THOMAS MORE

Optional Lab 2.x

Bastion Management Path

Toepasbare Cloud Security voor IT-Professionals

2025 - 2026

Contents

1. Bastion Management Path

1.1. Context

In Lab 2.4 you disabled public network access on the DB storage account. Only resources inside the VNet can now reach it through the private endpoint.

A common concern after locking down a database is: how do administrators access it for troubleshooting or maintenance?

This lab answers that question by showing the Azure Bastion management path.

1.2. Learning goals

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

- What Azure Bastion is and how it differs from a public SSH/RDP port.
- Why NSG rules must allow traffic from the Bastion subnet to the target VM.
- Why a jumpbox VM is used to reach private resources.
- How private DNS resolution works differently inside and outside the VNet.
- What a system-assigned managed identity is and how it enables credential-free authentication.
- How to use managed identities to access Azure services like Table Storage without storing credentials.

1.3. Estimated time

50 minutes (or 60 minutes with optional Part 8)

This lab is optional and instructor-led.

Suggested timing:

Time	Activity
5 minutes	Micro-theory
15 minutes	Deploy Bastion and management VM
5 minutes	Configure NSG to allow Bastion connectivity
5 minutes	Connect via Bastion and test DNS/TCP connectivity
5 minutes	Compare DNS resolution from laptop
10 minutes	Enable managed identity and grant storage access
5 minutes	Install Azure CLI and query tables using managed identity
10 minutes	**Optional:** Bastion tunnel demonstration
5 minutes	Reflection questions

> **Note**
>
> This lab requires Azure Bastion and a management VM deployed in the VNet. Both are created in Part 1 using the Azure Portal.

1.4. Before you start

Make sure you have:

- [ ] The resource group name and VNet name from Lab 2.1 (or Portal: Resource groups)
- [ ] Azure Portal access
- [ ] DB storage account public network access disabled (from Lab 2.4)

2. Micro-theory: Bastion and the management path

Azure Bastion provides browser-based SSH and RDP access to virtual machines inside a VNet.

The connection goes through the Azure Portal over HTTPS. The VM does not need a public IP address. No inbound SSH or RDP port needs to be open to the internet.

The management path in this lab:

Step	Path
1	Admin opens Azure Portal on their laptop
2	Admin connects to the management VM via Azure Bastion
3	A browser-based terminal session opens
4	The management VM is inside the VNet
5	The management VM resolves the DB storage account hostname to a private IP
6	The management VM connects to the DB storage account through the private endpoint

> **Note**
>
> Bastion is not used to access the DB storage account directly. Bastion is used to access a VM inside the VNet. That VM can then reach private resources.

3. Part 1 — Deploy Azure Bastion and a management VM

The `AzureBastionSubnet` (`10.0.10.0/27`) is already pre-provisioned in your VNet by Terraform.

3.1. Deploy Azure Bastion

1. In the Azure Portal, search for **Bastions** and click **+ Create**.
2. Fill in the **Basics** tab:

Setting	Value
Resource group	Your lab resource group
Name	`bas-<your-prefix>`
Region	Same region as your resources
Tier	Basic
Virtual network	Your lab VNet
Subnet	`AzureBastionSubnet` (pre-provisioned)
Public IP address	Create new — name it `pip-bastion-<your-prefix>`

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

> **Note**
> Bastion provisioning takes approximately 5-10 minutes.

3.2. Deploy a management VM

1. In the Azure Portal, search for **Virtual machines** and click **+ Create → Azure virtual machine**.
2. Fill in the **Basics** tab:

Setting	Value
Resource group	Your lab resource group
Virtual machine name	`vm-mgmt`
Region	Same region as your resources
Image	Ubuntu Server 24.04 LTS
Size	`Standard_B1s (or smallest available)`
Authentication type	SSH public key
Username	`azureuser`
SSH public key source	Generate new key pair

3. On the **Networking** tab:

Setting	Value
Virtual network	Your lab VNet

Subnet	`snet-private-endpoints`
Public IP	**None**
NIC network security group	Basic
Public inbound ports	None

4. Click **Review + create**, then **Create**. Download the SSH key when prompted.

> **Note**
>
> Setting Public IP to None means the VM has no direct internet access. It can only be reached through Azure Bastion.

Write down the VM name and private IP address (find it in: VM → **Overview** → **Private IP address**):

Item	**Value**
VM name	
Private IP address	

4. Part 2 — Configure NSG to allow Bastion connectivity

Before you can connect via Bastion, you need to allow inbound SSH traffic from the Bastion subnet to the management VM.

The management VM is in the `snet-private-endpoints` subnet, which has an NSG attached (`nsg-snet-private-endpoints`). By default, this NSG only has Azure default rules.

Azure Bastion sends RDP/SSH traffic from the `AzureBastionSubnet` (10.0.10.0/27) to target VMs in other subnets. We need to allow this traffic.

4.1. Add NSG rule to allow Bastion SSH

1. Navigate to the NSG resource: `nsg-snet-private-endpoints`.
2. Go to **Settings → Inbound security rules**.
3. Click **+ Add**.
4. Configure the rule:

Setting	Value
Source	IP Addresses
Source IP addresses/CIDR ranges	`10.0.10.0/27`
Source port ranges	*
Destination	Any
Service	Custom
Destination port ranges	22,3389
Protocol	Any
Action	Allow
Priority	100
Name	`allow-bastion-inbound`

5. Click **Add**.
6. Wait 30-60 seconds for the rule to propagate.

> **Note**
>
> The source IP range 10.0.10.0/27 is the AzureBastionSubnet CIDR. Bastion sends SSH (port 22) and RDP (port 3389) traffic from this subnet to reach VMs in other subnets.

Write down any issues encountered:

Your answer:

5. Part 3 — Connect to the VM via Bastion

Now that the NSG allows Bastion traffic, connect to the management VM:

1. Navigate to the management VM in the Azure Portal.
2. Click **Connect → Bastion**.
3. Enter `azureuser` as the username.
4. Select **SSH Private Key from Local File** and upload the key you downloaded when creating the VM.
5. Click **Connect**.

A browser-based terminal session opens.

Were you able to connect?

Your answer:

If the connection fails with "The target machine encountered an error and has closed the connection", check:

- NSG rule was saved and has priority 100
- Source is set to IP Addresses (not Service Tag)
- Source IP addresses/CIDR ranges is `10.0.10.0/27`
- Destination ports include `22,3389`
- Action is Allow
- Wait 60-90 seconds after adding the rule, then try connecting again

6. Part 4 — Test private DB storage access from the VM

From inside the Bastion terminal session, test DNS name resolution:

```
nslookup <DB_STORAGE_ACCOUNT_NAME>.table.core.windows.net
```

Write down the IP address returned:

Item	Value
Resolved IP address	
Is it a private IP (10.x.x.x)?	

Test TCP connectivity to the storage endpoint (port 443):

```
timeout 5 bash -c "</dev/tcp/<DB_STORAGE_ACCOUNT_NAME>.table.core.windows.net/443" &&
echo "Connection successful" || echo "Connection failed"
```

> **Note**
>
> This test opens a TCP connection to port 443 (HTTPS). If it succeeds, the private endpoint is reachable. Table Storage requires authentication for actual data operations, but we can verify network connectivity this way.

Write down the result:

Your answer:

7. Part 5 — Compare with resolution from your laptop

From your local machine (not through Bastion):

```
nslookup <DB_STORAGE_ACCOUNT_NAME>.table.core.windows.net
```

Write down the IP address returned by your laptop:

Item	Value
Resolved IP address	
Is it a public IP?	

Expected result: Your laptop resolves to a public IP address (not 10.x.x.x) because your laptop is not linked to the Private DNS zone.

8. Part 6 — Comparison table

Test	From management VM (inside VNet)	From your laptop (outside VNet)
DNS resolves to	Private IP (10.0.2.x)	Public IP
Why?	VNet linked to Private DNS zone	Not linked to Private DNS zone

The management VM can reach the storage account through the private endpoint (10.0.2.x). Your laptop cannot, because public network access is disabled on the DB storage account.

9. Part 7 — Query table data from the VM using Azure CLI

Now that you've proven network-level connectivity from inside the VNet, let's query actual table data using the Azure CLI from the management VM.

9.1. Enable managed identity on the VM

Instead of using device code authentication (which requires opening a browser on your laptop), we'll use a system-assigned managed identity on the VM. This allows the VM to authenticate to Azure services without storing any credentials.

1. In the Azure Portal, navigate to your management VM (`vm-mgmt`).
2. Go to **Security → Identity**.
3. Under the **System assigned** tab, toggle **Status** to On.
4. Click **Save**.
5. Wait 10-20 seconds for Azure to create the managed identity.

> **Note**
>
> A system-assigned managed identity is automatically tied to the VM's lifecycle. When the VM is deleted, the identity is also deleted. This is perfect for resources like VMs that need to access other Azure services.

Write down the Object (principal) ID shown on the Identity page:

Item	Value
Object (principal) ID	

9.2. Grant the managed identity access to Table Storage

Now grant the VM's managed identity permission to read table data from the DB storage account:

1. Navigate to the DB storage account (`stdb...`).
2. Go to **Access Control (IAM)**.
3. Click **+ Add → Add role assignment**.
4. On the **Role** tab, search for and select Storage Table Data Reader.
5. Click **Next**.
6. On the **Members** tab:
   - Select **Managed identity**
   - Click **+ Select members**
   - Filter by **Virtual machine**
   - Select your management VM (`vm-mgmt`)
   - Click **Select**
7. Click **Review + assign**.

> **Note**

> The Storage Table Data Reader role grants read-only access to table data. The VM can query tables but cannot insert, update, or delete entities.

9.3. Install Azure CLI and authenticate with managed identity

Back in the Bastion terminal session, install Azure CLI on the VM:

```
curl -sL https://aka.ms/InstallAzureCLIDeb | sudo bash
```

This will take 2-3 minutes to install.

Once installed, log in to Azure using the managed identity:

```
az login --identity
```

Expected output: You should see JSON showing the VM's managed identity subscription and tenant.

> **Note**
>
> The --identity flag tells Azure CLI to use the VM's system-assigned managed identity. No browser interaction needed!

9.4. Query table data

Query the `messages` table:

```
az storage entity query \
  --account-name <DB_STORAGE_ACCOUNT_NAME> \
  --table-name messages \
  --auth-mode login
```

Expected result: You should see JSON output showing the table entities (messages).

Query the `audit` table:

```
az storage entity query \
  --account-name <DB_STORAGE_ACCOUNT_NAME> \
  --table-name audit \
  --auth-mode login
```

Write down the number of entities returned from each table:

Table	Entity count
`messages`	
`audit`	

Were you able to query the tables using Azure CLI?

Your answer:

Note

The Azure CLI queries work because the VM is inside the VNet and uses the private endpoint (10.0.2.x). The traffic goes through the private endpoint, not the public internet. You're using Azure AD authentication (--auth-mode login) instead of access keys.

10. Part 8 — Alternative: Bastion tunnel (optional advanced feature)

Instead of connecting to the VM via Bastion browser session, you can use Azure Bastion tunnel to forward a local port on your laptop to the VM, allowing you to run Azure CLI commands directly from your laptop.

> **Note**
>
> This part is optional and requires the Azure Bastion **Standard SKU** (not Basic). The instructor can demonstrate this feature if desired.

10.1. Upgrade Bastion to Standard SKU (if needed)

If your Bastion is currently Basic tier, upgrade it to Standard:

1. Navigate to your Bastion resource in the Azure Portal.
2. Go to **Settings → Configuration**.
3. Change **Tier** from Basic to Standard.
4. Click **Apply**.
5. Wait 2-3 minutes for the upgrade to complete.

10.2. Use Bastion tunnel from your laptop

From your local machine (not through the browser Bastion session), open a terminal/PowerShell and run:

```
az network bastion tunnel `
  --name bas-<your-prefix> `
  --resource-group <your-resource-group> `
  --target-resource-id /subscriptions/<subscription-id>/resourceGroups/<rg>/
providers/Microsoft.Compute/virtualMachines/vm-mgmt `
  --resource-port 22 `
  --port 2222
```

This command:

- Opens a tunnel from your laptop's port 2222 to the VM's port 22 (SSH)
- Keeps running in the foreground (leave this terminal open)

10.3. Connect via the tunnel

Open a second terminal/PowerShell window and SSH to localhost:2222:

```
ssh azureuser@localhost -p 2222 -i <path-to-ssh-key>
```

You are now connected to the VM via the Bastion tunnel!

From this SSH session, you can run Azure CLI commands exactly as you did in Part 7:

```
az storage entity query \
  --account-name <DB_STORAGE_ACCOUNT_NAME> \
```

```
--table-name messages \
--auth-mode login
```

> **Note**
>
> The Bastion tunnel feature is useful for automation scenarios, CI/CD pipelines, or when you prefer native SSH/RDP clients over the browser-based terminal. The tunnel encrypts traffic through Azure Bastion without exposing SSH/RDP ports to the internet.

Write down whether you were able to use the Bastion tunnel:

Your answer:

11. Reflection questions

11.1. Question 1

You added an NSG rule to allow traffic from the Bastion subnet (10.0.10.0/27) to the management VM on ports 22 and 3389. Why is this NSG rule needed for Bastion to work?

Your answer:

11.2. Question 2

Why does the management VM resolve the DB storage account to a private IP, while your laptop does not?

Your answer:

11.3. Question 3

Why is Bastion safer than opening an inbound SSH port (port 22) directly on the management VM with a public IP?

Your answer:

11.4. Question 4

What is a system-assigned managed identity, and why is it better than storing credentials (like storage account access keys) on the VM?

Your answer:

11.5. Question 5

You were able to query tables using Azure CLI from the management VM. Why does Azure CLI work from the VM but would fail from your laptop (assuming you authenticated with `az login`)?

Your answer:

11.6. Question 6

An admin needs to run a one-off DB storage table query to investigate an incident. DB storage account public access is disabled. What is the correct path?

Your answer:

12. Final conclusion

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

```
Azure Bastion provides a secure browser-based terminal session
to a VM inside the VNet, without exposing SSH or RDP ports
to the public internet.

From inside the VNet (management VM):
- DNS resolves the DB storage account hostname to a private IP (10.0.2.x)
- TCP connection to port 443 succeeds (goes through private endpoint)
- Azure CLI queries work using managed identity authentication (no credentials
stored)
- Traffic goes through the private endpoint, not the public internet

From outside the VNet (your laptop):
- DNS resolves to a public IP
- Azure CLI would fail (no private endpoint access, even with valid credentials)
- With Bastion tunnel (Standard SKU), you can SSH from laptop to VM
  and run commands that access private endpoints

Key concepts:
- System-assigned managed identity eliminates credential management
- Private endpoints restrict network access to resources inside the VNet
- Bastion provides secure access without exposing management ports to the internet

Admins do not need to re-enable public access to query storage data.
They can:
1. Connect via Bastion browser session → use managed identity to query (Basic or
Standard SKU)
2. Use Bastion tunnel to forward SSH/RDP to laptop → run tools locally (Standard SKU
only)
```