SAS® Viya® 3.5 Administration: General Authorization

General Authorization: Overview . . . 2

General Authorization: How To (Authorization Window) . . . 2
- Introduction . . . 2
- Navigation . . . 3
- Examine Access . . . 3
- Provide Access . . . 4
- Limit Access . . . 5
- Add a Condition . . . 5
- Edit a Condition . . . 5
- Delete a Condition . . . 6
- Remove a Direct Setting . . . 6
- Identify the Source of Effective Access . . . 6
- Details for Share-Based Authorization Rules . . . 7

General Authorization: How To (Rules Page) . . . 8
- Introduction . . . 8
- Navigation . . . 8
- Rules Page . . . 8
- Add a Rule . . . 9
- Edit a Rule . . . 9
- Copy a Rule . . . 10
- Delete a Rule . . . 10
- Edit a Condition . . . 10
- Delete a Condition . . . 11
- Locate a Particular Rule . . . 11
- Manage Share-Based Rules . . . 12

General Authorization: How To (CLI) . . . 13
- Getting Access Information . . . 13
- Managing Rules . . . 13
- Managing Rules (Bulk Approach) . . . 14
- Details and Tips . . . 15

General Authorization: Concepts . . . 16
- Scope . . . 16

Terms . . . 17
Principals . . . 17
Inheritance . . . 18
Permissions . . . 18
Rule Targets . . . 19
Rule Attributes . . . 22
Rule Conditions . . . 23
Rule Precedence . . . 26
Authorization Explanations . . . 27
Access to Authorization Rules . . . 27
Sharing: Details for Administrators . . . 28
Configuration Properties: Authorization Service . . . 31

General Authorization: Guidelines . . . **31**

General Authorization: Troubleshooting . . . **32**
Unexpected Outcomes . . . 32
Unavailable Principals . . . 32
Unrecognized Principals . . . 32
Unintended Loss of Access . . . 33
A Deleted Rule Reappears . . . 34
Inability to Browse to an Object . . . 34
Not All Contributing Rules Are Visible . . . 34
A Pair of Predefined Rules Is Difficult to Interpret . . . 34

General Authorization: Interfaces . . . **35**
Interfaces to General Authorization . . . 35

General Authorization: Overview

To learn about the general authorization system, see Concepts.

To manage access, use the interface that best meets your needs. Here are suggestions:

- To adjust access to content (such as folders and reports), use the Authorization window.
- To manipulate rules directly, use the Rules page or the command-line interface.

General Authorization: How To (Authorization Window)

Introduction

These instructions explain how to set permissions on folders, reports, and other content objects using the Authorization window.

Navigation

Here is one way to access the Authorization window for content objects:

1. In the applications menu (☰), under **ADMINISTRATION**, select **Manage Environment**.
2. In the vertical navigation bar in SAS Environment Manager, click 📄.
3. On the **Content** page, locate and select the object.
4. In the toolbar at the top of the navigation pane, click 🔒, and select **View Authorization** or **Edit Authorization**.

   Note: Some items on the **Content** page do not participate in general authorization. For example, you cannot set permissions on a virtual folder.

   Note: If **Edit Authorization** is not available, you are not authorized to modify access to the selected object.

Examine Access

For each principal and permission, the following icons describe effective (net) access to the current object:

Icon	Meaning
✓	Authorized
◉	Conditional
⊘	Not Authorized
○	Unknown
◆	Direct (indicates that effective access comes from a direct setting)*
⤴	Not Authorized (but can share)**

* This icon indicates that effective access comes from a permission that is directly assigned to the specified principal on the current object. If a direct setting exists but does not win (does not determine effective access), a diamond is not displayed.

** This icon is applicable to only the **Secure** and **Secure (convey)** columns. This icon is displayed only if Secure access is not granted and sharing is possible. If Secure access is authorized, only the Authorized icon is displayed, because the ability to share is inherent in Secure access.

The scope of the display is as follows:

- There is always a row for Authenticated Users.
- There is always a row for you, the currently connected user who is using the display.

- There is a row for each principal that is assigned to a rule that affects access to the current object. The exception is that internal service principals (for example, sasapp or sas.folders) are not displayed in the Authorization window.
- If you add an identity and do not give that identity at least one direct setting, that identity is automatically removed from the display.
- You cannot directly remove a row. If you remove all direct settings for an identity and there is no other reason for that identity to be displayed, that identity is automatically removed from the display.
- For a non-container object (such as a report), only the Read, Update, Delete, and Secure permissions are displayed. The Create permission is not applicable to an individual content object.
- For a container (such as a folder), two sets of permissions are displayed:
  - The first set of permissions affects access to the object, including the ability to add members to and remove members from the object. This set of permissions has no effect on the folder's members.
  - The second set of permissions affects the access that this object conveys to its child members. See Inheritance.

TIP An effective access value of Not Authorized on the conveyed side of a folder's Authorization window does not guarantee that access to child members is not authorized. A direct setting on the child or another influencing rule might provide access to the child member. For example, when you create a top-level folder, the effective access values on the conveyed side for SAS Administrators are all Not Authorized. However, SAS Administrators does have effective access to a folder that you add below the top-level folder. That access comes from a predefined rule that gives SAS Administrators access to all folders.

Provide Access

1. Open the Edit Authorization window for the target object.
2. If the principal that you want to work with is not already listed, click .

   Note: If guest access is enabled, you must select either **Add Identities** or **Add Guest** after you click .
3. Click an effective access icon (for example,).
4. In the pop-up window, select **Grant** as the direct setting.

   Note: If you cannot change a direct setting, you do not have Secure permission for the current object.
5. In the Edit Authorization window, click **Preview**. Examine the impact of your unsaved change.

   Note: If there is a relevant prohibit setting anywhere in the system, that setting has precedence over the direct grant that you added. In that case, the effective (net) result is **Not Authorized** (), and a diamond is not displayed.
6. Click **Save**.

Limit Access

Any access that is not granted is implicitly denied. The preferred approach is to grant selectively and to avoid use of prohibit settings.

If you must add a prohibit setting, make sure that you do not inadvertently block your own access, particularly for the Read and Secure permissions. If you do block your own access, see Troubleshooting.

CAUTION! A prohibit setting has absolute precedence, even if a more specific grant setting exists.

Add a Condition

To provide access within a particular scope or set of circumstances, add a condition.

1. Open the Edit Authorization window for the target object.
2. If the principal that you want to work with is not already listed, click .

   Note: If guest access is enabled, you must select either **Add Identities** or **Add Guest** after you click .
3. Click an effective access icon (for example,).
4. In the pop-up window, select **Conditional Grant**.

   Note: A conditional prohibit setting does not provide access. A conditional prohibit setting blocks all access within its scope, regardless of any more specific grant settings. A conditional prohibit setting can limit access that is provided by a grant or conditional grant setting.
5. In the Condition window, create an expression that specifies the scope and circumstances in which access is granted. Your syntax is validated when you click **OK**. See Rule Conditions.
6. In the Edit Authorization window, click **Preview**. Examine the impact of your unsaved change.
7. Click **Save**.

Edit a Condition

1. In the Edit Authorization window for an object, click the effective access icon for the direct conditional setting that you want to modify.
2. In the pop-up window, next to the **Conditional Grant** or **Conditional Prohibit** direct setting, click .
3. In the Condition window, edit the expression. Your syntax is validated when you click **OK**. See Rule Conditions.
4. In the Edit Authorization window, click **Preview**. Examine the impact of your unsaved change.
5. Click **Save**.

Delete a Condition

1. In the Edit Authorization window for an object, click the effective access icon for the direct conditional setting that you want to delete.
2. In the pop-up window, next to the **Conditional Grant** or **Conditional Prohibit** direct setting, click ≤≠≥.
3. In the Condition window, delete the expression. Click **OK**.
4. In the Edit Authorization window, click **Preview**. Examine the impact of your unsaved change.
5. Click **Save**.

Remove a Direct Setting

1. Open the Edit Authorization window for an object.
2. In the cell that has the direct setting that you want to remove, click the effective access icon. In the pop-up window, select **(none)** as the direct setting.

   Note: If you cannot change the direct setting, you do not have Secure permission for the current object.
3. In the Edit Authorization window, click **Preview**. Examine the impact of your unsaved change.

   Note: Any identities that are no longer principals are automatically removed.
4. Click **Save**.

Identify the Source of Effective Access

To determine which rules and shares contribute to a particular effective access result, examine the origins information for that result.

1. Open the View Authorization window for the target content object.
2. Click the effective access icon for which you want origins information.
3. In the pop-up window, the **Contributing Rules** tab provides a read-only display of all applicable rules, except share-based rules. Here are tips:
   - If the Edit Authorization window contains pending changes, the **Contributing Rules** tab is disabled.
   - To view additional details, add columns to the table. Click ⋮ and select **Manage columns**.
   - To directly modify a rule, use the Rules page.
4. In the pop-up window, the **Contributing Shares** tab provides a read-only display of all relevant shares. Here are tips:

- If sharing is disabled, the **Contributing Shares** tab is not displayed.
- To view additional details, add columns to the table.
- To directly modify a rule, use the Rules page.
- To manage shares, see “Manage Share-Based Rules”.

Details for Share-Based Authorization Rules

Direct Settings

In the Authorization window, a share is not considered a direct setting.

A share does not cause a diamond to be displayed.

You cannot modify shares in the Authorization window.

Effective Access

In the Authorization window, effective access information reflects shares as follows:

- Effective access information reflects any access that is provided by shares.
- In the **Secure** and **Secure (convey)** columns, the Share icon indicates that sharing is possible even though Secure access is not granted.

  Note: If Secure access is granted, the Share icon is not displayed. The ability to share is inherent in Secure access. See “Examine Access” on page 3.

Contributing Shares

After you click an effective access icon in the Authorization window, a pop-up window that includes a **Contributing Shares** tab is displayed. The tab provides a read-only list of the shares that are relevant to the selected effective access result.

A share is relevant if it meets all of the following criteria:

- The share specifies the current object (or a parent of that object) as the target.
- The share specifies the current principal (or a group to which that principal belongs) as the recipient.
- The share type is relevant for the selected permission. For example, for the Update and Delete permissions, the `readEditShare` and `readEdit` types are relevant, but the `read` and `readShare` types are not relevant.

  Note: For the Secure permission, the `readShare` and `readEditShare` types are treated as relevant only because those types provide the ability to reshare. Sharing never creates a grant of the Secure permission.

  Note: To add the **Share Type** column to the display, click and select **Manage columns**.

If Re-sharing Is Disabled

- In the Authorization window, no sharing-related information is displayed for the Secure permission.

- There is no **Contributing Shares** tab for the Secure permission.

If Sharing Is Disabled

- In the Authorization window, no sharing-related information is displayed.
- SAS Drive hides all actions and information that are related to sharing.
- SAS Drive provides direct access to the Authorization window for authorized users.

General Authorization: How To (Rules Page)

Introduction

These instructions explain how to directly manage general authorization rules using SAS Environment Manager.

Navigation

In the applications menu (☰), under **ADMINISTRATION**, select **Manage Environment**. In the vertical navigation bar, select ⚿.

The **Rules** page is an advanced interface. It is available to only SAS Administrators. You can use a simpler interface to set permissions on content such as folders and reports.

Rules Page

Use the **Rules** page to manage authorization rules directly. Here are examples:

- View and filter rules.
- Enable and disable rules.
- Replace the principal in a rule.
- View and edit a rule's description or reason.
- Use an existing rule as the basis for a new rule.
- Work with rules that affect access to functionality.
- View conditions for multiple rules at the same time.

Here are additional details:

- To ensure that all rules that should be visible to you are displayed, refresh the display and click **Reset all** in the **Rules Filter** pane.

- On the **Rules** page, you cannot see rules that are directly assigned to objects for which you lack the Secure permission.
- The search field searches only the **Object URI**, **Description**, **Reason**, and **Condition** columns.
- To add, remove, or reorder columns, click , and select **Manage columns**.
- You cannot sort the values within a column.
- You can use the **Rules Filter** pane to view a subset of rules. Some filters take effect immediately, other filters take effect after you click **Apply**.
- You can clear a filter by clicking its **Reset** link. You can clear all filters by clicking **Reset all** at the top of the pane.
- The **Guest** principal type is always listed, regardless of whether guest access is enabled.
- Display names for users and groups are not available on the **Rules** page.
- For details about rule attributes, see Rule Attributes.

Add a Rule

1. On the **Rules** page, click .
2. In the New Rule window, provide values for at least the required attributes. Here are tips:
   - In some fields, you can click an icon to browse instead of directly entering a value.
   - If a warning indicates that the **Principal** value cannot be validated, make sure the value is an ID, not a display name. If the principal is a service account (such as sasapp or sas.folders), you can ignore the warning.
   - To populate the list of permissions, use the **Clear All**, **Select All**, and **Choose** buttons.
3. Click **Save**.
4. On the **Rules** page, right-click the new rule, and select **Properties**. Verify that the attributes of the new rule are as you intended.
5. If the rule affects a content object (such as a folder or report), use the Authorization window to verify that the results are as you intended.

Edit a Rule

1. On the **Rules** page, right-click a rule, and select **Edit**.

   Note: You cannot edit a share-based rule. See “Sharing: Details for Administrators”.
2. In the Edit Rule window, modify attributes as needed. Here are tips:
   - If a warning indicates that the **Principal** value cannot be validated, make sure the value is an ID, not a display name. If the principal is a service account (such as sasapp or sas.folders), you can ignore the warning.
   - If the rule does not have a condition, an **Add Condition** button is present. If the rule has a condition, an **Edit Condition** button is displayed.

3 Click **Save**.

4 On the **Rules** page, right-click the rule, and select **Properties**. Verify that the attributes of the new rule are as you intended.

5 If the rule affects a content object (such as a folder or report), use the Authorization window to verify that the results are as you intended.

Copy a Rule

1 On the **Rules** page, right-click a rule, and select **Copy**.

Note: You cannot copy a share-based rule. See "Sharing: Details for Administrators".

2 In the New Rule window, modify attributes as needed.

3 Click **Save**.

4 On the **Rules** page, right-click the new rule, and select **Properties**. Verify that the attributes of the new rule are as you intended.

5 If the rule affects a content object (such as a folder or report), use the Authorization window to verify that the results are as you intended.

Delete a Rule

1 On the **Rules** page, right-click a rule, and select **Delete**.

2 In the confirmation window, click **Delete**.

3 If the rule affects a content object (such as a folder or report), use the Authorization window to verify that the results are as you intended.

Edit a Condition

1 On the **Rules** page, right-click a rule, and select **Edit**.

2 In the Edit Rule window, click **Edit Condition**.

Note: If a rule does not have a condition, an **Add Condition** button is present. If a rule has a condition, an **Edit Condition** button is displayed.

3 In the Edit Condition window, edit the expression. Your syntax is validated when you click **OK**.

4 Click **Save**.

5 If the rule affects a content object (such as a folder or report), use the Authorization window to verify that the results are as you intended.

Delete a Condition

1. On the **Rules** page, right-click a rule, and select **Edit**.
2. In the Edit Rule window, click **Edit Condition**.

   Note: If a rule does not have a condition, an **Add Condition** button is present. If a rule has a condition, an **Edit Condition** button is displayed.
3. In the Edit Condition window, delete the expression.
4. Click **Save**.
5. On the **Rules** page, verify that the condition no longer exists. Right-click on the rule, select **Properties**, and verify that the **Condition** field is blank.
6. If the rule affects a content object (such as a folder or report), use the Authorization window to verify that the results are as you intended.

Locate a Particular Rule

Here are general tips:

- Filter requirements are cumulative. For example, if you set two filters, only rules that meet both criteria are displayed.
- Remember to click **Apply** after you set or modify certain filters.
- You can search for rules that contain specified text in the ObjectUri, Description, Reason, or Condition field. The search is not case-sensitive. The search looks for the specified text in any of the four supported fields.

Here are tips for locating a rule by date:

- In the **Rules Filter** pane, under **Date Modified**, click to select a date or date range.
- Each rule's **Date Modified** value indicates when the rule was created or most recently modified.
- To add the **Date Modified** column to the display, click , and select **Manage columns**.

Here are tips for locating a rule by identity:

- The **Modified By** filter is based on who created or last updated a rule.
- To add the **Modified By** column to the display, click , and select **Manage columns**.
- The **Principal** filter is based on who a rule is assigned to.
- Both the **Modified By** and the **Principal** filters use ID values, not display name values. For example, to display only rules that were created by userA, specify `usera` in the **Modified By** filter. To display only rules that are assigned to the SAS Administrators group, specify **SASAdministrators** in the **Principal** filter.
- Rules that are predefined or generated can have a **Modified By** value that does not correspond to a user or group that is known to the identities service.

Here are tips for locating a rule by the URI of the rule's target:

- To view only those rules that target a specific content object, use the technique that is appropriate for the type of URI, as follows:
  - Browse content objects for the object URI for a rule target. In the drop-down list under **Object URI**, select **URI**.
  - Browse container objects for the container URI for a rule target. In the drop-down list under **Container URI**, select **URI**.
- To view only those rules that do not specify an object URI, select **(blank URI)** from the drop-down list under **Object URI**.
- To view rules that either specify `/**` as the object URI or do not specify an object URI, select **(global URI)** from the drop-down list under **Object URI**.
- To view only those rules that do not specify a container URI, select **(blank URI)** from the drop-down list under **Container URI**.

For tips for locating share-based rules, see “Manage Share-Based Rules” on page 12.

Manage Share-Based Rules

Administrators can view and delete share-based rules on the **Rules** page in SAS Environment Manager. When you delete a share-based rule, the corresponding share is automatically deleted along with the rule.

On the **Rules** page, the following constraints apply to managing share-based rules:

- You cannot add share-based rules. Sharing is primarily a user-driven activity. Share-based rules are automatically created when users share content in SAS Drive.
- You cannot edit or copy share-based rules. Any changes to share-based rules must be coordinated with changes to the associated shares, and must conform to requirements that are specific to share-based rules.

All share-based rules are created with generated text in the **Description** field. Here is an example:

The user "userA" shared an object with the specified principal. (This is a share-based rule.)

On the **Rules** page, you can filter for share-based rules as follows:

- To display only share-based rules, enter the following text in the **Description** filter:

  This is a share-based rule
- To display only share-based rules for a particular share recipient, specify the preceding **Description** filter and specify the share recipient in the **Principal** filter.
- To display only share-based rules that were generated by a particular user, include that user’s ID in the **Description** filter text as follows:

  The user "userID" shared an object with the specified principal

> **TIP** You cannot instead use the **Modified by** filter, because all share-based rules are generated with a **Modified by** value of `sas.authorization`. Do not assume that all rules that have a **Modified by** value of `sas.authorization` are share-based rules.

General Authorization: How To (CLI)

The following examples assume that you have already signed in to SAS Viya at the command line. See the preliminary instructions in *SAS Viya Administration: Using the Command-Line Interfaces*.

Getting Access Information

Example: Show detailed properties of a specified rule.

```
sas-admin authorization show-rule --id d85144aa-79dc-4852-b949-645cc5ff8ffc --details
```

Example: Show effective access for a specified object URI. To return information about contributing rules, you must specify fulljson output in your profile. Specifying the `output` global option in the CLI command inline is insufficient.

```
sas-admin authorization explain --target-uri /SASHome/**
```

Example: List all the rules in the deployment.

```
sas-admin authorization list-rules
```

Managing Rules

Example: Give groupA Read access to reportA.

```
sas-admin authorization authorize --permissions Read --group groupA
--object-uri /reports/reports/33db163a-716e-4980-a5bc-6c42a0278c40
```

Example: Provide guest access to reportA.

```
sas-admin authorization authorize --permissions Read --guest
--object-uri /reports/reports/33db163a-716e-4980-a5bc-6c42a0278c40
```

Example: Grant Authenticated Users Read access to folderA and its child members.

```
sas-admin authorization authorize --permissions Read --authenticated-users
--object-uri /folders/folders/2414f911-d276-4357-8550-fcf03753c9e7/**
--container-uri /folders/folders/2414f911-d276-4357-8550-fcf03753c9e7
```

Example: Delete a rule.

```
1 sas-admin authorization show-rule --id d85144aa-79dc-4852-b949-645cc5ff8ffc --details
2 sas-admin authorization remove-rule --id d85144aa-79dc-4852-b949-645cc5ff8ffc
```

1 Review the rule's properties so that you are certain you are deleting the correct rule.

2 Delete the rule.

Example: Change the principal in an existing rule so that the rule is assigned to Group B, which has `groupB` as its ID.

```
1 sas-admin authorization show-rule --id cd75a376-c5d4-4951-9e57-cf441610628c --details
2 sas-admin authorization update-rule --id cd75a376-c5d4-4951-9e57-cf441610628c --group groupB
```

1 Review the rule's properties so that you are certain you are modifying the correct rule.

2 Modify the rule.

Example: Include the Update and Delete permissions in an existing rule that already grants the Read permission.

```
1 sas-admin authorization show-rule --id cd75a376-c5d4-4951-9e57-cf441610628c --details
2 sas-admin authorization update-rule --id cd75a376-c5d4-4951-9e57-cf441610628c
--grant --permissions Read,Update,Delete
```

1 Review the rule's properties so that you are certain you are modifying the correct rule.

2 Modify the rule.

Example: Edit the description in an existing rule.

```
1 sas-admin authorization show-rule --id 0e8a6ce7-e51a-40cc-aeda-ee2a5efb53ca --details
2 sas-admin authorization update-rule --id 0e8a6ce7-e51a-40cc-aeda-ee2a5efb53ca
--description 'This is a revised description.'
```

1 Review the rule's properties so that you are certain you are modifying the correct rule.

2 Modify the rule.

Managing Rules (Bulk Approach)

Adding Rules

Example: Add multiple rules, as specified in a referenced JSON file.

```
sas-admin authorization create-rules --file newrules.json
```

Note: The path to the input file is relative to the location of the CLI executable (sas-admin).

The input file is in JSON format. Here is an example that adds two rules:

```
[
   {
    "op": "add",
    "value": {
      "description": "Description for this rule.",
      "objectUri": "/folders/folders/156f833a-31ac-40f8-bf78-82e738daef36",
      "permissions": [
        "Secure"
      ],
      "principalType": "user",
      "principal": "userC",
      "type": "grant"
      }
   },
   {
    "op": "add",
    "value": {
      "description": "Description for this rule.",
      "objectUri": "/folders/folders/156f833a-31ac-40f8-bf78-82e738daef36",
      "permissions": [
       "Read",
```

```
        "Update",
        "Delete"
      ],
      "principalType": "group",
      "principal": "groupD",
      "type": "grant"
    }
   }
]
```

Because an ID for each rule is not specified in the preceding example, the rule ID is generated. For details about available parameters and values, see the Authorization REST API documentation on developer.sas.com.

Deleting Rules

Example: Delete multiple rules, as specified in a referenced JSON file.

```
sas-admin authorization remove-rules --file oldrules.json
```

Note: The path to the input file is relative to the location of the CLI executable (sas-admin).

The only required content for the remove-rules command is the ID of each rule that you want to remove. Here is an example of the input file for a remove-rules command:

```
[
   {
    "op": "add",
    "value": {
      "id": "rule123456788"
    }
   },
   {
    "op": "add",
    "value": {
      "id": "rule123456789"
    }
   }
]
```

Notice that even though this file is used in a removal command, the value of the op parameter is **add**.

TIP If you specify a rule ID in an input file that you use to create rules, you can reference the same file to remove the rules. The remove-rules command ignores values other than the rule ID.

Details and Tips

- To assign a rule to a principal type, use one of the following options:

Principal Type	Option
Guest	--guest
Authenticated Users	--authenticated-users

Principal Type	Option
Everyone	`--everyone`

- To assign a rule to a particular identity, you must provide a user ID or a group ID, not a name. For example, to assign a rule to the SAS Administrators custom group, specify: `--group SASAdministrators`

  CAUTION! The user ID and the group ID that you provide are not validated. Make sure the IDs that you provide are accurate.
- You can obtain the ID for a user or group from the **Users** page in SAS Environment Manager.
- You can obtain the objectURI for a content object (such as a report) from the **Content** page in SAS Environment Manager. Select the object in the navigation pane. On the right, the **URI** field in the **Basic Properties** section contains the object URI. To target the object URI for a content object (such as a report) or a container (such as a folder), append a suffix.
- You can obtain the ID for a rule from the **Rules** page in SAS Environment Manager. Right-click a rule and select **Properties**. The last field in the Properties window contains the rule's ID.
- When you use the show-rule command, always specify that you want details to be returned. Some of the fields that can be essential to interpreting a rule are excluded from the default response. For example, a condition is not included in the default response.
- When you use the update-rule command, specify only the options for the rule properties that you want to modify. For any option that you specify in the update-rule command, provide the complete replacement value or values.
- When you use the explain command, the returned information indicates the effective (net) access of each relevant principal for all permissions.

  Note: In the output from the explain command, the `grant` and `prohibit` values indicate effective (net) access, not direct settings. For example, a `prohibit` value in the output from the explain command is usually caused by the lack of any relevant grant, rather than by the existence of a relevant Prohibit rule.
- Enabling or disabling guest access involves more than running the enable-guest-access or disable-guest-access command.
- Throughout this topic, the term *rule* refers to an authorization rule in the general authorization system. To manage access to CAS objects (such as caslibs and tables), use the CAS authorization system.

General Authorization: Concepts

Scope

The authorization service provides the general authorization system, which manages access to the following types of resources:

- content, such as folders and reports

- functionality, such as applications, features, and services

Note: Access controls for CAS objects, such as caslibs and tables, are outside the scope of the general authorization system.

Terms

authorization rule
: a composite of authorization elements.

  Example: A rule grants groupA the Read permission for folderA.

condition
: in an authorization rule, the constraint expression. Most authorization rules do not include a condition.

  Example: `currentUser() == #preferenceOwner`

effective access
: for a particular principal, permission, and target, a context-neutral description of the net result of all relevant authorization rules. Effective access does not incorporate evaluation of any conditions.

  Values: Authorized, Not Authorized, Conditional

outcome
: in an access request, the authorization decision.

  Values: Authorized, Not Authorized

permission
: in an authorization rule, a type of access.

  Values: Add, Create, Delete, Read, Remove, Secure, Update

principal
: in an authorization rule, the user, group, or construct to which a rule is assigned.

  Examples: UserA, GroupA, Authenticated Users

setting
: an indication of whether (and to what extent) a rule provides access. Setting is a compound of rule type and presence (or absence) of a condition. Setting is not a rule attribute.

  Values: Grant, Conditional Grant, Prohibit, Conditional Prohibit

target
: in an authorization rule, the affected resource or resources (for example, an individual object, a set of objects, a service, or a service endpoint).

  Examples: folderA, reportA

Principals

The following types of principal are supported:

- Users (individual authenticated users and service accounts)
- User groups (custom groups and groups in your authentication provider)
- Authenticated Users (the principal type that represents all authenticated users)

- Everyone (the principal type that represents all principals)
- Guest (the principal type that facilitates guest access)

Note: Guest is part of Everyone but is not part of Authenticated Users.

Note: When a user or group is deleted, rules that are assigned to that principal are not automatically deleted. Such rules are reused if a new principal of the same type and ID is created. The general authorization system does not have an automated cleanup process for orphaned rules.

Inheritance

Access flows through a hierarchy of containers. Each container conveys settings to its child members. Each child member inherits settings from its parent container.

A rule that targets the object aspect of a container (the container's objectUri attribute) has different effects than a rule that targets the container aspect of a container (the container's containerUri attribute). This separation enables you to provide Write access to the objects in a container without providing Write access to the container itself.

The following table provides details, using folderA as an example container.

Rule Target	Potential Impact of the Rule
folderA (as an object)	Can affect the ability to read, update, or delete folderA. Can affect the ability to add or remove members for folderA. Settings are not conveyed to the objects within folderA.
folderA (as a container)	Settings are conveyed to folderA's child members.
folderA (as an object and as a container)	Can affect the ability to read, update, or delete folderA. Can affect the ability to add or remove members for folderA. Container settings are conveyed to folderA's child members.

Permissions

Table 1 *Definition of Each Permission*

Permission	Affected Activity
Create	Create a new object. This permission is applicable for a service, service endpoint, or media type.
Read	Read an object.
Update	Update or edit an object.
Delete	Delete an object.

Permission	Affected Activity
Secure	Set permissions on an object (manipulate the object's direct rules).
Add	Add a member to a container. This permission is required on the container's objectUri.
Remove	Move a member out of a container. This permission is required on the container's objectUri. **Note:** The folders service requires this permission on a folder to *move* a member, but not to *delete* a member. However, some clients impose an additional requirement, allowing only those users who have the Remove permission on a folder to delete members from that folder.

Table 2 *Default Mapping to HTTP Verbs*

HTTP Verb	Permission
POST	Create
DELETE	Delete
GET, OPTIONS, HEAD	Read
PUT, PATCH	Update

Rule Targets

About Rule Targets

Each rule affects a target resource (or set of resources), as identified in the rule's target-related attributes (objectUri, containerUri, and the media type properties). A rule can specify any, all, or none of the target-related attributes.

CAUTION! A rule that does not specify at least one target-related attribute affects access to all resources.

About Target URIs

TIP Do not confuse uniform resource identifiers (URIs) and SAS folder locations. Although both constructs use paths to reference resources, the two constructs are entirely independent and distinct. For example, a report might have a URI of `/reports/reports/qwe3429ryjw12567` and a folder location of `/SAS Content/Public/reportA`.

A target URI is the value in an authorization rule's objectUri or containerUri attribute. A target URI references a resource such as a service, service endpoint, or content object. The authorization service sees target URIs as strings. The authorization service does not know whether a particular URI represents an application, a feature, an object, or a collection of objects.

Authorization decisions for a particular resource evaluate all rules that have target URIs that match the requested URI. To compare a requested URI to the target URIs in authorization rules, the

authorization service uses Ant-style pattern matching. In that pattern matching, the authorization service supports a wildcard character, so that the target URI in a rule can be more general than the target URI in an individual request. Here are details:

- A single wildcard (`*`) matches a single element in a URI path. For example, it matches the name of a service or the ID of an object.
- A double wildcard (`**`) matches any number of consecutive elements in a URI path. For example, it matches a forward slash (`/`), the name of a service, the ID of an object, a multi-element path (`/lmn/xyz/`), or the absence of any element (null).

The following tables provide examples. The first table uses fictional URIs to demonstrate the principles. The other tables use selected URIs to illustrate how the principles apply in context.

Table 3 *Scope of a Target URI: Principles*

Rule's Target URI	Rule's Relevance
`/**`	Relevant to all requests.
`/abc`	Relevant to requests for exactly this URI: `/abc`.
`/abc/`	Relevant to requests for exactly this URI: `/abc/`.
`/abc/*`	Relevant to requests for URIs such as `/abc/lmn` (not `/abc`, `/abc/`, or URIs such as `/abc/lmn/` and `/abc/lmn/xyz`).
`/abc/*/xyz`	Relevant to requests for URIs such as `/abc/lmn/xyz` (not `/abc/xyz`).
`/abc/**`	Relevant to requests for `/abc`, `/abc/`, and URIs such as `/abc/lmn`, `/abc/lmn/`, and `/abc/lmn/xyz`.
`/abc/**/xyz`	Relevant to requests for `/abc/xyz`, and URIs such as `/abc/lmn/xyz` and `/abc/lmn/rst/xyz`.

Table 4 *Scope of a Target URI: Examples from Reports Service*

Rule's Target URI	Rule's Scope
`/reports`	The root of the reports service. Relevant to requests that omit a trailing slash.
`/reports/`	The root of the reports service. Relevant to requests that include a trailing slash.
`/reports/*`	All first-level endpoints within the service (not the service root, lower-level endpoints, or individual reports).
`/reports/*/`*report-ID*	All first-level endpoints within the service (for the specified report).
`/reports/**`	All reports, all endpoints within the service, and the service root.
`/reports/**/`*report-ID*	All endpoints within the service (for the specified report).
`/reports/reports/*`	All reports, and all second-level endpoints beneath the reports endpoint.

Rule's Target URI	Rule's Scope
`/reports/reports/`*report-ID*	The specified report.

Table 5 *Scope of a Target URI: Examples from Model Repository Service*

Rule's Target URI	Rule's Scope
`/modelRepository`	The root of the model repository service. Relevant to requests that omit a trailing slash.
`/modelRepository/`	The root of the model repository service. Relevant to requests that include a trailing slash.
`/modelRepository/*`	All first-level endpoints within the service (not the service root, lower-level endpoints, or individual objects).
`/modelRepository/**`	All objects that are managed by the service (repositories, projects, and models), all endpoints within the service, and the service root.
`/modelRepository/repositories/*`	All repositories, and all second-level endpoints that are beneath the repositories endpoint.
`/modelRepository/projects/*`	All projects, and all second-level endpoints that are beneath the projects endpoint.
`/modelRepository/models/*`	All models, and all second-level endpoints that are beneath the models endpoint.
`/modelRepository/models/`*model-ID*	The specified model.

TIP Not all services have identical structures. Before you create or customize a rule that uses a wildcard in its target URI, make sure you understand the structure of the service, and the resulting scope of the rule. (For services that have published APIs, you can use developer.sas.com to discover structure.)

objectUri

A rule that targets an objectUri affects access to the referenced resource. Here are examples:

- A rule that targets a folder's objectUri affects access to that folder.
- A rule that targets a report's objectUri affects access to that report.
- A rule that targets a service's objectUri (and does not target a specific object instance) affects access to functionality.

In general, rules that specify the objectUri for a content object (such as a folder or report) should include the `/**` suffix. Inadvertently omitting the suffix narrows the effects of a rule and can yield unintended results due to insufficient access.

containerUri

A rule that targets a folder's container URI affects access that the folder conveys to its child members.

Note: You cannot append the `/**` suffix to a containerUri. The `/**` suffix does not reference contained objects (such as reports within a folder). The `/**` suffix has nothing to do with container-based object inheritance.

Media Type Attributes

The three media type attributes (acceptType, acceptItemType, and contentType) provide specialized, advanced support for refining the scope of a rule. Most rules do not specify these attributes. To view or set these attributes, use the REST API.

You can use media type attributes to limit a rule's applicability so that it targets only a particular media type in only a specified context.

- If a rule specifies one media type property, the rule applies only to requests that specify a corresponding value for the corresponding attribute.

  For example, if a rule specifies `contentType = 'application/pdf'` then the rule applies only to requests in which the following query parameter is specified: `(contentType == application/pdf)`.

- If a rule specifies multiple media type attributes, the rule applies only to requests that specify corresponding values for all specified media type attributes.

  For example, if a rule specifies the value `application/pdf` in both the contentType and the acceptType attributes, then the rule applies to only requests for which the following compound set of parameters is specified: `(contentType == application/pdf and acceptType == application/pdf)`.

These attributes are processed as follows:

- In response to an authorization request that specifies one or more of the media type query parameters (contentType, acceptType, or acceptItemType), only matching rules are applied.
- In response to an authorization request that does not specify a media type query parameter (contentType, acceptType, or acceptItemType), any rules that specify one or more media type attributes are ignored.

Rule Attributes

acceptItemType
: a rarely used, target-related attribute. See "Media Type Attributes".

acceptType
: a rarely used, target-related attribute. See "Media Type Attributes".

condition
: an expression that limits the scope or applicability of a rule. See "Rule Conditions".

  Character limit: 5120

containerUri
: a relative URI that represents the container aspect of a container, such as a folder. A rule that targets a containerUri affects access that the container conveys to its child members. See "containerUri".

  Character limit: 500

contentType
: a rarely used, target-related attribute. See "Media Type Attributes".

description
: text that documents a rule for administrative purposes.

  Character limit: 1000

enabled
: the indication of whether a rule is enabled. By default, rules are enabled. To temporarily prevent a rule from being enforced, disable the rule.

mediaType
: a rarely used, target-related attribute. This attribute has no effect and is deprecated. It is replaced by the more specific attributes acceptType, acceptItemType, and contentType.

objectUri
: a relative URI that represents a resource such as a report, a folder, a service, or a service endpoint. See "objectUri".

  Character limit: 500

permissions
: a list of access types. At least one permission is required. See "Permissions".

principal
: the unique string that identifies a particular user or group by its ID. If principalType is `user` or `group`, you must specify a value for this attribute.

  Character limit: 100

principalType
: a type of principal (user or group) or a construct that represents a class of principals (Authenticated Users, Everyone, and Guest).

type
: the indication of whether a rule blocks (prohibit) or attempts to provide (grant) access. Prohibit rules have absolute precedence.

reason
: descriptive or explanatory text that is appropriate for end users.

  Note: In the current release, no client application displays this text to end users. The text is visible to administrators on the **Rules** page in SAS Environment Manager.

  Character limit: 1000

Rule Conditions

Overview

A condition is a Boolean expression that limits the scope of a rule.

- A rule that has no condition is always applied.

- A rule that has a condition that evaluates to `true` for a particular access request is applied in the authorization decision process for that access request.
- A rule that has a condition that evaluates to `false` for a particular access request is ignored in the authorization decision process for that access request.
- If a rule has an invalid condition, an error is logged and access is restricted as follows:
  - If a grant rule has an invalid condition, the rule is always ignored.
  - If a prohibit rule has an invalid condition, the rule is always applied.

You can specify a condition in inclusive or exclusive terms. Here are two examples:

- A rule grants the Read permission to GroupA for folderA, with a condition that the rule applies only on weekdays. A request from a member of GroupA to access folderA on Sunday is outside the condition. For that access request, the condition is false, so the rule is not applicable (it does not provide access).

  Note: A conditional grant rule provides access in specified circumstances, but it does not prevent access outside of those circumstances.
- A rule prohibits the Read permission for GroupA for folderA, with a condition that the rule does not apply on weekends. A request from a member of GroupA to access the folder on Sunday is inside the condition. For that access request, the condition is true, so the rule is applicable (access is blocked).

  Note: A conditional prohibit rule prevents access in specified circumstances, but it does not provide access outside of those circumstances.

Condition Syntax

- Conditions are written and stored in Spring Expression Language (SpEL).
- Boolean operators (AND, OR, and NOT) and parentheses are supported. For example, the following condition always evaluates to `true`:

  ```
  (4 < 6) and (5 > 3)
  ```
- Built-in functions correspond to attributes of the requestor or the environment. You must append `()` to each built-in function (for example, `currentUser()`).

  Note: You can use a constant instead of a function. However, functions are often more useful because they are dynamic. At request time, actual context-specific values are dynamically substituted into each function.
- Variables correspond to attributes of the target. You must prepend `#` to each variable (for example, `#userId`).

  Note: The available condition variables for a particular type of object are designated in the service for that object type. For example, the preference service designates the `userId` attribute on preference objects as available for use as a condition variable.

Built-In Functions

Table 6 *Location-Based Functions*

Function	Description	Type
locale()	Locale of the client that made the request (for example, `en_US`).	String

Function	Description	Type
remoteHost()	Name of the client machine that made the request.	String
remoteIp()	IP address of the client machine that made the request.	String
serverIp()	IP address of the middle-tier server that received the request.	String
serverName()	Machine name of the middle-tier server that received the request.	String
serverPort()	Port of the middle-tier server that received the request.	int

Table 7 *Target-Based Functions*

Function	Description	Type
contentType()	Content type of the target (for example, `application/vnd.sas.credential.domain+json`).	String
contentLength()	Length of the request.	long
uri()	URI of the target.	String

Table 8 *Time-Based Functions*

Function	Description	Type
timestamp	Coordinated Universal Time (UTC) timestamp.	ZonedDateTime
timestamp(zoneId)	Timestamp of the request, based on a specified zoneid.	ZonedDateTime
localTime(zoneId)	Time of the request, based on a specified zoneid.	LocalTime
localDate(zoneId)	Date of the request, based on a specified zoneid.	LocalDate
localDateTime(zoneId)	Date and time of the request, based on a specified zoneid.	LocalDateTime

Note: In the preceding table, *zoneid* refers to a time zone ID that is valid for java.time.ZoneId.

Table 9 *Other Functions*

Function	Description	Type
currentUser()	Identifier for the currently connected user.	String
groupsForCurrentUser()	Identifier for each group to which the current user belongs. Nested memberships are included. Unassumed memberships are not included.	List
method()	Method that the request invoked (for example, `GET`).	String

Function	Description	Type
protocol()	Protocol of the request (for example, `HTTP/1.1`).	String
header(headerName)	Headers for a specified headerName.	List

Examples of Conditions

This condition makes its associated rule applicable only for weekday requests (in the US Eastern time zone):

```
localDate('US/Eastern').dayOfWeek != T(java.time.DayOfWeek).SUNDAY and
localDate('US/Eastern').dayOfWeek != T(java.time.DayOfWeek).SATURDAY
```

This condition makes its associated rule applicable only if the target's user ID is the same as the requesting user's ID:

```
#userId == currentUser()
```

Evaluation of Conditions

Here are key points about evaluation of conditions:

- If a request does not meet the criteria in a condition, the request is outside that condition. If a request meets the criteria in a condition, the request is inside that condition.
- In a description of effective access, there is no request context, so conditions are not evaluated. Even an atypical condition that is always true (1=1) or never true (1>2) yields an effective access result of Conditional in certain scenarios. A condition is evaluated only in the context of a specific request.
- In an actual request, there is a request context. Any relevant conditions are evaluated, and a definitive answer is provided (Authorized or Not Authorized).

Rule Precedence

The *only* factor that affects precedence is the type of rule (grant or prohibit). Prohibit rules have *absolute* precedence. If there is a relevant prohibit rule, effective access is always Not Authorized.

Neither object inheritance nor identity hierarchy has precedence implications. Here are examples:

- A grant setting that is assigned to you has *less* precedence than a prohibit setting that is assigned to Authenticated Users.
- A direct grant on a report has *less* precedence than a prohibit setting that the report inherits from its parent folder.

In the following table, each row indicates the effective access result for a separate, independent scenario. For example, if the only relevant rule is a Conditional Prohibit, the result is Not Authorized (because there is no relevant grant setting).

Rules	Effective Access Result	Explanation
(none)	Not Authorized	Any access that is not granted is implicitly denied. (This result is due to the lack of a grant, so you can override it by adding a grant.)
Prohibit	Not Authorized	A relevant prohibit setting blocks access. (This result is due to a prohibit setting, so you cannot override it by adding grants. As long as the prohibit setting exists and is relevant, effective access is not authorized.)
Prohibit + (any other rules)	Not Authorized	A relevant prohibit setting has absolute precedence.
Conditional Prohibits	Not Authorized	No relevant grants, no access.
Grant	Authorized	A relevant grant provides access, if there are no relevant prohibit settings.
Grant + Conditional Grants	Authorized	Relevant grants provide cumulative access, if there are no relevant prohibit settings.
Grant + Conditional Prohibits	Conditional	Authorized for requests that are outside all of the prohibit conditions. Prohibit wins, but only within its scope.
Conditional Grants	Conditional	Authorized for requests that are inside any of the grant conditions.
Conditional Grants + Conditional Prohibits	Conditional	Authorized for requests that are outside all of the prohibit conditions *and* inside at least one grant condition.

Authorization Explanations

An explanation of effective access answers the question, Why does this principal have this effective access result for this permission and object?

Each explanation consists of these items:

- A list of contributing rules, which includes all relevant rules, except share-based rules. Rules that are relevant but not determinative are included.
- A list of contributing shares, which includes all relevant shares.

Access to Authorization Rules

Anyone who has the Secure permission for a resource can add, modify, and delete direct rules for that resource.

Resharing enables users to create share-based authorization rules, even if they do not have the Secure permission for the target resource. See the following topic.

Note: In the initial configuration, the SAS Administrators group can add, modify, and delete all authorization rules. A universal predefined rule grants all permissions to the SAS Administrators group. However, the SAS Administrators group is not unrestricted or exempt from authorization requirements.

Sharing: Details for Administrators

Introduction

This topic documents administrative aspects of the implementation of sharing that is provided by the authorization service. That implementation is available to end users in SAS Drive.

Note: For information about sharing of Model Studio projects, sharing of reports in SAS Visual Analytics, and sharing of generic (formerly SAS Data Management) projects, see your product documentation.

Who Can Share?

In the initial configuration, sharing is available as follows:

- Any user who has Secure access to an object can share that object with other users and groups.
- Any user with whom an object has been shared can further share that object, passing along some or all of the access that they received. Only the initiator of a chain of sharing has to have the Secure permission on the shared object.

  For example, after UserA shares reportA with UserB, UserB can share reportA with other users, even if there is no authorization rule that grants UserB the Secure permission for reportA. (However, if an authorization rule explicitly prohibits the Secure permission for UserB on reportA, UserB cannot further share reportA.)

For a more restrictive configuration, see “How to Prevent All Sharing” or “How to Prevent Re-sharing”.

What is a Share?

A share is of a set of attributes that is backed by a corresponding authorization rule.

When users share objects in SAS Drive, shares and corresponding share-based rules are generated. Each share specifies a particular type of access, a recipient user or group, and a target object. Each corresponding share-based rule translates the share information into an authorization rule. In authorization decisions, share-based rules are evaluated in the same way as other authorization rules.

Here are details about shares:

- Shares can only expand access. Users can remove shares, but they cannot use shares to prevent access.
- Shares do not provide guaranteed or comprehensive access.
  - A share does not provide access if there is a relevant Prohibit rule.
  - A share of one object does not provide access to related resources (such as data or linked reports), parent objects (such as folders), or embedded objects (such as images or files).

TIP Because sharing an object does not provide access to the object's parent folders, a share recipient might not be able to navigate to the shared object.

- Shares cannot provide variations of access that differ from the defined share types.
- Shares cannot be conditional.
- Shares cannot target a media type.
- Shares for a container object always target both the object's URI and the object's container URI.
- Shares are static. For example, a share of reportA by UserA to UserB is unaffected by any subsequent loss of UserA's access to reportA.

Effects of Sharing

Sharing a non-container object can expand access to that object. The following table provides details:

Table 10 *Effects of Sharing a Non-Container Object*

Share Type	Effects
read	Corresponds to a grant of the Read permission.
readShare	Corresponds to a grant of the Read permission. Also provides the ability to further share Read access.
readEdit	Corresponds to grants of the Read, Update, and Delete permissions.
readEditShare	Corresponds to grants of the Read, Update, and Delete permissions. Also provides the ability to further share some or all of the same access.

Sharing a container object can expand access to that object and its child members. The following table provides details:

Table 11 *Effects of Sharing a Container Object*

Share Type	Effects
read	Corresponds to a grant of the Read permission on the container's objectUri and containerUri.
readShare	Corresponds to a grant of the Read permission on the container's objectUri and containerUri. Also provides the ability to further share the same access to the container.
readEdit	Corresponds to grants of all permissions except Secure on the container's object URI and container URI.
readEditShare	Corresponds to grants of all permissions except Secure on the container's object URI and container URI. Also provides the ability to further share some or all of the same access to the container.

Note: The ability to re-share is not conveyed from a container object to its members. For example, the recipient of a readShare on a folder can re-share that folder, but cannot re-share individual items within that folder.

Attributes of a Share

enabled
: specifies whether a share is enabled. By default, shares are enabled.

resourceUri
: specifies the URI of the content object that is shared. Also referred to as the share *target*.

sharedBy
: specifies the ID of the user that created the share.

sharedWith
: specifies the ID of the user or group that receives the share. Also referred to as the share *recipient*.

sharedWithType
: specifies the type of identity that receives the share.

type
: specifies the type of access that the share provides. Also referred to as the share *level*.

  - Users select either `Can read` or `Can read and edit` when they create a share.
  - The service stores one of the following values for each share: `read`, `readEdit`, `readShare`, `readEditShare`.

    Note: The appended `Share` indicates that a share allows further sharing.

How to Prevent Re-sharing

Initially, downstream sharing is supported, even for users who lack Secure access to the object that is being shared. Disabling re-sharing has the following effects:

- New shares have values of `read` or `readEdit`, not `readShare` or `readEditShare`.
- Existing shares retain their values of `readShare` or `readEditShare`. However, the `Share` part of those values has no effect. Only those users who have Secure access to a target object can share that object.
- Any access that was already extended through re-sharing remains in effect. Disabling re-sharing does not revoke access that already exists.

To limit downstream sharing to only those users who have Secure access to the target object, set the authorization service's reshareEnabled property to `Off`.

Note: The change takes effect within 30 seconds. You do not have to restart the authorization service.

How to Prevent All Sharing

Initially, sharing is enabled. Disabling sharing has the following effects:

- No new shares can be created.
- All existing share-based rules are disabled.

To disable sharing, set the authorization service's sharingEnabled property to `Off`.

Note: The change takes effect within 30 seconds. You do not have to restart the authorization service.

Configuration Properties: Authorization Service

sas.authorization.maxAncestryCacheSize
: specifies the maximum number of ancestors to cache per object. The default value is `1000`. The cache enhances performance in container-based inheritance.

sas.authorization.remote (a supplemental property)
: disables enforcement in the general authorization system, if set to `false`. The default value is `true`. An administrator might temporarily disable authorization if rules that inadvertently prevent access are introduced. Do not disable authorization while the system is available to other users.

sas.authorization.reshareEnabled
: specifies whether resharing is enabled. When set to `Off`, prevents downstream sharing by users who lack Secure access to the object that is being shared. The default value is `On`.

  Note: If sharingEnabled is `Off`, setting this property has no effect.

  Note: You cannot enable or disable downstream sharing on a per-object or per-identity basis. The ability to perform downstream sharing is controlled only by this deployment-wide configuration property.

sas.authorization.rules.executorThreads (a supplemental property)
: specifies the number of threads that are available for bulk processing of authorization rules. The default value is `20`. Modify this value only if you are directed to do so by SAS Technical Support.

sas.authorization.sharingEnabled
: specifies whether sharing is enabled. When set to `Off`, prevents all sharing. The default value is `On`.

sas.authorization.warnOnCycles (a supplemental property)
: prevents cyclic warnings from being written to the authorization service log, if set to `false`. The default value is `true`. An administrator might temporarily set this property to `false` if cyclic warnings are causing the log to grow rapidly. After cyclic rules are corrected, set this property back to `true`.

General Authorization: Guidelines

These general guidelines contribute to simplicity and security.

- Minimize use of prohibit rules.
- Limit membership in administrative groups.
- Use groups, not individual users, as principals.
- Use folders, not individual objects, as targets.
- Use conditions only if you cannot efficiently express your authorization requirements another way.
- Perform a backup before and after you make significant changes to your system.

General Authorization: Troubleshooting

Unexpected Outcomes

Here are tips for troubleshooting an authorization outcome that differs from what you expect:

- Make sure all relevant rules are enabled. On the Rules page, right-click a rule to view all of its properties. Or, add the **Rule Status** column to the display.
- Make sure you understand the precedence model. See Authorization Decisions.
- Examine the origins information for the unexpected outcome. See Identify the Source of Effective Access.
- If the unexpected outcome relates to inheritance from a folder to objects in that folder, make sure you are using the second set of permissions in the folder's Authorization window to convey access to the folder's child members. See Inheritance.
- If the unexpected outcome relates to your access, and you have changed your memberships in the current session, sign out and then sign back in.
- If the unexpected outcome relates to your access, and you are a member of the SAS Administrators group, sign out and then sign back in, indicating whether you want that membership to be in effect.

Unavailable Principals

To grant access to a principal that is not in the identities service, use the **Rules** page in SAS Environment Manager or use the command-line interface.

Unrecognized Principals

If the **Rules** page or the Authorization window displays a warning icon next to a principal's name, that principal does not exist in the identities service.

- If the principal is a service account (for example, sas.folders or sasapp), you can ignore the warning icon.
- If you are using the New Rule or Edit Rule window, make sure that the correct value is selected in the **Principal type** field and the principal's unique identifier (not display name) is specified.
- If you are using the Authorization window, make sure the identity still exists.

  Note: Deletion of a custom group does not cause automatic deletion of rules in which that custom group is the principal.

Unintended Loss of Access

Reinstate Access: Instructions for Users

If you inadvertently block your own access to a resource, contact an administrator for assistance.

Note: Anyone who still has Secure access to the blocked resource can reinstate your access.

Reinstate Access: Instructions for Administrators

To reinstate access that is blocked by a prohibit rule, complete the following steps:

1. Opt in to your assumable membership in the SAS Administrators group.
2. Try to reinstate access by disabling, modifying, or deleting the prohibit rule. Here are some tips:
   - If the resource is a content object (such as a folder or report) and you cannot see the resource on the **Content** page, you lack Read access to the resource. Use the Rules page.
   - If the resource is a content object and you cannot make changes in the resource's Authorization window, you lack Secure access to the resource. Either delete the resource (if you have Delete access and the resource is not already in use) or proceed to the next step.
   - If the resource is not a content object, use the **Rules** page.
   - If you know who created (or last modified) a problematic rule or when a problematic rule was created (or last modified), use the **Modified By** or **Date Modified** filter on the **Rules** page to locate the problematic rule.

   If you cannot reinstate access, proceed to the next step.
3. To temporarily prevent users other than yourself from using the deployment, close current sessions for users other than yourself, and disallow new sessions.

   Note: A system administrator can disable logins through firewall rules or using LDAP. This disables new sessions, ends current sessions, and prevents others from using the deployment.
4. Temporarily disable self-enforcement of authorization requirements for the authorization service.
   a. In the configuration definition for the authorization service, add a supplemental property named `remote` with a value of `false`. See *SAS Viya Administration: Configuration Properties*.
   b. Restart the authorization service. See *SAS Viya Administration: General Servers and Services*.
5. Disable, modify, or delete the problematic rule or rules.
6. Enable self-enforcement of authorization requirements for the authorization service.
   a. In the configuration definition for the authorization service, remove the supplemental property named `remote`.
   b. Restart the authorization service.
7. Verify that access is reinstated.
8. Make the deployment available again by allowing new user sessions.

If you cannot reinstate access, contact SAS Technical Support for assistance.

A Deleted Rule Reappears

Some of the predefined rules are bootstrapped by their associated service. If you delete one of those rules, it reappears the next time the service starts. Modifications that you make to such rules are preserved. If you are sure you do not want one of those rules to be in effect, disable that rule (instead of deleting it).

Inability to Browse to an Object

If you have access to a content object but cannot navigate to that object, you lack Read access to at least one of the object's parent folders.

To access the object, use an alternate navigation method. Here are examples:

- Access the object from a direct link or reference.
- Search for the object, and open the object from the search results list.
- If the object was shared with you, access the object from the **Shared with me** pane in SAS Drive. (On the **All** tab, select **Shared**. In the center pane, make sure **Shared with me** is selected.)

Not All Contributing Rules Are Visible

If the number that is displayed on the **Contributing Rules** tab exceeds the number of visible listed rules, you are not currently authorized to see all contributing rules.

For the most comprehensive view, assume membership in the SAS Administrators group when you sign in.

See "Identify the Source of Effective Access".

A Pair of Predefined Rules Is Difficult to Interpret

The search service uses two predefined rules that might seem to block Read access to all resources for most authenticated users. No administrative action or adjustment is needed, because the rules do not actually have that effect.

Each of the two rules includes a specialized property that is not displayed in SAS Environment Manager. Those specialized properties, acceptType and acceptItemType, limit the applicability of the rules. The rules affect only requests that involve usage of the search service's index. The rules ensure that users are not offered search results that include resources that they are unable to access.

Here are the key properties of one of the rules:

Property	Value
objectUri	/**

Property	Value
principalType	authenticatedUsers
type	prohibit
permissions	read, create
condition	!(groupsForCurrentUser().contains('SASAdministrators') || clientId() == 'sas.searchIndex')
description	Prevent ordinary users from getting a collection of indexable data when requesting a collection of vnd.sas.search.indexable.data.
acceptItemType*	application/vnd.sas.search.indexable.data+json

* This property limits the applicability of the predefined rule. See "Media Type Attributes".

General Authorization: Interfaces

In the following table, the shaded part of each circle is an approximation of the amount of general authorization functionality that a particular interface exposes. The shading indicates relative coverage. The shading does not indicate alignment of coverage across interfaces.

Table 12 *Interfaces to General Authorization*

Interface		Description
●	REST API	The REST interface for general authorization.
◕	Rules page	The advanced enterprise graphical interface for managing rules directly.
◒	Authorization window	The basic enterprise graphical interface for managing access to content such as folders and reports.
◒	Command-line interface	A simple, scriptable interface for managing access to objects and resources.
◔	Share Window	An interface for simple sharing of content objects.

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies. Copyright © 2019, SAS Institute Inc., Cary, NC, USA. All Rights Reserved. November 2019 3.5-P1:calauthzgen

sas THE POWER TO KNOW.