openEuler 开源操作系统技术白皮书

创新项目总览

2025 年 11 月

OpenAtom openEuler 社区

目录

1 社区简介 5

2 技术生态 5

openEuler 覆盖全场景的创新平台 5

openEuler 对 Linux Kernel 的持续贡献 6

openEuler 软件包仓库 7

openEuler 开放透明的开源软件供应链管理 8

通过社区认证的 openEuler 发行版 8

openEuler 开源操作系统架构图 9

3 AI 创新 10

OS for AI 10

sysHAX 大语言模型异构协同加速运行时 10

sysTrace AI 易运维 11

GMEM 异构内存统一管理 13

AI for OS 15

openEuler Intelligence 15

开箱易用 24

功能描述 24

应用场景 25

4 场景化创新 25

服务器 25

sysSentry 统一故障管理框架 25

DPUDirect 直连聚合 27

vDPA 加速框架 28

云原生&虚拟化 30

众核高密 30

NestOS 云底座操作系统 34

KubeOS 容器操作系统 36

Kmesh 高性能服务治理框架 38

iSulad 轻量级容器引擎 39

StratoVirt 高可靠虚拟化引擎 42
Kuasar 机密容器 43
嵌入式 45
系统架构图 46
南向生态 46
嵌入式弹性虚拟化底座 46
混合关键性部署框架 47
北向生态 47
UniProton 硬实时系统 47
应用场景 48
边缘计算 48
openEuler 支持 KubeEdge 48
分布式软总线 50
5 基础能力创新 52
内核创新 52
高效并发与极致性能 54
oeAware 业务场景自适应无感调优 54
A-Tune 智能调优引擎 56
Gazelle 轻量级用户态协议栈 58
BiSheng JDK 毕昇 JDK 62
GCC for openEuler 66
LLVM for openEuler 68
Go for openEuler 72
编译器插件框架 75
强安全和高可靠 77
secGear 机密计算统一开发框架 77
virtCCA 机密计算 79
CCA 机密计算 81
极简运维 83
oeDeploy 软件部署工具 83
A-Ops 智能运维 84

6 开发者支持 86

基础设施 86

CVE Manager 漏洞管理 86

Compass-CI 88

OEPKGS openEuler 软件扩展仓库 90

开发者工具 92

DevStation 智能开发环境 92

DevStore 开发者软件商店 94

EPKG 新型软件包 95

QuickIssue 快捷的社区 issue 分类提交工具 97

兼容性与技术评测 98

OSV 技术评测 98

openEuler 兼容性全景清单 99

openEuler 技术测评 101

7 致谢 103

1 社区简介

OpenAtom openEuler（简称 openEuler 或开源欧拉）是由开放原子开源基金会孵化的全场景开源操作系统项目，面向数字基础设施四大核心场景（服务器、云计算、边缘计算、嵌入式），全面支持 ARM、x86、RISC-V、loongArch、PowerPC、SW-64 等多样性计算架构。

开源欧拉自贡献到基金会孵化以来，在产业、生态、国际化等方面已取得长足进步。产业上，超过 21 家厂商发布开源欧拉商业发型版，装机量突破 1000 万套，覆盖金融、通信、能源、政务、互联网等众多行业。生态上，秉承共建、共治、共享的理念，汇聚超过 2000 家成员单位，主要的捐赠人和贡献者企业包括 Intel、Arm 等国际领先的企业。国际化上，与 Linux 等国际组织技术合作构筑全球开源生态，40 多款国际项目原生支持开源欧拉。除此之外，开源欧拉一直重视开源软件供应链安全方面的实践与落地，2024 年开源欧拉社区成为首个通过 ISO18974 软件供应链安全认证的开源社区，证明开源欧拉在该领域的领先地位。

未来，面向智能化时代，将推动构筑基于开源欧拉的 AI 全栈开源体系，加速大模型推理技术普惠化，打造智能化数字底座。

发展历程

2019 年 9 月，华为将自研服务器操作系统 EulerOS 全面开源，并正式更名为 openEuler，迈出了从企业自用到产业共建的第一步；同年 12 月，openEuler 社区正式上线。2021 年 11 月，openEuler 整体捐赠给开放原子开源基金会，实现由“企业主导”向“产业共属”的历史性转变；一年后，凭借在多样性算力、云原生调度等方面的创新，openEuler 荣获 2022 世界互联网领先科技成果奖。2022 年 12 月，基金会批准成立 openEuler 项目群，开始系统接收外部项目捐赠，社区治理模式进一步升级。到 2024 年 12 月，openEuler 系操作系统在国内新增服务器市场份额已达 50.2%，并在通信、政府、金融、公共事业（电力）、能源五大关键行业实现占有率第一，标志着中国开源操作系统生态完成了从技术追赶到产业引领的跨越。

2 技术生态

openEuler 覆盖全场景的创新平台

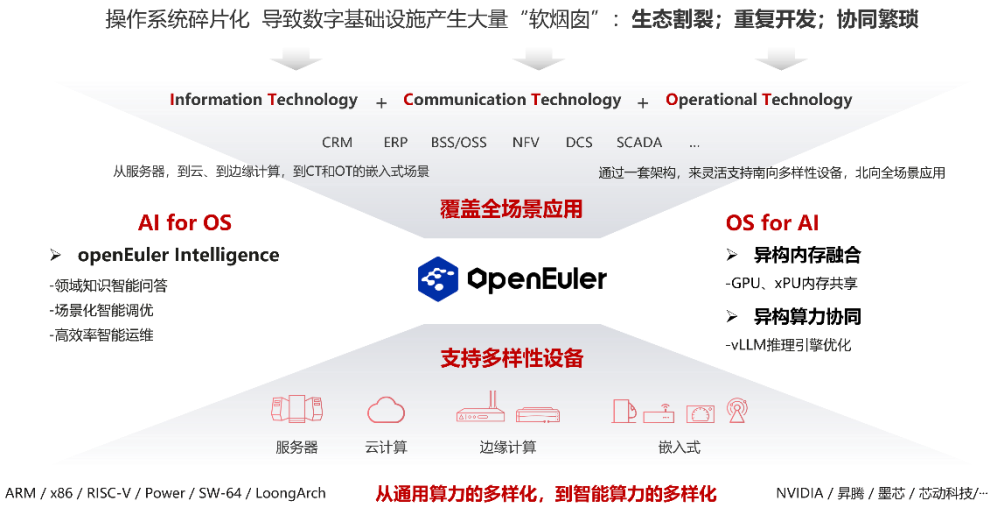
openEuler 是面向数字基础设施开源操作系统。通过一套操作系统架构，南向支持多样性设备，北向覆盖全场景应用，横向对接 OpenHarmony 等其他操作系统，通过能力共享实现生态互通。openEuler 突破性的实现了一套 OS 架构下，100%支持主流计算架构，是最佳支持多样性算力的开源操作系统。openEuler 开创性的提出全场景操作系统理念，通过全栈原子化解耦和榫卯架构，实现版本灵活构建、服务自由组合。通过一套操作系统架构，实现了对服务器、云计算、边缘计算和嵌入式等场景的支持。

openEuler 操作系统具备以下特征：

面向数字基础设施的全场景。支持服务器、云计算、边缘计算、嵌入式等应用场景，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。

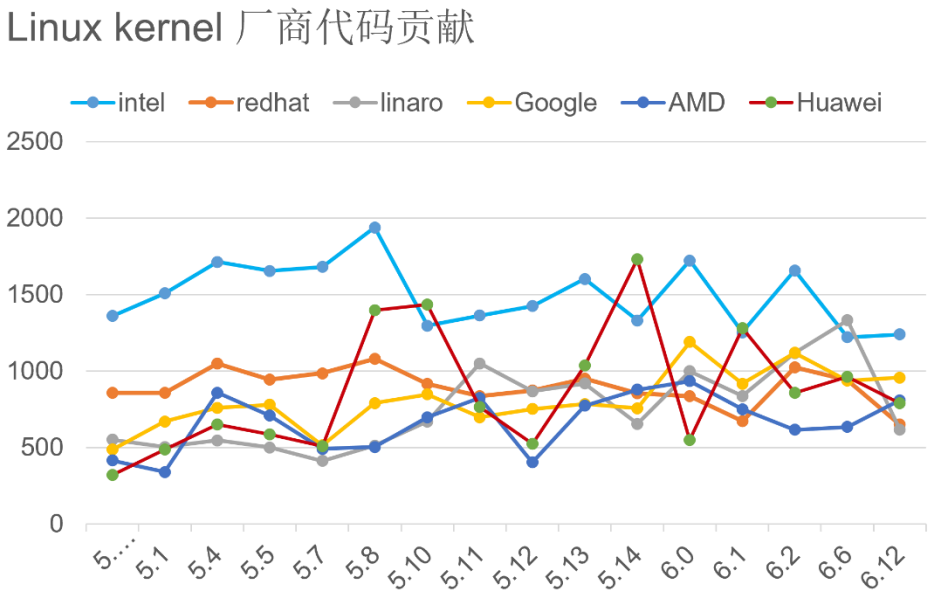
最佳多样性计算支持。支持 Intel、AMD、鲲鹏、飞腾、兆芯、龙芯、海光、申威等国内外主流芯片，同时支持包括 ARM、x86、RISC-V、DPU、IPU、NPU 等多种计算架构，将不同的负载匹配最佳运算能力，充分发挥多样性计算性能。

内核创新，自主演进。引领内核创新，24.03LTS 版本支持了 Linux 内核 6.6 版本，贡献国内第一，全球 top3；以多样性算力、以内存为中心架构创新，贡献上游社区，引领技术发展。



openEuler 对 Linux Kernel 的持续贡献

openEuler 社区成员（华为、龙芯中科、麒麟软件、统信软件、飞腾、成都菁蓉联创、中国电信、中国移动等）持续贡献 Linux Kernel 上游社区，回馈主要集中在：芯片架构、ACPI、内存管理、文件系统、 Media、内核文档、针对整个内核质量加固的 bugfix 及代码重构等内容。

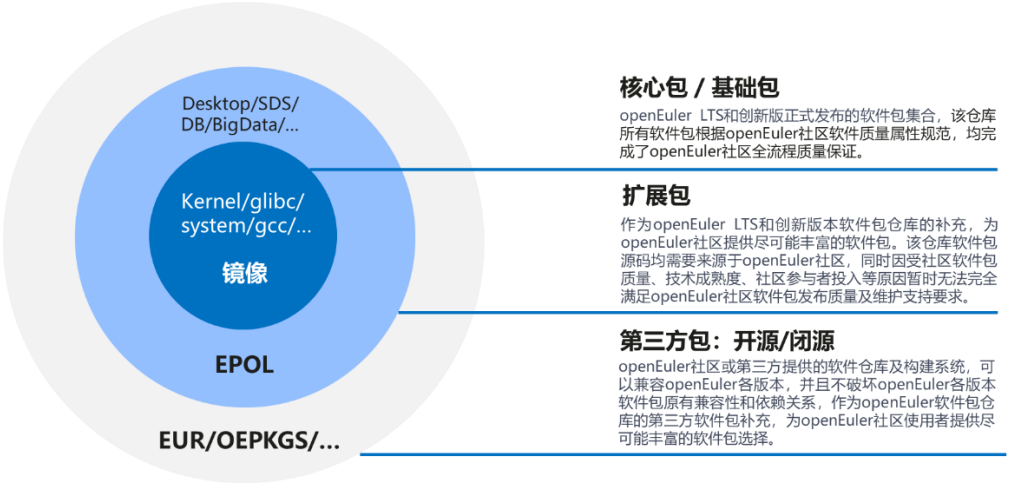


- **华为贡献历史排名 TOP5 之一，5.10/5.14/6.1 版本贡献排名第一，6.6 版本贡献世界 top3**
- **GCC 编译器中国首家 Maintainer，Linux 内核补丁 13000+，Docker 社区贡献排名 TOP5**

openEuler 社区拥有社区伙伴超过 2089 家，贡献者 2.2 万+，累计代码合入超 PR 225.8 万次，全球下载量达 424 万+。

openEuler 软件包仓库

openEuler 社区及其他第三方开发者共同提供了丰富易用的软件包，在最新的版本中，软件包总数已超过 3.6 万。openEuler 社区根据这些软件包的来源、质量属性、维护方式等不同维度划分为三类 openEuler 社区软件仓库，openEuler 社区版本使用者可以根据自己的需求配置不同的软件仓库；软件包可能会随其用量、稳定性、维护状态等在不同的软件仓库中依据社区规则重新分布。



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、运行依赖关系和上游社区，三者之前形成闭环且完整透明的软件供应链管理。

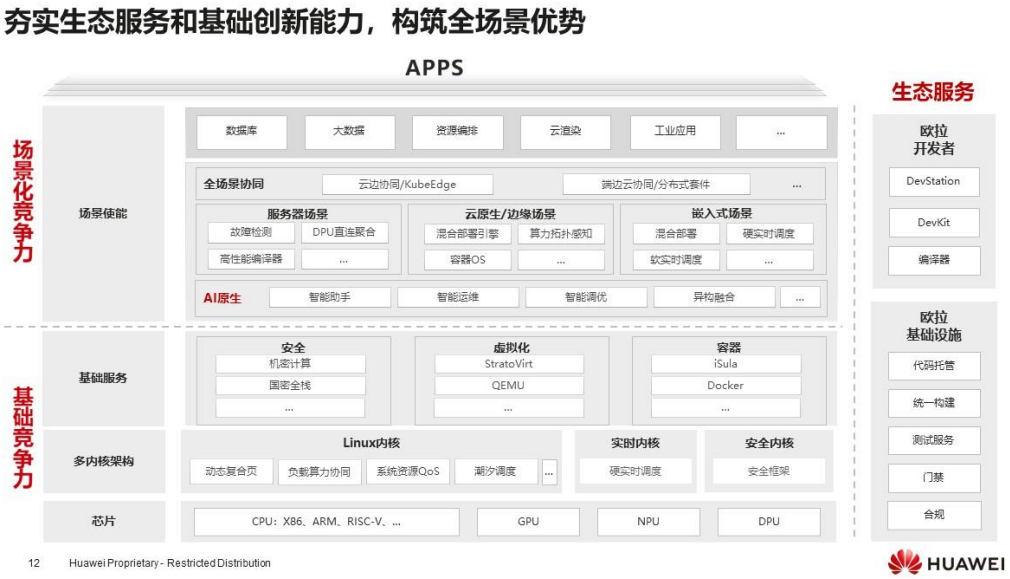
通过社区认证的 openEuler 发行版

按时间排序，详情链接：https://www.openeuler.openatom.cn/zh/download/commercial-release/

伙伴名称	系统名称
广州汇智通信技术有限公司	汇智 TeligenOS 服务器操作系统 V3
浪潮云信息技术股份公司	浪潮云启操作系统 23.12 LTS SP2
成都鼎桥通信技术有限公司	鼎桥 TDOS 服务器操作系统 V1.0
浪潮云信息技术股份公司	浪潮云启操作系统 23.12 LTS SP1
中科海微（北京）科技有限公司	SeawayEdgeV1.00
麒麟信安 Kylinsec	麒麟信安 Kylinsec V3.5.2
中软国际科技服务有限公司	磐石操作系统 CSIOS V1.0.0
江苏润和软件股份有限公司	润和企业级服务器操作系统 HopeOS V22
恒生电子股份有限公司	恒生 HUNDSUN LightOS 操作系统 V1.0
广东中兴新支点技术有限公司	新支点服务器操作系统 V6
超聚变数字技术有限公司	超聚变服务器操作系统 FusionOS 22 （免费使用授权）
浪潮云信息技术股份公司	浪潮云启操作系统 23.12 LTS
软通动力信息技术（集团）股份有限公司	天鹤操作系统 ISSEOS V22
北京凝思软件股份有限公司	凝思安全操作系统 V6.0
北京拓林思软件有限公司	拓林思企业级服务器操作系统
统信软件技术有限公司	统信服务器操作系统 V20

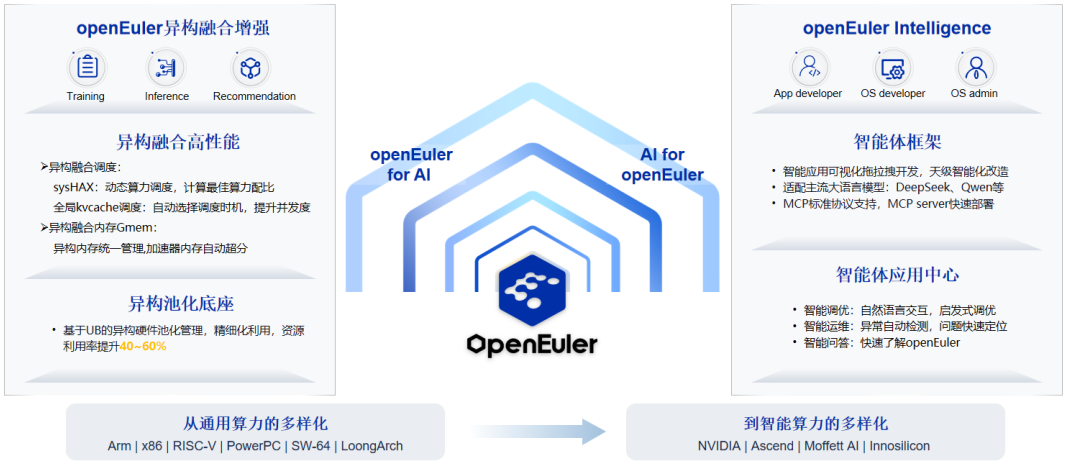
伙伴名称	系统名称
新华三技术有限公司	新华三 磐宁操作系统（NingOS) V3.0
中国移动云能中心	BCLinux for Euler V21.10
中科红旗（北京）信息科技有限公司	红旗 Asianux 服务器操作系统 V8.1
江苏润和软件股份有限公司	HopeStage
麒麟软件有限公司	银河麒麟高级服务器操作系统 V10
麒麟信安 Kylinsec	麒麟信安 Kylinsec V3.5.1
麒麟信安 Kylinsec	麒麟信安 Kylinsec V3.4-5
麒麟信安 Kylinsec	麒麟信安 Kylinsec V3.4-4
江苏润和软件股份有限公司	HopeEdge
中科院软件所	中科傲来服务器操作系统
普华软件	普华服务器操作系统 v5.1
同源 OS	同源 OS 8.1 欧拉发行版

openEuler 开源操作系统架构图



3 AI 创新

智能时代，操作系统需要面向 AI 不断演进，openEuler 提出了 AI for OS 和 OS for AI 的全新操作系统设计理念，AI for OS 方面，openEuler 在操作系统开发、部署、运维、调优全流程以 AI 加持，让操作系统更智能；对于 OS for AI，openEuler 已支持 ARM，x86，RISC-V 等全部主流通用计算架构，通过异构融合调度和异构融合内存等提升聚能算力能力，支持主流 AI 处理器，成为使能多样性算力的首选。



OS for AI

sysHAX 大语言模型异构协同加速运行时

SIG intelligence

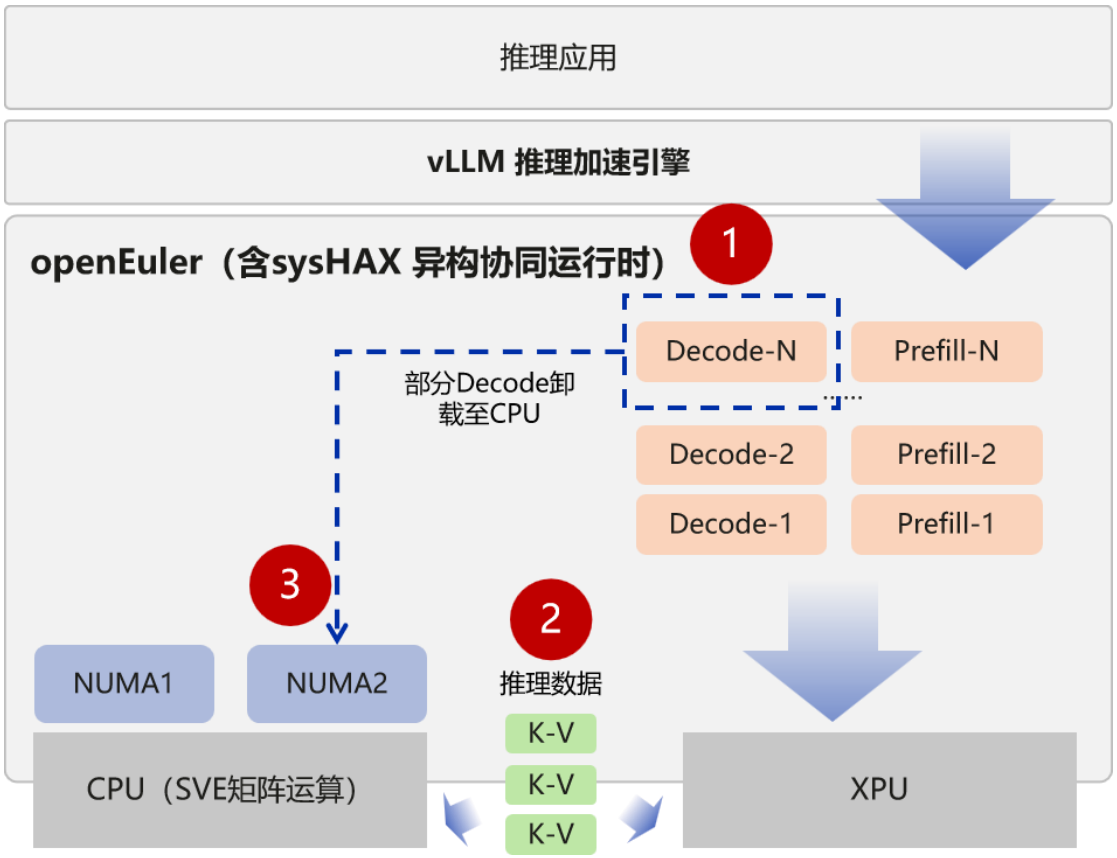
sysHAX 是一个大模型推理服务加速工具，通过 CPU 和 XPU 的合理配合，使能 CPU 能力，降低大模型的算力需求，提高大模型推理服务负载能力。

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 CPU+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：

- **CPU 与 XPU 算力协同**，通过 openEuler 社区 sysHax 特性，根据 XPU 负载情况，将部分 decode 任务卸载到 CPU，包括 CPU 算力填充 decode 任务，中小模型 LLM 推理吞吐量提升，CPU 算力填充 MOE 专家 任务，稠密 MLA 专家在 GPU，MOE 模型吞吐提升；同时提供算子并行下发加速及融合能力，降低在 CPU 侧开销。

- **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。



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。

仓库地址

https://gitee.com/openeuler/sysHAX

sysTrace AI 易运维

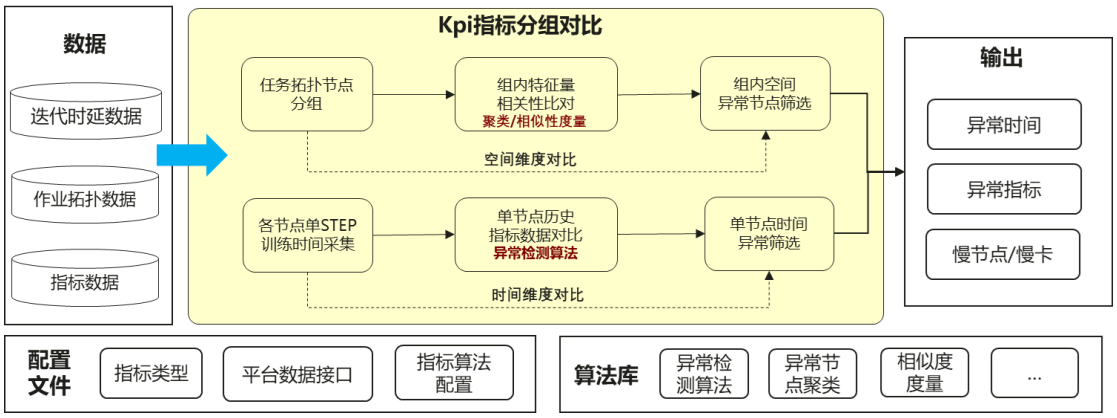
SIG	ops

sysTrace 是一款系统性能跟踪工具，聚焦 AI 任务过程中的性能问题跟踪与分析，AI 全栈观测，实现零侵扰、全栈跟踪，实时便捷提供故障分析。

技术挑战

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。常见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，sysTrace 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。

仓库地址

https://gitee.com/openeuler/sysTrace

GMEM 异构内存统一管理

SIG ops

GMEM（Generalized Memory Management）是一个异构通用内存管理框架，提供了异构内存互联的中心化管理机制，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。

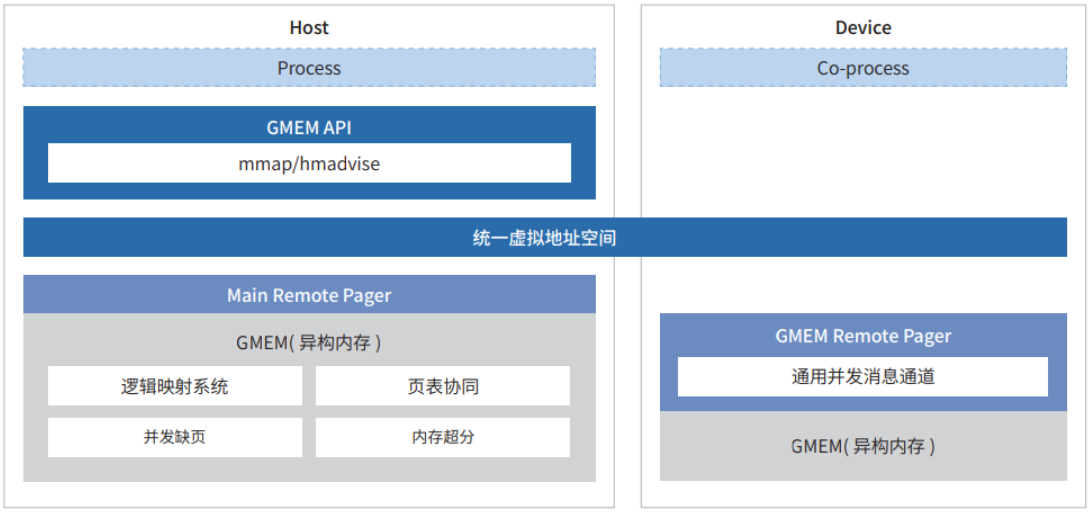
技术挑战

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。异构加速器品类繁多，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。

当前异构侧数据管理 CPU 与异构侧分离，数据显式搬移，易用性和性能难以平衡：异构设备 HBM 内存严重不足，应用手动 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，急需统一的有效对等内存管理机制，Linux 现有 HMM 框架搁浅，编程复杂度高且依赖人工调优，NV、AMD 虽尝试接入，但由于架构问题导致代码缺乏通用性，性能可移植性差，引起上游 OS 社区反弹。异构加速器领域亟需高效的统一内存管理机制。

基于这种背景，openEuler 提出 GMEM，一种易用性强，性能与可移植性好的异构通用内存管理框架。

功能描述



GMEM 提供了异构互联内存的中心化管理，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。

- **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。

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。)

仓库地址

https://gitee.com/openeuler/libgmem

AI for OS

openEuler Intelligence

SIG intelligence

openEuler intelligence 是基于 openEuler 操作系统的智能化平台，它集成了语义接口注册、工作流编排&调度和知识库等功能，可以为用户提供一些基础的智能化服务；也允许用户接入本地的自定义接口，提供高阶的智能化服务，支持应用智能化改造。

OS 智能体应用中心

openEuler Intelligence 支持 Agent 应用中心，自带智能问答、智能调优、智能诊断等智能应用。

问答智能体

openEuler Intelligence 智能问答支持用户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和使用通过智能体框架编排的工作流应用，便捷了新手用户对 openEuler 知识的获取和对 openEuler AI 能力的使用。

功能描述

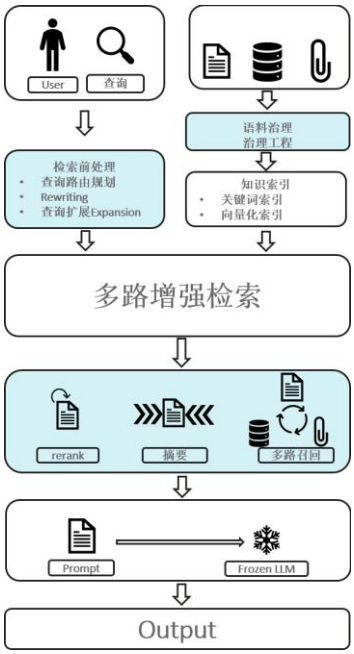
openEuler Intelligence 目前支持 Web 和智能 Shell 两个入口。用户可以使用 Web 入口通过可视化页面进行智能问答交互，也可以通过 Shell 入口借助 openai-api-key 调用智能体框架中的智能问答或者是编排好的工作流。

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。

问答准确率提升

RAG(检索增强技术)是指对大型语言模型输出进行优化，使其能够在生成响应之前引用训练数据来源之外的知识库。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等方面进行了增强，更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，能大幅度提升智能问答准确率，提升问答服务体验。



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。

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。

调优智能体

功能描述

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

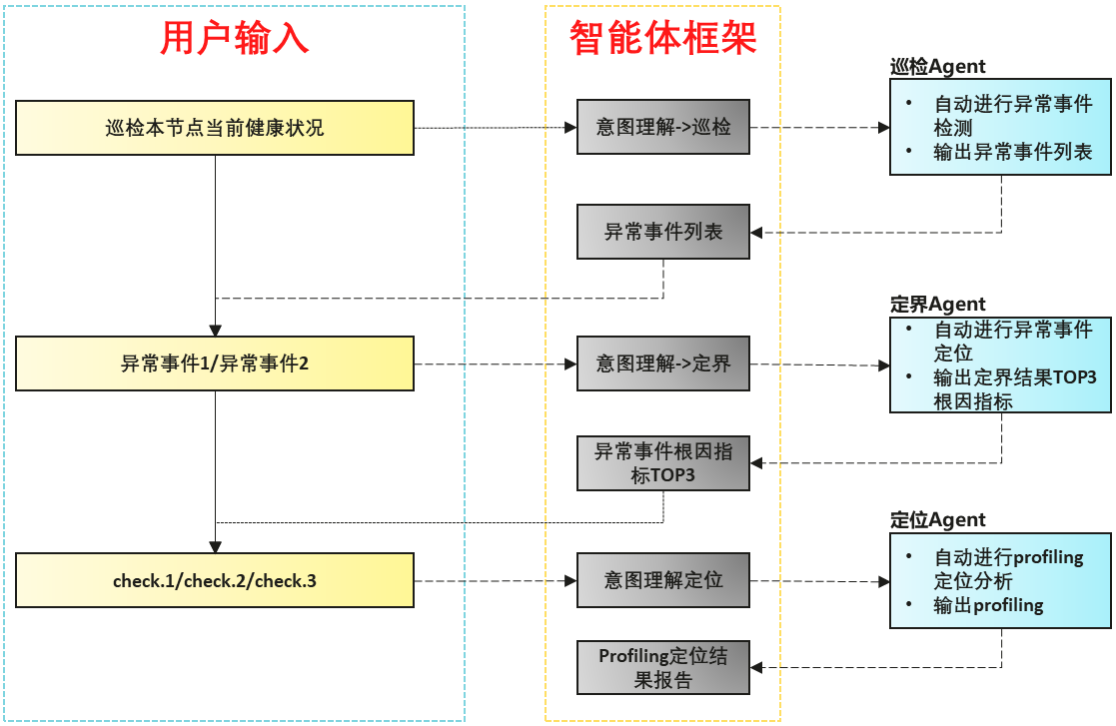
在上述功能入口，用户可通过与 openEuler Intelligence 进行自然语言交互，完成性能数据采集、系统性能分析、系统性能优化等作业，实现启发式调优。

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。

诊断智能体



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 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。

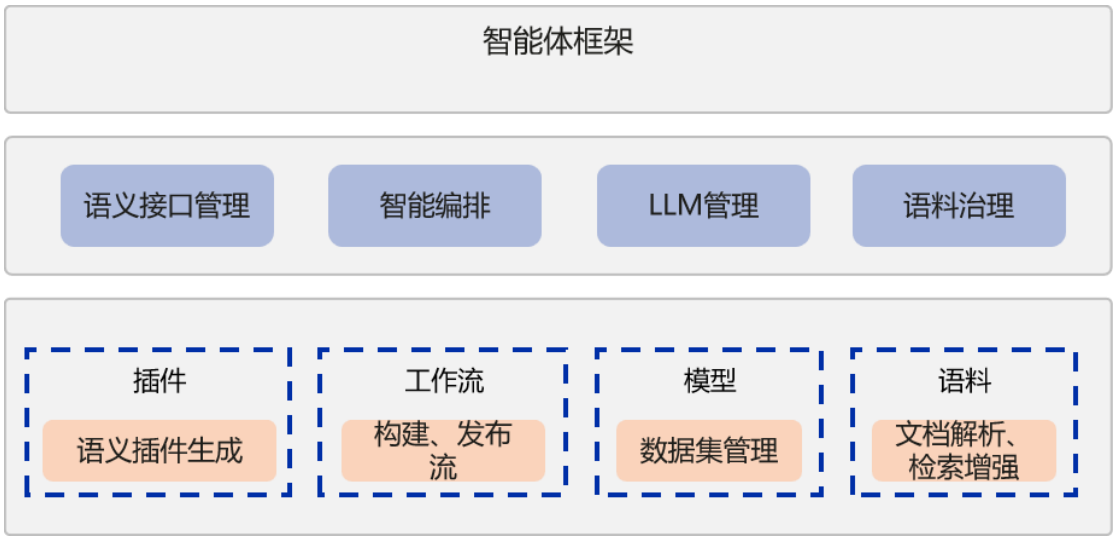
- 其中检测能力指的是：进行单机性能指标采集、性能分析、异常事件检测。
- 其中定界能力指的是：结合异常检测结果进行根因定位，输出 top3 根因指标及其描述。

- 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。

OS 智能体框架

在 openEuler 智能体不断演进的同时，openEuler Intelligence 形成了智能体框架，提供了智能体开发的基本能力，支持企业开发自己的 os 智能体，智能应用天级开发，工作流可视化编排。

openEuler Intelligence 智能体框架支持注册语义接口（含有自然语言注释的接口形式）和 mcp 服务、构建 Agent 和工作流应用等功能。支持 web 和客户端两种形式，为开发者和企业的使用提供了巨大的便利。



功能描述

openEuler Intelligence 允许用户将系统提供的语义接口、用户注册的语义接口和 mcp 服务以可视化的形式连线编排成工作流，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。

应用场景

openEuler Intelligence 智能体框架支持企业/个人通过简易的方式构建智能 Agent 应用，实现智能编排和调度工作流服务，完成生产环境复杂任务。

应用案例

天翼云 CTyun 基于 openEuler Intelligence 实现智能调优

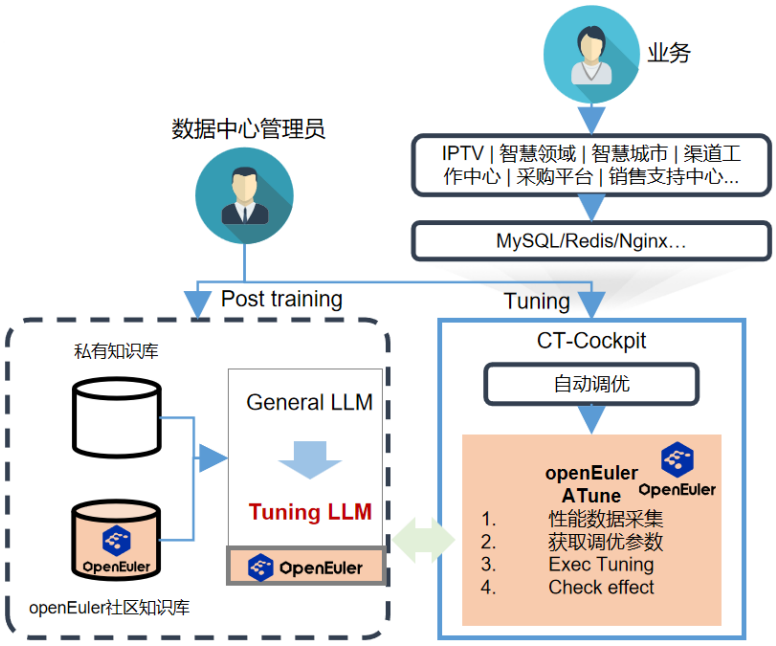
天翼云推出基于 openEuler 的自研操作系统 CTyunOS，实现智能调优系统与技术创新，服务中国电信集团云改数转战略，在电信、国资央企、政务等行业实现大规模生产环境部署，助力数字经济发展。

应用场景

天翼云是中国最大的电信云之一，操作系统数量多，业务规模大，场景变化多，在应用调优方面面临一些挑战：

解决方案

天翼云基于 openEuler 自研的操作系统 CTyunOS 将 openEuler Intelligence 集成到 CT-Cockpit 中，实现智能调优



客户价值

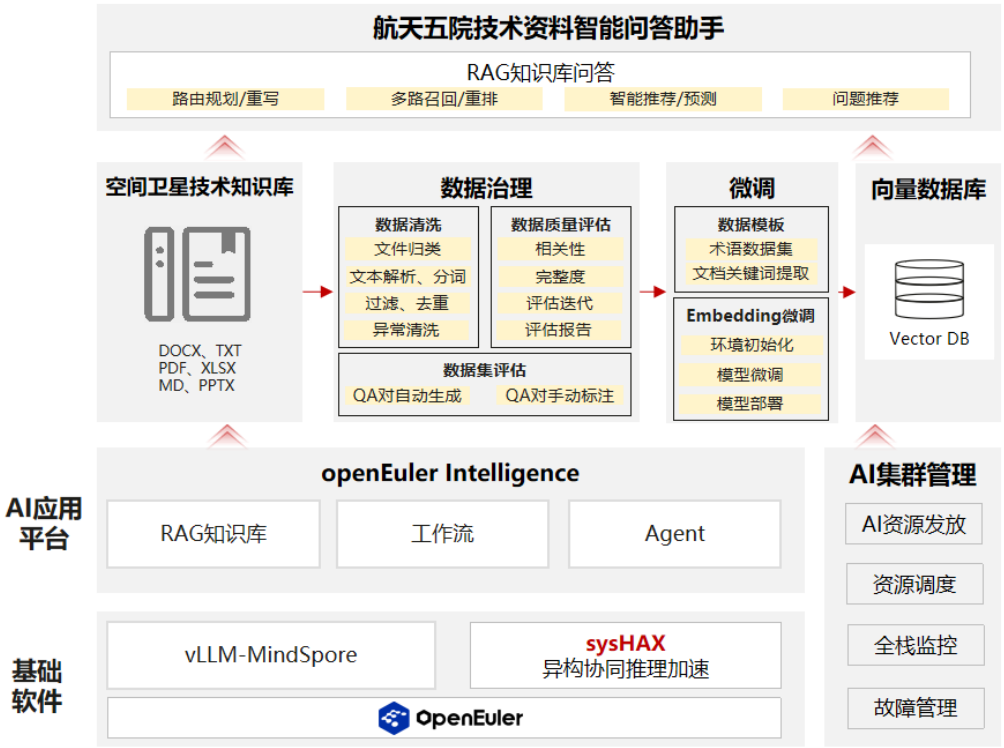
- 将大规模复杂应用的调优时间从数天缩短至数小时，无需更多专家参与；
- 通过 AI 调优实现 10%的性能提升。

openEuler Intelligence 联合华鲲振宇面向航天五院打造空间卫星技术资料智能问答助手

空间卫星技术资料智能问答助手可嵌入办公全链路，广泛适用于以下场景：智能问答、文档生成、摘要提取、事项划分和语义匹配等，可快速处理各类信息和任务；智能导办、辅助批示、智能校对、会议助手和智能督办等工具，则为工作流程提供全方位的智能化支持，有效提升工作效率和质量。

解决方案

openEuler Intelligence 联合华鲲振宇打造空间卫星技术资料智能问答助手，以“一站式 AI 助手”形态，让每一次交互更高效、更智慧。



客户价值

软硬协同：充分利用 CPU/xPU 算力，吞吐提升 10%；
数据安全：网络安全隔离，数据加密存储；
高并发低延时：支持 3000+文档分钟级入库；
高准确：问答准确率可测可信，至少达到 80%；
高效率：提供 AI 应用 Pipeline 可视化编排能力，支持 Function Call 及 MCP，缩短应用开发时间 50%

仓库地址

https://gitee.com/openeuler/euler-copilot-framework

https://gitee.com/openeuler/euler-copilot-rag

https://gitee.com/openeuler/euler-copilot-witchaind-web

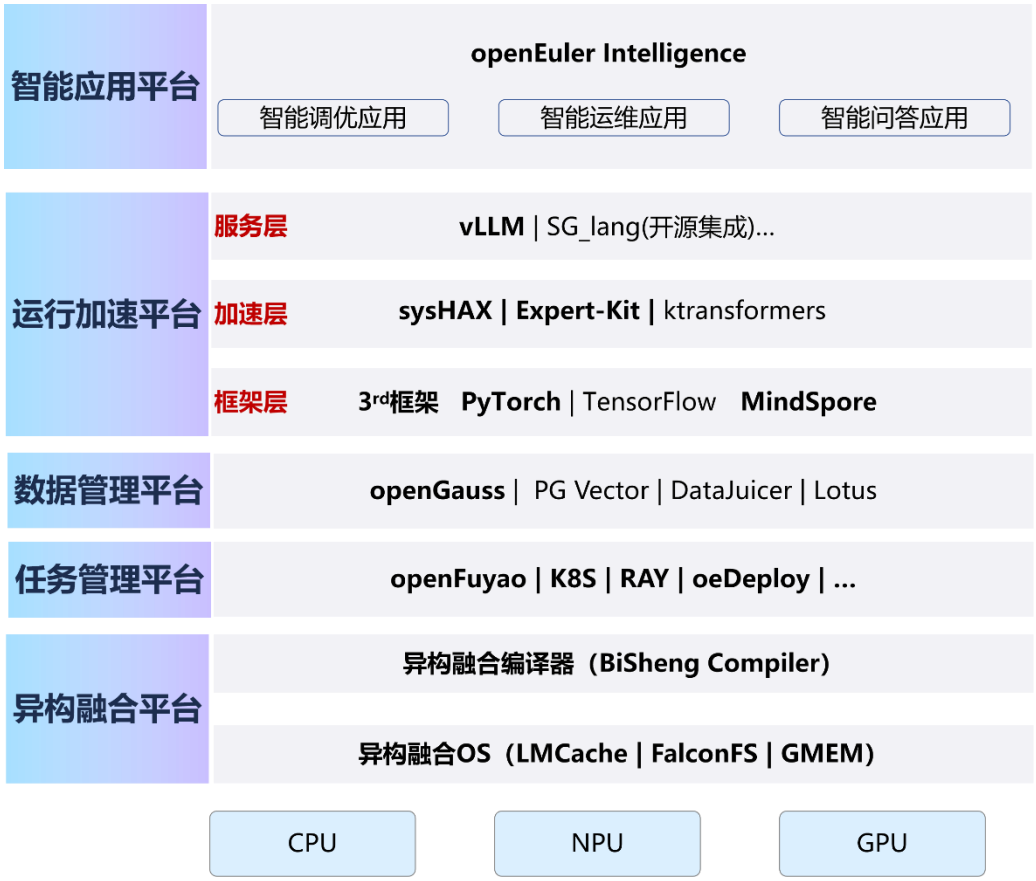
https://gitee.com/openeuler/euler-copilot-web

开箱易用

openEuler 提供开源 AI 基础软件栈 Intelligence Boom，集成了操作系统、数据库、AI 框架、模型优化工具；openEuler 在操作系统开发、运维、部署、调优全流程予以 AI 加持，为用户提供高效的开发运行环境和开箱即用的体验。

功能描述

openEuler 将部署安装、智能交互、AI 框架和异构算力融合集成于一体，提供可安装、开箱即用的 AI 能力：



1. openEuler Intelligence 是 openEuler 智能交互入口，通过可视化操作界面与低代码开发能力，赋能企业和开发者快速构建 AI 应用，实现智能运维、智能办公、数据分析等场景的高效落地，显著提升工作效能。
2. openEuler 全面兼容主流 AI 软件栈（驱动、SDK、训推框架、模型等），且致力于广泛兼容 AI 南北向生态，实现从底层硬件到上层应用的全链路贯通，保障各环节协同运作，为 AI 应用落地提供坚实生态支撑。
3. openEuler 聚能异构算力融合，GMEM 通过异构融合来高效管理内存，减小系统内存碎片；sysHAX 动态调度 K+X 算力资源，计算最佳算力配比，将 Decode 任务卸载至 CPU，最大程度利用 CPU 算力。
4. openEuler 提供部署工具 oeDeploy，支持主流 AI 开发软件和常用工具链的快速部署。

应用场景

openEuler 和 AI 深度结合，提供 Intelligence Boom 开源 AI 基础软件栈，强化操作系统原生智能，加速 Agent 生态成熟，适用于 AI 原生开发场景。

4 场景化创新

服务器

sysSentry 统一故障管理框架

SIG Base-service

sysSentry 是一款故障巡检框架，为用户提供在后台进行故障巡检的能力，支持对系统中 CPU、内存、磁盘、NPU 等硬件故障进行巡检和诊断。

技术挑战

传统的硬件故障管理缺少操作系统级别故障统一管理能力，故障信息格式碎片化、可靠性能力零散，sysSentry 通过提前发现系统中的软硬件故障并及时通知系统运维人员处理的方式，达到减少故障演变为现网事故、提升系统可靠性的目标。

功能描述

sysSentry 功能设计如下：

- **快速上报：**

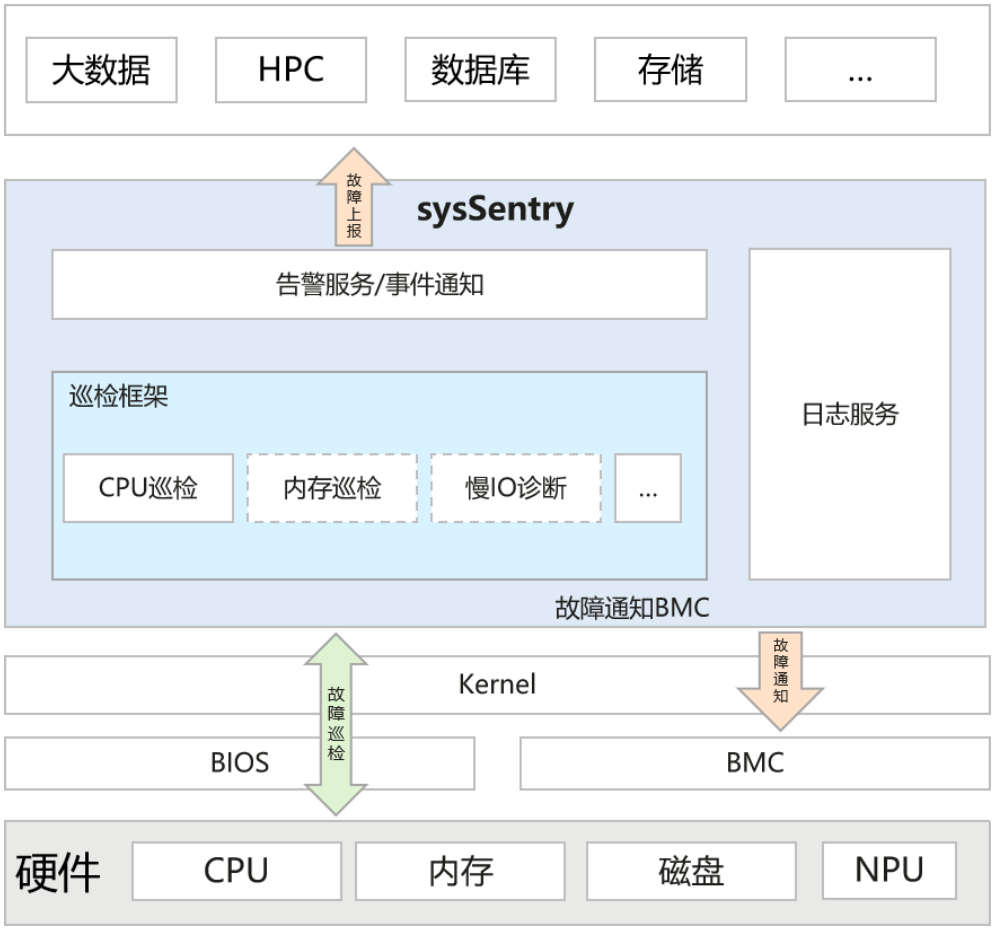
故障上报由被动感知->主动上报。业务管理服务按需订阅故障时间，故障发现秒级上报，业务快速恢复。

- **检测准确率高：**

  CPU 故障检测打通内核、BIOS、芯片，软硬结合高准确率检测，已知 CPU 硬件失效 98%检出。（内存、硬盘、XPU 等巡检能力陆续支持中）

- **插件化管理，容易扩展：**

  巡检功能插件化管理，不同硬件检测功能仅需新增配置即可实现天级接入，无需修改框架代码。



应用场景

sysSentry 统一故障管理框架实现故障主动快速上报，故障检测准确率高、易拓展，适用于服务器硬件维护场景。

仓库地址

https://gitee.com/openeuler/sysSentry

DPUDirect 直连聚合

SIG? DPU

直连聚合特性（DPUDirect）旨在为业务提供协同运行环境，允许业务在 HOST 和 DPU 之间灵活卸载及迁移。当前已实现进程级别无感卸载功能，提供跨 HOST 和 DPU 的协同框架，支持管理面进程无需改造进行拆分，并近无感知卸载到 DPU 上运行，卸载后进程同时保持对 HOST 侧业务进程的管理能力。直连聚合特性能够极大降低业务在 DPU 场景下的卸载成本，简化运维，大大降低后期维护成本。

技术挑战

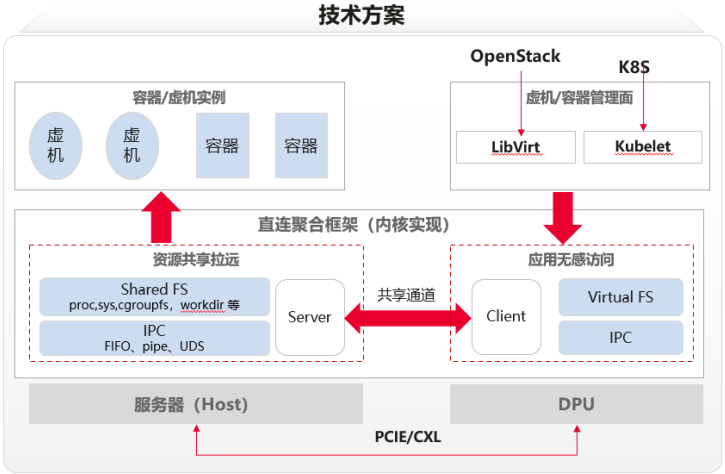
DPU 成为数据中心发展新趋势，当前业务管理面进程 DPU 卸载，基于用户态的拆分方案需要新增约 10K 代码，升级运维困难且容易被 CSP 锁定；运营商等客户，对面向典型场景的开放、标准化的透明无感卸载方案诉求强烈

功能描述

直连聚合框架，通过主机资源共享拉远，文件及 IPC 访问语义跨节点保持，实现管理面业务高效卸载；应用适配修改工作量由 10K+降低到<500LoC，极大降低软件迁移及维护成本

资源共享拉远：通过 Client/Server 模型的资源共享架构，实现 proc、sys、cgroupfs 和业务工作目录等虚机/容器管理目录的跨节点（Host & DPU）共享，提供无感访问通道。

应用无感卸载：基于文件系统及 IPC 访问语义跨节点保持，支撑虚机/容器管理面从 Host 到 DPU 的高效卸载，业务（近）无感知



应用场景

DPU 管理面无感卸载方案可应用于云计算中全卸载场景的容器或虚拟化管理面进程的 DPU 卸载；使用无感卸载方案，云上容器管理面进程（kubelet、dockerd）及虚拟化管理面进程（libvirtd）的卸载分别能够减少 10K+代码拆分工作量，业务卸载适配和维护的工作量降低近 20 倍，并且无需修改管理面业务逻辑，从而保证业务的软件兼容性和演进性。

仓库地址

https://gitee.com/openeuler/dpu-utilities

vDPA 加速框架

SIG Virt

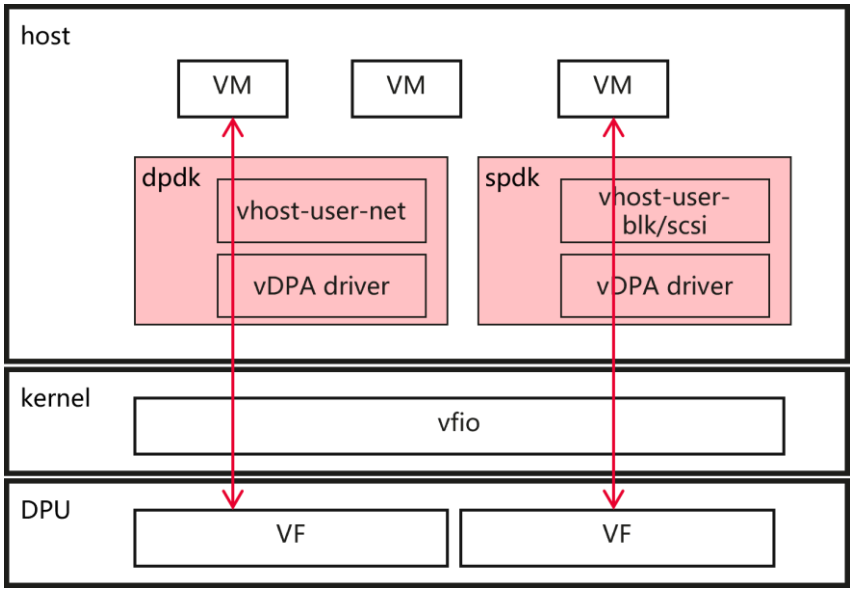
设备虚拟化由全虚拟化到 virtio 半虚拟化、vhost 数据面卸载、vfio 直通等方向演进，其整体发展趋势为：优化数据面提升 IO 性能，优化控制面提升设备管理灵活性。内核态 vDPA 旨在实现数据面直通的同时，支持网络、存储等各类设备，并实现虚拟机高性能热迁移。

技术挑战

目前开源主流的 vDPA 方案围绕用户态 dpdk、spdk 进行构建（如下图）。DPU 卡在主机侧呈现的设备通过 SR-IOV 能力创建出 VF，在用户态 vDPA 方案下，VF 通过 VFIO 驱动被纳管到 dpdk 以及 spdk 侧。dpdk/spdk 对外呈现 vhost-user 的 socket 设备，通过 qemu vhost-user-net/vhost-user-blk/vhost-user-scsi 的后端类型，从而给虚拟机呈现对应的 virtio 设备。

用户态 vDPA 方案下，存在如下几个问题：

- 主机侧资源消耗大：引入 DPU 卡的目的是卸载主机侧管理进程的资源，但是在用户态 vDPA 场景下，主机侧仍然需要运行 dpdk/spdk 组件，来呈现用户态的 vhost 设备，需要额外占用主机侧的 CPU/内存等资源
- 管理界面无法统一：从架构图上可以看出，vhost-user-net 以及 vhost-user-blk/scsi 设备通过 dpdk/spdk 两个组件管理，无法通过一套统一的方案实现，且当前仅支持存储和网络，virtio-fs 等其他 virtio 设备无法支持
- 热迁移时长过长：当前用户态 vDPA 虚拟机热迁移中断时间较长（超过 1s），导致业务在升级运维过程中出现明显的卡顿现象。

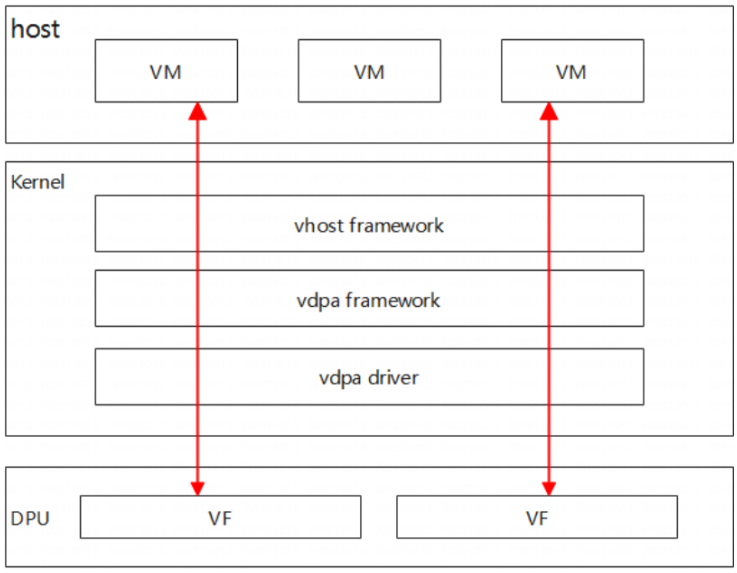


功能描述

与用户态 dpdk 相同，openEuler 构建的内核态 vDPA 利用内核中现有成熟的 vhost 子系统，为上层用户态软件提供 API 接口。

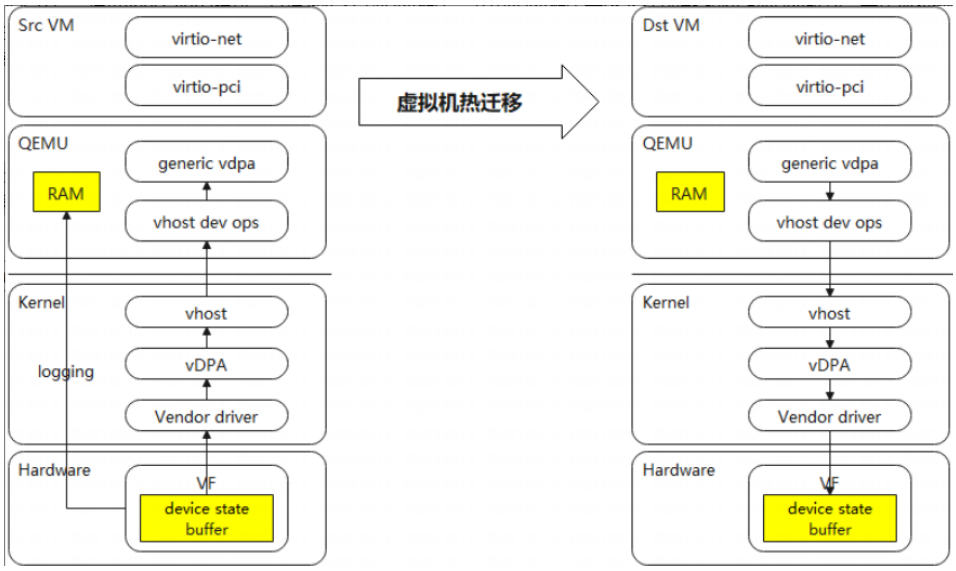
vhost 模块收到 qemu 下发的 IOCTL 后，通过 vhost-vdpa 模块，将请求转发至 vDPA 框架，并由 vDPA 框架最终下发到设备绑定的 vDPA 驱动，来实现对硬件的配置。

在这种方案下，主机侧不在需要额外用户态的进程占用额外的 CPU/内存等资源，影响主机侧的资源使用率，同时由于内核态 vDPA 不关注 VF 的设备类别，virtio-net/blk/scsi/fs 等 virtio 设备，可以通过同一套方案呈现给虚拟机方案统一。



在此基础框架之上通过补充下述能力实现内核态 vDPA 虚拟机高效热迁移：

- vDPA 设备硬件标脏：vDPA 虚拟设备支持硬件做脏页标记，实现内存标脏，确保虚机热迁移前后，源端主机和目的端主机的内存保持一致；
- vDPA 设备暂停/恢复：支持 vDPA 虚拟设备在虚机热迁移过程中的短暂暂停和恢复，以使虚机热迁移后快速恢复；
- vDPA 设备硬件状态迁移：支持 vDPA 设备所对应的硬件状态由源端主机迁移至目的端主机并保持一致。



应用场景

vDPA 方案可应用于云计算中智能网卡、DPU 卸载场景，使用该方案可以实现主机侧管理资源全卸载，支持网络、存储等设备形态，提升主机侧可售卖资源，降低云厂商成本。同时支持 vDPA 虚拟机高效热迁移能力，降低虚拟机迁移过程对业务的中断影响，实现高效热运维。

仓库地址

https://gitee.com/openeuler/qemu

https://gitee.com/openeuler/kernel

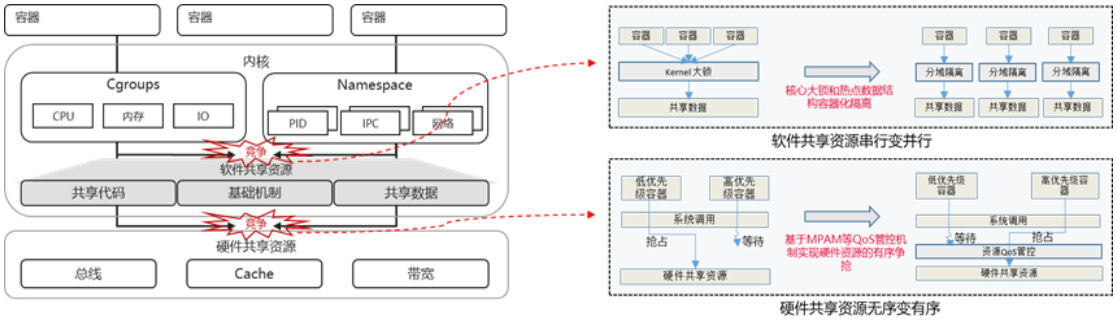
云原生&虚拟化

众核高密

SIG	Kernel

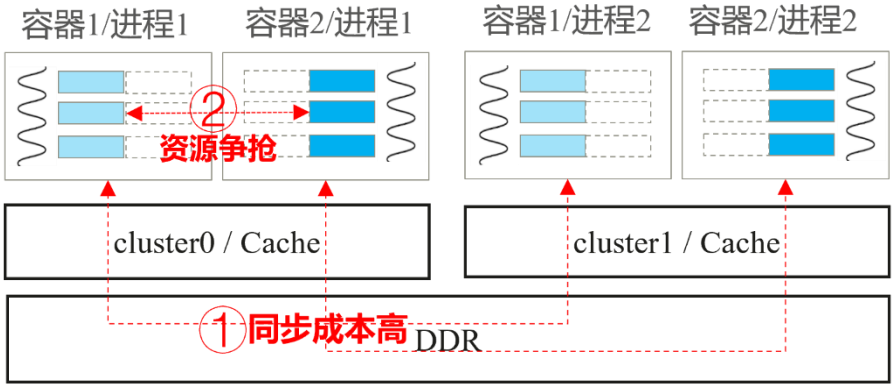
服务器芯片由多核进入众核时代（>256C），对操作系统提出新的挑战。提升 Rack 计算密度、降低数据中心 TCO，众核服务器已成为互联网行业主流选择，随着云技术和业务规模发展，容器化部署成为互联网行业的主流业务部署形态，在这种场景下，系统串行开销和同步开销限制可扩展性，干扰问题凸显，资源利用率低，影响容器部署扩展性的串行访问开销和同步开销主要来自软硬共享资源争用。

功能描述

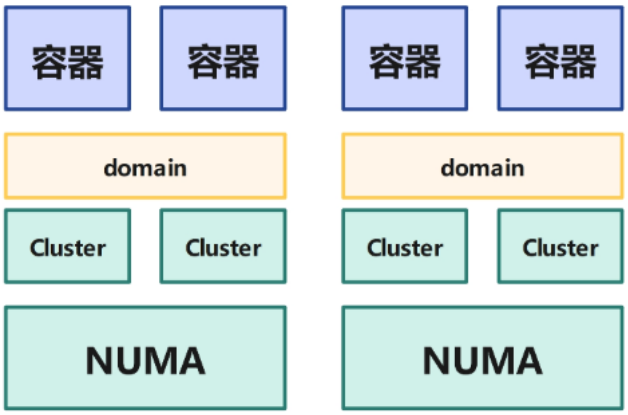


openEuler 主要采用轻量虚拟化按 NUMA 分域拆分资源、域内实现资源容器级隔离增强，降低因软硬件资源争用导致的性能干扰，提升容器部署扩展性。关键技术特性如下：

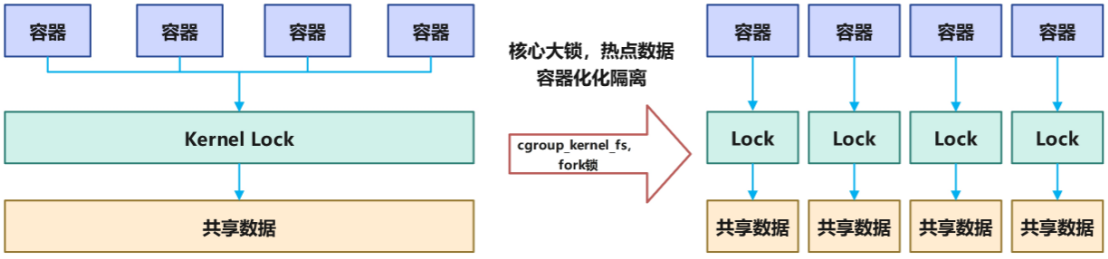
- 轻量虚拟化：虚拟设备中断卸载实现虚拟 virtio 设备中断卸载至硬件注入，降低存储虚拟化损耗；PMD 队列负载均衡实现 virtqueue 队列从单 reactor 均衡分散到多 reactor，消除 cpu 瓶颈；
- CPU 分域调度：CPU 基于硬件拓扑划分子域部署容器，一个容器一个独立子调度域，实现容器之间干扰隔离，降低跨 cluster cache 同步次数和 cache/NUMA 内存等硬件资源争抢，减轻容器之间的相互干扰。对于 redis 多并发场景性能提升 10%+；



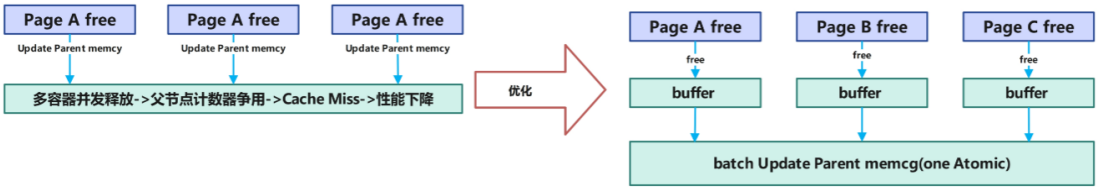
1. 容器内部跨 cluster 共享数据：同一个容器内的业务分散运行在多个 CPU cluster 上，cluster 间 cache 同步和跨 cluster 访问时延造成业务性能抖动；
2. 容器之间的资源竞争：多个容器业务的任务竞争 cache/内存带宽等资源，造成业务间的相互干扰和资源利用不均衡。



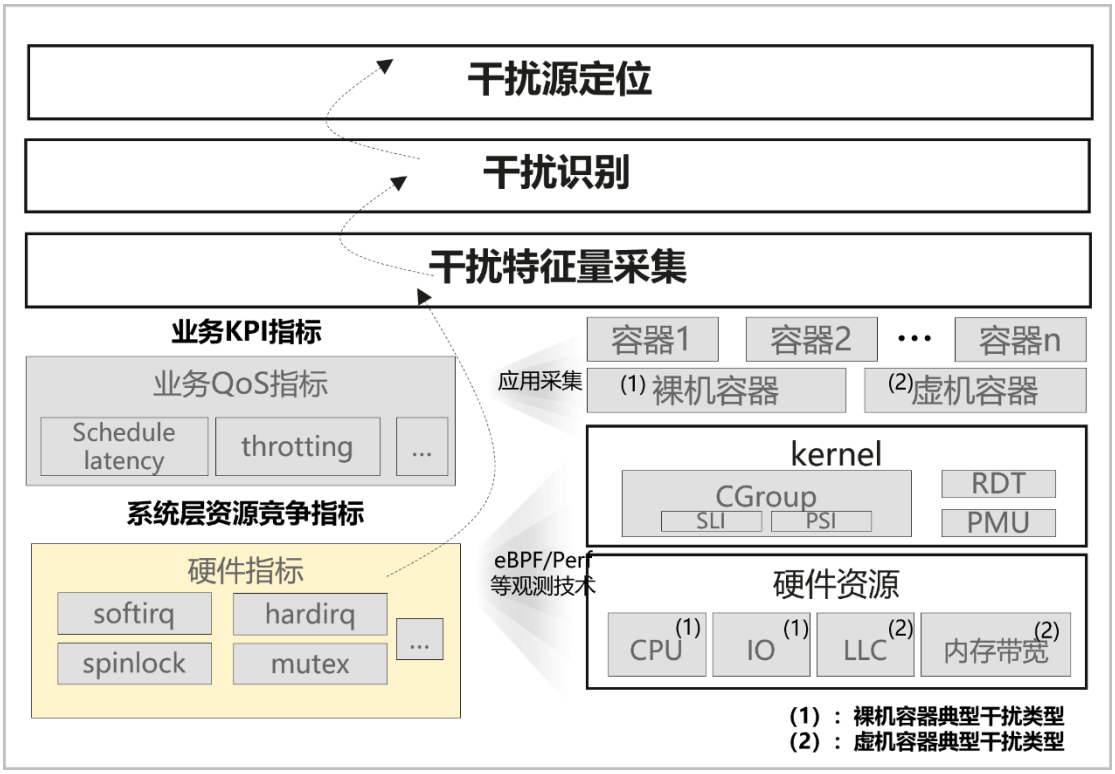
- 文件系统块分配干扰隔离：优化 ext4 块分配释放流程中的 group lock 和 s_md_lock 两个主要争抢的锁，以提高 EXT4 块分配流程的可扩展性。通过允许在当前目标块组被占用时尝试使用其他空闲块组进行分配，从而减少了多个容器争抢同一个块组造成的 CPU 浪费，并充分利用了 ext4 多块组的优势，缓解了 group lock 的竞争。其次通过将流式配的全局目标拆分到 inode 级别，从而减少了全局锁 s_md_lock 的竞争，文件数据也更加聚集。在 64 容器并发场景下，块分配和块释放混合场景 OPS 提升 5 倍以上，单块分配场景提升 10 倍以上。



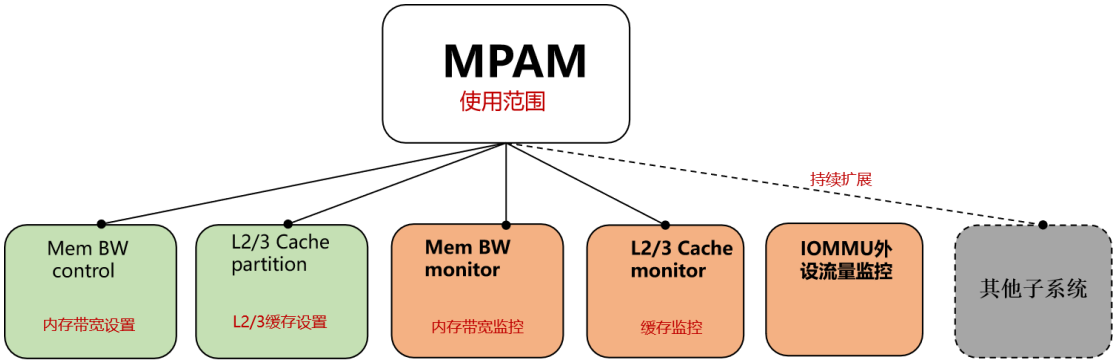
- 网络 tcp hash 干扰隔离：tcp_hashinfo bash、ehash 存在锁竞争，ehash 计算频繁，导致高并发下带宽下降，时延变大。将 tcp_hashinfo bash、ehash 的自旋锁改为 rcu，ehash 计算方式改为 lport 递增减少查询时间和计算次数，减少 tcp connect hash 的锁竞争。
- Cgroup 隔离增强：user namespace 通过 percpu counter 替换原来的原子操作，避免不同 namespace 相同父节点竞争访问，消除容器间 rlimit 计数干扰。 解决 will-it-scale/ signal1 用例线性度问题，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；



- 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。



MPAM 提供 cache 和 MB 控制和监控能力

支持静态+动态资源管控方式,提升资源利用率

1. 静态限制离线业务带宽，在线业务不受影响，隔离效果明显。实际场景不同负载下离线业务带宽流量通常占用系统总带宽 5%~20%之间，MPAM 控制精度在 1%左右
2. 动态调整离线业务带宽，在线业务低谷时，放开离线业务限制，避免对离线业务资源过度压制导致资源浪费

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。

NestOS 云底座操作系统

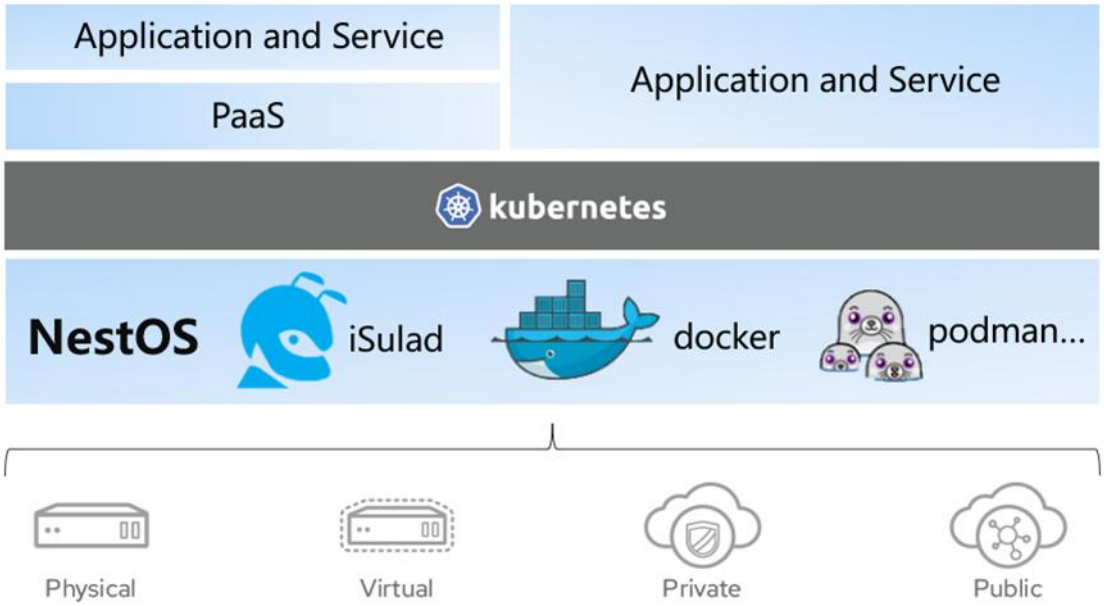
SIG CloudNative

NestOS 是在 openEuler 社区孵化的云底座操作系统,集成了 rpm-ostree 支持、ignition 配置等技术。采用双根文件系统、原子化更新的设计思路，使用 nestos-assembler 快速集成构建，并针对 K8S、OpenStack 等平台进行适配，优化容器运行底噪，使系统具备十分便捷的集群组建能力，可以更安全的运行大规模的容器化工作负载。

技术挑战

云原生场景中，容器和 kubernetes 等相关技术的应用越来越广泛，随之而来出现了多种多样的容器运行时及相关管理软件，因此在实际使用中容易出现，容器技术与容器编排技术实现业务发布、运维时与底层环境高度解耦而带来的运维技术栈不统一，运维平台重复建设等问题。

功能描述



1. 开箱即用的容器平台：NestOS 集成适配了 iSulad、Docker、Podman 等主流容器引擎，为用户提供轻量级、定制化的云场景 OS。
2. 简单易用的配置过程：NestOS 通过 ignition 技术，可以以相同的配置方便地完成大批量集群节点的安装配置工作。
3. 安全可靠的包管理：NestOS 使用 rpm-ostree 进行软件包管理，搭配 openEuler 软件包源，确保原子化更新的安全稳定状态。
4. 友好可控的更新机制：NestOS 使用 zincati 提供自动更新服务，可实现节点自动更新与重新引导，实现集群节点有序升级而服务不中断。
5. 紧密配合的双根文件系统：NestOS 采用双根文件系统的设计实现主备切换，确保 NestOS 运行期间的完整性与安全性。

应用场景

NestOS 适合作为以容器化应用为主的云场景基础运行环境，引入社区孵化项目 NestOS-Kubernetes-Deployer，辅助 NestOS 解决在使用容器技术与容器编排技术实现业务发布、运维时与底层环境高度解耦而带来的运维技术栈不统一，运维平台重复建设等问题，保证了业务与底座操作系统运维的一致性。

仓库地址

https://gitee.com/openeuler/NestOS

KubeOS 容器操作系统

SIG CloudNative

KubeOS 是面向云原生场景的容器操作系统，通过 Kubernetes 统一纳管容器和节点 OS，提供 API 运维化、原子化、轻量安全的云原生场景下集群 OS 的运维方案。

技术挑战

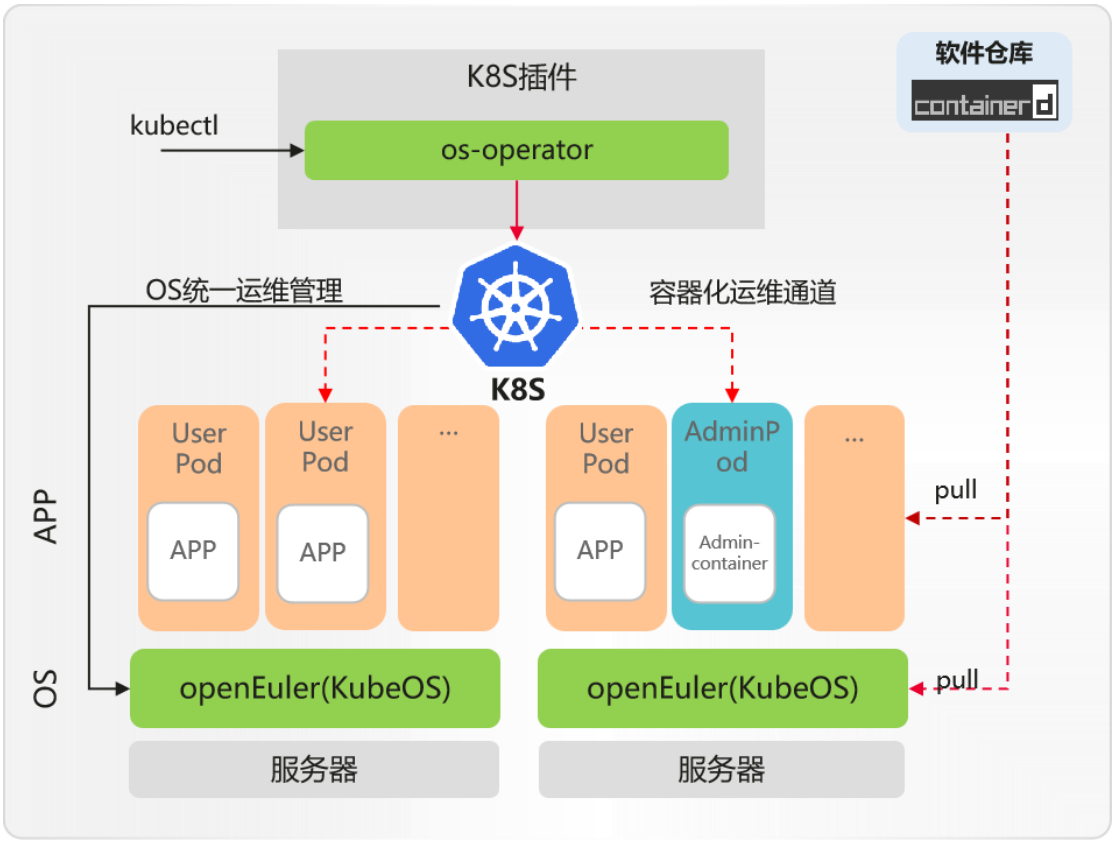
传统操作系统是面向通用场景设计，对云原生场景运行负载单一，使用 K8S 管理负载等情况考虑不足，随之而来的就是云原生场景下的 OS 的管理问题：操作系统的云原生改进，是提升 OS 管理和运维效率的有效方向。

1. 云原生场景下应用纷纷容器化，对 OS 有着新的挑战，传统的 OS 形态过重，不再完全适用。
2. 容器和 OS 的运维管理的分别独立进行，往往会出现管理功能冗余，两套调度系统协调困难的问题。
3. 单独的包管理会导致集群容器 OS 版本零散状态不统一等问题，缺乏统一的容器 OS 管理方式，容器 OS 的版本管理困难。

为解决以上问题，openEuler 提出了 KubeOS，一套通过 Kubernetes 统一纳管容器和 OS 的 OS 运维方案。

功能描述

KubeOS 基于 openEuler 提供了一套云原生场景下 OS 运维管理的解决方案。基于 openEuler 构建轻量化的容器操作系统，并将 OS 作为组件接入到 Kubernetes 中，使得可以通过 Kubernetes 统一对容器和 OS 进行原子化运维管理。



KubeOS 的主要特性如下:

1. 统一管理:KubeOS 将 OS 作为组件接入到集群中,使用 kubernetes 统一管理 OS 和业务容器,统一管理所有节点 OS。
2. 协同调度:OS 变更前感知集群状况,实现业务容器和 OS 的协同调度。
3. API 运维:使用 kubernetes 原生的声明式 API 管理运维 OS,运维通道标准化。
4. 原子管理:结合 kubernetes 生态,实现 OS 的原子升级/回滚能力,保证集群节点一致性。
5. 轻量安全:仅包含容器运行所需组件,减少攻击面和漏洞,提高安全性,降低 OS 运行底噪和重启时间,只读根文件系统,保证系统不被攻击和恶意篡改。

应用场景

KubeOS 主要应用在云原生场景的基础设施中,提供云服务的基础运行环境,助力云厂商,通信行业客户解决云原生场景 OS 运维难题。

仓库地址

https://gitee.com/openeuler/KubeOS

Kmesh 高性能服务治理框架

SIG ebpf

Kmesh 是一种高性能服务网格数据面软件，基于可编程内核，将流量治理逻辑从代理程序下沉到操作系统，实现流量路径多跳变为一跳，大幅提升服务网格下应用访问性能。

技术挑战

随着越来越多的应用云原生化，云上应用的规模、应用 SLA 诉求等都对云基础设施提出了很高的要求。数据中心集群规模越来越大，数据规模呈爆炸式增长，如何高效地实现数据中心内微服务间的流量治理一直是大家关心的问题。

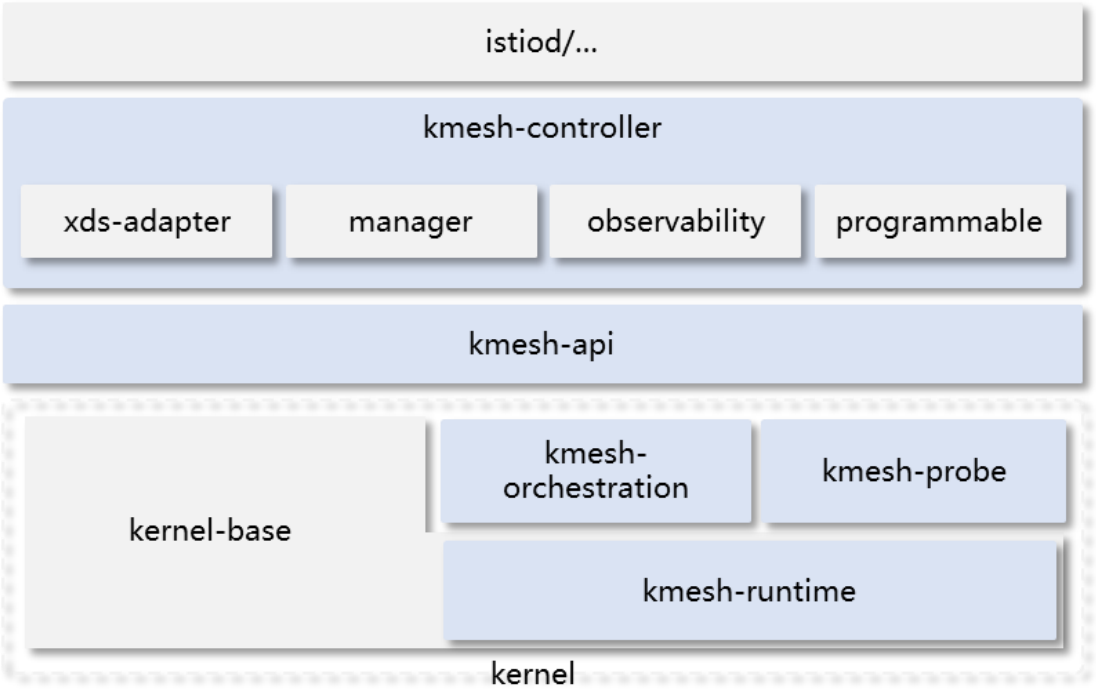
服务网格（serviceMesh）作为下一代微服务技术，将流量治理从服务中剥离出来，下沉到网格基础设施中，很好地实现了应用无感的流量编排；但其代理架构引入了额外的时延底噪开销（例如业界典型软件 istio，单跳服务访问时延增加 2~3ms），无法满足时延敏感应用的 SLA（Service Level Agreement）诉求。

如何实现应用无感的高性能流量治理，是当前面临的技术挑战。

功能描述

Kmesh 基于可编程内核，将流量治理下沉操作系统，实现高性能服务网格数据面；Kmesh 当前支持的主要特性包括：

1. 支持对接遵从 XDS 协议的网格控制面（如 istio）
2. 流量编排能力
    a. 负载均衡：支持轮询等负载均衡策略
    b. 路由：支持 L7 路由规则
    c. 灰度：支持按百分比灰度方式选择后端服务策略



如上 Kmesh 软件架构图所示，其主要部件包括：

1. kmesh-controller：Kmesh 管理程序，负责 Kmesh 生命周期管理、XDS 协议对接、观测运维等。
2. kmesh-api：Kmesh 对外提供的 API 接口层，主要包括 XDS 转换后的编排 API、观测运维通道等。
3. kmesh-runtime：kernel 中实现的支持 L3~L7 流量编排的运行时。
4. kmesh-orchestration：基于 eBPF 实现 L3~L7 流量编排，如路由、灰度、负载均衡等。
5. kmesh-probe：观测运维探针，提供端到端观测能力。

应用场景

Kmesh 适用于电子商务、云游戏、在线会议、短视频等时延敏感应用。http 测试场景下对比业界方案（istio）转发性能提升 5 倍。

仓库地址

https://gitee.com/openeuler/Kmesh

iSulad 轻量级容器引擎

SIG	iSulad

iSulad 是一个由 C/C++编写实现的轻量级容器引擎，具有轻、灵、巧、快的特点，不受硬件规格和架构限制，底噪开销更小，可应用的领域更为广泛。

技术挑战

容器是一种创建隔离环境，方便高效打包和分发应用的技术。由于其相比于虚拟化技术具备更高的分发效率，以及更小的运行开销，有效提升了开发和部署的效率，这使得越来越多的用户选择使用容器。随着 Docker 容器引擎、Kubernetes 容器编排调度，以及云原生概念的提出，容器生态越来越完善，容器技术也得以快速推广。

然而，随着容器技术的发展，用户对容器的需求场景越来越多样化。

1. 用户对于容器的启动速度和部署速度要求越来越高。
2. 用户对容器的资源开销要求越来越高。
3. 物联网、边缘计算领域的蓬勃发展对容器技术提出了新要求。

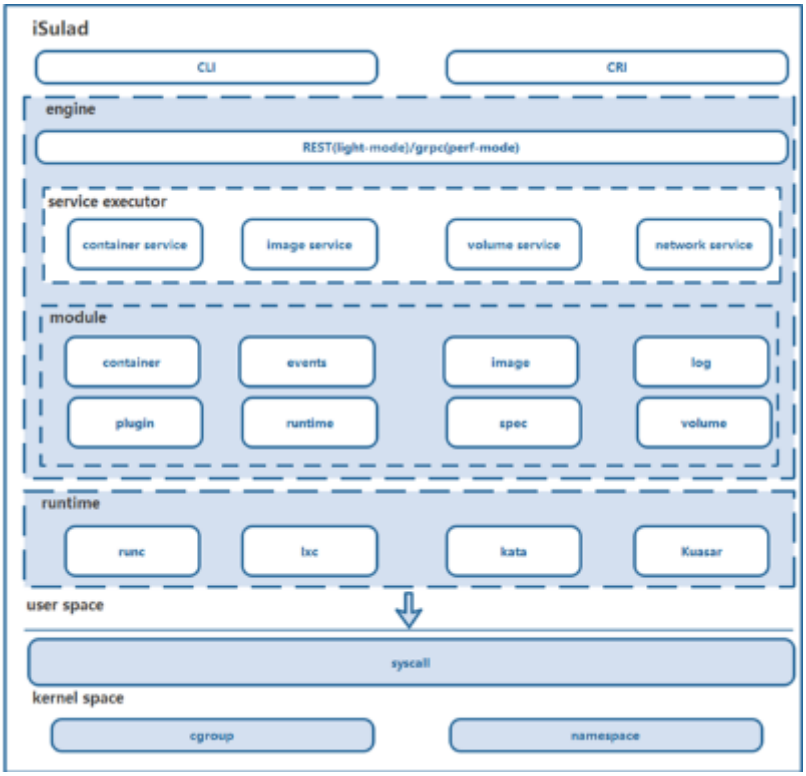
基于这种背景，openEuler 提出了 iSulad 容器解决方案，一种更加轻量、快速的容器引擎。

功能描述

iSulad 是 openEuler 提供的新的容器引擎，其统一的架构设计能够满足 CT 和 IT 领域的不同需求。相比 Golang 编写的 Docker，iSulad 资源占用更少，容器启动更快，可应用范围更广。

iSulad 的名字来自于南美的子弹蚁，其个头虽然小，但是力量巨大，被它咬一口，犹如被子弹打到那般疼痛，它是世界上最强大的昆虫之一。iSulad 也是如此，其虽然轻量，能力却不弱，可以为多种场景提供灵活、稳定、安全的底座支撑，与子弹蚁的形象不谋而合。

iSulad 容器引擎提供了与 Docker 类似的命令行，方便用户操作使用。其北向支持 CRI 接口，可以对接 Kubernetes，用户可以使用 iSulad 作为底座，通过 Kubernetes 进行容器的编排调度。iSulad 南向支持 OCI runtime 标准，能够灵活对接 runc、lxc、kata、kuasar 等多种容器运行时，兼容容器生态。



iSulad 软件架构

iSulad 核心能力，包括容器服务，镜像服务、卷服务以及网络服务。容器服务，用来负责容器生命周期的管理。镜像服务，负责提供对容器镜像的操作。iSulad 支持符合 OCI image 标准的镜像格式，保证 iSulad 能够支持业界主流镜像。此外，iSulad 还支持用于系统容器场景的 external rootfs 以及嵌入式场景的 embedded 镜像格式。卷服务，为用户提供容器数据卷管理的能力。网络服务，可以与符合 CNI 标准的网络插件一起，为容器提供网络能力。

iSulad 作为一款通用容器引擎，除了支持运行普通容器之外，还支持运行系统容器与安全容器。

1. 普通容器：传统的应用容器
2. 系统容器：在普通容器基础上的功能扩展，相比较普通容器，系统容器具备 systemd 管理服务的能力，支持在容器运行时动态添加/释放磁盘设备、网卡、路由以及卷。系统容器主要应用在重计算、高性能、大并发的场景下，可以解决重型应用和业务云化的问题。
3. 安全容器：安全容器是虚拟化技术和容器技术的结合，相比于普通容器共用同一台宿主机内核存在的安全隐患，安全容器通过虚拟化层实现容器间的强隔离，每个安全容器都有一个自己单独的内核和轻量级虚拟机运行环境，保证同一个宿主机上不同安全容器的运行互相不受影响。

与 Docker 相比，iSulad 不仅在容器启动速度上更快，而且在资源开销方面也更低。这是因为 iSulad 是通过 C/C++实现的，相比于其他语言其运行开销更小。其次，iSulad 在代码层面对调用链路进行了优化，相较于 docker 多次 fork 及 exec 调用二进制的方式，iSulad 较少调用次数，直接通过链接

库的方式进行函数调用，减少调用长度，从而使得其容器启动速度更快。此外，由于 C 语言是天然的系统级编程语言，使得在嵌入式、边缘测等终端设备上，Golang 实现的 Docker“望洋兴叹”，而 iSulad 却可以"大显身手"。

经过实验测试，iSulad 底噪开销仅为 Docker 的 30%，在 Arm 及 x86 环境下，iSulad 并发启动 100 个容器时间相较于 Docker 提升了 50%+。这使得用户使用 iSulad 进行业务部署时，不仅能够更快的启动业务容器，同时可以减少容器引擎带来的额外资源开销，避免其影响业务的正常性能。

应用场景

作为一款轻量级容器引擎，iSulad 在云计算、CT、嵌入式、边缘侧等场景应用广泛，应用场景覆盖银行、金融、通信、云业务等，助力客户提升容器启动性能 50%+。此外，iSulad+Kuasar+StratoVirt 方案也正在推进，共同构建 openEuler 社区的全栈安全容器解决方案。

仓库地址

https://gitee.com/openeuler/iSulad

StratoVirt 高可靠虚拟化引擎

SIG Virt

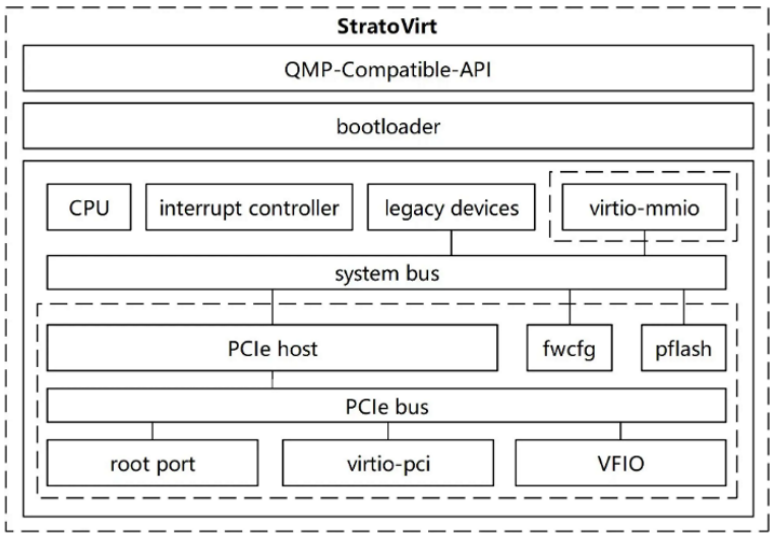
StratoVirt 是计算产业中面向云数据中心的企业级虚拟化平台，实现了一套架构统一支持虚拟机、容器、Serverless 三种场景。StratoVirt 在轻量低噪、软硬协同、Rust 语言级安全等方面具备关键技术竞争优势。

技术挑战

随着近几十年 QEMU 虚拟化软件的发展，核心开源组件代码规模越来越大，其中包含大量陈旧的历史代码，同时近年来 CVE 安全漏洞频出，安全性差、代码冗余、效率低问题越来越明显。业界逐步演进出以内存安全语言 Rust 实现的 Rust-VMM 等架构。安全、轻量、高性能的全场景（数据中心、终端、边缘设备）通用的虚拟化技术是未来的趋势。StratoVirt 作为 openEuler 开源平台上实现的下一代虚拟化技术应运而生。

功能描述

StratoVirt 是一种基于 Linux 内核虚拟化（KVM）的开源轻量级虚拟化技术，在保持传统虚拟化的隔离能力和安全能力的同时，降低了内存资源消耗，提高了虚拟机启动速度。StratoVirt 可以应用于微服务或函数计算等 Serverless 场景，保留了相应接口和设计，用于快速导入更多特性，直至支持通用虚拟化。



StratoVirt 软件架构

StratoVirt 的核心架构如下图所示，从上到下分为三层：

1. 外部 API: StratoVirt 使用 QMP 协议与外部系统通信，兼容 OCI，同时支持对接 libvirt；
2. bootloader: 轻量化场景下使用简单的 bootloader 加载内核镜像，而不像传统的繁琐的 BIOS 和 Grub 引导方式，实现快速启动；通用虚拟化场景下，支持 UEFI 启动；
3. 模拟主板 microvm：为了提高性能和减少攻击面，StratoVirt 最小化了用户态设备的模拟。模拟实现了 KVM 仿真设备和半虚拟化设备，如 GIC、串行、RTC 和 virtio-mmio 设备；
4. 通用机型：提供 ACPI 表实现 UEFI 启动，支持添加 virtio-pci 以及 VFIO 直通设备等，极大提高虚拟机的 I/O 性能；

应用场景

StratoVirt 配合 iSula 容器引擎和 Kubernetes 编排引擎可形成完整的容器解决方案，支持 Serverless 负载高效运行。

仓库地址

Kuasar 机密容器

SIG CloudNative

Kuasar 是一款支持多种类型沙箱统一管理的容器运行时，可同时支持业界主流的多种沙箱隔离技术，openEuler 基于 Kuasar 统一容器运行时并结合已有 openEuler 生态中 iSulad 容器引擎和 StratoVirt

虚拟化引擎技术，打造面向云原生场景轻量级全栈自研的安全容器，构建极低底噪、极速启动的关键竞争力。

技术挑战

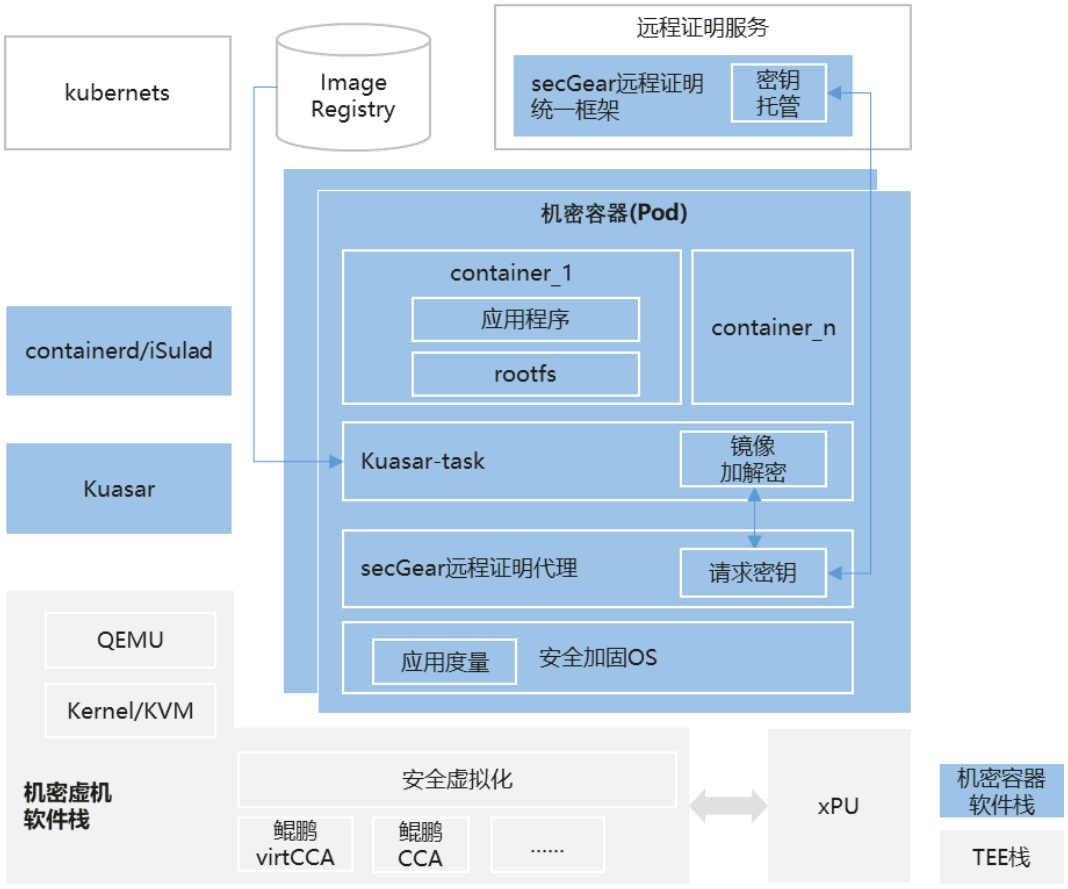
随着云原生技术的广泛应用，如何将云原生应用快速迁移到机密计算环境，在保护容器及容器内数据的机密性和完整性的同时保持和普通容器一致的开发和部署体验，成为亟需解决的问题。

基于这种背景，openEuler 提出 Kuasar，一种支持多种类型沙箱统一管理的容器运行时。

功能描述

Kuasar 机密容器充分利用 Sandboxer 架构优势，提供高性能、低开销的机密容器运行时。

- 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 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。

仓库地址

https://gitee.com/src-openeuler/kuasar

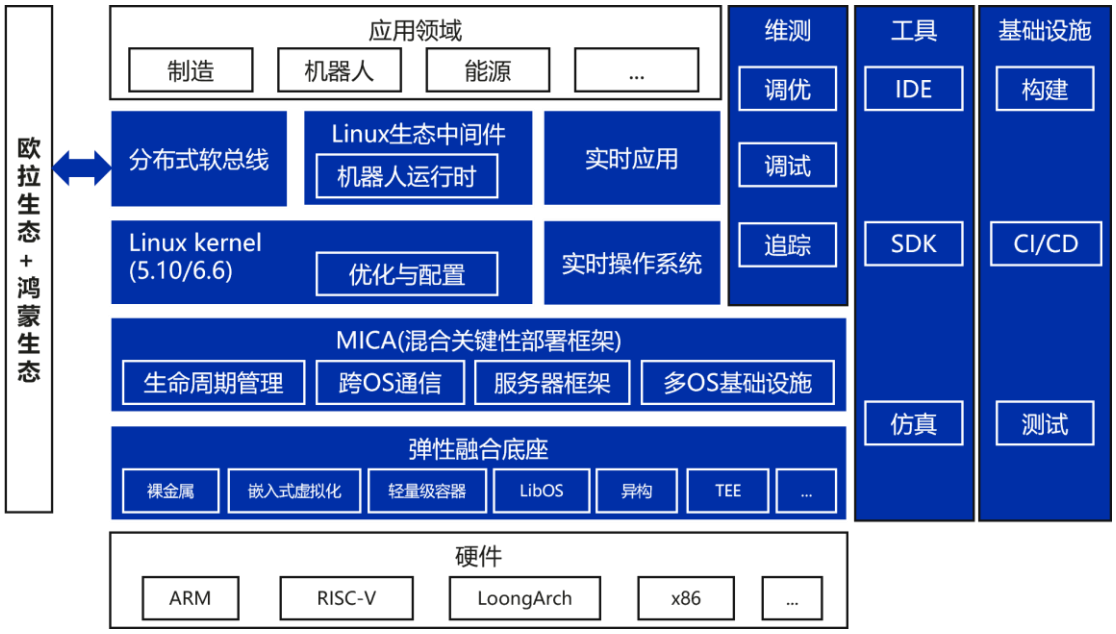
嵌入式

openEuler 面向嵌入式领域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 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：600+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。

边缘计算

openEuler 支持 KubeEdge

SIG Edge

openEuler 发布面向边缘计算的版本 openEuler Edge，集成 KubeEdge 边云协同框架，具备边云应用统一管理和发放等基础能力，并将通过增强智能协同提升 AI 易用性和场景适应性，增强服务协同实现跨边云服务发现和流量转发，以及增强数据协同提升南向服务能力。

技术挑战

边缘计算是未来 10 大战略技术趋势。随着智慧城市、自动驾驶、工业互联网等应用落地，海量数据将在边缘产生，集中式云计算在带宽负载、网络延时、数据管理成本等方面愈发显得捉襟见肘，难以适应数据频繁交互需求，边缘计算价值凸显。

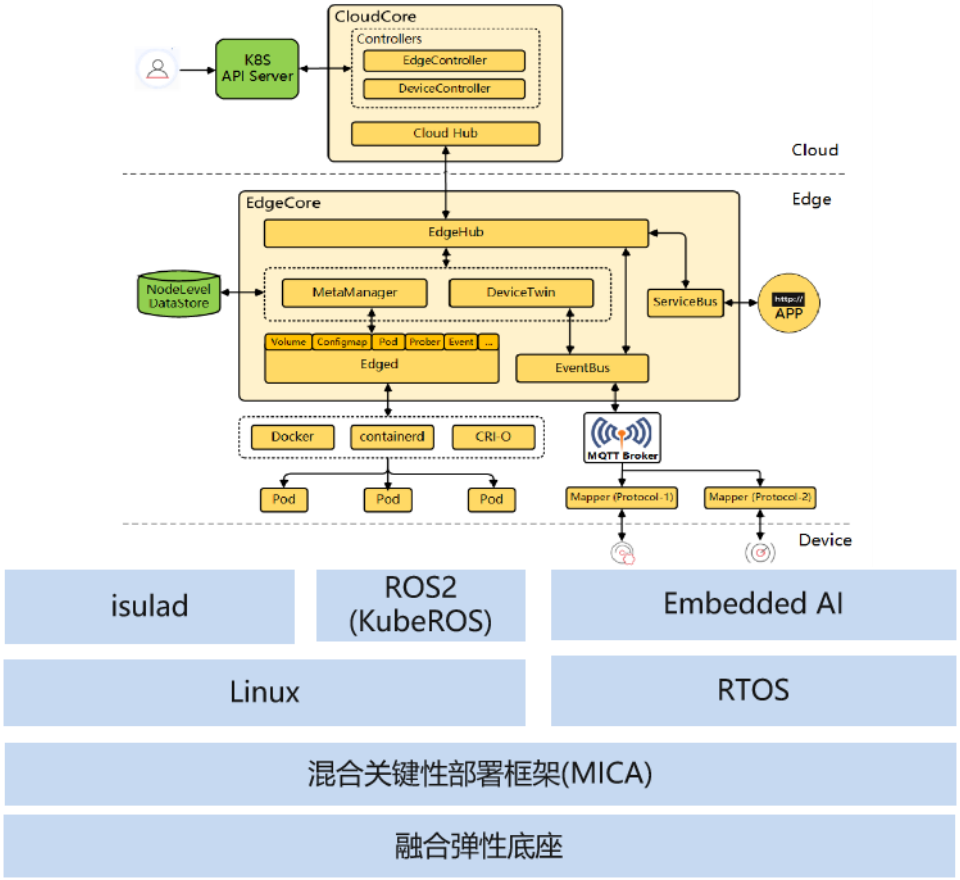
功能描述

边云协同基础框架功能如下：

1. 边云管理协同：实现边云之间的应用管理与部署、跨边云的通信，以及跨边云的南向外设管理等基础能力。
2. 边云服务协同：边侧部署 EdgeMesh Agent，云侧部署 EdgeMesh Server 实现跨边云服务发现和服务路由。
3. 边缘南向服务：南向接入 Mapper，提供外设 Pofile 及解析机制，以及实现对不同南向外设的管理、控制、业务流的接入，可兼容 EdgeX Foundry 开源生态。
4. 边缘数据服务：通过边缘数据服务实现消息、数据、媒体流的按需持久化，并具备数据分析和数据导出的能力。

5. 协同创新：isulad 对接 MICA，弹性底座的更新，支持 RTOS 的容器镜像，实现 KubeEdge 与混合关键性系统的协同创新

应用拓展：KubeEdge，KubeROS， 嵌入式 AI 部署，分布式软总线的结合



关键价值：

降低云边带宽、降低时延：业务逻辑在边侧自闭环，减少了边缘和云的网络消耗，提高响应速度，保护客户数据隐私

方便的从云端管理和部署 AI 应用：轻松的将现有丰富的机器学习、图像识别等应用部署到边侧应用场景

在工厂远程智能监控、工业物联网等需要云边协同的场景，基于 KubeEdge 实现原生容器应用编排功能扩展到边缘节点，方便的从云端管理和部署边侧应用

仓库地址

https://mirror.sjtu.edu.cn/openeuler/openEuler-23.03/edge_img/x86_64/openEuler-23.03-edge-x86_64-dvd.iso

https://github.com/kubeedge/kubeedge

分布式软总线

SIG	distributed-middleware

openEuler 秉承打造“数字化基础设施操作系统”的愿景，为实现端边领域的互通和协同，首次在服务器&边缘&嵌入式领域引入分布式软总线技术。分布式软总线作为分布式设备通信基座，为设备之间的互通互联提供统一的分布式协同能力，实现设备无感发现和数据高效传输。

技术挑战

边端设备之间的互联是实现边端设备协同工作的基石，涉及边端设备的发现、连接、组网、传输等环节。当前边端设备的互联存在以下难点：

1. 端设备形态不一：硬件能力各异，支持的连接方式参差不齐，比如 WiFi、蓝牙、NFC 等多种方式，缺少统一的方案覆盖各种连接方式。
2. 稳定快速组网难：如何在边端设备间自动构建和分配组网管理角色，实现网络的鲁棒性，在设备退出、掉电、故障后仍能保持组网的稳定。
3. 传输性能差：如何达成边端设备间最佳传输性能，特别是当部分端设备对功耗有一定的约束时。
4. 接口适配难：对于上层应用开发者，如何做到提供统一的接口，屏蔽底层硬件、组网的差异，能够让应用开发者不用关心底层实现，聚焦业务流程，实现一次开发、边端复用。

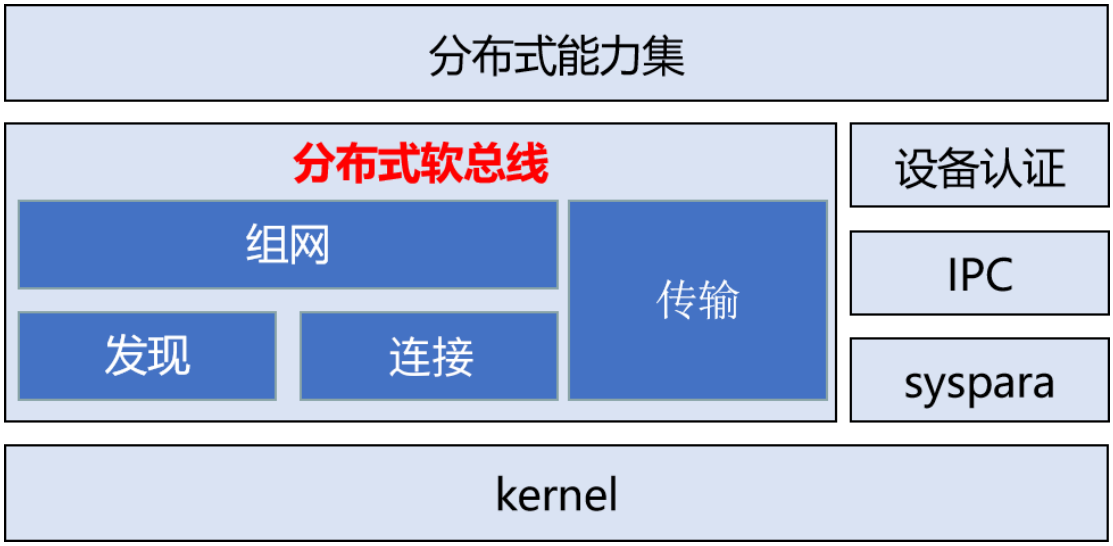
功能描述

主要能力：

1. 发现连接：提供基于 Wifi、有线网络及蓝牙等通信方式的设备发现连接能力。
2. 设备组网：提供统一的设备组网和拓扑管理能力，为数据传输提供已组网设备信息。
3. 数据传输：提供数据传输通道，支持字节、流、文件的数据传输能力。

模块架构

软总线主体功能分为发现、组网、连接和传输四个基本模块。



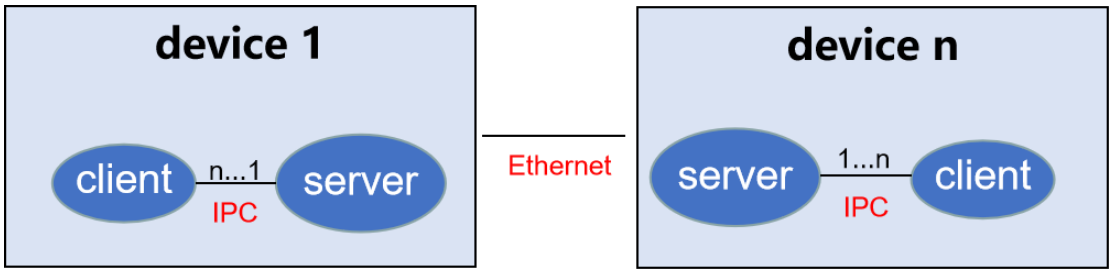
软总线与外部模块架构关系图

软总线南向支持 Wifi、有线网络及蓝牙等通信方式。并为北向的分布式应用提供统一的 API 接口，屏蔽底层通信机制。

软总线依赖于设备认证、IPC、日志和系统参数（SN 号）等周边模块，在嵌入式场景将这些依赖模块进行了样板性质的替换，以实现软总线基本功能。实际的周边模块功能实现，还需要用户根据实际业务场景进行丰富和替换，以拓展软总线能力。

部署示意：

1. 软总线支持局域网内多设备部署，设备间通过以太网通信。
2. 单设备上分为 server 和 client，二者通过 IPC 模块进行交互。
3. 单节点上支持多 client 同时接入单一 server。



部署示意图

如部署模型，软总线通过独立进程部署的方式对外提供服务，通过执行服务端主程序可拉起软总线进程提供对外服务。

应用场景

分布式软总线主要适用于 openEuler 边缘服务器、嵌入式设备及 OpenHarmony 嵌入式设备之间的自发现、互联互通。

多用于工业产线、园区设备管理场景，通过软总线统一接口和协议标准，让不同厂家、不同类型硬件设备之间自连接、自组网、新设备即插即用，实现数据与外设互通访问。

仓库地址

https://gitee.com/openeuler/dsoftbus_standard

5 基础能力创新

内核创新

SIG Kernel

openEuler 基于 Linux Kernel 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，包括但不限于：

- 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，仅仅是记录元数据并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 特性旨在针对 fuse 直接对接后端文件系统（透传）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，同时也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：
    1. 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，为混部场景实现动态调控共享资源提供了端到端能力；
    2. 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。
    3. 新增 L2 缓存隔离配置。提供 L2C 占用量和带宽流量的监控能力，为混部场景的系统性能提供了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。

- 内核多副本（Kernel Replication）特性：Kernel Replication 旨在优化 Linux 内核在 NUMA（非一致性内存访问）架构下的内核性能瓶颈。相关研究表明，Apache、MySQL、Redis 等数据中心关键应用中，Linux 内核态的执行对整体性能影响显著，例如：内核执行占整个应用程序执行 CPU cycles 数目的 61%、总指令执行数目的 57%，占 I-cache 失效率的 61%、I-TLB 失效率的 46%。在传统 Linux 内核中，代码段、只读数据段、内核页表（swapper_pg_dir）仅驻留在主 NUMA 节点中并无法被迁移，这就导致当进程或多线程应用跨多个 NUMA 节点部署时，涉及系统调用存在频繁的跨 NUMA，导致整机访存延时增加，从而影响整机性能。Kernel Replication 特性扩展了 mm_struct 中的 pgd 全局页目录表，在内核启动阶段会自动创建内核代码段、数据段以及对应页表的各 NUMA 节点副本，此机制允许相同的内核虚拟地址在不同 NUMA 节点上映射到所在各自节点的物理地址，从而提升访存的本地局部性、减少跨 NUMA 性能开销。功能完备性上，Kernel Replication 特性同时支持 vmalloc，module 动态加载，Kprobe/KGDB/BPF 等动态指令注入机制，KPTI/KASLR/KASAN 等安全机制，64K 大页等。易用性上，新增内核启动阶段 cmdline 配置选项（默认关闭），可以在 boot 阶段进行动态开关，确保可控性和兼容性。Kernel Replication 适用于高并发、多线程的服务器负载场景。
- HAOC 3.0 安全特性：HAOC (Hardware-assisted OS compartmentalization) 含义是基于硬件辅助的操作系统隔离技术。 基于 x86、ARM 处理器提供的硬件特性，设计复式架构内核，在内核中提供隔离执行环境，为 Linux 内核提供进一步的隔离能力，从而阻止攻击者实施横向移动和权限提升。当前版本提供了隔离执行环境 IEE， 允许后续选项在隔离执行环境中新增需要隔离的敏感资源，并保证这些敏感资源不能被普通的内核代码随意的访问，而只能通过对 IEE 提供的接口进行调用来实现访问。
- 内存管理 folio 特性：Linux 内存管理基于 page（页）转换到由 folio（拉丁语 foliō，对开本）进行管理，相比 page，folio 可以由一个或多个 page 组成，采用 struct folio 参数的函数声明它将对整个（1 个或者多个）页面进行操作，而不仅仅是 PAGE_SIZE 字节，从而移除不必要复合页转换，降低误用 tail page 问题；从内存管理效率上采用 folio 减少 LRU 链表数量，提升内存回收效率，另一方，一次分配更多连续内存减少 page fault 次数，一定程度降低内存碎片化；而在 IO 方面，可以加速大 IO 的读写效率，提升吞吐。全量支持匿名页、文件页的 large folio，提供系统级别的开关控制，业务可以按需使用。对于 ARM64 架构，基于硬件 contiguous bit 技术（16 个连续 PTE 只占一个 TLB entry），可以进一步降低系统 TLB miss，从而提升整体系统性能。24.03 LTS SP1 版本新增支持 anonymous shmem 分配 mTHP 与支持 mTHP 的 lazyfree，进一步增加内存子系统对于 large folio 的支持；新增 page cache 分配 mTHP 的 sysfs 控制接口，提供系统级别的开关控制，业务可以按需使用。
- ext4 文件系统支持 Large folio：iozone 性能总分可以提升 80%，iomap 框架回写流程支持批量映射 block。支持 ext4 默认模式下批量申请 block，大幅优化各类 benchmark 下 ext4 性能表现（华为贡献）。ext4 buffer io 读写流程以及 pagecache 回写流程弃用老旧的 buffer_head 框架，

切换至 iomap 框架，并通过 iomap 框架实现 ext4 支持 large folio。24.03 LTS SP1 版本新增对于 block size < folio size 场景的小 buffered IO（<=4KB）的性能优化，性能提升 20%。

- xcall/xint 特性：随着 Linux 内核的发展，系统调用成为性能瓶颈，尤其是在功能简单的调用中。AARCH64 平台上的 SYSCALL 共享异常入口，包含安全检查等冗余流程。降低 SYSCALL 开销的方法包括业务前移和批量处理，但需业务适配。XCALL 提供了一种无需业务代码感知的方案，通过优化 SYSCALL 处理，牺牲部分维测和安全功能来降低系统底噪，降低系统调用处理开销。

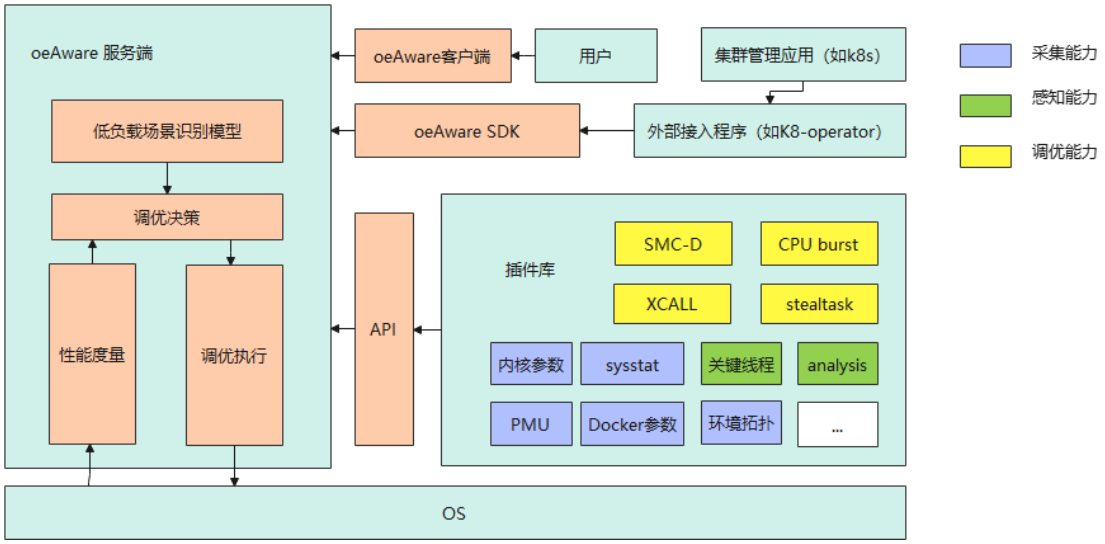
  内核为了使中断处理整体架构统一，将所有中断处理全部归一到内核通用中断处理框架中，同时随着内核版本演进，通用中断处理框架附加了很多与中断处理自身的功能关系不大的安全加固和维测特性，这导致中断处理的时延不确定性增大。xint 通过提供一套精简的中断处理流程，来降低中断处理的时延和系统底噪。

高效并发与极致性能

oeAware 业务场景自适应无感调优

SIG A-Tune

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。



技术挑战

传统调优特性入口分散，调优能力各自开发和推广，难以获取所有调优能力，且部分调优都是一个小闭环，存在开发资源的浪费，oeAware 作为智能调优的出口，用户通过 oeAware 就能完成 openEuler 的调优，随着时间积累，后续的调优可以获得很多现有的输入，只需要聚焦根据现有输入如何建模调优，不需重复造轮子。

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 (4) seep_tune：实时调整 core 频率，降低功耗 (5) dynamic_smt_tune：低负载时通过 smt 队列排队调度，降低 SMT0/1 之间干扰，降低低负载场景的应用之间的干扰率。
- Docker 调优插件：利用 cpuburst 特性在突发负载下环境 CPU 性能瓶颈。
- Numa 调优插件：自动绑核和内存迁移，适用于跨 NUMA 访存瓶颈场景。
- 网卡调优插件：网络 IO 密集型应用性能,如果存在网络 IO 中断性能瓶颈,通过实时监控网络流量与 CPU 之间的亲和性,实时绑定网卡队列中断。
- 透明大页优化：针对 LLC miss 场景较多的应用。
- sysboost：通过 ELF 文件合并优化，减少间接指令跳转。
- 算力统筹：将 host 上剩余的 cpu 资源临时分配给 cpu 资源不足的容器实例。

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。

仓库地址

https://gitee.com/openeuler/oeAware-manager

A-Tune 智能调优引擎

SIG A-Tune

A-Tune 是一款基于 AI 的操作系统性能调优引擎。A-Tune 利用 AI 技术，使操作系统“懂”业务，简化 IT 系统调优工作的同时，让应用程序发挥出色性能。

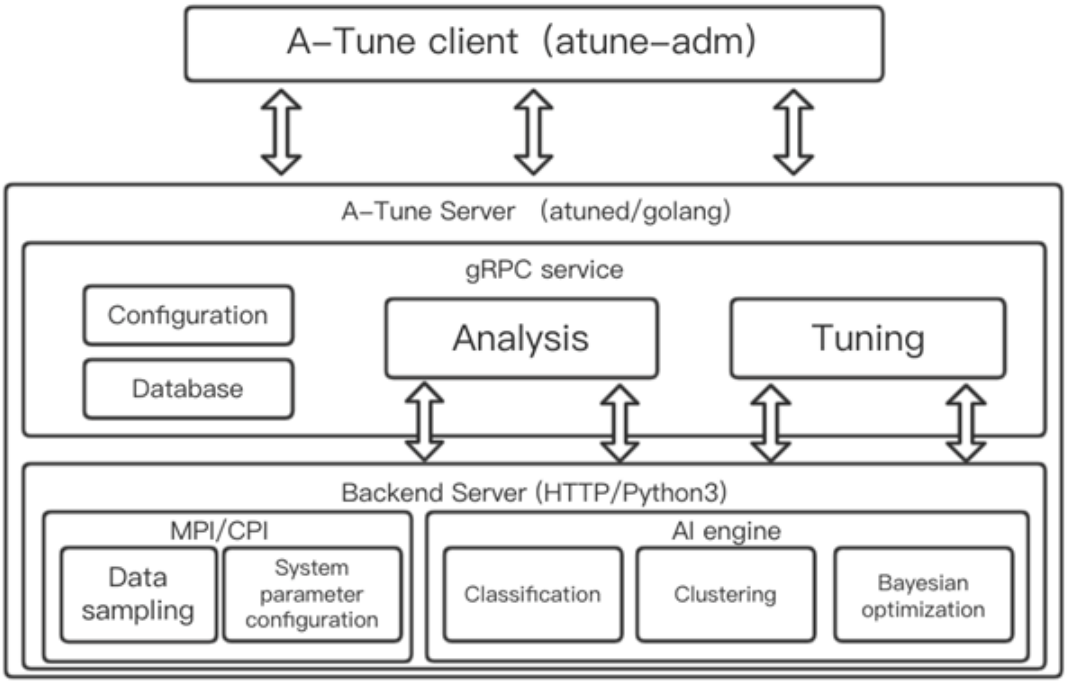
技术挑战

随着几十年来硬件和软件应用的不断发展，Linux 内核正变得越来越复杂，而整个操作系统也变得越来越庞大。在 openEuler 开源操作系统中，仅 sysctl 命令（用于运行时配置内核参数的命令）的参数（sysctl -a | wc -l）就超过 1000 个，而完整的 IT 系统从最底层的 CPU、加速器、网卡，到编译器、操作系统、中间件框架，再到上层应用，可调节参数超过 7000 个。而大部分使用者只使用了这些参数的默认配置，因此无法充分发挥系统最佳性能。然而，针对特定的应用场景进行调优存在以下几方面的难点：

1. 参数数量多，且参数间存在依赖关系
2. 上层应用系统种类多，不同应用系统的参数不同
3. 每个应用的负载也复杂多样，不同负载对应的最优参数值也不同

功能描述

A-Tune 的整体架构如图所示，其整体上是一个 C/S 架构。客户端 atune-adm 是一个命令行工具，通过 gRPC 协议与服务端 atuned 进程进行通讯。服务端中 atuned 包含了一个前端 gRPC 服务层(采用 golang 实现）和一个后端服务层。gRPC 服务层负责优化配置数据库管理和对外提供调优服务，主要包括智能决策（analysis）和自动调优（tuning）。后端服务层是一个基于 Python 实现的 HTTP 服务层，包含了 MPI（Model Plugin Interface）/CPI（Configurator Plugin Interface）和 AI 引擎。其中，MPI/CPI 负责与系统配置进行交互，而 AI 引擎负责对上层提供机器学习能力，主要包括用于模型识别的分类、聚类和用于参数搜索的贝叶斯优化。



A-Tune 软件架构

A-Tune 目前主要提供两个能力：智能决策和自动调优。

智能决策的基本原理是通过采集系统数据，并通过 AI 引擎中的聚类和分类算法对采集到的数据进行负载识别，得到系统中当前正在运行的业务负载类型，并从优化配置数据库中提取优化配置，最终选取适合当前系统业务负载的最优参数配置。具备以下功能：

1. 重要特征分析：自动选择重要特征，剔除冗余特征，实现精准用户画像
2. 两层分类模型：通过分类算法，准确识别当前负载
3. 负载变化感知：主动识别应用负载的变化，实现自适应调优

自动调优的基本原理是基于系统或应用的配置参数及性能评价指标，利用AI引擎中的参数搜索算法，反复迭代，最终得到性能最优的参数配置。具备以下功能：

1. 重要参数选择：自动选择重要的调优参数，减少搜索空间，提升训练效率
2. 调优算法构建：用户可从适用场景、参数类型、性能要求等方面选择最优算法
3. 知识库构建：将当前负载特征和最优参数增加到知识库，提升后续调优效率

应用场景

A-Tune 在 openEuler 等 Linux 环境里面应用比较广泛，应用场景涵盖大数据、数据库、中间件、高性能计算等场景。助力金融、电信等行业客户实现 MySQL、Redis、宝兰德中间件等应用性能提升12%~140%。

仓库地址

https://gitee.com/openeuler/A-Tune

Gazelle 轻量级用户态协议栈

SIG	high-performance-network

Gazelle 是一款高性能用户态协议栈。它基于 DPDK 在用户态直接读写网卡报文，共享大页内存传递报文，使用轻量级 LwIP 协议栈。能够大幅提高应用的网络 I/O 吞吐能力。专注于数据库网络性能加速，如 MySQL、redis 等。

技术挑战

1. **理想协议栈所需的设计原则：**

   报文在 app<->网卡的数据路径的传递过程，要避免拷贝、避免上下文切换、避免 cache miss 等情况。要充分利用网卡多队列、CPU 多核这类横向扩展的硬件能力，同时也要避免多核之间数据访问竞争问题。应用的网络线程模型是多样化的，要兼顾多种应用线程模型。

2. **理想协议栈应有特性：**

   性能方面体现出零拷贝、无上下文切换、避免 cache miss 等技术特点。线性度方面体现出分布式多核部署，无锁，独立协议栈上下文等技术特点。通用性方面体现出兼容不同应用网络模型特点，提供统一抽象层，避免应用面对协议、硬件的复杂性。

3. **当前现状：**

   内核协议栈：注重通用性的基础上，逐步补充加速技术，但是性能始终是硬伤。

   用户态协议栈：一切都为了性能，牺牲的是应用通用性、适装性。

功能描述

1. **功能特性：**

   高性能：

   零拷贝：基于 dpdk 实现报文直通应用

   无锁协议栈：基于 lwip 的无锁线程协议栈，支持线性扩展

   自适应调度：流量均衡调度，算力释放最大化

   动态绑核：应用/协议栈线程动态绑定，实现就近访问

   硬件卸载：CSUM 等功能硬件卸载，提升协议栈处理性能

   通用性：

   POSIX 兼容

   通用网络模型

   兼容内核协议栈

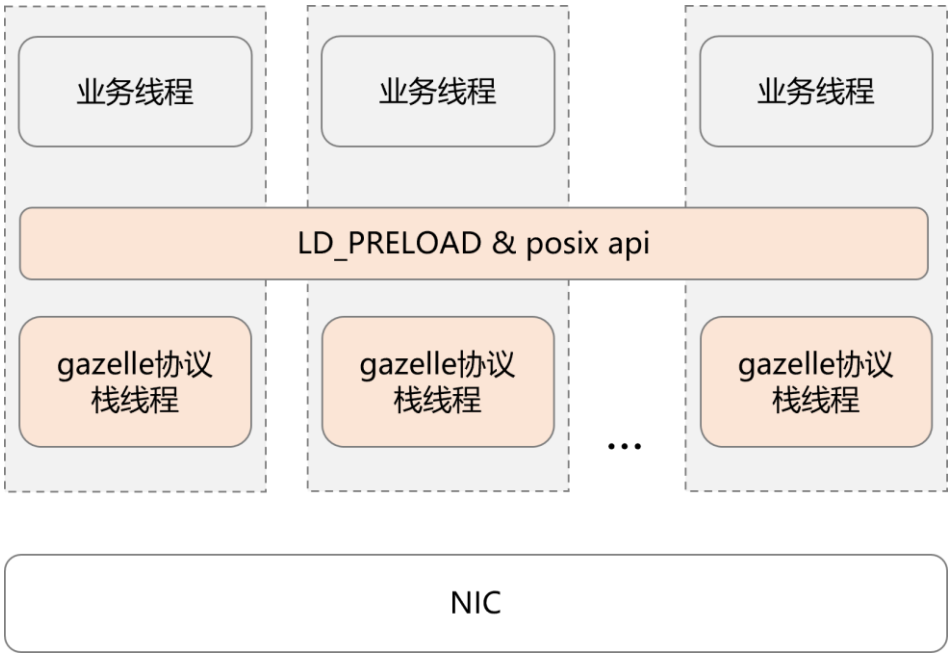
   免配套：

   应用免修改：零配套直接使用

gazelle 架构模型

2. **RTC 模式：**

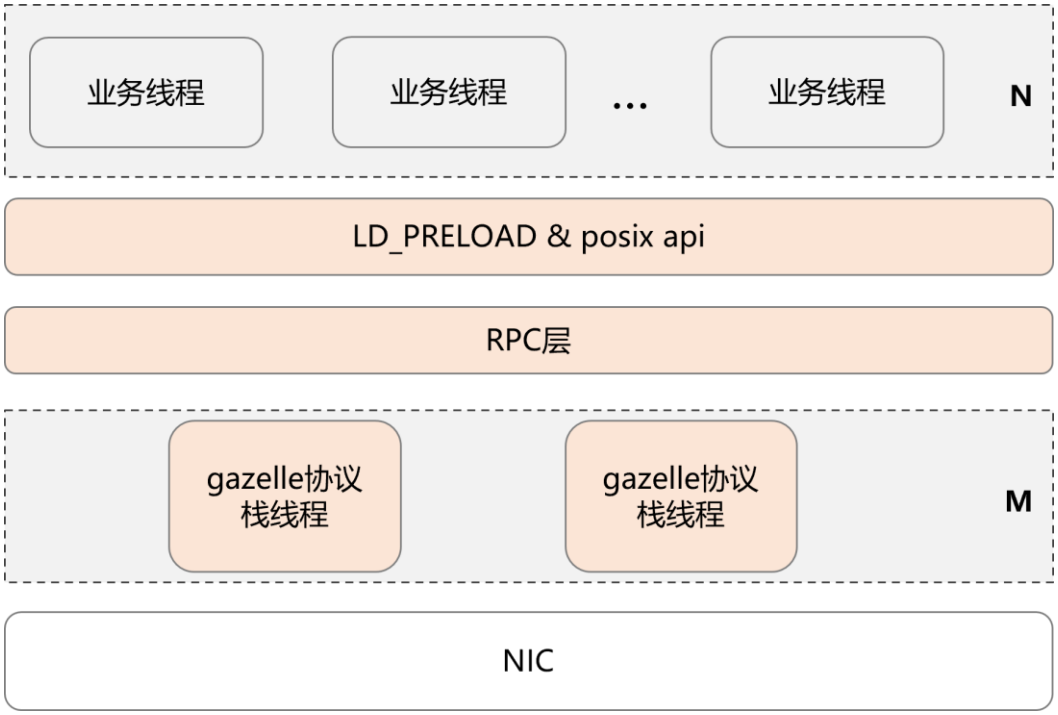
当业务网络线程数量不多，数量固定。业务线程 fd 不跨线程使用，也无阻塞调用时，使 gazelle 运行在 RTC 模式 (共线程模式)。此时，业务和协议栈在一个上下文运行，业务执行到 poll/epoll 时，在其内执行协议栈轮询收包。独占 cpu，不需要唤醒调度，可收获极致性能。



RTC 模式线程模型

3. **RTW 模式：**

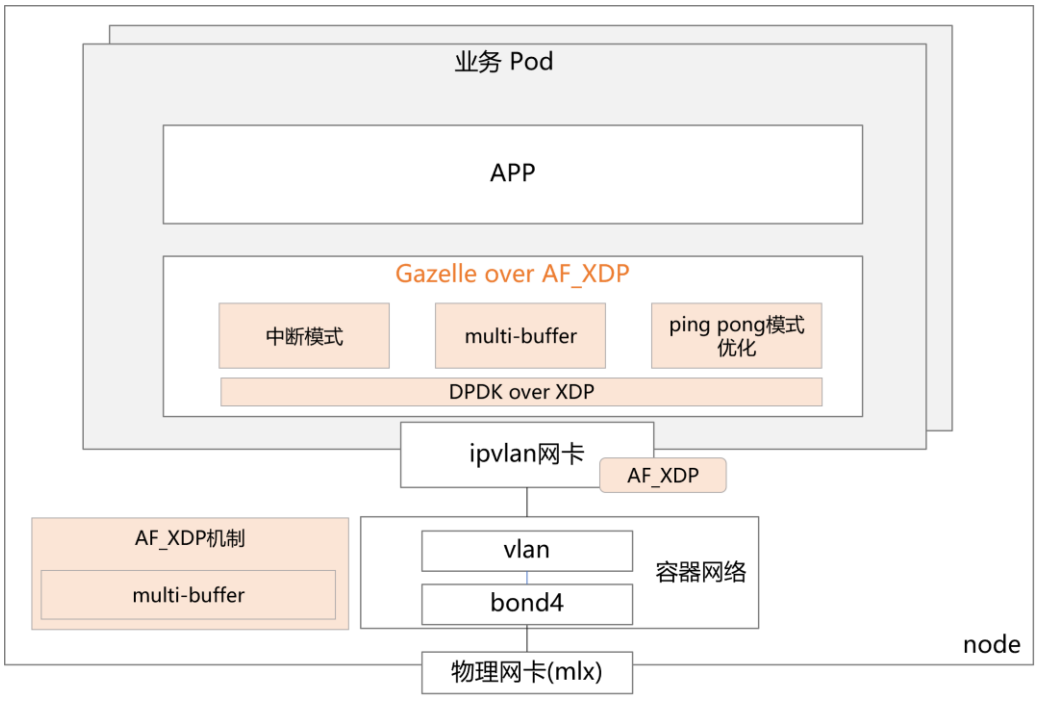
一般场景中，即当业务线程数量很多，fd 会跨线程使用，使 gazelle 运行在 RTW 模式（独立线程模式)。该模式下，协议栈线程和业务线程分离。当收到请求时，由协议栈触发并唤醒相应的业务线程，类似于 Linux 内核协议栈的软中断机制。业务线程 recv/send 通过无锁队列读写报文数据，不与协议栈产生锁竞争。其他控制面 socket 请求通过 rpc 发送到协议栈线程。



RTW 模式线程模型

4. **over AF_XDP 模式：**

在云原生容器网络场景中，通过使用该模式，可在突破网络瓶颈的同时，实现资源解耦。具体为：1.无需独占网卡，利用 AF_XDP 实现网卡队列直通用户态；2.无需独占 CPU，支持中断模式，负载较轻时，降低 CPU 使用率。



RTW 模式线程模型

应用场景

Gazelle 在数据库、中间件等组件，以及整机或容器化场景中，均可实现“部署即见效”的性能提升。其中，在整机 Redis-benchmark 测试中，对比内核态，gazelle 在 SET 和 GET 测试项中，均有 1.6~3.5 倍的性能提升。在 MySQL TPCC 测试中，也能收获 10%-20%的性能提升。

由于 Gazelle 在很大程度上突破了传统用户态协议栈在性能、兼容性和部署灵活性等方面的业界应用局限性，不仅显著提升了网络处理效率，还为高吞吐、低延迟场景下的系统优化提供了新的可能性。这种技术突破使得 Gazelle 能够适配更广泛的业务需求，涵盖云原生、边缘计算、高性能数据库、金融交易系统等多个前沿领域。然而，目前其潜力尚未被充分挖掘，许多创新应用场景仍处于探索和验证阶段。与此同时，Gazelle 社区正持续活跃演进，不断吸纳开发者和企业用户的反馈，完善功能生态，推动标准化和最佳实践的建立，未来有望成为下一代高性能网络基础设施的核心组件之一。

仓库地址

https://gitee.com/openeuler/gazelle

BiSheng JDK 毕昇 JDK

SIG Compiler

毕昇 JDK 是基于 OpenJDK 定制的 Huawei JDK 的开源版本，是一款高性能、可用于生产环境的 OpenJDK 发行版。毕昇 JDK 团队积累了丰富的开发经验，解决了许多实际业务中由原生 OpenJDK 缺陷引起的问题，毕昇 JDK 致力于为 JAVA 开发者提供一款稳定可靠、高性能、易调测的 JDK，也为用户在 AArch64 架构上提供一个更好的选择。

技术挑战

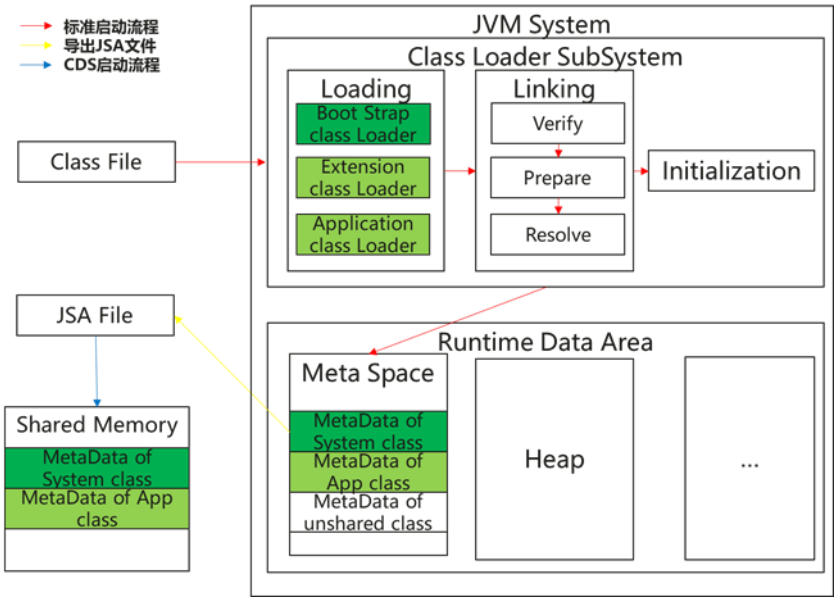
JDK 作为 java 运行的基础软件，性能和稳定性一直是 java 开发者和 java 产业关注的重点；众所周知，OpenJDK8 已进入维护期，引入大的特性比较困难，同时参与 OpenJDK 开发和维护工作的门槛相对较高。针对 java 启动时间慢、GC 吞吐量低、延时高、加解密性能弱等问题，毕昇 JDK 持续打造竞争力，引入新的优化特性。

功能描述

功能描述

功能描述 1：APPCDS 特性

Java 程序运行初始阶段，类的加载是一个比较耗时的过程，且在每次程序运行中均需要执行一遍。而 CDS（Class Data Sharing）技术，就是把类加载后的的数据保存到文件中，下次运行时，直接将加载后的类数据从文件中恢复到内存中，不需要再重新执行类的加载过程，从而提高性能。毕昇 JDK8 在 OpenJDK 提供的 CDS 特性基础上，扩展提供了 AppCDS 特性，增加了对应用类的支持。该特性在 Hive-Sql 场景平均性能提升 7%+；



APPCDS 业务流程图

功能描述 2：G1GC 堆内存伸缩特性

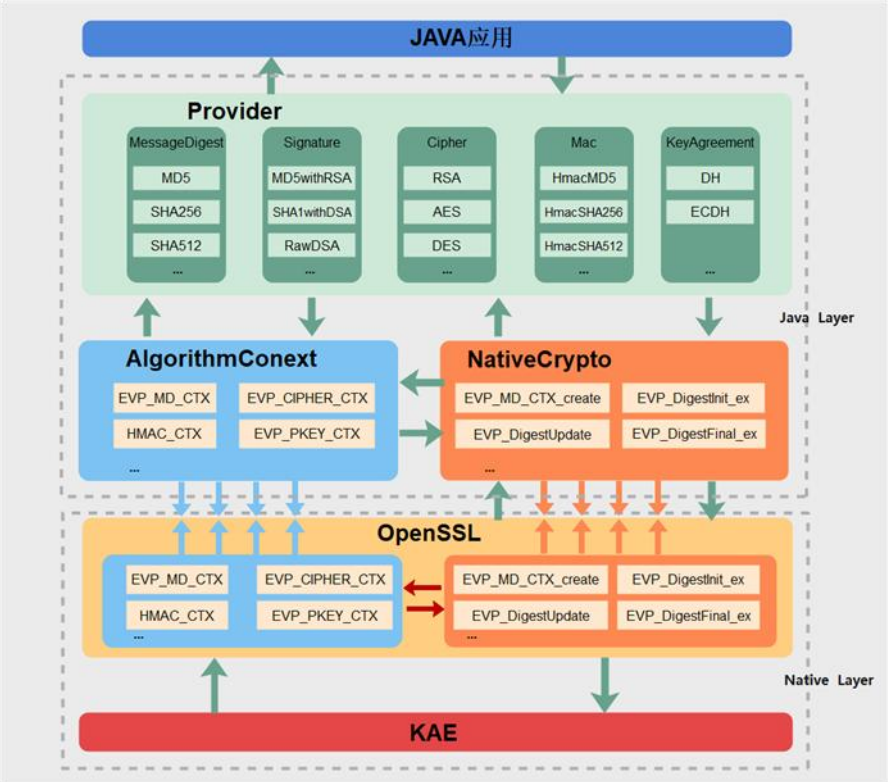
在 OpenJDK 8 中，G1GC 不会及时将空闲的 Java 堆内存释放给操作系统，其仅在 Full GC 时才会把空闲的 Java 堆内存进行释放。由于 G1 尽可能避免触发 Full GC，因此在许多情况下，除非强制从外部执行 Full GC，否则 G1 不会将空闲的 Java 堆内存释放给操作系统。

在按资源使用量付费的容器场景中，G1 不释放空闲的 Java 堆内存返回给操作系统的行为非常糟糕。即使在应用负载下降或不活跃时，G1 也会保留和占用所有 Java 堆。这会导致客户一直在为所有 JVM 占用资源付费，并且云提供商也无法充分利用其硬件资源。如果 JVM 能够检测到应用负载下降和 Java 堆有空闲内存的情况，并自动减少 JVM Java 堆占用情况，那么双方都会受益。在某 49 个微服务场景，开启 G1 堆内存回收特性比默认 G1GC 的实际物理内存减少 40%。

功能描述 3：KAE Provider 特性

KAE 加解密是鲲鹏加速引擎的加解密模块，鲲鹏硬加速模块实现了 RSA/SM3/SM4/DH/MD5/AES 算法，提供了高性能对称加解密、非对称加解密算法能力，兼容 openssl1.1.1a 及其之后版本，支持同步和异步机制。

毕昇 JDK 8 通过利用 Provider 机制，实现对鲲鹏服务器 KAE 加解密特性的支持，以帮助用户提升在鲲鹏 AArch64 服务器加解密业务的竞争力。在 Https 场景，性能提升 1 倍。



KAE Provider 业务架构图

KAE Provider 已支持算法列表：

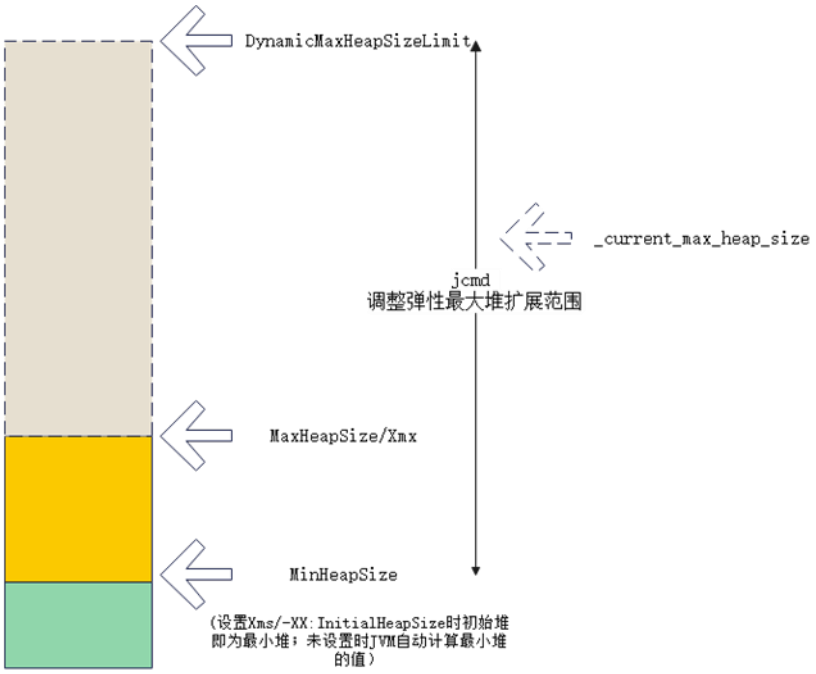
算法	说明
摘要算法	包括 MD5、SHA256、SHA384、SM3
对称加密算法 AES	支持 ECB、CBC、CTR、GCM 模式
对称加密算法 SM4	包括 ECB、CBC、CTR、OFB 模式
HMac	包括 HmacMD5、HmacSHA1、HmacSHA224、HmacSHA256、HmacSHA384、HmacSHA512
非对称加密算法 RSA	支持 512、1024、2048、3072、4096 位秘钥大小
DH	包括 DHKeyPairGenerator 和 DHKeyAgreement，支持 512、1024、2048、3072、4096 位秘钥
ECDH	包括 ECKeyPairGenerator 和 ECDHKeyAgreement，支持曲线

算法	说明
	secp224r1、prime256v1、secp384r1、secp521r1
RSA 签名	包括 RSASignature 和 RSAPSSSignature，私钥只支持 RSAPrivateCrtKey

功能描述 4：JVM 堆动态伸缩特性

目前互联网普遍使用容器化部署应用的模式，容器场景下容器资源可垂直扩容。而在 OpenJDK 中，最大堆内存只能在启动时指定，无法在运行期间动态调整堆大小，java 应用无法使用到容器扩容出的内存。

毕昇 JDK 通过支持内存伸缩的能力，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 以避免扩容导致的业务中断。本特可以解决 JDK 在容器化环境中无法动态扩展堆内存的痛点，支持垂直弹性扩容；同时支持堆上限缩容。



堆上限伸缩示意图

应用场景

毕昇 JDK 是基于 OpenJDK 开发和发行的 java 基础软件，在 openEuler 等 Linux 环境里面应用比较广泛，应用场景涵盖大数据、中间件、加解密敏感等场景。助力金融、中间件、运营商、互联网等行业客户实现在大数据 spark 性能提升 10%，加解密场景性能提升 100%+。

仓库地址

毕昇 JDK 8、11、17 和 21 均已开源，且每隔 3 个月进行版本升级和新特性合入，java 开发者可以在毕昇 JDK 开源社区获取最新信息、开展相关交流。

软件产品类型	交付类型	仓库链接
毕昇 JDK8	开源代码仓	https://gitee.com/openeuler/bishengjdk-8
毕昇 JDK11	开源代码仓	https://gitee.com/openeuler/bishengjdk-11
毕昇 JDK17	开源代码仓	https://gitee.com/openeuler/bishengjdk-17
毕昇 JDK21	开源代码仓	https://gitee.com/openeuler/bishengjdk-21

GCC for openEuler

SIG Compiler

GCC for openEuler 编译器基于开源 GCC（GNU Compiler Collection，GNU 编译器套装）开发。开源 GCC 是一种支持多种编程语言的跨平台开源编译器，采用 GPLv3（GNU General Public License, version 3）协议，是 Linux 系统上目前应用最广泛的 C/C++编译器，基本被认为是跨平台编译器的事实标准。而 GCC for openEuler 在继承了开源 GCC 能力的基础上，聚焦于 C、C++、Fortran 语言的优化，增强自动反馈优化、软硬件协同、内存优化、自动向量化等特性，并适配国产硬件平台，如鲲鹏、飞腾、龙芯等，充分释放国产硬件算力。

技术挑战

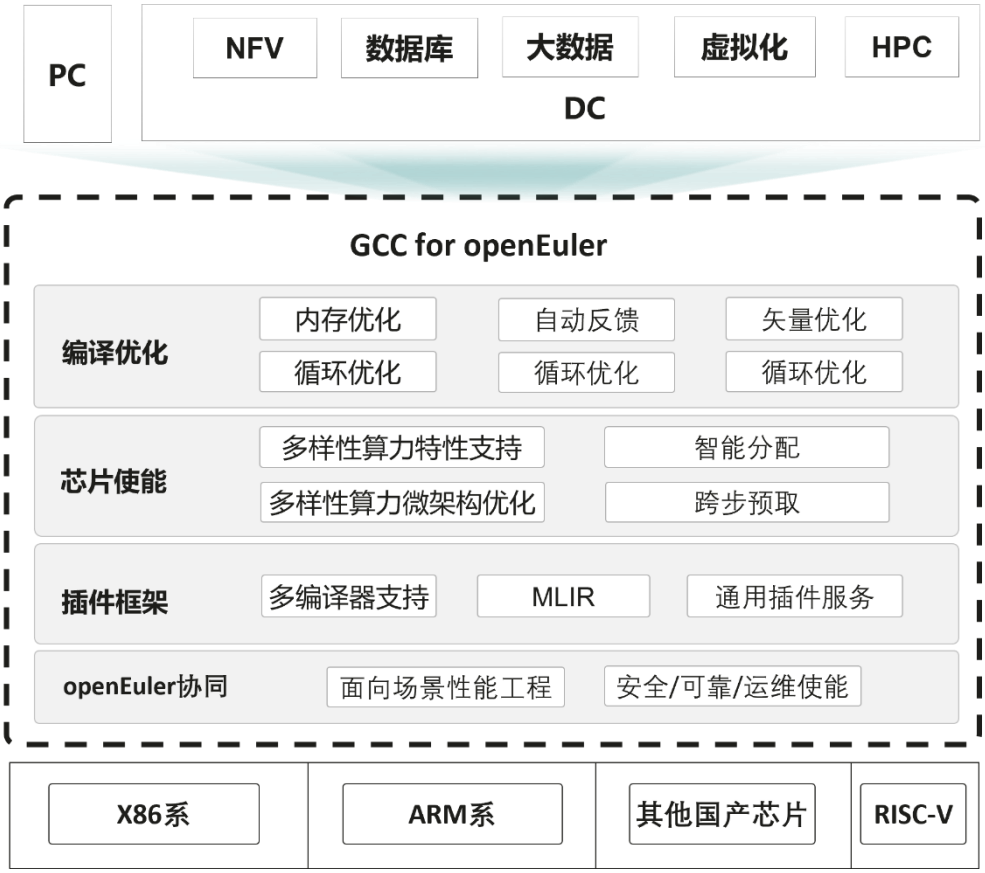
GCC 作为 Linux 内核的默认编译器，跨平台编译器的事实标准，是操作系统中至关重要的基础软件。GCC 的修改往往牵一发而动全身，对上层应用影响甚大，因此 GCC 开发者不仅要熟悉编译原理等基础知识，还要有充足的技术储备，加强特性的安全性、鲁棒性，在增强竞争力的同时，保证 GCC 本身的安全稳定。GCC for openEuler 致力于在开源 GCC 基础上，提供更多元化的竞争力特性，通

过编译优化、反馈优化等手段，提升上层软件的性能表现。在 Compiler SIG 双周例会上有 GCC 的固定议题，欢迎社区开发者随时与会交流。

功能描述

GCC for openEuler 支持鲲鹏、x86 等主流硬件平台，支持与 openEuler 性能/安全/可靠/运维工程进行协同，对接编译器插件框架，提供通用化插件功能，支持多样算力特性支持和微架构优化，实现内存智能分配、内存优化、自动矢量化等特性，并通过整合业界领先的反馈优化技术，实现自动反馈优化，提升数据库等场景应用性能。GCC for openEuler 在以下四个方向实现主要突破。

1. 基础性能：基于 GCC 开源版本，提升通用场景性能，服务多样算力。
2. 反馈优化：整合业界领先的反馈优化技术，实现程序全流程和多模态反馈优化，提升数据库等云原生场景重点应用性能。
3. 芯片使能：使能多样算力指令集，围绕内存等硬件系统，发挥算力优势，提升 HPC 等场景化性能。
4. 插件框架：使能多样算力差异化编译诉求，一套插件兼容不同编译框架，打通 GCC 和 LLVM 生态。



应用场景

GCC for openEuler 是基于开源 GCC 开发和发行的 GCC 基础软件，在 openEuler 等 Linux 环境里面应用比较广泛，应用场景涵盖数据库、虚拟化、HPC 等重要场景。实现 Arm 平台下，SPEC2017 基础性能相比开源 GCC 提升 20%，助力运营商、云厂商、安平等行业客户 MySQL 数据库性能提升 15%+。

仓库地址

GCC for openEuler 代码已开源，且随 openEuler 正式版本进行版本升级和新特性合入，GCC 开发者可以在 openEuler 开源社区获取最新信息、开展相关交流。

软件产品	交付类型	链接
GCC for openEuler	代码仓	https://gitee.com/openeuler/gcc
	软件包仓	https://gitee.com/src-openEuler/gcc

LLVM for openEuler

SIG Compiler

LLVM 项目是一个开源的编译器基础设施项目，它提供了一套用于编译程序的工具链和库。近年来，LLVM 项目越来越得到开发者的关注，社区非常活跃，商业公司也纷纷基于 LLVM 项目推出商业编译器。LLVM for openEuler 由华为毕昇编译器团队主要贡献，致力于在开源 LLVM 基础上与 openEuler 协同创新，包括兼容性、性能和开发态安全编码特性，并打造稳定的 LLVM 长效版本，为 openEuler 上的编译器提供第二选择。它适配多种硬件平台，如鲲鹏、龙芯、申威等，充分释放多样性硬件算力。

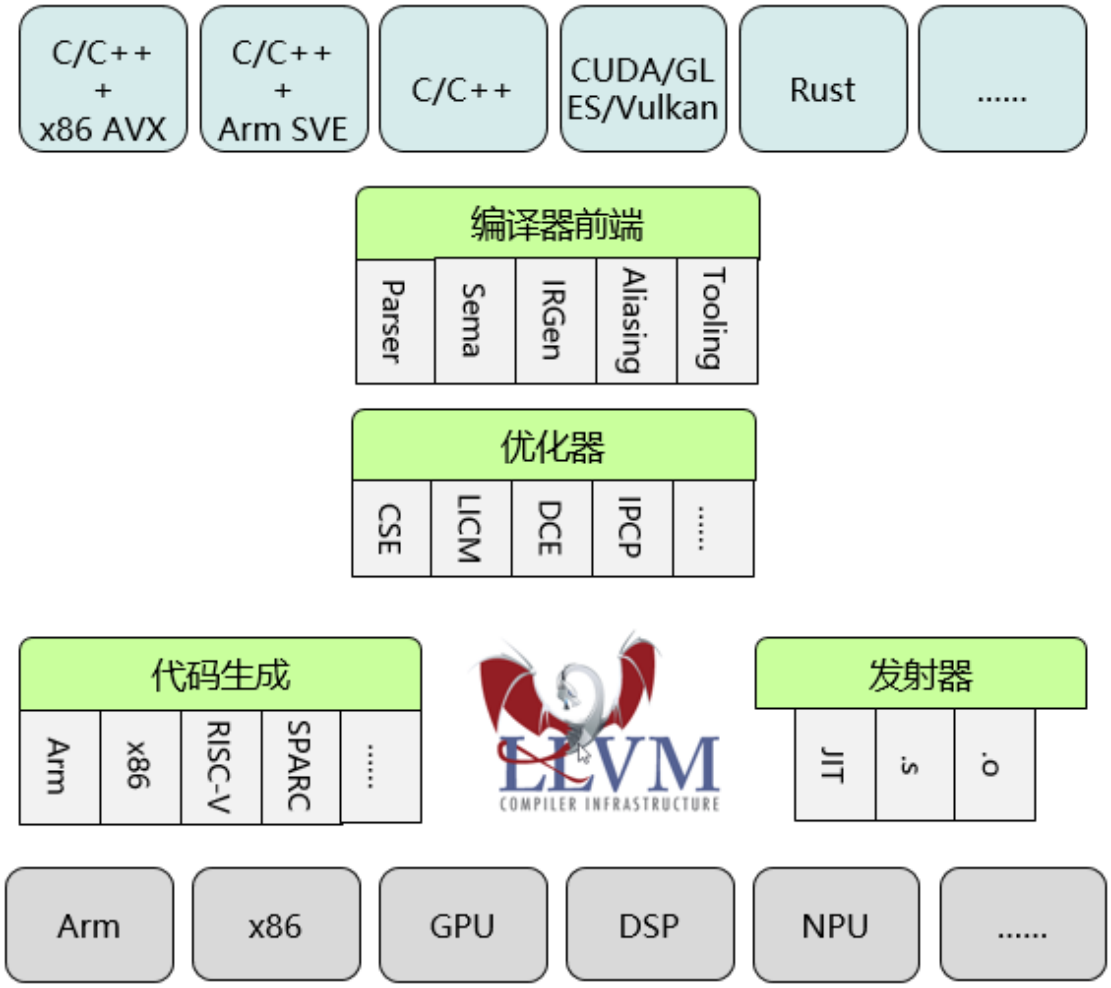
技术挑战

LLVM for openEuler 作为 openEuler 上编译器的第二选择，需要充分发挥优势，提供差异化竞争力，与 GCC 编译器形成优势互补，同时生态系统也需要进一步完善，这样才能通过 openEuler 为客户带来价值。从这两点出发，一方面 LLVM for openEuler 需要进一步提供强大而可扩展的优化能力，在计算主力场景，如数据库、大数据、分布式存储、虚拟化上提供更多性能收益，另一方面需要不断壮大社区和生态，兼容现存软件包，并为新开发的软件包提供更好的编译工作和服务。

功能描述

LLVM 采用了模块化架构设计，将编译过程分为多个独立阶段，如前端、优化和后端。这种设计使得 LLVM 更加灵活和可扩展，有助于各阶段模块分别演进创新，而通过统一的 IR 表示又将不同的模块有机的结合起来。目前 LLVM 项目包含多个子项目，如 clang、flang、llvm、mlir、lld 等。

架构描述



功能描述 1：Sanitizer

LLVM 的 Sanitizer 是一组用于进行动态代码分析和检测的工具，皆在帮助开发人员发现和调试常见的内存错误和安全问题，这些工具被设计为与 LLVM 编译器和运行时库紧密集成，提供了一种便捷的方式来检测和诊断代码中的问题。

功能	使用方法	探测问题列表
快速内存错误检测	-fsanitize=address	• Out-of-bounds accesses to heap/stack/globals • Use-after-free • Use-after-return • Use-after-scope • Double-free • invalid free • Memory leaks
数据竞争检测	-fsanitize=thread	• Data races
内存检测器	-fsanitize=memory	• uninitialized reads • use-after-destruction
未定义行为检测	-fsanitize=undefined	• integer-divide-by-zero • Bitwise shifts that are out of bounds for their data type • Dereferencing misaligned or null pointers • Signed integer overflow
硬件辅助内存错误检测	-fsanitize=hwaddress	• Same and AddressSanitizer
堆栈缓冲区溢出	-fsanitize=safe-stack	• 保护程序免受基于堆栈缓冲区溢出的攻击

功能描述 2：clang extra tools

Clang Extra Tools 是一组由 LLVM 项目提供的额外工具，用于与 Clang C/C++编译器一起使用，皆在提供对代码静态分析、代码重构和代码风格检查等功能的支持。

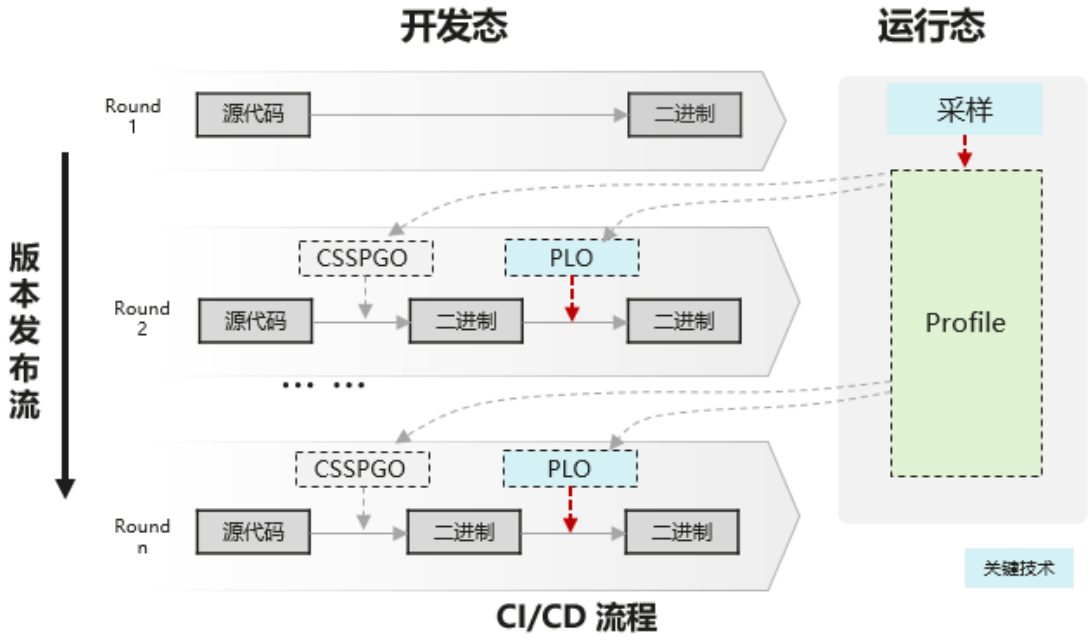
Clang-Tidy：Clang-Tidy 是一个强大的静态代码分析工具，用于检查 C、C++和 Objective-C 代码中的常见错误、潜在问题和代码风格违规。它可以自动检测和修复代码中的问题，帮助开发人员编写更高质量、更规范的代码。

Clang-Format：Clang-Format 是一个代码格式化工具，用于自动格式化 C、C++和 Objective-C 代码。它可以根据配置规则自动调整代码的缩进、换行、空格等，以保持一致的代码风格。Clang-Format 可以帮助团队在代码风格上达成一致，提高代码的可读性和维护性。

Clang-Check：Clang-Check 是一个用于编写自定义静态分析检查器的工具。它允许开发人员编写自定义的静态分析规则，用于检测代码中的特定问题或潜在错误。Clang-Check 提供了强大的 API 和框架，使开发人员能够根据自己的需求创建定制化的代码检查工具。

功能描述 3：CFGO 持续反馈优化

对于数据中心大型应用，前端性能瓶颈较大，具有高 i-cache 和 TLB 未命中率的场景推荐使用 CFGO 持续反馈优化，周期性地采集环境运行时信息进行持续反馈优化，保证低开销、性能泛化性。



应用场景

作为 C/C++/Fortran/Rust 语言编译器可以用于编译构建服务器、云计算、边缘计算、嵌入式场景应用编译构建。实现 ARM 平台，SPEC2017 基础性能相比上游 GCC 主力版本提升 30%+，计算主力场景性能提升 5%~10%。推荐用于计算主力场景和高性能计算场景，如大数据、数据库、分布式存储、虚拟化等等。

仓库地址

源码仓：

https://gitee.com/openeuler/llv m-project

制品仓：

https://gitee.com/src-openEuler/clang

https://gitee.com/src-openEuler/llvm

https://gitee.com/src-openEuler/lld

Go for openEuler

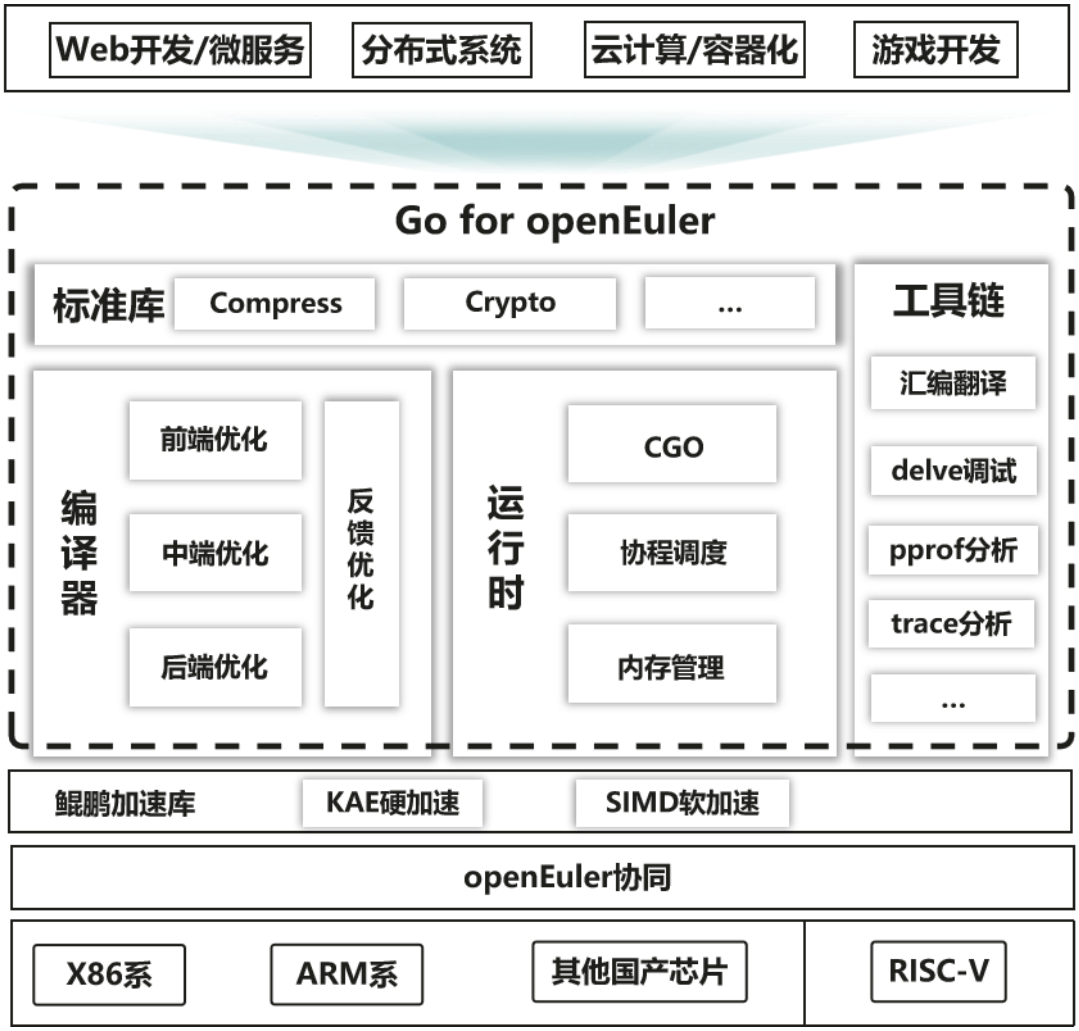
SIG golang

Go for openEuler 是基于开源 Golang 开发，是一款高性能、高可靠、易开发的 Golang 发行版，致力于打造生态兼容、极致体验、亲和 openEuler 的高性能编译器。主要面向云原生、微服务应用等对敏捷开发和运行性能都有要求的容器云场景，围绕业界主流 Go 业务负载进行优化，解决实际业务中由原生 Golang 能力不足导致的性能问题，并适配国产龙芯、鲲鹏等硬件平台，充分释放国产硬件算力。

技术挑战

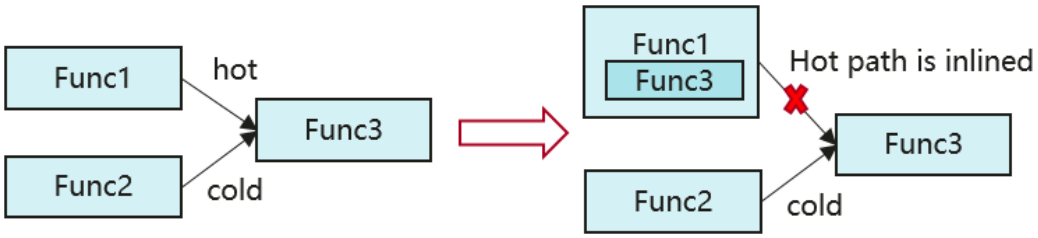
当前互联网许多业务场景如抖音、京东商城等热门软件都有使用 Go 开发微服务业务，应用如 etcd、cubefs、sonic 等核心应用，原生 Go 的编译器/标准库/运行时能力已难以满足这些大型互联网业务的性能需求。

功能描述



功能描述 1：CFGO 反馈优化

在保证程序功能不变的前提下，通过收集程序运行时信息，指导编译优化进行更准确的优化决策，获得性能更优的目标程序。基于程序局部性原理，使热指令紧密排布，优化 cache/TLB 命中，有效降低程序前端瓶颈，提升程序性能。



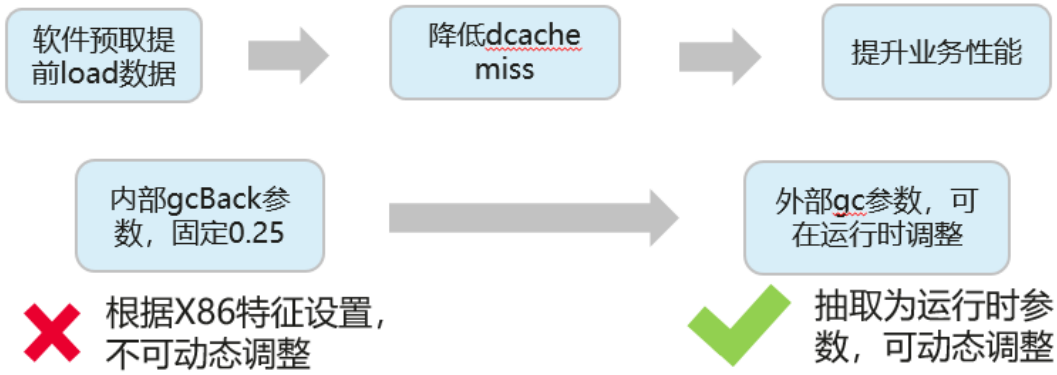
函数内联

功能描述 2：ARM 原子指令优化

在部分业务场景中，Golang 运行时调用 CAS 锁、LD/ST 指令开销较大，改为 ARM 亲合指令序列实现，可实现性能提升。

功能描述 3：运行时 GC 优化

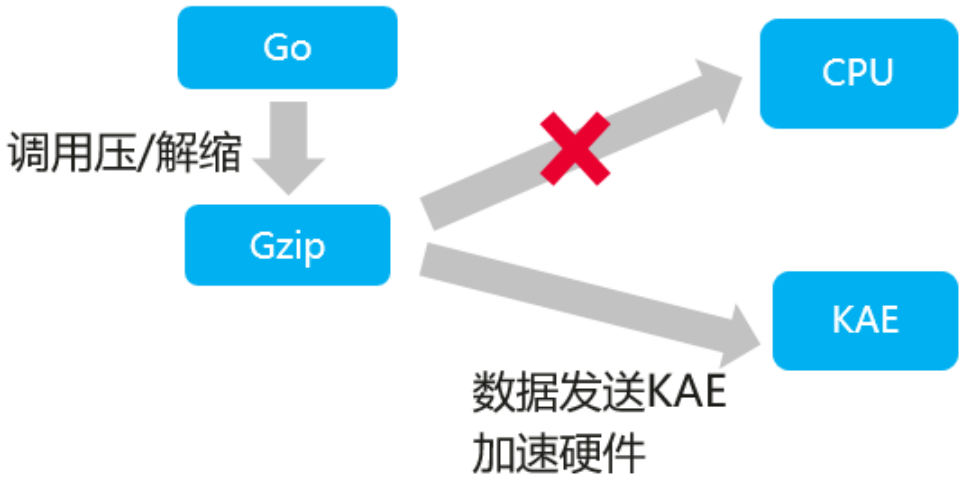
结合特征，插入软件预取；抽取 GC 协程开销资源参数为运行时参数，支持根据不同业务特征进行动态调整。



运行时 GC 优化

功能描述 4：结合鲲鹏 KAE 使能底层硬件加速：

改造 Golang 自身 Compress 库 Gzip 的压缩/解压缩逻辑实现使能底层硬件加速。



鲲鹏 KAE 使能

应用场景

Go for openEuler 是基于 Golang 开发的 Go 基础软件，在 openEuler 等 Linux 环境里应用广泛，应用场景涵盖云原生、分布式存储、云游戏等场景。助力互联网客户在主流业务场景性能提升 20%。

仓库地址

https://gitee.com/openeuler/golang

编译器插件框架

SIG Compiler

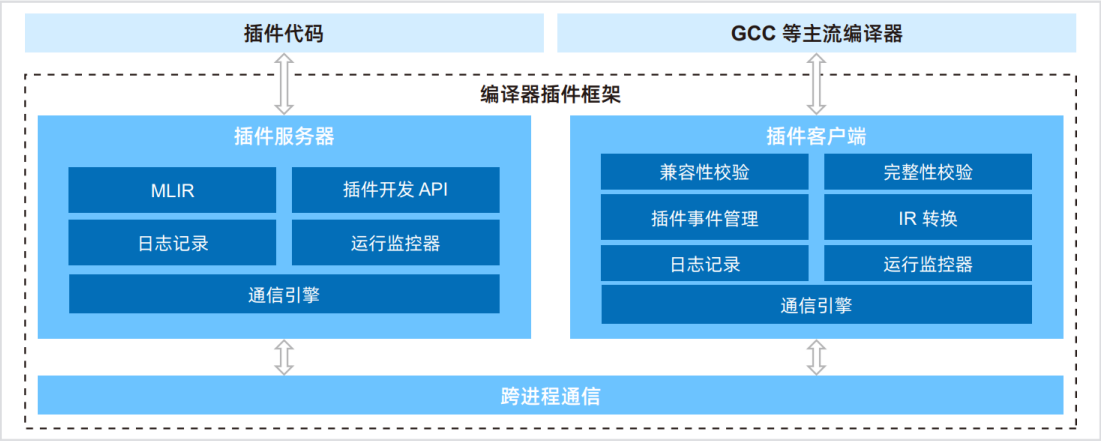
编译器插件框架（compiler plugin framework）提供面向 MLIR 的插件开发接口，以一次开发、多编译器落地为目标，避免编译工具的重复开发。编译器插件框架作为插件工具开发平台，提供对工具兼容性、完整性校验等公共能力的支持与维护，帮助用户以插件的形式提高优化特性的开发效率。

技术挑战

当前存在两款主流的编译器框架为 GCC 和 LLVM。市场上大量的编译工具和编译扩展能力，都是基于这两类编译器完成的。由于任何的编译工具都需要选择两种编译框架之一进行开发，而当其需要使能其他框架时，便会出现重复开发的问题。因为编译器框架的不同，导致即使是相同的工具设计逻辑，其代码也无法复用，需要在不同的编译器框架上做重复的代码开发并分别进行维护，这样直接拉高了工具的开发和维护成本。因此，当前编译工具的开发存在以下几方面的难点：

- 需要深入修改编译器，难度高、难维护。
- 编译工具需要同时使能两种编译框架时，存在重复开发的问题。
- 缺少兼容性等公共能力，拉高工具的开发和维护成本。

功能描述



1. 提供基于 MLIR 的插件开发能力，支持与 GIMPLE 等编译器中间表示的转换。
2. 插件框架提供对 19 类 GIMPLE 的支持。
3. 支持兼容性检测、二进制完整性校验等公共能力。
4. 支持安全编译选项校验、操作合法性校验等插件运行监控校验功能。
5. 支持插件客户端作为 GCC 插件加载，可以在完全不需要修改 GCC 编译器代码的情况下实现插件功能。
6. 支持为链接时优化（LTO，Link Time Optimization）使能插件框架。

应用场景

场景一：编译工具开发者构建工具，并需要完整性校验等公共能力。

有多编译器落地需求的工具开发者，可以使用编译器插件框架作为开发平台，基于 MLIR 进行一次工具开发，即可在 GCC 等主流编译器上使能工具。编译器插件框架统一提供对兼容性检测、二进制完整性校验等公共能力的支持和维护。

场景二：开发者以插件形式快速使能与验证编译相关工具。

编译器插件框架提供以插件的形式运行在 GCC 等主流编译器上。无需对编译器进行源码改动，提高开发效率。

仓库地址

https://gitee.com/openeuler/pin-gcc-client

https://gitee.com/openeuler/pin-server

强安全和高可靠

secGear 机密计算统一开发框架

SIG	confidential-computing

secGear 作为开源的 openEuler 机密计算组件，致力于提供简单、易用的机密计算软件栈及解决方案，降低机密计算的使用门槛，推动机密计算生态发展。

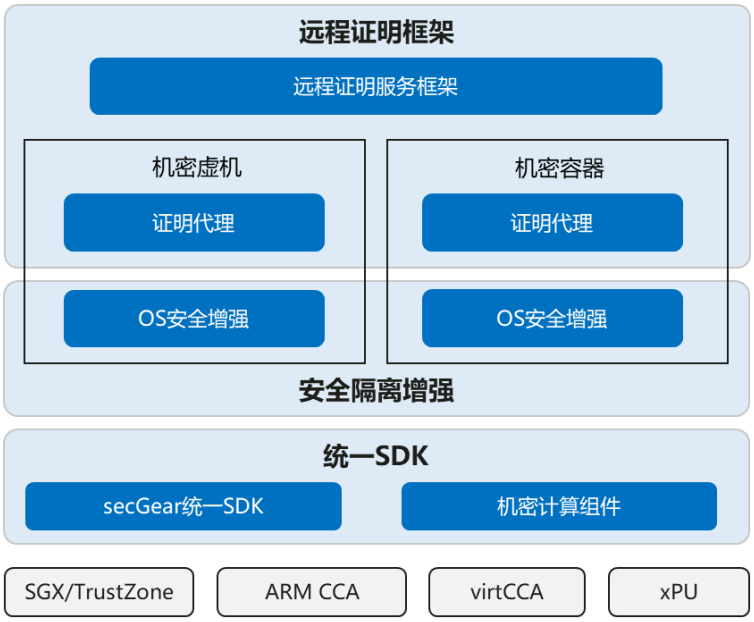
技术挑战

随着“机密计算”x“云原生”技术的不断发展，机密虚机、机密容器等越来越受到用户青睐。在提供 OS 级高安隔离运行环境的同时，仍面临新的挑战：

- 安全隔离：虚拟化、容器运行时等启动/运行链上关键组件不在保护范围内，容易被利用；OS 级隔离对外暴露的服务、端口可能成为攻击跳板；机密虚机/容器内的组件繁多，软件漏洞、后门等威胁安全运行。
- 安全易用：不同 TEE 技术厂商的底层实现存在差异，不仅不利于上层应用使能硬件能力，也不利于机密虚机/容器进行迁移。
- 安全互联：机密虚机/容器简化了安全应用的开发部署，但用户和云端机密应用/服务间、或机密虚机/容器间如何建立安全的互信互联，确保数据传输安全；当机密虚机/容器使用 xPU 等异构计算设备时，如何解决设备间的安全互信。

功能描述

面对 AI、云原生等技术浪潮，secGear 作为 openEuler 机密计算组件，将重点着眼于构建分布式云原生机密计算底座，南向纳管异构硬件 TEE，北向支持业务应用云原生部署。



其核心能力包括：

- 远程证明服务框架：包括证明服务、证明代理等功能，可以快速建立远程证明服务体系。此次新增
- 安全隔离增强：针对机密虚机/容器场景，提供配置加固、安全 OS 镜像构建等能力支持，缩小机密虚机/容器启动/运行过程中的攻击面。
- secGear 统一 SDK：屏蔽不同 SDK 接口差异，提供统一开发接口，实现不同架构共源码；提供零切换特性，提高 REE-TEE 频繁交互场景下应用性能；提供基于 SDK 的机密计算组件，支持跨 TEE 安全互联互通，如本地证明、跨 TEE 环境 RA-TLS 、安全通道。

当前新增的安全特性包括：

- 支持 Kuasar 机密容器镜像加解密及基于远程证明的密钥安全传输
- 针对 AI 推理场景，支持 NPU 固件度量及证明
- OS 安全增强配置加固指导及安全 OS 镜像裁剪制作

应用场景

secGear 在 openEuler AI 全栈安全、密态数据库、硬件密码机替代、大数据等场景应用广泛，助力金融、电信等行业客户业务快速迁移到机密计算环境，保护数据运行时安全。

仓库地址

https://gitee.com/openeuler/secGear

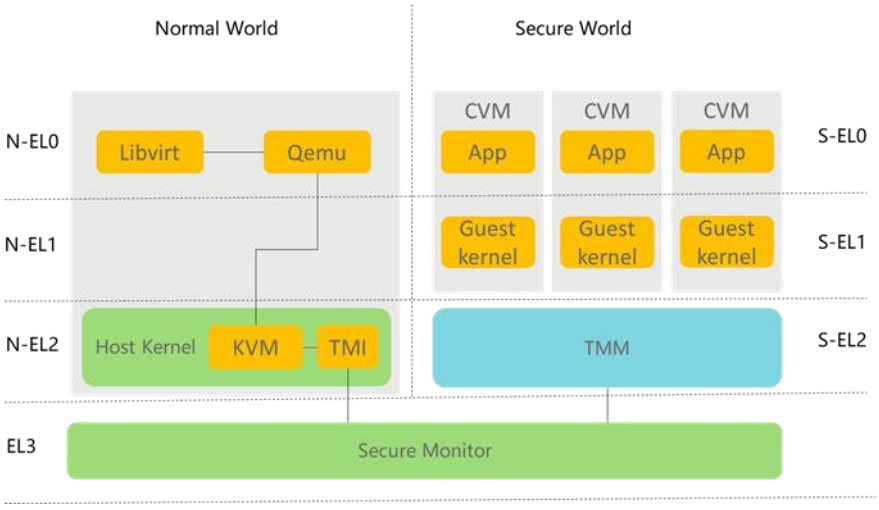
virtCCA 机密计算

SIG	confidential-computing

virtCCA 机密虚机特性实现安全域虚拟化，支持机密虚机和机密容器，基于鲲鹏平台 S-EL2 能力，在 TEE 侧实现机密虚机能力，实现现有普通虚机中的软件栈无缝迁移到机密环境中，让更广泛的应用快速迁移到可信环境，达成数据可信流通。

技术挑战

数据在传输和存储阶段，都有较成熟的加密措施，而对数据在计算状态下的安全保障能力不足，导致无法实现数据全生命周期的安全保护，机密计算是解决该问题的一项关键技术；传统的专用高安 TEE，应用需重开发或迁移，应用生态与安全 OS 绑定，使用门槛高，且无法满足云原生通用生态场景，如虚机和容器。virtCCA 机密虚机特性基于 Arm CCA 标准接口，在 Trustzone 固件基础上构建 TEE 虚拟化管理模块，实现机密虚机间的内存隔离、上下文管理、生命周期管理和页表管理等机制，支持客户应用无缝迁移到机密计算环境中。



功能描述

1. 设备直通：

   设备直通是基于鲲鹏平台，通过预埋在 PCIE Root Complex 里的 PCIE 保护组件，在 PCIE 总线上增加选通器，对 CPU 与外设间的通信进行选通，即对 SMMU 的 Outbound Traffic 和 Inbound Traffic 的控制流和数据流进行控制，以此保证整体数据链路的安全性。

   基于 virtCCA PCIPC 的设备直通能力，实现对 PCIE 设备的安全隔离和性能提升，存在以下优势：

(1) 安全隔离

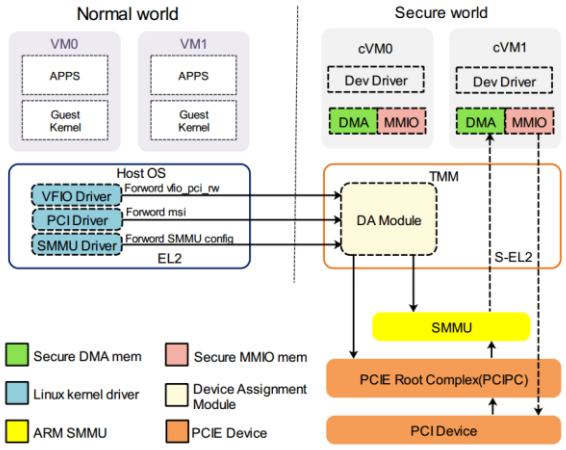
TEE 侧控制设备的访问权限，Host 侧软件无法访问 TEE 侧设备；

(2) 高性能

机密设备直通，相比业界加解密方案，数据面无损耗；

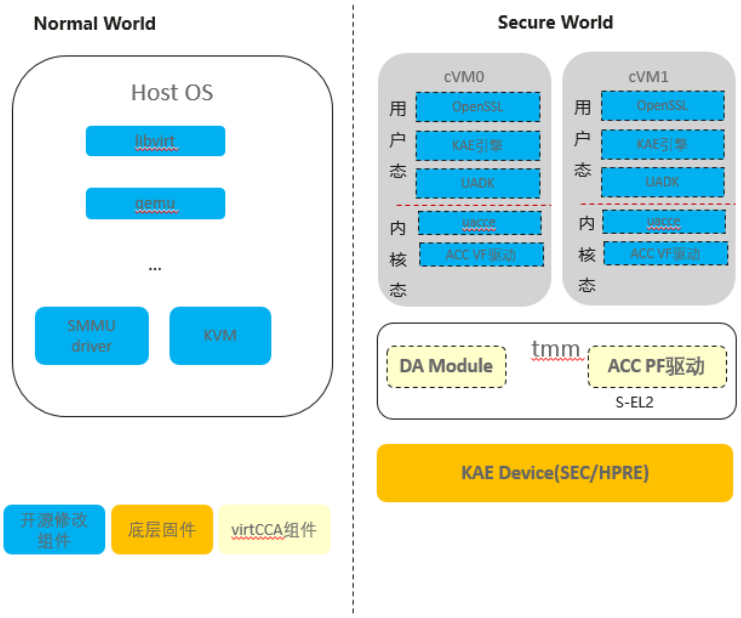
(3) 易用性

兼容现有开源 OS,无需修改开源 OS 内核驱动代码。



2. 国密硬件加速：

国密硬件加速是基于鲲鹏芯片，通过 KAE 加速器能力复用到安全侧，并采用 openEuler UADK 用户态加速器框架，提供客户机密虚机内国密加速性能提升以及算法卸载的能力。



应用场景

面向密态数据库、安全云主机、机密多方计算、数据可信流通以及 AI 模型数据保护。

仓库地址

https://gitee.com/openeuler/virtCCA_sdk

https://gitee.com/openeuler/virtCCA_driver

CCA 机密计算

SIG Virt

ARM CCA（Confidential Computing Architecture）是 ARM v9-A 新引入的机密计算架构规范，旨在为下一代计算设备定义标准化的机密计算解决方案。

openEuler 基于 ARM CCA 机密计算架构规范，实现了 OS 相关组件（KVM、QEMU、libvirt、Guest kernel）对 CCA 的支持，提供了原生支持 Realm 机密虚机的社区版本，满足基于 Realm 机密虚机保护使用中数据的安全诉求，同时提供了兼容传统应用生态及虚机管理软件的易用性。

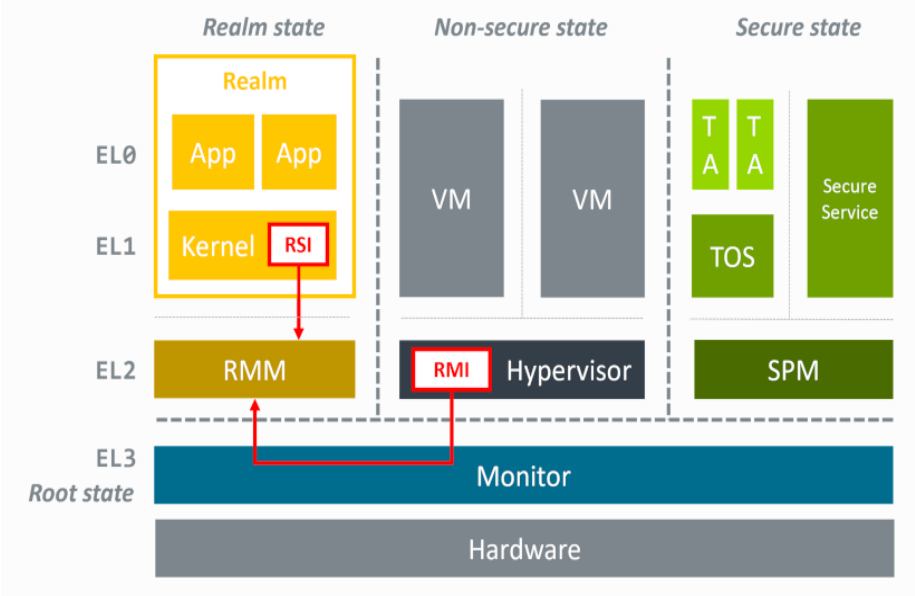
技术挑战

为实现普适、高效的机密计算，ARM CCA 面临多重技术挑战：

- **信任边界的确立**：在云场景中，Hypervisor、操作系统等底层软件栈可能极其复杂或存在漏洞，甚至可能是恶意的。CCA 的核心设计原则是“减少信任假设”，明确将 Hypervisor 和主机操作系统排除在信任边界之外。它们可以调度 Realm 的资源，但无法访问其内存内容。
- **动态内存管理的安全**：内存需要频繁地在不同的工作负载之间分配和释放。挑战在于如何在不暴露 Realm 内存的情况下，让系统管理器（如 Hypervisor）安全地进行内存管理，实现内存的清空和重复使用。
- **远程证明**：用户如何远程验证他们的代码确实运行在一个真实的、硬件强化的 ARM CCA 环境中，而不是一个模拟的恶意环境中？这需要一套基于硬件的密码学证明机制（远程证明），让硬件本身能够出具一份“证明报告”，供用户验证。

功能描述

ARM CCA 通过以下核心组件协同工作，构建一种隔离的、受保护的执行空间，在代码执行和数据访问方面与正常世界完全隔离，成为 Realm 机密域。



- **Realm 机密域**：Realm 是 CCA 的核心抽象，它是一种与正常世界（Non-secure）和安全世界（Secure，原来的 Trustzone）并行的新类型执行环境。Realm 是硬件隔离的，专为托管敏感代码和数据而设计。它独立于主机操作系统和 Hypervisor，它们可以管理 Realm 但无法访问其内部内容。
- **动态管理**：Hypervisor 可以应客户要求动态创建 Realm，并为其分配内存和 CPU 资源。但在 Realm 初始化后，Hypervisor 会将其控制权移交给一个受保护的安全虚拟化模块 RMM（Realm Management Monitor），此后 Hypervisor 便无法访问 Realm 内的秘密。
- **内存管理**：CCA 扩展了系统内存管理单元（MMU），使其能够识别和隔离 Realm 内存。任何从 Realm 外部（包括 Hypervisor）发起的访问尝试都会被硬件阻断，从而确保数据的机密性。
- **远程证明**：每个支持 CCA 的处理器都有一个基于硬件的唯一身份标识。当 Realm 启动时，它可以生成一份由硬件密码学签名的证明报告（Attestation Token）。用户可以获得这份报告，并验证其签名及组件度量值，从而确信他们的工作负载正在一个真实的、未被篡改的 ARM CCA 环境中运行。

应用场景

机密虚机主要应用于云计算环境，为核心工作负载提供高级别安全隔离。它使得客户能在不受信的公有云上处理敏感数据（如金融记录、医疗健康信息或知识产权），并确保其机密性和完整性连云服务商也无法窥探。这有效满足了数据主权、监管合规和跨机构隐私数据协作（如联合建模与分析）的关键需求。

仓库地址

https://gitee.com/openeuler/kernel/tree/OLK-6.6

https://gitee.com/openeuler/qemu/tree/qemu-8.2.0

https://gitee.com/openeuler/libvirt/tree/libvirt-9.10.0

极简运维

oeDeploy 软件部署工具

SIG ops

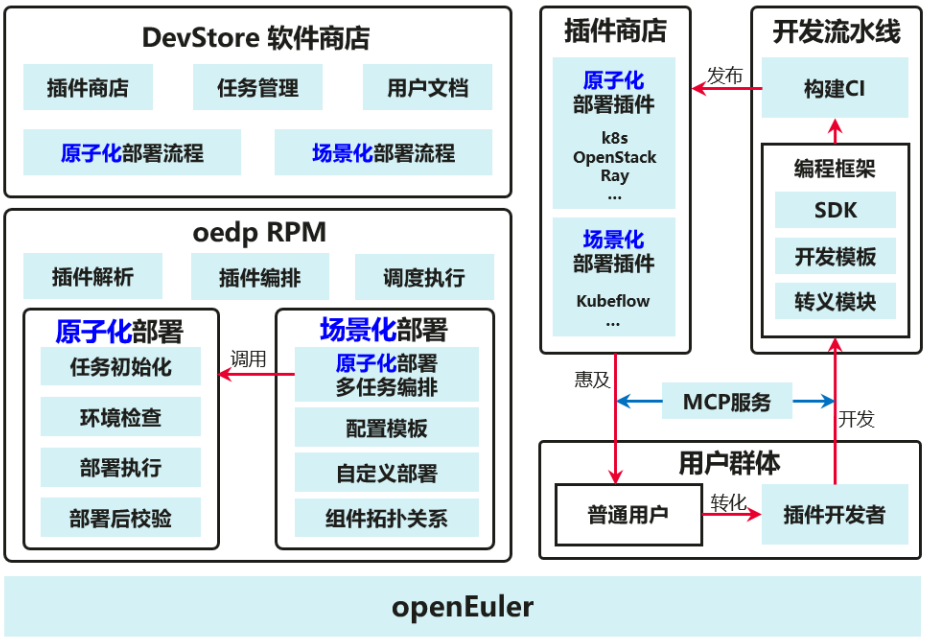
oeDeploy 是一款轻量级的软件部署工具，旨在帮助开发者快速、高效地完成各类软件环境部署，对单节点与分布式场景均可适配。

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。

仓库地址

https://gitee.com/openeuler/oeDeploy

A-Ops 智能运维

SIG ops

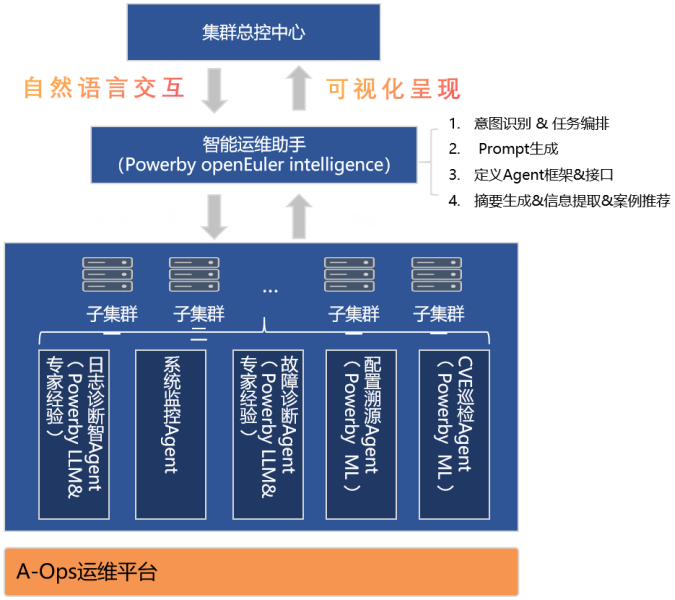
A-Ops 是一款基于操作系统维度的故障运维平台，提供从数据采集，健康巡检，故障诊断，故障修复的智能运维解决方案。A-Ops 运维智能化，结合 openEuler intelligence 通过对话交互实现运维操作以替代传统运维操作方式降低运维门槛，导航式操作，简化操作。

现交互场景支持 cve 提示和修复推荐，配置溯源配置异常追溯和配置基线同步，通过和助手交互，让运维助手帮忙执行日常运维操作。

技术挑战

云基础设施在近几年随着云原生、无服务化等技术的实施，其运维流程的复杂性变得越来越有挑战性。A-Ops 运维流程进行了智能化改造，实现智能运维助手，通过智能交互实现日常运维操作，降低运维门槛。

功能描述



aops 基于 openEuler Intelligence 集成智能运维助手，实现 cve 修复和配置溯源操作智能化：

- cve 修复：aops 自动显示集群 cve 情况，筛选高分高严重等级 cve 进行推荐并提供修复方式，用户通过助手辅助和页面混合交互实现 cve 的修复和结果查看。
- 配置溯源：通过助手查询基线配置异常的机器，页面展示异常机器和异常配置项，助手进行智能总结并提供推荐修复方式，可通过页面混合交互进行修复。

应用场景

Aops cve 修复和配置溯源实现运维智能化，通过对话交互实现运维操作以替代传统运维操作方式，协助运维人员日常运维，日常运维提供专家级指导，关联当前操作提供推荐操作，简化运维上手难度，提高使用体验。

仓库地址

https://gitee.com/openeuler/aops-zeus

https://gitee.com/openeuler/aops-apollo

https://gitee.com/openeuler/aops-vulcanus

https://gitee.com/openeuler/aops-hermes

https://gitee.com/openeuler/aops-ceres

https://gitee.com/openeuler/authHub

6 开发者支持

基础设施

CVE Manager 漏洞管理

SIG	Infrastructure/ security-committee

漏洞管理是 openEuler 社区对安全漏洞进行感知、收集、处理以及披露的流程、工具和机制的统称。

技术挑战

漏洞管理是 openEuler 社区对安全漏洞进行感知、收集、处理以及披露的流程、工具和机制的统称。

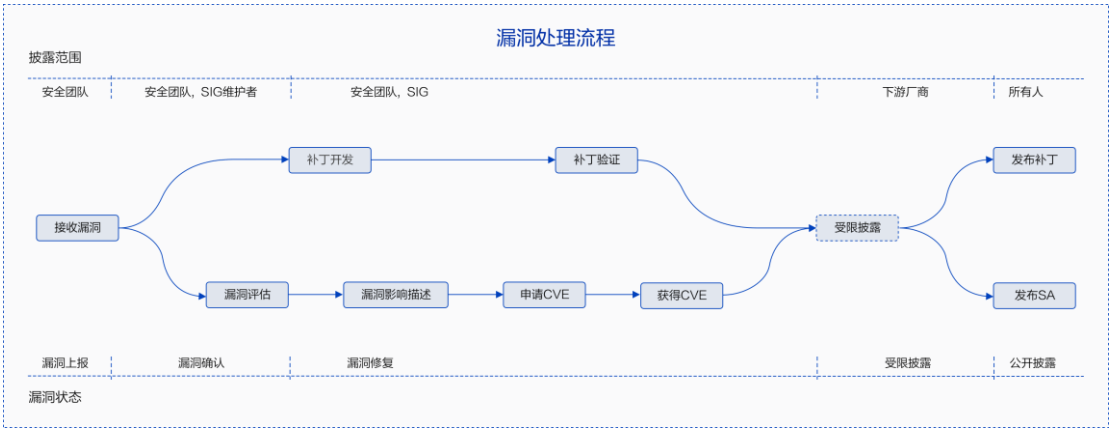
目标：

1. 及时感知
2. 高效分析
3. 快速修复
4. 受控披露

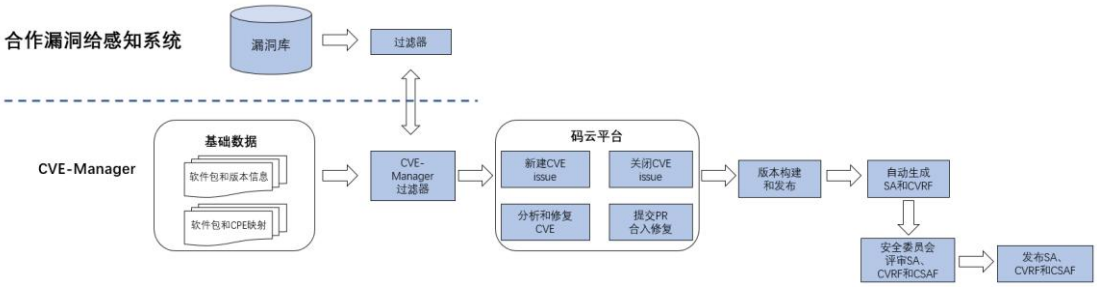
功能描述

openEuler 社区非常重视社区版本的安全性，openEuler 安全委员会负责接收、调查和披露 openEuler 社区相关的安全漏洞。我们鼓励漏洞研究人员和行业组织主动将 openEuler 社区的疑似安全漏洞报告给 openEuler 社区安全委员会。我们会快速的响应、分析和解决上报的安全问题或安全漏洞。

漏洞响应流程主要支持 openEuler 社区的 LTS 发行版和其分支版本。漏洞端到端的处理流程如下图。



openEuler 安全团队希望上报者将 openEuler 产品疑似安全漏洞上报给 openEuler 社区，并相互配合以负责任的方式修复和披露该问题。漏洞上报方式为通过 email 将 openEuler 产品的潜在安全漏洞发送到 openEuler 安全团队邮箱（openEuler-security@openEuler.org）。安全团队将在 48 小时内响应通过邮箱上报的疑似安全漏洞，并向上报者反馈漏洞处理的进展。



openEuler 社区通过原创的 CVE-Manager 项目从合作漏洞感知系统获取公开漏洞感知信息，然后通过机器人在码云平台对应项目软件包仓创建并维护漏洞相关记录，漏洞修复后进入通用版本构建发布以及安全公告发布流程。

openEuler 使用 CVSSv3 进行漏洞评分。

为了保护 openEuler 用户的安全，在进行调查、修复和发布安全公告之前，openEuler 社区不会公开披露、讨论或确认 openEuler 产品的安全问题。安全漏洞修复后 openEuler 社区会发布安全公告，安全公告内容包括该漏洞的技术细节、CVE 编号、CVSS 安全评分、严重性等级以及受到该漏洞影响的版本和修复版本等信息。安全公告提供邮件订阅功能，同时社区也提供 CVRF、CSAF 格式的安全公告。

应用场景

应用场景 1：对 openEuler 长期维护发行版进行公开漏洞的感知、分析、修复和披露。

应用场景 2：对 openEuler 长期维护发行版进行 0day 漏洞的收集、分析、修复和披露。

仓库地址

https://www.openEuler.org/zh/security/vulnerability-reporting/

https://gitee.com/openeuler/security-committee/blob/master/security-process.md

https://gitee.com/openeuler/cve-manager

Compass-CI

SIG CICD

Compass-CI 是一个可持续集成的开源软件平台。为开发者提供针对上游开源软件（来自 Github、Gitee、 Gitlab 等托管平台）的测试服务、登录服务、故障辅助定界服务和基于历史数据的分析服务。Compass-CI 基于开源软件 PR 进行自动化测试（包括构建测试，软件包自带用例测试等），构建一个开放、完整的任务执行系统。

技术挑战

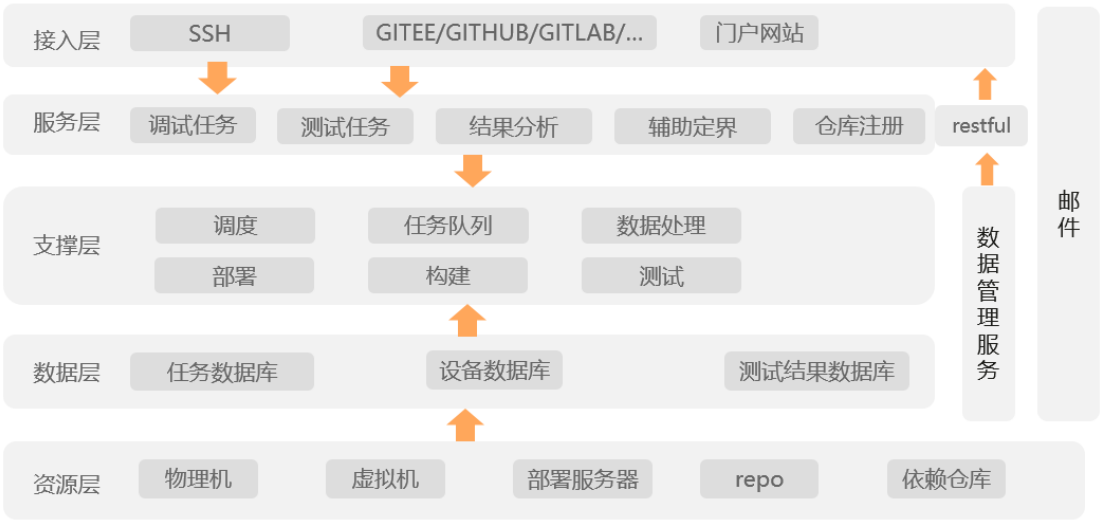
Linux 日趋复杂，开源软件开发者由于受到资源约束，往往只会对单一场景进行验证，面对众多的 Linux 发行版，开源软件如何快速引入及测试验证，是摆在开发者面前的首要问题。

资源单一，使用场景却是 {os, arch, machine, ..} 的矩阵组合。这导致大量的问题会在后续的使用过程中才会被发现，造成更大的修改成本。

问题复现困难，大量成本消耗在环境的准备中，无法进行快速的问题定位。

功能描述

Compass-CI 是一个通用全栈软件测试平台，集构建&测试系统、登录调测、测试分析比较、辅助定位于一体，通过主动测试数以万计的开源软件，暴露这些软件在芯片和操作系统上的问题，在第一时间自动定位问题并以报告的形式反馈给第三方软件开发者，方便第三方开发者能及时处理问题，保障软件质量。给社区开发者提供友好的开发体验，与社区开发者一起繁荣开源软件生态及提升开源软件质量。



1. 测试服务：支持开发者基于本地设备开发，往 github 提交代码，Compass-CI 自动获取代码开展测试，并向开发者反馈测试结果。
2. 调测环境登录：Compass-CI 提供 SSH 登录能力，测试过程中如果测出问题，开发者可根据需要登录环境进行调测。
3. 测试结果分析：Compass-CI 记录历史测试结果，对外提供 web 及命令行接口，支持开发者针对已有的测试结果进行分析，挖掘影响测试结果的因素。
4. 辅助定位：Compass-CI 在测试过程中可以自动识别错误信息，触发基于 git tree 的测试，找出引入问题模块的变化点。

应用场景

聚合开发者测试用例：开发者向代码托管平台提交代码、测试用例、测试工具时，Compass-CI 自动获取提交的代码开展构建测试，同时获取开发者编写到开源软件包的测试用例进行自动化测试，并反馈测试结果。

登录环境随时调测：测试过程中，发现 Bug 时，可随时提供调测资源服务，登录到环境进行复现、调试。

快照数据分析对比：测试过程中，全面监控系统运行信息（CPU/MEM/IO/网络等），对测试过程中的数据做快照归档，提供多次测试之间快照数据分析对比能力，协助开发者对测试结果开展分析，找出影响测试结果的因素。

辅助定界：测试过程中，发现有 Bug 时，自动触发 Regression 机制，找出首次引入问题 Commit 信息。

仓库地址

https://gitee.com/openeuler/compass-ci

https://gitee.com/compass-ci/lkp-tests

OEPKGS openEuler 软件扩展仓库

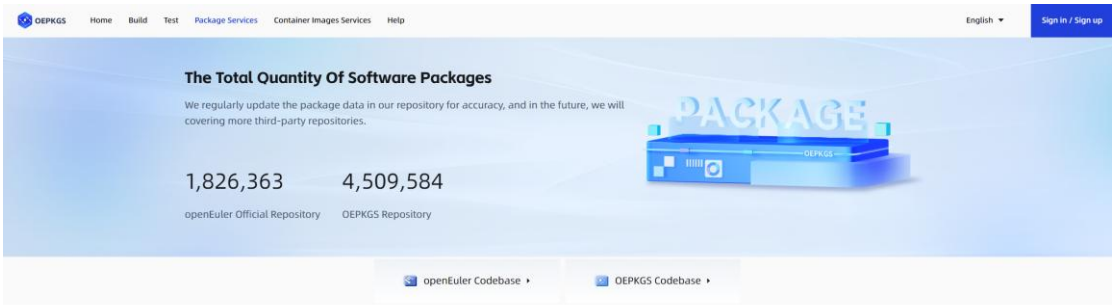
OEPKGS 开放软件包服务（Open External Packages Service）正式上线，为 openEuler 生态提供超过 3 万+源码包、百万级二进制软件包，可为 CentOS、Fedora 等系统向 openEuler 迁移的开发者、OSV、企业等用户提供一站式兼容性软件包、文件查询、下载、开源软件包的使用风险感知服务。

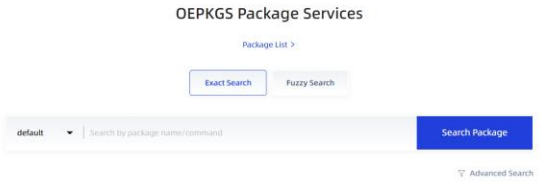
技术挑战

1. 文件、软件包软件包精准及模糊检索。
2. 软件包 CICD 门禁管理，依赖元数据分析及管理，软件包自由组合验证。
3. 开源软件风险感知，安全性及合规性风险分析。

功能描述

OEPKGS 由中国科学院软件研究所、中科南京软件技术研究院、openEuler 社区共同发起并提供支持的项目。旨在作为 openEuler 社区的扩展仓库，为开发者及广大用户提供丰富的软件包，同时提供一个成熟的 CICD 体系，支撑软件包引入溯源分析，源码构建，二进制扫描，基本功能验证，漏洞、合规风险感知，补丁、版本更新感知，保障软件仓库质量可靠及持续演进；结合 RPM 软件包检索、元数据分析、SBOM 和供应链分析、安全性及合规性风险分析等多项能力，同时提供一站式文件、软件包查询、风险感知查询、下载等服务，提升开发者及用户的使用体验。





应用场景

1. 面向创新场景开发者提供快速的软件包引入平台，快速将上游项目形成 RPM 包。
2. 面向迁移场景开发者提供多版本软件包引入能力，支持传统其他 Linux OS 快速替代。
3. 面向广大用户，提供 openEuler 已兼容的软件包查询，文件所在软件包的查询平台。
4. 面向企业用户，提供开源软件风险一站式查询平台，感知软件的风险。
5. 面向企业，提供闭源软件分发渠道，支持企业软件直达用户。

仓库地址

https://search.oepkgs.net/

https://gitee.com/src-oepkgs

开发者工具

DevStation 智能开发环境

SIG IDE/ intelligence

DevStation 是基于 openEuler 的智能开发者工作站，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。

全新安装工具 heolleo：heolleo 是一款专为 DevStation 设计的现代化客户端工具。其核心使命是简化 DevStation 的安装流程。采用模块化设计使其可以轻松扩展以支持不同的硬件架构（如 x86/ARM）、文件系统或引导加载器（GRUB 等）。支持从本地 ISO 镜像、网络地址（HTTP/FTP）等快速获取系统文件，提供灵活的安装方式:

1. 本地 ISO 安装:对于追求极致稳定、速度或需要在无网络、受限环境中部署系统的用户，heolleo 提供本地 ISO 安装模式。充分利用已有的系统镜像文件，提供一个高速、可靠且完全离线的安装体验，当前已实现自动化分区安装。

2. 网络安装：heolleo 的网络安装模式适应现代系统部署的趋势。可直接从互联网上的服务器获取最新系统文件，省去了手动下载镜像的步骤，能以最便捷的方式触及最新的 DevStation 版本。

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。

仓库地址

https://gitee.com/openeuler/mcp-servers

https://gitee.com/openeuler/heolleo

https://gitee.com/openeuler/oeDevPlugin

https://gitee.com/openeuler/devstation-config

DevStore 开发者软件商店

SIG ops

DevStore 是 openEuler 桌面版本的应用商店，是面向开发者的软件分发平台，支持 MCP 服务、oeDeploy 插件的检索与快捷部署功能。在 DevStation 平台上实现开箱即用。

功能描述

MCP 服务一键安装：DevStore 借助 openEuler 社区丰富的软件生态，以 rpm 软件包的形式处理 MCP 运行所需的软件依赖，并通过内置的服务管理工具，在智能体应用中快速部署 MCP 服务。自动帮助用户解决软件依赖与 MCP 配置问题，大幅提升用户体验。目前已支持 80+MCP 服务。

oeDeploy 插件快速部署：DevStore 借助 oeDeploy 工具实现主流软件的快速部署，大幅度降低开发者部署软件的时间成本。包括 Kubernetes、Kuberay、Pytorch、TenserFlow、DeepSeek 等 AI 软件，EulerMaker、openEuler Intelligence 等社区工具链，以及 RagFlow、Dify、AnythingLLM 等主流的 RAG 工具。

应用场景

DevStore 作为 openEuler 桌面端的软件商店，帮助开发者与初学者快速获取主流开发工具、AI 软件、MCP 服务等等，在详情页提供了丰富的用户文档与操作入口。所有复杂软件的部署操作都可以在一个操作界面完成，大幅降低学习成本、提升用户体验。

仓库地址

https://gitee.com/openeuler/DevStore

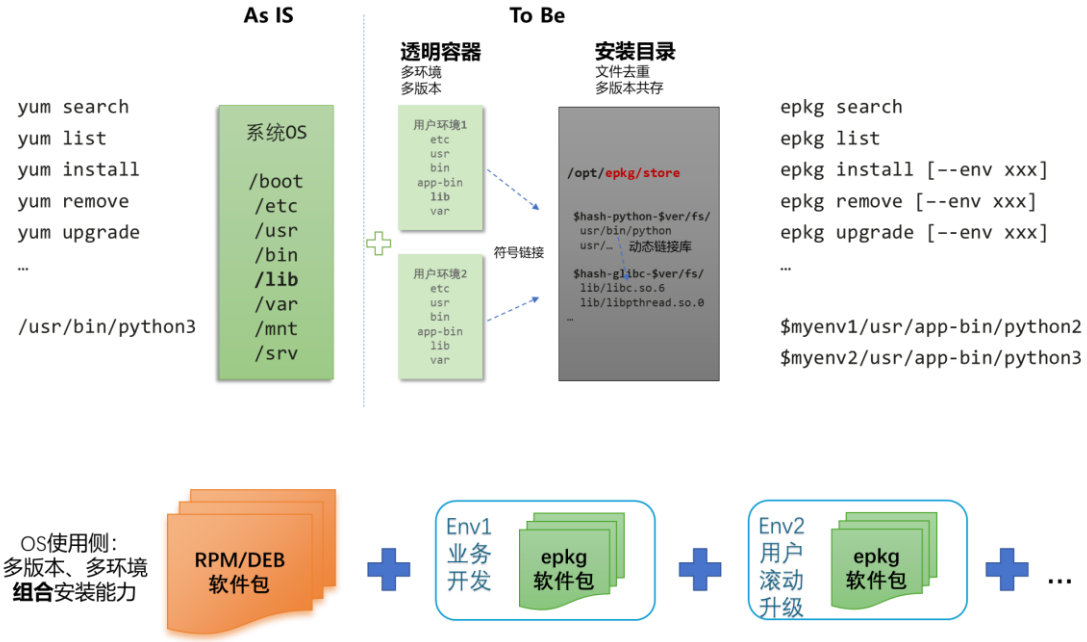
EPKG 新型软件包

SIG epkg

epkg 是 openEuler 提供的一款轻量级、可扩展的软件包管理工具，支持跨 OS 的软件管理、环境隔离、多源协同等能力。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命令安装不同版本的软件包。同时支持环境管理实现环境的创建/切换/使能/回退等操作，用户在误操作或安装软件后出现问题时，能够恢复环境。

功能描述

epkg 能够在现有 OS 上进行直接安装，通过 epkg env 和 epkg install 等命令，为不同环境指定相应的 repo 仓库，在各环境中安装不同版本软件包，并切换使用。



多版本兼容：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。

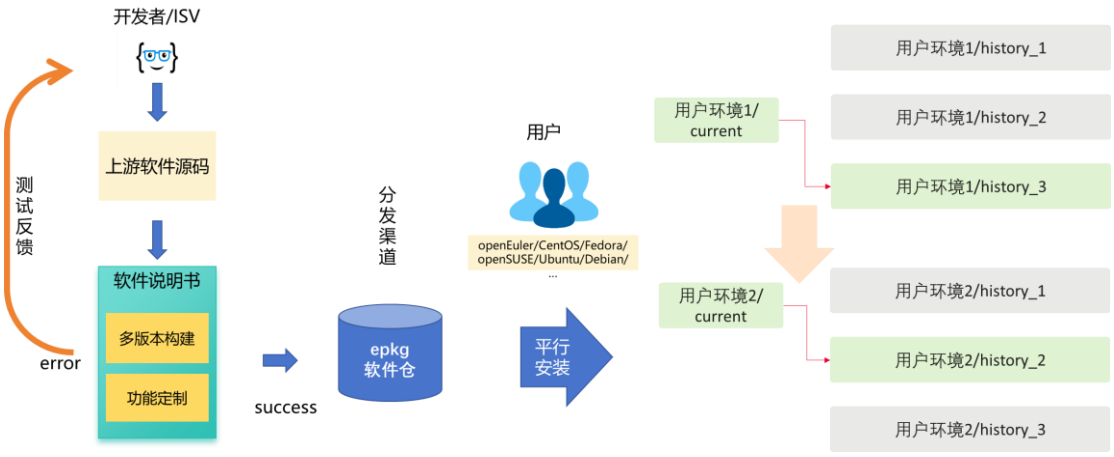
环境管理：支持环境管理，实现环境的创建/切换/使能等操作，用户通过环境的切换，在环境中使用不同的 channel，实现在不同的环境中使用不同版本的软件包。用户可以基于环境，快速实现软件包版本的切换。

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

FOR 开发者：避免面向 OS 的矩阵适配，软件分发渠道，测试反馈服务

FOR 用　户：提供海量的软件包，支持多环境安装、复现、升级，可任意回退



仓库地址

https://gitee.com/openeuler/epkg

QuickIssue 快捷的社区 issue 分类提交工具

QuickIssue 是 openEuler 基础设施团队开发的一个能够满足社区发展的问题跟踪系统。它也是一个更快捷的 issue 分类提交工具，在提交 issue 上具备独特的优势。

功能描述

QuickIssue 是一个更快捷的 issue 分类提交工具，在提交 issue 上有一些独特的优势：

1. QuickIssue 在 openEuler 官网提供统一的 issue 提交入口，开发者在提交 issue 时更便于查找对应仓库。
2. QuickIssue 提供了两种提交 issue 的方式，无论开发者是否有 Gitee 账号，都可以提交 issue。
3. QuickIssue 可以指导用户或开发者将 issue 提交到某个仓库中，也有默认的仓库可供开发者提交 issue。
4. QuickIssue 只为 openEuler 服务，保证查询、搜索、筛选等操作足够顺滑。
5. 可以和社区已有的 SIG 管理、贡献统计等服务互通信息。

QuickIssue 提供三个主要功能：新建 issue、查询 issue、查询 PR，可以为开发者提供更便捷的 issue 提交服务。

新建 issue

1. 统一了 issue 提交入口，openEuler 社区的所有 issue 都可以通过这个入口提交。
2. 解决了 issue 提交人没有代码托管平台账号的问题，可以直接使用邮件和验证码完成 issue 提交。

3. 优化了 issue 提交人查找 issue 归属仓库的途径，可以按图索骥找仓库，也可以直接提交默认仓库。

查询 issue

QuickIssue 服务会展示 openEuler 社区所有的 issue 信息。

针对不同的使用场景，QuickIssue 提供了较为便捷的筛选功能，开发者可以按照自己需求定向查找。如果需要查找通过邮件提交的 issue，可以在提交人一栏输入邮箱前半段筛选。

查询 PR

QuickIssue 会展示 openEuler 社区所有的 PR 信息。开发者通过 PR 状态、提交人、标签等信息可以筛选出满足场景的 PR。QuickIssue 中只包含 openEuler 社区的全量 PR，比代码托管平台自身所管理的 PR 信息量少很多，且系统采用了缓存数据，查询操作响应速度有保障。

应用场景

社区开发者提交 issue 入口，快速搜索查看社区 issue、PR。

仓库地址

https://quickissue.openeuler.openatom.cn/zh/issues

兼容性与技术评测

OSV 技术评测

openEuler OSV 技术测评是在开放原子基金会指导下成立的对 OSV 进行技术测评的技术规范，目前通过 openEuler 创新中心具体实施。

项目介绍

OSV 技术测评清单主要存放通过社区 OSV 基础测评的 OS 厂家、OS 版本信息。

OSV 技术测评主要通过对操作系统内核版本，KABI、内核配置，软件包范围、版本、配置、服务、命令、文件一致性检测，通过对 epol/OEPKGS 等仓库的复用度，运行时一致性检测，保持对 openEuler 系生态软件的可用性。

应用场景

通过技术测评，主要测评 OSV 对 openEuler 技术路线的一致性，保持与社区广泛兼容性生态复用，降低重复迁移适配工作量。

面向 OSV 版本开发过程，看护 OSV 版本前向兼容性。

面向迁移场景，制作差异数据库，与 x2openEuler 迁移工具结合，加速 OS 迁移及升级替代

仓库地址

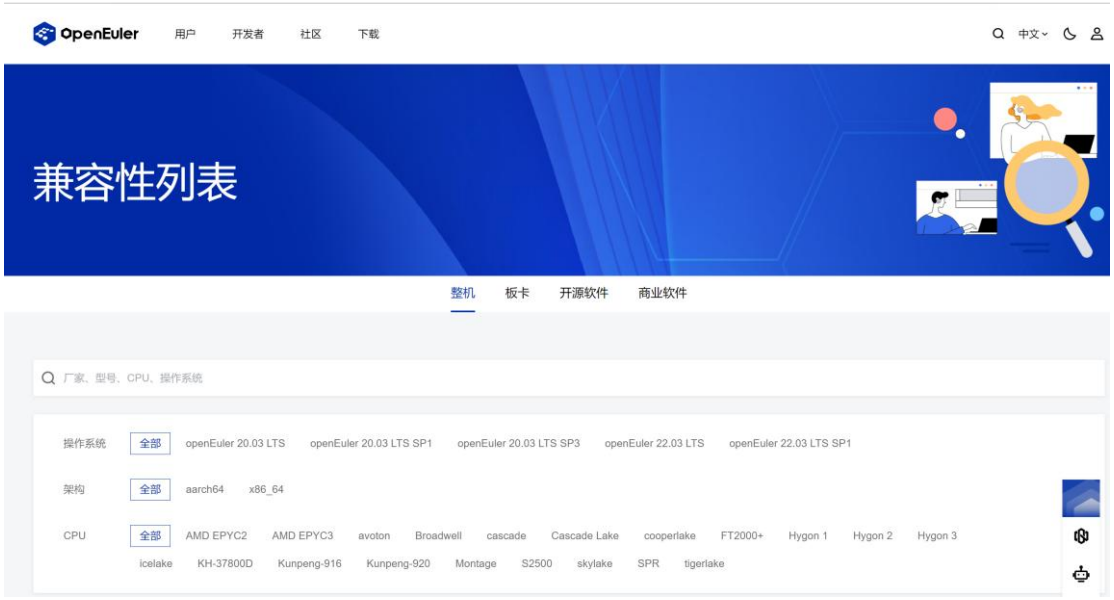
https://gitee.com/openeuler/oecp

openEuler 兼容性全景清单

openEuler 兼容性列表为广大用户提供整机、板卡、开源软件、商业软件查询平台。

项目介绍

openEuler 兼容性清单，分为整机、板卡、开源软件、商业软件等信息，链接参考：https://www.openEuler.org/zh/compatibility/ 。同时开源软件兼容性查询方式通过第三方进行补充，链接参考：https://search.OEPKGS.net/。



硬件兼容性：主要面向南向 CPU 架构、CPU 厂家、整机厂家、整机型号、部件芯片型号、部件芯片厂家等硬件，在社区建立了一套规范、一套流程、一套 CICD 流程，用于指导芯片及板卡厂家在社区建立独立仓库，并基于社区基础设施，保持持续演进；与对应厂家建立芯片功能、版本使能、生态适配三层能力看护，共同保障芯片功能、性能、兼容性、规范性、稳定性。当前已有 marvell、星云智联、云芯智联等公司在社区建仓，并跟随版本例行化运作，相关流程参考：

https://www.openEuler.org/zh/compatibility/hardware/

开源软件兼容性：主要基于业界主流上游活跃项目，按照 openEuler 社区包管理规范集成到 openEuler 社区及扩展社区，保持 openEuler 对主流软件的兼容性，同时建立一套软件包引入平台及机制，便于在 OS 迁移、升级场景快速引入及获取对应版本的软件包，相关流程参考：

https://www.openEuler.org/zh/compatibility/software/

商业软件兼容性：主要面向千行百业 ISV，提供一套对商业软件的测评体系，包括测评规范、流程、方案、工具链，支持 ISV 到创新中心进行测评，测评完成后，提供社区认证的证书，相关流程参考：

https://certification.openEuler.org/

应用场景

通过 openEuler 兼容性清单，用户可以查询及检索 openEuler 版本兼容的 CPU 架构、CPU 厂家、整机厂家、整机型号、部件芯片型号、部件芯片厂家、开源软件、商业软件的兼容性信息。

通过兼容性清单，可以获取 openEuler 兼容性的测试规范、测试方案、流程、相关工具。

通过兼容性清单，获取加入到兼容性的具体操作过程，帮助 IHV/ISV/开发者将关注的软硬件加入到兼容性清单。

仓库地址

https://gitee.com/openeuler/oec-hardware

https://gitee.com/openeuler/oec-application

https://gitee.com/openeuler/technical-certification

openEuler 技术测评

openEuler 技术测评是在开放原子基金会指导下成立的对商业软件、硬件、OSV 进行技术测评的技术规范，目前通过 openEuler 创新中心具体实施。

功能描述

openEuler 技术测评是基于 openEuler 操作系统，在多样性算力平台上，基于社区制定的统一规范，通过统一的设备测评工具、统一的检测标准，构建统一的 openEuler 生态体系。openEuler 技术测评对象主要包括三类：软件、硬件及操作系统。针对软/硬件产品、主要测评其对 openEuler 操作系统的兼容性，针对操作系统产品，主要测评其对 openEuler 技术路线的一致性。

当前广大的软件和硬件伙伴，面向操作系统做兼容性验证时，由于缺少平台化、工具化和自动化能力，面临测试效率低、算力成本高和反复测试等问题。

为此，我们基于 openEuler 社区，开发了 openEuler 生态服务平台，该平台可拉通多算力资源，通过提供资源的统一纳管、测试用例的自动执行、测试报告的自动生成等功能，形成一站式的生态服务，大幅提升伙伴生态服务体验。

1. 测试更便捷：如前所述，通过统一的资源调度，以及环境安装/用例执行/报告生成等全流程自动化平台。
2. 算力更丰富：社区协同 openEuler 生态创新中心，共建全算力硬件资源平台，当前已支持鲲鹏、x86 等主流算力能力，可为伙伴在国产操作系统的迁移、适配、测评中大幅降低算力成本。
3. 服务更高效：通过与基础软件、整机厂商联合互认证，已可为伙伴实现一次 openEuler 测试、多方获取认证证书的能力，大幅提高软件伙伴生态构建效率。

应用场景

通过兼容性技术测评，可以帮助客户从纷杂的产品和解决方案中，快速识别经过严格验证的解决方案；帮助社区通过统一的技术测评体系，繁荣满足操作系统的产业技术生态。

仓库地址

https://gitee.com/openeuler/technical-certification

7 致谢

谨以此章，向 openEuler 社区的贡献者致敬——正因有你们的代码、文档、测试与布道，才有了今天愈加坚韧与开放的 openEuler。

商标声明

在本手册中以及本手册描述的产品中，出现的商标，产品名称，服务名称以及公司名称，由其各自的所有人拥有。

免责声明

本文档可能含有预测信息，包括但不限于有关未来的财务、运营、产品系列、新技术等信息。由于实践中存在很多不确定因素，可能导致实际结果与预测信息有很大的差别。因此，本文档信息仅供参考，不构成任何要约或承诺，不对您在本文档基础上做出的任何行为承担责任。可能不经通知修改上述信息，恕不另行通知。

未经书面同意，任何单位和个人不得擅自摘抄、复制本手册内容的部分或全部，并不得以任何形式传播。