openEuler 22.09 Technical White Paper

1. Introduction

openEuler has evolved from a simple server operating system (OS) into a digital infrastructure OS that fits into any server, cloud computing, edge computing, and embedded deployment. It provides a secure, stable, and easy-to-use open source OS that is compatible with multiple computing architectures. openEuler suits operational technology (OT) applications and enables the convergence of OT and information and communications technology (ICT).

The openEuler open source community is a portal available to global developers, with the goal of building an open, diversified, and architecture-inclusive software ecosystem for all digital infrastructure scenarios. It has a rich history of helping enterprises develop their software, hardware, and applications.

The openEuler open source community was officially established on December 31, 2019, with the original focus of innovating diversified computing architectures.

On March 30, 2020, the Long Term Support (LTS) version openEuler 20.03 was officially released, which was a new Linux distribution with independent technology evolution.

Later in 2020, on September 30, the innovative openEuler 20.09 version was released through the collaboration efforts of multiple companies, teams, and independent developers in the openEuler community. The release of openEuler 20.09 marked a milestone not only in the growth of the openEuler community, but also in the history of open sourced software in China.

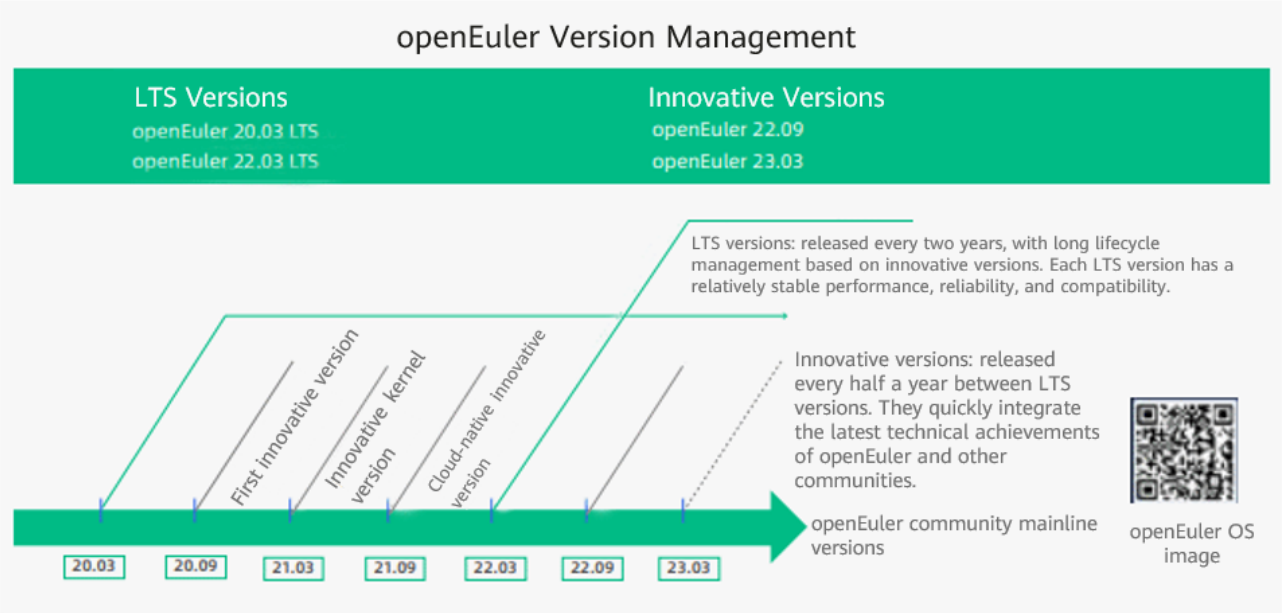
On March 31, 2021, the innovative kernel version openEuler 21.03 was released. This version is enhanced in line with Linux kernel 5.10 and also incorporates multiple new features, such as live kernel upgrade and tiered memory expansion. These features improve multi-core performance and deliver the computing power of one thousand cores.

Fast forward to September 30, 2021, openEuler 21.09 was released. This premium version is designed to supercharge all scenarios, including edge and embedded devices. It enhances server and cloud computing features, and incorporates key technologies including cloud-native CPU scheduling algorithms for hybrid service deployments and KubeOS for containers.

On March 30, 2022, openEuler 22.03 LTS was released based on Linux kernel 5.10. Designed to meet all server, cloud, edge computing, and embedded workloads, openEuler 22.03 LTS is an all-scenario digital infrastructure OS that unleashes premium computing power and resource utilization.

On September 30, 2022, openEuler 22.09 was released, which unleashes diversified computing power for all-scenario innovations. It further empowers application porting and the interworking between

openEuler and OpenHarmony.

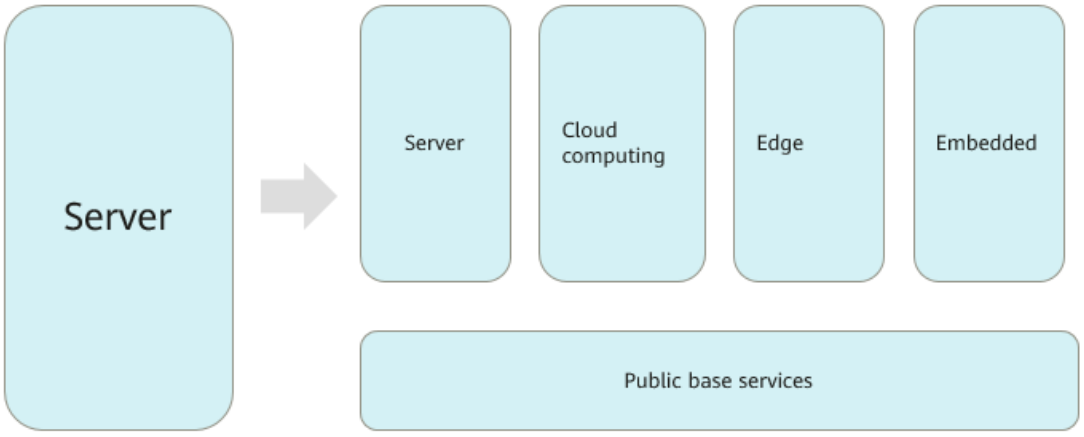


As an OS platform, openEuler releases an LTS version every two years. Each LTS version provides enhanced specifications and a secure, stable, and reliable OS for enterprise users.

openEuler is built on tried-and-tested technologies. A new openEuler innovative version is released every 6 months to quickly integrate the latest technical achievements of openEuler and other communities. The innovative tech is first verified in the openEuler open source community as a single open source project, and then these features are added to each new release, enabling community developers to obtain the source code.

Technical capabilities are first tested in the open source community, and continuously incorporated into each openEuler release. In addition, each release is built on feedback given by community users to bridge the gap between innovation and the community, as well as improve existing technologies. openEuler is both a release platform and incubator of new technologies, working in a symbiotic relationship that drives the evolution of new versions.

Innovative Platform for All Scenarios

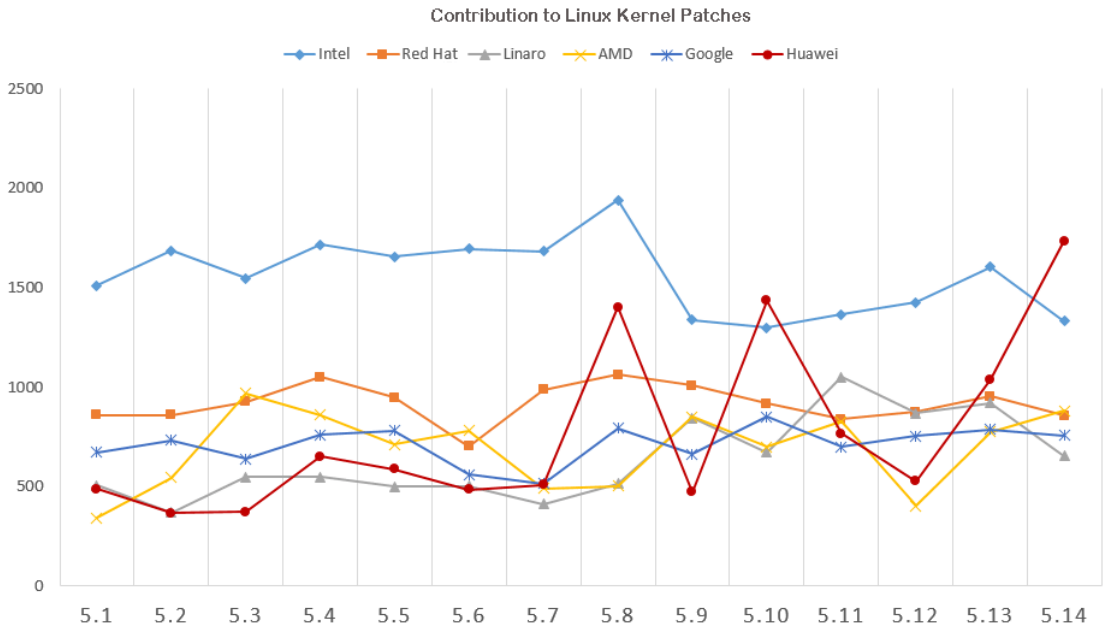


openEuler supports multiple processor architectures (x86, Arm, SW64, and RISC-V) and will support other brands (such as PowerPC) in the future, as part of a focus to continuously improve the ecosystem of diversified computing power.

The openEuler community is home to an increasing number of special interest groups (SIGs), which are dedicated teams that help extend the OS features from server to cloud computing, edge computing, and embedded scenarios. The openEuler OS covers all scenarios, and comprises the 22.09 Edge and 22.09 Embedded editions designed for edge computing and embedded deployments, respectively.

The OS is a perfect choice for ecosystem partners, users, and developers who plan to enhance scenario-specific capabilities. By creating a unified OS that supports multiple devices, openEuler hopes to enable a single application development for all scenarios.

Continuous Contribution to the Linux Kernel



As a major contributor to the Linux kernel, the kernel development team is responsible for enhancing the processor architectures, Advanced Configuration and Power Interface (ACPI), memory management, file systems, media, kernel documents, bug fixes, and code rebuilds. Over the past decade, openEuler has contributed more than 17,000 patches to the Linux kernel.

In Linux kernels 5.10 and 5.14, openEuler's code contribution ranks No.1 in the world. openEuler is committed to kernel innovation and has been continuously contributing to upstream communities.

Open and Transparent: The Open Source Software Supply Chain

The process of building an open source OS relies on supply chain aggregation and optimization. To ensure reliable open source software or a large-scale commercial OS, openEuler comprises a complete lifecycle management that covers building, verification, and distribution. The brand regularly reviews its software dependencies based on user scenarios, organizes the upstream community addresses of all the software packages, and verifies its source code by comparing it to that of the upstream communities. The build,

runtime dependencies, and upstream communities of the open source software form a closed loop, realizing a complete, transparent software supply chain management.

2. Platform Architecture

System Framework

openEuler is an innovative open source OS platform built on kernel innovations and a solid cloud base to cover all scenarios. It is built on the latest trends of interconnect buses and storage media, and offers a distributed, real-time acceleration engine and base services. It provides competitive advantages in edge and embedded scenarios, and is the first step to building an all-scenario digital infrastructure OS.

openEuler 22.09 runs on Linux kernel 5.10 and provides POSIX APIs and OS releases for server, cloud native, edge, and embedded environments. It is a solid foundation for intelligent collaboration across hybrid and heterogeneous deployments. openEuler 22.09 is equipped with a distributed soft bus and KubeEdge+ edge-cloud synergy framework, among other premium features, making it a perfect choice for collaboration over digital infrastructure and everything connected models.

In the future, the openEuler open source community will continue to innovate, aiming to promote the ecosystem and consolidate the digital infrastructure.

Kernel innovations:

• **Enhanced cloud-native scheduling**: openEuler suits hybrid deployments of online and offline cloud services. Its innovative CPU scheduling algorithm ensures real-time CPU preemption and jitter suppression for online services. Additionally, its innovative memory reclamation algorithm against out of memory (OOM) allows online services to run reliably based on their higher service priorities.

• **EulerFS**: A new file system is designed for non-volatile dual in-line memory modules (NVDIMMs). It uses technologies such as soft updates and dual-view directories to accelerate file metadata synchronization and thus improve file read and write performance.

• **Tiered memory expansion etMem**: With the user-mode swap function, the discarded

cold memory can be changed to the user-mode storage based on a preset policy. The user-mode swap delivers a higher performance than the kernel-mode swap and the whole swap process is transparent to users.

• **Enhanced memory reliability, availability, and serviceability (RAS)**: The tiered-reliability memory technology (technical preview feature) preferentially allocates high-reliability memory for sensitive data, such as kernels and key processes. This technology reduces the system breakdown rate and enhances system reliability.

Cloud base:

• **KubeOS for containers**: In cloud native scenarios, the OS is deployed and maintained in containers, allowing the OS to be managed based on Kubernetes, just as service containers.

• **Secure container solution**: Compared with the traditional Docker+QEMU solution, the iSulad+shimv2+StratoVirt secure container solution reduces the memory overhead and boot time by 40%.

• **Dual-plane deployment tool eggo**: OSs can be installed with one click for Arm and x86 hybrid clusters, while deployment of a 100-node cluster is possible within just 15 minutes.

New scenarios:

• **Edge computing**: openEuler 22.09 Edge is released for edge computing scenarios. It integrates the KubeEdge+ edge-cloud synergy framework to provide unified management, provisioning of edge and cloud applications, and other capabilities.

• **Embedded**: openEuler 22.09 Embedded is released for embedded scenarios, helping compress images under 5 MB and image loading within 5 seconds.

Flourishing community ecosystem:

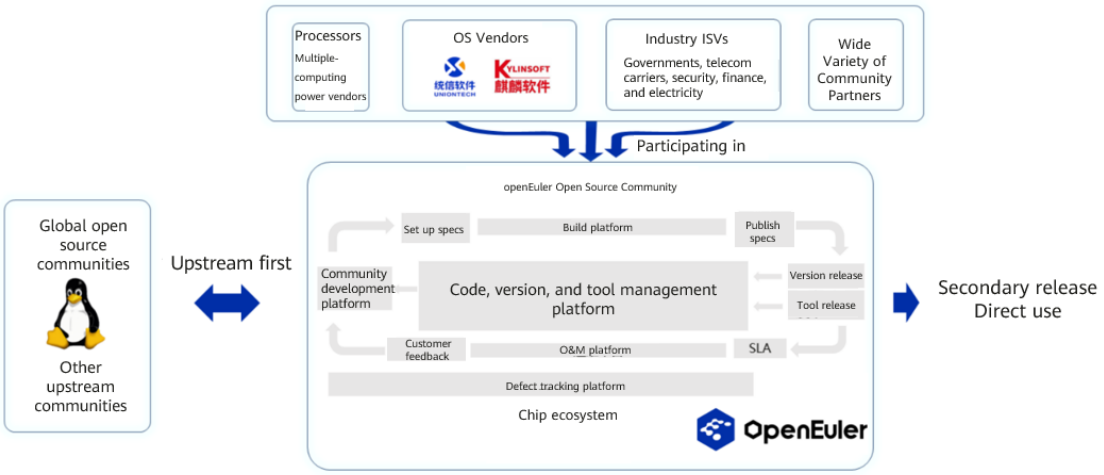
• **Desktop environments**: UKUI, DDE, Xfce, Kiran-desktop, and GNOME.

• **openEuler DevKit**: Supports OS migration, compatibility assessment, and various development tools such as secPaver which simplifies security configuration.

Platform Framework

The openEuler open source community partners with upstream and downstream

communities to advance the evolution of openEuler versions.



Hardware Support

The openEuler open source community works with multiple vendors to build a vibrant southbound ecosystem. While related features will be incorporated in subsequent 22.09 updates, openEuler 22.09 supports Intel Ice Lake and AMD Milan processors and provides basic support for the next-gen Intel Sapphire Rapids processor.

All openEuler versions support x86, Arm, ShenWei, Loongson, and RISC-V CPU architectures, and Intel, AMD, and Zhaoxin CPU chips. openEuler can run on servers from multiple hardware vendors and is compatible with NIC, RAID, Fibre Channel, GPU & AI, DPU, SSD, and security cards.

openEuler supports the following CPU architectures:

Hardware Type	x86	Arm	ShenWei	Loongson	RISC-V
CPU	Intel, AMD, Zhaoxin, Hygon	Kunpeng, Phytium	ShenWei	Loongson	Nuclei, StarFive, UC Techip

openEuler supports the following servers:

Hardware Type	x86	Arm	ShenWei	Loongson
Server	**Intel**: xFusion, H3C, Inspur, Lenovo,	**Kunpeng**: TaiShan, Tongfang, H3C,	ShenWei	Loongson

	Nettrix, SUPERCLOUD, PowerLeader, Supermicro, ZTE **AMD**: H3C, Lenovo, Supermicro **Hygon**: H3C, Sugon/Suma **Zhaoxin**: Zhaoxin	PowerLeader, Digital China, Yangtze Computing, Huanghe, Sichuan Hongxin, Xiangjiang Kunpeng, 100 Trust, Tiangong **Phytium**: QS, H3C, PowerLeader, Lenovo, Tongfang, Skysolidiss		

openEuler supports the following cards:

Hardware Type	x86	Arm
NIC	Huawei, Mellanox, Intel, Broadcom, Marvell, NetSwift	Huawei, Mellanox, Broadcom, Marvell, NetSwift, Intel
RAID	Huawei, Avago, PMC	Huawei, Avago, PMC
Fibre Channel	Huawei, Marvell, Qlogic, Emulex	Huawei, Marvell, Qlogic, Emulex
GPU & AI	Huawei, NVIDIA, AMD, Iluvatar CoreX, Intel	Huawei, NVIDIA, AMD, Iluvatar CoreX, Intel
DPU	Jaguar Microsystems	
SSD	Huawei, Samsung, Intel	Huawei, Samsung, Intel, Dera
Security	Sansec	Sansec

For the complete compatibility list, visit https://www.openeuler.org/en/compatibility/.

3. Operating Environments

Servers

To install openEuler on a physical machine, check that the physical machine meets the compatibility and hardware requirements.

For a full list, visit https://openeuler.org/en/compatibility/.

Item	Configuration Requirement
Architecture	AArch64, x86_64
Memory	At least 4 GB

Drive	At least 20 GB

VMs

openEuler supports the following virtual machines (VMs):

1. centos-7.9 qemu 1.5.3-175.el7 libvirt 4.5.0-36.el7 virt-manager 1.5.0-7.el7
2. centos-8 qemu 2.12.0-65.module_el8.0.0+189+f9babebb.5
   libvirt 4.5.0-24.3.model_el8.0.0+189+f9babebb virt-manager 2.0.0-5.el8
3. fedora 32 qemu 4.2.0-7.fc32 libvirt 6.1.0-2.fc32 virt-manager 2.2.1-3.fc32
4. fedora 35 qemu 6.1.0-5.fc35 libvirt 7.6.0-3.fc35 virt-manager 3.2.0-4.fc35

Item	Configuration Requirement
Architecture	AArch64, x86_64
CPU	2 CPUs
Memory	At least 4 GB
Drive	At least 20 GB

Edge Devices

To install openEuler on an edge device, check that the edge device meets the compatibility and minimum hardware requirements.

Item	Configuration Requirement
Architecture	AArch64, x86_64
Memory	At least 4 GB
Drive	At least 20 GB

Embedded Devices

To install openEuler on an embedded device, check that the embedded device meets the compatibility and minimum hardware requirements.

Item	Configuration Requirement
Architecture	AArch64, AArch32
Memory	At least 512 MB

Drive	At least 256 MB

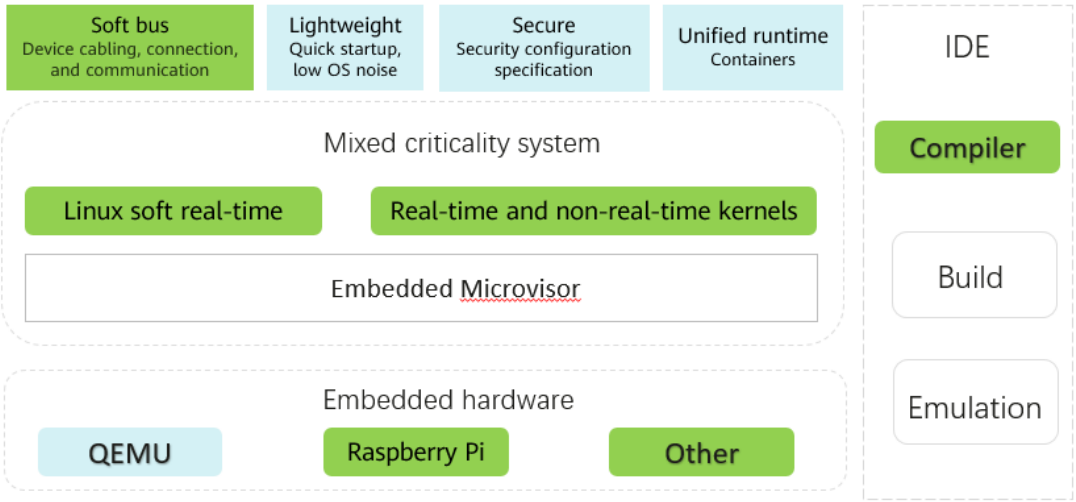
4. Scenario-specific Innovations

openEuler 22.09 includes the Edge edition for edge computing and the Embedded edition for embedded systems. The OS empowers all-scenario collaboration over the new generation of digital infrastructure.

Embedded

openEuler 22.09 Embedded offers a distributed soft bus and software package build capabilities to allow for mixed criticality deployment of real-time and non-real-time planes. Based on the participation of ecosystem partners, users, and developers of the openEuler open source community, openEuler 22.09 Embedded helps to design efficient embedded OS solutions. It will attain greater support for chip architectures such as PowerPC and RISC-V, and add capabilities such as deterministic latency, industrial middleware, and simulation systems.

Feature Description



openEuler 22.09 provides the following features for embedded scenarios:

1. **Lightweight deployment**: The open source Yocto is a small-scale and lightweight framework that allows you to customize OS images. It can compress OS images to under 5 MB and shorten OS startup time to under 5s.

2. **Support for diverse hardware types**: Among others, Raspberry Pi is a new device that can serve as the universal hardware for embedded deployments.
3. **Soft real-time kernel**: This capability is inherited from Linux kernel 5.10, and helps respond to soft real-time interrupts within microseconds.
4. **Mixed criticality deployment**: Real-time (Zephyr) and non-real-time planes can coexist on an SOC based on Raspberry Pi (new support in openEuler 22.09). The lifecycle management of real-time systems is also supported.
5. **Distributed soft bus (DSoftBus)**: The DSoftBus and HiChain point-to-point authentication module available in OpenHarmony are used to implement interconnection and interworking between embedded devices running on openEuler as well as between openEuler embedded devices and OpenHarmony devices (new support in openEuler 22.09).
6. **Embedded software packages**: Over 80 common embedded software packages can be built using openEuler.
7. **Hard real-time kernel (new support in openEuler 22.09)**: The open source Real Time Operating System (RTOS) kernel, UniProton, controls the context switchover latency down to 2 μs and the interrupt latency to 1 μs.

   The following features will be available soon:

1. **Unified APIs**: The hard real-time kernel supports POSIX APIs to simplify application development.
2. **Industry security certifications**: openEuler 22.09 is currently under review for different standards and certifications, including IEC61508 and EC62443.

Application Scenarios

Embedded systems help supercharge computing performance in a wide range of industries and fields, including aerospace, industrial control, telecommunications, automobiles, and healthcare. Mature 5G and AI technologies will enable embedded systems to meet the demands for intelligent management of IoT and edge computing devices.

5. Kernel Innovations

What's New in the openEuler Kernel

openEuler 22.09 runs on Linux kernel 5.10 and inherits the competitive advantages of community versions and innovative features released in the openEuler community.

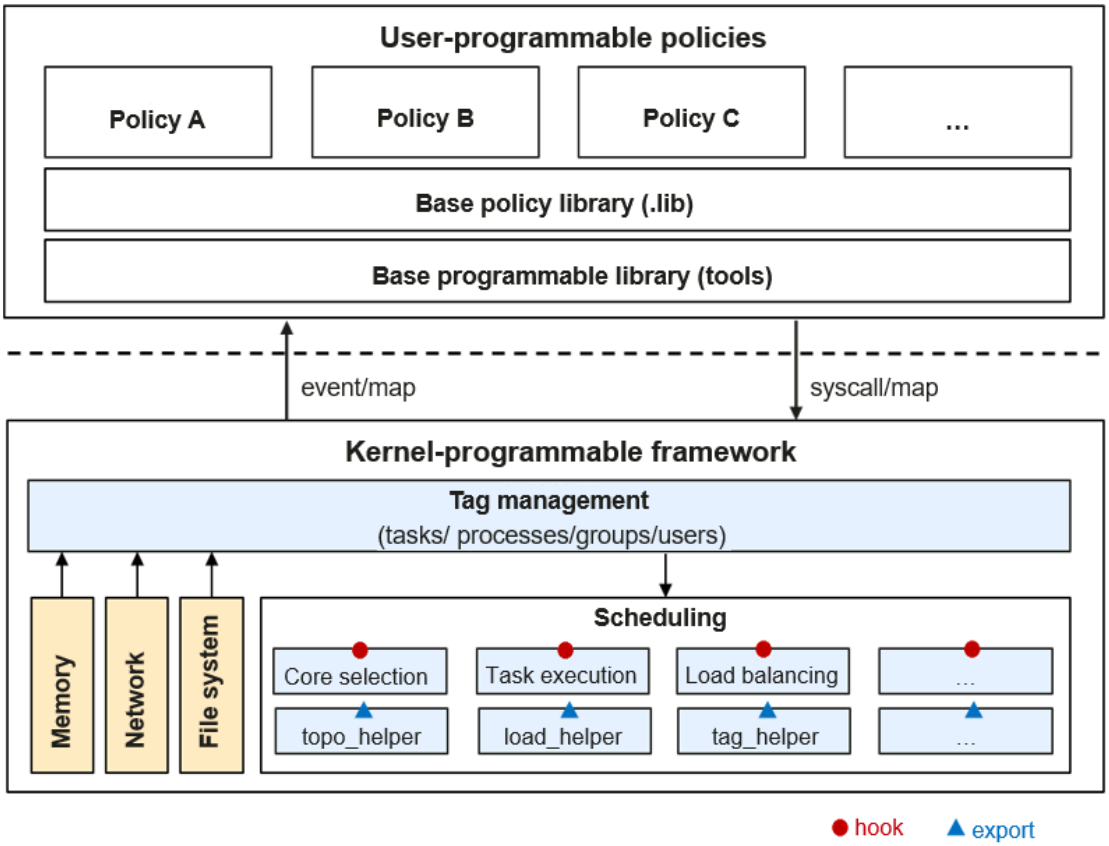
- **BPF CO-RE (Compile Once-Run Everywhere)**: Solves portability issues of BPF. After being compiled and passing kernel verification, the compiled program can run on kernels of different versions without recompilation.
- **Memory RAS – reliable memory**: The kernel, key processes, memory file system, and file cache use reliable memory to prevent kernel reset caused by multi-bit errors (MBEs). Compared with openEuler 22.03 LTS, the support for reliable memory is more complete and extensive.
- **Memory RAS – enhanced UCE tolerance**: In the event the copy_from_user experiences a multi-bit error (MBE), the affected processes can be killed to avoid kernel reset.
- **Kernel-programmable scheduling framework**: Preemption, core selection, task execution, and code examples.
- **BPF-based kernel cache**: Greatly improves Redis performance.
- **Support for AArch64 scalable matrix extension (SME)**: The next-generation SIMD is equipped, with more powerful functions than Arm's Neon, providing better HPC and machine learning performance for AArch64 platforms.
- **Rust for Linux driver framework**: Provides Rust-related infrastructure and facilitates Linux driver compiling.
- **Huge-page programs**: Reduces TLB misses and improves application performance.
- **ShangMi (SM) series cryptographic algorithms**: SM3 and SM4.
- **SM series cryptographic algorithms**: module signing.

Programmable Kernel

The eBPF-based programmable scheduling framework enables the kernel scheduler to extend scheduling policies and better meet varying loads. It has the following features:

- **Tag management mechanism**: The capability of tagging tasks and task groups is open. Users and kernel subsystems can tag specific workloads by calling interfaces. The scheduler can detect tasks of specific workloads by tag.
- **Policy extension**: The programmable scheduling framework supports policy extension for completely fair scheduling (CFS) preemption, core selection, and task execution, and adds new extension points and various auxiliary methods to extend policies.

Feature Description



- **Base library functions and policy library**: Provides basic library functions and custom scheduling policy templates for quick orchestration and extension of user-mode policies.
- **Tag management mechanism**: Supports user-defined extended tags for objects such as tasks, processes, groups, and users, and bears the semantics of collaborative scheduling between user-mode and kernel-mode components.
- **Scheduling component hook point and helper function**: Supports custom policy injections for CFS core selection, task execution, and preemption processes.

Application Scenarios

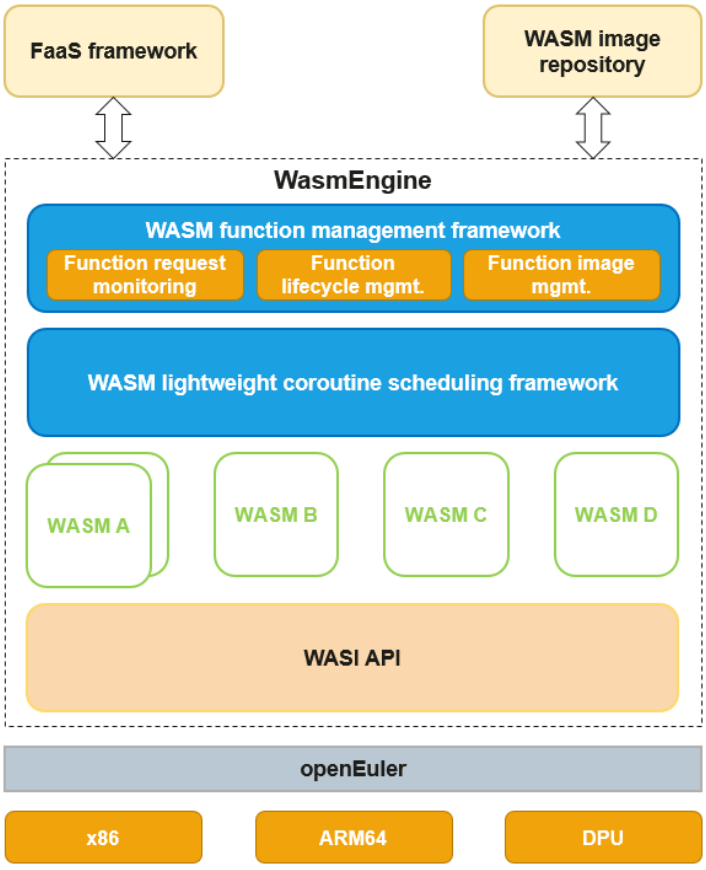
On the programmable kernel framework, developers and system administrators can create policies and dynamically load those policies to the kernel for execution.

WasmEngine

Framework as a Service (FaaS) is a new computing paradigm of cloud computing. It features agile development, auto scaling, pay per use, and minimized O&M, helping users build any types of applications and services with ease. Conventional container-based FaaS decouples custom computing capabilities from content delivery network (CDN) services and implements fast iteration and updates. However, its cold start speed and memory overhead of containers make it insufficient for quick execution and processing of tens of thousands of instances on a single node, such as those in high-concurrency, heavy traffic scenarios.

To solve this, openEuler provides WasmEngine sandbox solution based on the WebAssembly (WASM) technology to isolate functions in the WASM sandbox.

Feature Description



The functions of the lightweight WasmEngine are available thanks to the following two key

components:

1. **WASM function management framework**
   - Listens to and processes concurrent function requests.
   - Manages functions throughout their lifecycle.
   - Can work on Open Container Initiative (OCI) container images and manage local function image resources.
2. **WASM lightweight coroutine scheduling framework**

Abstracts the execution context of WASM instances, supports lightweight and high-performance user-mode coroutine scheduling models, and supports multiple WASM instance execution models such as JIT and AOT.

Application Scenarios

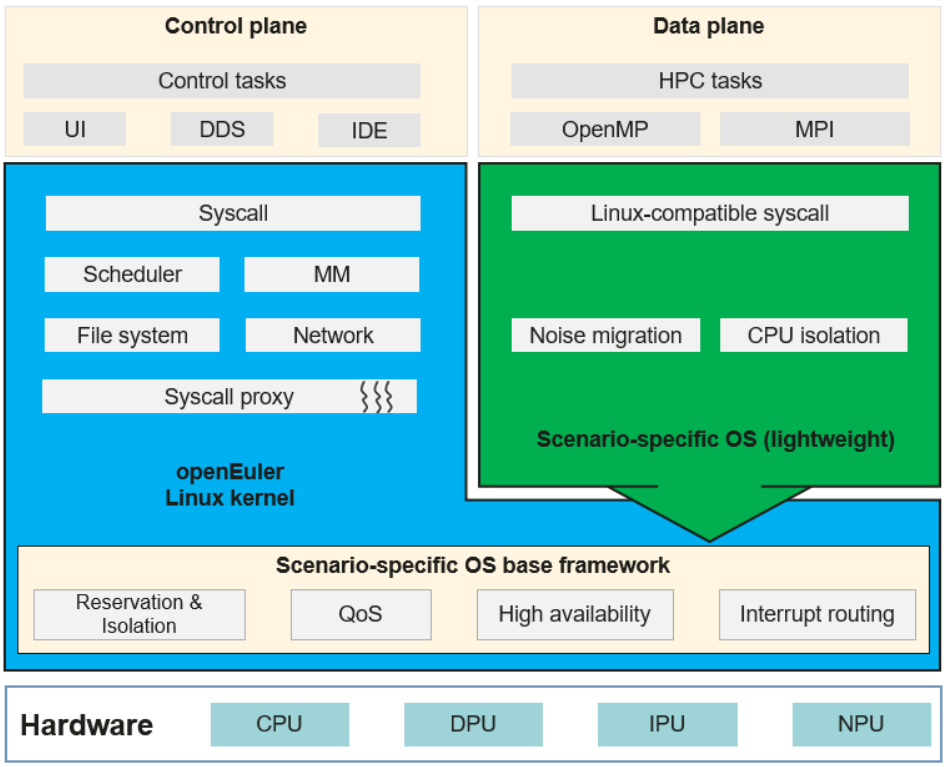
Stateless FaaS function tasks that run for a short period of time can be started on demand. For example, in the CDN edge computing scenario, custom request preprocessing functions allow for on-demand pulls and quick response.

Control-Data Plane Separation on HCK

High-performance computing (HPC) is a core foundation in large-scale clusters, providing concurrent computing and frequent data synchronization, but it is impacted by system noise, which can cause single-node fluctuations on overall performance and scalability of the cluster.

To ensure low system noise for HPC services, this design uses the computing base, High-performance Computing Kit (HCK), to separate the control and data planes and works with the HCK user-mode tool launcher to run applications on isolated CPUs, providing a low-noise isolated execution environment.

Feature Description



HCK provides the following functions:

1. Kernel-mode base

   - **CPU noise migration**: Migrates unwanted system noise from isolated cores.
   - **CPU isolation management**: Tags and reserves specified CPUs during system startup.
   - **Task domain management**: Creates task domains and configures affinity when processes are running on isolated CPUs.
   - **Topology filtering**: Controls the availability of CPU topology when some interfaces under proc and sysfs obtain the CPU topology.

2. User-mode tool

The user-mode tool connects to the sysfs interface provided by the preceding kernel-mode functions to run target applications on specified isolated CPUs.

Most HPC services run on the Bulk Synchronous Parallel (BSP) model featuring parallel computing + communication + synchronization. As mentioned, system noise has great

impact on service performance. System noise refers to non-application computing tasks executed during service running, including system/user-mode daemon processes, kernel daemon processes, memory management, system scheduling overheads, non-computing tasks of service applications, other noise (cache misses, page faults) caused by resource contention, among others. Analysis shows that when system noise is long (length) with a short noise interval, it has a much greater impact on the HPC application performance, while other issues include long application synchronization time and large number of execution nodes also affect performance. Large-scale systems like exascale computing systems will experience great performance deterioration from system noise. Therefore, measures must be taken to reduce system noise.

A control-data plane separation solution uses the OS to isolate HPC tasks from system management. The control-data plane separation isolates "noisy" computing tasks, alleviates resource contention, and ensures compatibility with the Linux ecosystem. These are described as follows:

To isolate HPC tasks and those tasks that cause noise, HPC tasks are run on the lightweight kernel, while system tasks, interrupt processing, and kernel threads are run on the Linux kernel to avoid interference on HPC tasks. The syscall proxy is used for scheduling. The kernel independently processes high-load system scheduling to ensure high-load tasks are run without affecting other system services.

Compatibility with the Linux ecosystem enables programs to run on the lightweight kernel without modification.

In short, the control-data plane separation aims to improve compatibility and universality to inherit the powerful Linux ecosystem (current kernel), and support scenario-specific OS development and operating tasks on the general framework.

Application Scenarios

For environments that call for a low level of system noise, a CPU isolation kernel base with control-data plane separation runs specific applications on isolated CPUs using a launcher.

6. Cloud Base

Enhanced StratoVirt Standard Virtualization

StratoVirt is an enterprise-grade virtualization platform designed for cloud data centers. "Strato" refers to "stratosphere" – the layer of the Earth's atmosphere immediately above the troposphere – and indicates a light protective layer that protects services on the openEuler platform.

StratoVirt offers the following protection features:

- **Robust security**: Offers language-level security based on Rust, while the modular design minimizes the attack surface and physically isolates each tenant.
- **Lightweight**: When running a simplified device model, StratoVirt can start it within 50 ms, and control the memory overhead within 4 MB.
- **Software and hardware collaboration**: Supports x86 VT and Kunpeng-V virtualization.
- **Lightning-fast scaling**: Scales devices within milliseconds, to provide flexible resource scaling for lightweight workloads.
- **Multi-scenario support**: A single architecture supports multiple scenarios, including serverless, secure containers, and standard VMs.

New Features

StratoVirt standard virtualization enhances the features in extended desktop VMs, and integrates northbound interfaces and common standard software and hardware ecosystems.

- **Positioning**: Desktop VMs in key scenarios with standard virtualization support VNC, USB keyboard and mouse, and virtio-gpu graphics desktop virtualization, expanding the application ecosystem.
- **Specifications**: Compatibility with common standards such as ACPI and UEFI enables VM edk2 boot, NUMA display, and CPU topology display.
- **Enhanced performance and abundant devices**: Virtio multi-queue and vhost-user-

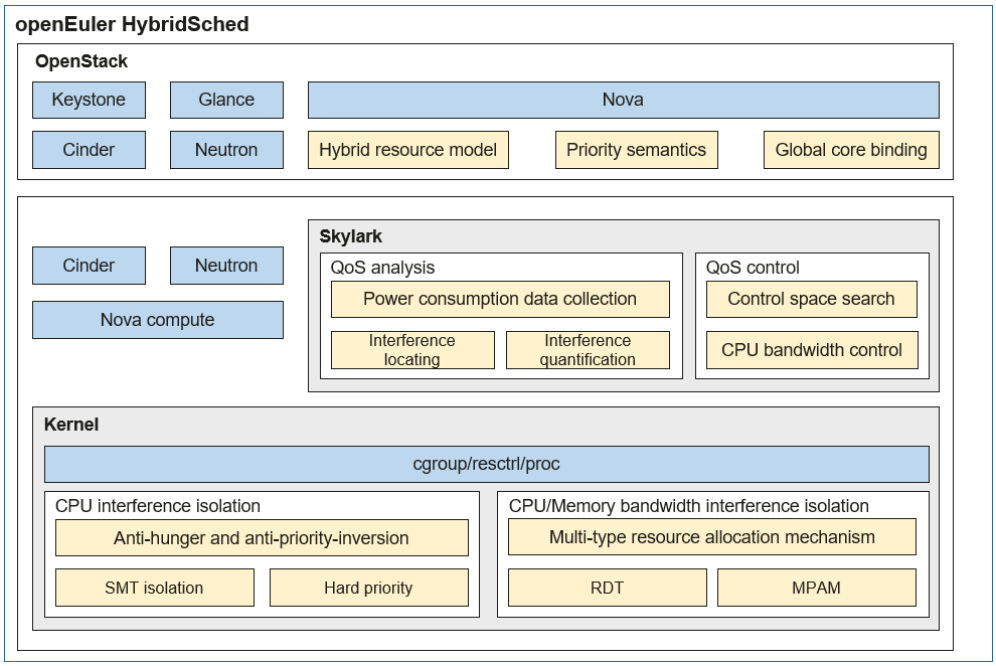
net improve IOPS throughput.

- **Northbound ecosystem**: The StratoVirt driver is available in libvirt, enriching the northbound software ecosystem.

HybridSched for Hybrid Virtualization Scheduling

Low resource utilization of cloud data centers is a common issue in the industry, and has fueled ways to improve this problem, such as deploying services based on priorities (hybrid deployment). The core technology of hybrid deployment is resource isolation and control. HybridSched is a full-stack solution for hybrid deployment of VMs, covering enhanced OpenStack cluster scheduling, the single-node QoS management component Skylark, and kernel-mode base resource isolation. In particular, Skylark is a QoS-aware resource scheduler used when high- and low-priority VMs are deployed together, improving physical machine resource utilization while ensuring the QoS of high-priority VMs.

Feature Description



- **Enhanced cluster scheduling**: Enhances OpenStack Nova to support priority-based semantic scheduling.
- **Power consumption control**: Limits the CPU bandwidth of low-priority VMs to reduce

the overall system power consumption and ensure the QoS of high-priority VMs.

- **Cache and memory bandwidth control**: Limits the LLC and memory bandwidth of low-priority VMs. Currently, only static allocation is supported.
- **CPU interference control**: Supports CPU time slice preemption in microseconds, SMT interference isolation, and anti-priority-inversion.

Application Scenarios

To improve resource utilization, services are classified into high- and low-priority services based on latency sensitivity, and deployed accordingly. Latency-sensitive services are recommended for high-priority VMs, such as web services, high-performance databases, real-time rendering, and machine learning and inference; while services not limited by latency can be used for low-priority VMs, such as video encoding, big data processing, offline rendering, and machine learning training.

7. Enhanced Features

Full-Stack Support for SM Cryptographic Algorithms

The openEuler OS now supports ShangMi (SM) cryptographic algorithms (SM2, SM3, and SM4) in key security features, and provides cryptographic services such as the SM cryptographic algorithm library, certificates, and secure transmission protocols for upper-layer applications.

Feature Description

SM cryptographic algorithms provide the following features:

- User-mode algorithm libraries, such as OpenSSL and Libgcrypt, support SM2, SM3, and SM4 algorithms.
- OpenSSH supports SM2, SM3, and SM4 algorithms.
- OpenSSL supports the Transport Layer Cryptography Protocol (TLCP) stack of the SM standards.
- SM3 and SM4 algorithms are supported for drive encryption (dm-crypt/cryptsetup).
- The SM3 algorithm is supported for password encryption in user identity authentication

(pam/libuser/shadow).

- The SM3 algorithm is supported for data digest in intrusion detection (AIDE).
- SM2, SM3, and SM4 algorithms are supported in the kernel cryptographic framework (crypto), allowing algorithm performance optimization using instruction sets such as AVX, CE, and NEON.
- The SM3 message digest algorithm and SM2 certificate are supported in Integrity Measurement Architecture and Extended Verification Module (IMA/EVM) of the kernel.
- The SM2 certificate is supported in kernel module signing and module signature verification.
- SM4-CBC and SM4-GCM algorithms are supported in Kernel Transport Layer Security (KTLS).
- SM3 and SM4 algorithms are supported in the Kunpeng Accelerator Engine (KAE).

Application Scenarios

The SM capabilities provided by openEuler safeguard applications running on openEuler. For example, the SM3 algorithm is used to encrypt service data based on the OpenSSL encryption API, and the SM3 or SM4 algorithm is used to encrypt drives in dm-crypt.

x2openEuler Porting Tool

x2openEuler is used to assess the feasibility and compatibility of porting from another OS to openEuler. It identifies compatibility risks in software, hardware, and configuration items through porting reports, and provides functions such as feasibility assessment, upgrade execution, and visualized batch porting for in-place upgrade to openEuler.

Feature Description

- **Software assessment**

Scans and assesses dependency packages (.rpm, .tar, .zip, .gzip, .jar, .py, .pyc, .sh, and .bin) for incompatibilities, and generates a report in HTML format.

- **Configuration collection and assessment**

Collects user environment data and generates JSON files. x2openEuler collects and assesses information about the systemd service, kernel parameters, and network and drive mounting configurations.

- **Hardware assessment**

Checks whether the system (x86/AArch64) and server boards (RAID/NIC/FC/IB/GPU/SSD/TPM) in the operating environment are in the openEuler compatibility list.

- **In-place upgrade**

Assesses whether the service software installed in the current environment can be upgraded to the openEuler version and upgrades the OS and services accordingly, and supports porting management and batch execution on a WebUI.

8. Copyright

9. Trademarks

any content in this document shall not be deemed as granting the permission or right to use any of the aforementioned trademarks and logos by implication, no objection, or other means. Without prior written consent, no one is allowed to use the name, trademark, or logo of the openEuler community in any form.

10. Appendixes

Appendix 1: Setting Up the Development Environment

Environment Preparation	URL
Downloading and installing openEuler	https://www.openeuler.org/en/download/
Preparing the development environment	https://gitee.com/openeuler/community/blob/master/en/contributors/prepare-environment.md
Building a software package	https://gitee.com/openeuler/community/blob/master/en/contributors/package-install.md

Appendix 2: Security Handling Process and Security Disclosure

Disclosure of Community Security Issues	URL
Security handling process	https://gitee.com/openeuler/security-committee/blob/master/security-process-en.md
Security disclosure	https://gitee.com/openeuler/security-committee/blob/master/security-disclosure-en.md