FunctionGraph

Service Overview

Issue 01

Date 2024-11-11

HUAWEI

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


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

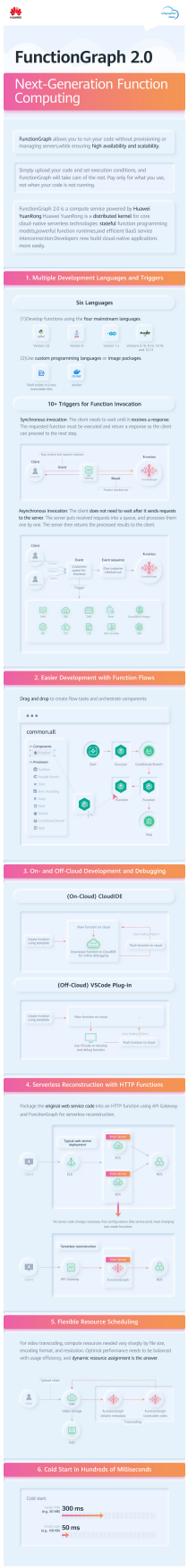
Notice

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

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

Contents

1 FunctionGraph Infographics

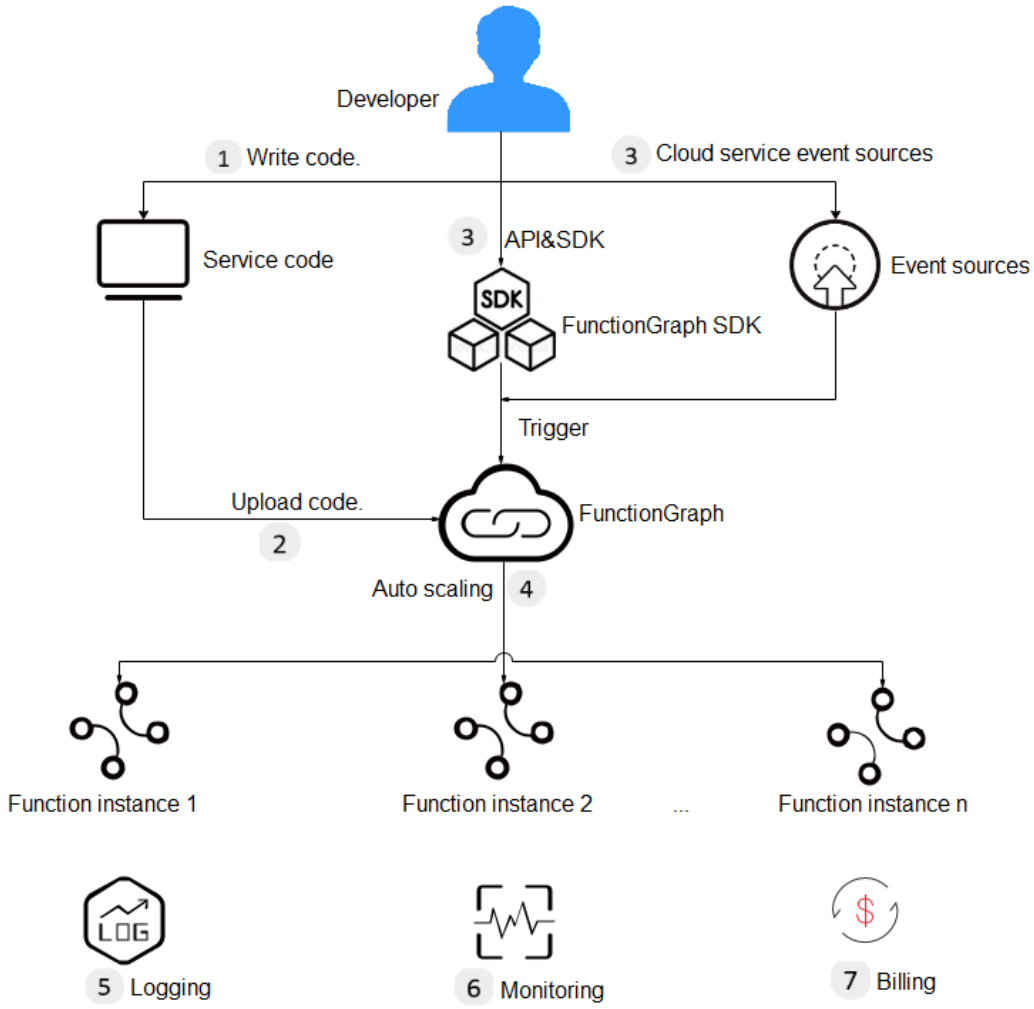


2 What Is FunctionGraph?

FunctionGraph hosts and computes event-driven functions in a serverless context while ensuring high availability, high scalability, and zero maintenance. All you need to do is write your code and set conditions. You pay only for what you use and you are not charged when your code is not running.

Figure 2-1 shows the process of using FunctionGraph.

Figure 2-1 Usage process



Feature Overview

1. Write code.

Write code in Node.js, Python, Java, C#, PHP, or Go. For details, see the **FunctionGraph Developer Guide**.

2. Upload code.

Currently, you can edit code inline, upload a ZIP or JAR file, or obtain a ZIP file from OBS. For details, see **Table 3-2**.

3. Trigger functions by API calls or cloud service events.

Call RESTful APIs or use cloud service event sources to trigger function execution and generate instances to implement service functions.

4. Auto scaling is implemented.

During function execution, FunctionGraph scales automatically based on the number of requests without the need for configurations. For details about the maximum number of function instances that can be run concurrently, see **Notes and Constraints**.

5. View logs.

View run logs of functions as FunctionGraph is interconnected with Log Tank Service (LTS). For details, see **Querying Function Logs**.

6. View monitoring information.

View graphical monitoring information. FunctionGraph is interconnected with Application Operations Management (AOM).

7. Billing mode

After a function is executed, you will be billed based on the number of function execution requests and execution duration.

3 Product Features

Function Management

FunctionGraph provides console-based function management.

- The Node.js, Java, Python, Go, C#, PHP, and custom runtimes are supported. **Table 3-1** provides the details.

  NOTE

  You are advised to use the latest runtime version.

  Table 3-1 Runtimes

Runtime	Supported Version
Node.js	6.10, 8.10, 10.16, 12.13, 14.18, 16.17, 18.15
Python	2.7, 3.6, 3.9, 3.10
Java	8.0 and 11
Go	1.x
C#	.NET Core 2.1 and .NET Core 3.1
PHP	7.3
Custom	-
Cangjie	1.0

- Multiple code entry modes

  FunctionGraph allows you to edit code inline, upload a ZIP file from Object Storage Service (OBS), or directly upload a ZIP or JAR file. **Table 3-2** lists the code entry modes supported for each runtime.

Table 3-2 Code entry modes

Runtime	Editing Code Inline	Uploading a ZIP File	Uploading a JAR File	Uploading a ZIP File from OBS
Node.js	Supported	Supported	Not supported	Supported
Python	Supported	Supported	Not supported	Supported
Java	Not supported	Supported	Supported	Supported
Go	Not supported	Supported	Not supported	Supported
C#	Not supported	Supported	Not supported	Supported
PHP	Supported	Supported	Not supported	Supported
Custom	Supported	Supported	Not supported	Supported
Cangjie	Not supported	Supported	Not supported	Supported

Trigger

Table 3-3 lists the invocation modes for different trigger types.

Table 3-3 Function invocation modes

Trigger	Function Invocation Mode
SMN trigger	Asynchronous invocation
APIG trigger	Synchronous invocation
Data Ingestion Service (DIS) trigger	Asynchronous invocation
Timer trigger	Asynchronous invocation
Log Tank Service (LTS) trigger	Asynchronous invocation
Cloud Trace Service (CTS) trigger	Asynchronous invocation
Document Database Service (DDS) trigger	Asynchronous invocation
Kafka trigger	Asynchronous invocation

Logs and Metrics

FunctionGraph graphically displays function monitoring metrics and collects function running logs, enabling you to view function statuses, and locate problems by querying logs.

To query logs, see **Managing Function Logs**.

For details about monitoring metric, see **Function Monitoring**.

For details about tenant-level function monitoring metrics, see **Introduction to Dashboard**.

Function Initialization

The initializer interface is introduced to:

- Isolate function initialization and request processing to enable clearer program logic and better structured and higher-performance code.
- Ensure smooth function upgrade to prevent performance loss during the application layer's cold start initialization. Enable new function instances to automatically execute initialization logic before processing requests.
- Identify the overhead of application layer initialization, and accurately determine the time for resource scaling and the quantity of required resources. This feature makes request latency more stable when the application load increases and more function instances are required.

Function Flows

Function flows are used to orchestrate functions. Multiple functions can be orchestrated into a flow for coordinating the execution of multiple distributed function tasks.

On the orchestration page, you can connect event triggers, functions, and flow controllers in a flowchart through lines. The output of each node is used as the input of the next node. An orchestrated flow will be executed according to the sequence specified in the flowchart. After it is successfully executed, you can view its execution records for easy diagnosis and debugging.

Function flows have the following features and advantages:

- Features
  a. Visualized function orchestration
  b. Function flow execution engine
  c. Error handling
  d. Visualized monitoring
- Advantages
  a. Build apps with less code

     Function flows allow you to orchestrate functions into a complete application without compiling code. Quick construction and rollout When services are changed, you can quickly adjust flows and roll out services without writing any code.

b. Comprehensive error handling

   Errors that occur in processes can be captured, retries are supported, and exceptions can be flexibly handled.

c. Visualized orchestration and monitoring

   Flows can simply be orchestrated by dragging elements.

   You can view flows on the monitoring page to quickly locate problems.

Unified Plug-in for Development and Debugging

- **VSCode plug-in (off-cloud):**

  Create a function using a template, view the function on the cloud, download the function to a local PC for debugging, use the VSCode plug-in to debug the function, and then push the function to the cloud.

HTTP Functions

You can set **Function Type** to **HTTP Function** on the function creation page. HTTP functions are designed to optimize web services. You can send HTTP requests to URLs to trigger function execution. HTTP functions support APIG and API Connect (APIC) triggers only.

NOTE

This feature is supported only by FunctionGraph v2.

Tracing

You can enable tracing for functions. Then you can go to the Application Performance Management (APM) console to view JVM and tracing information. Currently, only Java functions can be traced.

Custom Images

You can directly package and upload container images. The images are loaded and started by the platform and can be called in a similar way as HTTP functions. Unlike the previous code upload mode, you can use a custom code package, which is flexible and reduces migration costs.

NOTE

This feature is supported only by FunctionGraph v2.

4 Product Advantages

No Servers to Manage

FunctionGraph automatically runs your code and frees you from provisioning and managing servers, allowing you to focus on business innovation.

Auto Scaling

FunctionGraph automatically scales to suit fluctuations in resource demands and ensures that the service remains accessible even during peaks and spikes.

It automatically scales in/out resources based on the number of service requests, and distributes requests to function instances through automatic load balancing.

In addition, the system intelligently preheats instances for the traffic loads to reduce the impact of cold start on your services.

Event-based Triggering

FunctionGraph integrates with multiple cloud services using an event-based triggering mechanism to meet service requirements.

It is interconnected with the LTS and Cloud Eye services, allowing you to view function logs and metrics without the need for any configurations.

High Availability

If an instance becomes faulty, FunctionGraph starts another instance to process new requests and releases resources from the unhealthy instance.

Pay per Use

You will be billed based on the number of function requests and execution duration and will not be charged when your code is not running.

Reserved Instance Billing

Reserved instances can be created to initialize functions to eliminate the influence of cold start on your services. Reserved instances are always alive in the execution environment.

For the use of reserved instances, you will be billed based on the number of requests and the running duration of reserved instances. The minimum running duration is 60s.

Dynamic Resource Adjustment

Resource specifications can be dynamically adjusted to minimize resource usage and reduce costs.

5 Application Scenarios

FunctionGraph is suitable for various scenarios, such as real-time file processing, real-time data stream processing, web & mobile application backends, and AI application.

Scenario 1: Event-Driven Applications

Services are executed in event-driven mode and resources are provisioned based on demands. Developers do not need to be concerned about service peaks or troughs. Idle resources are not billed, reducing O&M costs. Event-driven applications include live streaming/transcoding, real-time data stream processing, and IoT rule/event processing.

- **Real-time file processing**

  When files are uploaded from a client to OBS, functions can be triggered to create image thumbnails in real time, convert video formats, aggregate and filter data files, or implement other file operations.

  Advantages:

  - FunctionGraph automatically allocates resources to run more function instances as the number of received requests increases.
  - Files are uploaded to OBS to trigger file processing functions.
  - You will be billed only for resources used to process files as needed (you are not billed for idle resources during lows in demand).

- **Real-time data stream processing**

  FunctionGraph works with DIS to process data streams in real time. FunctionGraph supports application activity tracking, sequential transaction processing, data stream analysis, data sorting, metric generation, log filtering, indexing, social media analysis, and IoT device data telemetry and metering.

  Advantages:

  - Data is collected by means of DIS streams to trigger data processing functions.
  - FunctionGraph automatically allocates resources to run more function instances as the number of received requests increases.
  - You will be billed only for resources used to process files as needed (you are not billed for idle resources during lows in demand).

Scenario 2: Web Applications

Interconnect FunctionGraph with other cloud services or your VMs to quickly build highly available and scalable web & mobile backends. Web applications include mini programs, web pages/apps, chatbots, and Backends for Frontends (BFF).

Advantages:

- FunctionGraph ensures high reliability of website data using OBS and CloudTable, and high-availability of website logic using API Gateway.
- FunctionGraph automatically allocates resources to run more function instances as the number of received requests increases.
- You will be billed only for resources used to process files as needed (you are not billed for idle resources during lows in demand).

Scenario 3: AI Applications

Intelligence evolution requires various services to be integrated for quick rollout. These services include third-party service integration, AI inference, and license plate recognition.

Advantages:

- FunctionGraph works with EI services for text recognition and content moderation to suit a wide range of scenarios – make adjustments whenever you need as demands change.
- You only need to apply for related services and write service code without having to provision or manage servers.
- You will be billed only for function execution and used EI services without having to pay for idle resources when service demands are low.

6 Function Types

6.1 Event Functions

NOTE

You need to choose between **Event Function** and **HTTP Function** when creating a function on FunctionGraph v2.

Overview

FunctionGraph supports event functions. An event can trigger function execution. Generally, it is in JSON format. You can create an event to trigger your function through the cloud service platform or CodeArts IDE Online. All types of triggers supported by FunctionGraph can trigger event functions.

NOTE

1. On the function creation page, **Function Type** is set to **Event Function** by default.
2. During testing, a function can be triggered by simply entering the specified event in JSON format.
3. You can also use triggers to trigger event functions.

Advantages

- Easy single-node programming

  You can edit event functions on FunctionGraph or CodeArts IDE Online or upload code packages there and deploy them with just a few clicks. There is no need for you to care about function concurrency or fault rectification.

- High-performance, high-speed runtimes

  Event functions can be started, scaled, and called within milliseconds. Faults can be detected and rectified within seconds.

- Complete tool chain

  FunctionGraph provides comprehensive logging, tracing, debugging, and monitoring, allowing developers to roll out functions in just three steps.

Restrictions

Event functions face event source restrictions. You need to comply with the function development rules of the function platform.

6.2 HTTP Functions

NOTE

This feature is supported only by FunctionGraph v2.

Overview

FunctionGraph supports event functions and HTTP functions. HTTP functions are designed to optimize web services. You can send HTTP requests to URLs to trigger function execution. HTTP functions support APIG and APIC triggers only.

NOTE

1. HTTP functions support the HTTP/1.1 protocol.
2. On the function creation page, **HTTP Function** is newly added.
3. The HTTP function must be set to **bootstrap**. You can directly write the startup command and **allow access over port 8000**.

Advantages

- Support for multiple frameworks

  You can use common web frameworks, such as Node.js Express and Koa, to write web functions, and migrate your local web framework services to the cloud with least modifications.
- Fewer request processing steps

  Functions can directly receive and process HTTP requests, eliminating the need for API Gateway to convert the JSON format. This accelerates request processing and improves web service performance.
- Premium writing experience

  Writing HTTP functions is similar to writing native web services. You can also use native Node.js APIs to enjoy local development-like experience.

Restrictions

- HTTP functions support APIG (dedicated), APIG (shared), and APIC triggers only.
- Multiple API triggers can be bound to the same function, but all the APIs must belong to the same APIG service.
- For HTTP functions, the size of the HTTP response body cannot exceed 6 MB.
- HTTP functions cannot be executed for a long time, invoked asynchronously, or retried.

7 Notes and Constraints

Account Resource Constraints

The following table provides the quotas for account resources. For details on how to query and modify quotas, see **Quotas**.

Table 7-1 Account resource constraints

Resource	Limit	Adjustable
Maximum number of functions that can be created under an account	400	No.
Maximum number of versions allowed for a function	20	No.
Maximum number of aliases allowed for a function	10	No.
Maximum number of triggers (DDS/ Kafka/Timer) allowed for a function version	10	No.
Size of a code deployment package (in ZIP or JAR format) that can be uploaded to the FunctionGraph console	40 MB	No.
Size of a code deployment package (in ZIP or JAR format) that can be edited inline during function API invocation	50 MB	No.
Size of an original code deployment package allowed during function API invocation	• ZIP: 1500 MB (after decompression) • OBS bucket: 300 MB (after compression)	No.
Maximum size of deployment packages allowed for an account	10 GB	No.

Resource	Limit	Adjustable
Number of concurrent executions per account	100	Yes
Maximum number of reserved instances that an account can create	90 (Number of concurrent executions per account x 90%)	Yes
Size of all environment variables of a function	4096 characters	No.
Maximum size of code that can be displayed on the console	20 MB	No.

Function Running Resource Constraints

Table 7-2 Function running resource constraints

Resource	Default	Adjustable
Ephemeral disk space (**/tmp** space)	512 MB	No.
Number of file descriptors	1024	No.
Total number of processes and threads	1024	No.
Maximum execution duration per request	259,200s	Yes
Valid payload size of invocation request body (synchronous invocation)	6 MB	No.
Valid payload size of invocation response body (synchronous invocation)	6 MB	No.
Valid payload size of invocation request body (asynchronous invocation)	256 KB	No.
Size of imported resources	≤ 50 MB ZIP file	No.
Image size per function	10 GB	No.
Size of exported resources	≤ 50 MB	No.
Instances per tenant	1000	Yes
Max. memory per function	10 GB	No.
Bandwidth	Unlimited	-

Resource	Default	Adjustable
Single log size	Unlimited	-
Maximum execution duration of initializer	259,200s	Yes

NOTE

- Valid payload size of invocation response body (synchronous invocation): The returned character string or the JSON character string of the serialized response body is less than or equal to 6 MB by default. The actual data size varies depending on the backend settings of FunctionGraph. The backend determines the size of the serialized data with a byte-level deviation. The actual valid payload size is 6 MB ± 100 bytes.
- You are not advised to invoke a function whose execution time exceeds 90s on the FunctionGraph console. To invoke such a function, use asynchronous invocation.
- The valid payload size of a request body is 6 MB when a Kafka or DDS trigger is used and is 4 MB when an APIG trigger is used.

8 Security

8.1 Shared Responsibilities

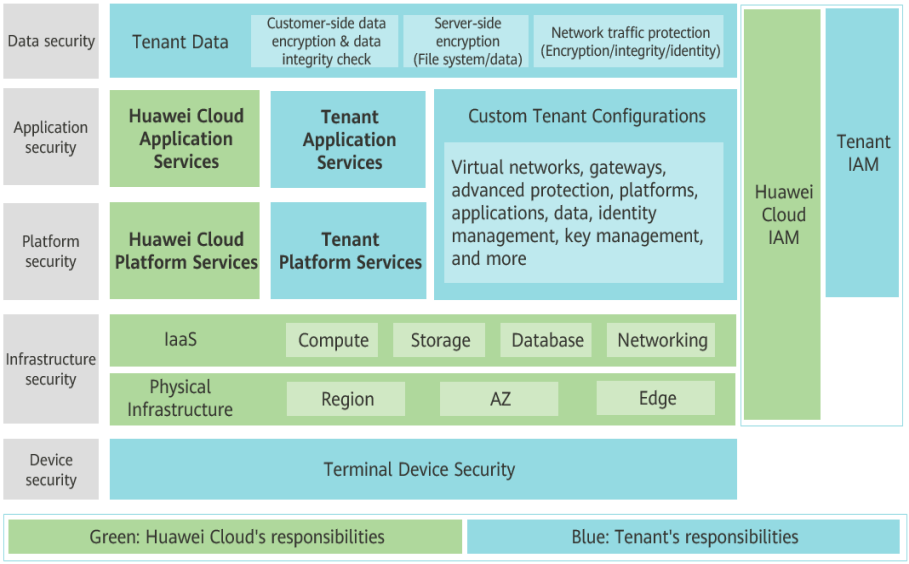
Huawei guarantees that its commitment to cyber security will never be outweighed by the consideration of commercial interests. To cope with emerging cloud security challenges and pervasive cloud security threats and attacks, Huawei Cloud builds a comprehensive cloud service security assurance system for different regions and industries based on Huawei's unique software and hardware advantages, laws, regulations, industry standards, and security ecosystem.

Figure 8-1 illustrates the responsibilities shared by Huawei Cloud and users.

- **Huawei Cloud**: Ensure the security of cloud services and provide secure clouds. Huawei Cloud's security responsibilities include ensuring the security of our IaaS, PaaS, and SaaS services, as well as the physical environments of the Huawei Cloud data centers where our IaaS, PaaS, and SaaS services operate. Huawei Cloud is responsible for not only the security functions and performance of our infrastructure, cloud services, and technologies, but also for the overall cloud O&M security and, in the broader sense, the security and compliance of our infrastructure and services.
- **Tenant**: Use the cloud securely. Tenants of Huawei Cloud are responsible for the secure and effective management of the tenant-customized configurations of cloud services including IaaS, PaaS, and SaaS. This includes but is not limited to virtual networks, the OS of virtual machine hosts and guests, virtual firewalls, API Gateway, advanced security services, all types of cloud services, tenant data, identity accounts, and key management.

elaborates on the ideas and measures for building Huawei Cloud security, including cloud security strategies, the shared responsibility model, compliance and privacy, security organizations and personnel, infrastructure security, tenant service and security, engineering security, O&M security, and ecosystem security.

Figure 8-1 Huawei Cloud shared security responsibility model



8.2 Asset Identification and Management

Environment variables involving sensitive information, such as an account, password, AK, and SK, should be **encrypted**. Unencrypted environment variables will be displayed in plaintext on the GUI or in API responses.

To use triggers, configure VPC access, use custom images, and mount SFS file systems for your function, create an agency for the function so that FunctionGraph can perform resource O&M on your behalf. For details, see **Configuring Agency Permissions**.

8.3 Identity Authentication and Access Control

Identity Authentication

You can access FunctionGraph through the console, APIs, or SDKs. All these access modes are implemented by sending requests via the REST APIs provided by FunctionGraph. FunctionGraph supports **token and AK/SK authentication**.

Access Control

Access to FunctionGraph is controlled through fine-grained permissions management in IAM, which enables you to secure access to your public cloud resources. For details, see **Permissions Management**.

8.4 Data Protection

To prevent your data, such as function metadata, from being obtained by unauthorized or unauthenticated entities or individuals, FunctionGraph encrypts the data for transmission. All API requests and internal communications are encrypted using TLS 1.2 or later.

8.5 Audit and Logs

Audit

Cloud Trace Service (CTS) records operations on the cloud resources in your account. You can use the logs generated by CTS to perform security analysis, trace resource changes, audit compliance, and locate faults.

After you enable CTS and configure a tracker, CTS records management traces of FunctionGraph for auditing.

For details about how to enable and configure CTS, see **Enabling CTS**.

With CTS, you can record operations associated with FunctionGraph for later query, audit, and backtracking. For details, see **Operations Logged by CTS**.

Logs

FunctionGraph is interconnected with Log Tank Service (LTS). For details, see **Managing Function Logs**.

8.6 Resilience

Huawei Cloud data centers are deployed around the world in accordance with relevant rules, and all of them are running properly. These data centers serve as a disaster recovery center for each other. If one of them is down, the involved customer applications and data are transferred to another one in compliance with the regulations to ensure service continuity. In order to minimize the service interruptions caused by hardware failures, natural disasters, or other disastrous events, Huawei Cloud provides a DR plan for all data centers.

FunctionGraph resources are deployed in multiple zones for higher availability, fault tolerance, and scalability.

8.7 Security Risk Monitoring

FunctionGraph uses Cloud Eye to help you monitor your functions and receive alarms and notifications in real time. The following metrics can be monitored: invocations, errors, duration (maximum, average, and minimum), throttles, and instance statistics.

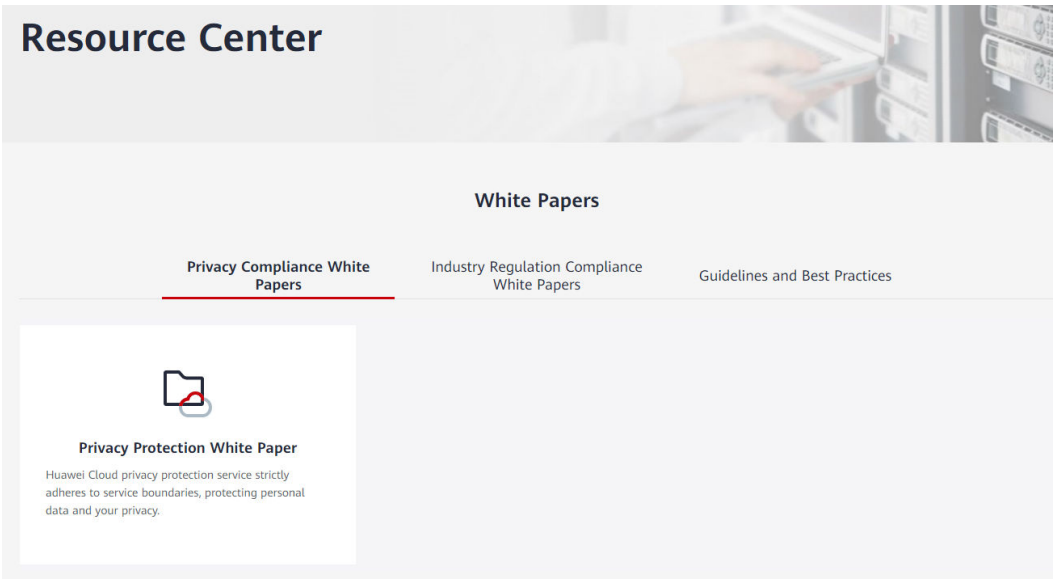
For details about FunctionGraph metrics, see **Monitoring**. To learn how to create alarm rules, see **Creating an Alarm Rule**.

8.8 Certificates

Resource Center

Huawei Cloud also provides the following resources to help users meet compliance requirements. For details, see **Resource Center**.

Figure 8-2 Resource center



8.9 Code Signature

When a function is created or modified, FunctionGraph encrypts the function code and generates a signature to prevent inconsistency caused by code file damage or tampering. The signature is stored in the function metainformation.

When executing a function, FunctionGraph generates a signature for the current code and compares it with the signature in the metainformation. Only code that passes the consistency check is executed. If the check fails, FunctionGraph will not execute the code but return an error.

9 Permissions Management

To assign different permissions to employees in your enterprise to access your FunctionGraph resources, IAM is a good choice for fine-grained permissions management. IAM provides identity authentication, permissions management, and access control, helping you secure access to your cloud resources.

With IAM, you can use your account to create IAM users for your employees, and assign permissions to the users to control their access to specific resource types. For example, some software developers in your enterprise need to use FunctionGraph resources but must not delete them or perform any high-risk operations. To achieve this result, you can create IAM users for the software developers and grant them only the permissions required for using FunctionGraph resources.

If your account does not need individual IAM users for permissions management, you may skip over this chapter.

IAM can be used free of charge. You pay only for the resources in your account. For more information about IAM, see **IAM Service Overview**.

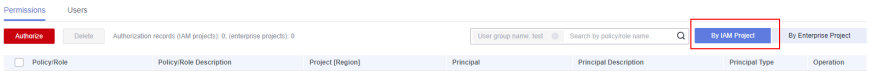
Why Is "Insufficient Permission" Displayed After Enterprise Project Authorization?

IAM project/Enterprise project: A custom policy can be applied to IAM projects or enterprise projects or both. Policies that contain actions supporting both IAM and enterprise projects can be assigned to user groups and take effect in both IAM and Enterprise Management. Policies that only contain actions supporting IAM projects can be assigned to user groups and only take effect for IAM. Such policies will not take effect if they are assigned to user groups in Enterprise Management. For details, see **Differences Between IAM Projects and Enterprise Projects**.

In FunctionGraph, only function resource APIs support enterprise project authorization. For other APIs that support only IAM project authorization:

1. Click **By IAM Project** during authorization.

Figure 9-1 Viewing authorization records by IAM project



2. When selecting the authorization scope, select **Region-specific projects** according to the minimum authorization principle.

FunctionGraph Permissions

By default, new IAM users do not have any permissions assigned. You need to add a user to one or more groups, and assign permissions policies to these groups. The user then inherits permissions from the groups it is a member of. This process is called authorization. After authorization, the user can perform specified operations on FunctionGraph based on the permissions.

FunctionGraph is a project-level service deployed and accessed in specific physical regions. To assign FunctionGraph permissions to a user group, specify the scope as region-specific projects and select projects (such as **cn-north-1**) in relevant regions (such as **CN North-Beijing1**) for the permissions to take effect. If **All projects** is selected, the permissions will take effect for the user group in all region-specific projects. When accessing FunctionGraph, the users need to switch to a region where they have been authorized to use the FunctionGraph service.

You can grant users permissions by using roles and policies.

- Roles: A type of coarse-grained authorization mechanism that defines permissions related to user responsibilities. This mechanism provides only a limited number of service-level roles for authorization. When using roles to grant permissions, you may also need to assign other roles on which the permissions depend. However, roles are not an ideal choice for fine-grained authorization and secure access control.
- Policies: A type of fine-grained authorization mechanism that defines permissions required to perform operations on specific cloud resources under certain conditions. This mechanism allows for more flexible policy-based authorization, meeting requirements for secure access control.

Table 9-1 lists all the system policies supported by FunctionGraph.

Table 9-1 Permissions description

Role/Policy Name	Description	Category	Dependency
FunctionGraph Administrator	This role has the permissions to manage functions, flows, and triggers, and invoke functions. (It will be unavailable soon and therefore not recommended.)	System-defined role	Tenant Guest
FunctionGraph Invoker	This role has the permissions to query functions, flows, triggers, and invoke functions.	System-defined role	N/A

Role/Policy Name	Description	Category	Dependency
FunctionGraph FullAccess	This policy grants all permissions for FunctionGraph.	System-defined policy	N/A
FunctionGraph ReadOnlyAccess	This policy grants read-only permissions for FunctionGraph.	System-defined policy	N/A
FunctionGraph CommonOperations	This policy grants permissions to query functions and triggers, and invoke functions.	System-defined policy	N/A

NOTE

If an IAM user granted the **FunctionGraph FullAccess** permission has no permission to create a certain type of trigger or use a certain function, the relevant service or function does not support fine-grained authentication. In this case, grant the admin permission for this service or function to the user. These services and functions include:

- CTS, APIG, and DIS: These do not support fine-grained authentication. Add the admin permission for them.
- SMN: This supports fine-grained authentication in some regions. If needed, add the admin permission for this service.
- IoTDA: This is a new trigger type and is not covered in **FullAccess**. When you create an IoTDA trigger, you will be prompted to create an agency and add the **iam:agencies:list** and **iam:agencies:createAgency** permissions.
- TMS, DNS, BSS, Cloud Eye, EG, and DMS: These are new functions and are not covered in **FullAccess**. Add the permissions for them as required.

For more information about the permissions required to use these triggers and relevant functions, see **Table 9-2**.

Table 9-2 Permissions required to use triggers and relevant functions

Trigger/Function	Permission
APIG	apig:groups:get apig:groups:list apig:apis:create apig:apis:delete apig:apis:update apig:apis:publish apig:apis:list apig:apis:get apig:apis:offline apig:apps:list apig:envs:list
APIG (dedicated)	apig:instances:get apig:instances:create apig:instances:update apig:instances:list apig:sharedInstance:operate
CTS	cts:notification:create cts:notification:delete cts:notification:update cts:operation:list cts:tracker:list cts:trace:list
DDS	dds:instance:get dds:instance:list
DIS	dis:streams:list
IoTDA	iotda:routingrules:create iotda:routingrules:delete iotda:routingrules:queryList iotda:routingrules:query iotda:routingactions:create iotda:routingactions:delete iotda:routingactions:query iotda:routingactions:queryList iotda:subscriptions:queryList iotda:rules:modifyStatus iotda:apps:queryList

Trigger/Function	Permission
LTS	lts:groups:create lts:groups:get lts:groups:list lts:groups:put lts:logstreams:delete lts:logstreams:list lts:topics:get lts:subscriptions:create lts:subscriptions:delete lts:subscriptions:put lts:structConfig:create lts:structConfig:get
OBS	obs:bucket:GetBucketLocation obs:bucket:GetBucketNotification obs:bucket:PutBucketNotification obs:bucket:ListBucket
SMN	smn:topic:list smn:topic:update
TMS	tms:predefineTags:list tms:tagValues:list
DNS	dns:recordset:create, dns:recordset:list, dns:recordset:update, dns:zone:create, dns:zone:delete, dns:zone:get, dns:zone:list
BSS	bss:bill:view bss:renewal:view
CES	ces:alarms:get ces:alarms:list ces:alarms:create
DMS	dms:instance:get

Trigger/Function	Permission
EG	eg:subscriptions:get eg:subscriptions:list eg:sources:list eg:sources:get eg:agency:create eg:subscriptions:create eg:subscriptions:delete eg:subscriptions:operate

Table 9-3 lists the common operations supported by each system-defined policy of FunctionGraph. Please choose proper system-defined policies according to this table.

Table 9-3 Common operations supported by each system-defined policy

Operation	FunctionGraph Invoker	FunctionGraph Administrator	FunctionGraph ReadOnlyAccess	FunctionGraph CommonOperations	FunctionGraph FullAccess
Creating functions	×	√	×	×	√
Querying functions	√	√	√	√	√
Modifying functions	×	√	×	×	√
Deleting functions	×	√	×	×	√
Invoking functions	√	√	×	√	√
Querying function logs	√	√	√	√	√
Viewing function metrics	√	√	√	√	√

Helpful Links

- IAM Service Overview

- **Creating a User Group and User and Granting Permissions**
- **Permissions Policies and Supported Actions**

10 Concepts

Function

Functions are code defined to handle events.

Event Source

An event source is a public cloud service or custom application that publishes events.

Synchronous Invocation

Clients wait for explicit responses to their requests from a function. Responses are returned only after the function is invoked.

Asynchronous Invocation

Clients do not care about the function invocation results of their requests. After receiving a request, FunctionGraph puts it in a queue, returns a response, and processes other requests when there are idle resources.

Trigger

A trigger is an event that triggers function execution.

Function Flow

You can drag, configure, and connect components on the UI and create function flow tasks to complete orchestration in complex scenarios.

Single-Instance Multi-Concurrency

The number of requests that can be concurrently processed by an instance.

Custom Images

You can directly package and upload container images. The platform then loads and starts these images to create functions.

Custom Function Execution

You can customize scripts and files to execute functions.

Function Logs

Logs generated during function invocation.

Function Monitoring

Monitoring information generated during function execution.

Function Version

FunctionGraph allows you to publish one or more versions throughout the development, testing, and production processes to manage your function code. The code and environment variables of each version are saved as a snapshot. After the function code is published, modify settings when necessary.

Function Alias

You can create an alias for a specific function version. To roll back to a previous version, use the corresponding alias to represent the version instead of modifying the function code.

Each function alias can be bound to a major version and an additional version for traffic shifting.

Dependency Package

FunctionGraph enables you to manage dependencies in a unified manner. You can upload dependencies from a local path, or through OBS if they are too large, and specify names for them.

For details about how to generate function dependencies, see **How Do I Create Function Dependencies?**

Tracing

Service calls can be traced and recorded, so that the execution paths and statuses of service requests in distributed systems can be restored to quickly locate performance bottlenecks and faults.

Bootstrap File

The **bootstrap** file is the startup file of an HTTP function. The HTTP function can only read **bootstrap** as the startup file name. If the file name is not **bootstrap**, the service cannot be started.

11 Relationships Between FunctionGraph and Other Services

Table 11-1 describes the cloud services that have been interconnected with FunctionGraph.

Table 11-1 Interconnected services

Service	Function
SMN	FunctionGraph functions are constructed to process SMN notifications. For details, see the **SMN User Guide**.
API Gateway	FunctionGraph functions are invoked over HTTPS by defining REST APIs with specified backend services. For details, see the **APIG User Guide**.
OBS	FunctionGraph functions are created to process OBS bucket events, such as object creation or deletion events. For example, when an image is uploaded to the specified bucket, OBS invokes the function to read the image and create a thumbnail. For details, see **OBS User Guide**.
DIS	FunctionGraph functions are created to periodically poll DIS streams for new records, such as website click streams, financial transactions, social media streams, IT logs, and location-tracking events. For details, see the *DIS User Guide*.
CTS	FunctionGraph functions are defined to analyze and process key information in logs according to the event notifications of specified service type and operations configured in CTS. • With CTS, you can record operations associated with FunctionGraph for later query, audit, and backtracking. For details, see **Operations Logged by CTS**. • CTS starts recording operations on cloud resources once being enabled. View traces of the last seven days on the CTS console.

Service	Function
Cloud Eye	FunctionGraph is interconnected with Cloud Eye to report monitoring metrics, allowing you to view function metrics and alarm messages through Cloud Eye. For details, see the **Cloud Eye User Guide**. • For details about function metrics supported by Cloud Eye, see **Monitoring Configuration**.
VPC	Functions can be configured to access resources in Virtual Private Clouds (VPCs) or to access the Internet through source network address translation (SNAT) by binding elastic IP addresses. For details, see the **VPC User Guide**.

