

目录

目录	1
术语说明	2
权限概述	2
KS3资源	2
被授权人	2
资源操作	3
访问策略	3
ACL（访问控制列表）	3
被授权者	4
授予的权限	4
存储空间权限和文件访问权限	4
存储空间权限	4
文件访问权限	4
设置	4
如何设置ACL	5
空间策略（Bucket Policy）	5
在策略中指定资源	5
在策略中指定被授权人	6
在策略中指定权限	6
在策略中指定条件	6
用户策略（User Policy）	6
在策略中指定资源	7
在策略中指定被授权人	7
在策略中指定权限	7
在策略中指定条件	8
访问策略对比	8
使用不同访问策略的准则	8
KS3如何对请求授权	8

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 AccountId	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 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
Policy	Policy	访问策略	<ul style="list-style-type: none"> 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。

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（跨源资源共享）- 存储空间跨源请求配置
- 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完全控制权限，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。

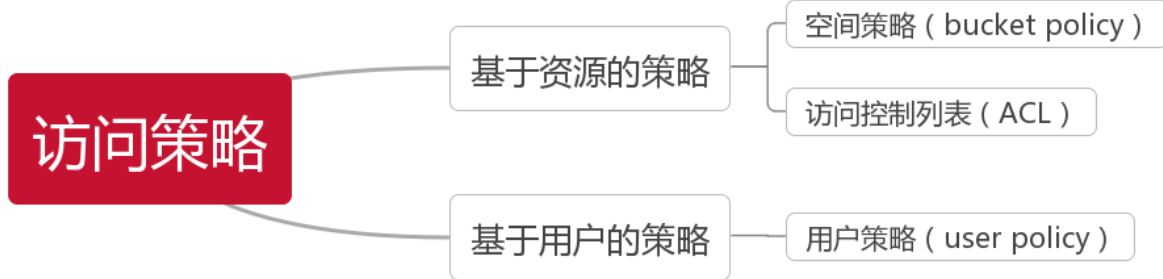
- 创建存储空间（Bucket）和对象/文件（Object）的 KS3 账户拥有这些资源的所有权
- IAM子用户不拥有任何资源，如果一个IAM子用户再被授权后，上传了一个对象/文件，那么该IAM用户所属的父账户拥有此对象/文件。
- 存储空间拥有者可以向其他KS3账户授予上传对象的跨账户权限。在这种情况下，上传对象的KS3账户拥有这些对象。存储空间拥有者对其他账户拥有的对象也同时拥有所有权限，账单也是由存储空间拥有者支付。

资源操作

KS3 提供一组操作用来处理KS3资源。

访问策略

访问策略选项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。



- **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, ACL定义了哪些用户被授予访问权限以及访问的类型, 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。

ACL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>

```

其中Owner元素指定资源所有者的账户ID和用户名，Grant元素指定被授权人和所授予的权限。

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来表示。

```

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

```

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

```

授予的权限

下表列出了KS3在ACL中支持的权限集。注意，对于文件ACL和存储空间ACL，ACL权限集相同。但根据上下文（存储空间ACL或文件ACL），这些ACL权限授予对存储空间或对象操作的特定权限不一样。下表列出了许可并介绍了它们在Bucket和Object许可语境中的含义。

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

从上表得知，ACL仅允许授予有限数量的权限。其中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。下表显示每个ACL权限如何映射到相应的访问策略权限。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权限时的相应访问策略	在Object授予ACL权限时的相应访问策略
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。

存储空间权限和文件访问权限

KS3支持一系列预定义的授权，称为预设ACL。每个预设ACL都有一组预定义的被授权者和权限。这些预设ACL与KS3控制台中存储空间权限和文件访问权限相对应。

- **存储空间权限**

提供存储空间的权限访问控制。目前存储空间有三种访问权限：private, public-read, public-read-write。

权限值	中文名称	权限对访问的限制
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

- **文件访问权限**

提供文件的权限访问控制。目前文件有两种访问权限：private, public-read。

权限值	中文名称	权限对访问的限制
private	私有	文件所有者拥有访问权限。其他人没有访问权限（默认）。
public-read	公开	所有用户（包括匿名用户）拥有文件的访问权限

- **设置**

- 1, 用户在调用接口创建存储空间、文件、修改权限时，使用x-ks-acl请求头来指定预设ACL权限。
- 2, 用户也可以在[KS3控制台](#)通过用户界面，点击 **XX桶** > **空间设置** > **基础设置** 直接设置访问权限。如图：



如何设置ACL

- 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](#)

名称	描述
x-kss-acl	设置预设ACL
x-kss-grant-read	为若干用户授予READ权限
x-kss-grant-write	为若干用户授予WRITE权限
x-kss-grant-full-control	为若干用户授予FULL_CONTROL权限。

- 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设置。
- 详情请见：
 - [PUT Bucket ACL API](#)
 - [PUT Object ACL API](#)

空间策略 (Bucket Policy)

存储空间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，则说明此条策略没有生效。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。
- **被授权人**：被允许或拒绝访问资源的用户和账户。
- **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。

下面是一条空间策略示例。此策略允许账户accountid下的子用户Dave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/*"]
    }
  ]
}
```

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：

- 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[在策略中指定权限](#)。

在策略中指定被授权人

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，accountid是主账户id，root是固定值。
- 3、krn:ksc:iam::(accountid):user/(userName) userName表示子账号，其中user是固定值，accountid是归属主账户id。
- 4、krn:ksc:iam::(accountid):role/(ruleNAME) ruleNAME表示角色名称，其中role是固定值，accountid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“accountid/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。

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。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；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；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。

在策略中指定条件

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

如何在控制台配置存储空间策略，详见文档[空间策略](#)。

用户策略(User Policy)

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，将策略授权给用户，子用户就拥有了相对应的权限。请参阅[授权管理](#)。

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

```
karn:ksc:ks3::bucketname
karn: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

```
karn: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。

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。

权限关键字	对应的KS3操作
<code>ks3:ListBuckets</code>	查询Bucket列表
<code>ks3:GetBucketLocation</code>	查询Bucket归属region
<code>ks3:PutBucket</code>	新建Bucket
<code>ks3>DeleteBucket</code>	删除Bucket
<code>ks3:ListBucket</code>	列举Bucket下的文件和查询bucket信息
<code>ks3:GetBucketAcl</code>	获取Bucket的ACL信息
<code>ks3:PutBucketAcl</code>	设置Bucket的ACL信息
<code>ks3:PutBucketPolicy</code>	创建Bucket的空间策略信息
<code>ks3:GetBucketPolicy</code>	获取Bucket的空间策略信息
<code>ks3>DeleteBucketPolicy</code>	删除Bucket的空间策略信息
<code>ks3:GetBucketCORS</code>	获取Bucket的CORS配置信息
<code>ks3:PutBucketCORS</code>	设置Bucket的CORS
<code>ks3>DeleteBucketMirror</code>	删除（清空）Bucket回源规则
<code>ks3:GetBucketMirror</code>	获取Bucket的CORS配置信息
<code>ks3:PutBucketMirror</code>	设置Bucket回源规则
<code>ks3:PutObject</code>	上传文件，包含post、PUT和分块上传
<code>ks3>DeleteObject</code>	删除文件
<code>ks3:GetObject</code>	GET Object 和 HEAD Object
<code>ks3:GetObjectAcl</code>	获取文件的ACL信息
<code>ks3:PutObjectAcl</code>	设置文件的ACL
<code>ks3:ListBucketMultipartUploads</code>	列出分段上传
<code>ks3:ListMultipartUploadParts</code>	列出分段
<code>ks3:AbortMultipartUpload</code>	取消分块上传
<code>ks3:PutBucketLifecycle</code>	创建生命周期管理规则
<code>ks3>DeleteBucketLifecycle</code>	删除生命周期管理规则
<code>ks3:GetBucketLifecycle</code>	查看生命周期管理规则
<code>ks3:PostObjectRestore</code>	解冻归档文件
<code>ks3:PutObjectTagging</code>	添加/更新对象的标签
<code>ks3:GetObjectTagging</code>	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

策略类型	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 Policy）。
 - 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

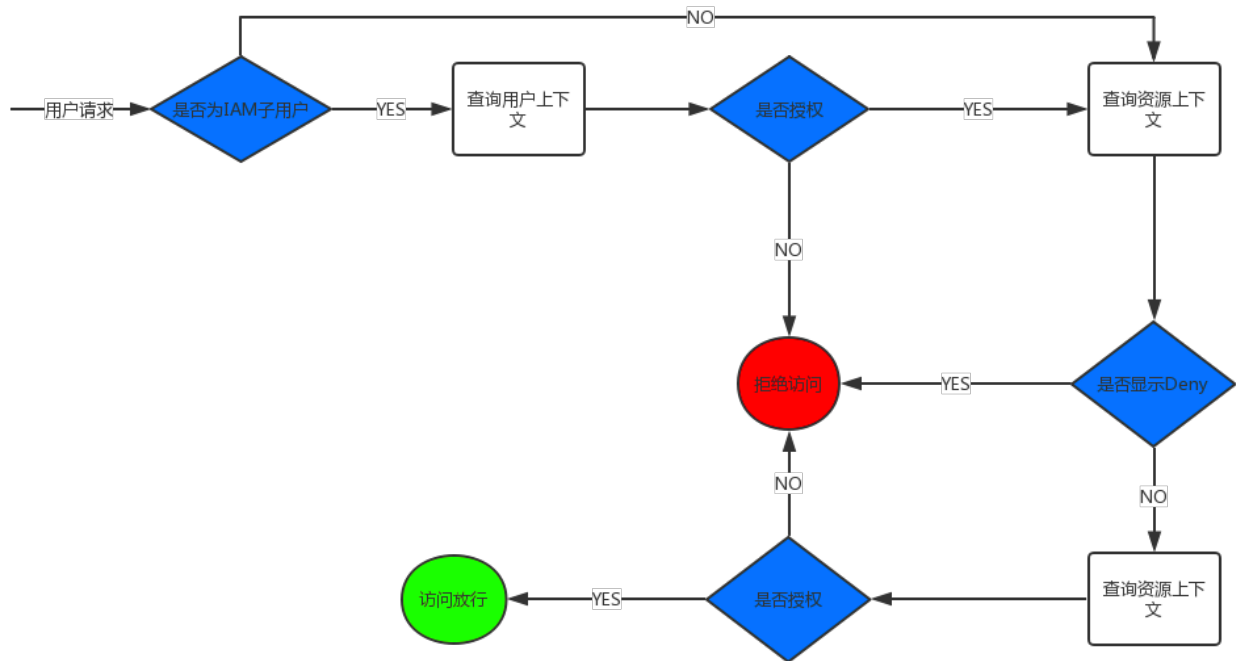
当 KS3 收到请求，例如，存储空间（Bucket）或文件（Object）操作时，它首先验证请求者是否拥有必要的权限。

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。