ALIBABA CLOUD

阿里云

物联网平台
权限管理

文档版本：20220526

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.使用阿里云账号登录控制台 05
2.RAM授权管理 06
2.1. RAM和STS介绍 06
2.2. 自定义权限 07
2.3. IoT授权映射表 16
2.4. RAM用户访问 21
2.5. 进阶使用STS 23
2.6. 日志导出服务关联角色 28
2.7. 设备文件上传服务关联角色 30
2.8. 数据转发到时序数据库（Lindorm）服务关联角色 32

1.使用阿里云账号登录控制台

阿里云账号具有该账号下所有资源的完全操作权限，并且可以修改账号信息。

登录物联网平台控制台

使用阿里云账号登录物联网控制台，建议您首先完成实名认证，以获取对物联网平台所有操作的完全权限。

1. 访问阿里云官网。
2. 单击**控制台**。
3. 使用阿里云账号和密码登录。

> **说明** 若忘记账号或密码，请单击登录框中**忘记登录名**或**忘记密码**，进入账号或密码找回流程。

4. 在控制台中，单击**产品与服务**，页面显示所有阿里云产品和服务名称。
5. 搜索物联网平台，并单击搜索结果中的**物联网平台**产品名，进入**物联网平台**控制台。

> **说明** 如果您还没有开通物联网平台服务，**物联网平台**主页会展示相关提示，您只需单击**立即开通**，便可快速开通物联网平台服务。

进入**物联网平台**控制台后，您便可以进行产品管理、设备管理、规则管理等操作。

创建访问控制

因为阿里云账号具有账号的完全权限，阿里云账号泄露会带来极严重的安全隐患。因此，若需要授权其他人访问您的阿里云资源，请勿将您的阿里云账号及密码直接泄露出去。

您可通过RAM访问控制的控制台创建RAM用户，并为RAM用户授予需要的访问权限。非账号所有者或管理员的其他人，可通过RAM用户访问资源。RAM用户访问的配置方法，请参见RAM用户访问和自定义权限。

2.RAM授权管理

本章节将详细介绍物联网平台账号权限控制相关事宜。

2.1. RAM和STS介绍

RAM和STS是阿里云提供的权限管理系统。

了解RAM和STS的详情，请参见访问控制产品帮助文档。

RAM的主要作用是控制账号系统的权限。通过使用RAM，创建、管理RAM用户，并通过给RAM用户授予不同的权限，控制RAM用户对资源的操作权限。

STS是一个安全凭证（Token）的管理系统，为阿里云主账号下的RAM用户提供短期访问权限管理。通过STS来完成对临时用户的访问授权。

背景介绍

RAM和STS解决的一个核心问题是如何在不暴露主账号的AccessKey的情况下，安全地授权他人访问。因为一旦主账号的AccessKey被泄露，会带来极大的安全风险：获得该账号AccessKey的人可任意操作该账号下所有的资源，盗取重要信息等。

RAM提供的是一种长期有效的权限控制机制。通过创建RAM用户，并授予RAM用户相应的权限，将不同的权限分给不同的用户。RAM用户的 AccessKey也不能泄露。即使RAM用户泄露也不会造成全局的信息泄露。一般情况下，RAM用户长期有效。

相对于RAM提供的长效控制机制，STS提供的是一种临时访问授权。通过调用STS，获得临时的AccessKey和Token。可以将临时AccessKey和Token发给临时用户，用来访问相应的资源。从STS获取的权限会受到更加严格的限制，并且具有时间限制。因此，即使出现信息泄露的情况，影响相对较小。

使用场景示例，请参见使用示例。

基本概念

使用RAM和STS涉及以下基本概念：

- RAM用户：在RAM控制台中，创建的用户，每个用户即一个RAM用户。创建时或创建成功后，均可为RAM用户生成独立的AccessKey。创建后，需为RAM用户配置密码和权限。使用RAM用户，可以进行已获授权的操作。RAM用户可以理解为具有某种权限的用户，可以被认为是一个具有某些权限的操作发起者。
- 角色（Role）：表示某种操作权限的虚拟概念，但是没有独立的登录密码和AccessKey。RAM用户可以扮演角色。扮演角色时，RAM用户拥有的权限是该角色的权限。
- 授权策略（Policy）：用来定义权限的规则，如允许RAM用户读取或者写入某些资源。
- 资源（Resource）：代表RAM用户可访问的云资源，如表格存储所有的Instance、某个Instance或者某个Instance下面的某个Table等。

RAM用户和角色可以类比为个人和其身份的关系。如，某人在公司的角色是员工，在家里的角色是父亲。同一人在不同的场景扮演不同的角色。在扮演不同角色的时候，拥有对应角色的权限。角色本身并不是一个操作的实体，只有用户扮演了该角色之后才是一个完整的操作实体。并且，一个角色可以被多个不同的用户同时扮演。

使用示例

为避免阿里云账号的AccessKey泄露而导致安全风险，某阿里云账号管理员使用RAM创建了两个RAM用户，分别命名为A和B，并为A和B生成独立的AccessKey。A拥有读权限，B拥有写权限。 管理员可以随时在RAM控制台取消RAM用户的权限。

现在因为某些原因，需要授权给其他人临时访问物联网平台接口的权限。这种情况下，不能直接把A的AccessKey透露出去，而应该新建一个角色C，并给这个角色授予读取物联网平台接口的权限。但请注意，目前角色C还无法直接使用。因为并不存在对应角色C的AccessKey，角色C仅是一个拥有访问物联网平台接口权限的虚拟实体。

需调用STS的AssumeRole接口，获取访问物联网平台接口的临时授权。在调用STS的请求中，RoleArn的值需为角色C的Arn。如果调用成功，STS会返回临时的AccessKeyId、AccessKeySecret和SecurityToken作为访问凭证（凭证的过期时间，在调用AssumeRole的请求中指定）。将这个凭证发给需要访问的用户，该用户就可以获得访问物联网平台接口的临时权限。

为什么RAM和STS的使用这么复杂？

虽然RAM和STS的概念和使用比较复杂，但这是为了账号的安全性和权限控制的灵活性而牺牲了部分易用性。

将RAM用户和角色分开，主要是为了将执行操作的实体和代表权限集合的虚拟实体分开。如果某用户需要使用多种权限，如读/写权限，但是实际上每次操作只需要其中的一部分权限，那么就可以创建两个角色。这两个角色分别具有读或写权限。然后，创建一个可以扮演这两个角色的RAM用户。当用户需要读权限的时候，就可以扮演其中拥有读权限的角色；使用写权限的时候同理。这样可以降低每次操作中权限泄露的风险。而且，通过扮演角色，可以将角色权限授予其他用户，更加方便了协同使用。

STS对权限的控制更加灵活。如按照实际需求设置有效时长。但是，如果需要一个长期有效的临时访问凭证，则可以只适用RAM用户管理功能，而无需使用STS。

在后面的章节中，我们将提供一些RAM和STS的使用指南和使用示例。如果您需要了解更多RAM和STS的代码详情，请参见RAM API和STS API。

2.2. 自定义权限

权限指在特定条件下，允许（Allow）或拒绝（Deny）对指定资源执行的相关操作。

操作步骤

权限的载体是授权策略。自定义权限，即在自定义授权策略时定义某些权限。

1. 登录访问控制RAM控制台。
2. 在左侧导航栏，选择**权限管理 > 权限策略**。
3. 在**权限策略**页面，单击**创建权限策略**。
4. 在**创建权限策略**页面，单击**脚本编辑**页签。
5. 输入权限策略内容，然后单击**下一步**。

   JSON格式的授权策略详情。需包含以下参数：

   - Action：表示要授权的操作。IoT操作都以 `iot:` 开头。定义方式和示例，请参见本文档中Action定义。
   - Effect： 表示授权类型，取值：*Allow*（允许）、*Deny*（拒绝）。
   - Resource： 表示要授权的资源。

     如果为RAM用户授予访问您的所有物联网平台资源的权限，取值为 `*` 。
   - Condition： 表示鉴权条件，请参见本文档中Condition定义。
6. 输入权限策略**名称**和**备注**，然后单击**确定**。

Action 定义

Action是API的名称。在创建IoT授权策略时，每个Action前缀均为 `iot:` ，多个Action以英文逗号（,）分隔，支持使用星号（*）通配符。IoT API名称定义，请参见IoT授权映射表。

下面介绍一些典型的Action定义示例。

- 定义单个API。

```
"Action": "iot:CreateProduct"
```

- 定义多个API。

```
"Action": [
"iot:UpdateProduct",
"iot:QueryProduct"
]
```

- 定义所有只读API，包含规则引擎数据流转目标产品的权限。

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": [
        "iot:Query*",
        "iot:List*",
        "iot:Get*",
        "iot:BatchGet*",
        "iot:Check*"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "rds:DescribeDBInstances",
        "rds:DescribeDatabases",
        "rds:DescribeAccounts",
        "rds:DescribeDBInstanceNetInfo"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": "ram:ListRoles",
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "mns:ListTopic",
        "mns:GetTopicRef"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
```

```
      "Action": [
        "dhs:ListProject",
        "dhs:GetProject",
        "dhs:ListTopic",
        "dhs:GetTopic"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "ots:ListInstance",
        "ots:GetInstance",
        "ots:ListTable",
        "ots:DescribeTable"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "ons:OnsRegionList",
        "ons:OnsInstanceInServiceList",
        "ons:OnsTopicList",
        "ons:OnsTopicGet"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "hitsdb:DescribeRegions",
        "hitsdb:DescribeHiTSDBInstanceList",
        "hitsdb:DescribeHiTSDBInstance"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "fc:ListServices",
        "fc:GetService",
        "fc:GetFunction",
        "fc:ListFunctions"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "log:ListShards",
        "log:ListLogStores",
        "log:ListProject"
      ],
```

```
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "cms:QueryMetricList"
      ],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

- 定义所有读写API，包含规则引擎数据流转目标产品的权限。

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": "iot:*",
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "rds:DescribeDBInstances",
        "rds:DescribeDatabases",
        "rds:DescribeAccounts",
        "rds:DescribeDBInstanceNetInfo",
        "rds:ModifySecurityIps"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": "ram:ListRoles",
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "mns:ListTopic",
        "mns:GetTopicRef"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "dhs:ListProject",
        "dhs:ListTopic",
        "dhs:GetProject",
        "dhs:GetTopic"
      ],
      "Resource": "*",
```

```
      "Effect": "Allow"
    },
    {
      "Action": [
        "ots:ListInstance",
        "ots:ListTable",
        "ots:DescribeTable",
        "ots:GetInstance"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "ons:OnsRegionList",
        "ons:OnsInstanceInServiceList",
        "ons:OnsTopicList",
        "ons:OnsTopicGet"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "hitsdb:DescribeRegions",
        "hitsdb:DescribeHiTSDBInstanceList",
        "hitsdb:DescribeHiTSDBInstance",
        "hitsdb:ModifyHiTSDBInstanceSecurityIpList"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "fc:ListServices",
        "fc:GetService",
        "fc:GetFunction",
        "fc:ListFunctions"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "log:ListShards",
        "log:ListLogStores",
        "log:ListProject"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": "ram:PassRole",
      "Resource": "*",
```

```
      "Effect": "Allow",
      "Condition": {
        "StringEquals": {
          "acs:Service": "iot.aliyuncs.com"
        }
      }
    },
    {
      "Action": [
        "cms:QueryMetricList"
      ],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

Condition定义

目前RAM用户授权策略支持访问IP限制、是否通过HTTPS访问、是否通过MFA（多因素认证）访问、访问时间限制等多种鉴权条件。物联网平台的所有API均支持上述条件。

- 访问IP限制。

  访问控制可以限制访问IoT的源IP地址，并且支持根据网段进行过滤。以下是典型的使用场景示例。

  - 限制单个IP地址和IP网段。例如，只允许IP地址为192.0.2.1或198.51.100.0/24网段的请求访问。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:*",
      "Resource": "*",
      "Condition": {
        "IpAddress": {
          "acs:SourceIp": [
            "192.0.2.1",
            "198.51.100.0/24"
          ]
        }
      }
    }
  ],
  "Version": "1"
}
```

- 限制多个IP地址。例如，只允许IP地址为192.0.2.1和198.51.100.1的请求访问。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:*",
      "Resource": "*",
      "Condition": {
        "IpAddress": {
          "acs:SourceIp": [
            "192.0.2.1",
            "198.51.100.1"
          ]
        }
      }
    }
  ],
  "Version": "1"
}
```

- HTTPS访问限制。

  访问控制可以限制是否通过HTTPS访问。

  示例：限制必须通过HTTPS请求访问。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:*",
      "Resource": "*",
      "Condition": {
        "Bool": {
          "acs:SecureTransport": "true"
        }
      }
    }
  ],
  "Version": "1"
}
```

- MFA访问限制。

  访问控制可以限制是否通过MFA（多因素认证）访问。MFA访问适用于控制台登录，使用API访问无需MFA码。

  示例：限制必须通过MFA请求访问。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:*",
      "Resource": "*",
      "Condition": {
        "Bool": {
          "acs:MFAPresent ": "true"
        }
      }
    }
  ],
  "Version": "1"
}
```

- 访问时间限制。

  访问控制可以限制请求的访问时间，即只允许或拒绝在某个时间点范围之前的请求。

  示例：用户可以在北京时间2019年01月01日凌晨之前访问，之后则不能访问。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:*",
      "Resource": "*",
      "Condition": {
        "DateLessThan": {
          "acs:CurrentTime": "2019-01-01T00:00:00+08:00"
        }
      }
    }
  ],
  "Version": "1"
}
```

典型使用场景

结合以上对Action、Resource和Condition的定义，下面介绍一些典型使用场景的授权策略定义和授权方法。

- 允许访问的授权策略示例。

  场景：定义访问IP地址为192.0.2.1/24网段的用户访问IoT的权限，且要求只能在2019-01-01 00:00:00之前访问和通过HTTPS访问。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:*",
      "Resource": "*",
      "Condition": {
        "IpAddress": {
          "acs:SourceIp": [
            "192.0.2.1/24"
          ]
        },
        "DateLessThan": {
          "acs:CurrentTime": "2019-01-01T00:00:00+08:00"
        },
        "Bool": {
          "acs:SecureTransport": "true"
        }
      }
    }
  ],
  "Version": "1"
}
```

- 拒绝访问的授权策略示例。

  场景：拒绝访问IP地址为198.51.100.1的用户对IoT执行读操作。

```
{
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "iot:Query*",
        "iot:List*",
        "iot:Get*",
        "iot:BatchGet*"
      ],
      "Resource": "*",
      "Condition": {
        "IpAddress": {
          "acs:SourceIp": [
            "198.51.100.1"
          ]
        }
      }
    }
  ],
  "Version": "1"
}
```

授权策略创建成功后，将此权限授予RAM用户。获得授权的RAM用户就可以进行权限中定义的操作。创建RAM用户和授权操作帮助，请参见RAM用户访问。

2.3. IoT授权映射表

定义授权策略，为RAM用户授予具体API或操作的访问权限。

为RAM用户授权的具体方法，请参见自定义权限。

下表中列举了您在创建物联网平台相关授权策略时，参数Action的可选值。

> **说明** 下表列举了部分API授权映射表，所有API的授权操作统一为 `iot:${API名称}` ，其中 `${API名称}` 为调用的具体API名称。

API名称或操作	RAM 授权操作（Action）	资源（Resource）	接口说明
CreateProduct	iot:CreateProduct	*	创建产品。
UpdateProduct	iot:UpdateProduct	*	修改产品。
QueryProduct	iot:QueryProduct	*	查询产品信息。
QueryProductList	iot:QueryProductList	*	查询产品列表。
DeleteProduct	iot:DeleteProduct	*	删除产品。
CreateProductTags	iot:CreateProductTags	*	创建产品标签。
UpdateProductTags	iot:UpdateProductTags	*	更新产品标签。
DeleteProductTags	iot:DeleteProductTags	*	删除产品标签。
ListProductTags	iot:ListProductTags	*	查询产品标签。
ListProductByTags	iot:ListProductByTags	*	根据标签查询产品。
RegisterDevice	iot:RegisterDevice	*	注册设备。
QueryDevice	iot:QueryDevice	*	查询指定产品下的所有设备列表。
DeleteDevice	iot:DeleteDevice	*	删除设备。
QueryPageByApplyId	iot:QueryPageByApplyId	*	查询批量注册的设备信息。
BatchGetDeviceState	iot:BatchGetDeviceState	*	批量获取设备状态。
BatchRegisterDeviceWithApplyId	iot:BatchRegisterDeviceWithApplyId	*	根据ApplyId批量申请设备。
BatchRegisterDevice	iot:BatchRegisterDevice	*	批量注册设备（随机生成设备名）。
QueryBatchRegisterDeviceStatus	iot:QueryBatchRegisterDeviceStatus	*	查询批量注册设备的处理状态和结果。

API名称或操作	RAM 授权操作（Action）	资源（Resource）	接口说明
BatchCheckDeviceNames	iot:BatchCheckDeviceNames	*	批量自定义设备名称。
QueryDeviceStatistics	iot:QueryDeviceStatistics	*	获取设备的统计数量。
QueryDeviceEventData	iot:QueryDeviceEventData	*	获取设备的事件历史数据。
QueryDeviceServiceData	iot:QueryDeviceServiceData	*	获取设备的服务记录历史数据。
SetDeviceProperty	iot:SetDeviceProperty	*	设置设备的属性。
SetDevicesProperty	iot:SetDevicesProperty	*	批量设置设备属性。
InvokeThingService	iot:InvokeThingService	*	调用设备的服务。
InvokeThingsService	iot:InvokeThingsService	*	批量调用设备服务。
QueryDevicePropertyStatus	iot:QueryDevicePropertyStatus	*	查询设备的属性快照。
QueryDeviceDetail	iot:QueryDeviceDetail	*	查询设备详情。
DisableThing	iot:DisableThing	*	禁用设备。
EnableThing	iot:EnableThing	*	解除设备的禁用状态。
ResetThing	iot:ResetThing	*	重置设备。
GetThingTopo	iot:GetThingTopo	*	查询设备拓扑关系。
RemoveThingTopo	iot:RemoveThingTopo	*	移除设备拓扑关系。
NotifyAddThingTopo	iot:NotifyAddThingTopo	*	通知云端增加设备拓扑关系。
QueryDevicePropertyData	iot:QueryDevicePropertyData	*	获取设备的属性历史数据。
QueryDevicePropertiesData	iot:QueryDevicePropertiesData	*	批量查询指定设备的属性上报数据。
GetGatewayBySubDevice	iot:GetGatewayBySubDevice	*	根据挂载的子设备信息查询对应的网关设备信息。
SaveDeviceProp	iot:SaveDeviceProp	*	为指定设备设置标签。
QueryDeviceProp	iot:QueryDeviceProp	*	查询指定设备的标签列表。
DeleteDeviceProp	iot:DeleteDeviceProp	*	删除设备标签。
QueryDeviceByTags	iot:QueryDeviceByTags	*	根据标签查询设备。
CreateDeviceGroup	iot:CreateDeviceGroup	*	创建分组。

API名称或操作	RAM 授权操作（Action）	资源（Resource）	接口说明
UpdateDeviceGroup	iot:UpdateDeviceGroup	*	更新分组信息。
DeleteDeviceGroup	iot:DeleteDeviceGroup	*	删除分组。
BatchAddDeviceGroupRelations	iot:BatchAddDeviceGroupRelations	*	添加设备到分组。
BatchDeleteDeviceGroupRelations	iot:BatchDeleteDeviceGroupRelations	*	将设备从分组中删除。
QueryDeviceGroupInfo	iot:QueryDeviceGroupInfo	*	查询分组详情。
QueryDeviceGroupList	iot:QueryDeviceGroupList	*	查询分组列表。
SetDeviceGroupTags	iot:SetDeviceGroupTags	*	添加或更新分组标签。
QueryDeviceGroupTagList	iot:QueryDeviceGroupTagList	*	查询分组标签列表。
QueryDeviceGroupByDevice	iot:QueryDeviceGroupByDevice	*	查询指定设备所在的分组列表。
QueryDeviceListByDeviceGroup	iot:QueryDeviceListByDeviceGroup	*	查询分组中的设备列表。
QuerySuperDeviceGroup	iot:QuerySuperDeviceGroup	*	根据子分组ID查询父分组信息。
QueryDeviceGroupByTags	iot:QueryDeviceGroupByTags	*	根据标签查询设备分组。
StartRule	iot:StartRule	*	启动规则。
StopRule	iot:StopRule	*	暂停规则。
ListRule	iot:ListRule	*	查询规则列表。
GetRule	iot:GetRule	*	查询规则详情。
CreateRule	iot:CreateRule	*	创建规则。
UpdateRule	iot:UpdateRule	*	修改规则。
DeleteRule	iot:DeleteRule	*	删除规则。
CreateRuleAction	iot:CreateRuleAction	*	创建规则中的数据转发方法。
UpdateRuleAction	iot:UpdateRuleAction	*	修改规则中的数据转发方法。
DeleteRuleAction	iot:DeleteRuleAction	*	删除规则中的数据转发方法。
GetRuleAction	iot:GetRuleAction	*	查询规则中的数据转发方法的详细信息。

API名称或操作	RAM 授权操作（Action）	资源（Resource）	接口说明
ListRuleActions	iot:ListRuleActions	*	获取规则中的数据转发方法列表。
Pub	iot:Pub	*	发布消息。
PubBroadcast	iot:PubBroadcast	*	向订阅了指定产品广播Topic的所有设备发送消息。
RRpc	iot:RRpc	*	发送消息给设备并得到设备响应。
CreateProductTopic	iot:CreateProductTopic	*	创建产品Topic类。
DeleteProductTopic	iot:DeleteProductTopic	*	删除产品Topic类。
QueryProductTopic	iot:QueryProductTopic	*	查询产品Topic类列表。
UpdateProductTopic	iot:UpdateProductTopic	*	修改产品Topic类。
CreateTopicRouteTable	iot:CreateTopicRouteTable	*	新建Topic间的消息路由关系。
DeleteTopicRouteTable	iot:DeleteTopicRouteTable	*	删除Topic路由关系。
QueryTopicReverseRouteTable	iot:QueryTopicReverseRouteTable	*	查询指定Topic订阅的源Topic。
QueryTopicRouteTable	iot:QueryTopicRouteTable	*	查询向指定Topic订阅消息的目标Topic。
GetDeviceShadow	iot:GetDeviceShadow	*	查询设备的影子信息。
UpdateDeviceShadow	iot:UpdateDeviceShadow	*	修改设备的影子信息。
SetDeviceDesiredProperty	iot:SetDeviceDesiredProperty	*	为指定设备批量设置期望属性值。
QueryDeviceDesiredProperty	iot:QueryDeviceDesiredProperty	*	查询指定设备的期望属性值。
BatchUpdateDeviceNickname	iot:BatchUpdateDeviceNickname	*	批量更新设备备注名称。
QueryDeviceFileList	iot:QueryDeviceFileList	*	查询指定设备上传到物联网平台的所有文件列表。
QueryDeviceFile	iot:QueryDeviceFile	*	查询指定设备上传到物联网平台的指定文件信息。
DeleteDeviceFile	iot:DeleteDeviceFile	*	删除指定设备上传到物联网平台的指定文件。
QueryLoRaJoinPermissions	iot:QueryLoRaJoinPermissions	*	查询LoRaWAN入网凭证列表。

API名称或操作	RAM 授权操作（Action）	资源（Resource）	接口说明
CreateLoRaNodesTask	iot:CreateLoRaNodesTask	*	生成批量注册LoRaWAN设备的任务。
GetLoraNodesTask	iot:GetLoraNodesTask	*	查询批量注册LoRaWAN设备任务的状态。
QueryDeviceCert	iot:QueryDeviceCert	*	查询单个设备的X.509证书。
QueryCertUrlByApplyId	iot:QueryCertUrlByApplyId	*	查询批量注册设备的X.509证书下载链接。
BatchAddThingTopo	iot:BatchAddThingTopo	*	批量添加设备拓扑关系。
QueryDeviceByStatus	iot:QueryDeviceByStatus	*	根据设备状态查询设备列表。
GenerateOTAUploadURL	iot:GenerateOTAUploadURL	*	生成固件文件的上传对象存储的信息。
CreateOTAFirmware	iot:CreateOTAFirmware	*	创建固件。
DeleteOTAFirmware	iot:DeleteOTAFirmware	*	删除指定固件。
ListOTAFirmware	iot:ListOTAFirmware	*	查询固件列表。
QueryOTAFirmware	iot:QueryOTAFirmware	*	查询指定固件的详细信息。
CreateOTAVerifyJob	iot:CreateOTAVerifyJob	*	创建固件验证批次。
CreateOTAStaticUpgradeJob	iot:CreateOTAStaticUpgradeJob	*	创建静态批量升级批次。
CreateOTADynamicUpgradeJob	iot:CreateOTADynamicUpgradeJob	*	创建动态升级批次。
ListOTAJobByFirmware	iot:ListOTAJobByFirmware	*	获取固件下的升级批次列表。
ListOTAJobByDevice	iot:ListOTAJobByDevice	*	获取设备所在的固件升级批次列表。
QueryOTAJob	iot:QueryOTAJob	*	查询指定升级批次的详情。
CancelOTAStrategyByJob	iot:CancelOTAStrategyByJob	*	取消动态升级批次所关联的动态升级策略。
CancelOTATaskByDevice	iot:CancelOTATaskByDevice	*	取消指定固件下状态为待升级的设备升级作业。
CancelOTATaskByJob	iot:CancelOTATaskByJob	*	取消指定批次下的设备升级作业。

API名称或操作	RAM 授权操作（Action）	资源（Resource）	接口说明
ListOTATaskByJob	iot:ListOTATaskByJob	*	查询指定升级批次下的设备升级作业列表。
CreateSubscribeRelation	iot:CreateSubscribeRelation	*	创建MNS或AMQP服务端订阅。
UpdateSubscribeRelation	iot:UpdateSubscribeRelation	*	修改MNS或AMQP服务端订阅。
QuerySubscribeRelation	iot:QuerySubscribeRelation	*	查询MNS或AMQP服务端订阅。
DeleteSubscribeRelation	iot:DeleteSubscribeRelation	*	删除MNS或AMQP服务端订阅。
CreateConsumerGroup	iotCreateConsumerGroup	*	创建一个消费组，用于创建AMQP服务端订阅。
UpdateConsumerGroup	iot:UpdateConsumerGroup	*	修改消费组名称。
QueryConsumerGroupByGroupId	iot:QueryConsumerGroupByGroupId	*	根据消费组ID查询消费组详情。
QueryConsumerGroupList	iot:QueryConsumerGroupList	*	查询用户所有消费组列表，或按消费组名称进行模糊查询。
QueryConsumerGroupStatus	iot:QueryConsumerGroupStatus	*	使用AMQP服务端订阅时，查询某个消费组的状态，包括在线客户端信息、消息消费速率、消息堆积数、最近消息消费时间。
ResetConsumerGroupPosition	iot:ResetConsumerGroupPosition	*	使用AMQP服务端订阅时，清空某个消费组的堆积消息。
DeleteConsumerGroup	iot:DeleteConsumerGroup	*	删除消费组。
CreateConsumerGroupSubscribeRelation	iot:CreateConsumerGroupSubscribeRelation	*	在AMQP订阅中添加一个消费组。
DeleteConsumerGroupSubscribeRelation	iot:DeleteConsumerGroupSubscribeRelation	*	从AMQP订阅中的多个消费组移除指定消费组。
配置AMQP服务端订阅	iot:sub	*	支持AMQP服务端订阅连接。

2.4. RAM用户访问

用户可以使用RAM用户访问物联网平台资源。本文介绍如何创建RAM用户，如何授予RAM用户访问物联网平台资源的权限，和RAM用户如何登录物联网平台控制台。

背景信息

您需先创建RAM用户，并通过授权策略授予RAM用户访问物联网平台的权限。创建自定义授权策略的方法，请参见自定义权限。

创建RAM用户

如果您已有RAM用户，请忽略此操作。

1. 用主账号登录访问控制RAM控制台。
2. 在左侧导航栏**身份管理**菜单下，单击**用户**。
3. 单击**创建用户**。
4. 输入**登录名称**和**显示名称**。
5. 在**访问方式**区域下，选择**控制台访问**或**Open API调用访问**，并设置具体的登录信息。

   > **说明** 为了保障账号安全，建议仅为RAM用户选择一种登录方式，避免RAM用户离开组织后仍可以通过访问密钥访问阿里云资源。

6. 单击**确定**。
7. 身份验证。阿里云可能会进行用户身份验证，并向您的账号预留的联系手机号中发送验证码。请将收到的验证码填入验证对话框中。

RAM用户创建完成后，RAM用户便可通过RAM用户登录链接登录阿里云官网和控制台。RAM用户的登录地址，请在**访问控制RAM控制台**的**概览**页面查看。

但是，在获得授权之前，该RAM用户无法访问您的阿里云资源。您需为RAM用户授予物联网平台的访问权限。

授权RAM用户访问物联网平台

在**访问控制RAM控制台**中，您可以在**用户**页，为单个RAM用户进行授权；也可以在**用户组**页，为整个群组授予相同的权限。下面我们以为单个RAM用户授权为例，介绍授权操作流程。

1. 用主账号登录访问控制 RAM 控制台。
2. 在左侧导航栏**身份管理**菜单下，单击**用户**。
3. 勾选要授权的RAM用户，单击下方**添加权限**。
4. 在授权对话框中，选中您要授予该RAM用户的物联网平台授权策略，再单击**确定**。

   > **说明** 如果您要为RAM用户授予自定义权限，请先创建授权策略。授权策略的创建方法，请参见自定义权限。

授权成功后，RAM用户便可访问授权策略中定义的资源，和进行授权策略中定义的操作。

RAM用户登录控制台

阿里云主账号登录是从阿里云官网主页直接登录，但是RAM用户需从**RAM用户登录**页登录。

1. 获取**RAM用户登录**页的链接地址。

   用主账号登录**访问控制RAM控制台**，在**概览**页面的**账号管理**区域下，查看用户登录地址，并将链接地址分发给RAM用户。

2. RAM用户访问**RAM用户登录**页面进行登录。

   RAM用户登录方式有以下三种：

   - 方式一：`<$username>@<$AccountAlias>.onaliyun.com`。例如：`username@company-alias.onaliyun.com`。

> **说明** RAM用户登录账号为UPN（User Principal Name）格式，即RAM控制台用户列表中所见的用户登录名称。<$username>为RAM用户名称，<$AccountAlias>.onaliyun.com为默认域名。

- 方式二：`<$username>@<$AccountAlias>`。例如：`username@company-alias`。

  > **说明** <$username>为RAM用户名称，<$AccountAlias>为账号别名。

- 方式三：如果创建了域别名，也可以使用域别名登录，格式为：`<$username>@<$DomainAlias>`。

  > **说明** <$username>为RAM用户名称，<$DomainAlias>为域别名。

3. 单击页面右上角**控制台**按钮，进入**管理控制台**。
4. 单击**产品与服务**，选择**物联网平台**，即可进入**物联网平台控制台**。

RAM用户登录**物联网平台控制台**后，便可在控制台中，进行已获授权的操作。

2.5. 进阶使用STS

STS权限管理系统是比访问控制（RAM）更为严格的权限管理系统。使用STS权限管理系统进行资源访问控制，需通过复杂的授权流程，授予子账号用户临时访问资源的权限。

背景信息

子账号和授予子账号的权限均长期有效。删除子账号或解除子账号权限，均需手动操作。发生子账号信息泄露后，如果无法及时删除该子账号或解除权限，可能给您的阿里云资源和重要信息带来危险。所以，对于关键性权限或子账号无需长期使用的权限，您可以通过STS权限管理系统来进行控制。

子账号获得临时访问权限的操作流程

步骤一：创建角色

RAM角色是一种虚拟用户，是承载操作权限的虚拟概念。

1. 使用阿里云主账号登录访问控制 RAM 控制台。
2. 单击**角色** > **新建角色**，进入角色创建流程。
3. 选择可信实体类型为**阿里云账号**，单击**下一步**。
4. 输入角色名称和备注，选择云账号为**当前云账号**或**其他云账号**，单击**完成**。

> **说明** 若选择**其他云账号**，需要填写其他云账号的ID。

步骤二：创建角色授权策略

角色授权策略，即定义要授予角色的资源访问权限。

1. 在访问控制 RAM 控制台左侧导航栏，单击**权限管理** > **权限策略**。

2. 单击**创建权限策略**。
3. 在**创建权限策略**页面，单击**脚本编辑**页签。
4. 输入策略内容，单击**下一步**。

   如果策略模式选择为**脚本配置**，授权策略内容的编写方法请参见语法结构。

   IoT资源只读权限的授权策略内容示例如下：

```
{
    "Version":"1",
    "Statement":[
        {
            "Action":[
                "rds:DescribeDBInstances",
                "rds:DescribeDatabases",
                "rds:DescribeAccounts",
                "rds:DescribeDBInstanceNetInfo"
            ],
            "Resource":"*",
            "Effect":"Allow"
        },
        {
            "Action":"ram:ListRoles",
            "Effect":"Allow",
            "Resource":"*"
        },
        {
            "Action":[
                "mns:ListTopic"
            ],
            "Resource":"*",
            "Effect":"Allow"
        },
        {
            "Action":[
                "dhs:ListProject",
                "dhs:ListTopic",
                "dhs:GetTopic"
            ],
            "Resource":"*",
            "Effect":"Allow"
        },
        {
            "Action":[
                "ots:ListInstance",
                "ots:ListTable",
                "ots:DescribeTable"
            ],
            "Resource":"*",
            "Effect":"Allow"
        },
        {
            "Action":[
                "log:ListShards",
                "log:ListLogStores"
```

```
                "log:ListLogStores",
                "log:ListProject"
            ],
            "Resource":"*",
            "Effect":"Allow"
        },
        {
            "Effect":"Allow",
            "Action":[
                "iot:Query*",
                "iot:List*",
                "iot:Get*",
                "iot:BatchGet*"
            ],
            "Resource":"*"
        }
    ]
}
```

IoT资源读写权限的授权策略内容示例如下：

```
{
    "Version":"1",
    "Statement":[
        {
            "Action":[
                "rds:DescribeDBInstances",
                "rds:DescribeDatabases",
                "rds:DescribeAccounts",
                "rds:DescribeDBInstanceNetInfo"
            ],
            "Resource":"*",
            "Effect":"Allow"
        },
        {
            "Action":"ram:ListRoles",
            "Effect":"Allow",
            "Resource":"*"
        },
        {
            "Action":[
                "mns:ListTopic"
            ],
            "Resource":"*",
            "Effect":"Allow"
        },
        {
            "Action":[
                "dhs:ListProject",
                "dhs:ListTopic",
                "dhs:GetTopic"
            ],
            "Resource":"*",
            "Effect":"Allow"
        },
```

```
        {
            "Action":[
                "ots:ListInstance",
                "ots:ListTable",
                "ots:DescribeTable"
            ],
            "Resource":"*",
            "Effect":"Allow"
        },
        {
            "Action":[
                "log:ListShards",
                "log:ListLogStores",
                "log:ListProject"
            ],
            "Resource":"*",
            "Effect":"Allow"
        },
        {
            "Effect":"Allow",
            "Action":"iot:*",
            "Resource":"*"
        }
    ]
}
```

5. 输入权限策略**名称**和**备注**，然后单击**确定**。

授权策略创建成功后，您就可以将该授权策略中定义的权限授予角色。

步骤三：为角色授权

角色获得授权后，才具有资源访问权限。您可以在**RAM角色管理**页，单击角色对应的**添加权限**按钮，为单个角色授权。同时为多个角色授权，请参见以下步骤。

1. 在访问控制 RAM 控制台页左侧导航栏，单击**权限管理 > 授权**。
2. 单击**新增授权**。
3. 在授权对话框中，**被授权主体**下，输入RAM角色名称，选中要授权的策略，再单击**确定**。

下一步，为子账号授予可以扮演该角色的权限。

步骤四：授予子账号角色扮演权限

虽然经过授权后，该角色已拥有了授权策略定义的访问权限，但角色本身只是虚拟用户，需要子账号用户扮演该角色，才能进行权限允许的操作。若任意子账号都可以扮演该角色，也会带来风险，因此只有获得角色扮演权限的子账号用户才能扮演角色。

授权子账号扮演角色的方法：先新建一个Resource参数值为角色ID的自定义授权策略，然后用该授权策略为子账号授权。

1. 在访问控制 RAM 控制台左侧导航栏，单击**权限管理 > 权限策略**。
2. 单击**新建权限策略**。
3. 在**创建权限策略**页面，单击**脚本编辑**页签。
4. 输入策略内容，单击**下一步**。

> **说明** 授权策略内容中，参数Resource 的值需为角色Arn。在**RAM角色管理**页面，单击角色名称，进入**基本信息**页，查看角色的Arn 。

角色授权策略示例：

```
{
    "Version":"1",
    "Statement":[
        {
            "Effect":"Allow",
            "Action":"iot:QueryProduct",
            "Resource":"角色Arn"
        }
    ]
}
```

5. 输入权限策略**名称**和**备注**，然后单击**确定**。
6. 授权策略创建成功后，返回**RAM访问控制**主页面。
7. 单击左侧导航栏中的**身份管理 > 用户**。
8. 在子账号列表中，勾选要授权的子账号，并单击下方的**添加权限**按钮。
9. 在授权对话框中，选中刚新建的角色授权策略，再单击**确定**。

授权完成后，子账号便有了可以扮演该角色的权限，就可以使用STS获取扮演角色的临时身份凭证，和进行资源访问。

步骤五：子账号获取临时身份凭证

获得角色授权的子账号用户，可以通过直接调用API或使SDK 来获取扮演角色的临时身份凭证：AccessKeyId、AccessKeySecret和SecurityToken。STS API和STS SDK详情，请参见访问控制文档中STS API和STS SDK。

使用API和SDK获取扮演角色的临时身份凭证需传入以下参数：

- RoleArn：需要扮演的角色Arn。
- RoleSessionName：临时凭证的名称（自定义参数）。
- Policy：授权策略，即为角色增加一个权限限制。通过此参数限制生成的Token的权限。不指定此参数，则返回的Token将拥有指定角色的所有权限。
- DurationSeconds：临时凭证的有效期。单位是秒，最小为900，最大为3600，默认值是3600。
- id 和secret：指需要扮演该角色的子账号的AccessKeyId和AccessKeySecret。

获取临时身份凭证示例

API示例：子账号用户通过调用STS的AssumeRole接口获得扮演该角色的临时身份凭证。

```
https://sts.aliyuncs.com?Action=AssumeRole
&RoleArn=acs:ram::1234567890123456:role/iotstsrole
&RoleSessionName=iotreadonlyrole
&DurationSeconds=3600
&Policy=<url_encoded_policy>
&<公共请求参数>
```

SDK示例：子账号用户使用STS的Python命令行工具接口获得扮演该角色的临时身份凭证。

```
$python ./sts.py AssumeRole RoleArn=acs:ram::1234567890123456:role/iotstsrole RoleSessionNa
me=iotreadonlyrole Policy='{"Version":"1","Statement":[{"Effect":"Allow","Action":"iot:*","
Resource":"*"}]}' DurationSeconds=3600 --id=id --secret=secret
```

请求成功后，将返回扮演该角色的临时身份凭证：AccessKeyId、AccessKeySecret和SecurityToken。

步骤六：子账号临时访问资源

获得扮演角色的临时身份凭证后，子账号用户便可以在调用SDK的请求中传入该临时身份凭证信息，扮演角色。

Java SDK示例：子账号用户在调用请求中，传入临时身份凭证的AccessKeyId、AccessKeySecret和SecurityToken参数，创建IAcsClient对象。

```
IClientProfile  profile = DefaultProfile.getProfile("cn-hangzhou", AccessKeyId,AccessSecret
);
RpcAcsRequest request.putQueryParameter("SecurityToken", Token);
IAcsClient client = new DefaultAcsClient(profile);
AcsResponse response = client.getAcsResponse(request);
```

2.6. 日志导出服务关联角色

下面介绍物联网平台日志导出服务关联角色（AliyunServiceRoleForIoTLogExport），以及如何删除该角色。

角色说明

物联网平台提供日志转储功能，可以将云端运行日志导出到您指定的阿里云日志服务实例Logstore中。为了实现该功能，需要获取日志服务的访问权限，因此，开通该功能时，系统自动创建相应的RAM角色：物联网平台日志导出服务关联角色（AliyunServiceRoleForIoTLogExport）。

角色名称：

```
AliyunServiceRoleForIoTLogExport
```

角色权限策略：

```
AliyunServiceRolePolicyForIoTLogExport
```

权限说明：

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": [
        "log:PostLogStoreLogs",
        "log:CreateProject",
        "log:GetLogStoreLogs",
        "log:CreateLogStore",
        "log:GetLogStore",
        "log:ListLogStores",
        "log:UpdateLogStore",
        "log:CreateConfig",
        "log:UpdateConfig",
        "log:GetConfig",
        "log:CreateIndex",
        "log:GetIndex",
        "log:UpdateIndex",
        "log:CreateSavedSearch",
        "log:UpdateSavedSearch",
        "log:GetSavedSearch",
        "log:DeleteSavedSearch",
        "log:ListSavedSearch",
        "log:CreateDashboard",
        "log:UpdateDashboard",
        "log:GetDashboard",
        "log:DeleteDashboard",
        "log:ListDashboard"
      ],
      "Resource": "acs:log:*:*:project/iot-log-*",
      "Effect": "Allow"
    },
    {
      "Action": "ram:DeleteServiceLinkedRole",
      "Resource": "*",
      "Effect": "Allow",
      "Condition": {
        "StringEquals": {
          "ram:ServiceName": "log-export.iot.aliyuncs.com"
        }
      }
    }
  ]
}
```

删除角色

当您不再需要AliyunServiceRoleForIoTLogExport角色时，可以将角色删除。

1. 逐一停止所有产品的日志转储功能。停止单个产品的日志转储功能，请参见停止云端运行日志转储。
2. 删除角色，请参见删除服务关联角色。

2.7. 设备文件上传服务关联角色

下面介绍物联网平台设备文件上传服务关联角色（AliyunServiceRoleForIoTDeviceFileUpload），以及如何删除该角色。

背景信息

设备文件上传配置的具体操作，请参见上传文件配置。

角色说明

物联网平台提供文件上传功能，可以让设备将文件上传到您指定的OSS Bucket中。为了实现该功能，需要获取您的OSS空间的访问权限，因此，开通该功能时，系统自动创建相应的RAM角色：物联网平台设备文件上传服务关联角色（AliyunServiceRoleForIoTDeviceFileUpload）。

- 角色名称：`AliyunServiceRoleForIoTDeviceFileUpload`。
- 角色权限策略：`AliyunServiceRolePolicyForIoTDeviceFileUpload`。
- 策略内容：

> **说明** 权限策略元素说明，请参见权限策略基本元素。

```
{
    "Version": "1",
    "Statement": [
        {
            "Action": "oss:GetBucketInfo",
            "Resource": "acs:oss:*:*:*",
            "Effect": "Allow"
        },
        {
            "Action": "oss:GetObject",
            "Resource": "acs:oss:*:*:*/aliyun-iot-device-file/*",
            "Effect": "Allow"
        },
        {
            "Action": "oss:DeleteObject",
            "Resource": "acs:oss:*:*:*/aliyun-iot-device-file/*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "oss:PutObject",
                "oss:PutObjectTagging"
            ],
            "Resource": "acs:oss:*:*:*/aliyun-iot-device-file/*",
            "Effect": "Allow"
        },
        {
            "Action": "ram:DeleteServiceLinkedRole",
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                "ram:ServiceName": "device-file-upload.iot.aliyuncs.com"
                }
            }
        }
    ]
}
```

删除角色

若您不再需要将设备文件保存至您的OSS Bucket时，可通过以下操作，删除设备文件上传服务关联角色 `AliyunServiceRoleForIoTDeviceFileUpload` 。

1. 在设备端修改文件上传逻辑，不再将文件上传至您自己的OSS空间中。
2. 在物联网平台控制台删除Bucket授权配置。
   i. 在**产品详情**页面，单击**上传文件配置**页签，找到目标Bucket配置。
   ii. 在目标OSS Bucket右侧**操作**栏，单击**删除**。

iii. 在弹出的对话框中，确认Bucket信息，单击**确认**。

3. 在RAM控制台，删除服务关联角色AliyunServiceRoleForIoTDeviceFileUpload。具体操作，请参见删除服务关联角色。

2.8. 数据转发到时序数据库（Lindorm）服务关联角色

本文介绍物联网平台云产品流转服务中数据转发到时序数据库（Lindorm）服务关联角色（AliyunServiceRoleForIoTRuleengineLindorm），以及如何删除该角色。

背景信息

服务关联角色SLR（Service Linked Role）的详细说明，请参见服务关联角色。

角色说明

物联网平台提供设备数据流转到时序数据库（Lindorm）功能，可以将设备上报数据转发到Lindorm实例下指定的数据表中。为了实现该功能，需要获取时序数据库（Lindorm）的访问权限。

物联网平台支持SLR的自动创建。您可在配置数据流转到时序数据库（Lindorm）时，通过授权操作完成SLR的创建。具体操作，请参见数据转发到时序数据库（Lindorm）。

角色名称：

```
AliyunServiceRoleForIoTRuleengineLindorm
```

角色权限策略：

```
AliyunServiceRolePolicyForIoTRuleengineLindorm
```

权限说明：

```
{
    "Version": "1",
    "Statement": [{
            "Action": [
                "lindorm:GetLindormInstanceList",
                "lindorm:GetLindormInstance",
                "lindorm:GetLindormInstanceEngineList",
                "lindorm:GetInstanceIpWhiteList",
                "lindorm:UpdateInstanceIpWhiteList"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": "ram:DeleteServiceLinkedRole",
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                "ram:ServiceName": "ruleengine-lindorm.iot.aliyuncs.com"
                }
            }
        }
    ]
}
```

删除角色

当您不再需要AliyunServiceRoleForIoT RuleengineLindorm角色时，可以将角色删除。

1. 逐一删除物联网平台云产品流转服务中已配置的流转到时序数据库（Lindorm）的数据目的。
2. 删除角色，请参见删除服务关联角色。