openEuler 24.03 LTS SP2 技术白皮书

1. 概述

OpenAtom openEuler（简称“openEuler”） 社区是一个面向数字基础设施操作系统的开源社区。由开放原子开源基金会（以下简称“基金会”）孵化及运营。

openEuler 是一个面向数字基础设施的操作系统，支持服务器、云计算、边缘计算、嵌入式等应用场景，支持多样性计算，致力于提供安全、稳定、易用的操作系统。通过为应用提供确定性保障能力，支持 OT 领域应用及 OT 与 ICT 的融合。

openEuler 社区通过开放的社区形式与全球的开发者共同构建一个开放、多元和架构包容的软件生态体系，孵化支持多种处理器架构、覆盖数字基础设施全场景，推动企业数字基础设施软硬件、应用生态繁荣发展。

2019 年 12 月 31 日，面向多样性计算的操作系统开源社区 openEuler 正式成立。

2020 年 3 月 30 日， openEuler 20.03 LTS （Long Term Support，简写为 LTS，中文为长生命周期支持）版本正式发布，为 Linux 世界带来一个全新的具备独立技术演进能力的 Linux 发行版。

2020 年 9 月 30 日，首个 openEuler 20.09 创新版发布，该版本是 openEuler 社区中的多个企业、团队、独立开发者协同开发的成果，在 openEuler 社区的发展进程中具有里程碑式的意义，也是中国开源历史上的标志性事件。

2021 年 3 月 31 日，发布 openEuler 21.03 内核创新版，该版本将内核升级到 5.10，并在内核方向实现内核热升级、内存分级扩展等多个创新特性，加速提升多核性能，构筑千核运算能力。

2021 年 9 月 30 日，全新 openEuler 21.09 创新版如期而至，这是 openEuler 全新发布后的第一个社区版本，实现了全场景支持。增强服务器和云计算的特性，发布面向云原生的业务混部 CPU 调度算法、容器化操作系统 KubeOS 等关键技术；同时发布边缘和嵌入式版本。

2022 年 3 月 30 日，基于统一的 5.10 内核，发布面向服务器、云计算、边缘计算、嵌入式的全场景 openEuler 22.03 LTS 版本，聚焦算力释放，持续提升资源利用率，打造全场景协同的数字基础设施操作系统。

2022 年 9 月 30 日，发布 openEuler 22.09 创新版本，持续补齐全场景的支持。

2022 年 12 月 30 日，发布 openEuler 22.03 LTS SP1 版本，打造最佳迁移工具实现业务无感迁移，性能持续领先。

2023 年 3 月 30 日，发布 openEuler 23.03 内核创新版本，采用 Linux Kernel 6.1 内核，为未来 openEuler 长生命周期版本采用 6.x 内核提前进行技术探索，方便开发者进行硬件适配、基础技术创新及上层应用创新。

2023 年 6 月 30 日，发布 openEuler 22.03 LTS SP2 版本，场景化竞争力特性增强，性能持续提升。

2023 年 9 月 30 日，发布 openEuler 23.09 创新版本，是基于 6.4 内核的创新版本（参见版本生命周期），提供更多新特性和功能，给开发者和用户带来全新的体验，服务更多的领域和更多的用户。

2023 年 11 月 30 日，发布 openEuler 20.03 LTS SP4 版本，其作为 20.03 LTS 版本的增强扩展版本，面向服务器、云原生、边缘计算场景，提供更多新特性和功能增强。

2023 年 12 月 30 日，发布 openEuler 22.03 LTS SP3 版本，是 22.03 LTS 版本增强扩展版本，面向服务器、云原生、边缘计算和嵌入式场景，持续提供更多新特性和功能扩展，给开发者和用户带来全新的体验，服务更多的领域和更多的用户。

2024 年 5 月 30 日，发布 openEuler 24.03 LTS，基于 6.6 内核的长周期 LTS 版本（参见版本生命周期），面向服务器、云、边缘计算、AI 和嵌入式场景，提供更多新特性和功能，给开发者和用户带来全新的体验，服务更多的领域和更多的用户。

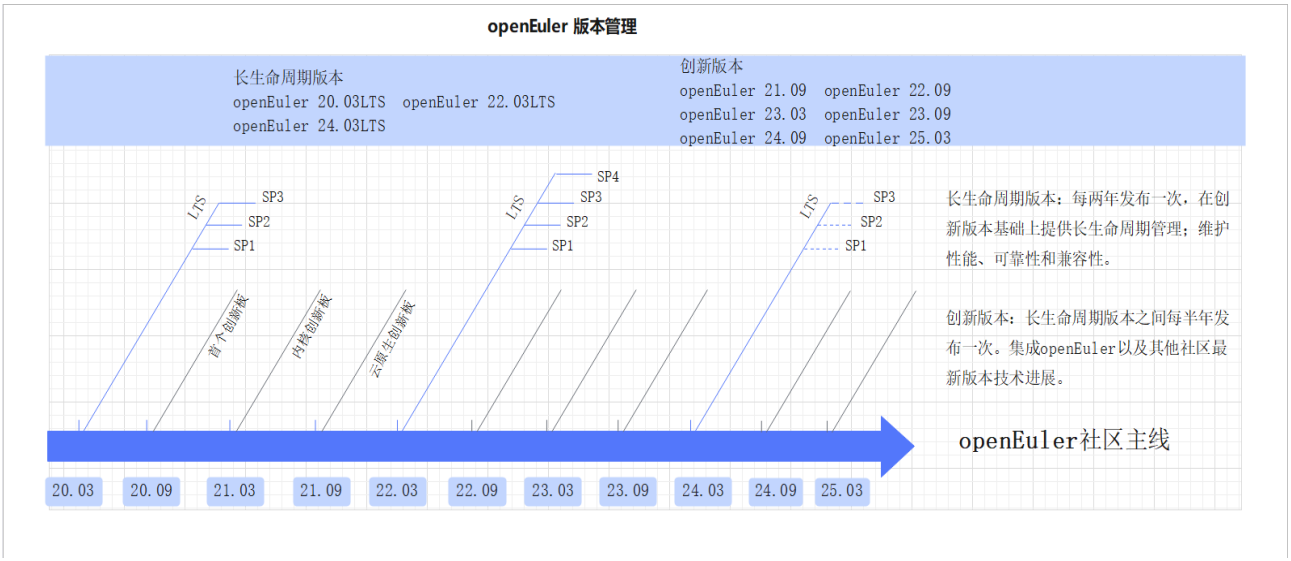
2024 年 6 月 30 日，发布 openEuler 22.03 LTS SP4，是 22.03 LTS 版本增强扩展版本，面向服务器、云原生、边缘计算和嵌入式场景，持续提供更多新特性和功能扩展，给开发者和用户带来全新的体验，服务更多的领域和更多的用户。

2024 年 9 月 30 日，发布 openEuler 24.09，基于 6.6 内核的创新版本，提供更多新特性和功能。

2024 年 12 月 30 日，发布 openEuler 24.03 LTS SP1，基于 6.6 内核的 24.03-LTS 版本增强扩展版本（参见版本生命周期），面向服务器、云、边缘计算和嵌入式场景，持续提供更多新特性和功能扩展，给开发者和用户带来全新的体验，服务更多的领域和更多的用户。

2025 年 3 月 30 日，发布 openEuler 25.03 是基于 6.6 内核的创新版本，面向服务器、云、边缘计算和嵌入式场景，提供更多新特性和功能，给开发者和用户带来全新的体验，服务更多的领域和更多的用户。

2025 年 6 月 30 日，发布 openEuler 24.03 LTS SP2，基于 6.6 内核的 24.03-LTS 版本增强扩展版本（参见版本生命周期），面向服务器、云、AI 和嵌入式场景，持续提供更多新特性和功能扩展，包括内核优化、异构协同推理、众核高密、机密容器、多核多实例混部等，给开发者和用户带来全新的体验，服务更多的领域和更多的用户。

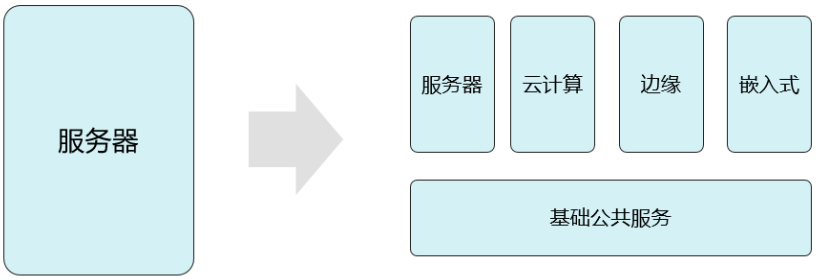


openEuler 作为一个操作系统发行版平台，每两年推出一个 LTS 版本。该版本为企业级用户提供一个安全稳定可靠的操作系统。

openEuler 也是一个技术孵化器。通过每半年发布一个创新版，快速集成 openEuler 以及其他社区的最新技术成果，将社区验证成熟的特性逐步回合到发行版中。这些新特性以单个开源项目的方式存在于社区，方便开发者获得源代码，也方便其他开源社区使用。

社区中的最新技术成果持续合入社区发行版，社区发行版通过用户反馈反哺技术，激发社区创新活力，从而不断孵化新技术。发行版平台和技术孵化器互相促进、互相推动、牵引版本持续演进。

openEuler 覆盖全场景的创新平台



openEuler 已支持 X86、ARM、SW64、RISC-V、LoongArch 多处理器架构及 PowerPC 芯片架构，持续完善多样性算力生态体验。

openEuler 社区面向场景化的 SIG 不断组建，推动 openEuler 应用边界从最初的服务器场景，逐步拓展到云计算、边缘计算、嵌入式等更多场景，openEuler 正成为覆盖数字基础设施全场景的操作系统。

openEuler 希望与广大生态伙伴、用户、开发者一起，通过联合创新、社区共建，不断增强场景化能力，最终实现统一操作系统支持多设备，应用一次开发覆盖全场景。

openEuler 开放透明的开源软件供应链管理

开源操作系统的构建过程，也是供应链聚合优化的过程。拥有可靠开源软件供应链，是大规模商用操作系统的基础。 openEuler 从用户场景出发，回溯梳理相应的软件依赖关系，理清所有软件包的上游社区地址、源码和上游对应验证。完成构建验证、分发、实现生命周期管理。开源软件的构建、运行依赖关系、上游社区，三者之前形成闭环且完整透明的软件供应链管理。

2. 平台架构

系统框架

openEuler 是覆盖全场景的创新平台，在引领内核创新，夯实云化基座的基础上，面向计算架构互联总线、存储介质发展新趋势，创新分布式、实时加速引擎和基础服务，结合边缘、嵌入式领域竞争力探索，打造全场景协同的面向数字基础设施的开源操作系统。

openEuler 24.03 LTS SP2 发布面向服务器、云原生、边缘和嵌入式场景的全场景操作系统版本，统一基于 Linux Kernel 6.6 构建，对外接口遵循 POSIX 标准，具备天然协同基础。同时 openEuler 24.03 LTS SP2 版本集成分布式软总线、K3s 边云协同框架等能力，进一步提升数字基础设施协同能力，构建万物互联的基础。

面向未来，社区将持续创新、社区共建、繁荣生态，夯实数字基座。

夯实云化基座

- 容器操作系统 KubeOS：云原生场景，实现 OS 容器化部署、运维，提供与业务容器一致的基于 K8S 的管理体验。
- 安全容器方案：iSulad+shimv2+StratoVirt 安全容器方案，相比传统 Docker+QEMU 方案，底噪和启动时间优化 40%。
- 双平面部署工具 eggo：ARM/X86 双平面混合集群 OS 高效一键式安装，百节点部署时间<15min。

全场景

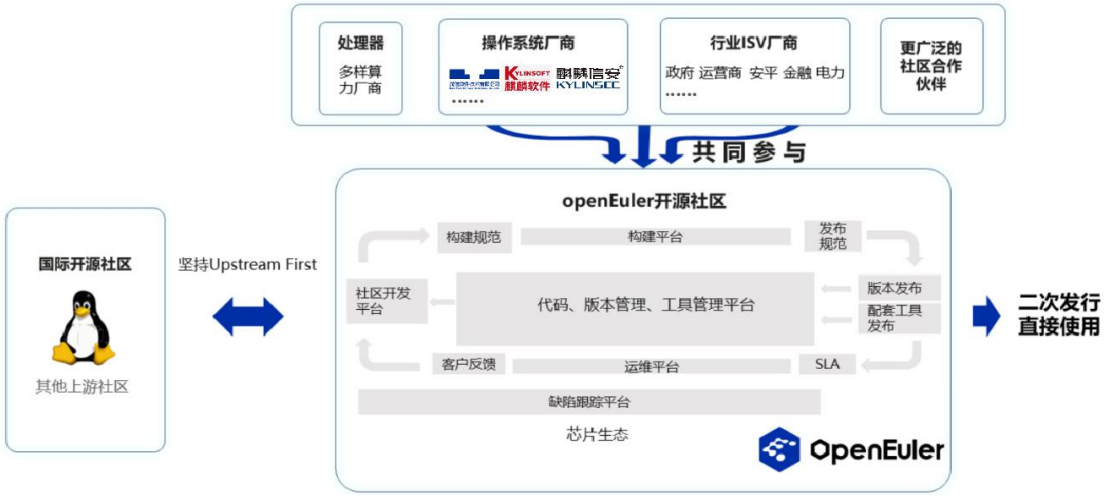
- 边缘计算：发布面向边缘计算场景的版本，支持 K3s 边云协同框架，具备边云应用统一管理和发放等基础能力。
- 嵌入式： 发布面向嵌入式领域的版本，该版本镜像大小 < 5M，启动时间 < 5s。
- AI 原生 OS：OS 使能 AI 软件栈，开箱即用；异构融合内存，调度，训推场景降本增效；智能化交互平台，赋能开发者及管理员。

繁荣社区生态

- 友好桌面环境：UKUI、DDE 、Kiran-desktop、GNOME 桌面环境，丰富社区桌面环境生态。
- openEuler DevKit：支持操作系统迁移、兼容性评估、简化安全配置 secPaver 等更多开发工具。

平台框架

openEuler 社区与上下游生态建立连接，构建多样性的社区合作伙伴和协作模式，共同推进版本演进。



硬件支持

openEuler 社区当前已与多个设备厂商建立丰富的南向生态，比如 Intel、AMD 等主流芯片厂商的加入和参与，openEuler 全版本支持 x86、Arm、申威、龙芯、RISC-V 五种架构，并支持多款 CPU 芯片，包括龙芯 3 号 、兆芯开先/开胜系列、Intel Sierra Forest/Granite Rapids、AMD EPYC 4/5 等芯片系列，支持多个硬件厂商发布的多款整机型号、板卡型号，支持网卡、RAID、FC、GPU&AI、DPU、SSD、安全卡七种类型的板卡，具备良好的兼容性。

支持的 CPU 架构如下：

硬件类型	x86_64	arm64	LoongArch	SW64	RISC-V
CPU	Intel、AMD、Hygon、兆芯	鲲鹏、飞腾	龙芯	申威	Sophgo、THead 等

全版本支持的硬件型号可在兼容性网站查询兼容性列表。

3. 运行环境

服务器

若需要在物理机环境上安装 openEuler 操作系统，则物理机硬件需要满足以下兼容性和最小硬件要求。

硬件兼容支持请查看 openEuler 兼容性列表：兼容性列表。

部件名称	最小硬件要求
架构	ARM64、x86_64、riscV、LoongArch64、SW64
内存	为了获得更好的体验，建议不小于 4GB
硬盘	为了获得更好的体验，建议不小于 20GB

虚拟机

openEuler 安装时，应注意虚拟机的兼容性问题，当前已测试可以兼容的虚拟机及组件如下所示。

1. 以 openEuler 24.03 LTS SP2 为 HostOS，组件版本如下：

- libvirt-9.10.0-16.oe2403sp2
- libvirt-client-9.10.0-16.oe2403sp2
- libvirt-daemon-9.10.0-16.oe2403sp2
- qemu-8.2.0-36.oe2403sp2
- qemu-img-8.2.0-36.oe2403sp2

2. 兼容的虚拟机列表如下：

HostOS	GuestOS(虚拟机)	架构
openEuler 24.03 LTS SP2	Centos 6	x86_64
openEuler 24.03 LTS SP2	Centos 7	aarch64
openEuler 24.03 LTS SP2	Centos 7	x86_64
openEuler 24.03 LTS SP2	Centos 8	aarch64
openEuler 24.03 LTS SP2	Centos 8	x86_64
openEuler 24.03 LTS SP2	Windows Server 2016	x86_64
openEuler 24.03 LTS SP2	Windows Server 2019	x86_64

部件名称	最小虚拟化空间要求
架构	ARM64、x86_64
CPU	2 个 CPU
内存	为了获得更好的体验，建议不小于 4GB
硬盘	为了获得更好的体验，建议不小于 20GB

边缘设备

若需要在边缘设备环境上安装 openEuler 操作系统，则边缘设备硬件需要满足以下兼容性和最小硬件要求。

部件名称	最小硬件要求
架构	ARM64、x86_64
内存	为了获得更好的体验，建议不小于 4GB
硬盘	为了获得更好的体验，建议不小于 20GB

嵌入式

若需要在嵌入式环境上安装 openEuler Embedded 操作系统，则嵌入式硬件需要满足以下兼容性和最小硬件要求。

部件名称	最小硬件要求
架构	ARM64、ARM32、x86_64
内存	为了获得更好的体验，建议不小于 512MB
硬盘	为了获得更好的体验，建议不小于 256MB

4. 场景创新

AI 场景创新

智能时代，操作系统需要面向 AI 不断演进。一方面，在操作系统开发、部署、运维全流程以 AI 加持，让操作系统更智能；另一方面，openEuler 已支持 ARM，x86，RISC-V 等全部主流通用计算架构，在智能时代，openEuler 也率先支持 NVIDIA、昇腾等主流 AI 处理器，成为使能多样性算力的首选。

OS for AI

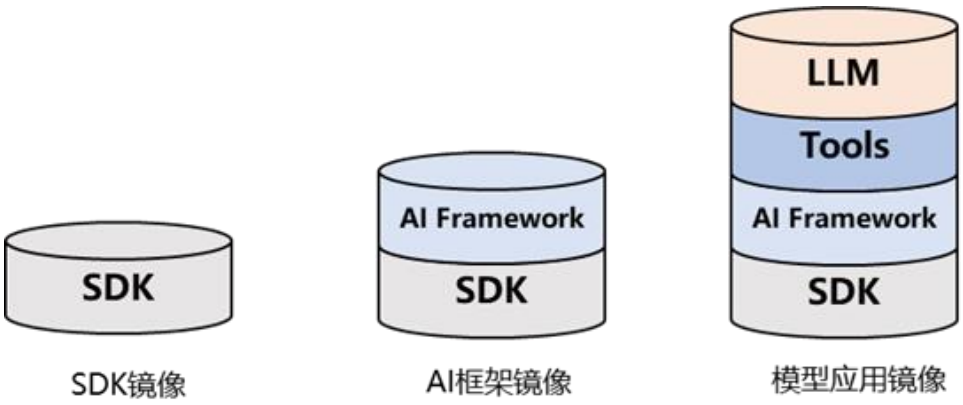
开箱易用

openEuler 兼容 NVIDIA、Ascend 等主流算力平台的软件栈，为用户提供高效的开发运行环境。通过将不同 AI 算力平台的软件栈进行容器化封装，即可简化用户部署过程，提供开箱即用的体验。同时，openEuler 也提供丰富的 AI 框架，方便大家快速在 openEuler 上使用 AI 能力。

功能描述

1. openEuler 已兼容 CANN、CUDA 等硬件 SDK，以及 TensorFlow、PyTorch、MindSpore 等相应的 AI 框架软件，支持 AI 应用在 openEuler 上高效开发与运行。

2. openEuler AI 软件栈容器化封装优化环境部署过程，并面向不同场景提供以下三类容器镜像。



- SDK 镜像：以 openEuler 为基础镜像，安装相应硬件平台的 SDK，如 Ascend 平台的 CANN 或 NVIDIA 的 CUDA 软件。
- AI 框架镜像：以 SDK 镜像为基础，安装 AI 框架软件，如 PyTorch 或 TensorFlow。此外，通过此部分镜像也可快速搭建 AI 分布式场景，如 Ray 等 AI 分布式框架。
- 模型应用镜像：在 AI 框架镜像的基础上，包含完整的工具链和模型应用。

相关使用方式请参考 openEuler AI 容器镜像用户指南。

应用场景

openEuler 使能 AI，向用户提供更多 OS 选择。基于 openEuler 的 AI 容器镜像可以解决开发运行环境部署门槛高的问题，用户根据自身需求选择对应的容器镜像即可一键部署，三类容器镜像的应用场景如下。

- SDK 镜像：提供对应硬件的计算加速工具包和开发环境，用户可进行 Ascend CANN 或 NVIDIA CUDA 等应用的开发和调试。同时，可在该类容器中运行高性能计算任务，例如大规模数据处理、并行计算等。
- AI 框架镜像：用户可直接在该类容器中进行 AI 模型开发、训练及推理等任务。
- 模型应用镜像：已预置完整的 AI 软件栈和特定的模型，用户可根据自身需求选择相应的模型应用镜像来开展模型推理或微调任务。

sysHAX 大语言模型异构协同加速运行时

功能描述

sysHAX 大语言模型异构协同加速运行时专注于单机多卡环境下大模型推理任务的性能提升，针对鲲鹏+xPU（GPU、NPU 等）的异构算力协同，显著提升大模型的吞吐量和并发量：

- 算子下发加速：适配 vllm v0.6.6 版本，进行调度引擎优化，降低 CPU 侧时延。
- CPU 推理加速：通过 NUMA 亲和调度、矩阵运算并行加速、SVE 指令集推理算子适配等方式，提升 CPU 的吞吐量。
- 异构融合调度：支持在 GPU 侧任务满载时，动态将推理请求的 prefill 阶段在 GPU 上执行，decode 阶段放在 CPU 上执行。

应用场景

sysHAX 大语言模型推理优化方案当前支持 DeepSeek、Qwen、baichuan、Llama 等 transformer 架构的模型。其中，CPU 推理加速能力已完成了对 DeepSeek 7B、14B、32B 以及 Qwen2.5 系列模型的适配。主要适用于以下典型场景：

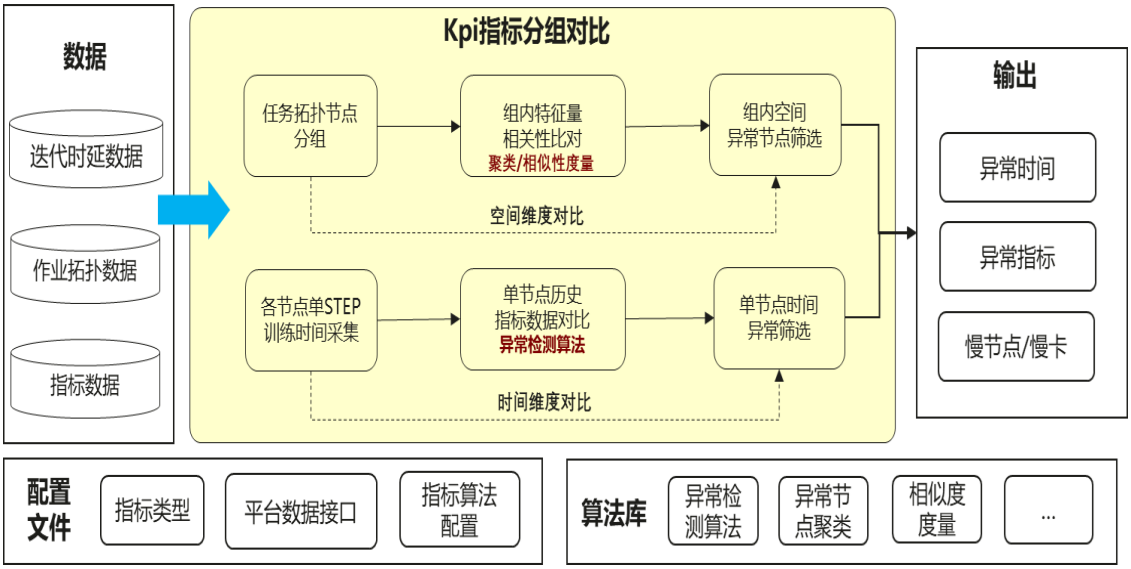
- 数据中心场景：sysHAX 通过上述技术，利用 CPU 填充推理任务，充分利用 CPU 资源，增加大模型并发量与吞吐量。

sysHAX 使用方式请参考 sysHAX 部署指南。

故障分组检测

AI 集群在训练过程中不可避免会发生性能劣化，导致性能劣化的原因很多且复杂。现有方案是在发生性能劣化之后利用日志分析，但是从日志收集到问题定界根因诊断以及现网闭环问题需要长达 3-4 天之久。基于上述痛点问题，我们设计了一套在线慢节点定界方案，该方案能够实时在线观测系统关键指标，并基于模型和数据驱动的算法对观测数据进行实时分析给出劣慢节点的位置，便于系统自愈或者运维人员修复问题。

功能描述



基于分组的指标对比技术提供了 AI 集群训练场景下的的慢节点/慢卡检测能力。这项技术通过 sysTrace 实现，新增内容包括配置文件、算法库、慢节点空间维度对比算法和慢节点时间维度对比，最终输出慢节点异常时间、异常指标以及对应的慢节点/慢卡 ip，从而提高系统的稳定性和可靠性。该特性主要功能如下：

- 配置文件：主要包括待观测指标类型、指标算法配置参数以及数据接口，用于初始化慢节点检测算法。
- 算法库：包括常用的时序异常检测算法 spot 算法，k-sigma 算法，异常节点聚类算法和相似度度量算法。
- 数据：采集到的各个节点的指标数据，以时序序列表示。

- 指标分组对比：包括组内空间异常节点筛选和单节点时间异常筛选。组内空间异常节点筛选根据异常聚类算法输出异常节点；单节点时间异常筛选根据单节点历史数据进行时序异常检测判断节点是否异常。

应用场景

sysTrace 支持慢节点异常检测，告警展示，异常信息落盘。

AI 模型训练场景：适用于 AI 模型大规模集群训练任务，通过该功能可快速发现慢节点，便于系统自愈或者运维人员修复问题。

AI 模型推理场景：适用于单一模型的多实例性能劣化检测，通过多实例间应用资源的对比情况，可快速发现性能劣化的实例，便于推理作业的调度和资源利用率的提升。

异构融合 GMem

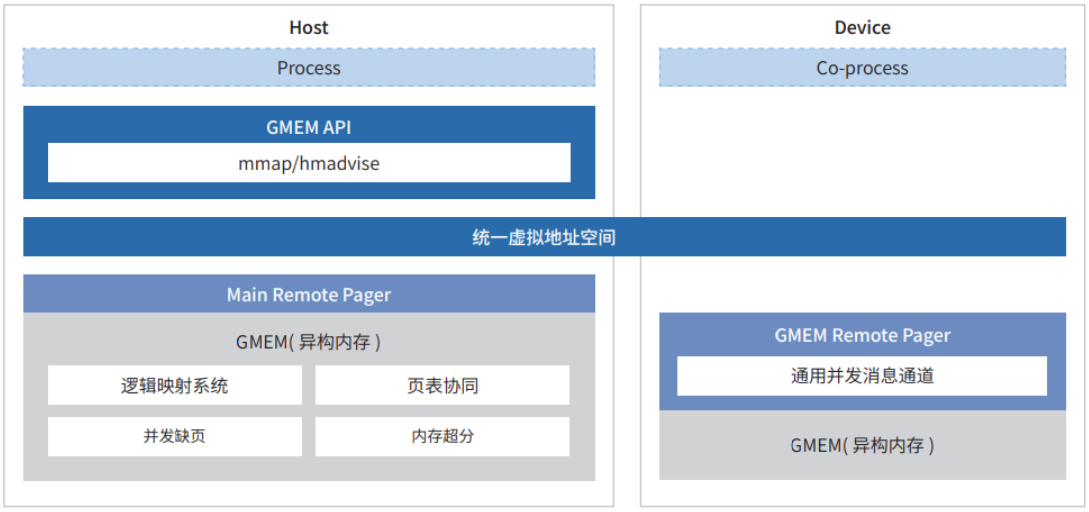
在后摩尔时代，GPU、TPU 和 FPGA 等专用异构加速器设备正不断涌现，它们与 CPU 类似，需要将数据放在本地内存（例如 LPDDR 或 HBM）中以提高计算速度。加速器厂商们也不可避免地需要开发复杂的内存管理系统。 现行加速器内存管理方案存在诸多缺陷：

- CPU 侧内存管理与加速器侧分离，数据显式搬移，加速器内存管理的易用性和性能难以平衡。
- 大模型场景下加速器设备 HBM 内存（Hign BandWidth Memory）严重不足，现有的手动 swap 方案性能损耗大且 通用性差。
- 搜推、大数据场景存在大量无效数据搬移，缺少高效内存池化方案。

Linux 现有的 HMM 框架，编程复杂度高且依赖人工调优，性能和可移植性差，引发 OS 社区反弹，最终导致 HMM 方案搁浅。异构加速器领域亟需高效的统一内存管理机制。异构通用内存管理框架 GMem （Generalized Memory Management），提供了异构内存互联的中心化管理机制，且 GMem API 与 Linux 原生内存管理 API 保持统一，易用性强，性能与可移植性好。加速器使用 GMem API 将内存接入统一地址空间后，可自动获得 GMem 面向异构内存编程优化的能力。与此同时，加速器驱动无需重复实现内存管理框架，大幅降低开发维护带来的成本。开发者使用一套统一申请、释放的 API，即可完成异构内存编程，无需处理内存搬移等细节。在加速器 HBM 内存不足时，GMem 可将 CPU 内存作为加速

器缓存，透明地超分 HBM，无需应用手动 swap。GMem 提供高效免搬移的内存池化方案，当内存池以共享方式接入后，可解决数据反复搬移的痛点。

功能描述



GMem 革新了 Linux 内核中的内存管理架构，其中逻辑映射系统屏蔽了 CPU 和加速器地址访问差异，remote_pager 内存消息交互框架提供了设备接入抽象层。在统一的地址空间下，GMem 可以在数据需要被访问或换页时，自动地迁移数 据到 OS 或加速器端。

- **异构内存特性**

为了结合加速器算力与 CPU 通用算力，实现统一的内存管理和透明内存访问，GMem 设计了统一虚拟内存地址空间机制，将原本的 OS 与加速器并行的两套地址空间合并为统一虚拟地址空间。

GMem 建立了一套新的逻辑页表去维护这个统一虚拟地址空间，通过利用逻辑页表的信息，可以维护不同处理器、不同微架构间多份页表的一致性。基于逻辑页表的访存一致性机制，内存访问时，通过内核缺页流程即可将待访问内存在主机与加速器进行搬移。在实际使用时，加速器可在内存不足时可以借用主机内存，同时回收加速器内的冷内存，达到内存超分的效果，突破模型参数受限于加速器内存的限制，实现低成本的大模型训练。

通过在内核中提供 GMem 高层 API，允许加速器驱动通过注册 GMem 规范所定义的 MMU 函数直接获取内存管理功能， 建立逻辑页表并进行内存超分。逻辑页表将内存管理的高层逻辑与 CPU 的硬件相关层解耦，从而抽象出能让各类加速器复用的高层内存管理逻辑。加速器只需要注册底层函数，不再需要实现任何统一地址空间协同的高层逻辑。

- **remote Pager 内存消息交互框架**

Remote Pager 作为 OS 内核外延的内存管理框架，设计并实现了主机和加速器设备之间协作的消息通道、进程管理、 内存交换和内存预取等模块，由独立驱动 remote_pager.ko 使能。通过 Remote Pager 抽象层可以让第三方加速器很容易地接入 GMEM 系统，简化设备适配难度。

- **用户 API**

用户可以直接使用 OS 的 mmap 分配统一虚拟内存，GMem 在 mmap 系统调用中新增分配统一虚拟内存的标志（MMAP_ PEER_SHARED）。

同时 libgmem 用户态库提供了内存预取语义 hmadvise 接口，协助用户优化加速器内存访问效率（参考 https:// gitee.com/openeuler/libgmem） 。

- **约束限制**

目前仅支持 2M 大页，所以 host OS 以及 NPU 卡内 OS 的透明大页需要默认开启。

通过 MAP_PEER_SHARED 申请的异构内存目前不支持 fork 时继承。

GMem 使用方法可参考以下链接:

https://gitee.com/openeuler/docs-centralized/tree/master/docs/zh/docs/GMEM

应用场景

- 异构统一内存编程

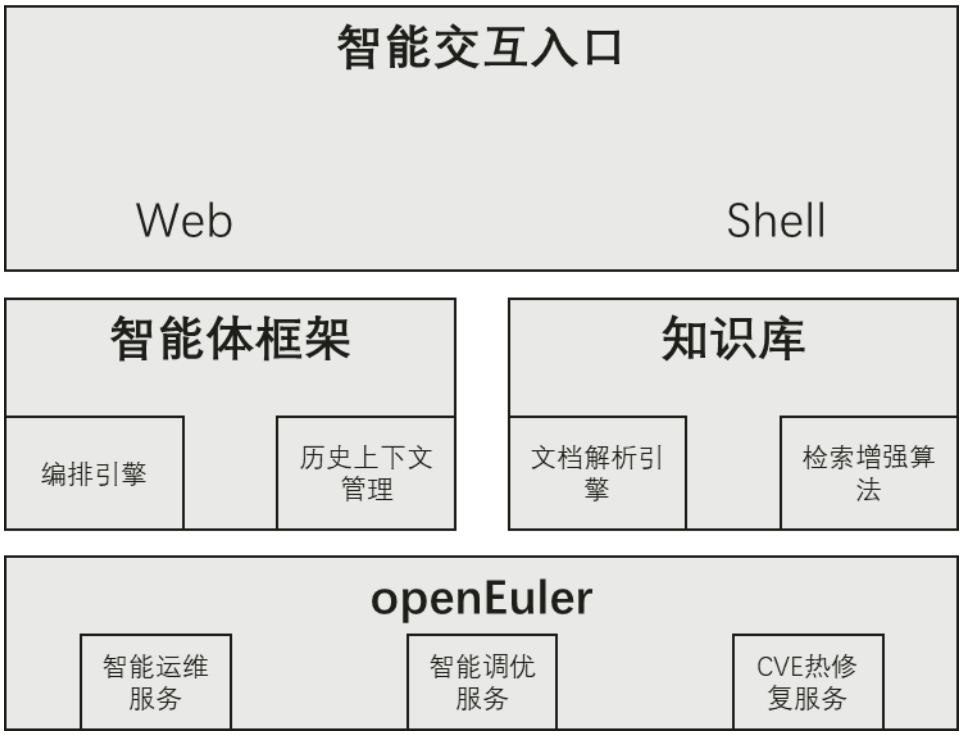
在面向异构内存编程时，使用 GMEM 可分配 CPU 和加速器之间的统一虚拟内存，CPU 内存与加速器内存可共享一个指针，显著降低了异构编程复杂度。当前基于 NPU 试点，驱动仅需百行修改即可接入 GMEM，替换原有约 4000 行内存管理框架代码。

- 加速器内存自动超分

使用 GMEM 接口分配内存时，将不受加速器的物理内存容量所限制，应用可以透明地超分内存（当前上限为 CPU 的 DRAM 容量）。GMEM 将较冷的设备内存页换出到 CPU 内存上，拓展了应用处理的问题规模，实现高性能、低门槛训推。 通过 GMEM 提供的极简异构内存管理框架，在超大模型训练中，GMEM 性能领先 NVIDIA-UVM。随着内存使用量增长，领先比例不断提升，在超分两倍以上时可领先 NVIDIA-UVM 60% 以上（数据基于 NPU-Ascend910 与 GPU-A100 硬件，在 相同 HBM 内存条件下测试）。

AI for OS

当前，openEuler 和 AI 深度结合，一方面，基于 openEuler 操作系统，研发出了 openEuler Intelligence，初步实现基于知识库的智能问答、基于语义接口的工作流编排等功能，在此基础上，openEuler Intelligence 集成了部分系统服务，让 openEuler 更智能。



智能问答

功能描述

openEuler Intelligence 目前支持 Web 和智能 Shell 两个入口。

- Web 入口：用户可以通过 Web 入口以可视化的形式进行知识库的构建和使用、知识库准确率的自动化测试与评估结果获取、openapi 形式的语义接口注册、基于语义接口的工作流应用的构建和使用，Web 入口便携了新手用户对 openEuler 知识的获取和对 openEuler AI 能力的使用。
- 智能 Shell 入口：用户可以通过 Shell 入口借助 openai-api-key 调用智能体框架中的智能问答或者是编排好的工作流，Shell 入口便携了运维人员对操作系统的使用，能通过亲和操作系统的方式与 openEuler 进行智能交互，降低运维成本。

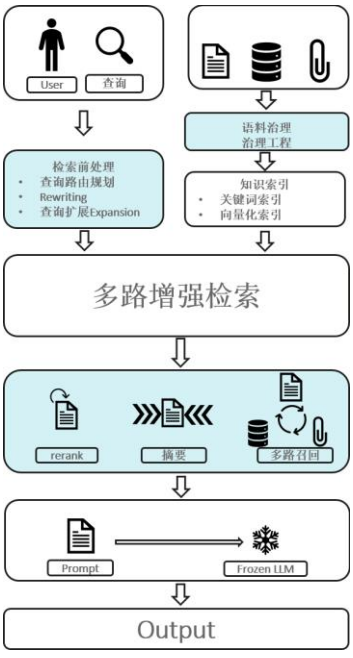
智能调度和推荐

- 智能调度：openEuler Intelligence 支持用户在一个应用中定义多个工作流，基于用户的查询，openEuler Intelligence 会自动的提取参数且选择最为合适的工作流进行工作。
- 智能推荐：openEuler Intelligence 基于用户的查询和工作流的运行结果，推荐用户接下来可能会使用的工作流，增加任务的完成概率，简便应用的使用。

工作流

- 语义接口：语义接口是指含有自然语言注释的接口形式，openEuler Intelligence 提供了两种语义接口注册方式：首先，openEuler Intelligence 允许用户将 api 接口以 openapi（3.0+）yaml 文件形式注册到系统中，用户需要在编写 openapi yaml 文件时，对接口添加自然语言注释，后续在这些接口调用过程中，大模型会根据接口的注释对接口进行选择和填参；其次，openEuler Intelligence 也允许用户将功能以 python 编码的形式注册到 openEuler Intelligence 中，这种形式对于 openEuler Intelligence 而言更为亲和。以上两种形式产生的语义接口最终都可以通过可视化形式编排为工作流。
- 工作流编排及调用：openEuler Intelligence 允许用户将系统提供的语义接口以及用户注册的语义接口以可视化的形式连线成工作流，并支持用户对工作流进行调试且以应用的形式进行发布和使用，工作流在调试和使用过程中会展示中间结果，降低用户调试成本，提升用户交互体验。

RAG



RAG(检索增强技术)是为了增强大模型长期记忆能力和降低大模型训练成本诞生的技术，相较传统 RAG，openEuler Intelligence 中的 RAG 技术在检索前处理、知识索引、检索增强算法和检索后处理方面做了改进：

- 检索前处理：对于信息缺失的用户查询，基于历史上下文和用户意图，提供查询改写的能力，提升检索准确率；
- 知识索引：对于格式和内容多样化的文档，提供摘要、文本特征提取、树形解析、ocr 等文档解析能力，建立片段特征索引，提升检索增强命中率；
- 检索增强算法：对于多样化的检索场景，提供 8 种检索方法，其中基于动态关键字权重的检索方法和基于大模型过滤的检索方法能极大提升开箱及用的准确率。
- 检索后处理：对于上下文缺失的片段，提供均匀随机化上下文补全的方法，增加片段信息完整度，降低最终模型拟合幻想程度；对于 token 阈值上限的片段（集合），提供基于杰卡德距离的 rerank 方法、基于通用词去除及随机丢弃的 token 压缩方法和基于二分的 token 截断方法，允许一些资源受限的场景也能正常进行智能问答。

通过以上能力，相较传统的 RAG 技术，openEuler Intelligence 中的 RAG 技术能适应多种文档格式和内容场景，在不为系统增加较大负担的情况下，增强问答服务体验。

语料治理

语料治理是 openEuler Intelligence 中的 RAG 技术的基础能力之一，其通过上下文位置信息提取、文本摘要和 OCR 增强等方式将语料以合适形态入库，以增强用户查询命中期望文档的概率：

- 上下文位置信息提取：对于文档内容，留存文档相对位置关系，包过片段的全局相对偏移和局部相对偏移，为上下文补全提供基础数据；
- 文本摘要：对复杂文档或者片段，通过滑动窗口+大模型对文本进行摘要，为后续多层次检索的方式提供基础数据；
- OCR 增强：对图文混合的文档，基于图片文字内容+图片上下文进行摘要，为后续针对图片的提问提供基础数据。

自动化测试

自动化测试是 openEuler Intelligence 中的 RAG 技术的基础能力之一，其通过自动化数据集生成和测试评估识别知识库和检索增强算法等配置的不足：

- 数据集生成：用户选择解析完成的文档，基于用户选择的文档，随机选择部分解析结果，并将这些解析结果发送给大模型，让大模型生成及过滤问答对，最终形成含有查询、标准答案和原始片段的高质量测试数据集；
- 测试评估：基于用户生成的数据集和配置的检索增强算法、大模型、topk 片段限制等参数展开测试，最终基于测试集中的查询、标准答案、原始片段、关联到的片段和大模型拟合结果进行打分，最终使用精确率、召回率、忠实值、可解释性、最长公共子串得分、编辑距离得分和杰卡德相似系数对测试结果进行评估。

应用场景

- 面向 openEuler 普通用户：深入了解 openEuler 相关知识和动态数据，比如咨询如何迁移到 openEuler。
- 面向 openEuler 开发者：熟悉 openEuler 开发贡献流程、关键特性、相关项目的开发等知识。
- 面向 openEuler 运维人员：熟悉 openEuler 常见或疑难问题的解决思路和方案、openEuler 系统管理知识和相关命令。

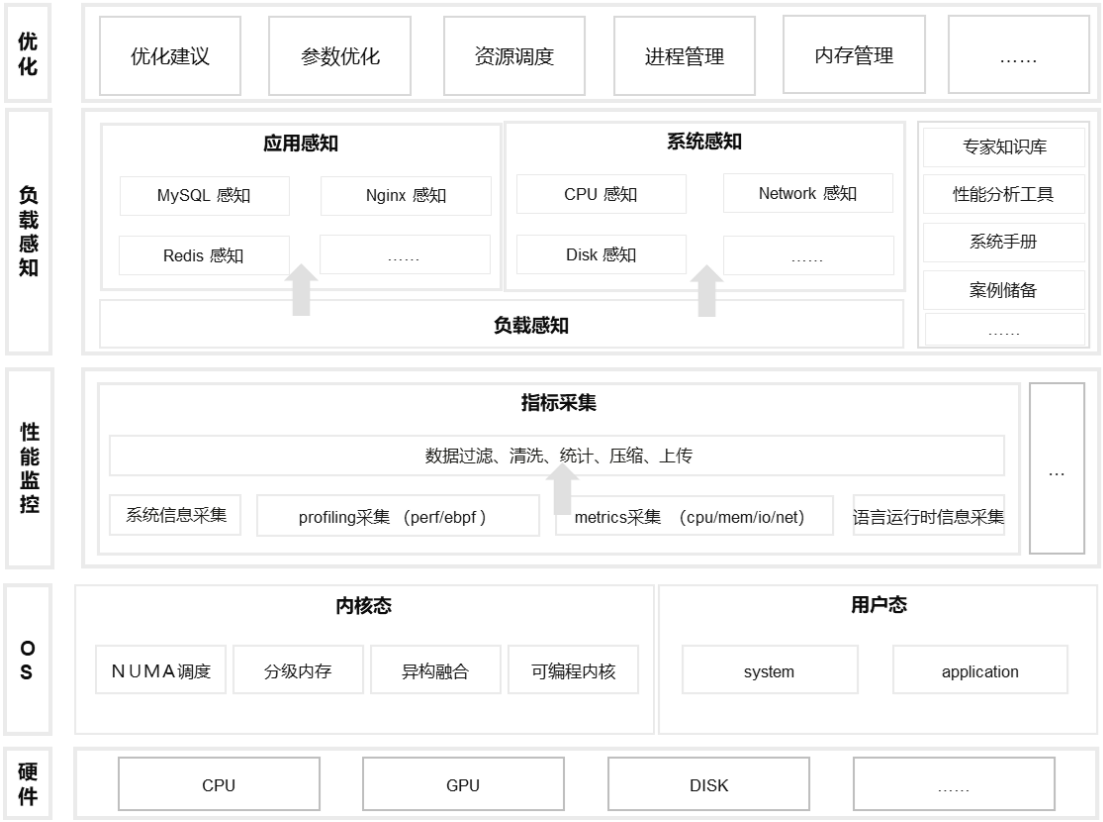
相关使用方式请参考 openEuler Intelligence 智能问答用户指南。

智能调优

功能描述

openEuler Intelligence 智能调优功能目前支持智能 shell 入口。

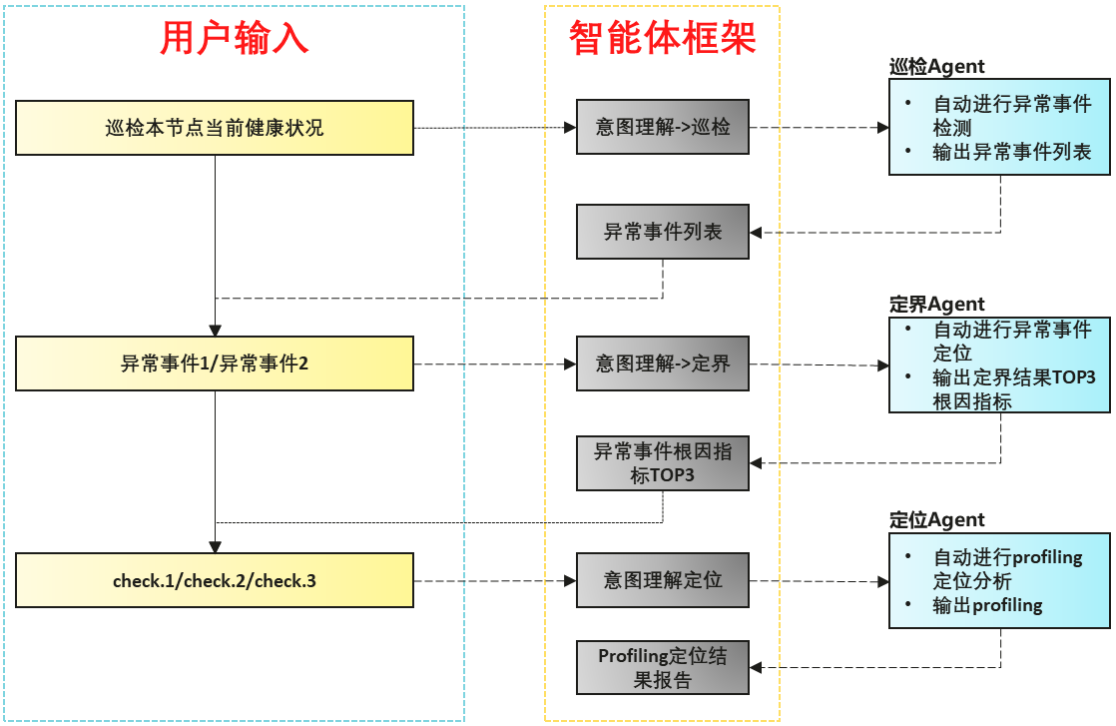
在上述功能入口，用户可通过与 openEuler Intelligence 进行自然语言交互，完成性能数据采集、系统性能分析、系统性能优化等作业，实现启发式调优。



应用场景

- 快速获取系统重要性能指标数据：可快速获取当前系统中 CPU/IO/DISK/NETWORK 等多个重要维度的性能指标以及指定应用的性能指标，帮助用户快速了解系统性能数据。
- 分析系统性能状况：可生成性能分析报告，报告从 CPU/IO/DISK/NETWORK 等多个重要维度分析系统性能状况以及分析指定应用的性能状况，并提示当前系统可能存在的性能瓶颈。
- 推荐系统性能优化建议：可生成一键式执行的性能优化脚本，用户在审核脚本内容后，可执行该脚本，对系统及指定应用的配置进行优化。

智能诊断



功能描述

1. 巡检：调用 Inspection Agent，对指定 IP 进行异常事件检测，为用户提供包含异常容器 ID 以及异常指标（cpu、memory 等）的异常事件列表。
2. 定界：调用 Demarcation Agent，对巡检结果中指定异常事件进行定界分析，输出导致该异常事件的根因指标 TOP3。
3. 定位：调用 Detection Agent，对定界结果中指定根因指标进行 Profiling 定位分析，为用户提供该根因指标异常的热点堆栈、热点系统时间、热点性能指标等信息。

应用场景

智能诊断接口在本次 openEuler 24.03 LTS SP2 版本的功能范围是：具备单机异常事件检测 + 定界 + profiling 定位的能力。

- 其中检测能力指的是：进行单机性能指标采集、性能分析、异常事件检测。
- 其中定界能力指的是：结合异常检测结果进行根因定位，输出 top3 根因指标及其描述。

- 其中 profiling 定位能力指的是：结合 profiling 工具对根因指标进行具体问题模块（代码）定位。

智能容器镜像

功能描述

openEuler Intelligence 目前支持通过自然语言调用环境资源，在本地协助用户基于实际物理资源拉取容器镜像，并且建立适合算力设备调试的开发环境。

当前版本支持三类容器，并且镜像源已同步在 dockerhub 发布，用户可手动拉取运行：

1. SDK 层：仅封装使能 AI 硬件资源的组件库，例如：cuda、cann 等。
2. SDK + 训练/推理框架：在 SDK 层的基础上加装 tensorflow、pytorch 等框架，例如：tensorflow2.15.0-cuda12.2.0、pytorch2.1.0.a1-cann7.0.RC1 等。
3. SDK + 训练/推理框架 + 大模型：在第 2 类容器上选配几个模型进行封装，例如 llama2-7b、chatglm2-13b 等语言模型。

当前支持的容器镜像汇总：

registry	repository	image_name	tag
docker.io	openeuler	cann	8.0.RC1-oe2203sp4
			cann7.0.RC1.alpha002-oe2203sp2
docker.io	openeuler	oneapi-runtime	2024.2.0-oe2403lts
docker.io	openeuler	oneapi-basekit	2024.2.0-oe2403lts
docker.io	openeuler	llm-server	1.0.0-oe2203sp3
docker.io	openeuler	mlflow	2.11.1-oe2203sp3
			2.13.1-oe2203sp3
docker.io	openeuler	llm	chatglm2_6b-pytorch2.1.0.a1-cann7.0.RC1.alpha002-oe2203sp2
			llama2-7b-q8_0-oe2203sp2
			chatglm2-6b-q8_0-oe2203sp2
			fastchat-pytorch2.1.0.a1-cann7.0.RC1.alpha002-oe2203sp2
docker.io	openeuler	tensorflow	tensorflow2.15.0-oe2203sp2
			tensorflow2.15.0-cuda12.2.0-devel-cudnn8.9.5.30-oe2203sp2

docker.io	openeuler	pytorch	pytorch2.1.0-oe2203sp2
			pytorch2.1.0-cuda12.2.0-devel-cudnn8.9.5.30-oe2203sp2
			pytorch2.1.0.a1-cann7.0.RC1.alpha002-oe2203sp2
docker.io	openeuler	cuda	cuda12.2.0-devel-cudnn8.9.5.30-oe2203sp2

应用场景

- 面向 openEuler 普通用户: 简化深度学习开发环境构建流程, 节省物理资源的调用前提, 比如实现在 openEuler 系统上搭建昇腾计算的开发环境。
- 面向 openEuler 开发者：熟悉 openEulerAI 软件栈，减少组件配套试错成本。

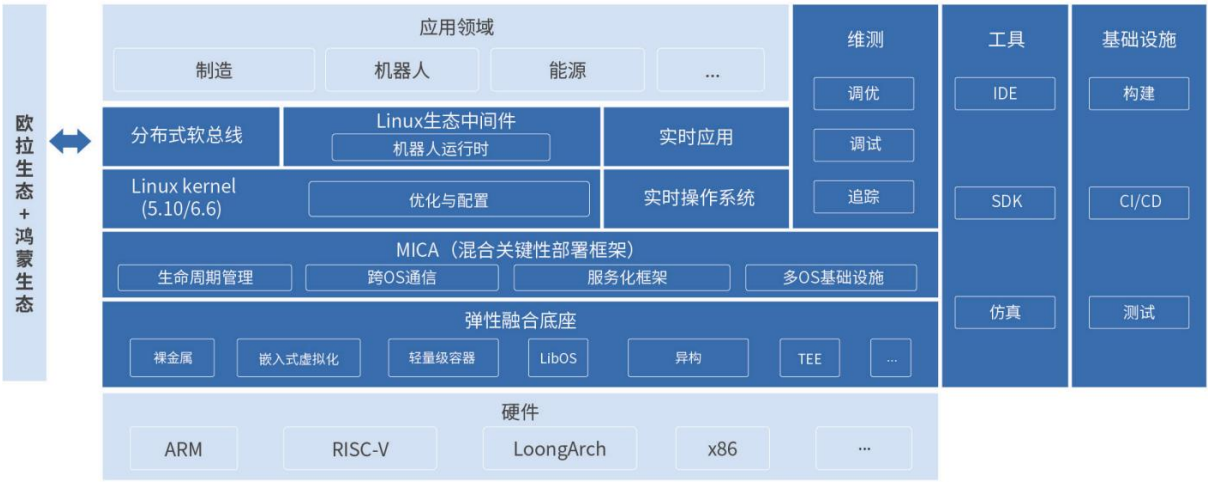
嵌入式场景创新

openEuler 发布面向嵌入式领域的版本 openEuler 24.03 LTS SP2，构建了一个相对完整的综合嵌入系统软件平台，在南北向生态、关键技术特性、基础设施、落地场景等方面都有显著的进步。

openEuler Embedded 24.03 LTS SP2 版本围绕以制造、机器人为代表的 OT 领域持续深耕，通过行业项目垂直打通，不断完善和丰富嵌入式系统软件栈和生态。在软件包生态方面，回合了 oebridge 特性，支持在线一键安装 openEuler 镜像仓软件包，并支持在 Yocto 镜像构建时通过 oebridge 直接安装 openEuler RPM 包快速实现镜像定制。此外，还扩展支持了 oedeploy 特性，能够快速完成 AI 软件栈、云原生软件栈的部署。在内核支持方面，持续完善了 meta-openeuler 的内核配置，配合 oeaware 实时调优功能实现干扰控制以增强系统实时性。

未来 openEuler Embedded 将协同 openEuler 社区生态伙伴、用户、开发者，逐步扩展支持龙芯等新的芯片架构和更多的南向硬件，完善工业中间件、嵌入式 AI、嵌入式边缘、仿真系统等能力，打造综合嵌入式系统软件平台解决方案。

系统架构图



南向生态

openEuler Embedded Linux 当前主要支持 ARM64、x86-64、ARM32、RISC-V 等多种芯片架构，未来计划支持龙芯等架构，从 24.03 版本开始，南向支持大幅改善，已经支持树莓派、海思、瑞芯微、瑞萨、德州仪器、飞腾、赛昉、全志等厂商的芯片。

嵌入式弹性虚拟化底座

openEuler Embedded 的弹性虚拟化底座是为了在多核片上系统（SoC, System On Chip）上实现多个操作系统共同运行的一系列技术的集合，包含了裸金属、嵌入式虚拟化、轻量级容器、LibOS、可信执行环境（TEE）、异构部署等多种实现形态。不同的形态有各自的特点：

1. 裸金属：基于 openAMP 实现裸金属混合部署方案，支持外设分区管理，性能最好，但隔离性和灵活性较差。目前支持 UniProton/Zephyr/RT-Thread 和 openEuler Embedded Linux 混合部署。
2. 分区虚拟化：基于 Jailhouse 实现工业级硬件分区虚拟化方案，性能和隔离性较好，但灵活性较差。目前支持 UniProton/Zephyr/FreeRTOS 和 openEuler Embedded Linux 混合部署，也支持 openHarmony 和 openEuler Embedded Linux 的混合部署。
3. 实时虚拟化：openEuler 社区孵化了嵌入实时虚拟机监控器 ZVM 和基于 rust 语言的 Type-I 型嵌入式虚拟机监控器 Rust-Shyper，可以满足不同场景的需求。

混合关键性部署框架

openEuler Embedded 打造了构建在融合弹性底座之上混合关键性部署框架，并命名为 MICA（MIxed CriticAlity），旨在通过一套统一的框架屏蔽下层弹性底座形态的不同，从而实现 Linux 和其他 OS 运行时便捷地混合部署。依托硬件上的多核能力使得通用的 Linux 和专用的实时操作系统有效互补，从而达到全系统兼具两者的特点，并能够灵活开发、灵活部署。

MICA 的组成主要有四大部分：生命周期管理、跨 OS 通信、服务化框架和多 OS 基础设施。生命周期管理主要负责从 OS（Client OS）的加载、启动、暂停、结束等工作；跨 OS 通信为不同 OS 之间提供一套基于共享内存的高效通信机制；服务化框架是在跨 OS 通信基础之上便于不同 OS 提供各自擅长服务的框架，例如 Linux 提供通用的文件系统、网络服务，实时操作系统提供实时控制、实时计算等服务；多 OS 基础设施是从工程角度为把不同 OS 从工程上有机融合在一起的一系列机制，包括资源表达与分配，统一构建等功能。

混合关键性部署框架当前能力：

1. 支持裸金属模式下 openEuler Embedded Linux 和 RTOS（Zephyr/UniProton）的生命周期管理、跨 OS 通信。
2. 支持分区虚拟化模式下 openEuler Embedded Linux 和 RTOS（FreeRTOS/Zephyr）的生命周期管理、跨 OS 通信。

北向生态

1. 北向软件包支持：700+嵌入式领域常用软件包的构建。
2. 软实时内核：提供软实时能力，软实时中断响应时延微秒级。
3. 分布式软总线基础能力：集成 OpenHarmony 的分布式软总线和 hichain 点对点认证模块，实现 openEuler 嵌入式设备之间互联互通、openEuler 嵌入式设备和 OpenHarmony 设备之间互联互通。
4. 嵌入式容器与边缘：支持 iSula 容器，可以实现在嵌入式上部署 openEuler 或其他操作系统容器，简化应用移植和部署。支持生成嵌入式容器镜像，最小大小可到 5MB，可以部署在其他支持容器的操作系统之上。

UniProton 硬实时系统

UniProton 是一款实时操作系统，具备极致的低时延和灵活的混合关键性部署特性，可以适用于工业控制场景，既支持微控制器 MCU，也支持算力强的多核 CPU。目前关键能力如下：

- 支持 Cortex-M、ARM64、X86_64、riscv64 架构，支持 M4、RK3568、RK3588、X86_64、Hi3093、树莓派 4B、鲲鹏 920、昇腾 310、全志 D1s。
- 支持树莓派 4B、Hi3093、RK3588、X86_64 设备上通过裸金属模式和 openEuler Embedded Linux 混合部署。
- 支持通过 gdb 在 openEuler Embedded Linux 侧远程调试。

应用场景

openEuler Embedded 可广泛应用于工业控制、机器人控制、电力控制、航空航天、汽车及医疗等领域。

5. 内核创新

openEuler 内核中的新特性

openEuler 24.03 LTS SP2 基于 Linux Kernel 6.6 内核构建，在此基础上，同时吸收了社区高版本的有益特性及社区创新特性。

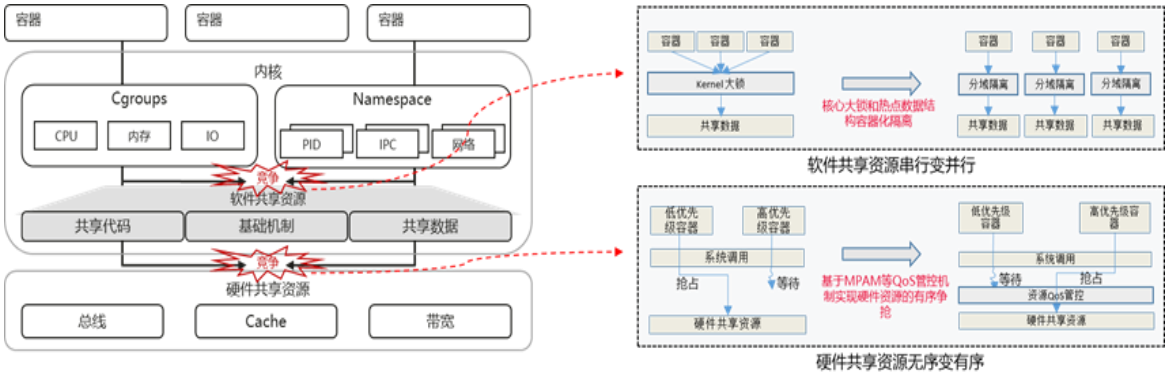
- fuse passthrough 特性支持：当前 fuse 在分布式存储、AI 中广泛使用。在直通场景中 fuse 用户态文件系统没有对读写 IO 进行额外处理，仅仅是记录元数据，向后端文件系统发起 IO。此时 fuse 的处理流程成为了整个系统的 IO 瓶颈。 fuse passthrough 特性旨在 fuse 直接对接后端文件系统（透传）的场景下，消除数据面 fuse 上下文切换、唤醒、数据拷贝开销，允许应用程序直接将读写 IO 在内核中发给后端文件系统，进而大幅提升读写性能。在实验室环境中，fuse passthrough 特性展现出了令人满意的性能提升。具体来说，在 fio 测试中，4K-1MB 粒度的读写测试均有 100%+的性能提升。同时通过了故障注入测试和稳定性测试。业务可以按需使用。

- MPAM 增强特性支持：新增 QoS 增强特性，拓展内存带宽和 L3 缓存控制方式，可按照使用量上限/保底/优先级方式配置，为混部场景动态调控共享资源提供端到端能力。新增 IO QoS 管控特性，联动 SMMU 对外围硬件设备或异构加速器的 IO 带宽流量进行隔离配置，支持 iommu_group 粒度级别监控，为异构场景下 IO QoS 管控方案提供控制侧新方案。此外，新增 L2 缓存隔离配置，提供 L2C 占用量和带宽流量监控能力，为混部场景下系统性能提供物理核级别的优化和分析手段。以上 MPAM 特性在业务实测场景展现出明显的性能提升，其中在混部场景下，Specjbb 作为在线业务的混部干扰率从 25.5%降低至 5%以下。

6. 云化基座

众核高密容器隔离

服务器芯片由多核进入众核时代（>256C），对操作系统提出新的挑战。提升 Rack 计算密度、降低数据中心 TCO，众核服务器已成为互联网行业主流选择，随着云技术和业务规模发展，容器化部署成为互联网行业的主流业务部署形态，在这种场景下，系统串行开销和同步开销限制可扩展性，干扰问题凸显，资源利用率低，影响容器部署扩展性的串行访问开销和同步开销主要来自软硬共享资源争用。



功能描述

本期主要采用轻量虚拟化按 NUMA 分域拆分资源、域内实现资源容器级隔离增强，降低因软硬件资源争用导致的性能干扰，提升容器部署扩展性。关键技术特性如下：

- 轻量虚拟化：虚拟设备中断卸载实现虚拟 virtio 设备中断卸载至硬件注入，降低存储虚拟化损耗；PMD 队列负载均衡实现 virtqueue 队列从单 reactor 均衡分散到多 reactor，消除 cpu 瓶颈；
- CPU 分域调度：CPU 基于硬件拓扑划分子域部署容器，一个容器一个独立子调度域，实现容器之间干扰隔离，降低跨 cluster cache 同步次数和 cache/NUMA 内存等硬件资源争抢，减轻容器之间的相互干扰。对于 redis 多并发场景性能提升 10%+；
- 文件系统块分配干扰隔离：优化 ext4 块分配释放流程中的 group lock 和 s_md_lock 两个主要争抢的锁，以提高 EXT4 块分配流程的可扩展性。通过允许在当前目标块组被占用时尝试使用其他空闲块组进行分配，从而减少了多个容器争抢同一个块组造成的 CPU 浪费，并充分利用了 ext4 多块组的优势，缓解了 group lock 的竞争。其次通过将流式配的全局目标拆分到 inode 级别，从而减少了全局锁 s_md_lock 的竞争文件数据也更加聚集。在 64 容器并发场景下，块分配和块释放混合场景 OPS 提升 5 倍以上，单块分配场景提升 10 倍以上；
- 网络 tcp hash 干扰隔离：tcp_hashinfo bash、ehash 存在锁竞争，ehash 计算频繁，导致高并发下带宽下降，时延变大。将 tcp_hashinfo bash、ehash 的自旋锁改为 rcu，ehash 计算方式改为 lport 递增减少查询时间和计算次数，减少 tcp connect hash 的锁竞争；
- Cgroup 隔离增强：user namespace 通过 percpu counter 替换原来的原子操作，避免不同 namespace 相同父节点竞争访问，消除容器间 rlimit 计数干扰。 解决 will-it-scale/signal1 用例线性度问题，64 个容器并发吞吐性能提升 2 倍。通过对 memcg 实现批量释放处理，避免大量的小内存释放对于相同父节点计数竞争，提升内存计数的可扩展性，tlb-flush2 测试用例 64 容器吞吐提升 1.5 倍；基于 eBPF 可编程内核能力，提供主机容器信息隔离与过滤机制，高效实现容器资源视图。相较业界 LXCFS 方案，本方案避免了内核态-用户态切换开销，消除了 LXCFS 进程的性能与可靠性瓶颈，单容器内资源视图吞吐量在单容器场景下性能提升 1 倍，在 64 容器场景下提升 10 倍；
- 干扰监测：干扰监测回答的是容器有没有被干扰、是什么干扰、干扰程度如何这三个问题，从结果上看干扰可以分为干扰导致指令得不到执行、干扰导致指令执行变慢和干扰导致指令执行变多三类，干扰监测从内核角度，针对每一类的典型干扰在运行时进行统计，当前支持在线统计 schedule latency、throttling、softirq、hardirq、spinlock、mutex 和 smt 干扰，性能开销在 5%以内；

- 鲲鹏内存/Cache QoS 管控机制 MPAM：内存带宽流量和各级缓存占用量，可按照使用量上限/保底/优先级方式进行配置，根据不同业务，以线程为粒度部署不同隔离策略。支持业务资源实时监控，在客户业务层面和线程级别，实时对共享资源的使用情况进行跟踪监控，将资源使用情况反馈给控制策略，形成闭环控制效果。此外，MPAM 联动 SMMU 扩展外设 IO QoS 方案，支持对外围设备和异构加速器 IO 带宽流量进行隔离配置，按设备粒度级别进行资源监控。

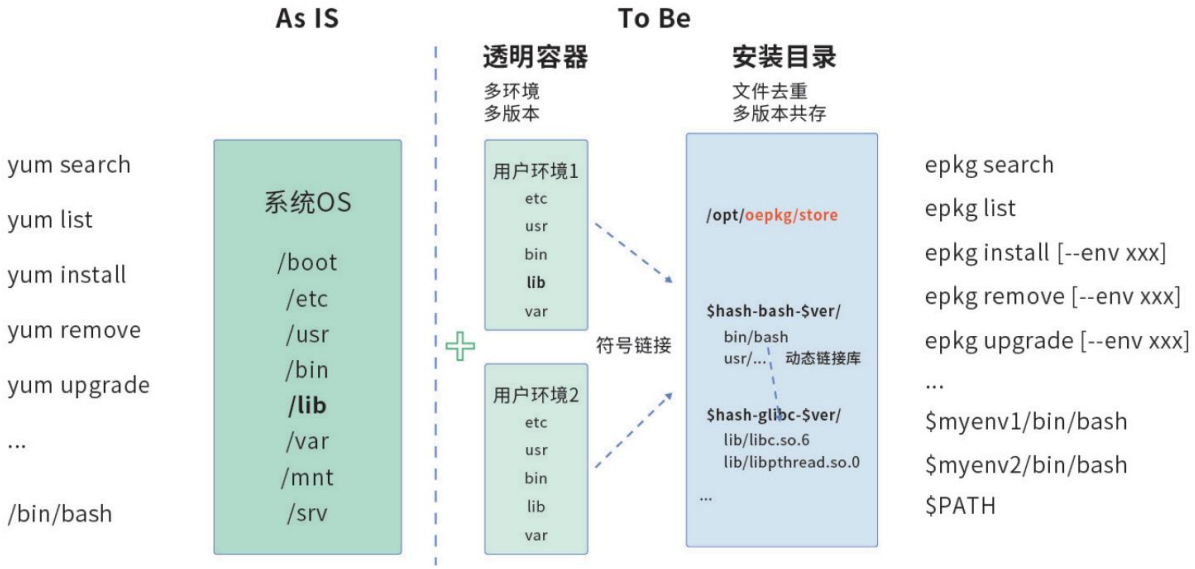
应用场景

众核服务器场景下，业务容器高密度部署场景，通过降低容器间干扰，提升容器部署密度，进而提升资源利用率。

7. 特性增强

epkg 新型软件包

epkg 是一款新型软件包，支持普通用户在操作系统中安装及使用。新的软件包格式相比现有软件包，主要解决多版本兼容性问题，用户可以在一个操作系统上通过简单地命令行安装不同版本的软件包。同时支持环境环境实现环境的创建/切换/使能等操作，来使用不同版本的软件包。目前 epkg 主要支持非服务类的软件包的安装和使用。



功能描述

多版本兼容：支持普通用户安装，支持安装不同版本的软件包，不同版本的同一软件包安装不冲突。使能用户在同一个节点上，快速安装同一软件包的不同版本，实现多版本软件包的共存。

环境管理：支持环境环境实现环境的创建/切换/使能等操作，用户通过环境的切换，在环境中使用不同的 channel，实现在不同的环境中使用不同版本的软件包。用户可以基于环境，快速实现软件包版本的切换。

普通用户安装：epkg 支持普通用户安装软件包，普通用户能够自行创建环境，对个人用户下的环境镜像管理，无需特权版本。降低软件包安装引起的安全问题。

应用场景

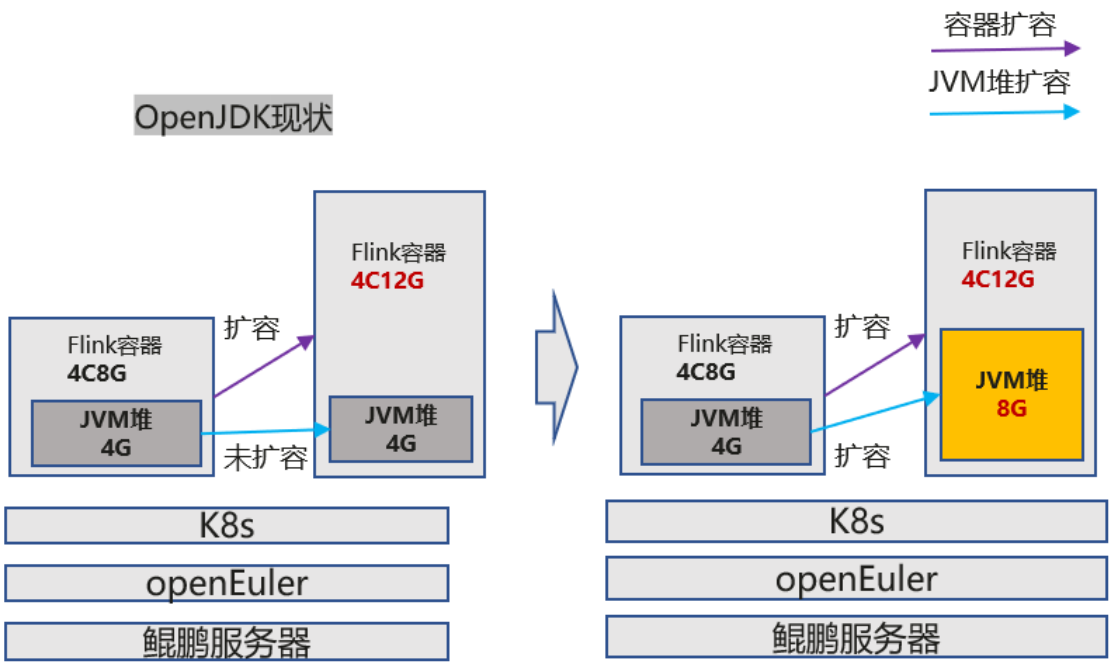
适用于用户希望安装软件包的多个版本的场景，用户能够通过切换环境使用不同的版本的软件包，解决兼容性难题。

使用文档参考：epkg 使用文档。

毕昇 JDK8 支持堆内存扩容

功能描述

互联网容器化部署应用的模式下，大部分客户容器场景下容器资源支持垂直伸缩，当前 OpenJDK8 的最大堆只能在启动时支持修改，无法支持在线动态扩缩，java 应用无法在线使用到容器扩容出的内存，需要 java 应用启动时重新设置最大堆；鉴于此问题，毕昇 JDK8 在 G1GC 实现堆内存上限在线伸缩能力，允许用户在应用运行时动态更新 Java 堆内存的上限，而无需重启 JVM。

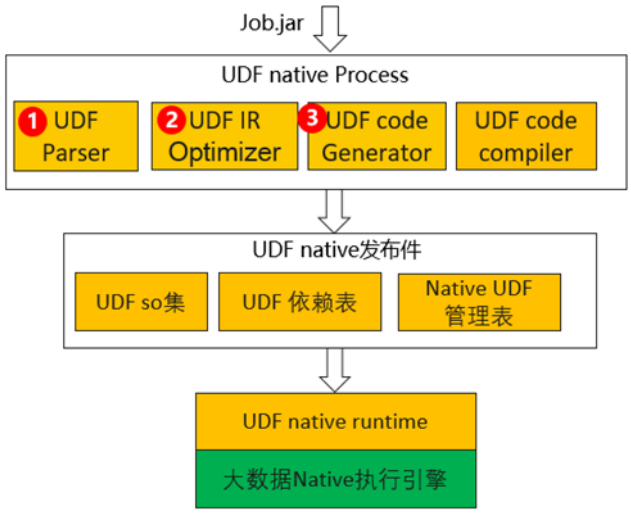


应用场景

互联网等容器场景业务，在容器在线扩容后需要 java 业务的堆内存大小也支持在线扩容的场景。

编译器 UDF 自动 native 框架

功能描述



针对开源大数据 JVM 执行效率低的缺点，UDF 自动 native 框架负责将 Java UDF 自动转换为 C/C++ Native UDF，并进一步从内存高效管理、硬件亲和等维度提升大数据处理性能。UDF 自动 native 框架致力于实现用户无感知、全自动的 Java UDF native 加速机制。UDF 自动 native 框架主要由 UDF parser、UDF IR Optimizer、UDF code Generator、UDF code compiler 等模块组成：

UDF parser 将业务 jar 包字节码自动转换为 IR 代码，并基于 UDF 特征提取出 UDF 代码；UDF IR Optimizer 从内存对象自动管理、硬件亲和加速等维度对 UDF IR 进行优化；UDF Code Generator 将 UDF IR 对等转换为 native 代码；UDF code compiler 将 UDF native 代码在线编译为 native 二进制。最后，UDF 的 native 二进制发布到大数据执行节点上，由大数据系统 native 执行引擎动态加载执行，提升大数据系统处理性能。

应用场景

当前版本的 UDF 自动 native 框架适用于对接 Flink 大数据 native 执行引擎，通过配套 Flink Datastream 场景 native 基础库，可以在用户无感知的情况下实现 Flink Datastream UDF 自动 native 加速。

AI 编译器

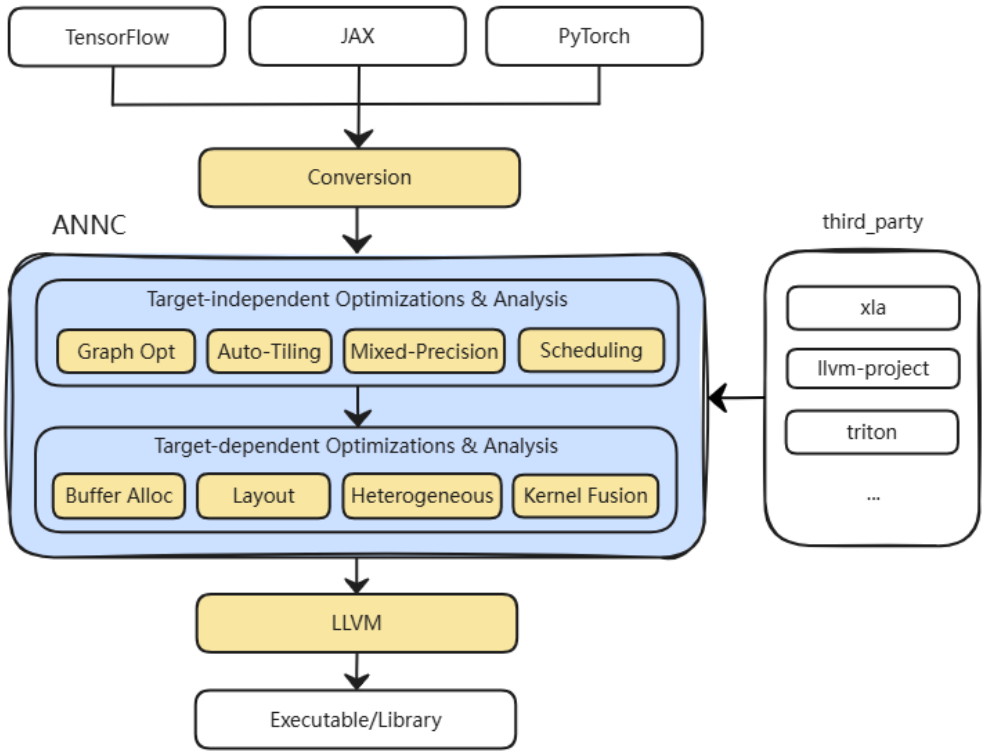
ANNC（Accelerated Neural Network Compiler）是专注于加速神经网络计算的 AI 编译器，聚焦于通过计算图优化，高性能融合算子生成和高效的代码生成和优化能力，加速推荐和大语言等模型的推理性能，并且从架构设计角度支持业界主流开源推理框架和不同硬件后端的接入，提高软件的可扩展性。

功能描述

计算图优化是通过优化神经网络的计算流程，从算法角度减少冗余操作、混合精度改写和自动子图调度优化计算负载和提高缓存利用率，从硬件架构角度优化张量数据布局、算子融合和转换、子图切分调度，进一步优化负载，充分利用硬件资源。

高性能融合算子库生成和对接包括前端计算图模式识别和转换，高性能算子库查询和对接，以及算子库自动生成三部分，从汇编指令层面上通过数据预取、模型并行和新型指令集应用等优化手段，减少访存，提高并行计算效率。

ANNC 旨在通过图编译优化和高性能算子生成和对接，提高 AI 推理速度和降低功耗，达到提高用户单位成本的推理效率的目的，同时通过软件兼容性和易用性设计减少用户的运营成本和环境影响。



应用场景

当前版本的 AI 编译器主要聚焦于图编译优化和算子库选择调用，面向主流搜索推荐系统中的粗排、精排模型能够取得较大收益，尤其是面向特征处理复杂的嵌入层网络，图编译优化能够取得更好的效果。

同时，ANNC 也预留了大语言模型等未来场景的性能优化途径，对接 LLM 推理框架，结合现有的图算融合、内存布局优化技术，以及 GEMM 等核心的性能优化手段，减小推理时延，提高推理吞吐。

oeDeploy 特性增强

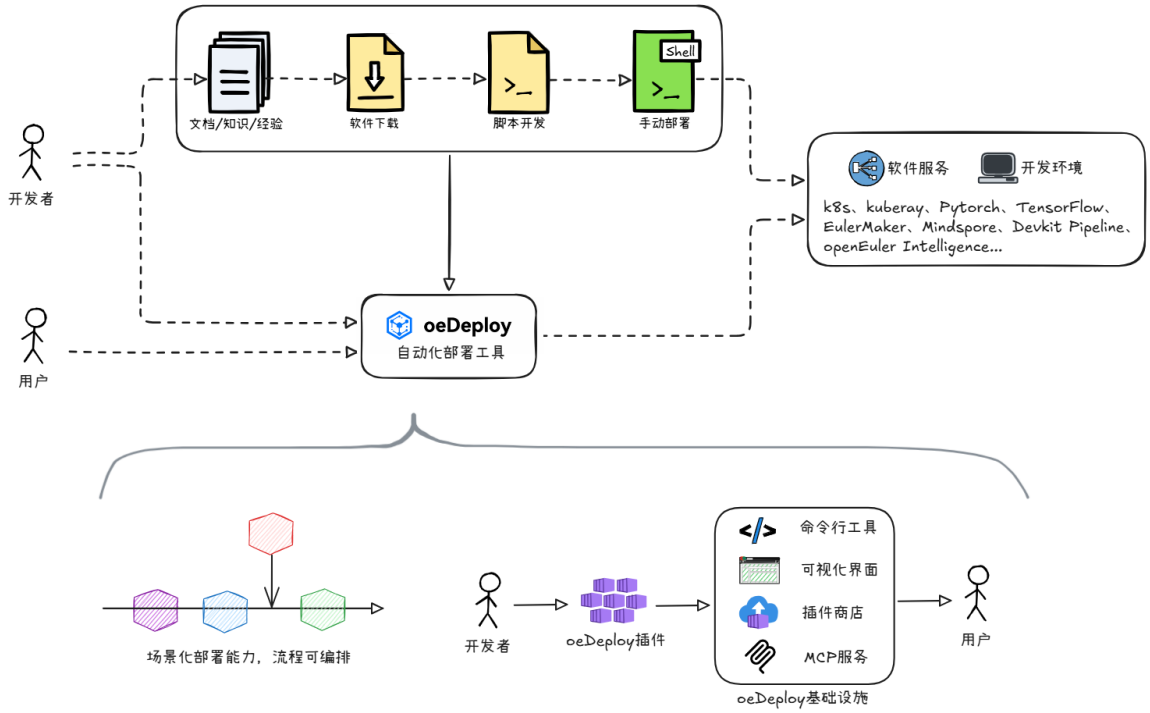
oeDeploy 是一款轻量级的软件部署工具，旨在帮助开发者快速、高效地完成各类软件环境部署，对单节点与分布式场景均可适配。

功能描述

多场景支持 & 主流软件一键部署：支持单节点应用与集群软件环境的一键部署，新版本 oeDeploy 增加了对多 master 节点的 Kubernetes 环境的快速部署，新增支持了 openEuler Intelligence、Devkit Pipeline 等社区工具链，以及 RAGFlow、anythingllm、Dify 等主流 RAG 软件。

灵活的插件化管理 & 优秀的部署体验：oeDeploy 提供可扩展的插件架构，灵活管理多种部署能力，开发者也可以快速发布自定义部署插件。新版本 oeDeploy 支持了插件源的管理，支持一键更新插件版本、一键完成插件初始化。oeDeploy 支持极简的命令行操作方式，也即将上线可视化工具与插件商店，用更少的代码，实现更高效的软件部署体验。

高效部署 & 智能开发：新版本 oeDeploy 发布了 MCP 服务，在 DevStation 中实现开箱即用，借助大模型的推理能力，支持用自然语言完成各类软件的一键部署，部署效率提升 2 倍；支持将用户文档快速转换成可以直接运行的 oeDeploy 插件，开发效率提升 5 倍。



应用场景

ISV 与开发团队可以将 oeDeploy 作为向客户交付软件产品的统一形式。借助 oeDeploy 提供的命令行工具与插件框架，开发团队只需较少的开发工作量就可以实现优秀的部署效果，降低用户的额外学习成本，提高客户满意度。

面向开发者与维护人员，oeDeploy 提供了复杂软件环境的快速部署能力，可以在几分钟内部主流的 AI 训推框架，大幅降低软件开发门槛，避免了重复操作。同时，开发者可以基于 oeDeploy 发布自定义的部署能力，让更多用户受益于一键部署的便利。借助大模型与 MCP 的能力，oeDeploy 可以让部署流程更加高效，开发过程更加智能。

DevStation 特性增强

DevStation 是基于 openEuler 的智能 LiveCD 桌面版开发者工作站，专为极客与创新者而生。旨在提供开箱即用、高效安全的开发环境，打通从部署、编码、编译、构建到发布的全流程。它融合了一键式运行环境与全栈开发工具链，支持从系统启动到代码落地的无缝衔接。无需复杂安装，即可体验开箱即用的开发环境，通过新增 MCP AI 智能引擎，快速完成社区工具链调用，实现从基础设施搭建到应用开发的效率飞跃。

功能描述

开发者友好的集成环境: 发行版预装了广泛的开发工具和 IDE, 如 VS Codium 系列等。支持多种编程语言，满足从前端、后端到全栈开发的需求。

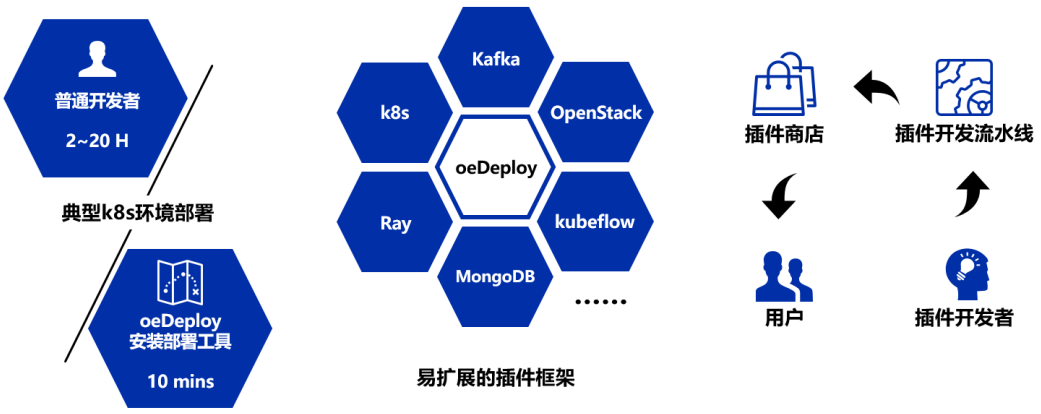
社区原生工具生态：新增 oeDeploy（一键式部署工具）、epkg（扩展软件包管理器）、devkit 和 openEuler Intelligence，实现从环境配置到代码落地的全链路支持。 oeDevPlugin 插件+oeGitExt 命令行工具支持：专为 openEuler 社区开发者设计的 VSCodium 插件，提供 Issue/PR 可视化管理面板，支持快速拉取社区代码仓、提交 PR，并实时同步社区任务状态。 openEuler Intelligence 智能助手：支持自然语言生成代码片段、一键生成 API 文档及 Linux 命令解释。

图形化编程环境：集成了图形化编程工具，降低了新手的编程门槛，同时也为高级开发者提供了可视化编程的强大功能，预装 Thunderbird 等办公效率工具。

MCP 智能应用生态构建：Devstation 深度集成 Model Context Protocol (MCP) 框架，构建完整的智能工具链生态，预装 MCP 智能工具链，支持 oeGitExt、rpm-builder 等核心 MCP Server，提供社区事务管理、RPM 打包等能力，将传统开发工具（如 Git、RPM 构建器）通过 MCP 协议进行智能化封装，提供自然语言交互接口。

系统部署与兼容性增强：广泛的硬件支持，特别优化对主流笔记本/PC 硬件的兼容性（触摸板、Wi-Fi 、蓝牙），重构内核构建脚本（kernel-extra-modules），确保裸机部署体验。灵活部署形态，支持 LiveCD（一键运行无需安装）、裸机安装、虚拟机部署。

应用场景



多语言开发环境：适用于需要同时开发多语言（如 Python、JavaScript、Java、C++）项目的开发者，无需手动配置环境，系统预装各种编译器、解释器和构建工具。

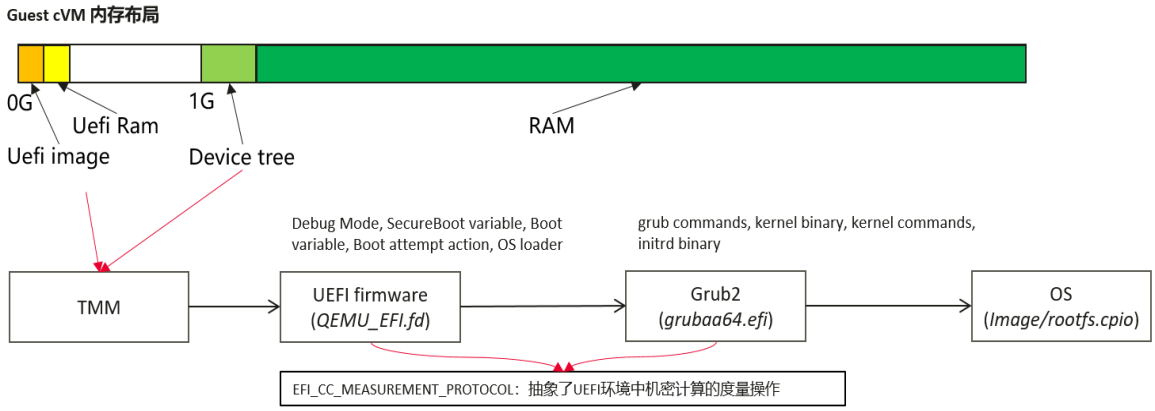
快速安装部署平台：Devstation 集成了 oeDeploy，可以对 kubeflow、k8s 等分布式软件实现分钟级部署，大幅减少开发者部署时间。oeDeploy 同时提供了统一的插件框架与原子化的部署能力，开发者只需遵循简单的开发规范，就可以快速发布自定义的安装部署插件，帮助更多用户解决安装部署问题。

面向测试/南向兼容性开发人员，提供硬件兼容性保障（支持主流笔记本/服务器），支持裸机部署测试驱动兼容性。

提升开发者效率：MCP RPM-builder 工具链落地，提升 MCP 易用性，支持 mcp servers 自动化打包构建 rpm 包并发布到社区，确保 mcp 一键安装功能 100%可用，构建完整的 MCP 智能应用生态 REPO，覆盖部署、测试、性能调优等应用场景，包括：查询分配的社区 issue，创建 PR 提交代码变更，通过 CI/CD 自动构建验证。

virtCCA 特性增强

功能描述



当前 virtCCA 架构在启动方式上存在特定约束：其仅支持 **kernel 与 rootfs 分离的启动模式**（即内核镜像与根文件系统分别挂载）。然而，在主流云平台环境中，虚拟机的启动流程普遍依赖 **GRUB 引导机制**，这要求将 UEFI 固件（如 EDK2）、内核（Kernel）及初始内存文件系统（initramfs）整合至单一磁盘镜像（如 QCOW2 格式）中。功能要点包括：

1. 单镜像封装

(1) 将 EDK2 固件、GRUB 引导程序、内核（Kernel）及 initramfs 整合至单一 QCOW2 磁盘镜像，形成完整启动栈。

(2) GRUB 通过配置文件（grub.cfg）定位内核路径，要求内核与 initramfs 必须位于同一文件系统（如 EXT4/XFS）。

2. 安全信任链传递

(1) Secure Boot 机制：EDK2 验证 GRUB 及内核的数字签名，确保启动组件未被篡改。

(2) 硬件资源协同：依赖 UEFI 运行时服务枚举硬件设备，为虚拟机管理程序（如 KVM）提供虚拟化资源池

3. 云原生优化

(1) 支持快照克隆、根文件系统动态扩容（依赖 initramfs 中的 cloud-init 工具）特性。

应用场景

云环境中采用 UEFI（Unified Extensible Firmware Interface）启动模式，已成为现代虚拟化架构的核心技术，virtCCA 架构支持 UEFI 启动，扩展了机密虚机的应用场景：

1. 快速实例启动与弹性伸缩

(1) UEFI 通过**并行硬件初始化**（如 CPU、内存、存储设备同步检测）显著缩短启动时间。

2. 大容量磁盘支持

(1) UEFI 依赖 **GPT 分区表**，突破传统 MBR 的 2TB 磁盘限制，支持**百 TB 级云盘**（如阿里云 ESSD 云盘），满足大数据存储（如 HDFS）、AI 训练等场景需求。

3. 自动化运维与批量部署

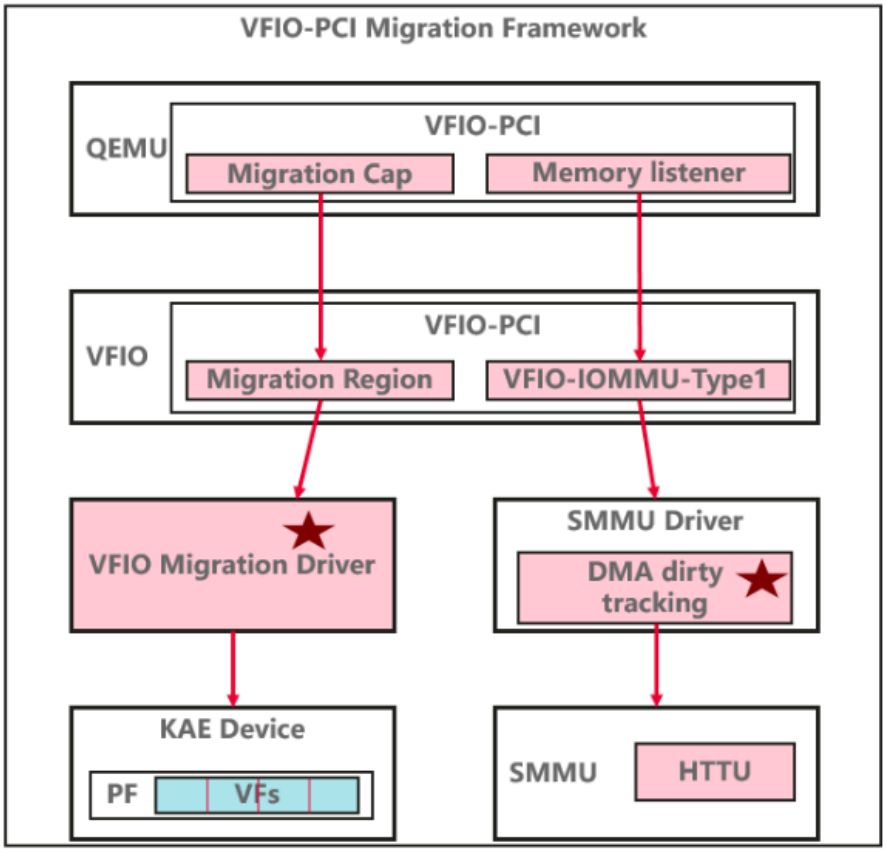
(1) **镜像标准化**：云服务商提供的 UEFI 兼容镜像默认启用 GPT 分区，简化用户部署流程。

虚拟化支持 vKAE 直通设备热迁移

功能描述

KAE 是基于鲲鹏 920 新型号处理器提供的硬件加速解决方案，包括 HPRE、SEC、ZIP 设备，可用于加解密和压缩解压缩，能够显著降低处理器消耗，提高处理器效率。vKAE 直通热迁移是指虚拟机在配置 KAE 直通设备时，进行热迁移的能力，可以为 KAE 设备的使用提供更强的灵活性和业务不中断的保障。

smmu 脏页跟踪是实现直通设备高效、可靠的热迁移的关键技术。在 ARM 架构中，通过纯软件方式进行脏页跟踪，会带来了较大的性能损耗。HTTU(Hardware Translation Table Udate)允许硬件自动更新 smmu 页表状态，在进行写操作时会自动置对应页表项的写权限位，热迁移时扫描页表的写权限位进行脏页统计。



应用场景

为虚拟机中使用 KAE 直通设备的场景提供热迁移支持，适用于对数据安全性和处理性能有较高要求的领域，如金融、云计算和大数据处理等，有效增强业务连续性与运行稳定性。

Kuasar 机密容器

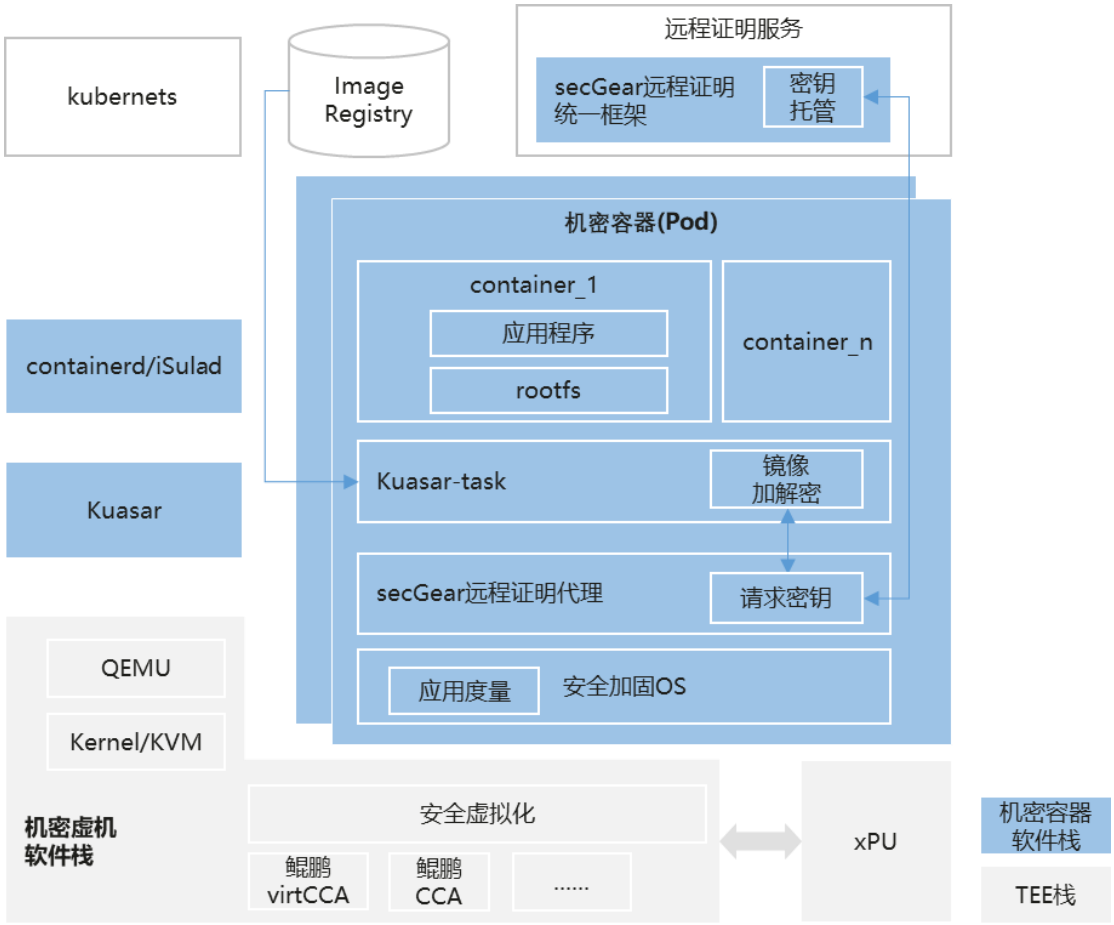
功能描述

Kuasar 统一容器运行时在支持安全容器的基础上添加了对机密容器的支持。用户可以通过配置 iSulad 的运行时参数，完成对 Kuasar 机密容器的纳管。

支持功能特性:

- 支持 iSulad 容器引擎对接 Kuasar 机密容器运行时，兼容 Kubernetes 云原生生态。

- 支持基于 virtCCA 的机密硬件，允许用户在鲲鹏 virtCCA 可信执行环境中部署机密容器。
- 支持 secGear 远程证明统一框架，遵循 RFC9334 RATS 标准架构，允许在机密计算环境中运行的容器向外部的受信任服务证明其可信性。
- 支持在机密容器内部拉取并解密容器镜像，保护容器镜像的机密性和完整性。



应用场景

Kuasar 机密容器在满足客户数据安全的诉求下，兼容云原生生态，确保用户的机密应用具备高可用性、弹性伸缩、快速交付 等云原生优势，在 AI 保护、数据可信流通、隐私保护等机密计算的业务中有广泛的应用场景。

Global Trust Authority 远程证明

功能描述

GTA 远程证明服务组件支持 TPM/vTPM 的远程证明，分为客户端和服务端。

- 服务端提供了远程证明服务框架兼容可信计算及机密计算，支持证书、策略等的增删改查，Quote 验证，随机数，JWT Token 生成等能力。
- 客户端支持采集本地 TPM 证据，并可与服务端交互，验证 Quote。

本组件在安全性及易用性上也提供了多种能力。

安全性上支持数据库完整性保护、数据链路加密，验证防重放，SQL 防注入，用户隔离，密钥轮换机制等一系列差异化安全竞争力。

易用性上支持护照模式及背调模式。客户端支持定时上报，响应挑战等多种验证模式。客户端及服务端支持 rpm 包及 docker 安装部署。

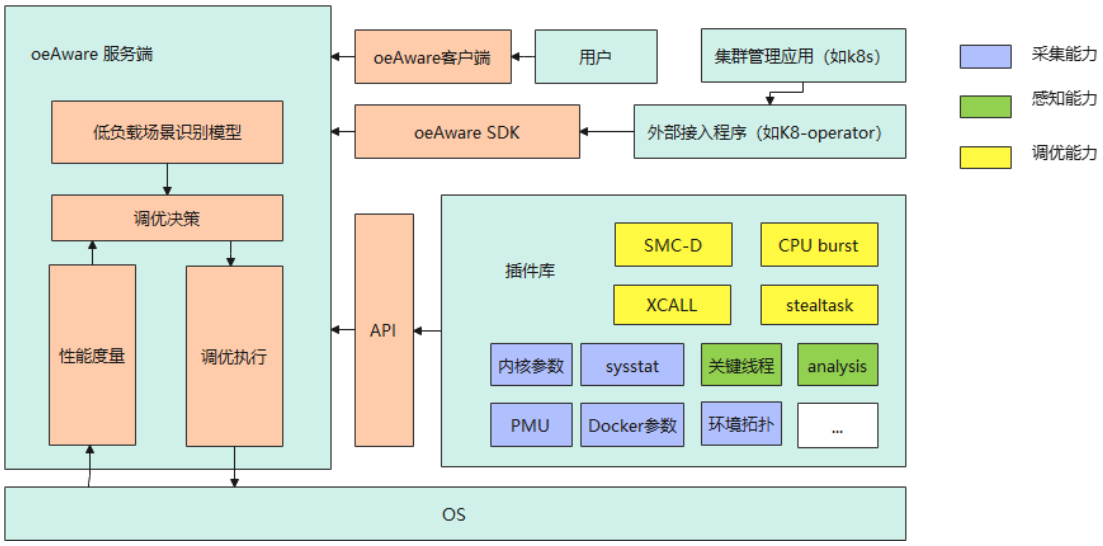
后续版本也将支持 virtCCA 机密计算验证能力。

应用场景

远程证明是开启机密计算及可信计算的前置条件。只有当运行环境在密码学意义上被严格证明安全可信后方可进行后续运算。所有涉及机密计算的端到端解决方案都应将远程证明作为整体解决方案中的核心一环，一旦运行环境被证明不可信，则应立即中止后续任务。在 AI 模型保护，用户隐私保护，密钥管理等多场景均有广泛应用。

oeAware 采集、调优插件等功能增强

oeAware 是在 openEuler 上实现低负载采集感知调优的框架，目标是动态感知系统行为后智能使能系统的调优特性。传统调优特性都以独立运行且静态打开关闭为主，oeAware 将调优拆分为采集、感知和调优三层，每层通过订阅方式关联，各层采用插件式开发尽可能复用。



功能描述

oeAware 的每个插件都是按 oeAware 标准接口开发的动态库，包含若干个实例，每个实例可以是一个独立的采集、感知或调优功能集，每个实例包含若干个 topic，其中 topic 主要用于提供采集或者感知的数据结果，这些数据结果可供其他插件或者外部应用进行调优或分析。

- SDK 提供的接口可以实现订阅插件的 topic，回调函数接收 oeAware 的数据，外部应用可以通过 SDK 开发定制化功能，例如完成集群各节点信息采集，分析本节点业务特征。
- PMU 信息采集插件：采集系统 PMU 性能记录。
- Docker 信息采集插件：采集当前环境 Docker 的一些参数信息。
- 系统信息采集插件：采集当前环境的内核参数、线程信息和一些资源信息（CPU、内存、IO、网络）等。
- 线程感知插件：感知关键线程信息。
- 评估插件：分析业务运行时系统的 NUMA 和网络信息，给用户推荐使用的调优方式。
- 系统调优插件：（1）stealtask ：优化 CPU 调优 （2）smc_tune（SMC-D）：基于内核共享内存通信特性，提高网络吞吐，降低时延 （3）xcall_tune ：跳过非关键流程的代码路径，优化 SYSCALL 的处理底噪
- Docker 调优插件：利用 cpuburst 特性在突发负载下环境 CPU 性能瓶颈。

约束限制

- SMC-D：需要在服务端客户端建链前，完成使能 smc 加速。比较适用于长链接多的场景。
- Docker 调优：暂不适用于 K8s 容器场景。
- xcall_tune：内核配置选项 FAST_SYSCALL 打开。

应用场景

stealtask 适用于希望提高 CPU 利用率的场景，例如 Doris，该调优实例可以有效提高 CPU 利用，避免 CPU 空转。

XCALL 适用于应用程序的 SYSCALL 开销较大的场景，XCALL 提供一套跳过非关键流程的代码路径，来优化 SYSCALL 的处理底噪，这些被跳过的非关键流程会牺牲部分维测和安全功能。

SMC-D 特别适用于需要高吞吐量和低延迟的应用场景，如高性能计算（HPC）、大数据处理和云计算平台。通过直接内存访问（DMA），SMC-D 能够显著减少 CPU 负载并提升交互式工作负载的速率。

cpuburst 适用于高负载容器场景，例如 Doris，能够缓解 CPU 限制带来的性能瓶颈。

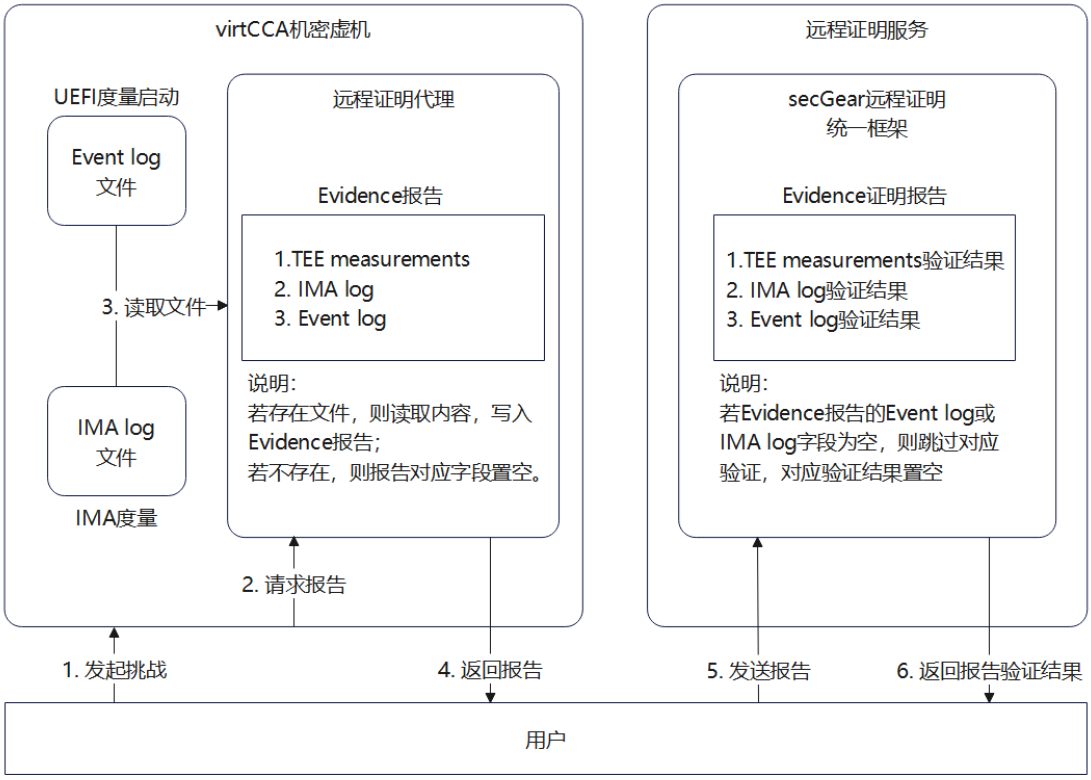
secGear 特性增强

secGear 远程证明统一框架是机密计算远程证明相关的关键组件，屏蔽不同 TEE 远程证明差异，提供 Attestation Agent 和 Attestation Service 两个组件，Agent 供用户集成获取证明报告，对接证明服务；Service 可独立部署，支持 iTrustee、virtCCA 远程证明报告的验证。当前版本新增了对 virtCCA 机密虚机 UEFI 度量启动的远程证明支持。同时为避免 UEFI 度量和 IMA 度量同时开启会产生冲突的情况，新增 virtCCA 的 IMA 度量寄存器自定义配置的功能。

功能描述

UEFI 度量启动时会将度量操作记录到 ACPI table CCEL (Confidential Computing Event log)。secGear 统一远程证明框架基于 ACPI table CCEL (Confidential Computing Event log)实现对 virtCCA 机密虚机 UEFI 度量启动的远程证明。secGear 远程证明通过对 Event log 文件内容处理，验证 virtCCA 机密虚机 UEFI 启动过程的完整性和可信性，并为后续远程证明提供可信的证据。

使用 secGear 远程证明时，远程证明代理会获取 virtCCA 机密虚机上的 Event log 文件内容。如果存在 Event log 文件，则将其内容及其他必要信息封装成报告，发送至远程证明服务进行报告校验，最终由远程证明服务给出包含 Event Log 验证结果的证明报告；如果不存在，则与 Direct Boot 模式相同，仅封装必要信息发送至远程证明服务进行报告校验，远程证明服务返回不包含 Event Log 验证结果的证明报告。



secGear统一远程证明框架远程证明流程

virtCCA 机密虚机开启 IMA 度量后，在 IMA 策略配置时支持使用 pcr 参数指定使用的 virtCCA 寄存器。如果同时启动 IMA 度量和 UEFI 度量，则需要在 IMA 策略配置时需要设置 pcr=4，指定使用 virtCCA 最后一个寄存器，避免度量扩展冲突。

应用场景

在金融、AI 等场景下，基于机密计算保护运行中的隐私数据安全时，远程证明是校验机密计算环境及应用合法性的技术手段，远程证明统一框架提供了易集成、易部署的组件，帮助用户快速使能机密计算远程证明能力。

GCC for openEuler CFGO 反馈优化特性增强

日益膨胀的代码体积导致当前处理器前端瓶颈成为普遍问题，影响程序运行性能。编译器反馈优化技术可以有效解决此类问题。

CFGO（Continuous Feature Guided Optimization）是 GCC for openEuler 的反馈优化技术名，指多模态（源代码、二进制）、全生命周期（编译、链接、链接后、运行时、OS、库）的持续反馈优化，主要包括以下两类优化技术：

- 代码布局优化：通过基本块重排、函数重排、冷热分区等技术，优化目标程序的二进制布局，提升 i-cache 和 i-TLB 命中率。
- 高级编译器优化：内联、循环展开、向量化、间接调用等提升编译优化技术受益于反馈信息，能够使编译器执行更精确的优化决策。

功能描述

GCC CFGO 反馈优化共包含三个子特性：CFGO-PGO、CFGO-CSPGO、CFGO-BOLT，通过依次使能这些特性可以缓解处理前端瓶颈，提升程序运行时性能。为了进一步提升优化效果，建议 CFGO 系列优化与链接时优化搭配使用，即在 CFGO-PGO、CFGO-CSPGO 优化过程中增加-flto=auto 编译选项。

- CFGO-PGO

CFGO-PGO 在传统 PGO 优化的基础上，利用 AI4C 对部分优化遍进行增强，主要包括 inline、常量传播、去虚化等优化，从而进一步提升性能。

- CFGO-CSPGO

PGO 的 profile 对上下文不敏感，可能导致次优的优化效果。通过在 PGO 后增加一次 CFGO-CSPGO 插桩优化流程，收集 inline 后的程序运行信息，从而为代码布局和寄存器优化等编译器优化遍提供更准确的执行信息，实现性能进一步提升。

- CFGO-BOLT

CFGO-BOLT 在基线版本的基础上，新增 aarch64 架构软件插桩、inline 优化支持等优化，进一步提升性能。

应用场景

CFGO 通用性较好，适用于整机性能瓶颈在 CPU 上，且 CPU 瓶颈在前端的 C/C++应用，如数据库、分布式存储等场景，一般可以取得 5~10%性能提升。

海光新安全处理器型号支持

功能描述

海光 CPU 内部包含密码协处理器（CCP），密码协处理器分为两类，其中一类是 PSP-CCP，另一类是 NTB-CCP。

PSP-CCP 为海光安全处理器提供 CPU 内置的密钥生成、安全启动、可信计算、CSV 等功能，支持 TKM 等模块内部的密码运算。

NTB-CCP 则供 C86 通用处理器核心使用，用来支持纯算力的密码学运算。

新增 PSP-CCP 和 NTB-CCP 的 vendor/device ID 的识别，为系统提供更丰富的硬件支持。

应用场景

密码协处理器为海光安全功能提供了密码学运算基础。PSP-CCP 为海光可信计算、机密计算等技术提供硬件基础；NTB-CCP 为操作系统中的密码库提供底层运算能力。海光安全功能所及的应用场景均是海光密码协处理器的应用场景。

海光可信计算特性

功能描述

当前海光可信计算内核驱动会调用内核 CCP 模块提供的命令接口，CCP 模块初始化时加密虚拟化特性的判断与可信功能并无关联，优化内核 CCP 模块，使可信功能解除与加密虚拟化特性的逻辑依赖，同时不会影响现有代码的运行，提高可信功能的健壮性。

应用场景

该特性主要是使海光可信功能更具独立性，主要应用于海光的可信计算技术，包括 TPM（Trusted Platform Module）、TCM（Trusted Cryptography Module）、TPCM（Trusted Platform Control Module）、TDM（Trusted Dynamic Measuring）、TSB（Trusted Secure Boot）。

海光密码加速特性

密码相关应用除了要求提供高效的密码运算能力外，还往往需要一种安全有效的密钥管理能力，密钥的安全是安全的基础，密钥如果被窃取则安全无从谈起。目前大量应用的密钥管理方式存在着简单粗放的问题，有的甚至把密钥以明文形式直接存储在文件系统中，密钥的安全存在巨大的隐患。密码技术的广泛应用迫切需要一种既安全又易用的密钥管理使用方式。

TKM 是海光的内生密码技术之一。传统的密钥管理方案通常使用单独的密码卡，通过 PCIE 等外接于主板上，与 CPU 通过总线连接。传统方案增加硬件成本的同时，加大了主板硬件设计的复杂度，也增加了暴露面，容易受总线类物理攻击。 CPU 利用内置安全处理器对密钥管理做了相关支持与拓展，在CPU内部实现了TKM（Trusted Key Management Module）模块。

功能描述

TKM 虚拟化支持和性能优化。

支持物理主机使用 TKM 的同时，通过 TKM 虚拟化技术虚拟出多个 vTKM 实例，支持虚拟机内使用 vTKM，主要特性如下：

- 虚拟机内使用 vTKM，vTKM 提供与主机 TKM 相同的密钥管理与密码运算功能；
- vTKM 实例之间相互隔离，每个 vTKM 实例代表一个独立的密钥空间。
- 支持每个虚拟机独占一个 vTKM 实例，拥有自己独立的密钥资源，相互之间不可访问，也支持多个虚拟机共享同一个 vTKM 实例，共享同一个密钥空间；
- 针对内核发送 vTKM 到 PSP 硬件上，新增了以下两个方面的优化：
- 为内核 PSP 驱动增加异步命令发送的功能支持，避免长时间阻塞单个 CPU 核心。
- 支持 PSP RING_BUFFER Overcommit 功能，发送命令给 PSP 硬件时，允许借助 RING_BUFFER 数据结构发送批量命令给 PSP，Overcommit 功能允许超额提交一组特定格式的空命令在队列中。PSP 读取到空命令时，将等待 C86 写入有效的命令后再开始执行。这样将有效减少 C86 到 PSP 的 IO 开销。

应用场景

TKM 应用场景:

TKM 的应用场景包括密码机服务器、工控系统、KMS 以及各种通用系统中的密钥管理等，根据需要基于 TKM 接口也可以进一步封装符合相关标准的接口。

TKM 提供的 GM/T 0018 标准接口，即 SDF 接口，可广泛应用在专用的密码设备中，比如服务器密码机、签名验签服务器、VPN 服务器等；另外，在需要密码服务的节点上使用支持 TKM 的海光服务器，改变原有通过网络访问池化密码资源的方式，通过本地密码资源直接访问，实现一种“分布式密码计算”方式，进而降低成本、降低网络带宽占用、优化性能。

支持树莓派

Raspberry Pi（树莓派）是由 Raspberry Pi 基金会与 Broadcom 公司合作开发的一系列小型单板计算机。凭借其价格低、体积小、能耗低、高可编程性以及丰富的生态系统等特点，树莓派在工业自动化、机器人技术、物联网、教育以及业余爱好者项目等领域得到了广泛应用。树莓派 4B 和树莓派 5 作为树莓派产品线的经典代表，采用 ARM 架构的处理器。其中树莓派 4B 是极具性价比的普及型单板计算机，树莓派 5 凭借其显著的性能突破和扩展能力成为一款在高性能边缘计算领域颇具竞争力的创新产品。

功能描述

作为开源硬件领域的一个较为高阶的硬件产品，树莓派 4B 和树莓派 5 支持 Raspberry Pi OS、Ubuntu、openEuler 等多种 Linux 发行版，外设丰富，具有较强的视频编解码能力，以及板载网络等功能，完全可以作为独立计算机系统使用。

应用场景

树莓派 4B 和树莓派 5 凭借其强大的性能和丰富的扩展能力，广泛应用于多个领域：

1. 教育与学习：学习 Python 等编程语言、借助外设接口进行电子实验；
2. 多媒体与娱乐：作为媒体中心或游戏机；

3. 物联网和智能家居：作为传感器节点或智能家居中枢，用于环境监测、家庭自动化控制和边缘计算；
4. 服务器与网络应用：用于家庭服务器、轻量级 Web 服务及容器化应用；
5. 创客与 DIY 项目：用于机器人控制、3D 打印管理和无人机飞控；
6. 科研与开发：用于 AI 实验、嵌入式开发原型验证；
7. 工业与自动化：实现设备监控、人机界面和机器视觉。

8. 著作权说明

9. 商标

10. 附录

附录 1：搭建开发环境

环境准备	地址
下载安装 openEuler	https://openeuler.org/zh/download/

开发环境准备	https://gitee.com/openeuler/community/blob/master/zh/contributors/prepare-environment.md
构建软件包	https://gitee.com/openeuler/community/blob/master/zh/contributors/package-install.md

附录 2：安全处理流程和安全批露信息

社区安全问题披露	地址
安全处理流程	https://gitee.com/openeuler/security-committee/blob/master/docs/zh/vulnerability-management-process/security-process.md
安全披露信息	https://gitee.com/openeuler/security-committee/blob/master/docs/zh/vulnerability-management-process/security-disclosure.md
安全保障策略总纲	https://gitee.com/openeuler/security-committee/blob/master/security-strategy-overview.md