openEuler

openEuler 20.09

技术白皮书

openEuler

目录

CONTENTS

01 概述

概述

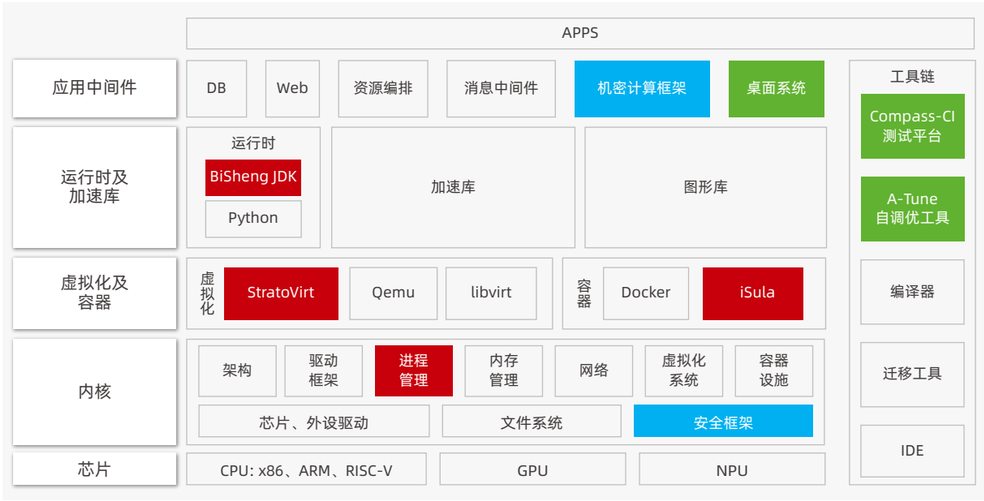
openEuler 是一个开源、免费的 Linux 发行版平台，将通过开放的社区形式与全球的开发者共同构建一个开放、多元和架构包容的软件生态体系。同时，openEuler 也是一个创新的平台，鼓励任何人在该平台上提出新想法、开拓新思路、实践新方案。

02 平台架构

系统框架

openEuler 操作系统面向对象主要是服务器，包括基础加速库、虚拟化、内核、驱动、编译器、OS 工具、OpenJDK 等组件。

创新架构，全栈优化，打造全场景协同的 One OS，为多样性架构释放算力。



释放算力：

Kernel: 分域调度 15%↑

StratoVirt 轻量级虚拟机：开销 80%↓，启动速度 10倍↑

iSula 2.0 云原生容器：空载资源消耗 68% ↓

BiSheng JDK：SpecJbb 20% ↑

安全可信：

IMA 完整性度量架构 ： 防止恶意篡改

secGear 机密计算框架：多平台安全应用开发效率倍级提升↑

繁荣生态：

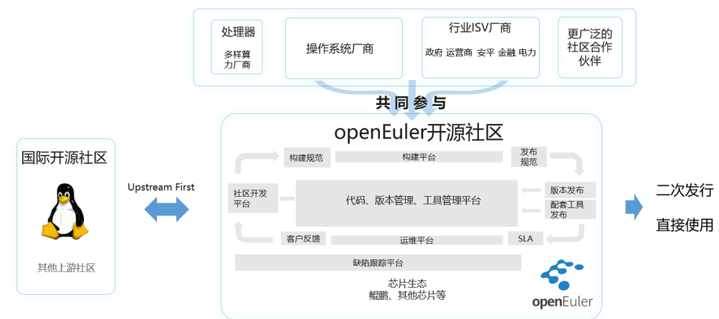
Compass-CI 开源软件自动化测试平台：1000+ 开源软件自动化测试

A-Tune 智能调优工具：10 大类场景，20+ 款应用

UKUI 桌面：轻量级 Linux 桌面环境

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，协作模式开发共同推进版本演进。



03 运行环境

物理机

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

openEuler 安装时，应注意硬件兼容性方面的问题，当前已支持的服务器类型如表 1 所示。

说明：

- TaiShan 200 服务器基于华为鲲鹏 920 处理器。
- 可以支持华为 TaiShan 服务器和 FusionServer Pro 机架服务器，后续将逐步支持更多款型服务器，OSV 和伙伴、客户也可自行适配扩大 openEuler 兼容性。

服务器形态	服务器名称	服务器型号
机架服务器	TaiShan 200	TaiShan 200
机架服务器	FusionServer Pro 机架服务器	FusionServer Pro 2288H V5 说明：服务器要求配置 Avago 3508 RAID 控制卡和启用 LOM-X722 网卡。

表1

最小硬件要求

部件名称	最小硬件要求	说明
架构	AArch64 x86_64	x86_64 支持 ARM 的 64 位架构。 支持 Intel 的 x86 64 位架构。
CPU	华为鲲鹏 920 系列处理器 Intel® Xeon® 处理器	
内存	不小于 4GB（为了获得更好的应用体验，建议不小于 8GB）	
硬盘	为了获得更好的应用体验，建议不小于 120GB	支持 IDE、SATA、SAS 等接口的硬盘。

虚拟机

虚拟化平台兼容性

openEuler 安装时，应注意虚拟化平台兼容性的问题，当前已支持的虚拟化平台为：

- openEuler 自有的虚拟化组件创建的虚拟化平台。
- 华为公有云的 x86 虚拟化平台。

最小虚拟化空间要求

部件名称	最小虚拟化空间要求
架构	AArch64 x86_64
CPU	2 个 CPU
内存	不小于 4GB（为了获得更好的应用体验，建议不小于 8GB）
硬盘	不小于 32GB（为了获得更好的应用体验，建议不小于 120GB）

04 openEuler 20.09 版本新特性

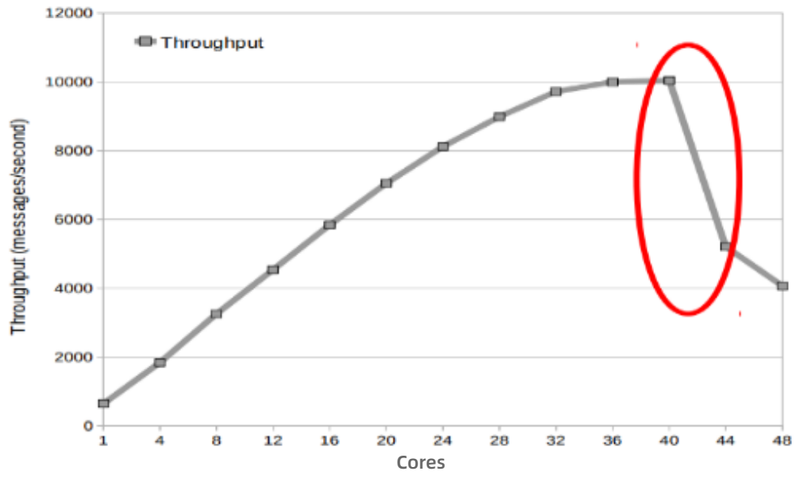
多核加速

概述

多核加速主要针对众核场景，从调度、锁和减少 CPU 共享资源冲突等方面，来提升 CPU 多核的并行度，从而实现业务的加速。

背景

Linux 内核在支持众核 CPU 上，扩展性受到严峻挑战，在 40 核+ CPU 时，部分业务场景会出现性能雪崩。



服务器的 CPU 数和每 CPU 核数都在快速增加，中端服务器会达到 96 或者128 核，高端服务器更会达到 256 核甚至更多。现有 Linux 内核以及用户态应用如 MySQL，未能充分利用多核，扩展性受到制约。

Linux 内核锁等关键机制不感知 NUMA，调度未充分利用众核。

功能描述

NUMA Aware Qspinlock

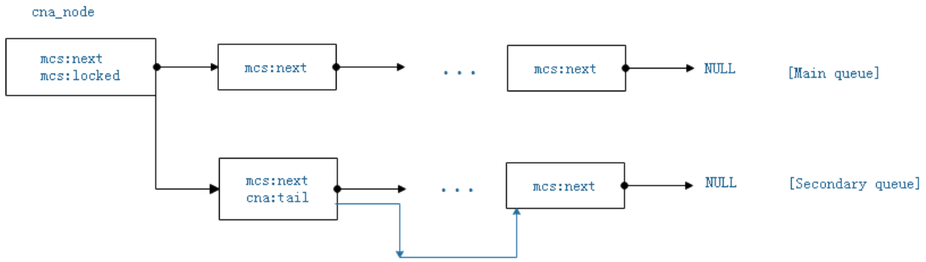
自旋锁可以小到一个比特位：如“零”代表锁可用。获取锁的线程会通过一条“比较并交换”原子指令来尝试置位，如果锁当前不可用，则重复执行这条指令直到成功，自旋指的就是这个重复执行的过程。

但是，这样的自旋锁是无序竞争的，不能保证先申请的进程先获得锁。排队自旋锁，按照要求锁的先后顺序，让最先要求锁的 CPU 在第一时间获得锁，后要求锁的 CPU 排队等候。从而增加了对公平性的支持，进程按照申请锁的顺序排队，先申请的进程优先获得锁。

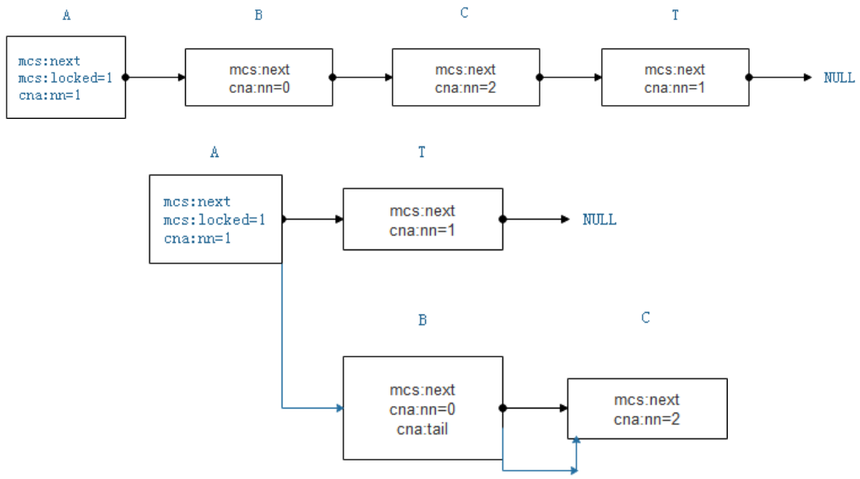
另外，自旋锁的性能也存在一个问题，因为自旋本身就有很大的开销：每一个尝试获取锁的操作都需要将包含锁的缓存行传输到执行操作的 CPU 上。锁竞争严重的情形下，这个缓存行会在自旋的 CPU 之间不断弹跳，导致性能严重下降。

鉴于以上情况，Peter Zijlstra、Waiman Long 等引入了 MCS 版本的自旋锁 Qspinlock。通过为每个处理器创建一个变量副本，每个处理器在申请自旋锁的时候在自己的本地变量上自旋等待，避免缓存同步的开销，并且通过一些技巧将把 MCS 自旋锁放进 4 个字节，避免了结构体变大 。

NUMA Aware Qspinlock 将等待自旋锁的线程组织成 2 个队列管理，主队列用于管理和当前锁的拥有者运行在同一 NUMA 节点的线程，另一个队列用于管理运行在其他节点上的线程，如下图所示：



在解锁的时候，锁的拥有者扫描主队列查找一个与其运行在同一 NUMA 上的线程。如果找到了（称之为线程 T），则把主队列中在锁的拥有者和 T 之间的所有线程移到第二队列的末尾，然后把锁传递给 T。如果没有找到，则把锁传递给第二队列的第一个线程。最终，如果第二队列为空，则把锁传递给主队列的下一个线程。为了避免第二队列中的线程饿死，这些线程会在特定几率随机数目的 NUMA 内锁传递后被移回到主队列。



通过 NUMA Aware Qspinlock，能大量减少跨 NUMA 的 Cache 同步和乒乓，从而提升整体系统的吞吐量。

Ktask

Ktask 是一个内核任务的并行处理通用框架，是将单线程处理的任务，差分成多个任务，交给 Ktask 框架执行，Ktask 将任务并行到多个 CPU 同时执行，提升系统并行度而提升性能。

MPAM 资源管控

MPAM 是 ARM64 架构 Cache QoS 以及内存带宽控制技术，能实现按进程，虚拟机以及 CPU 等粒度的 Cache 分配，内存带宽分配，因此实现关键进程的 Cache/内存带宽得到保障，提升关键业务性能。

鲲鹏 920 是 ARM64 业界第一个实现此功能的 CPU，openEuler 对此有了完整支持。

应用场景

- 应用场景 1

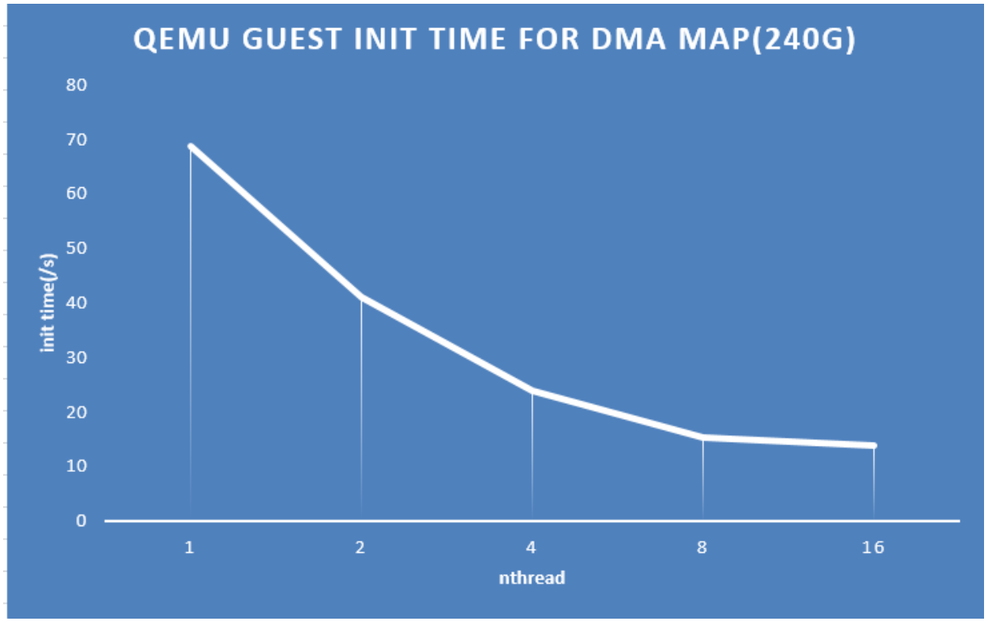
  NUMA Aware Qspinlock 针对 MySQL 等裸机业务，全局争抢同一个锁的场景，提升业务吞吐量。

- 应用场景 2

  Ktask 主要是针对内核的内存清零场景：

  - KVM Guest 中 VFIO 内存页的清零操作
  - 巨页的内存清零等

  在鲲鹏 920 服务器上，测试 240G VFIO-Enabled KVM Guest，得到性能数据如下



可以看出，在并行差分成 2 核处理时，有 1.67 倍的提升，4 核有 2.89 倍的提升。

- 应用场景 3

  MPAM 适合虚拟机以及关键业务的 Cache 以及内存带宽的隔离，针对关键虚拟机或者容器，划分固定的 Cache 和内存带宽，保证关键业务性能。

StratoVirt

概述

StratoVirt 是 openEuler 开源平台上实现的下一代虚拟化技术，Strato 意指大气层中的平流层，寓意为保护 openEuler 平台上业务平稳运行的轻薄保护层。

目前 openEuler 平台已经使用的是 Qemu 虚拟化技术，Qemu 代码量庞大、CVE 安全漏洞频出，业界逐步演进出以 rust 语言实现的 CrosVM、FireCracker 和 Rust-VMM 等架构。安全、轻量、高性能、低损耗的，组件灵活拆分，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。

功能描述

StratoVirt 功能主要支持轻量虚拟机和标准虚拟机两种模式：轻量虚拟机模式下，单虚机内存底噪小于 4MB，启动时间小于 50ms，且支持毫秒级时延的设备极速伸缩能力；标准虚拟机模式下，可支持完整的机器模型，启动标准内核，可以取代 Qemu 的作用，同时在代码规模和安全性上却有较大提升。

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。

StratoVirt 的优势

强安全性：采用 Rust 语言，支持 Seccomp，实现多租隔离

轻量低噪：采用极简设备模型时，启动时间 <50ms，内存底噪 <4M，支持 Serverless 负载

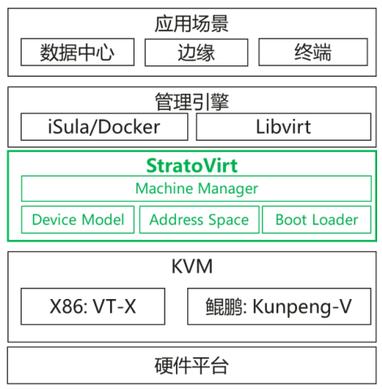
软硬协同：StratoVirt 支持 x86 的 VT，鲲鹏支持 Kunpeng-V，实现多体系硬件加速

极速伸缩：毫秒级设备扩缩能力，为轻量化负载提供灵活的资源伸缩能力

高扩展性：设备模型可扩展，支持 PCI 等复杂设备规范，实现标准虚拟机

异构增强：除支持常用的硬件 SR-IOV 直通方案，结合昇腾软件定义能力，实现更灵活异构算力分配

架构特点



- 面向 serverless 等轻量化场景，StratoVirt 仅实现运行业务负载所需的最小设备集，极大精简传统虚拟化冗余的设备类型，大大减少系统开销；
- 面向传统业务上云场景，StratoVirt 通过动态配置方式实现对标准虚拟机的支持，并将持续优化系统性能；
- StratoVirt 以一套软件架构同时支持多种虚拟化硬件加速技术，并将逐步从 CPU 通用计算向 AI 异构计算演进，推动算力融合；

StratoVirt gitee 地址：https://gitee.com/openeuler/stratovirt

iSula

概述

iSula 是 openEuler 开源平台上的容器技术项目，包括了容器全栈生态中的多个软件。其中通用容器引擎 iSulad 是一种新的容器解决方案，提供统一的架构设计来满足 CT 和 IT 领域的不同需求。相比 Golang 编写的 Docker，iSulad 使用 C/C++ 实现，具有轻、灵、巧、快的特点，不受硬件规格和架构的限制，开销更小，可应用领域更为广泛。

openEuler 20.09 相对 openEuler 20.03 LTS 版本有以下特性更新：

- iSulad 的性能优化，并发启动和容器生命周期操作性能有了很大的提升
- 新增容器镜像构建工具 iSula-build，提供了静态构建、IMA 构建等能力

性能提升

在 openEuler 20.09 版本中，相对于 openEuler 20.03 LTS 版本，iSulad 主要有以下优化：

- 对源码架构进行了重构和调整，提高代码可维护性和可扩展性，对外接口保持不变，用户不感知
- 优化了容器并发启动性能，百容器并发启动平均耗时从 18 秒降低到 2.2 秒（泰山 2288 服务器， iSulad 配置 overlay2 存储）

iSula-build 容器镜像构建

openEuler 20.09 新增了容器镜像构建工具 iSula-build，它提供了安全、快速的容器镜像构建能力。iSula-build 与 iSulad、iSula-transform 等一系列组件一起，构成了 iSula 全栈解决方案，同时也为容器镜像构建提供了全新的选择。

重点特性

iSula-build 有如下特点：

- 完全兼容 Dockerfile 语法

  iSula-build 完全兼容 Dockerfile 所有语法，支持多 Stage 构建，用户可以沿用 docker build 的使用习惯，不需要任何学习成本。

- 与 iSulad、Docker 快速集成

  iSula-build 的镜像导出形式多样，可以直接导入到 iSulad 和 Docker 的本地 Storage。同时，还支持导出到远端仓库和本地 tar 包，与周边组件的集成快速、方便。

- 镜像管理

  iSula-build 提供了本地镜像管理功能。除了 build 镜像之外，iSula-build 还提供了 import/save/load/tag /rm 等镜像管理功能，这使得其镜像构建的来源更加丰富，导出形式更加多样。

- 快速

  相比 docker build，iSula-build 不会为每一条 Dockerfile 指令启动一个容器，只有 RUN 指令才会在容器中执行，而且 Commit 的粒度是 Stage 而不是每一行指令。所以在通常的容器镜像构建场景，构建速度会有大幅提高。

- 安全

  支持 IMA(Integrity Measurement Architecture，完整性度量架构)。这是内核中的一个子系统，能够基于自

定义策略对通过 execve()、mmap() 和 open() 系统调用访问的文件进行度量。通过 iSula-build 构建的镜像能够保留 IMA 文件扩展属性，配合操作系统一起保证构建出来的容器镜像在运行侧可执行文件和动态库的完整性度量。

- 静态构建

静态构建，是指当构建镜像的输入，包括 Dockerfile 和命令行一致，并且指定构建时间戳，则在同一环境多次构建结果得到的容器镜像 ID 相同，在某些需要记录 Dockerfile 对应容器镜像 ID，或者需要固定镜像 ID 的场景下能发挥作用。

应用场景

iSula-build 目前的应用场景很明确，可以在通用场景无缝替换 docker build 构建容器镜像，同时提供了上述涉及的新特性。

Compass-CI

概念

Compass-CI 是一个可持续集成的软件平台。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、基于历史数据的分析服务。通过 Compass-CI，社区开发者将开源软件快速引入 openEuler 社区，补充更多的测试用例，共同构建一个健康完善的开源软件生态。

背景

开源社区的软件质量保障一直是一个难题，不同的开源软件质量差别较大，同时当前社区测试系统一般以测试为主，较少考虑社区开发者与测试系统的协同能力。

开源软件大多数基于个人 PC 开发及调测，缺乏强大易用的多元化测试集群环境。一般测试系统主要关注发现问题，缺乏为开发者提供软件调测、调优、定位、复现能力。

openEuler 开放 Compass-CI 测试平台，为开源软件提供基于鲲鹏集群测试服务，一键式登录调测、自动 git bisect、测试结果分析，大大提升社区开发者的开发调测体验。

功能描述

- **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。

- **调测环境登录**

Compass-CI 提供 SSH 登录能力，测试过程中如果遇到有问题，开发者可根据需要环境进行登录调测 。

- **测试结果比较**

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。

- **bug 辅助定界**

Compass-CI 测试过程中自动识别错误信息，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

- 应用场景1

聚合开发者测试用例：开发者往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用例自动化测试，并反馈测试结果。

- 应用场景2

测试过程中，全面监控系统运行信息（CPU/MEM/IO/网络等），对测试过程数据快照归档，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。

- 应用场景3

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。

- 应用场景4

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

Compass-CI 优点

Compass-CI 集开发调测、测试服务、测试结果分析、辅助定位为一体的综合平台，打造社区开发者极致开发体验。相比业绩其它持续集成软件相比，Compass-CI 平台具有如下特点：软件测试更简单、bug 调测更便捷、测试分析数据更全面。

下一步计划：

1、优化 git bisect 精准率及效率

2、优化数据分析联动能力，让数据分析更加聚焦

3、数据可视化优化，更加友好展示数据比较结果。

4、增强登录认证机制，如：GPG 认证

5、优化部署效率

名词解释

git bisect

它的原理很简单，就是将代码提交的历史，按照两分法不断缩小定位。所谓"两分法"，就是将代码历史一分为二，确定问题出在前半部分，还是后半部分，不断执行这个过程，直到范围缩小到某一次代码提交。

Compass-CI gitee 地址：https://gitee.com/openeuler/compass-ci

A-Tune

概述

A-Tune 是一款基于 openEuler 开发的，自动化、智能化性能调优引擎。它利用人工智能技术，对运行在操作系统上的业务建立精准模型，动态感知业务特征并推理出具体应用，根据业务负载情况动态调节给出最佳的参数配置组合，从而使业务运行于最佳系统性能状态下。

背景

系统调优是一个门槛高的系统性工程，强依赖工程师的技能和经验。一个简单的应用，除了自身代码外，支撑其运行的环境，如硬件平台、操作系统、数据库等都可能影响到应用的性能。如何在众多因素中找到性能瓶颈，需要工程师熟悉大量参数的含义、配置方法，以及业务场景，并不断积累经验，才能对系统进行快速精准调优。

随着业务复杂度和调优对象的增加，调优组合和调优所需成本指数级增长，大大超过了工程师的能力范围。为了降低调优门槛、提升调优效率，A-Tune 应运而生。

功能描述

A-Tune 利用 AI 技术，通过对各种类型的业务进行数据采集，建立精准业务模型，并制定相应的调优策略。A-Tune 具有如下功能：

1、在线静态调优

A-Tune 简化了系统调优，尽可能地屏蔽了硬件和操作系统底层细节，使用者无需感知底层细节，就可以实现快速调优。它利用 AI 技术，通过采集 52 个数据维度进行数据分析和机器学习，识别到具体的应用，快速匹配出多种配置组合，并从积累的优化模型库中找到最佳配置进行设置，满足多业务多场景下的性能调优。

2、离线动态调优

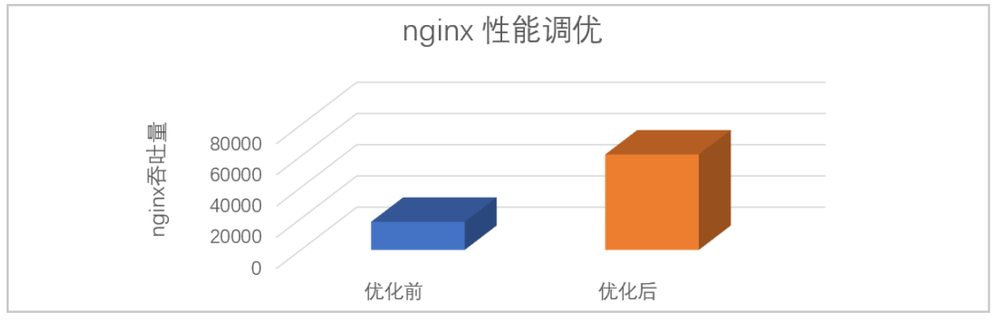
A-Tune 在离线场景下，采用重要参数搜索算法筛选出影响该业务场景下的重要参数，通过贝叶斯优化算法对筛选出的重要参数空间进行迭代搜索，不断优化参数配置，直到算法最终收敛，获取到最优配置。对于配置项多，业务复杂的场景能够极大提升调优效率。

应用场景

应用场景 1：微服务 nginx 调优

调优系统参数			
CPU	Memory	Network	Disk
128core	512G	10G/25G	HDD/SSD

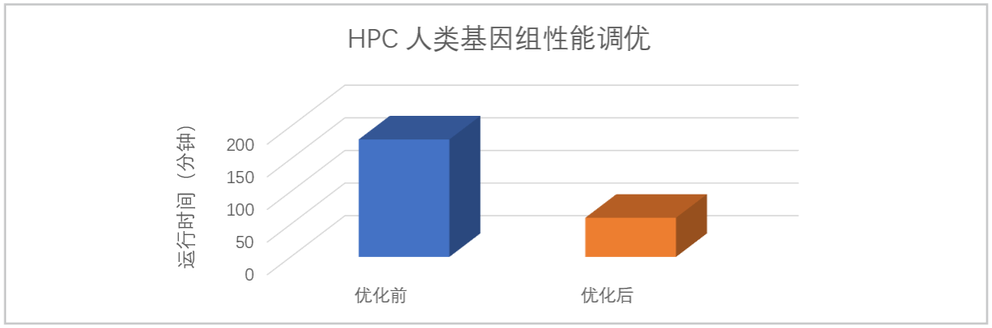
对 nginx 业务场景进行离线动态调优，A-Tune 能够自动感知需要进行硬件加速的场景，硬件加速自动触发后性能从 17998rps 优化到 61144rps，性能提升 240%。



应用场景 2：HPC 场景调优

调优系统（3 节点集群）参数			
CPU	Memory	Network	Disk
128core	1T	10G/25G	nvme

对 HPC 人类基因组业务场景进行离线动态调优，基因测序中合并场景的运行时间从 180 分钟优化到 60 分钟，优化效果提升 200%。



A-Tune 的特点

- 新增多种调优算法支持：

一种调优算法并不能适用所有的调优场景，因此新增多种调优算法：GP(Gaussian Process)，RF(Random Forest)，ET(Extremely Randomized Trees)，GBRT(Gradient BoostRegression Tree)，LHS(Latin Hyper-cube Sampling)，ABTEST，可供使用者适配多种调优场景

- 支持增量调优：

对于单次调优时间较长的系统，意外中断导致一次次重头开始调优是一样非常费时的事情，A-Tune 提供 Restart 的方式实现增量调优能力

- 新一代负载分类模型：

采用双层分类模型和特征工程方法自动选择重要维度并利用随机森林算法进行分类，识别粒度从业务大类识别增强到具体应用识别

- 敏感参数识别与自动筛选：

系统可调参数成百上千，过多的调优参数组合导致传统调优算法难以收敛，新增重要参数选择算法（基于模型精度的加权集成式重要参数选择算法），利用多轮增量式参数筛选裁剪参数空间加快调优算法收敛速度

- 支持多种环境部署：

支持虚拟机和物理机，支持 x86 和 ARM64 架构

- 支持引擎独立部署：

AI 计算会耗费大量的算力，与调优系统一起部署会占用系统本身的资源，新增 AI 引擎独立部署能力

- 支持自动化模型训练：

增加一键式自动化模型训练功能，满足用户自定义场景下应用负载模型训练自动化

IMA

概述

内核完整性度量架构，全称 Integrity Measurement Architecture，是内核中提供的一个强制访问控制（MAC）子系统，自 2.6 版本引入。

IMA 能够基于用户自定义的策略对通过特定系统调用（例如 execve、mmap）访问的文件进行度量，度量结果可被用于两个目的：

度量（measure）：检测对系统的意外或恶意修改，支持本地或远程证明。

评估（appraise）：度量文件并与预先存储的参考值比较，以保护本地文件完整性。

IMA 作为可信计算在 openEuler 中的实现之一，连接了信任链中的可信操作系统和可信应用。

需要注意的是，IMA 和常见的软件签名技术的区别在于，对软件签名只能保证软件在安装时是完整的，无法保证安装后软件不被篡改，而 IMA 在每次程序被访问之前都对其进行完整性校验，保证了程序直到运行前一刻都是完整且来源可信的。

背景

可信，根据可信计算工作组 TCG 的定义，是指对于一个特定的目标，实体的行为总是与预期的相符，则针对这个目标，该实体就是可信的。

一个可信计算系统由信任根、可信硬件平台、可信操作系统和可信应用组成，它的基本思想是首先创建一个安全信任根（TCB），然后建立从硬件平台、操作系统到应用的信任链，在这条信任链上从根开始，前一级认证后一级，实现信任的逐级扩展，从而实现一个安全可信的计算环境。

传统安全机制类似“头痛医头，脚痛医脚”，发现一个漏洞修复一个漏洞，无论防火墙、入侵检测还是杀毒软件，都属于黑名单机制，存在一定的滞后性，因为始终存在未被发现的漏洞和病毒，无法彻底杜绝安全风险。

可信计算采用的是白名单机制，即**只允许经过认证的程序在系统上运行**。在运行一个程序前，除了经过传统的操作系统权限判断，系统还要将其与一个可信的参考度量值对比，如果发现程序已经发生了更改（或原本就是一个未知的程序），就拒绝对该程序的访问。

应用场景

构造可信的本地环境

在运行关键应用的现网环境下，本地文件完整性是系统安全的重要前提，如果程序被篡改，或者有未知程序被执行，都可能导致系统行为偏离预期，从而为现网带来安全风险。

IMA 在开启 enforce 模式后，只允许内容及扩展属性和参考值相比完全一致的文件被访问。对于社区原生 IMA 而言，参考值是在现网环境的 fix 模式下生成的，对于 openEuler 提供的 IMA 摘要列表扩展而言，参考值被提前到构建阶段生成。但无论哪种方式，都采用的是“在可信环境下生成参考值并签名，在现网环境下验签并校验”的白名单机制，从而保证现网环境下执行的每个程序都是完整的。

鉴于 IMA 是一种十分严格的校验机制，我们不推荐在开发环境中开启 IMA 评估 enforce 模式，这将使得构建出的程序无法直接运行，除非手动为其生成参考值并转移到可信的环境下签名（这一过程十分繁琐）。IMA 评估最适合的场景是仅运行官方发布的可信应用的现网环境。

远程证明：验证对端完整性

仅仅保证本地环境的完整性还不够，如果与本机进行通信的对端有文件被篡改（例如生成通信密钥的程序），就可能导致错误通过合法的通信接口不断向外扩散。如果在与对端通信之前，能够通过一种技术证明对端环境也是完整可信的，再建立与对端的会话，就能最大限度避免不可信要素在网络中的传播。

远程证明就是这样一种技术，它能在测试者和被测试者隔开的情况下，通过一些证明手段，使测试者可以确定被测试者是可信的。和其他可信计算技术一样，远程证明依赖于系统的 TPM 芯片，TPM 的 PCR 寄存器扩展具有单向性，这一点保证了对系统的度量值无法被篡改，从而可以对外提供证明。

IMA 在应用层提供了对远程证明的支持，使得可信计算的信任链能够扩展到应用层，如果测试者能够获取到真实的度量日志，即可验证被测试平台加载的文件是否可信。

IMA的优势

纵向对比

IMA 是 openEuler 在 20.09 版本引入到内核的新特性，20.03 LTS 及之前版本均不提供对 IMA 以及可信计算的支持。IMA 的引入从无到有地填补了 openEuler 在可信计算领域的空缺，提供了在应用层保护文件完整性的能力。

20.09 版本不仅使能了 4.19 版本内核原生的 IMA 特性，同时也提供了一种更安全、更便捷的新方案——IMA 摘要列表扩展供用户选择，新方案在保证安全性的同时，显著提升了度量的效率，方便了用户的使用。

横向对比

社区原生 IMA

相比内核社区原生 IMA 机制，openEuler 内核提供的 IMA 摘要列表扩展从安全性、性能、易用性三个方面进行了改良，助力完整性保护机制在生产环境下落地：

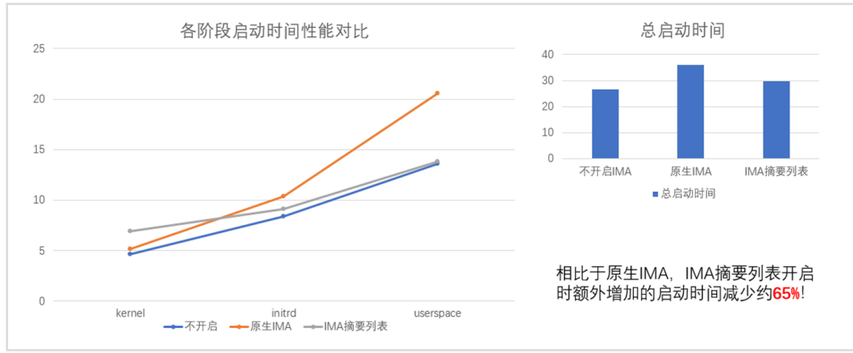
- 具备完整的信任链，安全性好

原生 IMA 机制要求在现网环境下预先生成并标记文件扩展属性，访问文件时将文件扩展属性作为参考值，信任链不完整。

IMA 摘要列表扩展将文件参考摘要值保存在内核空间中，构建阶段通过摘要列表的形式携带在发布的 rpm 包中，安装 rpm 包的同时导入摘要列表并执行验签，确保了参考值来自于软件发行商，实现了完整的信任链。

- 惊艳的性能

由于 TPM 芯片是一种低速芯片，因此 PCR 扩展操作成为了 IMA 度量场景的性能瓶颈。摘要列表扩展在确保安全性的前提下，减少了不必要的 PCR 扩展操作，相比原生 IMA 启动时间减少约 65%。



IMA 评估场景下，摘要列表扩展将签名验证统一移动到启动阶段进行，避免每次访问文件时都执行验签，相比原生 IMA 评估场景提升运行阶段文件访问的性能约 20%。

- 快速部署，平滑升级

原生 IMA 机制在初次部署或每次更新软件包时，都需要切换到 fix 模式手动标记文件扩展属性后再重启进入 enforce 模式，才能正常访问安装的程序。

摘要列表扩展可实现安装完成后开箱即用，且允许直接在 enforce 模式下安装或升级 rpm 包，无需重启和手动标记即可使用，实现了用户感知最小化，适合现网环境下的快速部署和平滑升级。

需要注意的是，IMA 摘要列表扩展将原生 IMA 的验签过程提前到启动阶段进行，也引入了一个假设，即内核空间的内存无法被篡改，这就使得 IMA 也依赖于其他安全机制（内核模块安全启动和内存动态度量）以保护内核内存的完整性。

但无论社区原生 IMA 机制还是 IMA 摘要列表扩展，都只是可信计算信任链中的一环，无法孤立地保证系统的安全性，安全自始至终都是一个构建纵深防御的系统工程。

其他 LSM 模块

SELinux/Apparmor/Smack/Tomoyo 和 IMA 一样都是 Linux 内核提供的强制访问控制（MAC）机制，从实现角度看，它们有很多共通之处，比如它们都利用了内核的 LSM 钩子，在文件被访问或进程尝试某种行为时触发校验，以便控制进程的行为和对文件的访问，校验的规则都可由用户自行定义，不同之处只是它们关注和保护的维度不一样。

但总的来说，IMA 之外的主流 LSM 模块都在试图对系统中进程的行为做不同粒度的控制，例如 SELinux 能够通过配置精确的规则来控制程序对不同文件的访问，而 IMA 控制的着眼点只在于文件本身而非文件的行为，如果文件是不完整的，那么对文件的访问就会天然地存在风险，应被拒绝或记录在度量日志中，就这个维度而言，IMA 的控制点更小，更精确地集中在文件完整性维度。除此之外，IMA 将可信基建立在 TPM 芯片之上，TPM 记录的单向性使得度量结果无法被篡改，从而能够支持远程证明这样的特殊场景。

名词解释

TPM

TPM（Trusted Platform Module）就是可信平台模块，它的主要作用是利用硬件保护的加密密钥带来强大的设备安全性。

TPM芯片通常作为可信计算系统的信任根（Trusted Computing Base）向上构建信任链，从而提供对整个系统的保护。

secGear

概述

在云上或数据中心中，为了保护敏感数据的安全性，通过一个安全隔离区来保护使用中的数据称为机密计算技术。硬件会提供一个可信的隔离执行环境供软件使用，让处理敏感数据的软件运行在可信执行环境中，实际上就是一个 Enclave 执行环境，而在普通环境里是无法访问到 Enclave 环境里的内存及相关代码内容。

Intel x86 处理器提供 SGX（Software Guard Extension）特性，划分一部分主存给 Enclave，对 Enclave 内存进行加密，CPU 加载内存的时候进行解密，外部攻击者无法访问到明文数据。

ARM 处理器提供 TrustZone 特性，在内存控制器上实现访问控制，划分给安全区的内存和设备普通模式下 CPU 无法访问，外部攻击者除非攻破安全区软件，否则无法访问到安全区里的内容的。

背景

secGear 是基于硬件 Enclave 技术为开发者的一个应用开发框架，开发者基于 secGear 框架做应用开发可以简化编写安全应用的复杂度，提升开发效率。

功能描述

- 提供 API 支持应用对 Enclave 的生命周期的管理和对 Enclave 安全函数的调用
- 为安全侧应用开发提供类 Posix 接口，实现非安全侧一致的编程体验
- 提供 Enclave 状态数据加密和持久化的支持，以及 Enclave 本地和远程证明的支持
- 提供必要的代码辅助生成工具及 Enclave 二进制签名工具
- 提供 C 编程语言支持
- 支持 Intel SGX 和 ARM Trustzone

应用场景

- 应用场景 1：多方计算

多方计算：多个组织或实体想利用多方所拥有的数据做运算而互相不泄露各自的数据给其他方。比如有三个公司，每个公司有各自的数据，而且每个公司的数据只能自己公司可以访问，有一种情景，三家公司的数据进行合并通过 AI 训练进行运算，如何避免数据的泄露呢？

这三家公司分别通过 Enclave 技术，将数据通过加密的通道传输给对方的 Enclave，通过远程证明的方式保证 Enclave 执行环境及运行的代码可信，这种情况就做到了各家公司共享数据做计算而不泄露各自的数据。

secGear 可以提供开发框架供开发者开发基于 Enclave 技术的多方计算安全应用。

- 应用场景 2：密钥管理服务

目前公有云的密钥管理服务，一般会基于硬件 HSM 来实现秘钥的安全管理。这些服务可以基于 secGear 开发框架来开发基于 Enclave 技术的安全模块替代硬件加密模块，实现对密钥的保护。

- 应用场景 3：安全数据库

安全数据库，在某些应用场景下，为了保护数据库里的内容，数据库所有者只希望应用能通过 SQL 访问返回的结果，对数据库访问的过程需要严格保护，这种情况下，可以基于 secGear 框架开发安全数据库应用，保障数据库的处理过程在可信的执行环境中运行，保证数据库的数据在处理过程中的安全。

secGear的特征优势

支持多架构：支持 x86、ARM 等多体系架构，不同的体系架构下硬件 Enclave 的实现和编程接口是不一样的，通过 secGear 开发框架可以让用户做到代码归一。一套代码通过在不同体系结构的硬件平台上编译，可以在不同的体系架构硬件上运行。

secGear gitee 地址：https://gitee.com/openeuler/secgear

05 社区治理

社区愿景

openEuler 社区的愿景是：通过社区合作，打造创新平台，构建支持多处理器架构、统一开放的操作系统社区，推动软硬件生态繁荣。

社区沟通和交流

openEuler 包含许多项目，这些项目被组织成社区团体。这些团体的沟通和交流渠道，包括邮件列表、IRC 频道等，可以在这些团队的 README 上找到。

IRC 聊天室

Internet Relay Chat (IRC) 是一个网络实时聊天工具，主要用于组织成员通过频道的形式进行交流，但同时也支持一对一私有信息交流。在 IRC 上您可以和其他任何在线的人交流。

虽然一个频道会有很多人，但这些人并不是随时都在系统前，因此如果您的信息没有人效应，请稍等一会以获得响应。

openEuler IRC 频道：https://openeuler.org/zh/community/irc.html

邮件

如果要开始一个开放主题的讨论，将电子邮件发送到相关的邮件列表是另一个不错的选择：

- 加入邮件列表

  访问该地址 https://openeuler.org/zh/community/mails.html 找到社区可用的邮件列表，请根据您的兴趣参照下面步骤去加入某个邮件列表。

  这里有两种加入方式.:

  网页

  - 点击https://openeuler.org/zh/community/mails.html 中的列表名字进入订阅页面。
  - 输入订阅邮箱并点击“订阅”按钮。
  - 登录邮箱并回复从 openeuler.org 发来的确认邮件。

  如果接收到从 @openeuler.org 发来的类似“Welcome”的邮件就意味着您已经订阅成功了。 注意：如果没有收到“Welcome”邮件，请确保回复邮件时标题保持与原邮件一致再发一次。

  邮箱

  - 发送一封以“subscribe”为标题的邮件到每个邮件列表的提供的订阅地址（订阅地址是列表地址+“-join”后缀组成）。
  - 回复从 openeuler.org 发送的确认邮件。

  以订阅 Dev(dev@openeuler.org) 作为例子，邮件如下:

  receivers: dev-join@openeuler.org

  subject: subscribe

  content: NA

- 发送邮件

  当前有 Announce 和 Discussion 两种列表类型，对于 Discussion 列表的发送方式，跟发送一般的邮箱到私有地址没有区别，按照平常的方式发送即可，用一对中括号加一个主题作为前缀添加到邮件主题里会是一种很好的实践，但是不是必须的，Announce 类型的邮件列表只是用于宣布消息或者注意事项，不接受邮件发布。

  注意：如果你不能在收件箱中接收到任何邮件信息，请优先检查是否将其归并到垃圾邮箱里了。

- 退订邮件

如果您想退订某个邮件列表，请参照如下步骤：

- 发送一封以 unsubscribe 为标题的邮件到该列表的退订地址（退订地址一般是列表地址+"-leave"为后缀组成）。
- 直接回复确认邮件。

当您收到一封退订确认邮件表示您已经退订成功。

- 获取帮助

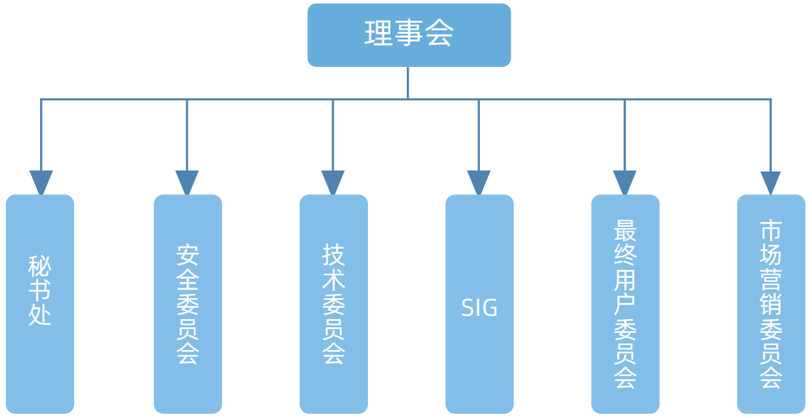
如果过程中遇到问题，请联系基础设施团队：

- 邮件: infra@openeuler.org
- IRC: #openEuler-infra

如果您发现任何有关邮件列表相关的 bug，请直接提交 issue 到 infrastructure

社区治理组织架构

社区治理组织架构包含六部分：



理事会负责决策 openEuler 社区发展方向、路线和策略。决策 openEuler 的预算、市场营销计划和其相应预算。宣传 openEuler 社区、技术项目。审视技术委员会、最终用户委员会、市场营销委员会的工作，给出指导意见。

秘书处

openEuler 社区尚处在完善阶段，秘书处来负责 openEuler 社区的运作、完善工作。现阶段，openEuler 社区秘书处的主要职责如下：

- 制定社区运营规划
- 制定费用预算，获得社区创始人批准
- 执行社区创始人筹备社区的工作
- openEuler 社区其他未明确分配到责任人的工作

安全委员会

安全委员会（SC）使命

openEuler 安全委员会（SC）是负责接收和响应 openEuler 产品安全问题报告、提供社区安全指导，开展安全治理的组织。它的使命是：为 openEuler 用户提供最安全的产品和开发环境。

工作职责

- 协助漏洞修复：确保及时修复已知漏洞。通过为软件包 Maintainer 们提供补丁帮助，帮助用户系统在成为攻击受害者之前进行漏洞修复，包括提供相关漏洞检测和修复工具。
- 响应安全问题：响应上报的安全问题，跟踪安全问题的处理进展，并遵循安全问题披露策略对安全问题在社区内进行披露和公告。
- 安全编码规则：普及安全编码知识是安全团队的目标。安全团队会努力创建文档或开发工具来帮助开发团队避免软件开发过程中的常见陷阱。安全团队还会尝试回答在开发和使用过程中遇到的任何问题。
- 参与代码审核：安全团队希望能够通过代码审核帮助团队提前发现代码中的漏洞。

技术委员会

openEuler 技术委员会（Technical Committee，以下和TC互换使用）是 openEuler 社区的技术决策机构，负责社区技术决策和技术资源的协调。 TC 委员（含主席）名额不得超过 9 名。主席及委员由理事会任命，任期一年。

技术委员会的主要职责如下：

- 负责回答理事会提出的技术问题，支撑理事会对战略蓝图的技术发展方向做出判断；
- 以远程协作的方式运行，每半年召开一次面对面沟通的正式会议，正式会议间定期召开例行线上公开会议；
- 对社区技术路线、接口定义、架构设计、构建发布等进行指导，并逐步构建社区规则；
- 协调跨项目合作，对社区跨项目技术问题进行指导，并逐步构建社区规则；
- 制定、指导项目孵化、开发、退出流程，支撑社区技术生态健康发展；
- 制定、指导软件包接纳、退出 openEuler 的流程，支撑 openEuler 开源版本的可信和可靠；
- 接受用户委员会的反馈（需求和问题），牵引社区资源将其落地至项目；
- 建立社区认证标准和平台，为社区认证（OS 商业发行版认证、硬件兼容性认证等）提供技术支撑；

组织会议

公开的会议时间：当前 TC 在北京时间，每双周周三的上午 10:00 召开公开的例行线上讨论会议。

具体会议参会方式，会提前一天在邮件列表中讨论确定。目前主要采用华为公司提供的 welink 语音会议系统。

TC 会议的例行议题包括：

- 对新的 SIG 申请进行评审
- 对 openEuler 新增软件进行评审
- 通报用户反馈，并讨论如何响应
- 审视原创项目运作情况（一次会议审视一个项目）
- 其他议题由 TC 成员在会前邮件发起
- 议题收集和整理通过

https://gitee.com/openeuler/community/tree/master/zh/technical-committee/next-meeting-topics.md 公开

欢迎任何感兴趣的开发者参加旁听

SIG

专注于一个领域的持久和开放的团队，该团队通过定期的任务和活动实现特定的交付目标。SIG 具有公开透明的程序，要遵循 openEuler 的行为准则。任何人都可以参与并作出贡献。所有的 SIG 都存在于https://openeuler.org/zh/sig.html。

- 子项目：子项目是 SIG 内为实现特定交付目标或成果而成立的，可以独立工作。子项目有一个或多个仓库，是 openEuler 社区内主要交付成果的输出组织，所有的子项目都在其所属的 SIG 内。项目的交付件要进入社区发行范围，可以向技术委员会提交申请。
- 子项目有两个阶段：孵化阶段和成熟阶段。成熟项目的软件包可以进入社区发行光盘。孵化项目可以申请软件包进入社区发行的 /extra（不在光盘内的额外的软件包）目录或 /experimental（探索、实验性质的软件包）目录
- SIG 组的建立：SIG 组的建议由相关提议人在 TC 例会上进行议题申报，并进行集体评议。如果申请经评议通过，提议人需要按照流程在社区提交 PR，建立相关的 SIG 页面等。PR 经委员会成员审议合入。新 SIG 组成立后，由 TC 技术委员会指定一个委员作为导师，在 SIG 组的运行初期进行指导，确保 SIG 组快速步入正轨。
- SIG 组的撤销：TC 委员会可以依据以下的原则，经过讨论将 SIG 撤销：
  - SIG 组的工作无法满足社区版本的要求，阻碍了 openEuler 社区版本的发布。
  - SIG 组无法正常运转，包括无固定例会，无法及时响应社区 Issue，所负责的软件无及时更新等。

  SIG 组的撤销依照如下流程实施：
  - 由 TC 委员会中的一个委员提出 SIG 组撤销申请。
  - 该申请在 TC 例会上进行讨论并投票决策。投票原则按照简单多数票原则。

  当 SIG 组被撤销后，该 SIG 组名下的软件包依照其合理归属划归其它 SIG 组。

最终用户委员会

最终用户委员会的主要职责是获取用户在使用 openEuler 操作系统时的需求， 来指导技术工作。

市场营销委员会

随着 openEuler 社区的发展，社区计划持续组织多种活动，或和其它社区、组织机构等联合举办宣传活动。负责组织 openEuler 的社区活动，包括但不限于 Meetup、Summit、在线直播等，同时协调社区成员参与活动，并在活动前后的协调渠道进行广泛宣传。 同时通过定期的 SIG 会议沟通活动机会、协调组织工作、协同宣传等等工作。

贡献

做出贡献的第一步是从 openEuler 的 SIG/项目列表中选择。开始参加 SIG/项目会议，加入 IRC 频道并订阅邮件列表。 SIG/项目通常会由一系列 help-wanted 的 Issue，这些 Issue 可以帮助新的贡献者参与进来。

签署 CLA

您必须首先签署“贡献者许可协议”（CLA），然后才能参与社区贡献。

社区行为准则

openEuelr 社区遵守开源社区《贡献者公约》V1.4中规定的行为守则，请参考 V1.4 版本

如需举报侮辱、骚扰或其他不可接受的行为，您可以发送邮件至 tc@openeuler.org，联系 openEuler 技术委员会处理。

贡献者们的承诺

为建设开放友好的环境，我们贡献者和维护者承诺：不论年龄、体型、身体健全与否、民族、性征、性别认同与表征、经验水平、教育程度、社会地位、国籍、相貌、种族、信仰、性取向，我们项目和社区的参与者皆免于骚扰。

我们的准则

有助于创造积极环境的行为包括但不限于：

- 措辞友好且包容
- 尊重不同的观点和经验
- 耐心接受有益批评
- 关注对社区最有利的事情
- 与社区其他成员友善相处

参与者不应采取的行为包括但不限于：

- 发布与性有关的言论或图像、不受欢迎地献殷勤
- 捣乱/煽动/造谣行为、侮辱/贬损的评论、人身及政治攻击
- 公开或私下骚扰
- 未经明确授权便发布他人的资料，如住址、电子邮箱等
- 其他有理由认定为违反职业操守的不当行为

我们的义务

项目维护者有义务诠释何谓“妥当行为”，并妥善公正地纠正已发生的不当行为。

项目维护者有权利和义务去删除、编辑、拒绝违背本行为标准的评论（comments）、提交（commits）、代码、wiki 编辑、问题（Issues）等贡献；项目维护者可暂时或永久地封禁任何他们认为行为不当、威胁、冒犯、有害的参与者。

适用范围

本行为标准适用于本项目。当有人代表本项目或本社区时，本标准亦适用于此人所处的公共平台。

代表本项目或本社区的情形包括但不限于：使用项目的官方电子邮件、通过官方媒体账号发布消息、作为指定代表参与在线或线下活动等。

代表本项目的行为可由项目维护者进一步定义及解释。

贯彻落实

可以致信 tc@openeuler.org，向项目团队举报滥用、骚扰及不当行为。

维护团队将审议并调查全部投诉，妥善地予以必要的回应。项目团队有义务保密举报者信息。具体执行方针或将另行发布。

未切实遵守或执行本行为标准的项目维护人员，经项目负责人或其他成员决议，可能被暂时或永久地剥夺参与本项目的资格。

社区贡献

在社区上总是有可以改进的文档（比如您正在阅读的），需要检视的代码，可以重构或注释的函数或变量，可以持续补充和优化的测试用例。我们将帮助您了解 openEuler SIG 的组织方式，并引导您顺利的开始您的第一个贡献。

找到您感兴趣的工作

了解 SIG

SIG 就是 Special Interest Group 的缩写，openEuler 社区按照不同的 SIG 来组织，以便于更好的管理和改善工作流程。

- SIG 组是开放的，欢迎任何人加入并参与贡献。
- SIG 都是针对特定的一个或多个技术主题而成立的。SIG 内的成员推动交付成果输出，并争取让交付成果成为 openEuler 社区发行的一部分。
- SIG 的核心成员主导 SIG 的治理。您可以在贡献的同时积累经验和提升影响力。
- 每一个 SIG 在 Gitee 上都会拥有一个或多个项目，这些项目会拥有一个或多个 Repository。SIG 的交付成果会保存在这些 Repository 内。
- 可以在 SIG 对应的 Repository 内提交 Issue、针对特定问题参与讨论，提交和解决问题，参与评审等。
- 您也可以通过邮件列表、IRC 或视频会议和 SIG 内的成员进行交流。

找到您感兴趣的 SIG 或项目

找到您感兴趣的 SIG 组，可以帮助您在正确的地方提出问题，并得到更快的社区响应。

- 方式一：如果您不了解有哪些 SIG 或项目，您可以查看 SIG 列表，它包含当前 openEuler 社区成立的所有 SIG 团队的清单。您可以通过该列表快速的定位到您感兴趣的领域所对应 SIG 团队。同时还会您提供该 SIG 团队的如下信息：
  - SIG 下的项目，以及项目的 Repository 地址
  - SIG 内的交流方式，包括邮件列表、IRC 或视频会议等
  - Maintainer 的联系方式
- 方式二：如果您知道感兴趣的项目名称，可以在 openEuler 的 Repository 列表下进行模糊搜索，从而快速定位到对应项目的首页地址。通常情况下，在该项目首页地址的 README.md 文件中，可以找到该项目所属的 SIG 信息、交流方式、成员和联系方式等。

如果上述两种方式都定位不到您感兴趣的 SIG，您可以向 community@openeuler.org 发求助邮件。建议您在邮件列表内用“【开发过程疑问】”作为标题，在内容中写出你寻找的 SIG 或项目的特征，我们会为您提供帮助。

开始您的贡献

- **找一个 Issue**
  - 找到 Issue 列表：在您感兴趣的项目的首页内（Gitee 上的项目的 Repository）的工具栏，点击“Issues”，您可以找到该 SIG 的 Issue 列表（如 Community 团队的 Issue 列表地址为https://gitee.com/openeuler/community/issues）
  - 找到愿意处理的 Issue：如果您愿意处理其中的一个 Issue，可以将它分配给自己。只需要在评论框内输入 /assign或 /assign @yourself，机器人就会将问题分配给您，您的名字将显示在负责人列表里。
  - 参与 Issue 内的讨论：每个 Issue 下面可能已经有参与者们的交流和讨论，如果您感兴趣，也可以在评论框中发表自己的意见。
- 提出问题或建议
  - 提出问题：如果您发现并想向社区上报问题或缺陷，问题提交的方式就是创建一个 Issue。您只要将问题以 Issue 的方式提交到该项目 Repository 的 Issue 列表内。提交问题时，请尽量遵守问题提交准则。
  - 提出建议：如果您想对 SIG 领域内贡献出自己的意见或建议，也可以通过提交 Issue 的方式分享给大家。大家可以在该 Issue 上充分的交流和讨论。为了吸引更广泛的注意，您也可以把 Issue 的链接附在邮件内，通过邮件列表发送给所有人。
- **搭建开发环境**

  1. 安装 openEuler

  2. 开发环境准备

  3. 载和构建软件包

  具体详情参见附录 1。
- **参与编码贡献**
  - **了解 SIG 和项目内的开发注意事项**

    每个 SIG 内的项目使用的编码语言、开发环境、编码约定等都可能存在差异的。如果您想了解并参与到编码类贡献，可以先找到该项目给开发者提供的贡献者指南——这个指南一般是在该 SIG 的首页地址内，以 CONTRIBUTING.md 文件的形式提供，或者就直接在该项目的 README.md 内。

    除了这些文件外，SIG 可能还会提供其他指南信息。这些信息位于 SIG 或其项目的特定社区目录中。如果您未找到相关信息，或者对相关信息有疑问，可以在该 SIG 内提交 Issue，或者把问题或疑问发送到该项目所属 SIG 的邮件列表。如果您认为长时间没有得到回应，可以向 community@openeuler.org 求助。
  - **下载代码和拉分支**

    如果要参与代码贡献，您还需要了解如何在 Gitee 下载代码，通过 PR 合入代码等。该托管平台的使用方法类似 GitHub，如果您以前使用 GitHub，本章的内容您可以大致了解甚至跳过。
  - **修改、构建和本地验证**

    在本地分支上完成修改后，进行构建和本地验证。
  - **提交一个 Pull-Request**

    当你提交一个 PR 的时候，就意味您已经开始给社区贡献代码了。
  - **如何新增软件包**

    openEuler 支持在 gitee 新增软件包的同时自动在 obs 的 openEuler:Fctory 上创建同名仓库。这样在向已创建的 gitee 仓库提交代码时，会自动对代码编译进行检测。
- **检视代码**

  openEuler 是一个开放的社区，我们希望所有参与社区的人都能成为活跃的检视者。

  对于贡献者，为了使您的提交更容易被接受，您需要：
  - 遵循 SIG 组的编码约定，如果有的话
  - 准备完善的提交信息
  - 如果一次提交的代码量较大，建议将大型的内容分解成一系列逻辑上较小的内容，分别进行提交会更便于检视者理解您的想法

- 使用适当的 SIG 组和监视者标签去标记 PR：社区机器人会发送给您消息，以方便您更好的完成整个 PR 的过程

对于检视者，强烈建议本着行为准则，超越自我，相互尊重和促进协作。《补丁审核的柔和艺术》一文中提出了一系列检视的重点，说明代码检视的活动也希望能够促进新的贡献者积极参与，而不会使贡献者一开始就被细微的错误淹没，所以检视的时候，可以重点关注包括：

- 贡献背后的想法是否合理
- 贡献的架构是否正确
- 贡献是否完善

注意：如果您的 PR 请求没有引起足够的关注，可以在 SIG 的邮件列表或 dev@openeuler.org 求助。

- 选择社区组件打包

制作 RPM 包，俗称打包，是指编译并捆绑软件与元数据例如软件全名、描述、正常运行所需的依赖列表等等的任务。这是为了让软件使用者可以使用软件包管理器舒服的安装、删除或者升级他们所使用的软件。

打包规则

openEuler 试图规范化多种多样的开源项目到一个连贯的系统。因此 openEuler 制定此打包指导来规范制作 RPM 的动作。

- openEuler 遵守一般的 Linux 基础标准(LSB)。该标准致力于减少各个发行版间的不同。
- openEuler 也遵守 Linux 文件系统层级标准(FHS)。该标准是关于如何管理 Linux 文件系统层级的参考。
- 除了遵守这些一般 Linux 发行版都会遵守的一般规则，本文档规范化了为 openEuler 社区版打包的实际细节问题。

- 社区安全问题披露
  - 安全处理流程
  - 安全披露信息

安全处理流程和安全批露信息请参考附录 2。

著作权说明

商标

附录

附录 1：搭建开发环境

环境准备	地址
下载安装 openEuler	https://openeuler.org/zh/download.html
开发环境准备	https://gitee.com/openeuler/community/blob/master/zh/contributors/prepare-environment.md
构建软件包	https://gitee.com/openeuler/community/blob/master/zh/contributors/package-install.md

附录 2：安全处理流程和安全批露信息

社区安全问题披露	地址
安全处理流程	https://gitee.com/openeuler/community/blob/master/zh/security-committee/security-process.md
安全披露信息	https://gitee.com/openeuler/community/blob/master/zh/security-committee/security-disclosure.md