

Lab 5.5

Auto-Remediation of Policy Violations

Toepasbare Cloud Security voor IT-Professionals

Contents

1. Auto-Remediation of Policy Violations	3
1.1. Context	3
1.2. Learning goals	3
1.3. Estimated time	3
1.4. Before you start	3
2. Understand What Can Be Remediated	5
2.1. Why this matters	5
3. Select a Remediable Policy	6
4. Trigger or Wait for Policy Evaluation	8
5. Create a Remediation Task	9
5.1. Navigate to Policy Compliance	9
5.2. Start remediation	9
6. Monitor Remediation Progress	10
6.1. In the Azure portal	10
6.2. Optional CLI check	10
7. Verify the Result	11
8. Alert-Only Policy for Risky Actions	12
8.1. Find an audit policy	12
8.2. Assign the audit policy	12
8.3. Review audit evidence	12
9. Important Limitation: Management Plane	14
10. Reflection Questions	15
11. Summary	16

1. Auto-Remediation of Policy Violations

1.1. Context

In Labs 5.2 and 5.3, you assigned Azure Policy guardrails and built an approval-based governance flow.

Those controls are useful, but they do not automatically solve every existing problem.

Some policies prevent future violations. Some policies only report violations. Some policies can actively change or deploy configuration to bring existing resources back into compliance.

In this lab, you will create and monitor an Azure Policy remediation task. You will also discuss when auto-remediation is useful, and when alert-only detection is safer.

Note

Azure Policy remediation tasks are only supported for policy assignments that use the `'DeployIfNotExists'` or `'Modify'` effect. `'Deny'` prevents matching create or update requests. `'Audit'` reports non-compliance and writes audit information, but it does not fix the resource.

1.2. Learning goals

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

- What an Azure Policy remediation task is.
- Why only some policy effects support remediation.
- Why remediation tasks require a managed identity on the policy assignment.
- How to create and monitor a remediation task.
- Why some violations should be auto-fixed and others should only trigger review or alerting.
- How audit-only policies can support detection without immediately breaking workloads.

1.3. Estimated time

45 minutes

Suggested timing:

Time	Activity
10 minutes	Review policy effects and select a remediable policy
10 minutes	Create a remediation task
10 minutes	Monitor remediation progress
10 minutes	Verify the result and inspect compliance
5 minutes	Reflection and discussion

1.4. Before you start

Make sure you have:

- [] Day 5 infrastructure deployed.

- [] Lab 5.3 policy assignments available, or permission to create a new policy assignment.
- [] Owner or Contributor access on your lab resource group.
- [] Permission to create policy assignments at the selected scope.
- [] Access to Azure Policy in the Azure portal.

Note

For policy assignments with `Modify` or `DeployIfNotExists`, Azure Policy uses a managed identity to make the required changes. If the assignment does not have the required managed identity and permissions, remediation will fail.

2. Understand What Can Be Remediated

Not every policy can fix resources.

Effect	What it does	Can remediate existing resources?
Deny	Blocks matching create or update requests.	No
Audit	Reports non-compliance and writes audit information.	No
Modify	Adds, updates, or removes supported properties or tags.	Yes
DeployIfNotExists	Deploys related configuration when it is missing.	Yes

Note

A common example of a `Modify` policy is adding or inheriting a required tag. A common example of a `DeployIfNotExists` policy is deploying diagnostic settings when they are missing.

2.1. Why this matters

A remediation task is useful for existing resources.

For example:

- A tag policy can add a missing tag to existing resources.
- A diagnostic settings policy can deploy missing diagnostic settings.
- A TLS configuration policy can configure an existing service to use a required minimum version.

But remediation is not always safe.

For example:

- Automatically disabling public network access might break a workload.
- Automatically changing application settings might interrupt an application.
- Automatically modifying shared infrastructure can affect other teams.

3. Select a Remediable Policy

For this lab, use a policy with the `Modify` or `DeployIfNotExists` effect.

A good classroom option is a tag policy, because the effect is visible and low risk.

Recommended example:

Add or replace a tag on resources

or:

text Inherit a tag from the resource group if missing

```
#note(
"The exact built-in policy names can change over time. Search for `tag` in Azure
Policy definitions and choose a built-in policy that uses the `Modify` effect. Your
instructor may also provide a prepared policy assignment for this lab."
)
```

== Check the policy effect

1. In the Azure portal, search for `*Policy*`.
2. Open `*Definitions*`.
3. Search for a tag-related policy.
4. Open the policy definition.
5. Inspect the policy rule.
6. Confirm that the effect is ``Modify``.

```
#question(
"Why is a tag policy a safer first remediation example than a policy that changes
network access?"
)
```

= Assign the Policy

If the policy is not already assigned from Lab 5.3, assign it now.

1. Open the selected policy definition.
2. Click `*Assign*`.
3. Set the scope to your lab resource group.
4. Configure the required parameters.

For example:

```
#table(
columns: (1fr, 2fr),
[*Parameter*], [*Example value*],
[Tag name], [`environment`],
[Tag value], [`lab`],
)
```

5. On the remediation or advanced tab, make sure a managed identity is created for

the assignment if the portal asks for one.
6. Review and create the assignment.

```
#note(  
"For `Modify` and `DeployIfNotExists` policies, the policy assignment identity needs  
permissions to perform the remediation. The Azure portal often helps configure this,  
but in real environments you should verify exactly which role was granted and at  
which scope."  
)
```

= Create a Non-Compliant Resource

Create or identify a resource in your lab resource group that does not have the required tag.

You can inspect resources with:

```
bash az resource list  
  --resource-group <RESOURCE_GROUP_NAME> --query "[].{name:name,type:type,tags:tags}"  
  --output table
```

If needed, remove the lab tag from a test resource:

```
az resource tag `  
  --resource-group <RESOURCE_GROUP_NAME> `  
  --name <RESOURCE_NAME> `  
  --resource-type <RESOURCE_TYPE> `  
  --tags ""
```

Note

Only do this on your own lab resources. Do not remove tags from shared or instructor-managed resources.

4. Trigger or Wait for Policy Evaluation

Policy compliance evaluation is not always instant.

You can wait for Azure Policy to evaluate the assignment, or ask the instructor whether it is acceptable to trigger a scan.

```
az policy state trigger-scan `
  --resource-group <RESOURCE_GROUP_NAME>
```

Note

Even after triggering a scan, compliance results can take several minutes to update in the portal.

5. Create a Remediation Task

5.1. Navigate to Policy Compliance

1. In the Azure portal, search for **Policy**.
2. Open **Compliance**.
3. Set the scope to your lab resource group.
4. Find the policy assignment you created or used.
5. Open the policy assignment or non-compliant policy result.

5.2. Start remediation

1. Open the **Remediation** tab.
2. Click **Create remediation task**.
3. Select the policy assignment.
4. Set the scope to your lab resource group.
5. Give the remediation task a clear name.

Suggested name:

remediate-tags-<your_prefix>

6. Leave the default settings unless your instructor tells you otherwise.
7. Create the remediation task.

Note

The remediation task runs in the background. It applies the `Modify` operation or deploys the `DeployIfNotExists` template to existing non-compliant resources in scope.

6. Monitor Remediation Progress

6.1. In the Azure portal

1. Go to **Policy**.
2. Open **Remediation**.
3. Find your remediation task.
4. Inspect the status and details.

You may see information such as:

- Current status.
- Number of resources selected.
- Number of resources attempted.
- Number of resources remediated successfully.
- Number of resources that failed.

Note

A failed remediation does not always mean the policy is wrong. Common causes include missing managed identity permissions, unsupported resource types, resource locks, or policy parameters that do not match the resource.

6.2. Optional CLI check

List remediation tasks for the resource group:

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

Show details for one remediation task:

```
az policy remediation show `
  --resource-group <RESOURCE_GROUP_NAME> `
  --name <REMEDIATION_TASK_NAME> `
  --output json
```

7. Verify the Result

After the remediation task completes, check the resource tags again:

```
az resource list `
  --resource-group <RESOURCE_GROUP_NAME> `
  --query "[].{name:name,type:type,tags:tags}" `
  --output table
```

Then return to Azure Policy compliance and check whether the resource is now compliant.

Note

Compliance state may lag behind the actual resource change. If the tag is already visible on the resource but the compliance page still shows non-compliant, wait a few minutes or trigger a new policy scan.

Question

Which changed first in your environment: the resource configuration or the compliance status in Azure Policy?

8. Alert-Only Policy for Risky Actions

Not every violation should be auto-remediated.

Some findings should create visibility first, because an automatic fix could break something.

Examples:

Public network access on a storage account. Public access on an application endpoint. **Network rules that are intentionally open for a temporary migration.** Settings used by a public static website or demo environment.

For those cases, an Audit policy may be safer than a Deny or auto-remediation policy.

8.1. Find an audit policy

1. In the Azure portal, open **Policy**.
2. Click **Definitions**.
3. Search for:

Storage accounts should disable public network access

4. Open the built-in policy if it is available in your tenant.
5. Inspect the policy rule and parameters.
6. Confirm which effects are allowed.

Note

Built-in policy names and available parameters can change. If this exact policy is not visible, search for `storage public network access` and choose a similar audit-capable storage policy.

8.2. Assign the audit policy

1. Click **Assign**.
2. Set the scope to your lab resource group.
3. Keep the effect as Audit if the policy allows effect selection.
4. Review and create the assignment.

Question

Why might `Audit` be safer than `Deny` for public network access during an investigation or migration?

8.3. Review audit evidence

After policy evaluation, inspect the policy compliance result.

You can also inspect Activity Log events in the Azure portal:

1. Open your resource group.
2. Open **Activity log**.
3. Filter for policy-related events or recent storage account operations.

Note

An audit policy creates visibility. It does not automatically mean an incident occurred. Use the signal as a reason to investigate ownership, business purpose, exposure, and risk.

9. Important Limitation: Management Plane

Azure Policy mainly evaluates Azure Resource Manager resource configuration and management-plane operations.

That means Azure Policy is useful for questions such as:

- Is this resource configured according to our rules?
- Was this resource created or updated in a non-compliant way?
- Is required configuration missing?

It is not a replacement for application-level or data-plane monitoring.

For example:

- Azure Policy can evaluate whether a Key Vault is configured securely.
- Key Vault diagnostic logs are needed to investigate who accessed a secret.
- Application Insights or Log Analytics are needed to investigate application behavior.

Note

This is why the next lab focuses on KQL queries and anomaly detection. Policy helps enforce and assess configuration. Logs help investigate behavior.

10. Reflection Questions

Your answer:

1. Which resources were remediated automatically?
2. Which policy effect made remediation possible?
3. What managed identity or permissions were needed for the remediation task?
4. Which kinds of violations would you auto-remediate in a real environment?
5. Which kinds of violations would you only audit or alert on first?
6. What could go wrong if auto-remediation is enabled too broadly?

11. Summary

In this lab, you created or used an Azure Policy assignment that supports remediation, started a remediation task, monitored its progress, and verified the result.

You also compared remediation with audit-only governance.

The key lesson is:

“Auto-remediation is powerful when the fix is predictable and safe. Audit-only detection is better when the fix needs human judgment.”

Next, you will use KQL queries to detect unusual patterns in monitoring data.