Well-Architected Framework (WAF)

Issue 01

Date 2025-06-25

HUAWEI

HUAWEI TECHNOLOGIES CO., LTD.

Copyright © Huawei Technologies Co., Ltd. 2025. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

Trademarks and Permissions

and other Huawei trademarks are trademarks 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 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.

Security Declaration

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process.* For details about this process, visit the following web page:
https://www.huawei.com/en/psirt/vul-response-process
For vulnerability information, enterprise customers can visit the following web page:
https://securitybulletin.huawei.com/enterprise/en/security-advisory

Contents

1 Well-Architected Framework (WAF) Introduction....1

2 Resilience Pillar....5

2.1 Introduction to the Resilience Pillar....5
2.2 Concepts....5
2.2.1 Concepts....6
2.2.2 Application Resilience....7
2.2.3 Shared Responsibility Model for Resiliency....7
2.2.4 Availability Objectives....8
2.2.4.1 Availability and SLO....8
2.2.4.2 RTO and RPO....10
2.2.4.3 Data Durability....10
2.2.5 Availability Requirements....10
2.3 Design Principles....10
2.4 Questions and Checklists....13
2.5 HA Design....16
2.5.1 RES01 Redundancy....16
2.5.1.1 Overview....16
2.5.1.2 RES01-01 High-Availability Deployment of Application Components....16
2.5.1.3 RES01-02 Multi-Location Deployment of Application Components....17
2.5.1.4 RES01-03 Anti-Affinity for Cloud Servers....18
2.5.2 RES02 Backup....18
2.5.2.1 Overview....18
2.5.2.2 RES02-01 Identifying and Backing Up Critical Data That Needs to Be Backed Up....19
2.5.2.3 RES02-02 Automatically Backing Up Data....19
2.5.2.4 RES02-03 Periodically Restoring Data from Backups....20
2.5.3 RES03 Cross-AZ DR....20
2.5.3.1 Overview....20
2.5.3.2 RES03-01 Deploying Clusters Across AZs....21
2.5.3.3 RES03-02 Synchronizing Data Across AZs....22
2.5.3.4 RES03-03 Interconnecting with DR Arbitrator for Automatic Switchover....22
2.5.3.5 RES03-04 Supporting DR Management....23
2.5.4 RES04 Cross-Region/Cross-Cloud DR....23
2.5.4.1 Overview....23

 Copyright © Huawei Technologies Co., Ltd.

2.5.4.2 RES04-01 Defining the RPO and RTO for an Application System 23
2.5.4.3 RES04-02 Deploying a DR System to Meet the DR Objectives 24
2.5.4.4 RES04-03 Automating the DR Process 25
2.5.4.5 RES04-04 Regularly Performing DR Drills to Check Whether the Recovery Can Meet DR Objectives 25
2.5.5 RES05 Network HA 26
2.5.5.1 Overview 26
2.5.5.2 RES05-01 Ensuring High Availability of Network Connections 26
2.5.5.3 RES05-02 Avoiding Unnecessarily Exposing Network Addresses 27
2.5.5.4 RES05-03 Isolating Network Bandwidth Among Services of Different Traffic Models 28
2.5.5.5 RES05-04 Reserving IP Resources for Expansion and HA 28
2.6 Comprehensive Fault Detection 28
2.6.1 RES06 Fault Detection 28
2.6.1.1 Overview 29
2.6.1.2 RES06-01 Fault Mode Analysis 29
2.6.1.3 RES06-02 Fault Detection 30
2.6.1.4 RES06-03 Subhealth Detection 31
2.6.2 RES07 Monitoring Alarms 32
2.6.2.1 Overview 32
2.6.2.2 RES07-01 Defining Key Metrics and Thresholds and Monitoring Such Metrics 32
2.6.2.3 RES07-02 Monitoring Logging 34
2.6.2.4 RES07-03 Sending Notifications When Exceptions Are Detected 34
2.6.2.5 RES07-04 Storing and Analyzing Monitoring Data 35
2.6.2.6 RES07-05 Tracking Requests End-to-End 35
2.7 Rapid Fault Recovery 36
2.7.1 RES08 Dependency Reduction and Degradation 36
2.7.1.1 Overview 36
2.7.1.2 RES08-01 Reducing Strong Dependencies 36
2.7.1.3 RES08-02 Using Loose Coupling to Reduce Dependencies 37
2.7.1.4 RES08-03 Minimizing the Impacts of Dependency Failures 37
2.7.2 RES09 Retries After Failures 38
2.7.2.1 Overview 38
2.7.2.2 RES09-01 Designing a Retry Mechanism for API Calls and Command Executions 38
2.7.2.3 RES09-02 Determining Whether to Retry from the Client Based on the Comprehensive Evaluation Results 38
2.7.2.4 RES09-03 Avoiding Creating Too Much Traffic Pressure from Excessive Retries 39
2.7.3 RES10 Fault Isolation 39
2.7.3.1 Overview 39
2.7.3.2 RES10-01 Isolating the Control Plane from the Data Plane 39
2.7.3.3 RES10-02 Deploying Application Systems in Multiple Locations 40
2.7.3.4 RES10-03 Adopting a Grid Architecture 40
2.7.3.5 RES10-04 Configuring Health Check and Automatic Isolation 42
2.7.4 RES11 Reliability Testing 42

 Copyright © Huawei Technologies Co., Ltd.

2.7.4.1 Overview ... 42
2.7.4.2 RES11-01 Chaos Testing ... 43
2.7.4.3 RES11-02 Stress Testing ... 44
2.7.4.4 RES11-03 Long-Term Stability Testing ... 44
2.7.4.5 RES11-04 DR Drills ... 45
2.7.4.6 RES11-05 Red/Blue Attack/Defense ... 45
2.7.5 RES12 Emergency Recovery ... 45
2.7.5.1 Overview ... 45
2.7.5.2 RES12-01 Setting Up an Emergency Recovery Team ... 46
2.7.5.3 RES12-02 Developing an Emergency Response Plan ... 46
2.7.5.4 RES12-03 Conducting Emergency Recovery Drills Periodically ... 46
2.7.5.5 RES12-04 Recovering from Faults Immediately ... 47
2.7.5.6 RES12-05 Organizing Emergency Recovery Backtracking ... 47
2.8 Overload Control ... 48
2.8.1 RES13 Overload Protection ... 48
2.8.1.1 Overview ... 48
2.8.1.2 RES13-01 Automatic Elastic Scaling ... 48
2.8.1.3 RES13-02 Application System Load Balancing ... 49
2.8.1.4 RES13-03 Overload Detection and Control ... 50
2.8.1.5 RES13-04 Proactive Capacity Expansion ... 50
2.8.1.6 RES13-05 Quota Restrictions on Automatic Capacity Expansion ... 51
2.8.1.7 RES13-06 Load Testing ... 51
2.9 Change Error Prevention ... 51
2.9.1 RES14 Configuring Error Prevention ... 52
2.9.1.1 Overview ... 52
2.9.1.2 RES14-01 Fool-proofing Check for Changes ... 52
2.9.1.3 RES14-02 Automatic Changes ... 52
2.9.1.4 RES14-03 Data Backup Before a Change ... 53
2.9.1.5 RES14-04 Runbooks for Standardized Changes ... 53
2.9.2 RES15 Hitless Upgrade ... 53
2.9.2.1 Overview ... 54
2.9.2.2 RES15-01 Automatic Deployment and Upgrade ... 54
2.9.2.3 RES15-02 Automatic Checks ... 54
2.9.2.4 RES15-03 Automatic Rollback ... 54
2.9.2.5 RES15-04 Gray Deployment and Upgrade ... 54
2.10 Reference Architecture ... 55
2.10.1 Overview ... 55
2.10.2 Typical Deployment Architecture for Internal Tools or OBT Applications (99% Availability) ... 56
2.10.3 Typical Deployment Architecture for Internal Knowledge Management Applications (99.9% Availability) ... 57
2.10.4 Typical Deployment Architecture for Information Management Applications (99.95% Availability) ... 60
2.10.5 Typical Deployment Architecture for E-commerce Applications (99.99% Availability) ... 63

 Copyright © Huawei Technologies Co., Ltd.

2.10.5.1 Single-Region Solution ... 63
2.10.5.2 Dual-Region Solution ... 65
2.10.6 Typical Deployment Architecture for Core Financial Applications (99.999% Availability) ... 67
2.10.7 Cross-Cloud Scenarios (99.99% Availability) ... 69
2.10.7.1 Overview ... 69
2.10.7.2 Cross-Cloud DR ... 69
2.10.7.3 Cross-Cloud Active-Active DR ... 72
2.11 Cloud Service Reliability ... 74
2.11.1 Overview ... 74
2.11.2 ECS ... 76
2.11.2.1 Reliability Functions ... 76
2.11.2.2 Common Faults ... 78
2.11.3 BMS ... 79
2.11.3.1 Reliability Functions ... 79
2.11.3.2 Common Faults ... 80
2.11.4 CCE ... 81
2.11.4.1 Reliability Functions ... 81
2.11.4.2 Common Faults ... 82
2.11.5 ELB ... 83
2.11.5.1 Reliability Functions ... 83
2.11.5.2 Common Faults ... 84
2.11.6 AS ... 84
2.11.6.1 Reliability Functions ... 84
2.11.6.2 Common Faults ... 85
2.11.7 DCS ... 85
2.11.7.1 Reliability Functions ... 85
2.11.7.2 Common Faults ... 86
2.11.8 DMS ... 87
2.11.8.1 Reliability Functions ... 87
2.11.8.2 Common Faults ... 87
2.11.9 RDS ... 88
2.11.9.1 Reliability Functions ... 88
2.11.9.2 Common Faults ... 89
2.11.10 TaurusDB ... 90
2.11.10.1 Reliability Functions ... 90
2.11.10.2 Common Faults ... 91
2.11.11 OBS ... 91
2.11.11.1 Reliability Functions ... 92
2.11.11.2 Common Faults ... 92
3 Security Pillar ... 94
3.1 Overview ... 94
3.1.1 Introduction to the Security Pillar ... 94

 Copyright © Huawei Technologies Co., Ltd.

3.1.2 Shared Responsibility Model 95
3.2 Concepts 95
3.2.1 Concept Description 95
3.2.2 Conceptual Models 98
3.3 Design Principles 99
3.4 Questions and Checklists 100
3.5 Cloud Security Governance Policies 102
3.5.1 SEC01 Cloud Security Governance Policies 102
3.5.1.1 SEC01-01 Establishing a Security Management Team 102
3.5.1.2 SEC01-02 Establishing a Security Baseline 103
3.5.1.3 SEC01-03 Compiling an Asset List 103
3.5.1.4 SEC01-04 Separating Workloads 104
3.5.1.5 SEC01-05 Performing Threat Modeling Analysis 105
3.5.1.6 SEC01-06 Identifying and Validating Security Measures 107
3.6 Infrastructure Security 107
3.6.1 SEC02 Identity Authentication 107
3.6.1.1 SEC02-01 Account Protection 107
3.6.1.2 SEC02-02 Secure Login Mechanism 108
3.6.1.3 SEC02-03 Security Management and Credential Usage 109
3.6.1.4 SEC02-04 Integrated Identity Management 109
3.6.2 SEC03 Permission Management 110
3.6.2.1 SEC03-01 Defining Access Control Requirements 110
3.6.2.2 SEC03-02 Assigning Appropriate Permissions on Demand 110
3.6.2.3 SEC03-03 Regularly Reviewing Permissions 111
3.6.2.4 SEC03-04 Securely Sharing Resources 112
3.6.3 SEC04 Network Security 112
3.6.3.1 SEC04-01 Partitioning the Network 112
3.6.3.2 SEC04-02 Controlling Network Traffic Access 113
3.6.3.3 SEC02-03 Minimizing Network Access Permissions 114
3.6.4 SEC05 Runtime Environment Security 115
3.6.4.1 SEC05-01 Cloud Service Security Configuration 115
3.6.4.2 SEC05-02 Vulnerability Management 116
3.6.4.3 SEC05-03 Reducing Attack Surfaces of Resources 116
3.6.4.4 SEC05-04 Key Security Management 117
3.6.4.5 SEC05-05 Certificate Security Management 118
3.6.4.6 SEC05-06 Using Hosted Cloud Services 119
3.7 Application Security 119
3.7.1 SEC06 Application Security 119
3.7.1.1 SEC06-01 Ensuring Open-Source Software Security 119
3.7.1.2 SEC06-02 Establishing Secure Coding Specifications 120
3.7.1.3 SEC06-03 Implementing White-Box Code Reviews 121
3.7.1.4 SEC06-04 Configuring Application Security 122

 Copyright © Huawei Technologies Co., Ltd.

3.7.1.5 SEC06-05 Performing Penetration Testing 122
3.8 Data Security and Privacy Protection 123
3.8.1 SEC07 General Data Security 123
3.8.1.1 SEC07-01 Identifying Data in Workloads 123
3.8.1.2 SEC07-02 Data Protection Control 124
3.8.1.3 SEC07-03 Monitoring Data Use 125
3.8.1.4 SEC07-04 Static Data Encryption 125
3.8.1.5 SEC07-05 Transmission Data Encryption 126
3.8.2 SEC08 Data Privacy Protection 127
3.8.2.1 SEC08-01 Specifying Privacy Protection Policies and Principles 127
3.8.2.2 SEC08-02 Proactive Notification to Data Subjects 128
3.8.2.3 SEC08-03 Data Subjects' Choice and Consent 129
3.8.2.4 SEC08-04 Data Collection Compliance 130
3.8.2.5 SEC08-05 Compliance of Data Usage, Retention, and Disposal 130
3.8.2.6 SEC08-06 Compliance of Personal Data Disclosure to Third Parties 131
3.8.2.7 SEC08-07 Data Subjects Have Rights to Access Their Privacy Data 132
3.9 Security Operations 132
3.9.1 SEC09 Security Awareness and Analysis 132
3.9.1.1 SEC09-01 Implementing Standardized Log Management 132
3.9.1.2 SEC09-02 Logging and Analyzing Security Incidents 133
3.9.1.3 SEC09-03 Implementing Security Audits 134
3.9.1.4 SEC09-04 Security Situation Awareness 134
3.9.2 SEC10 Security Incident Response 135
3.9.2.1 SEC10-01 Establishing a Security Response Team 135
3.9.2.2 SEC10-02 Developing an Incident Response Plan 136
3.9.2.3 SEC10-03 Automatic Response to Security Incidents 137
3.9.2.4 SEC10-04 Security Incident Drill 138
3.9.2.5 SEC10-05 Establishing a Review Mechanism 140
3.10 Reference Architecture 141
3.10.1 Organization-level Reference Architecture 141
3.10.2 Workload-Level Reference Architecture 145
3.11 Security Services 146
4 Performance Efficiency Pillar 148
4.1 Introduction 148
4.2 Concepts 149
4.3 Design Principles 150
4.4 Questions and Checklists 150
4.5 PERF01 Processes and Specifications 151
4.5.1 Full-lifecycle Performance Management 152
4.5.1.1 PERF01-01 Managing Performance Throughout the Lifecycle 152
4.5.2 Application Performance Programming Specifications 153
4.5.2.1 PERF01-02 Developing Application Performance Programming Specifications 153

 Copyright © Huawei Technologies Co., Ltd.

4.6 PERF02 Performance Planning 153
4.6.1 Performance Planning 153
4.6.1.1 PERF02-01 Setting Performance Targets 154
4.6.1.2 PERF02-02 Capacity Planning 155
4.7 PERF03 Performance Modeling 156
4.7.1 Selecting Appropriate Compute Resources 156
4.7.1.1 PERF03-01 Selecting Appropriate Compute Services 157
4.7.1.2 PERF03-02 Selecting VMs and Container Nodes of Appropriate Specifications 157
4.7.1.3 PERF03-03 Applying Auto Scaling 158
4.7.2 Selecting Appropriate Network Services 160
4.7.2.1 PERF03-04 Selecting Appropriate Networking Services 161
4.7.3 Selecting Appropriate Storage Services 162
4.7.3.1 PERF03-05 Selecting Appropriate Storage Services 162
4.7.4 Selecting Appropriate Application Middleware Services 165
4.7.4.1 PERF03-06 Selecting Appropriate Message Queuing Services 165
4.7.4.2 PERF03-07 Selecting an Appropriate Kafka 165
4.7.4.3 PERF03-08 Selecting an Appropriate RocketMQ 166
4.7.4.4 PERF03-09 Selecting an Appropriate RabbitMQ 166
4.7.5 Selecting Appropriate Database Resources 166
4.7.5.1 PERF03-10 Selecting Appropriate Relational Databases 167
4.7.5.2 PERF03-11 Selecting Appropriate Non-Relational Databases 168
4.8 PERF04 Performance Analysis 169
4.8.1 Performance Testing 169
4.8.1.1 PERF04-01 Defining Acceptance Criteria 169
4.8.1.2 PERF04-02 Selecting Appropriate Test Methods 170
4.8.1.3 PERF04-03 Performance Test Procedure 170
4.8.2 Collecting Performance Data 173
4.8.2.1 PERF04-04 Collecting Resource Performance Data 173
4.8.2.2 PERF04-05 Collecting Application Performance Data 174
4.8.3 Developing a Performance Observability System 174
4.8.3.1 PERF04-06 Developing a Performance Observability System 174
4.9 PERF05 Performance Optimization 175
4.9.1 Design Optimization 175
4.9.1.1 PERF05-01 Optimizing Design 175
4.9.2 Algorithm Optimization 177
4.9.2.1 PERF05-02 Optimizing General Algorithms 177
4.9.3 Resource Optimization 178
4.9.3.1 PERF05-03 Optimizing Resources for Web Scenarios 178
4.9.3.2 PERF05-04 Optimizing Resources for Big Data Scenarios 178
4.10 PERF06 Performance Assurance 179
4.10.1 Performance Assurance 179
4.10.1.1 PERF06-01 Applying Layered Assurance 179

4.10.1.2 PERF06-02 Automatically Demarcating and Locating Performance Issues....179
4.10.1.3 PERF06-03 Automating Alarming....180
4.11 Cloud Service Performance Optimization....180
4.11.1 Optimizing Cache Performance....180
4.11.2 Optimizing Message Queue Performance....187
4.11.2.1 Optimizing Kafka Performance....187
4.11.2.2 Optimizing RabbitMQ Performance....190
4.11.3 Optimizing Serverless Performance....193
4.11.4 Optimizing Database Performance....195
4.11.5 Optimizing AI Performance....198
4.11.6 Optimizing Big Data Performance....200
4.11.6.1 Optimizing Hive....200
4.11.6.2 Optimizing Spark Performance....203
4.11.6.3 Optimizing Flink Performance....204
5 Cost Optimization Pillar....207
5.1 Overview....207
5.2 Concepts....207
5.3 Design Principles....208
5.4 Problems and Checklists....209
5.5 COST01 Planning Organizations and Processes According to Cost Optimization Requirements....211
5.5.1 COST01-01 Planning Enterprise Organizations, and Aligning the Organizational Structure, Process, and Cost Management....211
5.5.2 COST01-02 Designing a Structured IT Governance System to Optimize Management Efficiency....211
5.5.3 COST01-03 Clarifying Team Responsibilities, and Cultivating and Sustaining an Organizational Cost-consciousness Culture....212
5.5.4 COST01-04 Establishing Cloud Resource and Permissions Management Policies....213
5.6 COST02 Planning and Managing Budgets....213
5.6.1 COST02-01 Establishing Cloud Budgets and Forecasts....213
5.6.2 COST02-02 Refining Budget Management and Tracking....214
5.7 COST03 Allocating Costs....214
5.7.1 COST03-01 Developing Cost Allocation Principles....214
5.7.2 COST03-02 Visualizing Cost Allocation Results....215
5.7.3 COST03-03 Allocating Shared Costs....216
5.8 COST04 Performing Continuous Cost Governance....216
5.8.1 COST04-01 Establishing Governance Frameworks to Continuously Optimize the Cost Allocation Ratio....217
5.8.2 COST04-02 Proactively Monitoring Costs....218
5.9 COST05 Setting Optimization Strategies and Objectives....218
5.9.1 COST05-01 Analyzing Service Trends and Optimization Benefits....218
5.9.2 COST05-02 Establishing Measurable Optimization Objectives....219
5.9.3 COST05-03 Regularly Reviewing and Verifying Optimization Results....220
5.10 COST06 Choosing the Right Billing Mode....220
5.10.1 COST06-01 Understanding the Characteristics of Different Billing Modes on the Cloud....220

 Copyright © Huawei Technologies Co., Ltd.

5.10.2 COST06-02 Selecting an Appropriate Billing Mode for Your Workloads 221
5.10.3 COST06-03 Tracking and Monitoring the Usage of Special Offerings 221
5.11 COST07 Managing and Optimizing Resources 222
5.11.1 COST07-01 Continuously Monitoring Resource Utilization 222
5.11.2 COST07-02 Releasing Idle Resources 222
5.11.3 COST07-03 Conducting Comprehensive Cloud Technology Selection Analysis 222
5.11.4 COST07-04 Changing Specifications Based on Workload Patterns and Resource Utilization 223
5.12 COST08 Optimizing Architectures 223
5.12.1 COST08-02 Implementing Cloud-Native Architecture Transformation 223
5.12.2 COST08-03 Decoupling Storage and Compute 223
5.12.3 COST08-04 Exploring Serverless 224
5.13 Cloud Services Used for Cost Optimization 224
6 Operational Excellence Pillar 226
6.1 Operational Excellence 226
6.2 Concepts 226
6.3 Design Principles 228
6.4 Questions and Checklists 230
6.5 OPS01 Building a Culture of Continuous Improvement with a Standardized O&M System 231
6.5.1 OPS01-01 Building a Growing Culture 231
6.5.2 OPS01-02 Designing a Standardized O&M Organization 231
6.5.3 OPS01-03 Standardizing O&M Processes and Tools 232
6.6 OPS02 Frequent, Small, Reversible Changes Through CI/CD 233
6.6.1 OPS02-01 Managing Requirements and Sprint Development 233
6.6.2 OPS02-02 Linking Source Code Versions to Deployed Applications and Applying Code Quality Best Practices 234
6.7 OPS03 Comprehensive Test and Verification System 234
6.7.1 OPS03-01 Promoting Developer Testing 234
6.7.2 OPS03-02 Performing Integration Testing in Multiple Environments and Mimicking Production with Staging Environments 235
6.7.3 OPS03-03 Performing Performance Testing 235
6.7.4 OPS03-04 Performing Echo Testing in the Production Environment 236
6.7.5 OPS03-05 Performing Chaos Testing and Drills 237
6.8 OPS04 Automated Build and Deployment 238
6.8.1 OPS04-01 Effectively Implementing Continuous Integration 238
6.8.2 OPS04-02 Using Continuous Deployment Models 239
6.8.3 OPS04-03 Implementing Infrastructure as Code 239
6.8.4 OPS04-04 Automating O&M Tasks 240
6.9 OPS05 O&M Preparation and Change Management 242
6.9.1 OPS05-01 Conducting a PRR 242
6.9.2 OPS05-02 Conducting a Change Risk Control 242
6.9.3 OPS05-03 Defining a Change Process 243
6.10 OPS06 Observability System 244
6.10.1 OPS06-01 Establishing an Observability System 244

 Copyright © Huawei Technologies Co., Ltd.

6.10.2 OPS06-02 Defining Observable Objects....245
6.10.3 OPS06-03 Developing and Implementing Observability Indicators....246
6.10.4 OPS06-04 Standardizing Application Logs....247
6.10.5 OPS06-05 Implementing Dependency Telemetry....247
6.10.6 OPS06-06 Performing Distributed Tracing....248
6.10.7 OPS06-07 Automation Measures Based on Observability Metrics....248
6.11 OPS07 Fault Analysis and Management....248
6.11.1 OPS07-01 Creating Alarms....248
6.11.2 OPS07-02 Creating Monitoring Dashboards....249
6.11.3 OPS07-03 Managing Events....249
6.11.4 OPS07-04 Supporting Fault Recovery Process....250
6.12 OPS08 Operations Status Measurement and Continuous Improvement....250
6.12.1 OPS08-01 Using Metrics to Measure Operations Objectives....250
6.12.2 OPS08-02 Reviewing and Improving Events....251
6.12.3 OPS08-03 Managing Knowledge....251
6.13 Examples....252
6.13.1 Improving System O&M Capabilities to Reduce O&M Costs and Difficulties with AOM....252
6.13.2 Gathering Various Device Logs Using LTS for Full-Link Issue Tracking and Service Operations Analysis....253
6.13.3 Performing Daily Service O&M and Compliance with LTS....254
6.14 Operational Excellence....254
6.14.1 CodeArts....254
6.14.2 RFS....255
6.14.3 COC....255
6.14.4 Cloud Eye....257
6.14.5 LTS....258
6.14.6 AOM 2.0....258
6.14.7 APM....258
6.14.8 CBH....259
6.14.9 ServiceStage....259
6.14.10 MAS....259

 Copyright © Huawei Technologies Co., Ltd.

1 Well-Architected Framework (WAF) Introduction

WAF provides a set of principles and best practices that guide you through cloud architecture designs.

Based on Huawei and industry best practices, WAF has five core architectural principles: resilience, security, performance efficiency, cost optimization, and operational excellence. It empowers customers to design exceptional cloud architectures, build cloud architecture governance systems, improve R&D productivity, develop modernized applications, and build O&M systems.

Architecture Pillars

- **Resilience**

  The resilience pillar leverages Huawei's expertise in resilience design and industry best practices to help enterprises build highly available, resilient application systems on Huawei Cloud. A highly resilient system can maintain acceptable service levels even in abnormal scenarios. Additionally, by standardizing change management, deployment, and emergency recovery processes, enterprises can reduce service disruptions and enhance availability.

- **Security**

  The security pillar is dedicated to ensuring the security, trustworthiness, and compliance of services. It protects workloads against security threats and reduces security risks through a series of best practices in Huawei Cloud architecture. The security pillar provides best practices to protect the confidentiality, integrity, and availability of systems, assets, and data on the cloud. It ensures legal and compliant use of data, and protects user privacy.

- **Performance efficiency**

  The performance efficiency pillar focuses on how to design a high-performance architecture. Performance is crucial as a fundamental quality attribute, and its failures can lead to serious outcomes. This pillar provides technical methods for designing and improving system performance. It supports software performance engineering and helps optimize system efficiency.

- **Cost optimization**

 Copyright © Huawei Technologies Co., Ltd.

The cost optimization pillar helps enterprises use cloud services effectively for building workloads, improve workload performance throughout the whole lifecycle, cut unnecessary costs, boost operations efficiency, and maximize the cost-effectiveness of cloud applications. This pillar provides systematic practice suggestions summarized from Huawei's cloud cost operations expertise and industry best practices.

- **Operational excellence**

  The operational excellence pillar uses proven methods to create, run, and improve software effectively while ensuring great customer experiences. It covers team organization, workload design, scaling operations, and ongoing enhancements.

Application Scenarios

- **Cloud architecture governance system**

  The cloud platform integrates virtualization, database, middleware, big data, and AI technologies with industry best practices to provide hosted cloud services for enterprises. After enterprises migrate services to the cloud, they are no longer limited by their own technical capabilities and can use advanced IT technologies. They can build a modern architecture governance system based on advanced cloud platforms and WA methodologies, enabling organizations, processes, tools, and products to stay ahead in the digital era.

  The cloud architecture governance system is different from the traditional IT architecture governance system. It uses the modern cloud platform and lightweight governance system to enhance security, improve resilience, optimize resource utilization, reduce costs, and drive agile innovation.

- **Cloud architecture design**

  The cloud platform encapsulates underlying software technologies, allowing enterprises to focus more on service application design. The cloud architecture design encourages the use of domain-driven design (DDD) as the starting point for architecture designs. It integrates resilience, security, performance efficiency, cost optimization, and operational excellence into the architecture design process.

- **Cloud architecture review**

  As business requirements and technology development change, the system architecture also needs to evolve and optimize. Architects assess workload architectures against WAF best practices to confirm they meet current cloud requirements and standards. Architecture review is a continuous process. Architects are advised to review the architecture at key milestones or regularly (for example, every six months).

- **R&D productivity improvement**

  Cloud application development has high maturity in terms of technology, tools, and engineering practices. After migrating services to the cloud, enterprises should upgrade the tool chain, reconstruct the R&D process, and improve the team's cloud R&D capabilities based on cloud best practices. Introducing the advanced DevSecOps system and deterministic O&M system will greatly improve the productivity of enterprises and truly achieve service agility. Based on Huawei's 20 years of digital practice and service experience with millions of enterprise customers, Huawei Cloud draws the essence of the DevSecOps concept in the industry, extracts typical features of the DevSecOps

 Copyright © Huawei Technologies Co., Ltd.

quality and efficiency management system, and explores an effective quality and efficiency methodology and best practices with value creation as the core.

- **Highly resilient and available applications**

  Huawei summarizes and refines a series of systematic design principles and best practices based on its internal resilience design experience and industry best practices to:

  – Help customers in leveraging Huawei Cloud's infrastructure to design high availability applications with DR capabilities for various failure scenarios.

  – Reduce service interruption duration and improve availability by standardizing change, deployment, and emergency recovery processes.

- **Security compliance system**

  Cloud security has become a multi-dimensional global challenge. Huawei Cloud WAF combines the advanced cloud security concepts and accumulated network security experience and advantages in the industry, and explores a set of effective cloud security strategies and practices based on the excellent security practices of world-leading Cloud Service Platforms (CSPs). As a result, Huawei Cloud has built a multi-layered security architecture that provides in-depth defense and complies with all relevant regulations. Moreover, Huawei Cloud builds security into and continues to improve the security of its most commonly used Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) cloud services.

- **Deterministic O&M system**

  The IT O&M industry is changing like never before. We are shifting from reactive O&M for stable devices to proactive O&M for agile services. Instead of focusing on the internal network, we are focusing on customer applications. Instead of focusing on system maintenance engineers, we are focusing more on R&D engineers. Such shifts create a lot of new challenges for O&M, but also create valuable opportunities for the development of organizations and individuals. Huawei Cloud Site Reliability Engineers (SREs) have established capabilities and are also overcoming new challenges. We have a quality management system, an O&M platform, and a global expert team. More importantly, we have worked with many customers to improve their application stability and have found better ways to make applications and platforms cost-effective, high-quality, and efficient.

- **Cloud financial system (FinOps)**

  FinOps, a combination of "Finance" and "DevOps", is a management practice that organizations use to optimize the financial performance of their cloud computing infrastructure. FinOps Foundation defines FinOps as "an operational framework and cultural practice which maximizes the business value of cloud and technology, enables timely data-driven decision making, and creates financial accountability through collaboration between engineering, finance, and business teams." Cloud cost management needs continuous optimization as enterprise cloud resource consumption runs through the entire cloud adoption process.

  The FinOps framework defines a simple lifecycle with three phases: Inform (cost visibility), Optimize (cost optimization), and Operate (cost operations), to guide enterprises through cost optimization. Enterprises need to balance cost, quality, and efficiency during cost optimization to prevent extremely low costs from affecting business efficiency and stability. Within an organization,

different service teams may be at varying phases of implementing FinOps practices. The FinOps framework guides enterprises to manage cloud costs in a refined manner through collaboration across teams and data-based decision-making. Costs of each service team are visible, enabling proactive control over overspending and waste. Enterprises make data-based decisions on cloud investment to prioritize spending on essential and strategic services. With FinOps, enterprises can continuously reduce the unit business cost.

- **Application optimization**

  Many current enterprise applications hinder business growth. Outdated, complicated, and inflexible systems are hard to upgrade. Maintaining costly infrastructure increases expenses. Complex deployment processes delay releases. Existing architectures and technologies fail to meet modern software needs. These issues create significant challenges for enterprises. But most enterprises still rely on these applications as essential components of their operations to deliver key functions and store critical data. Developers and operations personnel managing existing applications are dealing with growing IT complexity, rising tech debt, skill gaps, and security threats. These issues can slow innovation and block business success.

  To address these challenges, WAF provides optimization suggestions for enterprises that have a large number of existing applications. Enterprises can improve the resilience, security, performance, and resource utilization of existing applications, adapt to modern software development, and reduce operations costs.

- **Partner Competency Certification**

  Huawei Cloud defines and maintains a full set of partner competency certifications. Partners meeting the necessary skill requirements will obtain such certifications.

  By learning WAF, partners can understand and refer to the best practices of each pillar on the cloud to obtain more professional knowledge about cloud architecture designing. Partners should follow these best practices when designing solutions or offering professional services for customers to enhance architecture design quality and improve workloads consistently. Partners who submit real use cases and pass Huawei Cloud's review will receive certifications for their expertise in specific domains, scenarios, or industries.

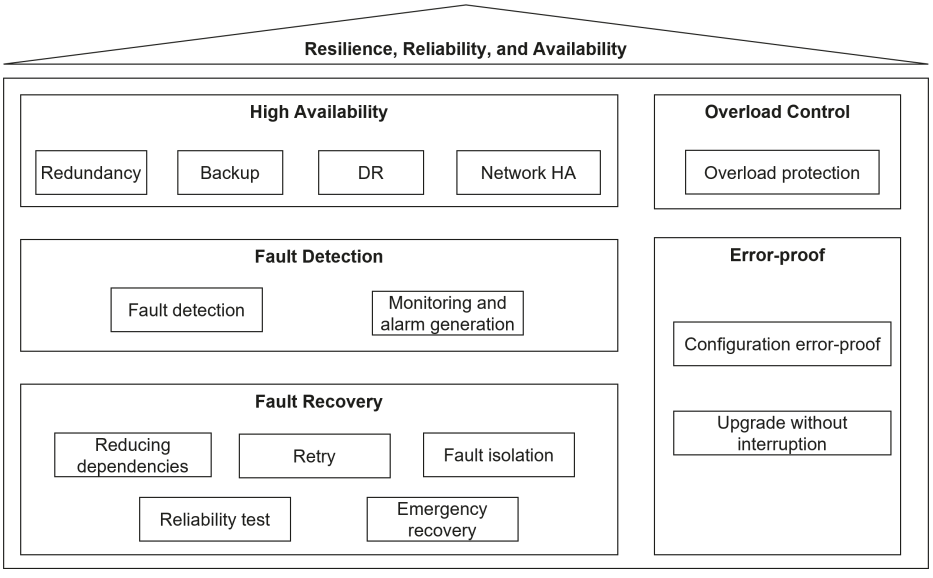
 Copyright © Huawei Technologies Co., Ltd.

2 Resilience Pillar

2.1 Introduction to the Resilience Pillar

The resilience pillar leverages Huawei's expertise in resilience design and industry best practices to help enterprises build highly available, resilient application systems on Huawei Cloud. A highly resilient system can maintain acceptable service levels even in abnormal scenarios. By standardizing change management, deployment, and emergency recovery processes, enterprises can reduce service disruptions and enhance availability.

The following figure shows the framework of the resilience pillar.



2.2 Concepts

 Copyright © Huawei Technologies Co., Ltd.

2.2.1 Concepts

Concept	Description
Resilience	Resilience is a measure of how likely a system is able to withstand and recover from failures and remain in a known operational state (even in a degraded state) in the event of faults. Resilience indicates how likely core functions and data can be restored to recover services.
Reliability	Reliability is a measure of how likely a product is able to perform its functions for a specified period of time under predefined conditions.
Availability	Availability is the percentage of time that a product is available for use.
Service-level indicator (SLI)	An SLI is a quantitative measurement of a particular aspect of a service's performance. You can use an SLI to measure your service's response to requests.
Service-level objective (SLO)	An SLO is a target or goal set for an SLI. For example, an SLO might specify that the request response success rate should be greater than *xx*% within a certain period of time, or the percentage of uptime should be higher than *xx*%.
Service-level agreement (SLA)	An SLA is a formal contract between a service provider and its customers that defines expected service performance, including penalties or compensations if the agreed-upon levels are not met.
Recovery point objective (RPO)	RPO is the maximum acceptable length of time that data can be lost in the event of an incident.
Recovery time objective (RTO)	RTO is the maximum acceptable length of time that an application can be unavailable after an incident. It is the maximum time a system takes to recover from a fault.

There is no industry-wide agreed upon definition of resilience. In a narrow sense, resilience is the ability to automatically or quickly recover from faults. In a broader sense, resilience also includes fault tolerance.

Fault tolerance is the ability of a system to continue providing services when one or more faults occur in some components of the system. More broadly, resilience encompasses not only fault recovery but also fault tolerance—the capacity to continue functioning despite failures.

Reliability is also categorized into narrow-sense reliability and broad-sense reliability. In a narrow sense, reliability engineering focuses on enhancing a system's ability to operate fault-free. In a broad sense, reliability engineering not only enhances reliability but also improves fault recovery (known as

 Copyright © Huawei Technologies Co., Ltd.

maintainability), along with other fault-related attributes such as availability and supportability.

Broadly speaking, there are no significant differences between resilience and reliability. The only difference is their focus. Reliability focuses on minimizing faults and ensuring a system operates without failures. Resilience, on the other hand, acknowledges that faults are inevitable and focuses on minimizing their impacts and recovering quickly.

2.2.2 Application Resilience

Application resilience refers to an application system's ability to provide and maintain an acceptable level of service despite various abnormal scenarios. These include infrastructure faults (such as database exceptions), external attacks (like DDoS attacks exceeding preset traffic limits), external dependency failures (such as access timeouts or unavailability), and regional disasters (such as large-scale power outages and floods).

Resilience design aims to ensure that:

- The system has a high-availability architecture, for example, there are no single points of failure (SPOFs).
- The system can quickly recover from faults, such as data loss, device faults, or site faults.

Compared with traditional data centers, Huawei Cloud provides infrastructure and cloud services with high availability, auto scaling, automatic backup, cross-AZ disaster recovery (DR), and cross-region DR, enabling customers to build highly reliable systems.

- EVS and OBS use distributed storage to prevent the impacts of hardware faults with a single disk, server, or rack.
- RDS provides automatic data backup as well as cross-AZ and cross-region data replication and switchover.

However, even if application systems on the cloud platform have these high-availability capabilities, they still need to have the capability of recovering from various occasional faults.

- The application systems need to be able to retry to re-establish links if hardware faults cause such links to instantaneously interrupt during a high-availability switchover or a cross-AZ switchover.
- The application systems need to be able to control traffic if service overload occurs due to sudden external traffic bursts.
- Some workloads heavily depend on hardware, such as local hard disks and GPUs. If the hardware is faulty, services are interrupted. In this case, the application systems need to build their own high-availability capabilities.

Different application systems have varying availability requirements, necessitating tailored resilience solutions to meet their specific needs.

2.2.3 Shared Responsibility Model for Resiliency

Resiliency is a responsibility shared between you and Huawei Cloud. The resilience of a cloud application system depends on the cloud infrastructure and the

 Copyright © Huawei Technologies Co., Ltd.

application system itself. If either is faulty, the application system is unable to provide services.

- Huawei Cloud is responsible for resiliency of the infrastructure. This infrastructure comprises the hardware, software, networking, and facilities that run Huawei Cloud services. Huawei Cloud uses all commercially reasonable efforts to make these cloud services available, ensuring service availability meets SLAs.
- Your responsibility is determined by the Huawei Cloud services that you select. This determines the amount of configuration work you must perform as part of your resiliency responsibilities. You can refer to the design principles and best practices in this white paper to build highly available application systems with detection and recovery capabilities for various abnormal scenarios.

2.2.4 Availability Objectives

2.2.4.1 Availability and SLO

Availability comprehensively measures reliability and resilience.

Availability is the percentage of time (such as 99.9%) services are up over a certain period of time (typically a month or year).

Availability = Available for use time/Total time x 100%

Availability is typically represented by a number of 9s or a combination of 9s and 5s. For instance, three 9s indicate an availability of 99.9%, while three 9s and one 5 indicates an availability of 99.95%.

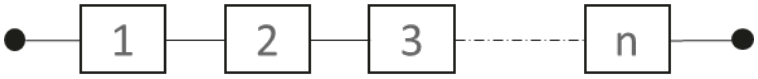
The system availability objective is defined by a service level objective (SLO). Different application systems have different availability objectives. Defining the objectives is essential for assessing the resilience of an application system. The following table describes the SLOs of common IT systems:

SLO	Maximum Unavailability (per Year)	Typical IT Services
99%	3.65 days	Batch processing, background tasks, and data extraction
99.9%	8.76 hours	Internal knowledge management systems and project tracking systems
99.95%	4.38 hours	Customer account management services and information management services
99.99%	52.56 minutes	E-commerce, B2B web services, and heavy-traffic media/content websites
99.999%	5.26 minutes	Banking, investment, finance, government, telecoms, and critical enterprise applications

 Copyright © Huawei Technologies Co., Ltd.

The system availability depends on the availability of each service unit in the system. There are two typical reliability models for service units:

- Serial model: If any unit fails, the entire system fails.



$$R_S = \prod_{i=1}^{n} R_i$$

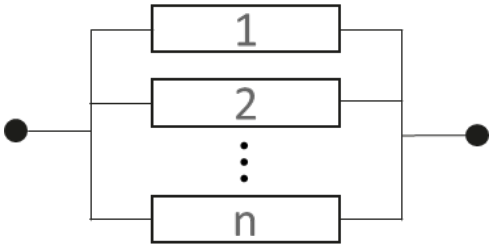
Mathematical model for reliability:

For example, if a system consists of two units connected in series, each with an availability of 99.9%, the system's overall availability is calculated as follows: Rs = 99.9% × 99.9% = 99.8%.

The availability of the serial system is lower than that of a single unit in the system. To improve the system availability:

  - Minimize the number of units in the system.
  - Improve the availability of each unit. Reduce their failure rates as much as possible.

- Parallel model: The system only fails if all of the units it comprises fail.



$$R_P = 1 - \prod_{i=1}^{n} (1 - R_i)$$

Mathematical model for reliability:

For example, if a system consists of two units connected in parallel, each with an availability of 99.9%, the system's overall availability is calculated as follows: Rs = 1 – (1 – 99.9%) x (1 – 99.9%) = 99.9999%.

Parallel connections can greatly enhance system availability. Common parallel technologies include active/standby, clusters, active-active, and multi-active setups.

To achieve an application system's availability objectives, you need to systematically define the availability requirements for its components and dependent components.

- **Availability requirements for dependent components**: The SLO of key dependent components should be one 9 higher than that of other services. For instance, if an application system's SLO is 99.9%, its key dependent components must achieve an SLO of 99.99%.

 Copyright © Huawei Technologies Co., Ltd.

- **Breakdown of the application system SLO** : Break down the system SLO, fault frequency, and cloud service SLA to determine how long application components can tolerate an interruption. Then, further break down these requirements to establish fault detection, manual intervention, and recovery time targets.
- **Enhancement of weak links in the application system**:
  - If the cloud service SLA is insufficient, strengthen protection and enhancements at the application layer.
  - Enhance availability with component redundancy and fail-back mechanisms. For example, set up an ELB cluster to add redundancy for ELB, and configure the system to fail back temporarily to SMN if there is ever a problem with DMS access.

2.2.4.2 RTO and RPO

Recovery Time Objective (RTO) and Recovery Point Objective (RPO) are two critical metrics used to define the recovery objectives in a disaster recovery plan.

- RTO is the maximum acceptable length of time that your application can be offline. Your RTO determines what DR method you will need for your application: cold backup, warm backup, or hot backup.
- RPO is the maximum acceptable length of time during which data might be lost from your application due to a major incident. RPO determines the backup frequency and method (whether online or offline, synchronous or asynchronous).

2.2.4.3 Data Durability

Data durability is a measure of how resilient your data is against loss within a certain period. It can be used to measure the reliability of a storage system. It is only a measure of how likely you are to lose data. It does not measure how much data might be lost. Data durability is estimated for a one-year period.

The main factors that affect data durability are the number of redundant copies, disk failure rate, and how long it takes to recover data. Each additional copy can increase data durability by 99% to 99.9%. Cloud object storage typically uses three copies to provide 11 to 12 nines of data durability.

2.2.5 Availability Requirements

As shown in the table *SLOs for Common IT Systems*, different IT systems have different SLOs. Not all application systems need to meet the highest availability requirements.

Higher availability targets cost more to achieve, so you need to balance resilience and costs when defining your actual targets.

Once the system's availability target is set, you can follow resilience best practices to optimize it to achieve the desired availability.

2.3 Design Principles

Faults, such as hardware failures, software errors, network delays, or any other anomalies triggered by traffic surges, are inevitable. Therefore, high availability

(HA) application systems must be designed with the possibility of failure in mind for all hardware and software at the layers of IaaS, PaaS, and SaaS, as well as in other application systems. The objective of resilience design is not to prevent these faults, but to minimize their impacts on the system and ensure continuous and stable operations. The design principles below need to be followed:

HA

A single point of failure (SPOF) may cause an entire system to break down, affect main functions, delay tasks, or cause major faults. To address this problem, the HA design of a system is critical.

HA design mainly uses redundancy or even multi-level redundancy, including remote DR, to ensure that no single point of failure exists in the case of a disaster.

- Redundancy mechanism: If possible, redundancy or even multi-level redundancy (such as 1+1, N+1, or N-way redundancy) must be considered for key components.
- Remote DR: An example is two-site three-center DR. This ensures that services can be provided even if a disaster occurs.
- Data redundancy: Periodic backup and multi-copy backup can be used to improve data durability and ensure data consistency.

The increase of redundancy means the increase of costs, which must be considered during HA design.

Comprehensive Fault Detection

Fault detection is the prerequisite of fault management. Both comprehensive detection and fast detection are important. Generally, the former is more important than the latter.

Fault detection involves the following aspects:

- Detection scope: All components are identified and tracked, with a focus on detecting faults that have a major impact.
  - Subhealth detection: Subhealth exceptions that do not cause system faults but cause system or service KPIs to deteriorate need to be detected, such as increased network latency, slow disks, and memory leakage.
  - Standby module detection: In a redundancy system, faults of both the active and standby modules need to be detected to avoid silent faults.
  - Components with lifespan limits: The health status of components with lifespan limits (such as local hard disks) must be monitored in a timely manner. Maintenance measures must be taken based on warnings to prevent serious impacts caused by unexpected faults.
- Detection speed: An appropriate detection speed needs to be determined based on overall service requirements.
- Detection impact: The interval of periodic fault detection must be determined based on its impact on the CPU usage and the impact of detection delay on the service recovery speed.
- Streamlined detection: The system and modules used for fault detection need to be simpler than the detected system and modules.

 Copyright © Huawei Technologies Co., Ltd.

After a fault is detected, it should be reported to the monitoring system and rectified in a timely manner.

Rapid Fault Recovery

Fault recovery refers to the capability of restoring a product's intended functionality after it was struck by a fault. Generally, the faster the recovery, the less the impact.

A proper fault recovery solution need to be designed based on the service requirements, technical implementation difficulty, technical solution complexity, and cost.

- Automatic recovery: If a fault is one that would affect services, the system should be able to recover from the fault automatically by means of protection switching, partial reset, or system services.
- Priority recovery: Faults that have a high probability of occurrence and great impact are preferentially rectified.
- Hierarchical reset: Resource reset at a lower level is designed to minimize the impact on services.
- Non-coupling recovery: The system startup should not be affected by partial system faults or component startup sequences.
- Layered protection: Network layering must be considered for system fault protection. The protection switching at the lower layer must be more sensitive than that at the upper layer to avoid ping-pong switching.

Monitoring the system status or key metrics of the system load helps determine whether a fault occurs in the system. If it does, the fault can be automatically rectified.

Fault analysis methods can be used to analyze fault types and their impacts and hazards and to design corresponding reliability and availability solutions, providing capabilities such as redundancy, isolation, degradation, and elasticity. In addition, fault injection testing (FIT) can be used to verify the effectiveness of reliability and availability solutions, maximizing service reliability and availability.

For some faults, after redundancy and automatic recovery are performed using various technical means, services may still be interrupted, requiring manual intervention such as backup recovery or DR. Therefore, an efficient emergency recovery process and platform must be established to quickly recover services and reduce the impact of faults.

Overload Control

When the request load exceeds the system capacity, requests may fail due to resource saturation. In the cloud, the system and resource usage can be monitored, so resources can be automatically added or deleted to maintain the optimal level to meet service requirements, without causing over-configuration or under-configuration.

Generally, service traffic is adjusted through dynamic resource management. It is not recommended that static thresholds be used to prevent overload, because this may cause a huge resource waste. The following aspects must be considered during overload control design:

 Copyright © Huawei Technologies Co., Ltd.

- Dynamic rate limiting: The rate limit is dynamically adjusted based on the system resource usage.
- Elastic scaling: The system automatically monitors the system resource usage and adds or deletes resources.
- Load balancing before overload control: When multiple processing units are deployed, load balancing is preferentially considered to reduce service impacts caused by resource insufficiency of a single processing unit. Then, overload control is performed to maximize the processing capability of the entire system.
- Early control: When the system is overloaded, service access should be controlled at the earliest possible stage of the process, in the front-end processing modules, or at the underlying protocol level to avoid unnecessary performance overhead caused by mid-process control.
- Priority assurance: When the system is overloaded, services with higher priorities are preferentially processed to maximize social benefits.

Change Error Prevention

When the system is upgraded or the configuration is changed, human errors need to be prevented to avoid causing system and service impacts or failures.

Generally, mistake proofing is used to reduce human errors. Mistake proofing is a behavior constraint method that prevents errors. It allows operators to perform operations intuitively and accurately. Such operations do not require intense concentration, extensive experience, or professional knowledge. Mistake proofing improves efficiency and user experience in many scenarios and helps reduce damage or replacement costs. It is a basic and common feature in excellent products.

The following solutions are usually used to prevent change errors:

- Role constraints: The permissions control is used to restrict the configuration scope of different roles, preventing errors caused by unauthorized configuration.
- Separation of query and modification: The configuration options are divided into different layers through the product interface design, with the query and modification pages separated to reduce the risk of inadvertent configurations.
- Configuration validation: The configuration validation mechanism is designed to ensure that necessary checks are performed before the configuration takes effect, preventing any configuration errors from taking effect. Automatic configuration changes reduce the possibility of manual errors.
- Deletion protection: A protection mechanism is used to prevent resources from being deleted by mistake. For example, the status check before deletion, resource locking, and recycle bin mechanism.

2.4 Questions and Checklists

During application resilience design, this checklist can help identify areas for improvement and guide their enhancement. Each item in the checklist represents a best practice, which will be explained in detail in the next section.

 Copyright © Huawei Technologies Co., Ltd.

Question	Checklist/Best Practice
RES01 What redundancy technologies do you use to ensure the high availability of your application system?	1. High-availability deployment of application components 2. Multi-location deployment of application components 3. Anti-affinity for cloud servers
RES02 How do you back up critical data in your application?	1. Identify and back up critical data that needs to be backed up. 2. Automatically back up data. 3. Periodically restore data from backups.
RES03 How do you enable cross-AZ disaster recovery for your application?	1. Cross-AZ cluster deployment 2. Cross-AZ data synchronization 3. Interconnection with DR arbitrator for automatic switchover 4. Disaster recovery management
RES04 How do you deploy cross-region or cross-cloud disaster recovery for your application?	1. Define the RPO and RTO for the application system. 2. Deploy the DR system to meet the DR objectives. 3. Automate the DR process. 4. Regularly perform DR drills to confirm that the recovery can meet objectives.
RES05 How do you ensure high availability of the networks?	1. Ensure high availability for network connections. 2. Avoid unnecessarily exposing network addresses. 3. Isolate network bandwidth allocated to services using different traffic models. 4. Reserve IP resources for expansion and high availability.
RES06 How do you detect faults?	1. Fault mode analysis 2. Fault detection 3. Subhealth detection
RES07 How do you monitor application system resources?	1. Define key metrics and thresholds and monitor such metrics. 2. Monitor logging 3. Send notifications when exceptions are detected. 4. Store and analyze monitoring data. 5. Track requests end to end.
RES08 How do you reduce the impact of dependencies?	1. Reduce strong dependencies. 2. Use loose coupling to reduce dependencies. 3. Minimize the impacts of dependency failures.

 Copyright © Huawei Technologies Co., Ltd.

Question	Checklist/Best Practice
RES09 How do you design a retry mechanism?	1. Design a retry mechanism for API calls and command executions. 2. Determine whether to retry from the client based on the comprehensive evaluation results. 3. Avoid creating too much traffic pressure from excessive retries.
RES10 How do you isolate faults?	1. Isolate the control plane from the data plane. 2. Deploy application components in multiple locations. 3. Adopt a grid architecture. 4. Configure health check and automatic isolation.
RES011 How do you perform reliability tests?	1. Chaos testing 2. Load testing 3. Long-term stability testing 4. DR drills 5. Red/blue team testing
RES012 How do you perform an emergency recovery?	1. Set up an emergency recovery team. 2. Develop an emergency response plan. 3. Periodically conduct emergency recovery drills. 4. Restore services as soon as possible after a problem occurs. 5. Organize emergency recovery backtracking.
RES013 How do you implement overload protection to adapt to traffic changes?	1. Automatic elastic scaling 2. Application system load balancing 3. Overload detection and traffic throttling 4. Automatic capacity expansion 5. Quota limiting for automatic capacity expansion 6. Load testing
RES14 How do you configure error prevention?	1. Make changes foolproof. 2. Automate changes. 3. Back up data before changes. 4. Provide runbooks to standardize changes.
RES15 How do you perform upgrades without interrupting services?	1. Automatic deployment and upgrade 2. Automatic checks 3. Automatic rollbacks 4. Canary deployments and upgrades

 Copyright © Huawei Technologies Co., Ltd.

2.5 HA Design

2.5.1 RES01 Redundancy

2.5.1.1 Overview

A highly available system must avoid single points of failure (SPOFs) to prevent the entire system from being unavailable due to the failure of a single node.

This section describes the redundancy design to eliminate SPOFs and improve availability.

2.5.1.2 RES01-01 High-Availability Deployment of Application Components

All components in an application system must be deployed in high availability (HA) mode to prevent SPOFs.

- **Risk level**

  High

- **Key strategies**

  The HA deployment solution varies depending on the capabilities of each component in the application system.

  - Using native HA instances: If a cloud service supports both single-node resources and active/standby or cluster resources, the key nodes must use active/standby or cluster resources, such as CCE HA clusters, RDS primary/standby instances, DDS clusters, and DCS master/standby or cluster instances. Multiple workloads need to run on CCE clusters to prevent service interruptions caused by a single node failure.
  - Single-node instances achieve HA through multiple instances: If a cloud service only supports single-node deployment, active/standby or load balancing among multiple nodes needs to be supported at the application layer. For example, for the ECS service, you can use ELB to enable load balancing and failover of stateless services among multiple ECSs. Alternatively, set up active/standby ECSs at the application layer.
  - Hardware-dependent instances achieve HA at the application layer: If an ECS uses hardware resources, such as local disks, FPGA passthrough, and IB NIC passthrough, a hardware failure will cause the ECS to fail, and the ECS cannot be automatically recovered through the VM HA function. To address this issue, avoid using hardware resources when designing the application system. If they have to be used, implement HA at the application layer so that services can be quickly recovered if the dependent hardware is faulty.
  - VM HA: If an ECS does not depend on special resources, VM auto recovery is supported. If the physical server accommodating the ECS is faulty, the ECS can automatically restart on another physical server. Workloads deployed on such an ECS must support auto recovery upon VM restart and be able to tolerate temporary performance degradation or interruptions during the HA process.

 Copyright © Huawei Technologies Co., Ltd.

To reconstruct a deployed application system for HA, perform the following steps:

a. Determine the key components of the application system. If a key component is faulty, the entire application system or key functions will be affected.

b. Check the HA capability of key components. Specifically, check whether failover is supported if a key component is faulty.

c. Optimize the key components that do not support HA as follows:

- If a cloud service instance is a single-node instance, for example, the ECS service, you can apply for multiple ECSs to bear the same service and use ELB to implement load balancing and failover. Alternatively, enable failover of multiple ECSs at the application layer to achieve HA.
- Failover is available for ECSs that do not depend on special resources. If the physical server accommodating an ECS is faulty, the ECS can automatically restart on another physical server. For workloads deployed on such an ECS, you need to check whether services can be automatically recovered after the ECS is restarted.
- Failover is unavailable for ECSs that depend on special resources, such as local disks, FPGA passthrough, and IB passthrough. Check whether these ECSs can be replaced with common ECSs (that do not depend on these special resources) to improve the availability.
- Local disks have a limited service life and are prone to faults after long-term use. You are advised not to attach local disks to ECSs, BMSs, and MRS clusters. Instead, use EVS disks with HA. If local disks must be used, you are advised to use RAID to improve the availability of local disks and implement HA at the application layer. In this way, if an instance is faulty, failover is supported and services can be recovered.

- **Related cloud services and tools**
  - Elastic Cloud Server (ECS)
  - Bare Metal Server (BMS)
  - Elastic Load Balance (ELB)
  - Cloud Container Engine (CCE)
  - Document Database Service (DDS)
  - Distributed Cache Service (DCS)
  - MapReduce Service (MRS)

2.5.1.3 RES01-02 Multi-Location Deployment of Application Components

Application components must be deployed in multiple data centers to prevent service interruptions caused by the fault of a single data center.

- **Risk level**

  High

- **Key strategies**

  Application data and resources can be deployed in multiple locations based on different requirements.

 Copyright © Huawei Technologies Co., Ltd.

- Multi-AZ deployment: Applications should be deployed in multiple AZs to prevent service interruptions caused by the fault of a single AZ.
- Multi-region deployment: For application systems with high availability requirements, multi-region deployment is required to prevent service interruptions caused by the fault of a single region.
- Multi-AZ deployment is preferred. Most workloads can achieve their availability objectives through multi-AZ deployment in a single region. Multi-region deployment is considered only when workloads have extremely high availability requirements or other service objectives.

2.5.1.4 RES01-03 Anti-Affinity for Cloud Servers

ECSs that run the same service in an application must be deployed on different physical servers to prevent service interruptions caused by a physical server fault.

- **Risk level**

  High

- **Key strategies**
  - For multiple ECSs that bear the same service, anti-affinity needs to be configured for the group housing these ECSs so that they can be scheduled to different physical servers. This prevents all services from being unavailable due to the fault of a single physical server.
  - If ECSs are automatically scaled using Auto Scaling (AS), anti-affinity needs to be configured for the ECS group. This prevents the ECSs automatically scaled using AS from running on the same physical server.
  - If CCE cluster nodes or node pools use ECSs, you are advised to configure anti-affinity for the ECS group to prevent ECS nodes in the CCE cluster from running on the same physical server.
- **Related cloud services and tools**
  - Elastic Cloud Server (ECS): ECS group
  - Auto Scaling (AS)
  - Cloud Container Engine (CCE)

2.5.2 RES02 Backup

2.5.2.1 Overview

Backup is needed for critical data in an application system. If there are virus attacks, accidental deletions, and software and hardware faults, data can be quickly restored from backups.

A disaster recovery (DR) system replicates data in real time but lacks multiple backup points. If standby data on the active side is deleted or modified by mistake, the incorrect data will be synchronized to the standby side. In this case, DR cannot replace backup.

The recovery point objective (RPO) (the maximum acceptable amount of data loss) in backup and restore is determined by the time point when the last backup was created. Different types of data may tolerate different amounts of data loss,

 Copyright © Huawei Technologies Co., Ltd.

which means they require different RPOs. To meet the RPO requirements of data backups, use periodic automatic backup, rather than manual backup.

2.5.2.2 RES02-01 Identifying and Backing Up Critical Data That Needs to Be Backed Up

The importance of different data varies. You need to specify the importance of all data in the application system and the corresponding recovery point objective (RPO) and recovery time objective (RTO) requirements. For example, for critical data, the allowed data loss time is short, so the data needs to be backed up more frequently. For general data, the allowed data loss time is relatively long, so the data can be backed up at a low frequency. For non-critical data, the data loss does not affect services, so the data does not need to be backed up.

- **Risk level**

  High

- **Key strategies**
  - Identify all data in the application system. Data can be stored in multiple resources, such as volumes of ECSs or BMSs, Relational Database Service (RDS) and Document Database Service (DDS) databases, Scalable File Service (SFS) file systems, and Object Storage Service (OBS).
  - Classify data by importance. In an application system, different types of data vary in importance and backup requirements. For example, critical data requires near-zero RPO, so it needs to be backed up in real time. Non-critical data can tolerate data loss, so backup is not required. Some important data can tolerate different levels of data loss, so it requires tailored backup policies.
  - Design backup solutions for the data to be backed up to meet the RPO and RTO requirements.

2.5.2.3 RES02-02 Automatically Backing Up Data

You can set a periodic backup policy to automatically back up data based on the RPO requirements of the data that needs to be backed up.

- **Risk level**

  High

- **Key strategies**

  Use Huawei Cloud backup services or third-party backup software to back up data, and set the automatic backup frequency based on the RPO requirements. Cloud Backup and Recovery (CBR) can be used to back up ECSs, BMSs, EVS disks, SFS Turbo file systems, and file directories. Most cloud services, such as RDS, DDS, and DCS, have native backup functions. KooGallery also provides various backup software to back up data.

  Huawei Cloud supports workload data backup. Typical backups include:

  - CBR backups: CBR provides snapshot-based backup and restore for disks (EVS disks) and servers (ECSs, Flexus instances, and BMSs), supports hybrid cloud backups, and can back up SFS Turbo file systems and databases (such as MySQL and SAP HANA) that are deployed on cloud servers. CBR supports one-off backup and periodic backup. Currently, the

backup time can only be on the hour. You can select multiple hours for backup. The minimum RPO is 1 hour. You need to select a proper backup period based on the data importance.

- Automatic database backup: Database services, such as RDS, DDS, and GaussDB, support default automatic backup. Incremental backup is automatically performed for instances every 5 minutes to ensure database reliability.
- DCS backup: DCS provides automatic and manual backup functions for non-single-node instances. You are advised to set an automatic backup policy.

In addition, you can use third-party backup software to back up data.

In Huawei Cloud, data is backed up to OBS to ensure high security.

- **Related cloud services and tools**
  - Cloud Backup and Recovery (CBR)
  - Relational Database Service (RDS)
  - Distributed Cache Service (DCS)

2.5.2.4 RES02-03 Periodically Restoring Data from Backups

You can periodically restore data to verify the integrity of backups and availability of the restoration process, and check that the data loss time and restoration time meet the RPO and RTO requirements.

- **Risk level**

  High
- **Key strategies**
  - Periodically restore data from backups to verify the integrity of the backups.
  - To prevent backup and restore from affecting production services, you can build a test environment and restore data from existing backups.

  Huawei Cloud allows you to manually restore data. You can periodically do this to test the restoration.
- **Related cloud services and tools**
  - Cloud Backup and Recovery (CBR)
  - Relational Database Service (RDS)
  - Distributed Cache Service (DCS)

2.5.3 RES03 Cross-AZ DR

2.5.3.1 Overview

Huawei Cloud Availability Zone (AZ) capability allows you to deploy a DR architecture for an application at a low cost to prevent service interruptions caused by a single-AZ fault. Application systems can be designed to access resource pools across multiple AZs. They use the cross-AZ data replication and switchover provided by cloud service instances or supported by applications to replicate data, balance loads, and perform failover across AZs. This helps application systems handle failures within specific AZs effectively.

 Copyright © Huawei Technologies Co., Ltd.

2.5.3.2 RES03-01 Deploying Clusters Across AZs

All components in an application are deployed across AZs for DR to prevent service interruptions in case of a single-AZ fault.

- **Risk level**

  High

- **Key strategies**
  - If a cloud service supports cross-AZ HA instances, use the cross-AZ HA instances of the cloud service first.
  - If a cloud service supports only single-AZ instances and does not support cross-AZ HA instances, use other cloud services or the application layer for cross-AZ DR. The following uses ECS as an example:
    - For stateless ECSs, use the cross-AZ scaling capability of AS or the cross-AZ load balancing capability of ELB for cross-AZ HA. If an AZ is faulty, services can be automatically and quickly switched over to another available AZ.
    - For stateful ECSs or BMSs, you are advised to implement cross-AZ DR at the application layer. Use automatic cross-AZ switchover or DR management tools for automatic DR switchover to reduce manual operations during disasters.

  To reconstruct a deployed application system to a cross-AZ instance, perform the following steps:

  a. Determine the key components of the application system. If a key component is faulty, the entire application system or key functions will be affected.

  b. Check the cross-AZ HA capability of the key components. Ensure that services of key components can automatically switch to another AZ during failures for seamless recovery.

  c. Optimize the key components that do not support HA as follows:

  - For services like RDS, DDS, and DCS that support cross-AZ HA instances and support reconstruction to cross-AZ HA instances, upgrade the single-AZ instances to cross-AZ instances.
  - For services like dedicated ELB, CCE clusters, DMS, and OBS buckets that support cross-AZ HA instances but cannot be reconstructed to cross-AZ HA instances, apply for new cross-AZ HA instances to replace the single-AZ instances.
  - For services like ECS using single-node instances, apply for multiple instances in multiple AZs to bear a service, and use cross-AZ ELB for cross-AZ load balancing and failover. Alternatively, use failover of cross-AZ instances at the application layer for cross-AZ HA.

- **Related cloud services and tools**

  Most cloud services of Huawei Cloud support multi-AZ instances, such as ELB, AS, CCE, DCS, DMS, RDS, and GaussDB. If one AZ is faulty, the other AZ can quickly take over services to keep services running smoothly.

 Copyright © Huawei Technologies Co., Ltd.

2.5.3.3 RES03-02 Synchronizing Data Across AZs

Stateful services require data synchronization between AZs to prevent data loss during an AZ failure. Stateless services do not need this synchronization.

- **Risk level**

  High

- **Key strategies**
  - If the cloud service for an application component supports cross-AZ HA instances, enable the cross-AZ data synchronization of the cloud service, such as RDS databases, DCS instances, or OBS buckets.
  - If the cloud service for an application component does not support cross-AZ HA instances but supports cross-AZ data synchronization through a synchronization service, use that service to synchronize data across AZs. For example, ECSs with stateful data do not support cross-AZ HA, but Storage Disaster Recovery Service (SDRS) can be used for cross-AZ data synchronization.
  - If the cloud service for an application component does not support cross-AZ HA instances and does not support or use cross-AZ data synchronization, such as BMSs with stateful data, replicate data at the application layer.
- **Related cloud services and tools**
  - Storage Disaster Recovery Service (SDRS)
  - ECS
  - RDS
  - DCS
  - OBS

2.5.3.4 RES03-03 Interconnecting with DR Arbitrator for Automatic Switchover

If stateful active/standby services deployed in different AZs support automatic switchover, DR arbitration needs to be interconnected to avoid dual-active or dual-standby deployment. In this way, if a link between AZs is interrupted, services can be automatically switched to another AZ without being affected. Cluster services are not involved.

- **Risk level**

  High

- **Key strategies**
  - Interconnect with DR arbitration for stateful active/standby services. In case of link disconnections between sites, there is no dual-active deployment, and data integrity remains unaffected.
  - Connect all related application components to the consistency arbitration. If the link is disconnected, all components can be switched to the same site to maintain E2E service availability.

 Copyright © Huawei Technologies Co., Ltd.

2.5.3.5 RES03-04 Supporting DR Management

DR management monitors DR status and RPO metrics while enabling service switchover during disruptions.

- **Risk level**

  High
- **Key strategies**
  - Monitor the DR status in real time to understand the DR operational status.
  - Verify application-level data, compare synchronization data between AZs, and monitor RPO metrics.
  - In typical deterministic fault scenarios, automatic DR or switchover does not require manual intervention. It keeps services running smoothly while meeting the RPO/RTO requirements.
  - In typical sub-health fault scenarios, service degradation or proactive switchover ensures service continuity.
- **Related cloud services and tools**
  - Multi-Site High Availability Service (MAS)

2.5.4 RES04 Cross-Region/Cross-Cloud DR

2.5.4.1 Overview

A DR system ensures full data protection to avoid regional disasters or meet cross-cloud DR needs. In case of a site-level disaster, this system reduces data loss for the production system. It allows the DR center to quickly take over services, keeping smooth service operations and minimizing damage.

In cross-region DR, application systems can be deployed across multiple regions with data replicated among the regions. This allows for disaster recovery if a region faces service disruptions or data loss.

In cross-cloud DR, if an application system has been deployed in an IDC or other clouds, you can deploy another application system in Huawei Cloud and replicate data from the IDC or another cloud to Huawei Cloud. This enables disaster recovery if the entire IDC or cloud fails or data gets lost.

2.5.4.2 RES04-01 Defining the RPO and RTO for an Application System

Before designing DR, determine the DR objectives of an application system based on the importance of the application system. Typically, the DR objectives are defined by the RPO and RTO metrics.

- RPO: indicates the allowed volume of data loss, which is related to the data replication period or continuous replication delay.
- RTO: indicates how long services can be interrupted during recovery. It depends on how the services are deployed and switched over at the disaster recovery site.
- **Risk level**

  High

 Copyright © Huawei Technologies Co., Ltd.

- **Key strategies**

  Service importance varies depending on the service systems. Define the importance of various services in an application system and the RPO/RTO requirements. Core services require minimal downtime to ensure quick service recovery and keep service continuity, so use active-active or multi-active DR for core services. Important services allow a certain duration of service interruptions, so use active/standby DR for important services. General services tolerate service interruptions in days, so use remote backups for general services. Interruptions of unimportant services do not have impact on customers, so they do not require DR.

2.5.4.3 RES04-02 Deploying a DR System to Meet the DR Objectives

Choose the right DR plan for each application system based on the DR objectives by considering the interruption probability and DR costs.

- **Risk level**

  High

- **Key strategies**

  In cross-region or cross-cloud DR scenarios, use a proper DR plan like remote backup, active/standby DR, or active-active DR, based on the availability needs. Deploy the production site on another cloud, in an IDC, or in a Huawei Cloud region as required.

  – Remote backup: Back up important data from the production site to a remote Huawei Cloud DR region. In case of a disaster at the production site, deploy a new service system in the remote DR region and use the latest backup data to restore data and services.
  – Active/Standby DR: Deploy a service system at both the production site and Huawei Cloud DR region. Replicate important data from the production site to the DR region asynchronously. Normally, the production site handles all services. In case of a disaster at the production site, the DR region is promoted to the primary site, and service traffic is switched to it.
  – Active-active or multi-active DR: Deploy a service system at both the production site and the Huawei Cloud DR region. Replicate important data between the sites asynchronously. Both sites provide services at the same time, with a global load balancer distributing traffic. If one site fails, the other takes over services seamlessly.

  To support cross-region active/standby DR for a region with an application system deployed, perform the following steps:

  a. Select another region as the DR region and deploy a set of the same application system with its workloads and database instances.
  b. Replicate key data between regions using either a cloud service or the application system.

     ▪ If cloud service instances allow cross-region DR, replicate data between the production site and the DR region. For example, for RDS database instances, apply for a DRS instance to replicate data between the primary region and the DR region in real time. For OBS buckets, replicate their data from the primary region to the DR region.

        - If cloud service instances do not allow cross-region DR but have critical data, enable cross-region data dual-write at the application layer to synchronize data.
    c. Apply for an external IP address for both the primary region and DR region, respectively, on the access side. Use DNS to resolve the application system to the external IP address in the primary region. If the primary region fails, update the DNS record to direct traffic to the external IP address of the DR region.
    d. Use the MAS service for DR orchestration to quickly switch services between the active and standby sites in case of a disaster.
- **Related cloud services and tools**
  - Cloud Backup and Recovery (CBR): allows data replication and recovery across regions.
  - Data Replication Service (DRS): allows real-time DR for databases such as RDS for MySQL and GaussDB(for MySQL). It supports both cross-region and cross-cloud DR scenarios.
  - Object Storage Service (OBS): supports cross-region replication and active-active DR.

2.5.4.4 RES04-03 Automating the DR Process

The DR process involves restoring services at the DR site, switching between primary and standby databases, and redirecting traffic to the DR site. DR management enables monitoring of DR status and RPO while allowing automated one-click switchovers during disasters. This reduces manual interventions.

- **Risk level**

  High
- **Key strategies**
  - Monitor the DR status in real time to understand the DR operational status.
  - Verify application-level data, compare synchronization data between AZs, and monitor RPO metrics.
  - Enable one-click automatic switchovers in disaster scenarios to reduce manual interventions and meet RPO/RTO requirements.
  - Orchestrate the DR process and perform DR drills.
- **Related cloud services and tools**
  - MAS

2.5.4.5 RES04-04 Regularly Performing DR Drills to Check Whether the Recovery Can Meet DR Objectives

Regular DR drills can verify whether the DR system is available and whether the data loss time and recovery time meet the RPO and RTO requirements.

- **Risk level**

  High
- **Key strategies**

 Copyright © Huawei Technologies Co., Ltd.

  - Conduct at least one DR drill every year to improve the proficiency of operators.
  - Time the recovery process during the drill to check whether the application system meets the RPO and RTO requirements.
  - Review and optimize the DR plan execution sequence and recovery time during the drill.
- **Related cloud services and tools**
  - MAS

2.5.5 RES05 Network HA

2.5.5.1 Overview

An application system relies on the network for internal and external communications. Any network exceptions can result in services interruptions. This makes the HA and DR capabilities of the network architecture crucial. The network design for an application system involves internal and external network connections, IP address management, and domain name resolution.

Huawei Cloud network HA involves the following scenarios:

- Public cloud network: HA for public network connections related to application systems can reduce service interruptions caused by network disconnections.
- Hybrid cloud network: For users using self-built local data centers (IDCs) or other clouds, hybrid cloud network interconnection becomes essential when they need to deploy some services on Huawei Cloud to meet their service development needs. If application systems are deployed across clouds (for example, active/standby DR or active-active DR across clouds), hybrid cloud network connections with HA enabled are required to reduce service interruptions caused by network disconnections.
- Inter-VPC access: When a service system spans multiple departments or teams, separate VPCs isolate their services. These teams and departments must connect across VPCs for mutual access.

2.5.5.2 RES05-01 Ensuring High Availability of Network Connections

Reliable external network connections prevent service interruptions when applications support external systems.

- **Risk level**

  High
- **Key strategies**
  - Network link redundancy: Multiple network paths must be supported to ensure high availability. If one connection fails, traffic will automatically switch to another available path.
  - Fast network link switchover: The system quickly switches services to a working network path upon a link failure by regularly checking for network connection issues.

  Public cloud networks rely on multiple EIPs and DNS for reliable connections. For high availability needs, smart DNS monitors EIP issues and switches

services automatically upon a failure. Redundant DNS systems are also required to prevent service disruptions from DNS failures or errors.

Link redundancy and switchover solutions in hybrid cloud networking scenarios:

- Redundancy with two Direct Connect connections: Your on-premises data center connects to a Huawei Cloud VPC over two connections that are terminated at two Direct Connect locations in the same region. BGP routes are used to route traffic between your on-premises data center and the VPC you want to access. You can set the virtual interface priorities to determine the active and standby links for services.

  For details, see **Connecting an On-Premises Data Center to a VPC over Two Connections in Load Balancing Mode (Virtual Gateway)**.
- Dual-VPN redundancy: Your on-premises data center connects to a Huawei Cloud VPC over two VPN connections to ensure reliability. If one VPN connection is faulty, the system can switch to the other VPN connection to ensure network continuity. The two VPN connections can be deployed in active-active or active/standby mode.

  For details, see **Connecting an On-premises Data Center to a VPC on the Cloud Through VPN (Active-Active Mode)** and **Connecting an On-premises Data Center to a VPC on the Cloud Through VPN (Active/Standby Mode)**.
- Direct Connect/VPN connections in active/standby mode: Your on-premises data center connects to a Huawei Cloud VPC over both Direct Connect and VPN connections working in active/standby mode. You can attach the connections to an enterprise router for automatic switchover between active and standby links. If the active connection is faulty, services are automatically switched to the standby one. This reduces the impact of network interruptions on services and O&M costs.

  For details, see **Allowing Direct Connect and VPN to Work in Active and Standby Mode to Link Data Center to Cloud**.

- **Related cloud services and tools**
  - Direct Connect
  - Virtual Private Network (VPN)

2.5.5.3 RES05-02 Avoiding Unnecessarily Exposing Network Addresses

There may be security risks when network addresses are exposed. Therefore, avoid exposing unnecessary network addresses.

- **Risk level**

  High
- **Key strategies**
  - Centrally manage external network addresses to avoid exposure. For example, use ELB, NAT Gateway, and WAF for public network access.
  - Restrict access to external IP addresses through network ports using security groups and NAT gateways to reduce security risks.
- **Related cloud services and tools**
  - VPC: security groups

 Copyright © Huawei Technologies Co., Ltd.

  - Elastic Load Balance (ELB)
  - NAT Gateway
  - Web Application Firewall (WAF)

2.5.5.4 RES05-03 Isolating Network Bandwidth Among Services of Different Traffic Models

If services of different traffic models share network bandwidth, one service can disrupt others through traffic preemption.

- **Risk level**

  High

- **Key strategies**
  - Services of the same traffic model can share the network bandwidth. The shared bandwidth must satisfy all connected services' needs.
  - Use separate bandwidth instances for services with different traffic models to avoid interference.
  - Use separate domain names for services with different features.

2.5.5.5 RES05-04 Reserving IP Resources for Expansion and HA

The cloud networks must support scalability and high availability to provide sufficient resources during automatic scaling or service expansion on the cloud.

- **Risk level**

  High

- **Key strategies**

  The cloud network planning and design must meet the following requirements:
  - Plan separate VPCs for each region based on service needs. Assign independent address spaces to each VPC. Reserve IP address spaces for creating VPCs.
  - Plan subnets and IP address spaces for each VPC based on service needs. Reserve IP address spaces for creating subnets.
  - Reserve IP address spaces for each subnet for network capacity expansion.
  - Connect your network to others like VPCs, IDCs, or other clouds only if their IP address spaces do not conflict.

2.6 Comprehensive Fault Detection

2.6.1 RES06 Fault Detection

2.6.1.1 Overview

A high availability system must have comprehensive fault detection capabilities to identify incidents that may cause faults, develop, activate, and have potential faults. Fault detection is essential for fault recovery in nearly all cases.

2.6.1.2 RES06-01 Fault Mode Analysis

Fault mode analysis is a design method that analyzes all potential fault modes of each component and their impact on product functions, classifies each potential fault mode based on its severity, identifies single points of failure and product weaknesses, and proposes preventive measures during system analysis and design to improve product reliability.

Huawei Cloud manages infrastructure faults. If application systems are deployed on Huawei Cloud, you do not need to pay much attention to detect and rectify faults of infrastructure such as data enters, power supplies, environments, compute servers, storage devices, and network switches. However, the impact of these infrastructure faults on application systems and the corresponding recovery measures must be considered, including data center (AZ- or region-level disaster) disasters, compute server failures or restarts, hard disk issues or subhealth, network communication interruptions, and packet loss. For application-related fault modes, such as software system faults, data faults, communication faults, overloads, and human errors, sufficient analysis and detection and recovery measures must be provided.

- **Risk level**

  High

- **Key strategies**

  Analyze the occurrence frequency and impact of each fault mode to determine the severity level. For fault modes of components with single points of failure, the severity must be high. Common cloud service fault modes include CPU overload, memory overload, high disk usage, data issues (such as accidental deletion), AZ failures, and region failures.

  a. Define severity levels.

     Severity measures the adverse impact of a fault on a system. The severity has four levels, including critical, major, minor, and warning.

     - Critical: Faults of this level will cause a system crash or severely affect the key functions of the system.
     - Major: Faults of this level will affect the key functions of a system, cause task delay, or cause minor system damage or severe potential faults.
     - Minor: Faults of this level will impair the secondary functions of a system or make these functions unavailable, but do not impact key functions. Immediate rectification is required.
     - Warning: Faults of this level will impair some secondary functions of a system (regular maintenance is required), but will not affect the key functions of the system. These faults include alarms or indicator failures.

 Copyright © Huawei Technologies Co., Ltd.

Critical and major faults are also called single points of failures (SPOFs). Critical faults may also arise the security concerns and cause failures in all or most functions. Major faults mainly affect the major functions. Minor faults require immediate rectification.

Generally, if a fault cannot be detected, it is considered as a potential fault, whose severity level should be increased by one level.

b. Identify all components and functional modules in a system.

Identify all components and external dependencies of an application system, such as providers and third-party services.

c. Identify faults.

Identify potential faults for each component. A single component may have multiple fault modes, and each fault mode must be analyzed. All fault modes must be included. If a fault mode is missing, it may not be considered in the design, leaving it unmonitored with no recovery solutions in place.

d. Analyze fault impact scope (blast radius).

Analyze the occurrence frequency and impact of each fault mode to determine the severity level. For fault modes of components with single points of failure, the severity must be high. Common cloud service fault modes include CPU overload, memory overload, high disk usage, data issues (such as accidental deletion), AZ failures, and region failures.

e. Provide fault detection and mitigation measures.

f. For each fault mode, analyze how to detect and rectify the fault, propose improvement suggestions, comprehensively consider the system complexity and cost, and preferentially resolve the fault mode with a higher severity level.

- **Related cloud services and tools**
  - Cloud Operations Center (COC): supports fault mode management.

2.6.1.3 RES06-02 Fault Detection

Automatic detection is required for all fault scenarios to detect and rectify faults in a timely manner.

- **Risk level**

  High

- **Key strategies**
  - Detect all faults.
  - Detect faults by region, AZ, service, method, instance, or container ID. Keep detection dimensions aligned with fault recovery modes.
  - Ensure alarms are generated or faults are automatically rectified after faults are detected.

  Based on detection types, fault detection can be classified into resource detection, function detection, and service detection.

  - Resource detection detects virtualized physical hardware resources and corresponding software resources in a cloud environment, including CPU, memory, network, and disk resources.

 Copyright © Huawei Technologies Co., Ltd.

- Function detection detects the internal modules of a product system to determine whether the module functions meet the design requirements. If a function of a product system is faulty, the function is not working as expected. Developers and testers must repeatedly verify module functionality before launching a product. Function detection can be performed using traditional log tracing and call chain technologies, such as Huawei Cloud APM.
- Service detection simulates user operation process to obtain the performance and operation result data of the process. Service detection is implemented using dialing test technologies. Dialing tests occupy network resources. Therefore, long-period dialing tests are usually performed during off-peak hours, which are sampling tests. Short-period dialing tests (for example, 5-minute dialing tests) can be performed regularly. Call chains can also be used for service detection.

There are many fault detection methods based on the fault type. The following describes some common fault detection methods for high availability systems.

- Value range check: In most applications, the result of an operation must be within a certain range. Such a range helps you verify whether the data meets the expected requirements.
- Data integrity check: Data can be corrupted during transfer, especially between hardware units. However, because the software layer can hide the differences between local memory transfers and transfers across remote links, data integrity checks must occur at multiple points. There are many ways to verify data integrity, most of which rely on redundancy or summary information contained in the data. Some methods use enough redundancy to detect errors and correct them. However, most methods only include enough extra information to detect whether the data is valid. Typical methods include parity check and cyclic redundancy check (CRC).
- Comparison test: If there is a redundant system, two systems run calculation in parallel and then their results are compared. If the results differ, there is a fault. This concept is also called voting. Comparisons can be made at any level of the system, including cycle-by-cycle comparisons on a memory bus and comparisons with results transferred over a network.
- Time detection is a basic fault detection method that identifies faults when expected events do not occur within a specified timeframe. A specialized approach, known as the heartbeat method, uses periodic message handshakes to verify whether a unit or subsystem maintains functional status.

2.6.1.4 RES06-03 Subhealth Detection

Components in a system can be fully faulty or in a subhealth state. Subhealth indicates that the overall service of a system remains within the threshold, but the service of some instances exceeds the threshold. Subhealth is a relative concept, which compares current performance with past data or overall system performance. Therefore, the detection and determination of subhealth vary. If subhealth is detected, a system needs to be isolated or recovered promptly to prevent service disruptions.

- **Risk level**

  High

- **Key strategies**

  Subhealth detection predicts system faults based on subhealth symptoms. A typical example is memory leakage. Memory leakage does not immediately cause system failures. The system becomes slow due to insufficient swap memory, and the memory usage keeps increasing. Therefore, monitoring the memory usage of instances is necessary. If the memory usage exceeds the threshold, an alarm is generated, and manual intervention is required to quickly rectify the fault, preventing service interruptions.

  Typical subhealth scenarios include packet loss or errors, hard disk performance deterioration, and CPU or memory overload. If a component in an application system is in subhealth state, the service success rate of the application system may decrease.

  Subhealth is not a fault. Therefore, thresholds are set for service monitoring metrics. When a metric exceeds the threshold, an alarm is generated and recovery is required.

2.6.2 RES07 Monitoring Alarms

2.6.2.1 Overview

An application system needs to be monitored so that maintenance personnel can quickly identify the system status and problems.

2.6.2.2 RES07-01 Defining Key Metrics and Thresholds and Monitoring Such Metrics

Before monitoring resources, you need to define key metrics and thresholds to quickly and effectively detect service performance and system status. In this way, you can intervene and rectify faults as soon as possible or locate and fix system defects.

- **Risk level**

  Medium

- **Key strategies**
  - Ensure key metrics are related to key performance indicators (KPIs) of workloads in a system and can be used to identity early warning signals of system performance deterioration. For example, the number of API requests processed by the system and the request success rate can indicate system performance issues more truly than basic metrics such as CPU usage and memory usage.
  - Application systems should be monitored at least from business status, service status, and resource status to ensure availability, effectiveness, and simplicity. Based on your business scale, you can choose Cloud Eye to monitor IaaS services or Application Operations Management (AOM) or Application Performance Management (APM) to monitor PaaS services. Alternatively, you can use Prometheus, Zabbix, and Zipkin to develop your own monitoring system, or use Grafana to visualize monitoring data and align time series.

 Copyright © Huawei Technologies Co., Ltd.

1. Business monitoring

The following four golden metrics are summarized based on the experience of monitoring a large number of distributed services and can be used as a reference for business monitoring:

- Latency: Both successful and failed requests have latency. It is vital to differentiate between the latencies of successful and failed requests.
- Traffic: monitors the system service load.
- Error rate: There are explicit failures (such as HTTP 500 errors) and implicit failures (such as HTTP 200 errors). It is vital to differentiate between explicit and implicit failures.
- Saturation: focuses on monitoring bottleneck resources with the most limited capacity in a system.

For Java application systems, Huawei Cloud users can use APM to monitor the latency and error rate based on traces. FunctionGraph and Cloud Storage Engine (CSE) can help monitor the traffic, latency, and error rate. Applications that use API Gateway to expose APIs can use the traffic, latency, and error rate monitoring capabilities provided by API Gateway. If the capabilities of cloud services cannot meet system requirements, you can use their open APIs to develop your own monitoring system or use Zipkin to monitor tracing, latency, and traffic.

2. Service monitoring

Affected by the redundancy configuration of service instances and fault tolerance protection of application systems, normal service metrics do not necessarily indicate that service instances are functioning normally. For example, in a VM cluster with the load balancer configured, the load balancer proactively isolates faulty nodes. As a result, although requests are shared by healthy nodes, the processing capacity of the application system is still diminished. For this reason, cloud services must be monitored.

Cloud service metrics vary with functions and features. As function providers, cloud services require metrics such as the latency, traffic, error rate, and utilization. In addition, key reliability events of service instances, such as dynamic scaling, overload control, fault self-healing, and migration, also indicate service robustness. If there is an exception, manual intervention is required. You can use Cloud Eye or develop your own monitoring service to monitor key events.

Cloud Eye monitors metrics of IaaS services (such as ECS, EVS, OBS, VPC, ELB and AS), RDS databases, and high-availability middleware (such as DCS and DMS), and allows users to report custom metrics. If you set up a monitoring system by yourself, you can also use the Cloud Eye SDKs to obtain metrics of a cloud service.

AOM monitors key metrics of microservice applications and nodes. You can view key metrics of CCE workloads in CSE, and key metrics of FunctionGraph on the FunctionGraph console.

3. Resource monitoring

Resource monitoring is used to identify resource bottlenecks and analyze system performance issues. Before monitoring application system resources, you need to define key metrics and thresholds of resources to quickly and effectively detect service performance and system status. In this way, you can intervene and rectify faults as soon as possible or locate and fix system defects.

 Copyright © Huawei Technologies Co., Ltd.

The Utilization Saturation and Errors (USE) Method is used to monitor resources by:

- Utilization: covers system resources, including but not limited to CPUs, memory, networks, and disks.
- Saturation: indicates the resource saturation, such as the CPU queue length. This saturation must be distinguished from the golden metric of service monitoring.
- Errors: resource processing errors, such as the network packet loss rate.

Cloud Eye provides fine-grained monitoring for VMs. Utilization and errors are also involved in other services. If the existing capabilities of cloud services cannot meet system requirements, you can create custom monitoring metrics using Cloud Eye or AOM. If you set up a monitoring system by yourself, the monitoring system must cover host resources, network devices, and third-party components such as Apache, Java, and MySQL. Open-source Zabbix is a common choice.

- **Related cloud services and tools**
  - Cloud Eye
  - Application Operations Management (AOM)
  - Application Performance Management (APM)

2.6.2.3 RES07-02 Monitoring Logging

You need to collect logs of application systems, analyze logs as needed, and configure alarm rules to trigger alarms. You can analyze how often specific keywords appear in a certain period.

- **Risk level**

  Medium
- **Key strategies**
  - Configure appropriate log keywords and occurrence thresholds to obtain correct monitoring information.
  - Update alarm rules in a timely manner when log information (such as keywords or keyword occurrence frequency) changes.
- **Related cloud services and tools**
  - Log Tank Service (LTS)

2.6.2.4 RES07-03 Sending Notifications When Exceptions Are Detected

If exceptions are detected during application system monitoring, real-time notifications and alarms need to be sent to related personnel and systems for prompt actions.

- **Risk level**

  Medium
- **Key strategies**
  - Use real-time and quick message notifications to ensure that related personnel can receive messages in a timely manner.
  - Message receivers must include O&M personnel so that faults can be rectified in a timely manner.

 Copyright © Huawei Technologies Co., Ltd.

  - O&M personnel need to have backups to avoid individual faults.

  Simple Message Notification (SMN) can send notifications to users by SMS or email. Services such as Cloud Eye, AOM, CTS, APM, and LTS have been interconnected with SMN. If a threshold rule is changed, message receivers will be notified by email or SMS so that they can detect and handle exceptions in a timely manner.
- **Related cloud services and tools**
  - SMN
  - Cloud Operations Center (COC): supports personnel management, shift management, and notification management. Notifications can be automatically sent to related personnel based on notification rules.

2.6.2.5 RES07-04 Storing and Analyzing Monitoring Data

Monitoring data (including statistics and logs) needs to be stored and managed throughout their lifecycle to meet monitoring requirements. You need to analyze monitoring data periodically to learn system statuses and trends.

- **Risk level**

  Medium
- **Key strategies**
  - Ensure the storage duration of monitoring data meet retention requirements.
  - Analyze monitoring data periodically to detect or predict system faults and reduce service interruptions.
- **Related cloud services and tools**
  - LTS: supports log analysis and transfer.

2.6.2.6 RES07-05 Tracking Requests End-to-End

End-to-end tracing of requests helps analyze and debug issues and improve performance.

- **Risk level**

  Low
- **Key strategies**
  - Trace requests as they process through service components to ensure complete traces for locating and analyzing faults.
- **Related cloud services and tools**
  - APM supports tracing. It comprehensively monitors traces and visualizes service executions and statuses, helping you quickly locate performance bottlenecks and faults.
    - In the displayed trace list, click the target trace to view its basic information.
    - On the trace details page, you can view the trace details, including the local method stack and remote call relationships.

 Copyright © Huawei Technologies Co., Ltd.

- Traces are associated with logs to improve user experience. You can directly switch from the tracing page to LTS to view logs.

2.7 Rapid Fault Recovery

In an application system, if Huawei cloud services use the high availability design, the cloud services can automatically detect and recover from faults after instances become faulty. However, for its own faults, the application system also needs to detect and quickly recover from faults to improve its reliability and stability.

2.7.1 RES08 Dependency Reduction and Degradation

2.7.1.1 Overview

Dependencies of application systems need to be identified and managed. Application system designers need to maintain a complete list of dependencies on other system components, including all dependencies inside and outside the system.

Application systems should have as few critical dependencies as possible, which can minimize the service interruptions caused by the unavailability of the dependencies.

2.7.1.2 RES08-01 Reducing Strong Dependencies

If system components strongly depend on each other, a fault in one component will directly affect other components, affecting system availability.

- **Risk level**

  Medium

- **Key strategies**

  Reduce dependencies using the following methods:

  - Select a higher redundancy level for critical dependencies to increase the availability of critical components.
  - Use asynchronous communication and retries upon timeouts for dependencies, or use the publish–subscribe pattern to separate requests from responses, so that dependencies can recover from short-term failures.
  - Ensure applications can continue to function when its dependencies cannot be accessed for a long time. This minimizes the impacts of partial faults on the overall system functions. For example, if the data that an application depends on is lost, the application can still run and maintain a predictable and recoverable state although the data is slightly outdated, alternative data is used, or even there is no data.
  - Avoid startup dependencies and circular dependencies. When an application system that depends on dependencies to start or load key configuration data is restarted due to certain reasons, it may be in the startup state for a long time and cannot respond to external requests. In this case, the application system should start with the default

configuration data, and then check the statuses of dependencies, or load the latest configuration data to recover itself.

2.7.1.3 RES08-02 Using Loose Coupling to Reduce Dependencies

If internal components of a system directly access each other, they become tightly coupled. The status change of one component directly affects other components, decreasing the availability of all components. A loosely coupled architecture minimizes dependencies between components. Each component can be changed or extended on its own without impacting others. This makes the system more adaptable, simpler to maintain, easier to upgrade, and highly dependable.

- **Risk level**

  Medium

- **Key strategies**
  - Use message queues, caches, and load balancers for component communication. These techniques make components loosely coupled and reduce the impacts of one component on others.
- **Related cloud services and tools**
  - Elastic Load Balance (ELB)
  - Distributed Cache Service (DCS)
  - Distributed Message Service (DMS) for Kafka
  - Distributed Message Service (DMS) for RabbitMQ
  - Distributed Message Service (DMS) for RocketMQ
  - EventGrid (EG)

2.7.1.4 RES08-03 Minimizing the Impacts of Dependency Failures

The availability of component dependencies must be enhanced to minimize the impacts of dependency failures on the components.

- **Risk level**

  Medium

- **Key strategies**

  To minimize the impacts on components when dependencies become faulty or are running slowly, consider the following technologies and principles:
  - Reduce the external dependencies of component dependencies.
  - Optimize performance and reduce the message response delay and load.
  - Use a priority queue to preferentially process requests from high-priority users. This ensures that the core functions of the application system are not affected when there is overflow traffic.
  - Ensure functions can be gradually degraded when there is overflow traffic.
  - Ensure there is default processing configuration if the functions of component dependencies are damaged, so that the application system can still run normally. The default processing configuration may be different from the actual configuration. In this case, ensure alarms can be generated to notify the system administrator.

 Copyright © Huawei Technologies Co., Ltd.

2.7.2 RES09 Retries After Failures

2.7.2.1 Overview

Although the cloud has certain high availability and automatic fault recovery capabilities, it still may cause short-time faults. An application system needs to handle such short-time faults, mainly by using the retry mechanism.

The following are typical scenarios where retries are required:

1. Connections may be interrupted during switchover of primary/standby instances, for example, DCS or RDS instances. In such a case, clients need to retry to restore connections.
2. If an instance is restarted due to a fault, the communication may be interrupted. For example, if the physical server where an ECS is located is faulty due to hardware reasons, the ECS is restarted or automatically restored on another physical server. During the restoration, the communication with the ECS is interrupted and a retry is required.
3. If an instance is overloaded, it cannot respond in a timely manner and a retry is required.

2.7.2.2 RES09-01 Designing a Retry Mechanism for API Calls and Command Executions

During retries, the service party repeatedly calls the APIs and executes commands. Repeated operations must have consistent results.

- **Risk level**

  High

- **Key strategies**

  In application system design, ensure that operations are idempotent. Idempotence means that performing an operation once or performing it multiple times has the same result. This ensures an operation is safe to retry. If idempotent operations are not supported, it is difficult for the client to retry an operation or the retry is more complex.

2.7.2.3 RES09-02 Determining Whether to Retry from the Client Based on the Comprehensive Evaluation Results

When the requests from a client time out or the client receives error responses, the client needs to determine whether to retry the requests. Retries help the client obtain desired results when requests fail but consume more of a server's time to get successful responses.

- **Risk level**

  High

- **Key strategies**
  - Retry requests if they time out when messages are lost due to intermittent disconnections or other temporary faults.
  - Retry requests based on error codes. For temporary faults, for example, if an error code indicates that the system is busy, wait for a period of time

 Copyright © Huawei Technologies Co., Ltd.

and then retry the request. For non-temporary faults, you do not need to retry requests.

- If requests are made through the SDK with the built-in message retry function, the client does not need to retry the requests.
- In a multi-layer service stack, ensure only the original requester retries the requests to avoid layer-by-layer retry.

2.7.2.4 RES09-03 Avoiding Creating Too Much Traffic Pressure from Excessive Retries

Retries from the client work well for temporary faults caused by intermittent disconnections. However, for faults caused by a sudden traffic spike, a retry may worsen the situation, which needs to be avoided.

- **Risk level**

  High

- **Key strategies**

  When the client retries requests, you are advised to:

  - Add exponential backoff with jitter to avoid traffic pressure on the server. Exponential backoff increases the delay between retries exponentially. Jitter introduces randomness to the delay interval between two retries and avoids retry peaks at the same time.
  - Limit the maximum number of retries or retry interval to prevent traffic spikes caused by message backlog.

2.7.3 RES10 Fault Isolation

2.7.3.1 Overview

If no measures are taken when a unit of a system is faulty, the fault may spread on a large scale, causing the entire system to fail. Fault isolation aims to limit faults to specific set of components, minimizing fault impact and preventing cascading failures.

Effective fault isolation requires dividing domains to minimize workload impacts.

2.7.3.2 RES10-01 Isolating the Control Plane from the Data Plane

The data plane processes services and has high availability requirements. The control plane does not directly process services, so its failure should not affect the service system.

- **Risk level**

  High

- **Key strategies**

  - If the control plane becomes faulty, isolate the control plane from the data plane to prevent the fault from affecting services.
  - Ensure the data plane can be recovered independently of the control plane. This prevents service system recovery failures caused by control plane faults.

 Copyright © Huawei Technologies Co., Ltd.

2.7.3.3 RES10-02 Deploying Application Systems in Multiple Locations

An application system can be deployed at multiple locations to prevent system unavailability due to infrastructure faults at one location.

- **Risk level**

  High

- **Key strategies**
  - Deploy the data and resources of an application system in multiple AZs to prevent service interruptions caused by faults in a single AZ.
  - Deploy an application system that has high availability requirements in multiple regions to prevent service interruptions caused by faults in a single region.
  - If a multi-AZ architecture can meet application availability requirements, multi-region deployment is not required.

2.7.3.4 RES10-03 Adopting a Grid Architecture

Adopting a grid architecture can minimize the impacts of workload faults.

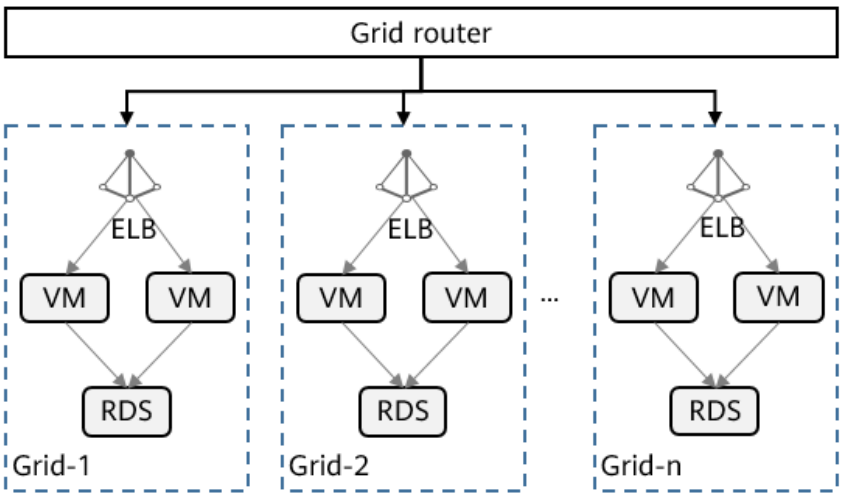
- **Risk level**

  High

- **Key strategies**

  In an application system, use multiple grids with same functions. Each grid has complete service functions and processes a replica of the workload. The grids do not interact with each other. If a grid becomes faulty, the services processed by that grid are affected, but other grids are not affected, reducing the blast radius.

  The following figure shows a typical grid architecture.



  Implementation procedure:

  a. Determine partition keys based on the following principles:

     - Partition keys must match service granularity or minimize cross-grid interactions. For a multi-user system, use user IDs as partition keys.

 Copyright © Huawei Technologies Co., Ltd.

For a system in which resources are objects, use resource IDs as partition keys.

- Partition keys must be directly included in all APIs or commands, or can be converted through other parameters.
- Ensure each grid independently process services, avoiding or reducing grid interactions.

b. Determine the number of grids and the size of each grid. Mode grids can be added later if the number of grids is insufficient. There are some tips:

- More grids mean smaller grid size. This limits blast radius, simplifies fault locating, and improves service availability. However, more grids lower resource utilization and increase costs.
- Fewer grids mean more resources in each grid. This simplifies O&M, improves resource usage, and reduces costs. Fewer grids are right for large customers with high resource demands.

c. Determine the grid mapping algorithm. The following are some options:

- Naive modulo mapping: Partition keys are used to modulo the number of grids. This algorithm ensures even distribution of data, but is not suitable when grids are added and deleted. Once a grid is added or deleted, service migration is required.
- Range-Hash/Hash mapping: Partition keys are used to partition data by range and then hash the data, or partition keys are directly used to hash data. Metadata management is complex.
- Full mapping: Every partition key is mapped to a grid. This comes with a severe read and write dependency on the mapping table and a read-your-writes consistency requirement. Generally, the metadata service needs to be introduced.
- Prefix range-based mapping: Partition keys are mapped to grids by prefix range. This algorithm offsets the downsides of the full mapping algorithm while providing flexibility.
- Mapping replacement: A specific key is forcibly allocated to a specific grid, which facilitates testing and isolation.

d. Design the grid routing layer as follows:

- The grid routing layer is the only shared component, so you must minimize changes to it for stability.
- Minimize changes to the service logic to maintain stability.
- Due to the large blast radius, the grid routing layer needs to be lightweight and simple enough, but have full functionality.
- In some cases, avoid routing all calls to help reduce the latency and the size of the grid routing layer.
- Allow horizontal expansion to prevent the grid routing layer from becoming a performance bottleneck.

e. Provide the grid migration function for quick grid reassignment based on keys when a grid is added or deleted. Details are as follows:

   - Copy data from the old grid mapped by the key to the new grid.
   - Update the grid routing layer route to map the partition key to the new grid.
   - Delete data from the old grid.

f. Deploy and update grid code as follows:

   - Combine the grid code deployment with cross-AZ and cross-region deployment to reduce the impact scope of faults through multi-layer isolation.
   - Use canary deployments to update the grid service unit code to reduce the possibility that multiple grid service units are faulty at the same time due to version problems.

2.7.3.5 RES10-04 Configuring Health Check and Automatic Isolation

If an unhealthy component is detected, it is automatically isolated to minimize the impact scope.

- **Risk level**

  High

- **Key strategies**
  - Perform health check periodically on system components to check whether they are normal.
  - Ensure abnormal components can be automatically isolated to prevent impacts on services.
- **Related cloud services and tools**
  - Elastic Load Balance (ELB): supports health check. ELB periodically sends requests to backend servers to detect their statuses and check whether they can process requests. If an unhealthy backend server is detected, the load balancer will stop routing requests to it.
  - Cloud Container Engine (CCE): supports container health check. When containers are running, CCE periodically checks the container health based on user requirements. If health check is not configured, application exceptions cannot be detected, and the pod cannot automatically restart to recover the application. As a result, the pod is normal, but the application is abnormal.

2.7.4 RES11 Reliability Testing

2.7.4.1 Overview

Reliability tests are performed to ensure that a system can meet the expected reliability objectives within the specified lifecycle. Unlike function tests, reliability tests involve service loads and faults, which poses higher requirements on the test environment and capabilities.

By proactively introducing faults, reliability tests and drills detect system risks in advance, and enhance test quality, risk contingency plans, monitoring and alarms, and fault emergency efficiency. In this way, faults can be effectively prevented, handled in a timely manner, and regression verification can be performed after faults are rectified. Distributed system resilience is built based on faults to continuously improve software quality, enhance the confidence of the team in software production and operation, and reduce similar problems during service running.

To ensure test effectiveness, the test environment must be the same as the production environment.

Huawei Cloud provides fault injection service (MAS-CAST), performance test service (CodeArts PerfTest), and multi-site high availability service (MAS), which can be used for fault injection tests, pressure load tests, long-term stability tests, and disaster drills.

2.7.4.2 RES11-01 Chaos Testing

Chaos engineering is the practice of injecting faults into a system to test its stability and fault tolerance.

- **Risk level**

  High

- **Key strategies**
  - Perform chaos testing in a real environment.
  - Perform chaos testing in a CI/CD pipeline.
  - Proactively inject faults to detect and rectify faults before they occur in the real world.
  - Inject faults in a controllable manner to reduce the impacts on customers.

  Chaos engineering metrics:
  - Fault scenario coverage: The coverage of fault scenarios is analyzed, such as 80% of redundancy scenarios and 60% of overload scenarios.
  - Hit rate of a fault scenario: This is the actual occurrence rate of the fault scenario.
  - Contingency plan quality: assesses both the effectiveness and execution efficiency of contingency plans.
  - Number and severity of identified risks: regularly, typically quarterly or annually, quantify and assess the severity of risks.
  - Number, severity, and type of mitigated risks: the number of downgraded risks, the number of mitigated risks, the number of added contingency plans, and the number of improved monitoring items
  - Improvement rate of fault recovery duration: reflects how quickly the average recovery time improves as a result of chaos engineering drills.
  - YoY fault decrease: the decrease in the number of faults in the current year compared to the previous year

- **Related cloud services and tools**

 Copyright © Huawei Technologies Co., Ltd.

  - MAS-CAST: provides test tools and injection methods for cloud applications. It supports reliability testing, stress testing, random fault injection in chaos engineering, and fault drills in the production environment for fault and service process orchestration.
  - COC: allows you to conduct automatic chaos drills covering from risk identification, emergency plan management, fault injection, and review and improvement.

2.7.4.3 RES11-02 Stress Testing

Stress testing increases the service load beyond the system's capacity to check the overload protection, service isolation, and graceful degradation capabilities of cloud services. To test the planned overall capacity and service dependencies of a system, cloud service applications are usually tested through full-link stress testing.

- **Risk level**

  High

- **Key strategies**
  - Construct a large number of interface messages to perform stress testing.
  - Simulate various service scenarios to perform stress testing.
  - Ensure continuous automatic testing.
  - Ensure alarms are automatically generated for performance deviations, so that the deviations can be located and handled in a timely manner.
- **Related cloud services and tools**
  - CodeArts PerfTest: provides performance testing for cloud applications developed based on protocols such as HTTP, HTTPS, TCP, UDP, HLS, RTMP, WebSocket, and HTTP-FLV. It can quickly simulate service peaks with a large number of concurrent users. It allows users to define the content and time series of packets and supports flexible combinations of multiple transactions to test and check service performance during peak hours.

2.7.4.4 RES11-03 Long-Term Stability Testing

In long-term stability testing, build service models based on user application scenarios, use the automatic tool to construct service volumes that cover 70% of system capacity specifications, and perform 24/7 load testing to evaluate system stability.

- **Risk level**

  High

- **Key strategies**
  - Simulate various service scenarios for testing.
  - Ensure continuous automatic testing.
  - Ensure alarms are automatically generated for deviations, so that the deviations can be located and handled in a timely manner.

 Copyright © Huawei Technologies Co., Ltd.

2.7.4.5 RES11-04 DR Drills

DR drills can check whether the DR system is available and whether the data loss time and recovery time meet the RPO and RTO requirements.

DR drills test cross-AZ or cross-region failover capabilities of services and verify the system's DR and response to disasters. DR drills involve multiple teams and are usually carried out as a dedicated project. DR drills help enterprises better verify RPO and RTO, detect and resolve issues in a timely manner, and improve system availability and reliability.

- **Risk level**

  High

- **Key strategies**
  - Simulate an AZ or region fault scenario and perform a recovery drill based on the DR switchover manual.
  - After the drill is complete, check whether the RPO and RTO requirements are met. If the requirements are not met, adjust the DR solution and perform the drill again.
- **Related cloud services and tools**
  - Multi Site High Availability Service (MAS): supports service DR management, workflow orchestration, and drill switchover in intra-city cross-AZ, intra-city dual-active, geo-redundant, and remote multi-active DR scenarios.

2.7.4.6 RES11-05 Red/Blue Attack/Defense

Red/Blue attack/defense can simulate various complex attack scenarios to help comprehensively evaluate application resilience and detect and resolve potential risks in a timely manner.

- **Risk level**

  High

- **Key strategies**
  - The blue team discovers various vulnerabilities from the third-party perspective and injects faults into the software and hardware that services depend on to continuously verify the reliability of service systems. The red team handles faults based on the predefined troubleshooting and emergency response process.
  - After attack/defense drills, review the duration and operations in the fault discovery, response, and recovery phases, and sort out the required improvements and implement them to enhance system stability.

2.7.5 RES12 Emergency Recovery

2.7.5.1 Overview

No matter how carefully an application system is designed, unrecoverable faults may still occur. If there is such a fault, emergency recovery is required.

 Copyright © Huawei Technologies Co., Ltd.

2.7.5.2 RES12-01 Setting Up an Emergency Recovery Team

To rectify emergency faults, set up an emergency recovery team, clarify each member's role, and provide training.

- **Risk level**

  High

- **Key strategies**
  - **Set up an emergency recovery team** with a chairperson and O&M owners for all components and key dependencies.
    - The chairperson organizes the emergency recovery team to quickly rectify faults.
    - The O&M owners of components or key dependencies locate and rectify faults.
  - **Formulate an emergency recovery solution**. All emergency recovery team members must receive emergency recovery training and get familiar with emergency recovery processes and methods.

2.7.5.3 RES12-02 Developing an Emergency Response Plan

Provide standard emergency recovery guides to resolve common issues.

- **Risk level**

  High

- **Key strategies**
  - Ensure the emergency response plan covers common scenarios.
  - Provide standard operation processes and operations for emergency recovery, so that related personnel can clearly understand their responsibilities and take steps when there are events.
  - Ensure each recovery operation is clear and unambiguous and can guide operators.
- **Related cloud services and tools**
  - COC: supports emergency response plan management.

2.7.5.4 RES12-03 Conducting Emergency Recovery Drills Periodically

Periodically test the emergency recovery process to ensure quick recovery from faults.

- **Risk level**

  High

- **Key strategies**
  - Conduct at least one emergency recovery drill every year to improve the proficiency of operators.
  - Strictly follow the emergency response plans during emergency recovery drills to check their accuracy.
  - After drills are complete, backtrack the emergency recovery process and optimize the emergency response plans.

 Copyright © Huawei Technologies Co., Ltd.

- **Related cloud services and tools**
  - COC: allows you to conduct automatic chaos drills covering from risk identification, emergency plan management, fault injection, and review and improvement.

2.7.5.5 RES12-04 Recovering from Faults Immediately

If an application system becomes faulty, locate and rectify faults as soon as possible.

- **Risk level**

  High

- **Key strategies**

  Use the following methods to quickly detect faults:

  - Monitoring: An application system must offer service monitoring data so that the maintenance team can track its status. Specially-assigned maintenance personnel should monitor the system and rectify faults quickly.
  - Alarms: After detecting a fault, an application system must generate an alarm and send it to all related personnel by SMS message or email, so that the personnel can quickly respond to the fault.
  - Prediction: The maintenance team needs to predict system risks based on the system status using methods such as data analytics and machine learning, and prevent and handle the risks in advance.

  During emergency recovery, the maintenance team needs to mitigate risks or recover services as soon as possible to quickly eliminate the impacts of service interruptions on customers, and then start locating and rectifying faults to shorten service interruptions.

  - Organization coordination: If there is a fault, the emergency recovery chairperson needs to quickly organize related personnel to recover services.
  - Emergency recovery: If there is a fault, the assigned O&M owner needs to quickly analyze the fault and recover the system based on the emergency response plan.

2.7.5.6 RES12-05 Organizing Emergency Recovery Backtracking

After emergency recovery, backtrack and optimize the emergency recovery process to prevent similar events from recurring.

- **Risk level**

  High

- **Key strategies**
  - Locate and rectify faults, optimize product capabilities, and reduce the occurrence of similar events.
  - Summarize and optimize the emergency recovery process.

 Copyright © Huawei Technologies Co., Ltd.

2.8 Overload Control

The component resources in the system are limited. When traffic bursts occur, the resources may be exhausted, which can impact services.

2.8.1 RES13 Overload Protection

2.8.1.1 Overview

If the system traffic exceeds a certain threshold, the system is overloaded, and some requests may fail. In this case, the failure triggers service retries, and the system load will increase, causing a vicious cycle. This may cause the access success rate to be far lower than the designed capacity, and even the entire system may become unavailable. Therefore, an overload protection mechanism must be designed for applications to ensure that a certain proportion of the designed capacity can be used normally in the overload state.

Overload protection helps prevent resource exhaustion caused by traffic bursts, flood attacks, or retry storms and ensure that the agreed number of requests can be processed normally in those cases.

2.8.1.2 RES13-01 Automatic Elastic Scaling

When the system traffic bursts, automatic elastic scale-out can be performed to reduce the risk of service interruptions.

- **Risk level**

  High

- **Key strategies**

  Elastic scaling requires technologies such as separation of service processing logic from data and state externalization to support rapid increase or decrease of system processing capabilities.

  There are two ways to scale a system. One is to change the processing capability of a single node, involving its CPU, memory, and storage, which is called vertical scaling. The other is to change the processing capability of the system by changing the number of nodes without changing the processing capability of a single node, which is called horizontal scaling.

  Horizontal scaling is recommended during system design. When horizontal scaling is used, services and data are decoupled. That is, the service processing logic must be separated from data, and data (states) is externalized. In this way, service nodes (including resources) are stateless and can be quickly added or removed as required, thereby scaling the system processing capabilities.

  The system can automatically detect node faults or resource insufficiency and then add nodes to implement horizonal scaling, increasing the system capacity to meet processing needs.

  Huawei Cloud provides the Auto Scaling (AS) service to automatically adjust resources, such as Elastic Cloud Server (ECS) instances and bandwidth, based

 Copyright © Huawei Technologies Co., Ltd.

on the load in an AS group and scaling rules. When service demands increase, AS automatically adds ECS instances or bandwidth resources to ensure normal service capabilities. When service demands decrease, AS automatically reduces ECS instances or bandwidth resources to reduce costs.

In addition, Huawei Cloud provides some cloud services with built-in scaling capabilities, which are transparent to users or only require simple configurations.

- Object Storage Service (OBS), Scalable File Service (SFS), and FunctionGraph automatically expand their service processing capabilities based on the number of requests, which is transparent to users.
- Relational Database Service (RDS) supports a maximum of five read replicas. Read replicas can be expanded online. The CPU and memory can be easily scaled up or down. The storage capacity can be expanded online.
- Cloud Container Engine (CCE) supports automatic scaling of nodes and workloads. Policies can be created to configure alarm-based scaling (triggered by the excessive CPU or memory usage), scheduled scaling, or periodic scaling.

- **Related cloud services and tools**
  - AS
  - CCE
  - RDS
  - OBS
  - SFS
  - FunctionGraph

2.8.1.3 RES13-02 Application System Load Balancing

For stateless cluster services, load balancing is used to ensure that traffic is evenly distributed. This prevents services from being affected when some components are idle and some components are overloaded. In addition, system resources can be fully utilized to improve system performance and reliability.

- **Risk level**

  High

- **Key strategies**
  - The granularity of traffic distribution must not be too coarse. Otherwise, some components may be overloaded.
  - Traffic is distributed based on the check results of the load status of each backend node.
  - If any backend node is faulty, traffic needs to be automatically distributed to other healthy nodes to prevent request failures.
- **Related cloud services and tools**
  - Elastic Load Balance (ELB) supports service load balancing, backend server health inspection, and automatic isolation of abnormal ECS instances.

 Copyright © Huawei Technologies Co., Ltd.

2.8.1.4 RES13-03 Overload Detection and Control

When an application system is overloaded, the system may be overwhelmed and cannot provide services effectively. Therefore, overload detection and control are required.

- **Risk level**

  High

- **Key strategies**

  Overload control (also called traffic control) is a process of ensuring that a system, when overloaded, successfully handles all or the agreed number of requests by employing techniques such as rate limiting, degradation, circuit breaking, and elastic scaling. Typical overload control methods are as follows:

  - Rate limiting: The system proactively discards some service requests when it is overloaded.
  - Degradation: When the system is overloaded, lossy services are provided. Measures such as reducing non-critical services and reducing service quality are taken to reduce the system load.
  - Circuit breaking: In a distributed system, if an application fails to invoke a third-party resource or service due to a third-party resource or service fault (including overload), the application stops invoking remote resources or services to prevent the fault from spreading.

- **Related cloud services and tools**

  Huawei Cloud provides some cloud services with built-in traffic control capabilities available for configuration.

  - API Gateway: Request throttling policies can be created to control the number of times that an API can be called by a single user or application within a specified period.
  - Cloud Service Engine: Rate limiting can be configured to specify the maximum number of requests that can be accepted within a specified period.

2.8.1.5 RES13-04 Proactive Capacity Expansion

When resource demands increase due to planned activities, proactive capacity expansion is required to prevent services from being affected due to insufficient resources.

- **Risk level**

  High

- **Key strategies**

  When more resources are required by application system services, resources can be proactively added to meet the demands and avoid affecting availability. In a typical scenario such as product promotions, if traffic bursts are predicted, manual expansion can be performed.

  Huawei Cloud service instances support proactive horizontal and vertical scaling. For example, with ECS, horizontal scaling means adding instances and vertical scaling means upgrading instance specifications. With RDS, vertical scaling means upgrading DB instance specifications.

 Copyright © Huawei Technologies Co., Ltd.

2.8.1.6 RES13-05 Quota Restrictions on Automatic Capacity Expansion

When an application system is automatically expanded to address insufficient resources, if the remaining quota is not enough, the automatic capacity expansion fails.

- **Risk level**

  High

- **Key strategies**

  To manage resource usage, quotas can be set to limit the number and capacity of resources available. For example, quotes can be set to limit the number of ECS instances or EVS disks that can be created. For dynamic resources, their quotas must be considered to avoid access being affected when the quota is used up. The maximum capacity configured for automatic expansion must not exceed the corresponding resource quota.

  Alarm policies can be configured to trigger notifications when any resource exceeds a specified usage level, avoiding its quota being used up.

- **Related cloud services and tools**

  A quota page on Huawei Cloud displays the total and used quotas of resources of different types in each service. An application can be submitted to increase the quota of a specified resource of a cloud service. And alarm policies can be configured to trigger notifications whenever a resource exceeds a specified usage level, so a quota application can be submitted in a timely manner if needed.

  When AS is involved in the application system, the possibility that the AS configuration is restricted by quotas must be considered. For example, the maximum number of instances specified in AS may fail to be created because of the insufficient quota left for resources in, for example, ECS, EVS, or Elastic IP, affecting service access.

2.8.1.7 RES13-06 Load Testing

Load testing can be used to measure whether the elastic expansion capability of the system meets service requirements.

- **Risk level**

  High

- **Key strategies**

  For details, see **RES11-02**.

2.9 Change Error Prevention

When a system is running, configuration changes such as modifications, manual scaling of workloads, and patch installations may lead to downtime in the production system. If a change fails, it can result in serious problems, like performance deterioration or service interruptions. To minimize service risks associated with these changes, changes to your workload or its environment must be anticipated and accommodated to achieve reliable operation of the workload.

Change operations are a part of O&M. For details, see "Operational Excellence Pillar" > "O&M Preparation and Change Management".

 Copyright © Huawei Technologies Co., Ltd.

2.9.1 RES14 Configuring Error Prevention

2.9.1.1 Overview

If the system and services are damaged or become invalid due to incorrect data manually configured, the product design can be used to reduce or avoid the impact caused by incorrect configuration.

2.9.1.2 RES14-01 Fool-proofing Check for Changes

Fool-proofing is when you put systems in place to prevent equipment operators from making mistakes. The goal is to eliminate product defects by preventing, correcting, or drawing attention to human errors as they occur.

- **Risk level**

  High

- **Key strategies**

  The following constraints and checks can help reduce configuration errors:

  - Role constraints: Permissions management limits the configuration scope for different roles, preventing errors due to unauthorized changes.
  - Separation of query and modification: Configuration options are organized into distinct layers via the product interface, with query and modification functions kept separate to minimize the risk of accidental configuration changes.
  - Configuration validation: A validation mechanism ensures that necessary checks are performed before applying any configuration changes, preventing invalid settings from being applied.
  - Deletion protection: Protective measures are implemented to prevent accidental deletion of resources. Examples include pre-deletion status checks, resource locking, and a recycle bin.

2.9.1.3 RES14-02 Automatic Changes

Automatic changes involve an automation module that provides and manages the environment (including compute, storage, network, and middleware services) as well as the installation and configuration of applications, implementing infrastructure as code. This approach addresses issues such as manual errors when deploying services, reliance on individual skills, and the lack of traceability and difficulty in rolling back manual configuration changes.

- **Risk level**

  High

- **Key strategies**

  - A configuration management tool centralizes the management of configuration information, tracks and records changes, and swiftly identifies their impact scope.
  - An automatic change engine assists organizations in planning and automating change processes, including predefined change templates, approval workflows, and automated testing and verification, thereby minimizing manual errors and delays.

 Copyright © Huawei Technologies Co., Ltd.

  - Change evaluation and risk management assess the impact of changes, identify potential risks and conflicts, and take corresponding measures.
  - Testing and verification are automated to ensure the correctness, performance, and reliability of changes, thereby reducing errors and delays associated with manual testing.
  - The change process is continuously monitored and audited to track and record the change status, promptly identify and resolve issues, and provide transparency and traceability.
- **Related cloud services and tools**
  - Cloud Operations Center (COC)
    - Through job management, you can create, modify, and delete jobs, and execute jobs on VMs. (Job management is currently only supported for ECSs.)
    - The change center provides a unified platform for engineers to manage change tasks. With the change center, engineers can submit tickets to manage change requests, review, and execution.

2.9.1.4 RES14-03 Data Backup Before a Change

The design for configuration data backup and restoration ensures that the configuration data can be quickly restored upon a configuration error.

- **Risk level**

  High
- **Key strategies**
  - Full data is backed up to prevent data damage during a change and ensure service continuity.
  - If a change fails and needs to be rolled back, backups can be used for data restoration.

2.9.1.5 RES14-04 Runbooks for Standardized Changes

Runbooks are operational manuals that detail the steps for change operations.

Standard runbooks must be prepared beforehand to ensure that both the changes and any necessary rollbacks are executed strictly according to the runbooks.

- **Risk level**

  High
- **Key strategies**
  - A runbook must include checks before the change, the change operations, checks after the change, and rollback procedures in case the change fails.

2.9.2 RES15 Hitless Upgrade

2.9.2.1 Overview

During software redeployment or upgrade, service interruptions must be minimized to reduce the impact on services.

2.9.2.2 RES15-01 Automatic Deployment and Upgrade

The deployment and upgrade processes are automated using code to solidify the dependencies between components and the installation and configuration procedures, thereby reducing the room for manual errors.

- **Risk level**

  High
- **Key strategies**
  - The deployment and upgrade processes are fully automated.

2.9.2.3 RES15-02 Automatic Checks

Basic test functions are integrated into the deployment and upgrade processes to enable automatic checks. There is no manual intervention required.

- **Risk level**

  High
- **Key strategies**
  - Basic test functions are integrated during deployment or upgrade. Once completed, the system automatically checks and tests the newly deployed code.
  - Fault injection functions are also integrated during deployment or upgrade. Upon completion, the system automatically injects faults to test the resilience of the newly deployed code.

2.9.2.4 RES15-03 Automatic Rollback

If an exception occurs during an upgrade or deployment, or if any check or test fails, the system supports automatic rollback to minimize the need for manual intervention and prevent rollback failures.

- **Risk level**

  High
- **Key strategies**
  - If an exception is detected, a one-click rollback can be performed.
  - The rollback process is fully automated.

2.9.2.5 RES15-04 Gray Deployment and Upgrade

During in-place upgrade and rollback, services may be interrupted. The interruption depends on the upgrade and rollback duration, which can greatly affect services. Gray deployment and upgrade can help minimize service downtime and enhance overall system availability.

- **Risk level**

 Copyright © Huawei Technologies Co., Ltd.

High

- **Key strategies**

  Canary and blue-green deployments are supported to facilitate gray upgrade or deployment. These methods gradually introduce new changes or switch user workloads. They work in conjunction with automatic rollback to minimize service interruption risks caused by deployment errors.

  A canary deployment (gray release) is a method that tests out a new feature to an early sub-segment of users deploying the feature globally. It usually runs on a single service instance (the "canary"). If any changes or errors are detected, the workloads in the canary can be removed and users can be rolled back to the old version. If everything works as intended, you can gradually add more users while monitoring logs, errors, and software health.

  A blue-green deployment is similar to a canary deployment. You create two production instances. At any given time, one application responds to user traffic and the other application continuously receives updates from the team's continuous integration (CI) server. When a new version of the environment (the green version) is deployed, all traffic from the user base continues going to the production environment (the blue version). The green version is tested first, and when completed, users are switched over to the green environment. The blue environment serves as a backup, enabling easy rollback in case of any problems with the green version.

- **Related cloud services and tools**
  - CodeArts Deploy: a visualized and automatic deployment service. It provides various ways to customize the deployment process, improving efficiency and reducing costs.
  - Cloud Service Engine (CSE): support for gray release.
  - Application Service Mesh (ASM): support for gray release.

2.10 Reference Architecture

2.10.1 Overview

This section uses typical web applications as an example to describe the typical architectures deployed with different availability objectives. In each scenario, design an architecture with the items listed in the table below taken into account, so as to achieve the availability objective.

Item	Impact on Application Availability
Redundancy	High availability of application components and automatic service recovery upon faults of some application nodes
Backup	Recovery capability when application data is damaged
DR	Recovery capability in the event of a disaster in a region, AZ, IDC, or other cloud sites

 Copyright © Huawei Technologies Co., Ltd.

Item	Impact on Application Availability
Monitoring metrics and alarms	Capabilities of detection and alarm reporting for application system faults
Auto scaling	Automatic recovery capability when the application capacity is insufficient
Change error prevention	Capability of reducing the downtime caused by changes in applications
Emergency recovery	Emergency recovery capability upon application faults

2.10.2 Typical Deployment Architecture for Internal Tools or OBT Applications (99% Availability)

Internal tools are proprietary and used internally. If these tools fail, they may cause disruptions in related operations and bring inconvenience to operators, but their RTO and RPO can be long. OBT applications handle experimental customer tasks and can be disabled if needed. These tools and applications do not require high availability—a 99% uptime standard (up to 3.65 days of downtime per year) is sufficient.

The service interruption time includes the interruption time due to faults and the interruption time caused by upgrades, configuration, and maintenance. Assume that the interruption time is as follows:

- Fault-caused interruptions: Assume that there are four fault-caused interruptions each year. It takes 1 hour to decide on the emergency recovery for each interruption and 2 hours to re-deploy and configure the application workload and recover the related data. Thus, the total yearly interruption time is 12 hours.
- Change-caused interruptions: Assume that an application is updated offline for six times every year, and each update takes 4 hours. Then the yearly update time is 24 hours.

As estimated above, the application system will be unavailable for 36 hours each year, which meets the availability design requirements.

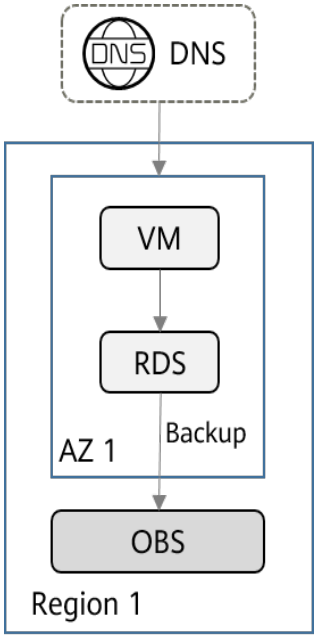
An internal tool usually consists of two main parts: a stateless application layer at the frontend and a database at the backend. The frontend stateless application layer can be composed of ECS or CCE instances (ECS is used as an example here); while the backend typically uses RDS for MySQL, depending on the service type. To meet the availability requirement, the recommended solution is as follows:

Item	Solution
Redundancy	Deploy ECS and RDS in single-node mode.
Backup	Enable automated backup for RDS databases. When a data fault occurs, the latest backup can be used to restore data, meeting availability requirements.

 Copyright © Huawei Technologies Co., Ltd.

Item	Solution
DR	DR deployment is not supported. If faults occur at a site, applications need to be re-deployed and backup data needs to be restored again.
Monitoring metrics and alarms	Use basic monitoring to check whether each application can return messages normally.
Auto scaling	Provide runbooks for troubleshooting common issues, facilitating manual scaling when required, such as performing a scale-out when the capacity becomes insufficient.
Change error prevention	Update software offline, stop applications during installation and restart, and deploy and roll back applications as instructed by runbooks.
Emergency recovery	Assign an application system owner, who can be contacted for emergency recovery.

The typical deployment architecture is as follows:



This architecture has the following features:

- The application system is deployed in a single AZ of a single region.
- The RDS database data is automatically backed up to OBS periodically to ensure that data can be quickly restored in the case of data loss.

2.10.3 Typical Deployment Architecture for Internal Knowledge Management Applications (99.9% Availability)

Internal knowledge management applications are proprietary and used internally. If these applications fail, they may cause disruptions in related operations and

 Copyright © Huawei Technologies Co., Ltd.

bring inconvenience to operators, but their RTO and RPO can be relatively long. These applications require an availability of 99.9%, that is, up to 8.76 hours of downtime per year.

The service interruption time includes the interruption time due to faults and the interruption time caused by upgrades, configuration, and maintenance. Assume that the interruption time is as follows:

- Fault-caused interruptions: Assume that there are four fault-caused interruptions each year. It takes 30 minutes to decide on the emergency recovery for each interruption and 30 minutes to recover application services. Thus, the total yearly interruption time is 240 minutes.
- Change-caused interruptions: Assume that an application is updated offline for eight times every year, and each update takes 30 minutes. Then the yearly update time is 240 minutes.

As estimated above, the application system will be unavailable for 480 minutes each year, which meets the availability design requirements.

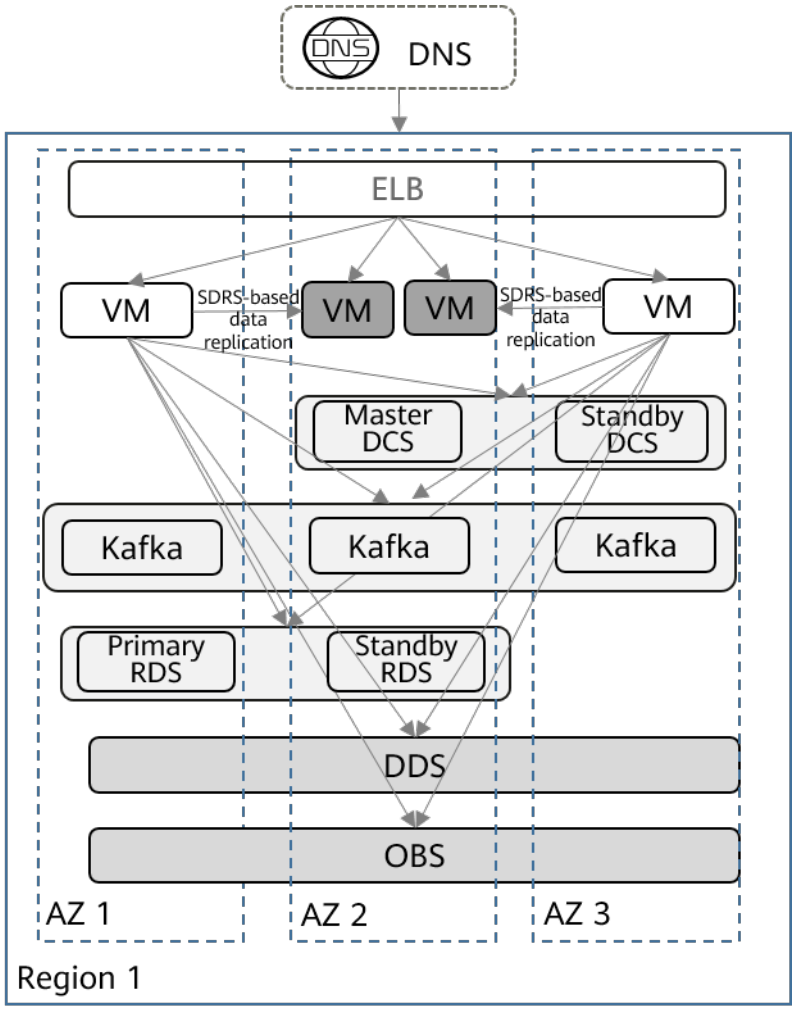
An internal knowledge management application usually consists of two main parts: a stateless application layer at the frontend and a database at the backend. The frontend stateless application layer can be composed of ECS instances; while the backend typically uses RDS for MySQL, depending on the service type. Other common backend options include DDS and some middleware, such as DCS or Kafka. To meet the availability requirement, the recommended solution is as follows:

Item	Solution
Redundancy	Deploy cloud service instances in HA mode, such as ELB, RDS, DCS, Kafka, and DDS instances.
Backup	Configure the RDS and DDS databases to automatically back up data, and configure the stateful ECS instances to automatically back up data through CBR. In case of a fault, the latest backup data can be used to restore services, meeting the availability requirements.
DR	Configure the applications to use services that support cross-AZ deployment. Deploy ELB and RDS across AZs. When a fault occurs in an AZ, services are automatically recovered. Configure the stateful ECS instances to use SDRS for cross-AZ DR. When a fault occurs in an AZ, a manual switchover can be performed.
Monitoring metrics and alarms	Monitor and check site statuses and instance workloads. If a site fails or an ECS or RDS instance is overloaded, an alarm is reported.
Auto scaling	Resources are sufficient for internal operations, making auto scaling unnecessary. For ECS instances, use ELB for fault detection and load balancing, and add or remove the ECS instances on demand based on the monitoring statistics, so as to enhance the application performance. For RDS, change the instance type or add read-only nodes during maintenance based on the workload monitoring statistics.

 Copyright © Huawei Technologies Co., Ltd.

Item	Solution
Change error prevention	Update software offline and replace it without changing its position. In addition, each application needs to be automatically deployed and rolled back as instructed by a runbook. The software is updated every one to two months.
Emergency recovery	Develop an emergency handling mechanism and designate related personnel to quickly make decisions and recover services. Provide solutions to common application and database problems as well as upgrade and deployment failures.

The typical deployment architecture is as follows:



This architecture has the following features:

- The application system is layered with stateful VMs and stateful databases.
- The application system is deployed across AZs in a single region of Huawei Cloud. Stateful VMs are deployed as active/standby pairs, with the active and standby ones in separate AZs and data replicated from the active one to the

 Copyright © Huawei Technologies Co., Ltd.

standby one. In this way, cross-DC application DR can be implemented on the cloud.

  - Access layer (external DNS): The external DNS is used for domain name resolution and traffic load balancing. The failure of a single AZ does not affect services.
  - Application layer (ELBs, application software, and VMs): For stateful applications, SDRS is used for cross-AZ data replication and DR failover between VMs, and CBR is used by applications for automatic data backup.
  - Middleware layer: Redis and Kafka clusters are deployed across AZs in HA mode. The failure of a single AZ does not affect services.
  - Data layer: RDS, DDS, and OBS are deployed across AZs in HA mode. The failure of a single AZ does not affect services.
- The RDS database data is automatically backed up to OBS periodically to ensure that data can be quickly restored in the case of data loss.

2.10.4 Typical Deployment Architecture for Information Management Applications (99.95% Availability)

Information management applications are proprietary and used internally. If these applications fail, they may cause disruptions in related operations and bring inconvenience to operators, but certain RTO and RPO of them are acceptable. These applications require an availability of 99.95%, that is, up to 4.38 hours of downtime per year.

Assume that the interruption time caused by faults and changes is as follows:

- Fault-caused interruptions: Assume that there are four fault-caused interruptions each year. It takes 20 minutes to decide on the emergency recovery for each interruption and 10 minutes to recover application services. Thus, the total yearly interruption time is 120 minutes.
- Change-caused interruptions: Assume that an application allows offline updates and online patching. If an application is updated offline for four times every year, and each update takes 30 minutes, then the yearly update time is 120 minutes. Note that services are not affected during online patching.

As estimated above, the application system will be unavailable for 240 minutes each year, which meets the availability design requirements.

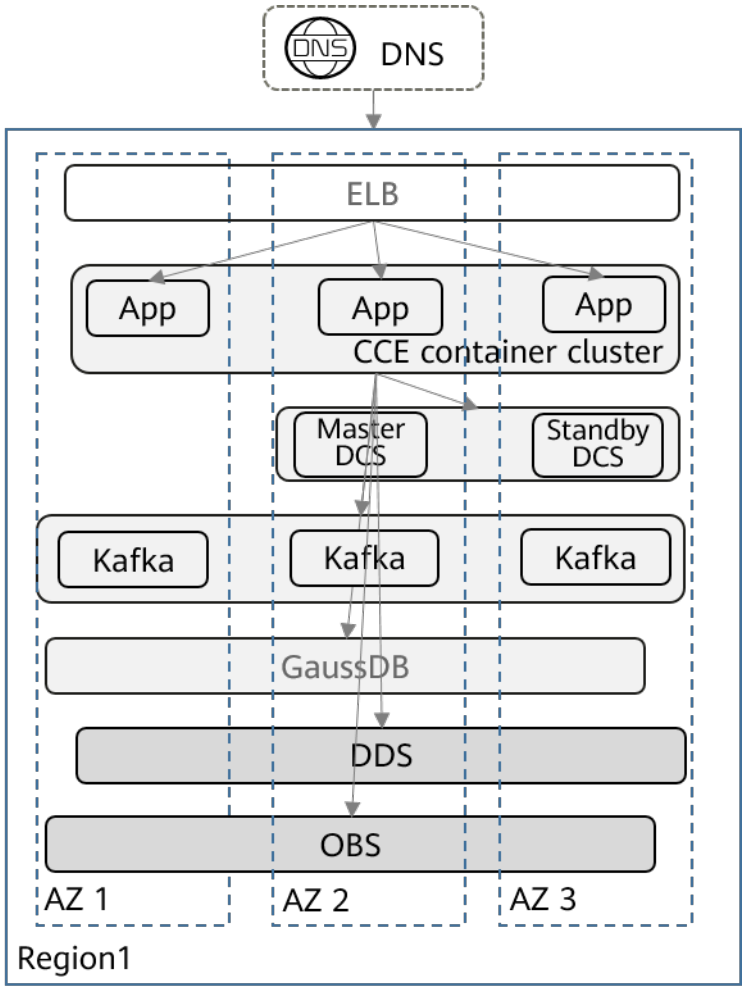
An information management application usually consists of two main parts: a stateless application layer at the frontend and a database at the backend. The frontend stateless application layer can be composed of ECS or CCE instances (CCE is used as an example here) with ELB for load balancing; while the backend typically uses GaussDB for its high performance and reliability. Other types of databases can also be used based on service requirements. For example, DDS and some middleware, such as DCS or Kafka are also common backend options. To meet the availability requirement, the recommended solution is as follows:

 Copyright © Huawei Technologies Co., Ltd.

Item	Solution
Redundancy	Deploy cloud service instances in HA mode, such as ELB, CCE, DCS, Kafka, GaussDB, and DDS instances.
Backup	Enable automated backup for GaussDB and DDS databases. When a data fault occurs, the latest backup can be used to restore data, meeting availability requirements.
DR	Deploy applications across three AZs. If an AZ is faulty, services are automatically recovered.
Monitoring metrics and alarms	Monitor and check the service running status and success metrics, and an alarm is reported when a fault occurs. Monitor the load status and resource failover of cloud service instances, and an alarm is reported when the load exceeds the threshold or the status is abnormal.
Auto scaling	Resources are sufficient for internal operations, making auto scaling unnecessary. For CCE containers, use ELB for fault detection and load balancing. Ensure that GaussDB instances can be automatically scaled up or down, and the read-only nodes can be automatically added or removed on demand based on the workload monitoring statistics.
Change error prevention	Update software offline and perform patch-related operations online. In addition, each application needs to be automatically deployed and rolled back as instructed by a runbook. The software is updated every one to two months.
Emergency recovery	Develop an emergency handling mechanism and designate related personnel to quickly make decisions and recover services. Provide solutions to common application and database problems as well as upgrade and deployment failures.

The typical deployment architecture is as follows:

 Copyright © Huawei Technologies Co., Ltd.



This architecture has the following features:

- The application system is layered with stateless applications and stateful databases.
- The application system is deployed across AZs in a single region of Huawei Cloud, so as to achieve active-active cross-DC deployment of the application layer on the cloud.
  - Access layer (external DNS): The external DNS is used for domain name resolution and traffic load balancing. The failure of a single AZ does not affect services.
  - Application layer (ELBs, application software, and containers): The stateless applications are deployed across AZs in HA mode. For these applications, ELBs are used for fault detection and load balancing, and CCE is used for workload monitoring and auto scaling.
  - Middleware layer: Redis and Kafka clusters are deployed across AZs in HA mode. The failure of a single AZ does not affect services.
  - Data layer: GaussDB, DDS, and OBS are deployed across three AZs in HA mode. These distributed databases ensure consistent data among different AZs. If a single AZ is faulty, services are not affected and no data is lost.
- The data in the GaussDB and DDS databases is automatically backed up in a scheduled manner to ensure data reliability.

 Copyright © Huawei Technologies Co., Ltd.

2.10.5 Typical Deployment Architecture for E-commerce Applications (99.99% Availability)

E-commerce applications are developed for external customers. These applications with a requirement for high availability (up to 99.99% uptime) and low mean time to recovery (a maximum of 52.56 minutes of downtime per year) must be architected to withstand component failures.

Assume that the fault interruption duration and change interruption duration are as follows:

- Fault-caused interruptions: Assume that there are three fault-caused interruptions each year. It takes 10 minutes to decide on the emergency recovery for each interruption and 5 minutes to recover application services. Thus, the total yearly interruption time is 45 minutes.
- Change-caused interruptions: Assume that applications support canary deployment or blue-green deployment, which are automatically completed. Software updates do not interrupt services.

As estimated above, the application system is unavailable for 45 minutes each year, meeting the design objectives for high availability.

The typical architecture of an e-commerce application consists of two main parts: a stateless application layer at the frontend and a database at the backend. The frontend stateless application layer can be composed of ECS or CCE instances; while the backend typically uses RDS for MySQL, depending on the service type. There are also sometimes DDS databases and middleware such as DCS and Kafka. To meet the availability requirement, the recommended solution is as follows:

2.10.5.1 Single-Region Solution

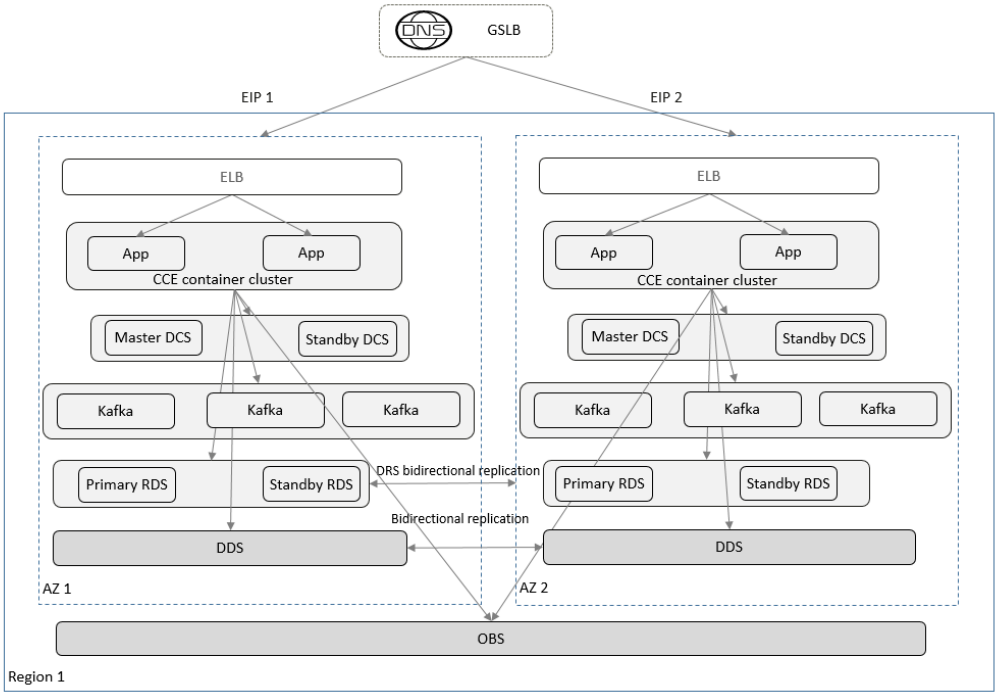
The following solution uses CCE as an example to describe the deployment architecture in a single region.

Item	Solution
Redundancy	Deploy cloud service instances in HA mode, such as ELB, CCE, DCS, Kafka, RDS, and DDS instances.
Backup	Enable automated backup for RDS and DDS databases. When a data fault occurs, the latest backup can be used to restore data, meeting availability requirements.
DR	Deploy a group of applications in two AZs separately and use bidirectional replication and active-active DR. When an AZ is faulty, service in it can be automatically recovered.
Monitoring metrics and alarms	Monitor and check site statuses and instance workloads. If a site fails or a CCE, DCS, Kafka, RDS, or DDS instance is overloaded, an alarm is reported.
Auto scaling	Use CCE clusters with auto scaling of workloads.

 Copyright © Huawei Technologies Co., Ltd.

Item	Solution
Change error prevention	Use canary or blue-green deployment for software updates. The deployment is automatically completed and rolled back upon a fault.
Emergency recovery	Develop an emergency handling mechanism and designate related personnel to quickly make decisions and recover services. Provide solutions to common application and database problems as well as upgrade and deployment failures. Periodically perform drills to identify problems in a timely manner.

A typical deployment architecture is as follows:



The architecture has the following features:

- The deployment architecture is layered with stateless applications and stateful databases.
- The application system is deployed in each of the two AZs in a region of Huawei Cloud to provide intra-city DR.
  - Access layer (external GSLB): The external GSLB is used for domain name resolution and traffic load balancing. When an AZ is faulty, service traffic load is automatically switched to another AZ.
  - Application layer (ELBs, application software, and containers): ELBs are used to detect faults and distribute loads for stateless applications, while containers enable elastic scaling.
  - Middleware layer: A DCS and a DMS for Kafka cluster are deployed in each AZ.

 Copyright © Huawei Technologies Co., Ltd.

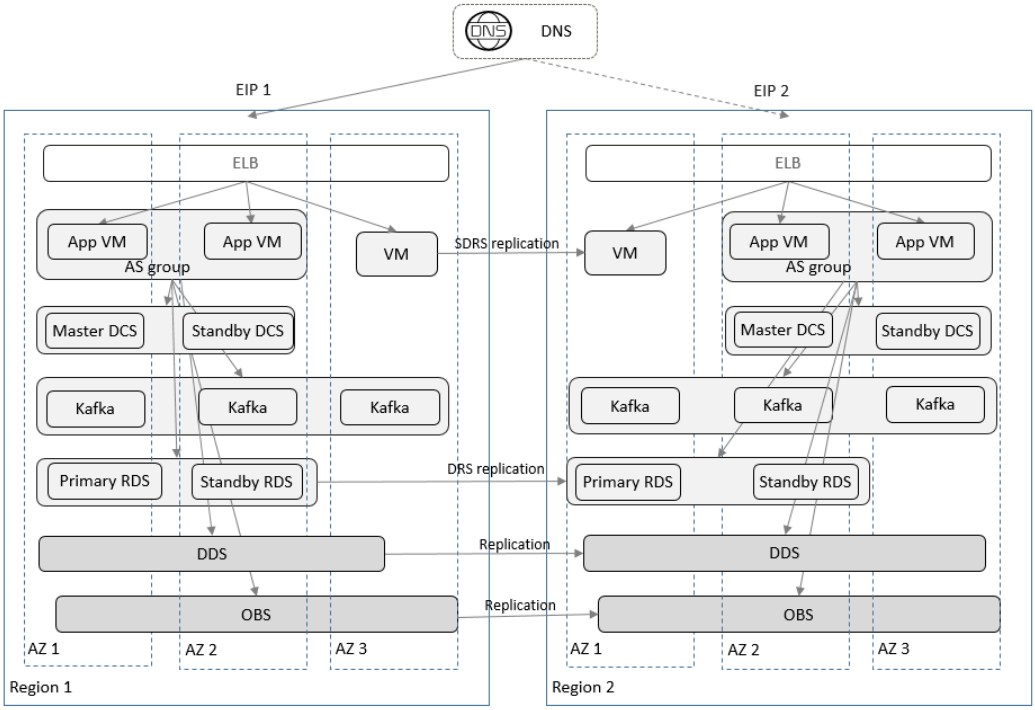
  - Data layer: A set of RDS databases are deployed in each AZ. DRS is used for cross-AZ bidirectional database replication and DR failover. Also, data can be automatically backed up in a scheduled manner to quickly restore in the event of data loss. OBS is deployed across AZs in HA mode. If a single AZ is faulty, services are not affected.
- The RDS database data is automatically backed up in a scheduled manner to ensure data reliability.

2.10.5.2 Dual-Region Solution

The following solution uses ECS as an example to describe the deployment architecture in dual regions.

Item	Solution
Redundancy	Deploy cloud service instances in HA mode, such as ELB, ECS, DCS, Kafka, RDS, and DDS instances.
Backup	Enable automated backup for RDS and DDS databases. When a data fault occurs, the latest backup can be used to restore data, meeting availability requirements.
DR	Deploy applications across AZs so that services can be automatically restored when an AZ is faulty. Enable master/standby and primary/standby DR across regions so that services can be quickly restored remotely when a region-level fault occurs.
Monitoring metrics and alarms	Monitor and check the service running status and success metrics, and an alarm is reported when a fault occurs. Monitor the load status and resource failover of ECS, DCS, Kafka, RDS, and DDS instances, and an alarm is reported when the load exceeds the threshold or the status is abnormal.
Auto scaling	For ECS instances, use ELB for fault detection and load balancing and add or remove ECS instances on demand based on the monitoring statistics, so as to enhance the application performance. For RDS for MySQL, change the instance type or add read-only nodes during maintenance based on the workload monitoring statistics.
Change error prevention	Use canary or blue-green deployment for software updates. The deployment is automatically completed and rolled back upon a fault.
Emergency recovery	Develop an emergency handling mechanism and designate related personnel to quickly make decisions and recover services. Provide solutions to common application and database problems as well as upgrade and deployment failures. Periodically perform drills to identify problems in a timely manner.

A typical deployment architecture is as follows:



The architecture has the following features:

- The deployment architecture is layered with stateless applications and stateful databases or VMs.
- A complete application system is deployed in each of the primary and standby regions with data synchronization. Cross-AZ HA in a region provides active-active capabilities across DCs in the same city. Data can be asynchronously synchronized between regions in real time. Master/standby and primary/standby DR is used to quickly restore services to another region when a region is faulty.
  - Access layer (external DNS servers and API gateways): It uses external DNS servers to resolve domain names. Service traffic is manually switched to another region if the current region becomes faulty.
  - Application layer (ELBs, application software, and VMs): ELBs detect faults and distribute loads. AS monitors load conditions and triggers automatic scaling. For stateful applications, SDRS facilitates cross-cloud VM data replication and DR failover. In addition, CBR is used for automated data backup.
  - Middleware layer: Redis and Kafka clusters are deployed across AZs for high availability.
  - Data layer: MySQL databases are deployed for high availability. DRS is used to implement cross-cloud database replication and DR failover. Data can be automatically backed up periodically to quickly restore services in the event of data loss. OBS also supports cross-region database replication.
- The RDS database data is automatically backed up to OBS periodically to ensure that data can be quickly restored in the case of data loss.

 Copyright © Huawei Technologies Co., Ltd.

2.10.6 Typical Deployment Architecture for Core Financial Applications (99.999% Availability)

Core financial applications require very short recovery time and can tolerate a small amount of data loss. The uptime needs to be up to 99.999%, that is, a maximum of 5.26 minutes of downtime per year.

Assume that the fault interruption duration and change interruption duration are as follows:

- Fault-caused interruptions: The duration must be as short as possible. Therefore, automatic recovery is required and no manual recovery can be triggered. Assume that there are four fault-caused interruptions each year. It takes 1 minute to recover automatically for each interruption. Thus, the total yearly interruption time is 4 minutes.
- Change-caused interruptions: Assume that applications support canary deployment or blue-green deployment, which are automatically completed. Software updates do not interrupt services.

As estimated above, the application system is unavailable for 4 minutes each year, meeting the design objectives for high availability.

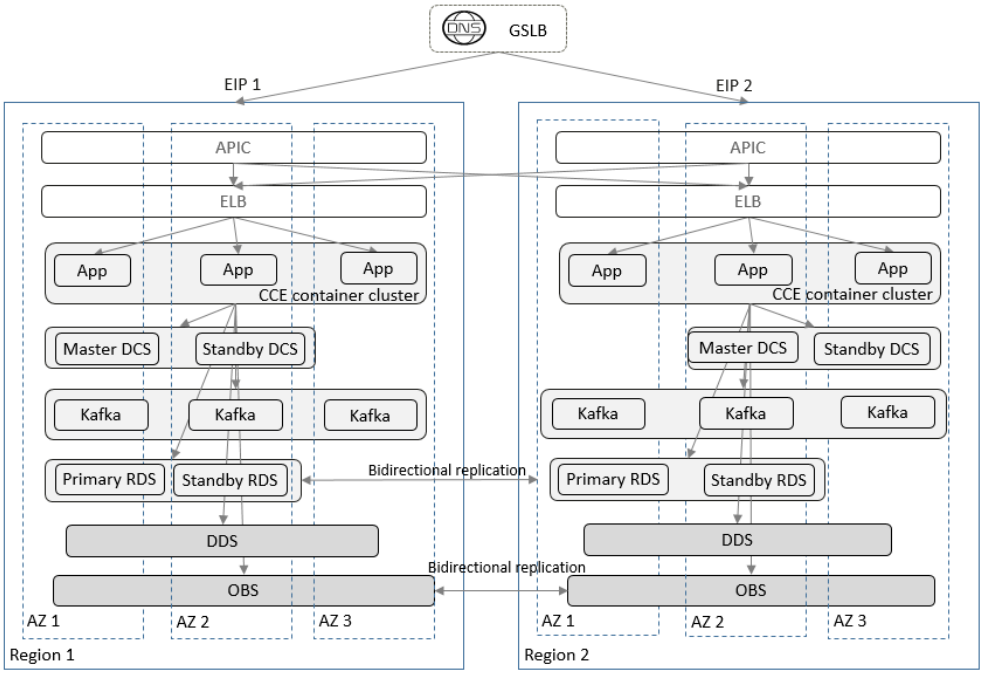
The typical architecture of financial applications consists of three layers: frontend web cluster, backend application cluster, and backend database cluster. The frontend stateless applications can use ECS or CCE (CCE is used as an example in this section), and the backend databases use RDS for MySQL to provide higher performance and availability. To meet the required availability, the following solution is recommended:

Item	Solution
Redundancy	Deploy cloud service instances in HA mode, such as ELB, CCE, DCS, Kafka, RDS, and DDS instances.
Backup	Enable automated backup for RDS and DDS databases. When a data fault occurs, the latest backup can be used to restore data, meeting availability requirements.
DR	Deploy applications across AZs so that services can be automatically restored when an AZ is faulty. Enable active-active DR across regions so that services can be quickly restored remotely when a region-level fault occurs.
Monitoring metrics and alarms	Monitor and check site statuses and instance workloads. If a site fails or a CCE, DCS, Kafka, RDS, or DDS instance is overloaded, an alarm is reported.
Auto scaling	Use CCE clusters with auto scaling of workloads.
Change error prevention	Use canary or blue-green deployment for software updates. The deployment is automatically completed and rolled back upon a fault.

 Copyright © Huawei Technologies Co., Ltd.

Item	Solution
Emergency recovery	Develop an emergency handling mechanism and designate related personnel to quickly make decisions and recover services. Provide solutions to common application and database problems as well as upgrade and deployment failures. Periodically perform drills to identify problems in a timely manner.

A typical deployment architecture is as follows:



The architecture has the following features:

- The deployment architecture is layered with stateless applications and stateful databases.
- A complete application system is deployed in each of the two regions. Cross-AZ HA in a region provides active-active capabilities across DCs in the same city. Data units are deployed between regions for cross-region active-active DR. If a region is faulty, services can be quickly restored to another region.
  - Access layer (external GSLB and API gateways): The external GSLB handles domain name resolution and traffic load balancing. Both regions provide services simultaneously, and if a region is faulty, service traffic is automatically redirected to another region. API gateways ensure traffic is correctly routed to the appropriate service units.
  - Application layer (ELBs, application software, and containers): ELBs are used to detect faults and distribute loads for stateless applications, while containers enable elastic scaling.
  - Middleware layer: Redis and Kafka clusters are deployed across AZs for high availability.

 Copyright © Huawei Technologies Co., Ltd.

  - Data layer: MySQL databases are deployed across AZs for high availability. DRS is used to implement cross-region bidirectional database replication and DR failover. Data can be automatically backed up periodically to quickly restore data in the event of data loss. OBS also supports cross-region bidirectional database replication.
- The data in RDS for MySQL and DDS databases is automatically backed up periodically to ensure data reliability.

2.10.7 Cross-Cloud Scenarios (99.99% Availability)

2.10.7.1 Overview

E-commerce applications are used by external customers, require high availability, and should withstand component faults. The availability objective is 99.99%, that is, the annual fault interruption duration should not exceed 52.56 minutes.

Assume that the fault interruption duration and change interruption duration are as follows:

- Fault-caused interruptions: Assume that there are three fault-caused interruptions each year. It takes 10 minutes to decide on the emergency recovery for each interruption and 5 minutes to recover application services. Thus, the total yearly interruption time is 45 minutes.
- Change-caused interruptions: Assume that applications support canary deployment or blue-green deployment, which are automatically completed. Software updates do not interrupt services.

According to the preceding evaluation, the application system is unavailable for 45 minutes every year, meeting the availability design objective.

The typical architecture of an e-commerce application is a stateless application layer at the frontend and databases at the backend. The frontend stateless applications can be deployed on ECSs or CCE instances. The backend databases can be RDS for MySQL or other databases based on service types. Middleware such as DCS and Kafka and DDS databases are also used. The following solutions are recommended to meet the availability objectives.

2.10.7.2 Cross-Cloud DR

If critical application systems are deployed in an IDC or another cloud environment and need to be backed up to Huawei Cloud for disaster recovery (DR), a cross-cloud DR solution is essential to ensure high availability. Assuming the systems in the IDC or other cloud already have 99.9% availability, using Huawei Cloud DR can bring it up to 99.99%.

A typical architecture for cross-cloud applications includes a frontend stateless application layer and a backend database layer. The frontend application layer uses VMs or containers (using services like Huawei Cloud CCE), while the backend database layer uses MySQL databases (like you get with Huawei Cloud RDS for MySQL) to support cross-cloud DR.

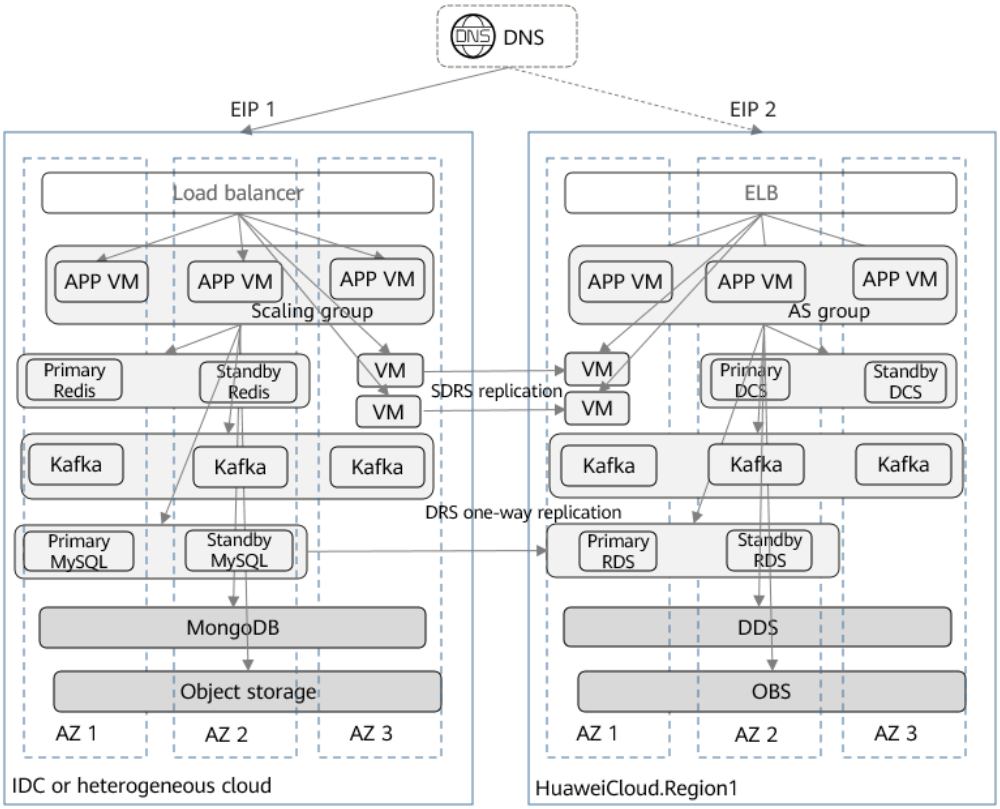
The recommended deployment on Huawei Cloud is as follows.

 Copyright © Huawei Technologies Co., Ltd.

Item	Solution
Redundancy	Deploy cloud services like ELB, CCE, DCS, Kafka, RDS, and DDS, to ensure high availability.
Backup	Enable automated backup for RDS and DDS databases. When a data fault occurs, the latest backup can be used to restore data, meeting availability requirements.
DR	Deploy applications across AZs. If an AZ fails, applications can be automatically recovered. Use cross-cloud DR to switch workloads to Huawei Cloud quickly if an IDC or another cloud environment fails.
Monitoring	Monitor and check site statuses and instance workloads. If a site fails or a CCE, DCS, Kafka, RDS, or DDS instance is overloaded, an alarm is reported.
Auto scaling	Use CCE clusters with auto scaling of workloads.
Change error prevention	Use canary or blue-green deployment for software updates. Canary and blue-green deployments can be completed automatically and rolled back if an update does not go smoothly.
Emergency recovery	Develop an emergency response system and designate emergency personnel to make decisions quickly and recover services. Provide solutions for common application and database issues, as well as upgrade and deployment failures. Conduct periodic drills to promptly identify and address potential problems.

A typical deployment architecture is as follows.

 Copyright © Huawei Technologies Co., Ltd.



The architecture has the following features:

- The application system uses a multi-layered architecture featuring stateless applications and stateful databases or VMs.
- The application system is deployed either in an IDC or on another cloud platform, as well as on Huawei Cloud. On Huawei Cloud, the system is distributed across AZs to facilitate active-active DR at the application level between different data centers. Data from other clouds can be synchronized to Huawei Cloud in real-time. If there is a failure in the IDC or another cloud, the primary/standby DR solution enables workloads to be rapidly switched over to Huawei Cloud.
  - Access layer (external DNS servers and API gateways): This layer utilizes external DNS servers for domain name resolution. If the IDC or another cloud environment fails, service traffic is manually redirected to Huawei Cloud.
  - Application layer (load balancers, application software, and VMs or physical hosts): Load balancers are employed for stateless applications to detect faults and distribute loads. On Huawei Cloud, Auto Scaling (AS) monitors loads and triggers automatic scaling. For stateful applications, Storage Disaster Recovery Service (SDRS) facilitates cross-cloud VM data replication and DR switchover. In addition, Cloud Backup and Recovery (CBR) is used for automated data backups.
  - Middleware layer: Redis and Kafka clusters are deployed across AZs for high availability.
  - Data layer: MySQL databases are designed for high availability. Data Replication Service (DRS) is used to enable cross-cloud database

 Copyright © Huawei Technologies Co., Ltd.

replication and DR switchover. Data is automatically backed up on a regular basis. If any data is lost, workloads can be quickly restored.

- To ensure data reliability, database data is automatically backed up periodically, enabling quick restoration in the event of data loss.

2.10.7.3 Cross-Cloud Active-Active DR

If critical application systems are deployed in an IDC or another cloud environment and need to be deployed on Huawei Cloud for active-active DR, a cross-cloud active-active DR solution is required to provide high availability. Assuming the systems in the IDC or other cloud already have 99.9% availability, using Huawei Cloud DR can bring it up to 99.99%.

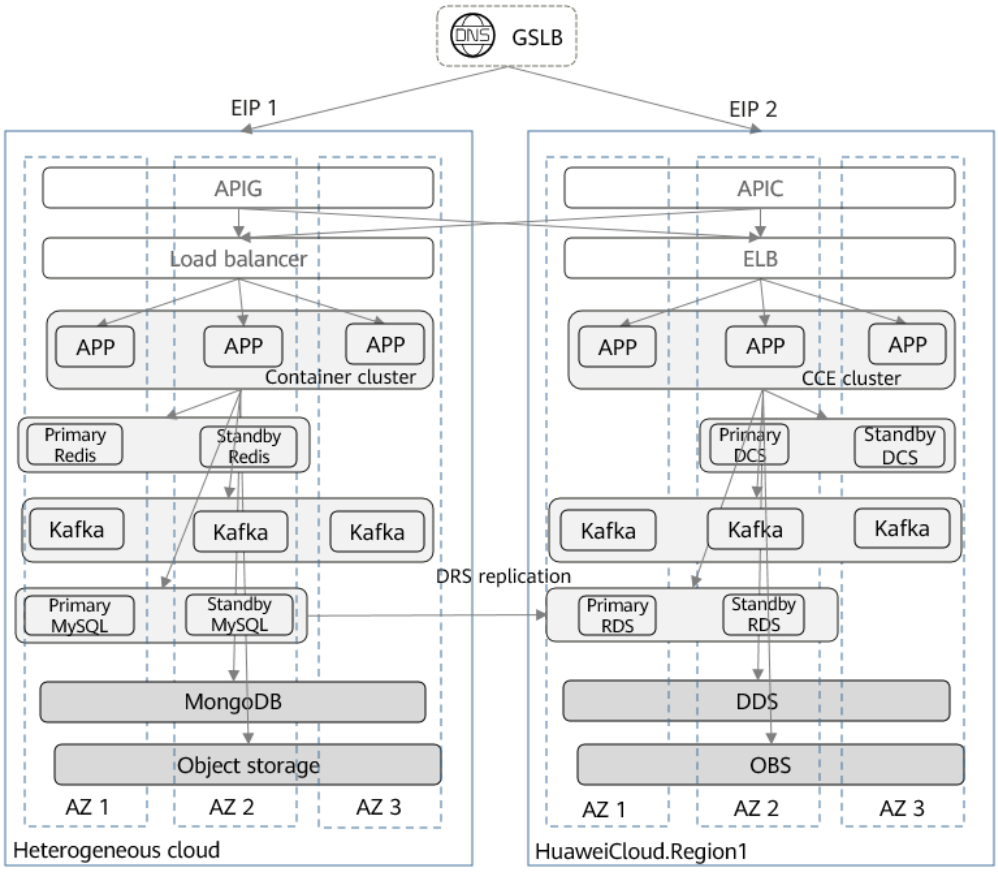
A typical architecture for cross-cloud applications includes a frontend stateless application layer and a backend database layer. The frontend application layer uses VMs or containers (using services like Huawei Cloud CCE), while the backend database layer uses MySQL databases (like you get with Huawei Cloud RDS for MySQL) to support cross-cloud active-active DR.

The recommended deployment on Huawei Cloud is as follows.

Item	Solution
Redundancy	Deploy cloud services like ELB, CCE, DCS, Kafka, RDS, and DDS, to ensure high availability.
Backup	Enable automated backup for RDS and DDS databases. When a data fault occurs, the latest backup can be used to restore data, meeting availability requirements.
DR	Deploy applications across AZs. If an AZ fails, applications can be automatically recovered. Use cross-cloud active-active DR to switch workloads to Huawei Cloud quickly if an IDC or another cloud environment fails.
Monitoring	Monitor and check site statuses and instance workloads. If a site fails or a CCE, DCS, Kafka, RDS, or DDS instance is overloaded, an alarm is reported.
Auto scaling	Use CCE clusters with auto scaling of workloads.
Change error prevention	Use canary or blue-green deployment for software updates. Canary and blue-green deployments can be completed automatically and rolled back if an update does not go smoothly.
Emergency recovery	Develop an emergency response system and designate emergency personnel to make decisions quickly and recover services. Provide solutions for common application and database issues, as well as upgrade and deployment failures. Conduct periodic drills to promptly identify and address potential problems.

A typical deployment architecture is as follows.

 Copyright © Huawei Technologies Co., Ltd.



The architecture has the following features:

- The application system uses a multi-layered architecture featuring stateless applications and stateful databases.
- The application system is deployed on both another cloud platform and Huawei Cloud. On Huawei Cloud, the system is distributed across AZs to facilitate active-active DR at the application level between different data centers. Data from other clouds is deployed in separate units and can be synchronized to Huawei Cloud in real-time. If there is a failure on a third-party cloud, the active-active DR solution enables workloads to be rapidly switched over to Huawei Cloud.
  - Access Layer (external GSLB and API Gateway): The external GSLB handles domain name resolution and traffic load balancing. Both clouds operate simultaneously, and if a third-party cloud fails, service traffic is automatically redirected to Huawei Cloud. API Gateway ensures traffic is correctly routed to the appropriate service units.
  - Application Layer (load balancers, application software, and containers): Load balancers are used to detect faults and distribute loads for stateless applications, while containers enable elastic scaling.
  - Middleware layer: Redis and Kafka clusters are deployed across AZs for high availability.
  - Data layer: MySQL databases are deployed across multiple AZs for high availability. DRS is used to implement cross-cloud database replication and DR switchover.

 Copyright © Huawei Technologies Co., Ltd.

- To ensure data reliability, database data is automatically backed up periodically, enabling quick restoration in the event of data loss.

2.11 Cloud Service Reliability

2.11.1 Overview

This section describes the functions available to enhance the reliability of typical cloud services used to run application systems. It also explains how you can use these functions to easily rectify common faults.

Table 2-1

Cloud Service	Description	Details
Elastic Cloud Server (ECS)	An ECS is a cloud server with its own vCPUs, memory, OS, and Elastic Volume Service (EVS) disks. After creating an ECS, you can use it like you would use your local PC or physical server.	ECS
Bare Metal Server (BMS)	A BMS combines the scalability of an ECS with the high performance of physical servers. It provides dedicated servers on the cloud, delivering the high performance and security required by core databases, mission-critical applications, high-performance computing (HPC) services, and big data.	BMS
Cloud Container Engine (CCE)	CCE is a highly scalable, high-performance, enterprise-class Kubernetes service you can use to run Docker containers. With CCE, you can easily deploy, manage, and scale containerized applications on the cloud.	CCE

 Copyright © Huawei Technologies Co., Ltd.

Cloud Service	Description	Details
Elastic Load Balance (ELB)	ELB automatically distributes incoming traffic across multiple backend servers based on the listening rules you configure. There are dedicated and shared load balancers: • Dedicated load balancers: You get exclusive access to load balancer resources. The performance of a dedicated load balancer is never affected by the loads on other load balancers. There are a wide range of specifications available for you to choose from. • Shared load balancers: These are deployed in clusters. They share resources, so the performance of one load balancer can be affected by the loads on other load balancers. There are no specifications for you to choose from.	**ELB**
Auto Scaling (AS)	AS helps you automatically scale ECSs and bandwidth resources to keep up with changes in demand based on the scaling policies you configure. It allows you to add ECSs or bandwidth resources to handle traffic spikes and also save money by removing ECSs or bandwidth resources that are sitting idle.	**AS**
Distributed Cache Service (DCS)	Huawei Cloud DCS is fast, distributed, online in-memory cache service compatible with Redis. It is reliable, scalable, usable out of the box, and easy to manage, meeting your requirements for high read/write performance and fast data access.	**DCS**

 Copyright © Huawei Technologies Co., Ltd.

Cloud Service	Description	Details
Distributed Message Service (DMS)	DMS supports diverse message types: • DMS for Kafka: compatible with open-source Kafka. It provides Kafka instances with isolated compute, storage, and bandwidth resources. • DMS for RabbitMQ: compatible with open-source RabbitMQ. It provides messaging services with numerous messaging features, flexible routing, high availability, monitoring, and alarming functions. It works for flash sales, flow control, and system decoupling scenarios. • DMS for RocketMQ: message-oriented middleware that delivers low latency, high flexibility, high throughput, dynamic expansion, easy management, and abundant messaging functions.	**DMS**
Relational Database Service (RDS)	RDS is a cloud-based web service that is reliable, scalable, and easy to manage.	**RDS**
TaurusDB	TaurusDB is a Huawei's proprietary enterprise-grade distributed database fully compatible with MySQL. It decouples storage from compute and uses Huawei-developed Data Function Virtualization (DFV), which scales to up to 128 TB per instance. A failover can be completed within seconds. It provides the superior performance and high availability of a commercial database at the price of an open-source database.	**TaurusDB**
Object Storage Service (OBS)	OBS is a secure, highly reliable object storage service that allows you to inexpensively store massive amounts of data.	**OBS**

2.11.2 ECS

2.11.2.1 Reliability Functions

Data Backup and Restoration

You can use Cloud Backup and Recovery (CBR) to back up and restore ECSs. CBR works based on the consistent snapshots for EVS disks.

For details, see **CBR Overview**.

 Copyright © Huawei Technologies Co., Ltd.

Automatic Fault Recovery

You can enable automatic recovery for ECSs. This will help ensure high availability and make it easier for them to be dynamically migrated. If a physical host accommodating ECSs breaks down, the ECSs will automatically be migrated to a functional physical host to minimize the impact on your services. During the process, the ECSs will restart.

For details, see **Can ECSs Automatically Recover After the Physical Host Accommodating the ECSs Becomes Faulty?**

When the system detects that a physical host accommodating ECSs is sub-healthy, the system automatically migrates the ECSs to another available physical host. During the migration, there is a brief service interruption of less than 1 second, and the performance is impacted.

Anti-Affinity

An ECS group allows you to create ECSs on different physical hosts to improve service reliability.

For details, see **Managing ECS Groups**.

Cluster HA

You can use a shared EVS disk to deploy an intra-AZ cluster or a mission-critical application with HA enabled. One shared EVS disk can be attached to up to 16 ECSs. You also need to deploy a shared file system or a cluster management system, such as Windows MSCS, Veritas VCS, or CFS.

For higher reliability, you are advised to attach a shared EVS disk to ECSs in the same anti-affinity ECS group.

For details, see **Shared EVS Disks and Usage Instructions**.

Load Balancing

You can use ELB to enable load balancing across multiple ECSs.

Health Checks

You can use ELB to perform health checks on ECSs.

For details, see **Configuring a Health Check**.

Auto Scaling

With Auto Scaling, you can scale ECSs to balance loads across AZs.

Cross-AZ DR

You can use ELB to enable failovers across AZs.

 Copyright © Huawei Technologies Co., Ltd.

Monitoring

Using Cloud Eye, you can monitor ECS resources such as vCPUs, memory, disks, and networks.

For details, see **Monitoring ECSs**.

2.11.2.2 Common Faults

Excessive CPU, Memory, or Disk Usage or Disk IOPS on an ECS

- Check: Use Cloud Eye to check the CPU, memory, or disk usage or disk IOPS.
- Recovery:
  a. Scale up resources or add ECSs for balancing loads based on service requirements.
  b. For stateless services, enable AS to automatically scale out resources.
  c. Enable overload protection for the application layer to keep high-priority services running smoothly.

Failed to Connect to Backend ECSs

- Check: Network connection failed.
- Recovery:
  a. Deploy at least two backend ECSs. For stateless services, configure ELB to ensure service reliability. For stateful services, enable multi-instance HA at the application layer.
  b. If this is a temporary failure, for example, if backend ECSs are being recovered, connect to the ECSs again at the application layer. For details, see **RES09 Retries After Failures**.
  c. If the connection to an overloaded ECS failed, rectify the fault by referring to "Excessive CPU, Memory, or Disk Usage or Disk IOPS on an ECS."

Unavailable or Abnormal ECSs

- Check: Configure a scheduled health check for backend servers of load balancers to check whether their key functions are normal.
- Recovery: Configure multiple ECSs for each application layer and use a load balancer to perform a health check on the ECSs. If an ECS is unavailable, the load balancer will not send service requests to the ECS any more.

ECSs, Disks Attached on Them, or Their Data Are Deleted Unexpectedly

- Check: N/A
- Recovery: For stateless services, use a template to quickly provision new ECSs. For stateful services, use CBR to periodically back up ECSs, and use the backups to quickly restore their data.

A Local Disk Used by an ECS Is Faulty

- Check: Check the status of local disks at the application layer.

 Copyright © Huawei Technologies Co., Ltd.

- Recovery: At the application layer, HA is enabled for disks of a single ECS using RAID, and data replication and HA across ECSs are also enabled. In this way, the ECS service can be recovered quickly when a local disk is faulty. You are advised to use EVS disks instead of local disks as they are more reliable.

2.11.3 BMS

2.11.3.1 Reliability Functions

Data Backup and Restoration

You can use CBR to back up any EVS disk (a system disk or data disk) attached to BMSs. CBR works based on the consistent snapshot technology for EVS disks. With CBR, you can use backup data to restore BMS data, ensuring data security and correctness.

For details, see **Backing Up a BMS**.

Cluster HA

You can use a shared EVS disk to deploy an intra-AZ cluster or a mission-critical application with HA enabled. One shared EVS disk can be attached to up to 16 BMSs. You also need to deploy a shared file system or a cluster management system, such as Windows MSCS, Veritas VCS, or Veritas CFS.

For details, see **Shared EVS Disks and Usage Instructions**.

Load Balancing

You can use ELB to enable load balancing across multiple BMSs.

Health Checks

You can use ELB to perform health checks on BMSs.

For details, see **Configuring a Health Check**.

Cross-AZ DR

You can use ELB to enable failovers across AZs.

Monitoring

Using Cloud Eye, you can monitor BMS resources such as CPUs, memory, disks, and networks.

For details, see **Monitoring Metrics**.

 Copyright © Huawei Technologies Co., Ltd.

2.11.3.2 Common Faults

Excessive CPU, Memory, or Disk Usage or Disk IOPS on a BMS

- Check: Use Cloud Eye to check the CPU, memory, or disk usage or disk IOPS.
- Recovery:
  a. Use a new BMS with higher specifications or add BMSs for balancing loads based on service requirements.
  b. Enable overload protection for the application layer to keep high-priority services running smoothly.

Failed to Connect to Backend BMSs

- Check: Network connection failed.
- Recovery:
  a. Deploy at least two backend BMSs. For stateless services, configure ELB to ensure service reliability. For stateful services, enable multi-instance HA at the application layer.
  b. If this is a temporary failure, for example, network overload, retry at the application layer. For details, see **RES09 Retries After Failures**.
  c. If the connection to an overloaded BMS failed, rectify the fault by referring to "Excessive CPU, Memory, or Disk Usage or Disk IOPS on a BMS."

Unavailable or Abnormal BMSs

- Check: Configure a scheduled health check for backend servers of load balancers to check whether their key functions are normal.
- Recovery: Configure multiple BMSs for each application layer and use a load balancer to perform a health check on the BMSs. If a BMS is unavailable, the load balancer will not send service requests to the BMS any more.

BMSs, Disks Attached on Them, or Their Data Are Deleted Unexpectedly

- Check: N/A
- Recovery: For stateless services, use a template to quickly provision new BMSs. For stateful services, use CBR to periodically back up EVS disks attached to BMSs, and use the backups to quickly restore their data.

A Physical Server or Local Disk Used by a BMS Is Faulty

- Check: Check the status of physical servers and local disks at the application layer.
- Recovery: At the application layer, HA is enabled for disks of a single BMS using RAID, and data replication and HA across BMSs are also enabled. In this way, the BMS service can be quickly recovered in case that a physical server or local disk is faulty. You are advised to use EVS disks instead of local disks as they are more reliable.

 Copyright © Huawei Technologies Co., Ltd.

2.11.4 CCE

2.11.4.1 Reliability Functions

Cluster HA

Three CCE master nodes can be deployed in HA mode to ensure cluster reliability.

Data Backup and Restoration

For persistent storage, you can use CCE to mount storage volumes created from EVS disks to a path of a container. CCE works with EVS to support snapshots. If data is lost, you can roll back the disk data to the state when a snapshot was created.

For details, see **Snapshots and Backups**.

Health Checks

Container health can be checked regularly when the container is running. If health check is not configured, a pod cannot detect application exceptions or automatically restart the application to recover it. As a result, a pod status can be normal even if the application in the pod is not.

Kubernetes provides the following health check probes:

- Liveness probe (livenessProbe): checks whether a container is still alive. It is similar to the **ps** command that checks whether a process exists. If the liveness probe of a container fails, the cluster restarts the container. If the liveness probe is successful, no further action is taken.
- Readiness probe (readinessProbe): checks whether a container is ready to process user requests. Upon that the container is detected unready, service traffic will not be directed to the container. An application may take a long time to start up and provide services due to some reasons, for example, it may need to load disk data or wait for the startup of an external module. In this case, application processes are running, but the applications are not ready to provide services. This is where the readiness probe enters. If the container readiness probe fails, the cluster masks all requests sent to the container. If the container readiness probe is successful, the container can be accessed.
- Startup probe (startupProbe): checks when a containerized application has started. If such a probe is configured, it disables liveness and readiness checks until it succeeds, ensuring that those probes do not interfere with the application startup. This can be used to perform liveness checks on slow starting containers to prevent them from getting terminated by the kubelet before they are started.

For details, see **Configuring Container Health Check**.

Anti-Affinity

CCE supports node anti-affinity. When creating a node pool, you can select an ECS group and configure an anti-affinity policy for it. Then ECSs in the same ECS group will be distributed to different hosts for higher reliability.

 Copyright © Huawei Technologies Co., Ltd.

CCE supports affinity and anti-affinity between workloads and nodes as well as between workloads:

- Node affinity: You can choose to deploy workloads in a specified node or AZ or not.
- Affinity or anti-affinity between workloads: Workloads are deployed on the same node, to minimize the usage of network resources, or on different nodes, to minimize the impacts of potential breakdowns.

For details, see **Scheduling Policies (Affinity and Anti-affinity)**.

Overload Control

CCE clusters support overload control. If this function is enabled, concurrent requests will be dynamically controlled based on the resource demands received by master nodes to ensure the stable running of the master nodes and the cluster.

For details, see **Enabling Overload Control for a Cluster**.

Auto Scaling

CCE supports auto scaling of workloads and nodes:

- Workload scaling: auto scaling at the scheduling layer to change the scheduling capacity of workloads. For example, you can use HPA, a scaling component at the scheduling layer, to adjust the number of pods used for an application. Adjusting the number of pods changes the scheduling capacity occupied by the current workload, thereby enabling scaling at the scheduling layer.
- Node scaling: auto scaling at the resource layer. When the planned cluster nodes cannot allow workload scheduling, ECS or CCI resources are provided to support scheduling.

Workload scaling and node scaling can work separately or together.

For details, see **Auto Scaling Overview**.

Cross-AZ DR

CCE allows you to create or scale clusters across AZs. Workloads are automatically distributed evenly among these AZs.

Monitoring

CCE can work with Application Operations Management (AOM) to monitor a wide array of resource objects such as clusters, nodes, workloads, and pods.

For details, see **Monitoring Overview**.

2.11.4.2 Common Faults

Excessive CPU, Memory, or Disk Usage of a CCE Cluster

- Check: Use AOM to check the CPU, memory, and disk usage of a CCE cluster.

 Copyright © Huawei Technologies Co., Ltd.

- Recovery:
  a. Change the cluster specifications or add resources based on service requirements.

Excessive CPU, Memory, Disk, GPU, or GPU Cache Usage or Disk IOPS of a CCE Node

- Check: Use AOM to check the CPU, memory, disk, GPU, or GPU cache usage or disk IOPS of a CCE node.
- Recovery:
  a. Change the node specifications or add nodes based on service requirements.

Excessive CPU, Memory, GPU, or GPU Cache Usage of a CCE Workload

- Check: Use AOM to check the CPU, memory, GPU, or GPU cache usage of a CCE workload.
- Recovery:
  a. Adjust the resource quotas of the workload or add workloads based on service requirements.

2.11.5 ELB

2.11.5.1 Reliability Functions

Cluster HA

Load balancers of ELB can be deployed in clusters. They can route traffic across AZs, ensuring that your services are uninterrupted. If servers in an AZ are unhealthy, ELB automatically routes traffic to healthy servers in other AZs.

Backend Server Health Checks

ELB periodically sends requests to backend servers to check whether they can process requests. If a backend server is identified as unhealthy, the load balancer will route requests to healthy servers, which improves service reliability. After the backend server recovers, the load balancer will resume routing requests to it.

For details, see **Health Check**.

Cross-AZ DR

Load balancers of ELB can be deployed in clusters. They can route traffic across AZs, ensuring that your services are uninterrupted. If servers in an AZ are unhealthy, ELB automatically routes traffic to healthy servers in other AZs.

Backend servers of ELB can be deployed across AZs. If an ECS in an AZ is unhealthy, ELB can forward traffic to ECSs in other AZs, improving the fault tolerance of applications.

 Copyright © Huawei Technologies Co., Ltd.

Monitoring

Using Cloud Eye, you can monitor ELB metrics such as the number of connections, bandwidth, and abnormal responses.

For details, see **Monitoring Metrics**.

2.11.5.2 Common Faults

Too Many Concurrent or New Connections and Excessive Bandwidth Usage of a Load Balancer?

- Check: Use Cloud Eye to monitor the number of concurrent connections, the number of new connections, and bandwidth usage of a load balancer.
- Recovery:
  a. Use dedicated load balancers and adjust the specifications of load balancers based on service requirements.

2.11.6 AS

2.11.6.1 Reliability Functions

Load Balancing

This service works with ELB to balance loads among ECSs created in an AS group.

Health Checks

A health check removes unhealthy ECSs from an AS group. Then, AS adds new ECSs to the AS group so that the number of ECSs is the same as the expected number. There are two types of AS group health checks.

- **ECS health check**: checks the ECS running status. If an ECS is stopped or deleted, it is considered as abnormal. **ECS health check** is the default health check mode for an AS group. The AS group periodically uses the check result to determine the running status of every ECS in the AS group. If an ECS is unhealthy, AS removes the ECS from the AS group.
- **ELB health check**: determines ECS running status using a load balancing listener. If the AS group uses load balancers, the health check method can also be **ELB health check**. If you add multiple load balancers to an AS group, an ECS is considered healthy only when all load balancers declare that it is healthy. If any load balancer detects that an ECS is unhealthy, the ECS will be removed from the AS group.

In both **ECS health check** and **ELB health check** methods, AS removes unhealthy ECSs from the AS group.

For details, see **Auto Scaling Health Check**.

 Copyright © Huawei Technologies Co., Ltd.

Cross-AZ DR

Backend servers of AS can be deployed across AZs. If an AZ is faulty, AS can automatically create ECSs in other AZs to quickly recover services.

If multiple AZs are selected, set **Multi-AZ Scaling Policy** to **Balanced** or **Sequenced**.

- **Balanced**: When scaling out an AS group, the system preferentially distributes ECSs evenly across AZs used by the AS group. If it fails in the target AZ, it automatically selects another AZ based on the sequenced policy.
- **Sequenced**: When scaling out an AS group, the system distributes ECSs to the AZs according to the order in which AZs are specified.

Monitoring

Using Cloud Eye, you can monitor AS group resources such as vCPUs, memory, disks, and networks.

For details, see **Monitoring Metrics**.

2.11.6.2 Common Faults

Auto Scaling Failure

- Check: Check the historical records of scaling actions of an AS group.
- Recovery:
  a. Rectify the fault based on the scaling action failure description.

2.11.7 DCS

2.11.7.1 Reliability Functions

Cluster HA

DCS provides master/standby, Proxy Cluster, and Redis Cluster instances. Each instance has multiple nodes for instance-level DR. If the master node is faulty, a failover is triggered automatically. Services may be interrupted for less than half a minute during exception detection and the failover.

Data Backup and Restoration

Data in DCS instances can be backed up to OBS. If a DCS instance becomes faulty, data in the instance can be restored from the backup. DCS instances support scheduled and manual backup. Backup is scheduled in days, can be stored for a maximum of seven days, and at least one backup file is retained. Users can perform manual backup before service system maintenance, upgrade, and other high-risk operations. The retention period is not limited.

DCS instances can be restored from a specified backup set. During the restoration, DCS instances do not accept data operation requests from clients because existing data is being overwritten by the backup data.

 Copyright © Huawei Technologies Co., Ltd.

For details, see **DCS Backup and Restoration Overview**.

Cross-AZ DR

Cross-AZ DR is supported for master/standby, Redis Cluster, and Proxy Cluster DCS instances. If one AZ is faulty, nodes in another AZ are not affected. The standby node automatically becomes the master node and takes over services.

Monitoring

Using Cloud Eye, you can monitor DCS resources such as vCPUs, memory, disks, and networks.

For details, see **DCS Metrics**.

2.11.7.2 Common Faults

Excessive CPU, Memory, Bandwidth, or Connection Usage of a DCS Instance

- Check: Use Cloud Eye to check the CPU, memory, bandwidth, or connection usage.
- Recovery:
  a. Change the specifications to expand resources based on service requirements.
  b. Enable overload protection for the application layer to keep high-priority services running smoothly. Switch some services that do not require high performance back to the original data source.

Failed to Connect to a Backend DCS Instance

- Check: Network connection failed.
- Recovery:
  a. If this is a temporary failure, for example, a master/standby switchover is in progress, connect to the DCS instance again at the application layer. For details, see **RES09 Retries After Failures**.
  b. If the connection to an overloaded DCS instance failed, rectify the fault by referring to "Excessive CPU, Memory, Bandwidth, or Connection Usage of a DCS Instance."
  c. For non-temporary failures, the data source at the application layer needs to be switched back to the original one for processing. This prevents service interruption due to cache faults.

Occasionally Failed to Read Data from or Write Data to a DCS Instance

- Check: Data failed to be read or written. Occasional timeout errors are normal in Redis because of network connectivity and client timeout configurations.
- Recovery:
  a. If this is a temporary failure, for example, the DCS instance is undergoing a master/standby switchover, retry at the application layer. For details, see **RES09 Retries After Failures**.

 Copyright © Huawei Technologies Co., Ltd.

b. If the connection to an overloaded DCS instance failed, rectify the fault by referring to "Excessive CPU, Memory, Bandwidth, or Connection Usage of a DCS Instance."

c. For non-temporary failures, the data source at the application layer needs to be switched back to the original one for processing. This prevents service interruption due to cache faults.

2.11.8 DMS

2.11.8.1 Reliability Functions

Cluster HA

A Kafka instance has multiple replicas for instance-level DR. If the leader replica is faulty, another leader is quickly elected to ensure uninterrupted Kafka services.

In a RabbitMQ cluster, data is replicated to all nodes through mirrored queues, preventing service interruption in case of a node breakdown.

A RocketMQ instance has one master and two slaves. To ensure strong data consistency, the slaves synchronize data from the master. When a node fault occurs, the master/slave roles are automatically switched by using the Raft protocol.

Cross-AZ DR

Kafka, RabbitMQ, and RocketMQ instances can be deployed across AZs. At least three AZs are required. If one AZ is faulty, the instance still provides services.

Monitoring

Kafka: Use Cloud Eye to monitor and report alarms for Kafka instances, instance nodes, instance topics, instance partitions, consumer groups of instance partitions, consumer groups of instance queues, and consumer groups of instances.

For details, see **Kafka Metrics**.

RabbitMQ: Use Cloud Eye to monitor and report alarms for RabbitMQ instances, instance nodes, and instance queues.

For details, see **RabbitMQ Metrics**.

RocketMQ: Use Cloud Eye to monitor and report alarms for RocketMQ instances, instance nodes, instance queues, consumer groups of instances, consumer groups of instance queues, and dead letter queues of instances.

For details, see **DMS for RocketMQ Metrics**.

2.11.8.2 Common Faults

Excessive CPU, Memory, Disk, or Bandwidth Usage

- Check: Use Cloud Eye to check the CPU, memory, disk, or bandwidth usage.

 Copyright © Huawei Technologies Co., Ltd.

- Recovery:
  a. If the CPU or memory usage is high, change the broker specifications or add brokers to expand resources based on service requirements.
  b. If the disk usage is high, scale up the instance storage based on service requirements.
  c. If the bandwidth usage is high, change the specifications to support higher bandwidth based on service requirements.
  d. Enable overload protection for the application layer to keep high-priority services running smoothly.

Message Creation Failure

- Check: Message creation failed.
- Recovery:
  a. If this is a temporary failure, retry at the application layer. For details, see **RES09 Retries After Failures**.
  b. If data still cannot be written after multiple retries, write the data to the local cache. After the service is available, write the data to the instance.
  c. If the connection to an overloaded instance failed, rectify the fault by referring to "Excessive CPU, Memory, or Bandwidth Usage."

2.11.9 RDS

2.11.9.1 Reliability Functions

Cluster HA

RDS supports an HA (primary/standby) architecture, which enables a failover to automatically complete in seconds.

Data Durability

RDS provides 99.9999999% (9 nines) data durability, keeping your data secure and reliable.

Data Backup and Restoration

RDS automatically backs up data every day and stores backups as packages in OBS. The backups can be stored for up to 732 days and can be restored with just a few clicks. You can set a custom backup policy and create manual backups at any time.

You can restore data from backups to any point in time within the backup retention period. In most scenarios, you can use backups to restore data to a new instance or an existing instance at any point in time within 732 days. After the data is verified, data can be migrated back to the primary DB instance.

For details, see **Introduction to Backups**.

 Copyright © Huawei Technologies Co., Ltd.

Storage Autoscaling

With storage autoscaling enabled, when RDS detects that you are running out of database space, it automatically scales up your storage.

For details, see **Configuring Storage Autoscaling**.

Cross-AZ DR

RDS supports cross-AZ high availability. When you buy primary/standby DB instances, you can select different AZs for them.

For details, see **Does RDS Support Cross-AZ High Availability?**

RDS for MySQL also supports cross-AZ active-active DR within a region using DRS, which is similar to cross-region DR.

Cross-Region DR

You can create a DR instance on the DRS console. If a natural disaster occurs in the region hosting the primary DB instance and both the primary and standby DB instances are unavailable, the DR instance is promoted to the primary DB instance. Then, change the connection address on your applications to the address of the DR instance for service recovery. DRS allows you to perform cross-region single-active or dual-active DR between a primary DB instance and a DR instance during disaster recovery.

For details, see **From MySQL to MySQL (Single-Active DR)** and **From MySQL to MySQL (Dual-Active DR)**.

Monitoring

Using Cloud Eye, you can monitor RDS resources such as vCPUs, memory, disks, and networks.

For details, see **Configuring Displayed Metrics**.

2.11.9.2 Common Faults

Excessive CPU, Memory, Disk, or Database Connection Usage or Disk IOPS on an RDS Instance

- Check: Use Cloud Eye to check the CPU, memory, disk, or database connection usage or disk IOPS.
- Recovery:
  a. Change the specifications to expand resources based on service requirements.
  b. Enable storage autoscaling to automatically increase storage capacity when needed.
  c. Enable overload protection for the application layer to keep high-priority services running smoothly.

 Copyright © Huawei Technologies Co., Ltd.

Failed to Connect to a Backend RDS Instance

- Check: Network connection failed.
- Recovery:
  a. If this is a temporary failure, for example, a primary/standby switchover is in progress, connect to the RDS instance again at the application layer. For details, see **RES09 Retries After Failures**.
  b. If the connection to an overloaded RDS instance failed, rectify the fault by referring to "Excessive CPU, Memory, Disk, or Database Connection Usage or Disk IOPS on an RDS Instance."

2.11.10 TaurusDB

2.11.10.1 Reliability Functions

Cluster HA

TaurusDB supports an HA architecture with a primary node and read replicas. If the primary node fails, a read replica is automatically promoted to the primary node to provide services, and the original primary node is demoted to a read replica.

TaurusDB also supports heterogeneous DR instances (MySQL nodes) to ensure HA. If your DB instance is unavailable due to unknown community bugs, misoperations, or AZ faults, the heterogeneous DR instance can take over the services.

Data Backup and Restoration

TaurusDB supports automated and manual backups. You can periodically back up databases. If a database is faulty or data is damaged, you can restore the database using backups to ensure data reliability.

TaurusDB supports same-region and cross-region backups. Cross-region backups mean that if an instance in a region is faulty, you can use the backups in another region to restore data to a new instance.

For details, see **Backup Principles**.

Auto Scaling

TaurusDB supports auto scaling, which can automatically adjust the instance specifications or the number of read replicas.

For details, see **Configuring Auto Scaling Policies for a DB Instance**.

Cross-AZ DR

TaurusDB instances can have nodes deployed in multiple AZs.

TaurusDB also supports cross-AZ active-active DR within a region using DRS, which is similar to cross-region DR.

 Copyright © Huawei Technologies Co., Ltd.

Cross-Region DR

You can create a DR instance on the DRS console. If a natural disaster occurs in the region hosting the primary DB instance and both the primary and standby DB instances are unavailable, the DR instance is promoted to the primary DB instance. Then, change the connection address on your applications to the address of the DR instance for service recovery. DRS allows you to perform cross-region single-active or dual-active DR between a primary DB instance and a DR instance during disaster recovery.

For details, see **From TaurusDB to TaurusDB (Single-Active DR)** and **From TaurusDB to TaurusDB (Dual-Active DR)**.

Monitoring

Using Cloud Eye, you can monitor TaurusDB resources such as vCPUs, memory, disks, and networks.

For details, see **TaurusDB Metrics**.

2.11.10.2 Common Faults

Excessive CPU, Memory, Disk, or Database Connection Usage or Disk IOPS on a TaurusDB Instance

- Check: Use Cloud Eye to check the CPU, memory, disk, or database connection usage or disk IOPS.
- Recovery:
  a. Change the specifications to expand resources based on service requirements.
  b. Enable auto scaling so that the specifications can be automatically upgraded or read replicas can be automatically added when an instance is overloaded.
  c. Enable overload protection for the application layer to keep high-priority services running smoothly.

Failed to Connect to a Backend TaurusDB Instance

- Check: Network connection failed.
- Recovery:
  a. If this is a temporary failure, for example, a primary/standby switchover is in progress, connect to the TaurusDB instance again at the application layer. For details, see **RES09 Retries After Failures**.
  b. If the connection to an overloaded TaurusDB instance failed, rectify the fault by referring to "Excessive CPU, Memory, Disk, or Database Connection Usage or Disk IOPS on a TaurusDB Instance."

2.11.11 OBS

2.11.11.1 Reliability Functions

Data Durability

OBS provides multi-level reliability assurance for storage media, servers, cabinets, data centers, and regions by leveraging the following technologies: slow disk or bad sector detection, intra-AZ device and data redundancy, cross-AZ data disaster recovery, cross-region replication, and more. OBS delivers up to 99.9999999999% (12 nines) of data durability, and up to 99.995% of data availability, far higher than a conventional architecture would offer.

For details, see **What Are the Durability and Availability of OBS?**

Data Backup and Restoration

OBS can store multiple versions of an object. You can quickly search for and restore different versions or restore data in the event of misoperations or application faults.

Cross-AZ DR

When creating a bucket, you can choose multi-AZ storage to store your data in multiple AZs for higher data reliability.

Cross-Region DR

OBS provides you with disaster recovery across regions, catering to your needs for remote backup.

Monitoring

Using Cloud Eye, you can monitor requests, traffic, latency, and error responses of OBS buckets and set alarms for them.

For details, see **Monitoring OBS**.

2.11.11.2 Common Faults

OBS Bucket Traffic Overload

- Check: Use Cloud Eye to check metrics such as the number of requests, request success rate, and upload/download bandwidth.
- Recovery:
  a. Adjust batch services at the application layer to avoid performing operations such as backups during peak hours.
  b. If this is a temporary failure, for example, network congestion, retry at the application layer. For details, see **RES09 Retries After Failures**.
  c. Enable overload protection for the application layer to keep high-priority services running smoothly.

 Copyright © Huawei Technologies Co., Ltd.

OBS Object Upload/Download Failure

- Check: The object upload or download failed.
- Recovery:
  a. If this is a temporary failure, for example, network congestion, retry at the application layer. For details, see **RES09 Retries After Failures**.
  b. If the connection to an overloaded OBS bucket failed, rectify the fault by referring to "OBS Bucket Traffic Overload."

OBS Bucket Data Mistakenly Deleted

- Check: N/A
- Recovery: Enable versioning for OBS buckets. If data is deleted, use the historical version to quickly recover the data.

 Copyright © Huawei Technologies Co., Ltd.

3 Security Pillar

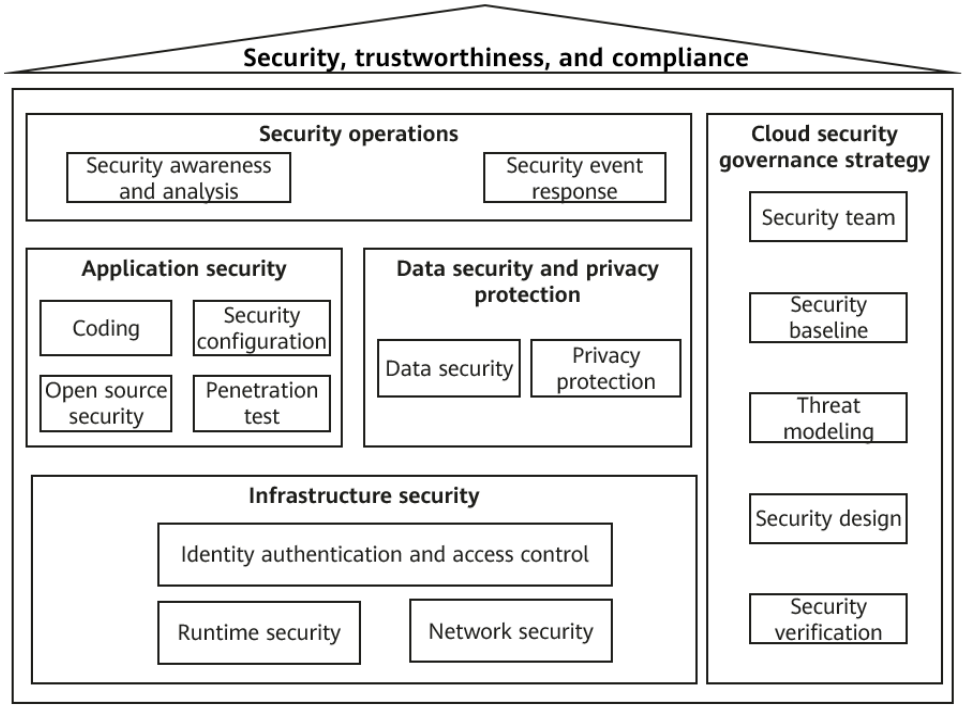
3.1 Overview

3.1.1 Introduction to the Security Pillar

Huawei takes security and privacy protection as our top priorities. The security pillar is dedicated to ensuring the security, trustworthiness, and compliance of services. It protects workloads against security threats and reduces security risks through a series of best practices in the Huawei Cloud architecture. The security pillar provides best practices to protect the confidentiality, integrity, and availability of systems, assets, and data on the cloud. It ensures legal and regulation-compliant use of data, and protects user privacy.

Security is an important aspect of modern applications. We need to systematically consider the security of workloads. The following figure shows the framework of the security pillar.

 Copyright © Huawei Technologies Co., Ltd.



3.1.2 Shared Responsibility Model

Huawei Cloud provides secure, reliable, and trustworthy infrastructure and services based on years of technical and governance capabilities in security, compliance, privacy, and data protection. Huawei Cloud proposes the "Seven layers of defense + One center" cybersecurity framework to ensure the security of cloud services through multiple layers of defense.

Huawei Cloud prioritizes security and compliance above all. You and Huawei Cloud share this responsibility. To define each party's role in protecting cloud security, Huawei Cloud has defined a shared responsibility model for security. Huawei Cloud is responsible for the security of the cloud, and Huawei Cloud customers are responsible for the security on the cloud.

3.2 Concepts

3.2.1 Concept Description

Concept	Description
Confidentiality	A system only allows authorized users to access the sensitive information in the system, preventing unauthorized access and disclosure. Confidentiality is usually implemented using encryption technologies, including data encryption and description. This ensures that only authorized users can access and understand data.

 Copyright © Huawei Technologies Co., Ltd.

Concept	Description
Integrity	Data is not tampered with or damaged during transmission or storage. Integrity is usually implemented by using technologies such as hash functions and digital signatures, to ensure that data is not tampered with or damaged during transmission or storage.
Availability	Systems and data are available whenever needed, not affected by attacks, faults, or other reasons. From the perspective of security, availability can be achieved through load balancing, elastic computing, event monitoring and alarming, and brute-force attack defense, such as anti-DDoS.
Auditability	The ability to effectively monitor, record, and audit systems or data processing. Auditability is usually implemented by using technologies such as audit logs, audit tracing, monitoring systems, and audit tools. System operations and events are recorded for subsequent audit and monitoring.
Non-repudiation	During a communication or transaction, one party cannot deny the message or behavior that has been sent or received. Non-repudiation is usually implemented by using technologies such as digital signatures, public key infrastructure (PKI), audit logs, and audit trails to ensure neither communication party can deny their behaviors or messages.
Account	Your Huawei Cloud resources belong to an account. Resource isolation and billing are also based on this account. It has full access permissions for the resources and cloud services it owns. Each account has independent identity authentication, access control, and resource isolation. By default, accounts are isolated from each other.
IAM	Identity and Access Management (IAM) provides identity authentication and permissions management to help you securely control access to your cloud services and resources.
IAM user	You can use your account to create IAM users and assign permissions for specific resources. Each IAM user has their own identity credentials (password or access keys) and uses cloud resources based on assigned permissions. An account and its IAM users have a parent-child relationship. The account owns the resources and makes payments for the resources used by IAM users. It has full access permissions for these resources. You use an account to create IAM users and grant permissions to them. You can also use this account modify or revoke the IAM users' permissions at any time.

 Copyright © Huawei Technologies Co., Ltd.

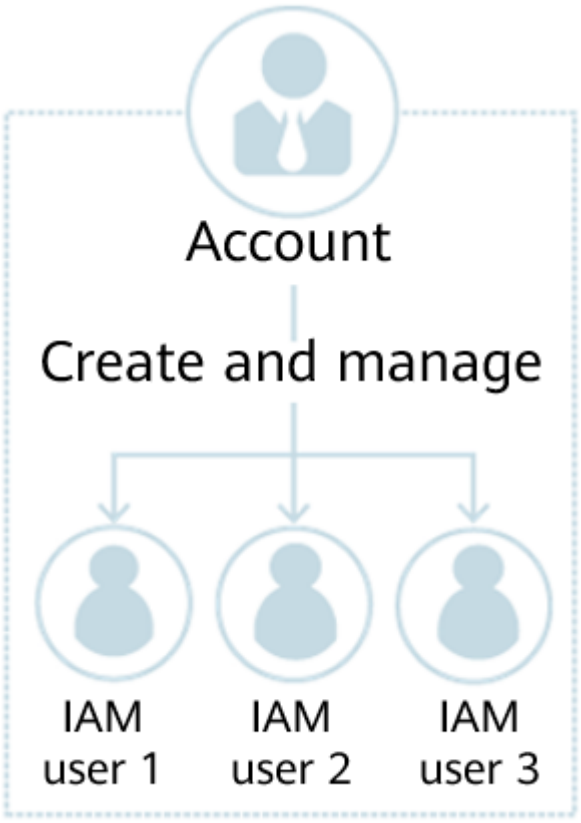
Concept	Description
MFA	Multi-factor authentication (MFA) is a security authentication process that requires users to provide two or more types of authentication factors to authenticate their identities. The factors include passwords, fingerprints, SMS verification codes, smart cards, and biometric recognition, improving user account security.
Security threat	Security threats refer to the risk factors or events that may cause the system, network, or data to be damaged or accessed. Security threats can be accidental or intentional and may cause a system to be attacked or damaged. Security threats can be external (such as hacker attacks and malware) or internal (such as employee negligence and internal leakage).
Threat modeling	Identify potential threats to a system to develop protection policies and build a secure system.
Security risk	The probability and impact of possible damage or loss brought by security threats to a system, network, or data in terms of confidentiality, integrity, or availability. Security risks are usually caused by threats, system vulnerabilities, improper security measures, or other factors. Security risks are usually assessed and managed through risk assessment, which determines the risk level and take corresponding control measures.
Playbook	In security response, a playbook is a predefined operation guide designed to help security teams take actions in a quick and orderly manner in the face of specific security events or threats. A playbook usually contains detailed steps, processes, tools, and division of labor to ensure security incidents can be effectively handled and potential losses and impacts can be reduced.
Data subject	A natural person who provides personal data and can be identified based on personal data or a combination of personal data. Natural persons have the right to complain or ask questions about how their personal data is processed. Example: the end users of a product, or the employees of a company.
Data controller	A natural or legal person, public authority, agency, or any other body that, alone or jointly with others, determines the purposes and means of personal data processing. They have control over the processing of personal data and assume the primary responsibility for personal data protection.
Data processor	A natural or legal person, public authority, agency, or any other body that processes personal data on behalf of a data controller. A data processor must provide adequate protection following the data controller's requirements.

 Copyright © Huawei Technologies Co., Ltd.

Concept	Description
Third party (in privacy protection)	Any natural person, legal person, public authority, agency, or other body other than the data subject, data controller, data processor, and any person directly authorized by the data controller or processor to process data.
Personal data	Any data relating to an identified or identifiable natural person, known as a data subject. Natural persons are identifiable if they can be directly or indirectly identified, by such information as a name, an identification number, location data, or one or more special characteristics specific to the physical, physiological, mental, economic, cultural, or social identity, and other relevant information of these natural persons.
Sensitive personal data	A subset of high-impact personal data. It refers to the personal data that is way closely related to an individual's fundamental rights and freedom and, if disclosed, may result in personal injury, financial losses, reputation damage, identity theft or fraud, or discrimination.

3.2.2 Conceptual Models

After you sign up for a Huawei Cloud account, you can create multiple IAM users under the account.



 Copyright © Huawei Technologies Co., Ltd.

Large enterprises usually need to manage multiple accounts in a unified manner. In this case, they can use an enterprise master account to manage multiple enterprise member accounts in a unified manner.

Both an account and its IAM user can create IAM users to manage resources. The Huawei Cloud Enterprise Center allows multiple independent HUAWEI IDs to be associated with each other.

3.3 Design Principles

According to International Organization for Standardization (ISO), the security of a computer system refers to the protection on its information assets (including hardware, software, network, and data) ensure confidentiality, integrity, and availability. The objectives of computer system security are to protect information systems from various threats, including unauthorized access, use, disclosure, destruction, modification, disruption, and unavailability, while ensuring that systems can continue to deliver services.

The basic elements of system security include confidentiality, integrity, availability, auditability, and non-repudiation. Confidentiality, integrity and availability are the three basic elements and are called the "CIA triad" for short.

To implement the basic elements defined for system security, the following common security design principles are extracted based on a large number of practices in the industry:

- **Zero Trust**
  - Zero Trust follows the security concept of "never trust, always verify." It assumes that no one or program is trustworthy, no matter whether they are internal users, external users, or network devices. To reduce the attack risks, the components in a system must be explicitly verified before any communication. Zero Trust transforms the existing static trust model based on authentication and default authorization into a dynamic trust model based on continuous risk evaluation and adaptive authorization.
  - Zero Trust does not determine the credibility based on the network location. It focuses on protecting resources, not CIDR blocks. Compared with the traditional security concepts, Zero Trust shifts the focus of network defense from static network-based borders to users, devices, and resources. All resources (such as people, objects, devices, applications, networks, data, and supply chains) require continuous identity authentication and trust evaluation, and dynamic security policies must be applied globally. Zero Trust reduces the attack surface and ensures system security through dynamic and continuous risk evaluation.
- **Defense in depth**
  - Multi-point and multi-layer security mechanisms are used to protect organization networks, assets, and resources.
  - It does not fully depend on the security capabilities of a single layer. The protection failure on a single layer will not lead to complete exposure to attacks.
  - If a system is attacked, it will still have certain resilience capabilities to ensure minimum system running and provide minimum services.

 Copyright © Huawei Technologies Co., Ltd.

- **Principle of least privilege (PoLP)**
  - Least roles: Delete unnecessary system administrators. Periodically check for and delete expired roles.
  - Least privileges: Grant users or entities the least privileges required to perform their tasks to reduce security risks.
  - Minimum exposure: Expose the least service endpoints and service application interfaces required for different access regions and objects.
  - Least credentials: Eliminate the dependency on long-term, static credentials.
- **Data security**
  - Classify and grade data, and define data protection measures at different levels.
  - Ensure proper encryption, backup, and access control of data to protect data confidentiality, integrity, and availability.
  - Protect privacy. Ensure the confidentiality and integrity of private data.
- **DevSecOps**
  - The core concept of DevSecOps is to incorporate security into the entire software development lifecycle. Security is considered in all the phases, including requirement analysis, design, development, test, deployment, O&M, and operation, to ensure system security and stability.
  - DevSecOps combines security with the automated process of DevOps to detect and fix security vulnerabilities more quickly, improving the efficiency and quality of software development.

3.4 Questions and Checklists

Question	Checklist/Best Practice
SEC01 How do we develop the overall cloud security governance policies?	1. Set up a security management team. 2. Establish a security baseline. 3. Sort assets and compile a list. 4. Separate workloads. 5. Perform threat modeling analysis. 6. Identify and verify security measures.
SEC02 How do we manage the identity authentication of human-machine interfaces and machine-machine interfaces?	1. Account protection 2. Secure login mechanism 3. Security management and use of credentials 4. Integrated identity management
SEC03 How do we manage the permissions of human and machine?	1. Define permission access requirements. 2. Assign appropriate permissions as needed. 3. Regularly review the permissions. 4. Securely share resources.

 Copyright © Huawei Technologies Co., Ltd.

Question	Checklist/Best Practice
SEC04 How do we design network security?	1. Divide a network into zones. 2. Control network traffic access. 3. Minimize network access permissions.
SEC05 How do we design the operating environment security?	1. Cloud service security configuration 2. Vulnerability management 3. Reduce the attack surface. 4. Key security management 5. Certificate security management 6. Use managed cloud services.
SEC06 How do we design application security?	1. Use open-source software in a secure and regulation-compliant manner. 2. Establish secure coding specifications. 3. Implement white-box code review. 4. Apply secure application configuration. 5. Perform penetration testing.
SEC07 How do we design data security?	1. Identify data within workloads. 2. Data protection control 3. Data operation monitoring 4. Static data encryption 5. Transmitted data encryption
SEC08 How do we design data privacy protection?	1. Specify privacy protection policies and principles. 2. Proactively notify data subjects. 3. Obtain data subjects' choice and consent. 4. Ensure data collection compliance. 5. Ensure data use, retention, and disposal compliance. 6. Ensure the compliance of personal data disclosure to third parties. 7. Data subjects have the right to access their privacy data.
SEC09 How do we implement security awareness and threat detection?	1. Manage logs in a standardized manner. 2. Security event recording and analysis 3. Perform security audit. 4. Security posture awareness

 Copyright © Huawei Technologies Co., Ltd.

Question	Checklist/Best Practice
SEC10 How do we respond to security incidents?	1. Establish a security response team. 2. Develop an incident response plan. 3. Automatically respond to security incidents. 4. Perform security incident drills. 5. Establish a review mechanism.

3.5 Cloud Security Governance Policies

3.5.1 SEC01 Cloud Security Governance Policies

The use of cloud services will not change the ultimate security goal of an enterprise, but will change the way the goal is achieved. To securely perform operations on and manage your workloads, you need to consider all aspects of security. The management team and security team of an enterprise need to formulate a cloud security strategy based on its overall security strategy and business strategy, and consider security as early as possible while planning the use of cloud services. Cloud security governance policies involve security teams, security baselines, security assets, security modeling, and core security control points. You need to plan as early as possible how to use cloud technologies and services to modernize security governance, and implement proper cloud security policies to ensure the security and compliance of service systems on the cloud.

3.5.1.1 SEC01-01 Establishing a Security Management Team

Specify the key roles responsible for workload security, compliance, and privacy protection in the cloud environment.

- **Risk level**

  High

- **Key strategies**
  - Clarify responsibilities and roles: Determine the responsibilities and roles of team members, including the roles in security architecture design, security test, and security operation. Clearly define the responsibilities and tasks of each role.
  - Cross-functional team: Set up a cross-functional security management team that consists of professionals from different domains, such as security operation, security architecture, and compliance, to ensure comprehensive security management.
  - Develop security policies and processes: Develop detailed security policies and processes, and specify security management standards and regulations. Team members should comply with these policies and processes to ensure consistency and effectiveness of security management.
  - Establish an emergency response plan: Develop and validate an emergency response plan to respond to security incidents and

 Copyright © Huawei Technologies Co., Ltd.

emergencies. The team should have a clear understanding of how to deal with security threats and handle security incidents.

3.5.1.2 SEC01-02 Establishing a Security Baseline

Establish security baselines that comply with compliance requirements, industry standards, and platform suggestions. Security baselines are the minimum security requirements for within a team. Periodically measure and improve the security of your workloads by checking them against the security baselines.

- **Risk level**

  High

- **Key strategies**

  Determine compliance requirements: Understand the organizational, legal, and compliance requirements that your workloads must comply with.

- **Related cloud services and tools**
  - **Huawei Cloud Compliance Center**
  - **Huawei Cloud Trust Center**
  - Huawei Cloud DJCP Multi-level Protection Scheme (MLPS) compliance solution: Based on our own security capabilities and the security and compliance ecosystem, Huawei Cloud provides a one-stop security solution that helps to quickly complete security rectification at a low cost and easily meet DJCP MLPS compliance requirements. Based on Huawei's 30 years of security experience, we help enterprises and organizations achieve their security and compliance goals, comply with national and industry laws and regulations, effectively monitor security risks and security incidents, and take effective measures in a timely manner to continuously reduce security risks and eliminate the losses caused by security incidents.

3.5.1.3 SEC01-03 Compiling an Asset List

Sort all the cloud resources involved in workloads, such as servers, IP addresses, domain names, databases, and certificates, and label the resources. In this way, you can quickly locate resources with security risks if a security incident occurs.

- **Risk level**

  High

- **Key strategies**
  - Consistency between the development state and runtime state: Implement cloud service resources based on the architecture design and documents. The architecture of the running workload should always be the same as designed.
  - Automatic asset stocktaking: Use security cloud services or tools to automatically discover and record cloud resources, including servers, storage, databases, and networks. This ensures the asset list is up-to-date and accurate.
  - Tags and metadata: Use tags and metadata to classify and describe cloud resources to better organize and manage resource lists. Tags can be used to quickly identify and filter resources, facilitating monitoring and security audit.

 Copyright © Huawei Technologies Co., Ltd.

- **Related cloud services and tools**
  - InnoStage Workbench: This solution workbench can assist in the visualized design of the cloud architecture.
  - SecMaster: SecMaster can automatically count assets on the cloud, flexibly manage diverse assets outside the cloud, and display the real-time security status of assets.
  - Config
  - Tag Management Service (TMS)

3.5.1.4 SEC01-04 Separating Workloads

Workloads are separated in the architecture design. The workloads of a system are divided into smaller parts, and each part runs and is managed independently, improving system security and maintainability.

- **Risk level**

  High

- **Key strategies**

  An enterprise, especially a large one, usually has workloads of different types (such as those in the production environment, development environment, and test environment) or different organizational units (OUs). Multiple OUs or workloads must be isolated.

  Workload separation is important in a cloud environment. From the perspective of security governance, the reasons are as follows:

  - Security: Workload separation can reduce security risks. By isolating different workloads in independent environments, you can reduce the impact of a faulty or attacked workload on other workloads.
  - Compliance: Some industries and regulations have strict requirements on data isolation and access control. By separating workloads, you can more easily meet compliance requirements, protect sensitive data, and ensure data privacy.
  - Manageability: Workloads are separated to facilitate system management and maintenance. Each workload has independent configuration and management requirements. Separating them can simplify the management process and reduce operation risks.
  - Flexibility: Separating workloads provides greater flexibility and scalability. Organizations can adjust and scale resources for different workloads as needed without affecting other parts.

  Huawei Cloud provides the following workload separation mechanisms:

  - Separation by VPC: Workloads are deployed in different VPCs. Each VPC has an independent network separated from others.
  - Separation by enterprise project: An enterprise project is a logical collection of cloud service resources. Workloads can be deployed in different enterprise projects for group-based resource management and permission control.
  - Separation by account: Deploy different workloads in different Huawei Cloud accounts. Each account has independent identity authentication, access control, and resource isolation. In this way, you can achieve stricter isolation and security. Grant minimum necessary permissions to each

 Copyright © Huawei Technologies Co., Ltd.

account to avoid excessive permission assignment. This can reduce security risks and permission abuse. If cross-account access is required, use appropriate identity authentication and authorization mechanisms, such as cross-account delegation and resource sharing.

- Combination of multiple workloads: Use two or more of the preceding methods to separate workloads.

- **Related cloud services and tools**
  - Virtual Private Cloud (VPC)
  - Enterprise Project Management Service (EPS)
  - Identity and Access Management (IAM)
  - Huawei Cloud Landing Zone
  - Organizations
  - Resource Governance Center (RGC)
  - Resource Access Manager (RAM)

3.5.1.5 SEC01-05 Performing Threat Modeling Analysis

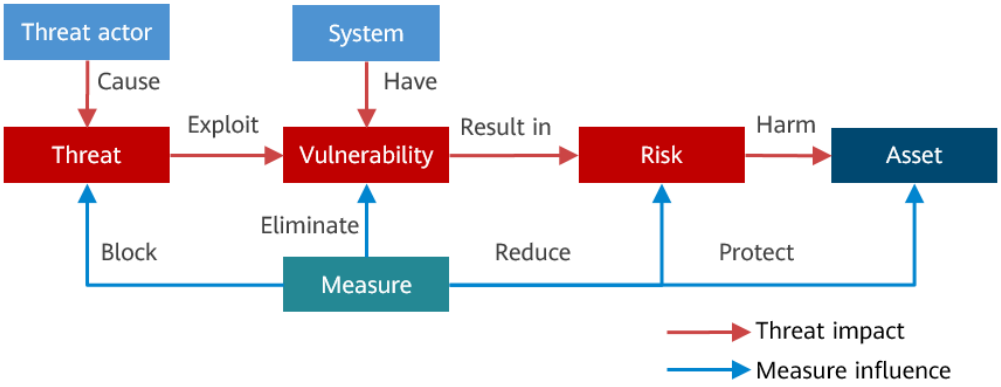
Threat modeling helps to systematically identify and evaluate potential threat sources, attack vectors, and attack methods that may pose threats to systems or organizations. In this way, you can understand system security risks, identify security issues in system design, and develop mitigation measures to reduce the risks and to improve system security and resilience.

- **Risk level**

  High

- **Key strategies**

  The following figure shows the model for analyzing the threats during system running.



  The concepts involved in this model are as follows:

  - Threat actor: An entity that intentionally exploits vulnerabilities. A threat actor can be a person, program, hardware, or system.
  - Vulnerability: A flaw in a system that allows its security to be compromised, including software, hardware, processes, or human-made defects. The existence of vulnerability indicates the lack of security measures or the defects in security measures.
  - Threat: Any potential hazards caused by exploiting vulnerabilities.

 Copyright © Huawei Technologies Co., Ltd.

- Risk: The possibility of vulnerability exploit by attackers and its business impact. Risks relate vulnerabilities, threats, and the possibility of exploits to the business impact.
- Asset: Any information or resource that is valuable to an organization. It should be protected by security policies.
- Measure: Measures include the mitigation measures for security purposes and the hardening measures for resilience purposes. They help to eliminate vulnerabilities, prevent threats, reduce the impact of risks, and protect assets.

To implement threat modeling, you need to think like an attacker to discover potential exposed surfaces, attack targets, and available attack methods, so as to discover potential security threats in the system and develop corresponding mitigation measures.

The general procedure for threat modeling is as follows:

a. Determine the scope: Specify the cloud system scope for threat modeling, including cloud services, data storage, and network architecture.

b. Collect information: Collect information about the cloud system, including the system architecture diagram, data flow, and access control policies.

c. Identify assets: Determine the critical assets in the cloud system, including data, applications, VMs, and storage.

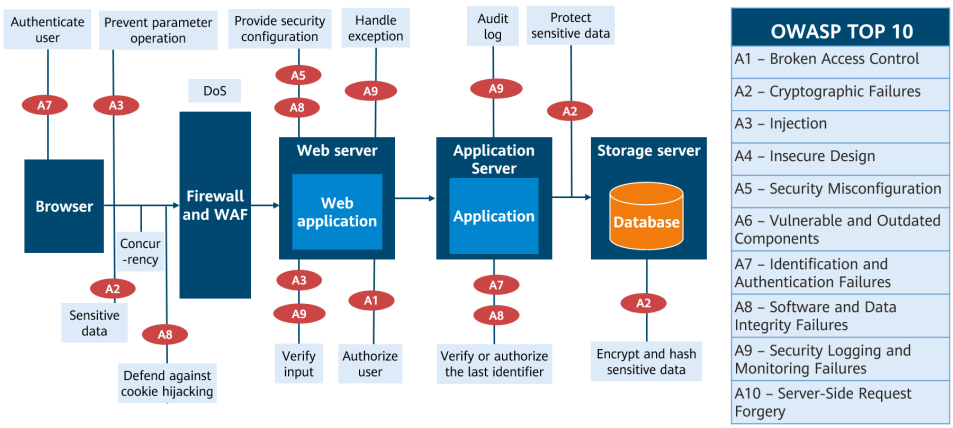
d. Identify threat sources and attack vectors: Determine the threat sources and attack paths that may pose threats to the cloud system, and consider the attack methods that different attackers may take.

e. Assess threat probability and impact: Assess the probability and possible impact of each threat, including data leakage and service interruption.

f. Develop security countermeasures: Develop security countermeasures and control measures based on identified threats, including access control, encryption, and monitoring.

g. Continuous improvement: Periodically review and update threat models to reflect new threats and security risks, continuous improving cloud system security.

The following table lists the top 10 web application system threats summarized by OWASP and their countermeasures.



- **Related cloud services and tools**

 Copyright © Huawei Technologies Co., Ltd.

3.5.1.6 SEC01-06 Identifying and Validating Security Measures

Validate the security measures involved in the workloads based on the security baselines developed by the team and the results of the threat modeling analysis. Ensure the measures work as expected and effectively protect the system, mitigating or eliminating security threats.

- **Risk level**

  High

- **Key strategies**
  - Based on the security design documents of the system, verify that security measures are correctly integrated into the system and comply with best practices and standards.
  - Review the system code as early as possible. (This process is called white-box security review.) Ensure the code complies with security best practices so that major vulnerabilities will not be left unresolved to subsequent phases.
  - Use security test tools to perform tests such as static code analysis, dynamic code analysis, and vulnerability scan to detect potential security issues.
  - Use attack simulation tools or techniques to simulate attack behaviors and analyze the security and weaknesses of the system.

3.6 Infrastructure Security

3.6.1 SEC02 Identity Authentication

3.6.1.1 SEC02-01 Account Protection

An account, which is the highest-privileged user in the Huawei Cloud tenant account system, has the highest permissions on the entire cloud environment. If an account is attacked or leaked, serious security problems and data leakage may occur. Therefore, it is vital to protect the account for the security of identity authentication.

- **Risk level**

  High

- **Key strategies**
  - Strong passwords: Use strong passwords to protect accounts. The password should be long and complex enough, and consist of digits, letters, and special characters.
  - MFA: Enable MFA to enhance account security. MFA requires additional identity authentication beyond just passwords, improving account security.
  - Limited routine operations: Create an IAM user and use it for routine management, rather than using accounts directly. Accounts should be used only for key operations, such as creating IAM users or modifying permissions.

 Copyright © Huawei Technologies Co., Ltd.

  - Using temporary credentials and rotating them periodically: Periodically change the password of the account and update the MFA device to reduce the risk of the credentials being guessed or embezzled.
  - Enabling audit logs: Enable audit logs to monitor account activities. This can help you detect abnormal behaviors and take measures in a timely manner.
  - Multi-account management: You need to specify an account as the main account (enterprise master account), and use this account to add member accounts (enterprise member accounts). The security of main account is prior to that of member accounts.
- **Related cloud services and tools**
  - IAM
  - Organizations
  - **Enterprise Management**
  - CTS

3.6.1.2 SEC02-02 Secure Login Mechanism

Use secure login mechanisms for accounts, IAM users, and third-party identity providers.

- **Risk level**

  High
- **Key strategies**
  - Enable MFA-based login for accounts and IAM administrators (IAM users with administrator permissions) to prevent risks caused by login credential leakage.
  - Configure IAM login verification policies, such as session timeout, account lockout, account suspension, and last login notification.
  - Configure the network ACL policy of IAM. Users can access Huawei Cloud only from specific IP address ranges, CIDR blocks, and VPC endpoints.
  - Use different passwords for different accounts or IAM users.
  - Do not share your passwords with others. Instead, create a user for each person who manages or uses Huawei Cloud resources.
  - Change the default password of the new user. When you create a user using IAM, a one-time login link can be sent to the new user via email. The created user needs to set a password when logging in through the link. Additionally, when the administrator customizes the password for the new user, password modification upon user activation can be set to mandatory.
  - Centralized identity control:
    - Single sign-on (SSO): Use the SSO solution to centrally manage user identity authentication information, simplify the user login process, enhance security, and improve user experience.
    - Multi-account: Centrally manage the accounts.
- **Related cloud services and tools**

 Copyright © Huawei Technologies Co., Ltd.

  - **IAM Identity Providers**
  - **Best Practices for Using Huawei Accounts**
  - OneAccess: Use OneAccess to associate with your organization's HR system for SSO.

3.6.1.3 SEC02-03 Security Management and Credential Usage

Use temporary credentials instead of long-term or permanent credentials for identity authentication to reduce or eliminate risks caused by credential leakage, sharing, or theft.

- **Risk level**

  High

- **Key strategies**
  - There are two types of credentials — long-term credentials include the login password of the user and permanent AK/SK, and short-term credentials include temporary AK/SK and permissions obtained through agency. Do not hard-code long-term credentials in code to prevent leakage. Use temporary credentials to call Huawei Cloud SDKs or APIs.
  - In certain scenarios where temporary credentials cannot be used, only then shall the long-term credentials be used. In this case, you should store the long-term credentials in a file outside the code or have them hosted by a third party, and use them as variables. Moreover, periodically audit and rotate your credentials to reduce risks.
  - Audit your identity providers and identities configured in IAM to ensure that only authorized identities can access your workloads.
  - Use Data Encryption Workshop (DEW) to host credentials. DEW allows you to centrally manage, retrieve, and securely store various credentials, including database account passwords, server passwords, SSH keys, and AKs.
  - Use Cloud Secret Management Service (CSMS) in DEW to periodically rotate credentials.
  - Use IAM agency to grant operation permissions to cloud services or other accounts.
- **Related cloud services and tools**
  - DEW
  - IAM

3.6.1.4 SEC02-04 Integrated Identity Management

Create a unified identity management system across the company to centrally manage the private cloud, public cloud, and the accounts on the public cloud.

- **Risk level**

  Medium

- **Key strategies**
  - Create a unified identity management system across the company to centrally store user identity information.

 Copyright © Huawei Technologies Co., Ltd.

  - The system supports identity federation between private and public cloud IAM systems, enabling seamless cross-cloud access for users.
  - The system is integrated with the company's HR process. User creation, change, and deletion can be triggered when an employee joins, change positions, or leaves the company.
  - With IAM Identity Center in a multi-account environment built on Landing Zone, you can centrally manage user identities and their access to resources, eliminating per-account IAM user creation and simplifying identity permission management.
  - The system establishes identity federation with IAM Identity Center, eliminating the need for separate federation with each account's IAM system.
- **Related cloud services and tools**
  - IAM Identity Center
  - IAM
  - OneAccess

3.6.2 SEC03 Permission Management

3.6.2.1 SEC03-01 Defining Access Control Requirements

Define which personnel or systems are authorized to access specific components, and select the appropriate identity type and method for authentication and authorization.

- **Risk level**

  High
- **Key strategies**

  Utilize Identity and Access Management (IAM) roles to define the access permissions of applications and components to resources. Implement a principle of least privilege access control model, ensuring that only essential permissions are granted. Assign permissions based on user roles and responsibilities, guaranteeing that users can access only the resources necessary for their work.
- **Related cloud services and tools**

  IAM

3.6.2.2 SEC03-02 Assigning Appropriate Permissions on Demand

Permissions management must adhere to the principles of on-demand allocation, least privilege, and separation of duties (SoD). Users should be assigned access permissions to critical service systems strictly based on their job responsibilities, preventing unauthorized access to sensitive systems and data. If temporary permissions are required, grant only the specific permissions necessary for performing designated tasks within a limited timeframe. Once the tasks are completed, these temporary permissions must be revoked.

- **Risk level**

  High

 Copyright © Huawei Technologies Co., Ltd.

- **Key strategies**
  - Organize user groups based on IT functions and add users to the corresponding groups. An IAM user group is a collection of IAM users. User groups let you specify permissions for multiple users.
  - It is recommended to grant permissions to user groups rather than individual users.
  - The "admin" group is the system's default administrator group, possessing operational permissions across all cloud service resources. Do not add all users to the "admin" group.
  - Assign only the minimum necessary permissions to user groups. For instance, some user groups may only access specific cloud services, or have read-only permissions on cloud service resources.
  - Do not include the *:* management permissions in custom IAM policies.
  - If enterprise projects are utilized, prioritize assigning permissions to user groups within those enterprise projects. To grant permissions for all resources across all regions or specific regions under an account, consider using IAM projects to streamline the authorization process.
- **Related cloud services and tools**
  - Identity and Access Management (IAM)
  - Enterprise Project Management Service (EPS)
  - Cloud Bastion Host (CBH): Use CBH to restrict and control the use and access of O&M accounts. CBH provides centralized management and fine-grained access control over O&M account permissions to systems and resources.
  - Organizations: In multi-account scenarios, leverage Service Control Policies (SCPs) within Organizations. The organization management account can use SCPs to define permission boundaries for member accounts, ensuring adherence to the organization's access control guidelines. SCPs can be associated with organizations, organizational units (OUs), and individual member accounts, impacting all accounts within their scope.

3.6.2.3 SEC03-03 Regularly Reviewing Permissions

Periodically review and update permissions to prevent permission creep and consistently remove unnecessary permissions.

- **Risk level**

  High
- **Key strategies**
  - Use IAM user groups to manage access permissions, and configure expiration times for temporary permissions.
  - Promptly adjust the permissions granted to a user group when its job functions or responsibilities change.
  - When delegating an account to another account, ensure an expiration time is configured for the delegation.
  - Monitor the last login time of IAM users to identify accounts that have been inactive for an extended period. For such accounts, manage their identity credentials and permissions in a timely manner.

- **Related cloud services and tools**

  IAM

3.6.2.4 SEC03-04 Securely Sharing Resources

In large enterprises, resources often need to be securely shared across various organizational units, departments, and teams.

- **Risk level**

  Medium

- **Key strategies**
  - Large enterprises typically comprise multiple organizational units and accounts, necessitating resource sharing across these accounts. Secure resource sharing must adhere to the following practices:
  - Use resource tags: Classify and label resources using tags to facilitate efficient management and policy application.
  - Share resources only with trusted entities: Employ Service Control Policies (SCPs) to restrict account permissions within an organization, ensuring that resources are shared exclusively within the defined organizational boundaries.
  - Establish specific service accounts for accessing shared resources.
- **Related cloud services and tools**
  - Organizations
  - Resource Access Manager (RAM): Provides secure cross-account resource sharing capabilities. If you manage multiple Huawei Cloud accounts, you can create resources once in a central account and use RAM to share these resources with other accounts, eliminating the need for redundant resource creation in each account.

3.6.3 SEC04 Network Security

3.6.3.1 SEC04-01 Partitioning the Network

Network partitioning splits a network into separate areas to isolate sensitive traffic and resources, enhancing overall security.

- **Risk level**

  High

- **Key strategies**

  Network partitioning can:
  - Isolate sensitive data: Sensitive data and applications are isolated in independent network partitions to reduce the risk of unauthorized access.
  - Provide scalability: Partitioning and layering help manage and scale complex network architectures, making maintenance and expansion easy.
  - Limit network traffic: Controlling communication traffic between different network partitions ensures that only authorized traffic can flow.

 Copyright © Huawei Technologies Co., Ltd.

  - Improve performance and availability: Network partitioning helps optimize network performance and availability, and avoid network congestion and single points of failure (SPOFs).

  Define the border of each partition and allocate addresses to each network partition for easy management and control. For example, for a web workload, you can create a web area, an app area, and a data area. The most important border lies between the public network (Internet) and your internal applications. This border is the first line of defense for your workloads. Huawei Cloud VPCs and subnets can be used to establish borders for each network partition.

  - VPC planning: Specify a proper CIDR range for a VPC to determine its IP address range.
  - Subnet planning: Create multiple subnets in a VPC and deploy different resources in different subnets.

- **Related cloud services and tools**

  Virtual Private Cloud (VPC)

3.6.3.2 SEC04-02 Controlling Network Traffic Access

Controlling network traffic helps ensure that traffic between network partitions is predictable and allowed. Based on the zero-trust principle, all incoming and outgoing traffic must be verified at the network level. Ensure that the capacity of network devices and bandwidth for each network area meets service requirements during peak hours.

- **Risk level**

  High

- **Key strategies**
  - When designing the network topology, check the connection requirements of each component. For example, you can check whether Internet accessibility (inbound and outbound), VPC connection, edge services, and external data centers are required. Do not place resources in a public subnet of a VPC unless the resources have to receive traffic from the Internet.
  - Use an in-depth defense method for inbound and outbound traffic. For example, all inbound traffic must receive intrusion detection to block malicious cyber attacks. NAT gateways can be used to establish unidirectional connections for outbound traffic.
  - Filter traffic. Firewalls and ACLs can be used to control the access traffic between internal and external networks, and to control the incoming and outgoing traffic of sensitive areas on internal networks. All network traffic must be checked to block traffic that does not comply with security standards in use. This safeguards system components against unauthorized access attempts originating from untrusted networks.
  - When using application load balancers, use secure certificates for layer-7 load balancers.
  - Enable VPC flow logs. VPC flow logs record information about traffic going to and coming from VPCs. You can use flow logs to monitor network traffic, analyze network attacks, and determine whether security groups and firewall rules need to be changed.

For more details about security logs, see **SEC09-01 Implementing Standardized Log Management**.

- **Related cloud services and tools**
  - Virtual Private Cloud (VPC) and VPC Endpoint (VPCEP)
  - Enterprise Router (ER)
  - Cloud Connect
  - Cloud Firewall (CFW) protects Internet and Virtual Private Cloud (VPC) borders on the cloud by real-time intrusion detection and prevention, global unified access control, full traffic analysis and visualization, log audit, and tracing. CFW employs AI for intelligent defense, and can be elastically scaled to meet changing business needs, helping you easily handle security threats. CFW is a basic service that enhances network security on the cloud.
  - Web Application Firewall (WAF) protects web applications, such as websites, from common web attacks, helping keep customer workloads stable, secure, and compliant with applicable laws and regulations.
  - Anti-DDoS Service (AAD) mitigates DDoS attacks in milliseconds to ensure continuity of your global services based on machine learning, protection policy tuning, and precise identification of DDoS attacks. Anti-DDoS is helpful to improve bandwidth utilization. Anti-DDoS protects EIPs against layer 4 to layer 7 DDoS attacks and notifies of alarms in real time. It helps improve bandwidth utilization and keep customer workloads stable.
  - NAT Gateway (NAT): An NAT gateway sits between the Internet and your VPC. It hides the IP addresses of internal networks, reducing the risk of attacks on virtual environments.
  - Elastic Load Balance (ELB) distributes traffic across multiple backend nodes.

3.6.3.3 SEC02-03 Minimizing Network Access Permissions

Ensure that only necessary personnel or components can access specific network resources.

- **Risk level**

  High

- **Key strategies**
  - Configure security groups and network ACLs to control incoming and outgoing traffic for cloud resources. Make sure only authorized traffic can access specific services and ports. Optimize the ACL of each network area based on service requirements and reduce the number of access control rules as much as possible.
  - Restrict public IP address exposure to the minimum required scope and disable external access to open high-risk ports and remote management ports. If it is not possible to disable all high-risk ports or remote management ports, open as few ports as possible.
  - Only open the network segments and ports required by services in the security group. Do not allow all IP addresses (0.0.0.0/0) to access the security group.

 Copyright © Huawei Technologies Co., Ltd.

- **Related cloud services and tools**
  - Virtual Private Cloud (VPC)
  - NAT Gateway
  - SecMaster: Cloud service baseline inspection

3.6.4 SEC05 Runtime Environment Security

3.6.4.1 SEC05-01 Cloud Service Security Configuration

Security configuration is the minimal security requirement for information systems. For cloud services, it provides essential protection and establishes the baseline for security operations. Proper configuration of cloud services can help you prevent security vulnerabilities and data leakage, improving overall system security. Failure to meet security configuration baseline requirements can lead to significant risks.

- **Risk level**

  High
- **Key strategies**
  - Configure security policies for services based on *Huawei Cloud Security Configuration Guide*:
    - Container security, for example, container security configuration. Insecure container configurations in CCE may cause container escape.
    - System vulnerabilities, for example, whether the OS version is the latest version and whether the used version has vulnerabilities.
    - Open necessary ports. For example, check whether high-risk ports such as 22 and 3306 are open to the public network.
    - Do not set the OBS bucket where important service data is stored as a public or publicly readable bucket.
  - Periodically check cloud service security configurations.
    - Comprehensive check: Ensure that the baseline check covers all key cloud service configuration items, including identity authentication, access control, and network security.
    - Periodic and real-time check: Set a periodic automatic check plan and provide the real-time check function to evaluate the security status of cloud services immediately when necessary.
    - Risk assessment: Evaluate risks based on the check results, to be specific, identify resources of different levels, such as critical, high, medium, low, and warning.
- **Related cloud services and tools**
  - **Security Features**: How to ensure cloud security has become the top concern for most enterprises and customers. Huawei Cloud is committed to ensuring the security of IaaS, PaaS, and SaaS cloud services and infrastructure, and providing advanced, stable, reliable, and secure

 Copyright © Huawei Technologies Co., Ltd.

products and services. This document describes how to configure Huawei Cloud services to meet your security objectives.

- Config
- SecMaster: Check the cloud service security configuration baseline using SecMaster to continuously protect customers' cloud service security.
- Host Security Service (HSS): The latest version of HSS supports features of host security and container security, which is formerly known as Container Guard Service (CGS).

3.6.4.2 SEC05-02 Vulnerability Management

With vulnerability management, you can detect and rectify security vulnerabilities in the system in a timely manner, avoiding potential security threats and attacks. Others may illegally obtain system access privileges through security vulnerabilities. You should obtain the latest security intelligence through trusted channels.

- **Risk level**

  High

- **Key strategies**

  You can install patches to fix security vulnerabilities in a timely manner to prevent malicious individuals or software from damaging service systems and data. Stay informed of the latest security advisories of Huawei Cloud and the industry, and implement mitigation measures to ensure workload security.

  - Stay informed of the latest security suggestions of Huawei Cloud and the industry.

    For details about the latest security information, see **Public Notices**.
  - Vulnerability scanning and identification: Huawei Cloud services are used to periodically scan systems and applications to detect potential vulnerabilities and security risks.
  - Automatic vulnerability scanning: Use automatic vulnerability scanning tools to periodically scan the runtime environment to detect potential vulnerabilities and security risks.
  - Vulnerability fixing and patch management: Develop a vulnerability fixing plan, fix confirmed vulnerabilities in a timely manner, and manage the release and application of security patches.
  - Detection and removal of malicious code at key nodes: Malicious code must be checked and removed at key network nodes, and the upgrade and update of the malicious code protection mechanism must be maintained.

- **Related cloud services and tools**
  - HSS
  - SecMaster
  - CodeArts Inspector

3.6.4.3 SEC05-03 Reducing Attack Surfaces of Resources

Enhance the OS, reduce unused components and external services, and use tools to improve cloud security and mitigate the attacks on resources.

 Copyright © Huawei Technologies Co., Ltd.

- **Risk level**

  High

- **Key strategies**
  - Enhancing OS and reducing components: Reduce unused components, libraries, and external services to reduce risks caused by accidental access. This includes OS packages, applications, and external software modules in the code.
  - Create secure VM images or container images.
  - Use third-party tools for security analysis: Use third-party static code analysis tools and dependency check tools to identify typical security problems and vulnerabilities, ensuring code security and compliance.
  - Applying various test methods: In addition to the use of tools, testing at the application level is also necessary, such as using fuzz testing to identify and fix potential vulnerabilities and errors.

- **Related cloud services and tools**

  HSS

3.6.4.4 SEC05-04 Key Security Management

Key security management is critical to the security of the entire workload. If keys are managed improperly, even strong cryptographic algorithms cannot ensure system security. The security management covers the entire lifecycle of keys, including key generation, transmission, use, storage, update, backup and restoration, and destruction.

- **Risk level**

  High

- **Key strategies**
  - Generating a key:
    - Manage keys hierarchically. Divide keys into at least two layers: root keys and working keys. The root keys encrypt working keys for protection.
    - Generate keys using secure random number generators to ensure the randomness and unpredictability of keys. Do not use weak or fixed keys.
  - Transmitting a key:
    - Use secure communication channels, such as encrypted channels or physical transmission, to transmit keys.
    - Ensure that keys are not stolen or tampered with during transmission.
  - Using a key:
    - Minimize the use of keys and avoid exposing keys unnecessarily.
    - Implement access control and permission management to restrict access to keys.

 Copyright © Huawei Technologies Co., Ltd.

  - Storing a key:
    - Use secure storage devices or encrypted storage to store keys.
    - Ensure that only authorized personnel can access the key storage.
  - Updating a key:
    - Periodically update keys to cope with security vulnerabilities and attacks.
    - Rotate keys securely to ensure service continuity.
  - Backup and restoration:
    - Periodically back up keys and store the backup securely.
    - Ensure that there is a reliable recovery mechanism to prevent key loss or damage.
  - Destroying a key:
    - Destroy keys when they are no longer needed.
    - Destroy keys securely, for example, use encrypted deletion or physical destruction.
- **Related cloud services and tools**

  DEW

3.6.4.5 SEC05-05 Certificate Security Management

Certificates are commonly used for transmission data encryption and identity authentication between systems. Centrally manage the usage and validity period of certificates and replace them in a timely manner.

- **Risk level**

  Medium
- **Key strategies**
  - Centrally managing certificates:
    - Establish a centralized certificate management system to store, track, and manage all certificates.
    - Ensure that all certificates are clearly identified, including the usage, owner, and validity period.
  - Validity period management:
    - Periodically check the validity period of the certificate. Update or replace the certificate that is about to expire in a timely manner.
    - Do not use expired certificates to prevent security vulnerabilities and service interruption.
  - Secure storage:
    - Store the certificates securely. Ensure that only authorized personnel can access the certificates.

 Copyright © Huawei Technologies Co., Ltd.

        - Implement extra protection on private keys, for example, use hardware security modules (HSMs) to store private keys.
      - Encrypted transmission:
        - Use encrypted channels, such as SSL and TLS, to transmit certificates to prevent certificates from being tampered with or stolen.
        - Do not transmit certificates on insecure networks for security purposes.
- **Related cloud services and tools**

  Cloud Certificate Manager (CCM): provides one-stop management, such as applying for, issuing, querying, and revoking SSL certificates.

3.6.4.6 SEC05-06 Using Hosted Cloud Services

Use Huawei Cloud services to host resources such as compute data, databases, and storage resources, avoiding development and O&M costs caused by user-construction.

- **Risk level**

  Low
- **Key strategies**
  - Use services for hosting resources, reducing security maintenance tasks in a shared responsibility model. For example, use Huawei Cloud database services instead of self-built relational database instances.
  - Use cloud services with serverless architectures. Huawei Cloud will process compute resources securely, relieving users of workload and human errors caused by server O&M and reducing security vulnerabilities. In this way, you can concentrate on service logic and application security.
- **Related cloud services and tools**
  - RDS for MySQL
  - GaussDB
  - FunctionGraph
  - CCI
  - EG

3.7 Application Security

3.7.1 SEC06 Application Security

3.7.1.1 SEC06-01 Ensuring Open-Source Software Security

Open-source software is indispensable in modern software development. More and more enterprises choose to use open-source software to develop and deploy software applications. Open-source software must be used in compliance with laws and regulations. You need to check the sources, vulnerability management,

traceability, centralized management, and lifecycle management of open-source software.

- **Risk severity**

  High

- **Key strategies**
  - Use reliable sources. Open-source software is publicly available, making it easy for hackers to identify and exploit its vulnerabilities for attacks. Make sure you obtain the open-source software directly from the official website of the authorized community, supplier, or vendor.
  - Comply with software license requirements. Make sure the open-source software in use comes with a proper license or a signed usage agreement. Open-source software must be used in accordance with its license terms and relevant laws to prevent intellectual property or licensing issues. Open-source obligations must be fulfilled to avoid damage to the reputation of products or enterprises.
  - Enable centralized management. You should centrally manage open-source software, unify the introduction of open-source software, establish a preferred library and roadmap library, and reduce the types and quantity of open-source software. You can encourage your teams to use preferred open-source software to ensure the quality and security.
  - Reduce the impact of open-source vulnerabilities. Open-source software vulnerabilities spread quickly and have a great impact. Once a security vulnerability was reported, the key to minimizing impact lies in quickly identifying affected products and fixing the vulnerability.
  - Enable traceability. The change process of open-source software must be controllable and recorded. The relationship between product versions and third-party software and vulnerabilities must be established.

3.7.1.2 SEC06-02 Establishing Secure Coding Specifications

Application security involves multiple phases, including requirements analysis, design, implementation, and deployment, among which implementation is an important phase. Establishing secure coding specifications helps the development team write more secure and high-quality code, reducing or even eliminating security risks caused by code errors.

- **Risk severity**

  High

- **Key strategies**
  - Release secure coding specifications for common programming languages used by the development team. The general secure coding specifications should include program input verification, program output coding, identity authentication, access control, secure encryption and decryption algorithms, exception handling, I/O operations, file uploads, serialization, and output formatting.
  - For languages used in web application scenarios, such as Java and Python, consider secure session management and SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF) prevention.

 Copyright © Huawei Technologies Co., Ltd.

  - For the C/C++ language, consider more about security risks such as buffer overflow, command injection, unsafe function usage, memory leaks, pointer manipulation errors, and array read/write boundary violations.
  - For JavaScript, consider more about security risks such as cross-site scripting (XSS) attacks.

3.7.1.3 SEC06-03 Implementing White-Box Code Reviews

White-box code review is a software quality assurance method that checks the internal structure, logic, and implementation details of source code to ensure that the code complies with best practices, programming specifications, and security standards. In white-box code review, team members check the quality, security, and readability of the code to identify potential issues and improvement opportunities.

- **Risk level**

  Medium

- **Key strategies**

  a. **Develop a review plan**:

     i. Determine the review frequency and schedule to ensure that code review is a continuous activity.

     ii. Determine the review scope, for example, new code snippets (reviewed each time they are submitted), new functions (reviewed when a function is completed), or large-scale code (reviewed periodically).

  b. **Train team members**:

     i. Provide training to ensure that team members understand how to perform effective code review.

     ii. Ensure that the team understands the purpose and importance of code review, and how to identify common issues and potential security vulnerabilities. It is recommended that the top issues be sorted into a list for self-check by developers and reviews by others.

  c. **Select proper tools**:

     i. Use code review tools, such as static code analysis tools, to help identify potential issues.

     ii. Ensure that the team is familiar with and can effectively use these tools.

  d. **Set clear standards and guidelines**:

     i. Develop clear code review standards and guidelines so that reviewers can consistently assess code quality.

     ii. Focus on security.

  e. **Assign roles and responsibilities**:

     i. Determine code reviewers, such as developers, architects, and security experts.

     ii. Ensure that each team member understands their roles and responsibilities for code reviews.

 Copyright © Huawei Technologies Co., Ltd.

f. **Record review results**:

   i. Record the findings, suggestions, and decisions after a review for future follow-up and improvement.

   ii. Ensure that issues are properly tracked and resolved.

g. **Encourage cooperation and discussion**:

   i. Encourage team members to cooperate and discuss with each other, and share their experience and opinions to improve the review quality.

   ii. Create an open atmosphere, so that team members can raise questions and suggestions to promote joint learning and growth.

h. **Make continuous improvement**:

   i. Regularly assess the code review process, collect feedback, and make necessary adjustments and improvements.

   ii. Focus on improving review efficiency and quality to ensure that the team continuously improves code quality and security.

3.7.1.4 SEC06-04 Configuring Application Security

Configurations of applications must be hardened to eliminate security vulnerabilities and risks caused by improper security configurations.

- **Risk level**

  High

- **Key strategies**

  Harden security configurations and configure web attack protection for applications, such as Nginx, Tomcat, Apache, Jetty, JBoss, PHP, and Redis, you may use in your workloads by referring to security configuration specifications.

  – Privilege escalation, such as capability privilege escalation, suid file privilege escalation, scheduled task privilege escalation, and sudo file configuration privilege escalation

  – Users for running services. For example, the users who run services can have only the minimum permissions. The **root** user is not allowed to run services.

  – Protection against common web attacks, such as SQL injection, XSS, file inclusion, directory traversal, sensitive file access, command and code injection, Trojans, and third-party vulnerability exploits

- **Related cloud services and tools**

  – Host Security Service (HSS)

  – Web Application Firewall (WAF)

3.7.1.5 SEC06-05 Performing Penetration Testing

Penetration testing is a security assessment method that simulates the behavior of attackers and simulates real attack scenarios to assess the security of systems, applications, or networks. Penetration testing aims to discover security vulnerabilities, weaknesses, and potential security risks in the target system to

 Copyright © Huawei Technologies Co., Ltd.

help organizations improve their security measures, harden systems, and protect systems from real attacks.

- **Risk level**

  High

- **Key strategies**

  a. Perform penetration testing for the entire system. You are advised to arrange the penetrating testing in the later period of the development phase to ensure that all system functions are close to the expected release status, and reserve enough time to resolve discovered issues.

  b. Use a structured process to determine the scope of penetration testing and maintain scenario relevance based on the threat models to ensure comprehensive assessment of system security.

  c. Use tools to automate common or repeatable tests to speed up penetration testing and improve efficiency.

  d. Analyze penetration test results to determine systematic security issues and provide useful information for further automated testing and developer training.

  e. Provide training for developers and make sure they know what the penetration test results mean and where they can obtain fixes. This will help resolve security issues in a timely manner.

3.8 Data Security and Privacy Protection

3.8.1 SEC07 General Data Security

3.8.1.1 SEC07-01 Identifying Data in Workloads

Evaluate data sensitivity and perform data classification and tiering based on business processes, data flow, distribution, and ownership in compliance with regulatory requirements.

- **Risk level**

  High

- **Key strategies**

  Follow the following steps to sort out and identify data:

  a. Analyze the service process.

     - Understand the service process, and determine the types and usage of data generated, processed, and stored in each phase based on the service process diagram.
     - Communicate with the business department, development team, and O&M engineers to obtain detailed information about data.

  b. Determine data distribution: Determine where data is stored, for example, on EVS disks, databases, or OBS.

  c. Evaluate data sensitivity.

 Copyright © Huawei Technologies Co., Ltd.

- Determine the type and content of data, for example, whether the data contains personal identity information (such as name, ID number, and address), financial data (such as bank account and transaction records), confidential business information (such as product R&D plan and customer list), or other data protected by laws and regulations.
- Consider the potential impact of the data. Assess the potential impact of data leakage or misuse on individuals, organizations, or society, including financial losses, reputational harm, and legal risks.
- Refer to related laws, regulations, industry standards, and internal compliance policies. The definition and requirements of sensitive data may vary depending on industries and regions. For example, patient data in the healthcare industry and customer transaction data in the financial industry are protected by specific regulations and standards.
- Combine the business strategy and risk tolerance of the organization. For data related to key services, even if the data is not sensitive in common sense, it can still be considered as highly sensitive due to its importance to services.

d. Use data detection and classification tools to automatically scan workloads to identify data. This can help you implement the proper control measures.

e. Create and maintain a data inventory. Create a data inventory of classified and categorized information, including the data name, description, source, distribution, sensitivity, and category level.

- **Related cloud services and tools**

  Data Security Center (DSC): DSC can accurately identify sensitive data in the database based on the sensitive data detection policy, automatically detect and analyze sensitive data usage from massive data, as well as scanning, classifying, and categorizing structured and unstructured data based on the data identification engine to eliminate data blind spots.

3.8.1.2 SEC07-02 Data Protection Control

Implement tiered data protection for each classified data category to safeguard data confidentiality and integrity.

- **Risk level**

  High

- **Key strategies**
  - Implement proper data protection measures, such as encryption and authentication.
  - Manage data access permissions. Be aware of who can access, modify, and delete data. This helps restrict data access permissions and reduce data leakage risks. Only authorized users can access data based on the least privilege principle and perform related operations.
  - Sensitive data should be anonymized before being shared or disclosed to prevent sensitive information leakage.

  - Protect data integrity. Periodically back up data and control version to protect your data from being tampered with or deleted. Isolate key data from other data to protect its confidentiality and integrity.
  - Ensure that OBS buckets that store important service data and sensitive data are not publicly readable to prevent unauthorized data access.
  - Develop a risk management plan: Understand the possible business impact of accidental data disclosure, change, and deletion. Develop a risk management plan accordingly.
- **Related cloud services and tools**
  - Database Security Service (DBSS)
  - DEW

3.8.1.3 SEC07-03 Monitoring Data Use

Implement restriction policies or establish a monitoring mechanism for data modification and batch operations based on data classification.

- **Risk level**

  High
- **Key strategies**
  - Implement restriction measures or establish a monitoring mechanism for data modification and batch operations.
  - Use DBSS to audit operations on databases. DBSS database audit provides the audit function in bypass mode. It records user access to the database in real time, generates fine-grained audit reports, and sends real-time alerts for risky operations and attacks, helping users locate internal violations and improper operations in the database, as well as ensuring data asset security.
  - Enable DBSS database audit alarms. After configuring alarm notifications, you can receive DBSS alarms on database risks.
  - Use CBH to identify and intercept high-risk database commands. With CBH, you can dynamically identify and intercept high-risk commands (including deleting databases, modifying key information, and viewing sensitive information) to interrupt database O&M sessions by setting database control policies and preset command execution policies. In addition, the system automatically generates a database authorization ticket and sends it to the administrator for secondary authorization.
- **Related cloud services and tools**
  - DBSS
  - CBH

3.8.1.4 SEC07-04 Static Data Encryption

Encryption prevents unauthorized users from accessing and stealing data. Sensitive static data should be encrypted by default to ensure confidentiality even if the data is accessed or disclosed without authorization.

- **Risk level**

  High

 Copyright © Huawei Technologies Co., Ltd.

- **Key strategies**
  - Enable default encryption. Enable default encryption for cloud services, such as EVS, RDS, OBS, and SFS, to automatically encrypt stored data. Enable encryption for databases, such as RDS and DWS, to reduce security risks caused by database dragging and data leakage.
  - Encrypt, mask, and anonymize sensitive data. In this way, even if sensitive data is stolen, the risk of data leakage can be reduced.
  - Monitor the use of encryption and decryption keys and select different encryption keys based on data usage, type, and classification.
- **Related cloud services and tools**

  DEW: Based on the integration between DEW and services such as OBS, EVS, and IMS, KMS can manage the keys of these services, encrypt service data, and encrypt local data using KMS APIs.

3.8.1.5 SEC07-05 Transmission Data Encryption

Encrypt data during transmission to prevent data from being stolen, tampered with, or viewed by unauthorized users.

- **Risk level**

  High
- **Key strategies**
  - Using encryption protocols: Ensure that secure encryption protocols are used to encrypt data and protect it from being stolen or tampered with during transmission. Use the latest TLS version (TLS 1.2 or later) to ensure that the strongest encryption standard is used.
  - Secure transmission channel: Ensure that the data transmission channel is secure. Do not use insecure networks or public networks to transmit sensitive data.
  - Ensure that sensitive data is encrypted when being transmitted between the cloud and the client. In this way, data is difficult to be decrypted even if it is stolen.
  - E2E encryption: Use E2E encryption to ensure that data stays encrypted during transmission, that is, from the data generation end to the data receiving end. Use HTTPS to encrypt communication between clients and servers for web application APIs.
  - Data integrity verification: Use hash functions, digital signatures, and message authentication codes (MACs) to verify data integrity to ensure that data is not tampered with during transmission.
- **Related cloud services and tools**
  - VPN
  - Direct Connect
  - Cloud Connect
  - Data Express Service (DES)
  - CCM

 Copyright © Huawei Technologies Co., Ltd.

3.8.2 SEC08 Data Privacy Protection

It refers to a series of measures and technologies are taken to ensure that personal data and sensitive information is properly protected during collection, storage, processing, and transmission to prevent unauthorized access, use, or leakage. Data privacy protection aims to protect personal privacy rights and the confidentiality and integrity of personal information.

3.8.2.1 SEC08-01 Specifying Privacy Protection Policies and Principles

It refers to the general policies and principles established and adhered to in the protection of personal privacy data.

- **Risk level**

  High

- **Key strategies**

  Specify the classification and impact of personal data. Personal data includes the natural person's email address, phone number, biometric identifiers (such as fingerprints), location data, IP address, medical information, religious belief, social security number, and marital status. Based on impact level, personal data can be categorized into high-impact, medium-impact, and low-impact personal data (as listed in the following table).

Impact Level	Description
High Impact	Improper disclosure of such data may violate laws and pose serious adverse impacts on the company's reputation, finance, or operations and pose major negative impacts on the entity of personal data. For example, ID number and fingerprints.
Medium Impact	Improper disclosure may have serious adverse impact on the company and result in a major adverse impact on the data subject. Personal data at this level includes account, address, and age.
Low Impact	Improper disclosure of such data will pose certain impacts, which are controllable to the company, and minor impacts on the subject of personal data, for example, gender.

  Factors that affect the privacy risk level include: personal data level (high, medium, or low), whether the data subject can be directly or indirectly identified, data volume, data attributes (for example, the disease history is more sensitive than the phone number), whether other data can be associated in the storage area, purposes of personal data collection/storage/processing/disclosure (for example, statistical analysis, research, tax management, and legal requirements), and role (controller/processor/equipment supplier).

  – Sensitive personal data is strictly protected. Generally, sensitive personal data includes life information (such as racial or ethnic origin and political opinions), identity information (such as ID card number and social

 Copyright © Huawei Technologies Co., Ltd.

security number), property information (such as bank account information and deposit information), health information (such as medical history, and diagnosis and treatment information), biometric feature information (such as fingerprint and iris), and other information (such as precise location information).

- Specify the roles involved in personal data, including data subjects, data controllers, data processors, and third parties.
- Specify privacy protection principles and comply with the principles of legitimacy, transparency, and security.
  - Lawfulness, fairness and transparency: Personal data shall be processed lawfully, fairly, and in a transparent manner in relation to the data subject.
  - Purpose limitation: Personal data shall be collected for specified, explicit, and legitimate purposes and not further processed in a manner that is incompatible with those purposes.
  - Data minimization: Personal data shall be adequate, relevant, and limited to what is necessary in relation to the purposes for which they are processed. Personal data shall be anonymized or pseudonymized wherever possible to reduce the risks to data subjects concerned.
  - Accuracy: Personal data shall be accurate and, where necessary, kept up to date. Every reasonable step shall be taken to ensure that personal data that is inaccurate, having regard to the purposes for which it is processed, is erased or rectified without delay.
  - Storage period minimization: Personal data shall be kept for no longer than the period that is necessary to achieve the purposes for which it is processed.
  - Integrity and confidentiality: Personal data shall be processed in a manner that ensures appropriate security of the personal data, including protection against accidental or unlawful destruction, loss, and alteration or unauthorized access and disclosure, using appropriate technical or organizational measures based on the state of the art, the implementation costs, and the impact and likelihood of privacy risk.
  - Accountability: The data controller shall be responsible for, and be able to demonstrate compliance with the aforesaid principles.
- Privacy protection shall be implemented throughout the full lifecycle of personal data. Each phase of the personal data lifecycle shall have corresponding privacy protection requirements, and mitigation measures shall be developed based on privacy risks in different phases.

3.8.2.2 SEC08-02 Proactive Notification to Data Subjects

It specifies that the data controller proactively provides data subjects (individuals) with information about data processing activities, such as the data collection purpose, data processing method, data use scope, and data storage period. This notification is usually presented in the form of a privacy policy, user agreement, or prompt message.

 Copyright © Huawei Technologies Co., Ltd.

- **Risk level**

  Medium

- **Key strategies**
  - The importance of proactive notification to data subjects is as follows:
    - Transparency and control: Through proactive notification, data subjects can understand how data handlers process their personal data, thereby increasing the transparency and controllability of personal data and enabling them to make informed consent decisions.
    - Compliance: In many privacy protection laws and regulations (such as the GDPR in Europe), proactive notification to data subjects is a legal requirement. Data processors need to provide specific information to data subjects to ensure that data processing activities comply with relevant laws and regulations.
    - Trust building: Data handlers can show their importance to privacy protection to data subjects through proactive notification, thereby building trust relationships.
    - Rights protection: Proactive notification helps protect data subjects' rights, including rights to know, access, and rectification, so that they can exercise their privacy rights.
  - The data controller must provide a privacy statement, which is intended to inform users of the privacy and data protection practices of the product and the options available to users in terms of personal information collection and processing.
  - The privacy statement and personal data description should describe the types, purposes, processing methods, and time limits of all personal data collected by the product.
  - A privacy statement must be provided on the product page to facilitate user access. For example, the link to the privacy statement should be provided on the web page footer.

3.8.2.3 SEC08-03 Data Subjects' Choice and Consent

It refers to the process in which the data processor needs to obtain the explicit consent of the data subject (individual) before the personal data is collected, processed, or used, and the data subject has the right to choose whether to agree to the processing of the personal data.

- **Risk level**

  High

- **Key strategies**
  - Before the collection or use of personal data, notify users, obtain users' consent, and allow users to disable the collection and use of personal data.
  - Before exporting an error report containing personal data from the data subject's system, provide a mechanism to notify the data subject and obtain his/her consent.

 Copyright © Huawei Technologies Co., Ltd.

  - If personal data is used for marketing, user profiling, or surveys, the data controller and equipment vendor must provide a mechanism to obtain data subjects' opt-in consent and allow them to withdraw their consents anytime.
  - Before setting or reading cookies (for marketing or advertising) on the data subject's system, provide a mechanism to obtain the data subject's consent and allow the data subject to withdraw the consent any time.
  - Obtain users' consent before modifying users' personal space (e.g., system or application configuration change, software download, and system or software upgrade).
  - Guardians' consent must be obtained for services provided for minors or when personal information containing age information is collected.
  - The data controller should provide a mechanism to record users' consent and consent withdrawal.

3.8.2.4 SEC08-04 Data Collection Compliance

It refers to that the data controller shall comply with relevant laws, regulations, and privacy protection guidelines when collecting personal data, and ensure that data collection activities comply with laws and regulations and respect data subjects' rights.

- **Risk level**

  High

- **Key strategies**
  - Personal data collected only after data subjects' authorization is obtained
  - Sensitive personal data collection must obtain explicit consent from data subjects.
  - The collection scope, purposes, and processing method of using personal data shall not exceed those specified in the privacy statement and must comply with the minimization principle.
  - Provide data subjects with a way to withdraw or change their consent after obtaining their consent.

3.8.2.5 SEC08-05 Compliance of Data Usage, Retention, and Disposal

Data handlers must comply with relevant laws, regulations, and privacy protection guidelines when processing personal data, including data use, retention, and destruction, to ensure that data processing activities comply with laws and regulations and respect data subjects' rights.

- **Risk level**

  High

- **Key strategies**
  - Obtain data subjects' authorization before using personal data. The use scope and method cannot exceed the collection purpose.
  - The system should set the privacy protection function to protect privacy by default.

 Copyright © Huawei Technologies Co., Ltd.

  - Ensure personal data security during the use of personal data, for example, record personal data addition, deletion, modification, and batch export operations in the O&M phase.
  - Follow the data minimisation principle when logging personal data for fault location.
  - The data controller must not collect or process the personal data any more after data subjects withdraw their consent.
  - Systems that provide user profiles shall provide a mechanism for users to opt out of such profiles.
- **Related cloud services and tools**
  - Data Security Center (DSC): Users can use the preset masking rules of DSC or customize masking rules to mask specified database tables. DSC supports various cloud scenarios, such as RDS and self-built databases on ECSs. In addition, DSC automatically provides data masking suggestions and allows users to configure masking rules only in a few clicks.
  - Database Security Service (DBSS): DBSS can be used to mask data. To mask sensitive information in entered SQL statements, users can enable the privacy data masking function of DBSS and configure privacy data masking rules to mask specified database tables and queries from specific source IP addresses, users, and applications.

3.8.2.6 SEC08-06 Compliance of Personal Data Disclosure to Third Parties

When sharing, transferring, or providing personal data to a third party, the data controller must comply with relevant laws, regulations, and privacy protection guidelines to ensure that data transfer activities comply with laws and regulations and respect data subjects' rights.

- **Risk level**

  High
- **Key strategies**
  - The product team needs to evaluate whether personal data is pushed to third-party applications. Evaluate whether highly sensitive user data is pushed without users' explicit consent. In addition, check whether third-party applications provide a proper protection mechanism for shared data.
  - Obtain users' consent before transferring their personal data to third parties and comply with the lawfulness principle.
  - The purposes and scope of the transfer must be limited to what has been stated for the collection.
  - Keep the personal data integral, accurate, and up to date. Prevent personal data from being tampered with, deleted, or abused in any phase.
  - The exporter must obtain an explicit promise from the receiver to ensure the integrity, accuracy, and security of personal data and prevent abuse or unauthorized disclosure.
  - If high-impact personal data (e.g. passwords, bank accounts, and batch personal data) needs to be transmitted, encrypt it before transmission or transfer it through a secure channel.

 Copyright © Huawei Technologies Co., Ltd.

  - If cross-border data transfer is involved, comply with local laws and regulations.

3.8.2.7 SEC08-07 Data Subjects Have Rights to Access Their Privacy Data

It refers to that the data subject has the right to require the data handler to provide access to their personal data according to relevant privacy protection laws and regulations.

- **Risk level**

  High

- **Key strategies**
  - Users must be provided with functions for personal data query and update. Compliant with the data subject participation principle, the function must be provided in real time and at no cost to the data subject.
  - Data subjects must be authenticated before accessing personal data.
  - The time when data is entered or updated must be recorded.
  - It is advised to provide necessary check methods. For example, when a user enters an email address on a web page, verify the email address format.
  - In the case of user registration information, provide users a way to modify their registration information.
  - Make user privacy preference settings and options easy to see and use.
  - Systems that collect, process, and store personal data should provide a mechanism for data subjects to restrict the processing of their personal data.
  - For systems that collect, process, and store personal data, a mechanism for exporting personal data provided by data subjects should be provided.

3.9 Security Operations

3.9.1 SEC09 Security Awareness and Analysis

3.9.1.1 SEC09-01 Implementing Standardized Log Management

Standardized log management must be implemented for logs of the identity, network, application, server, data, and O&M defense layers to monitor systems and user activities, enabling centralized log management and ensuring transparent and traceable security operations.

- **Risk level**

  High

- **Key strategies**
  - Track and monitor all access to network resources and key data. The system's activity recording mechanism and user activity tracking function can effectively reduce the threat of malicious activities to data. Common security logs include host security logs, OS logs, bastion host logs, IAM

 Copyright © Huawei Technologies Co., Ltd.

logs, WAF attack logs, CFW logs, VPC flow logs, and DNS logs. When an error or security event is reported for a system, you can trace, alert, and analyze the error or security event and quickly locate the cause of the threat.

- Ensure that the log retention duration meets requirements. HSS and cloud service logs transferred to LTS will be deleted automatically after the default log retention duration expires. So, you need to configure a log retention duration based on your service requirements. If you want to store logs for a longer period, you need to configure log transfer in LTS.
- Enable centralized security management and operations if you have many accounts in your organization. This helps collect logs, alarms, configurations, policies, and asset data across multiple cloud environments, accounts, and cloud service products. In doing this, your organization security operations and O&M efficiency can be improved, and accounts and resources can be managed centrally. Centralized log management enables centralized storage, analysis, modeling, threat analysis, orchestration, response, situation reporting, and security policy management.

- **Related cloud services and tools**
  - Log Tank Service (LTS): You can use LTS to collect logs for efficient, real-time decision-making analysis, device O&M management, and service trend analysis.
  - Web Application Firewall (WAF)
  - SecMaster

3.9.1.2 SEC09-02 Logging and Analyzing Security Incidents

Pre-incident forensics capabilities must be in place to support security incident investigation. Attacks and abnormal behaviors must be logged and analyzed. Security controls must be deployed at key network nodes, such as the border between the internal and external networks and the nodes housing ELB load balancers, to detect, restrict, or block cyber attacks. Technical measures must be taken to continuously monitor and analyze collected security logs to identify and analyze cyber attacks, especially emerging cyber attacks, and abnormal behaviors.

- **Risk level**

  High

- **Key strategies**
  - Build pre-incident forensics capabilities to support security incident investigation. Log and analyze attacks and abnormal behaviors. Security controls must be deployed at key network nodes to detect, restrict, or block cyber attacks. Technical measures must be taken to continuously monitor and analyze collected security logs to identify and analyze cyber attacks, especially emerging cyber attacks, and abnormal behaviors.
  - Analyze attack chains and trace attack sources based on security events. This includes all attack paths, initial access, execution, persistence, privilege escalation, defense evasion, credential access, information discovery, lateral movement, data collection, command control, data theft, and impact and damage.
  - Build online, nearline, and offline anomaly behavior analysis models based on a platform that integrates stream and batch processing. These

models can work on identity, network, application, data, O&M, and host defense layers. The ad hoc real-time analysis of security events can also be implemented, with results automatically aggregated into interactive dashboards and customizable reports.

- **Related cloud services and tools**
  - SecMaster

3.9.1.3 SEC09-03 Implementing Security Audits

Enable security audits for key operations on each cloud service and by any individual user. Protect audit logs and back them up periodically to avoid unexpected deletions, modifications, or overwriting.

- **Risk level**

  High
- **Key strategies**
  - Key operations on cloud services include high-risk operations (such as creating and deleting IAM users, restarting VMs, and changing security configurations), cost-sensitive operations (such as creating and deleting high-cost resources), and service-sensitive operations (such as network configuration changes).
  - Enable the key event notifications. If you enable key event notifications in CTS, CTS sends notifications to subscribers in real time through SMN.
  - Enable log transfer in CTS to send audit logs to OBS. You can set a log retention duration based on your compliance and service requirements.
  - Protect audit logs and back them up periodically to avoid unexpected deletions, modifications, or overwriting. You can also enable file verification for audit logs to ensure the integrity of audit files and prevent files from being tampered with.
  - Enable centralized fine-grained control for O&M account permissions to access systems and resources.
  - For details about data security audit, see **SEC07-03 Monitoring Data Use**.
- **Related cloud services and tools**
  - Cloud Trace Service (CTS): A tracker is automatically created when you enable CTS. The tracker identifies and associates with all cloud services you are using, and records all operations on the services. CTS allows you to collect, store, and query all operations on your cloud resources and use these records for security analysis, compliance auditing, resource tracking, and fault locating.
  - Cloud Bastion Host (CBH)
  - Database Security Service (DBSS)
  - SecMaster
  - Simple Message Notification (SMN)

3.9.1.4 SEC09-04 Security Situation Awareness

All access to network resources and critical data must be monitored. Implementing comprehensive activity logging mechanism and user behavior

 Copyright © Huawei Technologies Co., Ltd.

tracking can significantly reduce the threat of malicious activities to data. When an error or security event is reported for a system, you can trace, alert, and analyze the error or security event and quickly locate the cause of the threat.

- **Risk level**

  Medium

- **Key strategies**
  - Aggregate alerts/events from multiple security services to enable data association, retrieval, sorting, and comprehensive security operations assessment.
  - Generate periodic security status reports to summarize the security situation, including issues identified and actions and improvement measures taken.
  - Ensure that all security measures comply with applicable laws and regulations and industry standards, such as *DJCP*, *GDPR*, *HIPAA*, and *PCI DSS*.
  - Provide regular security training for employees to improve their awareness and understanding of cloud security.
- **Related cloud services and tools**
  - SecMaster
  - Cloud Eye: You can use Cloud Eye to receive alarm notifications of security events. Cloud Eye provides alarm notifications based on monitored metrics. If a monitored metric of a cloud service reaches the threshold specified in an alarm rule, Cloud Eye notifies you by email or text message, so that you can learn cloud service statuses in a timely manner. Cloud Eye can also send HTTP/HTTPS messages to an IP address of your choice, helping you build smart alarm handling programs.

3.9.2 SEC10 Security Incident Response

3.9.2.1 SEC10-01 Establishing a Security Response Team

Establish a security incident response team and specify roles and responsibilities.

- **Risk level**

  High

- **Key strategies**

  The security incident response team generally includes the following roles and responsibilities:
  - Security response experts lead the investigation of network security incidents, rate and report the incidents, trace attack sources, determine the impact scope, develop emergency handling measures, and promote service to control risks.
  - Attack source tracing experts trace the attack source based on the attack IOC information, including the attacker information, entire attack scope, attack source tracing diagram (attack path), and attack source tracing report.
  - Senior analysis experts analyze and reproduce vulnerabilities, perform reverse analysis on malicious samples, and output virus scanning scripts.

 Copyright © Huawei Technologies Co., Ltd.

  - Service security response experts assist security response personnel in investigating and analyzing incidents, execute log forensics, and provide service architecture, network architecture, and service logs to help analyze attack sources and impact scopes. Implement emergency response measures and cover security products.

3.9.2.2 SEC10-02 Developing an Incident Response Plan

The Incident Response Plan (IRP) is an important part of an organization's security policy. It aims to ensure that actions can be taken quickly and orderly when a security incident occurs, minimize losses, and restore normal operations as soon as possible.

- **Risk level**

  High

- **Key strategies**
  - Establish an incident response plan, including defining incident levels, response processes, and recovery strategies. Security incidents that affect service availability or can be perceived by tenants are classified into five levels: S1, S2, S3, S4, and S5.
  - Implement continuous monitoring, including logs, network traffic, and abnormal behavior of the cloud environment. A preliminary analysis is performed when a potential incident is detected to determine the severity.
  - Implement rapid security response actions to isolate affected systems or accounts, disconnect network connections, stop services, remove malicious files, fix vulnerabilities, replace abnormal systems, and harden systems. Ensure that all threats have been eliminated to avoid recurrence.
  - Formulate recovery policies to gradually recover affected services, ensure data and system consistency, and perform tests to ensure that all systems are running properly.
  - Conduct post-incident analysis to summarize the cause, response process, and lessons of the incident. Update the incident response plan and make improvements.
  - Incident response plans are regularly reviewed and updated to accommodate new threats and business needs.

Event Severity	Incident Response Time	Average Risk Control Time
S1	5 minutes	1 hour
S2	5 minutes	2 hours
S3	5 minutes	4 hours
S4	10 minutes	24 hours
S5	10 minutes	48 hours

 Copyright © Huawei Technologies Co., Ltd.

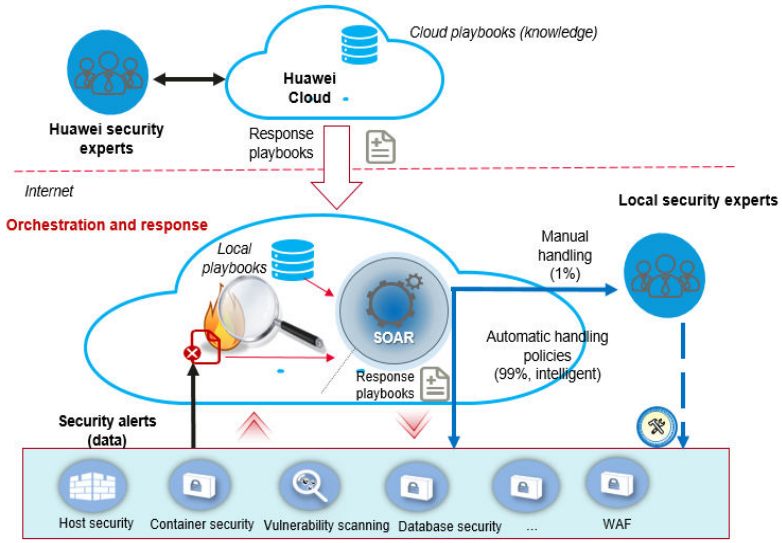
3.9.2.3 SEC10-03 Automatic Response to Security Incidents

The workflow of automatic response is the core component of security automation. It aims to reduce the response time of security incidents and improve the processing efficiency.

- **Risk level**

  High

- **Key strategies**



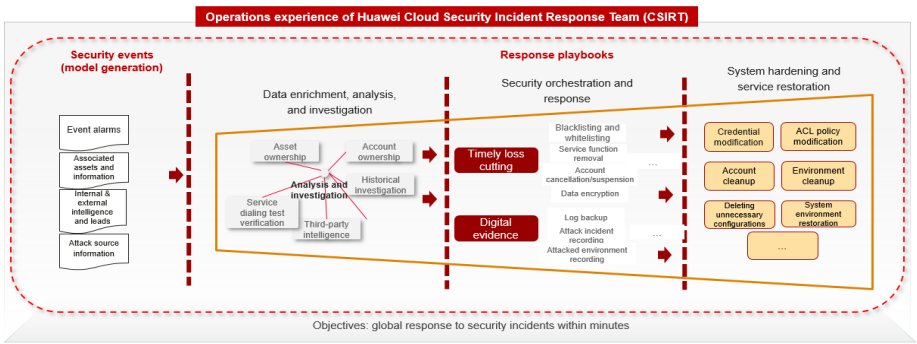
Security Orchestration, Automation and Response (SOAR)

  - Define response trigger conditions: Determine the conditions that trigger automatic response based on threat intelligence, abnormal behavior detection, and real-time monitoring results.
  - Develop response policies: Develop specific response actions for each type of threat or incident, such as isolation, repair, notification, and investigation.
  - Priority and level: Define the response priority based on the severity and urgency of incidents to ensure that important incidents are handled first.
  - Continuous monitoring: Use tools such as Security Information and Event Management (SIEM) and User and Entity Behavior Analysis (UEBA) to monitor networks, systems, applications, and user activities in real time.
  - Intelligent alarm: When an event that meets the predefined trigger condition is detected, an alarm is automatically generated and classified based on the priority.
  - Isolation and control: Infected devices or network segments are automatically isolated to prevent threat spreading.
  - Automatic fixing: For known vulnerabilities or issues, automatically install patches, modify configurations, or clear malware.
  - Evidence collection and recording: Automatically collect event-related logs, network packets, and other evidences and saves them for subsequent analysis.

 Copyright © Huawei Technologies Co., Ltd.

- Notification and communication: Alarms are sent to security team members, and notifications are sent to the IT department, management, or other related parties.
- Automatic analysis: Machine learning and data analysis tools are used to automatically analyze the nature, source, and impact scope of incidents.
- Human-machine collaboration: Security analysts review the results of automated analysis and, if necessary, perform manual analysis to confirm the severity of the incident and the subsequent steps.
- Decision-making support: Based on the analysis result, determine whether further manual intervention is required or adjust the automatic response policy.
- Automatic recovery: Automatic system recovery, data recovery, or service restart is performed for resolved incidents.
- Report generation: Automatically generates incident handling reports, including incident details, response actions, handling results, and recommended measures.
- Compliance check: Ensure that the entire response process complies with laws, regulations, and industry standards.
- Incidents review: Periodically review the handled incidents, evaluate the effect of automatic response, and identify improvements.
- Rule and policy update: Update automatic response rules and policies based on the review result to enhance the system adaptability.
- Training and drill: Regularly train and drill the security team on the automatic response process to ensure that personnel are familiar with the process and efficiently execute it in actual operations.

Implement the experience of top security experts in automatic threat handling based on a playbook. An incident response playbook is a set of defined response workflow. It leverages security orchestration, automation, and response (SOAR) technology to configure actions for different assets and protection components to achieve global response to single-point risks.



- **Related cloud services and tools**

  SecMaster: SOAR

3.9.2.4 SEC10-04 Security Incident Drill

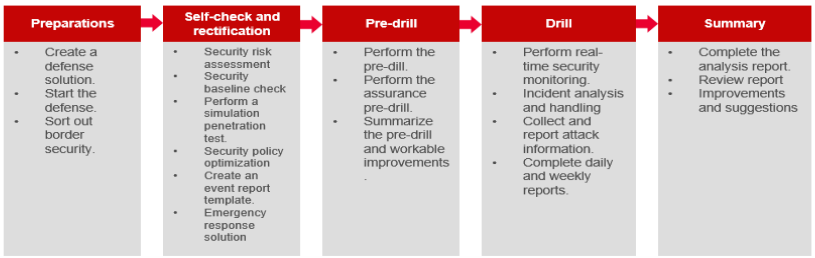
It is a simulated activity that enables organization members to perform operations and respond to security incidents in a simulated scenario. The purpose is to test

and improve their capabilities of responding to security incidents. Organizations can assess the effectiveness of their security incident response plans, identify potential problems, and make improvements to improve the team's response capabilities.

- **Risk level**

  High

- **Key strategies**



According to the security requirements of practical, systematic, and regular defense; dynamic defense, proactive defense, in-depth defense, precise defense, overall defense, and joint defense and control, practical attack-defense drills will become the norm. Promoting defense through attacks is an effective way to achieve precise defense and protection.

Event drills are used to simulate real attack scenarios. The attack chain is the same as that of actual cyber warfare, including information collection, boundary breakthrough, weapon delivery, and lateral movement. In actual attacks, weak password (simple password and repeated password) cracking, popular vulnerability exploitation, and phishing are the most frequently used.

Drills have rules, such as specifying the time, determining the target system, setting the score, and limiting the impact scope. However, there are no rules for actual attacks. So the regular security construction requirements must be higher than the specific requirements of attack-defense drills to truly improve the overall capability.

- Attack tactics analysis:
  - 0-day attacks, backdoor exploitation, VPN vulnerabilities, phishing, and social engineering attacks. With the increase of attack intensity and investment of attack resources, the proportion of 0-day attacks increases, and the diversity of social engineering attacks increases. Most attacks are intranet penetration attacks, and there are few direct intrusions. The overall attack tactics are closer to real network intrusions and comply with the goal of "attack-driven prevention".
- Changes in defense points
  - Single-point protection is changed to multi-point collaborative protection; large-scale blacklist interception is changed to collaborative protection; border in-depth interception is extended to intranet abnormal monitoring; passive monitoring and defense are changed to proactive deception and source tracing.
- Changes in user requirements

 Copyright © Huawei Technologies Co., Ltd.

- Product level: In addition to traditional intrusion prevention, WAF, and vulnerability scanning, the requirements for asset mapping, APT detection, security intelligence, and honeypots is increasing.
- Service level: In addition to security hardening and attendance services during the assurance period, the requirements for routine security inspection, security training, and internal attack-defense drills are increasing.

3.9.2.5 SEC10-05 Establishing a Review Mechanism

It helps the team learn from past security incidents and improve future security measures.

- **Risk level**

  Medium

- **Key strategies**

  a. Determine the purpose: It is important to specify the purpose before the review. Determine what you want to learn from this security incident and how to improve future security measures.

  b. Collect facts and data: Collect all information and data related to the security incident. Use the 5W2H method to sort out the incident, including the time, place, owner, process, cause, and impact of the incident.

  c. Set up a review team: Invite related team members and stakeholders to participate in the review. Ensure that representatives of all key domains are covered, such as technical personnel and security operations personnel.

  d. Analyze the root cause: Trace the event result to analyze the root cause of the event. Ask "why" more often to find out the root cause of the event. This helps to avoid similar events in the future.

  e. Identify errors and deficiencies: Identify errors and deficiencies that occur during security incidents. This includes technologies, processes, and personnel.

  f. Develop improvement measures: Develop specific improvement measures and action plans based on the review results. These measures include people, processes, and technologies. Ensure that these measures are feasible, specific, and can effectively solve problems.

  g. Implement improvement measures: Implement the improvement measures and monitor the implementation. Ensure that all relevant personnel understand and comply with these improvement measures.

  h. Regular review and update: Regularly review the results and the implementation of improvement measures, and update and adjust them as required. Continuous improvement is a lasting process.

  i. Documentation and sharing: The review results and improvement measures are recorded and shared within the team. This helps ensure that everyone can learn from it and avoid similar errors from happening again.

 Copyright © Huawei Technologies Co., Ltd.

j. Training and awareness improvement: Ensure that team members understand the importance of security incident review and are able to actively participate in the review.

3.10 Reference Architecture

To establish secure, trustworthy, and compliant workloads on the cloud, Huawei Cloud offers a comprehensive suite of security-related cloud services. Huawei Cloud customers leverage these services based on the best practices outlined in the Well-Architected framework. During discussions with customers, our solution architects frequently encounter the following inquiries:

- Is there a holistic overview that encapsulates the entire process of building secure workloads?
- Which cloud services are recommended for multi-account versus single-account environments?
- How should workload security be approached from a global to local, top-down, and multi-faceted perspective?

In response to these requirements, we have developed a security reference architecture. This architecture is designed to assist customers in effectively utilizing Huawei Cloud services to construct secure workloads. It provides architectural diagrams from various viewpoints, methodologies for categorizing security cloud services, and guidance on cloud service integration. It is important to note that this security reference architecture serves as a reference paradigm, not an absolute standard. Customers must customize and implement the security reference architecture based on their unique circumstances and specific requirements.

3.10.1 Organization-level Reference Architecture

Huawei Cloud provides the Landing Zone solution to assist enterprise customers in establishing a multi-account environment that is architecturally sound, secure, compliant, and scalable. The foundational step involves planning the organizational and account structure. In accordance with Conway's Law, the organization and account architecture on Huawei Cloud should align with, but not be a direct copy of, the enterprise's internal organizational and business structures.

Huawei Cloud offers a reference architecture. It is advisable to design organization levels and accounts based on your business architecture, geographical distribution, and IT functions.

- Structure different organizational levels and Organizational Units (OUs) on Huawei Cloud based on your business architecture. You can create independent member accounts for each business OU, corresponding to specific business systems. Large-scale business systems or those with stringent security isolation requirements (e.g., PCI-DSS and HIPAA compliance) should be allocated to an independent member account. Conversely, multiple smaller business systems with lower security isolation needs can share a single member account. For example, you might create independent member accounts for major business systems like a sales management system and a digital marketing system. For the R&D department, you could deploy the design and R&D systems of a single product within one member account.

 Copyright © Huawei Technologies Co., Ltd.

- Structure different organizational levels and OUs on Huawei Cloud based on your geographical presence. You can create independent member accounts for each geographical region OU, typically corresponding to a country or specific region. Deploy local customer relationship management systems and customer service systems within these member accounts. In the reference architecture provided, the organization in the China region is mapped to an OU, with independent member accounts created for branches such as Beijing and Shanghai to host localized application systems.
- For an enterprise's IT department, create an OU on Huawei Cloud and establish member accounts based on IT functions. This approach ensures isolated responsibilities and permissions within the IT management domain and facilitates unified management of multiple member accounts across the enterprise. In the provided reference architecture, two OUs are created: a "security OU" housing an account for security operations and log auditing, and an "infrastructure OU" containing an account for network operations, O&M monitoring, public services, and sandbox testing. The following table further describes these IT function accounts.
- In addition to the aforementioned accounts, each organization maintains a single management account. It is strongly recommended not to deploy any cloud resources within the management account. The management account primarily handles the following management tasks:
  - Unified organization and account management: Create and manage organizational structures and OUs, establish cost accounts under OUs, or invite existing accounts to become member accounts of OUs.
  - Unified financial management: Centrally manage all enterprise accounts on Huawei Cloud, including unified budget management, bill management, cost settlement, and cost analysis.
  - Unified control policy management: Set service control policies (SCPs) for each organizational unit and member account to define the maximum permissions for IAM users (including administrators of member accounts) under that member account. This prevents security risks arising from excessive permissions. When creating an SCP, you can apply it to an organizational unit, allowing the policy to be inherited by associated member accounts and lower-level organizational units.
  - Unified identity and permission management: Perform centralized user identity management and permission configuration for all enterprise accounts on Huawei Cloud. This also includes the unified setup of federated identities with external Identity Providers (IdPs).

The table below details the accounts described previously. The security operations account and log account are responsible for centralized security management across all enterprise accounts. Consequently, you must enable relevant security cloud services for other accounts as needed, allowing the security operations account and log account to aggregate security posture data and audit logs from those accounts. Furthermore, security policies and incident response directives from the security operations account can be uniformly disseminated to other accounts. To protect the cloud services enabled under each respective account, it is advisable to also enable relevant security cloud services directly within that account, as illustrated in the rightmost column of the table.

 Copyright © Huawei Technologies Co., Ltd.

Account	Function	Responsible Team	Recommended Cloud Service	Recommended Security Cloud Service
Management account	Centrally manage organizations and accounts, finances, governance policies, and identities and permissions.	IT governance team	Organizations, Resource Governance Center (RGC), Cost Center, and IAM Identity Center	SecMaster, Cloud Trace Service (CTS), and Config
Security operations account	Centrally manage and control security policies, rules, and resources in all accounts of the company, set security configuration baselines for member accounts of the company, and be responsible for information security of the entire company.	Security management team	Deploy services that support cross-account security management and control, such as SecMaster, Host Security Service (HSS), Data Security Center (DSC), Data Encryption Workshop (DEW), Cloud Certificate Manager (CCM), CodeArts Inspector, and Config.	CTS
Logging account	Centrally store and view audit logs and security-related logs (such as VPC flow logs and OBS access logs) of all accounts.	Compliance audit team	Cloud Trace Service (CTS), Log Tank Service (LTS), and Object Storage Service (OBS)	SecMaster, DSC, CTS, and Config

 Copyright © Huawei Technologies Co., Ltd.

Account	Function	Responsible Team	Recommended Cloud Service	Recommended Security Cloud Service
O&M monitoring account	Centrally monitor and maintain resources and applications under each member account, manage alarms, handle events, manage changes, and provide O&M security assurance measures	O&M team	Application Operations Management (AOM), Cloud Operations Center (COC), Log Tank Service (LTS), Application Performance Management (APM), and Cloud Bastion Host (CBH)	SecMaster, CTS, and Config
Network operations account	Centrally deploy and manage enterprise network resources (including resources used for protecting network boundaries), and ensure VPC connectivity in a multi-account environment. In particular, manage ingress and egress in a unified manner for Internet and on-premises IDCs.	Network management team	Enterprise Router, Domain Name Service (DNS), NAT Gateway, Elastic IP (EIP), Virtual Private Cloud (VPC), Direct Connect, Cloud Connect, Virtual Private Network (VPN), Cloud Firewall (CFW), Web Application Firewall (WAF), and Anti-DDoS Service (AAD)	SecMaster, CTS, and Config
Public service account	Centrally deploy and manage the enterprise public resources, services, and application systems, and share them with other member accounts of the enterprise.	Public service management team	Image Management Service (IMS), SoftWare Repository for Container (SWR), Scalable File Service (SFS), Object Storage Service (OBS), in-house NTP servers, and in-house Anti-DDoS servers	SecMaster, CTS, Config, HSS, and DSC

 Copyright © Huawei Technologies Co., Ltd.

Account	Function	Responsible Team	Recommended Cloud Service	Recommended Security Cloud Service
Business account	This account is created based on the business architecture and geographical architecture to deploy application systems that support R&D, production, supply, sales, and service domains.	Application DevOps team	Deploy cloud services as required by service systems.	SecMaster, CTS, Config, HSS, and DSC

3.10.2 Workload-Level Reference Architecture

For some small and medium-sized enterprises, a single Huawei Cloud account is sufficient for their IT system management. In such cases, customers deploy all their workloads within that one account.

Huawei Cloud provides a workload-level security reference architecture for a single account.

The security design of this architecture is as follows:

- **Network security**
  - Anti-DDoS Service is used to defend against DDoS attacks.
  - Web Application Firewall (WAF) is used to defend against web attacks.
  - SSL certificates are used for communication encryption.
  - Cloud Firewall (CFW) is implemented between Internet borders and VPCs.
- **Operating environment security**
  - Host Security Service (HSS) protects host and container security.
  - Network ACLs and security groups are used for access control in a VPC.
  - Vulnerability Scan Service (VSS) is used to periodically scan vulnerabilities of cloud resources.
- **Data security**
  - Data Security Center (DSC) ensures data security throughout the data lifecycle.
  - Data encryption is enabled by default.
  - Database Security Service (DBSS) is deployed for key databases.
  - Cloud Backup and Recovery (CBR) is used to prevent loss of key data.
- **Security operations**
  - SecMaster monitors the overall security of the cloud.
  - Services such as Cloud Log Service (LTS), Cloud Trace Service (CTS), Config, and Cloud Eye are used to manage cloud resources.

 Copyright © Huawei Technologies Co., Ltd.

  - Threat Detection Service (TDS) is used to detect malicious activities and unauthorized behaviors in logs of various cloud services.
  - Cloud Bastion Host (CBH) is used for O&M.

3.11 Security Services

Security Services

- **Security governance**
  - **Identity and Access Management (IAM)**: Authenticates identities and securely manage access to your services and resources
  - **Organizations**: Helps you govern multiple accounts within your organization. You can consolidate multiple Huawei Cloud accounts into a single organization and apply governance policies to them.
  - **Resource Governance Center (RGC)**: Helps you set up and govern a secure, scalable multi-account cloud environment.
  - **Resource Access Manager (RAM)**: Provides secure cross-account resource sharing capabilities. You can create resources once in an account and use RAM to share the resources with organizations, organizational units, or accounts.
  - IAM Identity Center helps you centrally manage your workforce identities and their access to multiple Huawei Cloud accounts. You can create identities for your entire enterprise at one go and give them single sign-on (SSO) access with managed permissions.
- **Network security**
  - Cloud Firewall (CFW): A next-generation cloud native firewall with elastic and flexible services, low deployment costs, and easy and efficient O&M.
  - **Anti-DDoS Service (AAD)**: Mitigates DDoS attacks in milliseconds to ensure continuity of your global services based on machine learning, protection policy tuning, and precise identification of DDoS attacks.
- **Threat detection**
  - **Web Application Firewall (WAF)**: Protects web applications, such as websites, from common web attacks, ensuring continuous and stable service running and meeting compliance and regulatory requirements.
  - **Config**: Allows you to centrally search for and evaluate your resources to ensure resource compliance.
  - **Host Security Service (HSS)**: Manages server and container risks and maintains compliance with cyber security standards. Fights intrusions such as ransomware, mining, infiltration, and container escapes.
  - **CodeArts Inspector**: Provides one-stop vulnerability management for software R&D and O&M. This includes real-time continuous asset evaluation and risk quantification, analysis, and handling. It quickly detects and responds to vulnerabilities for defense against potential security threats.
- **Data security**
  - **Data Security Center (DSC)**: The DSC is a next-generation cloud-native data security platform. It offers fundamental data security functionalities,

 Copyright © Huawei Technologies Co., Ltd.

including data classification and grading, sensitive data scanning, data security checks, data watermark tracing, and data masking. The Asset Map feature integrates the status of each phase within the data security lifecycle, providing a comprehensive view of the overall data security posture in the cloud.

  - **Data Encryption Workshop (DEW)**: Provides robust capabilities for key management, credential management, key pair management, and dedicated encryption functions. Its purpose is to ensure data security and key security while simplifying the overall key management process.
  - **Cloud Certificate Manager (CCM)**: Issues cloud certificates and provides lifecycle management for certificates. Currently, CCM provides SSL Certificate Management and Private Certificate Authority (PCA).
  - **Database Security Service (DBSS)**: Based on machine learning and big data analytics technologies, DBSS can audit your databases, detect SQL injection attacks, and identify high-risk operations.

- **Compliance and privacy protection**
  - **Compliance Center**: Provides a wide range of resources to help you better understand and meet global regulatory requirements.
  - **Cloud Trace Service (CTS)**: Collects, stores, and queries resource operation records. You can use these records to perform security analysis, track resource changes, audit compliance, and locate faults.
  - **Config**: Allows you to centrally search for and evaluate your resources to ensure resource compliance.
- **Security operations**
  - **SecMaster**: SecMaster is a next-generation cloud native platform that enables integrated and automatic security operations. You can manage cloud assets, security posture, security information, and incidents in one place and enjoy intelligent threat detection, easy security orchestration, and automatic response.

 Copyright © Huawei Technologies Co., Ltd.

4 Performance Efficiency Pillar

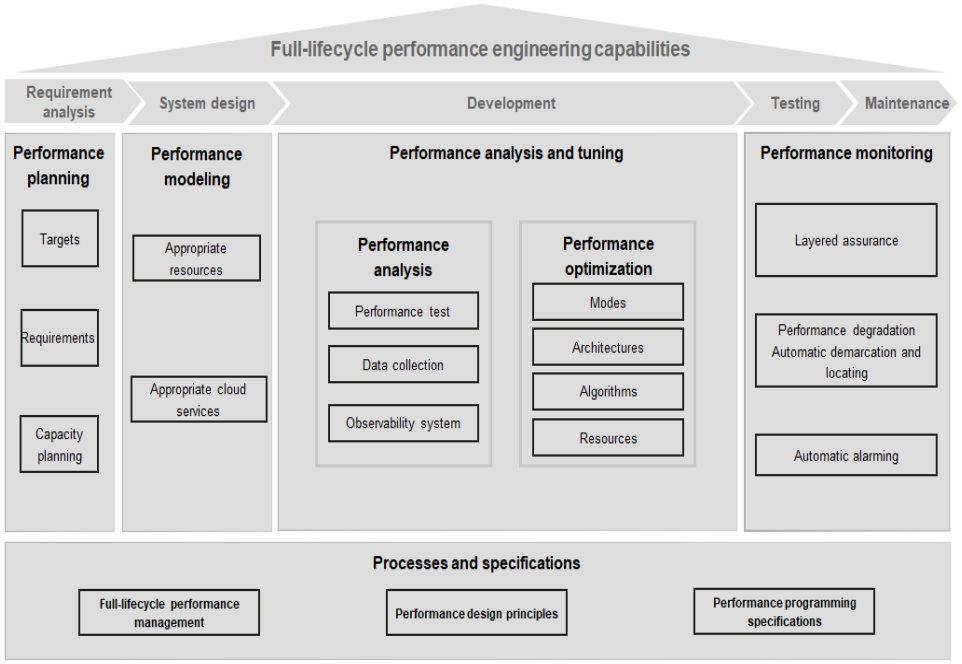
4.1 Introduction

How to design high-performance architecture is a common challenge. Performance is essential as a core quality attribute. The impact of performance issues can be severe. Many companies have faced such problems both internally and externally. This chapter describes some technical methods for implementing software performance engineering and improving system performance.

Early design choices significantly impact the need for and effectiveness of performance tuning. You need to manage performance actively at the beginning of design and development for highly performance-sensitive software.

Additionally, you need efficient technical measures for designing systems, architectures, solutions, and coding.

Figure 4-1 Performance efficiency pillar



 Copyright © Huawei Technologies Co., Ltd.

Many people believe that performance issues come from poor architecture or design rather than inefficient coding. While teams may introduce these issues early in development, they often fix them only during or after integration testing. In fact, performance issues often arise during coding, and fixing architectural problems is far more expensive. The following factors affect system performance:

- System and architecture design: a decisive factor that must be considered early in the design process.
- Implementation solution design: a main factor that can be modified only when there is no need to change the overall architecture.
- Coding implementation: an important factor that can be improved continuously.

Performance issues with systems or components are reflected in these ways:

- It takes too long to respond to requests.
- Too many resources are consumed.

Common performance issues of a system or component typically arise in these areas:

- Communication or invoking between entities (including databases)
- Frequent invoking of functions, issues with stored procedures in modules, and issues with data organizations
- Delay caused by contention for resources during parallel processing
- Waiting time and delay in serial processing of processes or threads

4.2 Concepts

Metric	Description
Performance	The degree to which the response timeliness of a software system or software meets the requirements. Timeliness can be measured by response time or throughput.
Responsiveness	The system's ability to achieve its response time or throughput targets.
Response time (RT)	The total amount of time taken responding to a user's request. It varies depending on the service system, for example, less than 500 ms for Internet services and less than 1s for financial services.
Scalability	The system's ability to continue to achieve its response time or throughput targets when it is required to meet more requirements for software functions.
Throughput (TPS)	The number of transactions, customer requests, or units of data that the system can process per unit of time.
Network bandwidth	The data transmission capability or rate during a specific period.

 Copyright © Huawei Technologies Co., Ltd.

Metric	Description
Network traffic	The amount of data transmitted over the network. It measures either the total data transmitted during a specified period or the data transmission rate on a specific node or connection on the network.
Network latency	The time it takes for data to travel from its sender to receiver.

4.3 Design Principles

The following are common principles for optimizing performance:

- Centralization: Identify key workloads, reduce unnecessary processing, and enhance the most important parts for better performance.
- Localization: Select the resources close to activities, functions, and results, and avoid indirect approaches that may increase traffic or workload and harm performance.
- Resource sharing: Share resources and reduce the contention for resources to improve the overall performance. For example, multiple processes can read the same part of a database simultaneously.
- Parallel processing: Leverage parallel processing to make up for the communication overhead and resource contention delay.
- Load distribution: Handle loads at different time or locations. Divide resources into small separate resource groups and allow different processes or threads to access them independently. Schedule multiple concurrent requests for processing within a time segment.

4.4 Questions and Checklists

Question	Checklist/Best Practice
PERF01 How do you establish processes and specifications?	1. Manage performance throughout the lifecycle 2. Develop application performance programming specifications
PERF02 How do you conduct performance planning?	1. Set performance targets 2. Conduct capacity planning

 Copyright © Huawei Technologies Co., Ltd.

PERF03 How do you conduct performance modeling?	1. Select appropriate compute services 2. Select VMs and container nodes of proper specifications 3. Apply auto scaling 4. Select appropriate networking services 5. Select appropriate storage services 6. Select appropriate message queuing services 7. Select appropriate Kafka specifications 8. Select appropriate RocketMQ specifications 9. Select appropriate RabbitMQ specifications 10. Select appropriate relational databases 11. Select appropriate non-relational databases
PERF04 How do you analyze performance?	1. Define performance acceptance criteria 2. Select appropriate test methods 3. Determine performance test procedure 4. Collect resource performance data 5. Collect application performance data 6. Develop performance observability system
PERF05 How do you optimize performance?	1. Optimize designs 2. Optimize general algorithms 3. Optimize resources for web scenarios 4. Optimize resources for big data scenarios
PERF06 How do you monitor performance?	1. Apply layered assurance 2. Automatically demarcate and locate performance issues 3. Automate alarming

4.5 PERF01 Processes and Specifications

4.5.1 Full-lifecycle Performance Management

4.5.1.1 PERF01-01 Managing Performance Throughout the Lifecycle

Performance activities are integrated into regular stages of the software lifecycle, including requirements, design, development, testing, and maintenance.

- **Risk level**

  High

- **Key strategies**

  Set performance targets

  Set specific, quantitative, measurable performance targets for each scenario. Understand the service requirements and expected quality before setting targets. Collaborate with stakeholders to fulfill function experience requirements instead of only focusing on technical metrics. Define clear performance requirements so you can quantitatively determine if the software achieves the targets accurately. The requirements include:

  – Define clear performance targets.
  – Avoid using qualitative or vague performance targets.
  – Define one or more targets for each performance scenario.
  – Define the appropriate level of granularity for performance metrics.

  Implement adequate functions

  Simplify the function design for components/modules/methods/classes and avoid making inclusive design in the early process. Do not invoke excessive functions or functions that have great impact on performance. Select appropriate cloud service types and specifications to take advantage of cloud elasticity. Designing overly complex components is either for general purposes or a poor practice. Avoid over-design when creating your own functions or invoking existing ones.

  Monitor performance

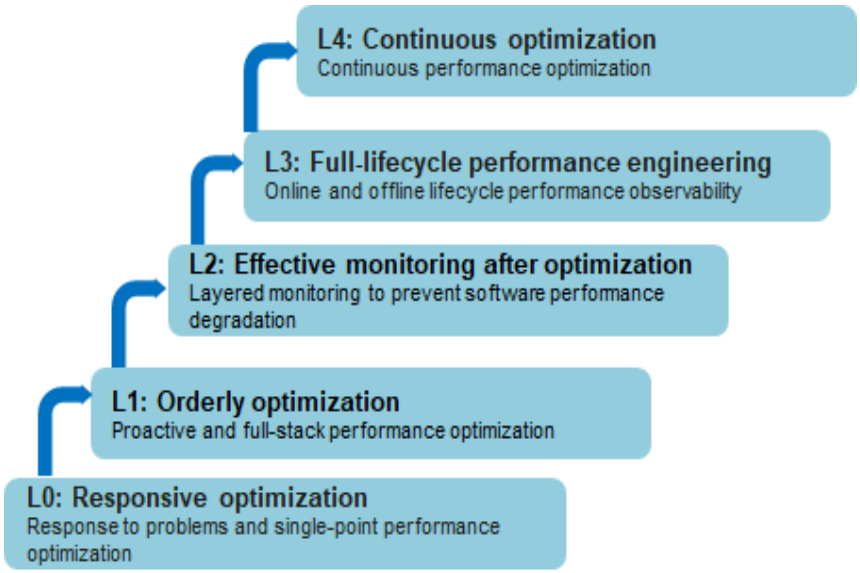
  During development and maintenance, take consistent measures (for example, insert code and detectors at key points) to test and analyze load scenarios, resource requirements, and performance targets. Use monitoring tools to analyze historical trends and identify dominant data flows and code implementation paths. Test performance metrics with commercial tools or consider their testability during design. The data to be tested includes response time, processing capacity, and the frequency of invoking functions and modules.

  Optimize performance for efficiency

  Update your targets as the business grows to meet new challenges and opportunities. Understand how the system works, evolves, and adapts as platforms or technologies change. Allocate enough time for ongoing performance optimization. Foster a culture focused on performance optimization to empower team members to actively monitor performance data. Utilize metric-driven new design patterns and technologies to enhance the system architecture.

 Copyright © Huawei Technologies Co., Ltd.

Figure 4-2 Performance optimization maturity model



4.5.2 Application Performance Programming Specifications

4.5.2.1 PERF01-02 Developing Application Performance Programming Specifications

- **Risk level**

  High

- **Key strategies**

  Improving performance efficiency involves every stage of the development process, including architecting, design, coding, compilation, and running. At the coding level, developers can apply many techniques to boost software performance while keeping code readable and easy to maintain. They can improve software efficiency and product competitiveness by using the high-performance coding best practices integrated into the specifications for different programming languages.

  Strategies for developing high-performance programming specifications:

  – Java: Combine the basic capabilities, concurrency models, deployment optimization, and toolchain assistance.

  – C/C++: Combine the basic capabilities, compilation, concurrency, efficient data structures and algorithms, high-performance libraries, and toolchain assistance.

4.6 PERF02 Performance Planning

4.6.1 Performance Planning

4.6.1.1 PERF02-01 Setting Performance Targets

- **Risk level**

  Medium

- **Key strategies**

  Setting performance targets is crucial for optimizing workload performance efficiency. Performance targets define the required performance levels for workloads and help measure the effectiveness of achieving these targets. They offer a standard for measuring and comparing workload efficiency, helping you highlight improvement areas. These targets align your tasks with your organization's goals, boosting your business success. In addition, performance targets guide resource allocation, helping workloads adapt to different requirements while maintaining optimal performance.

  – Set performance targets early

  Performance targets are key for teams to continuously making improvements. Before service deployment, you need to check system requirements and expected goals and then set performance targets to ensure that the system can meet the reliability and performance requirements without any bottlenecks.

  – Determine performance requirements

    Determine the maximum numbers of users and concurrent requests that meet your business goals.

    - Determine service performance metrics: These metrics include response time, throughput, and number of concurrent users. They must reflect the service requirements.
    - Prioritize services: Determine service priority by importance and urgency for performance testing and optimization.
    - Regularly review and update performance targets: Align your performance targets with new business goals.

  – Identify key performance metrics

  Key performance metrics help measure the status and performance of the workloads related to your business goals. Monitoring these metrics enables you to identify system bottlenecks, such as response time, throughput, and resource usage.

  – Set specific targets

  After determining key metrics, you need to set specific performance targets or thresholds for each metric. This helps you better monitor and manage performance and make optimizations to improve system performance and customer satisfaction.

  For example, you can set the page loading time to no more than 5 seconds for a shopping website. If it takes longer, you can optimize image sizes and reduce HTTP requests to speed up the page loading.

  – Document and expose performance targets

  Meeting performance targets is an ongoing process that needs the development and operations teams to work together. The development team should consider and optimize performance factors during development. The operations team should constantly monitor and improve performance during

 Copyright © Huawei Technologies Co., Ltd.

operations. Document and expose performance targets to help these teams easily check the targets and understand and achieve the common ones they share.

– Evaluate customer feedback

Customer feedback is vital for all organizations. Value customer voice and satisfaction and use them to improve continuously. Regularly collect and analyze customer feedback to understand their needs and expectations. Use this feedback to set your technical standards and improve continuously. Adjust your performance targets based on customers' evolving needs and expectations.

4.6.1.2 PERF02-02 Capacity Planning

- **Risk level**

  Medium

- **Key strategies**

  Capacity planning involves planning and configuring the system's required resources based on service needs and system performance, including the number of users, concurrent requests, and response time requirements. Capacity planning is critical to any organization. Effective capacity planning ensures sufficient resources to meet expected demands and prevents resource waste.

  – Collecting capacity data

    Collecting capacity data helps define technical requirements based on service objectives and it is also essential for forecasting capacity. To meet workload requirements, you need to collect capacity data, including a system's resource usage and key service data.

    - Resource usage data includes the usage of CPUs, memory, disk space, and network bandwidth, which is used to identify system bottlenecks.
    - Key service data includes the number of users, user behavior patterns, service types, and service periods, which is to identify their impact on workloads.

  – Forecasting demands

  Effective capacity planning involves preparing for future service needs. It usually uses workload data to forecast future demands. Forecasting demands is a complex process with multiple factors, including market trends, consumer behavior, and competition environment. The combination of multiple methods, such as historical data analysis, resource analysis, and trend analysis, is used as the basis for forecasting demands. Additionally, artificial intelligence (AI) machine learning algorithms are also used to more accurately forecast future demands and evaluate resource requirements of workloads.

  – Aligning forecasts with workload objectives

  To align the prediction with the workload objectives, forecasting needs to be periodically evaluated to compare the actual results with the forecast results and adjust the capacity prediction model as required. For example, if a new application or service is added to the system, the capacity prediction model needs to consider these new capacity requirements. This alignment ensures

 Copyright © Huawei Technologies Co., Ltd.

adequate resource allocation and prevents resource underutilization and potential overload.

– Determining resource demands

  The results of demand and forecast analysis are used for capacity evaluation and planning. Compute resources, storage resources, and network bandwidth required by the system must be determined to meet system performance demands.

  - Compute resources are selected and configured as required based on the predicted CPU, GPU, and memory demands.
  - Storage resources are calculated based on the predicted requirements. For example, if a large amount of data needs to be stored, a distributed storage system may be required.
  - Network bandwidth is calculated based on the predicted requirements. For example, if large-scale data transmission or real-time network communication is required, a high-speed network may be required.

– Understanding resource restrictions

For capacity planning, it is important to understand limitations on resources, such as processes, threads, CPU usage, memory, and disk space. Resource restrictions are set to ensure system stability. They prevent processes or applications from occupying too many system resources. Otherwise, other processes or applications cannot run properly, or even the system breaks down.

4.7 PERF03 Performance Modeling

4.7.1 Selecting Appropriate Compute Resources

Evaluating compute demands involves the specific compute demands of workloads, including factors such as instance types, scalability, and containerization. Different compute services have different features and characteristics that can impact workload performance. Select the most appropriate compute service to ensure workloads run efficiently. Consider the following strategies:

- Instance types

Different instance types are optimized for different workloads, such as CPU-optimized, memory-optimized, and GPU-optimized. Choose the instance type that meets your needs.

- Auto scaling

If workload demands are unpredictable, consider compute services with auto scaling capabilities that can automatically adjust compute capacity as required. Auto scaling ensures sufficient resources during peak hours and prevents over-allocation during off-peak hours.

- Containerization

 Copyright © Huawei Technologies Co., Ltd.

Containers have performance advantages over non-containerized workloads. If containers fit your architecture requirements, consider containerization. Containers improve compute performance through isolation, resource efficiency, fast startup times, and portability.

When using containers, consider design factors such as containerizing all application components. Linux-based container runtimes are used for lightweight images to provide a short lifecycle for containers so that they are immutable and replaceable. Relevant logs and metrics are collected from containers, container hosts, and basic clusters to monitor and analyze performance. Containers are just one component of the overall architecture. An appropriate container business process coordinator (such as Kubernetes) can enhance performance and scalability.

4.7.1.1 PERF03-01 Selecting Appropriate Compute Services

- **Risk level**

  Medium

- **Key strategies**

  Select appropriate compute cloud services based on application characteristics. The following factors are considered when you select compute cloud services:

  – Application deployment mode

  – Service migration mode (for example, migration or reconstruction to the cloud)

  If the service is deployed on a VM in the Internet data center (IDC), the application system is outdated, and the service is not planned for reconstruction, it is advised to deploy the application in the original IDC deployment mode using ECS or BMS to meet the performance requirements of the application and service.

  If the service side also needs to be containerized and microservices are reconstructed when the service is migrated to the cloud, select containerization (CCE/CCI) for deployment.

  If the service is microservice-based, it is advised to use the container service (CCE/CCI) for deployment.

- **Related services and tools**

  – ECS

  – BMS

  – CCE

  – CCI

4.7.1.2 PERF03-02 Selecting VMs and Container Nodes of Appropriate Specifications

- **Risk level**

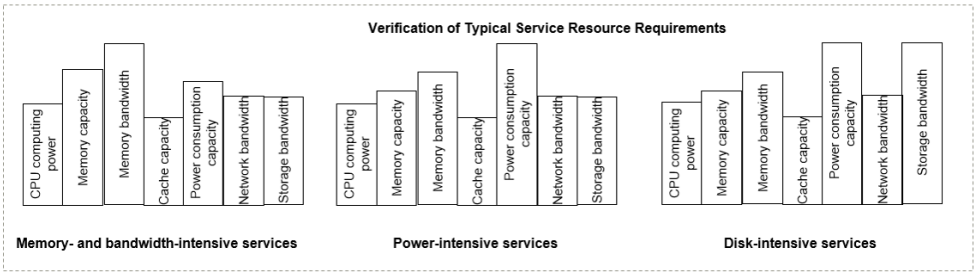
  Medium

- **Key strategies**

Server resources function like a bucket composed of multiple components. The capacity to handle peak demands is determined by whichever resource hits its limit first.

Different applications have different resource demands. For example:

- Power-intensive services (such as high-performance computing, artificial intelligence, and deep learning) consume computing capacity.
- Memory-intensive services (such as big data processing, image/video processing, game development, and database) consume memory and storage capacity.
- Storage-intensive services (such as large databases, big data analysis, large-scale file storage, and compilation and building) may consume storage bandwidth.



Select appropriate VM types and specifications based on service characteristics

For details about the VM types and specifications, see the **official documents**.

- **Related cloud services and tools**
  - ECS
  - BMS

4.7.1.3 PERF03-03 Applying Auto Scaling

- **Risk level**

  Medium

- **Key strategies**

  For scalable workloads like stateless applications, consider compute services with auto scaling to dynamically adjust resources as needed. Auto scaling ensures sufficient resources during peak hours and prevents over-allocation during off-peak hours. While both VM and container auto-scaling adjust application capacity dynamically, containers offer faster response times and better resource efficiency compared to VMs. Use Auto Scaling (AS) in VM scenarios and use Cluster AutoScaling (CA) and Horizontal Pod Autoscaling (HPA) in container scenarios.

  The following describes the auto scaling policies applicable to containers:

  CCE supports auto scaling for workloads and nodes.

  - **Workload scaling**: auto scaling at the scheduling layer to change the scheduling capacity of workloads. For example, you can use HPA, a scaling component at the scheduling layer, to adjust the number of pods used for an application. Adjusting the number of pods changes the scheduling capacity occupied by the current workload, thereby enabling scaling at the scheduling layer.

 Copyright © Huawei Technologies Co., Ltd.

- **Node scaling**: auto scaling at the resource layer. When the planned cluster nodes cannot allow workload scheduling, ECS resources are provided to support scheduling.

Workload scaling and node scaling can work separately or together.

For details, see **Using HPA and CA for Auto Scaling of Workloads and Nodes**.

The following table introduces the workload scaling components

Type	Component	Description	Reference
HPA	**Kubernetes Metrics Server**	It is a built-in component of Kubernetes and enables horizontal scaling of pods. It is based on Kubernetes HPA, but it also has a cooldown time window and scaling thresholds for applications.	**HPA Policy**
CustomedHPA	**CCE Advanced HPA**	An enhanced auto scaling feature, used for auto scaling of Deployments based on metrics (CPU usage and memory usage) or at a periodic interval (a specific time point every day, every week, every month, or every year).	**CustomedHPA Policy**
Prometheus	• **Prometheus** • **Cloud Native Cluster Monitoring**	An open-source system monitoring and alarm framework, which collects public metrics (CPU usage and memory usage) of kubelet in the Kubernetes cluster.	N/A

 Copyright © Huawei Technologies Co., Ltd.

Type	Component	Description	Reference
CronHPA	CCE Advanced HPA	CronHPA can scale in or out a cluster at a fixed time. It can work with HPA policies to periodically adjust the HPA scaling scope, implementing workload scaling in complex scenarios.	CronHPA Policy

Node scaling components are described as follows:

Component	Description	Scenario	Reference
CCE Cluster Autoscaler	An open source Kubernetes component for horizontal scaling of nodes, which is optimized by CCE in scheduling, auto scaling, and costs.	Online services, deep learning, and large-scale computing with limited resource budgets	Node Scaling
CCE Cloud Bursting Engine for CCI	Used to extend Kubernetes APIs to serverless container platforms (such as CCI), which means you no longer have to worry about node resources.	Online traffic surge, CI/CD, big data, and more	Elastic Scaling of CCE Pods to CCI

- **Related cloud services and tools**
  - AS
  - CCE
  - CCI

4.7.2 Selecting Appropriate Network Services

The following performance metrics need to be considered for evaluating appropriate network cloud services:

 Copyright © Huawei Technologies Co., Ltd.

- Network traffic: The expected network traffic of the workload is evaluated to understand the data transmission requirements and the frequency of network requests.
- Bandwidth requirements: The requirements are determined based on the amount of data transmitted and received over the network.
- Network latency: The latency required by the workload is evaluated. Using dedicated virtual networks and backbone networks, instead of traversing the public Internet, can effectively reduce the latency of the workload.
- Throughput: The throughput required by the workload must be considered. Configure network routing options to take advantage of network throughput.

4.7.2.1 PERF03-04 Selecting Appropriate Networking Services

- **Risk level**

  Medium

- **Key strategies**

  Select an appropriate type of network service based on network characteristics.

Scenario	Huawei Cloud Service	Selection Strategy
Cloud networking (intra-cloud and inter-cloud)	VPC	Defines and starts Huawei resources in a logically isolated virtual network to facilitate internal network management and configuration.
	Enterprise Router	Connects a VPC and a local network to a gateway, and supports route learning, dynamic route selection, and path changes, greatly improving network scalability and O&M efficiency and ensuring service continuity.
	Enterprise Switch	Connects a VPC and a local network to a gateway (layer 2 connection) to help enterprises flexibly build large-scale, high-performance, and highly reliable cloud and on-premises networks.
	NAT gateway	Allows multiple cloud servers to share a private IP address to access the local data center or other VPCs, and allows cloud servers to provide services for private networks.
Application networking (between users and the cloud)	ELB	Distributes HTTP/HTTPS traffic to expand the external service capabilities of application systems and improve the fault tolerance of applications.
	VPCEP	Connects a VPC to Huawei Cloud services without exposing data to the Internet. The network is more powerful and flexible.

 Copyright © Huawei Technologies Co., Ltd.

Scenario	Huawei Cloud Service	Selection Strategy
Network access (between users and PoP)	DNS	Provides highly available and scalable authoritative DNS services and DNS management services, routing end users to Internet applications in a reliable and cost-effective manner.
	EIP	Provides independent public IP addresses to connect the public Internet and VPC virtual networks. EIP can be flexibly bound to or unbound from resources such as ECSs, BMSs, virtual IP addresses, load balancers, and NAT gateways.
	Global Accelerator	Uses the Huawei Cloud global network to improve application availability, performance, and security, so that end users can quickly access cloud applications around the world and obtain high-quality experience.
	Direct Connect	Provides a high-speed, low-latency, stable, and secure dedicated connection channel between the local data center and Huawei Cloud VPC, implementing flexible, integrated, and scalable hybrid cloud computing environments.
	VPN	Securely connects to Huawei Cloud or on-premises resources and seamlessly extends existing data centers to Huawei Cloud.
Hybrid networking	Cloud Connect	Provides the capabilities of building, managing, and monitoring global WANs, helping users build a global cloud network with enterprise-scale and communications capabilities.

4.7.3 Selecting Appropriate Storage Services

Learn data characteristics (such as shareability, size, access mode, latency, throughput, and data durability) to select appropriate dedicated data storage for your workloads.

4.7.3.1 PERF03-05 Selecting Appropriate Storage Services

- **Risk level**

  Medium

- **Key strategies**

  During architecture design, select an appropriate storage service based on service scenarios and data characteristics. There are three types of storage available for you to choose from: Elastic Volume Service (EVS), Scalable File

 Copyright © Huawei Technologies Co., Ltd.

Service (SFS), and Object Storage Service (OBS). Their differences are described in the following table.

Dimension	SFS	OBS	EVS
Concept	Provides on-demand high-performance file storage, which can be shared by multiple cloud servers. It is similar to a remote directory for Windows or Linux OSs.	Provides massive, secure, reliable, and cost-effective data storage for users to store data of any type and size.	Provides scalable, high-performance, high-reliability, block storage that can be used to meet a wide variety of service requirements. An EVS disk is similar to a hard disk on a PC.
Data storage logic	Stores files. Data is sorted and displayed in files and folders.	Stores objects. Files are saved directly to OBS. The files automatically generate corresponding system metadata. You can also customize the metadata if needed.	Stores binary data and cannot directly store files. To store files, you need to format the disk with a file system first.
Access method	SFS file systems need to be mounted to ECSs or BMSs through the Network File System (NFS) or Common Internet File System (CIFS) protocol before they can be accessed. A network address must be specified or mapped to a local directory for access.	Accessible through the Internet or Direct Connect. The bucket address must be specified for access, and transfer protocols HTTP and HTTPS are used.	EVS disks can only be used and accessed from applications after being attached to ECSs or BMSs and initialized.

 Copyright © Huawei Technologies Co., Ltd.

Dimension	SFS	OBS	EVS
Scenario	High-performance computing (HPC), media processing, file sharing, content management, and web services Note: HPC: High bandwidth is required for shared file storage, such as gene sequencing and image rendering.	Big data analytics, static website hosting, online video on demand (VoD), gene sequencing, and intelligent video surveillance	HPC, enterprise core cluster applications, enterprise application systems, and development and testing Note: HPC: High-speed and high-IOPS storage is required, such as industrial design and energy exploration.
Capacity	PiB-level	EiB-level	TiB-level
Latency	3–10 ms	10 ms	Sub-milliseconds
IOPS/TPS	10,000 per file system	Tens of millions	128,000 for a single disk
Bandwidth	GiB/s	TiB/s	MiB/s
Data sharing	Yes	Yes	Yes
Remote access	Yes	Yes	No
Used independently	Yes	Yes	No
Hyperlink	**SFS official website**	**OBS official website**	**EVS official website**

 Copyright © Huawei Technologies Co., Ltd.

4.7.4 Selecting Appropriate Application Middleware Services

Huawei Distributed Message Service (DMS) for Kafka, RocketMQ, and RabbitMQ are available. You can select an appropriate message queue based on your service demands and version advantages.

4.7.4.1 PERF03-06 Selecting Appropriate Message Queuing Services

- **Risk level**

  Medium

- **Key strategies**

  The application scenarios of the three DMS versions are as follows:

  DMS for Kafka is compatible with open-source Kafka. It is a fully-managed service that delivers high reliability, concurrency, and scalability in real-time data transmission, stream data processing, system decoupling, and traffic balancing use cases.

  DMS for RocketMQ is compatible with open-source RocketMQ. It provides ordered message delivery, intentional delivery delay, message retry, dead letter messages, transactions, and session messages for e-commerce and finance scenarios.

  DMS for RabbitMQ is compatible with open-source RabbitMQ. It provides message broadcast, transactional messages, message routing, dead letter queues, and priority queues for flash sales, flow control, and system decoupling scenarios.

  For details about the version comparison, see the **official documents**.

- **Related cloud services and tools**
  - DMS for Kafka
  - DMS for RocketMQ
  - DMS for RabbitMQ

4.7.4.2 PERF03-07 Selecting an Appropriate Kafka

- **Risk level**

  Medium

- **Key strategies**

  Calculate the specifications required by services based on indicators such as the production traffic, consumption traffic, aging time, and number of replicas, and select appropriate Kafka specifications.

  Specification calculation

  Min. performance-required nodes = Max.((required storage bandwidth / storage bandwidth per node), (required network bandwidth / assured network bandwidth per node))

  Min. disk-required nodes = Max.(total disk capacity / maximum disk capacity per node)

  For details about how to select specifications, see the **official document**.

 Copyright © Huawei Technologies Co., Ltd.

4.7.4.3 PERF03-08 Selecting an Appropriate RocketMQ

- **Risk level**

  Medium

- **Key strategies**

  RocketMQ specifications are defined on multiple dimensions, such as resources, number of brokers, storage capacity, TPS per broker, maximum number of topics per broker, and maximum number of consumer groups per broker. You are advised to select an appropriate RocketMQ version and specification based on your needs.

  For details about the versions and their specifications, see the **official document**.

4.7.4.4 PERF03-09 Selecting an Appropriate RabbitMQ

- **Risk level**

  Medium

- **Key strategies**

  – Version selection: The RabbitMQ iterates with time and there are many versions. Pay attention to their status and differences when you select a version.

  For details, see **Version Support Notes**.

  – Specifications: The RabbitMQ service provides instances of different specifications. You are advised to select an appropriate specification based on service requirements.

  For details about the instance specifications, see the **official document**.

4.7.5 Selecting Appropriate Database Resources

Huawei Cloud database services vary in optimization methods and precautions. Select appropriate database resources by considering:

- Compatibility. Generally, you should select databases that are compatible with the original ones. The purpose of migrating databases to the cloud is to enable production work to focus more on the application layer. Before migrating databases to the cloud, it should be ensured that the databases in the system have been tested by real services. To avoid incompatibility between the database layer and application layer after migration, you should use cloud databases with the same ecosystem as the original ones.
- Migratability. The solution must support smooth migration. A key concern is to use the capabilities provided by the database migration service to evaluate the impact of database switchover on other components in the service system (such as service interruption and data transfer efficiency).
- Service scenario evaluation. To create a cloud-based service system, select a cloud database that meets your needs first. The evaluation of required databases is based on the features of the real service system, not on whether the database should be created on the cloud. For example, for an e-commerce system, select MySQL for user and buyer information management and transaction processing, and MongoDB for product information management.

 Copyright © Huawei Technologies Co., Ltd.

- Architecture design (performance, reliability, multi-region deployment, and security) constraints. They are the specific requirements of the service system in every dimension, and determine the requirements for database specifications and resource quantity. Grasping the requirements for service performance and reliability and evaluating the performance load capability and load stability of the cloud database can help you better select the database service that meets service requirements. When evaluating the required resource quantity, analyze the requirements of key indicators (such as the number of database connections and transaction processing performance) and the constraints of the deployment design (such as DR requirements). In terms of security, evaluate each design constraint (such as access control and data encryption) to determine whether the database cloud service meets the requirements.

4.7.5.1 PERF03-10 Selecting Appropriate Relational Databases

- **Risk level**

  Medium

- **Key strategies**

  Huawei Cloud provides multiple relational database services, including GaussDB, TaurusDB, and RDS (for MySQL, PostgreSQL, SQL Server, and MariaDB). GaussDB and TaurusDB are enterprise-level relational databases launched by Huawei based on the openGauss and MySQL ecosystems, respectively, offering better performance. RDS supports four open-source database engines and provides users with more stable, elastic, and easy-to-maintain cloud services via the cloud computing platform.

  Application scenarios:

  - GaussDB: enterprise-level distributed relational database based on the openGauss ecosystem led by Huawei. It features high availability and reliability on the cloud, security, auto scaling, one-click deployment, and quick backup and restoration. It is suitable for transaction applications with high concurrency, large data volume, and online transaction processing (OLTP), such as government affairs, finance, e-commerce, O2O, and telecom CRM/billing.
  - TaurusDB: Huawei-developed next-generation enterprise-level cloud native relational database. It is fully compatible with MySQL and features high performance, high scalability, high reliability, high compatibility, low cost, and non-middleware architecture. It is suitable for industries that require high security or high performance, such as the finance and gaming industries.
  - RDS: It features low cost, high performance, high security, and high reliability.
  - RDS for MySQL: MySQL is the most widely used open-source relational database. RDS for MySQL is suitable for website services, applications, and small- and medium-sized enterprises.
  - RDS for PostgreSQL: It is an open-source object cloud database management system that focuses on scalability and standards compliance. It is suitable for website services, location application systems, and complex data object processing.
  - RDS for SQL Server: Microsoft SQL Server is a commercial database with a mature enterprise-level architecture, and can easily cope with various

 Copyright © Huawei Technologies Co., Ltd.

complex environments. RDS for SQL Server is suitable for government, finance, healthcare, education, and gaming industries.

- RDS for MariaDB: MariaDB is one of the mainstream open-source community databases and is quite compatible with MySQL. RDS for MariaDB is suitable for applications of various scales.

Suggestions on selecting relational databases:

Principle 1: Compatibility-based

To ensure smooth migration to the cloud, you are advised to select the relational databases having the same ecosystem as the original databases. This prevents incompatibility between the database layer and application layer and negative impacts on other components in the service architecture.

Principle 2: Requirement-based

To create a service system on the cloud or select a service from different services of the same database, you are advised to consider the actual service requirements. For example, when selecting RDS for MySQL or TaurusDB, consider as many factors as possible, such as performance. For details about these factors, see the official documents.

4.7.5.2 PERF03-11 Selecting Appropriate Non-Relational Databases

- **Risk level**

  Medium

- **Key strategies**

  Huawei Cloud offers two non-relational database services: Document Database Service (DDS) and GeminiDB.

  - DDS is a MongoDB-compatible database service that is secure, highly available, reliable, scalable, and easy to use. It provides DB instance creation, scaling, redundancy, backup, restoration, monitoring, and alarm reporting functions with just a few clicks on the DDS console. It is applicable to games, IoT services, and Internet applications.
  - GeminiDB Redis API is a cloud-native, Redis-compatible NoSQL database service with a decoupled compute and storage architecture developed by Huawei. It stores multiple copies of data persistently in a shared storage pool to secure data and ensure strong consistency. It features high compatibility, cost-effectiveness, reliability, auto scaling, high availability, and lossless scaling. It is suitable for games, recommendation, GIAmgGAP, and push scenarios.
  - GaussDB Influx API is a cloud-native, InfluxDB-compatible NoSQL database service with a decoupled compute and storage architecture. It supports highly-concurrent I/O requests, compressed storage, SQL-like queries, multi-dimensional aggregation computing, and GUI-based data analysis. It features high write performance, robust scalability, high compression rate, and high query performance. It is suitable for IoT, finance, real-time monitoring of software and hardware devices, and data collection.
  - GeminiDB Cassandra API is a cloud-native in-house NoSQL database service with decoupled storage and compute. It is compatible with the Cassandra ecosystem and supports Cassandra Query Language (CQL), which gives you SQL-like syntax. It features security, reliability, high read/

write performance, elastic scaling, and easy management. It is suitable for Internet applications and industrial data collection.

- GeminiDB Mongo API is a cloud-native NoSQL database based on the Huawei-developed architecture with decoupled compute and storage. It is compatible with the MongoDB ecosystem. It features enterprise-level performance, robust scalability, high reliability, and visual management. It is widely used in game applications.
- GeminiDB HBase API is a distributed NoSQL database compatible with the HBase ecosystem. It is extended and optimized based on Apache HBase. It features high performance, high reliability, powerful and flexible scalability. It is suitable for finance, telecommunications, logistics, and games scenarios.

Similar to relational databases, the selection of non-relational databases is also based on compatibility and scenario evaluation.

Principle 1: Compatibility-based

To ensure smooth migration to the cloud, you are advised to select the relational databases having the same ecosystem as the original databases. This prevents incompatibility between the database layer and application layer and negative impacts on other components in the service architecture.

Principle 2: Requirement-based

To create a service system on the cloud or select a service from different services of the same database, you are advised to consider the actual service requirements, such as performance and security. For details about the product and specifications, see the official documentation.

4.8 PERF04 Performance Analysis

4.8.1 Performance Testing

Performance testing is a type of software testing that tests the system with the normal, peak, and abnormal loads simulated by performance test tools to measure response time, throughput, resource usage, stability, and other performance indicators. Run-time system performance is evaluated and verified to ensure that the system functions can meet the baseline requirements. This helps developers detect performance problems and prevent performance from deteriorating below the baseline. The following sections describe performance test.

4.8.1.1 PERF04-01 Defining Acceptance Criteria

- **Risk level**

  High

- **Key strategies**

  Acceptance criteria are used to evaluate whether a specified workload meets performance requirements. They should be defined properly before performance testing.

  - Checking performance targets

  Performance targets define the performance level required for a workload. Check the performance targets set for the workload. They may include

 Copyright © Huawei Technologies Co., Ltd.

response time, throughput, resource usage, or any other performance indicators. For example, the response time may be expected to be less than 2 seconds.

– Defining acceptance criteria

Convert performance targets into specific acceptance criteria that can be used to evaluate the performance. For example, if the response time is expected to be 2 seconds or less, the acceptance criterion can be 2 seconds or less for the average response time under the workload. Use the acceptance criteria to determine whether the performance level meets the requirements under the workload.

4.8.1.2 PERF04-02 Selecting Appropriate Test Methods

- **Risk level**

  High

- **Key strategies**

  The following lists the common performance test methods. Note that they are not orthogonal but coupled.

  – Performance acceptance: Performance acceptance testing verifies whether the system meets the claimed capability requirements in a specified running environment.

  – Load testing: The load on the system is increased tier by tier until the system reaches its performance limit. Generally, load testing is used to test the system performance capacity or for optimization.

  – Stress testing: It is to check the system's flow control mechanism and performance of the system under extreme loads.

  – Long-term stability testing: It is performed under certain load stress to test the performance stability. Testers usually run long-term stability test along with stress testing.

  – Configuration testing: It is to check the impact of software/hardware configurations and service models on the tested system's performance. This helps to find the optimal allocation principles of system resources and performance trends of different service models.

  – Concurrency testing: It is to test the system using concurrent requests and tasks to identify performance problems, such as process deadlocks and resource leaks.

- **Related cloud services and tools**

  – CodeArts PerfTest

4.8.1.3 PERF04-03 Performance Test Procedure

- **Risk level**

  High

- **Key strategies**

  1. Determine acceptance criteria

  Organize users, developers, maintenance administrators, and other roles to define the target baselines for service, data, and resource metrics based on the production environment capabilities. You need to enable the system to

 Copyright © Huawei Technologies Co., Ltd.

manage the maximum amount of data with the minimum resources for optimal user experience. You also need to determine SLAs for each scenario.

2. Create test plan

Design specific scenarios or conditions for performance testing under system loads. The performance test plan must be comprehensive, covering all networking scenarios using a design template. Based on performance testing requirements, test models, and network topology diagrams, it outlines key test points and breaks down test scenarios and steps. The test plan simulates real-world user behavior and system loads, providing performance testers with methods to evaluate service performance under various conditions, including test environments, tools, and monitoring metrics.

It enables replication of different load levels, such as concurrent user access, peak traffic periods, or specific scenarios. By testing these load levels, teams can identify performance bottlenecks and optimize resource allocation. The final deliverable is an executable performance test plan.

- Define user behaviors: Identify test scenarios by analyzing frequent user interactions (for example, login, search, batch operations, import/export, form submissions, and feature access) to replicate real-world behavior and load patterns. Break down each scenario into specific interaction steps, covering page flows, transactions, and mixed system load conditions.
- Establish data models: Determine required test background data. Generate realistic datasets with varied scenarios, user profiles, and data volumes to ensure comprehensive performance evaluation.
- Design test scripts: Create executable scripts containing system operations, HTTP requests, or API/UI interactions. Use performance testing tools with parameterization and dynamic data processing. Debug scripts to resolve errors, missing or incorrect actions, or data issues, ensuring reliable test execution.
- Configure test variables and parameters: Set variables and parameters to simulate authentic scenarios, including multi-user logins, data queries, and randomized actions reflecting diverse user behaviors and workload responses.
- Automate scripts: Automate scripts and continuously optimize them based on feedback, test results, or change requirements. Improve script logic, parameterization, and error handling, or add additional validation checkpoints. Configure automated scripts to enable iterative testing without manual script modifications.

3. Configure test environment

Set up a test environment (with the same software version, system networking, and hardware specifications as that of the production environment), and separate the test environment to prevent it from being affected by other factors.

Key considerations include:

- System networking and architecture: active/standby, cluster, or distributed networking and architecture analysis to identify dependency services.
- Hardware specifications: required number of servers and their specifications, including CPU frequency/cores, memory capacity, disk type and capacity, storage pool type and capacity, and network bandwidth.

 Copyright © Huawei Technologies Co., Ltd.

- Software environment: software versions and configurations, such as operating system version, service version, database version, and other performance-related configurations.

4. Complete test design

Design the preceding system loads, background data volumes, and data models for the tests.

5. Perform the tests

Execute performance tests using the selected tools. The process involves monitoring and recording key performance metrics, observing system behavior, and checking performance issues. Track and collect metrics, including response time, throughput, and CPU/memory utilization.

Apply the defined test plan to subject the workloads to expected levels. Conduct testing under varying load conditions, including normal, peak, and stress levels, to analyze workload behavior across different scenarios.

6. Analyze the results

Examine performance metrics collected from the performance tests to identify bottlenecks. For a performance issue, submit a ticket for optimization. Key aspects include:

- Review performance metrics: Evaluate metrics including response time, throughput, error rate, CPU/memory utilization, and network usage to assess overall service performance.
- Identify performance bottlenecks: Analyze metrics to pinpoint constraints such as high response time, resource utilization, database issues, network latency, and scalability limitations, guiding targeted optimizations and capacity adjustments.
- Correlate performance metrics: Examine the relationship between metrics, for example, how background data volume and resource utilization affect response time, to obtain valuable guidance for different scenarios, and determine which nodes require scaling when needed.
- Evaluate acceptance conditions: Compare results against predefined SLAs to determine production readiness. Implement optimizations or scaling when requirements are unmet.

7. Determine the baseline

A performance baseline is a baseline library established when the product is formed. The performance baseline is archived and released in the version design phase based on the layering principle. The baseline provides a reference point for comparing performance results within a period of time. The baseline should be the reference of the workload performance.

Consider the workload objectives and record the performance so that the baseline can be compared and optimized as the version iterates. Use these baseline metrics as the benchmark for future performance tests and use them to identify whether the service system has performance degradation.

To establish a performance baseline for future benchmarking, follow these steps:

- Determine performance metrics: Identify and agree upon metrics, such as:

  Response time, or the speed at which a service responds to a request

  Throughput, or the number of requests processed per unit of time

 Copyright © Huawei Technologies Co., Ltd.

Resource utilization, such as CPU, memory, and disk usage

- Record performance-related metrics: Record the performance metrics obtained during the test as baseline metrics. Compare them with the pre-defined SLA values.
- Compare future test results: In subsequent tests, evaluate performance metrics against the established baseline and thresholds. This comparison helps to detect potential performance degradation.

Principles and requirements for managing performance baselines:

- Baseline immutability: Published baselines cannot be modified within the same version. Any changes require alignment with product SEs and PMs, followed by official re-release.
- Baseline inheritance: Baseline values must be validated during each version's design phase. Complete overhaul of existing baselines is prohibited.
- Baseline ownership: Testing teams must strictly adhere to published baseline requirements for performance tests and evaluations.
- Validation rights: Teams may reject verification requests for non-baseline tests that lack product relevance.

Testing teams have the right to challenge unreasonable baseline content. Based on test data, live network applications, customer specifications, and standards, they may initiate reviews with SEs and product managers for baseline adjustments.

4.8.2 Collecting Performance Data

Collect metrics and logs that provide information about workload performance. Metrics refer to data including numerical values. They describe the system's status at specific time points and encompass logs containing various types of data organized into records.

Collected performance data enables workload monitoring and analysis. This information serves to identify performance bottlenecks, solve issues, optimize resource allocation, and support data-driven decisions for higher overall workload efficiency.

Without data-driven insights, potential performance issues and optimization opportunities may go undetected. This can lead to longer response time, reduced throughput, higher resource utilization, and degraded user experience. Additionally, insufficient performance metrics hinder timely issue diagnosis and troubleshooting, resulting in prolonged downtime and decreased operational efficiency.

4.8.2.1 PERF04-04 Collecting Resource Performance Data

- **Risk level**

  Medium

- **Key strategies**

  Each Huawei Cloud service provides a set of resource-specific metrics that reflect resource utilization. Collecting these performance metrics enables in-depth analysis of workload operations and behavior.

  Metrics help you:

  - Understand resource status and performance.
  - Configure alarm policies and metric dashboards in Cloud Eye.
  - Optimize the network performance by tracking and analyzing traffic on network paths.
- **Related cloud services and tools**
  - Cloud Eye

4.8.2.2 PERF04-05 Collecting Application Performance Data

- **Risk level**

  Medium
- **Key strategies**

  Application performance data (throughput, latency, and completion time) is usually collected by embedding code snippets or integrating tools into application code. The collected application performance metrics enable bottleneck identification, system behavior evaluation, availability risk detection, and capacity planning.

  Common monitoring strategies include:
  - APM tools: Use cloud-based or open-source APM to analyze performance data (metrics, logs, and traces).
  - Log-based tracing framework: Implement frameworks with log generation, formatting, and contextual correlation capabilities. By integrating these into code repositories, runtime performance data can be collected.
  - Customized detection: Develop custom code to collect unique performance metrics only when platform metrics are insufficient.
  - Use observability standards in the industry. Adopt industry standards–based tools like OpenTelemetry.

  Suggestion: Use the distributed tracing technology to identify request links between multiple services and components. Collect tracing data to analyze data flows from end to end, and identify bottlenecks or inefficient request segments for further optimization.
- **Related cloud services and tools**
  - Application Operations Management (AOM)
  - Application Performance Management (APM)
  - Log Tank Service (LTS)

4.8.3 Developing a Performance Observability System

4.8.3.1 PERF04-06 Developing a Performance Observability System

- **Risk level**

  Medium
- **Key strategies**

  Observability system is a comprehensive solution in cloud-native architectures that employs various tools and technologies to implement monitoring, alerting, logging, and troubleshooting for applications and infrastructure.

 Copyright © Huawei Technologies Co., Ltd.

Performance observability system focuses on performance metrics. By collecting and analyzing performance data, you can identify system bottlenecks, optimize resource allocation, and find performance optimization directions.

Performance monitoring objects: servers, OSs, databases, applications, network devices, and cloud services.

Common performance metrics: CPU, memory, hard disk, program response time, throughput, and number of concurrent requests.

4.9 PERF05 Performance Optimization

In performance optimization, avoid "premature optimization". Our basic strategy is to first make the system work, then consider how to make it faster. Optimization should generally only be undertaken when we have confirmed a performance bottleneck in specific code sections. Unless specialized tools identify the bottleneck, such efforts may prove wasteful. Additionally, the hidden cost of optimization, that is, reduced code clarity and maintainability, must be carefully weighed.

4.9.1 Design Optimization

4.9.1.1 PERF05-01 Optimizing Design

- **Risk level**

  Medium

- **Key strategies**

  – Fast path

  Reduce the processing load of the dominant workload to improve the response time. A piece of software can have multiple functions, but only a few of them are frequently used. The frequently used functions constitute a dominant workload. The fast path pattern reduces the processing load of these functions or simplifies the processing. This pattern simplifies the execution path

  by identifying the dominant workload, which can be determined based on the usage frequency of a function. Common fast paths include quick navigation keys and database indexes.

  – Prioritizing important tasks

  Preferentially allocate resources to important tasks. If you cannot do everything in the available time, ignore the least important tasks. This strategy handles the situation where the system capacity is exceeded due to instantaneous burst loads. Generally, the most important tasks are processed first. If the overload is not temporary, reduce the processing volume or upgrade the system. In the performance overload scenario, apply circuit breaking to non-critical features to ensure that the main functions are available.

  – Aggregation

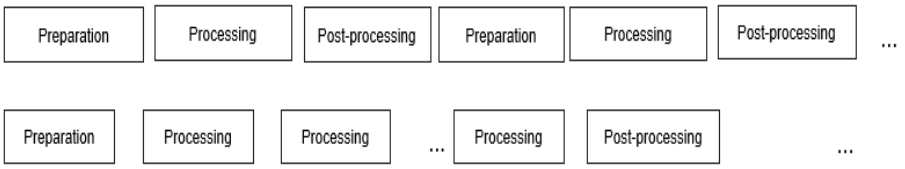
  Combine the functions that are used together in most scenarios to reduce the interactions.

 Copyright © Huawei Technologies Co., Ltd.

This pattern combines some sub-functions that are often called together. Place related or tightly coupled functions in a single object. Use local interfaces instead of external or high-overhead interfaces (such as CORBA interfaces) that present small-granularity objects. The aggregation pattern uses coarse-grained objects. Frequently accessed data should be combined into an aggregate to minimize requests for specific information. For example, the account class CustAcct can provide access functions getName(), getAddress(), and getZip(). If you need to create email tags, use a new function genMailLableInfo() to obtain all information at a time to reduce interactions.

– Batch processing

Combine frequent service requests to reduce the overhead of request initialization, transmission, and termination. When the overhead of initializing, transmitting, and terminating a request is high, the actual process time for the request may be longer than necessary. By combining requests into a batch, the overhead is shared among the requests. This approach improves processing efficiency. Batch saving in the database and Redis pipelines are common batch processing operations.



Batch processing model

– Bypass

Process the requests that require extensive resources on other objects or other locations to reduce contention delay. This pattern is similar to using an alternative route to bypass the traffic bottleneck and reach the destination. Solution 1: Divide the target object into small-granularity objects, and distribute them to different physical locations. Solution 2: Create a separate thread for each data region. Each thread is responsible for updating its assigned data region.

For a process that must collaborate with a single downstream process, if there is only one downstream process, it may struggle to keep up under high loads, causing a bottleneck. You can use multiple downstream processes. Each process updates only its own data.

– Flexible schedule

Process the requests that require extensive resources at different time segments to reduce the contention delay.

A large number of requests are sent out simultaneously at a certain time. As a result, the demand for returned results surges and the response is slow. However, at other times, there are few or no requests. This situation happens when requests are processed at a specific frequency or at a specific time of day. Identify the functions that are processed regularly, then change their processing times. Randomly distribute these processing times within a given time range.

– Space–time trade off

 Copyright © Huawei Technologies Co., Ltd.

Use more storage space to save execution time.

You can pre-store results or store frequently accessed data for easy calculation. You can also select specific algorithms, for example, the hash algorithm. Another method is OLAP. This technology summarizes data based on a certain hierarchical relationship and greatly reduces the time required for subsequent queries.

For example, when optimizing slow SQL statements, you can identify frequently accessed fields and set indexes to shorten the access latency.

– Handling valid workloads

  Identify the data that must be processed and prevent repeated processing of data. You can use multiple methods to reduce the load, such as processing only increments and changes.

  ▪ Incremental processing

  ▪ Change processing upon notification

Processing a large amount of unnecessary data heavily burdens the processing capacity. The issue might arise if the processing is oversimplified or lacks necessity analysis. You can analyze the data to be processed, filter the data that must be processed, and redesign the processing solution.

– Connections and synchronization for serial processing

Design close connections and effective synchronization measures for serial processing of threads or processes, to reduce the synchronous wait delay.

Divide a serial processing into multiple threads for parallel processing, while ensuring that each thread can be handled in a certain sequence and synchronization relationship. To achieve this, you need to design proper connections and synchronization. You can select scheduled query, message synchronization, or event association. The method should bridge the time gaps and meet performance objectives.

Even with this strategy, the overall performance of the system can only reach the performance level of the weakest processing stage (ignoring the overhead of thread switching and semaphore operations).

4.9.2 Algorithm Optimization

4.9.2.1 PERF05-02 Optimizing General Algorithms

- **Risk level**

  Medium

- **Key strategies**

  You can improve the design and implementation of algorithms to improve program performance. The following are some best practices:

  – Use the correct data structure: Selecting a proper data structure can greatly improve the algorithm efficiency. For example, you can use a hash table to quickly search for elements and use an array to quickly access elements.

  – Reduce memory allocation: Memory allocation is a time-consuming operation. You can pre-allocate memory or reuse allocated memory.

 Copyright © Huawei Technologies Co., Ltd.

- Reduce the number of loops: Loops are a common algorithm structure. However, too many loops will deteriorate program performance. You can use more efficient algorithms.
- Use parallel computing: For some compute-intensive algorithms, use parallel computing to improve program performance. You can use multi-thread or distributed computing.

4.9.3 Resource Optimization

4.9.3.1 PERF05-03 Optimizing Resources for Web Scenarios

- **Risk level**

  Medium

- **Key strategies**

  You can optimize the configured resources to improve performance. For example, optimize OS settings, network bandwidth, and database queries.

  Perform the following to optimize resources:

  - Monitor and analyze cloud service resource usage.to identify performance bottlenecks.
  - Adjust the configurations, such as CPU, memory, and network to optimize resource configurations.
  - Use cache technologies (such as CDN and Redis) to improve data access speed.
  - Optimize the code used by cloud service resources to improve code execution efficiency.
  - Optimize database index and query.
  - Use load balancing technology to distribute requests to multiple cloud service resources and improve the system processing capability.
  - Monitor resources performance and adjust resources based on actual situations.

4.9.3.2 PERF05-04 Optimizing Resources for Big Data Scenarios

- **Risk level**

  Medium

- **Key strategies**

  You can optimize resource usage and allocation to improve system performance and efficiency. The following are common methods to optimize resources:

  - Use distributed storage systems, such as Hadoop HDFS and Apache Cassandra, to store data on multiple nodes to improve data reliability and scalability.
  - Compress a large amount of data using compression algorithms to reduce the storage space and transmission bandwidth.
  - Use parallel computing frameworks, such as Apache Spark and Apache Flink, to distribute computing tasks to multiple nodes for parallel execution. This method improves the computing speed and efficiency.

- Optimize the memory allocation and usage policies, such as using memory cache and memory mapping, to improve the data processing and computing speed and efficiency.
- Use the load balancing technology to evenly distribute data and computing tasks to multiple nodes to prevent a single node overload and improve system availability and performance.
- Divide data into multiple partitions according to certain rules to better process and compute data.
- Optimize network parameters such as bandwidth and latency to improve data transmission speed and efficiency.
- Clean and preprocess data to improve data quality and accuracy, and reduce the computing error rate and workload.

4.10 PERF06 Performance Assurance

4.10.1 Performance Assurance

4.10.1.1 PERF06-01 Applying Layered Assurance

- **Risk level**

  High

- **Key strategies**

  Service deployment architecture divides resources into five layers (from the hardware infrastructure at the bottom layer to the applications at the top layer). For cloud services, you only need to focus on the virtual network, instances, and applications layers. Build models for each layer based on their resources characteristics, and set different monitoring metrics for each layer. Generally, you can classify severity levels as minor, major, and critical, each with its corresponding handling measures. For an application architecture that is sensitive or highly important to service reliability, a warning level can be added.

- **Related cloud services and tools:**
  - Cloud Eye
  - Application Operations Management (AOM)
  - Application Performance Management (APM)

4.10.1.2 PERF06-02 Automatically Demarcating and Locating Performance Issues

- **Risk level**

  Medium

- **Key strategies**

  Build layered performance models to determine whether the system performance deteriorates. When a deterioration event occurs, in order to automatically quickly locate and demarcate the root cause, you can apply a three-dimensional topology based on application models to associate

 Copyright © Huawei Technologies Co., Ltd.

architecture, space, and time data. Continuous resource and data governance are important to build architecture models and aggregate and visualize layered metrics.

- **Related cloud services and tools:**
  - Optimization Advisor (OA)
  - Cloud Eye
  - Application Operations Management (AOM)

4.10.1.3 PERF06-03 Automating Alarming

- **Risk level**

  Medium
- **Key strategies**

  Configure alarm policies on Cloud Eye to monitor resource risks in a timely manner and make necessary adjustments.
- **Related cloud services and tools:**
  - Optimization Advisor (OA)
  - Cloud Eye

4.11 Cloud Service Performance Optimization

4.11.1 Optimizing Cache Performance

The following section describes how to optimize cache performance based on specific suggestions and cases.

Suggestions on Using Redis

These suggestions help reduce Redis instability or exception occurrences to ensure a stable system.

Service Usage

Principle	Description	Level	Remarks
Nearest deployment for low latency	If the deployment sites are far from each other (across regions) or the latency is high (server-instance connection over a public network), the read/write performance will be greatly affected.	Required	If your service is latency-sensitive, do not create cross-AZ DCS Redis instances.

Principle	Description	Level	Remarks
Cold/Hot data separation	You can store frequently accessed data (hot data) in Redis, and lesser accessed data (cold data) in MySQL and Elasticsearch databases.	Suggested	Infrequently accessed data stored in the memory occupies Redis space and does not accelerate access.
Service data differentiation	Store unrelated service data in different Redis instances.	Required	This prevents services from affecting each other, and prevents single instances from being too large. This also enables quick service recovery in case of faults.
	Do not use the **SELECT** command for multi-DB on a single instance.	Required	Multi-DB on a single Redis instance causes weak isolation, and is no longer in active development by open-source Redis. You are advised not to depend on this feature in the future.
Proper memory eviction policy	Redis can still function when the memory is used up unexpectedly.	Required	The default policy is volatile-lru. Select a policy as required. **Redis Data Eviction Policies**
Redis use as cache	Do not rely too much on Redis transactions.	Suggested	After a transaction is executed, it cannot be rolled back.
	If data is abnormal, the cache can be cleared for data restoration.	Required	Redis does not have a mechanism or protocol to ensure strong data consistency. Therefore, services cannot over-rely on the accuracy of Redis data.
	When using Redis as cache, set expiration on all keys. Do not use Redis as a database.	Required	Set expiration as required, but a longer expiration is not necessarily better.
Cache breakdown prevention	Use Redis together with local cache. Store frequently accessed data in the local cache and regularly update it asynchronously.	Suggested	-

 Copyright © Huawei Technologies Co., Ltd.

Principle	Description	Level	Remarks
Cache penetration prevention	Non-critical path operations are passed through to the database. Limit the rate of access to the database.	Suggested	-
	If the requested data is not found in Redis, read-only DB instances are accessed. You can use domain names to connect to read-only DB instances.	Suggested	The idea is that the request does not go to the main database. You can use domain names to connect to multiple read-only DB instances. If a fault occurs, you can add such instances for emergency handling.
No use as a message queue	In pub/sub scenarios, do not use Redis as a message queue.	Required	• Unless otherwise required, you are not advised to use Redis as a message queue. • Using Redis as a message queue causes capacity, network, performance, and function issues. • If message queues are required, use Kafka for throughput and RocketMQ for reliability.
Proper specification selection	If service growth causes increases in Redis requests, use Proxy Cluster or Redis Cluster instances.	Required	Scaling up single-node and master/standby instances only expands the memory and bandwidth, but cannot enhance the computing capabilities.
	In production, do not use single-node instances. Use master/standby or cluster instances.	Required	-
	Do not use large specifications for master/standby instances.	Suggested	Redis forks a process when rewriting AOF or running the **BGSAVE** command. If the memory is too large, responses will be slow.
Degradation or disaster recovery measures	When a cache miss occurs, data is obtained from the database. Alternatively, when a fault occurs, allow another Redis to take over services automatically.	Suggested	-

 Copyright © Huawei Technologies Co., Ltd.

Data Design

Category	Principle	Description	Level	Remarks
Keys	Keep the format consistent.	Use the service name or database name as the prefix, followed by colons (:). Ensure that key names have clear meanings.	Suggested	For example: *service name*:*sub-service name*:*ID*.
	Minimize the key length.	Minimize the key length without compromising clarity of the meaning. Abbreviate common words. For example, **user** can be abbreviated to **u**, and **messages** can be abbreviated to **msg**.	Suggested	Use up to 128 bytes. The shorter the better.
	Do not use special characters except braces ({}).	Do not use special characters such as spaces, line breaks, single or double quotation marks, and other escape characters.	Suggested	Redis uses braces ({}) to signify hash tags. For cluster instances, braces in key names must be used correctly to avoid unbalanced shards.
Values	Use appropriate value sizes.	Keep the value of a key within 10 KB.	Suggested	Large values may cause unbalanced shards, hot keys, traffic or CPU usage surges, and scaling or migration failures. These problems can be avoided by proper design.
	Use appropriate number of elements in each key.	Do not include too many elements in each Hash, Set, or List. It is recommended that each key contains up to 5,000 elements.	Suggested	Time complexity of some commands, such as **HGETALL**, is directly related to the quantity of elements in a key. If commands whose time complexity is O(N) or higher are frequently executed and a key has a large number of elements, there may be slow requests, unbalanced shards, or hot keys.

 Copyright © Huawei Technologies Co., Ltd.

Category	Principle	Description	Level	Remarks
	Use appropriate data types.	This saves memory and bandwidth.	Suggested	For example, to store multiple attributes of a user, you can use multiple keys, such as **set u:1:name "X"** and **set u:1:age 20**. To save memory usage, you can also use the **HMSET** command to set multiple fields to their respective values in the hash stored at one key.
	Set appropriate timeout.	Do not set a large number of keys to expire at the same time.	Suggested	When setting key expiration, add or subtract a random offset from a base expiry time, to prevent a large number of keys from expiring at the same time. Otherwise, CPU usage will be high at the expiry time.

Command Use

Principle	Description	Level	Remarks
Exercise caution when using commands with time complexity of O(N).	Pay attention to the value of N for commands whose time complexity is O(N). If the value of N is too large, Redis will be blocked and the CPU usage will be high.	Required	For example, the **HGETALL**, **LRANGE**, **SMEMBERS**, **ZRANGE**, and **SINTER** commands will consume a large number of CPU resources if there is a large number of elements. Alternatively, you can use **SCAN** sister commands, such as **HSCAN**, **SSCAN**, and **ZSCAN** commands.
Do not use high-risk commands.	Do not use high-risk commands such as **FLUSHALL**, **KEYS**, and **HGETALL**, or rename them.	Required	**Renaming Critical Commands**

 Copyright © Huawei Technologies Co., Ltd.

Exercise caution when using the **SELECT** command.	Redis does not have a strong support for multi-DB. Redis is single-threaded, so databases interfere with each other. You are advised to use multiple Redis instances instead of using shared databases for one instance.	Suggested	-
Use batch operations to improve efficiency.	For batch operations, use the **MGET** command, **MSET** command, or pipelining to improve efficiency, but do not include a large number of elements in one batch operation.	Suggested	**MGET** command, **MSET** command, and pipelining differ in the following ways: • **MGET** and **MSET** are atomic operations, while pipelining is not. • Pipelining can be used to send multiple commands at a time, while **MGET** and **MSET** cannot. • Pipelining must be supported by both the server and the client.
Do not use time-consuming code in Lua scripts.	The timeout of Lua scripts is 5s, so avoid using long scripts.	Required	Long scripts: time-consuming sleep statements or long loops.
Do not use random functions in Lua scripts.	When invoking a Lua script, do not use random functions to specify keys. Otherwise, the execution results will be inconsistent between the master and replica nodes, causing data inconsistency.	Required	-

Follow the rules for using Lua on cluster instances.	Follow the rules for using Lua on cluster instances.	Required	• When the **EVAL** or **EVALSHA** command is run, the command parameter must contain at least one key. Otherwise, the client displays the error message "ERR eval/evalsha numkeys must be bigger than zero in redis cluster mode." • When the **EVAL** or **EVALSHA** command is run, a cluster DCS Redis instance uses the first key to compute slots. Ensure that the keys to be operated are in the same slot.
Optimize multi-key operation commands such as **MGET** and **HMGET** with parallel processing and non-blocking I/O.	Some clients do not treat these commands differently. Keys in such a command are processed sequentially before their values are returned in a batch. This process is slow and can be optimized through pipelining.	Suggested	For example, running the **MGET** command on a cluster using Lettuce is dozens of times faster than using Jedis, because Lettuce uses pipelining and non-blocking I/O while Jedis does not have a special plan itself. To use Jedis in such scenarios, you need to implement slot grouping and pipelining by yourself.
Do not use the **DEL** command to directly delete big keys.	Deleting big keys, especially Sets, using **DEL** blocks other requests.	Required	In Redis 4.0 and later, you can use the **UNLINK** command to delete big keys safely. This command is non-blocking. In versions earlier than Redis 4.0: • To delete big Hashes, use **HSCAN** + **HDEL** commands. • To delete big Lists, use the **LTRIM** command. • To delete big Sets, use **SSCAN** + **SREM** commands. • To delete big Sorted Sets, use **ZSCAN** + **ZREM** commands.

 Copyright © Huawei Technologies Co., Ltd.

4.11.2 Optimizing Message Queue Performance

The following document describes how to optimize message queue performance based on specific suggestions and metrics.

4.11.2.1 Optimizing Kafka Performance

Kafka performance optimization

Client configuration optimization

Producer configuration suggestions

For details, see **Setting Parameters for Kafka Clients**.

Consumer configuration suggestions

Parameter	Recommended Value	Description
max.poll.records	500	The maximum number of messages a consumer can consume at a time. The default value is 500. If consumption per message takes longer, use a smaller value and ensure that messages in a batch can be processed as **max.poll.interval.ms** specified.
max.poll.interval.ms	300000	The maximum interval between two consumption pulls. The default interval is 300s. If the interval exceeds 300s, the consumer is considered abnormal.
fetch.min.bytes	Adjusted based on services	The minimum size of data each fetch request returns. The default value is **1**. The bigger the value, the higher the throughput and latency.

Performance metrics

Kafka provides the following performance metrics to help analyze message accumulation, partition data skew, and traffic imbalance.

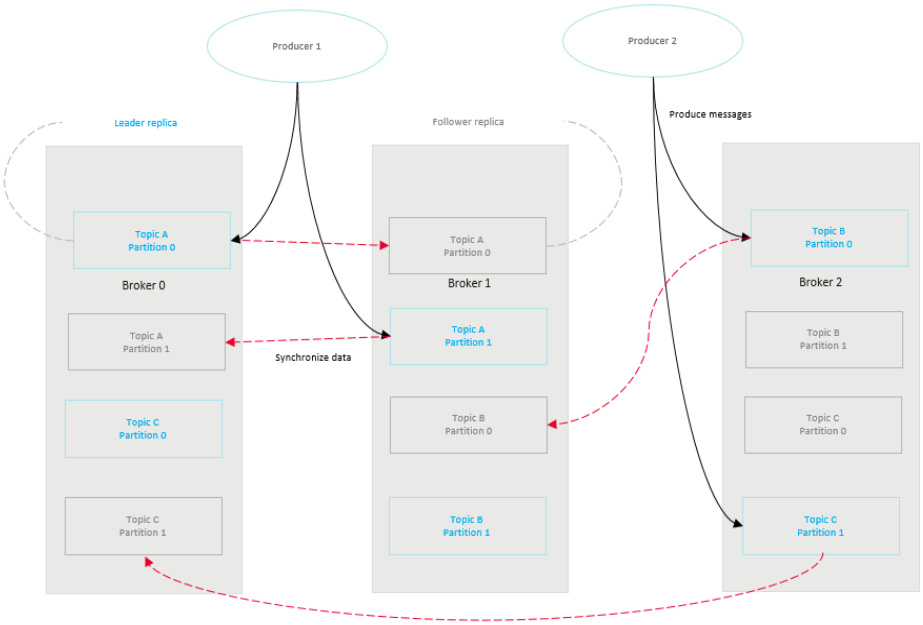
 Copyright © Huawei Technologies Co., Ltd.

Metric ID	Metric Name	Description
broker_disk_usage	Disk Capacity Usage	Disk usage of the Kafka VM
broker_cpu_core_load	Average Load per CPU Core	Average load of each CPU core of the Kafka VM
broker_memory_usage	Memory Usage	Memory usage of the Kafka VM
broker_cpu_usage	CPU Usage	CPU usage of the Kafka VM
group_msgs	Accumulated Messages	Total number of accumulated messages in all consumer groups of a Kafka instance
topic_messages_remained	Available Messages	Number of remaining messages that can be retrieved from the specified topic in the consumer group
broker_messages_in_rate	Message Creation Rate	Kafka node production speed per second
broker_connections	Connections	Number of connections on the Kafka node

Data partition optimization

Kafka divides each topic into multiple partitions for distributed message storage. Each partition has one or more replicas distributed on different brokers. Each replica stores a copy of full data. Messages are synchronized among replicas. The following figure shows the relationships between topics, partitions, replicas, and brokers.

 Copyright © Huawei Technologies Co., Ltd.



Uneven service data among brokers and partitions may happen, leading to low performance of Kafka clusters and low resource utilization.

Causes of uneven service data:

- The traffic of some topics is much heavier than that of others.
- Producers specified partitions when sending messages, leaving unspecified partitions empty.
- Producers specified message keys to send messages to specific partitions.
- The system re-implements flawed partition allocation policies.
- There are new Kafka brokers with no partitions allocated.
- Cluster changes lead to switches and migration of leader replicas, causing data on some brokers to increase.

Data compression

If CPU resources are sufficient, compress the data with compression algorithms.

Common compression algorithms include ZIP, gzip, Sappy, and LZ4. You need to consider the data compression rate and duration when selecting compression algorithms. Generally, an algorithm with a higher compression rate consumes more time.

Compression Mode	Compression Ratio	Client CPU Usage	Server CPU Usage	Disk Usage	Broker Bandwidth Usage
gzip	Medium	Medium	Low	Medium	Low
LZ4	Medium	Medium	Medium	Medium	Medium

 Copyright © Huawei Technologies Co., Ltd.

zstd	High	Medium	Low	Low	Low
Snappy	Low	High	High	High	High

For high TPS, use LZ4; for lower network I/O or CPU usage, use zstd. gzip is not recommended because it is a computing-sensitive compression algorithm. Batch message compression can be better used to obtain higher TPS.

4.11.2.2 Optimizing RabbitMQ Performance

Keep the queue length as short as possible.

Many messages may stack in queues, causing high memory usage. To release memory, RabbitMQ dumps these messages to disks, which is time-consuming. As a result, the message processing becomes slow or the production is blocked. Therefore, message stack may cause negative effects on brokers. Moreover, if a broker crashes and restarts, rebuilding indexes for much data and broker synchronization in the cluster mode are time-consuming.

Use lazy queues for stability.

Lazy queues is a new feature for RabbitMQ 3.6 and later. Messages in lazy queues are automatically stored to disks, which reduces memory usage but increases I/O workload and affects throughput. Lazy queues can better control performance and stabilize clusters. In contrast to non-lazy queues, their messages are stacked in memory and flushed to disks at a time when the memory is insufficient, causing unstable queue performance. To send a large number of messages at a time, or require the consumption speed to catch up with the production speed, lazy queues are recommended. Otherwise, do not use lazy queues when: a. high performance is needed b. queues are not stacked c. queues are configured with the max-length policy

Limit the queue length using TTL or max-length.

Message or queue TTL, and **max-length** can be used to limit the queue length. If the queue length reaches the **max-length**, the messages at the queue head are discarded or enter dead letter queues. Expired messages are also discarded or enter dead letter queues.

Note the queue quantity.

In RabbitMQ, a queue is processed by a thread. To obtain a high throughput, create multiple queues using the multi-core and distributed features of a server, and distribute different queues to different CPUs or different nodes. In the meantime, many queues may overload the CPU and memory, or slow down the response of RabbitMQ management APIs.

Allocate a queue name to a temporary queue automatically.

To use a temporary queue (e.g., an exclusive queue, automatic deletion queue, or non-persistent queue), RabbitMQ can automatically allocate a queue name by calling **queueDeclare()**.

Use automatic deletion queues as required.

 Copyright © Huawei Technologies Co., Ltd.

If queues that are no longer used are stored on a server for a long time, RabbitMQ performance may be affected. They can be automatically deleted by TTL, auto-delete, or exclusive settings.

Use priority queues under control.

Each priority uses an internal queue in the Erlang VM, which consumes some resources. Up to five priorities are enough in most scenarios.

Determine the message size.

Specifying the length of messages sent to RabbitMQ is a common problem. Note that the number of messages sent per second is more likely to hit the bottleneck than the message size. Either sending large messages, or sending multiple small messages is a good choice. It is better that the producer encapsulates multiple small messages into a large one, and then the consumer splits the message for processing. But processing a large message that contains many small messages is slow. In addition, if a small message fails to be processed, the entire large message needs to be resent. Therefore, consider the bandwidth and service architecture in determining the message size.

Connections and channels

A connection occupies about 100 KB memory (100+ KB when TLS is enabled). Thousands of connections cause heavy load on RabbitMQ servers. In extreme cases, RabbitMQ servers may break down due to OOM. The AMQP protocol can implement multiplexing over a single TCP connection. Create one TCP connection for a process. Each thread reuses this connection to create its own channel. The connection must have a long lifecycle because its creation is costly (of at least seven TCP packets). On the contrary, a channel can often be enabled and disabled. But sending messages reusing a channel is a good practice. Do not open a new channel every time a message is sent. Note the following:

- For thread safety, do not share channels among multiple threads.
- For proper latency, do not open and close connections or channels frequently.
- For high throughput, producers and consumers use separate connections.
- To avoid request timeout, do not use many connections and channels because they may affect API performance.

Message acknowledgment

To avoid message loss due to connection issues, consumers use the acknowledgment mechanism. Clients can return an ack message to servers after receiving or processing messages. This mechanism affects performance. If high throughput is required, disable manual acknowledgment. If messages are important to services, the messages should be acknowledged after they are processed. Similarly, producers use the confirm mechanism. When a server receives a message from a producer, the server returns an ack message to implement at-least-once delivery. Note, the confirm mechanism also affects performance.

Control the number of unacknowledged messages.

All unacknowledged messages are temporarily stored in the memory. Many unacknowledged messages may cause service OOM. To limit the number of unacknowledged messages, you can enable the **prefetch** function on the consumer which limits the maximum number of messages that can be pulled.

 Copyright © Huawei Technologies Co., Ltd.

Persistent resources

To prevent message loss from service breakdown, restart, or hardware faults, use persistent queues and messages. Persistent messages involve disk writes. When lazy queues are used, all messages including non-persistent messages are written to disks. If high performance is required, non-persistent messages can be used.

Enable TLS connections.

On AMQP, you can use TLS to connect to RabbitMQ. TLS encrypts and decrypts data, which affects performance.

Select proper QoS.

- When there is single or a few consumers and the consumption is fast, use large QoS to keep the client busy.
- If the message processing speed and bandwidth on the client remain unchanged, estimate proper QoS using formula RTT/Processing duration per message.
- When there are many consumers and the message processing is fast, use small QoS.
- When there are many consumers and the message processing is slow, you are advised to set QoS to 1 so that messages are evenly distributed to all consumers.

Note, if the client is pull-based or has auto-ack enabled, the prefetch function will not work. There is a common error: The number of prefetched messages is not limited. All messages are sent to one consumer. As a result, consumer OOM happens and messages are repeatedly delivered. For more information about the RabbitMQ prefetch function, see **here**.

Performance metrics

Metric ID	Metric Name	Description
channels	Channels	Number of channels in the RabbitMQ instance
queues	Queues	Number of queues in the RabbitMQ instance
connections	Connections	Number of connections in the RabbitMQ instance
connections_usage	Connection Usage	Percentage of current connections to the maximum number of connections
rabbitmq_disk_usage	Disk Capacity Usage	Disk usage of the RabbitMQ VM
rabbitmq_cpu_usage	CPU Usage	CPU usage of the RabbitMQ VM

 Copyright © Huawei Technologies Co., Ltd.

rabbitmq_memory_usage	Memory Usage	Memory usage of the RabbitMQ VM
rabbitmq_cpu_core_load	Average Load per CPU Core	Average load of each CPU core of the RabbitMQ VM

Learn more about **RabbitMQ Metrics**.

4.11.3 Optimizing Serverless Performance

Serverless function configuration

- Runtime

Compiled languages (such as Java and C#) may incur higher cold start latency due to initialization overhead, but offer lower execution latency after initialization. Use interpreted languages (such as Node.js and Python) for scenarios with uneven traffic or strict cold start requirements. For stable traffic or average latency-sensitive workloads, choose compiled languages like Java or Go.

- Memory

CPU resources scale with allocated memory. Larger memory sizes improve performance for CPU/memory-intensive tasks. Test different memory configurations to balance latency and cost.

Use dynamic memory allocation if a function requires varying resources across different scenarios. For details, see **Configuring Dynamic Memory**.

- Maximum instances

Ensure the maximum instance count matches actual concurrency demands. Insufficient instances may cause queuing delays or errors.

- Reserved instances

Reserved instances can be created and released by you as required. After you create reserved instances for a function, FunctionGraph preferentially forwards requests to the reserved instances. If the number of requests exceeds the processing capability of the reserved instances, FunctionGraph will forward the excessive requests to on-demand instances and automatically allocates execution resources to these requests. If your service traffic is uneven and cold start latency is required, you can configure a certain number of reserved instances. If your service traffic has a periodic pattern, you can configure scheduled scaling and intelligent recommendation policies for reserved instances to improve resource utilization. For details, see **Reserved Instance Management**.

Network

Public access: Functions can be accessed over the public network. However, the bandwidth is limited and shared by all tenants. This access mode is suitable for test services that have low requirements on bandwidth and reliability.

 Copyright © Huawei Technologies Co., Ltd.

VPC access: Functions can be accessed from a specified VPC. However, the cold start latency increases because the network link to the VPC is initialized during cold start.

For production with high bandwidth needs, use a VPC with a NAT gateway. If functions do not need to access the network, you are advised not to configure a VPC. For details, see **Configuring the Network**.

- Timeout

Avoid excessively long timeout to prevent service interruption from code exception. Set reasonable timeout aligned with service logic to enable failfast behavior and improve user experience.

Serverless function code

If the service logic can be implemented asynchronously, performance optimization (except for cost) is less critical. FunctionGraph performance largely depends on the executed logic.

Strategies:

- Use connection pools (HTTP, databases, Redis) to reuse established connections.
- Maintain HTTP connection pools to reduce initialization time if functions are invoked via APIs.
- Use global static variables and singletons to avoid reinitializing variable objects.
- Use interpretive languages (Node.js and Python) instead of compiled languages (Java and Go).
- Simplify the function code package to accelerate download from Huawei Cloud OBS before calling the functions.
- When calling other cloud service resources (such as DIS and OBS), optimize the SDK parameters, such as the timeout, number of connections, and retries
- When FunctionGraph accesses a third-party service or Huawei Cloud service, ensure the service has no performance bottleneck.

Cold start optimization

Serverless is the next-generation cloud computing paradigm due to its pay-per-use, auto scaling, and complexity shielding features. However, serverless also has a challenge in real-time applications: cold start. If the cold start and web service initialization time exceed 5 seconds, user experience will be greatly affected. Therefore, reducing the cold start time and improving user experience are urgent problems to be solved when building a serverless architecture.

The lifecycle of a serverless instance can be divided into three phases: initialization, execution, and shutdown.

When FunctionGraph is triggered, if no activated function instance is available, the function code is downloaded and an execution environment is created. The period from the time when a function is triggered to the time when a new FunctionGraph environment is created is called cold start. In the serverless architecture, the cold start cannot be avoided.

 Copyright © Huawei Technologies Co., Ltd.

Currently, FunctionGraph has optimized the cold start on the system side. For details about the cold start on the user side, see the following solutions.

- Proper runtime

FunctionGraph supports.NET Core, Go, Java, Node.js, and Python. Cold start time (512 MB memory): Java > Go > .NET > Node.js > Python.

- Memory

Given a fixed level of request concurrency, higher function memory leads to better cold start performance with more CPU resources allocated.

- Snapshot-based cold start

Cold start is quite prominent in Java applications. FunctionGraph has proposed a process snapshot–based cold start acceleration solution, which helps you break through the performance bottleneck while involving zero or few code changes. In this solution, the execution environment is restored from a snapshot captured after initialization, avoiding complex framework startup and service initialization. The startup latency of Java applications is significantly reduced, and the performance is improved by over 90%.

Performance metrics

FunctionGraph provides a series of metrics to help you optimize your functions. Once a function is called, you can view the metric changes on the monitoring page and log page of the function to further optimize the function configuration and code.

FunctionGraph provides two types of metrics: overview metrics and function metrics. For details, see the **official metric document**.

4.11.4 Optimizing Database Performance

This section describes how to optimize database performance based on specific suggestions and use cases.

1. **Optimizing Database Parameters**

   Database parameters should be designed based on specific workload needs. Huawei Cloud provides default parameters for all its database cloud services to meet the most common workload needs.

   Huawei Cloud provides multiple database services. The optimization methods used and precautions taken depend on the services. You are advised to use Huawei Cloud professional services to optimize your database parameters.

2. **Monitoring Performance Metrics**

   Performance monitoring helps you keep appraised of application and system loads and resource usages in real time. You can set alarm rules to automatically generate alarms for abnormal load, so that you can better use and maintain your cloud database system. Take GeminiDB as an example. You can check GeminiDB Redis API metrics on the console.

3. **Setting Data Partitions**

   GaussDB supports range partitioned tables, list partitioned tables, and hash partitioned tables. Partitioned tables can improve database query performance, enhance availability, facilitate maintenance, and balance I/O.

 Copyright © Huawei Technologies Co., Ltd.

To convert an ordinary table to a partitioned table, you need to create a partitioned table and import data to it from the ordinary table. When you design tables, plan whether to use partitioned tables based on workload requirements.

4. **Rewriting GaussDB SQL Statements**

   SQL statement rewriting allows databases to execute SQL statements faster while still ensuring accurate results. Adhering to specific rules, such as using UNION ALL instead of UNION, adding not null to the join columns, and converting NOT IN to NOT EXISTS, can significantly enhance query efficiency.

5. **Tuning GaussDB Statement Pushdown**

   Currently, there are three ways the GaussDB optimizer can develop statement execution policies in a distributed framework: generating a statement pushdown plan, a distributed execution plan, or a distributed execution plan for submitting statements. The third method sends many intermediate results from DNs to the CN for further execution. In this case, the CN performance bottleneck (in bandwidth, storage, and compute) is caused by statements that cannot be pushed down to DNs. So, you are not advised to use the query statements where only the third method applies.

   Statements cannot be pushed down if they have **functions that do not support pushdown** or **syntax that does not support pushdown**.

   In this case, you can rewrite these statements.

   Typical scenarios for statement pushdown include single-table query pushdown and multi-table query pushdown. In some special scenarios, for example, a statement containing the **with recursive** clause or a column-store table cannot be pushed down.

6. **Tuning GaussDB Subqueries**

   When an application runs a SQL statement to query the database, a large number of subqueries are used because they are more clear than a table join. Especially in complex query statements, subqueries have more complete and independent semantics, which makes SQL statements clearer and easy to understand. In GaussDB, subqueries are classified as subqueries and sublinks depending on where they are located in the SQL statement.

7. **Tuning GaussDB Operators**

   A query goes through many steps to produce its final results. Often, the whole query slows down because just one of the steps takes too long. This slow step is called a bottleneck. To fix this, you can run the **EXPLAIN ANALYZE** or **PERFORMANCE** command to find the bottleneck.

8. **Enabling Read/Write Splitting for a TaurusDB Instance**

   Read/write splitting enables read and write requests to be automatically routed through a proxy address.

   You can **create a proxy instance** after a TaurusDB instance is created. Using the IP address of the proxy instance, write requests are automatically routed to the primary node and read requests are routed to read replicas and the primary node by user-defined weights or active connections.

   When creating a proxy instance, you need to select the nodes (including the primary node and read replicas) to be associated with the proxy instance.

   For more information, see **Enabling Read/Write Splitting**.

 Copyright © Huawei Technologies Co., Ltd.

9. **What Should I Do If Locks on Long Transactions Block the Execution of Subsequent Transactions in TaurusDB?**

   Kill the long transactions.

10. **How Do I Handle a Large Number of Temporary Tables Being Generated for Long Transactions and High Memory Usage in TaurusDB?**

    Upgrade the instance specifications to keep the memory usage within a reasonable range, preventing a sudden increase in traffic from causing an OOM crash. Alternatively, optimize slow SQL statements as needed.

11. **How Do I Handle Slow SQL Statements Caused by Inappropriate Composite Index Settings in TaurusDB?**

    If the slow query was caused by insufficient CPU resources, upgrade the specifications. If it was caused by poor table design, missing indexes, or incorrect index settings, tune the statements.

12. **Using INSTANT to Add Columns Quickly in TaurusDB**

    TaurusDB is compatible with open-source MySQL 8.0.22, so you can use **ALGORITHM=INSTANT** to quickly add columns, preventing lock waiting from affecting workloads or SQL statement execution timeout.

13. **Using DAS Data Diagnosis**

    Intelligent O&M checks instance health using operational data analytics and intelligent algorithms and provides diagnosis results and suggestions.

14. **Monitoring Performance Metrics**

    The following uses **GaussDB** as an example.

    Table 4-1 Performance metrics

Metric ID	Metric Name	Description
rds001_cpu_util	CPU Usage	CPU usage of the monitored object
rds002_mem_util	Memory Usage	Memory usage of the monitored object
rds003_bytes_in	Data Write Volume	Average number of bytes sent by the VM of the monitored object in a measurement period
rds004_bytes_out	Outgoing Data Volume	Average number of bytes received by the VM of the monitored object in a measurement period
iops_usage	IOPS Usage	Percentage of IOPS capacity in use.
rds007_instance_disk_usage	Instance Disk Usage	Real-time data disk usage of the monitored instance
rds010_disk_usage	Disk Usage	Real-time data disk usage of the monitored node

    For more GaussDB metrics and other database metrics, see the official documentation.

 Copyright © Huawei Technologies Co., Ltd.

4.11.5 Optimizing AI Performance

1. **Practice: Improving model performance through training optimization**

   Parameter tuning policy: Adjust parameters such as the flash attention, parallel splitting policy, micro batch size, and recomputation policy.

   Improve performance by taking full advantage of the video random-access memory (VRAM), computing power, and parameter tuning.

   Performance breakdown

   If the performance after parameter tuning still cannot meet the commercial use requirements, you can perform profiling. After collecting performance data, break down and analyze performance from the operator, communications, scheduling, and memory dimensions, and add profiling code to the training script. The procedure is as follows: First, generate the profiling data directory structure. Then use the ATT tool to compare and analyze the end-to-end time consumption of the NPU and competing products. Finally, perform tracing analysis.

   Operator analysis

   Analyze operators through the summary file generated in profiling and use FA and MM operators.

2. **Practice: route planning acceleration**

   The ranktable route planning is a communication optimization capability used in distributed parallel training. When NPUs are used, network route affinity planning can be performed for communication paths between nodes based on the actual switch topology, improving the communication speed between nodes. This case describes how to complete a PyTorch NPU distributed training task in ModelArts Lite using ranktable route planning. By default, training tasks are delivered to the Lite resource pool cluster in Volcano job mode. For details, see **Best Practices**.

3. **Practice: training VRAM optimization**

   – **PyTorch memory pool management policies**

   - The PyTorch memory pool is managed by block. Block pools can be categorized as small memory pools and large memory pools. PyTorch requests memory from device drivers by block. The interface used by a user or PyTorch code to request memory from the memory pool is represented as the request or release of a tensor.
   - The tensor lifecycle is managed using a reference counting mode similar to a smart pointer. The channel between Python and C++ is streamlined. That means a Python tensor object is associated with a C++ tensor object. If a Python tensor object disappears, the corresponding C++ tensor object is destructed to release memory. A tensor object created in the C++ environment can be returned as a Python tensor object.
   - Tensor objects in C++ consist of viewTensor and storageTensor. The former contains various meta information of a tensor, such as shape, stride, and dataType. The latter contains the memory address and offset. viewTensor is displayed for tensor objects. This is the basic rule when multiple tensors correspond to the same memory after PyTorch performs view operations. When storageTensor requests

 Copyright © Huawei Technologies Co., Ltd.

memory, the PyTorch block pool will allocate an idle block, split it as needed, and return the address pointer.

- **Cross-stream memory reuse policies for PyTorch**
  - If you want to use a tensor obtained from the memory pool of a stream for another stream, perform the recordStream operation to add the information about the target stream to the block of the tensor. When the lifecycle of the tensor ends and the address is released, an **event_record** task is delivered to the target stream.
  - Each time the stream requests memory, an **event_query** operation is performed. If the event has been recorded, it indicates that the task has been executed on other streams. In this case, you can reclaim the block.
- **PyTorch memory statistics**
  - Generally, PyTorch memory statistics include allocated, active, and reserved memory. Allocated memory refers to the memory allocated to tensors on a host and not yet released. (Note that this is about memory allocation and release on a host, but not indicate the device status.) Active memory refers to the memory not yet released on a host plus the memory still occupied by other streams.
  - For example, a tensor requests memory on stream A to perform the allreduce collective communication operation on stream B. After the tensor is added, it is released. In this case, the allocated memory will be subtracted, but the active memory will not. The active memory will be subtracted only after the allreduce is executed and the result is obtained through **query_event**.
  - If PyTorch requests 100 MB memory from the device and then releases it. Then PyTorch requests another 20 MB memory and does not release it. In this case, the allocated memory is 20 MB, and the reserved memory is 100 MB. On the actual network, the reserved memory contains a large number of small blocks, which cannot be reused as a large memory block. As a result, there is a large difference between the reserved and allocated memory, which is called memory fragments.
- **Factors affecting PyTorch memory fragments**
  - Theoretically, more memory requests and releases within a single step always lead to more memory fragments. The PyTorch memory pool depends only on the training script logic on the host. The training script logic of each step is the same. Therefore, the memory status becomes stable after the first step.
  - Tensors of different lifecycles request and release memory alternately. To reduce memory fragments, tensors of a long lifecycle should request memory first. Workspace memory can be serially reused in an absolute sense. Therefore, a customized memory pool policy can be used to reduce the impact of memory fragments. The common operation of converting non-continuous memory to continuous memory is a common memory request on NPUs compared with GPUs.

 Copyright © Huawei Technologies Co., Ltd.

- **VRAM optimization policy**

  The number of parameters of a foundation model increases exponentially, far exceeding the capacity of physical memory with a single GPU. Therefore, VRAM optimization is required for training foundation models. VRAM optimization optimizes algorithms to reduce VRAM consumption, or enlarges the VRAM to obtain extra spaces through some replacement methods. As the physical size of VRAM is fixed, you can obtain extra spaces by prolonging time or transferring spaces. Prolonging time usually consumes computing power and bandwidth. Transferring spaces mainly consumes I/O bandwidth and has a delay, which may reduce throughput.

- **Performance metrics**

  Table 4-2 Performance metrics

Metric ID	Metric Name	Description
cpu_usage	CPU Usage	CPU usage of ModelArts
mem_usage	Memory Usage	Memory usage of ModelArts
gpu_util	GPU Usage	GPU usage of ModelArts
gpu_mem_usage	GPU Memory Usage	GPU memory usage of ModelArts
npu_util	NPU Usage	NPU usage of ModelArts
npu_mem_usage	NPU Memory Usage	NPU memory usage of ModelArts
disk_read_rate	Disk Read Rate	Disk read rate of ModelArts
disk_write_rate	Disk Write Rate	Disk write rate of ModelArts

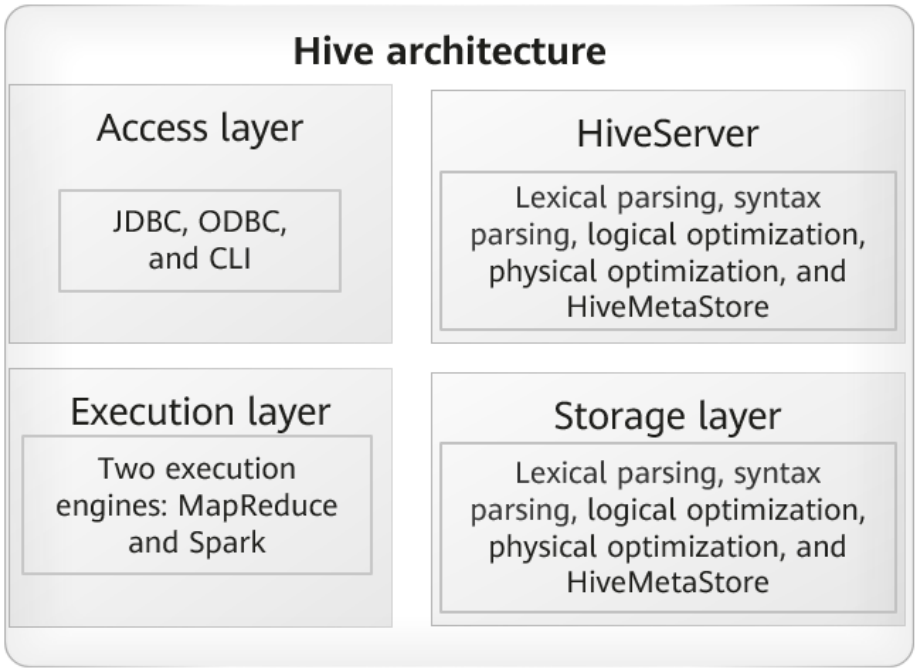
  For details about all metrics, see **ModelArts Metrics**.

4.11.6 Optimizing Big Data Performance

4.11.6.1 Optimizing Hive

Overview

Hive architecture



Hive provides the SQL capability of Hadoop. Hive SQL (HQL) was developed based on the standard SQL and is more suitable for the Hadoop distributed system. It is a de facto standard of the Hadoop system.

Hive tuning

After a user enters an HQL, Hive analyzes morphology and syntax of HQL, generates and optimizes an execution plan, and submits a task to YARN. Hive tuning covers the following parts:

- Access layer: user connection performance, such as the network speed, authentication, and the number of concurrent connections
- HiveServer: The execution plan is the main method of SQL optimization. You can view how Hive decomposes and orchestrates SQL statements through interfaces and optimize tasks based on the MapReduce or Spark execution status.
- HiveMetaStore: The MetaStore of Hive may be an independent external database. It affects the performance of HiveServer, mainly including the HiveMetaStore access time, access times, and number of concurrent connections.
- MapReduce/Spark: The execution status of MapReduce or Spark directly affects the Hive performance, such as the size of each task, how evenly tasks and resources are allocated, and how properly tasks are split.
- HDFS: I/O read at the base layer is a key factor to affect performance. Read and write performance as well as block size should be considered.

For details about MapReduce, Spark, and HDFS performance tuning, see their own performance tuning guides and documentation. This section mainly describes performance tuning for the first three aspects and also MapReduce and Spark.

Batch processing

Batch processing takes a long time and consumes a large number of resources. It can be tuned as follows:

 Copyright © Huawei Technologies Co., Ltd.

- Use ORC files and appropriate compression algorithms, such as Zlib and Snappy. Zlib delivers a higher compression ratio but takes a longer time and consumes more resources than Snappy. Snappy balances the compression ratio and decompression performance. Snappy is recommended.
- Use Map Join to reduce Shuffle times, remarkably improving performance.
- If different SQL statements are used to provide the same function, the fewer MapReduce tasks are generated, the better.
- The default Hive configurations are typical. You can adjust some parameters to meet service requirements, for example, the file block size, the number of Map and Reduce tasks, and compression algorithm.
- Use partitions properly. Avoid too many partitions. Specify a specific partition in the query SQL statement.

Measurement metrics

Measurement metrics help you quickly detect Hive execution problems and locate Hive performance problems. Check the following metrics in sequence: common metrics, access layer metrics, HiveMetaStore metrics, HiveServer metrics, and metrics of other components such as MapReduce, Spark, and HDFS. The following are available metrics:

- Common metrics

Common performance metrics of servers: CPU usage, memory usage, disk I/O read and write speed, the number of used Cores. These metrics help you measure task execution status on the servers and check whether cluster loads are balanced.

- Access layer metrics

Number of Hive connections, number of concurrent SQL statements, input cache value (or batch size), and maximum number of requests that can be concurrency handled by a HiveServer instance (The default value is 500, and the value is restricted by JVM memory and CPU processing capability.)

- HiveMetaStore

Number of HiveMetaStore connections, number of concurrent SQL statements, and statement execution plan

- HiveServer

You can view the execution plans of SQL statements, tune SQL statements, and set the parameters of the MapReduce or Spark engine, such as the number of jobs, number of Map tasks, and number of Reduce tasks.

- Common test standards

TPC-DS is used to compare the performance of Hive with that of peering products or its earlier versions. Standard tests only provide reference for performance tests.

Metrics monitoring

- Common metrics

On the **Hosts** page of Manager, check host resource usage, for example, CPUs, memory, and I/O usage in the clusters.

 Copyright © Huawei Technologies Co., Ltd.

- Access layer metrics

On the Hive service status page of Manager, check the number of HiveServer connections and the number of HQL statements successfully executed.

- HiveMetaStore metrics

On the Hive service status page of Manager, check the number of HiveMetaStore request connections and performance of key APIs.

- HiveServer-related metrics

4.11.6.2 Optimizing Spark Performance

Overview

Spark is an in-memory distributed computing framework. In iterative computation scenarios, the computing capability of Spark is 10 to 100 times higher than MapReduce, because data is stored in memory when being processed. Spark can use HDFS as the underlying storage system, enabling users to quickly switch to Spark from MapReduce. Spark provides one-stop data analysis capabilities, such as the streaming processing in small batches, offline batch processing, SQL query, and data mining. Users can seamlessly use these functions in a same application.

Features of Spark are as follows:

- Improves the data processing capability through distributed memory computing and directed acyclic graph (DAG) execution engine. The delivered performance is 10 to 100 times higher than that of MapReduce.
- Supports multiple development languages (Scala/Java/Python) and dozens of highly abstract operators to facilitate the construction of distributed data processing applications.
- Builds data processing stacks using SQL, Streaming, MLlib, and GraphX to provide one-stop data processing capabilities.
- Fits into the Hadoop ecosystem, allowing Spark applications to run on Standalone, Mesos, or Yarn, enabling access of multiple data sources such as HDFS, HBase, and Hive, and supporting smooth migration of the MapReduce application to Spark.

Cluster service deployment planning

Service scale and capacity parameter configuration

- Spark, as a memory-based computing engine, requires a large memory and a large number of CPU resources. You must properly plan memory and CPU resources based on the current service capacity and growth rate. The following aspects must be considered:
- If the program runs in yarn-client mode, pay attention to the data volume at the driver end and set a proper memory for the driver.
- The CPU and memory resources must be planned based on service objectives. During planning, you need to set **executor-memory**, **executor-cores**, and **Executor-num** based on the data distribution and service complexity. You also need to plan the number of CPUs and memory.
- Reserve some memory space (usually 20% of the total memory) as the buffer cache of the operating system.

- Consider data expansion after data blocks read from HDFS are decompressed.
- Plan some disks for storing cached data, logs, and shuffle data.

Optimization principles

- Increase CPU usage while reducing extra performance overhead.
- Increase memory usage.
- Optimize the service logic to reduce the calculation workload and I/O operations.

Typical service tuning

- Before tuning Spark parameters, you must optimize code logic through planning and design.
- Slow Spark jobs and low CPU utilization. Executor threads cannot be fully occupied. Reduce the number of cores of each executor, add more executors, and increase the number of partitions.
- Memory overflow may occur if too much data is processed by a task due to large data segments, or if the memory of a task is insufficient due to a low concurrency of the executor. Remove some executors and add data shards.
- A small data volume but a large number of small files. Reduce data fragments and execute the reduce operator and then the coalesce operator to reduce tasks and CPU load.
- Search for a large table with Spark SQL for a large number of columns but a small number of columns. Use the rcfile or parquet format to reduce file read costs. Select a proper compaction format to reduce memory load.

Metrics monitoring

Performance metrics include throughput, resource usage, and scalability.

- Throughput: Run the same computing tasks in the same resource environment and check how fast the task is complete.
- Resource usage: Run computing tasks and view the CPU, memory, and network usage under different loads.
- Scalability:

– Performance improvement curve upon horizontal scaling: Run the same computing tasks before and after resource scaling and then compare system performance.

– Performance decrease curve upon increase of system load: Increase system load under the same resource environment and compare system performance before and after the load increase.

4.11.6.3 Optimizing Flink Performance

Overview

Flink is a unified computing framework that supports both batch processing and stream processing. It provides a stream data processing engine that supports data distribution and parallel computing. Flink features stream processing and is a top open-source stream processing engine in the industry. Flink provides high-concurrency pipeline data processing, millisecond-level latency, and high reliability, making it suitable for low-latency data processing.

Cluster Service Deployment Architecture

Service Scale and Capacity Parameter Configuration

Flink, a stream data processing engine, relies on memory and CPU resources. You must plan memory and CPU resources based on the current service capacity and growth rate. The following aspects must be considered:

- The CPU and memory resources must be planned based on service objectives. Set JobManager memory, the number of TaskManagers, TaskManager memory, the number of slots in each TaskManager, and CPU cores based on current data distribution and service complexity.
- When planning the memory, reserve 20% of memory space (recommended) as the buffer cache of the operating system.
- Consider how data will expand after data blocks read from HDFS are decompressed.
- Plan certain disks for cache space for cache data and logs.

Optimization Objective

Flink performance optimization aims at efficiently completing service objectives without affecting the normal operation of other services. To achieve this goal, the system generally maximizes the utilization of physical cluster resources such as the CPU, memory, and disk I/O. In this case, certain resources may become bottlenecks.

Optimization Principles

- Improve CPU utilization and reduce extra performance costs.
- Improve memory utilization.
- Optimize the service logic to reduce processing workloads and I/O operations.

Typical Performance Optimization Method with DataStream

- Configuring memory: Adjust the ratio of the old to the new generation resources. When developing Flink applications, optimize the data partitioning or grouping of DataStream.
- Configuring the degree of parallelism (DOP): Set the DOP parameter based on the actual memory, CPU, data, and application logic. Tasks can be arranged based on their level of parallelism, in descending order of priority, as specified by the operator layer, execution environment layer, client layer, and system layer.
- Configuring process parameters: Set JobManager memory, number of TaskManagers, number of TaskManager slots, and TaskManager memory.
- Partitioning design methods: random partitioning, rebalancing (round-robin partitioning of elements to balance load in each partition), rescaling (round-robin partitioning of elements to distribute them in subsets of downstream operations), broadcast partitioning (broadcast each element to all partitions), and custom partitioning
- Configuring the Netty network communications: Modify configurations in the **conf/flink-conf.yaml** file on the client.

Metrics Monitoring

Performance metrics include throughput, resource utilization, and scalability.

 Copyright © Huawei Technologies Co., Ltd.

- Throughput: Run the same computing tasks in the same resource environment and check how fast the task completes.
- Resource usage: Run computing tasks and monitor the CPU, memory, and network usages under different loads.
- Scalability:

– Performance improvement curve with horizontal scaling: Run the same computing tasks before and after resource scaling and then compare system performance.

– Performance degradation curve as system loads increase: Increase the system load in the same resource environment and compare system performance before and after the change.

 Copyright © Huawei Technologies Co., Ltd.

5 Cost Optimization Pillar

5.1 Overview

Cost optimization helps enterprises use cloud services effectively to build workloads, improve workload performance throughout the entire lifecycle, cut unnecessary costs, boost operational efficiency, and maximize the cost-efficiency of cloud applications.

Cost optimization practices do not merely imply cost reduction; they represent a balance across dimensions such as security compliance and resilience, as well as the optimal investment for achieving business objectives.

Huawei integrates its cloud operations cost management expertise with industry best practices to develop systematic methodologies and recommendations. These include improving cost management efficiency, making rational decisions on cloud resource selection and allocation, establishing budget management mechanisms, ensuring continuous cost governance, improving resource utilization, and optimizing architecture.

5.2 Concepts

Concepts

Term	Definition
FinOps	FinOps, a fusion of Finance and DevOps, fosters collaboration among IT, finance, and business teams to integrate financial accountability into cloud operations. It enables data-driven decision-making while balancing speed, cost, and performance.
CFM	Huawei Cloud Cloud Financial Management (CFM), based on FinOps practices, establishes end-to-end cloud financial management capabilities to help customers enhance transparency and predictability of cloud spending, enabling more accurate, efficient allocation, control, and optimization of cloud costs.

 Copyright © Huawei Technologies Co., Ltd.

5.3 Design Principles

Organizational, Process, and Cost Management Alignment

A fundamental principle of cost optimization is ensuring alignment between organizational structure, processes, and cost management. Establishing clear responsibility and accountability mechanisms is critical—even the most advanced cost optimization tools will fail without proper execution.

In terms of processes, cost management should be an essential part of every cloud migration process. In terms of organization, it is important to allocate the necessary time, resources, and personnel to establish cloud financial management capabilities. For example, before you apply for cloud accounts and cloud resources, it is crucial to have well-defined processes to link organizations, projects, cloud accounts, and resources, ensuring that each department or team in your company is accountable for the resources they use. Introducing cloud financial management capabilities to the enterprise finance also helps understand the relationship between output and costs of each department, even clarifying how much cloud costs are incurred for every dollar earned by the company. This helps optimize spending while maintaining business productivity.

The execution phase of cost optimization typically requires cross-departmental coordination involving operations, maintenance, and R&D teams. This process also demands close collaboration between platform teams and business units, underscoring the need to establish a comprehensive process involving all organizational members, such as a process for releasing idle resources, which requires engagement from all relevant stakeholders.

Enterprises can also generate regular reports and share them with stakeholders. Additionally, it is beneficial to hold joint meetings, such as monthly meetings involving various roles, to review budget execution, discuss risk mitigation strategies, summarize optimization experiences, and plan next steps.

Planning Ahead with Proper Cost Modeling, Budget Planning, and Cost Forecasting

Understanding the costs of each organization and project is not easy, especially when many cloud resources are shared across different organizations and projects. Therefore, it is essential to establish a foundational cost model early on that is approved by the management team. This model will ensure overall cost traceability during future cost optimization.

After the cost model is in place, it is important to create a budget plan. Introduce this budget plan and cost management at the beginning of cloud-based projects, and set thresholds for different levels of budget usage to serve as cost alerts.

Additionally, for users with Huawei Cloud accounts, Huawei Cloud offers cost forecasting tools that provide forecasts based on actual cloud costs. By utilizing these tools, cloud users can set up predictive alerts based on budget situations.

Monitoring Costs and Resource Usage

As business operations expand, more cloud resources are demanded, while idle resources must be efficiently released to optimize costs. At the same time, the

 Copyright © Huawei Technologies Co., Ltd.

architecture of cloud-based applications continues to evolve. Enterprises should establish regular cloud cost monitoring and review mechanisms, and adjust and improve cost optimization strategies based on actual circumstances. For example, a rapidly growing business organization may prioritize agility and adopt a more flexible cloud cost optimization strategy to accelerate growth, while a stable business organization may focus more on cost efficiency and design a stricter cloud cost optimization strategy. Enterprises can also utilize the cloud cost management tools and platform provided by Huawei Cloud Cost Center to automate cost monitoring and optimization.

Cost Reduction and Optimization Through Diverse Billing Modes, Resource Efficiency Enhancements, and Architectural Improvements

The main factors affecting cloud spending are rates and usage. Different optimization measures can be implemented through the analysis of cloud business models and cost data. In terms of rates, cloud services can be billed on different bases, such as pay-per-use, yearly/monthly, resource package, and spot pricing, each suitable for different scenarios. You can choose the most appropriate billing mode based on your needs to adapt to different business forms and reduce costs. In terms of usage, enterprises can reduce resource consumption by optimizing resource efficiency, releasing idle resources, downsizing, or implementing architecture solutions like hybrid deployment and elastic scaling to reuse resources and release idle resources during off-peak times. This also helps achieve savings in resource usage. As investments in cost optimization, resource efficiency optimization, and architecture optimization gradually increase, their impacts on the business increase correspondingly. Enterprises can determine optimization objectives and prioritize optimization measures based on evaluations of business objectives, impact on business, and optimization investments and returns.

5.4 Problems and Checklists

During cost optimization design, this checklist can help identify areas for improvement and guide their enhancement. Each item in the checklist represents a best practice, which will be explained in detail in the next section.

Question	Checks and Best Practices
COST01 Have you planned organizations and processes according to cost optimization requirements?	1. Plan enterprise organizations, and align the organizational structure, process, and cost management 2. Design a structured IT governance system to optimize management efficiency. 3. Clarify team responsibilities, and cultivate and sustain an organizational cost-consciousness culture. 4. Establish cloud resource and permissions management policies.

 Copyright © Huawei Technologies Co., Ltd.

COST02 Is there any budget planning and management mechanism in your company?	1. Establish a cloud budgeting and forecasting process. 2. Implement granular budget management and rigorous expenditure tracking.
COST03 Are your costs being properly allocated across organization units?	1. Develop cost allocation principles. 2. Visualize cost allocation results. 3. Allocate shared costs.
COST04 Do you maintain ongoing cost governance practices to optimize your cloud expenditure?	1. Establish governance frameworks to continuously optimize cost allocation ratio. 2. Implement proactive cost monitoring to maintain financial control.
COST05 Have you set strategies and measurable objectives for cost optimization?	1. Analyze service trends and optimization benefits. 2. Establish measurable optimization objectives. 3. Regularly review and verify cost optimization outcomes.
COST06 Have you explored cost optimization through different billing modes?	1. Understand the characteristics of different billing modes on the cloud. 2. Choose the appropriate billing modes for your workloads. 3. Track and monitor the usage of special products.
COST07 Have you implemented comprehensive resource utilization management and optimization processes?	1. Continuously monitor resource utilization. 2. Release idle resources. 3. Conduct comprehensive cloud technology selection analysis. 4. Right-size resources based on workload patterns and resource utilization.
COST08 Does your cost optimization strategy include architectural improvements?	1. Design application architecture with geographic considerations. 2. Implement cloud-native transformation to reduce costs. 3. Adopt storage-compute decoupling to lower costs. 4. Explore the opportunity for using the Serverless architecture for cost optimization.

 Copyright © Huawei Technologies Co., Ltd.

5.5 COST01 Planning Organizations and Processes According to Cost Optimization Requirements

5.5.1 COST01-01 Planning Enterprise Organizations, and Aligning the Organizational Structure, Process, and Cost Management

- **Risk level**

  High

- **Key strategies**

  A fundamental principle of cost optimization is ensuring alignment between organizational structure, processes, and cost management. Establishing clear responsibility and accountability mechanisms is critical—even the most advanced cost optimization tools will fail without proper execution. A good practice is to create a team (cloud business office, cloud center of excellence, or FinOps team) at the beginning to cultivate and sustain an organizational cost-consciousness culture. The cost optimization lead can be an individual or team with a strong understanding of the organization and cloud finance. Team members should come from key decision-making and implementation departments, such as executive leadership (CXOs), finance, development, O&M, and data analytics.

  This team can function as a virtual team that establishes cost management processes when the company begins implementing cost optimization. For example, they can incorporate costs into key assessment metrics in the entire application and service lifecycle, and drive resolution of various discrepancies such as cost allocation. Ultimately, they define and implement a comprehensive accountability mechanism for cost-related matters within the company.

5.5.2 COST01-02 Designing a Structured IT Governance System to Optimize Management Efficiency

- **Risk level**

  High

- **Key strategies**

  An IT governance structure that aligns with your organization helps manage costs across the entire organization. As business operations scale, establishing subsidiaries and branches, and breaking down large departments into smaller ones become common, resulting in a more complex organizational hierarchy. The IT governance framework of a company is also influenced by its organizational structure. It is important to align the framework with the enterprise management model to help manage people, finances, and resources in a multi-level organizational manner, ensuring accountability for all resources. In this way, the company can easily allocate costs to business teams—holding each team accountable for the services they consume.

 Copyright © Huawei Technologies Co., Ltd.

- **Related services and tools**

  If you are a large-sized enterprise or corporate group, you are advised to create a hierarchical organization with multiple accounts for unified accounting management. For details, see **Enterprise Management and Organizations**. With Enterprise Center enabled, you can isolate resources and costs on an account-by-account basis, helping you quickly expand business.

  If you are a small- or medium-sized enterprise or you are using a standalone account, you can enable **Enterprise Project Management Service (EPS)** to group costs based on the organizational structure. If you have other multi-dimensional and fine-grained planning requirements, you can use **cost tags** as a supplement to organization planning. For example, you can use a specific tag to identify the team, application, and owner of a specific resource.

  If an enterprise uses master-member account relationship for cost management, there are unified accounting and independent accounting solutions for master and member accounts.

  – In unified accounting, an enterprise master account can centrally manage costs of both its own account and the associated member accounts. This includes cost analysis, budget tracking, cost anomaly detection, and cost optimization recommendations—significantly improving financial management efficiency.

  – In independent accounting, the master account and the member accounts manage their own costs, including cost analysis, budget tracking, cost anomaly detection, and cost optimization recommendations. When member accounts grant access permissions, the master account can analyze both its own and the member accounts' expenditures.

5.5.3 COST01-03 Clarifying Team Responsibilities, and Cultivating and Sustaining an Organizational Cost-consciousness Culture

- **Risk level**

  Medium

- **Key strategies**

  Effective cost optimization requires fostering cost awareness across teams and clarifying responsibilities. A proven approach is implementing well-defined KPIs, such as profit-to-cost ratios, per-unit product costs, and core resource utilization, and generating team-level reports. This drives both transparency and accountability in cloud spending.

  Cost optimization is an iterative process, not a one-time initiative. Teams need time to embrace responsibilities, refine implementation approaches, and improve the KPIs. The KPIs should evolve over time, fostering a culture of continuous improvement rather than expecting immediate perfection. Adopting a growth mindset is essential—taking mistakes as valuable learning and improvement experiences. This mindset reduces resistance to cost optimization for both inexperienced teams and the leadership. Then continuously refine budgets, IT expenditures, and forecasts through cyclical improvements.

  Finally, cost consciousness needs to be incorporated into processes and employee training.

 Copyright © Huawei Technologies Co., Ltd.

5.5.4 COST01-04 Establishing Cloud Resource and Permissions Management Policies

- **Risk level**

  High

- **Key strategies**

  Cost optimization is a cross-departmental effort. As cloud resources account for a significant portion of expenditures on the cloud, it is essential to establish clear policies that define how resources are managed. As mentioned above, a best practice is to use separate accounts to isolate resources across organizations or departments. Even within a single entity, segregating accounts or environments by function, such as development, testing, core services, and non-core services, enhances operational clarity and cost control.

  Despite decentralized accounts or environments, resource and permissions management should remain centralized. The central team, such as the cloud business office, cloud center of excellence, and FinOps team, must assign groups and roles to each account. This structured approach ensures that only authorized personnel can create, modify, or disable resources. Additionally, a unified resource and cost view is essential for centralizing bill management and tracking expenditures.

- **Related services and tools**

  You can use **IAM** to implement fine-grained permissions management to control resource access of users in a single account.

  For cross-account management, you can use service control policies (SCPs) provided by **Organizations** to centrally manage permissions of every account.

5.6 COST02 Planning and Managing Budgets

5.6.1 COST02-01 Establishing Cloud Budgets and Forecasts

- **Risk level**

  High

- **Key strategies**

  It is easy to provision and expand cloud resources, so using the cloud brings efficiency, innovation speed, and agility. However, this also introduces a highly variable cost and usage pattern. Enterprises must rethink their existing budgeting and forecasting methods to adapt to changes in the cloud.

  It is also essential to closely monitor historical expenditure changes and shifting service trends to make the most accurate budgeting possible. In addition, you can combine trend-based algorithms (using historical expenditures as inputs) and business driver-based algorithms (for example, new services on cloud or regional expansion) to improve the accuracy of financial forecasting.

- **Related services and tools**

  You can access Cost Center and enable cost analysis to forecast future costs based on your historical expenditures. **Cost analysis** of Cost Center considers

 Copyright © Huawei Technologies Co., Ltd.

the characteristics of different billing modes and uses machine learning alongside rule-based models to forecast costs and usage across these modes.

You can also use **Price Calculator** to estimate costs for your desired services or any combination of services based on their usage required for service rollout or regional expansion.

5.6.2 COST02-02 Refining Budget Management and Tracking

- **Risk level**

  High

- **Key strategies**

  A budget management mechanism should be established to manage the cloud costs of each project, service, or application throughout their lifecycle.

  Projects and services of an enterprise change over time. Generally, emerging projects and services require more cloud resources, while stable projects and services can focus more on continuously optimizing the cloud costs per unit of revenue. Projects and services at the end of their lifecycle should consider gradually releasing resources that are no longer needed.

  When formulating budgets, enterprises should track project lifecycles, regularly review budget usage, and dynamically adjust budgets based on service conditions to better control costs while ensuring that the budgets meet the actual needs.

  It is critical to establish response plans for cost overruns. These plans should include clear processes and owners, and identify the owners or teams who will receive alerts and take actions. These individuals or teams may be the finance team, operations team, department heads, or specific cost management teams. In this way, when an alert is triggered, necessary measures can be quickly taken, such as optimizing resources, stopping unnecessary services, or restricting the purchase of cloud resources for a department or project.

- **Related services and tools**

  Huawei Cloud provides budget management for you to track your costs by product, team, or project in a refined manner.

  In addition to viewing the budget progress in Cost Center, you can also configure budget alerts. If the actual or predicted usage reaches the specified threshold, you will receive an alert via SMS or email so that you can take measures in a timely manner.

  You can create a daily, weekly, or monthly budget report to inform relevant personnel so that the costs are always visible to the relevant business departments, finance departments, and CTO of your enterprise.

5.7 COST03 Allocating Costs

5.7.1 COST03-01 Developing Cost Allocation Principles

- **Risk level**

  High

 Copyright © Huawei Technologies Co., Ltd.

- **Key strategies**

  Cost allocation enables enterprises to allocate costs to business teams, making costs of each business team clearly visible. This is also the basis of team responsibilities specified above. Based on clear costs, business departments can provide cost-effective solutions with properly priced products and a balance between costs, stability, and performance. Enterprise managers can make informed decisions on cloud expenditures of each business, ensuring that expenditures of core and strategic businesses are within the budget and no waste occurs.

  Cost allocation follows the principles below to match business needs:

  - Costs are allocated to specific users. They are allocated to the entity that actually incurred costs not to the one that purchased the services.
  - Costs are allocated based on the actual consumption. For example, if you purchased a yearly subscription at USD 365 in January, the cost is allocated to January based on the expenditure. If you want to allocate the cost based on the consumption, the allocation is performed throughout the subscription period, with USD 1 recorded for each day within the year. This method of cost allocation better reflects cost accountability.
  - Shared costs should also be allocated within organizations. Good examples of sources for shared costs include shared resources, platform services, support plans, and untagged resources. When shared costs are allocated, the increased visibility encourages the various business units to use shared resources more appropriately to reduce waste.

- **Related services and tools**

  Huawei Cloud Cost Center supports allocation for costs of yearly/monthly resources and resource packages based on the actual users and consumption.

5.7.2 COST03-02 Visualizing Cost Allocation Results

- **Risk level**

  Medium

- **Key strategies**

  It is essential to allocate costs to responsible teams to ensure that each team can clearly understand their service costs in a timely and accurate manner, enhance cost awareness, and balance performance, reliability, and costs when building solutions.

- **Related services and tools**

  Huawei Cloud allows you to allocate costs based on your organizational plan. Once cloud services incur costs within your organizational plan, you can use Bill Management, **Cost Analysis**, and other tools to gain visibility into your costs.

  You can also use **cost categories** to define rules with multiple conditions (service type, bill type, linked account, enterprise project, and cost tag) and allocate costs to meaningful categories based on your business needs. You can view the cost allocation results using Cost Analysis.

  If you want to integrate detailed cost data with data from your own cloud management platform to generate tailored cost and usage analysis, you can subscribe to bill details, **dump cost data to OBS**, or **call customer**

 Copyright © Huawei Technologies Co., Ltd.

operations APIs. For example, you can combine the cost and usage details with the operations data of your enterprise to generate the KPIs for each business unit.

5.7.3 COST03-03 Allocating Shared Costs

- **Risk level**

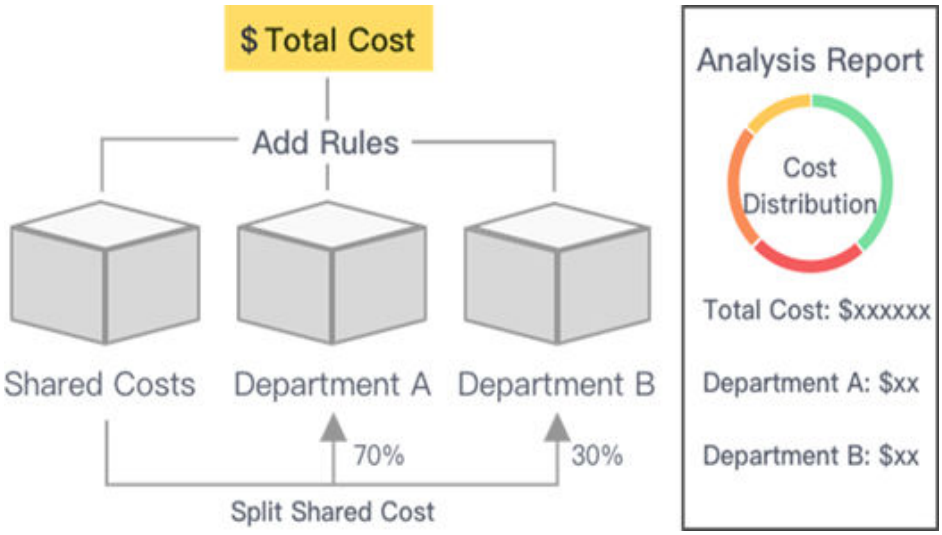
  Medium

- **Key strategies**

  Costs of resources shared by multiple teams, such as CDN and live bandwidths, should be allocated to each team based on their actual bandwidth usage.

  Costs of CCE clusters shared by multiple teams should be allocated based on the proportion of CPU and memory capacity allocated to and used by each team. The costs cover the cluster resources including CCE, ECS, and EVS resources.

  In addition, other shared resource costs, platform costs, support costs, and unallocated costs due to untimely tagging can also be allocated among various business departments according to certain rules, such as equal distribution, allocation based on consumption ratios, or allocation based on agreed ratios, in order to meet the need for fair allocation of shared costs.



- **Related services and tools**

  Cost categories of Huawei Cloud Cost Center offer a proportional method for splitting shared costs.

  Huawei Cloud Cost Center also supports allocation of shared CDN and live bandwidth costs by domain name. For details, see **Splitting Shared Costs**.

  Huawei Cloud **CCE** enables granular cost allocation at the pod level, which can be summarized by higher-level Kubernetes models, such as Workloads and Services.

5.8 COST04 Performing Continuous Cost Governance

 Copyright © Huawei Technologies Co., Ltd.

5.8.1 COST04-01 Establishing Governance Frameworks to Continuously Optimize the Cost Allocation Ratio

- **Risk level**

  Medium

- **Key strategies**

  Accurate and effective cost allocation is the basis for cost monitoring and optimization. Enterprises should focus on and improve the cost allocation ratio to strength cost governance.

  Tagging is a widely used method that allows flexible cost allocation across various business dimensions, such as products, applications, and responsible personnel. However, during implementation, companies may face challenges that reduce tag coverage, including:

  Heavy workload: As cloud resource usage grows, managing multiple tags across numerous assets becomes increasingly complex.

  Inconsistent tag application: Variability in departmental adoption, incorrect key-value assignments, and untagged resources create gaps in cost tracking.

  Evolving environments and requirements: Organizational changes, shifting stakeholder expectations, and adjustments to business or financial strategies impact tagging effectiveness.

  To address these challenges, organizations should introduce structured operations management, conduct regular reviews, and refine their tagging strategies through ongoing governance. Recommended approaches include:

  a. **Establish KPIs for improvement**: The primary goal of cost tagging is accurate cost allocation. It is advisable to define the Cost Allocation Ratio as a KPI, which measures tag coverage across resources. A higher ratio indicates improved efficiency in cost allocation and reporting as well as higher data reliability. During the governance process, tracking trends in the Cost Allocable Ratio helps assess whether tagging coverage within the organization is improving or declining.

  b. **Identifying tag deficiencies and errors**: Once the need for tag governance is determined, the first step is identifying all resources that are not tagged and those with incorrect tag key-value pairs. Governance should prioritize high-cost resources before gradually expanding to others. Organizations can use tools provided by cloud providers or develop tools by themselves to assess tag compliance during resource creation. A more robust approach integrates permissions management to identify resource creators and their organizational units and to automatically assign appropriate tags.

  c. **Regular review and optimization of specifications**: Tagging is not a one-time setup. Regularly auditing and refining tagging specifications ensures that cost tags remain adaptable to changing environments and evolving stakeholder demands. It is necessary to adjust tagging granularity based on feedback and expectations from management and finance teams. It is also essential to review and update tagging specifications in collaboration with stakeholders.

- **Related services and tools**

  You can view the proportion of allocated costs in **Cost Center** to assess tag coverage and enhance tag management.

 Copyright © Huawei Technologies Co., Ltd.

You can identify and manage untagged or wrongly tagged resources using **Cost Center**, TMS, and cloud service consoles.

You can use **resource compliance policies** preset in Config to identify non-compliant resources, such as resources without tags.

You can use **tag policies** preset in Organizations to standardize tags added to resources in your organization.

5.8.2 COST04-02 Proactively Monitoring Costs

- **Risk level**

  Medium

- **Key strategies**

  Monitor your costs and usage regularly, not just after the bill is generated or when you receive an anomaly notification. Regular monitoring and proactive cost analysis can help you identify cost trends in a timely manner and prevent any anomalies.

- **Related services and tools**

  You can create a **budget alert** and set a budget threshold, so you can receive notifications when the forecasted or actual costs exceed the set threshold. This helps you prevent potential cost overruns.

  You can create **cost monitors**. Huawei Cloud Cost Center **Cost Anomaly Detection** uses machine learning to model your historical expenditures. Cost growth that does not comply with the model are identified as cost anomalies, and root causes for the anomalies are provided. You can set monitoring alerts to periodically obtain anomaly records and respond quickly to maintain the expected costs.

  In Billing Center, you can enable available quota monitoring threshold. If the available quota falls below the threshold, an alert will be sent, preventing service interruptions due to quota exhaustion.

  You can enable **resource package monitoring**. If the remaining resource package quota is insufficient, an alert will be sent, preventing the resource package from being automatically changed to pay-per-use billing.

  You can use **preconfigured reports** for cost analysis or create commonly used cost analysis reports to quickly learn cost breakdowns and trends.

5.9 COST05 Setting Optimization Strategies and Objectives

5.9.1 COST05-01 Analyzing Service Trends and Optimization Benefits

- **Risk level**

  High

- **Key strategies**

Cloud cost management is a comprehensive initiative that requires regular review, assessment, and execution. While keeping in mind the optimization benefits, consider the associated costs, such as labor and time costs.

To reduce the overall cost, ensure that the optimization workload remains proportional to the potential savings. During cost optimization, consider the proportion of application costs relative to the total cost. For example, an application that accounts for 50% of the total cost should be reviewed more frequently and thoroughly than an application that accounts for 5% of the total cost. Another factor to consider when optimizing is the amount of work required to implement changes. If change testing and verification costs are high, do not frequently apply optimizations. Instead, consider using substitution automation testing and verification to further reduce labor costs.

As there are potential reductions in resource redundancy from cost optimization, the service trend should be taken into account. For example, a rapidly growing business organization may prioritize agility and adopt a more flexible cloud cost optimization strategy to accelerate growth, while a stable business organization may focus more on cost efficiency and design a stricter cloud cost optimization strategy. If your application targets a specific geographic location or market segment and you have predicted a decline in cloud service usage, increasing the optimization frequency may save more costs.

5.9.2 COST05-02 Establishing Measurable Optimization Objectives

- **Risk level**

  High

- **Key strategies**

  Cost optimization is a strategic investment and an ongoing process. To effectively showcase the value of cost optimization to decision-makers and stakeholders, it is essential to quantify the cost savings, especially the objectives of cost optimization. With clearly defined objectives, you can gain support from decision-makers and stakeholders for continuous optimization activities while establishing a structured process framework to measure cost optimization outcomes.

  A straightforward and measurable cost optimization objective is to report cost savings. For example, you can generate a report illustrating how much the company or organization can save without compromising quality or output.

  Efficiency improvement can also serve as a measurable optimization objective. For example, during the transition from the traditional IT architecture to containerization and serverless, resource utilization is improved, service development and deployment are accelerated, and thereby the response time to the market is shortened and workforce efficiency is enchained.

  The ultimate goal of cost optimization is to maximize the benefits of every dollar spent by enterprises or organizations. Do not focus only on reducing costs and ignore business value. Setting clear and measurable optimization objectives helps the cost optimization team mentioned, decision makers, and stakeholders to align and reach a consensus effectively.

 Copyright © Huawei Technologies Co., Ltd.

5.9.3 COST05-03 Regularly Reviewing and Verifying Optimization Results

- **Risk level**

  High

- **Key strategies**

  To ensure that your cloud applications are always cost-effective, you are advised to regularly review and verify cost optimization results to determine whether there are opportunities for new optimization measures.

  You can conduct review and verification at the application level based on cost allocation principles to continuously review the overall cost of each application. You can calculate the total cost of ownership (TCO) by taking into account cloud resource costs, R&D costs, and operations management costs (for example, hosted cloud services vs. non-hosted cloud services). The workload of verification should align with the expected benefits (for example, the analysis time is proportional to the application cost) and whether the associated costs bring positive revenue.

  When determining the frequency of review and verification, you should consider multiple factors, including the importance of cost optimization within enterprises or organizations, costs associated with test and verification, application complexity, and the adaptability of optimization changes. In addition, you can continuously improve the process, for example, by reducing the cost of testing and changes to increase the frequency of the overall optimization. Finally, after new services, resource types, and configurations are launched, you can also start the process to evaluate them to optimize the costs of your workloads.

5.10 COST06 Choosing the Right Billing Mode

5.10.1 COST06-01 Understanding the Characteristics of Different Billing Modes on the Cloud

- **Risk level**

  High

- **Key strategies**

  Huawei Cloud provides flexible billing modes, such as pay-per-use, yearly/monthly, resource package, and spot pricing, each suitable for different scenarios. You need to understand the characteristics of different billing modes and select the most appropriate one based on your needs to adapt to different business forms and reduce costs.

  – Pay-per-use: You may only need to use cloud resources temporarily or for unplanned, urgent purposes.

  – Yearly/monthly: You want to cut down costs by choosing a long-term subscription, and your demand is stable during the subscription term.

  – Resource package: Similar to a yearly/monthly subscription, you can buy a resource package in advance to cover your long-term usage of specific resource types.

 Copyright © Huawei Technologies Co., Ltd.

  - Spot pricing: You do not have high requirements on service stability and can handle occasional interruptions.

5.10.2 COST06-02 Selecting an Appropriate Billing Mode for Your Workloads

- **Risk level**

  Medium

- **Key strategies**

  Analyze each workload component to determine the most suitable billing mode for resources. For long-running components, consider committed discounts, changing to yearly or monthly billing, or purchasing resource packages. For short-term, dynamic workloads, opt for spot pricing or pay-per-use billing. Use the recommendations provided by cost management tools to analyze your workloads and apply them to maximize your savings.

- **Related services and tools**

  Huawei Cloud provides you with optimization recommendations for multiple billing modes based on your past usage to help you save costs while maintaining resource performance. You are advised to focus on optimization recommendations with high savings (large value for **Estimated Monthly Savings**) and low risks (small value for **Break-Even Time**).

  Cost optimization evaluation for changing pay-per-use billing to yearly/monthly: Automatically identifies long-running resources (such as ECS, EVS, RDS, ELB, and SFS Turbo) and generates optimization recommendations and estimated savings for changing from pay-per-use to yearly/monthly.

  Recommendations for purchasing resource package: Automatically analyzes your usage of pay-per-use resources and the coverage of Huawei Cloud resource packages (such as OBS Standard single-AZ storage packages, OBS Standard multi-AZ storage packages, OBS Internet outbound traffic packages, and SFS storage resource packages) and generates purchase recommendations and estimated savings for resource packages

5.10.3 COST06-03 Tracking and Monitoring the Usage of Special Offerings

- **Risk level**

  Low

- **Key strategies**

  After purchasing special offerings, such as resource packages, you should track their usage regularly. If a resource package is about to expire or has been used up, renew it in a timely manner. If the resource package coverage is low, purchase more resource packages. If the resource package utilization is low, purchase a resource package of a proper size upon expiration to avoid waste.

- **Related services and tools**

  Huawei Cloud Cost Center provides the utilization and coverage analysis of resource packages. You can use this function to learn about the utilization and coverage of purchased resource packages, and determine whether you

 Copyright © Huawei Technologies Co., Ltd.

have over-purchased (low utilization) or under-purchased (low coverage) to optimize future purchases.

Huawei Cloud Billing Center provides **remaining usage alerting for resource packages**. You can set a threshold based on the remaining usage percentage or absolute value, set a custom threshold to obtain timely alerts.

5.11 COST07 Managing and Optimizing Resources

5.11.1 COST07-01 Continuously Monitoring Resource Utilization

- **Risk level**

  High

- **Key strategies**

  Continuously define core resource usage metrics in your organization to identify patterns over different time periods (daily, weekly, or monthly), and review applications/projects with underutilized resources. The metrics include CPU utilization, memory consumption, CDN traffic, and database TPS.

5.11.2 COST07-02 Releasing Idle Resources

- **Risk level**

  Medium

- **Key strategies**

  Identify idle resources, such as load balancers with no traffic, unattached EVS disks, and unbound EIPs. If detected, release them to optimize resource utilization. Alternatively, monitor resource usage at fixed intervals (for example, daily at 12:00:00 or every weekend) and automate resource allocation as needed and release them once they are no longer needed.

- **Related services and tools**

  Huawei Cloud Optimization Advisor provides cost-based inspection to identify idle resources, such as ECSs, EIPs, EVS disks, and load balancers.

  Huawei Cloud **Cost Center** identifies idle resources, such as ECSs, EIPs, EVS disks, and load balancers, and also provides estimated savings based on historical expenditures. You can decide whether to apply the recommendations based on the resource utilization, estimated monthly savings, and feedback from service teams.

5.11.3 COST07-03 Conducting Comprehensive Cloud Technology Selection Analysis

- **Risk level**

  Medium

- **Key strategies**

  Regularly consult experts or Huawei partners to determine which services and functions are more cost-effective. You can also check Huawei blogs and other

 Copyright © Huawei Technologies Co., Ltd.

related sources. For example, you can replace common ECSs with Flexus instances in non-computing-intensive scenarios.

5.11.4 COST07-04 Changing Specifications Based on Workload Patterns and Resource Utilization

- **Risk level**

  Medium

- **Key strategies**

  Select proper resource size or type based on workload and resource characteristics. You can continuously monitor resource utilization. If it is lower than the set threshold, downgrade the resource specifications. If it is higher than the set threshold, upgrade the resource specifications.

5.12 COST08 Optimizing Architectures

5.12.1 COST08-02 Implementing Cloud-Native Architecture Transformation

- **Risk level**

  Medium

- **Key strategies**

  The cloud-native architecture modernizes applications by transitioning to containerized and microservice-based designs. It leverages cloud-native advantages such as automatic elastic scaling. Containerization can improve resource utilization and avoid idle resources, reducing computing costs, and microservice-based applications can streamline O&M, lowering O&M costs.

  Online services, such as advertising and e-commerce, have high SLA requirements and peak hours, resulting in dynamic resource allocation. In contrast, offline services, such as big data and transcoding, require strong computing power but can tolerate delays and occasional errors. To ensure the performance and stability of online services, resources are usually requested during peak hours. As a result, resources are wasted during off-peak hours. Hybrid deployment of online and offline services can effectively improve the overall utilization.

5.12.2 COST08-03 Decoupling Storage and Compute

- **Risk level**

  Medium

- **Key strategies**

  In traditional big data solutions, compute and storage are deployed together. Compute nodes must be added when disks are added, which is a waste of resources. Storage-compute decoupling is a data processing technology that separates data storage from data processing (compute). This technology allows independent optimization and scaling of storage and compute. It improves data processing efficiency, reduces costs, and meets the requirements of large-scale data storage and analysis.

 Copyright © Huawei Technologies Co., Ltd.

For example, in log analysis of a shopping guide website, storage capacity is frequently expanded, but compute resources are underutilized as demand remains relatively stable. Conversely, in an Internet recommendation service, storage capacity increases steadily and linearly while compute requirements experience sharp fluctuations, peaking at dozens of times the trough levels. This results in low utilization of compute resources, limiting flexibility in resource allocation. Using object storage instead of HDFS and local disks separates storage and compute. This ensures on-demand usage of compute and storage resources, preventing unnecessary resource binding and reducing costs by 30%.

5.12.3 COST08-04 Exploring Serverless

- **Risk level**

  Low

- **Key strategies**

  Serverless is the next-generation cloud-native paradigm. It simplifies development and O&M and reduces resource costs. The serverless architecture maximizes the utilization of compute, storage, and network resources, accelerates requirement release, and improves application R&D efficiency.

  You can use Cloud Eye to monitor the CPU and memory usage of serverless instances. Based on predefined conditions, the serverless computing power is automatically scaled out or in, improving resource utilization and reducing costs.

- **Related services and tools**

  Huawei Cloud products in serverless model include Object Storage Service (OBS), Cloud Application Engine (CAE), Cloud Container Instance (CCI), and FunctionGraph.

5.13 Cloud Services Used for Cost Optimization

Cloud Services Used for Cost Optimization

- **Cost Center** is a free cloud financial management service provided by Huawei Cloud. It offers a suite of tools to help you track, analyze, and explore your Huawei Cloud costs and usage. It also helps you detect cost anomalies to reduce unexpected expenditure spikes and find cost-saving opportunities.
- **Billing Center** provides accounting information, invoices, contracts, renewals, unsubscriptions, and changes so you can understand your expenditures better. In addition, Billing Center provides balance and resource package alerts to help you learn about your expenditures in a timely manner and maintain better expenditure control.
- **Enterprise Center** provides financial management services for large enterprises in a multi-account environment. It helps enterprises manage people, finance, and resources in a multi-level organization structure, meeting enterprise IT governance requirements. Enterprise Center supports two multi-account accounting management modes: unified accounting and independent accounting.
- **Enterprise Project Management (EPS)** allows you to manage people, finance, and resources in a single account. You can plan enterprise projects

 Copyright © Huawei Technologies Co., Ltd.

based on the organizational structure, centrally manage resources across different regions by enterprise project, and create user groups and users with different permissions for each enterprise project.

- **Tag Management Service (TMS)** is a visualized service that allows you to efficiently and centrally manage tags and categorize cloud resources across regions and services.
- **Optimize Adviser** analyzes your resource configurations and usage against Huawei Cloud best practices and provides self-service checks and optimization recommendations in terms of reliability, security, and performance. This helps you achieve efficient operations and cost savings.

 Copyright © Huawei Technologies Co., Ltd.

6 Operational Excellence Pillar

6.1 Operational Excellence

Operational excellence represents high quality, high efficiency, and sustainable customer experience. Operational excellence improves design, development, test, deployment, release, and O&M activities for high-quality delivery, and promotes continuous integration and continuous delivery (CI/CD). In addition, it helps build a deterministic O&M system, so that the R&D team can focus on developing valuable features instead of managing emergencies. In this way, operational excellence ensures smooth system running, balanced workloads, and better customer experiences. Operational excellence integrates best practices to focus on software development, efficient operations and maintenance, and providing excellent customer experiences. These practices cover team organization, workload design, handling large-scale operational workloads, and improving workloads over time.

6.2 Concepts

Term	Definition
Deterministic Operations	Deterministic operations is intended to create an O&M management system that makes risks avoidable, controllable, and manageable. Deterministic operations can help minimize risks through high-quality R&D and well-designed O&M processes and regulations. Technical measures are taken to make faults rectifiable, controllable, and preventable, and make their intervals, impact scope, and recovery time predictable. The uncertainty associated with digital transformation is reduced.

 Copyright © Huawei Technologies Co., Ltd.

Term	Definition
Infrastructure as code (IaC)	Infrastructure as Code (IaC) lets you manage infrastructure with code rather than manual interventions. Modern application environments depend on a variety of components, from operating systems and database connections to storage configurations. Developers must set up, update, and maintain infrastructure on a regular basis to develop, test, and deploy applications. Manual infrastructure management is time-consuming and error-prone, especially when managing applications at scale.
CI/CD	Continuous Integration (CI) promotes frequent small code commits to version control, with automated builds, tests, and packaging to improve collaboration and code quality. Continuous Delivery (CD) automates deploying code to environments (production, development, and testing) after CI completes.
Telemetering	Telemetry is the remote measurement and transmission of data from a target to a receiving station for recording, display, and analysis.
CMDB	The configuration management database (CMDB) is an information technology infrastructure library (ITIL) term. It is a database used by organizations to store information about software and hardware assets (usually called CIs). It tracks the status of assets (products, systems, software, devices, personnel), their relationships, and supports IT management of various service data consumption through open APIs.
MTTR	MTTR (Mean Time to Repair) is the average time from the occurrence of a fault to the confirmation of fault recovery. It encompasses three dimensions: Mean Time to Identify (MTTI), which measures the time to detect a fault; Mean Time to Know (MTTK), which assesses the time to diagnose the fault; and Mean Time to Fix (MTTF), which quantifies the time taken to implement a resolution.
Change risk control	Involves pre-event check, in-event interception, and post-event verification to prevent abnormal behaviors.
Safe production	Safe production aims to ensure the continuous stability, security, and quality of live networks by implementing E2E management across personnel, tools, product capabilities, and processes. This includes safety prevention, real-time monitoring, and post-audit checks to minimize or prevent network failures, with a key focus on avoiding incidents triggered by abnormal behaviors.
Rapid fault recovery	Rapid fault recovery leverages a library of predefined failure patterns to develop emergency plans, enhancing efficiency and reducing recovery time. The duration of uncertain recovery is made certain through chaos engineering drills.

 Copyright © Huawei Technologies Co., Ltd.

Term	Definition
Resource life cycle management	Resource application, creation, delivery, O&M, and final destruction and release.
Fault drills	Fault drills involve replicating common failure scenarios in a controlled environment to expose vulnerabilities. Through continuous drills and regression testing, they validate and improve systems, tools, processes, and team capabilities. This proactive approach identifies and resolves preventable critical issues in advance or verifies fault detection and repair mechanisms to shorten recovery times.
Managed O&M	Managed O&M is a professional service that comprehensively manages and maintains the IT infrastructure of enterprises or organizations. It aims to improve the availability, reliability, and security of IT systems. This service covers multiple aspects, including system monitoring, troubleshooting, system optimization, and security protection.

6.3 Design Principles

Building a Culture of Continuous Improvement with a Standardized O&M System

A strong team culture is essential for operational excellence. Operations is a continuous journey of improvement. Continuous learning from incidents and ongoing improvement are key to achieving operational excellence. To support continuous growth, teams should cultivate a culture of learning and improvement. When incidents arise, the priority should be system enhancement rather than individual blame. Punishing individuals often leads to negative consequences. For example, the O&M team may conceal incidents or their root causes out of fear, hindering improvements to system processes and operational capabilities. Fear of punishment can discourage frontline engineers from taking responsibility or initiating system changes. Similarly, departments may shift blame to others to protect their own teams. This creates a culture of fear, leading to organizational rigidity and stagnant operations. Without continuous iteration, code and architecture deteriorate, eventually making the system unmanageable. Culturally, we should emphasize learning and assigning responsibility at the organizational level, not blaming individuals.

Experience should be standardized by translating it into automation tools, processes, and organizational systems—forming a standardized O&M system. Non-standard O&M operations hinder large-scale success. Disorganized operations, with team members relying on individual approaches, lead to passive responses, low efficiency, frequent errors, and difficult troubleshooting. Standardized O&M is an efficient management approach that consolidates best practices and routine operations. By implementing standardized, process-driven, and tool-based O&M management, the O&M team can shift from disorder to

 Copyright © Huawei Technologies Co., Ltd.

order, simplify delivery, reduce skill dependency, improve efficiency, and lower operational costs.

Frequent, Small, Reversible Changes Through CI/CD

In software development, the phases of requirement analysis, design, development, testing, and deployment should be as short as possible, with frequent small iterative changes. Microservices, coupled with CI/CD, are a widely adopted practice. Known for their flexibility, scalability, and ease of maintenance, the microservice architecture have become the optimal choice for modern application development. It breaks an application into small, independent services, each handling specific business functions. These services can use different technology stacks and are developed, tested, deployed, and scaled by separate teams. They communicate through lightweight mechanisms. In CI/CD, the same team manages the development, testing, deployment, release, and O&M of microservices across different regions using pipelines.

DevOps-adopting organizations should extend the model beyond software project management. From an O&M perspective, frequent small iterations enable faster issue detection and easier rollbacks, minimizing the risk of widespread failures in the event of deployment issues.

X as Code: Automate Where Possible

Unlike traditional applications, a cloud application is defined as code—including the application itself, cloud infrastructure, security policies, and O&M. This means all aspects of operational excellence can be automated through code. For example, you can define infrastructure, deploy applications, update configurations, set security policies, and even automate routine O&M tasks and fault resolution.

Automation is key to building O&M expertise and establishing standardized practices. It limits human error, streamlines processes, and boosts efficiency. Even when manual intervention is needed—such as at decision points —automation can first verify permissions and provide the necessary context to support informed decisions, significantly reducing errors compared to fully manual processes.

Continuous Improvement Through Observability

Observability is the ability to infer the internal state of a system by analyzing its external output. The observability of cloud applications is typically assessed through three core approaches. The first is metrics—quantitative measurements of system performance, such as TPS, request latency, and number of calls. The second is logs—human-readable records of system events, including runtime information, errors, and security incidents. The third is traces—the path of a single request or transaction in the system, helping reveal how the system executes.

For cloud-based applications, observability quickly detects and resolves system faults, improving system recovery time. Observability also allows for the early detection of system issues, such as performance and capacity bottlenecks. Additionally, alarms triggered by observability can be linked to automated processes, enabling proactive responses like dynamic scaling, flow control, traffic switching, and node migration to resolve issues.

 Copyright © Huawei Technologies Co., Ltd.

6.4 Questions and Checklists

This checklist can help identify areas for improvement and guide their enhancement. Each item in the checklist represents a best practice, which will be explained in detail in the next section.

Question	Checklist/Best Practice
OPS01 Have you established a culture of continuous improvement with a standardized O&M system?	1. Build a growing culture. 2. Design a standardized O&M organization. 3. Standardize O&M processes and tools.
OPS02 Do you use CI/CD to implement frequent, small, and reversible changes?	1. Manage requirements and sprint development. 2. Link source code versions to deployed applications and applying code quality best practices.
OPS03 Do you have a comprehensive test and verification system?	1. Promote developer testing. 2. Perform integration tests in multiple environments and build a pre-production environment that is identical to the production environment. 3. Performance stress testing 4. Perform dialing test in the production environment. 5. Perform chaos testing and drills.
OPS04 Is the automated build and deployment process complete?	1. Effectively implement continuous integration. 2. Use continuous deployment models. 3. Implement infrastructure as code. 4. Perform automated O&M tasks.
OPS05 Is there an O&M preparation and change management system?	1. Conduct production readiness review. 2. Conduct change risk control. 3. Define a change process.

 Copyright © Huawei Technologies Co., Ltd.

OPS06 Has an observability system established?	1. Establish an observability system. 2. Define observable objects. 3. Develop and implement observability metrics 4. Standardize application logs. 5. Implement dependency telemetry. 6. Implement distributed tracing. 7. Introduce automation measures based on observability metrics.
OPS07 Is fault analysis and management conducted?	1. Create alarms. 2. Create a dashboard. 3. Support event management. 4. Support fault recovery process.
OPS08 Is there an operating status measurement and continuous improvement mechanism?	1. Use metrics to measure operational objectives. 2. Review the accident and make improvements. 3. Manage knowledge.

6.5 OPS01 Building a Culture of Continuous Improvement with a Standardized O&M System

6.5.1 OPS01-01 Building a Growing Culture

- **Risk level**

  High

- **Key strategies**

  There is no one-size-fits-all solution—every system is unique and complex. To achieve operational excellence, continuously refine existing practices and develop ones that best fit your environment. Therefore, the first step in any best practice is to build a team culture of continuous learning and improvement.

  Continuous learning and improvement rely on open communication and knowledge sharing. For example, best practices developed within your company/organization should be widely shared. Incident analysis should be documented to ensure root causes are fully understood. It is also critical to develop and promote standard processes and automation tools to reduce the likelihood and impact of future incidents—along with clear explanations so the team understands the 'why' behind them.

6.5.2 OPS01-02 Designing a Standardized O&M Organization

- **Risk level**

High

- **Key strategies**

  To achieve operational excellence, you need an O&M organization tailored to your specific needs. A well-structured O&M organization defines clear collaboration processes between teams—such as response times, service level objective (SLO), or service level agreement (SLA). It is also important to record inter-team communication to ensure there is enough data for future improvements.

  For example, one common O&M structure divides support into three tiers: frontline, second-line, and third-line teams. The Tier 1 team handles customer service requests and resolves most issues at the first point of contact. The second-line team is the engineers who handle requests escalated from the frontline. They resolve known issues within the SLA and escalate software defects to Tier 3. Their responsibilities include managing alarms, incidents, and faults, as well as executing maintenance handovers, emergency response plans, and drills. Their primary goal is to ensure the stability and availability of the live network. The third-line team focuses on rectifying software version defects.

  In a DevOps model, developers may take direct ownership of system operations and maintenance. A lean, expert enablement team should support continuous improvement and drive the implementation of process and automation tools across the organization.

  No matter how the organization is set up, there must be an overall process, and each team and member in the process has clear responsibilities. Additionally, you use clear methods—such as collecting operations and O&M data—to analyze how team efforts impact business outcomes, prioritize tasks effectively.

6.5.3 OPS01-03 Standardizing O&M Processes and Tools

- **Risk level**

  High

- **Key strategies**

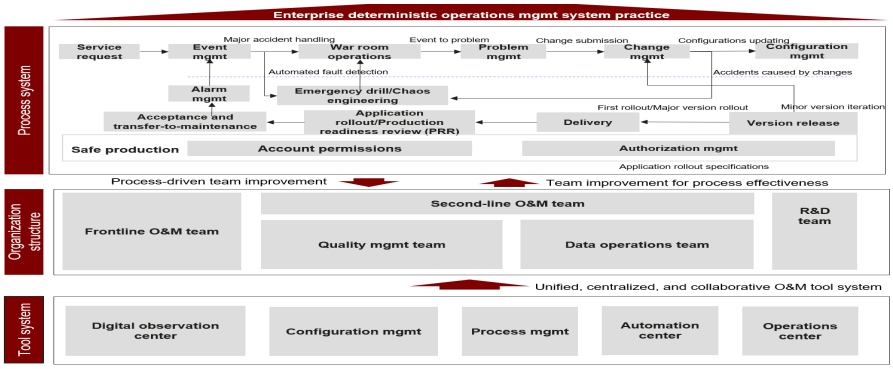
  Processes and tools embody operational experience. Standardizing them helps reduce the impact of individual factors and disorder during O&M. Standardized O&M tools offer a centralized interface and clear, user-friendly manuals, streamlining maintenance and boosting efficiency. Common O&M processes are as follows:

  - **Change management process**: Design to manage software and hardware changes in the production environment to avoid unexpected service interruptions or quality loss caused by changes. This can ensure that the enterprise environment is secure and stable while also improving system availability and meeting SLA requirements.
  - **Alarm and event management process**: Cover the acceptance, handling, and escalation of faults in both development and production environments. It ensures timely responses to user services and supports SLA fulfillment. Clear definitions of event levels, responsibilities, response times, and notification mechanisms are essential for maintaining service security and stability.

  - **Issue and backtracking process**: Apply to event review and analysis, identify root causes of faults, and implement solutions to prevent recurrence and minimize related impacts. Effective problem management improves product quality, enhances stability, and reduces faults in the live network.
  - **Product Readiness Review (PRR)**: Evaluate whether cloud services are ready for release by identifying any issues in the production environment and during the O&M phase. Note that, due to the iterative nature of cloud applications, product availability should be evaluated not only at the time of service release but also periodically or triggered by major events, such as e-commerce promotions.

  In addition, processes such as IT service and account management can be standardized using cloud tools like pipelines, monitoring and alarm reporting, log processing, and an O&M center. This standardization drives operational excellence for your enterprise. The best practices for key processes (change management, alarm and event management, issue and backtracking, and product readiness review) in the preceding sections are also described in other sections of this white paper.

- **Design suggestions:**



- **Related cloud services and tools**

  Cloud Operations Center (COC)

  Huawei Cloud Application Operations Management (AOM)

  Cloud Eye

  Huawei Cloud Log Tank Service (LTS)

  Application Performance Management (APM)

6.6 OPS02 Frequent, Small, Reversible Changes Through CI/CD

6.6.1 OPS02-01 Managing Requirements and Sprint Development

- **Risk level**

  High

- **Key strategies**

To ensure excellent cloud operations, prioritize availability, recoverability, and code quality during the design and development phases. You need to evaluate and understand the DFX requirements of software, including reliability, performance, serviceability, O&M, and deliverability, while incorporating regulatory, industry, and internal compliance requirements. In addition, Ensure adequate time and attention are allocated when prioritizing these requirements.

From a maintainability perspective, continuous iteration of cloud software updates is more O&M-friendly than disruptive one-time releases. Smaller updates reduce the risk of major incidents and keep development and O&M teams actively engaged, ensuring familiarity with processes and best practices. To enable iterative updates of cloud applications, sprint planning and tracking should start from the requirement phase, with iteration plans driven by specific requirements.

- **Related cloud services and tools**

  Huawei Cloud CodeArts Req

6.6.2 OPS02-02 Linking Source Code Versions to Deployed Applications and Applying Code Quality Best Practices

- **Risk level**

  High

- **Key strategies**

  In the code development phase, collaborative code design and management are required during development. Use a modern code repository to manage code. Ensure that the code is consistent after being merged and no change is lost. With proper version control, the modern code repository allows you to manage code versions, link source code to deployed application versions, and trace back to the source code once an application deployed on the cloud encounters any problem. If needed, you can roll back to a previous version using the corresponding source code.

  Throughout the software development lifecycle, developers should follow best practices like code reviews and pair programming to maintain code quality. Each line of code should be reviewed by at least two engineers before being merged. Enforce review standards by setting merge rules in the code repository. Finally, you are advised to use automated code checks to check code quality.

- **Related cloud services and tools**

  Huawei Cloud CodeArts Check

  Huawei Cloud CodeArts Repo

6.7 OPS03 Comprehensive Test and Verification System

6.7.1 OPS03-01 Promoting Developer Testing

- **Risk level**

  High

 Copyright © Huawei Technologies Co., Ltd.

- **Key strategies**

  Developer testing is a critical pillar of modern software engineering. Generally, developers can run tests repeatedly, whether on a local host or within the build phase, with minimal dependencies. This approach is exceptionally cost-effective at identifying potential issues early, especially when handling abnormal scenarios or unexpected user inputs. In essence, this approach compels developers to consider various usage scenarios in advance, significantly reducing the system load during later O&M.

  In the ever-evolving cloud landscape, software is continuously refined and refactored. Modifying legacy system code can be intimidating due to uncertainties about the potential ripple effects across interdependent modules. However, developer testing provides rapid feedback by immediately evaluating the impact of every adjustment. This instant feedback allows developers to confidently evolve code without unintended consequences.

  Moreover, it is essential to establish clear and suitable testing standards, such as setting targets for code and branch coverage, to evaluate your software.

6.7.2 OPS03-02 Performing Integration Testing in Multiple Environments and Mimicking Production with Staging Environments

- **Risk level**

  High

- **Key strategies**

  While developer testing is cost-effective, it does not verify the production environment configurations or the interactions across services and applications. To overcome these gaps, organizations can establish multiple cloud environments, including test, staging, and production environments. The test environment is ideal for joint commissioning to verify cross-team service interactions in coding, even though its configuration differs from that of the production environment. In contrast, the staging environment mirrors the production setup with identical deployment configurations, security controls, processes, and programs, ensuring that every deployment step, from dependency data verification to configuration and service checks, is executed as expected during the release process. Additionally, every change undergoes an integration testing process that combines functional tests, non-functional tests, and continuous monitoring techniques such as health checks.

6.7.3 OPS03-03 Performing Performance Testing

- **Risk level**

  High

- **Key strategies**

  System performance metrics are tested by simulating various routine, peak, and abnormal loads with automated testing tools. Performance testing primarily comprises load and stress tests. Essentially, performance testing is a structured series of tests that verify if the system can handle user needs effectively.

  Before application release, performance testing assesses maximum concurrent requests, response time, and stability under intense conditions, ensuring that

 Copyright © Huawei Technologies Co., Ltd.

the system meets the needs for the workload. The insights allow proper infrastructure configuration to optimize resource usage. The following performance test types are essential:

Load testing: simulates activities of one or more services within a given software-hardware-network ecosystem. It validates that performance metrics stay within acceptable ranges under projected virtual users. Load testing determines the user capacity, valid user threshold, optimal response time, and evaluates resource utilization, revealing the system's peak performance potential.

Stress testing: applies prolonged, extreme loads by simulating a vast number of virtual users within a given software-hardware-network ecosystem. This continuous pressure evaluates whether hardware devices or cloud services remain stable under severe loads, focusing on the system's reliability beyond normal operation limits.

Capacity testing: simulates different scales of test data and transactions within a given software-hardware-network ecosystem. In this way, it gathers performance metrics across different hardware or cloud service quantities under a set number of virtual users. These insights help determine both the optimal and maximum capacity of a service system.

Concurrency testing: checks for deadlocks and other performance issues when multiple users access the same application, module, or data record simultaneously. Given its time-sensitive nature, concurrency testing uses tools to simulate concurrent operations of virtual users through multi-threading or multi-processing approaches. Generally, it is part of performance testing.

Configuration testing: simulates a set number of virtual users running one or more services within a given software-hardware-network ecosystem. The test results establish a baseline. Later, when the system undergoes optimization or evaluation, the same service scenario is executed again. Comparing the new results against the baseline helps to verify if the enhancements have met expectations and gather the necessary data to make informed decisions on system selection.

Before performing performance testing, it is essential to choose the right testing tool for simulation and reproduction of specific scenarios. Each tool offers unique functions, such as static and dynamic resource testing, stress and load testing, end-to-end service request/response timing and metering, and testing of web applications, mobile apps, and APIs.

- **Related cloud services and tools**

  Huawei Cloud CodeArts PerfTest

6.7.4 OPS03-04 Performing Echo Testing in the Production Environment

- **Risk level**

  High

- **Key strategies**

  Echo testing simulates user actions on the primary software system. It verifies how this system interacts with external services, whether they are represented by separate accounts or distributed across various cloud regions. Echo testing regularly assesses key performance aspects, such as network quality, page

 Copyright © Huawei Technologies Co., Ltd.

performance, port performance, file transfer, and audio/video experience. By analyzing these metrics from multiple angles, visual insights are produced to inform rapid action to maintain high-quality and reliable service operations.

6.7.5 OPS03-05 Performing Chaos Testing and Drills

Chaos engineering is a practice of deliberately injecting faults into a system to test its resilience and reliability.

- **Risk level**

  High

- **Key strategies**

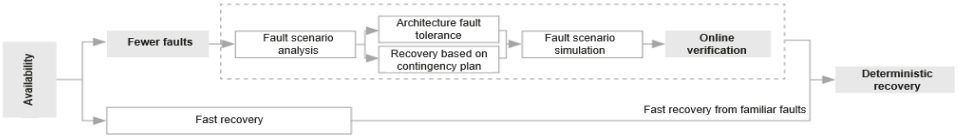
  Chaos engineering introduces faults into the system to assess its robustness under various failure scenarios. This approach tests system capabilities in fault tolerance, monitoring, emergency response, demarcation and locating, and rapid recovery.

  High availability design and verification: During the planning and design phase, the service system's architecture is crafted with HA and monitoring. Before launch, production readiness review (PRR) and performance testing verify that the system consistently delivers stable and reliable services. Chaos engineering constructs drill scenarios that examine multiple facets of the system, from application deployment architecture and service capacity to monitoring, alarms, and HA. These drills follow a structured progression: beginning with testing, advancing through attack-defense scenarios, and culminating in surprise attacks. Continuous drills verify critical capabilities online, such as architecture HA, monitoring, and PRR, to establish dynamic, ongoing risk governance. Chaos drills and HA design operate as dual engines to ensure enduring system stability.

  System risk mitigation and rapid service recovery: This involves analyzing potential risks (fault scenarios) and designing contingency plans. Then, it checks the fault scenario coverage and hit rate, and validates both the coverage and execution efficiency of these contingency plans. The outcome is a streamlined approach that minimizes faults and accelerates recovery, ultimately realizing deterministic recovery.

  Fewer faults: Identifying potential risks early is critical. This entails assessing the severity and impact of each fault scenario and testing the ability to mitigate these risks through regular drills.

  Fast recovery: By proactively introducing failures, O&M and development teams become well-acquainted with various fault scenarios and can validate contingency plans. This practice ultimately accelerates the recovery process.

  Chaos engineering metrics:

  – Fault scenario coverage: analyzes the extent that fault scenarios are covered. For example, the redundancy scenario coverage is 80% and the overload scenario coverage is 60%.

  – Hit rate of a fault scenario: measures the actual occurrence of a fault scenario.

 Copyright © Huawei Technologies Co., Ltd.

  - Contingency plan quality: assesses both the effectiveness and execution efficiency of contingency plans.
  - Number and severity of identified risks: regularly, typically quarterly or annually, quantify and assess the severity of risks that are proactively identified.
  - Number, severity, and type of mitigated risks: track risk mitigation, including the number of risks downgraded or mitigated, the addition of contingency plans, and enhancements in monitored items.
  - Improvement rate of fault recovery duration: reflects how quickly the average recovery time improves as a result of chaos engineering drills.
  - Year-over-year (YoY) fault decrease: compares the total number of faults in the current year against the previous year, showing the decrease in the number of faults over time.
- **Related cloud services and tools**

  MAS: Chaos Engineering

  COC: Chaos Drills

6.8 OPS04 Automated Build and Deployment

6.8.1 OPS04-01 Effectively Implementing Continuous Integration

- **Risk level**

  High
- **Key strategies**

  Continuous integration (CI) is a software development practice where software changes are regularly merged into a source code control system. The process initiates with code commits to the repository, triggering automated workflows that verify code updates against multiple baselines and tests. Then, developers receive feedback on test outcomes. When baseline checks and tests succeed, assets, including compiled code and container images, are generated and temporarily stored for software deployment and upgrade.

  CI rapidly delivers high-quality software by:
  - Running automated tests on code to detect major changes as early as possible.
  - Analyzing code to ensure standardization, quality, and configuration.
  - Running compliance and security checks to identify and eliminate vulnerabilities.
  - Running acceptance or function tests to confirm that the software behaves as expected.
  - Highlighting issues for prompt resolution.
  - Packaging updated code into ready-to-deploy assets.
- **Related cloud services and tools**

  CodeArts Pipeline

 Copyright © Huawei Technologies Co., Ltd.

6.8.2 OPS04-02 Using Continuous Deployment Models

Continuous deployment models facilitate rapid fault detection, significantly reducing the impact on end users.

A popular model within this approach is canary deployment, where new functions are initially introduced to a limited set of internal or external users. Once the release is confirmed to be stable, it is gradually rolled out to larger groups until the entire user base has transitioned to the new version.

Another common model is blue-green deployment, which leverages two identical production environments. The blue environment serves all live traffic, while the green environment is updated with new features and tested internally. Once confirmed to be stable, production traffic is gradually shifted from blue to green, mirroring the controlled approach of canary deployment. When the transfer is complete, the green environment takes over as the primary production instance, and the former blue environment is prepped for future updates. The two instances are clearly separated to prevent faults across environments.

- **Risk level**

  High

- **Key strategies**

  When choosing a deployment model, it is essential to allocate sufficient time between each stage to monitor key workload health metrics. Avoid compressing deployments into minutes. Instead, extend the intervals to hours or even days. This pacing ensures users across various regions and tasks have ample time to interact with each version. Additionally, longer intervals can better accommodate differences in time zones and usage patterns.

- **Related cloud services and tools**

  CodeArts Deploy

6.8.3 OPS04-03 Implementing Infrastructure as Code

IaC lets you manage infrastructure with code rather than manual interventions. Modern application environments depend on a variety of components, from operating systems and database connection to storage. Traditionally, developers spend significant time setting up, updating, and maintaining these elements regularly to support application development, testing, and deployment. This manual approach not only slows progress but also increases the risk of human errors, particularly when managing large-scale applications.

- **Risk level**

  High

- **Key strategies**

  Using declarative tools: When deploying and managing IaC, opting for a declarative approach offers significant advantages over an imperative one. Declarative tools let you define the desired final state of your environment using simplified syntax. In contrast, imperative tools require you to detail every step needed to reach that state, which not only complicates your definition files but also increases the risk of accumulating technical debt as code (including deployment script) grows more cumbersome over time.

 Copyright © Huawei Technologies Co., Ltd.

Using cloud platforms and integrated, industry-proven tools: Tools built in a cloud platform cover most of your requirements, eliminating the need to develop custom solutions. Moreover, as the platform evolves, providers continually updates these tools to enhance their functionality and performance, simplifying your IaC deployments.

Standardizing modular solutions: Each module encapsulates complex configurations or resource sets into a repeatable, standardized unit that meets specific objectives. In non-sensitive scenarios, consider integrating open-source modules to facilitate development.

Standardizing manual steps: Reduce manual interventions during deployment and maintenance. Document all necessary manual tasks in O&M guides and standard operating procedures for secure, consistent execution.

Reclaiming idle resources: IaC tools and configuration management platforms sometimes do not automatically remove idle resources. For instance, during a migration from a VM to a PaaS service, unused resources can linger and become isolated if not deleted by IaC tools or manually pruned. To prevent such issues, establish a standardized process to routinely scan for idle resources and implement a clear deletion policy.

- **Related cloud services and tools**

  Resource Formation Service (RFS)

6.8.4 OPS04-04 Automating O&M Tasks

You should automate as many tasks as you can in your daily development work to ease management and reduce human errors. To maximize the value of automation investment, prioritize simple, procedural, and long-term tasks. Application automation is not an all-or-nothing strategy. Even workflows that require manual intervention, such as decision-making points, can benefit from automation.

- **Risk level**

  High

- **Key strategies**

  Focus on tasks that benefit most from automation:

  - **Tasks that are highly procedural and prone to human error:** These tasks are clearly defined, highly automated, have no variables that add complexity, and are executed as part of the normal path. Examples: restarting servers, creating accounts, and transferring logs to data storage. These tasks may be performed as scheduled in response to events or monitoring alarms, or initiated as required due to external factors.
  - **Tasks that O&M engineers currently handle:** Automated services are provided for application DevOps teams to automatically execute O&M operations using scripts. For example, in a multi-tenant solution, database administrators frequently get asked to create new databases. If a self-service portal is built for operation personnel, they can independently create empty databases.
  - **Tasks that can greatly enhance efficiency once automated:** High-value automation requires the least management overhead and greatly enhances efficiency. For example, if automating database entries can save

 Copyright © Huawei Technologies Co., Ltd.

the operations team an hour each day, they will have extra time to focus on enhancing processes through automation.

- **Design suggestions**
  - **Pipeline definition, execution, and management**: Use continuous integration and continuous delivery (CI/CD) tools, such as **CodeArts Pipeline**, to automatically define pipelines and their running modes.
  - **Deployment**: Use tools such as Huawei Cloud **Resource Formation Service (RFS)**, Terraform, and Ansible to automate workload development and release. The infrastructure as code (IaC) approach can be used to deploy and optimize infrastructure using the same automation platform.
  - **Testing**: Many tools are available to automate the testing process. These tools can reduce the burden of the quality assurance team and maintain that tests are standardized and reliable.
  - **Scaling**: Use the functions provided by the platform and other tools, such as **RFS**, to automatically scale the infrastructure when the load increases or decreases.
  - **Monitoring and alerting**: Use tools provided by **Cloud Operations Center (COC)** and **Cloud Eye** to automatically register newly deployed resources and configure alarm-triggered operations to speed up problem rectification.
  - **Self-healing**: Use alarms generated by **Cloud Eye** to automatically perform operations and recover faulty components or jobs.
  - **Configuration management**: Use orchestration and policy tools to maintain that all resources run the same configuration and adhere to compliance requirements throughout the workload.
  - **Other management tasks**: Use scripts to automatically perform repetitive tasks, such as updating database records or DNS records.
  - **Approval**: Enable the system to automatically make approval decisions based on predefined rules to improve the efficiency of workflows with approval checkpoints. This approach promotes the use of standardized forms and templates to enhance process efficiency. Automated approvals can be risky in high-risk environments. Monitor and test your automatic approvals to maintain that specific criteria are defined for granting approvals.
  - **Onboarding new user and new employee onboarding**: You can automate many tasks associated with onboarding new application users or new employees, such as database updates and credential creation.
- **Related cloud services and tools**

  RFS

  CodeArts Pipeline

  CodeArts Deploy

  Cloud Operations Center (COC)

  Cloud Eye

  KooCLI

 Copyright © Huawei Technologies Co., Ltd.

6.9 OPS05 O&M Preparation and Change Management

6.9.1 OPS05-01 Conducting a PRR

- **Risk level**

  High

- **Key strategies**

  Product readiness review (PRR) baseline is set for service availability and O&M capabilities from the following dimensions: SLI/SLO, redundancy, DR, overload control, fault management, change capability, O&M, and secure production. It allows the frontend personnel to perform requirement planning, design, and development, as well as the production admission review before service rollout.

  Its core values include:

  (1) Providing accurate evaluation of product availability and maintenance capabilities, specifying criteria for cloud service launch, and rejecting the launch of services that do not meet the criteria

  (2) Continuously adding serviceability and O&M baselines to improve the level of standardization, reduce operational exceptions, and accelerate service launch

  (3) Continuously improving automatic verification and reducing manual evaluation to make product delivery and O&M more efficient

- **Related cloud services and tools**

  COC PRR

6.9.2 OPS05-02 Conducting a Change Risk Control

- **Risk level**

  High

- **Key strategies**

  Successful changes are maintained by developing risk control capabilities for various change scenarios, assessing and analyzing risks digitally, evaluating their impact, and implementing control measures to minimize or prevent risks. Change risks include service disruptions caused by adding, deleting, or modifying elements on the live network and changing statuses (such as version iteration, configuration change, and node scaling), as well as service loss risks stemming from change failures.

  Design suggestions

  Change risk control metrics: These metrics are change incident rate and number of major incidents caused by changes.

  Change incident rate: The ratio of change incidents that impact customers per month to the total changes.

  Formula: Change incident rate = Number of incidents caused by changes/Total number of changes

  Major incidents caused by changes: Incidents that are cased by changes per month and have great impact on customers.

 Copyright © Huawei Technologies Co., Ltd.

- **Related cloud services and tools**

  COC change management

6.9.3 OPS05-03 Defining a Change Process

- **Risk level**

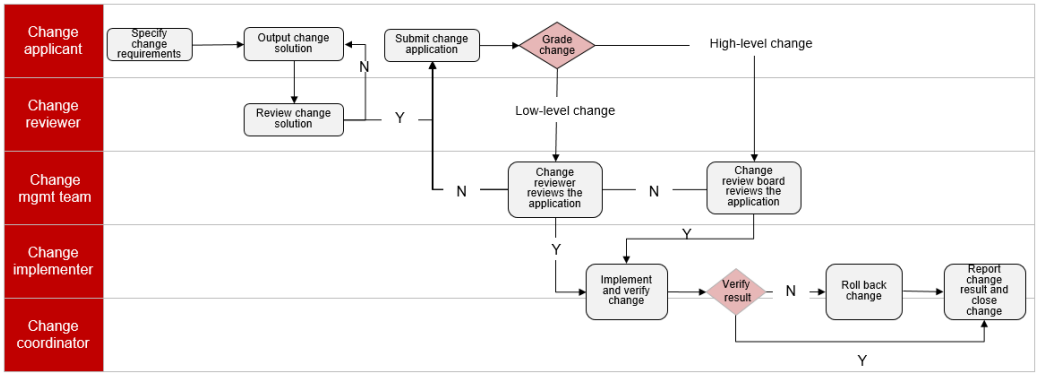
  High

- **Key strategies**

  Before establishing a standard change management process, define each stage clearly.

  - Change initiation: Before initiating a change, specify the change content and reason. Specifying the change information can reduce the workload of the change evaluator and clarify the change significance. Change information includes:
    - Basic information (title, time, changer, and reason).
    - Change information (system, scenario, and type).
    - Change solution (change implementation, rollback, and verification).
  - Change approval: The person in charge of the change system reviews the change to check the change risk level. If the change risk level cannot be controlled or predicted, more details should be provided for the change solution or the change should be prohibited. The change approval process can involve several persons, including the business owner, team TL, and technical TL. Personnel involved in the change can be determined based on factors such as the impact degree and scope of the change.
  - Change execution: The executor confirmed at the change initiation stage assigns the execution tasks to maintain that the change is accurately executed by the technical personnel related to the change content.
  - Change verification: After the change is complete, check the change object and content to maintain that the change does not affect actual business. After the check is complete, release the change result.
  - Change closure: After the change is complete, the change task is closed. Retain the change records to ease subsequent change data operations and analysis.

- **Design suggestions**



6.10 OPS06 Observability System

6.10.1 OPS06-01 Establishing an Observability System

Observability is a concept in system theory, which refers to whether the status of a system can be observed and reproduced by external systems. As cloud native and microservice architecture evolve, IT systems have higher demands for observability. Observability is how much you learn about the internal state or condition of a complex system by understanding its external output. A system that is highly observable allows for quicker and more accurate identification of issue root causes without extra testing or coding.

- **Risk level**

  High

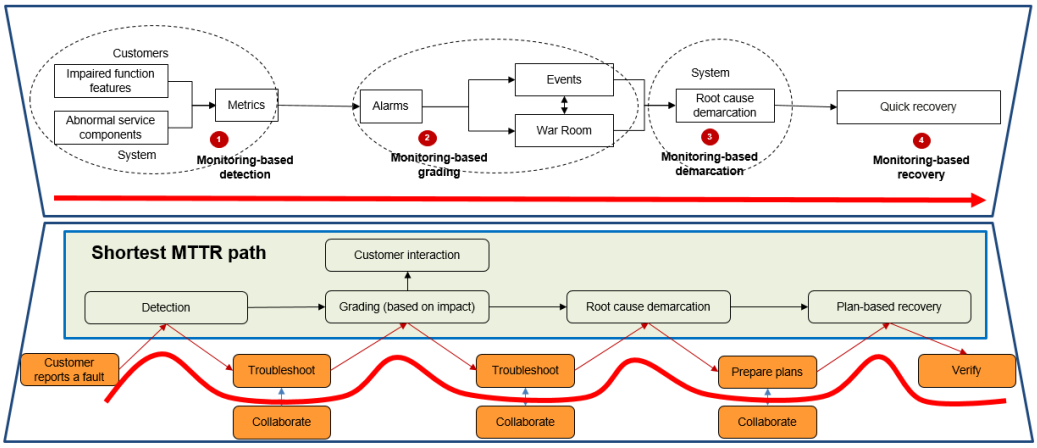
- **Key strategies**

  The observability system is built on the deterministic recovery proposition, determining the construction of deterministic recovery capabilities and SLOs achievement. The observability system plays a key role in the recovery duration of some faults. As shown in the following figure, the average MTTR duration consists of the average discovery duration, average demarcation duration, and average handling duration. The observability system determines the discovery duration and demarcation duration. In an event, the shorter the MTTR duration, the higher the possibility of achieving the overall SLO.

  MTTR = Average discovery duration + Average demarcation duration + Average handling duration

- **Design suggestions**

  The core design logic of the MTTR-oriented observability system is to identify the shortest recovery path. As shown in the following figure, in the fault recovery MTTR logic, when a service fault happens, it is detected, graded, and analyzed before being pinpointed, located, and fixed. These steps are mostly done manually. To shorten the time, monitoring-based detection, grading, and demarcation must be enabled. If this design can be achieved, the shortest path of MTTR can be formed.



 Copyright © Huawei Technologies Co., Ltd.

6.10.2 OPS06-02 Defining Observable Objects

- **Risk level**

  High

- **Key strategies**

  The following table shows how to classify observable objects.

Observability Layer	Function/Major Metric
IT resource monitoring	IT resource monitoring monitors and reports the performance and capacity of IT resources to ensure stable and reliable running of your services.
Application monitoring	Application monitoring tracks resources across different layers (applications, service components, and environments) based on application and resource management. Each layer has its own set of metrics that are monitored. Monitor alarm information at the business, application, middleware, and infrastructure layers, and bind the dashboards to display system charts, metric sources, and log sources in charts. It is important to focus on metrics like available memory, the number of WAITING threads, and the number of TIMED_WAITING threads.
Process monitoring	Process monitoring is used to monitor active processes on a host. By default, information such as CPU usage, memory usage, and the number of opened files of these processes is collected. If you have customized process monitoring, the number of processes containing keywords is also monitored. It is important to focus on metrics like the number of running processes, idle processes, and zombie processes.
Log monitoring	The log configuration service extracts specified keywords from logs. This helps you use the monitoring service to monitor and report alarms for key metrics in logs. It is important to focus on the metrics like log size, the number of access logs, and the number of error logs.
Custom monitoring	The custom monitoring page displays all the metrics defined by yourself. You can call APIs to report collected monitoring data of those metrics to the monitoring service.
Middleware monitoring	The monitoring platform allows you to quickly install and configure middleware plug-ins and offers ready-to-use dashboards for monitoring. Currently, the following middleware plug-ins are supported: MySQL, Redis, MongoDB, Nginx, Node, HAProxy, COMP_EXPORTER, COMP_REDIS_EXPORTER, and COMP_MYSQL_EXPORTER

 Copyright © Huawei Technologies Co., Ltd.

Server monitoring	It provides basic monitoring and OS monitoring of different monitoring granularities. Basic monitoring monitors metrics reported by ECSs. OS monitoring provides proactive, fine-grained OS monitoring for ECSs, and it requires the agent (a plug-in) to be installed on all ECSs to be monitored. It is important to focus on metrics like CPU_UTIL, DISK_READ_BYTES_RATE, and outband incoming rate.
On-premises component monitoring	Unified monitoring of on-premises components. The monitoring platform is interconnected with Grafana and Prometheus to monitor services, applications, and on-premises IDCs and middleware.
Network performance management monitoring	The full-link network that connects clients, networks, edges, and clouds is monitored. This helps you identify network faults quickly and keep track of the network's status. It is important to focus on metrics like application response time, DNS resolution time, TCP connection setup time, and access traffic.

6.10.3 OPS06-03 Developing and Implementing Observability Indicators

- **Risk level**

  High

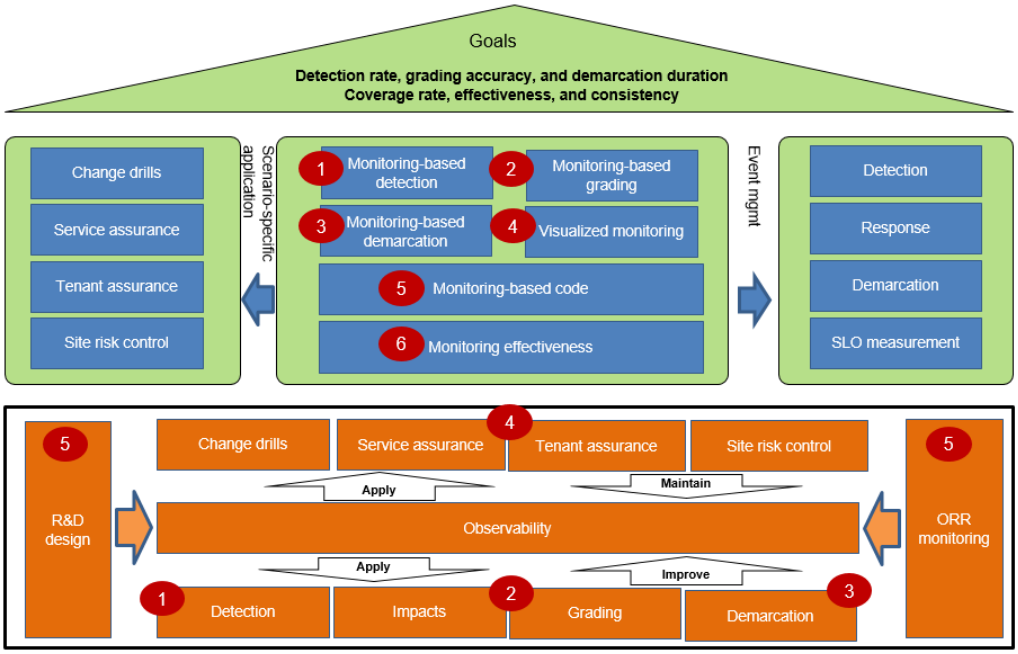
- **Key strategies**

  Metrics measure the data within a period. Observability metrics focus on the discovery rate, grading accuracy, demarcation duration, coverage, validity, and consistency. Observability design specifications, design requirements, and O&M management requirements are released centrally.

- **Design suggestions**

  The overall technical solution will be standardized and released. Architects of each service system should adhere to this standard during design to ensure capabilities can be utilized in design, running, and high availability scenarios.

 Copyright © Huawei Technologies Co., Ltd.



Observability metrics can be implemented using monitoring tools, and alarms can be sent when an exception occurs. Many monitoring tools are available, such as Prometheus, Grafana, Zabbix, and Huawei Cloud Cloud Eye. These tools can periodically collect metrics, provide visualized metric reports, and send alarms to help organizations detect problems in a timely manner.

For details, see **Best Practices of Cloud Eye**.

6.10.4 OPS06-04 Standardizing Application Logs

Logs are events that are timestamped and immutable, happening sequentially over time. The system needs to record key incidents and faults to help diagnose and rectify faults.

- **Risk level**

  High

- **Key strategies**

  Logs are very important for a system. They record all events in the system, including successful operations, failed operations, and warning information. Hence, logging is a fundamental requirement in observability design. Logs can record events and error information in log files or databases. This is useful for troubleshooting and problem diagnosis. However, logging is not enough. Logs must be effectively managed and analyzed. Excessive logs can be burdensome as they consume storage space and make it time-consuming to locate relevant information. So, logs need to be filtered and archived for better management.

- **Design suggestions**

  For details, see **LTS Best Practices**.

6.10.5 OPS06-05 Implementing Dependency Telemetry

- **Risk level**

 Copyright © Huawei Technologies Co., Ltd.

High

- **Key strategies**

  Dependency telemetry monitors the running status and performance of components and external services that workloads depend on. It collects high-value metrics on accessibility, timeout, and other key events related to dependencies such as DNS, databases, or third-party APIs. When you monitor an application to publish dependency-related metrics, logs, and traces, you can better understand potential bottlenecks, performance issues, or failures that could impact workloads.

6.10.6 OPS06-06 Performing Distributed Tracing

A trace indicates a series of causality-related distributed events that encode the end-to-end request flows through a distributed system.

- **Risk level**

  High

- **Key strategies**

  When a fault occurs in the system, the behavior and interaction of each component in the system need to be traced. Distributed traces in the system can quickly locate and rectify the fault.

- **Design suggestions**

  Link traces can be implemented by adding a trace identifier to the system. When the system receives a request, an identifier is added to the request and transferred throughout the system. Each component can add identifiers to their logs to facilitate troubleshooting when an issue occurs. Open-source tools such as Jaeger, Zipkin, SkyWalking, and CAT can be used for distributed traces. Huawei Cloud APM provides the tracing observation capability.

  For details, see **Locating the Causes of Request Errors**.

6.10.7 OPS06-07 Automation Measures Based on Observability Metrics

- **Risk level**

  High

- **Key strategies**

  Observability and automation O&M tools are combined for automated fault detection, recovery, and auto scaling, further improving the O&M response speed and accuracy and reducing delays and errors caused by human intervention.

6.11 OPS07 Fault Analysis and Management

6.11.1 OPS07-01 Creating Alarms

- **Risk level**

  High

 Copyright © Huawei Technologies Co., Ltd.

- **Key strategies**

  Respond to alarms based on the alarm operability principle. You do not need to respond to meaningless alarms, like a sudden increase in disk I/O volume or CPU usage. Following the operability principle can prevent many false alarms. In addition, you need to periodically collect statistics on and analyze alarm frequencies, identify high-frequency alarms, resolve the alarms, and clear the false alarms.

- **Design suggestions**
  - Optimize alarm thresholds: Properly increase the memory, CPU, and network I/O alarm thresholds.
  - Optimize log levels: Optimize improper log levels, for example, change some ERROR logs to WARNING logs.
  - Shield some logs: For applications whose log levels are difficult to adjust, shield some frequent log alarms based on keywords.
  - Enhance warning: Provide warnings for some operations that affect service parties.
  - Enhance emergency warning: Some hardware faults will occur in **/var/log/messages**. Match hardware alarms based on keywords for timely handling.

- **Related cloud services and tools**

  Application Operations Management (AOM)

  Cloud Operations Center (COC)

  Cloud Eye

6.11.2 OPS07-02 Creating Monitoring Dashboards

- **Risk level**

  High

- **Key strategies**

  A monitoring dashboard serves as custom monitoring platforms, allowing you to check core metrics and compare the performance data of different services from different dimensions in a monitoring item.

- **Huawei Cloud services and tools**

  Cloud Eye

  Cloud Operations Center (COC)

6.11.3 OPS07-03 Managing Events

- **Risk level**

  High

- **Key strategies**

  Events are things that need intervention. When events occur, they are handled through a process. How to communicate the activity status with teams? Who is responsible for response and handling? What tools are used to mitigate the events? All these questions need to be answered and a reliable response is required in the process. The process must be centralized and accessible to all involved in the workload. If there is no document storage, you can use the source code version control mechanism.

 Copyright © Huawei Technologies Co., Ltd.

Automated event responses are necessary to save time for service delivery and innovation. First, build a repeatable process and mitigate or solve the fundamental issues to improve efficiency.

- **Huawei Cloud services and tools**

  Cloud Eye

  Cloud Operations Center (COC)

6.11.4 OPS07-04 Supporting Fault Recovery Process

- **Risk level**

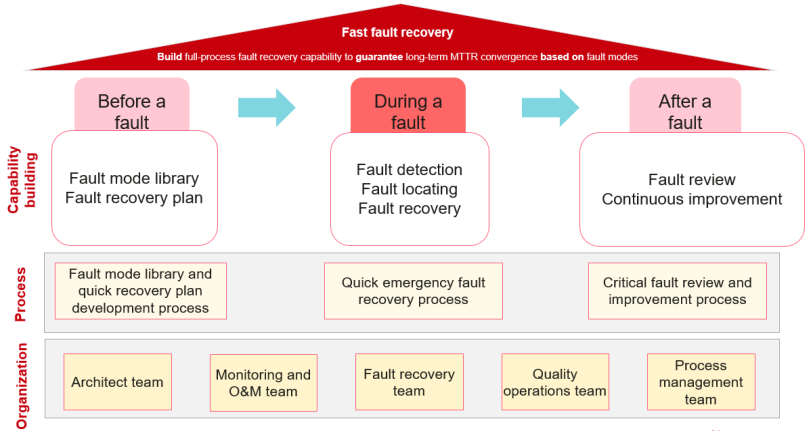
  High

- **Key strategies**

  If a fault occurs on the live network, fix it fast to minimize its impacts. A series of control processes must be performed throughout the fault lifecycle, including fault prevention, detection, locating, recovery, review, and continuous improvement (including fault drills). Recovery capabilities must be built based on the failure mode library to ensure long-term convergence of the mean time to repair for quick fault recovery.

- **Design suggestions**

Figure 6-1 Quick fault recovery process



6.12 OPS08 Operations Status Measurement and Continuous Improvement

6.12.1 OPS08-01 Using Metrics to Measure Operations Objectives

- **Risk level**

  High

- **Key strategies**

  Define clear objectives and key performance indicators (KPI) for successful operations. Set baselines as reference points and re-evaluate them

periodically. Determine the overall objectives of the service with service leaders and stakeholders, and the tasks of each operations team and the challenges they may face. Clarify KPIs of operations goal. The KPIs may include customer satisfaction, time to market (TTM), and average issue resolution duration. Identify key metrics and data sources based on KPIs. Customer satisfaction may be a combination of various metrics, such as call waiting or response duration, satisfaction score, and type of issues raised.

6.12.2 OPS08-02 Reviewing and Improving Events

Event analysis aims to standardize and guide event input and output after major events. It is used to guide the implementation of event backtracking, rectification of issues found in the backtracking report, and promotion of the summarized experience.

- **Risk level**

  High

- **Key strategies**

  To standardize recovery practices and improve availability and technical capabilities, after a critical live network fault was fixed, the recovery process is reviewed, the root causes are analyzed, and improvement measures are proposed. The postmortem process consists of review, analysis, summary, and action (RASA).

  – Review: All key links and roles in the recovery process root cause locating, decision-making, handling, contingency plan execution, rollback, and clearance.

  – Analysis: Root causes for faults and the potential for improvement in fault handling are analyzed.

  – Summary: The fault and the recovery process are summarized, including the fault nature, responsibility evaluation, and experience and lessons learned.

  – Action: Based on the summary, improvement measures are proposed and taken.

6.12.3 OPS08-03 Managing Knowledge

- **Risk level**

  High

- **Key strategies**

  Growing data volumes and complex service systems pose higher requirements on O&M personnel. O&M knowledge management is necessary to help O&M personnel obtain knowledge and learn and solve issues. O&M knowledge management integrates abundant O&M knowledge to help O&M personnel quickly solve issues and improve work efficiency. Generally, the O&M knowledge base system is used to carry O&M knowledge. This allows O&M personnel to easily access a range of information, such as network setup, server administration, and database upkeep. The system offers the following five main features and benefits:

  – Rich knowledge resources: A vast collection of O&M knowledge and expertise is gathered and organized, encompassing diverse topics and

 Copyright © Huawei Technologies Co., Ltd.

levels. Users can search for related knowledge and solutions through the system. In addition to solving some common issues, the system can also provide advanced technical support to help users solve complex issues.

- Quick issue locating: The O&M knowledge base system is equipped with a powerful search function. Users can search for knowledge by keywords. The system automatically matches related knowledge and provides corresponding solutions. Users can quickly locate issues and find solutions through simple operations, saving significant time and energy.
- Knowledge sharing and communication: The O&M knowledge base system also supports knowledge sharing and communication between users. Users can upload their experience and knowledge to the system and communicate and discuss with other users. This method enables users to broaden their knowledge and access additional solutions by communicating with others.
- Update and maintenance at any time: The O&M knowledge base system is regularly updated and maintained to ensure the content is up-to-date and accurate at all times. Users can obtain the latest knowledge and solutions through the subscription function. This approach helps users stay ahead and ready for challenging tasks.
- Various knowledge formats: The O&M knowledge base system supports displaying knowledge in multiple formats, such as text, images, and videos. This allows for meeting various user needs, enabling users to grasp knowledge more intuitively. Users can choose their preferred display formats.

6.13 Examples

6.13.1 Improving System O&M Capabilities to Reduce O&M Costs and Difficulties with AOM

A platform service has 10 million certified drivers and 5 million cargo users. The group's service covers 339 major cities in China and more than 110,000 routes, implementing multi-center operations architecture across the country.

Customer pain points:

- Difficult to ensure O&M in multi-cloud active-active DR scenarios: Many clusters need more than single-vendor DR to keep services running smoothly. A parallel active-active system is required to ensure zero switchover in case of faults. During this process, the customer's self-built O&M platform is insufficient to meet O&M requirements.
- Cloud service metrics cannot be collected: The customer's self-built O&M system cannot collect metrics of cloud services, which cannot meet the requirements of large-screen display.
- Insufficient alarm notification capabilities: The self-built O&M platform lacks sufficient alarm notification capabilities for multiple scenarios and does not support alarm noise reduction.

Solutions:

Benefits:

 Copyright © Huawei Technologies Co., Ltd.

- Reduced O&M costs and difficulties: Simplified management of multiple systems, decreased initial customer O&M resource investments, and lowered O&M costs.
- Improved operations analysis capabilities: Visual charts and out-of-the-box dashboards enable quick service operations analysis.
- Improved troubleshooting capabilities: Cloud-based multi-dimensional monitoring enables comprehensive service operations and maintenance, achieving rapid detection, location, and handling of faults through automated alarm rules.

6.13.2 Gathering Various Device Logs Using LTS for Full-Link Issue Tracking and Service Operations Analysis

A company specializes in IT information transmission, technical exchange, education and training, and professional technical talent services. The company has over 1,000 customers and partners with more than 32 million registered users.

Customer pain points:

- Unified collection tools on the device side, unavailable custom domain names, and complex fault locating.

Logs on multiple device sides, such as web, iOS, Android, Baidu mini program, and WeChat mini program, cannot be collected using tools of the same vendor. This forces users to switch between multiple tools when locating and analyzing issues, making troubleshooting more complex. Additionally, users cannot customize the server domain name for log reporting, raising compliance concerns among some users.

- The device logs report slowly and often get lost: Their reporting speed is tracked hourly. This loss impacts both end-to-end issue locating and service integrity analysis.
- Difficult service mining and analysis: Logs cannot be written to data lake insight (DLI). The data needs to be delivered to Kafka and then consumed by DLI, causing high links and high costs.

Solutions:

Benefits:

- Comprehensive collection and access of device logs and reporting of custom domain names: The multi-device software development kits (SDK) provided by LTS are integrated to comprehensively collect device logs and access LTS. The reporting of domain name of the server helps ensure service consistency and compliance on the user plane, reducing issue locating complexity and improving O&M efficiency.
- Millisecond-level reporting of device log data and zero data loss: This feature ensures millisecond-level reporting of device logs with no data loss. It enables full-link tracing and analysis from the frontend to the backend. In addition, service integrity analysis is supported.
- Easy and low-cost log obtaining, facilitating service mining and analysis: DLI-Flink integrates seamlessly with Connector to fetch logs from LTS in real time, enabling efficient service computing and analysis. Logs can also be stored on

 Copyright © Huawei Technologies Co., Ltd.

OBS for quick access by DLI, supporting offline service computing and analysis.

6.13.3 Performing Daily Service O&M and Compliance with LTS

A private high-tech enterprise focuses on IT, automotive, and renewable energy. It was listed in the 2022 Fortune Global 500 in August 2022.

Customer pain points:

- Managing projects is challenging due to the presence of numerous service departments and a high volume of logs.
- Managing operations and maintenance becomes challenging when there are excessive cloud resources and inexperienced monitoring staff.
- Long-term log storage is required for security and compliance. But it is limited by costly self-built ELK and understaffed O&M departments.

Solutions:

Benefits:

Full log access: LTS covers 170 service systems across automobile apps, software development, and traffic platforms. It covers services, applications, middleware, and infrastructure.

Issue demarcation in minutes: Log query in seconds and monitoring in minutes, configurable alarms and multi-channel notifications, and 90% issue awareness and demarcation within 30 minutes.

Compliant storage duration: Storage duration up to 365 days; lower storage costs with intelligent cold and hot storage; easy access for searches.

6.14 Operational Excellence

6.14.1 CodeArts

CodeArts is a one-stop, secure, and trustworthy DevSecOps platform built on Huawei's years of expertise in empowering digital transformation.

CodeArts consists of the following services:

- CodeArts Req: Provides requirement management and team collaboration services. It has multiple out-of-the-box scenario-based requirement models and object types (such as requirements, bugs, and tasks). It supports multiple R&D modes such as Integrated Product Development (IDP), DevOps, and Lean Kanban, and provides various functions including cross-project collaboration, baseline and change management, custom reports, online collaboration on Wiki, and document management.
- CodeArts Repo: Provides Git-based distributed code management and collaborative development capabilities, including member management, permission control, code hosting, code check, code review, code tracing, and continuous integration. It helps enterprises of different scales improve R&D quality and efficiency.

 Copyright © Huawei Technologies Co., Ltd.

- CodeArts Pipeline: Provides visual, customizable pipelines to shorten the delivery period and improve efficiency.
- CodeArts Check: Provides a variety of checks, such as code style, common quality issues, and cybersecurity risks. It generates detailed quality reports and facilitates closed-loop issue resolution, allowing enterprises to effectively manage code quality.
- CodeArts Build: Provides an easy-to-use platform that supports multi-language parallel builds on the cloud. Its distributed acceleration helps enterprises improve build efficiency while lowering overall costs even at large scale.
- CodeArts Deploy: Supports deployment in hosts and containers by using multiple languages and technology stacks, including Tomcat and Spring Boot. The plug-in encapsulation and orchestration of deployment functions help you quickly and efficiently release software.
- CodeArts TestPlan: Covers the entire process of test plan, test design, test cases, test execution, and test evaluation, and aims to help enterprises with collaborative, efficient, and trustworthy test activities before product release.
- CodeArts Artifact: Manages source code build products, including Maven and npm. It can seamlessly interconnect with local build tools and on-cloud CI/CD so that you can manage software package lifecycle to improve release quality and efficiency. It also provides features such as artifact package version control, granular permission management, security scanning, and more.
- CodeArts IDE Online: A cloud computing–based lightweight web IDE that enables users to quickly obtain and access the environment using a browser to complete coding, build, debugging, running, repo access, and command execution. It supports third-party service integration and extensions, and provides an independent extension market.

6.14.2 RFS

Resource Formation Service (RFS) is a new final-state resource formation engine for cloud services and fully supports Terraform (HCL and Provider). It is an upgrade of the Application Orchestration Service (AOS). Based on HCL syntax template, RFS automatically builds cloud service resources in batches to help you create, manage, and upgrade cloud service resources in an efficient, secure, and consistent manner.

6.14.3 COC

Cloud Operations Center (COC) is a secure and efficient O&M platform, offering one-stop, AI-powered solutions for all your centralized O&M needs. It encompasses Huawei Cloud deterministic operations scenarios and features essential functionalities such as fault management, batch O&M, and chaos drills, to improve cloud O&M efficiency while ensuring security compliance.

Unified resource management

- Application management: provides the capability of modeling the association between applications and resources to fulfill your requirements in centralized cloud resource management and cost reduction management.

 Copyright © Huawei Technologies Co., Ltd.

- Resource management: synchronizes and manages the resource instances used on various cloud platforms to build a resource O&M capability foundation.
- Configuration management: manages applications and resources, and centrally monitors their parameter configurations throughout their lifecycles.
- Compliance management: provides batch patch scanning and repair capabilities for resource O&M, ensuring both security compliance and efficiency.

All-round change management

- Solution review: enables Standard Operating Procedure (SOP) for change solutions, clarifying and electronizing change solutions and archiving them after review. Rules and processes can be decoupled to ensure that a change execution process is correct and that the change solution can be accumulated.
- Change review: reviews change tickets according to the preset review process to ensure the reliability, efficiency, and process compliance of change solutions.
- Risk assessment: manages changes based on scenario rules, process rules, and business rules to identify and prevent change risks in advance. The change calendar is used to identify change conflicts and reduce change risks caused by change dependencies between services.
- Assurance implementation: presets changes solutions, standardizes change steps, enables change operation observation, and ensures timely handling of change exceptions, delivering controllable, visible, and manageable change processes.

Deterministic fault management

- Unified incident center: provides an E2E and standard incident handling mechanism, covering incident discovery, incident handling, recovery verification, and continuous improvement.
- WarRoom and fault backtracking capabilities: triggers WarRoom requests intelligently for live-network incidents, shortening troubleshooting time. In addition, you can observe the troubleshooting progress in real time from the command center. Fault backtracking facilitates issue summary and experience accumulation, preventing issues from recurring and shortening the MTTR.
- Contingency plans: enables you to develop contingency plans for known faults and handle deterministic issues using the contingency plan automation mechanism.
- Failure modes: leverages professional risk analysis methods and expert knowledge bases to accumulate a failure mode base, helping you analyze potential risks of cloud applications and pass on O&M experience.

Resilience center optimization

- Full-lifecycle risk management: encompasses risk management in both application deployment and running scenarios throughout the lifecycles of applications and resources, serving you based on years of dynamic risk management experience accumulated on Huawei Cloud.
- Proactive O&M: promotes the quality and resilience of your key services through proactive O&M methods, including performance pressure tests, emergency drills/chaotic engineering, and resilience evaluation.

 Copyright © Huawei Technologies Co., Ltd.

- Rich fault drill tools: uses over 50 built-in drill attack tools based on Huawei Cloud best practices, enabling you to simulate complex and diversified service exception scenarios and develop countermeasures.
- Cost-effective and efficient capabilities: provides you with professional optimization suggestions without affecting your cloud services, helping you minimize costs, maximize efficiency, and optimize services.

6.14.4 Cloud Eye

Cloud Eye monitors cloud services and resources to keep you informed of any issues in real time.

Cloud Eye provides the following functions:

- Automatic monitoring: Monitoring starts automatically after you created resources such as Elastic Cloud Servers (ECSs). On the Cloud Eye console, you can view the service status and set alarm rules for these resources.
- Server monitoring: After you install the Agent (Telescope) on an ECS and Bare Metal Server (BMS), you can collect 60-second granularity ECS and BMS monitoring data in real-time. Cloud Eye tracks 40 different metrics, such as CPU, memory, and disk usage, for resources.

  For details, see **Server Monitoring**.
- Flexible alarm rule configuration: You can create alarm rules for multiple resources at the same time. After an alarm rule is created, you can flexibly manage it, for example at any time you can modify, enable, disable, or delete it.
- Real-time notification: You can enable **Alarm Notification** when creating alarm rules. When the cloud service status changes and the monitoring data of the metric reaches the threshold specified in an alarm rule, Cloud Eye notifies users by text messages, emails, or by sending messages to server addresses. In this way, you can monitor the cloud resource status and changes in real time.
- Panel: Panels enable you to view cross-service and cross-dimension monitoring data. It displays key metrics centrally, providing an overview of the service operating status and allowing monitoring details to be checked when troubleshooting.
- OBS dump: Raw data for each metric is kept for only two days on Cloud Eye. If you need to store the metrics longer than that, use Object Storage Service (OBS) to synchronize them to OBS buckets.
- Resource groups: A resource group allows you to add and monitor correlated resources and provides a collective health status for all resources that it contains.
- Website monitoring: It continuously monitors the remote server status, such as availability and connectivity, by simulating real user's access to remote servers.
- Log Monitoring: It enables you to monitor log content in real time. By creating alarm rules on Cloud Eye to monitor the logs collected by Log Tank Service (LTS), you can reduce your O&M costs for log monitoring and simplify the log monitoring process.

 Copyright © Huawei Technologies Co., Ltd.

- Event monitoring: You can query system events that are automatically reported to Cloud Eye and custom events reported to Cloud Eye through the API. When there are specified events, you will receive alarm notifications.

6.14.5 LTS

Log Tank Service (LTS) is a high-performance, low-cost, and reliable log platform. It offers various functions, such as full-stack log collection, search of tens of billions of logs in seconds, PBs of storage, log processing, visual charts, alarms, and log transfer. It is designed for application O&M, security compliance, and operations analysis.

LTS provides multiple modes to ingest massive logs to LTS. It is integrated with log search, SQL analysis, and log processing engines.

- Log ingestion for all device-cloud scenarios: covers 40+ cloud services, servers/containers, mobile devices, clouds, SDKs of different languages, and accounts.
- Log storage and search: takes just seconds for tens of billions of logs, and supports iterative search of hundreds of billions of logs and PBs of intelligent cold storage.
- SQL statistics display: provides over 100 SQL functions, multiple visual charts, and over 10 out-of-the-box dashboards.
- Real-time alarm: supports custom content and multiple notification channels (SMS, email, WeChat, DingTalk, and HTTP).
- One-stop processing: provides over 200 functions and log normalization, enrichment, anonymization, filtering, and splitting.
- Service integration: transfers logs to Object Storage Service (OBS)/Data Warehouse Service (DWS)/Data Ingestion Service (DIS)/Data Lake Insight (DLI)/Distributed Message Service (DMS) for building horizontal solutions.

6.14.6 AOM 2.0

Application Operations Management (AOM) is a one-stop, multi-dimensional O&M management platform for cloud applications. It integrates observable data sources, such as Cloud Eye, LTS, Application Performance Management (APM), real user experience, and backend link data. It also provides unified application resource management, one-stop observability analysis, and automated O&M solutions. With AOM, you can quickly detect faults, monitor applications, resources, and services in real time, and improve automated O&M capability and efficiency.

6.14.7 APM

Huawei Cloud Application Performance Management (APM) helps O&M personnel quickly identify application performance bottlenecks and locate root causes of faults, ensuring user experience.

You only need to install Agents for applications so that APM can monitor them in an all-round manner. APM can quickly locate error or slow APIs, reproduce call parameters, and detect system bottlenecks, facilitating online diagnosis. APM supports Java, Python, Node.js, Go, PHP, and .NET applications. The following table lists the application monitoring capabilities of APM.

 Copyright © Huawei Technologies Co., Ltd.

6.14.8 CBH

Cloud Bastion Host (CBH) is a unified security management and control platform. It provides account, authorization, authentication, and audit management services that enable you to centrally manage cloud computing resources.

CBH provides various functional modules, such as department, user, resource, policy, operation, and audit modules. It integrates functions such as single sign-on (SSO), unified asset management, multi-terminal access protocols, file transfer, and session collaboration. With the unified O&M login portal, protocol-based forward proxy, and remote access isolation technologies, CBH enables centralized, simplified, secure management and maintenance auditing for cloud resources such as servers, cloud hosts, databases, and application systems.

6.14.9 ServiceStage

ServiceStage is an application management and O&M platform that lets you deploy, roll out, monitor, and maintain applications all in one place. It supports technology stacks such as Java, PHP, Python, Node.js, Docker, and Tomcat, and supports microservice applications such as Apache ServiceComb Java Chassis (Java chassis) and Spring Cloud, making it easier to migrate enterprise applications to the cloud.

ServiceStage provides the following capabilities:

1. Application management: application lifecycle management and environment management.
2. Microservice application access: Java chassis, Go chassis, Spring Cloud and Dubbo microservice frameworks and Mesher.

   It can work with the microservice engine to implement service registry and discovery, configuration management, and service governance. For details, see the **Microservice Development Guide**.
3. Application O&M through logs, monitoring, and alarms

6.14.10 MAS

Chaos engineering of **Multi-Site High Availability Service (MAS)** is a practice of identifying and repairing potential risks by actively injecting faults. MAS-CAST offers different attack options for users to tailor the attack targets and policies, along with incorporating background traffic and resource monitoring. In addition to fault injection, systematic procedures and specifications allow users to conduct fault drills to assess the system's reliability and emergency response.

 Copyright © Huawei Technologies Co., Ltd.