽相同。因此，采用了事前完善的测试来验证 RASP 产品与应用的适配情况，通过测试后方可上线运行，同时也可以在测试环境获取一定的应用行为，进行上线前的规则学习及调参。同时在测试、投产、推广过程中，太平洋保险的安全部门与应用研发、应用运维部门紧密协同，平滑实施切换拦截策略，保障业务连续性。

（三）制造业

1. 需求分析

在当前的云计算发展中，安全团队面临着前所未有的挑战。随着组织越来越多地采用多云环境并优先开发云原生应用程序，为保障应用安全的复杂性呈指数级增长。为了应对这些不断变化的云安全需求，迫切需要整体的解决方案——云原生应用程序保护平台（CNAPP）。

很多制造行业公司，出于保障工厂稳定生产或用户高可用的考虑，都采用多云部署应用的架构，公有云、私有云、混合云甚至多家公有云厂商共同部署。

CNAPP 平台旨在整合多云威胁预防和检测，提供从代码到云的全面安全性。云原生应用程序保护平台（CNAPP）提供了一个集中和集成的云原生安全解决方案，用于监控、管理和检测云上安全风险，能够在云基础设施内进行主动管理和风险缓解。它涵盖了云安全的重要方面，包括云安全态势管理（CSPM）、云基础设施权限管理（CIEM）和云工作负载保护（CWP）。

随着各制造业厂商在云原生战略上的加速，保护云应用程序和工作负载也需要一种不同的方式来应对威胁。与传统的 IT 防护系统不同，在传统 IT 系统中，组织的信息主要包含在内部数据中心内，安全性侧重于防御外部威胁，然后云原生则不同于上述的方式。云安全必须满足跨提供商和云环境（公共、私有和混合）访问数据的要求。随着威胁的演变，云安全解决方案还必须适应动态变化的基础设施。

为了更好地保障应用安全、数据安全和云上基础设施等核心资产，云原生应用程序保护平台（CNAPP）在制造业中的应用日益广泛和深入。制造业也会打造一套全面的安全与合规解决方案，覆盖从研发到生产运营的各个环节。

2. 整体解决方案设计

图8 某制造业云原生安全运营平台架构

制造业 CNAPP 采用分布式的部署架构与微服务来提升系统的灵活性和可扩展性。CNAPP 在这种架构中通过 API 集成和代理集成，提供实时的安全监控与分析。特别是在微服务环境中，CNAPP 能够扫描和审查各个服务组件的安全配置与行为，确保每个服务的安全合规运行。

在竞争激烈的制造业中，敏捷开发和快速交付至关重要。CNAPP 可以无缝集成到 CI/CD 管道中，从代码提交到生产部署的每个阶段进行安全检测和风险评估。例如，通过基础设施即代码（IaC）扫描和容器扫描，确保代码和配置文件在进入生产环境前已经过严格的安全审查。例如汽车制造行业经常需要在高峰需求时扩展资源，例如在用户早晚出行高峰期，节假日出行高峰需要增加计算和存储资源。CNAPP 提供的自动扩展和安全策略能够动态适应资源变化，确保在扩展过程中不引入安全风险。安全配置管理功能帮助确保在不同环境中的一致性和安全性。

3. 实施技术路线

① 痛点安全风险闭环与能力覆盖提升

在云原生应用程序保护平台（CNAPP）的初期阶段，核心目标在于识别并解决安全风险，以及提升能力覆盖度。

首先，关于痛点安全风险的发现与解决，团队需要对云环境进行广泛的风险识别，通过自动化扫描和实时监控来发现潜在的安全问题。每个风险都应根据其严重程度和对业务的潜在影响进行评估，从而合理地分配资源和优先级，确保高风险问题得到及时处理。同时，通过持续监控和分析环境变化，可以确保新发现的风险被迅速识别并解决，从而维持一个防御稳固且动态响应的安全体系。

其次，能力覆盖度的提升是另一个关键领域。将安全功能全面集成到 DevOps 工作流程中，可以实现自动化响应和政策一致性。通过这种方式，安全不再是一个孤立的活动，而是成为应用程序开发和部署不可分割的一部分。此外，提升基础设施的可见性非常重要。通过先进的工具和仪表盘，团队可以获得应用、数据和网络流量的全面视图，从而快速识别潜在威胁并实施防御策略。

最后，扩展合规性支持也是提升能力覆盖度的重要组成部分。随着行业标准和法规的不断演变，确保我们的安全策略和实践与之保持一致至关重要。通过建立灵活的合规框架，企业可以在不影响业务敏捷性的前提下，快速适应新的合规要求。

通过在初期阶段的有效风险发现与能力提升，CNAPP 可以为企业提供全面、动态的安全保障，支持业务的持续发展和创新。

② 多场景适配，借助CNAPP集成化打造安全生态体系

制造业中多云场景以及复杂的生产环境，也是 CNAPP 面临的挑战之一，通过多场景适配和集成化，打造全面的安全生态体系，满足制造业多样化的安全需求。随着企业业务的扩展和应用场景的多样化，CNAPP 需要具备在不同云环境和应用架构中灵活适应的能力。无论是本地数据中心、混合云还是多云策略，CNAPP 都能提供一致性的安全保障，确保所有环境中的应用安全水平相一致。

CNAPP 通过集成各类安全工具和功能模块，形成一体化安全管理平台。它将威胁检测、防火墙、入侵检测等多个功能整合在一起，实现信息和操作的统一管理。通过这种集成化的安全框架，是安全工程师能够更高效地管理和响应安全事件，减少信息孤岛的出现。

通过多场景适配和集成化策略，CNAPP 需支持制造业公司在各种场景下的安全需求。

③ AI与自动化

当前制造业在云原生应用程序保护平台（CNAPP）的建设方向，重点将转向利用 AI 和自动化来提高处理效率和应急响应能力。

借助人工智能可以实现更智能的威胁检测和预测分析。AI 可以通过机器学习模型分析大量的日志和活动数据，从中识别出威胁并串联攻击链路。这种能力不仅能够快速响应新的攻击，还可以通过积累的经验，不断提升自动化处置能力。助力于风险发现、响应和修复的整个流程。在响应和修复阶段，通过自动化脚本和工具，安全团队可以在不需要人工干预的情况下，执行标准的修复流程，确保持续的环境保护。

通过人工智能和自动化的结合，CNAPP 能够大幅提高运营效率，减轻安全团队的负担。同时，这种高效的安全管理方式还能够为企业节省时间和成本，提高整个安全策略的灵活性和可扩展性，确保在复杂的云环境下实现强有力的保护。

4.价值体现

对于制造行业来说，专为满足复杂云环境和云原生应用程序的安全需求而设计的解决方案，CNAPP 使公司能够整合其整体安全信息。并总览在云上的安全态势。安全团队可以在应用程序生命周期的早期识别并优先解决问题。

有效的 CNAPP 解决方案简化了云安全运营上的操作，提供了跨云资源和环境的统一和场景化的风险视图。这有助于制造业中的开发团队与安全团队在开发的早期合作阶段优先考虑安全性，对于制造行业来说，优先融入安全设计，远比后期单点性的处置安全漏洞，对于交付效率上来说更让人接受，并克服传统安全解决方案的常见问题，比如可观测性不足、告警堆积、冗余的管理平台以及低效的安全运营方式。

通过 CNAPP 的建设与不断实践，公司能够相对轻松地管理安全风险，而不会减缓或中断运营。安全团队不会再筛选无数告警，并可以整合的高优先级告警，在很短的时间内为应急处置做出最佳决策。CNAPP 支持开发和运营团队之间的无缝协作，以简化风险管理和缓解。缩短了识别、分类和优先考虑云安全风险应急工作流程的时间，提高了管理和降低风险的效率。从而提高其整体安全投资回报率。

（四）互联网

1. 需求分析

作为一家受到海外监管的上市公司，老虎国际始终重视网络安全和安全合规建设。我们定期提供安全配置检查、漏洞修复、补丁管理和攻击识别的及时响应，确保合规要求全面落实和安全水位提升。

自 2021 年以来，老虎国际积极推进混合云的建设转型，构建完善的 DevSecOps 体系，并将产品研发安全视为重要核心。在云平台和 DevSecOps 的建设过程中，安全防护与 CI/CD 环节的安全检查成为老虎国际安全团队的首要任务，是保障软件应用和代码质量的基石。我们亟需一款能够全面覆盖传统主机安全，同时有效应对云原生环境中的安全挑战的产品。该产品还需无缝集成到我们的 DevSecOps 体系中的 CI/CD 流程，确保云原生环境内部所有容器和 Pod 的安全性得到充分保障。

2. 整体解决方案设计

技术架构	通过慎重评估及测试，我们最终采用集成青藤蜂巢和万相平台形成混合云安全整体解决方案，集群化部署。保证每一个上线的服务器和云原生集群均可纳入统一管理。

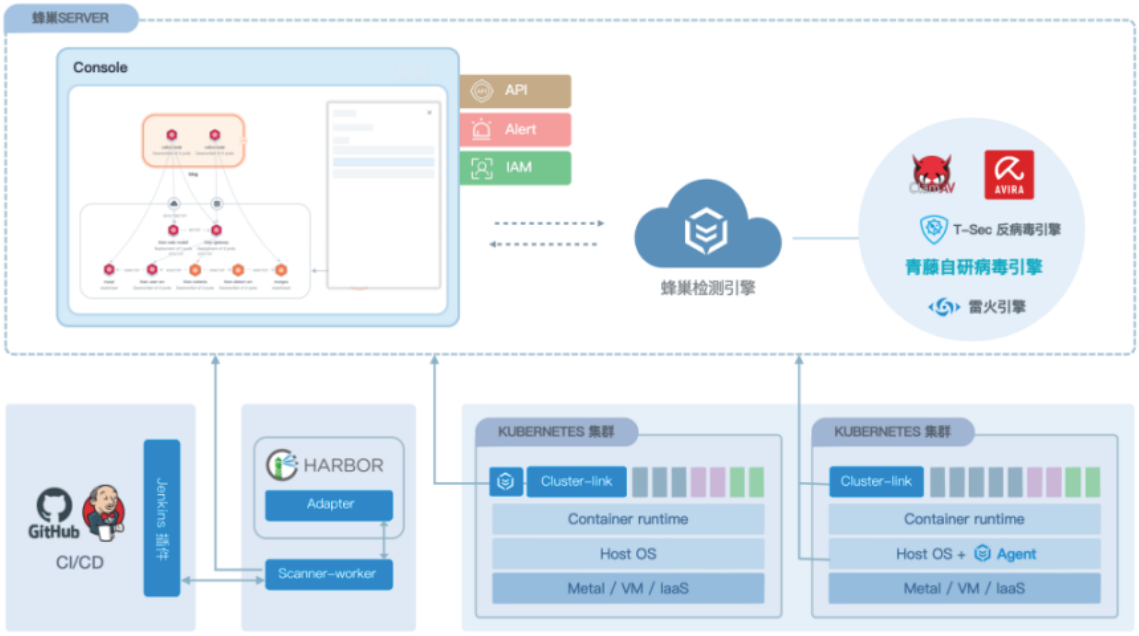


图9 老虎国际云原生安全平台

关键技术	在云原生方面采用蜂巢解决方案实现包括 K8s、Container、Docker 以及镜像安全，通过入侵事件告警可以快速定位混合云环境下的攻击特征、攻击目标和使用的工具命令，以便及时响应处置。 · **镜像安全：** 可以发现镜像存在的安全风险，了解镜像使用的 image 操作系统版本、是否含有病毒、木马和敏感文件等； · **应用风险：** 可以发现镜像中引用的不安全的组件漏洞，通过 POC 方式验证，确定漏洞的真实性。 · **集群安全：** 检测 k8s 及组件存在的安全风险，runc、docker 权限绕过等安全问题； · **容器雷达和微隔离：** 可以通过容器雷达和微隔离实现核心业务的服务间的细粒度隔离策略，快速了解业务间的访问关系、识别服务间的异常流量，发现潜在的恶意访问行为，为高风险高要求业务场景提供精细化管控的能力。 · **万相主机安全：** 在主机方面采用万相解决方案实现主机层面的安全态势感知能力。核心功能是资产清点、风险发现、入侵检测、日志采集和安全响应等，将这些功能统一纳入安全运营和安全应急响应的体系中，更高效的定位和处置安全风险。
主要能力	通过集成部署万相主机安全平台和蜂巢云原生安全平台，该方案可实现主机到混合云环境下的安全补丁和漏洞管理，对应用组件漏洞、木马病毒、敏感信息、开源许可、镜像阻断等主要功能提供详细可执行的处置方案，解决主机和混合云环境安全补丁、组件漏洞、木马病毒和敏感信息等安全风险，有效提升公司整体安全水位。 同时，在 DevSecOps 的 CI 阶段集成镜像检查，可以在 CI 阶段发现并阻止带有高危安全风险的镜像系统、应用程序集成，在上线前发现高危风险并及时处置。
技术优点	主机安全、容器安全协同统一防护；多 k8s 集群可以实现全面统一管理；镜像扫描模块可以分布部署和 DevSecOps 流程融合，通过与 LDAP 集成实现多业务多岗位的账号权限管控。

3. 实施技术路线

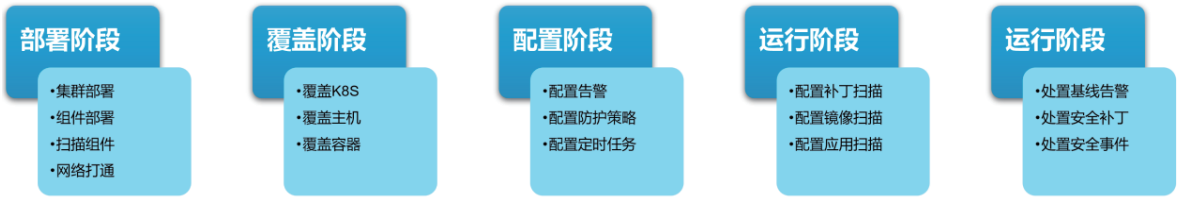


图10 老虎国际云原生安全实施技术路线

① **部署阶段：**青藤服务集群、集群组件和扫描组件安装部署，网络策略打通；

② **覆盖阶段：**青藤 Agent 安装覆盖所有主机、k8s 集群主机和容器；

③ **配置阶段：**配置告警和安全防护策略，确保实现自动告警；

④ **运行阶段：**配置漏洞定时扫描任务，可以及时发现漏洞风险；

⑤ **日常运营：**配置基线检查策略，定期对业务组发起基线检查并处置风险。

4. 价值体现

- 快速实现整体配置合规。基于 CIS Benchmark 基线快速检查容器、镜像、Docker、cri-o、K8s 主机、主机、MySQL、Nginx 的安全配置实现安全配置的 100% 符合。
- 混合云中 K8s 集群安全保障。快速发现基础设施包括 K8s、Docker 和 Harbor 镜像的安全风险，支持漏洞快速修复，对于操作系统漏洞实现 100% 处置、修复。
- 安全事件的快速发现、定位和响应。可以实时发现命令执行、反弹 shell、webshell 后门、本地提权、暴力破解、病毒后门、可疑操作等异常事件，可以实现 100% 的响应处置。
- 微隔离实现核心业务细粒度网络访问控制，从端口、进程、容器 pod 等维度配置精细化的 ACL 策略，支持一键隔离失陷主机和容器，实现重要业务的“零信任”安全，减少风险暴露。

06 CNAPP未来发展趋势和展望

云原生应用程序保护平台（CNAPP，Cloud-Native Application Protection Platform）是近年来网络安全领域的一个重要趋势。随着企业向云计算和容器化架构的转型，传统的安全模型已经无法满足现代应用环境的需求，CNAPP 因此应运而生。CNAPP 的核心目标是为云原生环境提供一套统一的安全解决方案，涵盖从开发到生产的整个生命周期。以下是 CNAPP 未来发展的一些主要趋势和展望：

（一）全面化

1. **全面的端到端安全：**CNAPP 将整合并扩展其安全覆盖范围，从开发阶段的代码扫描和依赖性管理，到生产环境的实时威胁检测与响应，确保应用程序生命周期的每个阶段都得到充分的安全保障。

2. **多云和混合云环境的支持：**CNAPP 将支持跨多个云服务提供商的一致安全性，包括统一的策略管理、跨云的威胁情报共享，以及无缝的日志和事件管理。

3. **DevSecOps 的进一步集成：**CNAPP 将进一步推动安全与开发和运维过程的深度集成，使安全性成为开发周期中的一个自然环节，通过工具链的集成，开发者能够在开发的早期阶段就检测和修复安全问题。

4. **开源工具的整合与支持：**CNAPP 将更多地与开源安全工具和框架集成，提供更广泛的功能支持，降低企业的安全成本，并借助开源社区的力量推动技术创新。

5. **与其他安全解决方案的融合：**CNAPP 将与其他安全解决方案（如零信任、SIEM、SOAR、EDR 等）进行更紧密的集成，形成一个更强大的安全生态系统，实现跨平台的威胁情报共享和联合响应。

（二）智能化

1. **安全检测智能化：**随着 AI 和机器学习技术的进步，CNAPP 将更多地依赖自动化和智能化的安全功能，如 AI 驱动的威胁检测系统，对异常行为进行实时分析，并自动响应。

2. **安全运营智能化：**通过安全大模型或安全智能体应用整合安全最佳实践和内部策略形成知识库，开发基于大模型的对话接口，实现多轮对话，保持对话上下文，与现有安全工具集成，提供 7*24 小时的智能辅助，加速问题诊断和解决过程。

（三）自动化

1. **合规性与治理的自动化：**CNAPP 将帮助企业自动化合规性检查和治理流程，确保应用程序和数据始终符合相关法规要求，减少合规性的复杂性。

2. **安全防护自动化：**CNAPP 将提供更深入的自动化安全功能来保护容器和微服务架构，包括容器镜像的安全扫描、运行时监控、以及对编排工具的全面安全管理。

总结

CNAPP 未来的发展将继续沿着更加全面化、智能化、自动化的方向前进。随着云原生架构的普及和复杂性的增加，CNAPP 的重要性将日益凸显。企业需要积极拥抱这一趋势，确保其云环境和应用程序的安全性能够与时俱进。

获取更多

《安全洞察·大咖说》