Cloud Container Engine

Skill Reference

Issue	01
Date	2026-06-05

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: https://www.huaweicloud.com/intl/en-us/

Contents

1 Using Huawei Cloud Cloud-Native Skills

NOTE

This section is intended for developers, O&M engineers, and architects who use Cloud Container Engine (CCE) and related cloud services. It describes the capacity positioning, usage, and reference of Huawei Cloud cloud-native Skills.

Skill Overview

What Are Skills?

Skills are open capabilities that convert professional knowledge, operation processes, and best practices into reusable capability units. In AI Agents, Skills are used to extend the professional capabilities of Agents so that Agents can automatically execute complex tasks in specific domains based on predefined processes and rules. The core features of Skills are as follows:

- Intent-driven: An Agent automatically understands when to trigger a Skill by reading the Skill's description. You do not need to explicitly specify the time.
- Scenario orchestration: A Skill can internally connect multiple steps to automatically collect contexts, analyze them, and output conclusions.
- Reusable: A Skill can run on different Agent platforms (web, CLI, and API). You do not need to adapt the Skill to each platform.
- Composable: Multiple Skills can be combined based on workflows. Agents automatically select and invoke appropriate Skills based on task requirements.
- Security guardrails: Risk constraints are defined within Skills. High-risk operations must be previewed and confirmed by users.

Huawei Cloud cloud-native Skills are encapsulated O&M capabilities of cloud services such as CCE, AOM, LTS, ELB, ECS, and HSS based on scenarios such as fault diagnosis, observability analysis, inspection and governance, and automatic recovery. They enable AI Agents to have professional cloud native O&M capabilities.

Scenarios

- Fault diagnosis: Pod CrashLoopBackOff, node NotReady, Ingress 502, PVC Pending, and other faults

- Observable analysis: AOM alarms, LTS logs, Kubernetes events, and pod/node metrics are aggregated to form diagnosis contexts.
- Inspection and governance: daily cluster health check, capacity trend prediction, cost optimization suggestions, and availability risk scanning
- Automatic recovery: controlled changes such as scaling, cordoning or draining nodes, restarting ECSs, and fixing HSS vulnerabilities
- Delivery solution: container migration planning, resource stocktaking, and dependency matrix analysis
- Cluster management: CCE cluster upgrade planning, workload management, and UCS cluster management and policy governance

Security Constraints and Risk Levels

- Core security constraints
  - Do not output AKs/SKs in scripts, logs, or reports.
  - Preview all operations, such as deletion, scaling, drain, and reboot, before confirming them.
  - Delete temporary kubeconfig files and certificate files after using them.
  - Use diagnosis, inspection, and migration planning Skills only for read-only queries and report generation.
- Risk levels

Risk Level	Example	Default Behavior
R0	list/get/query/ analyze	Direct execution
R1	Generating reports, solutions, and dashboards	Direct execution
R2	Restarting abnormal pods and providing suggestions after query	Preview by default, with configurable automatic execution
R3	Scaling, rollback, cordon, and uncordon	**confirm=true**
R4	Deleting, draining, and hibernating production clusters	**confirm=true** and strong risk warning
R5	Clearing data and performing irreversible cross-domain deletion	Forbidden by default

Usage Constraints

- Currently, Skills are mainly designed for CCE clusters and their associated cloud services (such as AOM, LTS, ELB, ECS, and HSS).
- All change actions are in preview mode by default and are not automatically executed.

Usage Description

Working Principles

A Skill works based on the intent matching mechanism. An Agent reads the description in the header of the **SKILL.md** file in the Skill directory. When your question matches the description, the Agent automatically triggers the Skill. A Skill defines a complete processing workflow, a list of tools that can be invoked, and risk constraints. The Agent executes tasks step by step based on the Skill's guidance.

For example, when you ask "What should I do if a pod keeps restarting?", the Agent matches the description of **pod-failure-diagnoser** as follows:

```
---
name: pod-failure-diagnoser
description: Diagnose CCE Pod failures such as CrashLoopBackOff, ImagePullBackOff, OOMKilled, Pending, Evicted, restart storms, or workload unavailable.
---
```

The Agent determines that the issue matches the description and automatically triggers **pod-failure-diagnoser** to execute the diagnosis process.

Obtaining Skills

Huawei Cloud cloud-native Skills are provided through an open repository at **GitHub**.

Each Skill uses a self-contained directory structure that contains the description and auxiliary files required to run the Skill.

```
skill-name/
├── SKILL.md       # Skill definition file, which is the only entry
├── references/    # Reference documents
├── scripts/       # Executable scripts
├── templates/     # Template files
└── demo/          # Demonstration examples
```

Skill Installation

- Method 1: Using npx

  ```
  # Install a single Skill.
  npx skills add huaweicloud/huaweicloud-skills --skill <skill-name>

  # Install all Skills.
  npx skills add huaweicloud/huaweicloud-skills
  ```

- Method 2: Using the GitHub repository for manual installation

  ```
  git clone https://github.com/huaweicloud/huaweicloud-skills.git

  # Install a specified Skill.
  npx skills add <path>/huaweicloud-skills/skills/<skill-name>
  ```

The loading paths and integration methods vary depending on the Agent platform. For details, see **Platform Integration Example**.

Authentication Configuration

Before using Skills related to Huawei Cloud products, configure authentication information based on the target cloud service.

- Interactive configuration
  ```
  Access Key Id: <your AK>
  Secret Access Key: <your SK>
  ```
- AccessKey authentication configuration using KooCLI
  ```
  hcloud configure set --cli-access-key="<your AK>" --cli-secret-key="<your SK>" --cli-mode="AKSK"
  ```

> **CAUTION**
>
> - Use plaintext AK/SK authentication only in the trusted local test environment to prevent credential leakage.
> - The cloud environment must comply with the principle of least privilege and follow the instructions provided in **Identity Authentication and Access Control**.
> - Do not write AKs/SKs into scripts, logs, reports, or code repositories.

Reference

Overview

Huawei Cloud cloud-native Skills are organized around cloud-native resource management and continuous O&M scenarios, covering capability domains such as resource lifecycle, observability and alarms, fault diagnosis and recovery, inspection and governance, solution and delivery, and multi-cloud and multi-cluster management.

Each Skill is provided as an independent directory, including the capability description, application scenarios, and necessary reference documents. You can select a single Skill or combine multiple Skills to complete cross-service and cross-step O&M tasks based on service requirements. The following lists available Skills by capability domain.

Lifecycle and Resource Management

Lifecycle and resource management covers CCE, CCI, and SWR. The product names are used only for grouping. Each row in the following table represents an independent Skill.

- CCE

Skill	Directory Path	Function
huawei-cloud-cce-cluster-management	skills/huawei-cloud-cce-cluster-management	Manages the full lifecycle of CCE clusters, node pools, nodes, add-ons, EIPs, and kubeconfig.

Skill	Directory Path	Function
cce-cluster-upgrade-planner	skills/cce/cce-cluster-upgrade-planner	Plans the CCE Kubernetes version upgrade and checks the upgrade path, add-on compatibility, different items, and upgrade window.
cce-workload-manager	skills/cce/cce-workload-manager	Manages CCE workloads and Kubernetes resources, including Deployments, StatefulSets, DaemonSets, jobs, CronJobs, HPAs, Services, ingresses, and ConfigMaps.

- CCI

Skill	Directory Path	Function
huawei-cloud-cci-instance-management	skills/cci/huawei-cloud-cci-instance-management	Manages CCI, including namespaces, networks, Deployments, StatefulSets, pods, EIPPools, logs, and metrics.

- SWR

Skill	Directory Path	Function
huawei-cloud-swr-image-management	skills/swr/huawei-cloud-swr-image-management	Manages SWR namespaces, repositories, tags, login credentials, and quotas.
huawei-cloud-swr-image-governance	skills/swr/huawei-cloud-swr-image-governance	Manages SWR permissions, retention policies, sharing policies, agencies, and immutability rules.
huawei-cloud-swr-image-automation	skills/swr/huawei-cloud-swr-image-automation	Manages SWR image synchronization, triggers, and automatic deployment processes.
huawei-cloud-swr-enterprise-instance	skills/swr/huawei-cloud-swr-enterprise-instance	Manages SWR Enterprise Edition, namespaces, repositories, artifacts, credentials, endpoints, and domain names.

Observability and Intelligent Alarms

Skill	Directory Path	Function
observability-context-builder	skills/observability-context-builder	Aggregates AOM alarms, metrics, LTS logs, pod logs, and Kubernetes events to form diagnosis contexts.
alarm-correlation-engine	skills/alarm-correlation-engine	Performs association analysis on AOM active and historical alarms, deduplicates and merges alarms, groups alarms by severity, and checks alarm rules.
log-analyzer	skills/log-analyzer	Queries and analyzes pod standard output, CCE LogConfig application logs, and LTS logs.
kubernetes-event-analyzer	skills/kubernetes-event-analyzer	Queries and analyzes Kubernetes warning events, repetition patterns, and pod, node, and workload exceptions.
metric-analyzer	skills/metric-analyzer	Queries and analyzes CCE pod, node, and ECS, ELB, EIP, and NAT metrics to identify threshold exceptions.

Fault Diagnosis and Self-Healing

Skill	Directory Path	Function
pod-failure-diagnoser	skills/pod-failure-diagnoser	Diagnoses pod faults such as CrashLoopBackOff, ImagePullBackOff, OOMKilled, Pending, Evicted, and frequent restarts.
workload-failure-diagnoser	skills/workload-failure-diagnoser	Diagnoses Deployment, StatefulSet, and DaemonSet release failures, rolling upgrade suspension, insufficient replicas, and probe exceptions.
node-failure-diagnoser	skills/node-failure-diagnoser	Diagnoses Node NotReady, resource pressure, NPD, CNI, kubelet, and container runtime exceptions.
autoscaling-diagnoser	skills/autoscaling-diagnoser	Diagnoses HPA and Cluster Autoscaler link faults.

Skill	Directory Path	Function
network-failure-diagnoser	skills/network-failure-diagnoser	Diagnoses Service, DNS, ingress, NetworkPolicy, ELB, EIP, NAT, and VPC network faults.
storage-failure-diagnoser	skills/storage-failure-diagnoser	Diagnoses PVC, PV, EVS, SFS, OBS, mounting, capacity, and deletion protection faults.
root-cause-analyzer	skills/root-cause-analyzer	Summarizes cross-domain evidence and outputs top root causes, impact scope, confidence, and recovery handover.
change-impact-analyzer	skills/change-impact-analyzer	Analyze the fault impacts caused by release, configuration, network, security policy, and node changes.
dependency-impact-analyzer	skills/dependency-impact-analyzer	Analyzes the fault propagation path and upstream and downstream impacts based on the Service, ingress, pod, and node topologies.
auto-remediation-runner	skills/auto-remediation-runner	Generates and executes controlled recovery actions. All high-risk changes are previewed by default and require explicit confirmation.

Inspection, Governance, and Continuous O&M

Skill	Directory Path	Function
daily-cluster-inspector	skills/daily-cluster-inspector	Performs periodic CCE health checks, quick inspections, and continuous O&M summaries.
availability-risk-scanner	skills/availability-risk-scanner	Scan for HA, AZ distribution, single replica, PDB, probe, affinity, gateway, and resource overcommitment risks.
capacity-trend-forecaster	skills/capacity-trend-forecaster	Analyzes periodic capacity trends, predicts resource bottlenecks, and simulates HPA and node scaling policies.
cost-optimization-advisor	skills/cost-optimization-advisor	Analyzes idle resources, excessive requests, low-usage nodes, and scaling policy optimization opportunities.

Skill	Directory Path	Function
ops-report-generator	skills/ops-report-generator	Summarizes inspection, capacity, availability, cost, and on-call contexts to generate weekly, monthly, SLA, capacity, and stability reports.

Solution and Delivery

Skill	Directory Path	Function
cce-cci-bursting-deployer	skills/cce-cci-bursting-deployer	Configures, deploys, and verifies the auto scaling capability from CCE to CCI 2.0, including VPCEP, virtual-kubelet, and smoke testing.
container-migration-planner	skills/container-migration-planner	Counts container platform resources and dependencies, and outputs migration batches, risks, and verification solutions. No real migration is performed.
Skill for full-link pressure test	skills/pressure-test	Builds a full-link pressure test from the k6 client through ELB and nginx-ingress to the service pod, collects observability data, and outputs a performance report.

Multi-Cloud and Multi-Cluster Management

UCS-related Skills are placed in this category and are no longer included in CCE lifecycle management.

Skill	Directory Path	Function
ucs-cluster-onboarding-manager	skills/ucs/ucs-cluster-onboarding-manager	Manages UCS clusters, lifecycle, fleet groups, kubeconfig, and resource quotas.
ucs-policy-governor	skills/ucs/ucs-policy-governor	Manages UCS policy instances, policy definitions, start and stop operations, execution statuses, and fleet compliance audit.

Usage

An Agent automatically matches capabilities based on the description in the **SKILL.md** file of each Skill. If manual locating is required, you can find the target Skill according to this document and then go to the corresponding directory to view the complete description and reference documents.

Platform Integration Example

Using Skills in OpenCode

OpenCode is an AI programming assistant for terminals. It allows you to load a Skill through the project directory or user directory.

- Skill types
  - Project-level Skill: Place the Skill directory in the **skills/** folder in the root directory of the project.

```
my-project/
├── src/
├── skills/
│   ├── pod-failure-diagnoser/
│   │   ├── SKILL.md
│   │   ├── manifest.json
│   │   ├── skill-profile.yaml
│   │   └── references/
│   ├── node-failure-diagnoser/
│   └── ...
```

    When OpenCode is started, it automatically scans the **skills/** folder in the project directory and loads all Skills. You can directly describe the issue in the dialog, and the Agent will automatically match the appropriate Skill based on the description.
  - User-level Skill: Place the Skill directory in the user configuration directory. User-level Skills take effect for all projects and are suitable for common O&M Skills.
    - Windows: **%USERPROFILE%\.opencode\skills**
    - Linux/macOS: **~/.opencode/skills/**
- Example

```
# Go to the project directory.
cd my-project

# Start OpenCode. Skills have been automatically loaded.
opencode

# Describe the issue in the dialog.
> My pod keeps restarting. Can you help me check?
# The Agent automatically triggers pod-failure-diagnoser.
```

Using Skills in OpenClaw

OpenClaw is an open-source, self-hosted gateway that connects chat applications and channels to AI Agents. You can run the gateway locally or on your own server and extend Agent capabilities through Skills.

OpenClaw can load Skills from the following directories:

Directory	Description
<workspace>/skills/	Skills in the current workspace, suitable for project-level customization
<workspace>/.agents/skills/	Project-level Skills for Agents in the current workspace
~/.agents/skills/	Skills that can be shared by multiple Agents
~/.openclaw/skills/	Skills managed by OpenClaw
skills.load.extraDirs	Skill directories that can be added through configuration

OpenClaw also loads the Skills that come with the installation. You can copy the required Skill directories to the corresponding loading directories. Example:

```
mkdir -p ~/.agents/skills
cp -R ./skills/pod-failure-diagnoser ~/.agents/skills/
cp -R ./skills/node-failure-diagnoser ~/.agents/skills/
```

Each Skill directory must contain **SKILL.md**. After OpenClaw loads Skills, the Agent can select an appropriate Skill based on your intent and execute tasks based on the workflow defined in the Skill.

For details about the positioning, Skill loading sequence, and directory description of OpenClaw, see the **OpenClaw documentation** and **OpenClaw Skills**.

Using Skills in Hermes

Hermes is a service orchestration platform for enterprise-class AI Agents. It supports Skill integration through declarative configuration.

Common Issues

When describing an issue, you can refer to the following table to quickly locate the recommended Skill.

Issue Description	Recommended Skill
Pod keeping restart, Pending, and OOMKilled	pod-failure-diagnoser
Release failure, rolling upgrade suspension, and insufficient replicas	workload-failure-diagnoser
Node NotReady, resource pressure, and node vulnerabilities	node-failure-diagnoser
HPA not scaling pods, CA not scaling nodes, and auto scaling not taking effect	autoscaling-diagnoser

Issue Description	Recommended Skill
Ingress 502, Service unreachable, ELB link exception	network-failure-diagnoser
PVC Pending, FailedMount, and capacity exhaustion	storage-failure-diagnoser
A large number of CCE alarms, which need to be combined for analysis	alarm-correlation-engine
Pod standard output or LTS application log query	log-analyzer
Kubernetes event trend analysis	kubernetes-event-analyzer
Query of CCE pod/node metrics and rankings by resource usage	metric-analyzer
Aggregation of logs, events, metrics, and alarms	observability-context-builder
Service unavailability, requiring comprehensive root cause analysis	root-cause-analyzer
Faults upon release, configuration, network, security policy, or node changes	change-impact-analyzer
Determining entries and upstream and downstream services affected by a service fault	dependency-impact-analyzer
Capacity expansion, restart, draining, and vulnerability fixing	auto-remediation-runner
Daily inspection or periodic health check	daily-cluster-inspector
Cost optimization and excessive request analysis	cost-optimization-advisor
Capacity trend prediction and scaling simulation	capacity-trend-forecaster
Availability risk scanning and PDB/probe check	availability-risk-scanner

Issue Description	Recommended Skill
Weekly, monthly, and SLA O&M reports	ops-report-generator
Container migration solution and resource stocktaking	container-migration-planner
Auto scaling configuration for scheduling CCE workloads to CCI	cce-cci-bursting-deployer
CCE cluster version upgrade planning	cce-cluster-upgrade-planner
CCE/UCS workload management	cce-workload-manager
UCS cluster management and fleet management	ucs-cluster-onboarding-manager
UCS policy governance and compliance audit	ucs-policy-governor
SWR image lifecycle management	huawei-cloud-swr-image-management
SWR image governance	huawei-cloud-swr-image-governance
SWR image automation	huawei-cloud-swr-image-automation
Pressure test solution and execution	Skill for full-link pressure test

Helpful Links

Document	Description	Path
CCE documentation	CCE documentation	**Huawei Cloud CCE Documentation**
Open Skill repository	Huawei Cloud cloud-native skill code repository	**huaweicloud/huaweicloud-skills**

2 Huawei Cloud Cloud-Native Skill Best Practices

2.1 Using AI CLI to Diagnose and Rectify CCE Workload Faults

Scenarios

In a CCE cluster, after workload release, scaling, or configuration change, there may be issues such as pod not ready for a long time, Deployment rolling upgrade suspended, frequent container restarts, image pull failures, scheduling failures, and service endpoint exceptions. Traditional troubleshooting usually requires O&M personnel to repeatedly switch between Deployments, ReplicaSets, pods, events, container logs, probe configurations, and monitoring data. The fault locating process is long and manual judgment is costly.

By combining AI CLI with cloud-native Skills, you can describe fault symptoms in natural language. The Agent automatically completes context identification, evidence collection, root cause analysis, recovery solution preview, user confirmation, recovery execution, and result verification. This helps O&M teams standardize workload troubleshooting into a repeatable process.

This practice applies to common scenarios of CCE workload fault diagnosis and recovery, and is not bound to a fixed namespace, workload name, or single test environment. In this document, **ai-diagnose-demo** is only a demo namespace. Replace it with your target cluster and service namespace.

Constraints

- The diagnosis accuracy of AI CLI depends on the cluster access permissions, log retention period, event integrity, and quality of observability data.
- All write operations must comply with the "preview + confirmation" mode. Rollback, restart, scaling, or configuration modification cannot be performed without confirmation.

- In the production environment, you are advised to separate diagnosis permissions from recovery permissions and record all AI CLI operations in audit logs.
- Rollback depends on the historical revisions of a workload. If a historical version has been cleared, restore it by re-releasing it or restoring its configurations.
- If faults involve database changes, data format changes, external dependencies, or message accumulation, evaluate data consistency and service compensation solutions before the rollback.
- In scenarios involving multiple clusters, namespaces, or abnormal objects, AI CLI should require users to confirm the target scope to avoid cross-service misoperations.
- Do not output sensitive information such as the AK/SK, token, certificate, and real project ID in prompts, diagnosis reports, or recovery previews.
- You are advised to verify the Skill process in a demonstration or test namespace to ensure the correctness and security of the process before promoting it to production clusters.

Prerequisites

- You have created a CCE cluster and have deployed the target workload.
- You have installed or accessed AI CLI, and have registered related Skills.
- AI CLI has permission to read workloads, pods, events, logs, version history, and server endpoints.
- To perform restoration, AI CLI must also have permission to roll back, restart, scale, or modify workload configurations.

Involved Skills

Skill	Function
huawei-cloud-cce-workload-failure-diagnoser	Collects workloads, pods, events, logs, and version history, and outputs diagnosis conclusions.
huawei-cloud-cce-auto-remediation-runner	Generates a recovery preview and performs actions such as rollback, restart, and scaling after user confirmation.

Procedure

Step 1: Start an AI CLI Session

Start the AI CLI based on the actual access mode of your enterprise. If you use the CLI, you can start an interactive session.

```
aicli chat
```

If AI CLI has been connected to the O&M platform, ChatOps tool, or pipeline, you can directly initiate a natural language request through the corresponding entry.

Step 2: Describe the Fault in Natural Language

You do not need to manually combine multiple Kubernetes commands. Instead, you need to describe the target cluster, namespace, and fault symptom. It is recommended that you specify whether recovery operations are allowed and whether a preview is required before recovery in the input. For example, the prompt could be as follows:

```
Help me diagnose the workload release exception in the ai-diagnose-demo namespace of the cce-ai-ops-demo cluster in CN North-Beijing4. The symptom is that the pod is not ready for a long time after the latest update. Please analyze the root cause and provide recovery suggestions. Before performing recovery operations, please let me confirm.
```

If you are not sure about the abnormal object, you can ask AI CLI to scan the namespace first.

```
Check the abnormal workloads in the ai-diagnose-demo namespace of the cce-ai-ops-demo cluster, find the objects that fail to be released or whose pods are not ready, and output the diagnosis conclusion and recovery suggestions.
```

Step 3: Confirm the Diagnosis Scope

AI CLI identifies the region, cluster, namespace, resource type, fault time window, and fault symptom based on your input. If there are multiple abnormal workloads in the namespace, AI CLI lists the candidate objects and exception summary for you to confirm the diagnosis scope.

Confirm the following information:

Information	Description
Target cluster	Clusters can be identified by cluster name, cluster ID, or region and cluster name.
Namespace	Used to limit the diagnosis scope to avoid cross-service misoperations.
Workload type	Common workloads such as Deployments, StatefulSets, and DaemonSets can be diagnosed.
Fault symptom	For example, the pod is not ready, the rolling upgrade is suspended, the container is restarted, the image fails to be pulled, or the scheduling fails.
Recovery boundary	Whether actions such as rollback, restart, scaling, or configuration modification are allowed.

Step 4: Automatically Collect Diagnosis Evidence

After AI CLI invokes the workload fault diagnosis Skill, evidence should be collected from the control plane to the data plane to avoid relying solely on single-point logs or a single event.

Diagnosis Dimension	Key Evidence	Diagnosis Value
Workload status	Desired replicas, ready replicas, available replicas, updated replicas, and status conditions	Check whether the release is complete and whether the availability is affected.
Version mapping	Revision history, ReplicaSet status, and replica distribution of old and new versions	Check whether a healthy historical version exists and whether the rollback conditions are met.
Pod lifecycle	Pending, Running, Ready, RestartCount, and container status	Check whether the fault occurs in the scheduling, startup, running, or ready phase.
Kubernetes events	Events such as FailedScheduling, FailedPull, Unhealthy, and BackOff	Quickly locate scheduling, image, probe, and container startup issues.
Container logs	Recent exception logs, startup logs, and health check logs	Identify internal application errors, dependency exceptions, or configuration issues.
Probe configurations	**readinessProbe**, **livenessProbe**, and **startupProbe** configurations and results	Check whether the health check path, port, protocol, and timeout configurations are proper.
Service endpoints	Service, EndpointSlice, ingress, or load balancing backend	Check whether the fault affects service traffic access.
Observability data	Metric, alarm, and log trends	Identify external factors such as resource pressure, abnormal traffic, and dependency jitter.

Step 5: Output the Diagnosis Conclusion and Recovery Suggestions

After the diagnosis is complete, AI CLI should output a structured conclusion to help you quickly determine whether recovery operations are required.

Recommended output:

Output Item	Content
Diagnosis conclusion	Specify the fault type, such as probe failure, image pull failure, container startup failure, scheduling failure, resource insufficiency, or configuration exception.
Impact scope	Describe the affected namespaces, workload types, number of unavailable replicas, and impact on service access.

Output Item	Content
Key evidence	List the status conditions, events, logs, or metrics that support the conclusion. Do not output sensitive information.
Cause analysis	Describe the phase in which the fault occurred in the release link and why the workload became unavailable.
Recovery suggestions	Provide one or more recovery options and describe the scenarios, risk levels, and expected effects.
Whether confirmation is required	Mark change actions, such as rollback, restart, scaling, and configuration modification, as requiring user confirmation.

Step 6: Preview the Recovery Solution

If you want AI CLI to continue rectifying the fault, AI CLI should invoke the automatic recovery Skill to generate a recovery preview. In the preview phase, only the plan is displayed, and the cluster status is not changed.

Common recovery actions:

Recovery Action	Scenario	Risk Control
Rolling back to a healthy revision	The new version is unavailable after release, but the old version can still stably carry services.	Verify the historical version, available replicas, and rollback impact scope.
Restarting an abnormal pod	A single pod enters the abnormal state, and no obvious configuration issue is found in the workload template.	Avoid insufficient available replicas caused by batch restart.
Temporary scale-out	The available replicas are insufficient, and the service capacity needs to be restored first.	Evaluate the resource quota, node capacity, and HPA policy.
Correcting the workload configuration	The probe, environment variable, image tag, Secret, or ConfigMap reference is incorrect.	Display configuration differences and confirm the change window and rollback method.
Suspending or resuming a release	The rolling upgrade is abnormal. You need to prevent the impact from expanding or continue the release.	Specify the release status and subsequent manual actions after the suspension or resumption.

The recovery preview should include the following information:

- Actions to be performed and the scope of target resources
- Key configuration or status differences before and after a change
- Risk level, service impact, and whether a change window is required
- Verification method and rollback path
- Specific statements that a user needs to confirm

Step 7: Confirm and Execute the Recovery

AI CLI can invoke the automatic recovery Skill to perform actions only after you confirm the recovery plan. You are advised to use clear expressions. For example:

```
Confirm that the recovery is performed based on the preview solution.
```

During the execution, AI CLI should continuously report the change status. If the recovery fails, subsequent changes should be stopped, and the failure cause, actions that have been performed, current cluster status, and recommended manual handling methods should be provided.

Step 8: Verify Recovery Results

After the recovery is complete, AI CLI needs to read the workload status and related evidence again to confirm whether the fault is actually rectified.

Recommended verification items:

Verification Item	Expected Result
Workload status	The desired replicas, ready replicas, and available replicas meet expectations, and the release status is stable.
Pod status	The pod is in the **Running** and **Ready** state, and there is no continuous restart, pull failure, or scheduling failure.
Events and logs	No similar high-frequency abnormal events occur, and no new key errors are recorded in container logs.
Service endpoints	The Service or EndpointSlice has available backends, and service traffic can be correctly forwarded.
Service detection	If health check or external detection has been configured, the detection result is normal.
Alarm status	Related alarms are cleared or enter the convergence state, and no new high-risk alarms are triggered.

Skill Execution Process

Phase	Input	Skill Action	Output
Target Identification	Region, cluster, namespace, and fault symptom in natural language	Parse the diagnosis scope, supplement missing information, and request user confirmation if necessary.	Clear diagnosis objectives and boundaries
Evidence collection	Target workload and time window	Query workloads, pods, events, logs, version history, and server endpoints.	Multi-dimensional diagnosis evidence
Cause analysis	Diagnosis evidence	Identify the fault phase, eliminate mismatched causes, and output confidence.	Root cause conclusion and key evidence
Recovery planning	Root cause, impact scope, and permission boundary	Generate a preview of rollback, restart, scaling, or configuration restoration.	Recovery plan, risk level, and verification method
User confirmation	User confirmation statement	Check whether the confirmed content matches the preview plan.	Executable recovery task
Recovery execution	Confirmed recovery task	Invoke Kubernetes APIs or CCE APIs to perform changes.	Execution result and intermediate status
Recovery verification	Workload status after recovery	Review status, events, logs, endpoints, and service probes.	Final conclusion and follow-up suggestions

Expected Results

After completing this practice, you can use AI CLI to complete the following closed-loop operations through natural language dialogs:

1. Identify the target CCE cluster, namespace, and abnormal workload.
2. Automatically aggregate evidence such as workload status, pod status, version history, events, logs, probes, and service endpoints.
3. Output explainable root cause analysis instead of just providing the result of a single command.
4. Preview the recovery actions to clarify the risks, impact scope, and verification methods.

5. Execute controlled recovery actions after user confirmation.
6. Automatically verify the recovery result and provide subsequent rectification suggestions.

Follow-up Suggestions

To improve the efficiency of troubleshooting CCE workload faults, you are advised to do the following based on this practice:

- Standardize prompts.

  Standardize the diagnosis prompt template in the team's SOP to specify the cluster, namespace, fault symptom, time window, and recovery boundary.
- Complete the pre-release check.

  Add image startup check, probe path verification, configuration reference verification, and basic smoke tests to the CI/CD pipeline.
- Configure proper release policies.

  Configure proper rolling update policies, PDBs, HPAs, and the number of revisions to be retained for key services to reduce the impact of release exceptions on services.
- Enhance observability.

  Access logs, metrics, events, and alarms so that AI CLI can determine root causes from more dimensions.
- Establish change audit.

  Record the operator, confirmation statement, execution action, execution result, and recovery verification conclusion of the recovery action triggered by AI CLI to facilitate review and compliance audit.

Helpful Links

- **Containerized Application Management**
- **Kubernetes Deployments**
- **Kubernetes Pod Lifecycle**

2.2 Using OpenClaw to Perform Periodic Inspection on CCE Clusters

Scenarios

In the production environment, you need to continuously monitor the node health, pod status, core add-ons, resource usages, Kubernetes events, AOM alarms, and service ingress statuses of CCE clusters. By interconnecting OpenClaw Agent with cluster inspection, you can configure periodic inspection tasks using natural language, enabling the Agent to automatically perform cluster health check, aggregate alarms, analyze exceptions, classify risks, generate reports, and send notifications.

In this practice, you are advised to perform a quick inspection first and then a deep inspection if any exception is detected.

- When the cluster is normal, the Agent outputs a concise health summary to reduce invalid noise.
- If an exception is detected during the quick inspection, the Agent automatically extends the inspection to dimensions such as the pod, node, event, AOM, ELB, and resource usage.
- The Agent queries AOM alarms generated in the last 24 hours and aggregates them by alarm type, severity, current status, and repetition frequency. It distinguishes active alarms, cleared alarms, burst alarms, and recurring alarms.
- An in-depth inspection supplements top *N* historical pod metrics and top *N* node CPU, memory, and disk metrics, helping determine whether exceptions are related to resource watermarks.
- AI grades risks based on inspection evidence and displays the impact scope, possible causes, and suggestions for next step in the report. The grading result is used for summary and suggestions. The tool is not required to return fixed fields.
- During the inspection, only read-only queries and report generation are performed. Change actions such as scale-out, deletion, restart, and drain are not automatically executed.

OpenClaw Agent can be used to:

- Perform scheduled CCE cluster inspection every day or week.
- Automatically generate inspection reports in Markdown and HTML formats.
- Push the inspection summary and report link to the O&M team via email.
- Archive historical inspection reports for trend comparison and review.
- When detecting a major risk, transfer the risk to related diagnosis capabilities for in-depth analysis.

Constraints

- During the inspection, only read-only queries and report generation are performed. No automatic rectification actions are taken.
- During the execution of an inspection task, APIs of cloud services such as CCE, AOM, LTS, and ELB are invoked, which may incur small expenditures for invoking APIs or log query costs.
- Reports are stored in OBS, which will incur storage expenditures.
- The email sending frequency is limited by the quota of SMTP or Huawei Cloud SES. You are advised to set a proper inspection frequency.
- Do not write AKs/SKs, tokens, certificates, or real project IDs into documents, code, or dialog output.

Precautions

- An in-depth inspection collects more metrics and context, including AOM alarms in the last 24 hours, top *N* historical pod metrics, and top *N* node CPU, memory, and disk metrics. The execution time may be significantly longer than that of a quick inspection.
- Top *N* pod resources are queried based on the historical metric time window. The query result may contain pods that existed in the query time window but

do not exist now. You can check whether the object still exists based on the current pod list.

- Risk levels are generated by AI based on the factual evidence returned by the tool. You are advised to view the associated events, logs, and metrics before deciding whether to proceed with the recovery process.

Prerequisites

- You have created a CCE cluster, and its status is **Running**.
- You have enabled the OpenClaw service and have initialized the Agent.
- You have connected the Agent to Huawei Cloud cloud-native capabilities.
- You have installed the Cloud Native Cluster Monitoring add-on in the target CCE cluster.
- You have configured AOM alarm rules for the target CCE cluster based on best practices. For details, see **Using AI CLI to Configure, Query, and Manage CCE AOM Alarms**.
- You have configured Huawei Cloud access credentials. You are advised to use OpenClaw key management or environment variable injection to avoid exposing AKs/SKs in documents, scripts, or dialogs.
- The inspection account has read-only permissions on CCE, AOM, LTS, ELB, and other related resources.
- If email notifications are required, you have prepared SMTP or Huawei Cloud SES.
- If you need to archive reports, you have prepared an OBS bucket or another storage location for reports.

Recommended Input

You can directly describe the target cluster, inspection period, inspection scope, and notification method in the OpenClaw dialog.

Scenarios and recommended input

Scenario	Recommended Input
Creating a daily inspection task	Create a daily inspection task for the test-ai-diagnoses cluster in CN North-Beijing4. The task should be executed at 9:00 a.m. every day. Perform a quick inspection first. If any exception is detected, perform an in-depth inspection. Send the report to the O&M team.
Executing the inspection task immediately	Execute the daily inspection task for the test-ai-diagnoses cluster in CN North-Beijing4 immediately. Perform quick check first. If any exception is detected, perform in-depth diagnosis.
Viewing the latest report	View the latest inspection report of the test-ai-diagnoses cluster and list the risks by severity.

Scenario	Recommended Input
Analyzing the trend of the last 7 days	Summarize the inspection results of the test-ai-diagnoses cluster in the last 7 days and tell me whether the number of risks has increased.
Performing in-depth analysis on exceptions	Continue to analyze the high-risk node issues in the inspection report and associate events, metrics, and related pods.

You are advised to focus on the following items in the inspection result:

Output	Focus
Inspection result	Whether the cluster is healthy and whether there are high-risk exceptions.
Exception group	Whether exceptions are concentrated on pods, nodes, events, AOM, ELB, or resources.
Impact scope	Which namespaces, nodes, workloads, or service entries are affected.
Risk trend	Whether issues are added, expanded, or resolved compared with the previous day or the last seven days.
Recommended action	Continue to observe, enter special diagnosis, perform scale-out evaluation, optimize rules, or transfer to the recovery process.

Procedure

Step 1: Create a Periodic Inspection Task for a CCE Cluster

Make the Agent create a periodic inspection task for a CCE cluster. The Agent will identify the region, cluster name, inspection time, report format, and notification method based on the input and generate an inspection plan.

1. Enter the following content in the OpenClaw dialog:
   Create a daily inspection task for the test-ai-diagnoses cluster in CN North-Beijing4. The task should be executed at 9:00 a.m. every day. Perform a quick inspection first. If any exception is detected, perform an in-depth inspection. Generate Markdown and HTML inspection reports and send them to ops-team@company.com.
2. The Agent automatically generates an inspection plan. Confirm the following information:

Configuration Item	Example	Description
Region	cn-north-4	Region where the target CCE cluster is located
Cluster name	test-ai-diagnoses	CCE cluster to be inspected
Inspection time	09:00 every day	Off-peak hours or before shift handover is recommended.
Inspection policy	Quick inspection first, then in-depth inspection if exceptions are found	Reduce unnecessary heavy checks by default.
Report format	Markdown and HTML	Easy to read and archive via email.
Recipient	ops-team@company.com	Used for receiving the inspection summary and report link.
Storage location	obs://your-bucket/reports/	Used for saving historical inspection reports.

3. After the task is generated, the Agent performs the inspection as planned. You are advised to trigger an inspection immediately to verify the configuration.
   Perform an inspection on the test-ai-diagnoses cluster immediately and send the test report.

Step 2: View the Inspection Report and Identify Risks

1. After the inspection is complete, the Agent generates an inspection summary and a complete report. You can directly view the latest inspection result.
   View the latest inspection report of the test-ai-diagnoses cluster and list the risks by severity.
2. The Agent returns the inspection summary first.

Inspection Item	Example Result	Focus
Inspection result	Warning	Whether the cluster has risks that need to be handled
Check item quantity	12 items: 9 passed, 2 warnings, and 1 failed	Whether there are new risks
Node health	3/3 nodes are normal.	Whether there are NotReady, resource pressure, or node events
Pod statuses	Two pods are abnormal.	Whether there are CrashLoopBackOff, Pending, or Evicted pods

Inspection Item	Example Result	Focus
AOM alarms	140 alarms in the last 24 hours, 4 of which are not cleared	Whether there are continuous, burst, or recurring alarms
Core add-ons	Normal	Whether CoreDNS, network, and storage add-ons are healthy
Top *N* pod resources	Top *N* CPU/memory metrics in the last 24 hours	Whether there are metrics of historical high-watermark pods or disappeared pods
Top *N* node resources	Top *N* node CPU/ memory/disk metrics	Whether there are risks in the node CPU, memory, and disk capacity

3. If an exception is detected during the inspection, the Agent determines the severity based on the evidence returned by the tool and outputs a list of issues.

Severity	Type	Resource	Issue	Evidence	Suggestion
High	Pod health	default/test	Available replicas: 0	The Deployment expects two replicas, but the number of ready replicas is 0. Related pods are in the **Pending** state.	Check pod events and image pull status, and restore workload availability first.
High	Node resources	192.168.32.2	Node CPU remaining high	Node CPU usage is 100%, and remains high in the last 24 hours.	Locate the high-CPU process or pod on the node, and evaluate migration or scale-out if necessary.
Medium	AOM alarms	default/test-*	Repeated image pull failure alarms	FailedPullImage and BackOffPullImage alarms occurred in the last 24 hours, and some alarms are not cleared.	Correct the image path or tag, and trigger the rolling update of the Deployment again.

4. For major or recurring issues, you can make the Agent continue the analysis.
   Continue to analyze the unavailability of the default/test replica, and associate AOM alarms, Kubernetes events, pod statuses, and related metrics generated in the last 24 hours.

The Agent will continue to aggregate the context based on the inspection report and output root cause clues, impact scope, and suggestions for next step. For example, for major alarms or high-risk resource exceptions, you can make the Agent further analyze the related pods, node metrics, events, and logs within the corresponding time window to determine whether the exceptions share the same root cause.

Step 3: View Historical Trends and Archive Reports

The value of a periodic inspection is not only to detect exceptions on the current day, but also to observe whether risks persist, escalate, or recover. You can make the Agent summarize the inspection results over a period of time.

```
Summarize the inspection results of the test-ai-diagnoses cluster in the last 7 days and list the changes and new issues of high, medium, and low risks by date.
```

The Agent can output a trend summary.

Date	Execution Status	Total Check Items	High Risks	Medium Risks	Low Risks	New Issues	Remarks
2026-05-31	Successful	12	1	2	1	1	Pod restart issues added
2026-05-30	Successful	12	1	1	1	0	Continuous node memory pressure
2026-05-29	Successful	12	2	2	2	2	Core add-on exceptions
2026-05-28	Successful	12	0	0	0	0	Cluster health

You can also view the report archive location.

```
The Markdown and HTML links of the inspection reports for the test-ai-diagnoses cluster in the last seven days are listed.
```

The report is expected to retain the following content.

Report Content	Description
Inspection summary	Overall cluster status, number of check items, and number of high/medium/low risks
Exception list	Displayed by pod, node, event, AOM, ELB, and resource

Report Content	Description
Risk trend	Comparison with the previous inspection or the trend in the last seven days
Root cause clues	Providing entries to related logs, events, and metrics for major exceptions
Recommended action	Continue to observe, enter special diagnosis, perform capacity evaluation, or transfer to the recovery process.

Expected Results

After completing this practice, you can use OpenClaw Agent to complete the following closed-loop operations:

1. Automatically perform a periodic inspection for the target CCE cluster.
2. Perform a quick inspection first by default. If any exception is detected, perform the in-depth diagnosis or parallel inspection.
3. Summarize exceptions by pod, node, event, AOM, ELB, and resource.
4. Make AI mark the risk severity and impact scope based on the inspection evidence.
5. Automatically generate Markdown and HTML inspection reports and push them to the O&M team via email.
6. Archive historical inspection reports and support daily viewing and trend comparison.
7. For major or persistent risks, continue to collaborate with logs, events, metrics, and related diagnosis capabilities for root cause analysis.

FAQs

Is In-Depth Inspection Required for Each Inspection?

Not recommended. It is recommended that a quick inspection be performed first by default. If any exception is detected, the in-depth diagnosis or parallel inspection is performed. This reduces unnecessary API invocations, log queries, and report noise, improving inspection efficiency.

Will High-Risk Issues Found in the Inspection Be Automatically Rectified?

No. In this practice, OpenClaw Agent is only used for inspection and report generation and does not perform rectification actions. If rectification is required, the Agent can transfer the corresponding diagnosis or rectification capability and confirm the action before any change is made.

Why Are Pods That Are Currently Inaccessible Displayed in the Top *N* Pod Resources?

Top *N* pod resources are used to analyze resource usages in a historical time window. By default, historical metrics of the last 24 hours are queried. Therefore,

pods that have been deleted or rebuilt may be displayed. You can make the Agent continue querying the current pod list and describe historical metric objects and current inventory objects separately to better understand resource usages.

Why Cannot I Receive the Email?

You are advised to check the email recipient, SMTP or SES configuration, sending records, email service quota, and enterprise email interception policy. If a report has been generated but the email fails to be sent, you can view the report on the OpenClaw console or in the report archive path.

How Long Should Historical Reports Be Retained?

You are advised to retain inspection reports of a production cluster for at least 30 days. If you need to perform monthly stability review, SLA statistics, or capacity trend analysis, you can retain the reports for 90 days or longer and configure OBS lifecycle policies to control storage costs. This ensures that historical data can be quickly accessed and storage costs are managed efficiently.

Helpful Links

- **Cloud Container Engine (CCE)**: Learn how to query CCE clusters, nodes, workloads, add-ons, and O&M products.
- **Application Operations Management (AOM)**: Learn how to query AOM metrics, alarms, logs, and application O&M description.

2.3 Using AI CLI to Configure, Query, and Manage CCE AOM Alarms

Scenarios

After a CCE cluster is brought online, the O&M team usually needs to initialize alarm rules as soon as possible and continuously pay attention to active alarms, historical alarms, recovery records, and notification rules during routine troubleshooting. By interconnecting AI CLI Agent with the **alarm-correlation-engine** Skill, you can use natural language to configure CCE AOM alarm rules, query alarms, aggregate and analyze alarms, and trace the root causes of major alarms.

Compared with viewing a single alarm list, this practice recommends that you analyze cluster alarms by time window. AI CLI automatically merges alarms of the same type, marks alarm severities, and distinguishes between normal alarms, unexpected alarms, and alarms that persist. This helps customers quickly determine which alarms are the most urgent, which resources are affected, and what to check next.

AI CLI can be used to:

- Create recommended rules in the AOM alarm center for a specified CCE cluster in one click.
- Automatically create a default cluster-level notification rule if no notification rule is specified.

- Preview the number of rules, rule types, notification methods, and missing parameters before the rule creation. Create rules after customer confirmation.
- Automatically query the alarm rule list to confirm that 50 rules have been created.
- Query active alarms, historical alarms, and clearance records by cluster and time window.
- Deduplicate, group, and mark the severity of alarms, identify burst, persistent, and normal alarms, and analyze root cause clues.

This practice uses the **test-ai-diagnoses** cluster in CN North-Beijing4 as an example to demonstrate how to configure, query, and aggregate and analyze alarms using natural language. In this example, CCE AOM alarm rules are configured in batches for the cluster, and the alarm situation in the last four hours is analyzed.

Constraints

- You do not need to specify an existing notification rule. If no notification rule is specified, AI CLI automatically invokes the default cluster-level notification rule.
- When a notification rule is automatically created, an available SMN topic name or topic URN must be provided.
- Metric alarms depend on the AOM CCE Prometheus instance associated with the target cluster.
- Event alarms depend on the CCE event reporting link. You are advised to ensure that the log collection and node fault detection add-ons are normal.
- Alarm rules can be created only after being confirmed by the target customer.
- Severity marking is only an auxiliary tool for troubleshooting and cannot replace the production change approval and manual confirmation mechanisms.
- You are advised to explicitly specify a time window for alarm analysis, such as the last 30 minutes, last 4 hours, or a specific start and end time.
- Do not write AKs/SKs, tokens, certificates, or real project IDs into documents, code, or dialog output.

Prerequisites

- You have registered the **alarm-correlation-engine** Skill in AI CLI.
- You have configured Huawei Cloud access credentials. You are advised to inject them through environment variables or security credential files to avoid exposing AKs/SKs in dialogs or documents.
- You have associated an AOM CCE Prometheus instance with the target CCE cluster.
- You have prepared an available SMN topic, for example, topic named **test**. If an AOM notification rule already exists, you can reuse it.
- The execution account has the permissions to query and create AOM alarm rules, and query and create notification rules.
- The creation of alarm rules has been confirmed by the target customer.

Involved Skills

Skill	Function
alarm-correlation-engine	Queries AOM alarms and active alarms, merges and analyzes alarms, creates alarm rules, and queries notification rules.
huawei-cloud-cce-cluster-management	Queries the CCE cluster list and confirms the cluster name, ID, and status.
observability-context-builder	Continues to aggregate metrics, logs, and Kubernetes event contexts after alarm analysis.

Procedure

Step 1: Create CCE Cluster Alarm Rules in One Click

You can use AI CLI to intelligently configure default AOM alarm rules for the target cluster based on the CCE cluster alarm best practices. You only need to specify the region, cluster name, and notification topic. AI CLI will automatically understand the configuration intent, supplement cluster information, generate a creation plan, and complete rule creation and result verification after your confirmation.

Enter the following in the OpenClaw dialog box:

```
Help me create CCE AOM alarm rules in batches for the test-ai-diagnoses cluster in CN North-Beijing4. Use the test topic for subscription.
```

AI CLI generates an alarm rule configuration plan to help confirm the creation scope and notification method.

Intelligent Processing Item	Description
Identifying a cluster	AI CLI queries a CCE cluster by region and cluster name, and checks the cluster ID, status, and configurability.
Supplementing notification configuration	AI CLI preferentially reuses existing AOM notification rules. If no notification rule is specified, it creates a default cluster-level notification rule based on the SMN topic.
Generating a rule plan	AI CLI generates 50 default alarm rules based on the recommended practices for CCE clusters, distinguishing between metric alarms and event alarms.
Previewing impact scope	AI CLI displays the number of rules, rule types, notification rules, and missing parameters. No rule is directly created.

Intelligent Processing Item	Description
Waiting for customer confirmation	AI CLI creates rules only after the customer confirms them.
Automatically creating rules	AI CLI creates default alarm rules in batches. If a rule with the same name already exists, it automatically skips the creation.
Automatically performing result acceptance	After the rule creation is complete, AI CLI queries the alarm rule list immediately to confirm the number of rules, rule types, and number of failed rules.

The 50 rules created by default include:

Type	Quantity	Example
Prometheus metric alarms	38	Abnormal pod status, frequent pod restarts, node CPU usage, node disk availability, and abnormal node kubelet
CCE event alarms	12	Pod memory OOM, insufficient node disk space, abnormal node status, node scale-out timeout, and unavailable cluster

After AI CLI displays the preview, you need to confirm that the number of rules, notification method, and target cluster are correct, and then reply:

```
Start the creation.
```

After the execution is complete, AI CLI returns the creation result and automatically queries and confirms that the rules have been created. Focus on the following acceptance items:

Check Item	Expected Result
Total number of rules for **test-ai-diagnoses**	50
Metric alarms	38
Event alarms	12
Creation failed	0

If some rules already exist, AI CLI skips the rules with the same names to avoid duplicate creation. You can also enter the following content at any time to check:

```
Query the alarm rules of the test-ai-diagnoses cluster in CN North-Beijing4.
```

Step 2: Aggregate and Analyze Alarms by Time Window and Trace Root Causes

To check whether a cluster has risks, you are advised to specify a time window so that AI CLI can query active alarms, historical alarms, and clearance records at the same time, and aggregate and analyze alarms by severity.

```
Analyze the AOM alarms of the test-ai-diagnoses cluster in CN North-Beijing4 in the last 4 hours, aggregate the alarms by severity, mark normal and burst alarms, and tell me the three issues that need to be handled first.
```

You can also specify an exact time range to analyze the change window, fault window, or shift handover window.

```
Analyze the AOM alarms of the test-ai-diagnoses cluster in CN North-Beijing4 from 09:00 to 12:00 today, aggregate the alarms by severity, and output the first occurrence time and handling priority of burst alarms.
```

AI CLI returns the alarm situation summary of the current time window. Pay attention to the following information:

Output Item	Example
Total alarms	42 alarms detected in the last 4 hours, 7 of which are not cleared.
Alarms by severity	Critical: 1, High: 3, Medium: 9, Low: 29
Alarms by type	3 groups of burst alarms, 2 groups of uncleared alarms, and 5 groups of frequent alarms
First occurrence time	The earliest burst alarm occurred at 09:17, and the burst occurred from 09:20 to 09:35.
Major affected objects	kube-system, default/nginx-demo, and node 192.168.0.12
Handling priority suggestion	The current risks are mainly related to node resource pressure and service pod restart. You are advised to first analyze the recent changes of node disks and pods.

Then, AI CLI aggregates multiple original alarms into alarm groups and sorts them by severity. You can handle the alarms in the following sequence:

Priority	Severity	Alarm Group	Alarm Characteristic	First Occurrence Time	Judgment	Handling Suggestion
P0	Critical	Unavailable cluster or abnormal core component	Burst alarms, not cleared	09:17	The cluster control plane or service scheduling capability may be affected.	Analyze the root cause immediately and check CCE events, core component pods, and AOM metrics.
P1	High	Node disk space insufficient, associated with multiple pod eviction events	Persistent alarms, not cleared	09:24	Node resource pressure may cause unstable service replicas.	Check the pods, eviction events, and disk metrics on the associated node.
P2	Medium	Frequent pod restarts, mainly in default/ nginx-demo	Burst alarms, some cleared	10:06	Possibly related to recent releases, probe configurations, or resource limits	Query pod logs, events, and Deployment version history.
P3	Low	Short-term CPU threshold alarms	Frequent alarms, automatically cleared	Multiple occurrences in the last seven days	No continuous impact. The traffic may fluctuate for a short period of time.	Observe the trend and adjust the threshold or HPA policy if necessary.

For critical, high, or user-concerned alarm groups, you can continue to use AI CLI to analyze the root causes in the same time window.

```
Continue to analyze P1 warning. Keep the time window to the last 4 hours. Help me associate the pods on this node, recent events, related metrics, and possible root causes.
```

AI CLI continues to query the context of the alarm group and provides root cause clues.

Root Cause Analysis Item	Available Information
Related resources	Alarm node, affected pods, namespaces, workloads, and Services
Related events	Events such as eviction, scheduling failure, probe failure, image pull failure, and node exception
Related metrics	Trends of CPU, memory, disk, and network resources, and changes before and after the alarm is triggered
Timeline	First occurrence time, burst time, recovery time, and related event occurrence time of the alarm
Preliminary root cause	For example, node disk pressure, workload release exception, core component exception, or insufficient capacity
Suggestions for the next step	Continue troubleshooting, clear resources, scale out, adjust thresholds, roll back the release, or keep observing.

You can also ask AI CLI to output only critical and uncleared alarm groups:

```
View only the high- and critical-severity alarms that are not cleared in the last 4 hours, group them by affected resource, and provide the first occurrence time and root cause analysis suggestions for each group.
```

If the alarm is related to service unavailability, you can ask AI CLI to output a complete root cause analysis and recovery suggestions:

```
Continue root cause analysis based on these critical alarms, keep the time window from 09:00 to 12:00, and provide recovery suggestions.
```

To view the original details, enter the following:

```
Display the alarm details of the last 4 hours, including the alarm name, status, severity, resource, first occurrence time, and description.
```

Expected Results

After completing this practice, you can use AI CLI to complete the following closed-loop operations:

- Create 50 AOM alarm rules for the target cluster in one click based on the CCE cluster alarm best practices.
- Automatically identify the target cluster, prepare notification rules, preview the creation plan, and execute the creation after user confirmation.
- Automatically query the alarm rule list to confirm that rules for 38 metric alarms, 12 event alarms, and 0 failed alarms have been created.
- Query active alarms, historical alarms, and clearance records by cluster and time window.
- Automatically aggregate, deduplicate, and mark the severity of alarms to distinguish between normal alarms, burst alarms, and alarms that remain uncleared.
- Output the first occurrence time, burst time, affected resources, and handling queue sorted by priority.

- For critical and high-severity alarms, associate events, logs, metrics, and resource statuses within the same time window, and output possible root causes and the next diagnosis path.

FAQs

Prometheus Instance Not Found

- Symptom

  Example error:

  ```
  {
    "success": false,
    "error": "The Prometheus instance corresponding to the target cluster is not found."
  }
  ```

- Handling suggestion

  a. Query the AOM Prometheus instance and check whether a CCE instance is associated with the target cluster.

  b. Query the CCE add-on and check whether the components related to cloud native monitoring have been installed.

  c. If monitoring is just enabled on the console, wait until the instance association information is synchronized and try again.

Why Is the Number of Queried Notification Rules Inconsistent with That on the Console?

You are advised to use AI CLI to query the AOM notification rules in the current region again. If the inconsistency persists, check whether the current credential, project, and region match those on the console.

Why Does Automatic Notification Rule Creation Fail?

Check whether the SMN topic exists and whether the current account has permission to access AOM notification rules and SMN topics. If yes, create the notification rule again.

There Are Many Alarms, But I Don't Know Which One to Handle First

You are advised to use AI CLI to re-aggregate and analyze alarms by time window and output the severity and handling priority. For example:

```
Aggregate AOM alarms of the test-ai-diagnoses cluster in the last 4 hours by resource and severity, and output the five alarm groups that need to be handled first.
```

If there are still a large number of high- or critical-severity alarms after aggregation, narrow down the scope and analyze the alarm groups that are not cleared, affect core namespaces, are associated with multiple resources, or burst.

Follow-up Suggestions

- After the rules are created, you are advised to query the alarm rules of the target cluster immediately to ensure that the number of rules, notification method, and enabling status meet your expectations.
- At the early stage after the new cluster rollout, you are advised to check active alarms and high-frequency historical alarms every day to check

whether there are alarms with strict thresholds, repeated notifications, or alarms that are not cleared for a long time.

- You are advised to use a fixed time window for analysis during on-duty troubleshooting, for example, "last 30 minutes", "last 4 hours", or "after the change". This avoids interference from irrelevant historical alarms.
- For critical and high-severity alarm groups, you are advised to continue associating logs, events, metrics, and workload versions to form a root cause analysis link.
- For high-frequency alarms that have no impact on services, you are advised to analyze the triggering objects and time periods, and then evaluate the threshold or notification scope optimization solution.
- For production clusters, you are advised to periodically export the alarm rule list as an important material for change audit and fault review.
- After alarms are triggered, you are advised to check the alarm aggregation and severity marking results, and then query related logs, events, and metrics. Do not perform recovery actions based on a single alarm.

Helpful Links

- **Configuring Custom Alarms on AOM**: Learn how to configure custom alarms on AOM for the CCE alarm center and key configuration items such as Prometheus instances.
- **Configuring AOM Alarm Rules**: Learn how to configure AOM alarm rules, rule parameters, and alarm triggering logic.
- **Monitoring CCE Metrics**: Learn how to use AOM to monitor CCE metrics and understand the relationships between metrics, events, and alarm notifications.
- **Creating a Topic**: Learn how to create a topic in SMN before notification rules are automatically created or associated.
- **Cloud Container Engine (CCE) documentation**: Learn about CCE clusters, add-ons, O&M, and workloads.

2.4 Scheduling CCE Workloads to CCI 2.0 Using AI CLI and Skills

CCE Cloud Bursting Engine for CCI is a virtual kubelet that extends Kubernetes APIs to CCI. It allows CCE workloads to be created on CCI 2.0 in short-term high-load scenarios. This section describes how to use AI CLI to load **cce-cci-bursting-deployer** and complete the entire process from cluster pre-check to scaling verification with a single prompt.

Constraints

Environment and Function Constraints

- Only CCE standard and CCE Turbo clusters that use the VPC network model are supported.
- The subnet where a cluster is located cannot overlap with 10.247.0.0/16, or the subnet conflicts with the Service CIDR block in the CCI namespace.

- DaemonSets are not supported.
- After the CCE Cloud Bursting Engine for CCI add-on is installed, a namespace named **bursting-*{cluster-ID}*** will be created in CCI and managed by the add-on. Do not use this namespace directly in CCI.
- For details about the Kubernetes versions and constraints, see **Functions of CCE Cloud Bursting Engine for CCI**.

IAM Permission Requirements

API Action	Permission	Purpose
cce:cluster:get	Obtaining cluster details	Read cluster network specifications (VPC, subnet, and ENI).
cce:addon:list	Listing add-ons	Check the installation status of virtual-kubelet.
cce:addon:create	Creating an add-on	Install the virtual-kubelet add-on.
cce:addon:update	Updating an add-on	Configure bursting parameters.
vpcep:endpoint:create	Creating VPC endpoints	Create VPC endpoints for SWR and OBS.
vpcep:endpoint:list	Listing VPC endpoints	Check existing VPC endpoints.
vpcep:service:list	Listing VPCEP services	Discover public service details.
vpc:subnet:list	Listing subnets	Verify subnet IDs.
vpc:routetable:list	Listing route tables	Query the ID of the OBS gateway route table.

Prerequisites

- Before using the CCE Cloud Bursting Engine for CCI add-on, you have granted CCI permissions to use CCE on the CCI console.
- If you use CCI 2.0 to interconnect with the CCE Cloud Bursting Engine for CCI add-on, purchase VPC endpoints. For details, see **Environment Configuration**.
- You have installed AI CLI and configured the Huawei Cloud credential environment variables **HUAWEI_AK**, **HUAWEI_SK**, and **HUAWEI_PROJECT_ID**.
- You have created a CCE standard or CCE Turbo cluster that uses the VPC network model. The Kubernetes version is 1.21 or later.
- You have obtained the OBS VPCEP service name. The VPC endpoint for OBS requires accurate **obs_endpoint_service_name**. The value needs to be obtained from the Huawei Cloud service ticket and cannot be inferred from the public service name in a similar region. If this value is not provided in advance, the configuration process will be interrupted during VPC endpoint creation and will only continue after you provide the value. Before the

configuration, obtain and record the information through a service ticket to avoid process interruption.

Procedure

Step 1: Create a CCE Turbo Cluster

Create a CCE Turbo cluster and record the cluster ID and region. If a cluster that meets the requirements already exists, skip this step.

Step 2: Enable Auto Scaling Using AI CLI

cce-cci-bursting-deployer uses a preview-first design. Read operations (precheck, verify, discover, and diagnose) can be executed immediately, while a preview solution of write operations (VPC endpoint creation, add-on installation, and workload deployment) is returned first. The write operations are executed only after user confirmation.

Enter the following prompt in AI CLI to start the entire process:

```
I need to enable auto scaling of CCI 2.0 for my CCE cluster (cluster ID: xxx, region: cn-north-4).
Please follow the complete process: precheck → VPC endpoint creation → add-on installation → smoke test deployment → verification.
Preview each write operation before executing it.
```

The Skill will automatically proceed in the following sequence: (Each step involving a write operation will be paused for confirmation.)

1. Cluster precheck

   The Skill invokes **huawei_precheck_cce_cci_bursting** to automatically parse the cluster network topology, distinguishes the subnet roles of **cci_subnet_id** (Neutron UUID) and **vpcep_subnet_id** (VPC UUID), checks the status of the virtual-kubelet add-on, and performs NodeCheck to check the physical node's addon headroom.

2. Node capacity check (NodeCheck)

   If the precheck report indicates that physical node resources are insufficient, the Skill invokes **huawei_check_cce_cci_node_capacity** to view capacity details and preview the node pool scale-out solution. After user confirmation, the scale-out is performed.

3. VPC endpoint creation

   The Skill invokes **huawei_ensure_cce_cci_vpcep** to automatically discover and create VPC endpoints compatible with SWR and OBS. Existing VPC endpoints are automatically reused and will not be created again.

> **CAUTION**
>
> The VPC endpoint for OBS requires accurate **obs_endpoint_service_name**. The value needs to be obtained from the Huawei Cloud service ticket and cannot be inferred from the public service name in a similar region. If this value is not obtained in advance, the Skill will be interrupted in this step and prompt you to provide the value. After **obs_endpoint_service_name** is added, the Skill continues to create the VPC endpoint for OBS and resumes the subsequent process.

4. Add-on installation

   The Skill invokes **huawei_setup_cce_cci_bursting** to install or update the virtual-kubelet add-on after confirming that the VPC endpoint dependencies are ready. It automatically parses and writes the region project ID. This operation is idempotent. The existing add-on is only updated, and will not be uninstalled and reinstalled.

5. Auto scaling readiness verification

   The Skill invokes **huawei_verify_cce_cci_bursting** to check whether the virtual-kubelet add-on and the bursting-node are ready. If the verification fails, the Skill invokes **huawei_diagnose_cce_cci_bursting_addon** to return a structured diagnosis report.

6. Smoke test deployment

   The Skill invokes **huawei_discover_cce_cci_smoke_images** to discover the tenant's image in SWR Basic Edition and then invokes **huawei_deploy_cce_cci_smoke_workload** to create a Deployment. The Deployment automatically adds the **bursting.cci.io/burst-to-cci: enforce** label to forcibly schedule the workload to CCI. If the **image** parameter is not specified, the discovered tenant image is automatically used.

7. Final verification

   The Skill invokes **huawei_verify_cce_cci_bursting** again to ensure that all pods are in the **Running** state on the CCI virtual node and are visible on the CCE console.

Step 3: Configure a ScheduleProfile Policy

After the basic auto scaling configuration is complete, you can create a ScheduleProfile to control workload scheduling.

Scheduling Policy	Description	Scenario
localPrefer	Workloads are preferentially scheduled to CCE. If resources are insufficient, workloads are scheduled to CCI.	Routine scale-out
enforce	Workloads are forcibly scheduled to CCI.	Test verification and temporary CI/CD tasks

Scheduling Policy	Description	Scenario
auto	Workloads will be scheduled to the CCE cluster or CCI based on the scoring results provided by the scheduler.	Flexible scheduling

Enter the following prompt in AI CLI to create a ScheduleProfile:

```
Help me create a ScheduleProfile in the default namespace.
Match the app=nginx label and set the policy to localPrefer.
The maximum number of local instances is 20, and the CCI scale-in priority is 10.
```

NOTE

The label policy has a higher priority than the ScheduleProfile. If a pod has both the **bursting.cci.io/burst-to-cci: off** label and the **enforce** profile, the pod will not be scheduled to CCI.

The auto scaling configuration and scheduling policy for scheduling CCE workloads to CCI 2.0 are complete.

Common Issues and Diagnosis Methods

If you encounter any issues during auto scaling configuration, you can describe the issue in AI CLI. The Skill will invoke the corresponding diagnosis tool to return a structured report.

Symptom	Diagnosis Prompt
CCI Pod ImagePullBackOff or image pull timeout	Failed to pull the CCI pod image. Please diagnose.
Virtual node not ready	The virtual node is not ready. Please diagnose the add-on.
addon Pod Pending or repeatedly restarting	The add-on pod is pending. Help me check the node capacity
"region mismatch" in the add-on log	The add-on log reports a region mismatch. Please diagnose.
IAM denied or project ID missing in the add-on log	The add-on reports IAM denied. Please diagnose.
OBS gateway node failed; VPC endpoint creation failed	Failed to create the VPC endpoint for OBS. Use the service name (actual **obs_endpoint_service_name**) obtained from the Huawei Cloud service ticket to try again.

Helpful Links

- Scaling CCE Pods to CCI
- Functions of CCE Cloud Bursting Engine for CCI
- CCI 2.0 Quick Start
- CCI 2.0 Environment Configuration

2.5 Building an Intelligent O&M Agent for the CCE Production Environment Based on Hermes and Lark

This document describes how to build a ChatOps on-duty Agent for the CCE production environment by connecting Hermes to Lark. The Agent can periodically scan alarms on the live network, automatically merge and analyze alarms, generate recovery solutions, and execute recovery actions after users confirm the recovery on the Lark mobile app. The high CPU usage alarm in this document is only a small case for verifying the closure process. You can extend custom capabilities such as pod restart diagnosis, node exception handling, scheduling failure recovery, capacity inspection, release change association, and daily report generation based on the same idea.

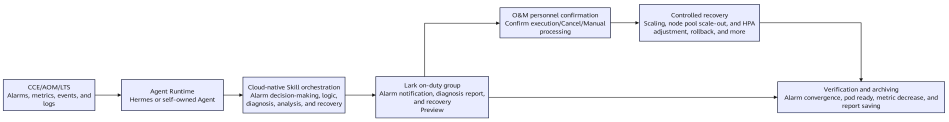
Scenarios

In a production environment, a CCE cluster may continuously generate a large number of alarms, covering multiple dimensions such as workloads, pods, nodes, networks, storage, auto scaling, resource capacity, and availability risks. The larger the live network scale, the more complex the numbers of alarms, alarm sources, and handling paths. On-duty personnel need to frequently switch between the Lark alarms, AOM metrics, Kubernetes events, pod logs, workload configurations, HPA statuses, node capacity, and service ticket system. This can lead to issues such as alarm fatigue, slow response, incomplete evidence, lack of review for recovery actions, and difficulty in accumulating review materials.

By building an intelligent O&M Agent for the CCE production environment, you can consolidate alarm detection, alarm merging, context collection, root cause analysis, recovery preview, user confirmation, recovery execution, effect verification, and result archiving into a set of reusable ChatOps on-duty capabilities. The Agent can use Hermes or your existing ChatOps, AIOps, service ticket assistant, or self-developed on-duty robot. Huawei Cloud cloud-native Skills provide standard query, analysis, and controlled recovery capabilities for resources such as CCE, AOM, LTS, node pools, and workloads.

Solution Architecture

This solution uses the "Agent Runtime + cloud-native Skills + Lark confirmation" architecture. Agent Runtime is responsible for task scheduling, alarm distribution, context orchestration, and Lark interaction. Cloud-native Skills provide capabilities such as alarms, metrics, logs, events, root cause analysis, and recovery actions. Lark carries alarm notification, user review, and closed-loop results.



You are advised to split Agent permissions based on the capability boundary.

Capability Domain	Typical Action	Recommended Control Mode
Alarm governance	Query, merge, classify, and route active and historical AOM alarms.	Read-only permission, allowing scheduled automatic execution.
Diagnosis analysis	Aggregate metrics, events, logs, workloads, and node statuses.	Read-only permission, allowing automatic orchestration by the Agent.
Recovery preview	Generate recovery solutions such as those for scaling, rollback, HPA, and node pools.	Only preview is generated, and resources are not modified.
Recovery execution	Execute the change action.	Must be confirmed via Lark
Audit data archiving	Save alarm, analysis, confirmation, execution, and verification records.	You are advised to access the service ticket, OBS, or internal knowledge base.

Constraints

- Use the recommended configurations of cloud services for alarm rules and metric specifications, and perform cross-verification based on real-time metrics.
- The preview and confirmation mechanisms must be retained for recovery actions, especially production operations such as scaling, rollback, and node pool changes.
- Lark messages should be readable to on-duty personnel. The conclusion, evidence, affected objects, optional solutions, and confirmation entry should be provided first.
- For complex recovery links, you are advised to make alarm fingerprints, solution IDs, target resources, and execution records persistent for confirmation and audit.
- Do not expose sensitive information such as AKs/SKs, tokens, certificates, and project IDs in prompts, Lark messages, screenshots, or documents.

- In the pilot phase, you are advised to start with read-only inspection and manual confirmation for recovery, and then gradually expand to more automatic recovery policies.

Prerequisites

Before performing this practice, you are advised to prepare the Agent running environment and CCE observability objects, and then gradually expand the automation scope.

- You have prepared the CCE clusters, namespaces, or service scope for inspection and diagnosis.
- AOM alarms, cloud native monitoring metrics, or existing inspection objects have been connected.
- You have prepared Hermes or your own Agent Runtime and connected it to Lark or the service ticket system.
- You have imported Huawei Cloud cloud-native Skills related to CCE to the Agent for querying alarms, metrics, events, workloads, pods, and nodes, and performing controlled recovery.
- You have prepared access credentials and have configured them securely to ensure that sensitive information is not exposed in prompts or documents.

Orchestratable Capabilities

You can combine the following Skills as needed to build an intelligent O&M process for countless alarms.

Capability	Representative Skill	Function
Alarm detection and merging	alarm-correlation-engine	Queries active and historical alarms, merges duplicate alarms, and identifies alarms that need to be handled.
Observability context	observability-context-builder	Aggregates metrics, logs, events, and resource statuses.
Metric analysis	metric-analyzer	Analyzes trends of CPU, memory, network, and disk usages.
Pod diagnosis	pod-failure-diagnoser	Analyzes pod statuses, restarts, logs, and events.
Workload diagnosis	workload-failure-diagnoser	Analyzes Deployments, ReplicaSets, HPAs, Services, and endpoints.
Node diagnosis	node-failure-diagnoser	Analyzes node statuses, resource watermarks, and scheduling capabilities.
Change association	change-impact-analyzer	Associates release, configuration, and resource changes before and after an alarm is generated.

Capability	Representative Skill	Function
Root cause analysis	root-cause-analyzer	Summarizes evidence and outputs root causes, confidence, and suggestions.
Controlled recovery	auto-remediation-runner	Generates a recovery preview and executes the recovery action after confirmation.

For live-network alarm governance, Agent capabilities can be classified into the following levels.

Level	Purpose	Example
Alarm entry	Receives and discovers alarms from different sources.	AOM alarms, inspection tasks, Lark messages, and service ticket events
Alarm governance	Reduces alarm noise and determines handling priorities.	Deduplication, merging, classification, routing, silence, and summary
Intelligent diagnosis	Locates candidate causes from multi-source data.	Alarms, metrics, logs, events, changes, and resource statuses
Controlled recovery	Converts recommended actions into reviewable recovery solutions.	Scaling, HPA adjustment, node pool scale-out, and rollback
Closed-loop operations	Consolidates handling results into reusable experience.	Lark notifications, service ticket archiving, daily reports, and review materials

Procedure

This practice is based on the high CPU usage alarm of the **default/chat-app** workload in the **demo-recovery** cluster. It aims to verify the end-to-end capabilities of the intelligent O&M Agent in the CCE production environment. The high CPU usage alarm is just one type of alarm on the live network. The process focuses on demonstrating the complete link from normal inspection, alarm detection, automatic analysis, user confirmation, controlled recovery, to closed-loop verification.

Step 1: Start the Inspection Robot

After initializing the Agent, load CCE-related Skills and configure the inspection period and Lark notification recipient.

The inspection robot should be able to provide clear results in both "no alarm" and "alarm" states so that on-duty personnel can determine whether the inspection link is normal.

Normally, the message indicating that the **demo-recovery** cluster is normal and the environment is normal will be displayed.

Step 2: Receive a High CPU Usage Alarm

When AOM generates a high CPU usage alarm, the inspection robot outputs an alarm summary in Lark and adds the alarm to the automatic analysis process. In the production environment, the same entry can also receive other types of alarms, such as pod restart, node exception, scheduling failure, no backend for a Service, and HPA not taking effect.

After receiving an alarm, the Agent should pay attention to both the AOM alarm status and real-time metrics to avoid making decisions based on a single signal.

Step 3: Automatically Analyze Alarms and Generate a Recovery Preview

After detecting high CPU usage, the inspection robot automatically collects alarms, pod metrics, node watermarks, workload statuses, and affected objects to generate a diagnosis report. The report should highlight observable facts, evidence chains, and optional recovery solutions.

The recovery preview should include the following items:

Item	Content
Alarm summary	Alarm name, severity, status, and trigger time
Affected objects	Cluster, namespace, workload, pod, and node
Key evidence	CPU watermark, node watermark, pod status, and related alarms
Candidate solutions	Replica scale-out, resource adjustment, node pool scale-out, manual handling, and other solutions
Change impact	Resource usage, scheduling conditions, cost changes, and rollback methods
User confirmation	Must provide clear confirmation statements or buttons.

Step 4: Confirm the Recovery Solution in Lark

This is a key review step in the production change process. In this phase, the Agent only waits for user confirmation and does not perform any operations that change the live network status, such as scaling, rollback, restart, or node pool change. The Agent will only execute the recovery operation based on the confirmed solution after the user clearly replies with a confirmation statement such as "Confirm execution" or "Confirm execution of solution A/B" in Lark.

The confirmation statement must match the solution number in the recovery preview. For example, "Confirm execution of solution A" corresponds to adding replicas, and "Confirm execution of solution B" corresponds to upgrading resource specifications or expanding capacity. You can also replace the confirmation action with a Lark card button, service ticket approval, or enterprise approval process. However, the core requirement remains unchanged: Without manual confirmation, the Agent only performs analysis and preview and does not make changes to the live network.

Step 5: Perform Recovery and Continuous Review

In this case, the workload is first scaled out by increasing the number of replicas of **chat-app** from 2 to 4. After the scale-out, the Agent continues to review the pod statuses and node capacity and finds that one of the new pods is pending due to insufficient CPU resources on the node.

This branch highlights an important aspect of production recovery: recovery actions require continuous verification. A successful scale-out request does not necessarily mean that all pods have been successfully scheduled or that the alarm has been cleared. The Agent should continue to provide the verification results to the user and propose the next steps.

Step 6: Add Capacity and Complete Closed-Loop Operations

When the scale-out is limited by the node capacity, the Agent can generate a new capacity recovery solution, such as adding a node pool, adjusting workload requests, optimizing the HPA upper limit, or integrating with CCI for auto scaling. In this example, a node pool is added for scale-out. After the new nodes are online, Kubernetes automatically schedules the pending pods.

After the recovery action is performed, the Agent inspects the alarms, pods, nodes, and CPU metrics again and sends the closed-loop result to Lark.

Hermes Task Prompt Reference

The following prompts can be used as a task template for the Hermes ChatOps on-duty Agent. It downplays specific commands and environment variables, retaining only the role, process, output structure, and security boundaries.

```
You are the intelligent O&M Agent of the CCE production environment. You are responsible for performing alarm inspection, alarm merging, automatic analysis, recovery preview, recovery after user confirmation, recovery verification, and Lark closed-loop notification in the target CCE environment.

Prerequisites:
- CCE-related cloud-native Skills have been imported to the current Agent in advance.
```



```
- The target cluster, inspection scope, notification channel, and access credentials have been provided by the runtime environment.
- All notifications are sent to Lark or the on-duty channel specified by the customer.

Purposes:
1. Periodically scan active alarms and recent historical alarms in the target CCE environment.
2. Deduplicate, merge, classify, route, and summarize the impact scope of alarms.
3. When the inspection is normal, output a concise health summary and do not exit silently.
4. When alarms need to be handled, automatically aggregate the context, including AOM alarms, real-time metrics, Kubernetes events, pod/workload/node statuses, log summaries, and recent changes.
5. Generate a diagnosis report for on-duty personnel, including the alarm summary, affected objects, key evidence, possible causes, recommended solutions, and actions to be confirmed.
6. For actions involving resource changes, only a recovery preview is generated, and no action is directly performed on the live network.
7. The recovery action is executed only after the user clearly replies with a confirmation statement such as "Confirm execution" or confirms a specific solution in Lark.
8. After the execution, verify the alarm statuses, pod statuses, workload replicas, node capacity, and key metrics, and send the closed-loop result to Lark.

Recommended structure of the inspection report:
- Inspection summary: cluster, time window, number of active alarms, and key resource statuses
- Alarm discovery: alarm name, severity, status, affected objects, and current observed value
- System analysis: alarm statuses, real-time metrics, pod/workload/node statuses, related events, and recent changes
- Recovery solution: Provide two to three optional solutions, including the scenarios, impact scope, rollback method, and verification method.
- User confirmation: Prompt the user to reply "Confirm" or select a specific solution.

Security boundaries:
- Only read-only operations are allowed during alarm scanning, evidence collection, and root cause analysis.
- Recovery cannot be performed based on a single alarm.
- For all write operations, the recovery preview, impact scope, rollback method, and verification method must be provided first.
- The solution number and meaning in the same alarm must be consistent. After the user confirms the solution, the solution is executed.
- Before user confirmation, do not perform any operations that change the live network status, such as scaling, rollback, restart, or node pool change.
- Do not expose sensitive information such as AKs/SKs, tokens, certificates, and project IDs in the output.
- If the evidence is insufficient, list possible causes and information that requires manual review. Do not make judgments for the user without sufficient evidence.

Lark output requirements:
- When the inspection is normal, output a concise health summary.
- When an alarm is detected, output the alarm summary and affected objects first, and then output the key evidence and candidate solutions.
- If recovery is required, clearly prompt the user to reply "Confirm" or select a specific solution. Before receiving manual confirmation, wait for confirmation and do not perform the change.
- After the recovery is complete, output the execution actions, verification results, remaining risks, and subsequent suggestions.
```

Diagnosis Results

The focus of this case is not that capacity expansion is necessary when the CPU usage is high. Instead, it demonstrates a transferable method: Alarm detection by the Agent → Evidence aggregation by the Skill → Manual confirmation and recovery → System execution and verification. You can replace any of the phases with your own tools, approval processes, and business rules.

Phase	Result
Alarm detection	The container CPU usage is greater than 80%.
Automatic analysis	Aggregate AOM alarms, pod metrics, node watermarks, and workload statuses.

Phase	Result
Recovery preview	Provide optional solutions such as adding replicas and wait for confirmation from Lark.
First recovery	Scale the workload from two replicas to four replicas.
Process review	A new pod is pending due to insufficient node resources.
Add action	Add node pools to expand the scheduling capacity.
Closed-loop verification	Clear active alarms, check that the pod and node are normal, and output the result on Lark.

Extended Scenarios

You can extend this practice from multiple dimensions based on your requirements and existing tools to adapt to different O&M scenarios and service requirements. The following provides suggestions and examples for extending this practice from different dimensions.

Extension Direction	Example
Replacing the Agent	Use Hermes, OpenClaw, AI CLI, enterprise ChatOps robot, or self-developed Agent. Different Agents provide different functions and integration capabilities. You can select an appropriate Agent based on your technology stack and requirements.
Changing the entry	Triggered by AOM alarms, Lark messages, service tickets, scheduled tasks, release events, or manual inquiries. Different entries can adapt to different alarm sources and trigger modes, improving the flexibility and response speed of alarm handling.
Changing the Skill combination	Orchestrate different Skills for scenarios such as pods, nodes, networks, storage, HPAs, and costs. By combining different Skills, you can provide more accurate diagnosis and recovery capabilities for specific O&M scenarios.
Changing approval methods	Use Lark reply, Lark card button, service ticket approval, and change approval flow. Different approval methods can adapt to different enterprise approval processes and security requirements, ensuring the compliance and security of recovery actions.
Changing recovery actions	Scaling, HPA adjustment, node pool scale-out, rollback, node isolation, and stopping abnormal tasks. Different recovery actions can address different fault types and recovery requirements, improving the flexibility and effectiveness of recovery.

Extension Direction	Example
Changing the archiving mode	Output to Lark, service tickets, OBS, daily reports, knowledge bases, or audit systems. Different archiving modes can meet different recording and audit requirements, ensuring that the alarm handling process can be traced and reviewed.

The following table lists the typical extension scenarios.

Scenario	Orchestration Approach
Frequent pod restarts	Aggregate the number of restarts, previous logs, OOM events, probe configurations, and events to generate a preview of rollback or resource adjustment. This helps quickly locate and resolve frequent pod restarts.
Pod Pending	Analyze the node capacity, taint tolerance, affinity, PVC, image pull, and quota to generate scheduling recovery suggestions. This helps resolve pod scheduling failures.
Abnormal nodes	Associate node statuses, resource watermarks, component statuses, and events to generate isolation, migration, or node pool scale-out preview. This helps quickly handle node exceptions.
No backend for the Service	Analyze the Deployment, Endpoint, Service Selector, and release statuses to locate release or selector issues. This helps locate the causes and generates corresponding recovery suggestions.
HPA not taking effect	Analyze metric collection, request configuration, HPA upper and lower limits, and scaling events. This helps diagnose the causes and generates corresponding recovery suggestions.
Periodic inspection	Periodically output alarms, resource watermarks, abnormal pods, node risks, and cost optimization suggestions. This helps promptly identify and address potential issues and optimize resource utilization.

Expected Results

After completing this practice, you can obtain the following benefits:

1. After a CCE alarm is reported to Lark, the Agent automatically starts the diagnosis link.
2. O&M personnel can view the alarm summary, evidence, and candidate solutions without repeatedly switching between multiple systems.
3. Recovery actions are confirmed on Lark before execution, reducing misoperations.

4. After the recovery is executed, the alarm convergence, pod status, node capacity, and metric trends are automatically verified.
5. The alarm handling process can be archived, audited, and reviewed.
6. The same Agent + Skill orchestration approach can be extended to more CCE O&M scenarios.

