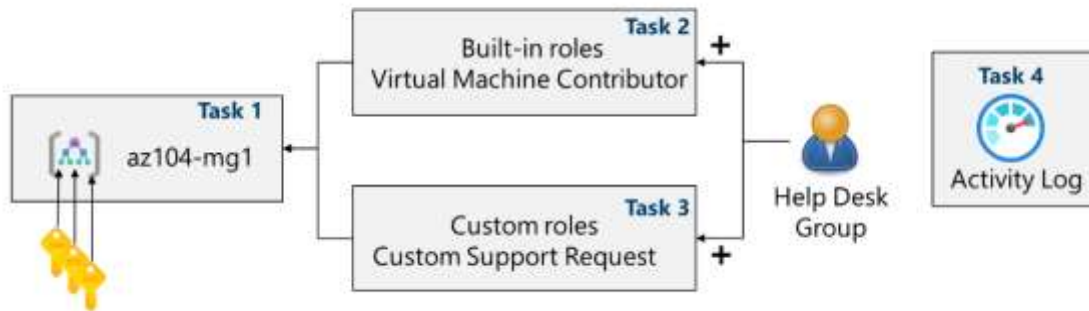


### Architecture diagram



Perfect drop, Muhammad. Here's how we document **Task 1: Implement Management Groups** — clean, tactical, and written as a record of what *we did*, not what *should be done*. You can paste this directly into your Word doc under Lab 01a.

---

#### ◆ Task 1: Implement Management Groups

We launched the Azure portal using the Cloud Express Pass credentials provided in the lab. After bypassing the “Action Required” prompts and skipping the “Stay signed in?” and “Welcome” screens, we accessed **Microsoft Entra ID** to review the **Access management for Azure resources** setting under **Properties**. This confirmed that directory-level access control was enabled.

We then searched for **Management Groups** and created a new group using the following configuration:

- **Management Group ID:** az104-mg101
- **Display Name:** az104-mg1

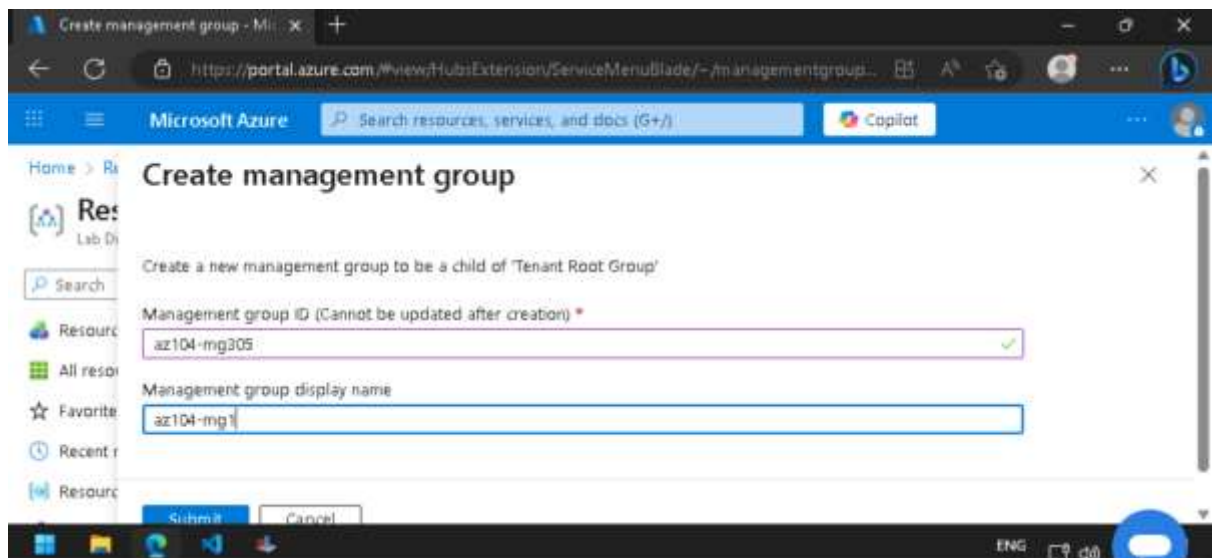
After submitting, we refreshed the blade until the new group appeared. The hierarchy displayed the **root management group**, confirming that all groups and subscriptions roll up to it — allowing for centralized policy and RBAC inheritance.

This setup lays the foundation for organizing subscriptions logically and applying governance controls at scale. In our scenario, this enables Help Desk staff to create support requests across all subscriptions without needing individual access to each one.

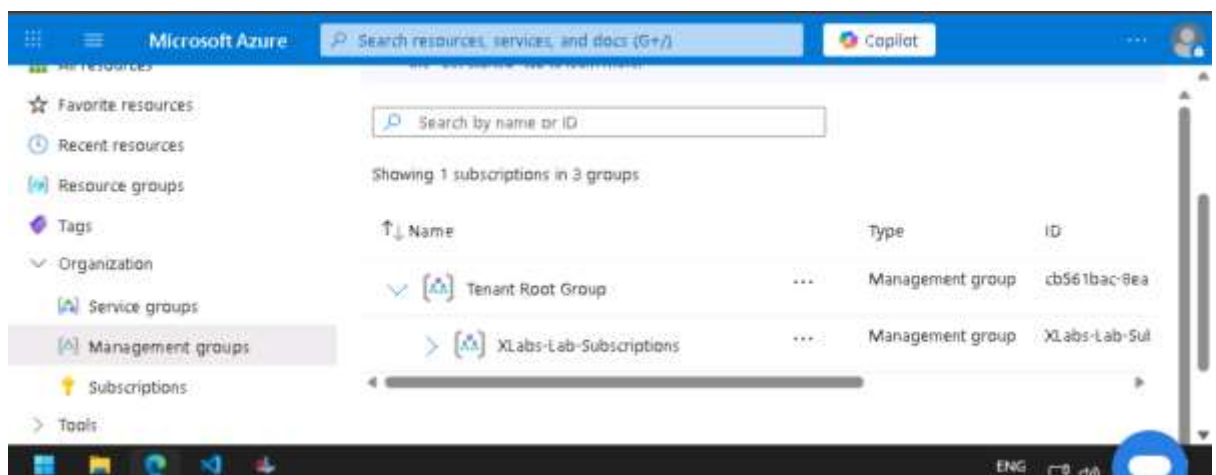
---

#### 📸 Suggested Screenshots

- **Management Group creation blade** with ID and display name filled in



- **Hierarchy view** showing the new group under the root



## ◆ Task 2: Review and Assign a Built-in Azure Role

We accessed the **az104-mg1** management group and opened the **Access control (IAM)** blade. Under the **Roles** tab, we reviewed the available built-in roles, including Owner, Contributor, Reader, and others. Each role displayed its permissions, JSON definition, and current assignments.

We selected **+ Add → Add role assignment**, searched for **Virtual Machine Contributor**, and confirmed its scope: the role allows management of virtual machines without access to their OS, networking, or storage — ideal for Help Desk operations.

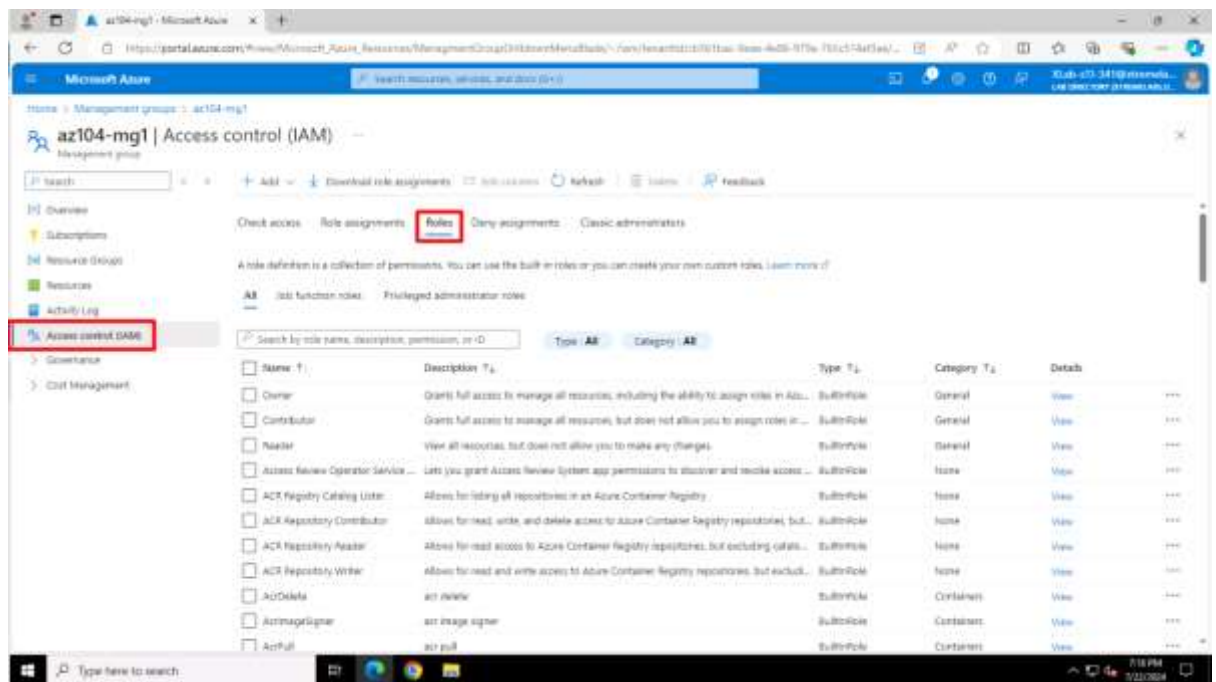
On the **Members** tab, we selected the **helpdesk group** (created earlier if not already present), then clicked **Review + assign** twice to finalize the role assignment.

Back on the **Role assignments** tab, we verified that the helpdesk group now holds the **Virtual Machine Contributor** role at the management group level.

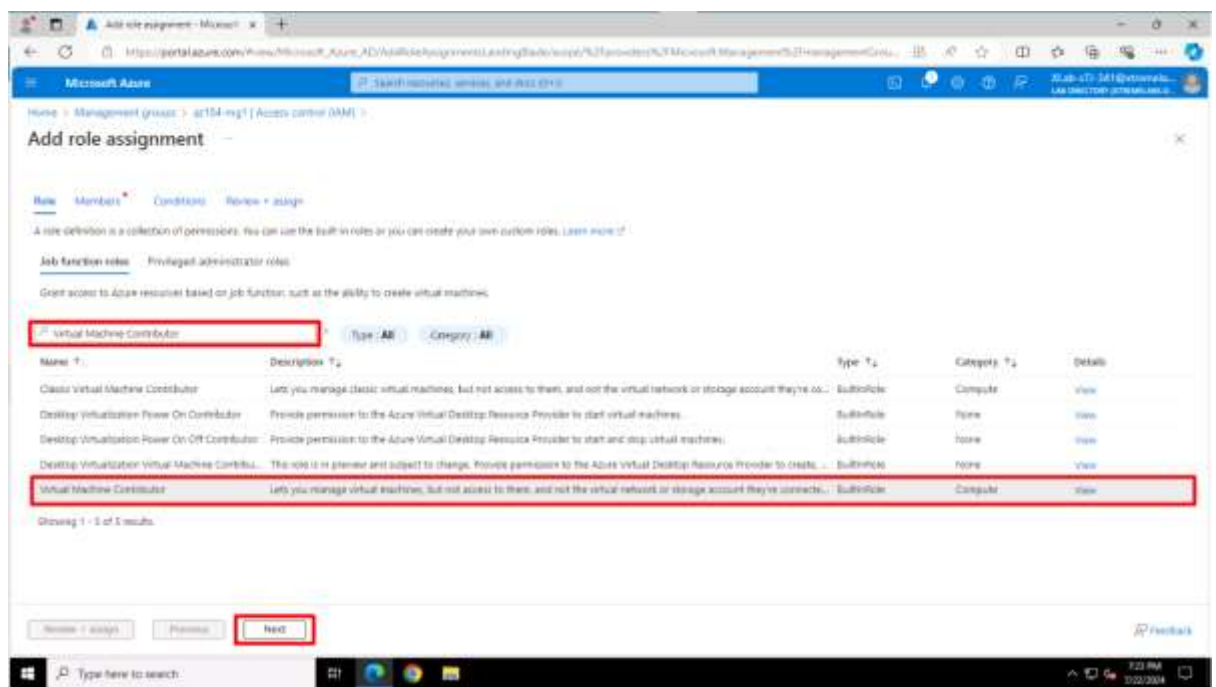
This setup promotes **least privilege access**, ensuring Help Desk staff can manage VMs without overreaching into sensitive infrastructure.

📷 Suggested Screenshots

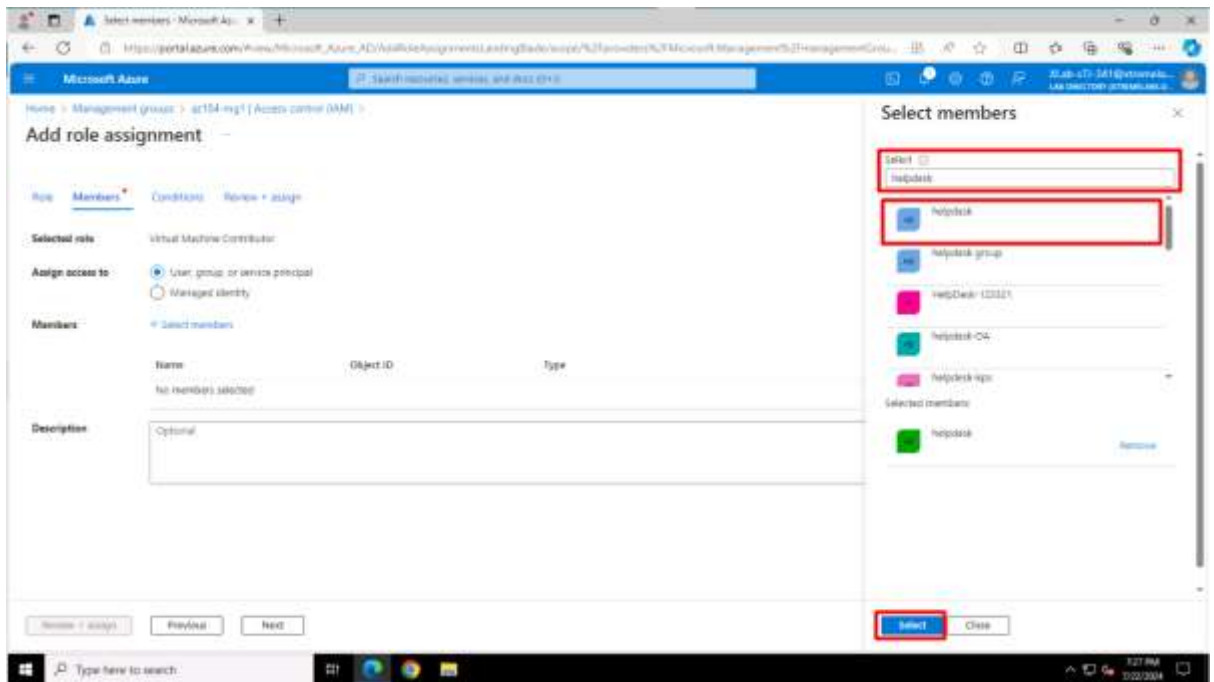
- **Roles tab** showing built-in roles list



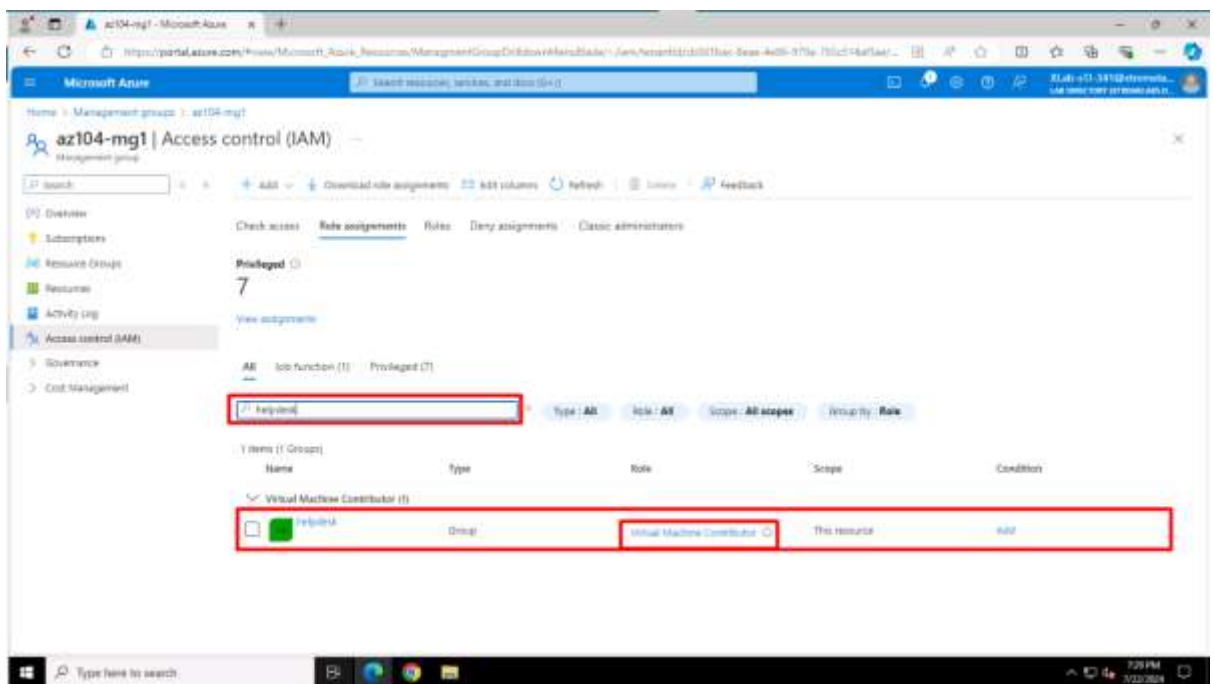
- **Role assignment blade** with Virtual Machine Contributor selected



- **Members tab** showing helpdesk group selected



- **Role assignments tab** confirming the assignment



### ◆ Task 3: Create a Custom RBAC Role

We continued working within the **az104-mg1** management group and navigated to the **Access control (IAM)** blade, selecting the **Check access** tab. From there, we initiated the creation of a custom role by selecting **Add** under the “Create a custom role” section.

On the **Basics** tab, we configured the role with the following settings:

- **Custom Role Name:** Custom Support Request 101
- **Description:** A custom contributor role for support requests.

We chose to **clone the Support Request Contributor** role as the baseline, then moved to the **Permissions** tab to exclude unnecessary access. Using the resource provider search, we filtered for **.Support** and selected **Microsoft.Support**. We excluded the permission **“Other: Registers Support Resource Provider”**, adding it as a **NotAction** to ensure Help Desk users couldn’t register support resource providers.


On the **Assignable scope** tab, we confirmed that the **az104-mg1** management group was listed, then reviewed the JSON structure showing the customized **Actions**, **NotActions**, and **AssignableScopes**.

After confirming the configuration, we selected **Review + Create**, then **Create**, successfully generating a custom RBAC role tailored to Help Desk support needs.

This role enforces **least privilege** by removing unnecessary permissions while still enabling support request functionality.

#### Suggested Screenshots

- **Custom role creation blade** with name and description



Basics Permissions Assignable scopes JSON Review + create

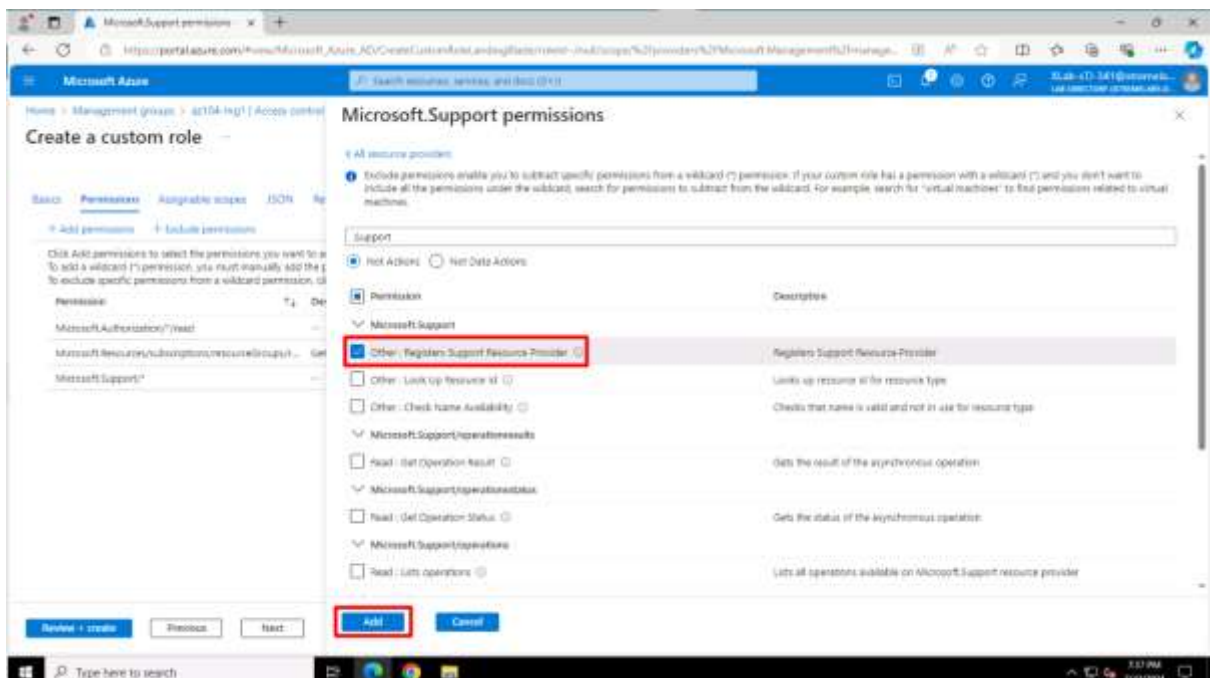
To create a custom role for Azure resources, fill out some basic information. [Learn more](#)

Custom role name \* Custom Support Request 305

Description A custom contributor role for support requests.

Baseline permissions ☐ Clone a role ☒ Start from scratch ☐ Start from JSON

- **Permissions tab** showing excluded permission



Microsoft Azure Search resources, services, and docs (0/1)

Home > Management groups > az104-mg1 > Access control > Microsoft.Support permissions

Create a custom role

Basics Permissions Assignable scopes JSON Review + create

Click Add permissions to select the permissions you want to add. To add a wildcard (\*) permission, you must manually add the permission to the list. To exclude specific permissions from a wildcard permission, click the Not Actions button.

Permissions

Microsoft.Authorization/\* (wildcard)

Microsoft.Resources/subscriptions/resourceproviders/\* (wildcard)

Microsoft.Support/\*

Not Actions Not Data Actions

Permissions

Microsoft.Support

Other: Registers Support Resource Provider

Other: Lock up resource id

Other: Check name Availability

Microsoft.Support/operationresults

Read: Get Operation Result

Microsoft.Support/operationstatus

Read: Get Operation Status

Microsoft.Support/operations

Read: Get operations

Review + create Previous Next Add Cancel

- **Assignable scope tab** with management group selected



#### ◆ Task 4: Monitor Role Assignments with the Activity Log

We navigated to the **az104-mg1** management group and opened the **Activity Log** to audit recent access control changes. The log provided visibility into subscription-level events, including role assignments and custom role creation.

We filtered the log to focus on **“Role Assignment”** operations and reviewed entries related to the creation and assignment of the **Custom Support Request 101** role. Each event included timestamped details of who performed the action, what role was assigned, and the scope of the assignment.

This step validated that our RBAC changes were properly recorded and aligned with governance policies. It also reinforced the importance of using the Activity Log for **security compliance and access auditing**.

#### 📸 Suggested Screenshots

##### • Activity Log filtered by Role Assignment

