100 XP

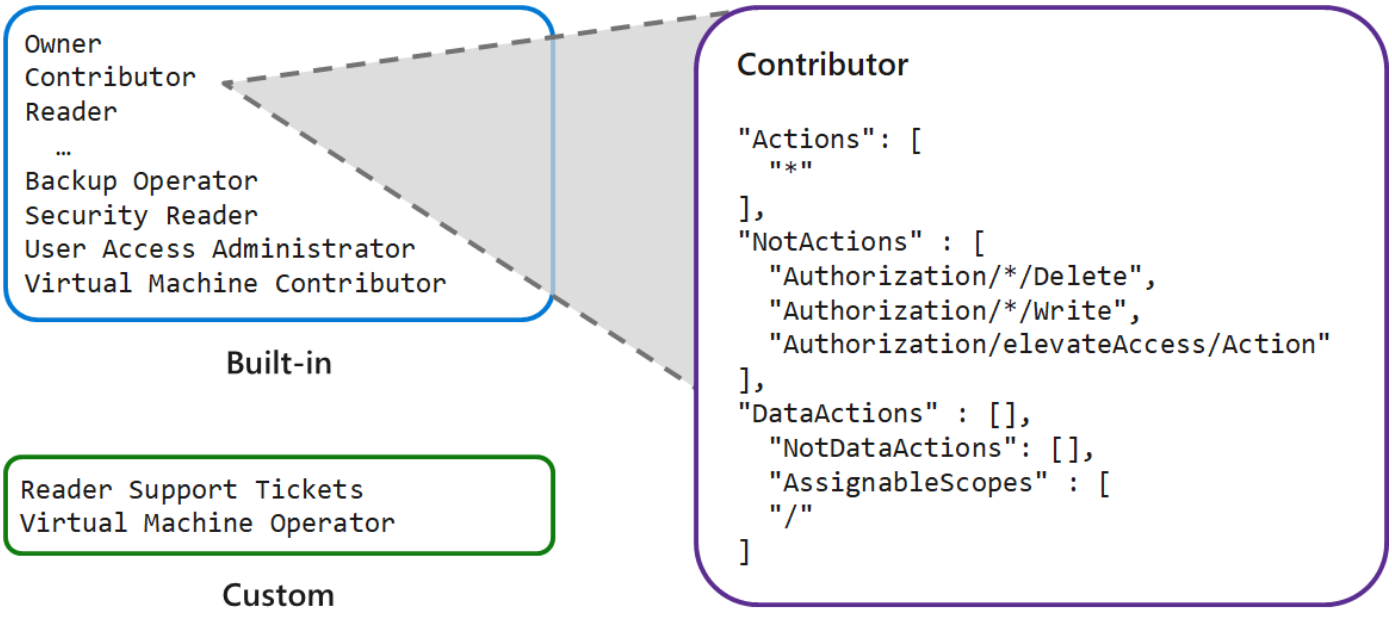
Create a role definition

4 minutes

A role definition consists of sets of permissions that are defined in a JSON file. Each permission set has a name, such as *Actions* or *NotActions* that describes the purpose of the permissions. Some examples of permission sets include:

- *Actions* permissions identify what actions are allowed.
- *NotActions* permissions specify what actions aren't allowed.
- *DataActions* permissions indicate how data can be changed or used.
- *AssignableScopes* permissions list the scopes where a role definition can be assigned.

The following diagram shows details for the *Contributor* role in Azure RBAC, which has three sets of permissions.



The *Actions* permissions show the *Contributor* role has all action privileges. The asterisk `"*"` wildcard means "all." The *NotActions* permissions narrow the privileges provided by the *Actions* set, and deny three actions:

- `Authorization/*/Delete`: Not authorized to delete or remove for "all."
- `Authorization/*/Write`: Not authorized to write or change for "all."

- `Authorization/elevateAccess/Action`: Not authorized to increase the level or scope of access privileges.

The *Contributor* role also has two *DataActions* permissions to specify how data can be affected:

- `"NotDataActions": []`: No specific actions are listed. Therefore, all actions can affect the data.
- `"AssignableScopes": ["/"]`: The role can be assigned for all scopes that affect data.

Here's another example of a role definition in PowerShell:

PowerShell

```
Name: Owner
ID: 01010101-2323-4545-6767-987453021523
IsCustom: False
Description: Manage everything, including access to resources
Actions: {*}             # All actions allowed
NotActions: {}           # No actions denied
AssignableScopes: {/}    # Role can be assigned to all scopes
```

Things to know about role definitions

Review the following characteristics of role definitions:

- Azure RBAC provides built-in roles and permissions sets. You can also create custom roles and permissions.
- The *Owner* built-in role has the highest level of access privilege in Azure.
- The system subtracts *NotActions* permissions from *Actions* permissions to determine the *effective permissions* for a role.
- The *AssignableScopes* permissions for a role can be management groups, subscriptions, resource groups, or resources.

Role permissions

Use the *Actions* and *NotActions* permissions together to grant and deny the exact privileges for each role. The *Actions* permissions can provide the breadth of access and the *NotActions* permissions can narrow the access.

The following table shows how the *Actions* or *NotActions* permissions are used in the definitions for three built-in roles: *Owner*, *Contributor*, and *Reader*. The permissions are narrowed from the *Owner* role

to the *Contributor* and *Reader* roles to limit access.

Role name	Description	Actions permissions	NotActions permissions
Owner	Allow all actions	*	n/a

Contributor	Allow all actions, except write or delete role assignment	*	`- Microsoft.Authorization/*/Delete`
			`- Microsoft.Authorization/*/Write`
			`- Microsoft.Authorization/elevateAccess/Action`

Reader	Allow all read actions	`/*/read`	n/a

Role scopes

After you define the role permissions, you use the *AssignableScopes* permissions to specify how the role can be assigned. Let's look at a few examples.

- Scope a role as available for assignment in two subscriptions:

  ```
  "/subscriptions/c276fc76-9cd4-44c9-99a7-4fd71546436e", "/subscriptions/e91d47c4-76f3-4271-a796-21b4ecfe3624"
  ```

- Scope a role as available for assignment only in the Network resource group:

  ```
  "/subscriptions/c276fc76-9cd4-44c9-99a7-4fd71546436e/resourceGroups/Network"
  ```

- Scope a role as available for assignment for all requestors:

  ```
  "/"
  ```

Things to consider when creating roles

Consider the following points about creating role definitions in Azure RBAC:

- **Consider using built-in roles**. Review the list of built-in role definitions in Azure RBAC. There are over 100 pre-defined role definitions to choose from, such as *Owner*, *Backup Operator*, *Website*

Contributor, and *SQL Security Manager*. Built-in roles are defined for several categories of services, tasks, and users, including General, Networking, Storage, Databases, and more.

- **Consider creating custom definitions**. Define your own custom roles to meet specific business scenarios for your organization. You can modify the permissions for a built-in role to meet the specific requirements for your organization. You can also create custom role definitions from scratch.
- **Consider limiting access scope**. Assign your roles with the minimum level of scope required to perform the job duties. Some users like administrators require full access to corporate resources to maintain the infrastructure. Other users in the organization can require write access to personal or team resource, and read-only access to shared company resources.
- **Consider controlling changes to data**. Identify data or resources that should only be modified in specific scenarios and apply tight access control. Limit users to the least of amount of access they need to get their work done. A well-planned access management strategy helps to maintain your infrastructure and prevent security issues.
- **Consider applying deny assignments**. Determine if you need to implement the deny assignment feature. Similar to a role assignment, a deny assignment attaches a set of deny actions to a user, group, or service principal at a particular scope for the purpose of denying access. Deny assignments block users from performing specific Azure resource actions even if a role assignment grants them access.

Next unit: Create a role assignment

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆