目录

术语说明

术语	全称	中文	说明
Account	Member Account	账户（账号）	指一个金山云账户，账户是资源的拥有者，任何资源均隶属于某个账户
User	IAM User	IAM用户	指一个金山云账户下的IAM用户(又称:子用户)，子用户不拥有资源，能够操作主账户授予权限的资源
Role	IAM Role	角色	指一个金山云账户下创建的角色
Group(IAM)	IAM Group	用户群组	金山云多个相同职能的用户（IAM用户的集合）
AccountId	Ksyun Account Id	金山云账户ID	指一个金山云账户的ID，为最长20位的数字标识
Policy	Policy	访问策略	访问策略选项大致可分为基于资源的策略和用户策略两类 • 基于资源的策略 ： 附加到资源（存储桶和对象）的访问策略,如空间策略和访问控制列表 (ACL) • 用户策略 ： 访问策略附加到账户中的用户
ACL	Access Control List	访问控制列表	KS3使用ACL（访问控制列表）方便用户管理Bucket和Object的访问权限，每个Bucket和Object都拥有一个ACL
Bucket Policy	Bucket Policy	空间策略	可以通过添加空间策略向其他主账户或IAM子用户授予对相应存储空间（Bucket）及其中文件（Object）的权限。存储桶策略相比ACL更加灵活，授权的粒度更细
User Policy	User Policy	用户策略	可以在账户中创建IAM子用户、用户组和角色，并通过附加访问策略授予它们对包括KS3在内的所有金山云资源的访问权限
KRN	Kingsoft Resource Name	金山云资源名称	指金山云全局唯一的资源标识名称，用于在策略文档中标识资源，形如krn:ksc::::/。例如：krn:ksc:iam::(11233):user/peter，表示11233账户下的peter子用户
AK	AccessKeyId	访问密钥ID	指一个访问密钥的唯一标识，标识访问者的身份
SK	SecretAccessKey	私有访问密钥	用于加密签名字符串和KS3用来验证签名字符串的密钥，需要用户保管好，不能泄露。
STS	Security Token Service	安全凭证（Token）管理系统	用来授予临时的访问权限

权限概述

默认情况下，所有 KS3 资源都是私有的，包括存储空间（Bucket）、对象(Object)和相关子资源（例如：Lifecycle 配置和 ACL 配置）。只有资源拥有者，即创建该资源的 KS3 账户可以访问该资源。存储空间的创建者也具有该存储空间下所有资源的访问权限。

资源拥有者可以选择通过编写**访问策略**授予他人访问权限。授予访问权限，指的是资源拥有者可以决定哪些资源、对什么人、在何种条件下、授予执行什么操作的权限。因此描述一个访问策略，通常包括四个元素：资源、被授权人、条件(非必需)、操作。下面依次说明访问策略的各个元素和具体的访问策略类型。

访问策略的元素

KS3资源

存储空间和对象/文件是KS3的主要资源，他们都有相关联的资源。 存储空间的子资源包括：

- Bucket Policy - 空间策略
- ACL - 存储空间访问控制列表
- Lifecycle - 存储生命周期配置信息
- CORS（跨域资源共享） - 存储空间跨域请求配置
- Logging - KS3的存储空间访问日志

对象/文件的子资源包括：

- ACL - 存储对象访问权限列表。

被授权人

- **账户（主账户）**

账户或称主账户是客户在金山云资源归属、资源计量、资源计费的主体。任何客户在使用金山云的服务前，都需要首先注册生成一个金山云账户，一般使用用户名作为账户的登录标识。

账户是其名下所有云计算资源的所有者，拥有名下全部资源的控制权限，拥有资源的订单、账单；云计算资源可被所属账户随意操作访问。

- **IAM用户（子用户）**

IAM用户是账户下的授权实体，也是归属于账户的一种资源。IAM用户不拥有任何云计算资源、不能独立计量和计费，只能被主账户授权管理其名下的各种资源，其所管理的资源归属于主账户（由主账户付费），且没有独立的账单。

IAM用户在获得主账户的授权后，能够被设置密码和访问密钥，从而登录控制台和使用openAPI管理主账户的资源。

- **角色（Role）**

IAM角色是一种虚拟用户（或影子账户），它是IAM用户类型的一种。这种虚拟用户有确定的身份，也可以被赋予一组权限(Policy)，但它没有确定的身份认证密钥（登录密码或AccessKey）。与普通IAM用户的差别主要在使用方法上，IAM角色需要被一个授信的实体用户扮演，扮演成功后实体用户将获得IAM角色的临时安全令牌，使用这个临时安全令牌就能以角色身份访问被授权的资源。

- **关于资源归属的说明**

默认情况下，所有KS3资源都是私有的，只有资源拥有者才能访问资源，资源拥有者是指创建资源的KS3账户。

- 创建存储空间(Bucket)和对象/文件（Object）的 KS3 账户拥有这些资源的所有权限
- IAM子用户不拥有任何资源，如果一个IAM子用户再被授权后，上传了一个对象/文件，那么该IAM用户所属的父账户拥有此对象/文件。
- 存储空间拥有者可以向其他KS3账户授予上传对象的跨账户权限。在这种情况下，上传对象的KS3账户拥有这些对象。存储空间拥有者对其他账户拥有的对象也同时拥有所有权限，账单也是由存储空间拥有者支付。

条件

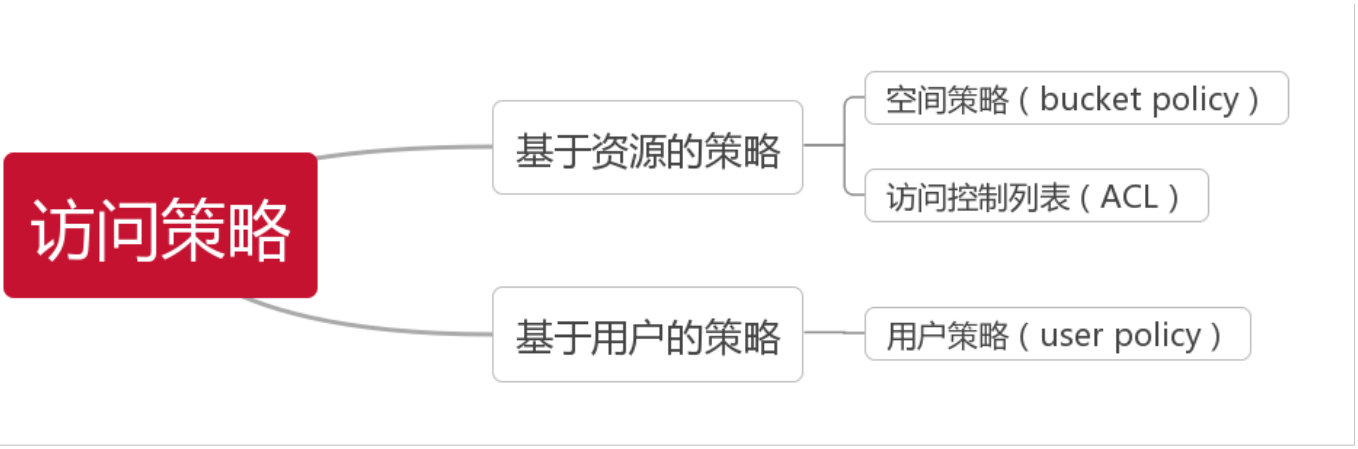
在授予权限时指定的条件。访问请求只有在满足指定条件时，访问策略才可以生效。KS支持指定IP地址、指定请求中带有的请求头等条件设置。

资源操作

KS3 提供一系列针对KS3资源的API操作，详情请见API概览。

访问策略

访问策略分为基于资源的策略和用户策略两类。附加到资源（存储空间和文件）的访问策略称为基于资源的策略，例如，存储空间策略（bucket policy）和访问控制列表（ACL）就是基于资源的策略；访问策略附加到您账户中的用户，这些策略称为用户策略（user policy）。



- ACL访问策略： 每一个存储空间和对象都关联一个ACL。ACL可以向其他的KS3账户授予基本的读写权限，ACL使用XML格式来表示，以下的例子指定存储空间的拥有者授予另外一个用户Read权限。

```
<AccessControlPolicy>
<Owner>
 <ID>Owner-User-Id</ID>
 <DisplayName>Owner-User-Name</DisplayName>
</Owner>
<AccessControlList>
  <Grant>
    <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
       <ID>User2-Id</ID>
       <DisplayName>User2-Name</DisplayName>
   </Grantee>
   <Permission>READ</Permission>
 </Grant>
...
</AccessControlList>
</AccessControlPolicy>
```

- 空间策略（Bucket Policy）： 用户可以在存储空间设置空间策略，向其他KS3账户或者IAM子用户（包含本账户下的子用户和其他账户下的子用户）授予存储空间以及其中文件/对象的权限。空间策略可以很好补充ACL访问策略，授权的权限更多。空间策略使用JSON文件来表示。下面为一个空间策略的例子，允许账户ID为11123的用户可以对mybucket和mybucket下所有文件做任何操作。

```
{
"Version":"2008-10-17",
"Statement":[{
   "Sid":"1",
   "Effect":"Allow",
```

```
        "Principal":{"KSC":["krn:ksc:iam:::11123:root"]},
        "Action":["ks3:*"],
        "Resource":["krn:ksc:ks3:::mybucket","krn:ksc:ks3:::mybucket/*"]
    }]
    }
```

- 用户策略（User Policy）：

用户可以使用 身份与访问控制中心（Identity and Access Mutual，简称:IAM）管理对 KS3 资源的访问权 限。使用 IAM，用户可以在主账户中创建 子用户（IAM 用户）、组和角色，并通过附加访问策略授予它们对KS3资源的访问权限。

```
{
    "Version":"2015-11-01",
    "Statement":[{
    "Effect":"Allow",
    "Action":["ks3:*"],
    "Resource":["krn:ksc:ks3:::mybucket","krn:ksc:ks3:::mybucket/*"]
    }]
}
```

ACL（访问控制列表）

KS3使用ACL（Access Control List : 访问控制列表）方便用户管理存储空间（Bucket）和文件（Object)的访问权限，每个Bucket和Object都拥有一个ACL，可以通过设置ACL定义向哪些用户授予哪些访问权限。

当KS3收到针对某个资源的请求后，将检查相应的ACL以验证请求者是否拥有所需的访问权限。

ACL使用XML格式来表示，示例：

```
<AccessControlPolicy>
    <Owner>
       <ID>Owner-User-Id</ID>
       <DisplayName>Owner-User-Name</DisplayName>
    </Owner>
    <AccessControlList>
      <Grant>
        <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
          <ID>User2-Id</ID>
          <DisplayName>User2-Name</DisplayName>
        </Grantee>
        <Permission>READ</Permission>
     </Grant>
...
    </AccessControlList>
</AccessControlPolicy>
```

示例中Owner元素中需要设置资源拥有者的账户ID和用户名，Grant元素中需要设置被授权人的账户ID和用户名，以及所授予的权限。下面依次对被授权人、授予的权限、预设ACL的方式以及KS3支持操作ACL的方式进行说明。

被授权者

被授权者可以是KS3账户也可以是所有用户（包括匿名用户），但不能是IAM子用户。

- **KS3账户**：在授权语句中，使用账户ID和用户名来表示被授权的KS3账户。

  ```
  <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser"><ID>ID</ID><DisplayName>GranteesEmail</DisplayNam
  e>
  </Grantee>
  ```

- **所有用户（包含匿名用户）**：在授权语句中，使用http://acs.ksyun.com/groups/global/AllUsers表示授权所有用户。

  ```
  <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group"><URI>http://acs.ksyun.com/groups/global/AllUsers</URI></Gr
  antee>
  ```

授予的权限

下表列出了KS3在ACL中支持的权限集。

ACL权限	对存储空间授权	对文件授权
READ	允许被授权者列出存储空间中的文件	允许被授权者读取文件数据及其元数据
WRITE	允许被授权者创建、覆盖和删除存储空间（Bucket）中的任意文件	不适用
FULL_CONTROL	授予被授权者在存储空间（Bucket）上的READ、WRITE 权限	授予被授权者在文件上的READ权限

> **注意**：1、对于文件ACL和存储空间ACL，ACL权限集相同，但这些ACL权限授予允许被授权者进行的具体操作不同，例如存储空间ACL的READ权限允许被授权者列出存储空间中的文件，而文件ACL的权限则是允许被授权者读取文件数据及其元数据。
>
> 2、文件ACL不支持WRITE权限。

ACL的每个权限允许一个或多个 KS3 操作，例如：文件的读权限，包含了Head Object和GET Object等KS3操作。 下表展示了每个ACL权限允许的具体操作。ACL 主要用于授予基本读/写权限，这与文件系统权限类似。

ACL权限	在Bucket授予ACL权限时允许的操作	在Object授予ACL权限时允许的操作
READ	List Bucket,List Multipart Upload	Get Object,Head Object, List Parts
WRITE	Put Object, Post Object, Put Object Copy, Upload Part Copy,Delete Object,Initiate Multipart Upload, Upload Part, Complete Multipart Upload, Abort Multipart Upload	不适用
FULL_CONTROL	等同于授予READ和WRITE ACL权限	等同于授予READ ACL权限。

预设ACL权限

KS3支持在创建桶或上传对象时通过头域设置桶或对象的权限控制策略,这种方式设置的权限称为预设ACL。 使用预设 ACL 时，需要在 PUT Bucket/Object 或 PUT Bucket/Object acl 中携带相应的头部，这些头部中x-kss-acl只能对所有用户进行授权，x-kss-grant-read、x-kss-grant-write和x-kss-grant-full-control可以更细粒度地对指定用户授权。

- **存储空间预设ACL**

存储空间预设ACL可选择的头部有4个，描述和示例见下表：

头域	描述	示例
x-kss-acl	设置任何用户（包括匿名用户）的操作权限，取值包括：private、public-read、public-read-write，默认值为private： • private：私有，空间所有者拥有所有的访问权限。其他人没有访问权限（默认）； • public-read：公开读，空间所有者拥有所有的访问权限，任何用户（包括匿名用户）都拥有列举空间下所有文件的权限； • public-read-write：公开读写，任何用户（包括匿名用户）都可以向存储空间写入文件、删除文件、列举空间下所有文件的权限；	指定任何用户（包括匿名用户）都拥有列举空间下所有文件的权限： • x-kss-acl:public-read
x-kss-grant-read	为若干用户授予READ权限	给id为1234578和3344211的两个用户授予READ权限: • x-kss-grant-read:id=“1234578”,id=“3344211”
x-kss-grant-write	为若干用户授予WRITE权限	给id为1234578和3344211的两个用户授予WRITE权限: • x-kss-grant-write:id=“1234578”,id=“3344211”
x-kss-grant-full-control	为若干用户授予FULL_CONTROL权限	给id为1234578和3344211的两个用户授予FULL_CONTROL权限: • x-kss-grant-full-control:id=“1234578”,id=“3344211”

- **文件预设ACL**

文件预设ACL可选择的头部有3个，描述和示例见下表：

头域	描述	示例
x-kss-acl	设置任何用户（包括匿名用户）的操作权限，取值包括：private、public-read，默认值为private： • private：私有，文件所有者拥有所有的访问权限。其他人没有访问权限（默认）； • public-read：公开读，所有用户（包括匿名用户）拥有文件的访问权限；	指定任何用户（包括匿名用户）都拥有文件的访问权限： • x-kss-acl:public-read
x-kss-grant-read	为若干用户授予READ权限	给id为1234578和3344211的两个用户授予READ权限: • x-kss-grant-read:id=“1234578”,id=“3344211”
xx-kss-grant-full-control	为若干用户授予FULL_CONTROL权限	给id为1234578和3344211的两个用户授予FULL_CONTROL权限: • x-kss-grant-full-control:id=“1234578”,id=“3344211”

操作方式

支持控制台和API两种操作方式：	操作方式	参考文档
控制台	• 控制台空间管理-权限设置 • 控制台内容管理-文件管理	
API	用户在创建存储空间或上传对象时：用户可以在请求头中通过设置以下的请求头来指定ACL，详情请见： • PUT Bucket API • PUT Object API 用户在已有资源上设置ACL：可以在请求头或者在请求内容中中设置,详情请见： • PUT Bucket ACL API • PUT Object ACL API	
SDK	• JAVA • PHP • Python • Android • IOS • JavaScript • Node.js • C# • Go	

空间策略（Bucket Policy）

空间策略作用于所配置的KS3空间及空间内对象,包含以下几个元素：

- **效果（Effect）**：代表本条策略的效果，可以是允许或拒绝，对应Effect的值为Allow或者Deny。该元素是必填项。
- **资源（Resource）**：代表本条策略针对哪些资源起作用，可以设置为存储空间(Bucket)、对象(Object)以及相对应的子资源。该元素是必填项。
- **被授权人（Principal）**：代表授权给哪些用户和账户。该元素是必填项。
- **权限（Action）**：代表对于指定的资源授权了哪些权限。该元素是必填项。
- **条件（Condition）**：在授予权限时指定的条件。访问请求只有在满足指定条件时，访问策略才可以生效。该元素是非必填项。

下面是一条空间策略示例,此策略允许账户accountid下的子用户Dave在ip为101.226.XXX.185的条件下对examplebucket存储桶具有 ks3:listbucket 和 ks3:getobject 的权限。

```
{
    "Version":"2015-11-01",
    "Statement": [
    {
        "Sid":"1",
        "Effect":"Allow",
        "Principal":{krn:ksc:iam::(accountid):user/Dave},
        "Action":[
            "ks3:ListBucket",
            "ks3:GetObject"
         ],
        "Resource":["krn:ksc:ks3:::examplebucket","krn:ksc:ks3:::examplebucket/*"],
    "Condition": {
            "IpAddress": {
                "ksc:SourceIp": ["101.226.XXX.185"]
            }
         }
   }
  ]
}
```

下面依次对空间策略包含的元素的配置方式进行说明。

指定效果

包含允许（Allow）和拒绝（Deny）两种授权效果，允许（Allow）代表允许设定的用户对指定资源的访问权限，拒绝（Deny）可确保设定的用户无法访问指定资源，即使有其他策略授予了访问权限的情况下也是如此。

指定资源

在KS3中，使用资源名称（KRN）来标识资源，KRN的格式如下：

```
krn:ksc:ks3:::bucketname
krn:ksc:ks3:::bucketname/keyname
```

其中bucketname表示存储空间名称，keyname表示对象/文件（Object）的名称，支持通配符*（匹配多个字符）和？（匹配单个字符）。

以下为 KS3 资源KRN的示例：

- 表示 mybucket 存储空间

```
krn:ksc:ks3:::mybucket
```

- 表示 mybucket存储空间中全部对象

```
krn:ksc:ks3:::mybucket/*
```

- 表示您拥有的所有存储空间与对象

```
krn:ksc:ks3:::*
```

注意：

- 在实际使用中，为了简化输入，在控制台写入Resource时，只需要写入空间名称和文件名称，可省略前面的krn:ksc:ks3:::。后台程序会自动将简化输入转换为标准输入。
- Bucket与Object级别的操作需要对应与之相匹配的资源表示，详见指定权限。

指定被授权人

Principal 元素用于指定被允许或拒绝访问资源的子用户、账户、服务或其他实体。元素 Principal 仅在空间策略中起作用；用户策略中不必指定，因为用户策略直接附加到特定用户。下面是指定 Principal 的示例。

```
1、*表示所有用户。
2、krn:ksc:iam::(accountid):root //标识主账号，acountid是主账户id，root是固定值。
3、krn:ksc:iam::(acountid):user/(userName) userName表示子账号，其中user是固定值，acountid是归属主账户id。
4、krn:ksc:iam::(acountid):role/(ruleNAME) ruleNAME表示角色名称，其中role是定值，acountid是归属主账户id。
```

注意：在实际使用中，为了简化输入，在控制台写入Principal时，对于主账户只需要填入主账户ID即可；对于子用户，写入格式为“acountid/userName”，其中accountid为用户主账号ID，userName为子用户名。后台程序会自动将简化输入转换为标准输入。

指定权限

KS3在策略中指定的一组权限。每一个权限，都会映射到特定 KS3 操作。

权限关键字	对应的KS3操作	操作级别
ks3:ListBucket	列举bucket下的文件和查询bucket信息	Bucket 级别
ks3:DeleteBucket	删除当前Bucket	Bucket 级别
ks3:GetBucketAcl	获取Bucket的ACL信息	Bucket 级别
ks3:PutBucketAcl	设置Bucket的ACL信息	Bucket 级别
ks3:GetBucketCORS	获取bucket的CORS配置信息	Bucket 级别
ks3:PutBucketCORS	设置Bucket的CORS	Bucket 级别
ks3:ListBucketMultipartUploads	列出分段上传	Bucket 级别
ks3:PutObject	上传文件，包含post、PUT和分块上传	Object 级别
ks3:DeleteObject	删除文件	Object 级别
ks3:GetObject	GET Object 和 HEAD Object	Object 级别
ks3:GetObjectAcl	获取文件的ACL信息	Object 级别
ks3:PutObjectAcl	设置文件的ACL	Object 级别
ks3:ListMultipartUploadParts	列出分段	Object 级别
ks3:AbortMultipartUpload	取消分块上传	Object 级别
ks3:PostObjectRestore	解冻归档存储对象	Object 级别
ks3:PutObjectTagging	添加/更新对象的标签	Object 级别
ks3:GetObjectTagging	查询对象的标签	Object 级别
ks3:DeleteObjectTagging	删除对象的标签	Object 级别

注意： 不同级别的操作需要指定与之对等的资源。 1.如您想授权ks3:ListBucket操作则其对应的资源必须为bucket， 例如：krn:ksc:ks3:::bucket01代表名为bucket01的存储空间； 2.如果您想授权ks3:PutObject操作，则需要指定文件资源，例如:krn:ksc:ks3:::bucket01/*,代表bucket01中的所有文件； 3.需要同时授权Bucket与Object级别权限需要同时指定两种对应的资源。

指定条件

访问策略语言支持在授予权限时指定该策略生效的条件。在Condition 元素（或 Condition 块）中，可以指定策略生效的条件。Condition 元素是可选的。

如果在策略中指定了条件，那么用户的请求必须满足条件，策略才可以生效。

KS3支持的Condition如下：

Condition	功能
ksc:SourceIp	指定IP地址

ksc:RequestHeader 指定请求中带有的请求头
ksc:SubnetID 指定请求来自SubnetID对应的VPC子网

IP地址（ksc:SourceIp）支持的条件运算符如下：

条件运算符	value取值范围	说明
IpAddress	严格的IP地址格式和CIDR格式，只支持IPV4	客户请求的源IP地址是value中指定的IP地址或范围内的地址，策略生效
NotIpAddress	严格的IP地址格式和CIDR格式，只支持IPV4	客户的请求的源IP地址是指定IP地址和范围外的IP地址，策略生效

请求头（ksc:RequestHeader）支持的条件运算符如下：

条件运算符	value取值范围	说明
StringEquals	键值对形式的字符串，如"x-kss-cdn:kingsoftcdn"	请求中带有指定header ，并且请求头的value值恰好匹配（区分大小写），策略生效
StringNotEquals	键值对形式的字符串，如"x-kss-cdn:kingsoftcdn"	请求中带有指定header ，并且请求头的value值不匹配（区分大小写），策略生效
StringEqualsIgnoreCase	键值对形式的字符串，如"x-kss-cdn:kingsoftcdn"	请求中带有指定header ，并且请求头的value值恰好匹配（不区分大小写），策略生效
StringNotEqualsIgnoreCase	键值对形式的字符串，如"x-kss-cdn:kingsoftcdn"	请求中带有指定header ，并且请求头的value值不匹配（不区分大小写），策略生效
StringLike	键值对形式的字符串，可包括字符串中任何一个多字符匹配的通配符 (*) 或单字符匹配的通配符 (?)，如"x-kss-cdn:*"	KS3的请求中带有指定header ，并且请求头的value值可模糊匹配（区分大小写），策略生效
StringNotLike	键值对形式的字符串，可包括字符串中任何一个多字符匹配的通配符 (*) 或单字符匹配的通配符 (?)，如"x-kss-cdn:*"	请求中带有指定header ，并且请求头的value值不模糊匹配（不区分大小写），策略生效

VPC子网ID（ksc:SubnetID）支持的条件运算符如下：

条件运算符	value取值范围	说明
StringEquals	严格的AccountID格式和SubnetID格式	客户的请求来自value中指定的SubnetID对应的VPC子网，策略生效
StringNotEquals	严格的AccountID格式和SubnetID格式	客户的请求不是来自value中指定的VPC子网，策略生效

操作方式

支持控制台和API两种操作方式：	操作方式	参考文档
控制台	空间策略	
API	• Put Bucket Policy • Get Bucket Policy • Delete Bucket Policy	
SDK	• Python • Android	

用户策略(User Policy)

用户策略（User Policy）是基于用户的授权策略。通过设置用户策略，您可以集中管理您的用户，以及控制用户可以访问您名下哪些资源的权限，用户策略包含以下几个元素：

- **Effect**：当用户请求特定操作（可以是允许或拒绝）时的效果。如果没有显式授予（允许）对资源的访问权限，则说明此条策略不生效。也可显式拒绝对资源的访问，这样可确保用户无法访问该资源，即使有其他策略授予了访问权限的情况下也是如此。
- **资源**：存储空间(Bucket)、对象(Object)以及相对应的子资源。
- **操作**：对于每个资源，所支持的一组操作。
- **指定条件**：在授予权限时指定的条件。访问请求只有在满足指定条件时，访问策略才可以生效。

下面是一条用户策略示例。此策略允许对 examplebucket 存储桶具有ks3:listbucket和ks3:getobject的权限。创建完此策略后，将策略授权给子用户，子用户就拥有了相对应的权限。其他示例请参阅用户权限示例。

```
{
    "Version":"2015-11-01",
    "Statement": [
    {
        "Sid":"1",
        "Effect":"Allow",
        "Action":[
            "ks3:ListBucket",
            "ks3:GetObject"
        ],
        "Resource":["krn:ksc:ks3:::examplebucket","krn:ksc:ks3:::examplebucket/*"]
    }
   ]
}
```

注意：和空间策略(Bucket policy)不同，用户策略中不必指定被授权人，因为用户策略直接附加到特定用户。

在策略中指定资源

在KS3中，使用资源名称（KRN）来标识资源，KRN的格式如下：

```
krn:ksc:ks3:::bucketname
krn:ksc:ks3:::bucketname/keyname
```

其中bucketname表示存储空间名称，keyname表示对象/文件（Object）的名称，支持通配符*（匹配多个字符）和？（匹配单个字符）。

以下为 KS3 资源KRN的示例：

- 该KRN表示 mybucket 存储桶

```
krn:ksc:ks3:::mybucket
```

在策略中指定被授权人

用户策略中不必指定，因为用户策略直接附加到特定用户。如何将自定义策略授权给IAM子用户或用户组或角色，请参照访问控制。

在策略中指定权限

KS3在策略中指定的一组权限。每一个权限，都会映射到特定 KS3 操作。权限分为Service级别操作、Bucket级别操作以及Object级别的操作。各API接口功能描述的更多信息，请参见API概览。

Service级别

Service级别操作对应的是ks3:ListBuckets，用来列举所有属于该用户的Bucket列表。

权限关键字	对应的KS3操作
ks3:ListBuckets	查询Bucket列表

> 注意： ListBuckets是服务级别的操作，与Bucket级别ListBucket不同，ListBuckets能查看到所有的Bucket列表，但不能看到Bucket中具体的Object内容，而ListBucket可以看到Bucket下的Object信息。

Bucket级别

Bucket级别操作的对象均为Bucket,权限关键字与对应的操作如下: 权限关键字	对应的KS3操作
ks3:GetBucketLocation	查询Bucket归属region
ks3:PutBucket	新建Bucket
ks3:DeleteBucket	删除Bucket
ks3:ListBucket	列举Bucket下的文件和查询bucket信息
ks3:GetBucketAcl	获取Bucket的ACL信息
ks3:PutBucketAcl	设置Bucket的ACL信息
ks3:PutBucketPolicy	创建Bucket的空间策略信息
ks3:GetBucketPolicy	获取Bucket的空间策略信息
ks3:DeleteBucketPolicy	删除Bucket的空间策略信息
ks3:GetBucketCORS	获取Bucket的CORS配置信息
ks3:PutBucketCORS	设置Bucket的CORS
ks3:DeleteBucketMirror	删除（清空）Bucket回源规则
ks3:GetBucketMirror	获取Bucket的CORS配置信息
ks3:PutBucketMirror	设置Bucket回源规则

Object级别

Object级别操作的对象均为Object,权限关键字与对应的操作如下: 权限关键字	对应的KS3操作
ks3:PutObject	上传文件，包含post、PUT和分块上传
ks3:DeleteObject	删除文件
ks3:GetObject	GET Object 和 HEAD Object
ks3:GetObjectAcl	获取文件的ACL信息
ks3:PutObjectAcl	设置文件的ACL
ks3:ListBucketMultipartUploads	列出分段上传
ks3:ListMultipartUploadParts	列出分段
ks3:AbortMultipartUpload	取消分块上传
ks3:PutBucketLifecycle	创建生命周期管理规则
ks3:DeleteBucketLifecycle	删除生命周期管理规则
ks3:GetBucketLifecycle	查看生命周期管理规则
ks3:PostObjectRestore	解冻归档文件
ks3:PutObjectTagging	添加/更新对象的标签

ks3:GetObjectTagging	查询对象的标签
ks3:DeleteObjectTagging	删除对象的标签

在策略中指定条件

访问策略语言可使您在授予权限时指定条件。在 Condition 元素（或 Condition 块）中，可以指定策略生效的条件。

KS3支持的Condition如下：

Condition	功能
ksc:SourceIp	指定IP地址

访问策略对比

	ACL	空间策略（Bucket Policy）	用户策略（User Policy）
策略类型	基于资源的策略	基于资源的策略	基于用户的策略
可支持的资源操作	较少，只支持简单的读写操作	较多，但不支持service操作，如不支持查询bucket列表	最多，支持service操作，如：ks3:ListBuckets（查询bucket列表）。
是否支持授权给其它账户（账号）	支持	支持	不支持，但可通过创建角色，选择授信账户来实现跨账户的授权。
是否支持授权给IAM子用户	不支持	支持	支持
是否支持授权给角色	不支持	支持	支持

使用不同访问策略的准则

1. 建议使用ACL的场景
   - 如果不需要实现复杂的授权逻辑，只是简单的设置存储空间（Bucket）和文件(Object)公开还是私密，建议使用ACL。
2. 建议使用Bucket Policy的场景
   - 如果想方便的将资源细粒度的授权给其它账户，实现跨账号访问，推荐使用存储空间策略（Bucket Pollicy）。
   - 如果想将资源细粒度的授权给IAM子用户，IAM子用户不需要登录控制台，可以使用存储空间策略（Bucket Policy）。
3. 建议使用User Policy的场景
   - 如果想将资源授权给主账户下不同的IAM子用户，IAM子用户需要登录控制台，需要使用用户策略(User Policy)。
   - 如果希望让IAM子用户扮演角色，使用临时权限，需要使用用户策略(User Policy)。

KS3如何对请求授权

当 KS3 收到请求，例如，存储空间（Bucket）或文件（Object)操作时，它首先验证请求者是否拥有必要的权限。

KS3 对所有相关访问策略、用户策略和基于资源的策略（空间策略、存储空间 ACL、对象 ACL）进行评估， 以决定是否对该请求进行授权。如果任何一个策略现实拒绝（Deny）用户的请求，无论其它的策略是否授权，KS3都会拒绝用户的这次请求。

以下是一些示例说明：

- 如果请求者是IAM子用户，则 KS3 需要确定该IAM子用户所属的父账户是否通过用户策略（User Policy）授予该用户执行操作的必要权限；但是，如果父账户是资源拥有者，也可以通过空间策略(Bucket Policy)授予该IAM子用户执行操作的必要权限。
- IAM子用户除了需要来自父账户的授权外，还需要得到资源拥有者向IAM子用户（使用空间策略）或父账户（使用空间策略、存储桶 ACL 或对象 ACL）授予权限；但是如果父账户是资源拥有者，在通过用户策略授权后，不再需要空间策略的授权。
- 如果请求者是主账户，则KS3只需要判断是否为资源拥有者；如果不是资源拥有者，必须得到资源拥有者向该主账户（使用空间策略、存储空间 ACL 或对象 ACL）授予权限。
- 如果请求者是匿名用户，则KS3需要判断ACL是否公开，或者资源拥有者通过空间策略进行了匿名授权。

流程图如下：

- 用户上下文：如果请求者是IAM子用户，则该子用户必须拥有来自其所属的父账户的授权，同时KS3还要判断其父账户是否也拥有对应的权利。

  注意：如果父账户同时也是资源拥有者，那么通过空间权限(bucket policy或ACL)授予该IAM子用户执行操作的必要权限也是可以的。

- 资源上下文：请求者必须拥有来自资源拥有者的权限才能执行特定的操作。资源拥有者可以通过用户策略、空间策略、ACL向主账号和IAM子用户授予权限。

注意：如果IAM子用户的父账户是资源所有者，已经通过用户策略获得了权限，在空间策略没有显示拒绝但是没有显示授权的情况下，也是可以通过资源上下文的验证。