ALIBABA CLOUD

阿里云

函数计算

函数管理

文档版本：20201022

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
`Courier字体`	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.服务管理

1.1. 服务简介

本文主要介绍什么是服务以及服务的属性。

服务定义

服务是函数计算资源管理的单位。从业务场景出发，一个应用可以拆分为多个服务。从资源使用维度出发，一个服务可以由多个函数组成。例如一个数据处理服务，分为数据准备和数据处理两部分。数据准备函数资源需求小，可以选择小规格实例。数据处理函数资源需求大，可以选择大规格实例。创建函数前必须先创建服务，同一个服务下的所有函数共享一些相同的设置，例如服务授权、日志配置。您可以通过控制台或者Funcraft工具创建和管理服务，详情请参见服务操作。

服务属性

在创建服务时，您需要指定下述信息：

参数	是否必选	说明
ServiceName	是	服务名称，在同一地域内唯一，创建后不可修改，需要符合以下约束： • 由英文大小写字母、数字（0~9）、下划线（_）和短划线（-）组成。 • 必须以英文字母（a~z）、（A~Z）或下划线（_）开始。 • 大小写敏感。 • 长度为1~128字符。
Description	否	服务的描述信息。
NasConfig	否	配置NAS选项后，可以让指定服务下的函数访问NAS文件系统时如同访问本地文件系统一样。
Role	否	授予函数计算执行函数所需的权限，使用场景包括： • 授权函数计算服务使用您的日志服务资源来存储、分析函数运行日志的权限。 • 授权函数计算服务访问其他云资源的权限。 关于权限更多信息，请参见权限简介。
LogConfig	否	设置日志服务的日志项目和日志仓库，用于存储和分析函数运行的日志。 强烈建议您开启日志服务，并配置该属性，否则您无法查看函数运行日志。 注意 使用阿里云的日志服务有资源预留的费用，即使您没有产生任何日志，仍需付费（最低￥0.04/每天）。详情请参按量付费。
VpcConfig	否	配置VPC选项可让函数访问指定的VPC。

参数	是否必选	说明
InternetAccess	否	设为true时可让函数访问公网。

1.2. 服务操作

本文介绍如何在函数计算控制台上创建、更新、删除服务。

背景信息

函数计算提供了以下几种方式创建、更新、删除服务。

- 通过控制台操作，即本文介绍的方式。
- 通过Funcraft工具操作，详情请参见功能概览。
- 通过VSCode插件操作，详情请参见 Aliyun Serverless VSCode Extension插件。

创建服务

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏中，单击**服务/函数**。在列表中选择**新建服务**。

4. 在**新建服务**页面填写服务名称，单击**创建**。

enter_name

在**服务/函数**页面的服务列表中可以查看已创建的服务。

更新服务

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 在服务列表中单击目标服务。然后选择**服务配置 > 修改**。

5. 根据需要修改相应的参数，单击**提交**。

删除服务

删除服务前请确保您已删除该服务下的所有函数，详情请参见删除函数。

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 在服务列表中单击目标服务，将鼠标移至 上，然后在列表中选择**删除服务**。

2.函数管理

2.1. 函数简介

函数是系统调度和运行的单位。函数必须从属于服务，同一个服务下的所有函数共享一些相同的设置，例如服务授权、日志配置。

函数属性

在创建函数时，您需要指定以下信息：

属性	是否必选	描述
FunctionName	是	函数名称。在当前服务内唯一，并符合以下约束： • 由英文大小写字母、数字（0~9）、下划线（_）和短划线（-）组成。 • 必须以英文字母（a~z）、（A~Z）或下划线（_）开始。 • 大小写敏感。 • 长度为1~128字符。
Runtime	是	函数运行时的环境类型。
Code	是	代码包。Java语言需要上传JAR包，其他语言上传ZIP包，可以存放在OSS上，或者直接上传代码包。
Handler	是	入口函数，函数计算系统运行您的函数的调用入口。
Description	否	函数的描述。函数计算系统并不会使用该属性值，但建议您为函数设置一个简洁、清晰的描述。
Timeout	否	函数的最大运行时间，单位为秒。
InstanceType	否	函数实例类型。取值： • ElasticInstance：弹性实例 • EnhancedInstance：性能实例
MemorySize	否	函数运行所需的内存资源，单位为MB。取值范围为[128, 3072]，以64 MB为步长递进。
Initializer	否	函数计算系统运行您的初始化函数的调用入口。
InitializationTimeout	否	Initializer最大运行时间，单位为秒。

除函数名字外，其他属性均可后续修改。

支持的函数运行环境列表

运行环境	说明	文档链接
nodejs6	Node.js 6.10.3版本	

运行环境	说明	文档链接
nodejs8	Node.js 8.9.0版本	Node.js运行环境
nodejs10	Node.js 10.15.3版本	
nodejs12	Node.js 12.16.1版本	
python2.7	Python 2.7版本	Python运行环境
python3	Python 3.6版本	
php7.2	PHP 7.2.7版本	PHP运行环境
java8	Java 8版本	Java运行环境
dotnetcore2.1	.NET Core 2.1版本	.NET Core运行环境
custom	无	简介
custom-ontainer	无	简介

相关文档

函数操作

2.2. 函数操作

同一个服务下可以创建多个函数，这些函数共享服务配置的日志资源和角色信息，但彼此相互独立，互不影响。本文介绍如何创建、配置、删除函数。

背景信息

函数计算提供了以下几种方式创建、配置、删除函数：

- 使用控制台进行函数操作，即本文介绍的内容。
- 使用fcli进行函数操作。

创建函数

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 单击**新建函数**，选择创建方式，然后单击**下一步**。本文以创建事件函数为例。

   

5. 在**配置函数**区域，填写相关信息，然后单击**完成**。

   

参数	操作	示例值
所在服务	○ 若已创建服务：在列表中选择已存在的服务。 ○ 若未创建服务：填写自定义的服务名称，系统将自动为您创建服务。	service

参数	操作	示例值
绑定日志	使用系统自动创建服务时需要配置该参数。 绑定日志后，您查看函数执行日志，方便函数开发调试。	示例中选择了已有服务，该参数无需配置。
函数名称	填写自定义的函数名称。	function
运行环境	选择您熟悉的语言，例如Python、Java、PHP、Node.js等。	python3
函数实例类型	选择适合您的实例类型，取值： ○ **弹性实例** ○ **性能实例** 更多信息请参见实例规格。	弹性实例
函数入口	填写函数入口。格式为[文件名].[函数名]。	index.handler
函数执行内存	设置函数执行内存。默认内存大小为512 MB，最大为3072 MB。	512 MB
超时时间	设置超时时间。默认超时时间为60秒，最长为600秒。 超过设置的超时时间，函数将以执行失败结束。	60
实例并发度	单个实例能够并发处理的请求数。	python3不支持设置

在**服务/函数**页面，单击目标服务，可以查看已创建的函数。

更新函数

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 单击目标函数所在的服务。
5. 在函数列表中找到目标函数，单击**操作**列的**配置**。
6. 根据需要修改相应的参数，单击**提交**。函数计算支持修改函数入口、运行环境、实例类型、函数执行内存、超时时间、单实例并发度、是否配置函数初始化入口、描述信息、环境变量。

> 说明 实例类型仅支持从弹性实例修改为性能实例，不支持从性能实例修改为弹性实例。

删除函数

删除服务前请确保您已删除该服务下的所有触发器，详情请参见删除触发器。

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。

3. 在左侧导航栏，单击**服务/函数**。
4. 单击目标函数所在的服务。
5. 在函数列表中找到目标函数，单击**操作**列的**删除**。

3.版本管理

3.1. 版本简介

阿里云函数计算支持版本管理功能，帮助您更高效地管理服务、函数以及触发器。您可以通过版本管理实现大多数软件开发生命周期中持续集成、持续发布。

版本（Version）定义

版本相当于服务的快照，包括服务的配置、服务内的函数代码及函数配置，不包括触发器。当您发布版本时，函数计算会为服务生成快照，并自动分配一个版本号与其关联，以供后续使用。版本发布后不能更改，且版本号单调递增，不会被重复使用。

函数计算提供服务级别的版本控制功能，支持您为自己的服务发布一个或多个版本。例如，如果没有版本功能，您在服务上的每次改动都会立刻影响到生产环境，无法控制发布的时机。有了版本功能，您可以在测试稳定后发布服务版本，用稳定的版本来服务线上请求，并且可以继续在LATEST版本上开发测试。

注意事项

- 新创建的服务，只有一个LATEST版本，在未发布任何版本前，LATEST版本是您拥有的唯一服务版本，LATEST版本不能被删除。
- 版本发布后，已发布的版本不可更改。

3.2. 别名简介

阿里云函数计算支持别名管理，别名是版本控制过程中涉及的一种资源。您可以通过结合别名和版本实现软件开发生命周期持续集成、持续发布的功能。

别名定义

别名可以理解为指向特定服务版本的指针，是函数计算的一种资源。别名无法脱离服务或版本单独存在。使用别名访问服务或函数时，函数计算会将别名解析为其指向的版本，调用方无需了解别名指向的具体版本。

函数计算支持为服务的版本创建别名。别名指向特定服务版本，可以更改。别名不能指向其他别名。您可以利用别名来轻松实现发布、回滚以及灰度发布等功能。

例如，假设第三方是通过HTTP触发器来触发您的函数，如果没有别名，每次新版本上线，您需要手动修改HTTP触发器关联的版本号，并且在修改的过程中会影响客户端的使用。而如果使用别名进行版本管理，您可以实现版本的平滑升级。

您可以设置别名PROD指向稳定的版本1。若您的函数下存在触发器，设置触发器关联别名PROD，详情请参见在触发器中使用别名。客户端直接通过别名PROD调用版本1下的函数。

发布版本1

版本1发布后，您可以继续在LATEST版本上开发新功能。由于客户端是通过别名调用对应版本下的函数，当需要发布新版本2时，只需要将别名PROD更新为指向版本2，此时，客户端调用时解析出的版本即为版本2，这样就可以完成版本的更新迭代。您也可以更改别名的指向，将别名PROD重新指向版本1回滚到之前的版本。通过这种方式发布，不需要频繁更新触发器，也不会影响客户端的使用。

发布版本2

别名属性

在创建别名时，您需要指定以下信息：

> 说明 除别名名称外，其他属性均可后续修改。

参数	是否必选	描述
AliasName	是	别名名称。在当前函数计算服务内唯一，并符合如下约束： • 由英文大小写字母、数字（0~9）、下划线（_）和短划线（-）组成。 • 必须以英文字母（a~z）、（A~Z）或下划线（_）开始。 • 大小写敏感。 • 长度为1~128字符。 • 别名名称不能为LATEST，LATEST为函数计算占用的默认版本名称。
VersionId	是	别名指向的版本。别名不能指向系统预留的版本，即LATEST。
Description	否	别名的描述信息。
AdditionalVersionWeight	否	别名指向的灰度版本以及灰度权重。 • 灰度版本只在调用函数时生效。 • 由版本号和对应的权重组成，例如，2:0.05表明调用函数时，版本2为灰度版本，切5%的流量到灰度版本，95%的流量默认到主版本。

3.3. 版本操作

为了在函数计算的环境中更好地管理您的服务及函数，您可以通过版本功能来发布多个版本的服务，并在开发工作流中使用这些版本。本文介绍如何在函数计算控制台发布、查看以及删除版本。

发布版本

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击服务/函数。
4. 单击目标服务，选择版本管理 > 发布版本。

5. 在从LATEST发布新版本对话框，填写版本描述，然后单击确定。
   版本列表中可查看刚发布的版本。

查看版本

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击服务/函数。
4. 单击目标服务，单击版本管理页签。
   您可以查看版本。

删除版本

删除一个版本将删除该版本中包含的函数和配置，并不会删除指向该版本的别名或者触发器。建议您在删除版本前先移除指向该版本的别名和触发器，删除后该版本中的函数将不能被触发。

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 单击目标服，单击**版本管理**页签。
5. 找到目标版本，单击**操作**列中的**删除版本**。

相关文档

您还可以通过Fcli命令行工具发布、查看以及删除版本，详情请参见初次使用fcli。

3.4. 别名操作

函数计算支持为服务版本创建别名。别名可以理解为指向特定服务版本的指针，是函数计算的一种资源。本文介绍如何在函数计算控制台创建、更新、查看以及删除别名。

创建别名

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 单击目标服务，然后单击**版本管理**页签。
5. 在版本列表找到需要创建别名的版本，单击**操作**列中的**新建别名**。
6. 在**新建别名**对话框中，填写别名的相关信息，单击**确定**。

参数说明

参数	是否必填	操作	示例
名称	是	填写自定义的别名名称。	alias
描述	否	填写该别名的描述信息。	无
灰度版本	否	若您需要将部分请求切到灰度版本处理，则在**灰度版本**列表中选择灰度版本。	2
权重	否	当需要灰度版本处理部分请求时，该参数必填。填写灰度版本的权重。	30

在目标服务的**版本管理**页签下，单击**别名**页签。您可以看到刚创建的别名。本示例中该别名指向版本1，灰度版本为版本2，灰度版本权重为30%。

查看别名

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击服务/函数。
4. 单击目标服务，选择版本管理 > 别名。
   您可以在别名列表中看到已创建的别名。

更新别名

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击服务/函数。
4. 单击目标服务，选择版本管理 > 别名。
5. 找到目标别名，单击操作列中的更新别名。
6. 在新建别名对话框，修改别名信息，然后单击确定。
   参数说明请参见参数说明。

删除别名

删除一个别名只会删除别名本身，并不会删除别名指向的版本，也不会删除指向此别名的触发器。建议删除别名前先修改触发器关联的别名。

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击服务/函数。
4. 单击目标服务，选择版本管理 > 别名。
5. 找到目标别名，单击操作列中的删除别名。
6. 在提示框中，单击确定。
   您可以在别名列表中看到目标别名已被移除。

API参考

函数计算提供了以下API用于管理别名：

- CreateAlias
- GetAlias
- UpdateAlias
- ListAliases
- DeleteAlias

其他与别名相关的API如下：

- GetService
- GetFunction
- GetFunctionCode
- ListFunctions
- InvokeFunction
- CreateTrigger

- UpdateTrigger

3.5. 版本管理示例

3.5.1. 示例介绍

本示例介绍如何使用函数计算的版本管理功能管理版本。

版本管理流程

1. 准备函数
   当您初次创建一个服务以及该服务下的函数时，该服务的版本号为LATEST。您可以调试LATEST版本下的函数直至版本稳定运行。
2. 发布版本
   当LATEST版本的服务稳定时，就可以发布该版本的服务，让稳定的版本来服务线上的请求。
3. 使用别名切换流量
   版本上线后，您可以创建一个别名，设置别名指向该版本。当该版本更新时，只需要将别名指向的版本更改为更新的版本。这样，调用方无需关心服务的具体版本，只需要使用正确的别名即可。如果您的函数下存在触发器，只需要将触发器与别名关联，实现切换版本时不影响触发器的使用。

查看执行函数的版本

在执行函数时，如果需要查看函数运行时的版本或别名，需要在函数中添加以下代码（本文以Node.js语言为例）。

```
module.exports.handler = function(eventBuf, context, callback) {
    var qualifier = context['service']['qualifier']
    var versionId = context['service']['versionId']
    console.log('Qualifier from context:', qualifier);
    console.log('VersionId from context:', versionId);
    callback(null, null);
};
```

其中：

- qualifier：调用函数时传入的版本信息，可以是版本号，也可以是别名。
- versionId：函数执行时根据qualifier解析出的具体版本号。

3.5.2. 准备函数

您需要创建服务及函数，并调试函数确保函数能稳定执行，为发布版本做准备。本文介绍如何通过函数计算控制台或API调用LATEST版本下的函数。

前提条件

1. 创建服务
2. 创建函数

本示例创建的服务名称为service，创建的函数名称为function。

通过控制台执行LATEST版本下的函数

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击服务/函数。
4. 找到目标服务service下的目标函数function，单击函数名称。
5. 单击代码执行页签，在您的函数代码中添加查看函数版本的代码。查看函数版本的代码示例如下。

Nodejs | Python | PHP | C#

```
module.exports.handler = function(eventBuf, context, callback) {
 var qualifier = context['service']['qualifier']
 var versionId = context['service']['versionId']
 console.log('Qualifier from context:', qualifier);
 console.log('VersionId from context: ', versionId);
 callback(null, qualifier);
};
```

6. 单击执行。
执行完成后，在页面底部可以查看执行日志。从日志中可以看到表示版本信息的字段qualifier的值为LATEST，即本次执行的函数为LATEST版本下的函数。

通过API调用LATEST版本下的函数

InvokeFunction API调用LATEST版本有以下两种情况：

- 未使用版本管理时调用LATEST版本的函数
  当没有指定版本信息时，会默认调用LATEST版本下的函数。请求格式如下：

  ```
  POST /services/{serviceName}/functions/{functionName}/invocations
  ```

  本文的请求示例如下：

  ```
  POST /services/{service}/functions/{function}/invocations
  ```

- 指定版本后缀调用LATEST版本下的函数
  通过在request path的serviceName后加 `.LATEST` 后缀来指定调用LATEST版本下的函数。请求格式如下所示：

  ```
  POST /services/{serviceName}.LATEST/functions/{functionName}/invocations
  ```

  本文的请求示例如下：

  ```
  POST /services/{service}.LATEST/functions/{function}/invocations
  ```

后续步骤

1. 发布版本

2. 使用别名切换流量

3.5.3. 发布版本

当LATEST版本的服务稳定时，就可以发布该版本的服务，让稳定的版本来服务线上的请求。同时您可以继续在LATEST版本上开发更多的功能。

前提条件

准备函数

发布版本

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击服务/函数。
4. 单击目标服务，选择版本管理 > 发布版本。

5. 在从LATEST发布新版本对话框，填写版本描述，然后单击确定。
   版本列表中可查看刚发布的版本。

通过控制台执行新版本下的函数

1. 在服务/函数页面，找到目标服务service。
2. 在函数列表的右上角，选择新版本的版本号。

3. 单击目标函数名称function，然后单击代码执行页签。
4. 单击执行。
   执行完成后，在页面底部可以查看执行日志。从日志中可以看到函数执行时的版本信息qualifier为1，解析出的versionId为1，即本次执行的函数为版本1下的函数。

通过API执行新版本下的函数

通过在request path的serviceName后加分隔符“.”，并用qualifier指定版本来调用特定版本下的函数。请求格式如下所示：

```
POST /services/{serviceName}.{qualifier}/functions/{functionName}/invocations
```

本文示例如下：

```
POST /services/{service}.{qualifier}/functions/{function}/invocations
```

后续步骤

使用别名切换流量

3.5.4. 使用别名切换流量

在有了版本的基础上，您可以使用别名控制版本。服务的调用方无需了解服务版本，就可以通过别名调用正确的服务版本。

前提条件

1. 准备函数
2. 发布版本

步骤一：创建别名

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 单击目标服务service，然后单击**版本管理**页签。
5. 在版本列表找到需要创建别名的版本，单击**操作**列中的**新建别名**。

6. 在**新建别名**对话框中，填写别名名称，然后单击**确定**。
   切换到**别名**页签下，可以看到刚创建成功的别名alias以及该别名指向的版本1。

（可选）步骤二：在触发器中使用别名

若您的函数中创建了触发器，您可以将触发器与别名关联。当线上版本切换时，触发器可以不受影响，平滑切换到别名指向的版本。

本文以OSS触发器为例说明如何在触发器中使用别名。

> **说明** 触发器本身并没有版本，触发器可以被指向特定的服务版本或别名。

1. 在**服务/函数**页面，找到目标服务service。
2. 在LATEST版本下，单击目标函数名称。
3. 单击**触发器**页签。
4. 在触发器列表，单击目标触发器名称。
5. 在**修改触发器**页面，将**触发版本/别名**由LATEST修改为刚创建的别名。然后单击**确定**。

步骤三：执行指定别名下的函数

您可以通过控制台或API调用的方式验证是否执行了正确版本的函数。

1. 在**服务/函数**页面，找到目标服务service。
2. 单击目标函数名称。
3. 在函数详情页的右上角，选择服务别名PROD。

4. 单击**代码执行**页签。
5. 单击**执行**。
   执行完成后，在页面底部可以查看执行日志。从日志中可以看到函数执行时的版本信息qualifier为PROD，解析出的versionId为1，即本次执行的函数为别名PROD下的函数，该别名指向版本1。

通过在request path的serviceName后加分隔符“.”，并用qualifier指定别名来调用特定别名下的函数。请求格式如下：

```
POST /services/{serviceName}.{qualifier}/functions/{functionName}/invocations
```

本文示例如下：

```
POST /services/{service}.{qualifier}/functions/{function}/invocations
```

通过控制台执行指定别名下的函数

通过API调用指定别名下的函数

步骤四：发布灰度版本

当新版本开发完成后，需要使用灰度版本帮助新版本的稳定发布。

1. 发布新版本，操作步骤请参见发布版本。
   发布完成后，您可以在版本列表查看新发布的版本。
2. 在**版本管理**页签下，单击**别名**页签。
3. 在别名列表中找到步骤一中创建的指向版本1的别名PROD，单击**操作**列的**更新别名**。
4. 在**更新别名**对话框，将新版本设置为灰度版本，并填写灰度版本的权重。然后单击**确定**。

   设置完成后，您可以将一部分线上流量切换到新版本。

步骤五：全量切换至新版本

当灰度版本运行稳定后，您可以将线上的流量全量切换到新版本。

1. 在**版本管理**页签下，单击**别名**页签。
2. 在别名列表中找到步骤四中指向版本1和灰度版本2的别名PROD，单击**操作**列的**更新别名**。
3. 在**更新别名**对话框，将主版本设置为新版本且不设置灰度版本。然后单击**确定**。

   设置完成后，您的线上流量被全量切换至新版本。

3.6. 常见问题

如何确定被调用的服务的版本？

当别名的灰度发布功能被使用时，函数计算将会按照您指定的权重来分配流量，您可以通过以下方式来确定被调用的服务的版本。

- 通过日志确定
  每次函数调用，您都可以在配置的日志服务中看到相应的请求。日志中的isDarkLaunch字段表示此次调用命中的是灰度发布的版本。日志中的externalServiceVersion字段表示此次调用使用的具体服务版本。
  更多版本管理相关的日志字段，请参见调用统计及监控报警。

- 通过context入参确定
  每次函数调用，context入参的service参数中会包括qualifier和versionId两个字段。
  - qualifier：调用函数时传入的版本信息，可以是版本号，也可以是别名。
  - versionId：函数执行时根据qualifier解析出的具体版本号。
- 通过同步函数调用响应确定
  同步函数调用的响应包含x-fc-invocation-service-version header，可以指示已调用的服务版本。

4.配置并查看函数日志

您可以将函数执行的日志存储至阿里云日志服务，再根据日志服务中存储的函数日志来执行代码调试、故障分析、数据分析等操作。本文介绍如何使用控制台来为函数计算的服务配置日志项目（Project）和日志仓库（Logstore），并查看函数执行的日志。

背景信息

日志服务SLS（Log Service）是阿里云提供的针对日志类数据的一站式服务，通过日志服务存储函数日志需要在函数对应的服务中配置日志项目和日志仓库，并授予该服务访问日志服务的权限。函数日志会打印到配置的日志仓库中，同一个服务下的所有函数日志都会打印到同一个日志仓库中。

操作步骤

1. 登录函数计算控制台，为服务配置日志项目和日志仓库。
   - 您可以在创建服务时通过勾选**绑定日志**配置。详细步骤请参见创建服务。
     服务创建成功后，函数计算会在后台为您创建并绑定相应的日志服务的日志项目和日志仓库，并被授予在您的日志服务资源中写入函数日志的权限。

     > **说明** 函数计算在后台为您创建的日志服务资源计费模式为按量计费，详情请参见计费概述。

     enter_name
   - 您可以在更新服务时配置。详细步骤请参见更新服务。配置前请确保相应资源已在日志服务创建。详细步骤请参见创建日志项目和日志仓库。
     如下图所示，您需在**日志配置**区域选择您已创建的**日志项目**和**日志仓库**，并在**权限配置**区域配置权限以允许函数计算在您的日志服务中写入函数执行的日志。权限的详细内容请参见权限简介。
2. 使用日志服务控制台查看日志。详细步骤请参见查询日志。
   在配置了日志项目和日志仓库的服务中新建一个默认函数，输出日志hello world，您的函数运行时，产生的日志都会输出到日志库中，可以在日志服务控制台查看。

更多信息

除了通过控制台，您还可以使用Fcli来配置相应日志服务资源并查看函数执行的日志。详情请查看初次使用fcli。

5.权限管理

5.1. 权限简介

本文介绍函数计算中权限的应用场景、类型以及管理机制。

应用场景

使用函数计算构建应用时，会涉及各种权限。例如：

- 当您想使用阿里云日志服务（SLS）收集函数运行日志时，您需要授予函数计算将函数运行日志写入您指定的日志库中的权限。
- 当您想使用阿里云对象存储服务（OSS）触发器时，您需要授予OSS调用函数的权限。
- 当您需要不同的人员管理的函数需要访问账户中的阿里云资源时，例如OSS中的数据，您可以创建RAM角色并授予函数访问阿里云资源的权限。

权限类型

访问阿里云云产品，需要拥有对该产品的访问权限。函数计算涉及的权限主要有以下几种：

- 函数计算访问阿里云其他产品，需要授予函数计算访问其他产品的权限。
  该权限是服务级别的，当某服务配置了指定权限，同一服务下的所有函数都继承该权限。您的函数代码中使用 `context.credentials` 来访问其他云产品，示例代码如下。

```
// 使用context访问OSS。
var OSSClient = require('ali-oss').Wrapper;
exports.handler = function (event, context, callback) {
  console.log(event.toString());

  var ossClient = new OSSClient({
    accessKeyId: context.credentials.accessKeyId,
    accessKeySecret: context.credentials.accessKeySecret,
    stsToken: context.credentials.securityToken,
    region: 'oss-cn-shanghai',
    bucket: 'my-bucket',
  });

  ossClient.put('my-object', new Buffer('hello, fc')).then(function (res) {
    callback(null, 'put object');
  }).catch(function (err) {
    callback(err);
  });
};
```

  如何配置服务权限，请参见配置服务权限。

- 事件源触发函数执行，需要授予事件源访问函数计算的权限。
  该权限是触发器级别的权限，每个触发器需要根据实际需要设置相应的权限。
  触发器的权限是在创建触发器过程中配置，具体操作步骤请参见：

- 创建OSS触发器
- 创建MNS主题触发器
- 创建日志服务触发器
- 创建TableStore触发器
- 创建CDN事件触发器
- 创建定时触发器

- RAM子账号访问函数计算资源需要授予相应的权限。
  该权限是RAM账号级别的。访问控制（RAM）允许在一个云账号下创建并管理多个身份，并允许给单个身份或一组身份分配不同的权限，从而实现不同权限组拥有不同的云资源访问权限。您可以通过主账号给RAM子账号授权，让子账号有权限操作函数计算相关资源。
  如何为RAM子账号设置权限，请参见为RAM用户授权。
  函数计算RAM子账号的操作权限、资源访问权限及系统权限策略如下：
  - 函数计算RAM操作（Action）

接口名称	描述	Action
ListServices	获取服务列表。	fc:ListServices
GetService	获取指定服务。	fc:CreateService
CreateService	新建一个服务。	fc:GetService
UpdateService	更新指定服务。	fc:UpdateService
DeleteService	删除指定服务。	fc:DeleteService
ListFunctions	获取服务下的函数列表。	fc:ListFunctions
GetFunction	获取指定函数的配置信息。	fc:GetFunction
CreateFunction	新建一个新函数。	fc:CreateFunction
UpdateFunction	更新函数，包括配置和代码。	fc:UpdateFunction
DeleteFunction	删除指定的函数。	fc:DeleteFunction
InvokeFunction	触发函数，分为同步和异步触发。	fc:InvokeFunction
ListTriggers	获取函数下的触发器列表。	fc:ListTriggers
GetTrigger	获取指定触发器。	fc:GetTrigger
UpdateTrigger	更新指定函数触发器配置。	fc:UpdateTrigger
DeleteTrigger	删除指定函数的触发器。	fc:DeleteTrigger

- 函数计算RAM资源

资源	描述
acs:fc:<region>:<account-id>:services/<serviceName>	特定服务资源。
acs:fc:<region>:<account-id>:services/*	所有服务资源。
acs:fc:<region>:<account-id>:services/<serviceName>.<qualifier>	特定版本的服务资源。
acs:fc:<region>:<account-id>:services/<serviceName>.*	特定服务的所有版本资源。
acs:fc:<region>:<account-id>:services/<serviceName>/functions/<functionName>	特定服务下的特定函数资源。
acs:fc:<region>:<account-id>:services/<serviceName>/functions/*	特定服务下的所有函数资源。
acs:fc:<region>:<account-id>:services/<serviceName>.*/functions/*	特定服务的所有版本下的所有函数资源。
acs:fc:<region>:<account-id>:services/<serviceName>.<qualifier>/functions/*	特定服务的指定版本下的所有函数资源。
acs:fc:<region>:<account-id>:services/<serviceName>/functions/<functionName>/triggers/<triggerName>	特定服务下的特定函数下的特定触发器资源。
acs:fc:<region>:<account-id>:services/<serviceName>/functions/<functionName>/triggers/*	特定服务下的特定函数下的所有触发器资源。

- 函数计算系统权限策略
  函数计算默认提供三个系统策略AliyunFCReadOnlyAccess、AliyunFCInvocationAccess和AliyunFCFullAccess。您也可以自定义策略进行更细粒度的权限管理，具体请参考权限策略基本元素。

- AliyunFCReadOnlyAccess系统策略：表示允许对函数计算所有资源进行读操作。

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": [
        "fc:Get*",
        "fc:List*"
      ],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

- AliyunFCInvocationAccess系统策略：表示允许对所有函数资源进行执行操作。

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": [
        "fc:InvokeFunction"
      ],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

- AliyunFCFullAccess系统策略：表示允许对所有函数计算资源进行所有执行操作。

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": "fc:*",
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

- 自定义策略：表示允许对杭州区域下的foo服务下的bar函数进行执行操作。

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": [
        "fc:InvokeFunction"
      ],
      "Resource": "acs:fc:cn-hangzhou:*:services/foo/functions/bar",
      "Effect": "Allow"
    }
  ]
}
```

权限管理机制

访问控制服务RAM（Resource Access Management）是阿里云提供的资源访问控制服务。函数计算使用RAM基于角色的权限管理机制。

- 授权基本思路
  策略（policy）表示访问指定服务的能力，当您为指定的角色（role）绑定指定策略后，该角色就具有访问指定服务的能力。当有第三方需要访问这个服务时，只需要扮演具有访问能力的角色即可。关于更多策略（policy）和角色（role）的内容请参见访问控制。
- 授权示例
  - 函数计算访问阿里云其他产品
    以函数计算访问日志服务为例，RAM提供系统授权策略AliyunLogFullAccess。该策略具有对日志服务完全的操作权限。在配置服务权限时，您可以为该服务绑定一个角色（可以新建一个角色也可以使用已有角色）。然后将策略AliyunLogFullAccess绑定到该角色上，这样函数计算就可以访问日志服务了。
  - 事件源访问函数计算
    以OSS事件源触发函数计算代码执行为例，RAM提供系统授权策略AliyunOSSEventNotificationRole。该策略具有OSS事件源触发函数计算代码执行的权限。在创建触发器时，您可以为触发器绑定一个角色（可以新建角色也可以使用已有角色）。然后将策略AliyunOSSEventNotificationRole绑定到该角色上，这样OSS事件源就可以触发函数计算代码执行了。
  - RAM子账号访问函数计算
    以授予RAM子账号对函数计算中所有资源读权限为例，函数计算提供系统授权策略AliyunFCReadOnlyAccess。在创建RAM子账号后，您可以为该子账号绑定一个角色（可以新建角色也可以使用已有角色）。然后将策略AliyunFCReadOnlyAccess绑定到该角色上，这样该子账号就可以对读函数计算中的所有资源了。

5.2. 配置服务权限

本文介绍如何在函数计算控制台为函数计算授予访问其他云产品的权限。

前提条件

创建服务

背景信息

函数计算访问阿里云其他产品，需要授予函数计算访问其他云产品的权限。该权限是服务级别的，当某服务配置了指定权限，同一服务下的所有函数都继承该权限。

操作步骤

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 在服务列表，单击目标服务名称。
5. 选择**服务配置 > 修改**。

6. 在**权限配置**区域，配置相关参数，然后单击**提交**。
   - 若您还未创建角色
     a. 新建角色。

参数	操作
角色创建方式	在列表中选择**新建角色**。
系统模板授权	在列表中选择您希望新建角色拥有的权限策略，可多选。

     b. 单击**点击授权**。
     页面跳转至**角色快捷创建**页面。
     c. 填写自定义的**角色名称**和**策略名称**，然后单击**同意授权**。

   - 若您已创建了角色
     a. 为已有角色授权。

参数	操作
角色创建方式	在列表中选择**选择已有角色**。
已存在角色	在列表中选择已创建的角色。
系统模板授权	在列表中选择您希望该角色拥有的权限策略，可多选。

     b. 单击**点击授权**。

c. 填写策略名称，然后单击同意授权。

6.允许指定的VPC调用函数

函数创建完成后，默认可以通过公网地址和阿里云内网地址调用函数。从安全性的角度出发，如果您希望函数只可以通过特定的VPC来调用，而无法通过公网和内网调用，则需要为服务绑定指定的VPC。本文介绍如何绑定一个指定的VPC以允许该VPC调用函数的操作步骤。

前提条件

您已完成以下操作：

- 创建服务
- 创建函数

注意事项

- 同一个服务最多绑定20个VPC。
- 设置仅允许指定VPC调用函数后，使用触发器调用函数不受影响。
- VPC绑定后对服务的所有版本和别名生效。
- 设置允许指定VPC调用函数后，会拒绝来自公网和其他VPC的调用请求，`StatusCode`为403，`ErrorCode`为`AccessDenied`，错误信息为`Resource access is bound by VPC: VPCID`。

绑定VPC

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 找到目标服务，进入服务配置，单击**修改**。
5. 在**网络配置**中打开**允许函数访问 VPC 内资源**，并选择**VPC ID**。
6. 如果您的服务已有角色，在权限配置中选择已有角色，并在系统模板授权中选择**AliyunVPCReadOnlyAccess**，单击**点击授权**。
7. 单击**提交**。
   操作完成后，此服务下的所有函数都只可以通过指定VPC访问。

7.访问VPC内资源

7.1. 配置函数访问VPC内资源

默认情况下，函数计算无法访问VPC中的资源，如果您想让函数计算能够访问VPC中的资源，您需要手动为服务配置VPC功能和相关权限。

前提条件

- 创建服务
- 创建函数
- 创建专有网络和交换机
- 创建安全组

背景信息

使用VPC功能会带来一些额外的费用，如果可以使用RAM授权方式访问的资源建议您不要使用VPC功能，例如OTS。因此，在配置VPC前，需要先判断您的场景是否需要使用VPC功能。

函数计算访问VPC的配置是服务级别的，为一个服务配置了访问VPC的能力后，那么此服务下的所有函数都可以访问VPC。

> **说明** 如果您的VPC资源不在函数计算当前可用区，可以通过在VPC环境中创建一个与函数计算相同可用区的VSwitch，并在函数计算的服务的VPC配置中设置此VSwitchID。由于同一VPC内不同交换机之间内网互通，因此函数计算可以通过该VSwitch访问在其他可用区的VPC内资源。

vpcConfig下有三个字段vpcId、vSwitchIds、securityGroupId，每一个字段都不允许为空。

- vpcId：要访问的VPC ID。
- vSwitchIds：一系列交换机的列表，至少要提供一个VSwitch ID。
  该字段限定了函数计算可以访问的子网，建议在vSwitchIds中设置两个或多个VSwitch，如果一个可用区出现故障或IP地址不足，您的函数可以在其他子网下运行。如果指定了多个VSwitch ID，函数计算在创建ENI的时候会随机选取一个。
- securityGroupId：ENI所在安全组的ID。

```
"vpcConfig": {
    "vpcId": "string",
    "vSwitchIds": [ "string" ],
    "securityGroupId": "string"
 }
```

此安全组是ENI所在的安全组，也就是函数计算所在的安全组，安全组限定了函数计算在VPC中的出入站规则。需要设置您的VPC所在安全组的入站规则为允许函数计算所在的安全组访问。否则，函数计算无法顺畅地访问您的VPC内资源。

函数计算的服务中有internetAccess字段，布尔类型，用于表示此服务是否可以访问公网，默认值为true，表示此服务可以访问互联网。您可以将internetAccess字段设置为false，表示此服务下的所有函数都无法连接互联网。

判断是否需要配置VPC

vpcConfig属性

访问公网

原理说明

专有网络VPC是基于阿里云创建的自定义私有网络，不同的VPC之间彻底逻辑隔离。您可以在自己创建的VPC内创建和管理云产品实例，例如ECS、SLB、RDS等，以便这些资源不被公共互联网访问。

函数计算访问VPC内的资源原理如下：

VPC是您的私有网络，需要授予弹性网卡ENI访问VPC的权限，并将此弹性网卡ENI插入到执行您的函数的实例上，从而使函数可以访问您VPC内的资源。关于弹性网卡ENI的内容请参见弹性网卡概述。

创建弹性网卡ENI时，需要您提供VPC ID、安全组ID、交换机ID等配置信息，函数计算使用这些信息配置弹性网卡ENI，您的函数可以通过ENI安全访问指定VPC中的资源。

函数计算访问VPC内资源的使用示例请参见访问数据库概述。

注意事项

- 华东1（杭州）、华东2（上海）、华北2（北京）、华南1（深圳）这四个地域如果无法使用VPC功能，则需要根据控制台的申请提示申请开通。
- 函数计算支持访问的可用区列表如下。如果您的资源所在的可用区不在以下列表中，请参见遇到VSwitch is in unsupported zone的错误怎么办？。

地域	地域ID	VPC
华东1（杭州）	cn-hangzhou	cn-hangzhou-f,cn-hangzhou-g,cn-hangzhou-h
华东2（上海）	cn-shanghai	cn-shanghai-b,cn-shanghai-e,cn-shanghai-g,cn-shanghai-f
华北1（青岛）	cn-qingdao	cn-qingdao-c
华北2（北京）	cn-beijing	cn-beijing-h,cn-beijing-c,cn-beijing-e,cn-beijing-f
华北3（张家口）	cn-zhangjiakou	cn-zhangjiakou-b,cn-zhangjiakou-a
华北5（呼和浩特）	cn-huhehaote	cn-huhehaote-a,cn-huhehaote-b
华南1（深圳）	cn-shenzhen	cn-shenzhen-e,cn-shenzhen-d
西南1（成都）	cn-chengdu	cn-chengdu-a, cn-chengdu-b
中国香港	cn-hongkong	cn-hongkong-c
新加坡	ap-southeast-1	ap-southeast-1a,ap-southeast-1b

地域	地域ID	VPC
澳大利亚（悉尼）	ap-southeast-2	ap-southeast-2a,ap-southeast-2b
马来西亚（吉隆坡）	ap-southeast-3	ap-southeast-3a
印度尼西亚（雅加达）	ap-southeast-5	ap-southeast-5a,ap-southeast-5b
日本（东京）	ap-northeast-1	ap-northeast-1b,ap-northeast-1a
英国（伦敦）	eu-west-1	eu-west-1a
德国（法兰克福）	eu-central-1	eu-central-a,eu-central-1a,eu-central-1b
美国（硅谷）	us-west-1	us-west-1a,us-west-1b
美国（弗吉尼亚）	us-east-1	us-east-1b，us-east-1a
印度（孟买）	ap-south-1	ap-south-1a,ap-south-1b

网络访问能力

根据对网络的不同设置，函数存在四种类型的网络访问能力，您可以根据自己的需求设置。

是否允许函数访问公网	是否允许函数访问VPC内资源	网络访问能力
是	是	函数可以访问公网，也可以访问VPC。
是	否	函数可以访问公网，不可以访问VPC。
否	是	函数不访问公网，可以访问VPC。
否	否	函数不可以访问公网，也不可以访问VPC。

配置网络及权限

函数计算访问VPC的配置以及权限的配置是服务级别的，为一个服务配置了访问VPC的能力后，此服务下的所有函数都可以访问VPC。

> **说明** 如果您的VPC资源不在函数计算当前可用区，可以通过在您的VPC环境中创建一个与函数计算相同可用区的交换机，并在函数计算的服务的VPC配置中设置此交换机ID。由于同一VPC内不同交换机之间内网互通，因此函数计算可以通过该交换机访问在其他可用区的VPC内的资源。

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击服务/函数。

4. 在服务列表，单击目标服务名称。
5. 单击**服务配置**，在**服务配置**页签单击**修改**。

6. 在**网络配置**区域，根据实际情况修改网络配置。

参数	操作
允许函数访问公网	是否允许函数访问公网。 ○ **打开开关**：允许函数访问公网。 ○ **关闭开关**：不允许函数访问公网。
允许函数访问VPC内资源	是否允许函数访问VPC内资源。 ○ **打开开关**：允许函数访问VPC内资源。 打开开关后，需要配置以下参数： ■ **专有网络**：在列表中选择要访问的VPC。 ■ **交换机**：在列表中选择交换机，可多选。 ■ **安全组**：在列表中选择ENI所在的安全组。 ○ **关闭开关**：不允许函数访问VPC内资源。
仅允许指定VPC调用函数	是否允许指定的VPC调用函数，详情请参见允许指定的VPC调用函数。

7. 在**权限配置**区域，配置权限，然后单击**点击授权**。
由于函数计算是通过ENI访问VPC中的资源，因此您需要授予访问VPC的服务对ENI的创建、描述以及删除等权限，想了解更多权限介绍请参见权限定义。

参数	操作
角色创建方式	在列表中选择角色创建方式，取值： ○ **选择已有角色**：您已经创建过角色时选择。 ○ **新建角色**：您未创建过角色选择。
已经存在角色	在列表中选择已创建的角色。 当**角色创建方式**选择**选择已有角色**时需要填写。
系统模板授权	在列表中选择**AliyunECSNetworkInterfaceManagementAccess**。 该权限模板包含以下权限： ○ vpc:DescribeVSwitchAttributes ○ ecs:CreateNetworkInterface ○ ecs:DeleteNetworkInterface ○ ecs:DescribeNetworkInterfaces ○ ecs:CreateNetworkInterfacePermission ○ ecs:DescribeNetworkInterfacePermissions

8. 单击**提交**，完成网络及权限配置。

8.挂载NAS

8.1. 配置NAS

阿里云文件存储NAS（Network Attached Storage）是一种分布式的网络文件存储，为ECS、HPC、Docker、BatchCompute等提供安全、无限容量、高性能、高可靠、简单易用的文件存储服务。

前提条件

- 配置函数访问VPC内资源
  NAS目前只能在私有的VPC环境才能添加挂载点，因此您必须确保配置正确的VPC才能访问指定的NAS文件系统。
- 创建通用型NAS文件系统
- 添加挂载点

背景信息

阿里云函数计算支持与NAS无缝集成。这使您的函数可以像访问本地文件系统一样访问存储在其中一个NAS文件系统上的文件。您所要做的是在服务上配置NAS，其中包括NAS的地域、挂载点、分组等信息。配置成功后，该服务下的函数就可以像访问本地文件系统一样访问指定的NAS文件系统。

使用NAS作为函数计算的挂载点的好处如下：

- 可以将临时文件存储到NAS中，临时文件大小不受系统限制。
- 多个函数可以共用一个NAS，实现文件共享。

配置NAS

函数计算的NAS配置是服务级别的，为一个服务配置了NAS挂载点后，那么此服务下的所有函数都可以访问指定NAS文件系统上的文件。

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 在服务列表，单击目标服务名称。
5. 选择**服务配置 > 修改**。

6. 在**NAS文件系统**区域，配置相关参数，然后单击**提交**。

参数	操作	备注
用户	在文本框中填写自定义的用户ID。	详情请参见下文的NAS用户和用户组
用户组	在文本框中填写自定义的组ID。	
NAS挂载点	在列表中选择已创建的NAS挂载的目录。	详情请参见下文的地址配置
远端目录	在文本框中填写远端目录。	
本地目录	在文本框中填写本地目录。	

NAS用户和用户组

在配置函数的NAS挂载时，首先需要配置UserID（用户ID）和GroupID（用户组ID），这两个值等同于文件系统中的用户和组的概念，请根据需求设置文件的拥有者和相应的组权限，确保文件读写权限一致。

UserID和GroupID取值范围从-1到65534，不包括0（为了执行安全，函数计算暂时不提供root用户的方式），其中-1代表系统默认值。UserID和GroupID值配置是可选的，如果不填写UserID，系统会使用-1作为UserID值；如果不填写GroupID， 系统会用UserID值作为Group ID值。

建议您将UserId和GroupId设置为具体的值（1-65534的任意数字），这样该服务下不同函数都可以共享这些文件资源。

> 说明 上传至NAS的文件权限与本地文件权限相同。

地址配置

在NAS配置的第二部分增加挂载点配置（nasMountConfig）。一个服务最多可以挂载5个NAS挂载点。

每个挂载点配置（nasMountConfig）由远程目录（ServerAddr）和本地目录（MountDir）组成。本地目录与远程目录结合，形成了从NAS文件系统中的某个目录到本地文件系统中的一个目录的映射。

- 远程目录（ServerAddr）
  远程目录描述了服务需要访问的NAS文件系统的目录，由挂载点（MountPoint）和绝对目录（absolute directory）两部分组成。挂载点可以通过NAS控制台来添加。将挂载点和绝对目录拼接就可以得到远程目录。例如，如果NAS文件系统的挂载点是*xxxx-nas.aliyuncs.com*，您希望被访问的绝对目录是*/workspace/document*，对应完整的远程目录就是*xxxx-nas.aliyuncs.com:/workspace/document*。
  您可以登录NAS控制台，在文件系统列表中，单击**操作**列的**管理**。然后单击左侧导航栏的**挂载使用**，在挂载点列表中获取挂载点。
- 本地目录（MountDir）
  本地目录是指本地文件系统的挂载点，请不要使用通用的Linux和Unix系统目录，例如*bin*、*opt*、*var*、*dev*等挂载NAS。函数计算允许您使用*mnt*、*home*等非系统目录挂载NAS。

相关文档

- 使用Funcraft配置NAS请参见使用Funcraft配置NAS、配置示例。
- 使用SDK配置NAS请参见SDK列表。

8.2. 访问NAS示例

函数计算的服务配置NAS挂载点后，您可以通过编写代码访问NAS中的文件，就像访问本地文件系统一样。本文提供编写读写NAS文件的函数代码示例。

前提条件

- 配置NAS
- 创建函数

创建写NAS的函数

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 找到目标服务下的目标函数，单击函数名称。
5. 单击**代码执行**页签，在代码编辑器中编写代码。本文以Python 2.7为例，代码示例如下。

```
import json
import logging
import random
import string
import os

def handler(event, context):
  logger = logging.getLogger()
  evt = json.loads(event)
  root_dir = evt["root_dir"]
  sub_dir = randomString(16)
  logger.info('uid : ' + str(os.geteuid()))
  logger.info('gid : ' + str(os.getgid()))
  file_name = randomString(6)+'.txt'
  newDir = root_dir + '/' + sub_dir + '/'
  content = "NAS here I come"
  os.mkdir(newDir)
  fw = open(newDir+file_name, "w+")
  fw.write(content)
  fw.close()
  return sub_dir + '/' + file_name

def randomString(n):
  return ''.join(random.SystemRandom().choice(string.ascii_uppercase + string.digits) for _ in range
(n))
```

其中，通过事件传入的 `root_dir` 是配置NAS时填写的本地挂载路径，详情请参见地址配置。

创建读NAS的函数

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击服务/函数。
4. 找到目标服务下的目标函数，单击函数名称。
5. 单击**代码执行**页签，在代码编辑器中编写代码。本文以Python 2.7为例，代码示例如下。

```
# -*- coding: utf-8 -*-

def handler(event, context):

    f = open("/mnt/test/test.txt", "r")

    print(f.readline())
    f.close()
    return 'ok'
```

该函数的执行结果就是通过写NAS函数写入的内容。

9.预留实例

9.1. 预留实例简介

函数计算为您提供了按量实例和预留实例两种实例资源。本文介绍了两种实例资源的特点，您可以根据实际需要选择不同类型的实例。

按量实例

按量实例是指函数实例的分配和释放完全由函数计算系统负责，有函数调用请求时，函数计算动态调度资源，为您提供弹性可靠的执行环境，极大地降低了管理应用资源的难度。

但是资源的动态调度不可避免地存在冷启动延时，对于时延敏感的在线业务有一定影响。

预留实例

预留实例是将函数实例的分配和释放交由您管理，根据实例的运行时长计费。

预留实例的执行环境是长驻的，可以彻底消除冷启动对业务的影响。

当您预留了实例，函数计算系统收到函数调用请求时，会优先将请求转发给您的预留实例，当函数请求的峰值超过预留实例处理能力时，剩余的部分请求将会转发给您的按量实例，由函数计算系统自动为您分配执行环境。

预留实例的执行时长根据实例的运行时长计费，其执行时长的计量是从函数计算系统启动预留实例开始，到您主动释放为止。因此，即使预留实例未执行任何请求，只要没有释放预留实例，您都需要为预留实例付费。

> **说明** 当您调用API释放实例时，系统保证新的请求不会再路由到该实例上。

具体产品定价和计费请参见计费说明。

9.2. 预留实例操作

本文介绍如何在函数计算控制台创建和修改预留实例。

操作步骤

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 在服务列表区域单击目标服务，选择**预留资源 > 新建预留**。
5. 在**预留实例数**对话框中，将页面拉至底部，配置相关参数，然后单击**确定**。

参数	说明
服务别名	在列表中选择指向需要在预留实例上执行的目标函数的别名。
函数名称	在列表中选择需要在预留实例上执行的目标函数。

参数	说明
预留实例个数	在文本框中填写预留实例的个数。 说明 您可以根据预留实例数对话框中的实例数监控图中的实际使用实例数量设置预留实例个数。

在预留资源列表，您可以看到刚创建的预留实例。

修改预留实例个数

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击服务/函数。
4. 在服务列表，单击目标服务。
5. 单击预留资源页签，找到目标函数。
6. 单击操作列的预留。
7. 在预留实例数对话框，修改预留实例个数，然后单击确定。

说明 如果需要删除预留实例，只需要将预留实例个数设置为0即可。

10.实例并发度管理

10.1. 单实例多并发简介

本文介绍单实例多并发的背景信息、优势、适用场景以及设置多并发的影响。

背景信息

函数计算按实例占用时长计费。假设访问数据库需要10秒，那么当并发的3个请求分别在3个实例内被处理后，3个实例总的执行时长是30秒。如果能让这3个请求在同一个实例内并发处理，这样实例的占用时间为10秒。为了帮助您节省实例资源费用，函数计算支持单实例多并发功能。函数计算允许您为函数设置一个实例并发度（InstanceConcurrency），即单个函数实例可以同时处理多少个请求。下面示意图可以看出单并发和多并发的区别。

假设同时有3个请求需要处理，当实例并发度设置为1时，函数计算需要创建3个实例来处理这3个请求，每个实例分别处理1个请求；当实例并发度设置为10时（即1个实例可以同时处理10个请求），函数计算只需要创建1个实例就能处理这3个请求。

> **说明** 默认情况下，函数的实例并发度为1，也就是一个实例内同时只会处理一个请求。当您设置了实例并发度大于1后，函数计算在弹性伸缩时，会尽可能地充分利用一个实例的并发度后再创建新的实例。

单实例多并发优势

- 减少执行时长，节省费用。
  例如，偏I/O的函数可以在一个实例内并发处理，减少实例数从而减少总的执行时长。
- 请求之间可以共享状态。
  多个请求可以在一个实例内共用数据库连接池，从而减少和数据库之间的连接数。
- 降低冷启动概率。
  由于多个请求可以在一个实例内处理，创建新实例的次数会变少，冷启动概率降低。
- 减少占用VPC IP
  在相同负载下，单实例多并发可以降低总的实例数，从而减少VPC IP的占用。

单实例多并发应用场景

并不是所有的函数都适合开启单实例多并发功能。该功能的适用性如下。

场景	适用性	理由
函数中有较多时间在等待下游服务的响应	适用	等待响应一般不消耗资源，在一个实例内并发处理可以节省费用。
函数中有共享状态且不能并发访问	不适用	例如全局变量，多请求并发执行修改共享状态可能会导致错误。
单个请求的执行要消耗大量CPU及内存资源	不适用	多请求并发执行会造成资源争抢，可能会导致内存不足（OOM）或者延时增加。

设置单实例多并发的影响

设置了单实例多并发（InstanceConcurrency > 1）之后，与单并发（InstanceConcurrency = 1）有以下几个方面的区别：

- 计费
  - 单实例单并发
    函数实例在同一时间只能处理1个请求，1个请求处理完了再处理下一个请求。计费时长从处理第一个请求开始，到最后一个请求结束为止。

    
  - 单实例多并发
    多个请求在一个实例并发处理时，以实例的实际占用时间作为计费的执行时长，即从第一个请求开始，到最后一个请求结束期间的时长。

    

  更多计费详情请参见计费说明。
- 并发度流控
  函数计算一个地域（Region）中按量实例数的上限默认值为300，一个地域可以同时处理的最大请求数为“300 × InstanceConcurrency”。例如，设置InstanceConcurrency = 10时，则一个地域最多允许同时处理3000个并发请求。当并发请求数超过函数计算可以处理的最大请求数时，会收到流控错误（ResourceExhausted）提示。

  > **说明** 如果您想要扩大一个地域的按量实例数上限，请联系我们。
- 日志
  - 在单并发模式下，在调用函数时指定HTTP头 `X-Fc-Log-Type: Tail` ，函数计算会在响应头 `X-Fc-Log-Result` 中包含本次调用所产生的函数日志。在多并发模式下，由于多个请求并发执行，无法获取某个特定请求的日志，响应头中不再包含本次调用的函数日志。

- 针对Node.js Runtime，原来的日志方式是使用 `console.info()` 函数，该方式会把当前请求的Request ID包含在日志内容中。当多请求在同一个实例并发处理时，当前请求可能有很多个，继续使用 `console.info()` 打印日志会导致Request ID错乱，Request ID都会变成 `req 2` 。打印日志示例如下。

  ```
  2019-11-06T14:23:37.587Z req1 [info] logger begin
  2019-11-06T14:23:37.587Z req1 [info] ctxlogger begin
  2019-11-06T14:23:37.587Z req2 [info] logger begin
  2019-11-06T14:23:37.587Z req2 [info] ctxlogger begin
  2019-11-06T14:23:40.587Z req1 [info] ctxlogger end
  2019-11-06T14:23:40.587Z req2 [info] ctxlogger end
  2019-11-06T14:23:37.587Z req2 [info] logger end
  2019-11-06T14:23:37.587Z req2 [info] logger end
  ```

  此时应该使用 `context.logger.info()` 函数打印日志，该方式仍保留了请求的独立Resquest ID。代码示例如下。

  ```
  exports.handler = (event, context, callback) => {
    console.info('logger begin');
    context.logger.info('ctxlogger begin');

    setTimeout(function() {
      context.logger.info('ctxlogger end');
      console.info('logger end');
      callback(null, 'hello world');
    }, 3000);
  };
  ```

- 错误处理
  多个请求在一个实例并发处理时，由于一个请求处理不当导致进程退出或者崩溃，会导致正在并发处理的其他请求也收到错误信息。这要求您在编写函数时，尽量捕获请求级别的异常，不影响其他请求。
  Node.js代码示例如下。

  ```
  exports.handler = (event, context, callback) => {
    try {
      JSON.parse(event);
    } catch (ex) {
      callback(ex);
    }

    callback(null, 'hello world');
  };
  ```

- 共享变量

多个请求在一个实例并发处理时，同时修改一个共享的变量，可能会导致错误。这要求您在编写函数时，对于非线程安全的变量修改要进行互斥保护。Java代码示例如下。

```
public class App implements StreamRequestHandler
{
    private static int counter = 0;

    @Override
    public void handleRequest(InputStream inputStream, OutputStream outputStream, Context conte
xt) throws IOException {
        synchronized (this) {
            counter = counter + 1;
        }
        outputStream.write(new String("hello world").getBytes());
    }
}
```

- 监控指标
  设置函数的实例并发度后，在相同的负载下，可以在控制台的实例数监控图中看到函数的实例数有明显地减少。

  

使用限制

限制项	描述
支持的Runtime	• Node.js Runtime • Java Runtime • Custom Runtime
实例并发度取值范围	1~100
调用响应中的函数日志（X-Fc-Log-Result）	InstanceConcurrency > 1时不支持

更多信息

设置单实例并发度

10.2. 设置单实例并发度

本文介绍如何在函数计算控制台设置单实例并发度。

操作步骤

您可以在创建或更新函数时，指定函数的实例并发度（InstanceConcurrency）。

- 创建函数时配置实例并发度的具体操作步骤请参见创建函数。

  

- 更新函数时配置实例并发度的具体操作步骤请参见更新函数。

如果同时使用了预留实例功能，预留实例也可以并发处理多个请求，详情请参见预留实例简介。

更多信息

使用Node.js SDK设置实例并发度，请参见设置实例并发度。

10.3. 函数级按量实例伸缩控制

本文介绍按量实例伸缩控制的背景信息、应用场景、使用限制、使用说明以及TPS计算公式。

背景信息

为了防止意外的过度调用函数导致费用失控，每个账号在当前地域中按量实例数存在限制，该限制为用户级别，所有函数共享按量实例数的最大限制值。例如，账号123456789在某地域的按量实例数上限为300，该账号下有3个函数function-a、function-b、function-c。在某一时刻所有正在处理调用的函数按量实例数之和最大为300。

除了用户级别的实例数限制，函数计算为函数的调用提供了更细粒度的按量调用实例数限制，您可以通过控制台或API设置函数级别实例限制数来防止单个函数过度调用导致的实例占用，保护后端资源，避免预期外的费用开销。例如，账号123456789下有函数function-a、function-b、function-c。您可以为函数function-a设置按量实例数上限10，调用函数function-a时最多只能占用10个实例。

应用场景

- 保护其他函数的正常并发度。
  例如，有function-a、function-b两个函数共享用户级实例限制数，其中function-a是需要保护的重点业务函数，而function-b有可能被过度调用而影响function-a的正常请求。此时，可以单独为function-b设置实例限制防止function-b抢占大量的按量实例数，使function-a分配不到足够的实例。
- 保护下游服务。
  例如，在函数计算中需要大量访问RDS数据库，由于数据库处理能力有限，您需要保护RDS不被打垮，您可以为访问RDS的函数设置实例限制。
- 禁止某个函数的调用。
  例如，如果发现某个函数调用异常，可以设置最大函数实例数为0，禁止其调用。
- 防止意外的过度调用导致费用失控。
  例如，浏览器端或客户端用户的操作行为不受控制，设置函数级实例数限制可以防止调用失控而产生意外费用。
- 配合预留模式使用。
  通过设置函数级按量实例数限制配合预留实例数达到只用预留实例、只用按量实例或混合使用。

设置函数级按量实例限制后的调用行为

调用类型	调用行为
同步调用	函数调用所需要占用的实例数超过所设置的值后，超出的请求会被拒绝，并收到 `ResourceExhausted` 流控错误。
异步调用	函数调用所需要占用的实例数超过所设置的值后，请求不会被拒绝，请求会在队列里以所有实例满负荷执行的速度逐渐被消费。

更多详情请参见函数调用。

函数级按量实例与预留实例的配合使用

如果您给指定的函数分配了预留实例资源，则优先使用预留实例资源，在预留实例资源用满的情况下，再使用按量实例资源。按量实例资源与预留实例资源配合使用示例如下。

按量实例限制	预留实例限制	结果
0	10	不使用按量实例，只使用预留实例，最多可用10个预留实例。 当当前预留实例不足以支撑并发请求时，新请求会被流控，收到429错误。
20	0	不使用预留实例，只使用按量实例，最多可用20个按量实例。
50	30	优先使用30个预留实例，用满后再使用按量实例，最多使用50个按量实例，总共最多使用80个实例资源。

使用限制

- 每个账号在当前地域下最多设置100条函数级按量实例数限制规则。
- 函数级按量实例数限制规则必须设置在指定别名或LATEST版本之上。
- 每条限制规则的实例限制值不超过账号级实例限制值300。
- 可以针对函数的多个不同别名设置不同的实例限制。

TPS计算公式

TPS指一秒钟内一个函数所能处理的请求数目。您可以结合TPS和您的业务需求，设置函数按量实例数。

TPS的计算公式为：TPS = 1 / DurationInSecond × InstanceConcurrency × MaxInstances

假设一个函数执行平均时长（DurationInSecond）为0.1s，该函数的按量实例数上限（MaxInstances）为5。如果单个实例并发度（InstanceConcurrency）为2，那么这5个函数实例每秒能处理50个（1/0.1× 2 × 5）这样的请求，即TPS为100。

更多信息

设置按量实例伸缩控制

10.4. 设置按量实例伸缩控制

本文介绍如何在函数计算控制台设置函数级按量实例数限制。

操作步骤

1. 登录函数计算控制台。
2. 在顶部菜单栏，选择地域。
3. 在左侧导航栏，单击**服务/函数**。
4. 在服务列表单击目标服务。
5. 单击**按量资源**，在**按量资源**页签单击**配置实例数**。

6. 在**配置按量资源实例**对话框，将页面拉至底部，设置相关参数，然后单击**确定**。

参数	操作
服务版本/别名	在列表中选择指向需要在按量实例上执行的目标函数的别名或LATEST版本。
函数名称	在列表中选择需要在按量实例上执行的目标函数。
最大实例数	在文本框中填写用于执行目标函数的最大按量实例个数。

更多信息

- 使用Java SDK设置按量实例，请参见Example。
- 使用Go SDK设置按量实例，请参见设置按量实例。