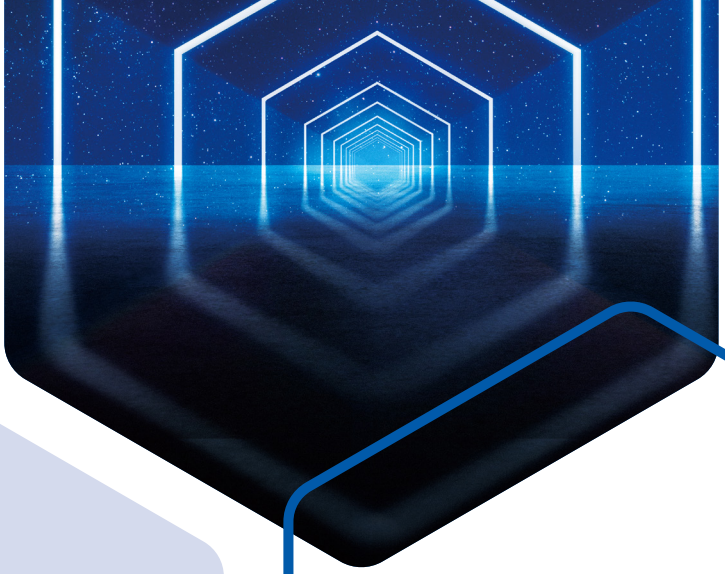
openEuler 22.03 LTS SP3

技术白皮书



CONTENTS

01 概述 01

02 平台架构 04

03 运行环境 07

04 场景创新 10

05 openEuler 内核中的新特性 15

06 云化基座 17

07 特性增强 19

08 著作权说明 57

09 商标 58

10 附录 59

01 概述

openEuler 社区，全称为 OpenAtom openEuler 社区，是一个面向数字基础设施操作系统的开源社区，简称 openEuler 或者 openEuler 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, 还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。

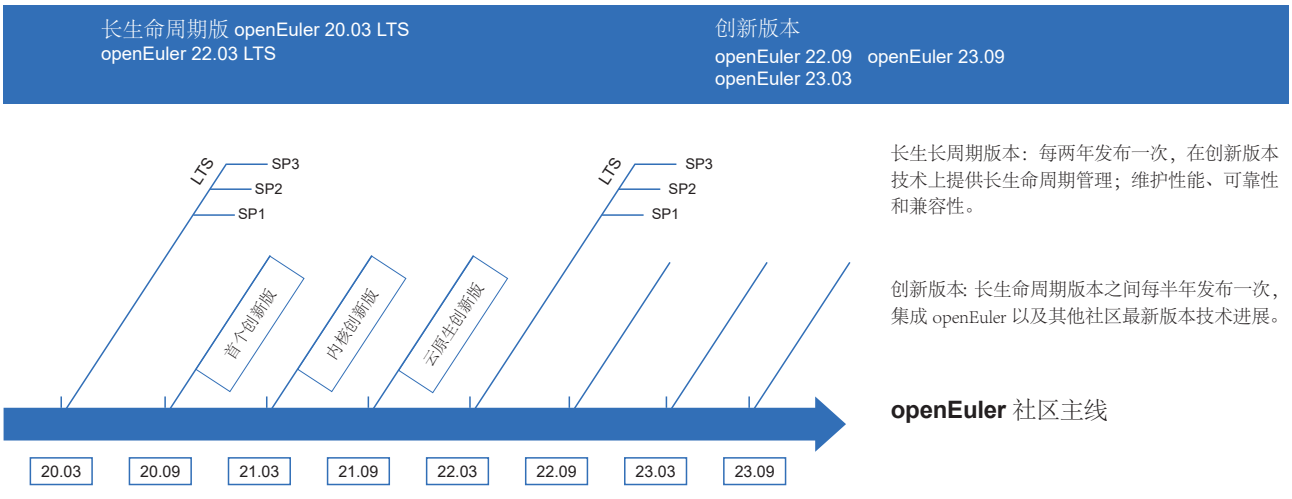
2023 年 6 月 30 日，发布 openEuler 22.03 LTS SP2 版本，场景化竞争力特性增强，性能持续提升。

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。

openEuler 版本管理

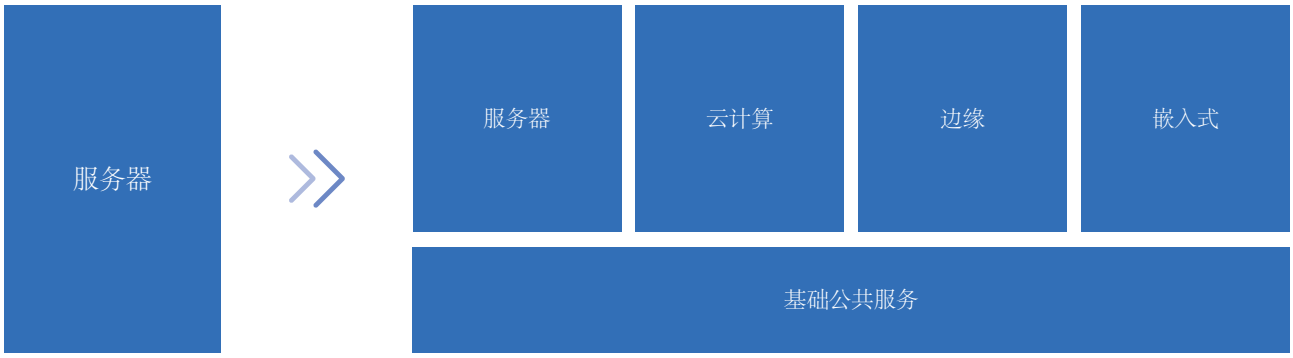


openEuler 作为一个操作系统发行版平台，每两年推出一个 LTS 版本。该版本为企业级用户提供一个安全稳定可靠的操作系统。

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，逐步扩展 PowerPC 等更多芯片架构支持，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，新增发布面向边缘计算的版本 openEuler Edge、面向嵌入式的版本 openEuler Embedded。

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。

02 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 22.03 LTS SP3 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 5.10 构建，对外接口遵循 POSIX 标准，具备天然协同基础。同时 openEuler 22.03 LTS SP3 版本集成分布式软总线、KubeEdge+ 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。

新场景

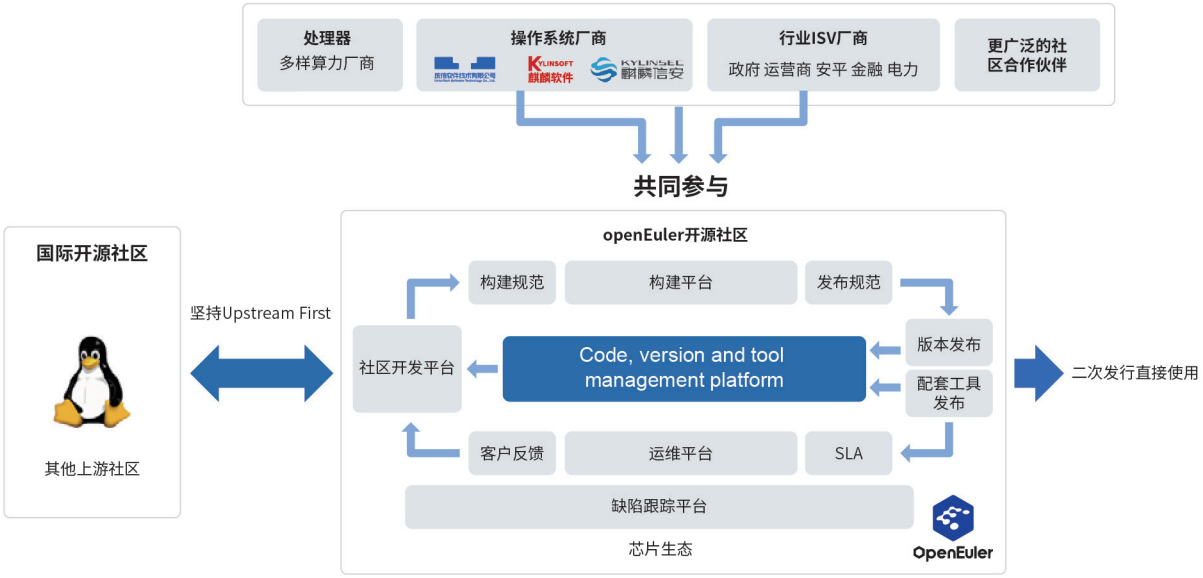
- 边缘计算：发布面向边缘计算场景的版本 openEuler 22.03 LTS SP3 Edge，支持 KubeEdge+ 边云协同框架，具备边云应用统一管理和发放等基础能力。
- 嵌入式： 发布面向嵌入式领域的版本 openEuler 22.03 LTS SP3 Embedded，镜像大小 <5M，启动时间 <5s。

繁荣社区生态

- 友好桌面环境：UKUI、DDE 、Xfce、Kiran-desktop、GNOME 桌面环境，丰富社区桌面环境生态。
- 欧拉 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，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 IceLake/ Sapphire Rapids、AMD EPYC Milan /Genoa 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	Arm
CPU	Intel、AMD、兆芯、海光	鲲鹏、飞腾

支持的整机如下：

硬件，类型	x86	Arm
整机	**Intel**: 超聚变、超微	**鲲鹏：**泰山
	AMD: 超微、新华三	**飞腾：**青松、宝德、新华三
	海光：中科可控	
	兆芯：兆芯	

支持的板卡类型如下：

硬件类型	x86	Arm
网卡	华为、Mellanox、Intel、星云智联、云芯智联、Netswift、云脉、沐创	华为、Mellanox、Intel、星云智联、云芯智联、Netswift、云脉、沐创
Raid	Avago、云芯智联、PMC	Avago、云芯智联、PMC
FC	Marvell、Qlogic、Emulex	Marvell、Qlogic、Emulex
GPU&AI	Nvidia	Nvidia
SSD	华为	华为

全版本支持的硬件型号可在兼容性网站查询：https://www.openeuler.org/zh/compatibility/。

03 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：https://openeuler.org/zh/compatibility/。

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

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-22.03-LTS-SP3 为 HostOS，软件包版本如下：

- libvirt-6.2.0-60.oe2203sp3
- libvirt-client-6.2.0-60.oe2203sp3
- libvirt-daemon-6.2.0-60.oe2203sp3
- qemu-6.2.0-86.oe2203sp3
- qemu-img-6.2.0-86.oe2203sp3

2. 兼容的虚拟机列表如下：

Host OS	GuestOS(虚拟机)	架构
openEuler 22.03 LTS SP3	CentOS 6	x86_64
openEuler 22.03 LTS SP3	CentOS 7	AArch64
openEuler 22.03 LTS SP3	CentOS 7	x86_64
openEuler 22.03 LTS SP3	CentOS 8	AArch64
openEuler 22.03 LTS SP3	CentOS 8	x86_64
openEuler 22.03 LTS SP3	Windows Server 2016	AArch64
openEuler 22.03 LTS SP3	Windows Server 2016	x86_64
openEuler 22.03 LTS SP3	Windows Server 2019	AArch64
openEuler 22.03 LTS SP3	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 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
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

04 场景创新

AI 专项

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。

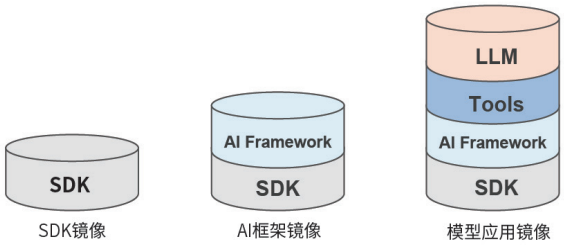
openEuler for AI

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，即可简化部署过程，提供开箱即用的体验。

功能描述

1）openEuler 已兼容 CANN，CUDA 等硬件 SDK，以及 TensorFlow、PyTorch 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。
- **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。

AI for openEuler

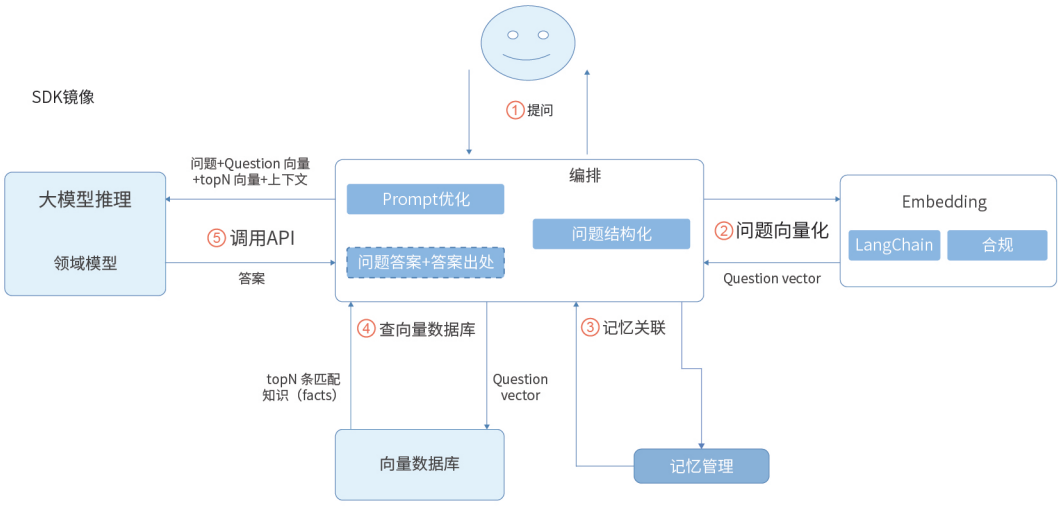
当前，欧拉和 AI 深度结合，一方面使用基础大模型，基于大量欧拉操作系统的代码和数据，训练出 EulerCopilot，初步实现代码辅助生成、智能问题智能分析、系统辅助运维等功能，让 openEuler 更智能。

EulerCopilot- 智能问答

功能描述

EulerCopilot 智能问答平台目前支持 web 和智能 shell 两个入口。

- Web 入口：操作简单，可咨询操作系统相关基础知识，openEuler 动态数据、openEuler 运维问题解决方案、openEuler 项目介绍与使用指导等等。
- 智能 Shell 入口：自然语言和 openEuler 交互，启发式的运维。



应用场景

1、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。

2、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。

3、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。

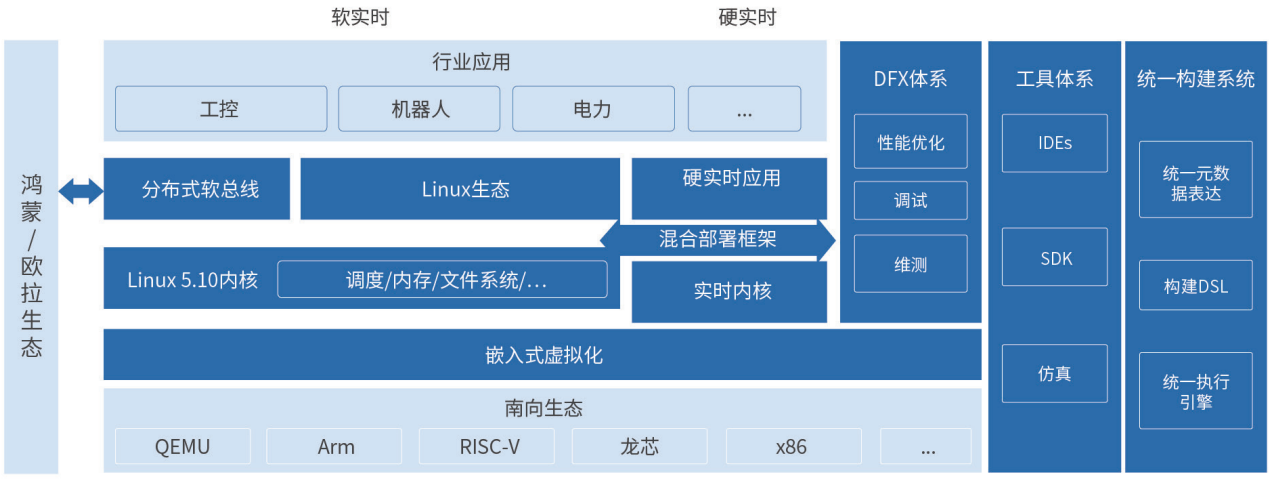
相关使用方式请参考 EulerCopilot 智能问答服务使用指南。

嵌入式

openEuler 发布面向嵌入式领域的版本 openEuler 22.03 LTS SP3 Embedded，提供更加丰富的嵌入式软件包构建能力，支持实时 / 非实时平面混合关键部署，并集成分布式软总线。

openEuler Embedded 围绕工业和机器人领域持续深耕，通过行业项目垂直打通，不断完善和丰富嵌入式技术栈和生态。openEuler 22.03 LTS SP3 Embedded 支持嵌入式虚拟化弹性底座，提供 Jailhouse 虚拟化方案、openAMP 轻量化混合部署方案，用户可以根据自己的使用场景选择最优的部署方案。同时支持 ROS humble 版本，集成 ros-core、ros-base、SLAM 等核心软件包，满足 ROS2 运行时要求。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 RISC-V、龙芯等芯片架构，丰富工业中间件、ROS 中间件、仿真系统等能力，打造嵌入式领域操作系统解决方案。

系统架构图



南向生态

openEuler Embedded Linux 当前主要支持 ARM64、x86-64 两种芯片架构，22.03 LTS SP3 版本新增支持 RK3588 芯片，未来还将支持龙芯、飞腾等芯片。

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 FreeRTOS 和 openEuler Embedded Linux 混合部署。

3）实时虚拟化：openEuler 社区自研虚拟化 ZVM，兼顾性能、隔离性和灵活性，综合最优。目前支持 Zephyr 和 Linux 混合部署。

混合关键性部署框架

openEuler Embedded 的混合关键性部署框架构建在融合弹性底座之上，通过一套统一的框架屏蔽下层弹性虚拟化底座形态的不同，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。

混合关键性部署框架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）的生命周期管理、跨 OS 通信。

北向生态

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 FreeRTOS 和 openEuler Embedded Linux 混合部署。

3）实时虚拟化：openEuler 社区自研虚拟化 ZVM，兼顾性能、隔离性和灵活性，综合最优。目前支持 Zephyr 和 Linux 混合部署。

硬实时系统（UniProton）

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：

1）支持 Cortex-M、ARM64、X86_64 架构，支持 M4、RK3568、X86_64、Hi3093、树莓派 4B。

2）支持树莓派 4B、Hi3093、X86_64 设备上通过裸金属模式和 openEuler Embedded Linux 混合部署。

3）支持通过 gdb 在 openEuler Embedded Linux 侧远程调试。

4）支持 890+ POSIX 接口，支持文件系统、设备管理、shell 控制台、网络。

应用场景

嵌入式系统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。

openEuler 内核中的新特性

05

openEuler 内核中的新特性

openEuler 22.03 LTS SP3 基于 Linux Kernel 5.10 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：

- 内存动态隔离和释放：提供安全、稳定的内存页面动态隔离与解除隔离的功能，支持隔离时安全迁移原内存。隔离过程中对待隔离、取消隔离页面进行额外的记录与检查，确保页面的来源可靠，降低内核风险。 通过本特性隔离的 UCE 风险页面在进行数据复制时，基于 MCS 做到触发 UCE 内核免复位，提升风险操作的内核可靠性。
- 支持 CPU 在线巡检：静默数据损坏 (SDC) 可能导致数据丢失和数据被破坏。通过执行巡检指令，发现存在静默故障的核，提前对故障核进行隔离，避免出现更严重的故障，提高系统可靠性。
- 负载算力协同：在多核服务器中运行用户体验敏感应用（如云桌面系统）时，通过负载算力协同技术能够保障算力供给的及时性和有效性。负载算力协同技术具有以下特性：高负载场景下，支持轻量级的任务搜索算法，提高空闲 CPU 拉取 runnable 任务的效率，实现多核间快速负载均衡，最大化 CPU 资源利用率；算力竞争场景下，支持按优先级对业务进行分级管控，有效避免优先级翻转的问题，实现高优先级的前台任务绝对压制低优级的后台任务，保障关键任务的算力供给。
- 支持功耗感知调度：在面向业务层面收集访存带宽，CPU 负载等数据，确保业务关键线程资源得到满足。引入物理拓扑，调压域感知新的维度，减少单 DIE 调频、单 DIE 调压带来的调频降功耗的局限，保障在低负载下能最小化功耗。

  1）根据物理拓扑构建逃逸通道，根据 CPU 负载和访存带宽瓶颈自动调节节能级别，低负载集中业务减少功耗，高负载扩散业务的策略保证 QOS。

  2）定时器收集负载，带宽等信息，感知访存带宽，避免跨 DIE 访问。

  3）OS 新增调压域、idle 静态功耗感知、业务标签等机制，实现智能感知调频、CPU idle、DVFS，来优化业务性能。

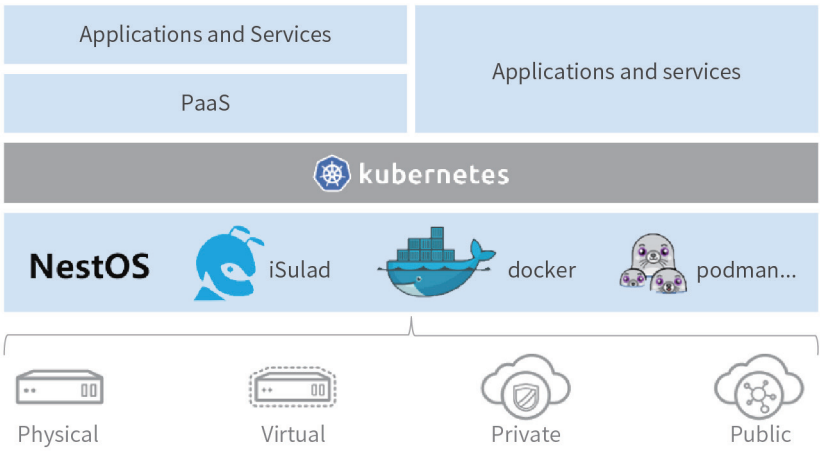
- 核隔离特性增强：该特性将系统 CPU 分成了 housekeeping 和 non-housekeeping 两部分，non-housekeeping cpu 主要用于执行业务进程，housekeeping cpu 主要运行 OS 周期性的时钟维护等背景进程等噪声。该特性将 OS 背景进程、中断等噪声集中在 housekeeping CPU 上，防止这些噪声对业务进程运行产生影响，提升业务性能，多用于 HPC 业务场景。
- 支持对资源竞争度量及处理性能监控单元指标低负载采集：在全局和 cgroup v1 中支持对于 CPU/IO/memory/irq 的压力采集和监测（PSI），并且对于 CPU/memory 的压力来源，进行细粒度的划分；使得多业务共享节点资源场景下，通过 PSI 等指标度量系统资源竞争，反映业务进程的吞吐和延时，分析系统资源瓶颈，可以协助了解特定业务进程的资源需求，动态调节业务的部署和资源的分配，以保证在线业务的服务质量和系统的健康程度。
- KVM TDP MMU：是 Linux 5.10 之后，内存虚拟化领域中用于提升 KVM 可扩展性的一个改进方案，由 Intel 提交到 openEuler 社区。相对于传统的 KVM MMU，TDP MMU 提供了更高效的并发 Page Fault 处理机制，从而使得 KVM 对大型虚拟机（多 vCPU，大内存）有了更好的支持。于此同时，通过采用新的 EPT/NPT 遍历接口，KVM TDP MMU 摒弃了传统内存虚拟化中对 rmap 数据结构的依赖，使得主机内存有了更好的利用率。openEuler 22.03 SP3 提供了 Linux 5.10 至 5.15 版本的 KVM TDP MMU 支持，后续优化将在 SP4 版本保持跟进。

06 云化基座

NestOS 容器操作系统

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件能力，可以更安全的运行大规模的容器化工作负载。

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，解决了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。

07 特性增强

SysCare 热补丁能力

在 Linux 世界，有一个困扰大家已久的难题：如何在不影响业务的情况下，快速可靠地修复漏洞、解决故障。

当前常见的方法是采用热补丁技术：在业务运行过程中，对问题组件直接进行代码级修复，业务无感知。然而，当前热补丁制作方式复杂，补丁需要代码级匹配，且管理困难，特别是用户态组件面临文件形式、编程语言、编译方式、运行方式的多样性问题，当前还没有简便统一的补丁机制。

为了解决热补丁制作和管理的问题，SysCare 应运而生。

SysCare 是一个系统级热修复软件，为操作系统提供安全补丁和系统错误热修复能力，主机无需重新启动即可修复该系统问题。SysCare 将内核态热补丁技术与用户态热补丁技术进行融合统一，用户仅需聚焦在自己核心业务中，系统修复问题交予 SysCare 进行处理。后期计划根据修复组件的不同，提供系统热升级技术，进一步解放运维用户提升运维效率。

功能描述

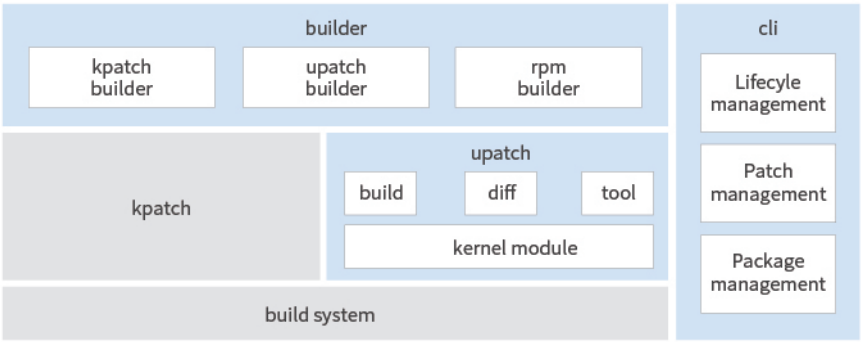
1. 热补丁制作

用户仅需输入目标软件的源码 RPM 包、调试信息 RPM 包与待打补丁，无需对软件源码进行任何修改，即可生成对应的热补丁 RPM 包。

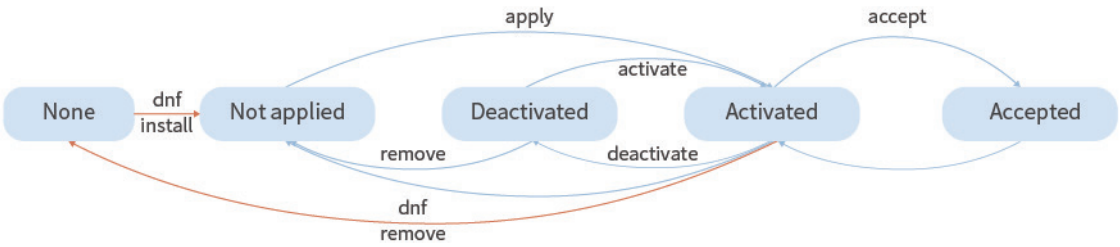
2. 热补丁生命周期管理

SysCare 提供一套完整的，傻瓜式补丁生命周期管理方式，旨在减少用户学习、使用成本，通过单条命令即可对热补丁进行管理。依托于 RPM 系统，SysCare 构建出的热补丁依赖关系完整，热补丁分发、安装、更新与卸载流程均无需进行特殊处理，可直接集成放入软件仓 repo。

SysCare 整体架构



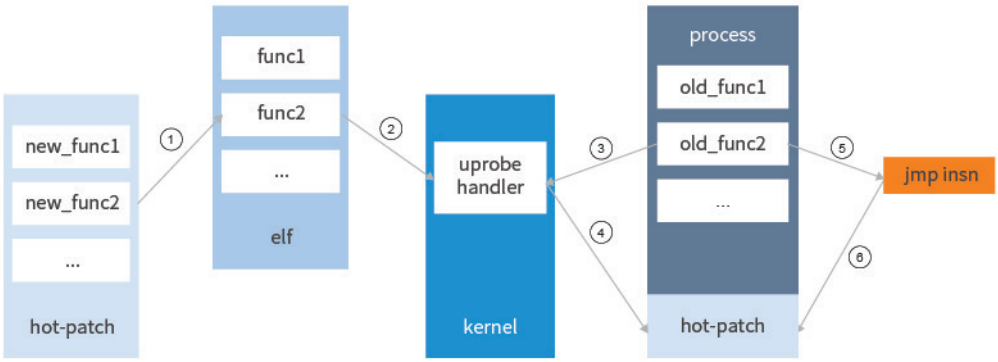
热补丁生命周期



3. 针对 ELF 文件（程序可执行文件）的用户态热补丁

使用 uprobe 技术，将热补丁与 ELF 文件绑定。在 ELF 文件运行时，通过 uprobe 触发补丁生效，这样无需监控进程。因此，无论进程是否已经运行都可以在打补丁后或新进程运行时使补丁生效。同时，该技术也可以给动态库打热补丁，解决了动态库热补丁的难题。补丁生效流程如下图所示。

补丁生效流程



1）执行 uprobe 系统调用，在待修改函数 func 处增加 uprobe 断点。

2）注册 uprobe handler。

3）进行运行到 func 时调用 uprobe handler。

4）将 patch 映射到当前进程地址空间。

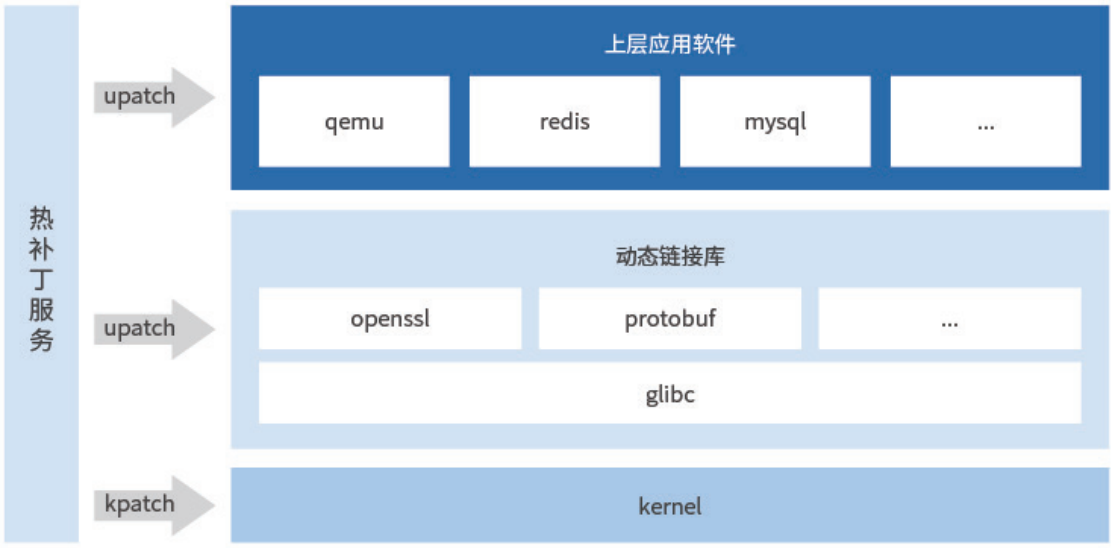
5）进行安全检查并将 func 的第一条指令修改为 jump 指令，指向 patch 地址。

6）跳转到 patch 地址执行。

4. 内核热补丁与用户态热补丁融合

SysCare 基于 upatch 和 kpatch 技术，覆盖应用、动态库、内核，自顶向下打通热补丁软件栈，提供用户无感知的全栈热修复能力。

热补丁应用范围



5. 新增特性

- 支持制作热补丁时配置热补丁软件包依赖关系。
- 支持用户态补丁多补丁管理。
- 支持用户态热补丁冲突检测。
- 支持用户态热补丁强制覆盖冲突。

6. 约束限制

- 当前仅支持 64 位系统。
- 当前仅支持 ELF 格式的热修复，不支持解释型语言，不支持纯汇编修改。
- 当前仅支持 GCC / G++ 编译器，且不支持交叉编译。
- 暂不支持 LTO 优化。

应用场景

应用场景 1： CVE 补丁快速修复。

应用场景 2： 现网问题临时定位。

GCC for openEuler

GCC for openEuler 编译器是面向 openEuler 生态的高性能编译器，基于开源 GCC（GNU Compiler Collection，GNU 编译器套装）开发和全场景优化。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，增强指令优化、内存优化、自动向量化等特性，并适配更多硬件平台，如鲲鹏、飞腾、龙芯等，充分释放国产硬件算力，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，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。

1. 极致性能：GCC for openEuler 通过拓展结构体优化、鲲鹏亲和优化、反馈优化、流水线优化等多种编译优化能力，基础性能测试 SPECCPU2017 intrate 超开源 GCC 10.3 20%，云场景典型应用 MySQL、Redis、Nginx、ceph 场景化性能提升 5%~10%。
2. 便捷易用：GCC cross 编译器工具链支持交叉编译，面向嵌入式场景，已用于 openEuler Embedded 构建系统。编译器插件框架无需深入修改编译器内部逻辑，帮助开发者可以实现独立编译优化和编译工具的便捷开发。
3. 生态成熟：GCC for openEuler 支持 C/C++、Fortran、Objective-C/C++ 等编程语言，并对分布式存储、数据库、Web、核心网等领域的典型应用进行了优化增强，最高可达 15% 的性能提升。架构及芯片上，支持 ARM、X86、RISC-V、SW-64、LoongArch 等架构，广泛涵盖了国内的鲲鹏、飞腾、兆芯、海光、龙芯、申威，以及国外的 Intel、AMD 等芯片算力。
4. 可靠可信：GCC 依托开源生态，覆盖基础功能测试、可靠性测试、兼容性测试等，生成百万测试用例，全面构筑质量加固防护网。同时依托 openEuler 社区漏洞处理机制，12H 内公开漏洞感知率 90% 以上，防止恶性攻击事件。

GCC for openEuler 新增能力主要包括以下 3 点：

1. 多版本 GCC 共存支持：提供以 GCC 12.3.0 为基线的 gcc-toolset-12 系列软件包，主要支持 OpenMP 语言规范，其中 fortran 支持 OpenMP 4.5 语言规范，C/C++ 支持部分 OpenMP 5.0 语言规范。
2. LLC 分配优化特性：通过分析程序中主要的执行路径，对主路径上的循环进行访存的复用分析，计算排序出 TOP 的热数据，并插入预取指令将数据预先分配至 LLC 中，减少 LLC miss。
3. 新增 10 余个 CPUBench 优化特性。主要通过对特定场景的智能识别并缩减指令，达到性能提升。

功能描述

多版本 GCC 支持：

- **LLC 分配优化**

  1）支持将程序内识别到的热数据预取分配至 LLC 中，降低 LLC miss，达到性能提升。

  2）新增选项 -fllc-allocate。

- **基础性能优化**

  1）CRC 优化：识别 CRC 软件循环代码，生成高效硬件指令。

  2）If-conversion 增强：增强 If conversion 优化，使用更多的寄存器以减少冲突。

  3）乘法计算优化：Arm 相关指令合并优化，实现低位乘法算法的识别，并以高效的高位乘法指令输出。

  4）cmlt 指令生成优化：对一些四则运算生成 cmlt 指令，减少指令数。

  5）向量化优化增强：识别并简化向量化过程中生成的冗余指令；允许更短的数组进行向量化处理。

  6）maxmin 和 uzp1/uzp2 指令联合优化：识别 maxmin 和 uzp1/uzp2 指令联合优化机会，减少指令数从而提升性能。

  7）ldp/stp 优化：识别某些性能表现差的 ldp/stp，将其拆分成 2 个 ldr 和 2 个 str。

  8）AES 指令优化：识别 AES 算法代码，使用硬件指令加速。

应用场景

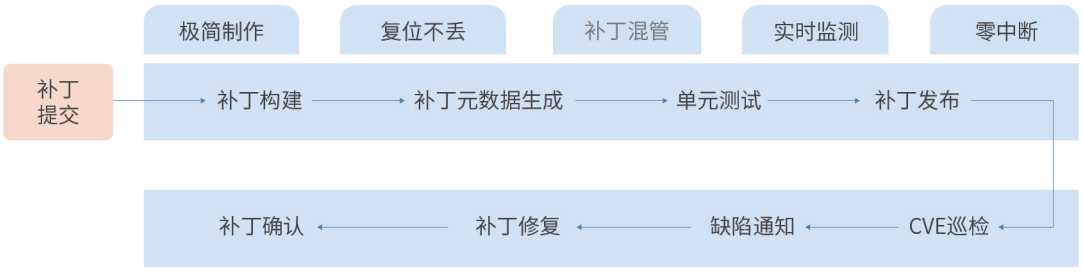
通用计算领域，运行 SPECCPU 2017 测试，相比于上游社区的 GCC 10.3 版本可获得 20% 左右的性能收益；运行 CPUBench 测试，相比于上游社区的 GCC 10.3 版本可获得 10% 左右的性能收益。

高性能计算领域，运行 WRF、NEMO 应用，相比于上游社区的 GCC 10.3 版本能够获得 10% 左右的性能收益。通过对 OpenFOAM、WRF、Relion 等 HPC 应用的热点函数中的热数据优先预取分配至 LLC 中，可有效减少 LLC miss，提高运行效率。

A-Ops 智能运维

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，故障修复到修复任务回退的智能运维解决方案。A-Ops 项目包括了诺干子项目：覆盖故障发现（gala），故障定位支撑（X-diagnosis），缺陷修复（apollo）等。

本次发布的 aops-apollo 项目是智能补丁管理框架，集成了漏洞扫描、CVE 修复（冷补丁 / 热补丁）、热补丁回退等核心功能。系统可以对发布的安全公告实行定时下载同步，可设置定时任务执行漏洞扫描，保证系统平稳运行的同时，运维人员可通过 AOps 工具实现对漏洞的修复和回退。



AOps补丁管理平台

功能描述

aops-apollo 内核智能补丁管理

- 热补丁源管理：openEuler 的漏洞信息通过安全公告对外发布时，会同时在 update 源中发布修复所用的软件包。 默认 openEuler 系统安装后自带对应 OS 版本的冷补丁 update 源，用户也可以通过设置 repo 来自行配置冷 / 热补丁的 update 源。
- 缺陷扫描：通过对集群手动和定时扫描，检查集群是否受 CVE 影响，并提供冷 / 热补丁修复选择。
- 冷热补丁混合管理：支持冷补丁、热补丁独立修复，也支持冷热补丁混合修复，实现在网热补丁静默收编，减少热补丁维护成本。
- 热补丁生命周期管理：热补丁移除，回退，查询等生命周期管理。

应用场景

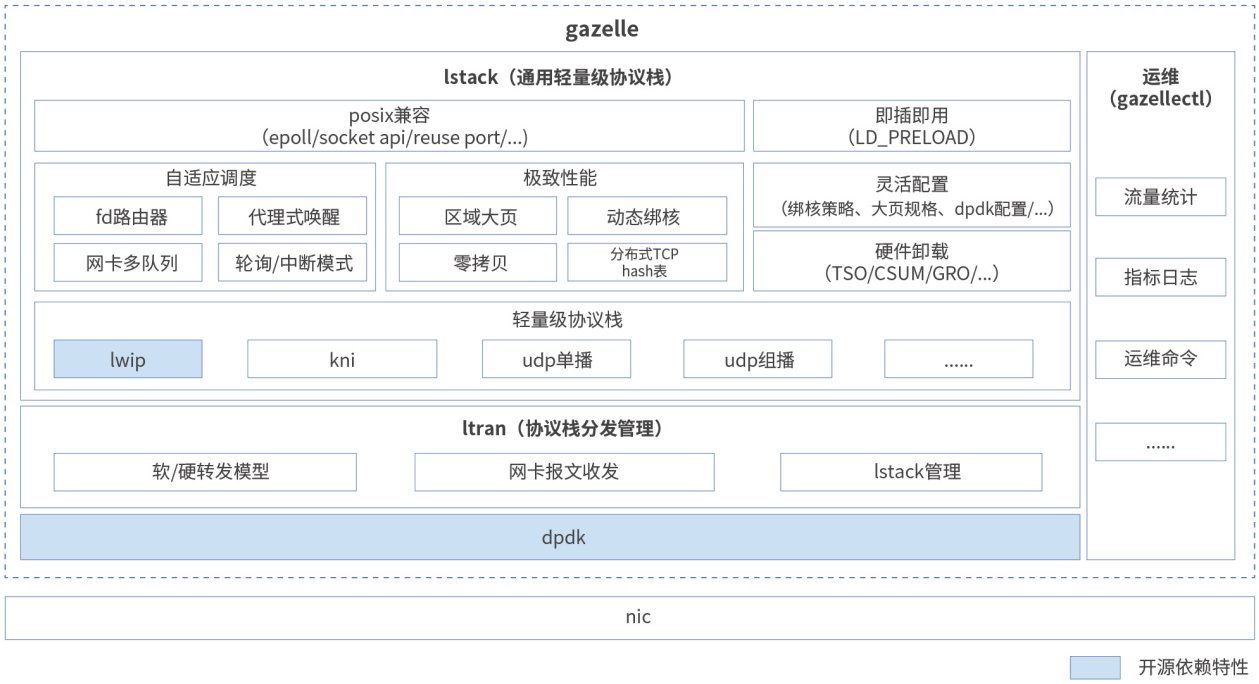
针对有运维诉求的管理员或个人开发者，可以通过漏洞管理服务，对单机 / 集群快速实现内核漏洞感知、修复，基于任务粒度对冷热补丁进行回退，极大减少补丁管理成本，保障集群安全，提升漏洞修复效率。

Gazelle 特性增强

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，兼顾高性能与通用性。本次版本新增 UDP 协议及相关接口支持，丰富用户态协议栈。

功能描述

gazelle 功能架构图



- 高性能（超轻量）：基于 dpdk、lwip 实现高性能轻量协议栈能力。
- 极致性能：基于区域大页划分、动态绑核、全路径零拷贝等技术，实现高线性度并发协议栈。
- 硬件加速：支持 TSO/CSUM/GRO 等硬件卸载，打通软硬件垂直加速路径。
- 通用性（posix 兼容）：接口完全兼容 posix api，应用零修改，支持 udp 的 recvfrom 和 sendto 接口。
- 通用网络模型：基于 fd 路由器、代理式唤醒等机制实现自适应网络模型调度，udp 多节点的组播模型，满足任意网络应用场景。
- 易用性（即插即用）：基于 LD_PRELOAD 实现业务免配套，真正实现零成本部署。
- 易运维（运维工具）：具备流量统计、指标日志、命令行等完整运维手段。

新增特性

- 新增支持单 vlan 模式、bond4 与 bond6 模式、网线插拔后网卡自愈功能。
- 全面支持鲲鹏 920 虚拟机单实例 redis 应用，最大支持链接数 5k+，性能提升约 30%+。
- 支持 netperf TCP_STREAM/TCP_RR（包长 1463 Byte 以下）参数测试。
- 对 gazelle 的 lstack、lwip、gazellectl 模块日志增强，便于定位。

验证过的网卡

MLNX CX4/CX5 网卡。

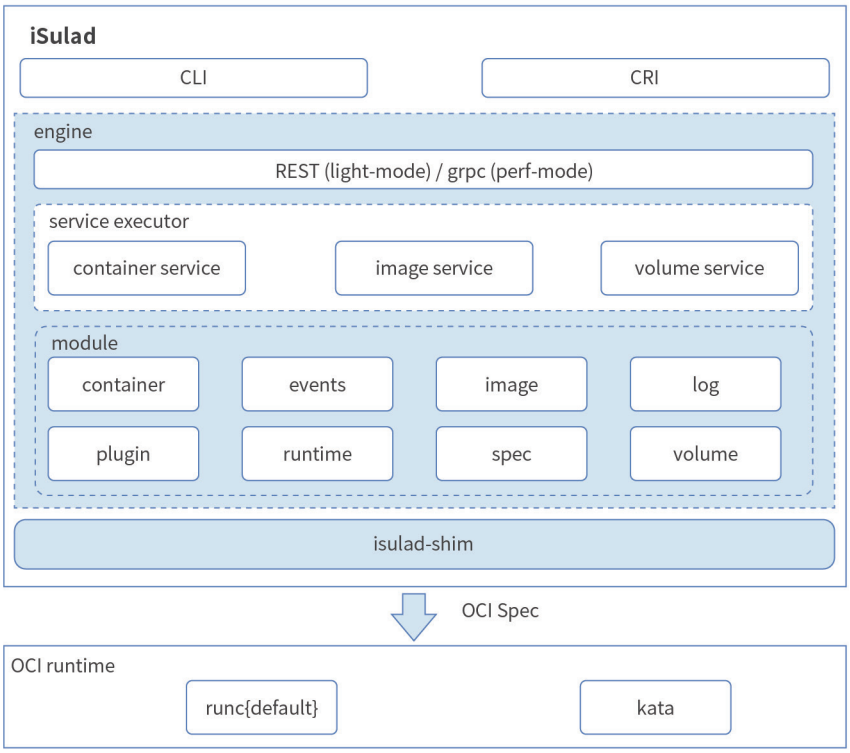
应用场景

适用于提升网络协议栈成为性能瓶颈点的应用提升业务处理性能。

iSulad 支持 OCI runtime

OCI（Open Container Initiative）是一个轻量级、开放的治理项目，在 Linux 基金会的支持下成立，致力于围绕容器格式和运行时创建开放的行业标准，使得任何符合 OCI 运行时标准的容器运行时都可以使用 OCI 镜像来运行容器，OCI 的出现使得整个容器社区都在朝着标准化的方向发展。iSulad 作为一款轻量级容器引擎，南向支持标准 OCI 接口，能够灵活对接 runc、kata 等多种 OCI 运行时，一步到位兼容容器生态。

功能描述



随着 OCI 标准规范的日益成熟，符合 OCI 运行时标准规范的容器运行时能满足各种场景需求。而 runc 是第一个 OCI Runtime 的参考实现，也是业界广泛使用的 OCI runtime。因此，针对 iSulad 对接 oci runtime 的功能进行了统一排查，修复了原有支持功能的缺陷，并补齐了 isula top 接口与 isula attach 接口，完善了与 OCI runtime 对接的功能之后切换 iSulad 的默认运行时为 runc。切换默认运行时后，修改与 oci runtime 对接的 isulad-shim 的依赖库为独立且裁剪后的静态工具库，与切换前相比，能有效防止由于工具库升级导致的已有进程崩溃问题，同时运行的容器具有底噪更低的优势。

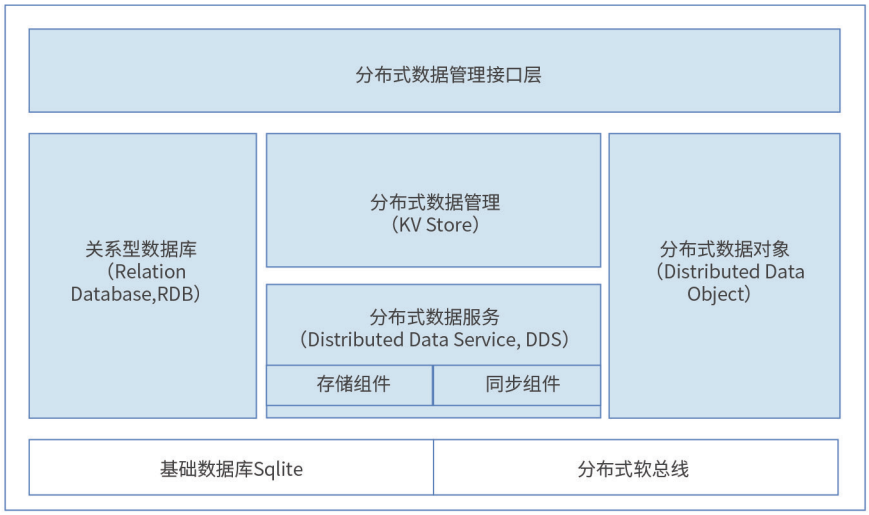
应用场景

可以应用于容器引擎 + oci runtime 场景中，切换默认运行时为 runc 后，还可以应用于底噪需求更低以及热升级的场景中。

基于软总线的分布式数据管理系统

分布式数据管理系统是从 OH 社区迁移而来的在软总线生态之上的一个数据管理能力，其在分布式软总线动态组网的基础上封装了 100+ 的的通用接口，为网络上各个设备结点提供可选的强一致或弱一致性等不同的数据同步能力。

功能描述



- **关系型数据库：**是一种基于关系模型来管理数据的数据库，底层使用 SQLite 作为持久化存储引擎，支持 SQLite 具有的所有数据库特性。
- **KV 数据库：**基于 SQLite 实现的 KV 数据库，提供键值管理能力，为应用程序提供设备间数据的分布式协同能力。
- **分布式数据对象：**分布式数据对象管理框架是一款面向对象的内存数据管理框架，让开发者能以使用本地对象的方式在设备间使用相同应用的数据对象。
- **分布式数据服务：**在通过可信认证的设备间，分布式数据服务支持数据相互同步，为用户提供在多种终端设备上一致的设备间数据访问体验。
- **分布式软总线：**网络链路层的发现连接。
- **SQlite：**开源三方件，原生 SQLite 能力。

软总线容器化支持

当前业务软件迁向容器化的已经是大势所趋，所以在新版本中使能了软总线以及相关依赖软件的容器化部署，以及多客户端的支持，大大简化了服务安装部署测试步骤，可以更方便地与业务软件兼容，便于在社区上推广应用。

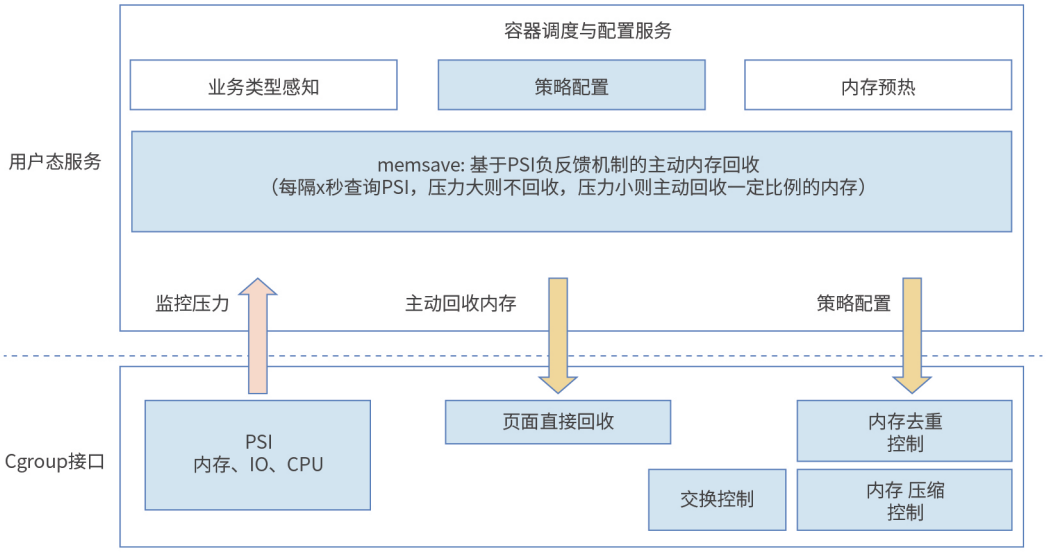
应用场景

基于 openEuler 的分布式数据管理，在 openEuler 嵌入式、边缘服务器以及 OH 原生设备之间，只需要少量开发代码，就能便捷地进行数据同步，跨平台算力协同。

内存超分

内存超分面向云原生容器场景，提供高效内存回收机制，实现内存可用空间的提升。本特性支持 cgroup 粒度的：内存去重、内存主动回收、内存压缩、swap 使用量限制与 swap 设备限制。除此之外，本特性针对内存回收，压缩算法等关键内核模块进行了性能优化，包括二次压缩算法、TLB 批量刷新、透明大页交换等性能优化。最后，本特性提供用户态主动回收示例服务程序：memsave，供用户参考，通过主动回收接口定时触发容器粒度的内存回收，并通过 PSI 机制自适应调整回收量，在节省容器内存使用的同时，尽可能减少对容器业务性能的影响。

功能描述



cgroup 内存资源管控与策略扩展

- **内存主动回收功能：**支持直接回收指定类型的内存，包括单独指定的回收文件页、匿名页。
- **基于水线的回收控制功能：**支持配置 min、low、high 水线做“被动”回收，支持异步后台回收避免影响业务性能。
- **内存去重功能** 支持方便快捷得使能容器内进程的内存全量计入 KSM 去重，应用无需调用 madvise 来标记参与去重的内存区域。
- **Swap 空间隔离与控制功能：**支持容器单独配置 swap 后端设备（例如 zram 设备、存储设备等）、swap 空间使用上限、主动换入功能、swap 启用开关。

基础优化

- **内存压缩：**支持 zram 二次压缩，兼顾不同压缩算法在压缩率、压缩 / 解压缩速度上的优势。
- **内存回收：**优化 unmap、migrate 等流程 TLB 刷新；优化回收速度，降低回收流程锁冲突；优化透明大页交换。

业务基于 PSI 机制最优化决策

- cgroup v1/v2 支持 PSI。
- 基于 PSI 负反馈机制主动回收内存。结合业务负载与集群情况共同决策，避免内存超分对性能 / 可靠性影响。

应用场景

本特性适用于容器混部、serverless 服务等云原生场景，致力于达到同等内存成本下更高的服务质量的效果：业务服务数，业务部署数量；或者同等业务量下可以配置更少的内存。同时尽量减少对业务性能的影响。

内存超分在部分场景下可能会对业务造成负面影响，需要充分评估并选择合适方案搭配使用， 例如在容器混部场景下，可以尽可能多的回收离线任务的内存，尽可能保障在线业务等访存敏感的任务，比如针对在线业务，可以配置压缩内存 zram 设备为 swap 后端以降低对性能的影响，或者针对该类容器关闭回收服务；针对离线任务可以激进得主动回收内存。

动态完整性度量特性

DIM（Dynamic Integrity Measurement）动态完整性度量特性通过在程序运行时对内存中的关键数据（如代码段）进行度量，并将度量结果和基准值进行对比，确定内存数据是否被篡改，从而检测攻击行为，并采取应对措施。

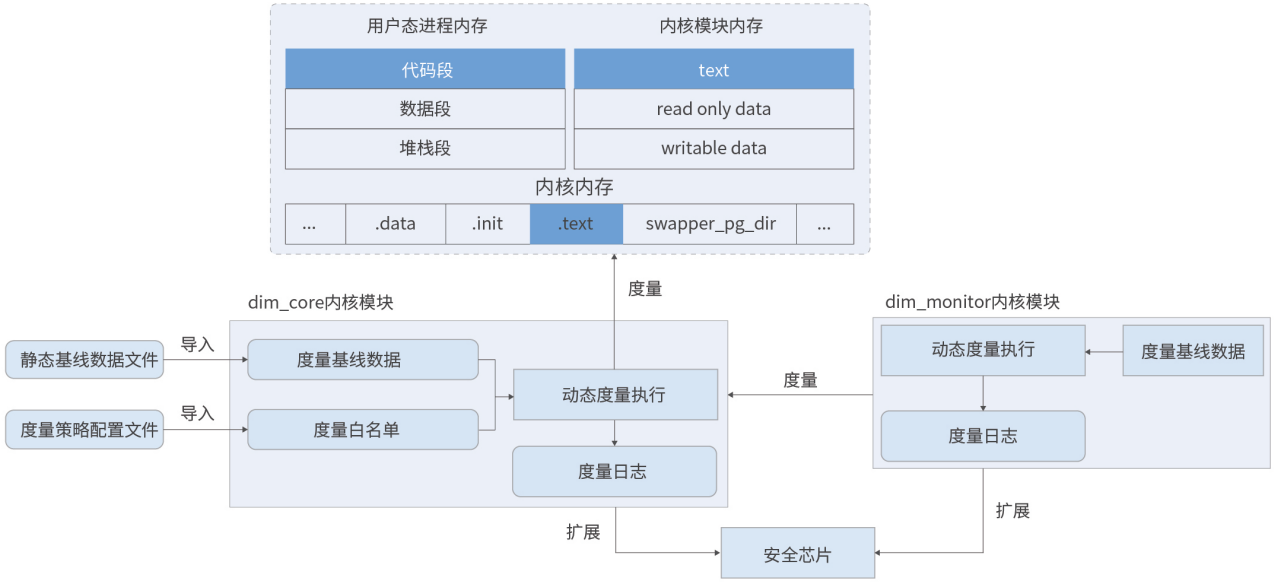
功能描述

DIM 动态完整性度量特性支持如下功能：

- 支持度量用户态进程、内核模块、内核内存代码段数据。
- 支持将度量结果扩展至 TPM 2.0 芯片 PCR 寄存器，用于对接远程证明。
- 支持配置度量策略，支持度量策略签名校验。
- 支持工具生成并导入度量基线数据，支持基线数据签名校验。
- 支持配置国密 SM3 度量算法。

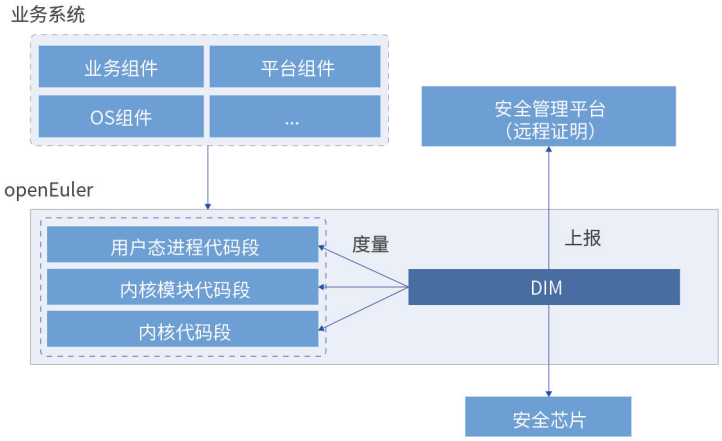
DIM 动态完整性度量特性包含两个软件包 dim_tools 和 dim：

- dim_tools：提供 dim_gen_baseline 命令行工具，通过解析 ELF 二进制文件生成指定格式的代码段度量基线。
- dim：提供 dim_core 和 dim_monitor 内核模块。dim_core 为动态完整性度量核心模块，解析并导入用户配置的度量策略和度量基线，获取内存中的度量目标数据并执行度量功能；dim_monitor 对 dim_core 的代码段和关键数据执行度量保护，防止由于 dim_core 被篡改而导致度量功能失效。



应用场景

DIM 动态完整性度量特性可作为 OS 提供的基础安全机制，为信息系统各个组件提供内存数据的完整性保护。其典型使用场景如下。



用户通过配置动态完整性度量策略，为系统中关键程序配置动态度量功能。DIM 完成目标数据度量后，将度量结果上报至安全管理平台，用户可以查询当前系统中的关键进程内存数据是否完整。在安全要求较高的场景下，用户还可对接可信计算远程证明机制，通过 TPM 证明度量结果的完整性。

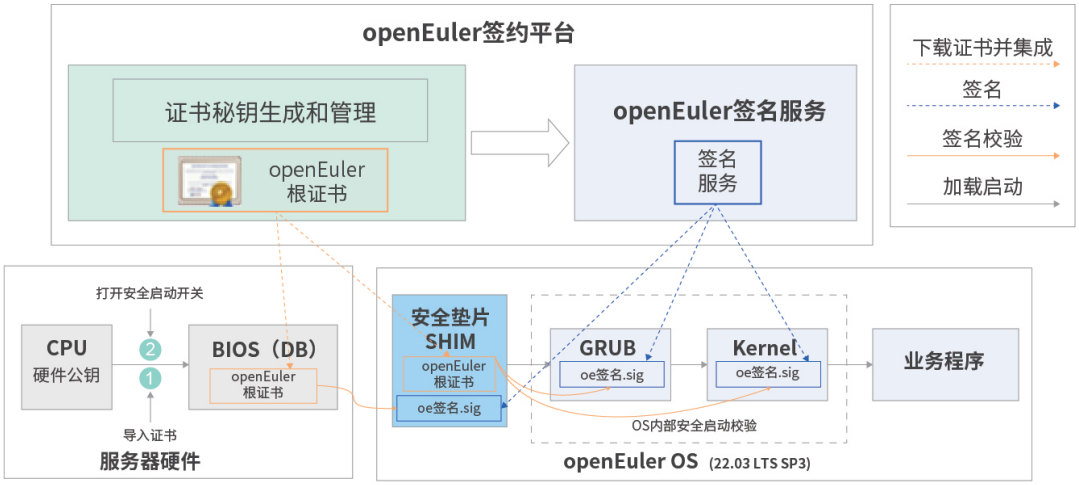
安全启动

安全启动（Secure Boot）是利用公私钥对启动部件进行签名和验证。在启动过程中，前一个部件验证后一个部件的数字签名，如果能验证通过，则运行后一个部件；如果验证不通过，则停止启动。通过安全启动可以保证系统启动过程中各个部件的完整性，防止没有经过认证的部件被加载运行，从而防止对系统及用户数据产生安全威胁。

安全启动涉及的验证组件： BIOS->shim->grub->vmlinuz（依次验签通过并加载），其中 vmlinuz 是内核镜像。

功能描述

由 openEuler 签名平台提供签名服务，突破开源社区签名私钥管理困难问题，解决了社区安全启动组件长期无签名的问题，实现社区秘钥的统一管理和安全启动功能，达到了增强系统完整性保护和秘钥规范性管理效果。



1. openEuler 签名平台生成签名公私钥和证书，并进行秘钥证书管理。同时签名平台对 openEuler 内部的 EulerMaker 软件包构建提供签名服务。
2. 在 EBS 软件包构建过程中进行签名，使用 openEuler 签名平台对安全启动的 EFI 组件（shim/grub/vmlinux）使用 signcode 方式进行签名。
3. 在系统启动过程中进行签名验签，保障系统组件没有被篡改。

约束限制

- 当前社区签名平台只支持对 openEuler 社区内部构建的组件进行签名，暂不支持对外部工程构建的文件及客户文件进行签名。
- 当前签名平台提供的签名算法只支持国际 RSA 算法。

应用场景

当前 22.03 LTS SP3 版本已支持安全启动功能，默认没有开启。用户可根据需要使能安全启动功能，包括打开安全启动开关和导入根证书，具体操作如下。

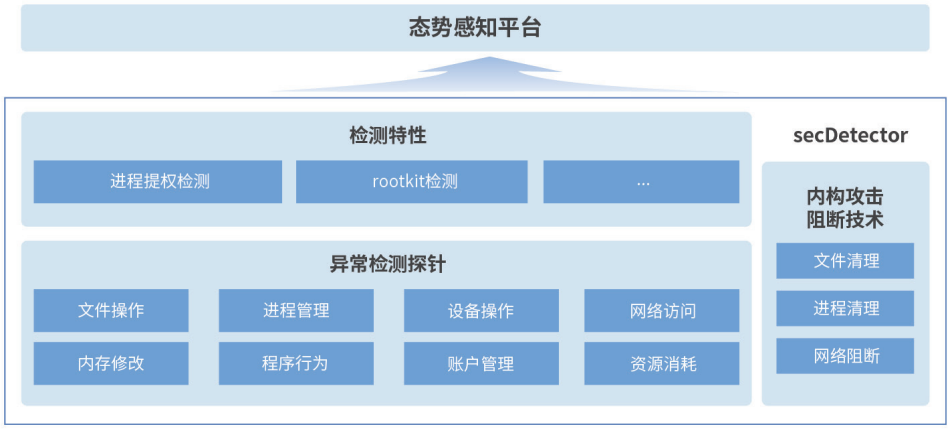
1. 从 openEuler 证书中心，<https://www.openeuler.org/zh/security/security-bulletins/> 获取签名根证书，证书名称“openEuler Shim Default CA”。
2. 设备启动后，进入 BIOS 界面，将 openEuler 根证书导入 BIOS 的 db 证书库中，并在 BIOS 中开启安全启动开关，保存设置并重启。

说明：由于 BIOS 型号不同，BIOS 界面的进入方式及配置选项存在差异，具体 BIOS 界面操作方法可参考对应 BIOS 厂商提供的资料。

OS 入侵检测

secDetector 是专为 OS 设计的内构入侵检测系统，旨在为关键信息基础设施提供入侵检测及响应能力，为第三方安全工具减少开发成本同时增强检测和响应能力。secDetector 基于 ATT&CK 攻击模式库建模提供更为丰富的安全事件原语，并且可以提供实时阻断和灵活调整的响应能力。

功能描述



secDetector 由检测特性集模块、异常检测探针模块、攻击阻断模块组成，异常检测探针功能是采集由攻击引发的 OS 系统事件，基于权威 MITRE ATT&CK 威胁模式库设计的 8 类系统异常检测探针（文件操作、进程管理、网络访问、程序行为、内存篡改、资源消耗、账户管理、设备操作），覆盖识别 APT 的完备信息。攻击阻断技术，对于规则清晰的事件，可根据预设规则提供实时阻断的响应措施：恶意对象清理（进程、文件、网络、模块、脚本）；动作阻止（进程提权、文件修改），在信息采集、检测的过程中就可以对攻击行为进行实时阻断。基于异常检测探针模块和攻击阻断模块的能力，构建一些基于内构入侵检测系统相比外挂式入侵检测系统有优势的攻击检测特性，如进程提权、内核 rootkits 检测等。

检测特性集、异常检测探针、攻击阻断等功能都支持扩展，用户可按应用场景增加或增强检测响应能力，高度可扩展性得益于技术实现架构上的支持。

secDetector 在技术实现架构上分为四个部分：SDK、service、检测特性集合 cases、检测框架 core。

- SDK 是以一个用户态动态链接库 lib 的形态承载，被部署到需要使用 secDetector 入侵检测系统的安全感知业务中。SDK 用于和 secDetector 入侵检测系统的 service 通讯，完成所需的工作（例如订阅，去订阅，读取现有消息等功能）。secDetector 提供的异常信息被定义成不同的 case，安全感知业务可以根据自身需求订阅。
- service 是以一个用户态服务应用的形态承载，向上管理、维护安全感知业务的 case 订阅信息，向下维护 case 的运行情况。框架 core 和检测特性集合 case 采集到的信息由 service 统一收集，按需转发给不同的安全感知业务。安全感知业务对于底层检测特性集合 case 和框架 core 的配置、管理的需求也由 service 进行统一的管理和转发。不同的安全感知业务可能会需求同样的 case，service 会统计出所有安全感知业务需求 case 的并集，向下层注册。
- 检测特性集合 cases 是一系列异常检测探针，根据异常信息的不同会有不同的形态，比如内核异常信息检测的每个探针会以内核模块 ko 的形态承载。一个 case 代表一个探针，一个探针往往是一类异常信息或者一类异常事件的信息。比如进程类探针会关注所有进程的创建、退出、属性修改等事件信息，内存修改类探针会收集内核模块列表和安全开关等信息。因此一个探针可能会包含多个事件的监控，而这些对不同事件的监控逻辑可能无法部署在同一个执行流当中。我们使用工作流（workflow）的概念表示一个探针在同一个执行流中的工作，一个探针可以包含一个或者多个工作流。比如对于某个进程探针而言，进程创建检测和进程属性修改检测就是不同的工作流。一个 workflow 被定义为由四类功能单元组成：事件发生器、信息采集器、事件分析器、响应单元。

- 检测框架 core 是每一个 case 依赖的基础框架，提供 case 的管理和 workflow 所需的通用的基础功能单元。内核异常信息检测框架会以内核模块 ko 的形态承载。一个检测特性 case 可以将自己注册到框架中，或者从框架中去注册。框架还可以提供特定的交互接口以满足外部的动态请求。

应用场景

secDetector 作为一套灵活的 OS 内构入侵检测系统，有三种使用模式：

1. 直接被系统用户开启用作一些基础异常事件的告警和处置。
2. 被安全态势感知服务集成，补齐系统信息采集缺陷，用于 APT 等复杂的安全威胁分析和重点事件布控实时阻断。
3. 由安全从业人员或安全感知服务提供商二次开发，基于可拓展框架构建精确、高效、及时的入侵检测与响应能力。

EulerMaker 特性

EulerMaker 构建系统是一款软件包构建系统，完成源码到二进制软件包的构建，并支持开发者通过搭积木方式，组装 和定制出适合自己需求的场景化 OS。主要提供增量 / 全量构建，分层定制与镜像定制的能力。

功能描述

- **增量 / 全量构建**，基于软件包变化，结合软件包依赖关系，分析影响范围，得到待构建软件包列表，按照依赖顺序下发并行构建任务。
- **构建依赖关系**，提供工程软件包构建依赖表，支持筛选及统计软件包依赖及被依赖的软件包内容。
- **分层定制，**支持在构建工程中，基于 spec 或 yaml，叠加配置层模型，实现针对软件包的版本、patch、构建依赖、安装依赖、编译选项及构建流程等内容的定制。
- **镜像定制**，支持开发者通过配置 repo 源，生成 iso、qcow2、容器等 OS 镜像，并支持对镜像进行软件包列表定制。
- **支持本地任务复现**，通过命令行在本地复现构建任务，方便定位构建问题。
- **一键工程创建**，基于 yaml 配置实现一键工程创建，支持批量加包，大大简化用户操作。

应用场景

社区开发者及合作伙伴基于统一构建系统建设自己的用户个人仓，OS 核心仓，定制出适合自己需求的场景化 OS。

DPU 直连聚合

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

随着智能网卡演进到 DPU/IPU 的形态，这一新兴计算单元正逐渐成为云与数据中心基础设施的重要组成部分。DPU/IPU 除了承担 IO 数据面加速的需求外，也在逐渐增加对管理面及控制面组件卸载的支持。数据中心基础设施相关管控组件全卸载到 DPU 能够带来更优的架构和更灵活的部署方式。当前主流卸载方案大都通过组件拆分完成卸载，拆分方案存在以下问题：

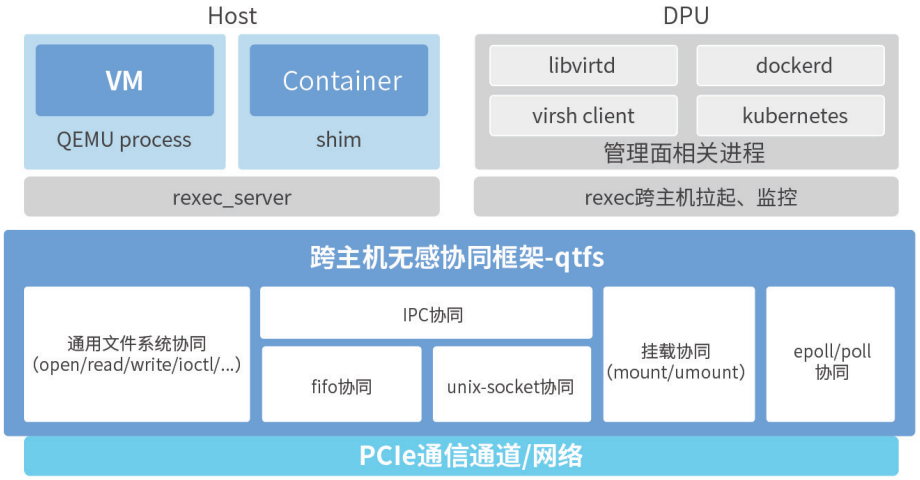
- 组件拆分要求开发者对卸载组件具备代码层级的了解。云厂商维护组件较多，相关组件卸载至 DPU 带来的拆分工作量巨大。
- 拆分工作在组件升级时很难继承，组件升级维护的成本较高，需要移植适配相关拆分代码。

项目介绍

由如下架构图所示， 直连聚合特性在 HOST 和 DPU 的操作系统层面构建了一个跨主机无感协同框架，该框架为卸载到 DPU 侧的管理面进程和 HOST 侧的业务进程提供一致的运行时视图，达到应用对卸载低感知或零感知的效果。用户只需要少量适配管理面业务代码，保证业务的软件兼容性和演进性，降低组件维护成本。

功能描述

DPU 管理面无感卸载框架架构图如下图所示，通过在 HOST 及 DPU 操作系统层面构建一个跨主机无感协同框架，为卸载到 DPU 侧的管理面进程和 HOST 侧的业务进程提供一致的运行时视图，达到应用对卸载低感知或零感知的效果。



从实现层面，本方案提供的机制可以结合定制策略实现不同场景下的进程无感卸载目标，方案包含的协同机制介绍如下：

- **文件系统协同：**支持跨主机文件系统的访问，为 HOST 和 DPU 进程提供一致的文件系统视图；除通用文件系统外还包括对 proc、sys、dev 等特殊文件系统的支持。
- **IPC 协同：**实现跨 HOST 和 DPU 进程间的无感通信，当前支持进程无感使用 fifo 及 unix domain socket 进行跨主机通信。
- **挂载协同：**对特定目录下的挂载操作拉远至 HOST 端，可用于适配容器 overlay 镜像构造场景；支持卸载后的管理面进程为 HOST 侧业务进程构造工作目录，提供跨节点的统一文件系统视图。
- **epoll 协同：**支持远程普通文件及 fifo 类文件跨主机访问的 epoll 操作，支持阻塞读写操作。
- **进程协同：**通过 rexec 工具进行远程可执行文件拉起，能够接管远端拉起进程的输入输出流并进行状态监控，保证两端进程生命周期一致性。

通过以上机制的结合，在不同场景下使用定制策略，可满足管理面进程接近无感、无需过多业务拆分改造的业务要求，并可达成卸载到 DPU 的业务目标。

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 & 二维码

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

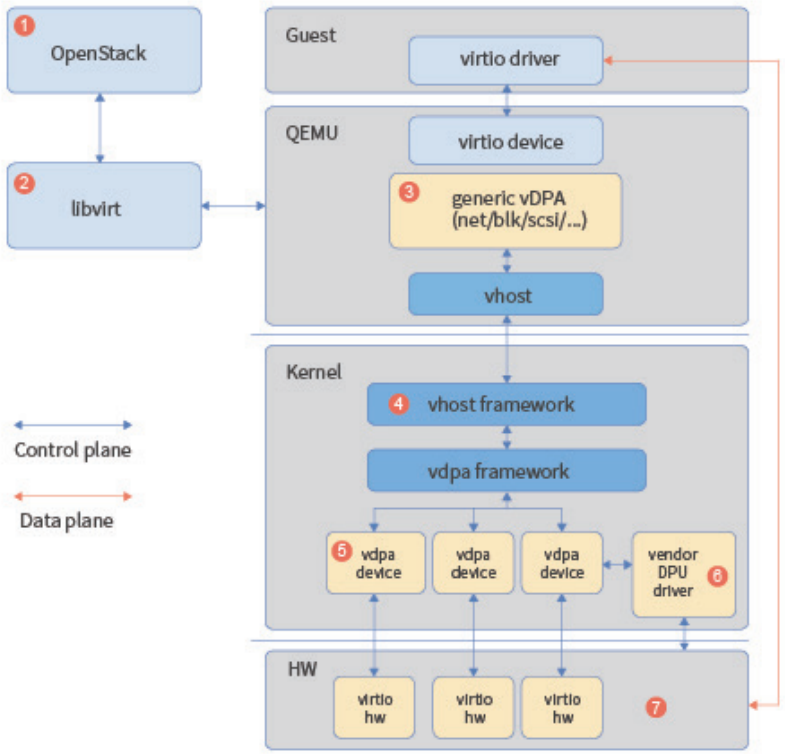
vDPA 网卡直通虚机热迁移

内核态 vDPA 框架，为设备虚拟化提供了一种性能与直通持平，且支持跨硬件厂商热迁移的方案，通过通用 vDPA 框架，实现智能网卡 /DPU 卡等多种硬件形态的架构统一。

通过扩展 vDPA 框架和 vhost 框架接口，实现同厂商 vDPA 设备虚拟机的热迁移能力，满足 vDPA 直通虚拟机的热迁移基本能力。同时预埋跨厂商热迁移的功能代码，满足后续跨厂商设备热迁移能力演进诉求。

功能描述

内核态 vDPA 的整体架构如下。特性主要完成 libvirt/qemu/ 内核组件对于通用 vDPA 框架以及热迁移能力的支持。



序号	组件	功能
1	OpenStack	通过 openstack 纳管支持通用 vDPA 的设备。
2	libvirt	支持 vDPA 设备资源查询，设备生命周期管理。
3	generic vDPA	实现通用 vDPA 设备模型，支持多种 virtio 设备类型。
4	vhost-vdpa	对接 vhost 子系统，打通 vDPA 设备的管理路径。
5	vdpa device	对接 vDPA 框架支持 virtio 设备生命周期、热迁移等能力。
6	vendor DPU driver	厂商提供的 vdpa 驱动，支持对接 vdpa 框架。
7	virtio hardware	满足 virtio 协议规范的硬件。

内核模块中，vDPA 框架的接口已经满足了通用 vDPA 的诉求，为支持 vDPA 设备的热迁移，对开源 vDPA 框架的接口做如下扩展：

设备标脏接口

当前内核态 vDPA 热迁移方案依赖设备支持硬件标脏。

支持 vDPA 设备标脏，主要包含两处改动：支持设备在 virtio feature 协商完成后，设置设备开启 / 关闭标脏的 feature；支持设置标脏的脏页地址、脏页大小，以及同步脏页的接口。

整体流程简单描述如下：

1. 迁出端 libvirt 和 qemu 发起迁移。
2. 迁出端 qemu 通过新增的 IOCTL，设置硬件标脏的脏页地址以及脏页大小。
3. 迁出端 qemu 通过设置 feature，打开设备硬件标脏能力。
4. 迁出端 qemu 通过脏页同步接口，获取脏页内存。
5. 迁出端 qemu 合并设备脏页，并迭代拷贝到迁入端 qemu 中。

设备状态接口

虚拟机热迁移除了迁移虚拟机的内存外，需要迁移虚拟机设备的状态信息，来保证迁出端和迁入端的设备状态一致。新增 vDPA 设备的设备状态保存 / 恢复接口。

整体流程简单描述如下：

1. 迁出端 qemu 脏页迁移完成，虚拟机停机。
2. 迁出端 qemu 调用 vDPA 设备状态保存接口，查询 vDPA 设备状态。
3. 迁出端 qemu 将 vDPA 设备状态发送至迁入端 qemu。
4. 迁入端 qemu 调用 vDPA 设备状态恢复接口，设置 vDPA 设备状态。
5. 迁入端 qemu 恢复虚拟机。

迁移状态接口

考虑到不同硬件厂商可能会针对不同的热迁移状态，做硬件资源回收等操作，新增热迁移状态设置接口，用于 qemu 将热迁移的不同状态告知 vDPA 硬件，用于 vDPA 硬件做定制处理。

应用场景

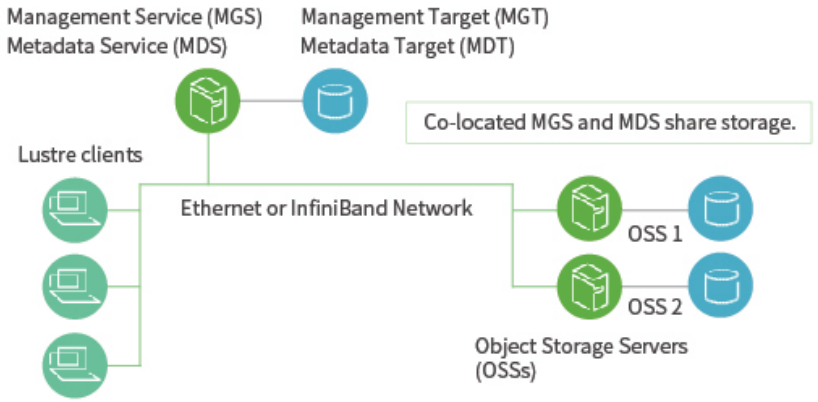
内核态 vDPA 方案适用于硬件支持 virtio 协议的智能网卡 /DPU 卡场景，可以实现与硬件持平的 IO 性能，并支持同厂商 vDPA 设备的虚拟机热迁移。

同时在 DPU 卡场景，可以实现一套框架支持 virtio-net、virtio-scsi 等所有 virtio 设备的支持，减少 openstack/libvirt/qemu 相关设备模型支持的开发工作量，保证方案的兼容性和可演进性。

Lustre server 软件包

Lustre 是一个开源、分布式并行文件系统软件平台，具有高可扩展、高性能、高可用等特点。Lustre 运行于 Linux 系统之上，提供符合 POSIX 标准的 UNIX 文件系统接口。

一个部署好的 Lustre 软件集群系统包含一个管理服务（MGS）及一个或多个 Lustre 文件系统实例，这些组件通过 Lustre 网络（LNet）互联起来。



- MGS （管理服务）：MGS 存储了集群中所有 Lustre 文件系统的配置信息。Lustre 文件系统和 Lustre 客户端通过联系 MGS 来获取这些配置信息。
- MGT （管理目标）：存储配置信息的实体，一个对应 MGT 对应一个硬盘，通常 MGT 和 MDT 共享硬盘部署。
- LNet（Lustre 网络）：LNet 是 Lustre 抽象化的网络 API，提供了通信的基础设施，每个网络协议需要实现一个 LNet driver，例如 TCP，IB 网络协议等。

Lustre 文件系统组件

- 每个 Lustre 文件系统由一下组件组成：
- MDS （元数据服务）：MDS 管理存储在 MDT 中的文件系统元数据（例如文件名，目录，权限和文件布局等），并提供给 Lustre 客户端使用。
- MDT （元数据目标）：MDT 存储云数据的组件，通常每个 MDT 对应一个硬盘。
- OSS （对象存储服务）：用户的文件数据被拆分存储在一个或多个对象中，OSS 是管理这些对象的服务。
- OST（对象存储目标）：OST 存储文件数据对象，每个 OST 对应一个硬盘。
- Lustre Client：Lustre client 为挂载和使用 Lustre 文件系统的客户端，通常为计算节点。

功能描述

高可扩展性和高性能

一个 Lustre 系统可在客户端节点数量、磁盘存储量、带宽各方面进行扩展或裁剪。其可扩展性和性能取决于系统中可用的磁盘、网络带宽、以及服务器的处理能力。组件的扩展和性能特性如下所示。

- 客户端扩展性：支持最大客户端节点数 100000，已知生产环境 5000+ 客户端，许多或在 10000~20000 之间。
- 客户端性能：单客户端支持 I/O 性能为 90% 网络带宽；聚合 I/O 性能为 50TB/s, 50M IOPS。
- OSS 扩展性：单 OSS 支持 1 到 32 个 OST； 单 OST 支持 5 亿对象，1024TB 容量。支持最大 1000 个 OSS，4000 个 OST。
- OSS 性能：单 OSS 支持 15GB/s，1.5M IOPS；聚合支持 50TB/s，50M IOPS。
- MDS 扩展性：单 MDS 支持 1 到 4 个 MDT；单 MDT，Ldiskfs 后端下，支持 40 亿个文件，16TB 容量；单 MDT，ZFS 后端下，支持 640 亿个文件，64TB 容量。
- MDS 性能：1M/s 的创建性能；2M/s 的 stat 性能。
- 文件系统扩展性：单个文件，Ldiskfs 后端下文件最大大小为 32PB；聚合情况下，512PB 容量，1 万亿个文件。

其他功能特性

- **性能增强的 ext4 文件系统：**Lustre 的后端存储文件系统 Ldiskfs 为 ext4 改进版，大大改进了分布式文件性能。Lustre 也可以使用 ZFS 作为后端文件系统，以获得更高的扩展性和数据完整性功能。
- **符号 POSIX 标准：**POSIX 测试可以在 Lustre 客户上完整通过，就在本地 ext4 文件系统通过一样，只有少量例外。Lustre 文件系统支持 mmap() 文件 I/O 操作。
- **高性能异构网络：**Lustre 支持高性能低延迟的 RDMA 网络，如 InfinitiBand(IB)、Intel OmniPath。
- **高可用：**Lustre 支持双活故障切换。Lustre 文件系统可与各种高可用管理程序一起工作，以实现自动故障切换消除单节点故障。
- **安全：**默认情况下，Lustre 只允许特权端口建立网络连接。UNIX 组成员身份会在 MDS 上进行验证。还支持对文件进行加密。
- **访问控制列表（ACL）及扩展属性：**Lustre 遵循 UNIX 文件的安全模型，并使用 POSIX ACL 进行增强。还支持额外的访问控制，如 root squash。
- **互操作性：**Lustre 文件系统可以运行在各类 CPU 架构和大小端混合的集群上。在依次发布的版本间，Lustre 具有互操作性。
- **基于对象的架构：**客户端与盘上文件结构相互隔离，可在不影响客户端的情况下升级存储架构。
- **配额：**支持基于用户和组配额。
- **容量增长：**通过在集群中新增 OST 和 MDT，可以增加 Lustre 文件系统的容量和聚合带宽，而无需中断服务。
- **控制文件布局：**跨 OST 的文件布局，可以以文件，目录或整个文件系统为单位进行配置。这使得我们可以在文件系统内，根据特定应用需求来优化文件 I/O。Lustre 文件系统使用 RAID-0 将数据条带化，并在 OST 间平衡空间用量。
- **网络数据完整性保护：**从客户端发送到 OSS 的所有数据具有校验码，可防止数据在传输过程中遭到破坏。
- **MPI I/O：**Lustre 架构具有专用的 MPI ADIO 层，优化了并行 I/O 以匹配底层文件系统的架构特点。
- **NFS 和 CIFS 导出：**Lustre 文件系统可以通过 NFS（基于 Linux knfsd 或者 Ganesha）或者 CIFS（基于 Samba）重新导出，使其可以共享到非 Linux 客户端上（如，Windows 和 OS X）。
- **灾难恢复工具：**Lustre 文件系统提供在线分布式文件系统检查工具 LFSCK。在发生重大文件系统错误的情况下，该工具可恢存储组件间的一致性。即使存在不一致，Lustre 文件系统仍可运行。
- **性能监视：**Lustre 文件系统提供了多种机制来进行性能监控和调优。

应用场景

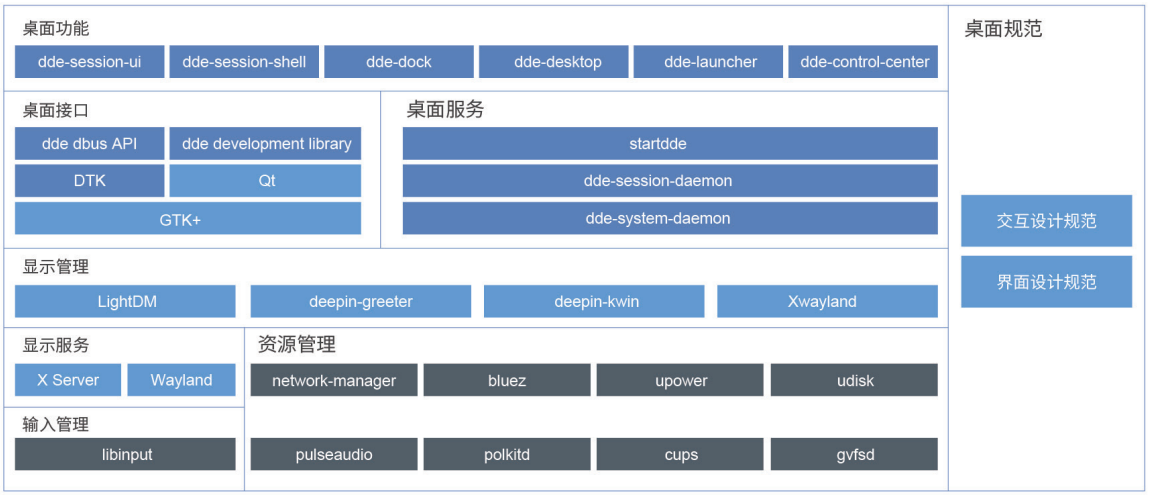
Lustre 并行文件系统适用于需要海量文件存储和高性能文件存储的应用场景。根据社区调查报告 Lustre 广泛应用于研究机构、科学计算、媒体、工业制造、金融、教育等领域行业的 HPC 和 AI 计算场景。

DDE 组件更新支持服务器场景优化

统信桌面环境（DDE）是统信软件为统信操作系统（UniontechOS）开发的一款桌面环境，统信桌面操作系统、统信 操作系统服务器版和统信操作系统专用设备版均在使用统信桌面环境。

功能描述

统信桌面环境专注打磨产品交互、视觉设计，拥有桌面环境的核心技术，主要功能包含：登录锁屏、桌面及文件管理器、 启动器、任务栏（DOCK）、窗口管理器、控制中心等。由于界面美观、交互优雅、安全可靠、尊重隐私，一直是用户首选桌面环境之一，用户可以使用它进行办公与娱乐，在工作中发挥创意和提高效率，和亲朋好友保持联系，轻松浏览网页、 享受影音播放。



统信桌面环境的核心技术是拥有统一界面元素设计、讲究细节交互设计的 DTK 框架及 Qt、GTK+ 等三方图形库。 显示服务、输入管理、资源管理较为底层，一般是基于 golang 开发的后端服务，为上层 GUI 程序提供桌面环境中所 需功能接口，如创建用户、设置屏幕亮度、设置设备音量、管理网络连接等功能。 显示管理、桌面接口、桌面服务属于 shell 层，一般是基于 DBus 接口协议与后端服务进行通信，为定义用户界面、交 互操作提供支撑，如登录界面、窗口外观、GUI 应用程序控件等。

应用场景

桌面功能属于应用层，一般是面向用户可操作的功能界面，比如启动器、任务栏（DOCK）等。

FangTian 视窗引擎特性合入

FangTian 视窗引擎，基于 openEuler 构建桌面环境基础底座，打造 openEuler 视窗根技术。提供显示服务，窗口管理，图形绘制、合成、送显等。

功能描述

- **视窗显示：**提供 Buffer 分配轮转、Vsync、渲染、合成、显示等能力。通过数据驱动接口及统一渲染架构，达成 FangTian 视窗引擎的高性能、低内存的目标。
- **视窗管理** 提供窗口的创建、销毁、移动、缩放、布局等能力。通过独立的窗口策略模块，适应移动端、PC 端等多种设备的多种场景。
- **FT 显示协议：**ArKUI 框架通过 FT 协议与视窗引擎做交互。它提供统一渲染及数据驱动接口，用于降低渲染负载，减少跨进程数据的交互量，提升应用的动画能力和性能。
- **ArkUI 框架：**是一套构建鸿蒙应用界面的声明式 UI 开发框架。通过迁移适配，当前在 openEuler 上也可以使用这套框架。基于 ArkUI 开发的鸿蒙应用能在 openEuler 运行。

主要特性

- 支持 Linux 原生 Wayland 应用及鸿蒙应用可同时运行。
- 支持鸿蒙应用高性能显示：50 窗 @60FPS。

约束限制：

- ArkUI 部分控件功能未使能，仅支持 x86_64 架构。
- Wayland 协议的兼容未考虑扩展协议。

应用场景

FangTian 视窗引擎可融合多个应用生态，为桌面环境提供高性能的底座。

sysMaster 支持虚机场景

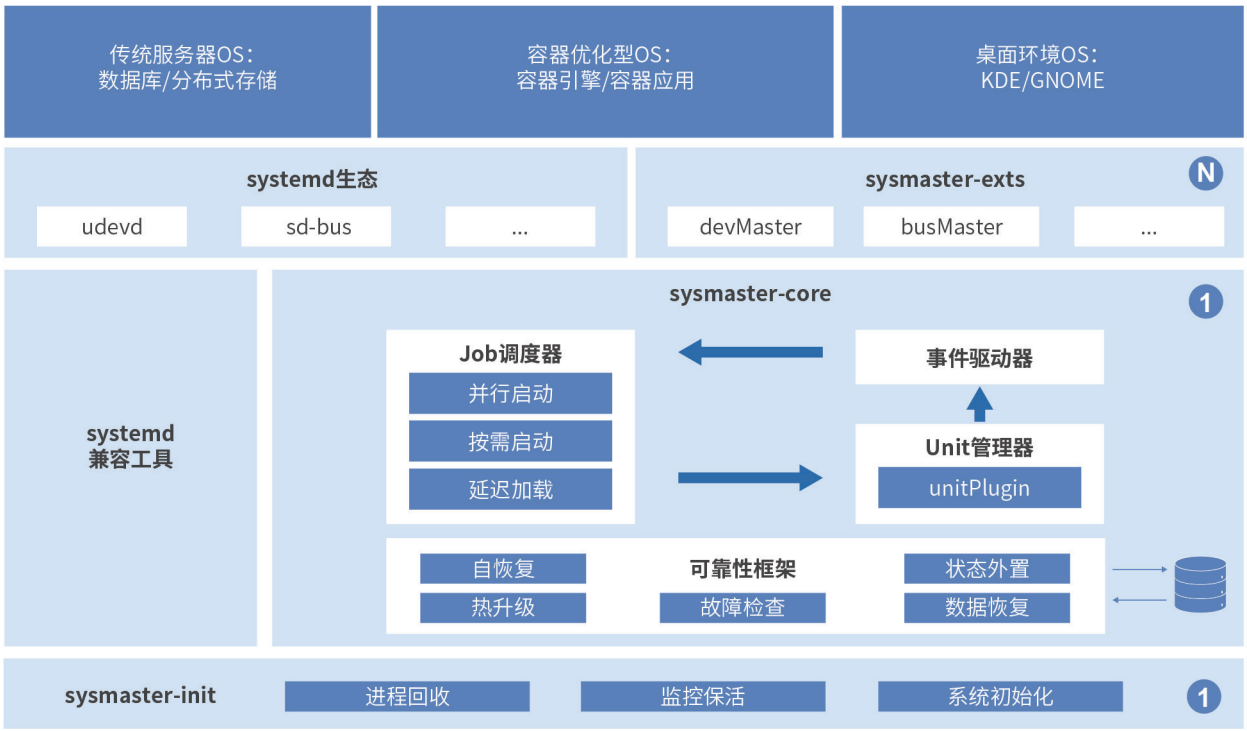
sysMaster 是一套超轻量、高可靠的服务管理程序集合，是对 1 号进程的全新实现，旨在改进传统的 init 守护进程。它使用 Rust 编写，具有故障监测、秒级自愈和快速启动等能力，从而提升操作系统可靠性和业务可用度。

sysMaster 支持进程、容器和虚拟机的统一管理，并引入了故障监测和自愈技术，从而解决 Linux 系统初始化和服务管理问题，致力于替代现有 1 号进程，其适用于服务器、云计算和嵌入式等多个场景。

功能描述

sysMaster 实现思路是将传统 1 号进程的功能解耦分层，结合使用场景，拆分出 1+1+N 的架构。如下面 sysMaster 系统架构图所示，主要包含三个方面。

- **sysmaster-init：**新的 1 号进程，功能极简，代码千行，极致可靠，提供系统初始化 / 僵尸进程回收 / 监控保活等功能，可单独应用于嵌入式场景。
- **sysmaster-core:** 承担原有服务管理的核心功能，引入可靠性框架，使其具备崩溃快速自愈、热升级等能力，保障业务全天在线。
- **sysmaster-exts:** 使原本耦合的各组件功能独立，提供系统关键功能的组件集合（如设备管理 devMaster 等），各组件可单独使用，可根据不同场景灵活选用。



sysMaster 组件架构简单，提升了系统整体架构的扩展性和适应性，从而降低开发和维护成本。其主要特点如下：

- 支持服务管理、设备管理等功能，具有自身故障秒级自愈和版本热升级能力。
- 具备快速启动的能力，更快的启动速度和更低的运行底噪。
- 提供迁移工具，支持从 Systemd 快速无缝迁移到 sysMaster。

- 结合容器引擎 (iSulad) 和 Qemu，提供统一的容器实例和虚拟化实例的管理接口。

  本次发布的 0.5.1 版本，支持在容器、虚机两种场景下，以 sysMaster 的方式管理系统中的服务。

新增特性

- 新增支持 devMaster 组件，用于管理设备热插拔。
- 新增支持 sysMaster 热升级、热重启功能。
- 新增支持在虚机中以 1 号进程运行。

约束限制

- 当前仅支持 64 位系统。
- 当前仅支持 sysMaster 使用的 toml 配置格式。
- 当前仅支持系统容器和虚拟机两种使用场景。

未来，sysMaster 将继续探索在多场景下的应用，并持续优化架构和性能以提高可扩展性和适应性。同时，我们还将开发新的功能和组件以满足容器化、虚拟化、边缘计算等场景的需求。让 sysMaster 成为一个强大的系统管理框架，为用户提供更好的使用体验和更高的效率。

应用场景

sysMaster 致力于替代容器、虚机、服务器及边缘设备上现有 1 号进程。

仓库地址

https://gitee.com/openeuler/sysmaster

migration-tools 项目发布

migration-tools 是由统信软件开发的一款操作系统迁移软件，面向已部署业务应用于其他操作系统且具有国产化替换需求的用户，帮助其快速、平滑、稳定且安全地迁移至 openEuler 系操作系统。

功能描述

迁移软件的系统架构分为：Server 模块、Agent 模块、配置模块、日志模块、迁移评估模块、迁移功能模块。

- **Server 模块：**Server 模块为迁移的软件的核心，采用 pythonflaskweb 框架研发，负责接收任务请求，同时处理相关 执行指令并分发至各 Agent。
- **Agent 模块：**Agent 模块安装在待迁移的操作系统中，负责接收 Server 发出的任务请求，执行迁移等功能。
- **配置模块：**为 Server 模块和 Agent 模块提供配置文件的读取功能。
- **日志模块：**提供迁移的全部运行过程记录日志。
- **迁移评估模块：**提供迁移前的基础环境检测、软件包对比分析、ABI 兼容性检测等评估报告，为用户的迁移工作提供依据。
- **迁移功能模块：**提供一键迁移、迁移进度展示、迁移结果判断等功能。

应用场景

在金融、电信、能源等关键行业，涉及大量存量硬件设备（AMD64 等架构）中操作系统的国产化替代，需要将原存量 操作系统中的应用软件、系统组件迁移至 openEuler 操作系统中时，都可以使用 migration-tools 进行迁移。

utshell 项目发布

utshell 是一个延续了 bash 使用习惯的全新 shell，它能够与用户进行命令行交互，响应用户的操作去执行命令并给予 反馈。并且能执行自动化脚本帮助运维。

功能描述

utshell 功能具体如下：

- 命令执行：可以执行部署在用户机器上的命令，并将执行的返回值反馈给用户。
- 批处理：通过脚本完成自动任务执行。
- 作业控制：能够将用户命令作为后台作业，从而实现多个命令同时执行。并对并行执行的任务进行管理和控制。
- 历史记录：记录用户所输入的命令。
- 别名功能：能够让用户对命令起一个自己喜欢的别名，从而个性化自己的操作功能。

应用场景

utshell 适用于传统服务器系统以及云原生环境，有人值守或无人值守下自动化脚本运维的生产环境。也适用于桌面命 令行爱好者的日常使用。

utsudo 项目发布

Sudo 是 Unix 和 Linux 操作系统中常用的工具之一，它允许用户在需要超级用户权限的情况下执行特定命令。然而， 传统 Sudo 在安全性和可靠性方面存在一些缺陷，为此 utsudo 项目应运而生。 utsudo 是一个采用 Rust 重构 Sudo 的项目，旨在提供一个更加高效、安全、灵活的提权工具，涉及的模块主要有通 用工具、整体框架和功能插件等。

功能描述

- **访问控制：**可以根据需求，限制用户可以执行的命令，并规定所需的验证方式。
- **审计日志：**可以记录和追踪每个用户使用 utsudo 执行的命令和任务。
- **临时提权：**允许普通用户通过输入自己的密码，临时提升为超级用户执行特定的命令或任务。
- **灵活配置：**可以设置参数如命令别名、环境变量、执行参数等，以满足复杂的系统管理需求

应用场景

通过部署 utsudo，管理员能够更好地管理用户权限，确保合适的授权级别，防止未经授权的特权操作，从而降低安全 风险。常见的应用场景有：系统管理维护、用户权限控制、多用户环境等。

i3 相关软件包发布

i3wm 简称 i3，是一个平铺式窗口管理器，通过各种键盘操作，就可以管理当前会话中的窗口布局，并支持多显示器。该版本支持在 openEuler 中使用 i3wm。

功能描述

下面是 i3 中的一些基本操作和对应快捷键。（注：下面的 mod 在通常的 PC 上是 win 键）

- mod+d：调出 dmenu，用于快速启动进程。
- mod+enter：启动 terminal。
- mod+ ↑ /mod+ ↓ /mod+ ← /mod+ →：调整 focus 的窗口。
- mod+shift+q：关闭当前 focus 的窗口。
- mod+shift+r：热加载配置文件。
- mod+shift+e：关闭 i3。
- mod+shift+l：锁屏。

更多的参考可以参考上游文档。

TPCM 特性

可信计算在近 40 年的研究过程中，经历了不断的发展和完善，已经成为信息安全的一个重要分支。中国的可信计算技术近年发展迅猛，在可信计算 2.0 的基础上解决了可信体系与现有体系的融合问题、可信管理问题以及可信开发的简化问题，形成了基于主动免疫体系的可信计算技术 -- 可信计算 3.0。相对于可信计算 2.0 被动调用的外挂式体系结构，可信计算 3.0 提出了以自主密码为基础、控制芯片为支柱、双融主板为平台、可信软件为核心、可信连接为纽带、策略管控成体系、安全可信保应用的全新的可信体系框架，在网络层面解决可信问题。

可信平台控制模块（Trusted Platform Control Module，TPCM）是一种可集成在可信计算平台中，用于建立和保障信任源点的基础核心模块。它作为中国可信计算 3.0 中的创新点之一和主动免疫机制的核心，实现了对整个平台的主动可控。

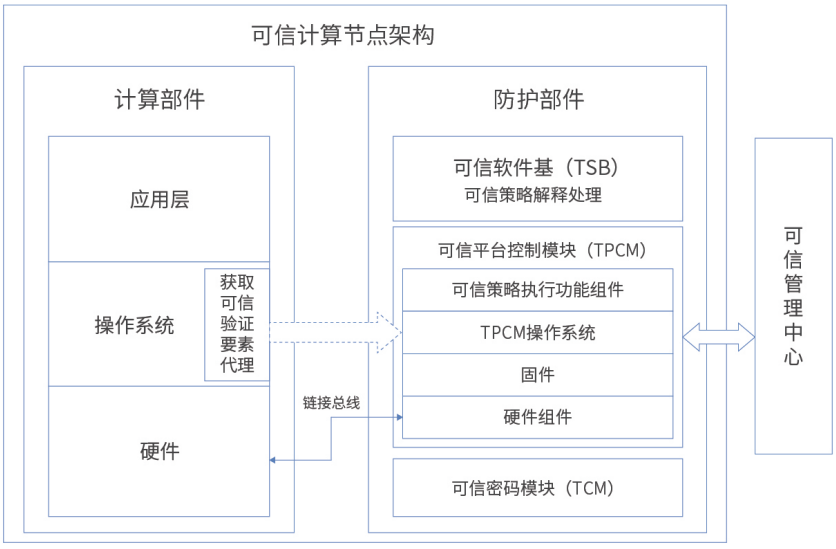
TPCM 可信计算 3.0 架构为双体系架构，分为防护部件和计算部件，以可信密码模块为基础，通过 TPCM 主控固件对防护部件和计算部件及组件的固件进行可信度量，可信软件基 (Trusted Software Base，TSB) 对系统软件及应用软件进行可信度量，同时 TPCM 管理平台实现对可信度量的验证及可信策略同步和管理。

下列缩略语适用于本章节：

- TCM：可信密码模块（Trusted Cryptography Module）
- TPCM：可信平台控制模块（Trusted Platform Control Module）
- TSB：可信软件基（Trusted Software Base）

功能描述

如下图所示，整体系统方案由防护部件、计算部件和可信管理中心软件三部份组成。



- **可信管理中心**：对可信计算节点的防护策略和基准值进行制定、下发、维护、存储等操作的集中管理平台，可信管理中心由第三方厂商提供。
- **防护部件：**独立于计算部件执行，为可信计算平台提供具有主动度量和主动控制特征的可信计算防护功能，实现运算的同时进行安全防护。防护部件包括 TPCM 主控固件、可信软件基，以及 TCM 可信密码模块。TPCM 是可信计算节点中实现可信防护功能的关键部件，可以采用多种技术途径实现，如板卡、芯片、IP 核等，其内部包含中央处理器、存储器等硬件，固件，以及

操作系统与可信功能组件等软件，支撑其作为一个独立于计算部件的防护部件组件，并行于计算部件按内置防护策略工作，对计算部件的硬件、固件及软件等需防护的资源进行可信监控，是可信计算节点中的可信根。TPCM 需与 TSB、TCM、可信管理中心和可信计算节点的计算部件交互，交互方式如下：

a)TPCM 的硬件、固件与软件为 TSB 提供运行环境，设置的可信功能组件为 TSB 按策略库解释要求实现度量、控制、支撑与决策等功能提供支持。

b)TPCM 通过访问 TCM 获取可信密码功能，完成对防护对象可信验证、度量和保密存储等计算任务，并提供 TCM 服务部件以支持对 TCM 的访问。

c)TPCM 通过管理接口连接可信管理中心，实现防护策略管理、可信报告处理等功能。

d)TPCM 通过内置的控制器和 I/O 端口，经由总线与计算部件的控制器交互，实现对计算部件的主动监控。

e) 计算部件操作系统中内置的防护代理获取预设的防护对象有关代码和数据提供给 TPCM，TPCM 将监控信息转发给 TSB，由 TSB 依据策略库进行分析处理。

- 计算部件：主要包括硬件、操作系统和应用层软件。其中操作系统分为引导阶段和运行阶段，在引导阶段 openEuler 的 shim 和 grub2 支持可信度量能力，可实现对 shim、grub2 以及操作系统内核、initramfs 等启动文件的可信度量防护；在运行阶段，openEuler 操作系统支持部署可信验证要素代理（由第三方厂商可信华泰提供），它负责将数据发送给 TPCM 模块，用以实现运行阶段的可信度量防护。

约束限制

- 适配服务器：TaiShan 200（型号 2280）
- 适配 BMC 插卡型号：BC83SMMC

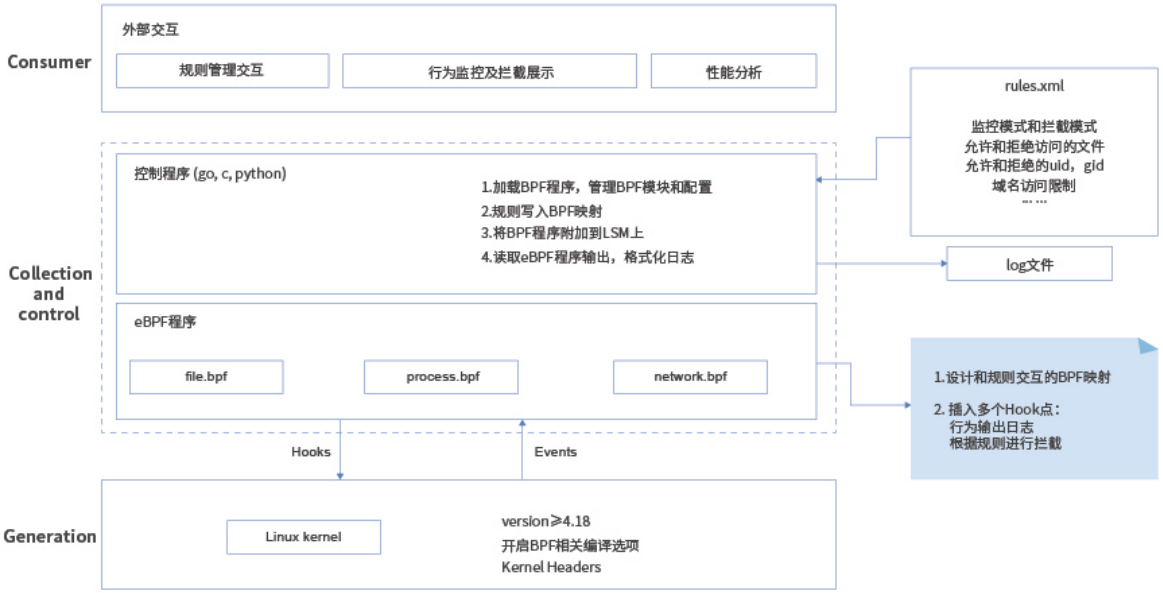
应用场景

通过 TPCM 特性构成一个完整的信任链，保障系统启动以后进入一个可信的计算环境。

支持 safeguard 软件包

针对操作系统、内核安全，safeguard 是一个基于 eBPF 的 Linux 安全防护系统，可以实现安全操作的拦截及审计记录。项目采用 libbpfgo 库，使用 go 语言实现顶层控制。

功能描述



文件

- 追踪文件系统的活动，包括文件的打开、关闭、读写、删除等。
- 修改文件系统的行为，例如拦截某些文件操作，或者实现自定义的安全策略。

安全策略

1. 拦截或重定向某些文件操作，使用 eBPF 来拦截对敏感文件的读写操作，或者重定向对某些文件的访问到其他位置。
2. 实现自定义的访问控制，使用 eBPF 来检查对文件的访问者的身份、权限、环境等信息，然后根据一些规则来允许或拒绝访问。
3. 实现自定义的审计和监控，使用 eBPF 来记录对某些文件的操作的详细信息，如操作者、时间、内容等，并将这些信息输出到日志。

进程

- 追踪进程的生命周期，例如进程的创建、终止等。
- 修改进程的行为，例如注入或修改某些系统调用，或者实现自定义的调度策略。

网络

- 追踪网络的活动，例如网络包的发送、接收、转发、丢弃等。
- 修改网络的行为，例如过滤或重写某些网络包，或者实现自定义的路由策略。

应用场景

safeguard 是一种基于 KRSI(eBPF+LSM) 的 Linux 安全审计和管控解决方案，可以实现对系统的全面监控和保护。下面是一些可能的应用场景：

- **容器安全：**safeguard 可以对容器内部的行为进行审计和控制，例如记录容器的进程、文件、网络活动，限制容器访问特定的资源或端口，检测容器的异常行为等。这样可以有效地防止容器被恶意攻击或滥用，提高容器的安全性和稳定性。
- **云服务安全：**safeguard 可以对云服务提供商的客户机进行审计和控制，例如记录客户机的操作系统、应用程序、用户等信息，限制客户机执行特定的命令或系统调用，检测客户机的恶意行为或漏洞利用等。这样可以有效地保护云服务提供商的资源和信誉，防止客户机被黑客入侵或破坏。
- **安全合规：**safeguard 可以对系统的安全合规性进行审计和控制，例如记录系统的配置、权限、日志等信息，限制系统修改特定的设置或文件，检测系统的违规行为或异常事件等。这样可以有效地满足各种安全标准和法规要求，提高系统的可信度和合法性。

08 著作权说明

09 商标

openEuler 白皮书上使用和显示的所有商标、标志皆属 openEuler 社区所有，但注明属于其他方拥有的商标、标志、商号除外。未经 openEuler 社区或其他方书面许可，openEuler 白皮书所载的任何内容不应被视作以暗示、不反对或其他形式授予使用前述任何商标、标志的许可或权利。未经事先书面许可，任何人不得以任何方式使用 openEuler 社区的名称及 openEuler 社区的商标、标记。

附录 10

附录 1：搭建开发环境

环境准备	地址
下载安装 openEuler	https://openeuler.org/en/download/
开发环境准备	https://gitee.com/openeuler/community/blob/master/en/contributors/prepare-environment.md
构建软件包	https://gitee.com/openeuler/community/blob/master/en/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/security-process-en.md
安全披露信息	https://gitee.com/openeuler/security-committee/blob/master/security-disclosure-en.md