THOMAS MORE

Lab 1.1

Discovering Overprivileged Access in Azure

Toepasbare Cloud Security voor IT-Professionals

2025 - 2026

Contents

1. Discovering Overprivileged Access in Azure

1.1. Context

You are joining the security review of a small cloud application.

The application is already deployed in Azure. It consists of several resources, such as an App Service, Storage Account, Key Vault, logging resources, and supporting infrastructure.

The application works, but the team is not sure whether the access model is secure.

Your task is to investigate the environment and identify whether any identity has more permissions than it actually needs.

This lab focuses on **identity**, **roles**, **scopes**, and **least privilege**. You will not change or exploit anything yet. That comes later.

1.2. Learning goals

By the end of this lab, you should be able to explain:

- Which identities exist in the lab environment.
- Which identity belongs to the backend application.
- Which roles are assigned.
- At which scope those roles are assigned.
- Which role assignment is too broad.
- Why excessive access is dangerous.
- What a safer permission model could look like.

1.3. Estimated time

45 minutes

Suggested timing:

Time	Activity
5 minutes	Scenario briefing
10 minutes	Resource discovery
15 minutes	Identity and role investigation
10 minutes	Risk analysis worksheet
5 minutes	Group discussion

1.4. Scenario

The development team deployed a cloud application quickly so it could be tested.

During deployment, some permissions may have been assigned too broadly.

You are asked to investigate the following question:

“Does any identity in this resource group have more access than it needs?”

You are not asked to fix the issue yet. You are not asked to attack the application yet. The goal of this lab is to **find and explain the risk**.

1.5. Before you start

Make sure you have:

- [ ] Azure CLI available
- [ ] Access to the correct Azure tenant
- [ ] Access to the correct Azure subscription
- [ ] The lab resource group name
- [ ] The backend App Service name

The course setup already expects you to verify Azure CLI, Terraform, the correct subscription, and the deployed lab resources before starting the IAM labs.

2. Confirm your Azure context

Before inspecting permissions, first confirm that you are working in the correct Azure environment.

Run:

```
az account show --output table
```

Check the output.

Write down:

Item	Value
Subscription name	
Subscription ID	
Tenant ID	
Signed-in user	

2.1. Question 1

Why is it important to check the active subscription before running Azure CLI commands?

Your answer:

3. Discover the resources in your lab environment

List all resources in your lab resource group.

Replace <RESOURCE_GROUP_NAME> with the name of your assigned lab resource group.

```
az resource list `
  --resource-group <RESOURCE_GROUP_NAME> `
  --output table
```

> **Command Examples**
> The examples in this lab use PowerShell line continuation with a backtick (`). If you use Bash, replace the backtick with a backslash (\).

You should see several Azure resources.

Typical resource types may include:

```
Microsoft.Web/sites
Microsoft.Web/serverfarms
Microsoft.Storage/storageAccounts
Microsoft.KeyVault/vaults
Microsoft.OperationalInsights/workspaces
Microsoft.Insights/components
```

This matches the course architecture where the lab environment uses a small set of Azure services such as App Service, Storage Account, Key Vault, and Log Analytics.

3.1. Fill in the table

Resource name	Resource type	What do you think it is used for?

3.2. Question 2

Which resource looks like the backend application?

Your answer:

4. Find the backend managed identity

The backend application may have a **managed identity**.

A managed identity is an Azure identity assigned to a resource, such as an App Service. The application can use this identity to access other Azure resources without storing credentials in code.

Run:

```
az webapp identity show `
  --resource-group <RESOURCE_GROUP_NAME> `
  --name <BACKEND_APP_NAME>
```

Look for these fields:

```
principalId
tenantId
type
```

Write down the result:

Item	Value
Backend App Service name	
Managed identity type	
Principal ID	
Tenant ID	

4.1. Question 3

Is this a human identity or a workload identity?

Your answer:

4.2. Question 4

Why might a backend application need a managed identity?

Your answer:

5. List role assignments in the resource group

> **Note**
> In this lab you only inspect the environment. Do not remove, add, or change role assignments yet. The fix will be done in a later lab.

Now inspect the role assignments on the resource group.

Run:

```
az role assignment list `
  --resource-group <RESOURCE_GROUP_NAME> `
  --output table
```

This shows who has access to the resource group and which roles they have.

You may see users, groups, service principals, or managed identities.

5.1. Fill in the table

Principal name / ID	Principal type	Role	Scope

5.2. Question 5

Which identities have access to the resource group?

Your answer:

6. Inspect the backend identity specifically

Now inspect the role assignments for the backend managed identity.

Use the `principalId` you found earlier.

```
az role assignment list `
  --assignee <BACKEND_PRINCIPAL_ID> `
  --all `
  --output table
```

Write down all roles assigned to this identity.

Identity	Role	Scope
Backend managed identity		
Backend managed identity		

6.1. Expected investigation point

You may find something like this:

```
Role: Contributor
Scope: /subscriptions/.../resourceGroups/<RESOURCE_GROUP_NAME>
```

If you see this, do not change it yet.

This is the main finding for the lab.

7. Understand what the role means

Now investigate the meaning of the `Contributor` role.

Run:

```
az role definition list `
  --name Contributor `
  --output json
```

Look for the `permissions` section.

You may see fields like:

```
actions
notActions
dataActions
notDataActions
```

The output may be long. Focus on the general meaning.

7.1. Question 6

What does the Contributor role generally allow?

Your answer:

7.2. Question 7

Does Contributor mean “full administrator”? If not, what is the main difference?

Your answer:

7.3. Question 8

Why is Contributor at resource group scope risky for an application identity?

Your answer:

8. Compare role assignments

In this lab environment, not every role assignment is necessarily bad.

Some access may be normal. Some access may be too broad.

Complete the table below. For the human role you can start with: az ad (**tip**: use `--id`)

Identity	Type	Role	Scope	Looks acceptable?	Why / why not?
Your student user	Human				
Backend managed identity	Workload				

Use these questions to help you:

- Does this identity need access?
- Does it need this specific role?
- Does it need access at this scope?
- Could the scope be narrower?
- Could the role be less powerful?

9. Identify the overprivileged identity

Now write down your main finding.

9.1. Finding

The overprivileged identity is:

9.2. Evidence

The role assigned to this identity is:

The scope of the role assignment is:

9.3. Why this is risky

This is risky because:

9.4. What the identity probably needs instead

A safer permission model would be:

10. Write a short security finding

Write your finding as if you were reporting it to the application team. In the next lab we will use the finding to attempt an exploit, and finally solve the issue.

Use the structure below.

10.1. Finding title

10.2. Evidence

10.3. Risk

10.4. Impact

10.5. Recommendation

11. Reflection questions

Answer the following questions individually or in pairs.

11.1. Question 9

What is the difference between a **human identity** and a **workload identity**?

Your answer:

11.2. Question 10

Why is scope important in Azure RBAC?

Tip: Azure RBAC scope overview

Your answer:

11.3. Question 11

Why is `Contributor` at resource group scope usually too broad for an application identity?

Your answer:

11.4. Question 12

What could happen if an attacker gained control over an application that uses an overprivileged managed identity?

Your answer:

12. Final conclusion

At the end of this lab, your conclusion should look similar to this:

```
The backend managed identity has Contributor access at resource group scope.
This is too broad because the backend application does not need to manage the full
resource group.
The application should only receive the permissions it actually needs, preferably
scoped to the specific resource it must access.
```

This lab covers the first two steps: **discover excessive access** and **understand the risk**.