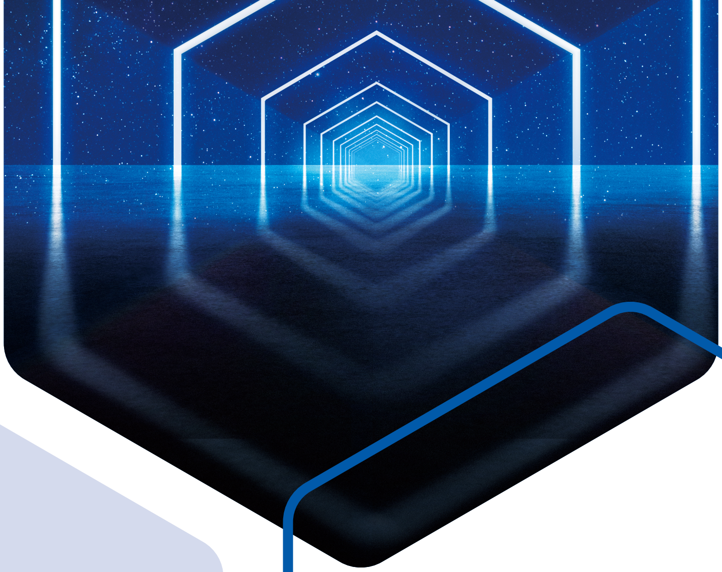
openEuler 24.09
Technical White Paper



CONTENTS

Introduction 01

01 Introduction

The openEuler open source community is incubated and operated by the OpenAtom Foundation.

openEuler is a digital infrastructure OS that fits into any server, cloud computing, edge computing, and embedded deployment. This secure, stable, and easy-to-use open source OS 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 to further enhance all-scenario innovations.

On December 30, 2022, openEuler 22.03 LTS SP1 was released, which is designed for hitless porting with best-of-breed tools.

On March 30, 2023, openEuler 23.03 was released. Running on Linux kernel 6.1, it streamlines technical readiness for Linux kernel 6.x and facilitates innovations for hardware adaptation and other technologies.

On June 30, 2023, openEuler 22.03 LTS SP2 was released, which enhances scenario-specific features and increases system performance to a higher level.

Later on September 30, 2023, the openEuler 23.09 innovation version was released based on Linux kernel 6.4. It provides a series of brand-new features to further enhance developer and user experience.

On November 30, 2023, openEuler 20.03 LTS SP4 was released, an enhanced extension of openEuler 20.03 LTS. openEuler 20.03 LTS SP4 provides excellent support for server, cloud native, and edge computing scenarios.

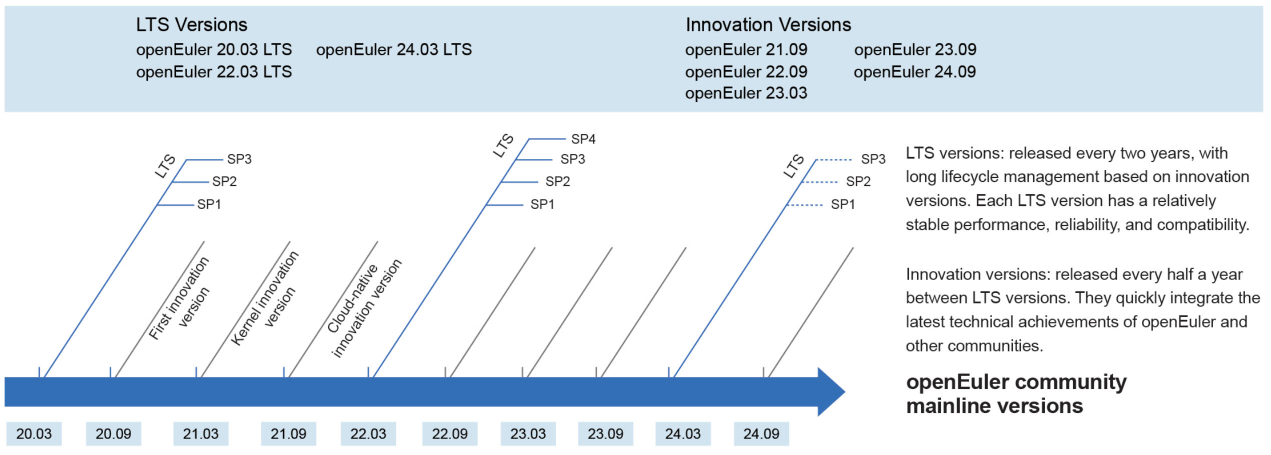
On December 30, 2023, openEuler 22.03 LTS SP3 was released. Designed to improve developer efficiency, it features for server, cloud native, edge computing, and embedded scenarios.

On May 30, 2024, openEuler 24.03 LTS was released. This version is built on Linux kernel 6.6 and brings new features for server, cloud, edge computing, AI, and embedded deployments to deliver enhanced developer and user experience.

On June 30, 2024, openEuler 22.03 LTS SP4 was released. Designed to improve developer efficiency, it further extends features for server, cloud native, edge computing, and embedded scenarios.

More recently, openEuler 24.09 was released on September 30, 2024. It is an innovation version built based on Linux kernel 6.6 and brings more advanced features and functions.

openEuler Version Management



LTS versions: released every two years, with long lifecycle management based on innovation versions. Each LTS version has a relatively stable performance, reliability, and compatibility.

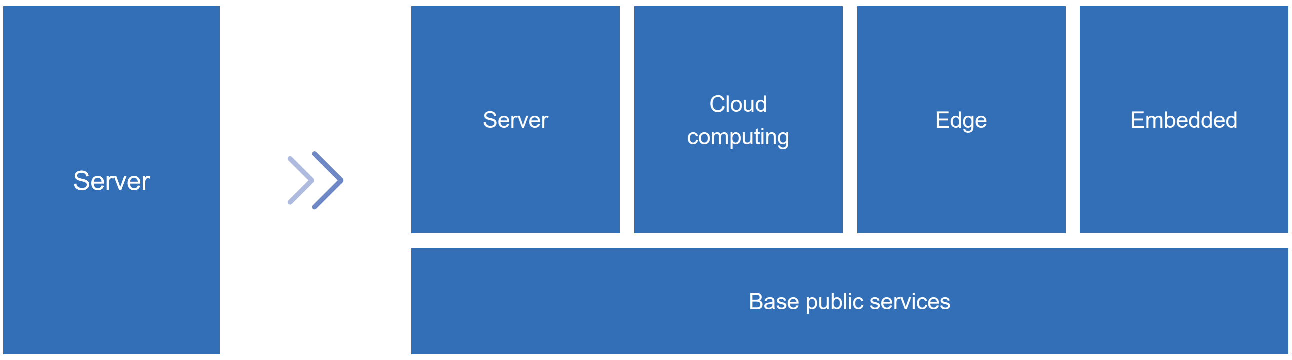
Innovation versions: released every half a year between LTS versions. They quickly integrate the latest technical achievements of openEuler and other communities.

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 innovation 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, RISC-V, LoongArch, and PowerPC), 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. openEuler is built to be used in any scenario, and comprises openEuler Edge and openEuler Embedded that are 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.

Open and Transparent: 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.

Platform Architecture 02

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 24.09 runs on Linux kernel 6.6 and provides POSIX-compliant 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 24.09 is equipped with a distributed soft bus and KubeEdge+ edge-cloud collaboration 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.

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

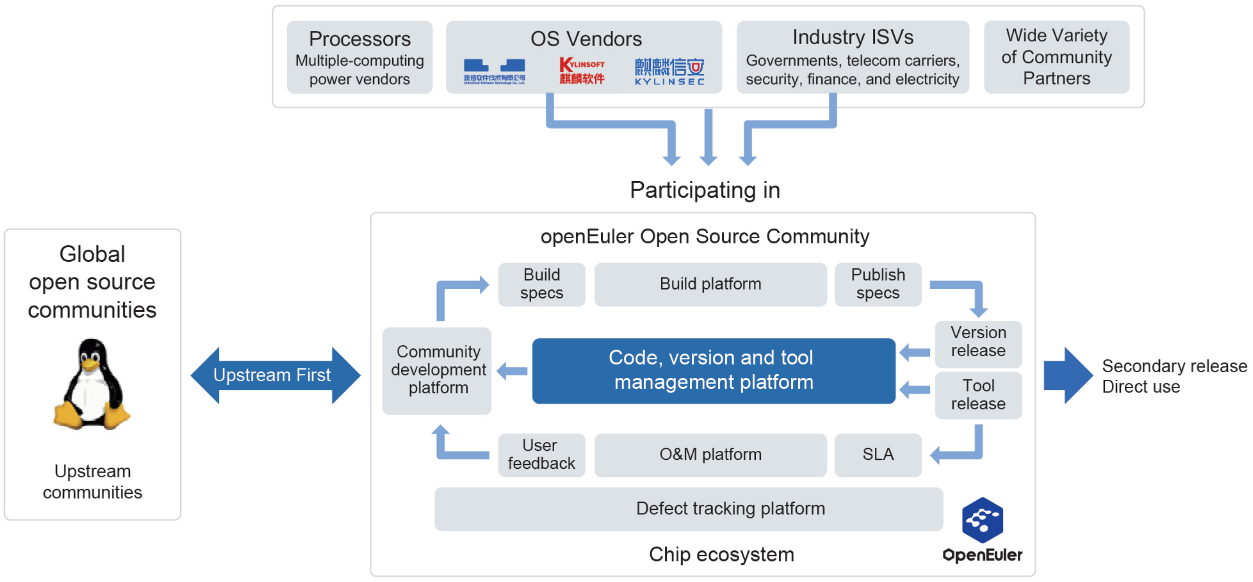
- **Edge computing**: openEuler 24.09 Edge is released for edge computing scenarios. It integrates the KubeEdge+ edge-cloud collaboration framework to provide unified management, provisioning of edge and cloud applications, and other capabilities.
- **Embedded**: openEuler 24.09 Embedded is released for embedded scenarios, helping compress images to under 5 MB and shorten the image loading time to under 5 seconds.
- **AI-native OS**: The OS enables AI software stacks with out-of-the-box availability. Heterogeneous convergence of memory, scheduling, and training/inference resources reduces AI development costs and improves efficiency. The intelligent interaction platform of the OS streamlines development and administration.

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

Visit https://www.openeuler.org/en/compatibility/ to see the full hardware list.

03 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, RISC-V
Memory	≥ 4 GB
Drive	≥ 20 GB

VMs

Verify VM compatibility when installing openEuler.

Hosts running on openEuler 24.09 support the following software packages:

- libvirt-9.10.0-12.oe2409
- libvirt-client-9.10.0-12.oe2409
- libvirt-daemon-9.10.0-12.oe2409
- qemu-8.2.0-17.oe2409
- qemu-img-8.2.0-17.oe2409

openEuler 24.09 is compatible with the following guest OSs for VMs:

Guest OS	Architecture
CentOS 6	x86_64
CentOS 7	AArch64
CentOS 7	x86_64
CentOS 8	AArch64
CentOS 8	x86_64
Windows Server 2016	AArch64
Windows Server 2016	x86_64
Windows Server 2019	AArch64
Windows Server 2019	x86_64

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

Embedded Devices

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

Item	Configuration Requirement
Architecture	AArch64, AArch32
Memory	≥ 512 MB
Drive	≥ 256 MB

Scenario-specific Innovations 04

AI

AI is redefining OSs by powering intelligent development, deployment, and O&M. openEuler supports general-purpose architectures like Arm, x86, and RISC-V, and next-gen AI processors like NVIDIA and Ascend. Further, openEuler is equipped with extensive AI capabilities that have made it a preferred choice for diversified computing power.

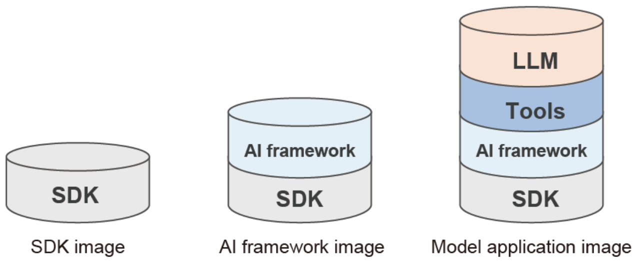
OS for AI

openEuler offers an efficient development and runtime environment that containerizes software stacks of AI platforms with out-of-the-box availability. It also provides various AI frameworks to facilitate AI development.

Feature Description

openEuler supports TensorFlow, PyTorch, and MindSpore frameworks and software development kits (SDKs) of major computing architectures, such as Compute Architecture for Neural Networks (CANN) and Compute Unified Architecture (CUDA), to make it easy to develop and run AI applications.

Environment setup is further simplified by containerizing software stacks. openEuler provides three types of container images:



- **SDK images**: Use openEuler as the base image and install the SDK of a computing architecture, for example, Ascend CANN and NVIDIA CUDA.
- **AI framework images**: Use an SDK image as the base and install AI framework software, such as PyTorch and TensorFlow. You can use an AI framework image to quickly build a distributed AI framework, such as Ray.
- **Model application images**: Provide a complete set of toolchains and model applications.

For details, see the openEuler AI Container Image User Guide.

Application Scenarios

openEuler uses AI container images to simplify deployment of runtime environments. You can select the container image that best suits your requirements and complete the deployment in a few simple steps.

- **SDK images**: You can develop and debug Ascend CANN or NVIDIA CUDA applications using an SDK image, which provides a compute acceleration toolkit and a development environment. These containers offer an easy way to perform high-performance computing (HPC) tasks, such as large-scale data processing and parallel computing.
- **AI framework images**: This type of containers is designed to support AI model development, training, and inference.
- **Model application images**: Such an image contains a complete AI software stack and purpose-built models for model inference and fine-tuning.

AI for OS

AI makes openEuler more intelligent. The openEuler Copilot System, an intelligent Q&A platform, is developed using foundation models and openEuler data. It assists in code generation, problem analysis, and system O&M.

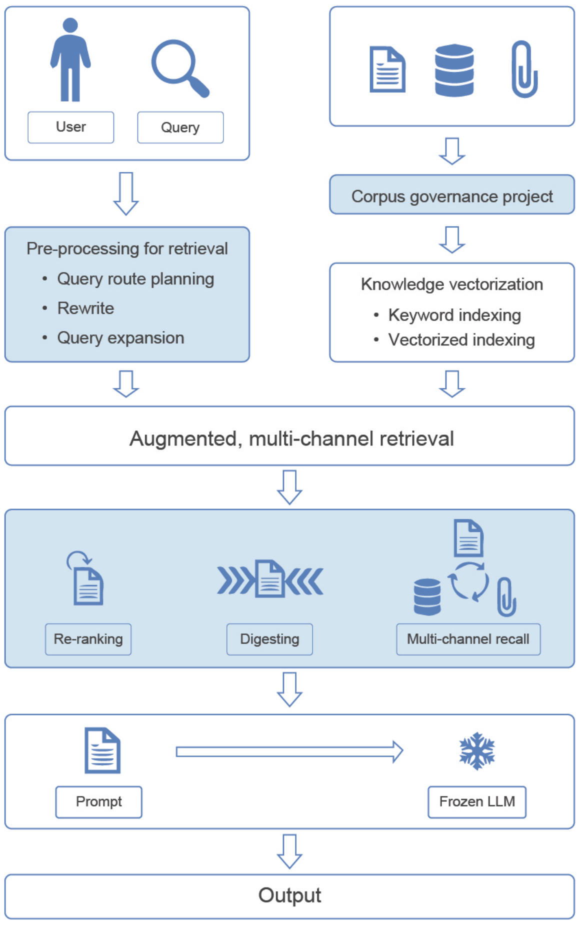
Intelligent Q&A

Feature Description

The openEuler Copilot System is accessible via web or shell.

- **Web**: Provides essential OS knowledge, openEuler data, and O&M, project, and usage information.
- **Shell**: Delivers intuitive user experience when accessed using natural languages.

RAG



Retrieval-augmented generation (RAG) is a technique for enhancing the long-term memory capability of large language models (LLMs). Used by the openEuler Copilot System, RAG is essential to reducing model training costs and has the following highlights:

- **Pre-processing for retrieval**: Query requests are allocated to the most appropriate information source through route planning for document retrieval. Then compound queries are disassembled to expand the retrieval coverage. After that, current queries are rewritten based on historical information to increase the retrieval hit rate.

- **Knowledge indexing**: In the corpus governance project, fragments are extracted from those documents to be stored into the knowledge base. Then derivatives are generated for complicated fragments like code snippets and optical character recognition (OCR)-generated text, replacing fragments in subsequent retrieval. After that, based on the fragments and fragment derivatives, keywords are extracted and vector features are generated to construct data structures for retrieval in multi-channel recall.
- **Multi-channel recall**: For queries that have been rewritten during pre-processing, target fragments are retrieved using vectorized retrieval, keyword retrieval, and Chat2DB. Then the retrieval results are re-ranked using model capabilities and the BM25 method, where irrelevant fragments are filtered out. Finally, digesting, polishing, and other techniques are employed to process the key content of the retrieval results, compress the token length, and enhance natural language features.

These capabilities enable RAG for the openEuler Copilot System to accommodate to more document formats and content, and enhance Q&A services without increasing system load.

Corpus Governance

Corpus governance is one of the basic RAG capabilities in the openEuler Copilot System. It imports corpuses into the knowledge base in a supported format using fragment relationship extraction, fragment derivative construction, and OCR, increasing the retrieval hit rate.

- **Fragment relationship extraction**: It represents the abstract relationship between fragments and between fragments and their derivatives. After a user's query hits the target fragment, the system retrieves related fragments (such as context) based on the fragment relationship, enhancing the integrity of the final retrieval result.
- **Fragment derivative generation**: For complicated fragments (such as code snippets), digests, descriptions, and case questions (raised by users about the fragments) are generated to replace these complicated fragments in searches. This significantly alleviates impact of complicated fragments in the retrieval process and enhances the accuracy and integrity of the retrieval result.
- **OCR**: Images in a document are extracted as fragment derivatives using image-to-text conversion to ensure that queries hit fragment derivatives associated with the original images. OCR greatly compensates for retrieval results that contain no images.

The preceding corpus governance methods can enhance the Q&A service experience in multi-round dialogs, content integrity, and image-text display.

Application Scenarios

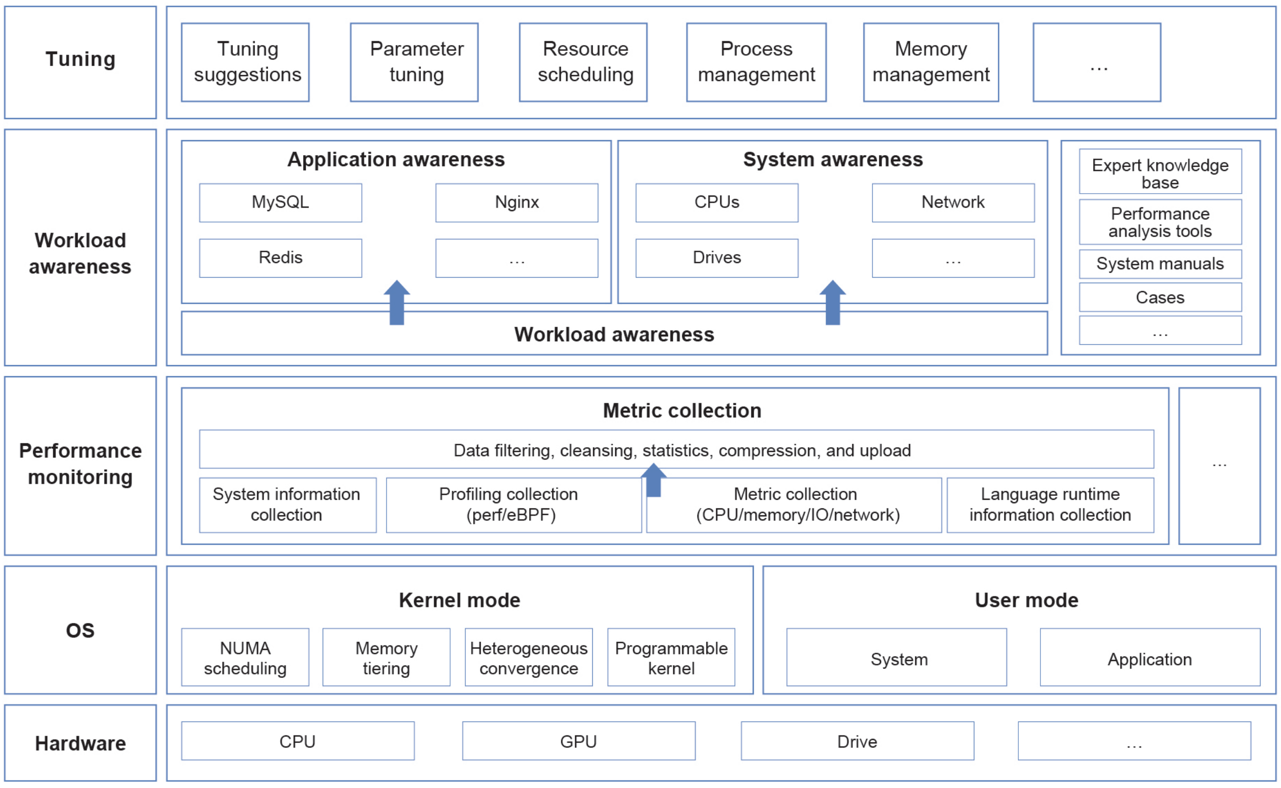
- **Common users** who are seeking best practices, such as porting applications to openEuler.
- **Developers** who want to learn contribution processes, key features, project development, and other extensive knowledge of openEuler.
- **O&M personnel** who are using the Q&A system to solve common problems and improve system management.

For details, see the openEuler Copilot System Intelligent Q&A Service User Guide.

Intelligent Tuning

Feature Description

The openEuler Copilot System supports the intelligent shell entry. Through this entry, you can interact with the openEuler Copilot System using a natural language and perform heuristic tuning operations such as performance data collection, system performance analysis, and system performance tuning.

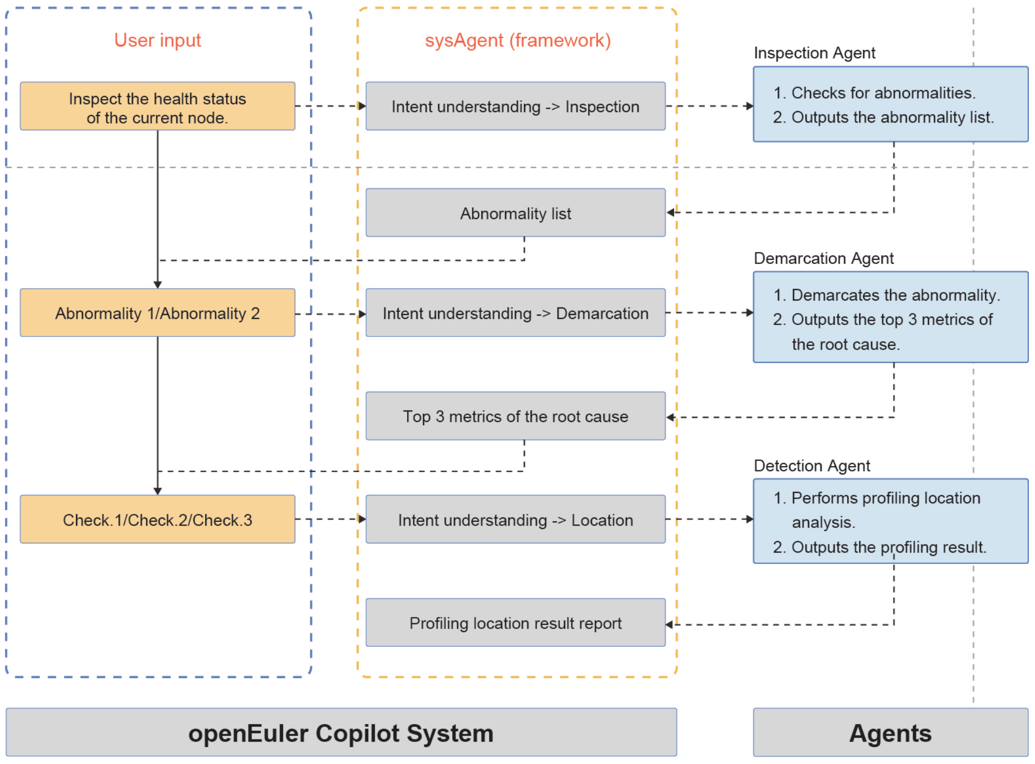


Application Scenarios

- **Gaining insights from key performance metrics**: You can learn about the system performance status based on collected performance metrics like CPU, I/O, drive, network, and application.
- **Analyzing system performance**: Performance analysis reports are generated, making it easier to locate performance bottlenecks across the entire system and in individual applications.
- **Receiving performance tuning suggestions**: The openEuler Copilot System generates a performance tuning script, which can be executed with one click to tune the system and specific applications.

Intelligent Diagnosis

Feature Description


User input
sysAgent (framework)
Inspection Agent
Inspect the health status of the current node.
Intent understanding -> Inspection
1. Checks for abnormalities.
2. Outputs the abnormality list.
Abnormality list
Demarcation Agent
Abnormality 1/Abnormality 2
Intent understanding -> Demarcation
1. Demarcates the abnormality.
2. Outputs the top 3 metrics of the root cause.
Top 3 metrics of the root cause
Detection Agent
Check.1/Check.2/Check.3
Intent understanding -> Location
1. Performs profiling location analysis.
2. Outputs the profiling result.
Profiling location result report
openEuler Copilot System
Agents

- **Inspection**: The Inspection Agent checks for abnormalities of designated IP addresses and provides an abnormality list that contains associated container IDs and abnormal metrics (such as CPU and memory).
- **Demarcation**: The Demarcation Agent analyzes and demarcates a specified abnormality contained in the inspection result and outputs the top 3 metrics of the root cause.
- **Location**: The Detection Agent performs profiling location analysis on the root cause, and provides useful hotspot information such as the stack, system time, and performance metrics related to the root cause.

Application Scenarios

In openEuler 24.09, the intelligent shell entry enables capabilities like single-node abnormality inspection, demarcation, and profiling location.

- The inspection capabilities refer to single-node performance metric collection, performance analysis, and abnormality inspection.
- The demarcation capability is to locate the root cause based on the abnormality inspection result and output the top 3 metrics of the root cause.
- The profiling location capability refers to using a profiling tool to locate the faulty modules (code snippets) based on the root cause.

Intelligent Deployment

Feature Description

The openEuler Copilot System can invoke environment resources through a natural language, assist in pulling container images for local physical resources, and establish a development environment suitable for debugging on existing compute devices.

This system supports three types of containers, and container images have been released on Docker Hub. You can manually pull and run these container images.

- **SDK layer**: encapsulates only the component libraries that enable AI hardware resources, such as CUDA and CANN.
- **SDKs + training/inference frameworks**: accommodates TensorFlow, PyTorch, and other frameworks (for example, tensorflow2.15.0-cuda12.2.0 and pytorch2.1.0.a1-cann7.0.RC1) in addition to the SDK layer.
- **SDKs + training/inference frameworks + LLMs**: encapsulates several models (for example, llama2-7b and chatglm2-13b) based on the second type of containers.

The following table lists the container images supported by the openEuler Copilot System:

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

Application Scenarios

- **Common openEuler operations**: Simplify the process of building a deep learning development environment while saving physical resources. For example, set up an Ascend development environment that runs on openEuler.
- **openEuler development**: Developers familiarize themselves with the openEuler AI software stack to reduce the trial-and-error cost of installing components.

openEuler Embedded

openEuler 24.09 Embedded is designed for embedded applications, offering significant progress in southbound and northbound ecosystems, technical features, infrastructure, and implementation over previous generations.

openEuler Embedded provides a closed loop framework often found in operational technology (OT) applications such as manufacturing and robotics, whereby innovations help optimize its embedded system software stack and ecosystem.

openEuler 24.09 Embedded supports Linux kernels 5.10 and 6.6, and users can select either Linux kernel or customize a kernel to better fulfill their board support package (BSP) and application requirements.

openEuler 24.09 Embedded is equipped with an embedded virtualization base that is available as multiple solutions, including Jailhouse partitioning-based virtualization, OpenAMP bare metal hybrid deployment, and Zephyr-based Virtual Machine (ZVM) & Rust-Shyper real-time virtualization. You can select the most appropriate solution to suit your services. The mixed-criticality (MICA) deployment framework is built on the embedded virtualization base to mask discrepancies between different bases and provides a unified set of APIs for different runtimes.

The MICA deployment framework supports over 600 northbound software packages, including the ROS Humble version. ROS Humble contains ros-core, ros-base, and SLAM software packages to implement the ROS 2 runtime. SDKs with embedded features of ROS 2 are provided for ROS 2 colcon cross-compilation. openEuler 24.09 Embedded is already included in the BMC ecosystem and has been adapted to openBMC.

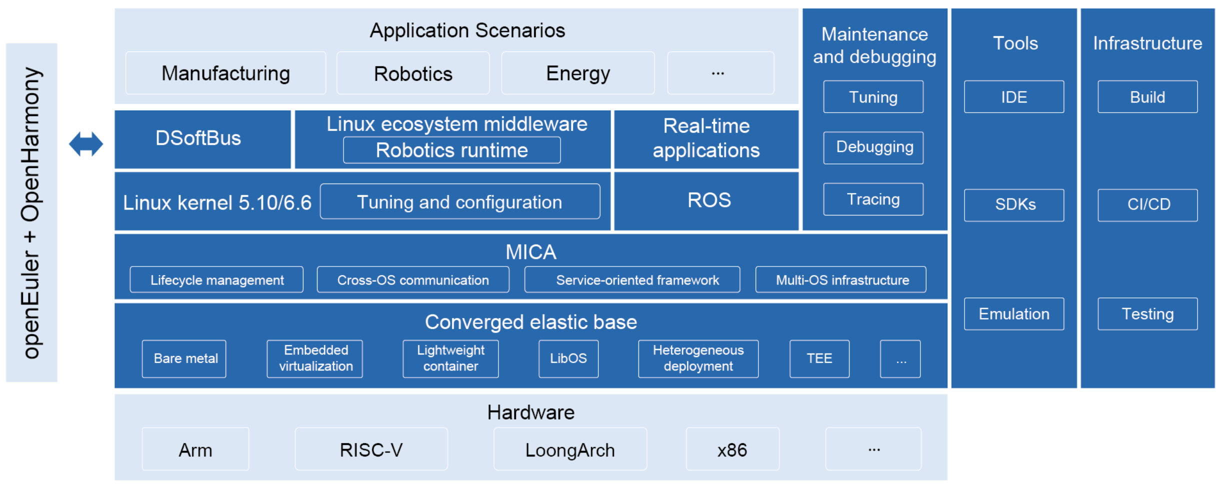
The MICA deployment framework supports a variety of southbound hardware, including Phytium, HiSilicon, Renesas, TI, and Allwinner, and also EulerPi (hardware development board designed for developers) and HiEuler Pi (native development board for openEuler Embedded).

Infrastructure tools available on openEuler 24.09 Embedded include the meta-tool oebuild and build tool Yocto 4.0 LTS. In addition, the LLVM toolchain is introduced to help better build images and generate SDKs. Compared with the GCC toolchain, LLVM has its advantages in performance, size, and security.

openEuler Embedded has multiple commercial and enterprise editions, which have been applied in BMC, industrial controller, robot controller, and other fields.

Future versions of openEuler Embedded will integrate contributions from ecosystem partners, users, and community developers, increase support for chip architectures such as LoongArch and more southbound hardware, and optimize industrial middleware, embedded AI, embedded edge, and simulation system capabilities.

System Architecture



Southbound Ecosystem

openEuler Embedded Linux supports mainstream processor architectures like AArch64, x86_64, AArch32, and RISC-V, and will extend support to LoongArch in the future. Since its 24.03 version, openEuler has a richer southbound ecosystem and supports chips from Raspberry Pi, HiSilicon, Rockchip, Renesas, TI, Phytium, StarFive, and Allwinner.

Embedded Virtualization Base

openEuler Embedded uses an elastic virtualization base that enables multiple OSs to run on a system-on-a-chip (SoC). The base incorporates a series of technologies including bare metal, embedded virtualization, lightweight containers, LibOS, trusted execution environment (TEE), and heterogeneous deployment.

- The bare metal hybrid deployment solution runs on OpenAMP to manage peripherals by partition at a high performance level; however, it delivers poor isolation and flexibility. This solution supports the hybrid deployment of UniProton/Zephyr/RT-Thread and openEuler Embedded Linux.
- Partitioning-based virtualization is an industrial-grade hardware partition virtualization solution that runs on Jailhouse. It offers superior performance and isolation but inferior flexibility. This solution supports the hybrid deployment of UniProton/Zephyr/FreeRTOS and openEuler Embedded Linux or of OpenHarmony and openEuler Embedded Linux.
- Real-time virtualization is available as two community hypervisors, ZVM (for real-time VM monitoring) and Rust-Shyper (for Type-I embedded VM monitoring).

MICA Deployment Framework

The MICA deployment framework is a unified environment that masks the differences between technologies that comprise the embedded elastic virtualization base. The multi-core capability of hardware combines the universal Linux OS and a dedicated real-time operating system (RTOS) to make full use of all OSs.

The MICA deployment framework covers lifecycle management, cross-OS communication, service-oriented framework, and multi-OS infrastructure.

- Lifecycle management provides operations to load, start, suspend, and stop the client OS.
- Cross-OS communication uses a set of communication mechanisms between different OSs based on shared memory.
- Service-oriented framework enables different OSs to provide their own services. For example, Linux provides common file system and network services, and the RTOS provides real-time control and computing.
- Multi-OS infrastructure integrates OSs through a series of mechanisms, covering resource expression and allocation and unified build.

The MICA deployment framework provides the following functions:

- Lifecycle management and cross-OS communication for openEuler Embedded Linux and the RTOS (Zephyr or UniProton) in bare metal mode
- Lifecycle management and cross-OS communication for openEuler Embedded Linux and the RTOS (FreeRTOS) in partitioning-based virtualization mode

Northbound Ecosystem

- **Northbound software packages**: Over 600 common embedded software packages can be built using openEuler.
- **Soft real-time kernel**: This capability helps respond to soft real-time interrupts within microseconds.
- **DSoftBus**: The distributed soft bus system (DSoftBus) of openEuler 24.09 Embedded integrates the DSoftBus and point-to-point authentication module of OpenHarmony. It implements interconnection between openEuler-based embedded devices and OpenHarmony-based devices as well as between openEuler-based embedded devices.

- **Embedded containers and edges**: With iSula containers, openEuler and other OS containers can be deployed on embedded devices to simplify application porting and deployment. Embedded container images can be compressed to 5 MB, and can be easily deployed into the OS on another container. openEuler 24.09 Embedded supports KubeEdge to streamline cloud-edge-device collaboration.

UniProton

UniProton is an RTOS that features ultra-low latency and flexible MICA deployments. It is suited for industrial control because it supports both microcontroller units and multi-core CPUs. UniProton provides the following capabilities:

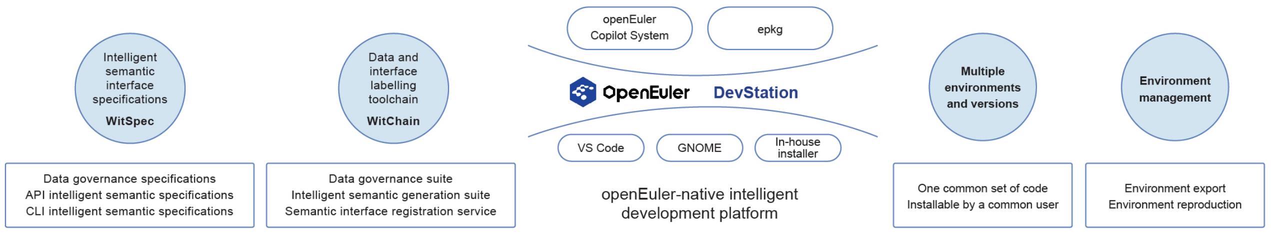
- Compatible with processor architectures like Cortex-M, AArch64, x86_64, and riscv64, and supports M4, RK3568, RK3588, x86_64, Hi3093, Raspberry Pi 4B, Kunpeng 920, Ascend 310, and Allwinner D1s.
- Connects with openEuler Embedded Linux on Raspberry Pi 4B, Hi3093, RK3588, and x86_64 devices in bare metal mode.
- Can be debugged using GDB on openEuler Embedded Linux.
- Over 890 POSIX APIs available for file systems, device management, shell consoles, and networks.

Application Scenarios

openEuler Embedded helps supercharge computing performance in a wide range of industries and fields, including industrial and power control, robotics, aerospace, automobiles, and healthcare.

DevStation

DevStation is a Linux desktop distribution introduced by openEuler that streamlines software development and deployment. Pre-installed with a host of development tools and IDEs, DevStation supports multiple programming languages and integrates container technologies (such as Docker and iSula) to facilitate test environment setup. It also equips built-in AI frameworks such as TensorFlow and PyTorch, with optimized hardware accelerator support. The distribution uses an easy-to-use GUI and other tools for debugging and automated testing, giving all-round support for both entry-level and senior developers.


Intelligent semantic interface specifications
WitSpec
Data governance specifications
API intelligent semantic specifications
CLI intelligent semantic specifications
Data and interface labelling toolchain
WitChain
Data governance suite
Intelligent semantic generation suite
Semantic interface registration service
openEuler Copilot System
epkg
OpenEuler DevStation
VS Code
GNOME
In-house installer
openEuler-native intelligent development platform
Multiple environments and versions
One common set of code
Installable by a common user
Environment management
Environment export
Environment reproduction

Feature Description

Developer-friendly integrated environment: supports multiple programming languages and comes with pre-installed development tools and IDEs such as VS Code to streamline front-end, back-end, and full-stack development.

Software package management and auto-deployment: provides convenient software package management tools, which allow you to complete installations with a few clicks and update development environments with ease. The built-in container technologies such as Docker and iSula facilitate application containerization and automatic deployment, and the epkg package manager is available for version deployment to simplify the use of various development tools.

GUI-based programming: realizes intuitive programming.

AI development: provides a complete AI model development and training environment, which is pre-installed with machine learning frameworks such as TensorFlow and PyTorch and is readily compatible with hardware accelerators including GPUs and NPUs. DevStation also integrates the openEuler Copilot System to provide AI assistance in solving most OS problems.

Debugging and testing: provides built-in debugging tools, such as GDB, CUnit, GTest, and perf, and test and tuning tools for fast debugging and automated testing.

Versioning and collaboration: integrates versioning tools such as Git and SVN and remote collaboration tools such as Slack, Mattermost, and GitLab, promoting team development and remote collaboration.

Security and compliance check: offers security scanning and code compliance check tools to help developers detect and fix potential security vulnerabilities when coding.

Application Scenarios

Multi-language development: DevStation is suitable for projects employing multiple languages such as Python, JavaScript, Java, and C++. With various pre-installed compilers, interpreters, and build tools, developers do not need to manually configure the environment.

Quick deployment and testing: DevStation provides virtualization support through built-in Docker and iSula. Developers can quickly build and deploy applications, and automate local tests across multiple environments. DevStation will integrate the openEuler deployment tool to streamline development environment installations.

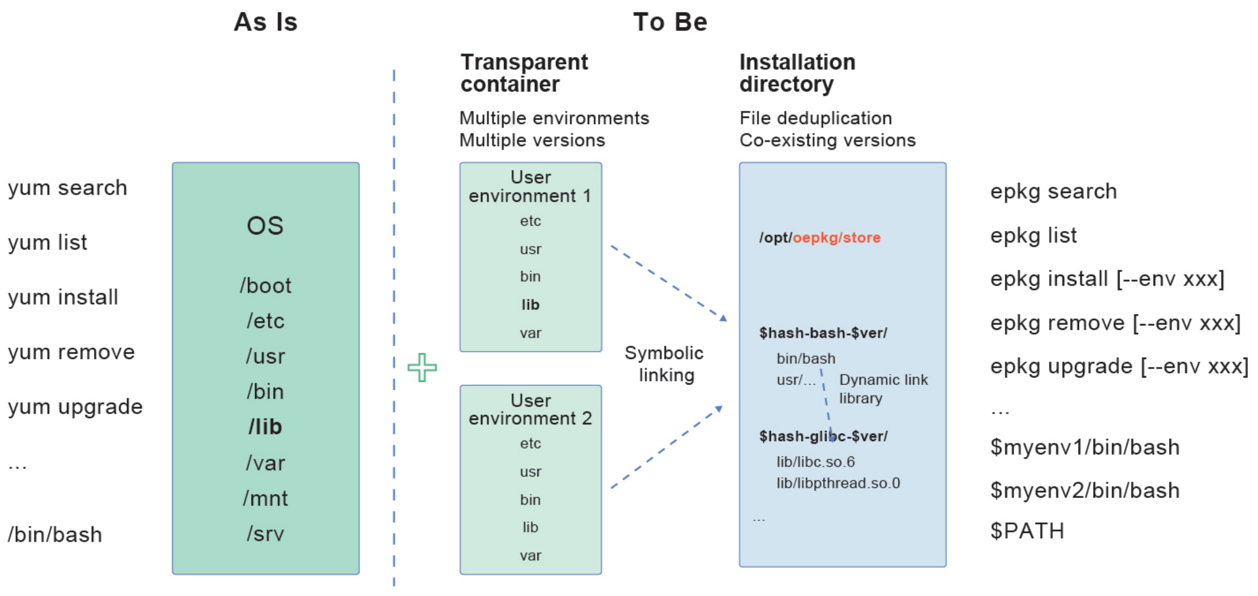
AI development and data science: Pre-installed databases, model training tools, and heterogeneous computing accelerators combine to create a complete AI development environment for model training and inference.

Game and multimedia development: DevStation comes with multimedia editing software to support image, audio, and video processing.

Programming training: DevStation is suitable for both entry-level and senior developers, as it provides built-in development tutorials and intelligent Q&A services using the openEuler Copilot System.

epkg

epkg is a new software package manager that supports the installation and use of non-service software packages. It solves version compatibility issues so that users can install and run software of different versions on the same OS by using simple commands to create, enable, and switch between environments.



Feature Description

Version compatibility: enables multiple versions of the same software package to run on the same node without version conflicts.

Environment management: allows users to create, enable, and switch between environments. Users can use channels of different environments to use software packages of different versions. When an environment is switched to another, the software package version is also changed.

Installation by common users: allows common users to install software packages, create environments, and manage their environment images, reducing security risks associated with software package installation.

Application Scenarios

epkg solves compatibility issues in installing multiple versions of the same software package. Users only need to switch between environments to use different package versions.

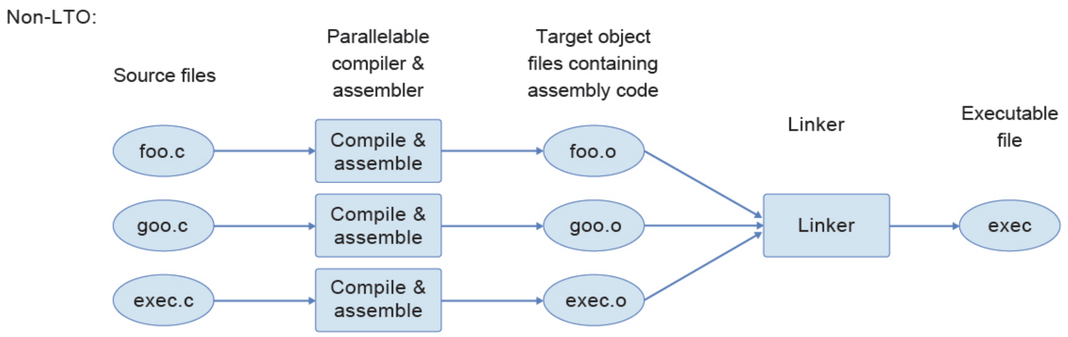
For details, refer to the *epkg User Guide*.

LTO for openEuler

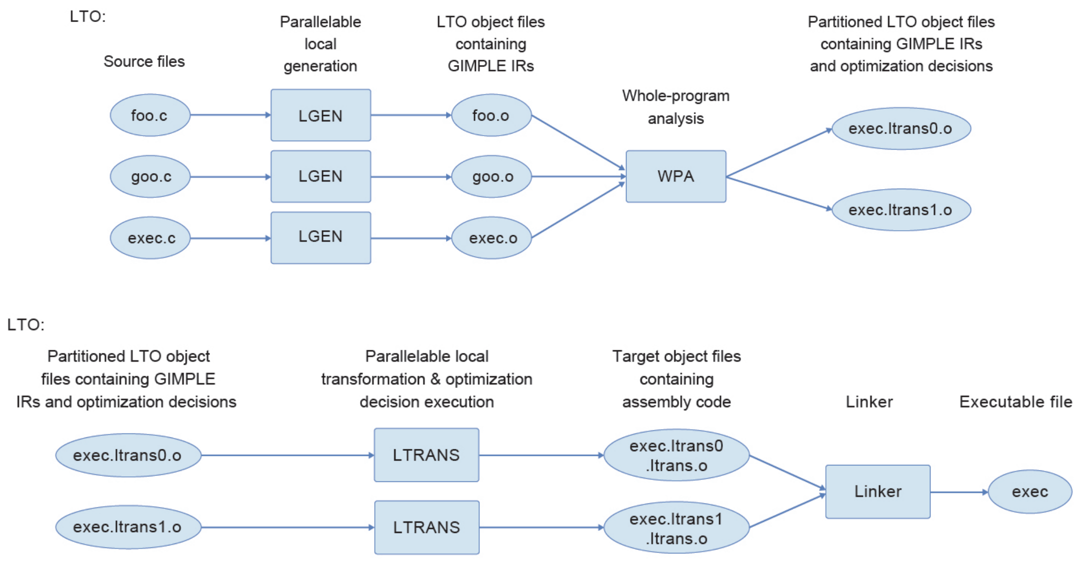
Link-time optimization (LTO) is a compilation technique that reschedules compilation optimization to the link stage. Compared with single-file compilation optimization during the compile stage, this technique makes more aggressive optimization decisions to offer more optimizations, smaller binary size, and higher performance.

Feature Description

In a conventional compilation process, GCC directly compiles and optimizes a single source file (also called single compilation unit or SCU) to generate a target object file (.o file) that contains assembly code. The linker then parses and relocates symbols in the symbol table of the .o file to generate an executable file. In this process, the linker, although having cross-file function call information, processes assembly code and thus cannot perform compilation optimization. Other stages that allow compilation optimization do not have such cross-file global information. This compilation framework compiles only the small number of modified compilation units, which improves efficiency but misses several opportunities for cross-file optimization.



The LTO design highlights opportunities for optimization at the link stage when cross-compilation-unit function call information is available. To achieve this, LTO retains Intermediate Representation (IR) information required for compilation optimization to the link stage. The linker then calls the LTO plugin to perform whole-program analysis and generate optimization decisions. After that, compilation optimization is performed to generate more efficient IRs. The IRs are then converted into target object files that contain assembly code, and the linker parses object files to generate an executable file.


LTO:
Source files
Parallelable local generation
LTO object files containing GIMPLE IRs
Whole-program analysis
Partitioned LTO object files containing GIMPLE IRs and optimization decisions
foo.c
goo.c
exec.c
LGEN
LGEN
LGEN
foo.o
goo.o
exec.o
WPA
exec.ltrans0.o
exec.ltrans1.o
LTO:
Partitioned LTO object files containing GIMPLE IRs and optimization decisions
Parallelable local transformation & optimization decision execution
Target object files containing assembly code
Linker
Executable file
exec.ltrans0.o
exec.ltrans1.o
LTRANS
LTRANS
exec.ltrans0 .ltrans.o
exec.ltrans1 .ltrans.o
Linker
exec

Application Scenarios

In openEuler 24.09, the default compilation options in **openEuler-rpm-config** are modified to enable LTO for 532 applications during software build using the trustlist mechanism. In this way, products generated by the linker, such as executable files and dynamic libraries, shrink by 300 MB, accounting for 14% of the 532 applications' linkage product size.

Applications for which LTO is enabled can be found in the LTO patch.

openEuler GCC Toolset 14

openEuler GCC Toolset 14 offers more flexible and efficient compilation environments. Users can easily switch between different GCC versions to access new hardware features and improved performance brought by latest GCC optimizations.

Feature Description

To enable new compute features and make the most of hardware features, openEuler GCC Toolset 14 offers a minor GCC 14 toolchain that is later than the major GCC of openEuler 24.09, enabling users to select the most appropriate compilation environment. By using openEuler GCC Toolset 14, users can easily switch between different GCC versions to use new hardware features.

To decouple from the default major GCC and prevent conflicts between the dependency libraries of the minor and major GCC versions, the software package of the openEuler GCC Toolset 14 toolchain is named starting with **gcc-toolset-14-**, followed by the name of the original GCC software package.

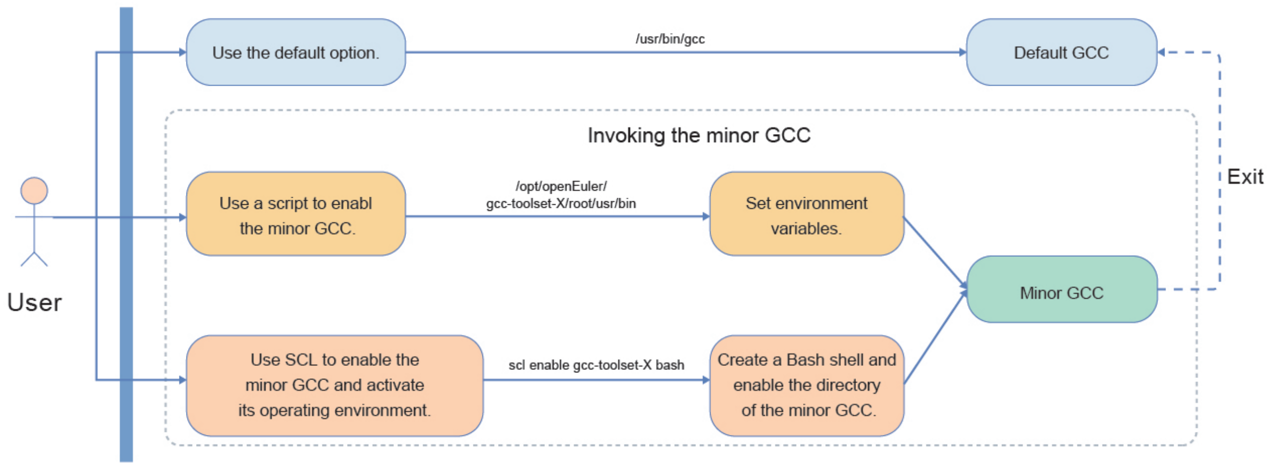
For version management, this solution introduces the SCL tool that provides an **enable** script in the /**opt/openEuler/gcc-toolset-14** directory to register the environment variables of openEuler GCC Toolset 14 with SCL. Then, users can use the tool to start a new Bash shell that uses the environment variables of the minor version configured in the **enable** script. In this way, it is easy to switch between the major and minor GCC versions.

Application Scenarios

Major GCC: The default GCC 12.3.1 is used for compilation.

Minor GCC: If the features of GCC 14 are required to build applications, use SCL to switch to the build environment of openEuler GCC Toolset 14.

The following figure shows the method of switching between the major and minor GCC.


Use the default option.
/usr/bin/gcc
Default GCC
Invoking the minor GCC
Exit
User
Use a script to enabl the minor GCC.
/opt/openEuler/ gcc-toolset-X/root/usr/bin
Set environment variables.
Minor GCC
Use SCL to enable the minor GCC and activate its operating environment.
scl enable gcc-toolset-X bash
Create a Bash shell and enable the directory of the minor GCC.

05 Kernel Innovations

openEuler 24.09 runs on Linux kernel 6.6 and inherits the competitive advantages of community versions and innovative features released in the openEuler community.

- **Folio-based memory management**: Folio-based Linux memory management is used instead of page. A folio consists of one or more pages and is declared in **struct folio**. Folio-based memory management is performed on one or more complete pages, rather than on *PAGE_SIZE* bytes. This alleviates compound page conversion and tail page misoperations, while decreasing the number of least recently used (LRU) linked lists and optimizing memory reclamation. It allocates more continuous memory on a per-operation basis to reduce the number of page faults and mitigate memory fragmentation. Folio-based management accelerates large I/Os and improves throughput, and large folios consisting of anonymous pages or file pages are available. For AArch64 systems, a contiguous bit (16 contiguous page table entries are cached in a single entry within a translation lookaside buffer, or TLB) is provided to reduce system TLB misses and improve system performance. In openEuler 24.09, multi-size transparent hugepage (mTHP) allocation by anonymous shmem and mTHP lazyfreeing are available. The memory subsystem supports large folios, with a new sysfs control interface for allocating mTHPs by page cache and a system-level switch for feature toggling.
- **Multipath TCP (MPTCP)**: MPTCP is introduced to let applications use multiple network paths for parallel data transmission, compared with single-path transmission over TCP. This design improves network hardware resource utilization and intelligently allocates traffic to different transmission paths, thereby relieving network congestion and improving throughput.

  MPTCP features the following performance highlights:

  » Selects the optimal path after evaluating indicators such as latency and bandwidth.

  » Ensures hitless network switchover and uninterrupted data transmission when switching between networks.

  » Uses multiple channels where data packets are distributed to implement parallel transmission, increasing network bandwidth.

  In the lab environment, the Rsync file transfer tool that adopts MPTCP v1 shows good transmission efficiency improvement. Specifically, a 1.3 GB file can be transferred in just 14.35s (down from 114.83s), and the average transfer speed is increased from 11.08 MB/s to 88.25 MB/s. In simulations of path failure caused by unexpected faults during transmission, MPTCP seamlessly switches data to other available channels, ensuring transmission continuity and data integrity.

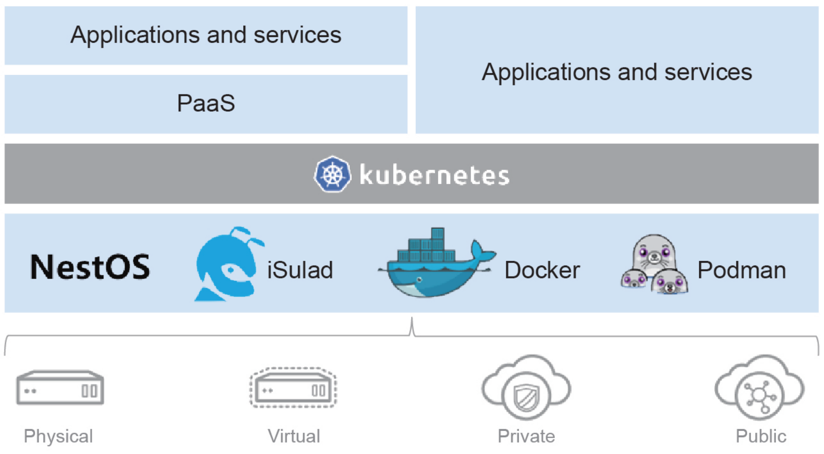
  In openEuler 24.09, MPTCP-related features in Linux mainline kernel 6.9 have been fully transplanted and optimized.
- **Large folio for ext4 file systems**: The IOzone performance can be improved by 80%, and the writeback process of the iomap framework supports batch block mapping. Blocks can be requested in batches in default ext4, optimizing ext4 performance in various benchmarks. For ext4 buffer I/O and page cache writeback operations, the buffer_head framework is replaced with the iomap framework that adds large folio support for ext4. In version 24.09, the performance of small buffered I/Os (≤ 4 KB) is optimized when the block size is smaller than the folio size, typically seeing a 20% performance increase.
- **CacheFiles failover**: In on-demand mode of CacheFiles, if the daemon breaks down or is killed, subsequent read and mount requests return **-EIO**. The mount points can be used only after the daemon is restarted and the mount operations are performed again. For public cloud services, such I/O errors will be passed to cloud service users, which may impact job execution and endanger the overall system stability. The CacheFiles failover feature renders it unnecessary to remount the mount points upon daemon crashes. It requires only the daemon to restart, ensuring that these events are invisible to users.
- **PGO for Clang**: Profile-guided optimization (PGO) is a feedback-directed compiler optimization technology that collects program runtime information to guide the compiler through optimization decision-making. Based on industry experience, PGO can be used to optimize large-scale data center applications (such as MySQL, Nginx, and Redis) and Linux kernels. Test results show that LLVM PGO provides over 20% performance increase on Nginx, in which a 10%+ performance increase is brought by kernel optimizations.

06 Cloud Base

NestOS

NestOS is a community cloud OS that uses nestos-assembler for quick integration and build. It runs rpm-ostree and Ignition tools over a dual rootfs and atomic update design, and enables easy cluster setup in large-scale containerized environments. Compatible with Kubernetes and OpenStack, NestOS also reduces container overheads.

Feature Description



- **Out-of-the-box availability**: integrates popular container engines such as iSulad, Docker, and Podman to provide lightweight and tailored OSs for the cloud.
- **Easy configuration**: uses the Ignition utility to install and configure a large number of cluster nodes with a single configuration.
- **Secure management**: runs rpm-ostree to manage software packages and works with the openEuler software package source to ensure secure and stable atomic updates.
- **Hitless node updating**: uses Zincati to provide automatic node updates and reboot without interrupting services.
- **Dual rootfs**: executes dual rootfs for active/standby switchovers, to ensure integrity and security during system running.

Application Scenarios

openEuler introduces the NestOS-Kubernetes-Deployer tool to resolve problems such as inconsistent and repeated O&M operations across stacks and platforms. These problems are typically caused by decoupling of containers from underlying environments when using container and container orchestration technologies for rollout and O&M. NestOS is developed for containerized cloud applications and ensures consistency between services and the base OS.

07 Enhanced Features

SysCare

SysCare is a system-level hotfix software that provides security patches and hot fixing for OSs. It can fix system errors without restarting hosts. SysCare combines kernel-mode and user-mode hot patching to take over system repair, freeing up valuable time for users to focus on core services. Looking ahead, SysCare will introduce live OS update capabilities, further improving O&M efficiency.

Feature Description

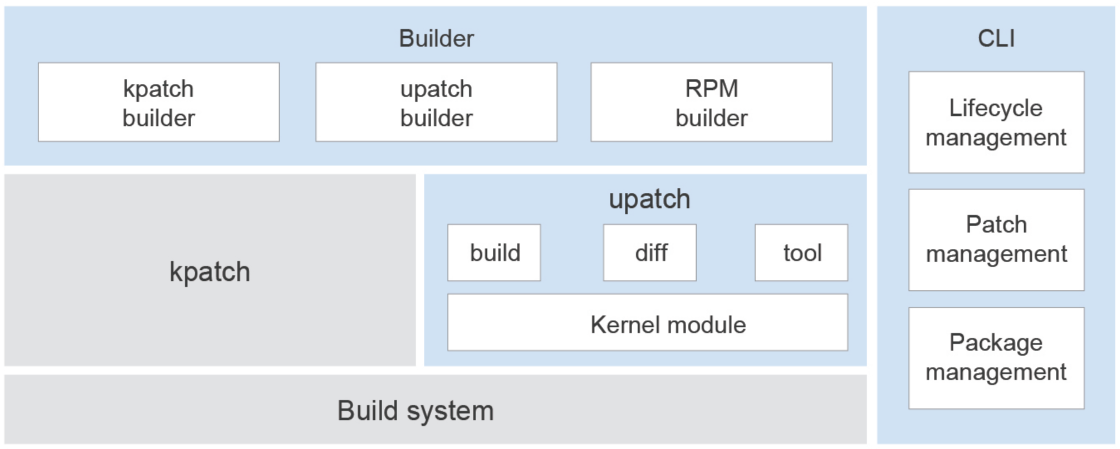
- **Hot patch creation**

  Hot patch RPM packages can be generated by providing the paths of the source RPM package, debuginfo RPM package, and patches to be applied, eliminating the need to modify the software source code.

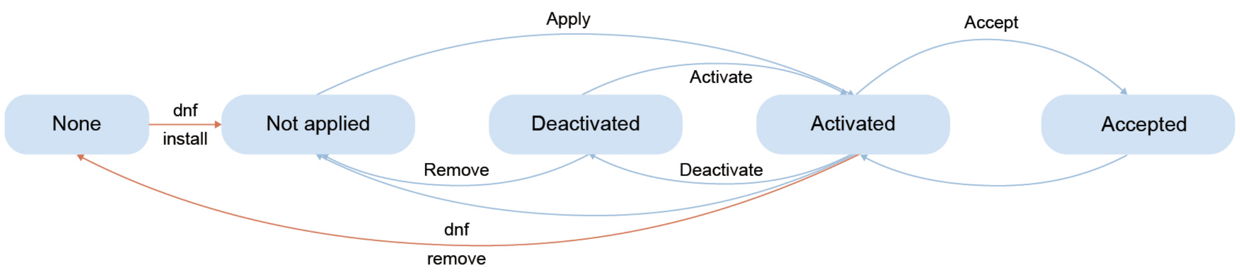
- **Patch lifecycle management**

  SysCare simplifies patch lifecycle management, offering a complete and user-friendly solution. With a single command, users can efficiently manage hot patches, saving time and effort. SysCare leverages the RPM system to build hot patches with complete dependencies. This allows for easy integration into the software repository and simplifies distribution, installation, update, and uninstallation of hot patches.

SysCare architecture



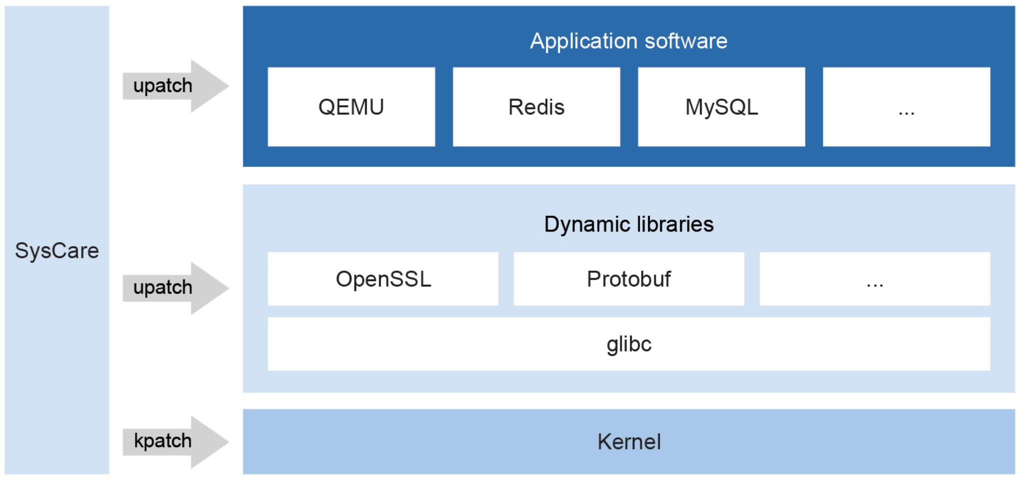
Lifecycle of a hot patch



- **Kernel-mode and user-mode hot patch integration**

  By utilizing the upatch and kpatch technologies, SysCare delivers seamless hot fixing for the entire software stack from applications and dynamic libraries to the kernel, eliminating the need for disruptive downtime.

Application scope of hot patches



New features

- The user-mode hot patch creation tool now offers preliminary support for embedded environments through configurable cross compilation.
- Kernel modules related to user-mode hot patch creation have been removed, streamlining the process for containerized applications.
- A new user-mode process stack inspection mechanism prevents inconsistencies in multi-threaded scenarios, mitigating crashes and exceptions.

Constraints

- Only available for 64-bit OSs
- Only available for ELF files, with no support for interpreted languages or pure assembly modification
- Only available for the GCC and G++ compilers, not supporting cross compilation
- TLS variable modification not supported
- LTO optimization not supported
- Compiler configuration interface for upatch-build only and not for syscare-build

Application Scenarios

Scenario 1: quick fix of common vulnerabilities and exposures (CVEs)

Scenario 2: temporary locating for live-network issues

NRI Plugin Support of iSula

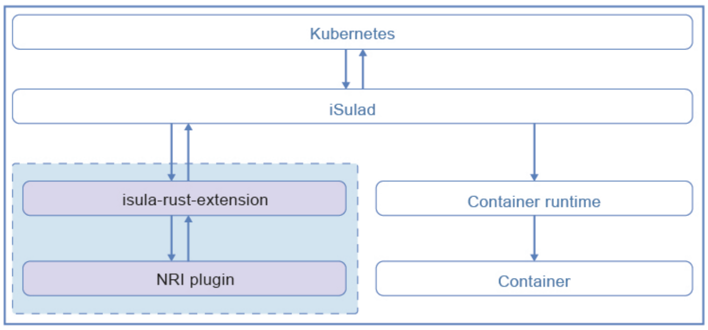
Node Resource Interface (NRI) is a public interface for controlling node resources and provides a generic framework for pluggable extensions for CRI-compatible container runtimes. It offers a fundamental mechanism for extensions to track container states and make limited modifications to their configurations. This allows users to insert custom logic into OCI-compatible runtimes, enabling controlled changes to containers or performing additional operations outside the scope of OCI at certain points in the container lifecycle. The newly added support for NRI plugins of iSulad reduces costs for container resource management and maintenance, eliminates scheduling delays, and ensures information consistency in Kubernetes environments.

Feature Description

An NRI plugin establishes a connection with iSulad through the NRI runtime service started within the isula-rust-extension component. This connection enables the plugin to subscribe to both pod and container lifecycle events.

- For pods, subscribable events include creation, stopping, and removal.
- For containers, subscribable events include creation, post-creation, starting, post-start, updating, post-update, stopping, and removal.

Upon receiving a CRI request from Kubernetes, iSulad relays this request to all NRI plugins subscribed to the corresponding lifecycle events. The request received by an NRI plugin includes metadata and resource information for the relevant pod or container. The plugin can then adjust the resource configuration of the pod or container as required. Finally, the NRI plugin communicates the updated configuration to iSulad, which in turn relays it to the container runtime, making the updated configuration effective.



Constraints

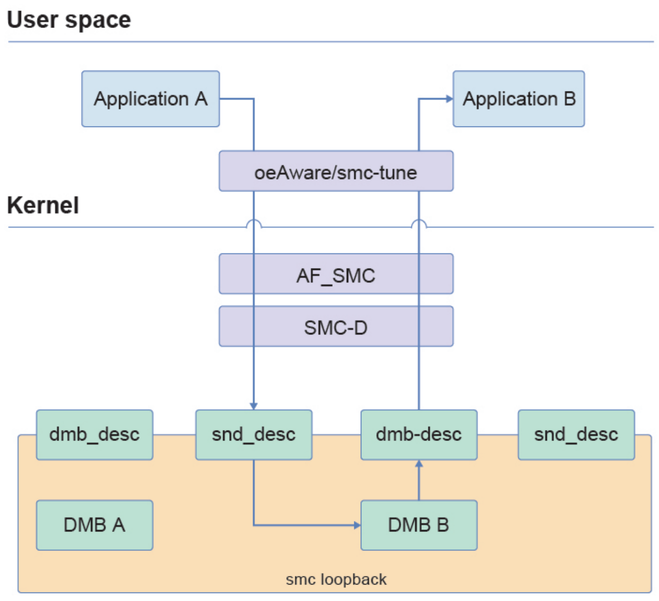
- Supports only Kubernetes CRI v1.
- Supports only NRI API version 0.6.1.

Application Scenarios

NRI support of iSulad can be utilized for resource management within Kubernetes environments. By implementing NRI plugins, users gain the capability to subscribe to container lifecycle events, enabling real-time tracking of container resource status. Based on predefined resource management logic, these plugins can then dynamically modify container configurations within the scope permitted by the NRI API. For instance, NRI facilitates the development of standardized plugins to address resource utilization and priority scheduling challenges inherent in scenarios where online and offline services are deployed together.

oeAware for Local Network Acceleration

The Shared Memory Communications - Direct Memory Access (SMC-D).feature of the Linux kernel optimizes and accelerates Shared Memory Communications (SMC) within the OS, particularly for inter-process communication (IPC). oeAware seamlessly leverages this feature to transparently accelerate TCP transmissions using SMC-D.



Feature Description

SMC-D introduces a loopback device accessible to all containers as a global system device, enabling accelerated TCP communication between local processes. This design ensures consistent SMC-D performance across various virtual environments.

- **Efficient communication**: SMC-D employs the Internal Shared Memory (ISM) technology, bypassing traditional network communication overhead to achieve low latency and high throughput.
- **Transparent replacement**: SMC-D can seamlessly replace existing TCP protocol stacks, eliminating the need for extensive application modifications.
- **Automatic negotiation**: SMC-D features automatic negotiation and dynamic fallback to TCP, guaranteeing compatibility and stability across different environments.

Constraints

- SMC acceleration must be enabled before the server-client connection is established.
- This feature is most effective in scenarios with numerous persistent connections.

Application Scenarios

SMC-D is particularly suitable for applications requiring high throughput and low latency, such as high-performance computing (HPC), big data processing, and cloud computing platforms. Through Direct Memory Access (DMA), SMC-D significantly reduces CPU load and enhances the speed of interactive workloads.

- **HPC**: In HPC environments demanding high throughput and low latency, SMC-D can significantly enhance data transfer efficiency.
- **Data centers**: Within data centers, SMC-D can accelerate inter-process communication, reducing network latency and CPU load.
- **Container communication**: SMC-D facilitates fast inter-process communication between containers within the same OS, making it ideal for microservice architectures.

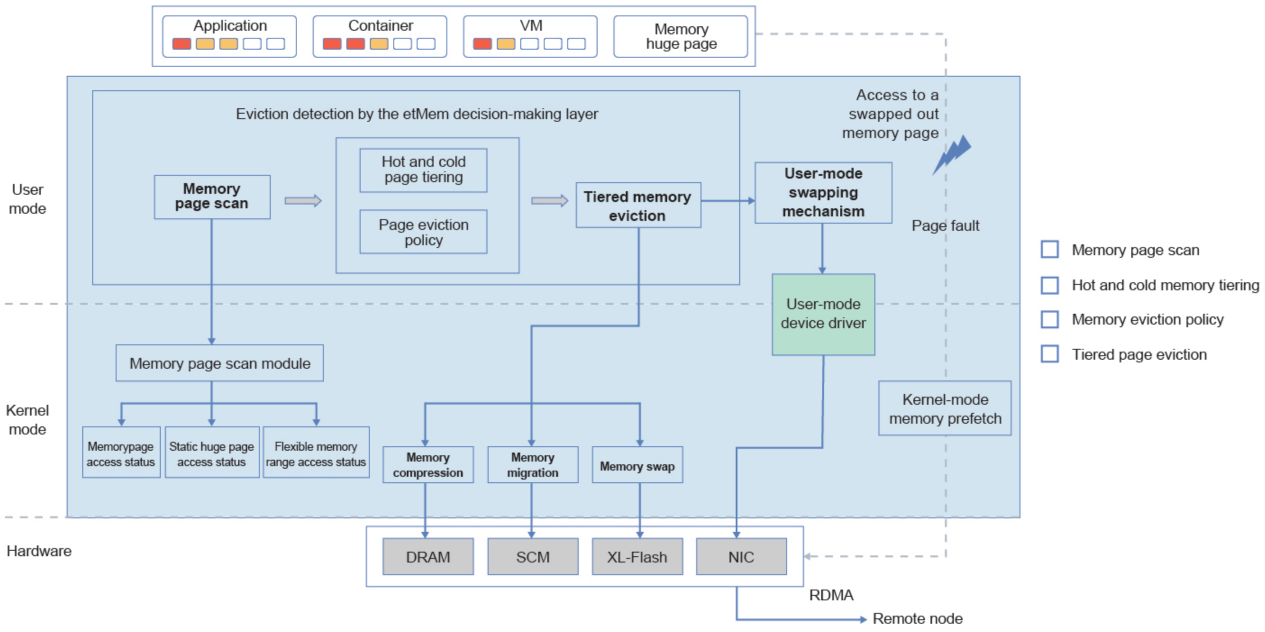
etMem for VM Memory Overcommitment

As CPU computing power continues to advance, particularly with the decreasing cost of Arm cores, memory cost and capacity have become critical bottlenecks constraining service costs and performance. Therefore, saving memory costs and expanding memory capacity have become urgent issues to be addressed in the field of storage.

etMem is a tiered memory expansion technology that utilizes a multi-tiered memory storage architecture consisting of DRAM combined with memory compression/high-performance storage media. This architecture enables the tiering of memory data, migrating cold data from memory media to high-performance storage media, effectively expanding memory capacity and reducing memory costs.

Feature Description

etMem leverages a swap-in/swap-out mechanism using DRAM, storage-class memory (SCM), and XL-Flash to create a multi-tiered storage system. This system is transparent to applications, compatible with existing application ecosystems, and achieves local tiered memory, ultimately reducing memory costs.



- **Memory page scan**: The kernel-mode memory page scan module traverses page tables, obtaining access bits for each page. This information determines whether a page has been accessed within the scan period. The scan period, frequency, and memory range can be configured.
- **Hot and cold memory tiering**: Based on the memory page scan results, the system statistically identifies the distribution of hot and cold data in memory. If multi-tiered storage is available, pages are tiered based on their temperature. Currently, only two tiers (hot and cold) are supported, while the implementation of multiple tiers is under technological research.
- **Memory eviction policy**: Memory pages to be evicted are determined based on the hot/cold tiering and migration policy. etMem provides a default policy and supports user-defined or third-party eviction policies.
- **Tiered page eviction**: Pages are evicted to different backend media using the kernel-mode memory compression, swapping, or move_pages migration mechanism. Both memory swapping and memory compression rely on the eviction mechanism of etMem to specify target pages. They leverage existing open source implementations for the actual swapping,

compression, and migration operations, utilizing the designated entry point for the memory page eviction mechanism instead of the Least-Recently-Used (LRU) eviction mechanism of Linux.

- **VM memory expansion**: etMem supports memory expansion in VM environments. For VM memory overcommitment scenarios, it analyzes the second-level page tables of a VM to determine the hot/cold memory within the VM. This enables hot/cold data identification and ultimately achieves VM memory overcommitment.

Application Scenarios

Compared to passive memory expansion technologies like kswap, etMem enables proactive memory swapping at the process level, including preemptive swapping. This ensures performance for critical service processes. This feature is suitable for scenarios where server memory is limited and service processes have varying priorities, such as databases, virtualization, and container management. It reduces service memory footprint while minimizing performance fluctuations in critical service processes.

- **Traditional services**: With the current memory technology reaching its limits and memory costs remaining high, memory expansion enables services to allocate memory data with varying temperatures to different tiered media, reducing DRAM usage and overall memory costs.
- **Memory overcommitment in virtualization**: In cloud server and multi-VM scenarios, etMem can effectively reduce memory costs.

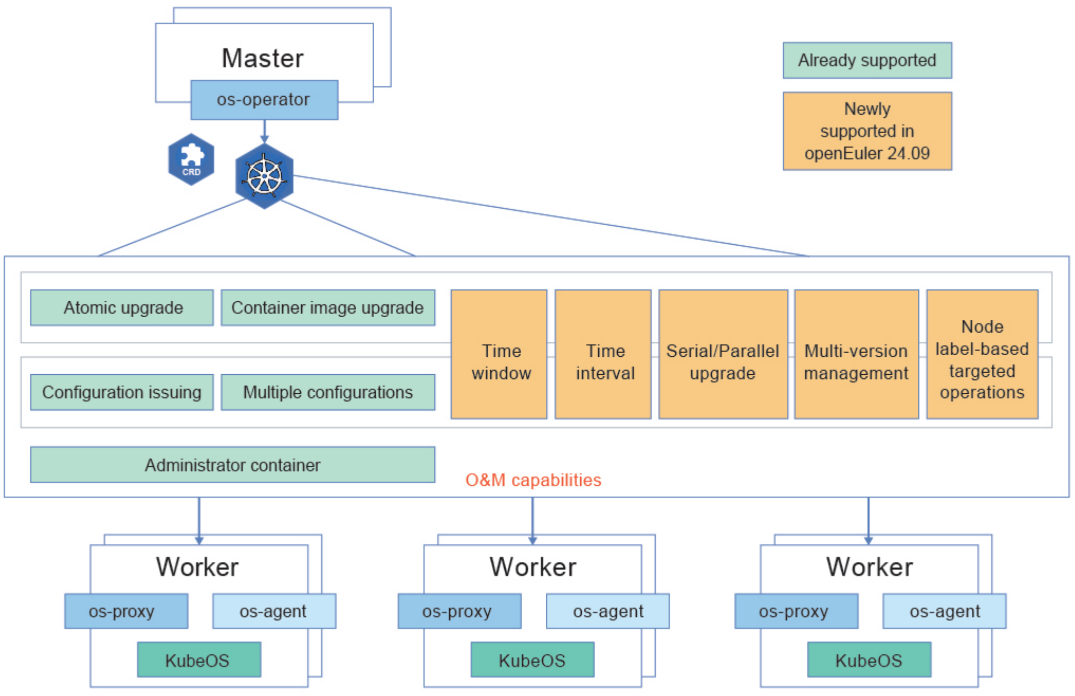
Repository

https://gitee.com/openeuler/etmem

Declarative API Support of KubeOS

KubeOS is a lightweight and secure OS designed for cloud-native environments. It simplifies O&M by providing unified tools for Kubernetes-based systems. KubeOS is specifically designed for running containers. It features a read-only root directory, includes only the essential components for the container runtime, and utilizes dm-verity security hardening. This minimizes vulnerabilities and attack surfaces while improving resource utilization and boot speed. KubeOS can leverage native Kubernetes declarative APIs to unify the upgrade, configuration, and maintenance of worker node OSs within a cluster. This approach simplifies cloud-native operations, addresses challenges associated with OS version fragmentation across cluster nodes, and provides a unified solution for OS management.

Feature Description



The figure shows the O&M policies added by the unified O&M framework of KubeOS.

- Configurable maintenance windows and intervals
  - KubeOS allows users to define time windows for upgrades and configuration changes, ensuring these operations occur within specified periods.
  - KubeOS enables the configuration of time intervals between successive upgrade or configuration tasks. Once an operation is completed, the system waits for the specified interval before issuing the next one.
- Serial and parallel O&M policies
  - The parallel policy enables concurrent upgrades or configuration changes on multiple nodes (user-defined).
  - The serial policy executes upgrades or configuration tasks on a specified number of nodes (user-defined) sequentially. The next batch of nodes is processed only after all nodes in the current batch have completed the operation.

- Multi-version OS management
  - KubeOS supports multiple OS Custom Resources (CRs) within a single cluster, allowing for different OS versions.
  - Node selection for upgrades or configuration changes is based on the **nodeselector** field within the OS CR, enabling targeted operations based on node labels.

Application Scenarios

KubeOS simplifies OS management in cloud-native environments, addressing pain points such as heavy manual intervention and long OS upgrade times. The unified O&M framework of KubeOS provides flexible and multi-dimensional policies, empowering users to perform scheduled, serial, and multi-version OS management across clusters.

Third-Party PKI Certificate Support of IMA Digest Lists

The Integrity Measurement Architecture (IMA) is a kernel subsystem that measures files access through system calls such as **execve()**, **mmap()**, and **open()** based on custom policies. These measurements can be used for local and remote attestation to implement file integrity protection.

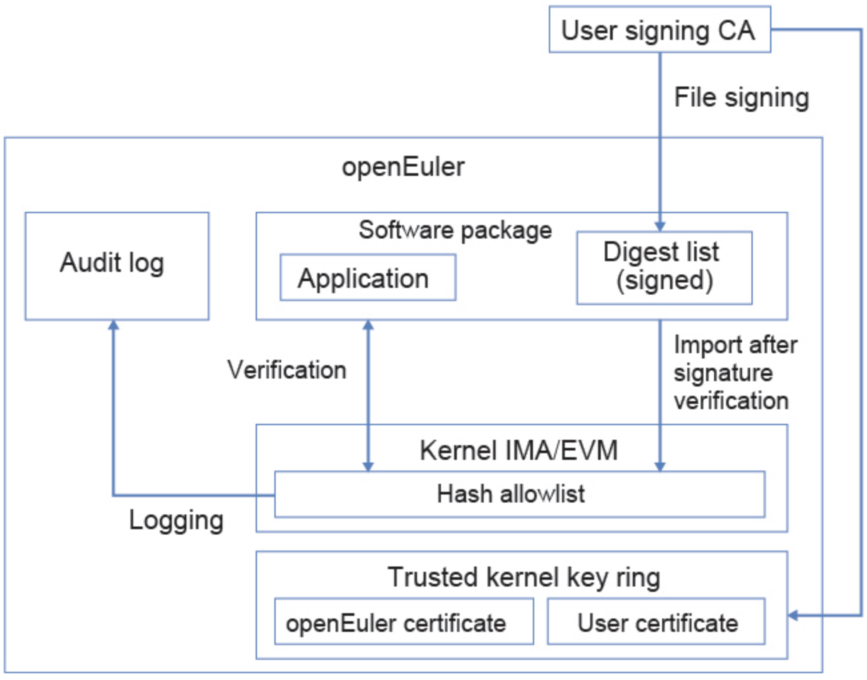
IMA digest lists, an openEuler enhancement to the native IMA mechanism of the kernel, were introduced in openEuler 21.03. This feature improves overall performance, security, and usability by pre-generating and signing digest lists during the RPM package building process.

Feature Description

Similar to secure boot, the IMA appraisal mode aims to ensure system integrity by verifying the integrity of executed applications and accessed critical files. If the verification fails, access is denied.

When IMA digest lists are enabled, the verification process relies on an allowlist of trusted hash values maintained by the kernel. This allowlist can be dynamically updated by adding or removing digest list files from the user space via kernel interfaces. To ensure the authenticity of the allowlist, a signature verification mechanism is employed.

openEuler 24.09 introduces support for user-imported certificates. This allows users to pre-set custom signing certificates during kernel compilation. These custom certificates can then be used for signature verification of digest list files in subsequent operations, enhancing the security and flexibility of the IMA appraisal process.



Application Scenarios

This feature enables users to implement secure boot for user-mode applications, ensuring that only pre-authenticated applications can run in the system, effectively preventing the execution of malicious code.

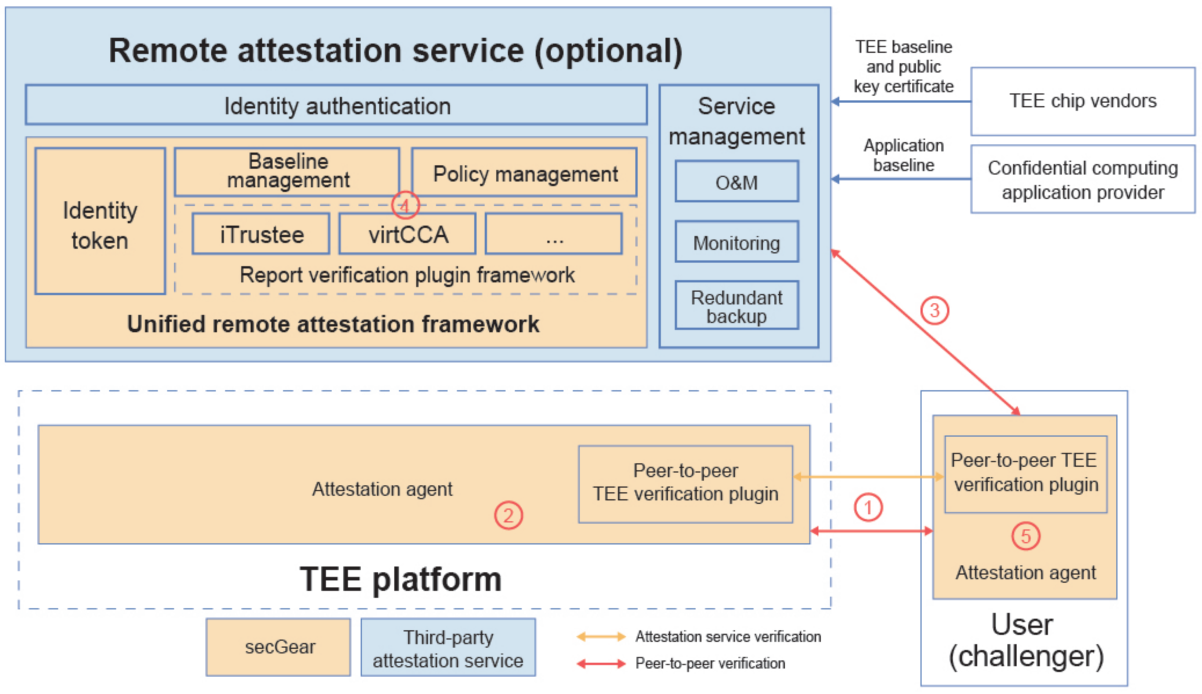
secGear for Remote Attestation

The unified remote attestation framework of secGear addresses the key components related to remote attestation in confidential computing, abstracting away the differences between different Trusted Execution Environments (TEEs). It provides two components: attestation agent and attestation service. The agent is integrated by users to obtain attestation reports and connect to the attestation service. The service can be deployed independently and supports the verification of iTrustee and virtCCA remote attestation reports.

Feature Description

The unified remote attestation framework focuses on confidential computing functionalities, while service deployment and operation capabilities are provided by third-party deployment services. The key features of the unified remote attestation framework are as follows:

- **Report verification plugin framework**: Supports runtime compatibility with attestation report verification for different TEE platforms, such as iTrustee, virtCCA, and CCA. It also supports the extension of new TEE report verification plugins.
- **Certificate baseline management**: Supports the management of baseline values of Trusted Computing Bases (TCB) and Trusted Applications (TA) as well as public key certificates for different TEE types. Centralized deployment on the server ensures transparency for users.
- **Policy management**: Provides default policies for ease of use and customizable policies for flexibility.
- **Identity token**: Issues identity tokens for different TEEs, endorsed by a third party for mutual authentication between different TEE types.
- **Attestation agent**: Supports connection to attestation service/peer-to-peer attestation, compatible with TEE report retrieval and identity token verification. It is easy to integrate, allowing users to focus on their service logic.


Remote attestation service (optional)
Identity authentication
Service management
Identity token
Baseline management
Policy management
4
iTrustee
virtCCA
...
Report verification plugin framework
Unified remote attestation framework
O&M
Monitoring
Redundant backup
TEE baseline and public key certificate
TEE chip vendors
Application baseline
Confidential computing application provider
3
Attestation agent
2
Peer-to-peer TEE verification plugin
1
TEE platform
Peer-to-peer TEE verification plugin
5
Attestation agent
User (challenger)
secGear
Third-party attestation service
Attestation service verification
Peer-to-peer verification

Two modes are supported depending on the usage scenario: peer-to-peer verification and attestation service verification.

Attestation service verification process:

1. The user (regular node or TEE) initiates a challenge to the TEE platform.
2. The TEE platform obtains the TEE attestation report through the attestation agent and returns it to the user.
3. The user-side attestation agent forwards the report to the remote attestation service.
4. The remote attestation service verifies the report and returns an identity token in a unified format endorsed by a third party.
5. The attestation agent verifies the identity token and parses the attestation report verification result.

Peer-to-peer verification process (without the attestation service):

1. The user initiates a challenge to the TEE platform, which then returns the attestation report to the user.
2. The user uses a local peer-to-peer TEE verification plugin to verify the report.

Note: The attestation agents used for peer-to-peer verification and attestation service verification are different. During compilation, the compilation options determine whether to compile attestation agents for the attestation service mode or peer-to-peer mode.

Application Scenarios

In scenarios like finance and AI, where confidential computing is used to protect the security of privacy data during runtime, remote attestation is a technical means to verify the legitimacy of the confidential computing environment and applications. secGear provides components that are easy to integrate and deploy, helping users quickly enable confidential computing remote attestation capabilities.

gala-anteater for Minute-Level Container Interference Detection

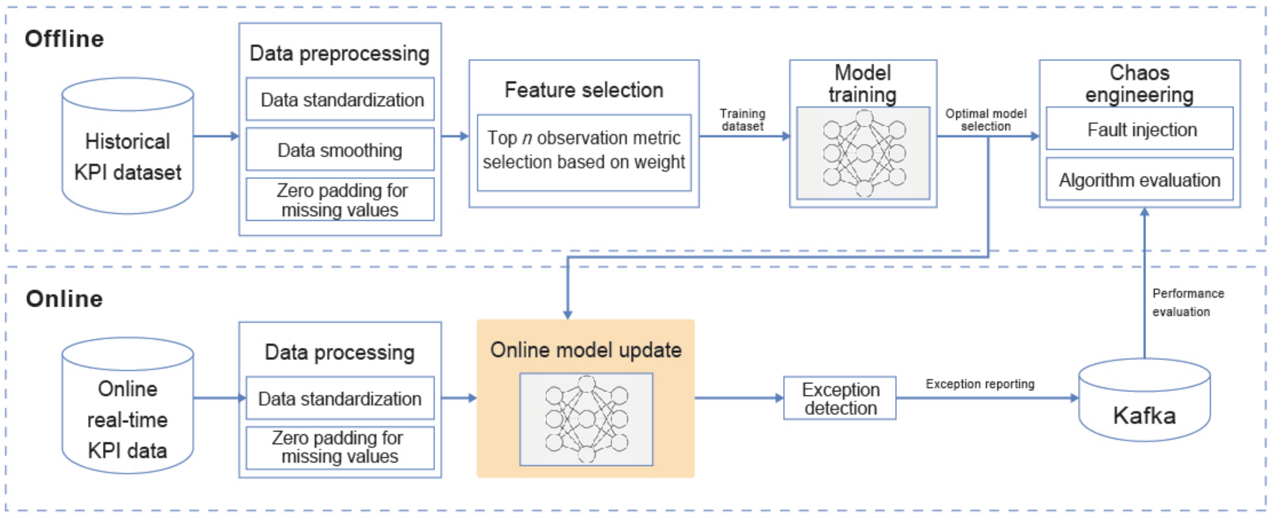
gala-anteater is an AI-powered exception detection platform for gray faults in the OS. Integrating various exception detection algorithms, it achieves system-level fault detection and reporting through automated model pre-training, online incremental learning, and model updates.

In high-density online container deployment scenarios, the presence of disorderly resource contention can lead to inter-container interference. gala-anteater enables minute-level identification of interference sources (CPU or I/O), aiding O&M personnel in swiftly tracing and resolving issues to ensure service QoS.

Feature Description

gala-anteater leverages a combination of offline and online learning techniques to facilitate offline model training and online updates, ultimately enabling real-time online exception detection.

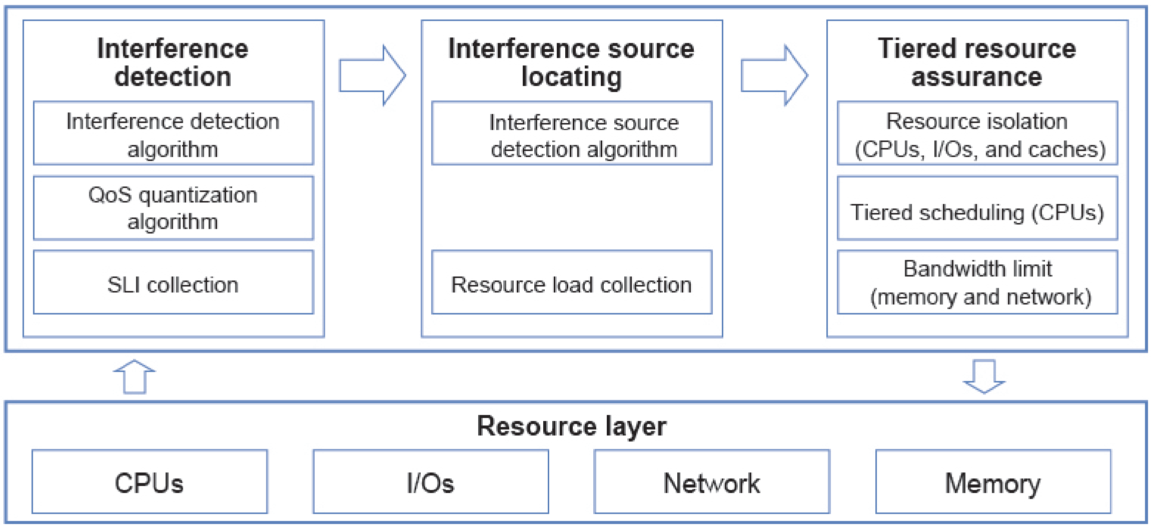
- **Offline**: Initially, historical KPI datasets undergo preprocessing and feature selection to generate a training set. This set is then used to train and optimize an unsupervised neural network model (such as a variational autoencoder). Finally, a manually labeled test set aids in selecting the optimal model.
- **Online**: The trained model is deployed online, where it undergoes further training and parameter tuning using real-time data. This continuously refined model then performs real-time exception detection within the online environment.



Application Scenarios

gala-anteater supports both application-level and system-level exception detection and reporting.

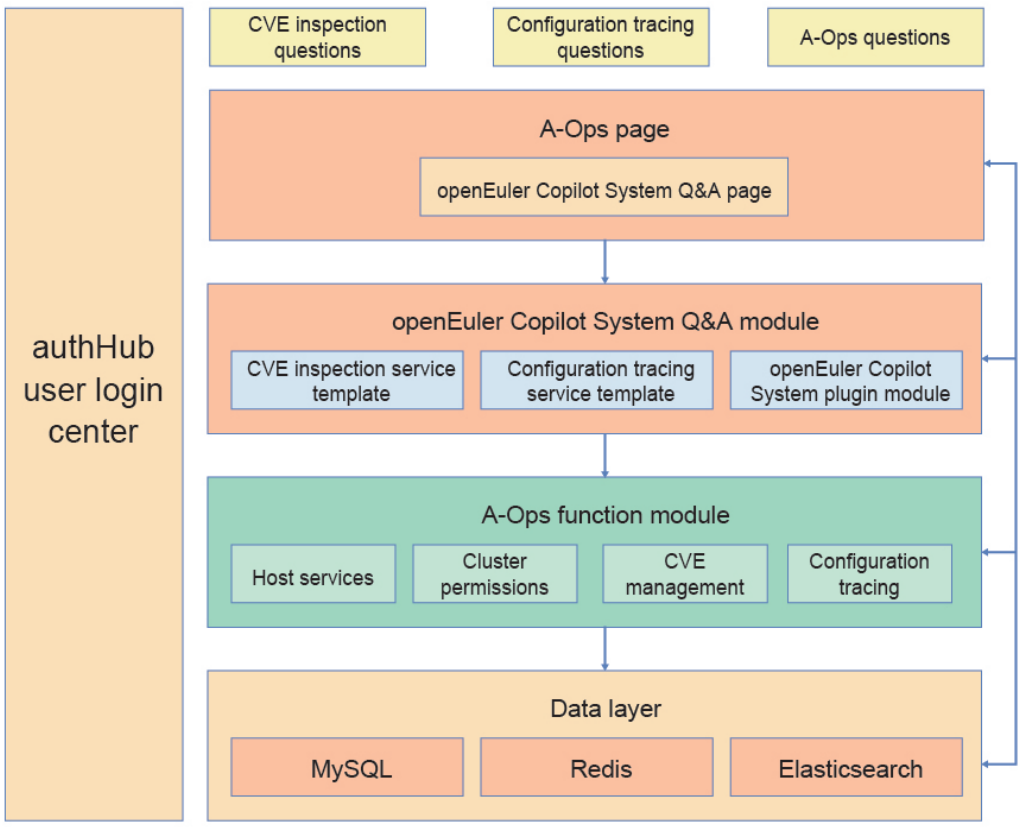
In actual application scenarios, it can be integrated with gala-gopher and Rubik to achieve real-time interference detection, locating, and self-healing. gala-anteater leverages real-time resource-level metrics collected by gala-gopher for interference detection and locating. The reported detection results are then used by Rubik to implement tiered resource assurance, enabling rapid recovery from interference.


Interference detection
Interference detection algorithm
QoS quantization algorithm
SLI collection
Interference source locating
Interference source detection algorithm
Resource load collection
Tiered resource assurance
Resource isolation (CPUs, I/Os, and caches)
Tiered scheduling (CPUs)
Bandwidth limit (memory and network)
Resource layer
CPUs
I/Os
Network
Memory

Intelligent O&M Assistant of A-Ops

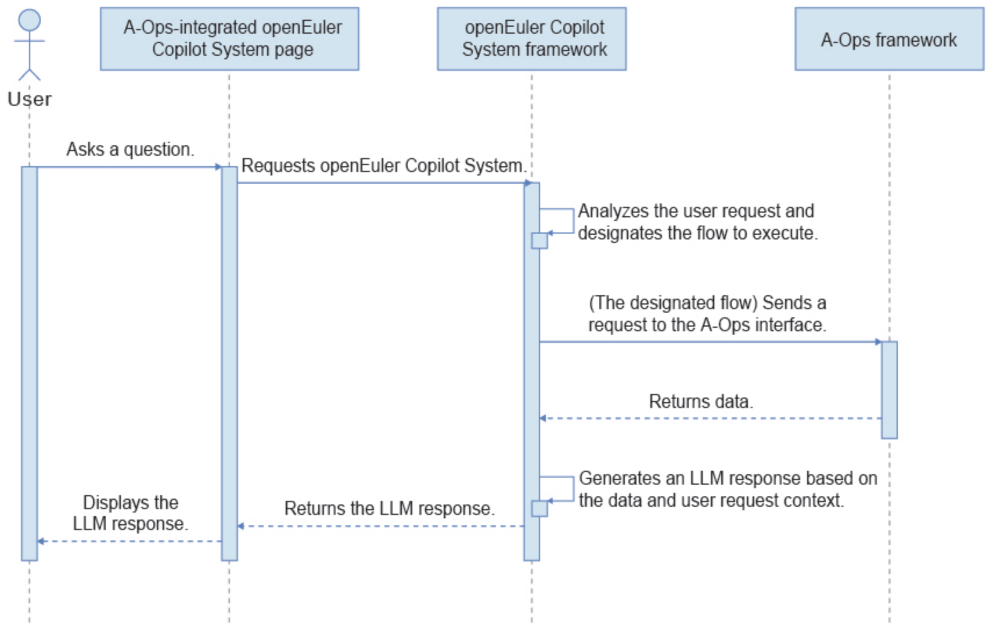
A-Ops leverages openEuler Copilot System plugins to provide an intelligent O&M assistant. This assistant allows users to perform CVE inspection and configuration tracing through natural languages.

Feature Description


authHub user login center
CVE inspection questions
Configuration tracing questions
A-Ops questions
A-Ops page
openEuler Copilot System Q&A page
openEuler Copilot System Q&A module
CVE inspection service template
Configuration tracing service template
openEuler Copilot System plugin module
A-Ops function module
Host services
Cluster permissions
CVE management
Configuration tracing
Data layer
MySQL
Redis
Elasticsearch

A-Ops offers the following functionalities through openEuler Copilot System plugins:

- **Unified user login center**: A-Ops integrates with openEuler Copilot System to provide authHub, a unified user login center for streamlined authentication.
- **CVE inspection and configuration tracing**: A-Ops leverages service scenario flow templates to facilitate heuristic CVE inspection and configuration tracing, enabling users to interact with the system using natural languages.

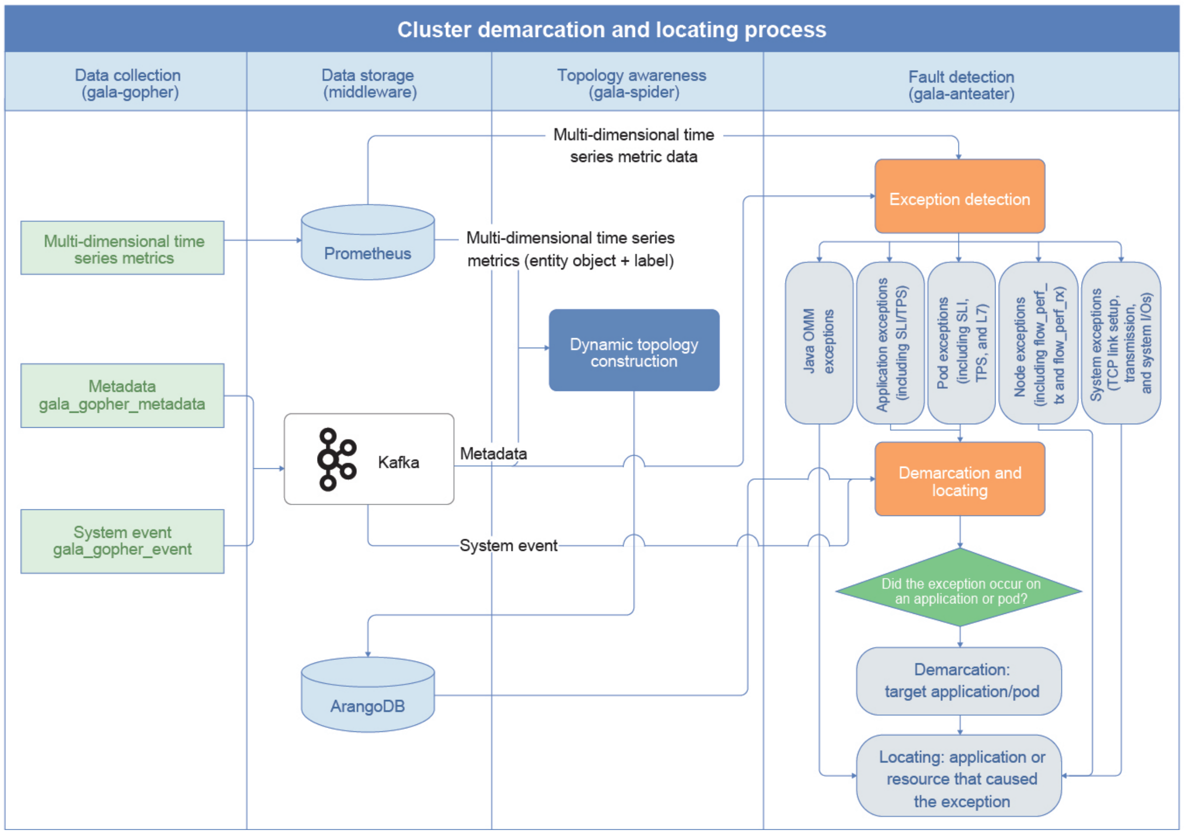

User
A-Ops-integrated openEuler Copilot System page
openEuler Copilot System framework
A-Ops framework
Asks a question.
Requests openEuler Copilot System.
Analyzes the user request and designates the flow to execute.
(The designated flow) Sends a request to the A-Ops interface.
Returns data.
Generates an LLM response based on the data and user request context.
Returns the LLM response.
Displays the LLM response.

Application Scenarios

The intelligent O&M assistant simplifies complex operations by allowing users to interact using natural languages. This streamlines the workflow for CVE inspection and configuration tracing, ultimately enhancing user experience.

Minute-level Demarcation and Locating of Microservice Performance Problems (TCP, I/Os, and Scheduling)

Feature Description


Cluster demarcation and locating process
Data collection (gala-gopher)
Data storage (middleware)
Topology awareness (gala-spider)
Fault detection (gala-anteater)
Multi-dimensional time series metric data
Multi-dimensional time series metrics
Prometheus
Multi-dimensional time series metrics (entity object + label)
Exception detection
Java OMM exceptions
Application exceptions (including SLI/TPS)
Pod exceptions (including SLI, TPS, and L7)
Node exceptions (including flow_perf_tx and flow_perf_rx)
System exceptions (TCP link setup, transmission, and system I/Os)
Dynamic topology construction
Metadata gala_gopher_metadata
Kafka
Metadata
Demarcation and locating
System event gala_gopher_event
System event
Did the exception occur on an application or pod?
Demarcation: target application/pod
ArangoDB
Locating: application or resource that caused the exception

gala-gopher, gala-spider, and gala-anteater implement a topological root cause analysis approach to facilitate fault detection and root cause locating in large-scale clusters. In openEuler 24.09, fine-grained capabilities have been introduced for cloud-native environments based on Layer 7 protocols like HTTPS, PostgreSQL, and MySQL, enabling O&M teams to quickly locate the source of faults, thus enhancing system stability and reliability. The following features are available:

- **Metric collection**: gala-gopher uses eBPF to collect and report network and I/O metrics.
- **Cluster topology**: gala-spider receives data reported by gala-gopher and constructs a container- and process-level call relationship topology.
- **Fault detection**: gala-anteater classifies the metrics reported by the application based on the fault detection model to determine whether an exception has occurred in the system.
- **Root cause locating**: gala-anteater locates the root cause node of the exception based on node exception and topology information.

Application Scenarios

Minute-level demarcation and locating of application problems are suited to the following scenarios:

- **Cloud Kubernetes pods**: Enterprises with cloud environments can use gala-gopher to collect metrics of Kubernetes containers and processes, gala-spider to construct a call relationship topology, and gala-anteater to detect faults and locate root causes based on their requirements and IT resource status.
- **Bare metal deployments**: Enterprises with bare metal deployments can use gala-gopher to collect metrics of processes and containers, gala-spider to construct a call relationship topology, and gala-anteater to detect faults and locate root causes based on their requirements and IT resource status.
- **Large-scale VMs**: Enterprises with large-scale VMs can use gala-gopher to collect metrics of VMs, gala-spider to construct a call relationship topology, and gala-anteater to detect VM faults and locate root causes based on their requirements and IT resource status.

ROS 2 Humble and ROS 1 Noetic

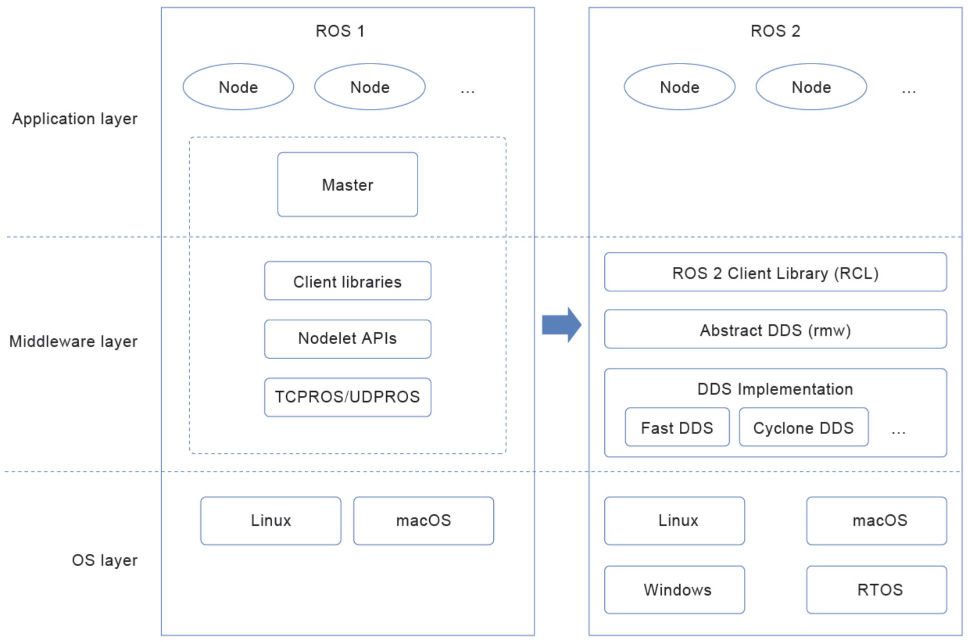
The Robot Operating System (ROS) is a set of software libraries and tools that help simplify and accelerate the build of robot applications. From drivers to state-of-the-art algorithms, and with powerful developer tools, ROS uses its unique communication architecture to closely integrate software and hardware components, building an efficient and collaborative working platform for developers. The ROS project has developed a vast robotics software ecosystem over decades, comprising three main categories of software:

- **Core components**: core software packages related to middleware, software package compilation and build, communication, messaging, and APIs
- **Algorithm libraries**: common algorithm packages in the robotics domain, such as GMapping, Navigation, MOVEit, and Controller, as well as drivers for executors and sensors
- **Development and debugging tools**: rviz (a 3D visualizer for debugging), Gazebo (a simulator), rqt (a graphical debugging tool), message recording and playback tools, command line utilities, tf, and launch

ROS is available in two versions: ROS 1 and ROS 2. The ROS SIG has designated ROS 2 Humble and ROS 1 Noetic as the long-term maintenance versions to support ROS applications on openEuler.

Feature Description

Architectures of ROS 1 and ROS 2



- **OS layer**

  ROS 1 is not a conventional OS, as it does not directly run on a computer like Windows and Linux, but instead, operates on top of various OSs, including Linux (such as Ubuntu), macOS, Windows, and real-time operating systems (RTOSs). This layer provides fundamental OS support, including hardware drivers and process management.

- **Middleware layer**

  » ROS 1 offers a vast array of middleware for robot development. Its core uses the TCPROS/UDPROS communication system, which is built on TCP/UDP and uses publish/subscribe and client/server models to implement data transmission of multiple communication mechanisms.

- » ROS 1 incorporates nodelets for in-process communication in scenarios demanding high-performance real-time data transmission.
- » ROS 1 also provides a substantial collection of libraries that can be directly used at the application layer for robot development, including those for data type definition, coordinate transformation, and motion control.
- » The middleware layer is the core of ROS 2, incorporating Data Distribution Service (DDS) along with middleware for robot development. DDS is a standard set by the Object Management Group (OMG) to facilitate decentralized data communication. It enables nodes to communicate independently of a central management node (for example, the Master node in ROS 1), thereby improving system robustness and scalability.
- » The middleware layer can be further divided into the following layers:
  - – **DDS Implementation layer**: This layer is compatible with multiple DDS vendors. You can select DDS implementations from different vendors to fit your requirements, such as eProsima Fast DDS and Eclipse Cyclone DDS.
  - – **Abstract DDS (rmw) layer**: The ROS middleware (rmw) interface is a middleware abstraction layer implemented in C. It directly interacts with DDS and provides APIs for the ROS 2 Client Library (RCL) layer. This layer enables user programs to be ported across different DDS implementations from various vendors.
  - – **RCL layer**: It includes a core RCL interface that provides access to ROS 2 concepts like nodes and actions, and language-specific APIs such as rclcpp (C++ client library), rclpy (Python client library), and rcljava (Java client library). The basic functionality of all ROS 2 elements is implemented in a single C library called RCL; then the language-specific libraries adapt this functionality to the particularities of each language. APIs for other languages can be implemented. This ensures consistent program running in different programming languages.

- **Application layer**
  - » At the application layer, ROS 1 runs a Master, which manages the normal running of the entire system. The Master provides a registration list and computation graph search through Remote Procedure Call (RPC) interfaces, enabling ROS nodes to search for and establish connections with each other and manage global parameters.
  - » The ROS 1 community is home to a wide range of robot application packages, the modules of which are executed by nodes. Developers can easily reuse these packages by leveraging standard input and output interfaces provided by ROS, thereby improving development efficiency.
  - » ROS 2 provides function packages for application programs. These function packages contain source code, data definition, and interfaces, enabling developers to build robot applications with specific functionalities.

Constraints

- **Restrictions on ROS 1**
  - » **Single points of failure**

    The ROS 1 Master node manages the communication and status of all nodes, and its failure can cause downtime across the entire system because the communication between nodes is interrupted.
  - » **Platform dependency**

    ROS 1 is designed to run on the Linux platform and is not well-suited to other OSs. This limits the wide application of ROS 1 on different robot platforms.
  - » **Low security**

    ROS 1 communication protocols (such as TCP/IP) have poor data encryption and security, and are unsuitable for scenarios that have higher security requirements.
  - » **Poor timeliness**

    For robot applications that require real-time processing, ROS 1 may fall short in delivering satisfactory performance because it was not designed with real-time processing as a primary consideration.

- **Compilation and build systems**

  For compilation and build systems (such as rosbuild and catkin) used by ROS 1 in large-scale or Python projects, there may be compilation and link errors that affect development efficiency.

- **Restrictions on ROS 2**

  Although ROS 2 provides improvements in design and implementation to address the restrictions on ROS 1, it also faces new challenges and limitations:

  - **Increased complexity**

    ROS 2 adopts a more powerful communication protocol DDS and programming models, but in doing so, this design increases system complexity and steepen the learning curve.

  - **Compatibility**

    ROS 2 has made significant changes in APIs and communication protocols, but this prevents the code developed for ROS 1 from directly running in ROS 2. Developers have to refactor the code for ROS 1 if they want to port existing projects to ROS 2.

  - **DDS implementation diversity**

    As an international standard, DDS has multiple implementations, which can cause performance and interface differences that trouble developers, despite ROS 2's support through the rmw layer.

  - **Higher resource consumption**

    ROS 2 provides higher reliability and scalability than its predecessor, but consumes more resources (CPUs, memory, and network bandwidth), making it unsuitable for embedded systems with limited resources.

  - **Low ecosystem maturity**

    Despite recent advancements, the ROS 2 ecosystem remains less mature than that of ROS 1. Some tools and libraries commonly used in ROS 1 may not be fully supported or optimized in ROS 2.

Application Scenarios

ROS provides powerful support for modern robotics, such as industrial automation, agricultural automation, healthcare (such as surgical robots), home service robots (such as floor sweeping robots), entertainment (such as robot competitions), safety assurance (such as disaster response), and unmanned driving.

utsudo

sudo is one of the commonly used utilities for Unix-like and Linux OSs. It enables users to run specific commands with the privileges of the super user. utsudo is developed to address issues of security and reliability common in sudo.

utsudo uses Rust to reconstruct sudo to deliver more efficient, secure, and flexible privilege escalation. It includes modules such as the common utility, overall framework, and function plugins.

Feature Description

Basic features:

- **Access control**: Limits the commands that can be executed by users, and specifies the required authentication method.
- **Audit log**: Records and traces all commands and tasks executed by each user.
- **Temporary privilege escalation**: Allows common users to temporarily escalate to a super user for executing privileged commands or tasks.
- **Flexible configuration**: Allows users to set arguments such as command aliases, environment variables, and execution parameters to meet system requirements.

Enhanced features:

utsudo 0.0.2 comes with openEuler 24.09 to provide the following features:

- **Privilege escalation**: Escalates the privilege of a process run by a common user to the **root** privilege.
- **Plugin loading**: Parses plugin configuration files and dynamically loads plugin libraries.

Application Scenarios

utsudo reduces security risks by enabling administrators to better manage user privileges and authorization, and prevent unauthorized privileged operations.

It is suitable for management and maintenance, user permission control, and multi-user environments.

utshell

utshell is a new shell that introduces new features and inherits the usability of Bash. It enables interaction through command lines, such as responding to user operations to execute commands and providing feedback, and can execute automated scripts to facilitate O&M.

Feature Description

Basic features:

- **Command execution**: Runs and sends return values from commands executed on user machines.
- **Batch processing**: Automates task execution using scripts.
- **Job control**: Executes, manages, and controls multiple user commands as background jobs concurrently.
- **Historical records**: Records commands entered by users.
- **Command aliases**: Allows users to create aliases for commands to customize their operations.

Enhanced features:

utshell 0.5 runs on openEuler 24.09 to perform the following operations:

- Parses shell scripts.
- Runs third-party commands.

Application Scenarios

utshell is well suited to conventional servers, cloud-native environments, and attended or unattended production sites where automated O&M scripts are executed, as well as the demands of individual users.

GCC for openEuler

The baseline version of GCC for openEuler has been upgraded from open source GCC 10.3 to GCC 12.3, supporting features such as automatic feedback-directed optimization (AutoFDO), software and hardware collaboration, memory optimization, Scalable Vector Extension (SVE), and vectorized math libraries.

- The default language of GCC for openEuler has been upgraded from C14/C++14 to C17/C++17, enabling GCC for openEuler to support more hardware features like Armv9-A and x86 AVX512-FP16.

Item	GCC 10.3.0	GCC 11.3.0	GCC 12.3.0
Release date	2021-04-08	2022-04-21	2023-05-08
C standard	C17 by default C2x supported	C17 by default C2x supported	C17 by default C2x supported
C++ standard	C++14 by default C++17 supported C++2a experimental optimization C++20 partially supported	C++17 supported C++2a experimental optimization C++20 partially supported	C++17 supported C++2a experimental optimization C++20 partially supported
New architecture features	Armv8.6-a(bfloat16 Extension/Matrix Multiply Extension) SVE2 Cortex-A77 Cortex-A76AE Cortex-A65 Cortex-A65AE Cortex-A34	Armv8.6-a, +bf16, +i8mm Armv8.6-r Cortex-A78 Cortex-A78AE Cortex-A78C Cortex-X1	Armv8.7-a, +ls64 atomic load and store Armv8.8-a, +mop, accelerate memory operations Armv9-a Ampere-1 Cortex-A710 Cortex-X2 AVX512-FP16 SSE2-FP16

- GCC for openEuler supports structure optimization and instruction selection optimization, leveraging Arm hardware features to improve system running efficiency. In the benchmark tests such as SPEC CPU 2017, GCC for openEuler has proven to deliver higher performance than GCC 10.3 of the upstream community.
- Further, it fuels AutoFDO to improve the performance of MySQL databases at the application layer.

Feature Description

- **SVE vectorization**: Significantly improves program running performance for Arm-based machines that support SVE instructions.
- **Memory layout**: Rearranges the positions of structure members so that frequently accessed structure members are placed in continuous memory space, increasing the cache hit ratio and improving program performance.
- **SLP transpose optimization**: Improves the analysis of loops with consecutive memory reads during loop splitting, and adds analysis to transpose grouped stores in the superword level parallelism (SLP) vectorization stage.

- **Redundant member elimination**: Eliminates structure members that are never read and deletes redundant write statements, which in turn reduces the memory occupied by the structure and alleviates subsequent bandwidth pressure, while improving performance.
- **Array comparison**: Implements parallel comparison of array elements to improve execution efficiency.
- **Arm instruction optimization**: Simplifies the pipeline of ccmp instructions for a wide range of deployments.
- **AutoFDO**: Uses perf to collect and parse program information and optimizes feedback across the compilation and binary phases, boosting mainstream applications such as MySQL databases.

Application Scenarios

In terms of general-purpose computing, GCC for openEuler reported 20% gains over GCC 10.3 in the SPEC CPU 2017 test.

With AutoFDO enabled, MySQL performance is 15% higher; while with kernel-mode PGO enabled, the UnixBench performance is boosted by over 3%.

KTIB

Kylin Trust Image Builder (KTIB) is a self-developed, open source image builder with components such as static compliance scanning, single-step image building, push-button image building, and image management. It provides a comprehensive image building solution, helping ensure the security and compliance of images.

Feature Description

- **Rapid building of the container image rootfs**
  - » KTIB supports custom building of the container image rootfs.
  - » Users can easily customize and extend rootfs using configuration files, meeting various service needs.
- **Static compliance scanning**
  - » Static compliance scanning can be performed on Dockerfiles before image building.
  - » Default scanning policy files based on CIS specifications are provided.
  - » Policy files are customizable. Users can modify the file content by referring to the official document, defining scan items as required.
  - » JSON-formatted scan reports can be generated with corresponding modification suggestions.
- **Single-step image building**
  - » KTIB allows users to execute Dockerfile instructions (FROM, COPY, RUN, etc.) one by one using the shell commands.
  - » This step-by-step building method allows users to view and debug each building step, aiding in troubleshooting and optimizing the building process.
  - » Users can selectively execute specific building instructions, gaining control over the image building process.
- **Push-button image building**

  KTIB can automatically execute all instructions in the Dockerfile to generate an image.
- **Image management**

  KTIB provides commands for pushing, pulling, listing, removing, and logging in to images.

Application Scenarios

- KTIB is applicable to scenarios requiring customized file systems, such as development environments for special applications, build environments demanding special dependencies, or production environments that must adhere to specific security policies.
- KTIB is suitable for enterprises and organizations that need to follow security and compliance standards. In development and production environments, KTIB ensures that Dockerfiles meet security and compliance requirements, reducing potential security vulnerabilities and configuration issues.
- KTIB is applicable to scenarios where frequent adjustments to the build steps are required during development. Single-step building enables developers to debug the image building process in detail and quickly identify and resolve issues.
- KTIB is well-suited to production environments and automated deployment scenarios.

NDK

NestOS Kubernetes deployer (NKD) is a Kubernetes cluster deployment tool for cloud container scenarios, covering various aspects such as infrastructure creation, cluster deployment, and configuration management.

Feature Description

- NDK automates infrastructure and Kubernetes cluster deployment and destruction as well as node scaling for common OSs in libvirt and OpenStack virtualization environments.
- NDK automates OS installation and Kubernetes cluster deployment as well as node scaling for common OSs in bare-metal environments.
- NDK supports containerd, CRI-O, Docker, and iSulad container runtimes.

Application Scenarios

NKD is suitable for rapid deployment of container platforms within enterprises, seamlessly supporting application development and management of microservice architectures. It enables software development teams to efficiently build development and testing environments, deeply integrate CI/CD processes, and significantly accelerate application iteration cycles. Additionally, it offers a manageable Kubernetes cluster management solution tailored for universities and research institutions, meeting the needs of teaching and research projects.

QSemOS-plugin

QSemOS-plugin is a set of plugins developed by the National Center of Technology Innovation for embedded OS development scenarios, including three plugins for soft real time, hard real time, and hybrid deployment. For enterprise and individual developers, the full-process toolchain includes project creation, development, compilation, debugging, simulation, and burning, helping developers quickly build customized development environments and simplify development processes.

Feature Description

- **Embedded plugin for soft real time**

  This plugin is a visualized development tool that aids in developing embedded soft real-time OS kernels, Yocto projects, and applications. It allows for the creation of a development environment for embedded soft real-time OSs (comprising the Linux kernel, bootloader, rootfs, and openEuler Yocto) and applications (C and C++). Additionally, users can combine various tools based on their specific scenarios to form a guided, visualized integrated development toolchain, enhancing development efficiency.

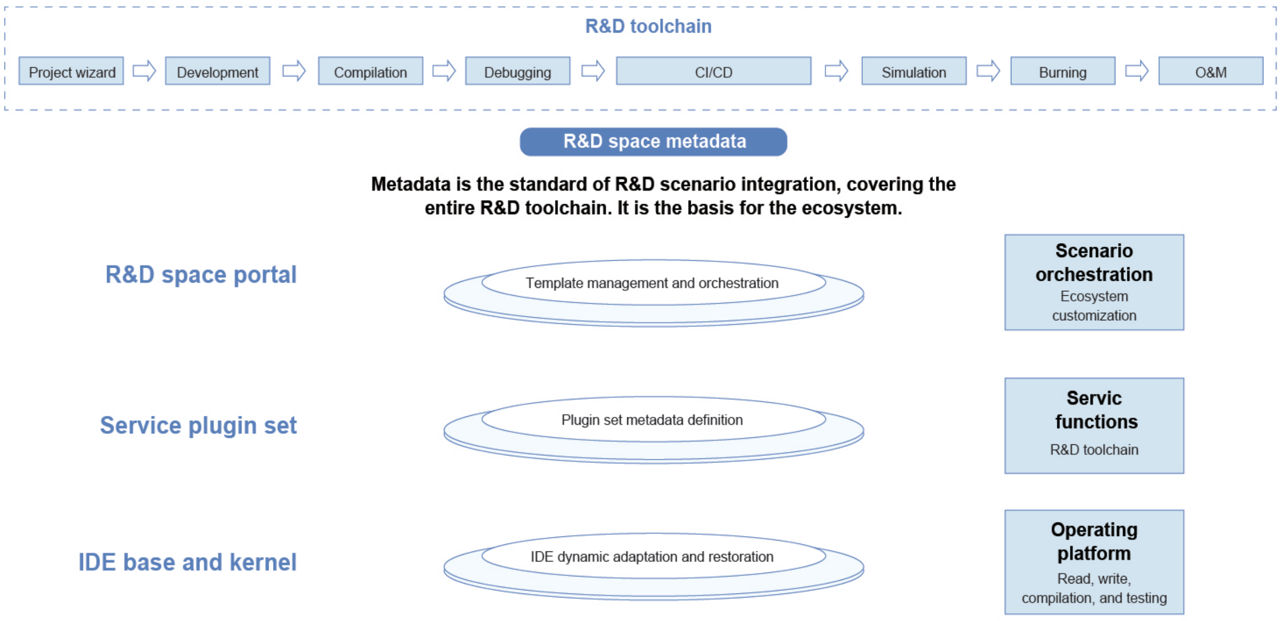
- **Embedded plugin for hybrid deployment**

  This plugin primarily facilitates OS deployment. To quickly deploy virtualized environments based on Jailhouse and Xvisor frameworks, the plugin prepares host and guest configuration files and their corresponding image files before the deployment, available to users through an IDE.

- **Embedded plugin for hard real time**

  This plugin presents a graphical user interface (GUI) that provides compilation, burning, and debugging of UniProton and Zephyr real-time OSs. This combination of toolchain and service demands optimizes development efficiency.

Vertical service framework of the embedded R&D space


R&D toolchain
Project wizard
Development
Compilation
Debugging
CI/CD
Simulation
Burning
O&M
R&D space metadata
Metadata is the standard of R&D scenario integration, covering the entire R&D toolchain. It is the basis for the ecosystem.
R&D space portal
Template management and orchestration
Scenario orchestration
Ecosystem customization
Service plugin set
Plugin set metadata definition
Servic functions
R&D toolchain
IDE base and kernel
IDE dynamic adaptation and restoration
Operating platform
Read, write, compilation, and testing

Application Scenarios

QSemOS-plugin is an embedded OS development toolchain for robots, industrial control systems, and high-end CNC machine tools, providing a full suite of tools for development, security, testing, verification, migration, tuning, and O&M management related to OSs in the industrial field.

08 Copyright Statement

Trademark 09

10 Appendixes

Appendix 1: Setting Up the Development Environment

Environment Setup	URL
Downloading and installing openEuler	https://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 Disclosure

Security Issue Disclosure	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