



# **Manage access policies**

StorageGRID software

NetApp

February 12, 2026

# Table of Contents

Manage access policies .....	1
Use access policies .....	1
Access policy overview .....	1
Consistency for policies .....	3
What is session policy ? .....	3
Use ARN in policy statements .....	4
Specify resources in a policy .....	4
Specify principals in a policy .....	5
Specify permissions in a policy .....	6
Use PutOverwriteObject permission .....	10
Specify conditions in a policy .....	11
Specify variables in a policy .....	15
Create policies requiring special handling .....	15
Write-once-read-many (WORM) protection .....	17
Session policy example .....	17
Example: Set up a session policy that allows object retrieval .....	18
Example bucket policies .....	18
Example: Allow everyone read-only access to a bucket .....	18
Example: Allow everyone in one account full access, and everyone in another account read-only access to a bucket .....	19
Example: Allow everyone read-only access to a bucket and full access by specified group .....	20
Example: Allow everyone read and write access to a bucket if client in IP range .....	21
Example: Allow full access to a bucket exclusively by a specified federated user .....	22
Example: PutOverwriteObject permission .....	23
Example group policies .....	24
Example: Set group policy using Tenant Manager .....	25
Example: Allow group full access to all buckets .....	25
Example: Allow group read-only access to all buckets .....	25
Example: Allow group members full access to only their "folder" in a bucket .....	26

# Manage access policies

## Use access policies

StorageGRID uses the Amazon Web Services (AWS) policy language to allow S3 tenants to control access to buckets and objects within those buckets. The StorageGRID system implements a subset of the S3 REST API policy language. Access policies for the S3 API are written in JSON.

### Access policy overview

StorageGRID supports three kinds of access policies:

- **Bucket policies**, which are managed using the GetBucketPolicy, PutBucketPolicy, and DeleteBucketPolicy S3 API operations or the Tenant Manager or Tenant Management API. Bucket policies are attached to buckets, so they are configured to control access by users in the bucket owner account or other accounts to the bucket and the objects in it. A bucket policy applies to only one bucket and possibly multiple groups.
- **Group policies**, which are configured using the Tenant Manager or Tenant Management API. Group policies are attached to a group in the account, so they are configured to allow that group to access specific resources owned by that account. A group policy applies to only one group and possibly multiple buckets.
- **Session policies**, which are included as part of making an AssumeRole request. Session policies only apply to the given session, further defining the permissions which the user has, in addition to those granted by group and bucket policy.



There is no difference in priority among group, bucket, and session policies.

StorageGRID bucket and group policies follow a specific grammar defined by Amazon. Inside each policy is an array of policy statements, and each statement contains the following elements:

- Statement ID (Sid) (optional)
- Effect
- Principal/NotPrincipal
- Resource/NotResource
- Action/NotAction
- Condition (optional)

Policy statements are built using this structure to specify permissions: Grant <Effect> to allow/deny <Principal> to perform <Action> on <Resource> when <Condition> applies.

Each policy element is used for a specific function:

Element	Description
Sid	The Sid element is optional. The Sid is only intended as a description for the user. It is stored but not interpreted by the StorageGRID system.

Element	Description
Effect	Use the Effect element to establish whether the specified operations are allowed or denied. You must identify operations you allow (or deny) on buckets or objects using the supported Action element keywords.
Principal/NotPrincipal	<p>You can allow users, groups, and accounts to access specific resources and perform specific actions. If no S3 signature is included in the request, anonymous access is allowed by specifying the wildcard character (*) as the principal. By default, only the account root has access to resources owned by the account.</p> <p>You only need to specify the Principal element in a bucket policy. For group policies, the group to which the policy is attached is the implicit Principal element.</p>
Resource/NotResource	The Resource element identifies buckets and objects. You can allow or deny permissions to buckets and objects using the Amazon Resource Name (ARN) to identify the resource.
Action/NotAction	The Action and Effect elements are the two components of permissions. When a group requests a resource, they are either granted or denied access to the resource. Access is denied unless you specifically assign permissions, but you can use explicit deny to override a permission granted by another policy.
Condition	The Condition element is optional. Conditions allow you to build expressions to determine when a policy should be applied.

In the Action element, you can use the wildcard character (\*) to specify all operations, or a subset of operations. For example, this Action matches permissions such as s3:GetObject, s3:PutObject, and s3:DeleteObject.

```
s3:*Object
```

In the Resource element, you can use the wildcard characters (\*) and (?). While the asterisk (\*) matches 0 or more characters, the question mark (?) matches any single character.

In the Principal element, wildcard characters aren't supported except to set anonymous access, which grants permission to everyone. For example, you set the wildcard (\*) as the Principal value.

```
"Principal": "*"

```

```
"Principal": {"AWS": "*"

```

In the following example, the statement is using the Effect, Principal, Action, and Resource elements. This example shows a complete bucket policy statement that uses the Effect "Allow" to give the Principals, the

admin group federated-group/admin and the finance group federated-group/finance, permissions to perform the Action `s3:ListBucket` on the bucket named `mybucket` and the Action `s3:GetObject` on all objects inside that bucket.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::27233906934684427525:federated-group/admin",
          "arn:aws:iam::27233906934684427525:federated-group/finance"
        ]
      },
      "Action": [
        "s3:ListBucket",
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::mybucket",
        "arn:aws:s3:::mybucket/*"
      ]
    }
  ]
}
```

The bucket policy has a size limit of 20,480 bytes, and the group policy has a size limit of 5,120 bytes.

## Consistency for policies

By default, any updates you make to group policies are eventually consistent. When a group policy becomes consistent, the changes can take an additional 15 minutes to take effect, because of policy caching. By default, any updates you make to bucket policies are strongly consistent.

As required, you can change the consistency guarantees for bucket policy updates. For example, you might want a change to a bucket policy to be available during a site outage.

In this case, you can either set the `Consistency-Control` header in the `PutBucketPolicy` request, or you can use the `PUT Bucket` consistency request. When a bucket policy becomes consistent, the changes can take an additional 8 seconds to take effect, because of policy caching.



If you set the consistency to a different value to work around a temporary situation, be sure to set the bucket-level setting back to its original value when you are done. Otherwise, all future bucket requests will use the modified setting.

## What is session policy ?

A session policy is an access policy that temporarily restricts the permissions available during a specific

session, such as when a user assumes a group. A session policy can only allow a subset of permissions, and it can't grant additional permissions. The group itself might have broader permissions.

## Use ARN in policy statements

In policy statements, the ARN is used in Principal and Resource elements.

- Use this syntax to specify the S3 resource ARN:

```
arn:aws:s3:::bucket-name
arn:aws:s3:::bucket-name/object_key
```

- Use this syntax to specify the identity resource ARN (users and groups):

```
arn:aws:iam::account_id:root
arn:aws:iam::account_id:user/user_name
arn:aws:iam::account_id:group/group_name
arn:aws:iam::account_id:federated-user/user_name
arn:aws:iam::account_id:federated-group/group_name
```

Other considerations:

- You can use the asterisk (\*) as a wildcard to match zero or more characters inside the object key.
- International characters, which can be specified in the object key, should be encoded using JSON UTF-8 or using JSON \u escape sequences. Percent-encoding is not supported.

### [RFC 2141 URN Syntax](#)

The HTTP request body for the PutBucketPolicy operation must be encoded with charset=UTF-8.

## Specify resources in a policy

In policy statements, you can use the Resource element to specify the bucket or object for which permissions are allowed or denied.

- Each policy statement requires a Resource element. In a policy, resources are denoted by the element Resource, or alternatively, NotResource for exclusion.
- You specify resources with an S3 resource ARN. For example:

```
"Resource": "arn:aws:s3:::mybucket/*"
```

- You can also use policy variables inside the object key. For example:

```
"Resource": "arn:aws:s3:::mybucket/home/${aws:username}/*"
```

- The resource value can specify a bucket that does not yet exist when a group policy is created.

## Specify principals in a policy

Use the Principal element to identify the user, group, or tenant account that is allowed/denied access to the resource by the policy statement.

- Each policy statement in a bucket policy must include a Principal element. Policy statements in a group policy don't need the Principal element because the group is understood to be the principal.
- In a policy, principals are denoted by the element "Principal," or alternatively "NotPrincipal" for exclusion.
- Account-based identities must be specified using an ID or an ARN:

```
"Principal": { "AWS": "account_id"}  
"Principal": { "AWS": "identity_arn" }
```

- This example uses the tenant account ID 27233906934684427525, which includes the account root and all users in the account:

```
"Principal": { "AWS": "27233906934684427525" }
```

- You can specify just the account root:

```
"Principal": { "AWS": "arn:aws:iam::27233906934684427525:root" }
```

- You can specify a specific federated user ("Alex"):

```
"Principal": { "AWS": "arn:aws:iam::27233906934684427525:federated-  
user/Alex" }
```

- You can specify a specific federated group ("Managers"):

```
"Principal": { "AWS": "arn:aws:iam::27233906934684427525:federated-  
group/Managers" }
```

- You can specify an anonymous principal:

```
"Principal": "*" 
```

- To avoid ambiguity, you can use the user UUID instead of the username:

```
arn:aws:iam::27233906934684427525:user-uuid/de305d54-75b4-431b-adb2-eb6b9e546013
```

For example, suppose Alex leaves the organization and the username `Alex` is deleted. If a new Alex joins the organization and is assigned the same `Alex` username, the new user might unintentionally inherit the permissions granted to the original user.

- The principal value can specify a group/user name that does not yet exist when a bucket policy is created.

## Specify permissions in a policy

In a policy, the Action element is used to allow/deny permissions to a resource. There are a set of permissions that you can specify in a policy, which are denoted by the element "Action," or alternatively, "NotAction" for exclusion. Each of these elements maps to specific S3 REST API operations.

The tables lists the permissions that apply to buckets and the permissions that apply to objects.



Amazon S3 now uses the `s3:PutReplicationConfiguration` permission for both the `PutBucketReplication` and `DeleteBucketReplication` actions. StorageGRID uses separate permissions for each action, which matches the original Amazon S3 specification.



A delete is performed when a put is used to overwrite an existing value.

### Permissions that apply to buckets

Permissions	S3 REST API operations	Custom for StorageGRID
<code>s3:CreateBucket</code>	<code>CreateBucket</code>	Yes.  <b>Note:</b> Use in group policy only.
<code>s3&gt;DeleteBucket</code>	<code>DeleteBucket</code>	
<code>s3&gt;DeleteBucketMetadataNotification</code>	DELETE Bucket metadata notification configuration	Yes
<code>s3&gt;DeleteBucketPolicy</code>	<code>DeleteBucketPolicy</code>	
<code>s3&gt;DeleteReplicationConfiguration</code>	<code>DeleteBucketReplication</code>	Yes, separate permissions for PUT and DELETE
<code>s3:GetBucketAcl</code>	<code>GetBucketAcl</code>	
<code>s3:GetBucketCompliance</code>	GET Bucket compliance (deprecated)	Yes



Permissions	S3 REST API operations	Custom for StorageGRID
s3:GetBucketConsistency	GET Bucket consistency	Yes
s3:GetBucketCORS	GetBucketCors	
s3:GetEncryptionConfiguration	GetBucketEncryption	
s3:GetBucketLastAccessTime	GET Bucket last access time	Yes
s3:GetBucketLocation	GetBucketLocation	
s3:GetBucketMetadataNotification	GET Bucket metadata notification configuration	Yes
s3:GetBucketNotification	GetBucketNotificationConfiguration	
s3:GetBucketObjectLockConfiguration	GetObjectLockConfiguration	
s3:GetBucketPolicy	GetBucketPolicy	
s3:GetBucketTagging	GetBucketTagging	
s3:GetBucketVersioning	GetBucketVersioning	
s3:GetLifecycleConfiguration	GetBucketLifecycleConfiguration	
s3:GetReplicationConfiguration	GetBucketReplication	
s3:ListAllMyBuckets	<ul style="list-style-type: none"> <li>ListBuckets</li> <li>GET Storage Usage</li> </ul>	Yes, for GET Storage Usage.  <b>Note:</b> Use in group policy only.
s3:ListBucket	<ul style="list-style-type: none"> <li>ListObjects</li> <li>HeadBucket</li> <li>RestoreObject</li> </ul>	
s3:ListBucketMultipartUploads	<ul style="list-style-type: none"> <li>ListMultipartUploads</li> <li>RestoreObject</li> </ul>	
s3:ListBucketVersions	GET Bucket versions	

Permissions	S3 REST API operations	Custom for StorageGRID
s3:PutBucketCompliance	PUT Bucket compliance (deprecated)	Yes
s3:PutBucketConsistency	PUT Bucket consistency	Yes
s3:PutBucketCORS	<ul style="list-style-type: none"> <li>DeleteBucketCors†</li> <li>PutBucketCors</li> </ul>	
s3:PutEncryptionConfiguration	<ul style="list-style-type: none"> <li>DeleteBucketEncryption</li> <li>PutBucketEncryption</li> </ul>	
s3:PutBucketLastAccessTime	PUT Bucket last access time	Yes
s3:PutBucketMetadataNotification	PUT Bucket metadata notification configuration	Yes
s3:PutBucketNotification	PutBucketNotificationConfiguration	
s3:PutBucketObjectLockConfiguration	<ul style="list-style-type: none"> <li>CreateBucket with the <code>x-amz-bucket-object-lock-enabled: true</code> request header (also requires the <code>s3:CreateBucket</code> permission)</li> <li>PutObjectLockConfiguration</li> </ul>	
s3:PutBucketPolicy	PutBucketPolicy	
s3:PutBucketTagging	<ul style="list-style-type: none"> <li>DeleteBucketTagging†</li> <li>PutBucketTagging</li> </ul>	
s3:PutBucketVersioning	PutBucketVersioning	
s3:PutLifecycleConfiguration	<ul style="list-style-type: none"> <li>DeleteBucketLifecycle†</li> <li>PutBucketLifecycleConfiguration</li> </ul>	
s3:PutReplicationConfiguration	PutBucketReplication	Yes, separate permissions for PUT and DELETE

### Permissions that apply to objects

Permissions	S3 REST API operations	Custom for StorageGRID
s3:AbortMultipartUpload	<ul style="list-style-type: none"> <li>AbortMultipartUpload</li> <li>RestoreObject</li> </ul>	
s3:BypassGovernanceRetention	<ul style="list-style-type: none"> <li>DeleteObject</li> <li>DeleteObjects</li> <li>PutObjectRetention</li> </ul>	
s3>DeleteObject	<ul style="list-style-type: none"> <li>DeleteObject</li> <li>DeleteObjects</li> <li>RestoreObject</li> </ul>	
s3>DeleteObjectTagging	DeleteObjectTagging	
s3>DeleteObjectVersionTagging	DeleteObjectTagging (a specific version of the object)	
s3>DeleteObjectVersion	DeleteObject (a specific version of the object)	
s3:GetObject	<ul style="list-style-type: none"> <li>GetObject</li> <li>HeadObject</li> <li>RestoreObject</li> <li>SelectObjectContent</li> </ul>	
s3:GetObjectAcl	GetObjectAcl	
s3:GetObjectLegalHold	GetObjectLegalHold	
s3:GetObjectRetention	GetObjectRetention	
s3:GetObjectTagging	GetObjectTagging	
s3:GetObjectVersionTagging	GetObjectTagging (a specific version of the object)	
s3:GetObjectVersion	GetObject (a specific version of the object)	
s3:ListMultipartUploadParts	ListParts, RestoreObject	

Permissions	S3 REST API operations	Custom for StorageGRID
s3:PutObject	<ul style="list-style-type: none"> <li>• PutObject</li> <li>• CopyObject</li> <li>• RestoreObject</li> <li>• CreateMultipartUpload</li> <li>• CompleteMultipartUpload</li> <li>• UploadPart</li> <li>• UploadPartCopy</li> </ul>	
s3:PutObjectLegalHold	PutObjectLegalHold	
s3:PutObjectRetention	PutObjectRetention	
s3:PutObjectTagging	PutObjectTagging	
s3:PutObjectVersionTagging	PutObjectTagging (a specific version of the object)	
s3:PutOverwriteObject	<ul style="list-style-type: none"> <li>• PutObject</li> <li>• CopyObject</li> <li>• PutObjectTagging</li> <li>• DeleteObjectTagging</li> <li>• CompleteMultipartUpload</li> </ul>	Yes
s3:RestoreObject	RestoreObject	

## Use PutOverwriteObject permission

The s3:PutOverwriteObject permission is a custom StorageGRID permission that applies to operations that create or update objects. The setting of this permission determines whether the client can overwrite an object's data, user-defined metadata, or S3 object tagging.

Possible settings for this permission include:

- **Allow:** The client can overwrite an object. This is the default setting.
- **Deny:** The client can't overwrite an object. When set to Deny, the PutOverwriteObject permission works as follows:
  - If an existing object is found at the same path:
    - The object's data, user-defined metadata, or S3 object tagging can't be overwritten.
    - Any ingest operations in progress are cancelled, and an error is returned.
    - If S3 versioning is enabled, the Deny setting prevents PutObjectTagging or DeleteObjectTagging operations from modifying the TagSet for an object and its noncurrent versions.

- If an existing object is not found, this permission has no effect.
- When this permission is not present, the effect is the same as if Allow were set.



If the current S3 policy allows overwrite, and the PutOverwriteObject permission is set to Deny, the client can't overwrite an object's data, user-defined metadata, or object tagging. In addition, if the **Prevent client modification** checkbox is selected (**Configuration > Security settings > Network and objects**), that setting overrides the setting of the PutOverwriteObject permission.

## Specify conditions in a policy

Conditions define when a policy will be in effect. Conditions consist of operators and key-value pairs.

Conditions use key-value pairs for evaluation. A Condition element can contain multiple conditions, and each condition can contain multiple key-value pairs. The condition block uses the following format:

```
Condition: {
  condition_type: {
    condition_key: condition_values
```

In the following example, the IpAddress condition uses the SourceIp condition key.

```
"Condition": {
  "IpAddress": {
    "aws:SourceIp": "54.240.143.0/24"
    ...
  },
  ...
```

## Supported condition operators

Condition operators are categorized as follows:

- String
- Numeric
- Boolean
- IP address
- Null check

Condition operators	Description
StringEquals	Compares a key to a string value based on exact matching (case sensitive).
StringNotEquals	Compares a key to a string value based on negated matching (case sensitive).

Condition operators	Description
StringEqualsIgnoreCase	Compares a key to a string value based on exact matching (ignores case).
StringNotEqualsIgnoreCase	Compares a key to a string value based on negated matching (ignores case).
StringLike	Compares a key to a string value based on exact matching (case sensitive). Can include * and ? wildcard characters.
StringNotLike	Compares a key to a string value based on negated matching (case sensitive). Can include * and ? wildcard characters.
NumericEquals	Compares a key to a numeric value based on exact matching.
NumericNotEquals	Compares a key to a numeric value based on negated matching.
NumericGreaterThan	Compares a key to a numeric value based on "greater than" matching.
NumericGreaterThanEquals	Compares a key to a numeric value based on "greater than or equals" matching.
NumericLessThan	Compares a key to a numeric value based on "less than" matching.
NumericLessThanEquals	Compares a key to a numeric value based on "less than or equals" matching.
Bool	Compares a key to a Boolean value based on "true or false" matching.
IpAddress	Compares a key to an IP address or range of IP addresses.
NotIpAddress	Compares a key to an IP address or range of IP addresses based on negated matching.
Null	Checks if a condition key is present in the current request context.
IfExists	Appended to any condition operator, except the Null condition, to check for absence of that condition key. Returns TRUE if the condition key is not present.

### Supported condition keys

Condition keys	Actions	Description
aws:SourceIp	IP operators	<p>Will compare to the IP address from which the request was sent. Can be used for bucket or object operations.</p> <p><b>Note:</b> If the S3 request was sent through the Load Balancer service on Admin Nodes and Gateways Nodes, this will compare to the IP address upstream of the Load Balancer service.</p> <p><b>Note:</b> If a third-party, non-transparent load balancer is used, this will compare to the IP address of that load balancer. Any X-Forwarded-For header will be ignored because its validity can't be ascertained.</p>
aws:username	Resource/Identity	Will compare to the sender's username from which the request was sent. Can be used for bucket or object operations.
s3:delimiter	s3:ListBucket and s3:ListBucketVersions permissions	Will compare to the delimiter parameter specified in a ListObjects or ListObjectVersions request.
s3:ExistingObjectTag/<tag-key>	s3:DeleteObjectTagging s3:DeleteObjectVersionTagging s3:GetObject s3:GetObjectAcl s3:GetObjectTagging s3:GetObjectVersion s3:GetObjectVersionAcl s3:GetObjectVersionTagging s3:PutObjectAcl s3:PutObjectTagging s3:PutObjectVersionAcl s3:PutObjectVersionTagging	Will require that the existing object has the specific tag key and value.

Condition keys	Actions	Description
s3:max-keys	s3:ListBucket and s3:ListBucketVersions permissions	Will compare to the max-keys parameter specified in a ListObjects or ListObjectVersions request.
s3:object-lock-mode	s3:PutObject	Compares to the object-lock-mode expanded from the request header in the PutObject, CopyObject, and CreateMultipartUpload request.
s3:object-lock-mode	s3:PutObjectRetention	Compares to the object-lock-mode expanded from the XML body in the PutObjectRetention request.
s3:object-lock-remaining-retention-days	s3:PutObject	Compares to the retain-until-date specified in the x-amz-object-lock-retain-until-date request header or computed from the bucket default retention period to make sure that these values are within the allowable range for the following requests: <ul style="list-style-type: none"> <li>• PutObject</li> <li>• CopyObject</li> <li>• CreateMultipartUpload</li> </ul>
s3:object-lock-remaining-retention-days	s3:PutObjectRetention	Compares to the retain-until-date specified in the PutObjectRetention request to ensure that it is within the allowable range.
s3:prefix	s3:ListBucket and s3:ListBucketVersions permissions	Will compare to the prefix parameter specified in a ListObjects or ListObjectVersions request.
s3:RequestObjectTag/<tag-key>	s3:PutObject s3:PutObjectTagging s3:PutObjectVersionTagging	Will require a specific tag key and value when the object request includes tagging.
s3:x-amz-server-side-encryption-customer-algorithm	s3:PutObject	Compares to the sse-customer-algorithm or to the copy-source-sse-customer-algorithm expanded from the request header in the PutObject, CopyObject, CreateMultipartUpload, UploadPart, UploadPartCopy, and CompleteMultipartUpload request.



## Specify variables in a policy

You can use variables in policies to populate policy information when it is available. You can use policy variables in the `Resource` element and in string comparisons in the `Condition` element.

In this example, the variable `${aws:username}` is part of the `Resource` element:

```
"Resource": "arn:aws:s3:::bucket-name/home/${aws:username}/*"
```

In this example, the variable `${aws:username}` is part of the condition value in the condition block:

```
"Condition": {
  "StringLike": {
    "s3:prefix": "${aws:username}/*"
    ...
  },
  ...
}
```

Variable	Description
<code>\${aws:SourceIp}</code>	Uses the <code>SourceIp</code> key as the provided variable.
<code>\${aws:username}</code>	Uses the <code>username</code> key as the provided variable.
<code>\${s3:prefix}</code>	Uses the service-specific <code>prefix</code> key as the provided variable.
<code>\${s3:max-keys}</code>	Uses the service-specific <code>max-keys</code> key as the provided variable.
<code>\${*}</code>	Special character. Uses the character as a literal <code>*</code> character.
<code>\${?}</code>	Special character. Uses the character as a literal <code>?</code> character.
<code>\${\$}</code>	Special character. Uses the character as a literal <code>\$</code> character.

## Create policies requiring special handling

Sometimes a policy can grant permissions that are dangerous for security or dangerous for continued operations, such as locking out the root user of the account. The StorageGRID S3 REST API implementation is less restrictive during policy validation than Amazon, but equally strict during policy evaluation.

Policy description	Policy type	Amazon behavior	StorageGRID behavior
Deny self any permissions to the root account	Bucket	Valid and enforced, but root user account retains permission for all S3 bucket policy operations	Same
Deny self any permissions to user/group	Group	Valid and enforced	Same
Allow a foreign account group any permission	Bucket	Invalid principal	Valid, but permissions for all S3 bucket policy operations return a 405 Method Not Allowed error when allowed by a policy
Allow a foreign account root or user any permission	Bucket	Valid, but permissions for all S3 bucket policy operations return a 405 Method Not Allowed error when allowed by a policy	Same
Allow everyone permissions to all actions	Bucket	Valid, but permissions for all S3 bucket policy operations return a 405 Method Not Allowed error for the foreign account root and users	Same
Deny everyone permissions to all actions	Bucket	Valid and enforced, but root user account retains permission for all S3 bucket policy operations	Same
Principal is a non-existent user or group	Bucket	Invalid principal	Valid
Resource is a non-existent S3 bucket	Group	Valid	Same
Principal is a local group	Bucket	Invalid principal	Valid
Policy grants a non-owner account (including anonymous accounts) permissions to put objects.	Bucket	Valid. Objects are owned by the creator account, and the bucket policy does not apply. The creator account must grant access permissions for the object using object ACLs.	Valid. Objects are owned by the bucket owner account. Bucket policy applies.

## Write-once-read-many (WORM) protection

You can create write-once-read-many (WORM) buckets to protect data, user-defined object metadata, and S3 object tagging. You configure the WORM buckets to allow the creation of new objects and to prevent overwrites or deletion of existing content. Use one of the approaches described here.

To ensure that overwrites are always denied, you can:

- From the Grid Manager, go to **Configuration > Security > Security settings > Network and objects**, and select the **Prevent client modification** checkbox.
- Apply the following rules and S3 policies:
  - Add a PutOverwriteObject DENY operation to the S3 policy.
  - Add a DeleteObject DENY operation to the S3 policy.
  - Add a PutObject ALLOW operation to the S3 policy.



Setting DeleteObject to DENY in an S3 policy does not prevent ILM from deleting objects when a rule such as "zero copies after 30 days" exists.



Even when all of these rules and policies are applied, they don't guard against concurrent writes (see Situation A). They do guard against sequential completed overwrites (see Situation B).

### Situation A: Concurrent writes (not guarded against)

```
/mybucket/important.doc  
PUT#1 ---> OK  
PUT#2 -----> OK
```

### Situation B: Sequential completed overwrites (guarded against)

```
/mybucket/important.doc  
PUT#1 -----> PUT#2 ---X (denied)
```

### Related information

- [How StorageGRID ILM rules manage objects](#)
- [Example bucket policies](#)
- [Example group policies](#)
- [Example session policy](#)
- [Manage objects with ILM](#)
- [Use a tenant account](#)

## Session policy example

Use the following example to build a StorageGRID session policy.

## Example: Set up a session policy that allows object retrieval

In this example, the session's principal is only allowed to retrieve objects from bucket1. All other actions are implicitly denied, except for StorageGRID-specific actions, such as using the [s3:PutOverwriteObject](#) permission. The session policy can be provided as a JSON file while calling the AssumeRole API.

```
{
  "Statement": [
    {
      "Action": "s3:GetObject",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::bucket1/*"
    }
  ]
}
```

## Example bucket policies

Use the examples in this section to build StorageGRID access policies for buckets.

Bucket policies specify the access permissions for the bucket that the policy is attached to. You configure a bucket policy by using the S3 PutBucketPolicy API through one of these tools:

- [Tenant Manager](#).
- AWS CLI using this command (refer to [Operations on buckets](#)):

```
> aws s3api put-bucket-policy --bucket examplebucket --policy
file://policy.json
```

## Example: Allow everyone read-only access to a bucket

In this example, everyone, including anonymous, is allowed to list objects in the bucket and perform GetObject operations on all objects in the bucket. All other operations will be denied. Note that this policy might not be particularly useful because no one except the account root has permissions to write to the bucket.

```

{
  "Statement": [
    {
      "Sid": "AllowEveryoneReadOnlyAccess",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [ "s3:GetObject", "s3:ListBucket" ],
      "Resource":
[ "arn:aws:s3:::examplebucket", "arn:aws:s3:::examplebucket/*" ]
    }
  ]
}

```

### Example: Allow everyone in one account full access, and everyone in another account read-only access to a bucket

In this example, everyone in one specified account is allowed full access to a bucket, while everyone in another specified account is only permitted to List the bucket and perform GetObject operations on objects in the bucket beginning with the `shared/` object key prefix.



In StorageGRID, objects created by a non-owner account (including anonymous accounts) are owned by the bucket owner account. The bucket policy applies to these objects.

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "95390887230002558202"
      },
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::examplebucket",
        "arn:aws:s3:::examplebucket/*"
      ]
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "31181711887329436680"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::examplebucket/shared/*"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "31181711887329436680"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::examplebucket",
      "Condition": {
        "StringLike": {
          "s3:prefix": "shared/*"
        }
      }
    }
  ]
}

```

### Example: Allow everyone read-only access to a bucket and full access by specified group

In this example, everyone including anonymous, is allowed to List the bucket and perform GetObject operations on all objects in the bucket, while only users belonging the group `Marketing` in the specified account are allowed full access.

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::95390887230002558202:federated-
group/Marketing"
      },
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::examplebucket",
        "arn:aws:s3:::examplebucket/*"
      ]
    },
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": ["s3:ListBucket", "s3:GetObject"],
      "Resource": [
        "arn:aws:s3:::examplebucket",
        "arn:aws:s3:::examplebucket/*"
      ]
    }
  ]
}

```

### Example: Allow everyone read and write access to a bucket if client in IP range

In this example, everyone, including anonymous, is allowed to List the bucket and perform any Object operations on all objects in the bucket, provided that the requests come from a specified IP range (54.240.143.0 to 54.240.143.255, except 54.240.143.188). All other operations will be denied, and all requests outside of the IP range will be denied.

```

{
  "Statement": [
    {
      "Sid": "AllowEveryoneReadWriteAccessIfInSourceIpRange",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [ "s3:*Object", "s3:ListBucket" ],
      "Resource":
[ "arn:aws:s3:::examplebucket", "arn:aws:s3:::examplebucket/*" ],
      "Condition": {
        "IpAddress": { "aws:SourceIp": "54.240.143.0/24" },
        "NotIpAddress": { "aws:SourceIp": "54.240.143.188" }
      }
    }
  ]
}

```

### Example: Allow full access to a bucket exclusively by a specified federated user

In this example, the federated user Alex is allowed full access to the `examplebucket` bucket and its objects. All other users, including 'root', are explicitly denied all operations. Note however that 'root' is never denied permissions to Put/Get/DeleteBucketPolicy.



```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::95390887230002558202:federated-user/Alex"
      },
      "Action": [
        "s3:*"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket",
        "arn:aws:s3:::examplebucket/*"
      ]
    },
    {
      "Effect": "Deny",
      "NotPrincipal": {
        "AWS": "arn:aws:iam::95390887230002558202:federated-user/Alex"
      },
      "Action": [
        "s3:*"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket",
        "arn:aws:s3:::examplebucket/*"
      ]
    }
  ]
}

```

## Example: PutOverwriteObject permission

In this example, the `Deny` Effect for `PutOverwriteObject` and `DeleteObject` ensures that no one can overwrite or delete the object's data, user-defined metadata, and S3 object tagging.

```

{
  "Statement": [
    {
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:PutOverwriteObject",
        "s3:DeleteObject",
        "s3:DeleteObjectVersion"
      ],
      "Resource": "arn:aws:s3:::wormbucket/*"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::95390887230002558202:federated-
group/SomeGroup"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::wormbucket"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::95390887230002558202:federated-
group/SomeGroup"
      },
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::wormbucket/*"
    }
  ]
}

```

## Example group policies

Use the examples in this section to build StorageGRID access policies for groups.

Group policies specify the access permissions for the group that the policy is attached to. There is no `Principal` element in the policy because it is implicit. Group policies are configured using the Tenant Manager or the API.

## Example: Set group policy using Tenant Manager

When you add or edit a group in the Tenant Manager, you can select a group policy to determine which S3 access permissions the members of this group will have. See [Create groups for an S3 tenant](#).

- **No S3 Access:** Default option. Users in this group don't have access to S3 resources, unless access is granted with a bucket policy. If you select this option, only the root user will have access to S3 resources by default.
- **Read Only Access:** Users in this group have read-only access to S3 resources. For example, users in this group can list objects and read object data, metadata, and tags. When you select this option, the JSON string for a read-only group policy appears in the text box. You can't edit this string.
- **Full Access:** Users in this group have full access to S3 resources, including buckets. When you select this option, the JSON string for a full-access group policy appears in the text box. You can't edit this string.
- **Ransomware Mitigation:** This sample policy applies to all buckets for this tenant. Users in this group can perform common actions, but can't permanently delete objects from buckets that have object versioning enabled.

Tenant Manager users who have the Manage all buckets permission can override this group policy. Limit the Manage all buckets permission to trusted users, and use Multi-Factor Authentication (MFA) where available.

- **Custom:** Users in the group are granted the permissions you specify in the text box.

## Example: Allow group full access to all buckets

In this example, all members of the group are permitted full access to all buckets owned by the tenant account unless explicitly denied by bucket policy.

```
{
  "Statement": [
    {
      "Action": "s3:*",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::*"
    }
  ]
}
```

## Example: Allow group read-only access to all buckets

In this example, all members of the group have read-only access to S3 resources, unless explicitly denied by the bucket policy. For example, users in this group can list objects and read object data, metadata, and tags.

```

{
  "Statement": [
    {
      "Sid": "AllowGroupReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:ListBucket",
        "s3:ListBucketVersions",
        "s3:GetObject",
        "s3:GetObjectTagging",
        "s3:GetObjectVersion",
        "s3:GetObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::*"
    }
  ]
}

```

### Example: Allow group members full access to only their "folder" in a bucket

In this example, members of the group are only permitted to list and access their specific folder (key prefix) in the specified bucket. Note that access permissions from other group policies and the bucket policy should be considered when determining the privacy of these folders.

```

{
  "Statement": [
    {
      "Sid": "AllowListBucketOfASpecificUserPrefix",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::department-bucket",
      "Condition": {
        "StringLike": {
          "s3:prefix": "${aws:username}/*"
        }
      }
    },
    {
      "Sid": "AllowUserSpecificActionsOnlyInTheSpecificUserPrefix",
      "Effect": "Allow",
      "Action": "s3:*Object",
      "Resource": "arn:aws:s3:::department-bucket/${aws:username}/*"
    }
  ]
}

```

## Copyright information

Copyright © 2026 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

## Trademark information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.