THOMAS MORE

Lab 1.2

Demonstrating the Impact of Excessive and Incorrect Permissions

Toepasbare Cloud Security voor IT-Professionals

2025 - 2026

Contents

1. Demonstrating the Impact of Excessive and Incorrect Permissions 4
   1.1. Context 4
   1.2. Learning goals 4
   1.3. Estimated time 4
   1.4. Scenario 5
   1.5. Before you start 5
2. Confirm your Azure context 6
3. Confirm the backend managed identity 7
4. Confirm the role assignment 8
5. Prediction 9
6. Micro-theory: management plane and data plane 10
   6.1. Management plane 10
   6.2. Data plane 10
7. Test 1: Try to read the Key Vault secret 11
   7.1. Expected result 11
   7.2. Question 1 11
   7.3. Question 2 11
   7.4. Question 3 12
8. Test 2: Check the current tags 13
9. Test 3: Demonstrate excessive management-plane access 14
   9.1. Expected result 14
   9.2. Question 4 14
   9.3. Question 5 14
10. Verify the change 15
    10.1. Question 6 15
11. Check the Activity Log 16
    11.1. Question 7 16
12. Investigate what Contributor allows 17
    12.1. Question 8 17
13. Compare the test results 18
    13.1. Question 9 18
    13.2. Question 10 18
14. Attacker thinking exercise 19
    14.1. Question 11 19
15. Evidence bundle 20
16. Write a short security finding 21
    16.1. Finding title 21
    16.2. Evidence 21
    16.3. Risk 21
    16.4. Impact 21
    16.5. Recommendation 21
17. Example finding 22
    17.1. Finding title 22

17.2. Evidence 22
17.3. Risk 22
17.4. Impact 22
17.5. Recommendation 22
18. Reflection questions 23
18.1. Question 12 23
18.2. Question 13 23
18.3. Question 14 23
18.4. Question 15 23
19. Final conclusion 24
20. Bridge to Lab 1.3 25

1. Demonstrating the Impact of Excessive and Incorrect Permissions

1.1. Context

In Lab 1.1 you investigated the role assignments in your Azure lab environment.

You found that the backend application has a **managed identity**. This identity belongs to the App Service and can be used by the application to access Azure resources without storing credentials in code.

You also found that the backend managed identity has a broad role assignment:

```
Role: Contributor
Scope: Resource group
```

In this lab you will demonstrate why that finding matters.

You will not fix the issue yet. The fix will be done in Lab 1.3.

1.2. Learning goals

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

- What the backend managed identity can do because it has Contributor at resource group scope.
- What the backend managed identity still cannot do because it lacks the correct Key Vault data-plane role.
- The difference between management-plane access and data-plane access.
- Why broad access is not the same as correct access.
- How to collect evidence for a security finding.

1.3. Estimated time

60 minutes

Suggested timing:

Time	Activity
5 minutes	Scenario briefing and prediction
10 minutes	Confirm identity and role assignment
10 minutes	Test Key Vault access
15 minutes	Test and verify management-plane impact
10 minutes	Activity Log and Contributor investigation
10 minutes	Evidence-based finding

1.4. Scenario

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

To avoid permission problems during development, the backend managed identity was given `Contributor` access on the full resource group.

At first, this may look convenient:

“The backend can manage whatever it needs in the lab environment.”

But this creates a security problem.

The backend application should not be able to manage the full resource group. If the application is compromised, an attacker may be able to use the application’s identity to change Azure resources.

At the same time, this broad role does not automatically mean that the backend can read secrets from Key Vault. Key Vault secret access is data-plane access, while `Contributor` is mainly management-plane access.

The key lesson of this lab is:

“Broad access is not the same as correct access.”

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 backend application URL
- [ ] The result from Lab 1.1

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

2. Confirm your Azure context

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

Run:

```
az account show --output json
```

Write down:

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

3. Confirm the backend managed identity

Find the managed identity of the backend App Service.

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. Confirm the role assignment

Now inspect the role assignments for the backend managed identity.

Use the `principalId` from the previous step.

Run:

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

You should find a role assignment similar to this:

Identity	Role	Scope
Backend managed identity	Contributor	Resource group

Write down the exact result:

Identity	Role	Scope
Backend managed identity		

5. Prediction

Before calling the endpoints, predict what will happen.

Test	Expected result	Why?
`GET /api/security/security/secret-demo`		
`POST /api/security/security/impact-demo/tag-self`		

Use these hints:

- Reading a Key Vault secret is data-plane access.
- Updating an App Service tag is management-plane access.

6. Micro-theory: management plane and data plane

Azure permissions can affect different types of actions.

6.1. Management plane

Management-plane actions create, change, configure, or delete Azure resources.

Examples:

- Add or change tags on a resource.
- Change App Service configuration.
- Restart an App Service.
- Create new resources.
- Delete resources.
- Change diagnostic settings.

The `Contributor` role is powerful in the management plane. It allows an identity to manage many Azure resources, but it does not allow that identity to assign Azure RBAC roles to other identities.

6.2. Data plane

Data-plane actions use or read the data inside a service.

Examples:

- Read a Key Vault secret value.
- Download a blob from Storage.
- Query data from a database.
- Read messages from a queue.

For Key Vault, reading a secret value requires the correct Key Vault data-plane permission.

> **Note**
>
> This is why the backend identity can be overprivileged and still not have the exact permission the application needs.

7. Test 1: Try to read the Key Vault secret

First, test whether the backend application can perform the action it probably needs: reading a secret from Key Vault.

Call the secret demo endpoint:

```
curl https://<BACKEND_APP_URL>/api/security/security/secret-demo
```

Or open the endpoint in your browser:

```
https://<BACKEND_APP_URL>/api/security/security/secret-demo
```

7.1. Expected result

The request should fail.

The exact error message may differ, but it will likely indicate that the backend identity is not authorized to read the Key Vault secret.

Write down what you observe:

Item	Observation
Endpoint called	`/api/security/security/secret-demo`
HTTP status code	
Result	
Error message	

7.2. Question 1

The backend identity has `Contributor` on the resource group. Why does the secret request still fail?

Your answer:

7.3. Question 2

Does this failure mean the backend identity is safe?

Your answer:

7.4. Question 3

Is there anything in the `main.tf` that could explain this? If so, what and why?

Your answer:

8. Test 2: Check the current tags

Before changing anything, inspect the current tags on the backend App Service.

Run:

```
az webapp show `
  --resource-group <RESOURCE_GROUP_NAME> `
  --name <BACKEND_APP_NAME> `
  --query tags `
  --output json
```

Write down the current state:

Item	Value
Existing tags	
Does `security-lab-impact` already exist?	

9. Test 3: Demonstrate excessive management-plane access

Now call the impact demo endpoint.

Run:

```
curl.exe -X POST https://<BACKEND_APP_URL>/api/security/impact-demo/tag-self
```

This endpoint uses the backend App Service's managed identity to modify the App Service resource itself.

The endpoint performs a harmless change by adding or updating a tag.

Expected tag:

```
security-lab-impact = demonstrated
```

9.1. Expected result

The request should succeed.

Write down the result:

Item	Observation
Endpoint called	`/api/security/impact-demo/tag-self`
HTTP status code	
Result	
Message returned by the API	

9.2. Question 4

Which identity performed this action?

Your answer:

9.3. Question 5

Why was the backend identity able to modify the App Service resource?

Your answer:

10. Verify the change

Now verify that the App Service resource was changed.

Run:

```
az webapp show `
  --resource-group <RESOURCE_GROUP_NAME> `
  --name <BACKEND_APP_NAME> `
  --query tags `
  --output table
```

You should see a tag similar to:

```
security-lab-impact    demonstrated
```

Also verify this in the Azure Portal:

```
Azure Portal
→ Resource groups
→ <RESOURCE_GROUP_NAME>
→ <BACKEND_APP_NAME>
→ Tags
```

Write down the evidence:

Evidence	Value
Resource changed	
Tag name	
Tag value	
Verified with CLI?	
Verified in Portal?	

10.1. Question 6

Why is changing a tag a safe demonstration for this lab?

Your answer:

11. Check the Activity Log

The tag change is not only visible on the App Service. Azure should also record the management action.

Open the Azure Portal and go to:

```
Azure Portal
→ Resource groups
→ <RESOURCE_GROUP_NAME>
→ Activity log
```

Filter or search for the update event.

Useful filters:

- Timespan: Last hour
- Event category: Administrative
- Resource: backend App Service
- Operation: write or update

Write down the evidence:

Evidence	Value
Operation name	
Resource	
Caller / identity	
Status	
Time	

11.1. Question 7

Why is the Activity Log useful evidence in a security investigation?

Your answer:

12. Investigate what Contributor allows

Now inspect the `Contributor` role definition.

Run:

```
az role definition list `
  --name Contributor `
  --query "[0].permissions[0].actions" `
  --output table
```

The output may contain broad action patterns.

Pick three actions or action patterns that look risky in this lab environment.

Action or action pattern	Why could this be risky?

12.1. Question 8

What more dangerous actions might be possible with `Contributor` at resource group scope?

Your answer:

13. Compare the test results

Complete the table.

Test	Plane	Expected result	Actual result	What does it prove?
Read Key Vault secret	Data plane	Fails		
Modify App Service tag	Management plane	Works		

13.1. Question 9

What is the most important difference between these two actions?

Your answer:

13.2. Question 10

Why is the current access model both too broad and incomplete?

Your answer:

14. Attacker thinking exercise

For each action, decide whether it is likely possible with the current permission model.

Possible action	Likely possible?	Why?
Add or change App Service tags		
Read Key Vault secret value		
Change App Service app settings		
Assign Owner to another user		
Delete resources in the resource group		
Disable or change diagnostic settings		

14.1. Question 11

If an attacker gained control over the backend application, what could they potentially do with this managed identity?

Your answer:

Your answer:

15. Evidence bundle

Before writing your finding, collect the following evidence.

Evidence item	Collected?	Notes
Backend managed identity `principalId`		
Role assignment showing `Contributor` at resource group scope		
Failed `/api/security/secret-demo` result		
Successful `/api/security/impact-demo/tag-self` result		
Tag visible on the App Service		
Activity Log entry showing the update		

16. Write a short security finding

Write your finding as if you were reporting it to the application team.

Use the structure below.

16.1. Finding title

16.2. Evidence

16.3. Risk

16.4. Impact

16.5. Recommendation

17. Example finding

You can compare your answer with this example.

17.1. Finding title

Backend managed identity has excessive `Contributor` access on the resource group.

17.2. Evidence

The backend App Service has a system-assigned managed identity. This identity has the `Contributor` role assigned at resource group scope.

The `/api/security/impact-demo/tag-self` endpoint successfully used this identity to modify the App Service resource by adding or updating a tag.

The `/api/security/secret-demo` endpoint failed because the identity does not have the correct Key Vault data-plane permission.

17.3. Risk

The backend application can perform management-plane actions in the resource group. If the application is compromised, an attacker may be able to use the managed identity to change Azure resources.

17.4. Impact

Possible impact includes unauthorized configuration changes, changes to monitoring settings, modification of resources, creation of new resources, or disruption of the environment.

17.5. Recommendation

Remove the `Contributor` role assignment from the backend managed identity.

Grant the backend only the specific permission it requires, such as Key Vault secret access scoped to the Key Vault.

Verify that the application still works after the change, and verify that the previous management-plane action no longer works.

18. Reflection questions

Answer the following questions individually or in pairs.

18.1. Question 12

Why was `Contributor` convenient for the development team?

Your answer:

18.2. Question 13

Why is convenience dangerous when assigning cloud permissions?

Your answer:

18.3. Question 14

What is the difference between a role being powerful and a role being appropriate?

Your answer:

18.4. Question 15

How does this lab prepare for the RBAC fix in Lab 1.3?

Your answer:

19. 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 allows the backend application to perform management-plane actions in the
resource group. The lab demonstrated this by using the backend identity to modify a
tag on the App Service resource.

However, the backend identity still cannot read the Key Vault secret, because it does
not have the correct Key Vault data-plane permission.

The current access model is therefore both too broad and incorrect.

The backend identity should not have Contributor on the resource group. It should
only receive the specific permission it needs, scoped to the specific resource it
must access.
```

20. Bridge to Lab 1.3

In the next lab, you will fix the access model.

You will move from this unsafe state:

```
Backend managed identity
→ Contributor
→ Resource group scope
```

To a safer state:

```
Backend managed identity
→ Key Vault Secrets User
→ Key Vault scope
```

Then you will verify both sides of the fix:

Endpoint	Before fix	After fix
`POST /api/security/impact-demo/tag-self`	Works	Fails
`GET /api/security/secret-demo`	Fails	Works