Object Storage Service

Permission Configuration Guide (Paris Region)

Issue 01

Date 2022-05-20

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


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

Notice

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

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

Security Declaration

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process.* For details about this process, visit the following web page:
https://www.huawei.com/en/psirt/vul-response-process
For vulnerability information, enterprise customers can visit the following web page:
https://securitybulletin.huawei.com/enterprise/en/security-advisory

Contents

1 Introduction to OBS Access Control

By default, OBS resources (buckets and objects) are private. Only resource owners can access their OBS resources. Without authorization, other users cannot access OBS. OBS permission control refers to granting permissions to other accounts or IAM users by editing access policies. For example, if you have a bucket, you can authorize another IAM user to upload objects to your bucket. You can also open buckets to non-public cloud users, so that anyone can access your buckets as public resources over the Internet. OBS offers different methods to help resource owners grant resource permissions to others as required, keeping data secure.

OBS Permission Control Model

OBS provides multiple permission control mechanisms, including IAM permissions, bucket policies, object ACLs, and bucket ACLs. **Table 1-1** describes the mechanisms and application scenarios.

Figure 1-1 OBS permission control mechanisms

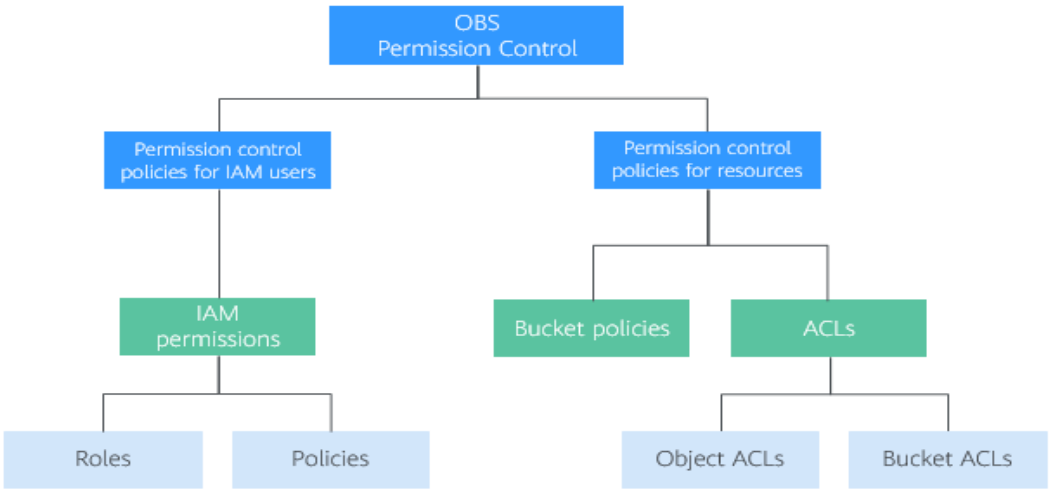


Table 1-1 OBS permission control mechanisms and application scenarios

Method	Description	Scenario
IAM permissions	IAM permissions define the actions that can be performed on your cloud resources. In other words, IAM permissions specify what actions are allowed or denied. After an IAM user is created, the administrator needs to add the user to a group. IAM can grant the user group required OBS access permissions, and then all users in the group automatically inherit the permissions of the user group.	• Controlling access to cloud resources as a whole under an account • Controlling access to all OBS buckets and objects under an account • Controlling access to specified OBS resources under an account
Bucket policies	A bucket policy is attached to a bucket and objects in the bucket. Bucket owners can use bucket policies to grant IAM users or other accounts the permissions to operate buckets and objects in the buckets. ACLs of buckets and objects supplement bucket policies, and in many cases, bucket policies replace ACLs.	• Granting other accounts the permissions to access OBS resources • Configuring bucket policies to grant IAM users various access permissions to different buckets

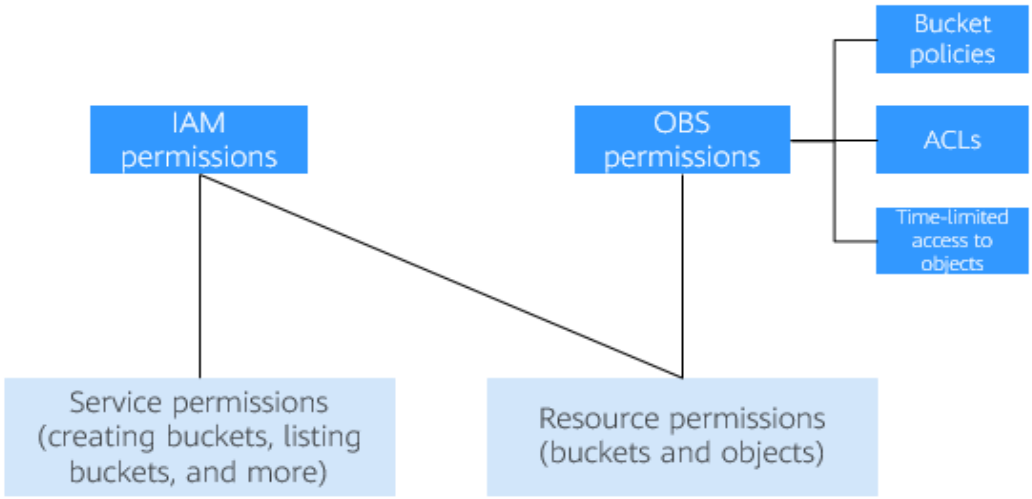
Method	Description	Scenario
Object ACLs	Controls access to objects for accounts or user groups. Object owners can configure the object access control list (ACL) to grant basic read and write permissions to specified accounts or user groups. **NOTE** • By default, an object ACL is created upon the creation of the object. The object owner has full control over the object. • An object owner is the account that uploads the object, but may not be the owner of the bucket that stores the object. For example, account **B** is granted the permission to access a bucket of account **A**, and account **B** uploads a file to the bucket. In that case, instead of the bucket owner account **A**, account **B** is the owner of the object. By default, account A is not allowed to access this object and cannot read or modify the object ACL.	• If object-level access control is required, a bucket policy can be used to grant the access permission to an object or a set of objects. After the access permission is granted to an object set, it is not practical to configure a bucket policy to grant the access permission to an object in the object set separately. Then the object ACL is recommended for easier access control over single objects. • An object is accessed through a URL. Generally, if you want to grant anonymous users the permission to read an object through a URL, use the object ACL.
Bucket ACLs	Controls access to buckets for accounts or user groups. Bucket owners can configure the bucket ACL to grant basic read and write permissions to specified accounts or user groups. **NOTE** • By default, a bucket ACL is created upon the creation of the bucket. The bucket owner has full control over the bucket. • Bucket ACLs do not provide fine-grained permission control. Generally, IAM permissions and bucket policies are recommended.	• Granting an account the read and write access to a bucket, so that data in the bucket can be shared or external buckets can be added. For example, after account **A** grants account **B** the read and write access to a bucket, account **B** can access the bucket by adding an external bucket through OBS Browser+ or using APIs and SDKs. • Grant the log delivery user write access to the target bucket that stores access logs.

Relationship Between OBS Permissions and IAM Permissions

OBS provides multiple permission control mechanisms, including time-limited access to objects, object ACLs, bucket ACLs, and bucket policies. Some service-level

permissions (for example, creating a bucket and listing all buckets) cannot be configured through OBS and can only be configured on IAM. OBS permissions apply only to resources (buckets and objects). To grant both OBS service-level and resource-level permissions, you must use IAM permissions or both IAM and OBS permissions.

Figure 1-2 Relationship between OBS permissions and IAM permissions



OBS Permission Control Elements

The following factors determine the authorization result:

- **Principal (authorized user)**
- **Effect**
- **Resource**
- **Action**
- **Condition**

For details about elements, see **Bucket Policy Parameters**.

Table 1-2 describes elements in different permission control mechanisms.

Table 1-2 OBS permission control elements in different permission control mechanisms

Method	Principal	Supported Effect	Authorized Resource	Authorized Action	Condition Configuration
IAM Permissions	IAM user	• Allow • Deny	All or specified OBS resources	All permissions to access OBS	Supported

Method	Principal	Supported Effect	Authorized Resource	Authorized Action	Condition Configuration
Bucket Policy	• Account • IAM user • Anonymous users	• Allow • Deny	Specified bucket and resources in the bucket	All permissions to access OBS	Supported
Object ACL	• Account • Anonymous users	Allow	Specified object	• Obtains the content and metadata of a specified object. • Obtains the content and metadata of an object with a specified version. • Obtains information about an object ACL. • Obtains information about the ACL for an object of a specified version. • Configures an object ACL. • Configures the ACL for an object of a specified version.	Not supported

Method	Principal	Supported Effect	Authorized Resource	Authorized Action	Condition Configuration
Bucket ACL	• Account • Anonymous users • Log delivery user groups	Allow	Specified bucket	• Identifies whether a bucket exists. • Lists objects in a bucket, and obtains the bucket metadata. • Lists versioned objects in a bucket. • Lists multipart uploads. • Performs PUT upload, POST upload, multipart upload, initialization of uploaded parts, and merging of parts. • Deletes an Object. • Deletes an object of a specified version. • Obtains bucket ACL information. • Configures a bucket ACL. • Obtains object content. • Obtains object metadata.	Not supported

How to Select IAM Permissions, Bucket Policies, and ACLs

Based on the advantages and disadvantages of the three elements, you are advised to preferentially use IAM permissions and bucket policies.

- Select IAM permissions in the following scenarios:
  - Grant the same permissions to numerous IAM users under the same account.
  - Grant the same permissions to all OBS resources or multiple buckets.
  - Configure OBS service-level permissions, such as creating and listing buckets.
  - Restrict the permissions of temporary access keys used for temporarily authorized access to OBS.
- Select bucket policies in the following scenarios:
  - Grant permissions across accounts or grant permissions to anonymous users.

  - Grant different permissions to different IAM users under the same account.
- Still do not know what to select?

  Identify the problem you are most concerned with:

  - What the user can do - IAM permissions recommended

    You can search for an IAM user and check the permissions of the user group to which the user belongs to know what the user can do.
  - Who can access the bucket? — Use bucket policies.

    You can query the bucket and check the bucket policy to know who can access the bucket.

NOTE

It is better for you to use the same method for access control, because as the number of IAM permissions and bucket policies increase, access maintenance will become increasingly difficult.

When to Select an ACL?

- As a supplement to IAM permissions and bucket policies:

  IAM permissions and bucket policies have granted access permissions to an object set, but you want to grant access permissions to a single object.
- To allow an object to be accessible to all anonymous Internet users, configuring object ACL operations is more convenient.

  When uploading an object, you can use the ACL header to specify the read and write permissions of the object.

Relationship Between Bucket ACLs and Bucket Policies

Bucket ACLs are used to control basic read and write access to buckets. Custom settings of bucket policies support more actions that can be performed on buckets. Bucket ACLs supplement bucket policies. In many cases, bucket policies can replace bucket ACLs to manage access to buckets. **Relationship Between Bucket Policies and Bucket ACLs** shows the mapping between bucket ACL access permissions and bucket policy actions.

OBS Permission Control Principles

- Least privilege

  Never grant IAM users more than the minimum level of access needed to complete a task. For example, if an IAM user only needs to upload and download objects to a directory, you do not need to assign the user the read and write permissions for the entire bucket.
- Separation of duties

  Management of resources or of permissions can be assigned to different IAM users. For example, you can let one IAM user assign permissions, and let other IAM users manage OBS resources.
- Restriction by condition

  To enhance the security of the resources in a bucket, specific conditions can be configured to control when a permission is applied. For example, a bucket

policy with conditions contained can be configured for OBS to accept requests only from a specific IP address.

How Do Access Control Mechanisms Work When They Conflict?

In the OBS permission control elements, there are allow and deny effects, which indicate the permission to allow or deny an operation.

Based on the least-privilege principle, decisions default to deny, and an explicit deny statement always takes precedence over an allow statement. For example, IAM permissions grant a user access to an object, a bucket policy denies the user's access to that object, and there is no ACL. Then access will be denied.

If no method specifies an allow statement, then the request will be denied by default. Only if no method specifies a deny statement and one or more methods specify an allow statement, will the request be allowed. For example, if a bucket has multiple bucket policies with allow statements, adding such a new bucket policy applies the allowed permissions to the bucket, but adding a new bucket policy with a deny statement will make the permissions work differently. The deny statement will take precedence over allow statements, even if the denied permissions are allowed in other bucket policies.

Figure 1-3 Authorization process

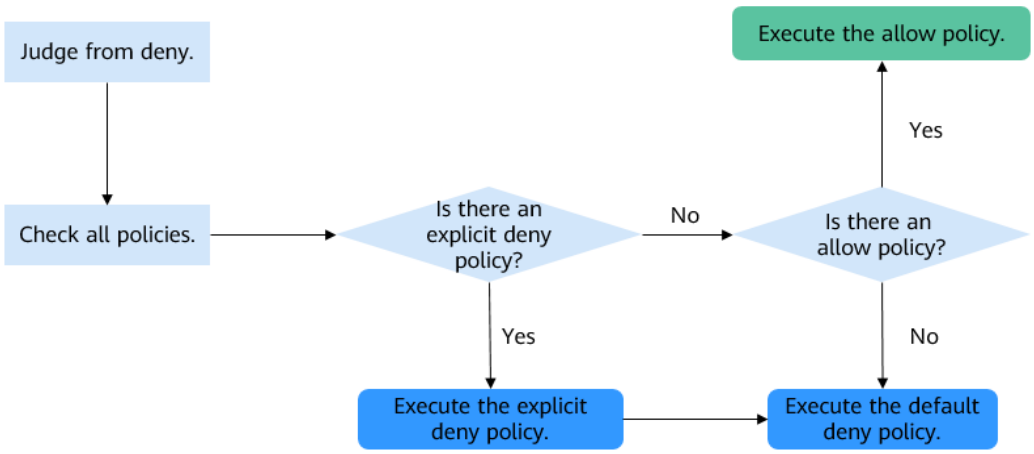


Figure 1-4 describes how bucket policies, IAM permissions, and ACLs work (allow or deny) when you grant the IAM users of your account the access to OBS buckets and resources in the buckets. ACLs are applied to accounts and do not control IAM users' read and write permissions for the buckets and the sources in the buckets under their account.

Figure 1-4 Working mechanisms (allow or deny) of bucket policies and IAM permissions in the same account

Bucket Policy	IAM Policy		
	Deny	Allow	Default Deny
Deny	Deny	Deny	Deny
Allow	Deny	Allow	Allow
Default Deny	Deny	Allow	Deny

Permissions configured
The final result of all settings is Deny
The final result of all settings is Allow

[Figure 1-5](#) describes how bucket policies, IAM permissions, and ACLs work (allow or deny) when you grant any other account and the IAM users of this account the access to OBS buckets and resources in the buckets.

Figure 1-5 Working mechanisms (allow or deny) of bucket policies, IAM permissions, and ACLs in cross-account access grant scenarios

Bucket Policy	IAM Policy			ACL
	Deny	Allow	Default Deny	
Deny	Deny	Deny	Deny	Allow
				Default Deny
Allow	Deny	Allow	Deny	Allow
				Default Deny
Default Deny	Deny	Allow	Deny	Allow
		Deny	Deny	Default Deny

Permissions configured
The final result of all settings is Deny
The final result of all settings is Allow

NOTE

- If both the bucket policy and IAM policy are set to **Default Deny**, but the ACL is set to **Allow**, the final result is **Deny**. ACLs are used to supplement bucket policies.

Concepts

- Domain: An account that is automatically created during your registration. This account has full access control over its resources and IAM users.
- IAM user: A user created by the administrator in IAM. An IAM user may be an employee, a system, or an application. An IAM user has access permissions to specified resources. IAM users have identity credentials (passwords and access keys) and can log in to the management console or call APIs.
- Anonymous user: A common visitor who has not registered.
- A log delivery user group: A user group who only delivers access logs of buckets and objects to the specified target bucket. OBS does not create or upload any file to a bucket automatically. If you want to record access logs for a bucket, you must grant the log delivery user group required permissions, so that OBS can write the access logs to the specified bucket. This user group is only used to record internal logs of OBS.

2 Permission Control Mechanisms

2.1 IAM Permissions

IAM Permissions Overview

By default, newly created IAM users do not have any permissions. You need to add the user to one or more groups, and attach permission policies or roles to these groups. Users inherit permissions from the groups to which they are added and can perform specified operations on cloud services based on the permissions.

IAM permissions take effect on all OBS buckets and objects. To grant an IAM user the permission to operate OBS resources, you need to assign one or more OBS permission sets to the user group to which the user belongs.

OBS is a global service because it is available for all physical regions. IAM permissions are assigned to users in the global project, and users do not need to switch regions when accessing OBS.

You can grant permissions to users by roles and policies.

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

NOTE

Due to data caching, a role and policy involving OBS actions will take effect 10 to 15 minutes after it is attached to a user, an enterprise project, and user group.

IAM presets system permissions for each cloud service so that you can quickly configure basic permissions. **Table 2-1** describes all system permissions of OBS.

Custom policies can be created to supplement the system-defined policies of OBS.

Table 2-1 OBS system permissions

Role/Policy Name	Description	Type	Dependency
Tenant Administrator	Users with this permission can perform all operations on all services except IAM.	System-defined role	N/A
Tenant Guest	Users with this permission can perform read-only operations on all services except IAM.	System-defined role	N/A
OBS FullAccess	Users with this permission are OBS administrators and can perform any operations on all OBS resources under the account.	System-defined policy	N/A
OBS Buckets Viewer	Users with this permission can list buckets, obtain basic bucket information, and obtain bucket metadata.	System-defined policy	N/A
OBS ReadOnlyAccess	Users with this permission can list buckets, obtain basic bucket information, obtain bucket metadata, and list objects (not the objects that have been versioned). **NOTE** If a user with this permission fails to list objects on OBS Console, there may be multiple versions of objects in the bucket. In this case, you need to grant the user the **obs:bucket:ListBucketVersions** permission so that the user can view different versions of objects on OBS Console.	System-defined policy	N/A
OBS OperateAccess	Users with this permission can perform all OBS ReadOnlyAccess operations and perform basic object operations, such as uploading objects, downloading objects, deleting objects, and obtaining object ACLs. **NOTE** If a user with this permission fails to list objects on OBS Console, there may be multiple versions of objects in the bucket. In this case, you need to grant the user the **obs:bucket:ListBucketVersions** permission so that the user can view different versions of objects on OBS Console.	System-defined policy	N/A

The following table lists the common operations supported by each system-defined policy or role of OBS. Select the policies or roles as required.

Table 2-2 Permissions and the allowed operations on OBS resources

Operation	Tenant Administrator	Tenant Guest	OBS FullAccess	OBS Buckets Viewer	OBS ReadOnly Access	OBS Operate Access
Listing buckets	Yes	Yes	Yes	Yes	Yes	Yes
Creating buckets	Yes	No	Yes	No	No	No
Deleting buckets	Yes	No	Yes	No	No	No
Obtaining basic bucket information	Yes	Yes	Yes	Yes	Yes	Yes
Controlling bucket access	Yes	No	Yes	No	No	No
Managing bucket policies	Yes	No	Yes	No	No	No
Modifying bucket storage classes	Yes	No	Yes	No	No	No
Listing objects	Yes	Yes	Yes	No	Yes	Yes
Listing versioned objects	Yes	Yes	Yes	No	No	No
Uploading a file	Yes	No	Yes	No	No	Yes
Creating a folder	Yes	No	Yes	No	No	Yes
Deleting a file	Yes	No	Yes	No	No	Yes
Deleting a folder	Yes	No	Yes	No	No	Yes

Operation	Tenant Administrator	Tenant Guest	OBS FullAccess	OBS Buckets Viewer	OBS ReadOnly Access	OBS Operate Access
Downloading a file	Yes	Yes	Yes	No	No	Yes
Deleting files with multiple versions	Yes	No	Yes	No	No	Yes
Downloading files with multiple versions	Yes	Yes	Yes	No	No	Yes
Modifying object storage classes	Yes	No	Yes	No	No	No
Restoring files	Yes	No	Yes	No	No	No
Undeleting a file	Yes	No	Yes	No	No	Yes
Deleting fragments	Yes	No	Yes	No	No	Yes
Controlling object access	Yes	No	Yes	No	No	No
Configuring object metadata	Yes	No	Yes	No	No	No
Obtaining object metadata	Yes	Yes	Yes	No	No	Yes
Managing versioning	Yes	No	Yes	No	No	No
Managing logging	Yes	No	Yes	No	No	No
Managing event notifications	Yes	No	Yes	No	No	No

Operation	Tenant Administrator	Tenant Guest	OBS FullAccess	OBS Buckets Viewer	OBS ReadOnly Access	OBS Operate Access
Managing lifecycle rules	Yes	No	Yes	No	No	No
Managing static website hosting	Yes	No	Yes	No	No	No
Managing CORS rules	Yes	No	Yes	No	No	No
Managing URL validation	Yes	No	Yes	No	No	No
Managing domain names	Yes	No	Yes	No	No	No
Managing cross-region replication	Yes	No	Yes	No	No	No
Configuring an object ACL	Yes	No	Yes	No	No	No
Configuring the ACL for an object of a specified version	Yes	No	Yes	No	No	No
Obtaining an object ACL	Yes	Yes	Yes	No	No	Yes
Obtaining the ACL of a specified object version	Yes	Yes	Yes	No	No	Yes

Operation	Tenant Administrator	Tenant Guest	OBS FullAccess	OBS Buckets Viewer	OBS ReadOnly Access	OBS Operate Access
Performing a multipart upload	Yes	No	Yes	No	No	Yes
Listing uploaded parts	Yes	Yes	Yes	No	No	Yes
Canceling a multipart upload	Yes	No	Yes	No	No	Yes

Application Scenarios of IAM Permissions

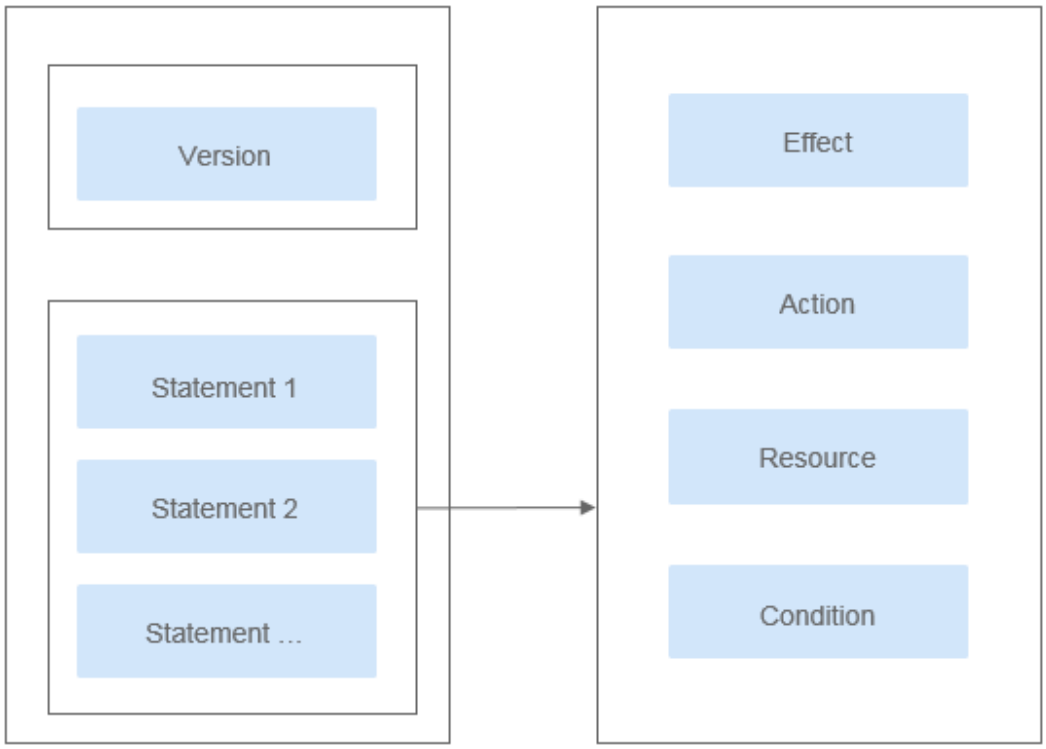
IAM permissions are used to authorize IAM users under an account.

- Controlling access to cloud resources as a whole under an account
- Controlling access to all OBS buckets and objects under an account
- Controlling access to specified OBS resources under an account

Policy Structure and Syntax

A policy consists of a version and statements. Each policy can have multiple statements.

Figure 2-1 Policy structure



Policy syntax example:

```
{
   "Version": "1.1",
   "Statement": [
            {
         "Effect": "Allow",
         "Action": [
            "obs:bucket:HeadBucket",
            "obs:bucket:ListBucket",
            "obs:bucket:GetBucketLocation"
         ],
         "Resource": [
            "obs:*:*:bucket:*"
         ],
         "Condition": {
            "StringEndWithIfExsits": {
               "g:UserName": ["specialCharacter"]
            },
            "Bool": {
               "g:MFAPresent": ["true"]
            }
         }
      }
   ]
}
```


Table 2-3 Policy syntax parameters

Parameter	Description
Version	The version number of a policy. • **1.0**: RBAC policies. An RBAC policy consists of permissions for an entire service. Users in a group with such a policy assigned are granted all of the permissions required for that service. • **1.1**: Fine-grained policies. A fine-grained policy consists of API-based permissions for operations on specific resource types. Fine-grained policies, as the name suggests, allow for more fine-grained control on specific operations and resources than RBAC policies. For example: You can restrict an IAM user to access only the objects in a specific directory of an OBS bucket.

Parameter	Description
Statement	Detailed descriptions of a policy, including **Effect**, **Action**, **Resource**, and **Condition**. **Resource** and **Condition** are optional. • **Effect** The valid values for **Effect** are **Allow** and **Deny**. System policies contain only **Allow** statements. For custom policies containing both **Allow** and **Deny** statements, the **Deny** statements take precedence. • **Action** Actions allowed on resources. An action is in the format of *Service name*:*Resource type*:*Action*. A policy can contain one or more actions. You can use a wildcard (*) to indicate all of the services, resource types, or actions depending on their location in the action. There are two types of OBS resources: buckets and objects. • **Resource** Resources on which the policy takes effect. A resource is in the format of *Service name*:*Region*:*Domain ID*:*Resource type*:*Resource path*. You can use a wildcard (*) to indicate all of the services, regions, domain IDs, resource types, or resource paths depending on their location in the resource. In the JSON view, if **Resource** is not specified, the policy takes effect for all resources. The value of **Resource** supports uppercase (A to Z), lowercase (a to z) letters, digits (0 to 9), and the following characters: **-_*./**. If the value contains invalid characters, use the wildcard character (*). OBS is a global service. Therefore, set **Region** to *****. **Domain ID** indicates the ID of the resource owner. Set it to ***** to indicate the ID of the account to which the resources belong. Examples: – **obs:*:*:bucket:***: all OBS buckets – **obs:*:*:object:my-bucket/my-object/***: all objects in the **my-object** directory of the **my-bucket** bucket • **Condition** When creating a custom policy, you can add condition elements to control when the policy takes effect. A condition consists of a condition key and an operator. Condition keys are either global or service-level and are used in the condition elements of a policy statement. Global condition keys (starting with **g:**) are available for actions of all services, while service-level condition keys (starting with a service name acronym like **obs:**) are available only for actions of a specific service. An operator is used together with a condition key to form a complete condition statement.

Parameter	Description
	OBS has a group of predefined condition keys that can be used in IAM. For example, to define an allow permission, you can use the condition key **obs:SourceIp** to filter matching requesters by IP address. The condition keys and operators supported by OBS are the same as those in the bucket policy. When configuring condition keys in IAM, start the condition keys and operators with **obs:**. For detailed condition information, see **Bucket Policy Parameters**. The value of **Condition** can contain only uppercase letters (A to Z), lowercase letters (a to z), digits (0 to 9), and the following characters: **-,./_@#$%&**. If the value contains unsupported characters, consider using the condition operators (like StringMatch) for fuzzy match. Examples: – **StringEndWithIfExists":{"g:UserName":["specialCharacter"]}**: The statement is valid for users whose names end with **specialCharacter**. – **"StringLike":{"obs:prefix":["private/"]}**: When listing objects in a bucket, you need to set prefix to **private/** or include **private/**.

Configuring IAM Permissions

- **Creating a User and Granting OBS Permissions**
- **Creating a Custom Policy**

Example Custom Policies

- **Example 1: Grant all OBS permissions to users.**

  This policy allows users to perform any operation on OBS using the API, SDKs, OBS Console, or tools.

  When a user logs in to OBS Console, the user accesses resources of other services, such as audit information in CTS, acceleration domain names in CDN, and keys in KMS. Therefore, in addition to the OBS permissions, you need to grant users the permissions for other services. CDN is a global service, while CTS, SMN, and KMS are regional ones. You need to configure the **Tenant Guest** permission for the global project and regional projects based on the services and regions that you use.

```
{
    "Version": "1.1",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "obs:*:*"
            ]
        }
    ]
}
```


- **Example 2: Grant the read-only permission on a bucket to users (any directory).**

  This policy allows users to list and download all objects in bucket **obs-example**.

```
{
   "Version": "1.1",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "obs:object:GetObject",
            "obs:bucket:ListBucket"
         ],
         "Resource": [
            "obs:*:*:object:obs-example/*",
            "obs:*:*:bucket:obs-example"
         ]
      }
   ]
}
```

- **Example 3: Grant the read-only permission on a bucket to users (specified directory).**

  This policy allows users to only download objects in the **my-project/** directory of bucket **obs-example**. Objects in other directories can be listed but cannot be downloaded.

```
{
   "Version": "1.1",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "obs:object:GetObject",
            "obs:bucket:ListBucket"
         ],
         "Resource": [
            "obs:*:*:object:obs-example/my-project/*",
            "obs:*:*:bucket:obs-example"
         ]
      }
   ]
}
```

- **Example 4: Grant the read and write permissions on a bucket to users (specified directory).**

  This policy allows users to list, download, upload, and delete objects in the **my-project** directory of bucket **obs-example**.

```
{
   "Version": "1.1",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "obs:object:GetObject",
            "obs:object:ListMultipartUploadParts",
            "obs:bucket:ListBucket",
            "obs:object:DeleteObject",
            "obs:object:PutObject"
         ],
         "Resource": [
            "obs:*:*:object:obs-example/my-project/*",
            "obs:*:*:bucket:obs-example"
         ]
      }
   ]
}
```


- **Example 5: Grant all permissions on a bucket to users.**

  This policy allows users to perform any operation on bucket **obs-example**.

```
{
   "Version": "1.1",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "obs:*:*"
         ],
         "Resource": [
            "obs:*:*:bucket:obs-example",
            "obs:*:*:object:obs-example/*"
         ]
      }
   ]
}
```

- **Example 6: Deny a user the permission to upload objects.**

  A policy with only "Deny" permissions must be used in conjunction with other policies to take effect. If the permissions assigned to a user contain both "Allow" and "Deny", the "Deny" permissions take precedence over the "Allow" permissions.

  The following method can be used if you need to assign permissions of the **OBS OperateAccess** policy to a user but also forbid the user from uploading objects. Create a custom policy for denying object upload, and assign both policies to the user. Then the user can perform all **OBS OperateAccess** permissions except uploading objects. The following is an example of a deny policy:

```
{
   "Version": "1.1",
   "Statement": [
      {
         "Effect": "Deny",
         "Action": [
            "obs:object:PutObject"
         ]
      }
   ]
}
```

- **Example 7: Grant users the permissions required to change a bucket's storage class and to delete certain objects in the bucket.**

  This policy allows users to change the storage class of bucket **obs-example** and to delete object **my-object.txt** in the bucket.

```
{
   "Version": "1.1",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "obs:bucket:ListAllMyBuckets",
            "obs:bucket:ListBucket"
         ]
      },
      {
         "Effect": "Allow",
         "Action": [
            "obs:object:DeleteObject",
            "obs:bucket:PutBucketStoragePolicy"
         ],
         "Resource": [
            "OBS:*:*:object:obs-example/my-object.txt",
```

```
            "OBS:*:*:bucket:obs-example"
        ]
    }
  ]
}
```

2.2 Bucket Policies

Overview

A bucket policy applies to an OBS bucket and objects in the bucket. By leveraging bucket policies, the owner of a bucket can authorize IAM users or other accounts the permissions to operate the bucket and objects in the bucket.

NOTE

- Creating a bucket and obtaining a bucket list are service-level operations. To obtain such operation permissions, you need to configure **IAM permissions**.
- Due to data caching, after a bucket policy is configured, it takes 5 minutes at most for the policy to take effect.

Bucket Policy Overview

A bucket policy is attached to a bucket and objects in the bucket. An OBS bucket owner can use bucket policies to grant IAM users, other accounts, or anonymous users the permissions to operate buckets and objects in the buckets. OBS provides standard and advanced bucket policies.

Standard Bucket Policies:

There are three options for standard bucket policies.

- **Private**: No access beyond the bucket ACL settings is granted.
- **Public Read**: Any user can read objects in the bucket.
- **Public Read and Write**: Any user can read, write, and delete objects in the bucket.

After a bucket is created, the default bucket policy is **Private**. Only the bucket owner has the full control permissions over the bucket. To ensure data security, it is recommended that you do not use the **Public Read** or **Public Read and Write** policies.

Table 2-4 Standard bucket policies

Parameter	Private	Public Read	Public Read and Write
Effect	N/A	Allow	Allow
Principal	N/A	* (Any user)	* (Any user)
Resources	N/A	* (All objects in a bucket)	* (All objects in a bucket)

Parameter	Private	Public Read	Public Read and Write
Actions	N/A	• GetObject • GetObjectVersion • HeadBucket • ListBucket	• GetObject • GetObjectVersion • PutObject • DeleteObject • DeleteObjectVersion • HeadBucket • ListBucket
Conditions	N/A	N/A	N/A

Custom Bucket Policy:

The following three modes are provided to facilitate quick configuration of a custom bucket policy:

- **Read-only**: With the **Read-only** mode, you only need to specify the **Principal** (authorized users). Then the authorized users have the read permission for the bucket and objects in the bucket, and can perform all GET operations on these resources.
- **Read and write**: With the **Read and write** mode, you only need to specify the **Principal** (authorized users). Then the authorized users have the full control permissions for the bucket and objects in the bucket, and can perform any operation on these resources.
- **Customized**: With the **Customized** mode, you can define the specific operation permissions that you want to authorize to users and accounts by configuring the parameters of **Effect**, **Principal**, **Resources**, **Actions**, and **Conditions**. For details, see **Bucket Policy Parameters**.

NOTE

On OBS Console, when you use the custom bucket policy to authorize other users with resource operation permissions, you also need to authorize the users with the bucket read permission **ListBucket** (leave the resource name blank to indicate that the policy takes effect on the entire bucket). Otherwise, the users have no permission to access the bucket.

Bucket Policy Application Scenarios

- You can use bucket policies to grant other accounts the permissions to access OBS resources.
- You can configure bucket policies to grant IAM users various access permissions to different buckets.

Policy Structure and Syntax

A bucket policy is in JSON format. The format is as follows:

```
{
"Statement" : [
   {
    statement1
   },
```



```
  {
   statement2
  },
 ......
 ]
}
```

Example:

```
{
  "Statement":[
     {
        "Sid": "ExampleStatementID1",
        "Principal":{
           "ID":[
              "domain/account ID",
              "domain/account ID:user/User ID"
           ]
        },
        "Effect":"Allow",
        "Action":[
           "CreateBucket",
           "DeleteBucket"
        ],
        "Resource":"000-02/key01",
        "Condition":{
           "NumericNotEquals":{
              "Referer":"sdf"
           },
           "StringNotLike":{
              "Delimiter":"ouio"
           }
        }
     }
  ]
}
```

A bucket policy comprises one or more statements. Each statement contains the following elements:

Table 2-5 Statement elements

Element	Description	Mandatory or Optional
Sid	ID of a statement. The value is a string that describes the statement.	Optional
Principal	Domains (accounts) and users (IAM users) to which the statement applies. The wildcard (*) is supported, indicating all users. ● When permissions are granted to all IAM users in a domain (account), the principal format is *domain/domainid:user/**. ● When a user is authorized, the principal format is *domain/domainid:user/userId* or *domain/domainid:user/userName*.	Optional. Select either **Principal** or **NotPrincipal**.

Element	Description	Mandatory or Optional
NotPrincipal	An exception to a list of principals in the statement. You can deny access to all principals except the ones named in the **NotPrincipal** element. This parameter has the same value format as **Principal**.	Optional. Select either **Principal** or **NotPrincipal**.
Effect	Whether the permission in a statement is allowed or denied. The value is **Allow** or **Deny**.	Mandatory
Action	Actions which a statement applies to. This parameter specifies a set of all the operations supported by OBS. Its values are case insensitive. The value supports a wildcard character (*) that indicates all actions, for example, **"Action": ["List*", "Get*"]**.	Optional. Select either **Action** or **NotAction**.
NotAction	An exception to a list of actions in the statement. All actions are performed except the ones specified in **NotAction**. The value of this element is similar to **Action**.	Optional. Select either **Action** or **NotAction**.
Resource	Resources on which the statement takes effect. The wildcard (*) is supported, indicating all resources.	Optional. Select either **Resource** or **NotResource**.
NotResource	An exception to a list of resources in a statement. A policy is not applied to the resources specified in **NotResource**. The value of this parameter is similar to that of **Resource**.	Optional. Select either **Resource** or **NotResource**.
Condition	Conditions for a statement to take effect.	Optional

For details about each element, see **Bucket Policy Parameters**.

Bucket Policy Example

- **Example 1: Grant an IAM user the specified operation permission on all objects in a specified bucket.**

  The following example policy grants the PutObject and PutObjectAcl permissions to the IAM user whose ID is **71f3901173514e6988115ea2c26d1999** under account **b4bf1b36d9ca43d984fbcb9491b6fce9** (account ID).

```
{
   "Statement":[
   {
     "Sid":"AddCannedAcl",
     "Effect":"Allow",
     "Principal": {"ID": ["domain/b4bf1b36d9ca43d984fbcb9491b6fce9:user/
71f3901173514e6988115ea2c26d1999"]},
     "Action":["PutObject","PutObjectAcl"],
     "Resource":["examplebucket/*"]
```



```
  }
 ]
}
```

- **Example 2: Grant all permissions for a specified bucket to an IAM user.**

  The following example policy grants all operation permissions (including bucket operations and object operations) of **examplebucket** to the user whose ID is **71f3901173514e6988115ea2c26d1999** in account **b4bf1b36d9ca43d984fbcb9491b6fce9** (account ID).

```
{
   "Statement":[
   {
     "Sid":"test",
     "Effect":"Allow",
     "Principal": {"ID": ["domain/b4bf1b36d9ca43d984fbcb9491b6fce9:user/
71f3901173514e6988115ea2c26d1999"]},
     "Action":["*"],
     "Resource":[
       "examplebucket/*",
       "examplebucket"
     ]
   }
 ]
}
```

- **Example 3: Grant all permissions except the object deletion permission to an OBS user.**

  The following example policy grants a user (user ID **71f3901173514e6988115ea2c26d1999**) of an account (ID **b4bf1b36d9ca43d984fbcb9491b6fce9**) all permissions for the **examplebucket** bucket, excluding the permission to delete objects.

```
{
   "Statement":[
   {
     "Sid":"test1",
     "Effect":"Allow",
     "Principal": {"ID": ["domain/b4bf1b36d9ca43d984fbcb9491b6fce9:user/
71f3901173514e6988115ea2c26d1999"]},
     "Action":["*"],
     "Resource":["examplebucket/*"]
   },
   {
     "Sid":"test2",
     "Effect":"Deny",
     "Principal": {"ID": ["domain/b4bf1b36d9ca43d984fbcb9491b6fce9:user/
71f3901173514e6988115ea2c26d1999"]},
     "Action":["DeleteObject"],
     "Resource":["examplebucket/*"]
   }
 ]
}
```

- **Example 4: Grant the read-only permission on a specified object to anonymous users.**

  The following example policy grants the **GetObject** (download object) permission of **exampleobject** in bucket **examplebucket** to anonymous users, allowing everyone to read data of the exampleobject object.

```
{
   "Statement":[
   {
     "Sid":"AddPerm",
     "Effect":"Allow",
     "Principal": "*",
     "Action":["GetObject"],
```



```
      "Resource":["examplebucket/exampleobject"]
    }
  ]
}
```

- **Example 5: Restrict access to a specific IP address.**

  The following policy grants all users the permission to perform any OBS operation. However, the requests must be from the specified IP address range. The IP address range that is allowed by the statement is 192.168.0.* with an exception of 192.168.0.1.

  Use **IpAddress** and **NotIpAddress** conditions, and use the **SourceIp** (in OBS range) condition key. The value of **SourceIp** is the CIDR notation described in RFC 4632.

```
{
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "*",
      "Resource": "examplebucket/*",
      "Condition": {
        "IpAddress": {"SourceIp": "192.168.0.0/24"},
        "NotIpAddress": {"SourceIp": "192.168.0.1/32"}
      }
    }
  ]
}
```

2.3 ACLs

An ACL is a list that defines grantees and their granted permissions.

Bucket and object ACLs are attached to accounts. By default, an ACL is created when a bucket or object is created, authorizing the owner the full control over the bucket or object.

To implement simple and practical authorization for users, the OBS ACL has the following features:

- The ACL takes effect for both the account and the users under the account.
- When the owner of a bucket is the same as the owner of an object, the ACL configured on the bucket takes effect on the bucket and objects in the bucket by default.
- An ACL can be carried when a bucket is created, or an ACL can be configured after a bucket is created. An object can carry an ACL when it is uploaded. You can also configure the ACL after the object is uploaded successfully.

ACLs are write and read control rules attached to accounts, whose permission granularity is not as fine as bucket policies and IAM policies. Generally, it is recommended that you use IAM permissions and bucket policies for access control.

Table 2-6 lists users to whom you can grant bucket access permissions by configuring an ACL.

Table 2-6 Authorized users supported by OBS

Principal	Description
Specific User	ACLs can be used to grant accounts with bucket/object access permissions. Once a specific account is granted with certain bucket/object access permissions, all IAM users who have OBS resource permissions under this account can have the same access permissions to operate the bucket or object. You can configure bucket policies to grant different permissions to different IAM users.
Owner	The owner of a bucket is the account that created the bucket. The bucket owner has all bucket access permissions by default. The read and write permissions to the bucket ACL are permanently available to the bucket owner, and cannot be modified. An object owner is the account that uploads the object, but may not be the owner of the bucket that stores the object. The object owner has all control over the object by default. The read and write permissions to the object ACL are permanently available to the object owner, and cannot be modified. **NOTICE** Do not modify the bucket owner's read and write access permissions for the bucket.
Anonymous users	Visitors who have not registered. If the permissions to access a bucket or an object are granted to anonymous users, everyone can access the object or bucket without identity authentication. **NOTICE** If the permissions to access a bucket or an object are granted to anonymous users, everyone can access the object or bucket without identity authentication.
Log delivery user groups **NOTE** Only the bucket ACL supports authorizing permissions to the log delivery user.	A log delivery user group only delivers access logs of buckets and objects to the configured target bucket. OBS does not create or upload any file to a bucket automatically. Therefore, if you want to record access logs for buckets, you need to grant the permission to a log delivery user group who will deliver the access logs to your specified target bucket. This user group is only used to record internal logs of OBS. **NOTICE** After logging is enabled, the log delivery user will be automatically granted the permission to read the bucket ACL and write the bucket where logs are saved. If you manually disable such permissions, bucket logging fails.

Bucket ACL

Table 2-7 lists the access permissions of a bucket ACL.

Table 2-7 Access permissions controlled by a bucket ACL

Permission Related Concepts	Option	Description
Access to Bucket	Read	A grantee with the read access to a bucket can obtain the list of objects in the bucket and the metadata of the bucket.
	Read	A grantee with this permission can obtain the object content and metadata.
	Write	A grantee with the write access to a bucket can upload, overwrite, and delete any object in the bucket.
Access to ACL	Read	A grantee with the read access to a bucket ACL can obtain the ACL of the bucket. The bucket owner has this permission permanently by default.
	Write	A grantee with the write access to a bucket ACL can update the ACL of the bucket. The bucket owner has this permission permanently by default.

Table 2-8 lists the access permissions of an object ACL.

Table 2-8 Access permissions controlled by an object ACL

Permission Related Concepts	Option	Description
Access to Object	Read	A grantee with the read access to an object can obtain the content and the metadata of the object.
Access to ACL	Read	A grantee with the read access to an object ACL can obtain the ACL of the object. The object owner has this permission permanently by default.
	Write	A grantee with the write access to an object ACL can update the ACL of the object. The object owner has this permission permanently by default.

NOTE

Every time you change the bucket or object access permission setting in an ACL, it overwrites the existing setting instead of adding a new access permission to the bucket or object.

Application Scenarios of Bucket ACLs

You can configure bucket ACLs to:

- Grant an account the read and write access to a bucket, so that data in the bucket can be shared or external buckets can be added. For example, after account **A** grants account **B** the read and write access to a bucket, account **B** can access the bucket by adding an external bucket through OBS Browser+ or using APIs.
- Grant the log delivery user group with the write access to the target bucket, so that access logs can be delivered to the target bucket.

Application Scenarios of Object ACLs

You can configure object ACLs to:

- Control access to objects. A bucket policy can control access to a single object or a set of objects. If you want to further separately control access to a single object in the set of objects for which a bucket policy has been configured, the object ACL is recommended.
- Access an object through a URL. Generally, if you want to grant anonymous users the permission to read an object through a URL, use the object ACL.

Configuring an ACL Using Header Fields

Access Control Policies

You can set an access control policy in a header when creating a bucket or uploading an object (for details about the examples, see **Creating a Bucket** and **Uploading Objects - PUT**). Only the access control policies predefined in OBS are available. The **x-obs-acl** is special, which can be configured with six types of permissions. No matter what type of permissions is configured, the owner has full control permission for the buckets or objects. The following table lists the predefined policies.

Table 2-9 Predefined access control policies in OBS

Policy	Description
private	Indicates that a bucket or object can be accessed only by its owner.
public-read	If this permission is set for a bucket, everyone can obtain the object list, multipart tasks, bucket metadata, and multiple object versions. If this permission is set for an object, everyone can obtain the content and metadata of the object.

Policy	Description
public-read-write	If this permission is configured for a bucket, everyone can obtain the object list, multipart uploads, bucket metadata, and object versions, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and cancel multipart uploads. If this permission is set for an object, everyone can obtain the content and metadata of the object.
public-read-delivered	If this permission is set for a bucket, everyone can obtain the object list, multipart tasks, bucket metadata, and multiple object versions, and obtain the content and metadata of the objects in the bucket. This permission does not apply to objects.
public-read-write-delivered	If this permission is configured for a bucket, everyone can obtain the object list, multipart uploads, bucket metadata, and object versions, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and cancel multipart uploads. Users can also obtain content and metadata of objects in the bucket. This permission does not apply to objects.
bucket-owner-full-control	If this permission is configured for an object, the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket of any other user, the bucket owner does not have the permissions on your object. After you grant this policy to the bucket owner, the bucket owner can have full control over your object.

NOTE

By default, the access control policy is **private**.

You can also use the following header fields to set access control policies when creating a bucket or uploading an object.

Table 2-10 Header fields for setting bucket or object ACLs

Header	Description
x-obs-grant-read	Used to grant the READ permission to all users in a specific account.
x-obs-grant-write	Used to grant the WRITE permission to all users in a specific account.
x-obs-grant-read-acp	Used to grant the READ_ACP permission to all users in a specific account.

Header	Description
x-obs-grant-write-acp	Used to grant the WRITE_ACP permission to all users in a specific account.
x-obs-grant-full-control	Used to grant the FULL_CONTROL permission to all users in a specific account.
x-obs-grant-read-delivered	Used to grant the READ permission for buckets and objects in the buckets to all users in a specific account, and objects inherit the permissions of their bucket. This permission does not apply to objects.
x-obs-grant-full-control-delivered	Used to grant the FULL_CONTROL permission for buckets and objects in the buckets to all users in a specific account, and objects inherit the permissions of their bucket. This permission does not apply to objects.

3 Access Requests

3.1 Accessing OBS Using Permanent Access Keys

OBS provides REST APIs that supports authenticated requests and anonymous requests. Anonymous requests are typically used for scenarios that require public access, such as accessing a hosted static website. In most scenarios, accessing OBS resources require authenticated requests. An authenticated request contains a signature value. The signature value is calculated based on the requester's access keys (a pair of AK and SK) as the encryption factor and the specific information carried by the request body. The signature calculation process is included in the SDK. You only need to prepare the access keys when initializing the SDK. Then the signature calculation is implemented automatically. However, if a client uses the REST APIs to develop a program to access OBS, the client needs to calculate the signature based on the signature algorithm defined by the OBS and add the signature to the request.

Users can create permanent access keys (a pair of AK and SK) on the **My Credentials** page.

- AK stands for the access key ID. It is the unique ID associated with the secret access key (SK). An AK is used together with an SK to encrypt and sign a request.
- They can identify a request sender and prevent the request from being modified.

An AK is also the unique identifier of an IAM user. OBS identifies a user based on its AK and SK, and then checks the permissions.

For details about how to obtain the permanent access keys, see **Obtaining Access Keys (AK/SK)**.

3.2 Accessing OBS Using Temporary Access Keys

Temporary Access Keys

OBS can be accessed through temporary access keys and the security token, which can be obtained on IAM. You can assign the temporary access keys (including the

security token) to a third-party application and an IAM user, so they can access OBS within a specified period of time.

You can obtain the temporary access keys and security token by calling the IAM API in **Obtaining a Temporary Access Key and Security Token Through a Token**.

Temporary AK/SK and security token comply with the least privilege principle and can be used to temporarily access OBS. When you use a temporary AK/SK pair to call an API for authentication, you must use the temporary AK/SK and security token at the same time and add the **x-obs-security-token** field to the request header.

Temporary access keys have the following advantages over permanent access keys of IAM users:

- Temporary access keys are valid for 15 minutes to 24 hours. You do not need to expose the permanent access keys of IAM users, reducing security risks.
- When obtaining temporary access keys, you can pass policy parameters to further restrict the temporary permissions granted to users. This ensures that IAM users can effectively control permissions granted to other users.

For details, see **User Signature Authentication**.

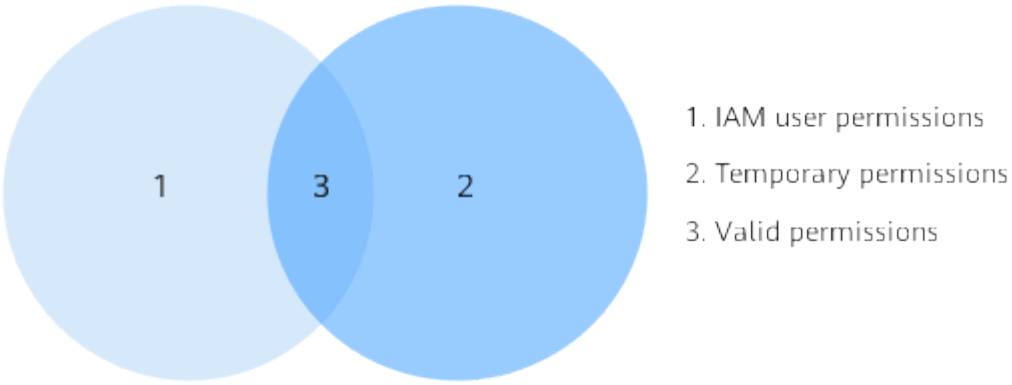
Permissions of the Temporary Access Keys

When an IAM user calls the IAM API in **Obtaining a Temporary Access Key and Security Token Through a Token**, the user can specify parameter **policy** to add a temporary policy for the temporary access keys to further restrict the permissions granted to other users. The format and content of a temporary policy are consistent with those specified in **IAM Permissions**.

- If policy parameters are not specified, no temporary policies are used. The temporary access keys inherit the IAM user's permissions.
- If policy parameters are specified, a temporary policy is enabled. Then the temporary access keys confine the granted permissions according to the temporary policy and the IAM user permissions.

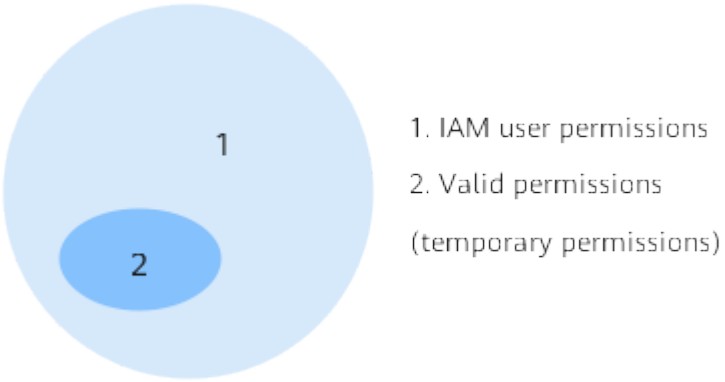
As shown in the following figure, circle 1 indicates the original permissions of an IAM user, and circle 2 indicates the temporary permissions specified by a temporary policy. The overlapped part 3 is the scope of permissions enabled by the temporary access keys.

Figure 3-1 Intersection of IAM user permissions and temporary policy permissions



Temporary access keys comply with the least privilege principle. Configure a temporary policy within the original permission scope of an IAM user. Otherwise you may be confused about why permissions enabled by a temporary policy are not effective. As illustrated by the following figure, the finally effective permissions are the authorized temporary permissions.

Figure 3-2 Restricting temporary permissions within the scope of IAM user permissions



A temporary policy authentication starts from the Deny statements. Unspecified permissions are denied by default.

NOTE

Therefore, you are advised to specify only the allowed permission.

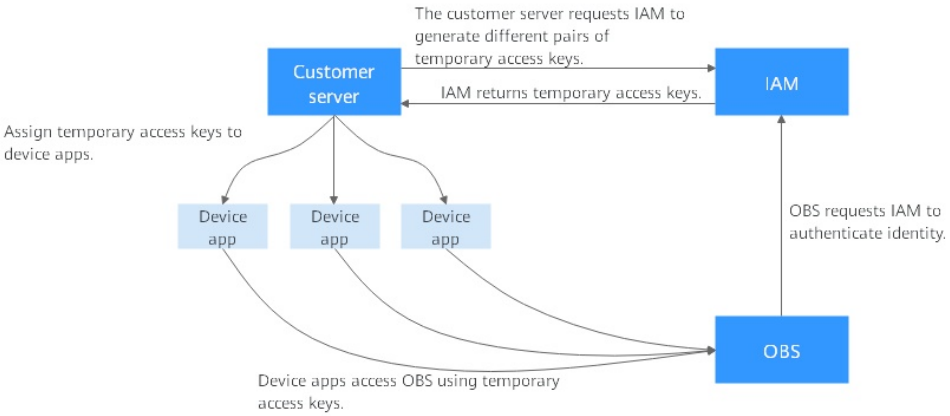
Application Scenarios

Temporary access keys are used to authorize third parties to temporarily access OBS. For example, some companies have their user management systems, which manage device app users and local enterprise users. These users do not have IAM user permissions, so IAM users can grant temporary access keys to these users when they need to access OBS.

Typical application scenario:

A company has a large number of device apps that need to access OBS. Different apps represent different end users who require different access permissions. In this case, temporary access keys can be used to access OBS.

Figure 3-3 Application scenarios of temporary access keys



1. If the customer's server can obtain permanent access keys for IAM users, the server can send requests to IAM to generate different temporary access keys for different apps.

   IAM users can obtain the temporary access keys and security token by calling the IAM API in **Obtaining a Temporary Access Key and Security Token Through a Token**. When calling this API, pass the **policy** parameter to set a temporary policy. An example is provided as follows:

   ```
   {
       "auth": {
           "identity": {
               "methods": [
                   ... ...
               ],
               "policy": {
                   ... ...
               }
           }
       }
   }
   ```

   The policy's syntax and format are the same as those specified in **IAM Permissions**. For details, see **Permissions and Supported Actions**.

2. IAM generates temporary access keys with different permissions and validity periods based on the passed policy parameters and returns the access keys to the customer server.

3. Then the customer server distributes the temporary access keys to device apps that require such permissions.
4. A device app can use the temporary access keys to access OBS through OBS SDKs or APIs. Temporary access keys are valid for a short period of time. If the device app needs to prolong its use of OBS, it should send a request to the customer server for updating temporary access keys before they expire.

Configuration Example

For details, see **Granting Temporary Access to OBS**.

3.3 Accessing OBS Using a Temporary URL

You can use a temporary URL to access OBS and perform operations such as bucket creation or object upload and download. This section describes how to share objects using a temporary URL.

Sharing Objects

You can share objects (files or folders) stored in OBS with all users within a specified period.

Sharing a file

All URLs generated during file sharing are temporary and remain valid for a limited period of time.

A temporary URL uses V4 temporarily authorized requests. The following is a temporary URL sample:

```
https://oss.regionid.prod-cloud-ocb.orange-business.com/bucketname/objectname?X-Amz-Algorithm=xxx&X-Amz-Credential=xxx&X-Amz-Date=xxx&X-Amz-Expires=900&X-Amz-Signature=xxx&X-Amz-SignedHeaders=xxx&response-content-disposition=xxx
```

After you share an object by choosing **More** > **Copy Object URL** on OBS Console, the system will generate a URL that contains the temporary authentication information, valid for 900 seconds since its generation by default. Each time you click **Copy Object URL**, OBS will obtain the authentication information again to generate a new sharing URL whose validity period is reset.

Sharing a folder

Folder sharing is temporary and has a validity period. You need to prepare a six-digit extraction code before sharing a folder. After the sharing task is created, OBS aggregates the download links of all objects in the folder to a static website that is hosted by a public OBS bucket. Then anyone who has the created temporary URL and access code can access the static website and download the shared files.

Limitations and Constraints

- The validity period of files shared through OBS Console is fixed at 900s. If you want a file to be accessed permanently, you can **configure a bucket policy to grant the public read permission on the file to anonymous users**.
- A folder shared through OBS Console is valid for one minute to 18 hours. If you need a longer validity period for a shared folder, use the client tool OBS

Browser+ that allows a validity period of up to one year. If you want a folder to be accessed permanently, you can **configure a bucket policy to grant the public read permission on the folder to anonymous users**.

- Only buckets 3.0 support file and folder sharing. You can view the bucket version in the **Basic Information** area on the **Overview** page of a bucket.
- To share a cold object, restore it first.

Configuration Procedure

For details about how to share files, see **Temporarily Sharing Objects with Anonymous Users**.

3.4 Accessing OBS Using an IAM Agency

The IAM agency is a function of Identity and Access Management (IAM). In some OBS application scenarios (such as CDN private bucket retrieval and cross-region replication), IAM agencies are required to grant other users or cloud services the permission to access OBS and manage OBS resources for the delegating party, thus implementing secure and efficient agent maintenance.

For details about IAM agencies, see **Identity and Access Management User Guide**.

4 Typical Permission Control Scenarios

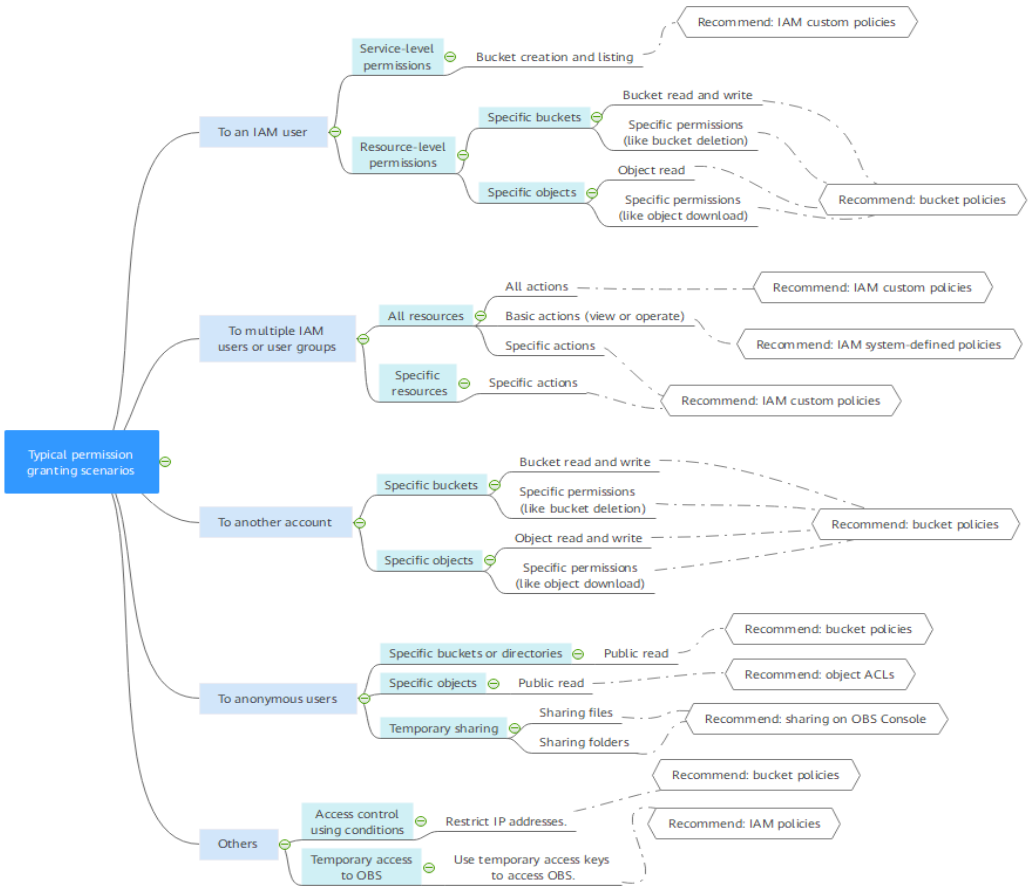
The following typical scenarios are provided to help you better configure OBS permission control.

Factors to consider before configuring permission control:

1. **Who are granted**: Grantees can be a single IAM user, multiple IAM users or user groups, other accounts, and anonymous users.
2. **What resources will be accessed**: Such resources can be all OBS resources (requiring service-level permissions), specified buckets, and specified objects.
3. **What permissions are granted**: In addition to configure basic permissions, such as read and read/write permissions, you can also customize permissions based on your needs.

OBS provides various permission control mechanisms for different scenarios. The following figure can help you quickly find the best method that matches your requirements.

Figure 4-1 Typical permission scenarios



The following table lists the permission control cases in typical scenarios for your reference.

Table 4-1 Configuration cases in typical scenarios

Scenario	Configuration Case
Granting permissions to an IAM user under the current account	**Granting an IAM User the Permissions Required to List and Create Buckets**
	Granting an IAM User the Read/Write Permission for a Bucket
	Granting an IAM User the Specified Permissions for a Bucket
	Granting an IAM User the Read Permission for Specific Objects
	Granting an IAM User the Specified Permissions for Certain Objects

Scenario	Configuration Case
Granting permissions to multiple IAM users or user groups under the current account	**Granting IAM User Groups All Permissions for All OBS Resources**
	Granting IAM User Groups Basic Permissions for All OBS Resources
	Granting IAM User Groups the Specified Permissions for All OBS Resources
	Granting IAM User Groups the Specified Permissions for Certain OBS Resources
Granting permissions to other accounts	**Granting Other Accounts the Read/Write Permission for a Bucket**
	Granting Other Accounts the Specified Permissions for a Bucket
	Granting IAM Users Under an Account the Access to a Bucket and the Resources in It
	Granting Other Accounts the Read Permission for Certain Objects
	Granting Other Accounts the Specified Permissions for Certain Objects
Granting permissions to anonymous users	**Granting Anonymous Users the Public Read Permission for a Bucket**
	Granting Anonymous Users the Read Permission for a Directory
	Granting Anonymous Users the Read Permission for Certain Objects
	Temporarily Sharing Objects with Anonymous Users
Granting temporary permissions	**Granting Temporary Access to OBS**
Restricting access to specified IP addresses	**Restricting Access to a Bucket for Specific IP Addresses**

5 Configuration Cases in Typical Permission Control Scenarios

5.1 Granting Permissions to an IAM User Under the Current Account

5.1.1 Granting an IAM User the Permissions Required to List and Create Buckets

Scenario

This topic describes how to grant an IAM user the permissions required to create and list buckets. An IAM user with this permission can create buckets. The created buckets are still owned by the account of the IAM user. The IAM user can view all buckets under the account.

Recommended Configuration

Permissions to create and list buckets are at OBS service-level, which can be implemented only through IAM. You are advised to use IAM custom policies.

Procedure

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 In the navigation pane, choose **Policies**.

Step 4 Click **Create Custom Policy** in the upper right corner.

Step 5 Configure parameters for a custom policy.

Table 5-1 Parameters for configuring a custom policy

Parameter	Description
Policy Name	Name of the custom policy
Policy View	Set this parameter based on your own habits. **Visual editor** is used here.
Policy Content	• Select **Allow**. • Select **Object Storage Service (OBS)**. • Select **obs:bucket:CreateBucket** from **ReadWrite** actions and **obs:bucket:ListAllMyBuckets** from **ListOnly** actions. • Select **All** for resources.
Scope	The default value is **Global services**.

Step 6 Click **OK**. The custom policy is created.

Step 7 **Create a user group and assign permissions**.

Add the created custom policy to the user group by following the instructions in the IAM document.

Step 8 Add the IAM user you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for a custom policy to take effect after the authorization.

----End

5.1.2 Granting an IAM User the Read/Write Permission for a Bucket

Scenario

This topic describes how to grant an IAM user the read/write permission for an OBS bucket.

Recommended Configuration

You are advised to use bucket policies to grant resource-level permissions to an IAM user.

Configuration Precautions

After the configuration is complete, read and write operations (uploading, downloading, and deleting all objects in the bucket) can be performed using APIs or SDKs. However, if you log in to OBS Console or OBS Browser+ to perform those

operations, an error is reported indicating that you do not have required permissions. .

If you want an IAM user to perform read and write operations on OBS Console or OBS Browser+, configure custom IAM policies by referring to **Follow-up Procedure**.

After the configuration is complete, the system still displays a message indicating that you do not have the permission to access the bucket. This is normal because the console invokes other advanced configuration APIs, but you can still perform operations allowed in read/write mode.

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 Configure parameters for a bucket policy.

Table 5-2 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Read and write**.
Principal	• Select **Include** > **Current account**. • **Username**: Select an IAM user whom you want to grant permissions to.
Resources	• **Include** • **Resource Name**: Enter ***.

Step 6 Click **OK**. The bucket policy is created.

----End

Follow-up Procedure

To perform read and write operations on OBS Console or OBS Browser+, you must add the **obs:bucket:ListAllMyBuckets** (for listing buckets) and **obs:bucket:ListBucket** (for listing objects in a bucket) permissions to the custom IAM policy.

NOTE

obs:bucket:ListAllMyBuckets applies to all resources, while **obs:bucket:ListBucket** applies to the authorized bucket only. Therefore, you need to add two permissions to the policy.

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 In the navigation pane, choose **Policies**.

Step 4 Click **Create Custom Policy** in the upper right corner.

Step 5 Configure parameters for a custom policy.

Table 5-3 Parameters for configuring a custom policy

Parameter	Description
Policy Name	Name of the custom policy
Policy View	Set this parameter based on your own habits. **Visual editor** is used here.
Policy Content	[Permission 1] • Select **Allow**. • Select **Object Storage Service (OBS)**. • Select **obs:bucket:ListAllMyBuckets** from the actions. • Select **All** for resources. [Permission 2] • Select **Allow**. • Select **Object Storage Service (OBS)**. • Select **obs:bucket:ListBucket** from the actions. • For **Resources**, select **Specific**, and for **bucket**, select **Specify resource path**, and click **Add Resource Path**. Enter the bucket name in the **Path** text box, indicating that the policy takes effect only for this bucket.
Scope	The default value is **Global services**.

Step 6 Click **OK**. The custom policy is created.

Step 7 **Create a user group and assign permissions**.

Add the created custom policy to the user group by following the instructions in the IAM document.

Step 8 Add the IAM user you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for a custom policy to take effect after the authorization.

----End

5.1.3 Granting an IAM User the Specified Permissions for a Bucket

Scenario

This topic describes how to grant an IAM user the permissions required to perform specific operations on an OBS bucket. Below describes how to grant the bucket deletion permission.

If you need to configure other permissions, select the corresponding actions from the **Action Name** drop-down list in the bucket policy. For details about the actions supported by OBS, see **Action/NotAction**.

Recommended Configuration

You are advised to use bucket policies to grant resource-level permissions to an IAM user.

Configuration Precautions

After the configuration is complete, you can delete buckets using APIs. However, if you log in to OBS Console or OBS Browser+ to delete buckets, an error is reported indicating that you do not have required permissions.

This is because when you log in to OBS Console or OBS Browser+, more APIs (such as **ListAllMyBuckets** and **ListBucketVersions**) are called to load the list of buckets and versioned objects, but your permissions do not cover those APIs. In such case, your access is denied or your operation is not allowed.

If you want an IAM user to delete buckets on OBS Console or OBS Browser+, allow the **ListBucketVersions** permission in the bucket policy and configure a custom IAM policy to grant the **ListAllMyBuckets** permission by referring to **Follow-up Procedure**.

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 Configure parameters for a bucket policy.

Table 5-4 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Customized**.
Effect	Select **Allow**.

Parameter	Description
Principal	• Select **Include** > **Current account**. • **Username**: Select an IAM user whom you want to grant permissions to.
Resources	Select **Include** > **Entire bucket**.
Actions	• **Include** • **Action Name**: – DeleteBucket – ListBucketVersions (required when the authorized user needs to access OBS on OBS Console or OBS Browser+) To configure other permissions, select the corresponding actions. For details about the actions supported by OBS, see **Action/NotAction**.

Step 6 Click **OK**. The bucket policy is created.

----End

Follow-up Procedure

To successfully delete buckets on OBS Console or OBS Browser+, you need to allow the **obs:bucket:ListAllMyBuckets** (for listing buckets) permission in the IAM policy.

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 In the navigation pane, choose **Policies**.

Step 4 Click **Create Custom Policy** in the upper right corner.

Step 5 Configure parameters for a custom policy.

Table 5-5 Parameters for configuring a custom policy

Parameter	Description
Policy Name	Name of the custom policy
Policy View	Set this parameter based on your own habits. **Visual editor** is used here.
Policy Content	• Select **Allow**. • Select **Object Storage Service (OBS)**. • Select **obs:bucket:ListAllMyBuckets** from the actions. • Select **All** for resources.

Parameter	Description
Scope	The default value is **Global services**.

Step 6 Click **OK**. The custom policy is created.

Step 7 **Create a user group and assign permissions**.

Add the created custom policy to the user group by following the instructions in the IAM document.

Step 8 Add the IAM user you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for a custom policy to take effect after the authorization.

----End

5.1.4 Granting an IAM User the Read Permission for Specific Objects

Scenario

This topic describes how to grant an IAM user the read permission for an object or a set of objects in an OBS bucket.

Recommended Configuration

You are advised to use bucket policies to grant resource-level permissions to an IAM user.

Configuration Precautions

After the configuration is complete, you can read (download) specific objects using APIs. However, if you download an object from OBS Console or OBS Browser +, an error is reported indicating that you do not have required permissions.

This is because when you log in to OBS Console or OBS Browser+, the **ListAllMyBuckets** API is called to load the bucket list and some other APIs will also be called on other pages, but your permissions do not cover those APIs. In such case, your access is denied or your operation is not allowed.

If you want an IAM user to perform read operations on OBS Console or OBS Browser+, configure custom IAM policies by referring to **Follow-up Procedure**.

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 Configure parameters for a bucket policy.

Table 5-6 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Read-only**.
Principal	• Select **Include** > **Current account**. • **Username**: Select an IAM user whom you want to grant permissions to.
Resources	• **Include** • **Resource Name**: Enter the object or the set of objects that will be accessed. For one object, enter *object name*. For a set of objects, enter *object name prefix + *, * + object name suffix, or **.

Step 6 Click **OK**. The bucket policy is created.

----End

Follow-up Procedure

To perform read operations on OBS Console or OBS Browser+, you must add the **obs:bucket:ListAllMyBuckets** (for listing buckets) and **obs:bucket:ListBucket** (for listing objects in a bucket) permissions to the custom IAM policy.

NOTE

obs:bucket:ListAllMyBuckets applies to all resources, while **obs:bucket:ListBucket** applies to the authorized bucket only. Therefore, you need to add two permissions to the policy.

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 In the navigation pane, choose **Policies**.

Step 4 Click **Create Custom Policy** in the upper right corner.

Step 5 Configure parameters for a custom policy.

Table 5-7 Parameters for configuring a custom policy

Parameter	Description
Policy Name	Name of the custom policy

Parameter	Description
Policy View	Set this parameter based on your own habits. **Visual editor** is used here.
Policy Content	[Permission 1] • Select **Allow**. • Select **Object Storage Service (OBS)**. • Select **obs:bucket:ListAllMyBuckets** from the actions. • Select **All** for resources. [Permission 2] • Select **Allow**. • Select **Object Storage Service (OBS)**. • Select **obs:bucket:ListBucket** from the actions. • For **Resources**, select **Specific**, and for **bucket**, select **Specify resource path**, and click **Add Resource Path**. Enter the bucket name in the **Path** text box, indicating that the policy takes effect only for this bucket.
Scope	The default value is **Global services**.

Step 6 Click **OK**. The custom policy is created.

Step 7 **Create a user group and assign permissions**.

Add the created custom policy to the user group by following the instructions in the IAM document.

Step 8 Add the IAM user you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for a custom policy to take effect after the authorization.

----End

5.1.5 Granting an IAM User the Specified Permissions for Certain Objects

Scenario

This topic describes how to grant an IAM user the specified permissions on certain objects in a bucket. Below explains how to grant the object download permission.

If you need to configure other permissions, select the corresponding actions from the **Action Name** drop-down list in the bucket policy. For details about the actions supported by OBS, see **Action/NotAction**.

Recommended Configuration

You are advised to use bucket policies to grant resource-level permissions to an IAM user.

Configuration Precautions

After the configuration is complete, you can download objects using APIs. However, if you log in to OBS Console or OBS Browser+ to download an object, an error is reported indicating that you do not have required permissions.

This is because when you log in to OBS Console or OBS Browser+, APIs (such as **ListAllMyBuckets** and **ListBucket**) are called to load the bucket list and object list and some other APIs will also be called on other pages, but your permissions do not cover those APIs. In such case, your access is denied or your operation is not allowed.

If you want an IAM user to successfully download objects on OBS Console or OBS Browser+, configure custom IAM policies by referring to **Follow-up Procedure**.

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 Configure parameters for a bucket policy.

Table 5-8 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Customized**.
Effect	Select **Allow**.
Principal	• Select **Include** > **Current account**. • **Username**: Select an IAM user whom you want to grant permissions to.
Resources	• Choose **Include** > **Specific resources**. • **Resource Name**: Enter the object or the set of objects that will be accessed. For one object, enter *object name*. For a set of objects, enter *object name prefix + *, * + object name suffix, or **.

Parameter	Description
Actions	• **Include** • Action Name: Select **GetObject**. To configure other permissions, select the corresponding actions. For details about the actions supported by OBS, see **Action/NotAction**.

Step 6 Click **OK**. The bucket policy is created.

----End

Follow-up Procedure

To perform specific operations on OBS Console or OBS Browser+, you must add the **obs:bucket:ListAllMyBuckets** (for listing buckets) and **obs:bucket:ListBucket** (for listing objects in a bucket) permissions to the custom IAM policy.

NOTE

obs:bucket:ListAllMyBuckets applies to all resources, while **obs:bucket:ListBucket** applies to the authorized bucket only. Therefore, you need to add two permissions to the policy.

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 In the navigation pane, choose **Policies**.

Step 4 Click **Create Custom Policy** in the upper right corner.

Step 5 Configure parameters for a custom policy.

Table 5-9 Parameters for configuring a custom policy

Parameter	Description
Policy Name	Name of the custom policy
Policy View	Set this parameter based on your own habits. **Visual editor** is used here.

Parameter	Description
Policy Content	[Permission 1] • Select **Allow**. • Select **Object Storage Service (OBS)**. • Select **obs:bucket:ListAllMyBuckets** from the actions. • Select **All** for resources. [Permission 2] • Select **Allow**. • Select **Object Storage Service (OBS)**. • Select **obs:bucket:ListBucket** from the actions. • For **Resources**, select **Specific**, and for **bucket**, select **Specify resource path**, and click **Add Resource Path**. Enter the bucket name in the **Path** text box, indicating that the policy takes effect only for this bucket.
Scope	The default value is **Global services**.

Step 6 Click **OK**. The custom policy is created.

Step 7 **Create a user group and assign permissions**.

Add the created custom policy to the user group by following the instructions in the IAM document.

Step 8 Add the IAM user you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for a custom policy to take effect after the authorization.

----End

5.2 Granting Permissions to Multiple IAM Users or User Groups Under the Current Account

5.2.1 Granting IAM User Groups All Permissions for All OBS Resources

Scenario

This topic describes how to grant multiple IAM users or user groups all permissions for all OBS resources. Users with this permission can perform any OBS operation.

Recommended Configuration

IAM custom policies

Procedure

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 In the navigation pane, choose **Policies**.

Step 4 Click **Create Custom Policy** in the upper right corner.

Step 5 Configure parameters for a custom policy.

Table 5-10 Parameters for configuring a custom policy

Parameter	Description
Policy Name	Name of the custom policy
Policy View	Set this parameter based on your own habits. **Visual editor** is used here.
Policy Content	• Select **Allow**. • Select **Object Storage Service (OBS)**. • Select all actions. • Select **All** for resources.
Scope	The default value is **Global services**.

Step 6 Click **OK**. The custom policy is created.

Step 7 **Create a user group and assign permissions**.

Add the created custom policy to the user group by following the instructions in the IAM document.

Step 8 Add the IAM user you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for a custom policy to take effect after the authorization.

----End

5.2.2 Granting IAM User Groups Basic Permissions for All OBS Resources

Scenario

This topic describes how to use the OBS-related system roles and policies preset in IAM to grant basic operation permissions for all OBS resources to multiple IAM

users or user groups. The following table lists the permissions supported by preset system roles and policies.

Table 5-11 OBS system permissions

Role/Policy Name	Description	Type
Tenant Administrator	Users with this permission can perform all operations on all services except IAM.	System-defined role
Tenant Guest	Users with this permission can perform read-only operations on all services except IAM.	System-defined role
OBS FullAccess	Users with this permission are OBS administrators and can perform any operations on all OBS resources under the account.	System-defined policy
OBS Buckets Viewer	Users with this permission can list buckets, obtain basic bucket information, and obtain bucket metadata.	System-defined policy
OBS ReadOnlyAccess	Users with this permission can list buckets, obtain basic bucket information, obtain bucket metadata, and list objects (not the objects that have been versioned). **NOTE** If a user with this permission fails to list objects on OBS Console, there may be multiple versions of objects in the bucket. In this case, you need to grant the user the **obs:bucket:ListBucketVersions** permission so that the user can view different versions of objects on OBS Console.	System-defined policy
OBS OperateAccess	Users with this permission can perform all OBS ReadOnlyAccess operations and perform basic object operations, such as uploading objects, downloading objects, deleting objects, and obtaining object ACLs. **NOTE** If a user with this permission fails to list objects on OBS Console, there may be multiple versions of objects in the bucket. In this case, you need to grant the user the **obs:bucket:ListBucketVersions** permission so that the user can view different versions of objects on OBS Console.	System-defined policy

Recommended Configuration

IAM system roles and policies

Configuration Precautions

After a system role or policy is configured according to this case, if you log in to the system using OBS Console or OBS Browser+, a message may be displayed indicating that you do not have the permission.

Authorized permissions are valid, though operations on the console or client are restricted. You can call the APIs directly.

With **OBS OperateAccess** configured, you can upload or download objects on OBS Console or OBS Browser+.

Procedure

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 **Create a user group and assign permissions**.

Add system roles or policies that meet the service scenario requirements to the user group by following the instructions provided in the IAM document.

Step 4 Add the IAM user you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for the configured permissions to take effect.

----End

5.2.3 Granting IAM User Groups the Specified Permissions for All OBS Resources

Scenario

This topic describes how to grant multiple IAM users or user groups specified permissions for all OBS resources.

Recommended Configuration

IAM custom policies

Configuration Precautions

After the configuration is complete, you can perform allowed operations using APIs. However, if you log in to OBS Console or OBS Browser+ to perform those operations, an error is reported indicating that you do not have required permissions.

This is because when you log in to OBS Console or OBS Browser+, APIs (such as **ListAllMyBuckets** and **ListBucket**) are called to load the bucket list and object list and some other APIs will also be called on other pages, but your permissions do

not cover those APIs. In such case, your access to OBS Console or OBS Browser+ is denied or your operation is not allowed.

To allow IAM users to operate buckets and objects on OBS Console or OBS Browser+, add at least the **obs:bucket:ListAllMyBuckets** and **obs:bucket:ListBucket** permissions to the custom policy.

Procedure

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 In the navigation pane, choose **Policies**.

Step 4 Click **Create Custom Policy** in the upper right corner.

Step 5 Configure parameters for a custom policy.

Table 5-12 Parameters for configuring a custom policy

Parameter	Description
Policy Name	Name of the custom policy
Policy View	Set this parameter based on your own habits. **Visual editor** is used here.
Policy Content	• Select **Allow**. • Select **Object Storage Service (OBS)**. • Select the actions to be authorized. • Select **All** for resources.
Scope	The default value is **Global services**.

Step 6 Click **OK**. The custom policy is created.

Step 7 **Create a user group and assign permissions**.

Add the created custom policy to the user group by following the instructions in the IAM document.

Step 8 Add the IAM user you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for a custom policy to take effect after the authorization.

----End

5.2.4 Granting IAM User Groups the Specified Permissions for Certain OBS Resources

Scenario

This topic describes how to grant specified operation permissions for certain OBS resources (can be a bucket or an object) to multiple IAM users or user groups.

Recommended Configuration

IAM custom policies

Configuration Precautions

After the configuration is complete, you can perform allowed operations using APIs. However, if you log in to OBS Console or OBS Browser+ to perform those operations, an error is reported indicating that you do not have required permissions.

This is because when you log in to OBS Console or OBS Browser+, APIs (such as **ListAllMyBuckets** and **ListBucket**) are called to load the bucket list and object list and some other APIs will also be called on other pages, but your permissions do not cover those APIs. In such case, your access to OBS Console or OBS Browser+ is denied or your operation is not allowed.

To allow IAM users to operate buckets and objects on OBS Console or OBS Browser+, add at least the **obs:bucket:ListAllMyBuckets** and **obs:bucket:ListBucket** permissions to the custom policy.

NOTE

obs:bucket:ListAllMyBuckets applies to all resources. You need to select all resources.

obs:bucket:ListBucket applies only to the authorized bucket. You can select all resources or a specified bucket as needed.

Procedure

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 In the navigation pane, choose **Policies**.

Step 4 Click **Create Custom Policy** in the upper right corner.

Step 5 Configure parameters for a custom policy.

Table 5-13 Parameters for configuring a custom policy

Parameter	Description
Policy Name	Name of the custom policy

Parameter	Description
Policy View	Set this parameter based on your own habits. **Visual editor** is used here.

Parameter	Description
Policy Content	[Permission 1] It is mandatory when an authorized user needs to perform operations on OBS Console or OBS Browser+. • Select **Allow**. • Select **Object Storage Service (OBS)**. • Select **obs:bucket:ListAllMyBuckets** from the actions. • Select **All** for resources. [Permission 2] • Select **Allow**. • Select **Object Storage Service (OBS)**. • Select the actions to be authorized. • Choose **Specific resources** > **Bucket** to specify bucket resources. [Format] **obs:*:*:bucket:***bucket name* [Note] For bucket resources, IAM automatically generates the prefix of the resource path: **obs:*:*:bucket:**. For the path of a specific bucket, add the *bucket name* to the end. You can also add a wildcard character (*) to indicate any bucket. Examples are given as follows: – **obs:*:*:bucket:*** (indicating any OBS bucket) – **obs:*:*:bucket:examplebucket** (indicating that the policy applies to bucket **examplebucket**) To perform operations on OBS Console or OBS Browser +, grant the **obs:bucket:ListBucket** permission to a specified bucket. • Choose **Specific resources** > **Object** to specify an object resource. [Format] Objects in a specified directory: **obs:*:*:object:***Bucket name*/*Prefix*/*** Specified object: **obs:*:*:object:***Bucket name*/*Object name* [Note] For object resources, IAM automatically generates the prefix of the resource path: **obs:*:*:object:** For the path of a specific object, add the *bucket name/object name* to the end. You can also add a wildcard character (*) to indicate any object in a bucket. Examples are given as follows:

Parameter	Description
	– **obs:*:*:object:my-bucket/my-object/*** (indicating any object in the **my-object** directory of bucket **my-bucket**) – **obs:*:*:object:my-bucket/exampleobject** (indicating object **exampleobject** in bucket **my-bucket**)
Scope	The default value is **Global services**.

Step 6 Click **OK**. The custom policy is created.

Step 7 **Create a user group and assign permissions**.

Add the created custom policy to the user group by following the instructions in the IAM document.

Step 8 Add the IAM user you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for a custom policy to take effect after the authorization.

----End

5.2.5 Granting IAM User Groups the Specified Permissions for a Folder

Scenario

This topic describes how to grant specified permissions for a folder in an OBS bucket to multiple IAM users or user groups.

Recommended Configuration

IAM custom policies

Configuration Precautions

After the configuration is complete, you can perform allowed operations using APIs. However, if you log in to OBS Console or OBS Browser+ to perform those operations, an error is reported indicating that you do not have required permissions.

This is because when you log in to OBS Console or OBS Browser+, APIs (such as **ListAllMyBuckets** and **ListBucket**) are called to load the bucket list and object list and some other APIs will also be called on other pages, but your permissions do not cover those APIs. In such case, your access to OBS Console or OBS Browser+ is denied or your operation is not allowed.

To allow IAM users to operate buckets and objects on OBS Console or OBS Browser+, add at least the **obs:bucket:ListAllMyBuckets** and

obs:bucket:ListBucket permissions to the custom policy. (In this case, these two permissions are configured in permission 2 and 3.)

NOTE

obs:bucket:ListAllMyBuckets applies to all resources. You need to select all resources.

obs:bucket:ListBucket applies only to the authorized bucket. You can select all resources or a specified bucket as needed.

Procedure

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 In the navigation pane, choose **Policies**.

Step 4 Click **Create Custom Policy** in the upper right corner.

Step 5 Configure parameters for a custom policy.

Table 5-14 Parameters for configuring a custom policy

Parameter	Description
Policy Name	Name of the custom policy
Policy View	Set this parameter based on your own habits. **Visual editor** is used here.

Parameter	Description
Policy Content	[Permission 1] • Select **Allow**. • Select **Object Storage Service (OBS)**. • Select all the object-related permissions under **ReadOnly**, **ReadWrite**, and **Permissions**. • On the **All** tab, choose **Specific** > **Specify resource path** to specify a folder. [Path Format] **obs:*:*:object:***Bucket name*/*Folder name*/*** [Notes] For bucket resources, IAM automatically generates the prefix of the resource path **obs:*:*:object:**. You can add *Bucket name/Object name* at the end of the generated path prefix to specify a resource path. Wildcards (*) are also supported. For example, **OBS:*:*:object:example-002/folder-001/*** indicates any object in folder **folder-001** of bucket **example-002**. [Permission 2] It is mandatory when an authorized user needs to perform operations on OBS Console or OBS Browser+. • Select **Allow**. • Select **Object Storage Service (OBS)**. • Select **obs:bucket:ListBucket** from the actions. • On the **All** tab, choose **Specific** > **Specify resource path** to specify a bucket. [Path Format] **obs:*:*:bucket:***Bucket name* • On the **(Optional) Add request condition** tab, click **Add Request Condition**. – **Condition key**: Select **obs:prefix** from the drop-down list. – **Operator**: Select **StringMatch** from the drop-down list. – **Value**: *Folder name*/ [Notes] If you want a user to have only the permission to list a folder in the bucket, add a request condition for action **obs:bucket:ListBucket**. **prefix** is included in the request for listing objects in a bucket. In this way, when you specify **prefix** to list objects whose names start with *Folder name*/, the objects in the bucket can be listed. [Permission 3] It is mandatory when an authorized user needs to perform operations on OBS Console or OBS Browser+.

Parameter	Description
	• Select **Allow**. • Select **Object Storage Service (OBS)**. • Select **obs:bucket:ListAllMyBuckets** under **ListOnly**. • Select **All** for **Resources**.
Scope	The default value is **Global services**.

Step 6 Click **OK**. The custom policy is created.

Step 7 **Create a user group and assign permissions**.

Add the created custom policy to the user group by following the instructions in the IAM document.

Step 8 Add the IAM user you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for a custom policy to take effect after the authorization.

----End

Verification

Step 1 Log in to OBS Console as an IAM user.

Step 2 In the bucket list, click bucket **example-002** to go to the overview page.

NOTE

After the configuration is complete, it is normal if the system still displays a message indicating that you do not have required permissions, because OBS Console also calls other APIs for advanced settings, but you can still perform the operations allowed on the folder.

Step 3 In the navigation pane, select **Objects**. It is normal that a message indicating no permission is displayed and no object can be viewed.

NOTE

The reason why there is no required permission is that listing objects on OBS Console is to list objects in the root folder. This rule does not match the configured custom policy for listing objects in folder **folder-001/**.

Step 4 In the search box, enter **folder-001/** to view the list of objects in **folder-001**. Objects **222.txt** and **111.txt** are displayed.

Step 5 Click **Create Folder** to create folder **folder-002**.

Step 6 Click **Upload Object** to upload file **333.txt**.

NOTE

If some other permissions are required, hover your cursor over the username and choose **Identity and Access Management** > **Permissions**, and then repeat the operations above to configure custom policies as needed.

----End

5.3 Granting Permissions to Other Accounts

5.3.1 Granting Other Accounts the Read/Write Permission for a Bucket

Scenario

This topic describes how to grant other accounts (excluding the IAM users under them) the read/write permission for OBS buckets. For details about how to grant permissions to an IAM user, see **Granting IAM Users Under an Account the Access to a Bucket and the Resources in It**.

Recommended Configuration

You are advised to use bucket policies to grant permissions to other accounts.

Configuration Precautions

After the configuration is complete, the authorized account can perform read and write operations (upload, download, or delete all objects in a bucket) by using APIs or by adding external buckets through OBS Browser+. Currently, access to buckets of other accounts is not allowed on OBS Console.

When you use OBS Browser+ to access the added external bucket, a message may still be displayed indicating that you do not have required permissions.

Error cause: The loading on the OBS Browser+ bucket details page invokes some other OBS APIs. However, such operations are not allowed by the read and write permissions. Therefore, a message "Access denied. Check the response permission" or "This operation is not allowed on the requested resource" is displayed, however, existing permissions are not affected.

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 Configure parameters for a bucket policy.

Table 5-15 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Read and write**.
Principal	• Select **Include** > **Other account**. • **Account ID**: Enter the ID of the account which you want to grant permissions to. You can obtain it from the **My Credentials** page of the account. • **User ID**: Enter the account ID, which can be obtained from the **My Credentials** page of the account. **NOTE** In this example, permissions are granted to an account, excluding any IAM user under the account. Therefore, the user ID is the same as the account ID.
Resources	• **Include** • Resource Name: Enter *.

Step 6 Click **OK**. The bucket policy is created.

Step 7 (Optional) Click **Create Bucket Policy** again.

If the authorized account wants to access the OBS bucket on OBS Browser+ by mounting an external bucket, you need to add a ListBucket permission.

Step 8 (Optional) Configure the ListBucket permission.

Table 5-16 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Customized**.
Effect	Select **Allow**.
Principal	• Select **Include** > **Other account**. • **Account ID**: Enter the ID of the account which you want to grant permissions to. You can obtain it from the **My Credentials** page of the account. • **User ID**: Enter the account ID. **NOTE** In this example, permissions are granted to an account, excluding any IAM user under the account. Therefore, the user ID is the same as the account ID.
Resources	Select **Include** > **Entire bucket**.
Actions	• **Include** • **Action Name**: ListBucket

Step 9 (Optional) Click **OK**. The bucket policy is created.

----End

5.3.2 Granting Other Accounts the Specified Permissions for a Bucket

Scenario

This topic describes how to grant other accounts (excluding the IAM users under them) specific operation permissions for OBS buckets. For details about how to grant permissions to an IAM user, see **Granting IAM Users Under an Account the Access to a Bucket and the Resources in It**.

The following example explains how to grant the permissions to configure a bucket ACL and obtain the bucket ACL configuration information. If you need to configure other permissions, select the corresponding actions from the **Action Name** drop-down list in the bucket policy. For details about the actions supported by OBS, see **Action/NotAction**.

Recommended Configuration

You are advised to use bucket policies to grant permissions to other accounts.

Configuration Precautions

After the configuration is complete, the authorized account can configure and obtain a bucket ACL by using APIs or SDKs or by adding external buckets through OBS Browser+. To do this by adding external buckets, the **ListBucket** permission is also required. Currently, access to buckets of other accounts is not allowed on OBS Console.

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 Configure parameters for a bucket policy.

Table 5-17 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Customized**.
Effect	Select **Allow**.

Parameter	Description
Principal	• Select **Include** > **Other account**. • **Account ID**: Enter the ID of the account which you want to grant permissions to. You can obtain it from the **My Credentials** page of the account. • **User ID**: Enter the account ID, which can be obtained from the **My Credentials** page of the account. **NOTE** In this example, permissions are granted to an account, excluding any IAM user under the account. Therefore, the user ID is the same as the account ID.
Resources	Select **Include** > **Entire bucket**.
Actions	• **Include** • **Action Name**: – PutBucketAcl – GetBucketAcl – ListBucket (required when the authorized account wants to access the OBS bucket on OBS Browser+ by mounting an external bucket) To configure other permissions, select the corresponding actions. For details about the actions supported by OBS, see **Action/NotAction**.

Step 6 Click **OK**. The bucket policy is created.

----End

5.3.3 Granting IAM Users Under an Account the Access to a Bucket and the Resources in It

Scenario

This topic describes how to grant IAM users the permissions to access OBS buckets and resources in them.

The following describes how to grant the permissions to upload and download objects in a bucket. If you need to configure other specified permissions, configure the corresponding permissions in the bucket policy and IAM permissions.

Recommended Configuration

To grant permissions to IAM users under other accounts, you need to **configure both bucket policies** and **IAM permissions**.

For example, to allow IAM user **A** of account **A** to access bucket **B** of account **B**, you need to:

1. Configure a bucket policy that allows IAM user **A** to access bucket **B**.

2. Configure IAM permissions for account **A** to allow IAM user **A** to access bucket **B**.

The permissions allowed by both bucket policies and IAM permissions take effect.

Configuration Precautions

After the configuration is complete, the authorized IAM user can upload and download objects through APIs. In addition, the user can upload and download objects by mounting external buckets on OBS Browser+. To add external buckets, the **ListBucket** permission is also required. Currently, access to buckets of other accounts is not allowed on OBS Console.

Procedure 1: Configure a Bucket Policy That Allows Specified Operations

The bucket owner or a user who has the permission to configure bucket policies needs to configure a bucket policy that allows specified operations.

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 Configure a bucket policy that allows uploads and downloads.

Table 5-18 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Customized**.
Effect	Select **Allow**.
Principal	• Select **Include** > **Other account**. • **Account ID**: Enter the ID of the account which you want to grant permissions to. You can obtain it from the **My Credentials** page of the account or the IAM user. • **User ID**: Enter the ID of the IAM user under the authorized account. You can obtain the ID on the **My Credentials** page of the IAM user. The wildcard character (*) is supported, indicating that the setting takes effect for all IAM users under the account.

Parameter	Description
Resources	• Choose **Include** > **Specific resources**. • **Resource Name**: Enter the object or the set of objects that will be accessed. – For one object, enter *object name*. – For a set of objects, enter *object name prefix + *, * + object name suffix, or **. Set this parameter to * if all objects need to be downloaded.
Actions	• **Include** • **Action Name**: – GetObject – GetObjectVersion – PutObject – (Optional) ListBucket: Select this operation if you need to use OBS Browser+ to add external buckets. To configure other specified operation permissions on objects, select the corresponding actions. For details about the actions supported by OBS, see **Action/NotAction**.

Step 6 Click **OK**. The bucket policy that allows upload and download is created.

Step 7 (Optional) Click **Create Bucket Policy** again to configure a bucket policy that allows objects in the bucket to be listed. (Perform this step when you need to use OBS Browser+ to add external buckets.)

Table 5-19 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Customized**.
Effect	Select **Allow**.
Principal	• Select **Include** > **Other account**. • **Account ID**: Enter the ID of the account which you want to grant permissions to. You can obtain it from the **My Credentials** page of the account or the IAM user. • **User ID**: Enter the ID of the IAM user under the authorized account. You can obtain the ID on the **My Credentials** page of the IAM user. The wildcard character (*) is supported, indicating that the setting takes effect for all IAM users under the account.
Resources	Select **Include** > **Entire bucket**.

Parameter	Description
Actions	• **Include** • **Action Name**: ListBucket To configure other specified permissions on buckets, select the corresponding actions. For details about the actions supported by OBS, see **Action/NotAction**.

Step 8 Click **OK**. The bucket policy for listing objects in the bucket is created.

----End

Procedure 2: Configure an IAM Permission That Allows Specified Operations

The account to which the authorized IAM user belongs needs to configure the IAM permission for the IAM user to perform specified operations on the specified bucket. The allowed operations must be the same as those specified in the bucket policy.

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 In the navigation pane, choose **Policies**.

Step 4 Click **Create Custom Policy** in the upper right corner.

Step 5 Configure parameters for a custom policy.

Table 5-20 Parameters for configuring a custom policy

Parameter	Description
Policy Name	Name of the custom policy
Policy View	Set this parameter based on your own habits. **Visual editor** is used here.

Parameter	Description
Policy Content	• Select **Allow**. • Select **Object Storage Service (OBS)**. • Select the actions to be authorized. – ReadOnly > **obs:bucket:ListBucketVersions** and **obs:object:GetObjectVersion** – ReadWrite > **obs:object:PutObject** – ListOnly > **obs:bucket:ListBucket** (Select this operation if you need to use OBS Browser+ to add external buckets.) • Choose **Specific** > **object** to specify an object resource. The specified object or object set must be consistent with the bucket policy. – Select **Any** if the resource set in the bucket policy is *****. – If the resource specified in the bucket policy is a specified object or a set of objects, you need to specify the object or the set of objects the same as that in the bucket policy through the resource path. [Format] obs:*:*:object:*bucket name/object name* Select **Any** as the bucket policy in this example is set to *****. • Choose **Specific** > **bucket** > **Specify resource path** to specify bucket resources. Click **Add Resource Path** and enter the name of the authorized bucket in the **Path** text box, for example, **example-bucket**. The complete path of the resource is as follows: **OBS:*:*:bucket:example-bucket**.
Scope	The default value is **Global services**.

Step 6 Click **OK**. The custom policy is created.

Step 7 **Create a user group and assign permissions**.

Add the created custom policy to the user group by following the instructions in the IAM document.

Step 8 Add the IAM user you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for a custom policy to take effect after the authorization.

----End

5.3.4 Granting Other Accounts the Read Permission for Certain Objects

Scenario

This case describes how to grant other accounts (excluding IAM users under the account) the read permission for an object or a type of objects in an OBS bucket. For details about how to grant permissions to an IAM user, see **Granting IAM Users Under an Account the Access to a Bucket and the Resources in It**.

Recommended Configuration

You are advised to use bucket policies to grant permissions to other accounts.

Configuration Precautions

After the configuration is complete, you can read (download) specific objects using APIs. However, if you download an object from OBS Console or OBS Browser +, an error is reported indicating that you do not have required permissions.

This is because when you log in to OBS Console or OBS Browser+, the **ListAllMyBuckets** APi is called to load the bucket list and some other APIs will also be called on other pages, but your permissions do not cover those APIs. In such case, your access is denied or your operation is not allowed.

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 Configure parameters for a bucket policy.

Table 5-21 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Read-only**.

Parameter	Description
Principal	• Select **Include** > **Other account**. • **Account ID**: Enter the ID of the account which you want to grant permissions to. You can obtain it from the **My Credentials** page of the account. • **User ID**: Enter the account ID, which can be obtained from the **My Credentials** page of the account. **NOTE** In this example, permissions are granted to an account, excluding any IAM user under the account. Therefore, the user ID is the same as the account ID.
Resources	• **Include** • **Resource Name**: Enter the object or the set of objects that will be accessed. For one object, enter *object name*. For a set of objects, enter *object name prefix + *, * + object name suffix, or **.

Step 6 Click **OK**. The bucket policy is created.

----End

5.3.5 Granting Other Accounts the Specified Permissions for Certain Objects

Scenario

This case describes how to grant other accounts the specified permissions for a specified object in an OBS bucket. The following describes how to grant the permission to download an object.

If you need to configure other permissions, select the corresponding actions from the **Action Name** drop-down list in the bucket policy. For details about the actions supported by OBS, see **Action/NotAction**.

For details about how to grant permissions to an IAM user, see **Granting IAM Users Under an Account the Access to a Bucket and the Resources in It**.

Recommended Configuration

You are advised to use bucket policies to grant permissions to other accounts.

Configuration Precautions

After the configuration is complete, you can download objects using APIs. However, if you log in to OBS Console or OBS Browser+ to download an object, an error is reported indicating that you do not have required permissions.

This is because when you log in to OBS Console or OBS Browser+, APIs (such as **ListAllMyBuckets** and **ListBucket**) are called to load the bucket list and object list

and some other APIs will also be called on other pages, but your permissions do not cover those APIs. In such case, your access is denied or your operation is not allowed.

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 Configure parameters for a bucket policy.

Table 5-22 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Customized**.
Effect	Select **Allow**.
Principal	• Select **Include** > **Other account**. • **Account ID**: Enter the ID of the account which you want to grant permissions to. You can obtain it from the **My Credentials** page of the account. • **User ID**: Enter the account ID, which can be obtained from the **My Credentials** page of the account. **NOTE** In this example, permissions are granted to an account, excluding any IAM user under the account. Therefore, the user ID is the same as the account ID.
Resources	• Choose **Include** > **Specific resources**. • **Resource Name**: Enter the object or the set of objects that will be accessed. For one object, enter *object name*. For a set of objects, enter *object name prefix + *, * + object name suffix, or **.
Actions	• **Include** • Action Name: Select **GetObject**. To configure other permissions, select the corresponding actions. For details about the actions supported by OBS, see **Action/NotAction**.

Step 6 Click **OK**. The bucket policy is created.

----End

5.4 Granting Permissions to Anonymous Users

5.4.1 Granting Anonymous Users the Public Read Permission for a Bucket

Scenario

If a bucket needs to be accessed by anonymous users, you can configure a bucket policy and bucket ACL to grant the access permission to anonymous users. The following uses a bucket policy as an example.

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 On the **Bucket Policies** tab page, select the **Public Read** policy for the bucket in the **Standard Bucket Policies** area.

----End

Verification

Step 1 After the permission is set, in the **Basic Information** area of the bucket overview page, locate **Access Domain Name**. Share the URL of the access domain name over the Internet so that all Internet users can access the bucket.

Step 2 On the **Objects** tab page of the bucket, click the target object name and find the object link. Share the object link over the Internet so that all Internet users can access the object.

----End

5.4.2 Granting Anonymous Users the Read Permission for a Directory

Scenario

If all objects in a folder need to be accessible to anonymous users, you can configure a bucket policy to grant anonymous users the permission to access the folder.

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 Configure parameters according to the following table, so that you can grant anonymous users the permission to access the folder and objects in it.

Table 5-23 Parameters for granting the permission to access a specified directory

Parameter	Value
Policy Mode	Select **Read-only**.
Principal	• **Include** • Select **Other account**, and enter an asterisk (*) as the account ID, indicating all anonymous users.
Resources	• **Include** • Select **Specific resources**. • Set this parameter to all objects in the selected folder. If the folder name is **folder-001**, enter the value **folder-001/***.

Step 6 Click **OK**.

----End

Verification

After the permission is set, click an object in the folder. Its URL is displayed under **Link**. Share the URL over the Internet, so that all users can access or download the object through the Internet.

5.4.3 Granting Anonymous Users the Read Permission for Certain Objects

Scenario

Enterprise A stores a large volume of map data in OBS, and offers the data for public query. This enterprise sets a read permission for anonymous users, and provides the data URLs on the Internet. Then all users can read or download the data through the URLs.

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket to be operated. The **Overview** page of the bucket is displayed.

Step 3 In the navigation pane, click **Objects**.

Step 4 Click the name of the object to be operated.

Step 5 On the **Object ACL** tab page, click the target object and click **Object ACL**.

Step 6 In **Public Permissions** > **Anonymous User**, click **Edit** and select the object read permission for anonymous users.

Step 7 Click **Save** to save the permission setting.

----End

Verification

After the permission is set, click the object. Its URL is displayed under **Link**. Share the URL over the Internet, so that all users can access or download the object through the Internet.

5.4.4 Temporarily Sharing Objects with Anonymous Users

Scenario

If you want to open an object to all users for a limited period of time, you can use the object sharing function.

Procedure for Sharing a File

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Objects**.

Step 4 Locate the file to be shared and click **Share** in the **Operation** column.

Once the **Share File** dialog box is opened, the URL is effective and valid for five minutes by default. If you change the validity period, the authentication information in the URL changes accordingly, and the URL's new validity period starts upon the change.

Step 5 Perform URL related operations.

- Click **Open URL** to preview the file on a new page or directly download it to your default download path.
- Click **Copy Link** to share the link to other users, so that they can enter the link to a web browser to access the file.
- Click **Copy Path** to share the file path to users who have access permissions to the bucket. Then the users can search for the file by pasting the path to the search box of the bucket.

NOTE

Within the URL validity period, anyone who has the URL can access the file.

----End

Procedure for Sharing a Folder

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Objects**.

Step 4 Locate the folder you want to share and click **Share** in the **Operation** column. The **Share Folder** dialog box is displayed.

Step 5 Configure parameters.

Table 5-24 Parameters for sharing a folder

Parameter	Description
URL Validity Period	The validity period is measured by minutes or hours, and ranges from one minute to 18 hours. The default value is five minutes. Within the URL validity period, anyone who has the URL can access the folder.
Access Code	A six-digit code. An extraction code is required to access a shared folder.

Step 6 Click **Create Share** to generate a sharing URL for the folder.

Step 7 You can send the URL and extraction code to other users. Then they can access the folder using the URL and the extraction code.

----End

Verification

Verify that other users can access the shared folder through the URL.

Step 1 Open a web browser, enter the shared URL, and open it.

Step 2 In the dialog box that is displayed, enter the access code and access objects in the shared folder.

----End

Verify that other users can access the shared folder through OBS Browser+.

Step 1 Start OBS Browser+.

Step 2 On the login page, click **Authorization Code Login**.

Step 3 Enter the authorization code and access code.

Step 4 Click **Log In** to access the shared folder.

----End

5.5 Granting Temporary Access to OBS

Scenario

This case describes how to use temporary access keys (temporary AK/SK and security token) to access OBS in temporary authorization mode.

Assume that you want to enable an IAM user (user name: APPServer) to access the APPClient folder in bucket **hi-company** and apply for two different temporary access keys to distribute to APP-1 and APP-2. APP-1 can only access files in APPClient/APP-1. APP-2 can access only the files in APPClient/APP-2.

Procedure

Step 1 Log in to the management console using a cloud service account.

Step 2 On the top menu bar, choose **Service List** > **Management & Deployment** > **Identity and Access Management**. The IAM console is displayed.

Step 3 Create an IAM user **APPServer**. For details, see **Creating an IAM User**.

Step 4 Create a user-defined policy that allows access to the AppClient folder in bucket hi-company.

1. In the navigation pane, choose **Policies**.
2. Configure parameters for a custom policy.

   NOTE

   Before configuring an IAM policy, you need to understand what permissions are required. An IAM user only has the permissions defined by the policy. In this example, user **APPServer** only has full permissions on objects in the **APPClient** folder.

Table 5-25 Parameters for configuring a custom policy

Parameter	Description
Policy Name	Name of the custom policy
Policy View	Set this parameter based on your own habits. **JSON** is used here.

Parameter	Description

Policy Content	{
	"Version": "1.1",
	"Statement": [
	{
	"Action": [
	"obs:object:*"
	],
	"Resource": [
	"obs:*:*:object:hi-company/APPClient/*"
	],
	"Effect": "Allow"
	}
	]
	}

Scope	The default value is **Global services**.

3. Click **OK**. The custom policy is created.

Step 5 **Create a user group and assign permissions**.

Add the created custom policy to the user group by following the instructions in the IAM document.

Step 6 Add the IAM user (**APPServer**) you want to authorize to the created user group by referring to **Adding Users to or Removing Users from a User Group**.

NOTE

Due to data caching, it takes about 10 to 15 minutes for a custom policy to take effect after the authorization.

Step 7 The IAM user (APPServer) obtains temporary access keys (temporary access keys and security token) for **APP-1** and **APP-2**.

To obtain temporary access keys with different permissions, you need to set a temporary policy by adding the policy parameter in the request body. For details, see **Obtaining a Temporary Access Key and Security Token Through a Token**.

The following is a sample request for obtaining a pair of temporary access keys. The temporary policy parameters are displayed in bold.

A sample request for obtaining a pair of temporary access keys for the device app APP-1:

```
{
   "auth": {
   "identity": {
      "policy": {
      "Version": "1.1",
      "Statement": [
         {
         "Action": [
            "obs:object:*"
         ],
         "Resource": [
            "obs:*:*:object:hi-company/APPClient/APP-1/*"
         ],
         "Effect": "Allow"
         }
      ]
      },
```



```
        "token": {
        "duration-seconds": 900

        },
        "methods": [
        "token"
        ]
    }
    }
}
```

A sample request for obtaining a pair of temporary access keys for the device app APP-2:

```
{
    "auth": {
    "identity": {
        "policy": {
        "Version": "1.1",
        "Statement": [
            {
            "Action": [
                "obs:object:*"
            ],
            "Resource": [
                "obs:*:*:object:hi-company/APPClient/APP-2/*"
            ],
            "Effect": "Allow"
            }
        ]
        },
        "token": {
        "duration-seconds": 900

        },
        "methods": [
        "token"
        ]
    }
    }
}
```

----End

Verification

After **APP-1** and **APP-2** have the temporary access keys, they can access OBS through OBS APIs. **APP-1** can access only files in the **APPClient/APP-1** folder, and **APP-2** can access only files in the **APPClient/APP-2** folder.

5.6 Restricting Access to a Bucket for Specific IP Addresses

Scenario

This case describes how to restrict the source IP addresses that can access an OBS bucket. The following shows how to deny a client access whose source IP address is within the range of 114.115.1.0/24.

Recommended Configuration

Bucket policy

Procedure

Step 1 In the navigation pane of OBS Console, choose **Object Storage**.

Step 2 In the bucket list, click the bucket name you want to go to the **Overview** page.

Step 3 In the navigation pane, choose **Permissions**.

Step 4 On the **Bucket Policies** page, click **Create Bucket Policy** under **Custom Bucket Policies**.

Step 5 Configure parameters for a bucket policy.

Table 5-26 Parameters for creating a bucket policy

Parameter	Description
Policy Mode	Select **Customized**.
Effect	Select **Deny**.
Principal	• Choose **Include** > **Other account**. • **Account ID**: Enter *, which indicates that the setting takes effect for all registered users and anonymous users. • **User ID**: Leave the user ID blank.
Resources	Select **Include** > **Entire bucket**.
Actions	• **Include** • Action Name: Select *, which indicates all permissions.
Conditions	• **Conditional Operator**: **IpAddress** • **Key**: Select **SourceIp**. • **Value**: Enter **114.115.1.0/24**. **NOTE** Use commas (,) to separate multiple IP addresses.

NOTE

If you want to allow clients whose IP addresses are outside the configured range to access your bucket, grant access permissions to anonymous users by referring to **Granting Permissions to Anonymous Users**.

Step 6 Click **OK**. The bucket policy is created.

----End

Verification

Initiate an access request from an IP address within the range of 114.115.1.0/24. The access is denied. Initiate an access request from an IP address outside the range of 114.115.1.0/24. The access is allowed.

Related Scenarios

To allow only a specified IP address to access the OBS bucket, set **Condition Operator** to **NotIpAddress** and specify the allowed IP address as the **Value**.

A Appendix

A.1 Bucket Policy Parameters

A policy in JSON format is described as follows:

```
{
"Statement" : [{
    statement1
 },
 {
    statement2
 },
  ......
 ]
}
```

Example:

```
{
"Statement" : [{
    "Sid": "ExampleStatementID1",
    "Principal": "*",
    "Effect": "Allow",
    "Action": "ListBucket",
    "Resource": "examplebucket",
    "Condition": "some conditions"
 },
 {
    "Sid": "ExampleStatementID2",
    "Principal": "*",
    "Effect": "Allow",
    "Action": "PutObject",
    "Resource": "examplebucket",
    "Condition": "some conditions"
 },
......
]
}
```

A policy is comprised of one or more statements. Each statement contains the following elements:

Table A-1 Statement elements

Element	Description	Mandatory/ Optional
Sid	ID of a statement. The value is a string that describes the statement.	Optional
Principal	Domains and users to which a statement applies. The wildcard (*) is supported, indicating all users. When permissions are authorized to all users under a domain, the format of **Principal** is **domain/***domainid***:user/*** . When permissions are authorized to a specific user under a domain, the format of **Principal** is **domain/***domainid***:user/***userId* or **domain/***domainid***:user/***userName*.	Optional. Select either **Principal** or **NotPrincipal**.
NotPrincipal	An exception to a list of principals in the statement. You can deny access to all principals except the ones named in the **NotPrincipal** element. This parameter has the same value format as **Principal**.	Optional. Select either **NotPrincipal** or **Principal**.
Action	Actions which a statement applies to. This parameter specifies a set of all the operations supported by OBS. Its values are case insensitive. The value supports a wildcard character (*) that indicates all actions, for example, **"Action": ["List*", "Get*"]**.	Optional. Select either **Action** or **NotAction**.
NotAction	An exception to a list of actions in the statement. All actions are performed except the ones specified in **NotAction**. This parameter has the same value format as **Action**.	Optional. Select either **Action** or **NotAction**.
Effect	Whether the permission in a statement is allowed or denied. The value is **Allow** or **Deny**.	Mandatory
Resource	Resources on which the statement takes effect. The wildcard (*) is supported, indicating all resources.	Optional. Select either **Resource** or **NotResource**.
NotResource	An exception to a list of resources in a statement. A policy is not applied to the resources specified in **NotResource**. This parameter has the same value format as **Resource**.	Optional. Select either **Resource** or **NotResource**.
Condition	Conditions for a statement to take effect.	Optional

NOTE

A statement must contain either **Action** or **NotAction**, either **Resource** or **NotResource**, and either **Principal** or **NotPrincipal**.

Principal/NotPrincipal

Principal or **NotPrincipal** supported by OBS includes anonymous users, specific tenants, specific users, federated users, and agencies.

- All (anonymous users)

  ```
  "Principal": {"ID": "*"}
  ```

  In the example, the wildcard (*) is used as a placeholder for Everyone/Anonymous. We strongly recommend that you do not use wildcards in the **Principal** element of the role's trust policy unless you have restricted access by using the **Condition** element in the policy.

- Specific tenants

  If the tenant identifier is used as the authorizer in the policy, permissions in the policy statement can be granted to all roles, including all the users, contained in this tenant. The following example demonstrates how to specify a tenant as an authorizer.

  ```
  "Principal": { "ID": " domain/domainIdxxxx:user/*" }
  ```

  You can grant permissions to multiple tenants, as described in the following example:

  ```
  "Principal": {
    "ID": [
      "domain/domainIDxx1:user/useridxxxx",
      "domain/domainIDxx2:user/*"
    ]
  }
  ```

- Specific users

  In the **Principal** element, user names are case sensitive.

  ```
  "Principal": {"ID": "domain/domainIDxxx:user/user-name" }
  "Principal": {
    "ID": [
      "domain/domainIDxxx:user/UserID1",
      "domain/domainIDxxx:user/UserID2"
    ]
  }
  ```

- Federated users (using SAML identity provider)

  ```
  "Principal": { "Federated": "domain/domainIDxxx:identity-provider/provider-name" }
  "Principal": { "Federated": "domain/domainIDxxx:group/groupname" }
  ```

- Agencies

  ***** indicates all agencies of a tenant.

  ```
  "Principal": { "ID": "domain/domainIDxxx:agency/agencyname" }
  "Principal": { "ID": "domain/domainIDxxx:agency/*" }
  ```

The principals on OBS Console refer to the users which the bucket policies apply to. These users can be accounts and IAM users. Principals can be specified in either of the following ways:

- **Include**: The policy applies to specified users.
- **Exclude**: The policy applies to users except the specified ones.

Specifying IAM users under the current account

With **Principal** set to **Current account**, you can select one or more IAM users under this account, so the bucket policy applies to the selected IAM users.

Specifying another account

With **Principal** set to **Other account**, you can enter an account ID. If you want to grant access only to IAM users under the account, you need to enter user IDs, and use commas (,) to separate one user ID from another.

NOTE

To obtain the account ID and user ID, log in to the console as an IAM user and go to the **My Credentials** page.

Specifying anonymous users

To grant the bucket access to anyone, set **Principal** to **Other account** and enter a wildcard (*) as the account ID.

NOTICE

Exercise caution when granting bucket access permissions to anonymous users. If you grant the access permissions to anonymous users, anyone can access your bucket, and all the traffic and storage fees incurred will be borne by the bucket owner. You are advised to set restrictions on access requests. For example, you can allow the access requests from only one IP address.

Action/NotAction

If a policy applies to a bucket, configure bucket-related actions; if the policy applies to the objects in a bucket, configure object-related actions.

Actions can be specified in either of the following ways:

- **Include**: The bucket policy applies to specified actions.
- **Exclude**: The bucket policy applies to actions except the specified ones.

Bucket Actions

Table A-2 Action description

Type	Value	Description
General	*	Indicates that all operations can be performed on a resource.
	Get*	Indicates that all GET operations can be performed on a resource.
	Put*	Indicates that all PUT operations can be performed on a resource.
	List*	Indicates that all LIST operations can be performed on a resource.
Bucket	CreateBucket	Creates a bucket.
	DeleteBucket	Deletes a bucket.

Type	Value	Description
	ListBucket	Lists objects in a bucket, and obtains the bucket metadata.
	ListBucketVersions	Lists versioned objects in a bucket.
	ListBucketMultipartUploads	Lists multipart upload tasks.
	GetBucketAcl	Gets the bucket ACL information.
	PutBucketAcl	Configures a bucket ACL.
	GetBucketCORS	Gets the CORS configuration of a bucket.
	PutBucketCORS	Configures CORS for a bucket.
	GetBucketVersioning	Gets the bucket versioning information.
	PutBucketVersioning	Configures versioning for a bucket.
	GetBucketLocation	Gets the bucket location.
	GetBucketLogging	Gets the bucket logging information.
	PutBucketLogging	Configures logging for a bucket.
	GetBucketWebsite	Obtains the static website configuration information of a bucket.
	PutBucketWebsite	Configures static website hosting for a bucket.
	DeleteBucketWebsite	Cancels the static website hosting of a bucket.
	GetLifecycleConfiguration	Obtains the lifecycle rules of a bucket.
	PutLifecycleConfiguration	Configures a lifecycle rule for a bucket.

Object Actions

Table A-3 Action description

Type	Value	Description
General	*	Indicates that all operations can be performed on a resource.
	Get*	Indicates that all GET operations can be performed on a resource.

Type	Value	Description
	Put*	Indicates that all PUT operations can be performed on a resource.
	List*	Indicates that all LIST operations can be performed on a resource.
Object	GetObject	Gets the content and metadata of an object.
	GetObjectVersion	Gets the content and metadata of a specified object version.
	PutObject	Performs PUT upload, POST upload, multipart upload, initialization of uploaded parts, and merging of parts.
	GetObjectAcl	Gets the object ACL information.
	GetObjectVersionAcl	Gets the ACL information of a specified object version.
	PutObjectAcl	Configures the ACL for an object.
	PutObjectVersionAcl	Configures the ACL for a specified object version.
	DeleteObject	Deletes an object.
	DeleteObjectVersion	Deletes a specified object version.
	ListMultipartUpload-Parts	Lists uploaded parts.
	AbortMultipartUpload	Cancels a multipart upload.

Resource/NotResource

The resources supported by OBS are as follows:

- *bucketname* (bucket operation): The **Action** drop-down list box contains the list of supported bucket actions. If you want to perform the listed operations on the bucket, set **Resource** to the bucket name.
- *bucketname/objectname* (object operation): The **Action** drop-down list box contains the list of supported object actions. If you want to respond to an object in a bucket, set **Resource** to *bucketname/objectname*. **objectname** supports wildcards. For example, if you have permissions on the directory object in a bucket, set **Resource** to *"bucketname/directory/*"*. If you have permissions on all the objects in a bucket, set **Resource** to *"bucketname/*"*. If permissions for both a bucket and its objects need to be granted, set **Resource** to **["examplebucket/*","examplebucket"]**.

The following example policy grants all operation permissions on **examplebucket** (including the bucket and its objects) to user1 whose user ID is

71f3901173514e6988115ea2c26d1999 under account **b4bf1b36d9ca43d984fbcb9491b6fce9** (account ID).

```
{
   "Statement":[
   {
     "Sid":"test",
     "Effect":"Allow",
     "Principal": {"ID": ["domain/b4bf1b36d9ca43d984fbcb9491b6fce9:user/
71f3901173514e6988115ea2c26d1999"]},
     "Action":["*"],
     "Resource":["examplebucket/*","examplebucket"]
   }
 ]
}
```

On OBS Console, you can apply a bucket policy to the following resources: the current bucket, and all objects in a bucket.

Resources can be specified in either of the following ways:

- **Include**: The bucket policy applies to specified OBS resources.
- **Exclude**: The bucket policy applies to OBS resources except the specified ones.

Applying a bucket policy to a bucket

To apply a bucket policy to the current bucket, keep the resource text box empty. When configuring actions for the policy, select bucket related actions.

Applying a bucket policy to specified objects

To apply the bucket policy to specified objects in a bucket, object-related actions must be configured in the policy. The configuration format is as follows:

- For an object, enter the object name (including its folder name if any). For example, if the specified resource is the **example.jpg** file in the **imgs-folder** folder in the bucket, enter the following content in the resource text box:

  imgs-folder/example.jpg
- For an object set, the wildcard asterisk (*) should be used. The asterisk (*) indicates an empty string or any combination of multiple characters. The format rules are as follows:
  - Use only one asterisk (*) to indicate all objects in a bucket.
  - Use *Object name prefix** to indicate objects starting with this prefix in a bucket. Example:

    imgs*
  - Use **Object name suffix* to indicate objects ending with this suffix in a bucket. Example:

    *.jpg

NOTE

Use commas (,) to separate one object (or object set) from another.

Condition

In addition to the effect, principals, resources, and actions, you can also specify the conditions under which a bucket policy takes effect. The bucket policy takes effect

only when its condition expressions match values contained in the request. Conditions are optional. You can choose whether to configure them.

For example, if account A needs to have full control over an object uploaded by account B to bucket **example** of account A, the **x-obs-acl** key must be specified in the upload request and the policy effect must be set to **Allow** for account A. The complete condition expression is as follows:

Condition Operator	Key	Value
StringEquals	x-obs-acl	bucket-owner-full-control

A condition consists of three parts: condition operator, key, and value. If there are multiple identical keys in the same condition operator, only the last key is retained. Condition operators and keys are mutually restricted. If you select a condition operator of the string type, for example, **StringEquals**, the key can only be of the string type, for example, **UserAgent**. Likewise, if a key of the date type is selected, for example, **CurrentTime**, the condition operator can only be of the date type, for example, **DateEquals**.

- **Condition operators**

  A condition operator, a condition key, and a condition value together constitute a complete condition statement. A policy can be applied only when its request conditions are met. **Table A-4** lists the condition operators available for statements. String condition operators are not case-sensitive unless otherwise specified.

Table A-4 Condition operators

Type	Element	Description
String	StringEquals	Strict matching. Short version: streq
	StringNotEquals	Strict negated matching. Short version: strneq
	StringEqualsIgnoreCase	Strict matching, ignoring case. Short version: streqi
	StringNotEqualsIgnoreCase	Strict negated matching, ignoring case. Short version: strneqi
	StringLike	Loose case-sensitive matching. The values can include a multi-character match wildcard (*) or a single-character match wildcard (?) anywhere in the string. Short version: strl

Type	Element	Description
	StringNotLike	Negated loose case-sensitive matching. The values can include a multi-character match wildcard (*) or a single-character match wildcard (?) anywhere in the string. Short version: strnl
Numeric	NumericEquals	Strict matching. Short version: numeq **Numeric** indicates a data type expressed in numbers.
	NumericNotEquals	Strict negated matching. Short version: numneq
	NumericLessThan	"Less than" matching. Short version: numlt
	NumericLessThanEquals	"Less than or equals" matching. Short version: numlteq
	NumericGreaterThan	"Greater than" matching. Short version: numgt
	NumericGreaterThanEquals	"Greater than or equals" matching. Short version: numgteq
Date	DateEquals	Strict matching. Short version: dateeq
	DateNotEquals	Strict negated matching. Short version: dateneq
	DateLessThan	The date is earlier than a specific date. Short version: datelt
	DateLessThanEquals	The date is earlier than or equal to a specific date. Short version: datelteq
	DateGreaterThan	The date is later than a specific date. Short version: dategt
	DateGreaterThanEquals	The date is later than or equal to a specific date. Short version: dategteq
Boolean	Bool	Strict Boolean matching
IP address	IpAddress	Specified IP address or IP address range
	NotIpAddress	All IP addresses excluding the specified IP address or IP address range

NOTE

Elements in a condition are case sensitive. The date format complies with the ISO 8601 standard, for example, **2015-07-01T12:00:00Z**.

Each condition can contain multiple key-value pairs. The **Condition** combination in the following figure indicates that the request time ranges from **2015-07-01T12:00:00Z** to **2018-04-16T15:00:00Z** and the request IP address range is **192.168.176.0/24** or **192.168.143.0/24**.

```
"Condition" : {
  "DateGreaterThan" : {
  "CurrentTime" : "2015-07-01T12:00:00Z"
  },
  "DateLessThan": {
  "CurrentTime" : "2018-04-16T15:00:00Z"
  },
  "IpAddress" : {
  "SourceIp" : ["192.168.176.0/24","192.168.143.0/24"]
  }
}
```

- **Condition keys**

Keys in a condition can be classified into three types: general keys, keys related to bucket actions, and keys related to object actions.

The following table lists the keys that are not related to actions.

Table A-5 General keys

Key	Type	Description
CurrentTime	Date	Date when the request is received by the server. The date format must comply with ISO 8601.
EpochTime	Numeric	Time when the request is received by the server, which is expressed as seconds since 1970.01.01 00:00:00 UTC, regardless of the leap seconds
SecureTransport	Bool	Whether requests are encrypted using SSL **NOTE** The value can be either **true** or **false**. Any other values you enter will become **false** by default.
SourceIp	IP address	Source (client) IP address of the request
UserAgent	String	Requested client software agent
Referer	String	Link from which the request is sent

Keys in a condition must be used in certain actions. The following table lists the mapping between actions and the keys in a condition.

Table A-6 Keys related to bucket actions

Action	Optional Key	Description	Remarks
ListBucket	prefix	Type: String. Lists objects that begin with the specified prefix.	If **prefix**, **delimiter**, and **max-keys** are configured, the key-value pair meeting the conditions must be specified in the List operation for the bucket policy to take effect. For example, if a bucket policy (with the condition operator set to **NumericEquals**, the key to **max-keys**, and the value to **100**) that allows anonymous users to read data is configured for a bucket, the anonymous users must add **?max-keys=100** to the end of the bucket domain name for listing objects. The listed objects are the first 100 objects in alphabetic order.
	delimiter	Type: String. Groups objects in a bucket.	
	max-keys	Type: Numeric. Sets the maximum number of objects. Returned objects are listed in alphabetic order.	
ListBucketVersions	prefix	Type: String. Lists multi-version objects whose name starts with the specified prefix.	
	delimiter	Type: String. Groups objects of different versions in a bucket.	
	max-keys	Type: Numeric. Sets the maximum number of objects. Returned objects are listed in alphabetic order.	

Action	Optional Key	Description	Remarks
PutBucketAcl	x-obs-acl	Type: String. Configures the bucket ACL. When modifying a bucket ACL, you can use the request that contains a canned ACL setting in its header. Value options of a canned ACL setting: **private|public-read|public-read-write|bucketowner-read|log-delivery-write**.	None

Table A-7 Keys related to object actions

Action	Optional Key	Description
PutObject	x-obs-acl	Type: String. Configures the object ACL. When uploading an object, you can use the request that contains a canned ACL setting in its header. Value options of a canned ACL setting: **private|public-read|public-read-write|bucketowner-read|bucket-owner-full-control|log-delivery-write**.
	x-obs-copy-source	Type: String. Specifies names of the source bucket and the source object. Format: **/***bucketname***/***keyname*
	x-obs-metadata-directive	Type: String. Specifies whether to copy the metadata from the source object or replace with the metadata in the request. The value can be **COPY** or **REPLACE**.
	x-obs-server-side-encryption	Type: String. Specifies that objects in a bucket are encrypted using SSE-KMS before they are stored. The value is **kms**.
PutObjectAcl	x-obs-acl	Type: String. Configures the object ACL. When uploading an object, you can use the request that contains a canned ACL setting in its header. Value options of a canned ACL setting: **private|public-read|public-read-write|bucketowner-read|bucket-owner-full-control|log-delivery-write**.

Action	Optional Key	Description
GetObjectVersion	versionId	Type: String. Obtains the object with the specified version ID.
GetObjectVersionAcl	versionId	Type: String. Obtains the ACL of the object with the specified version ID.
PutObjectVersionAcl	versionId	Type: String. Specifies a version ID.
	x-obs-acl	Type: String. Configures the ACL of the object with the specified version ID. When uploading an object, you can use the request that contains a canned ACL setting in its header. Value options of a canned ACL setting: **private|public-read|public-read-write|bucketowner-read|bucket-owner-full-control|log-delivery-write**.
DeleteObjectVersion	versionId	Type: String. Deletes the object with the specified version ID.

Policy Permission Judgment Logic

A policy may pose any of the three results for each statement: **Explicit Deny**, **Allow**, and **Default Deny**. If a bucket policy contains multiple statements, the policy determines which statement prevails according to the following rules:

1. If conditions in any statement of a policy are not met, the policy poses a default deny result.

2. An explicit deny overrides an allow.

3. An allow overrides a default deny.

4. Statements can be in any order in a policy.

Table A-8 Statement results

Result	Description
explicit deny	A statement defines effect="deny". All requests for resources to which the statement applies are denied. No permission is returned.
allow	A statement defines effect="allow". All requests for resources to which the statement applies are allowed.
default deny	Conditions defined in a statement are not met. Requests are denied.

If an ACL and a bucket policy are applied together to an account, an explicit deny in the bucket policy overrides the allow in the ACL.

If a bucket policy and an IAM policy are applied together to an account, an explicit deny overrides the allow, and an allow overrides the default deny.

SSE-KMS server-side encrypted object does not support Bucket ACL/Policy for cross-tenant authorization.

A.2 Relationship Between Bucket Policies and Bucket ACLs

Mapping Between Bucket ACLs and Bucket Policies

Bucket ACLs are used to control basic read and write access to buckets. Custom settings of bucket policies support more actions that can be performed on buckets. Bucket ACLs supplement bucket policies, and in many cases, can be replaced by bucket policies to manage access to buckets, except when permissions are granted to a log delivery user group. **Table A-9** shows the mapping between bucket ACL access permissions and bucket policy actions.

Table A-9 Mapping between bucket ACLs and bucket policies

ACL Permission	Option	Mapped Action in a Custom Bucket Policy
Access to bucket	Read	• HeadBucket • ListBucket • ListBucketVersions • ListBucketMultipartUploads
	Write	• PutObject • DeleteObject • DeleteObjectVersion
Access to ACL	Read	• GetBucketAcl
	Write	• PutBucketAcl

Mapping Between Object ACLs and Bucket Policies

Object ACLs are used to control basic read and write access to objects. The custom settings of bucket policies allow you to specify more actions that can be performed on objects. **Table A-10** describes the mapping between object ACL access permissions and bucket policy actions.

Table A-10 Mapping between object ACLs and bucket policies

Object ACL Permission	Option	Mapped Action in a Custom Bucket Policy
Access to object	Read	• GetObject • GetObjectVersion
Access to ACL	Read	• GetObjectAcl • GetObjectVersionAcl
	Write	• PutObjectAcl • PutObjectVersionAcl

B Change History

Date	What's New
2022-05-20	This is the first official release.

