THOMAS MORE

Lab 2.4

Protecting the Data Layer

Toepasbare Cloud Security voor IT-Professionals

2025 - 2026

Contents

1. Protecting the Data Layer

1.1. Context

In Lab 2.3 you restricted access to the internal backend. The direct public path to the internal backend is now blocked.

The DB storage account still has public network access enabled. Anyone who can reach the Table Storage endpoint and has valid credentials — or a leaked access key — can use it from anywhere on the internet.

In this lab you will move the DB storage account toward private access using a private endpoint. After the private endpoint is in place, you will disable public network access on the storage account.

> **Note**
>
> This lab uses Azure Table Storage as the data layer. Cosmos DB is the more common production choice for globally distributed workloads, but it is not available in West Europe on Visual Studio subscriptions due to regional capacity limits. The private endpoint concepts you practice here apply directly to Cosmos DB: the only differences are the resource type, sub-resource name (`Sql` instead of `table`), and the private DNS zone (`privatelink.documents.azure.com`).

1.2. Learning goals

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

- What a private endpoint is and what it does.
- Why a private DNS zone is required alongside a private endpoint.
- How to verify that a private endpoint is working correctly.
- The difference between removing public access and removing data access.
- How defense in depth applies to the data layer.

1.3. Estimated time

70 minutes

Suggested timing:

Time	Activity
5 minutes	Micro-theory and context
5 minutes	Verify starting state
15 minutes	Create private endpoint in the Portal
10 minutes	Verify private endpoint and DNS
10 minutes	Test with Azure Storage Explorer before and after disabling public access
10 minutes	Test data path failure and add VNet integration to fix
15 minutes	Reflection questions

1.4. Before you start

Make sure you have:

- [ ] The DB storage account name and resource group name from Lab 2.1 (or Portal: Storage accounts → Overview)
- [ ] The `public_backend_url` from Lab 2.1
- [ ] Azure Portal access
- [ ] Azure Storage Explorer installed on your local machine

2. Micro-theory: private endpoints and private DNS

2.1. Private endpoint

A private endpoint is a network interface inside a VNet that connects to an Azure PaaS service.

When a private endpoint exists, the service gets a private IP address inside the VNet. Traffic travels through the VNet instead of the public internet.

2.2. Private DNS zone

When you resolve a storage account hostname from outside the VNet, you get the public IP address.

When the private endpoint is created, a private DNS zone maps the same hostname to the private IP address. Resources inside the VNet get the private IP. Resources outside get the public IP (or nothing, if public access is disabled).

This means the application does not need to change its connection string. DNS does the routing.

Service	Sub-resource	Private DNS zone
Azure Table Storage	`table`	`privatelink.table.core.windows.net`
Azure Cosmos DB	`Sql` (or `MongoDB` etc.)	`privatelink.documents.azure.com`
Azure Key Vault	`vault`	`privatelink.vaultcore.azure.net`

> **Note**
>
> The private DNS zone must be linked to the VNet for internal name resolution to work.

3. Part 1 — Verify the starting state

3.1. Check storage account public access in the Portal

Navigate to your DB storage account in the Azure Portal (name starts with `stdb`).

Go to **Security + networking → Networking**.

Write down the current setting for public network access:

Your answer:

3.2. Test public reachability

```
curl -I https://<DB_STORAGE_ACCOUNT_NAME>.table.core.windows.net
```

Write down the HTTP response code:

Your answer:

3.3. Generate a SAS token and test public data access

To prove the storage account is truly publicly accessible, generate a Shared Access Signature (SAS) token and query the `WelcomeMessage` table from your browser.

Use the Access Key to generate a Shared Access Signature (SAS):

1. Open your DB storage account in the Azure Portal (name starts with `stdb`).
2. Navigate to **Security + networking → Shared access signature**.
3. Check the **Table** service box, set your allowed permissions (e.g., Read), and click **Generate SAS and connection string**.
4. Copy the generated **Table service SAS URL**.
5. Paste that URL into your browser.
6. Insert your table name (`WelcomeMessage`) between the endpoint (`/`) and the query string (`?`).

Example URL format:

```
https://stdb<your-prefix>.table.core.windows.net/WelcomeMessage?
sv=2022-11-02&ss=t&srt=o&sp=r&se=2026-05-28T03:39:17Z&st=2026-05-27T19:24:17Z&spr=https&sig=...
```

Navigate to the URL in your browser.

If the endpoint is reachable, your browser will show or download an XML file. Open it with a text editor.

Does it return table data?

Your answer (paste a snippet of the XML response or describe what you see):

This confirms the storage account is reachable from the public internet. Anyone with a SAS token (or an access key) can read your data.

3.4. Confirm the data path works

```
curl https://<PUBLIC_BACKEND_URL>/api/data-demo
```

Does it return data from the data layer?

Item	Value
HTTP status	
`correlationId`	
Data returned?	

4. Part 2 — Create the private endpoint

In the Azure Portal, navigate to your DB storage account (name starts with `stdb`).

Go to **Security + networking → Networking → Private endpoint connections** and click **+ Private endpoint**.

Fill in the wizard:

Basics tab:

Setting	Value
Resource group	Your lab resource group
Name	`pe-table-<your-prefix>`
Region	Same region as your resources

Resource tab:

Setting	Value
Connection method	Connect to an Azure resource in my directory
Resource type	`Microsoft.Storage/storageAccounts`
Resource	Your DB storage account
Target sub-resource	`table`

Virtual Network tab:

Setting	Value
Virtual network	Your lab VNet
Subnet	`snet-private-endpoints`
Private IP configuration	Dynamically allocate IP address

DNS tab:

Setting	Value
Integrate with private DNS zone	Yes
Private DNS zone	`privatelink.table.core.windows.net` (created automatically)

Click **Review + create**, then **Create**.

> **Note**
>
> Private endpoint provisioning takes 1-3 minutes. Wait for the deployment to complete before verifying.

The portal creates:

Resource	Purpose
Private endpoint	Private NIC for the DB storage account inside the VNet
Private DNS zone	`privatelink.table.core.windows.net`
DNS zone VNet link	Allows the VNet to resolve private DNS names

Connection between the private endpoint and the storage account is automatic when you select the sub-resource in the wizard. The private DNS zone is also created and linked to the VNet automatically when you select that option in the wizard.

Write down any issues encountered during creation:

Your answer:

5. Part 3 — Verify the private endpoint

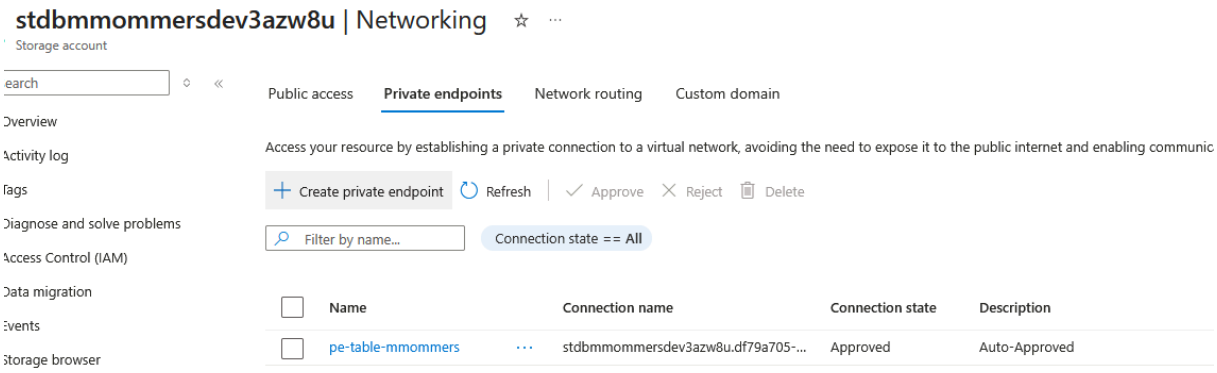
5.1. In the Azure Portal

1. Navigate to your DB storage account → **Networking → Private endpoint connections**.
2. Confirm the endpoint status shows **Approved**.

Write down the connection state:

Your answer:

Example:



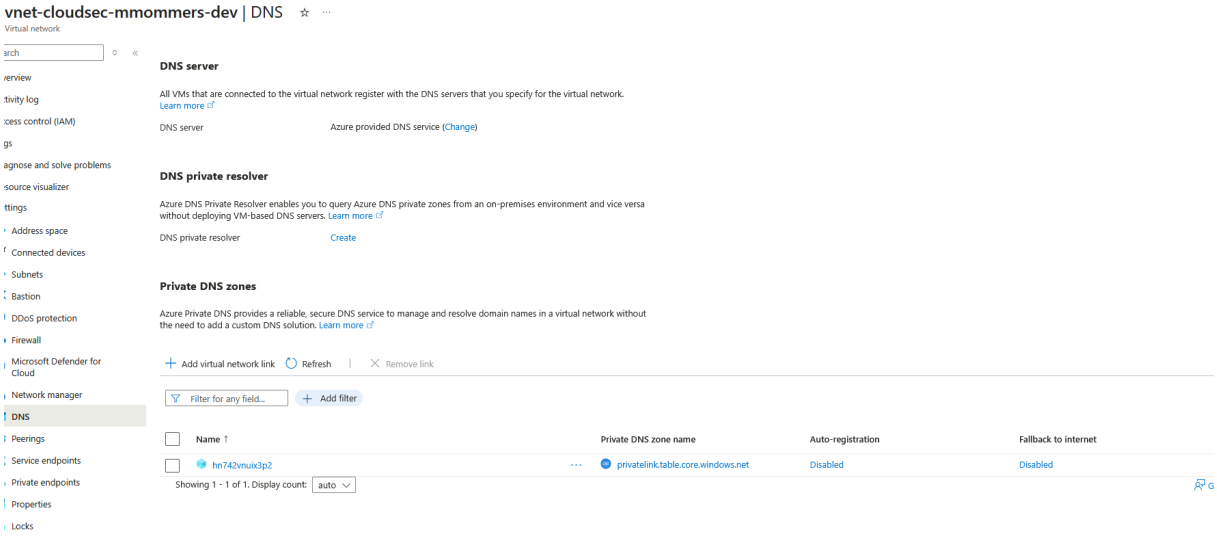
5.2. In the private DNS zone

1. Navigate to **Private DNS zones (part of VNet) → privatelink.table.core.windows.net**.
2. Confirm a DNS A record exists for your DB storage account name.

Write down the private IP address from the DNS record:

Item	Value
DB storage account name	
Private IP address	

DNS record added to the private DNS zone in the wizard.



Click **Save** and wait for the change to be applied (usually under 1 minute).

6. Part 4 — Verify access with Azure Storage Explorer

1. Open Azure Storage Explorer on your local machine.
2. Sign in with your Azure credentials (if not already signed in).
3. Navigate to your subscription → Storage Accounts → DB storage account (`stdb...`) → Tables
4. Try to expand the Tables node and open the `messages` table.

Can you browse the table contents?

Your answer:

> **Note**
> At this point, public network access is still enabled, so Storage Explorer can connect from your laptop through the public endpoint.

7. Part 5 — Disable public network access

In the Azure Portal, navigate to your DB storage account.

Go to **Security + networking → Networking**.

Under **Public network access**, select **Disabled from all networks**.

7.1. After disabling public access

After waiting about 1 minute, refresh Azure Storage Explorer. Try to expand the Tables node again.

What happens?

Your answer:

> **Note**
>
> You should see a connection error or timeout. Storage Explorer from your laptop can no longer reach the DB storage account because public network access is disabled and your laptop is not inside the VNet.

8. Part 6 — Test after disabling public access

8.1. Test public endpoint with SAS token (should be blocked)

Use the SAS token URL you created in Part 1 to test if you can still access the storage account from your laptop:

```
curl "https://<DB_STORAGE_ACCOUNT_NAME>.table.core.windows.net/messages?<SAS_TOKEN>"
```

Write down the result:

Item	Value
HTTP status or error	
Public access blocked?	

> **Note**
>
> Even with a valid SAS token, the request should fail because public network access is disabled. Network-level controls take precedence over authentication.

8.2. Test data path through the application (will fail)

```
curl https://<PUBLIC_BACKEND_URL>/api/data-demo
```

Write down the result:

Item	Value
HTTP status	
Data returned?	
`correlationId`	

> **Note**
>
> This test will fail. The internal backend cannot reach the DB storage account because public access is disabled and the internal backend does not have VNet integration configured yet. You will fix this in the next step.

8.3. Full trace demo

```
curl https://<PUBLIC_BACKEND_URL>/api/trace-demo
```

Does the trace still complete end to end?

Your answer:

> **Note**

> This will also fail for the same reason: the internal backend cannot reach the DB storage account without VNet integration.

9. Part 7 — Fix the data path with VNet integration

To allow the internal backend to reach the DB storage account's private endpoint, you need to configure VNet integration.

In the Azure Portal, navigate to your internal backend App Service (name contains `internal-backend`).

Go to **Settings → Networking → Outbound traffic → VNet integration**.

Click **+ Add VNet integration**.

Configure the integration:

Setting	Value
Virtual Network	Your lab VNet
Subnet	`snet-app-integration`

Click **OK**.

Wait about 30 seconds for the integration to activate.

9.1. Test the data path again (should now work)

```
curl https://<PUBLIC_BACKEND_URL>/api/data-demo
```

Write down the result:

Item	Value
HTTP status	
Data returned?	
`correlationId`	

```
curl https://<PUBLIC_BACKEND_URL>/api/trace-demo
```

Does the trace now complete end to end?

Your answer:

> **Note**
>
> With VNet integration, the internal backend's outbound traffic goes through the VNet. DNS resolves the storage account FQDN to the private endpoint's private IP (10.0.2.x), and the request succeeds even though public access is disabled.

10. Reflection questions

10.1. Question 1

We disabled public network access on the DB storage account. The application tests in Part 6 failed. What is missing, and why did it break?

Your answer:

10.2. Question 2

Why does DNS resolution matter for private endpoints?

Your answer:

10.3. Question 3

If an attacker obtains a storage account access key, can they now use it from their laptop?

Your answer:

10.4. Question 4

Table Storage data is encrypted at rest by default. A private endpoint adds network-level protection. The internal backend uses managed identity for authentication (Storage Table Data Contributor/ Reader). What layer does each control protect?

Control	What it protects
Encryption at rest	
Private endpoint	
Managed identity (RBAC)	

> **Note**
>
> Cosmos DB provides the same three layers: encryption at rest (default), private endpoints, and managed identity via built-in data plane roles. The controls map identically.

11. Final conclusion

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

```
The DB storage account now has a private endpoint inside the VNet.
The private DNS zone resolves the storage account hostname to a private
IP for resources inside the VNet.

Public network access is disabled. A direct connection from the internet
to the Table Storage endpoint is now refused.

The internal backend can still reach Table Storage through the private
endpoint. The application data path continues to work.

We removed the public data path, not the data access itself.
```