THOMAS MORE

Lab 1.4

Designing a Zero Trust Access Model

Toepasbare Cloud Security voor IT-Professionals

2025 - 2026

Contents

1. Designing a Zero Trust Access Model

1.1. Context

In the previous labs you investigated, tested, and improved the access model of the Azure lab environment.

You started with a backend application identity that had broad access at resource group scope. You then tested the impact of that access and changed the Terraform configuration so the application received more specific permissions.

This lab closes Day 1 by translating those technical changes into a simple Zero Trust access model.

This is not a heavy technical lab. It is a short design and reflection exercise.

The goal is to describe access in a way that is explicit, limited, verified, monitored, and removable.

1.2. Learning goals

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

- How the RBAC fix from Lab 1.3 supports least privilege.
- Which identities need access in the lab environment.
- Which access should be permanent and which access should be temporary.
- How access could be monitored.
- How access could be revoked after the lab or after a real incident.
- How Zero Trust thinking changes the way we design cloud permissions.

1.3. Estimated time

30 minutes

Suggested timing:

Time	Activity
5 minutes	Recap the Day 1 access problem
10 minutes	Fill in the current access model
10 minutes	Design a Zero Trust target model
5 minutes	Discuss remaining risks and Day 2 bridge

1.4. Scenario

The application now works with a safer permission model than before.

However, fixing one role assignment is not the same as designing a complete access model.

The team asks you to answer this question:

“If we had to run this application in a real environment, who should have access, to what, for how long, under which conditions, and how would we know if that access is misused?”

This lab helps you answer that question.

1.5. Before you start

Make sure you have the results from the previous labs:

- [ ] The overprivileged identity from Lab 1.1
- [ ] The impact test results from Lab 1.2
- [ ] The Terraform changes from Lab 1.3
- [ ] The final behavior of `/api/security/impact-demo/tag-self`
- [ ] The final behavior of `/api/security/secret-demo`

> **Note**
>
> You do not need to make more Terraform changes in this lab unless the instructor explicitly asks for it. Focus on the access model and the reasoning behind it.

2. Zero Trust in this lab

Zero Trust does not mean that nobody receives access.

Zero Trust means that access is:

- **Explicit**: access is intentionally granted, not accidental.
- **Limited**: access is scoped to what is needed.
- **Verified**: identities and conditions are checked.
- **Monitored**: activity can be reviewed and investigated.
- **Removable**: access can be revoked when it is no longer needed.

In Day 1, you mainly worked on the first two parts: explicit and limited access.

The next step is to think about verification, monitoring, and revocation.

2.1. Question 1

Which part of the original access model violated least privilege the most?

Your answer:

2.2. Question 2

Which change in Lab 1.3 made the access model more explicit or more limited?

Your answer:

3. Current access model

Start by describing the access model as it exists after your Lab 1.3 fix.

Fill in the table below.

Identity	Type	Access to what?	Role / permission	Scope	Why is this needed?
Student user	Human				
Backend managed identity	Workload				
Instructor / operator	Human				
Terraform runner	Human or workload				

3.1. Guiding questions

Use these questions while filling in the table:

- Does this identity need access during normal application runtime?
- Does this identity only need access during deployment?
- Could the access be scoped to a single resource instead of the full resource group?
- Is this management-plane access or data-plane access?
- Would the application still work if this access was removed?

4. Target Zero Trust model

Now design a better target model.

You do not need to implement every idea today. The goal is to describe what a safer real-world model would look like.

Who or what needs access?	To what?	At what scope?	For how long?	Under what conditions?	How is it monitored?	How is it revoked?
Developer						
Backend managed identity						
Operator / instructor						
CI/CD or deployment identity						

4.1. Example ideas

You may use the ideas below, but adapt them to your own lab environment.

Question	Example answer
Who needs access?	Developer, backend managed identity, operator, deployment identity
To what?	App Service, Key Vault, logs, Terraform-managed resources
At what scope?	Specific resource where possible, resource group only when justified
For how long?	Temporary for troubleshooting, permanent only for runtime needs
Under what conditions?	MFA, approved device, approved workflow, pull request review
How is it monitored?	Activity logs, App Service logs, Key Vault access logs, alerts
How is it revoked?	Remove group membership, remove role assignment, disable identity, rotate secret

5. Runtime access versus deployment access

A common cloud security mistake is mixing runtime permissions and deployment permissions.

The application identity normally needs access only while the application runs.

A deployment identity may need access while Terraform or a pipeline changes the environment.

Those are different jobs, so they should usually use different identities.

5.1. Question 3

Which permissions belong to the backend application at runtime?

Your answer:

5.2. Question 4

Which permissions belong to a deployment process, such as Terraform or a CI/CD pipeline?

Your answer:

5.3. Question 5

Why is it risky when a runtime application identity also has broad deployment-style permissions?

Your answer:

6. Monitoring and investigation

Access is safer when it can be observed.

In this lab environment, think about which events would be useful to monitor after Day 1.

Event to monitor	Why does it matter?	Where might you look?
Role assignment changed		
App Service configuration changed		
Key Vault secret accessed		
Managed identity used unexpectedly		
Terraform applied outside the expected lab window		

6.1. Question 6

Which single event from the table would you prioritize first, and why?

Your answer:

7. Revocation and cleanup

A Zero Trust access model should include a removal plan.

Access that was needed during a lab, deployment, troubleshooting session, or incident should not automatically stay forever.

Fill in the cleanup plan below.

Access or resource	When should it be removed?	How would you remove it?
Temporary role assignments		
Student lab resources		
Demo secrets		
Unused managed identities		

8. Group discussion

Discuss your target model with another student or group.

Compare the following points:

- Did you give the backend identity only the access it needs?
- Did you separate runtime access from deployment access?
- Did you avoid broad resource group permissions where possible?
- Did you include a monitoring idea?
- Did you include a revocation or cleanup step?

Write down one improvement suggested by someone else.

Improvement suggestion: