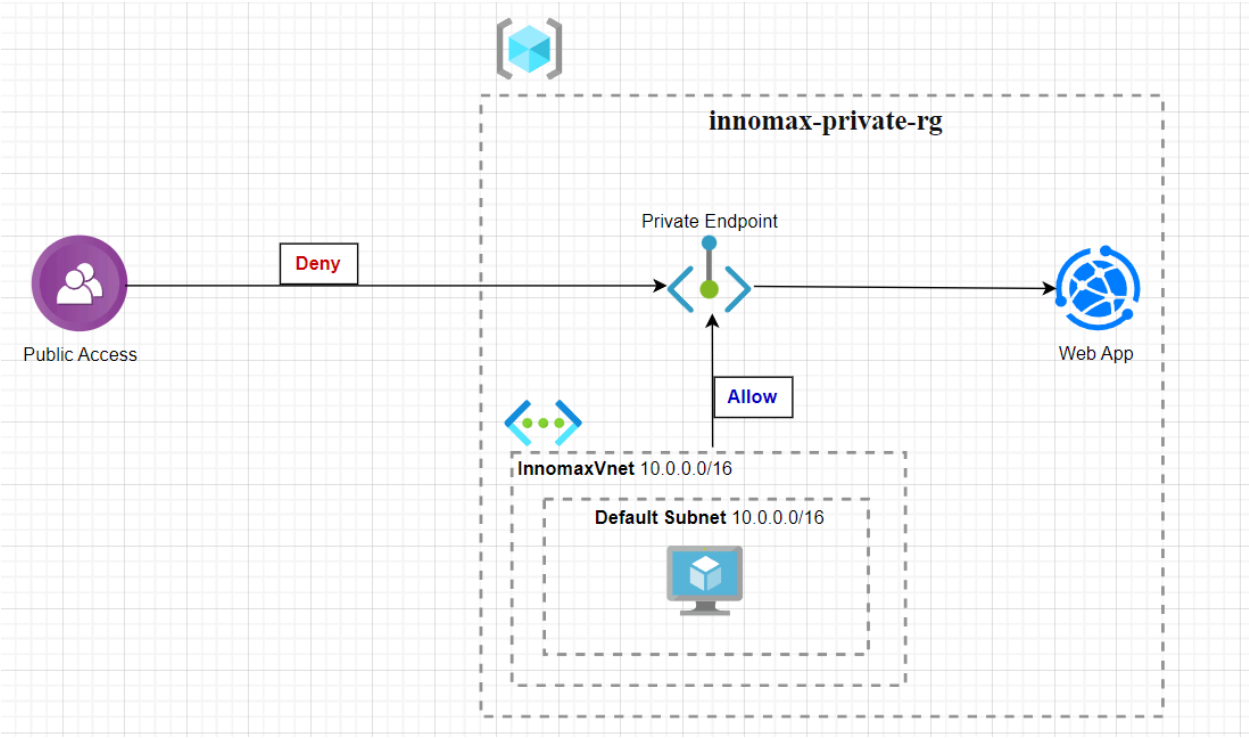
# Azure Web Apps Using Private Endpoints

# Implementing Zero-Trust Network Security for Azure Web Apps Using Private Endpoints




**Author**: Sai Min Thu, www.innomax.space, https://www.youtube.com/@SaiMinThuu, www.linkedin.com/in/saiminthu

**Date**: 6.9.2025

**Lab Objective**: To demonstrate how to completely remove public internet access from an Azure App Service Web App and secure it within a private virtual network using Private Endpoints, adhering to a zero-trust network model.

---

In today's threat landscape, the principle of "never trust, always verify" is paramount. While Azure Web Apps are publicly accessible by default, many

enterprise scenarios require workloads to be isolated from the public internet to meet strict compliance and security requirements.

This guide provides a step-by-step walkthrough of configuring an Azure Web App to be accessible only through a private network connection via an Azure Private Endpoint. We will:

1. Establish a foundational resource group and virtual network.
2. Deploy a basic web application.
3. Implement core security controls by creating a Private Endpoint and integrating with Private DNS.
4. Enforce network isolation by applying access restrictions.
5. Validate the security configuration.

## 2. Phase 1: Building the Foundation

### 2.1. Creating the Resource Group

All Azure resources are deployed into a container called a resource group, which helps manage their lifecycle collectively.

**Steps:**

1. Navigate to the **Azure Portal**.
2. Use the global search bar at the top to find and select **Resource groups**.
3. On the Resource groups blade, click the **+ Create** button.
4. **Basics Tab:**

**Subscription:** Choose your subscription.

**Resource Group:** Enter **innomax-private-rg**.

**Region:** Select a region closest to your users (e.g., East US).

5. Click **Review + create**, then **Create** after validation passes.

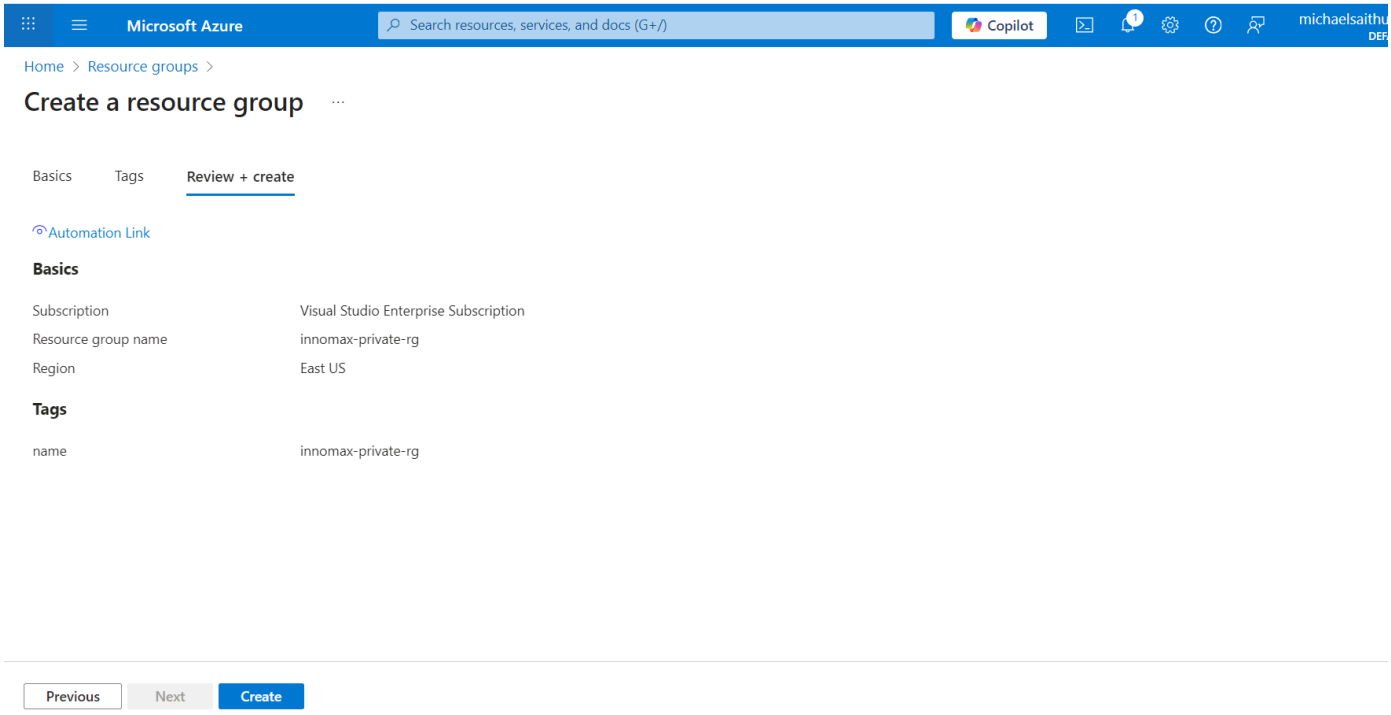



**Figure 1-1**

## 2.2. Provisioning the Virtual Network (VNet)

The Virtual Network (VNet) acts as our private, isolated network boundary in the cloud.

**Steps:**

1. In the portal search bar, search for and select **Virtual networks**.

2. Click **+ Create**.

3. **Basics Tab:**

**Subscription & Resource Group:** Select your subscription and the **innomax-private-rg** group created earlier.

**Name:** Enter **innomaxVnet**.

**Region:** Keep the same region as your resource group.

4. **IP Addresses Tab:**

Leave the default IPv4 address space (10.0.0.0/16) and default subnet (10.0.0.0/24).

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

**Documentation Notes:**

**VNet Name: innomaxVnet**

**Address Space:** 10.0.0.0/16

**Default Subnet:** 10.0.0.0/24

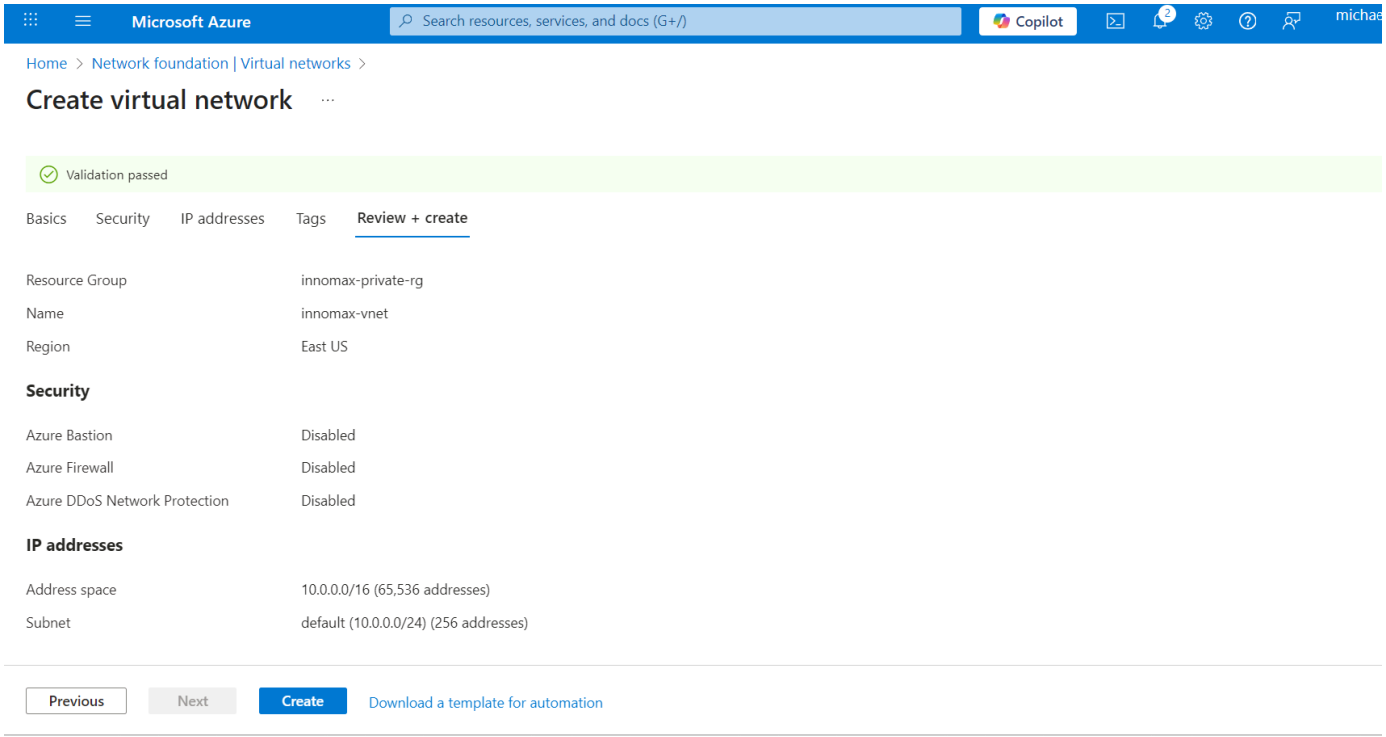



Figure 1-2

## 2.3. Deploying the Web App and App Service Plan

The App Service Plan defines the compute resources for your app, and the Web App is the application itself.

**Steps:**

Search for and select **App Services** in the portal.

Click + **Create**.

**Basics Tab:**

**Subscription & Resource Group:** Select your subscription and the **innomax-private-rg** group.

**Web App Name:** Enter a globally unique name (e.g., **innomax-app-space**).

**Publish:** Code

**Runtime Stack:** Select your preference (e.g., .NET 8 (STS)).

**Operating System:** Windows (typically auto-selected for .NET).

**Region:** Same region as before.

4. **Hosting Tab (Click 'Create new' under App Service Plan):**

**Name: webapp-plan**

**Pricing Tier:** Click **Change size** and select a production-ready tier like **B1** (Basic) or **S1** (Standard) for Private Endpoint support. The free (F1) tier does not support this feature.

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

**Documentation Notes:**

**Web App Name:** innomax-app-space

**App Service Plan:** webapp-plan (B1: Basic tier)

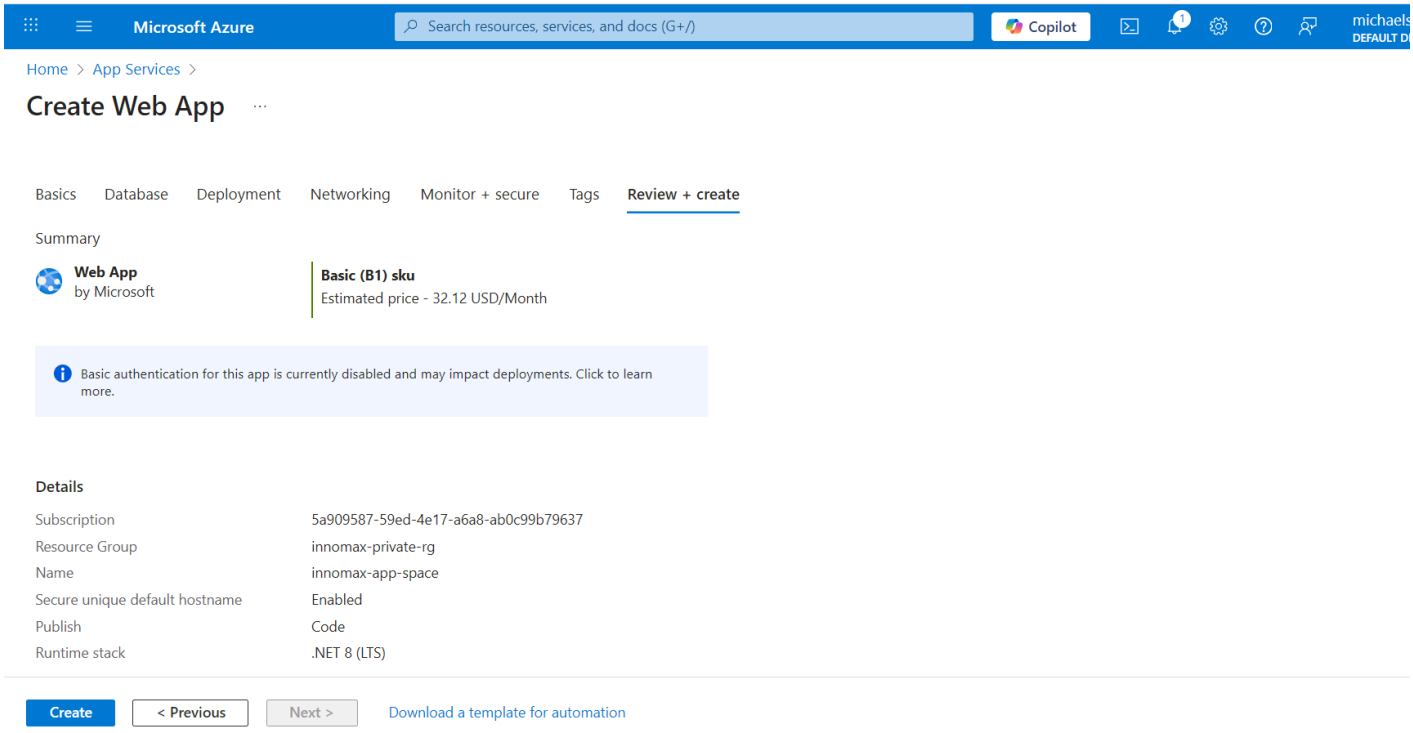



Figure 1-3

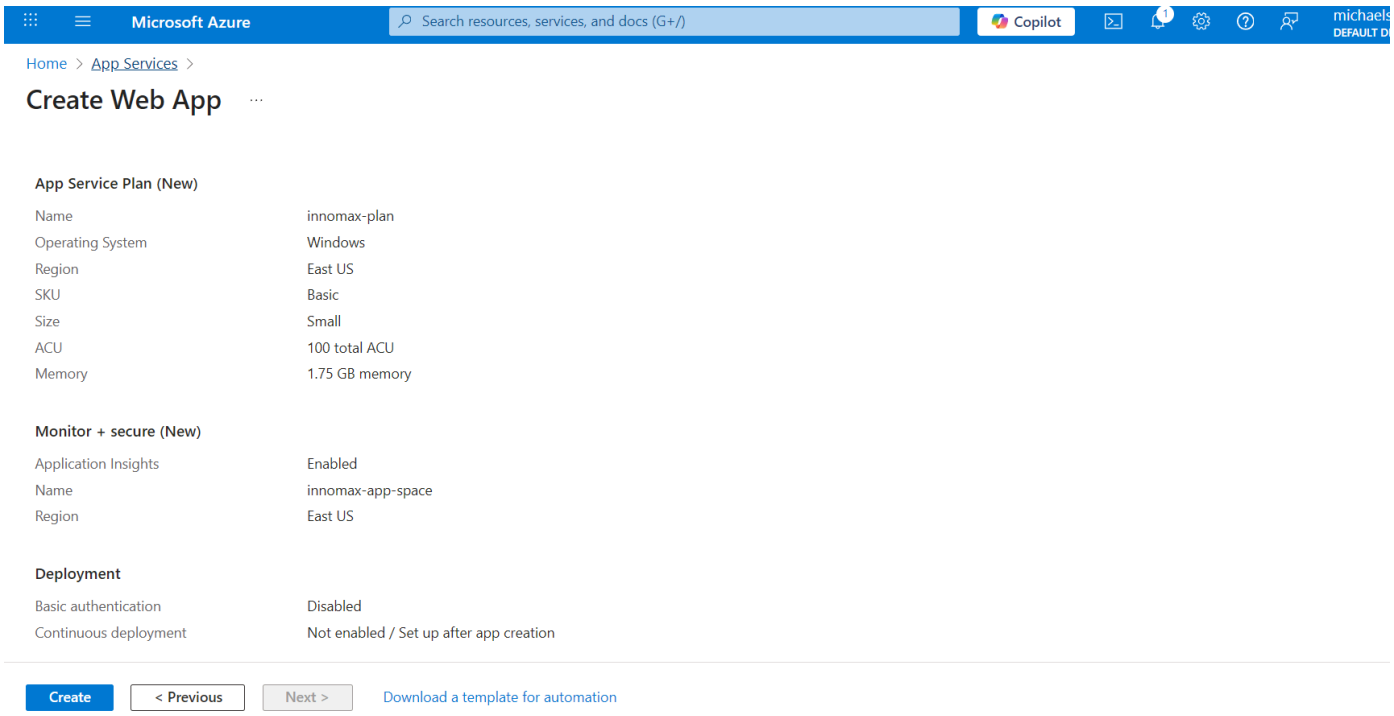



Figure 1-4

---

## 3. Phase 2: Implementing Security Controls

### 3.1. Creating the Private Endpoint

The Private Endpoint creates a private IP address inside your VNet, effectively making the Web App a first-class citizen on your private network.

**Steps:**

1. Navigate to your newly created **Web App** resource in the portal.
2. In the left-hand menu, under *Settings*, select **Networking**.
3. Under the "Outbound traffic" section, click on **Private endpoint connections**.
4. Click **+ Private endpoint**.
5. **Basics Tab:**

   **Name: Innomax-pe**

   **Subscription & Resource Group:** Select your subscription and the **innomax-private-rg** group.

6. **Resource Tab:**

   **Target sub-resource: sites** (This automatically refers to your web app).

7. **Networking Tab:**

   **Virtual network:** Select your **InnomaxVnet**.

   **Subnet:** Select the default subnet (10.0.0.0/24).

**Private DNS Integration: Crucially, ensure this is set to Yes.** This automatically creates the necessary DNS records so resources in the VNet can find your web app using its default name.

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

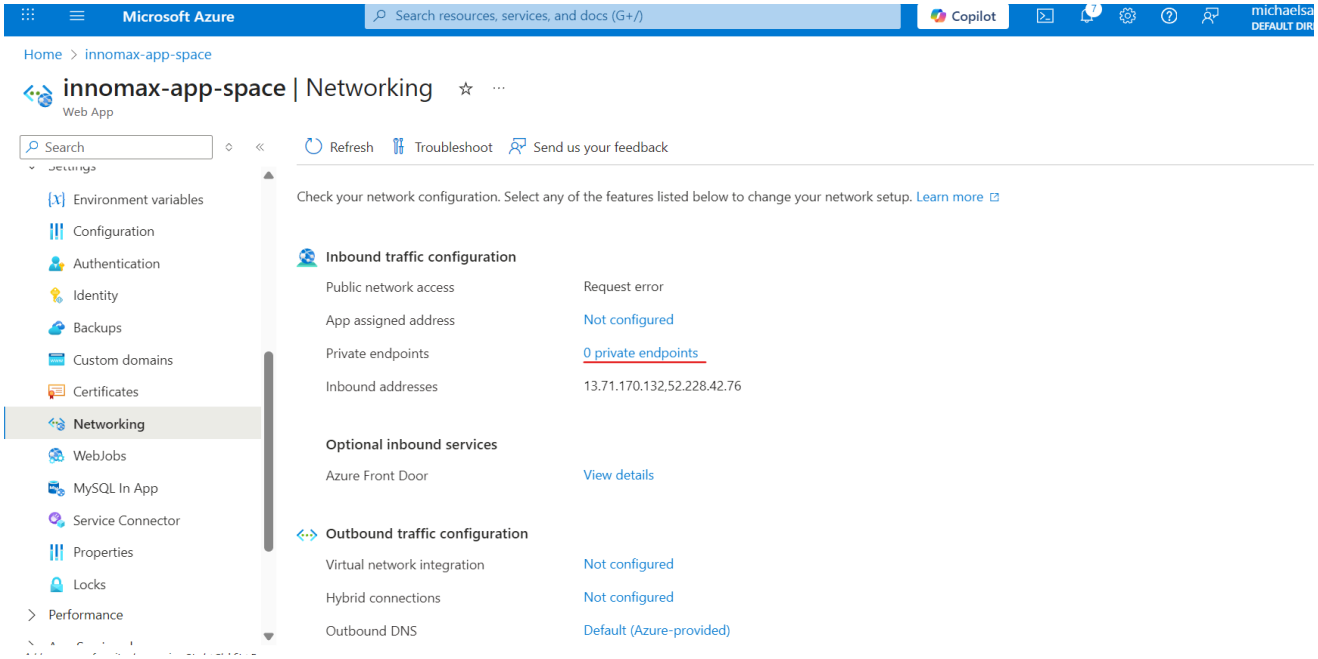



Figure 2-1

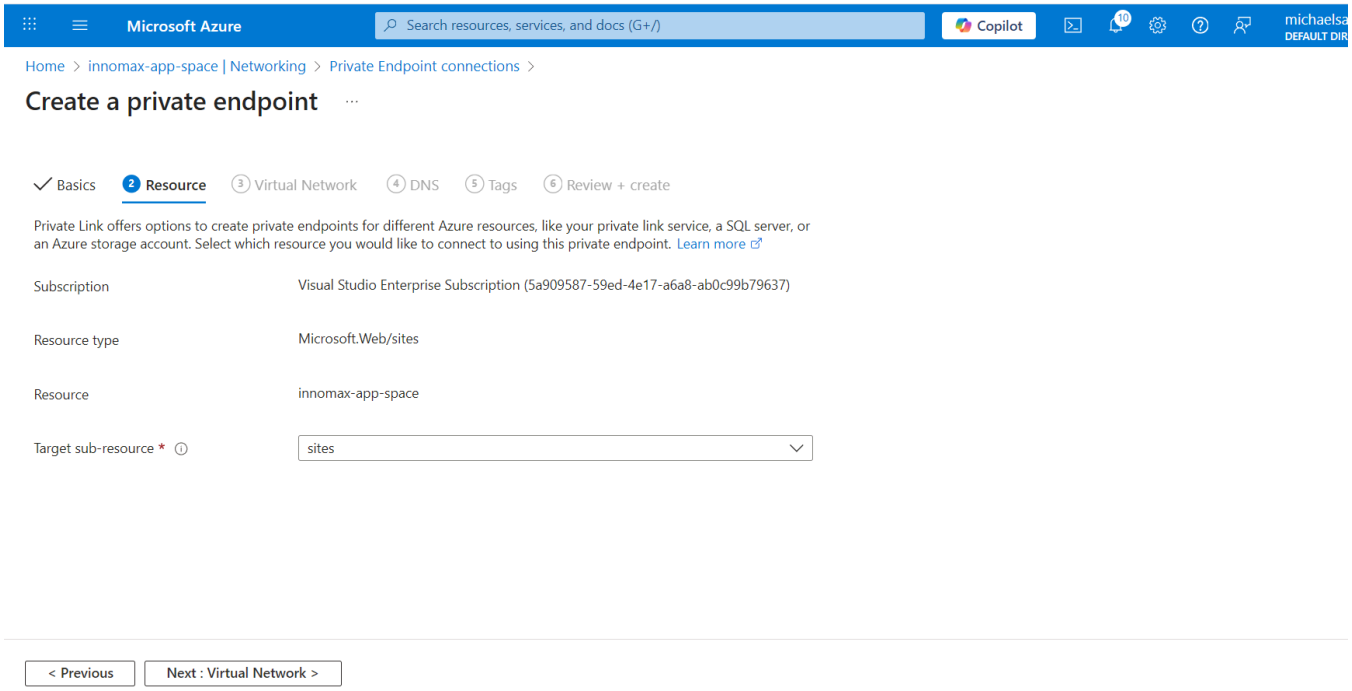



Figure 2-2

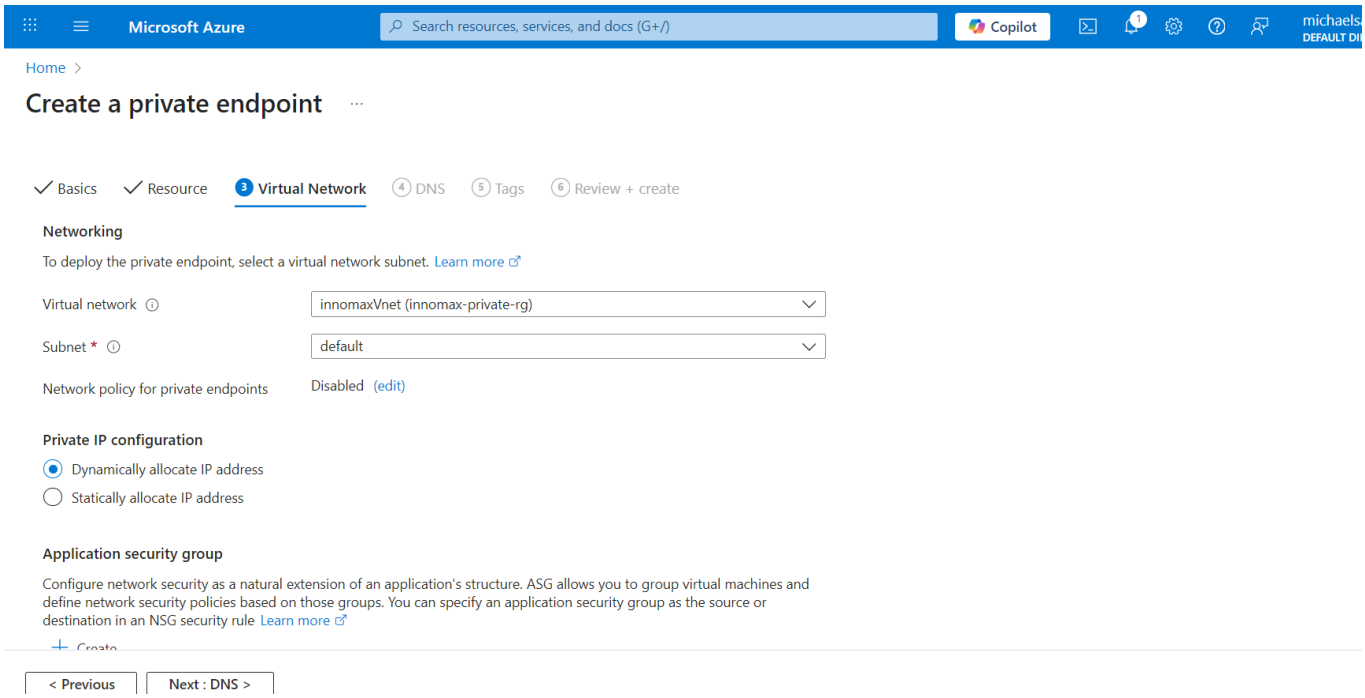



Figure 2-3

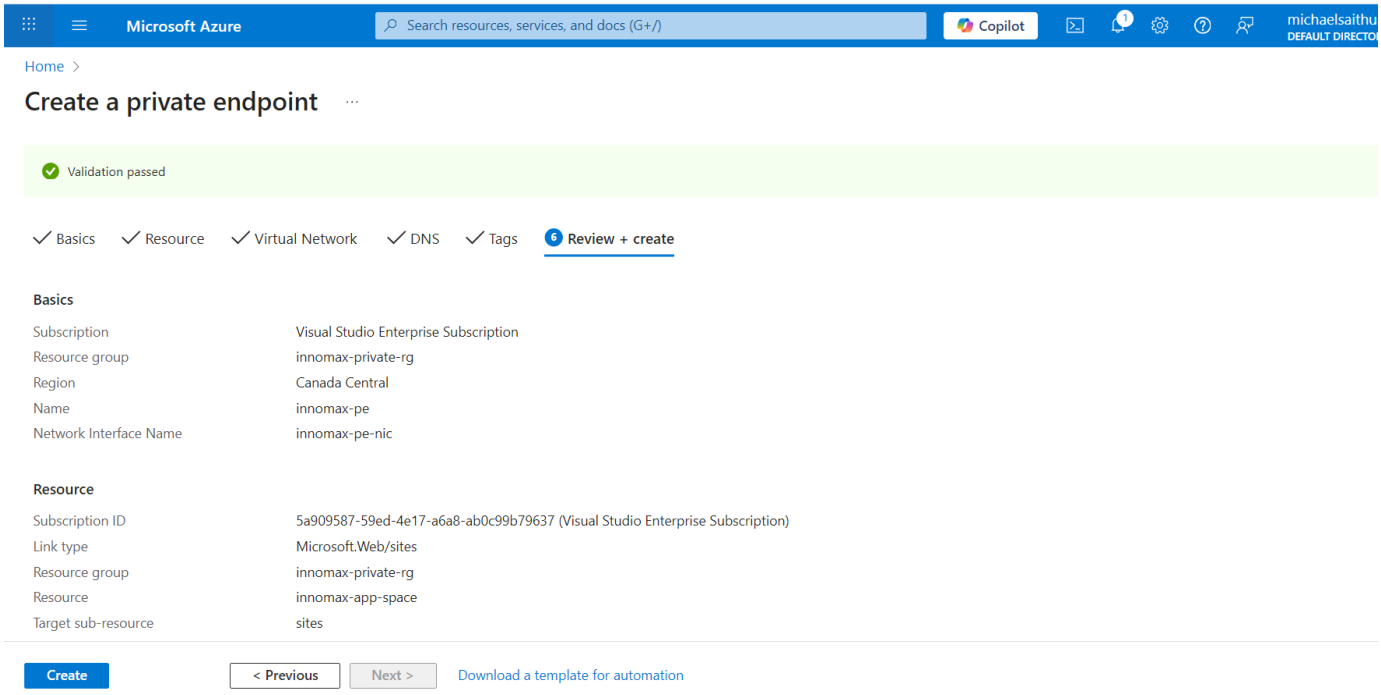



Figure 2-4

### 3.2. Restricting Public Access (Zero-Trust Enforcement)

Now that private access is configured, we must explicitly block all public traffic.

**Steps:**

1. Back in your Web App's **Networking** blade, under the "Inbound traffic" section, click on **Access restriction**.

2. You will see a default rule Allow all that allows traffic from any public IP (0.0.0.0/0).

3. **Click + Add rule** to create a Allow rule.

**Rule name: InnoAllowPrivate**

**Action: Allow**

**Priority:** 100 (or any number lower than the default Allow rule's priority, which is likely 65000). Lower numbers have higher precedence.

Leave the other fields blank to apply this deny rule to **all** traffic not matching other rules.

4. Click **Add**.

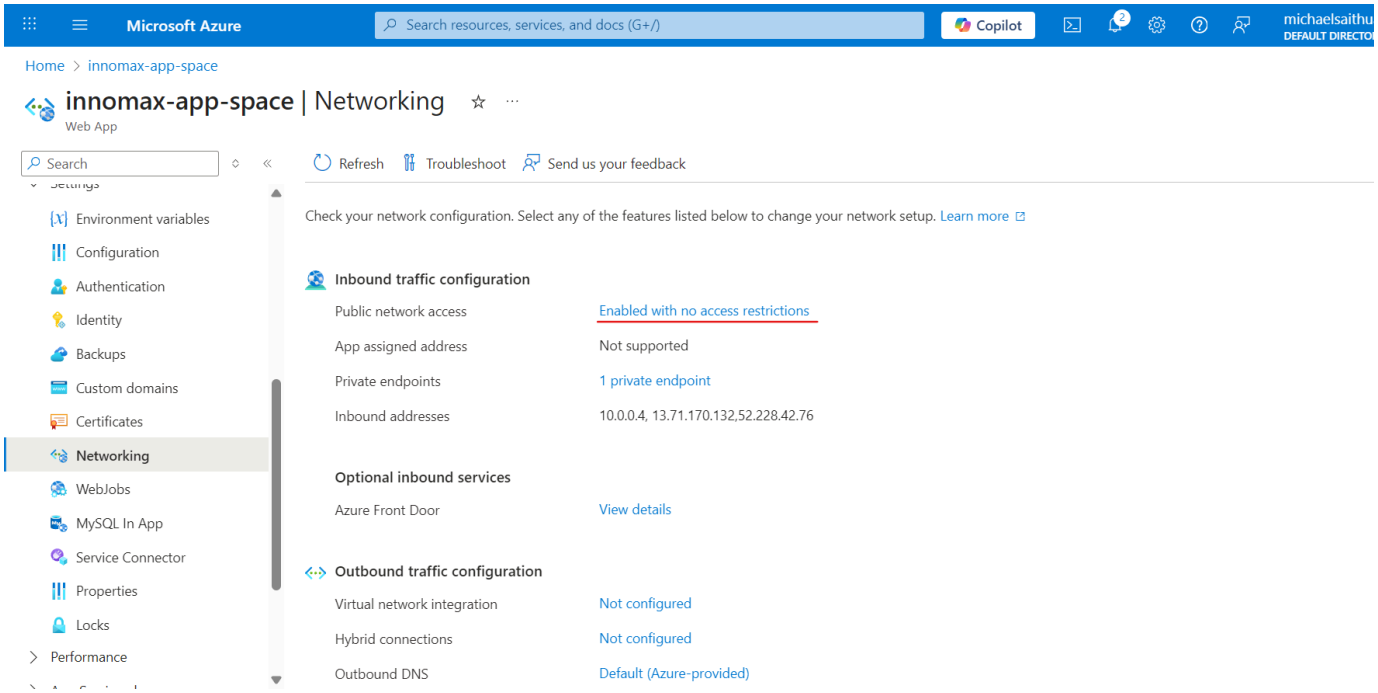



Figure 2-5

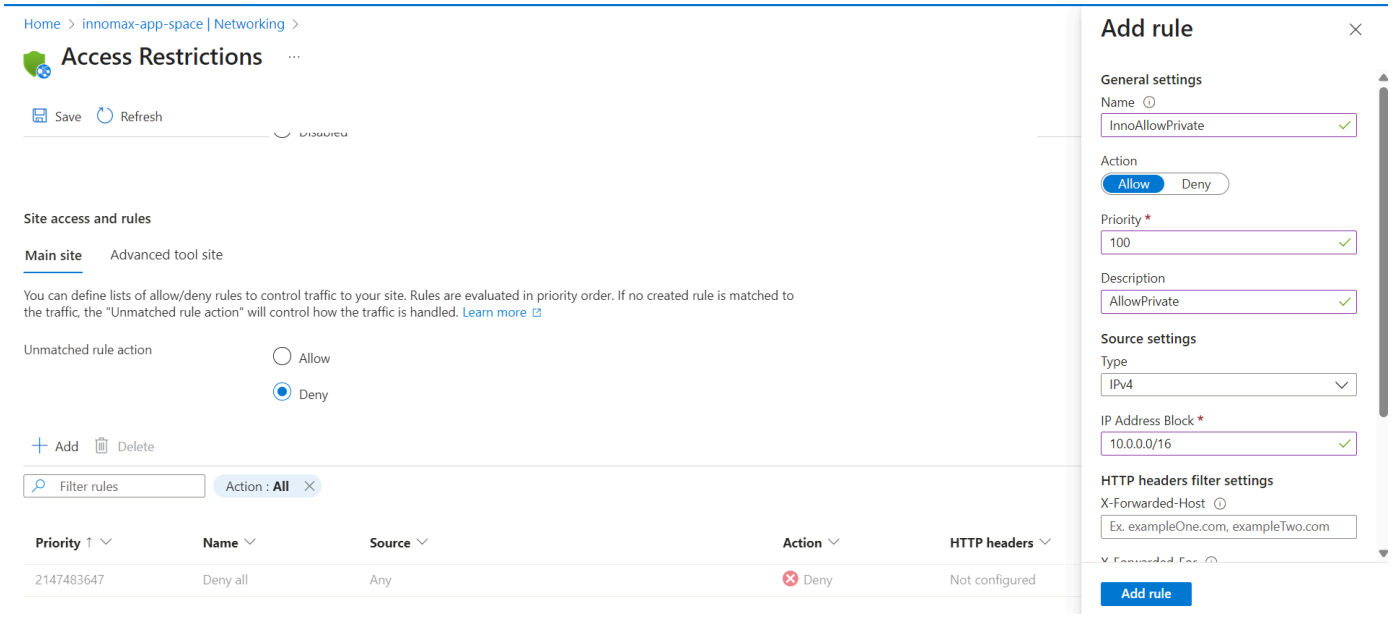



Figure 2-6

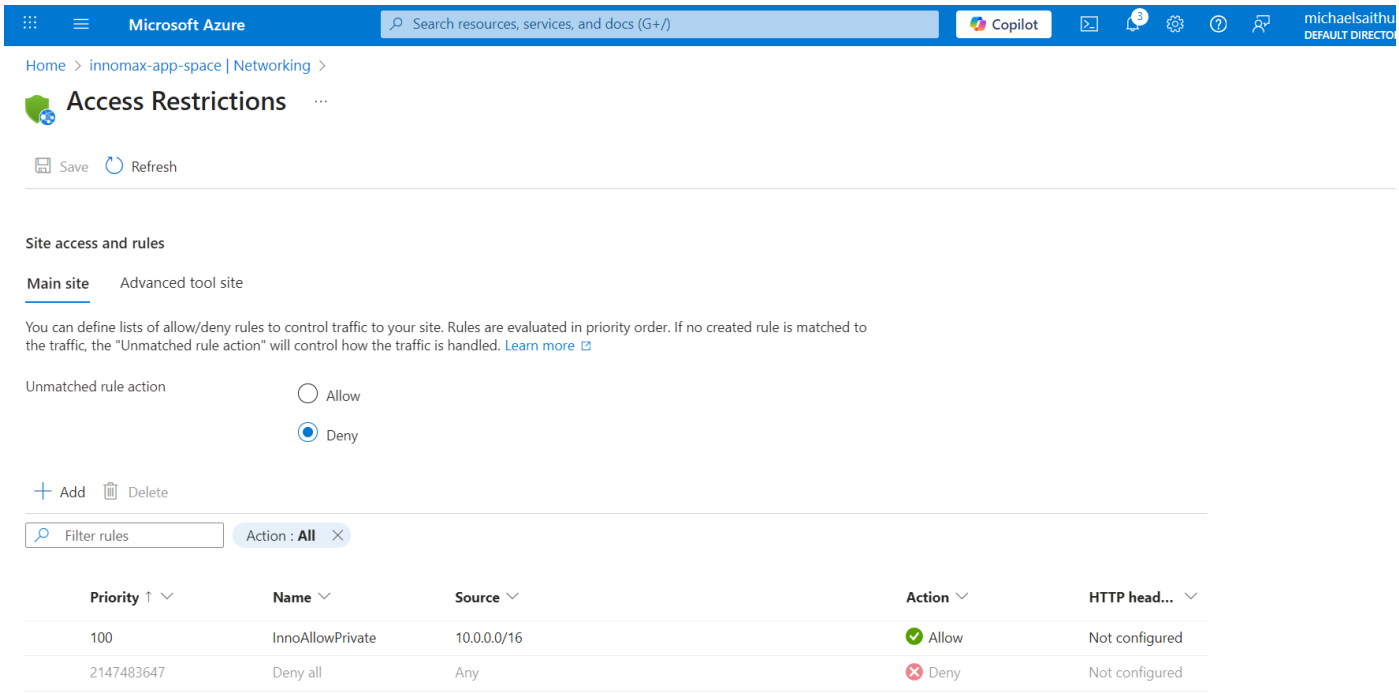



Figure 2-7

## 4. Validation and Results

The success of this security implementation is proven by testing connectivity from two different paths.

### Test 1: Public Internet Access (Should FAIL)

**Action:** Open a web browser on your local machine (which is on the public internet) and try to navigate to your Web App's URL: innomax-app-space-e4hzd7b0bmeybref.canadacentral-01.azurewebsites.net

**Expected Result:** The connection will time out or, more specifically, you will receive a **403 - Forbidden** error. This confirms that the public access restriction is working correctly.

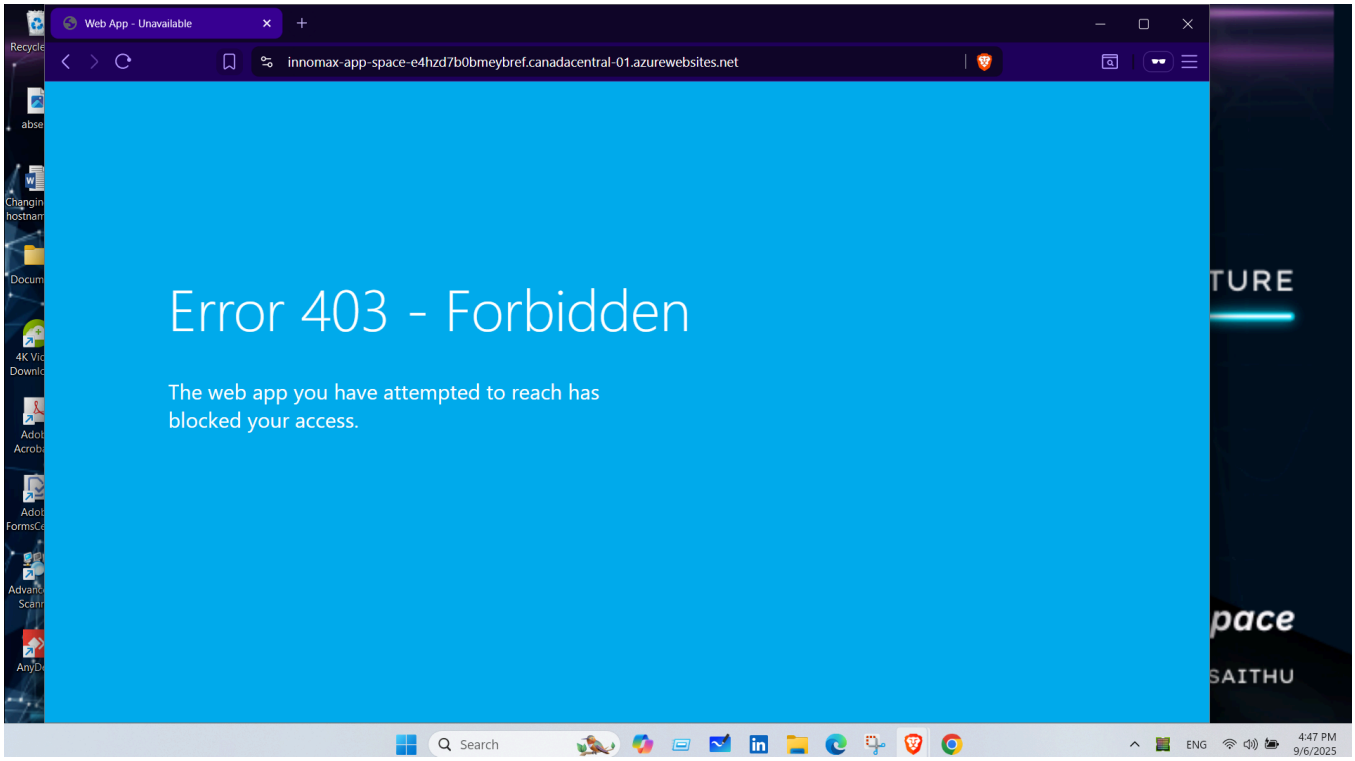


Figure 2-8

**Test 2: Private Network Access (Should SUCCEED)**

**Prerequisite:** This test requires a Virtual Machine (VM) deployed within the **webapp-vnet**.

**Action:** Connect to that VM (via Bastion or RDP/SSH) and attempt to access the same Web App URL from the VM's browser.

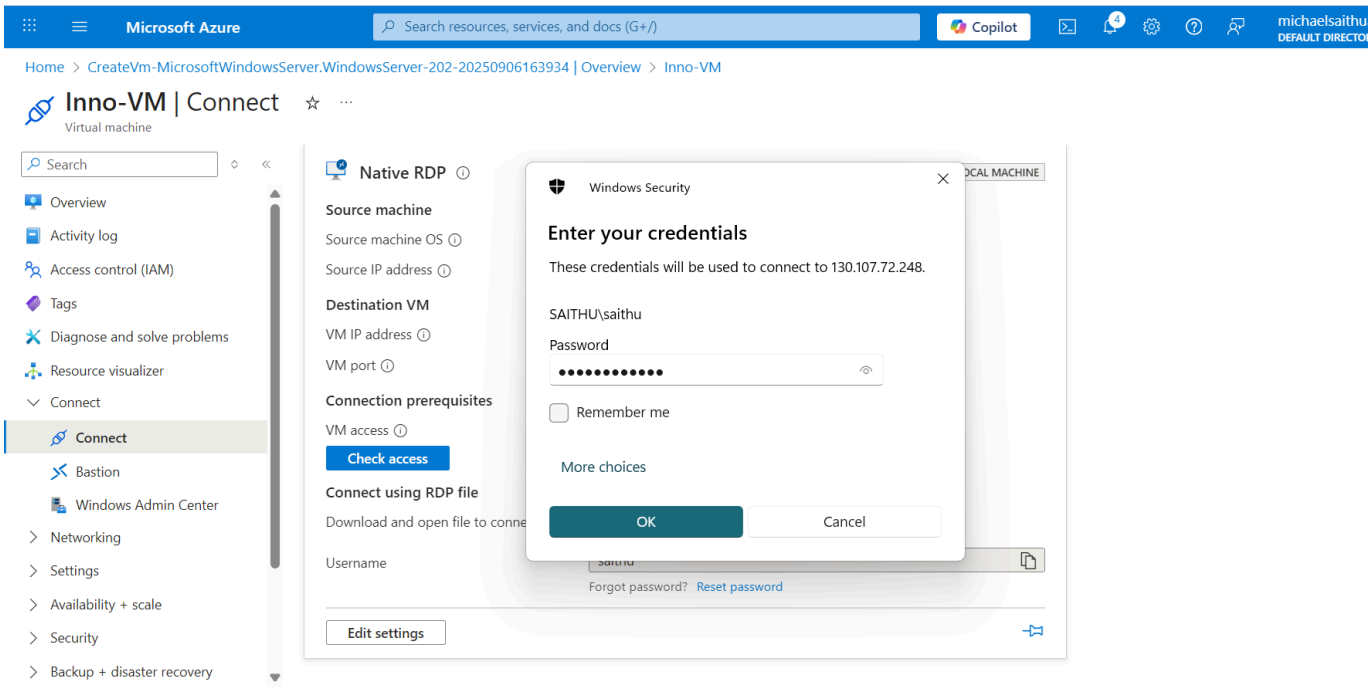



Figure 2-9

**Expected Result:** The web app will load successfully. This is because the VM is inside the VNet, and the Private Endpoint provides a private route to the app. The

private DNS zone automatically resolves the web app's name to its private IP address.

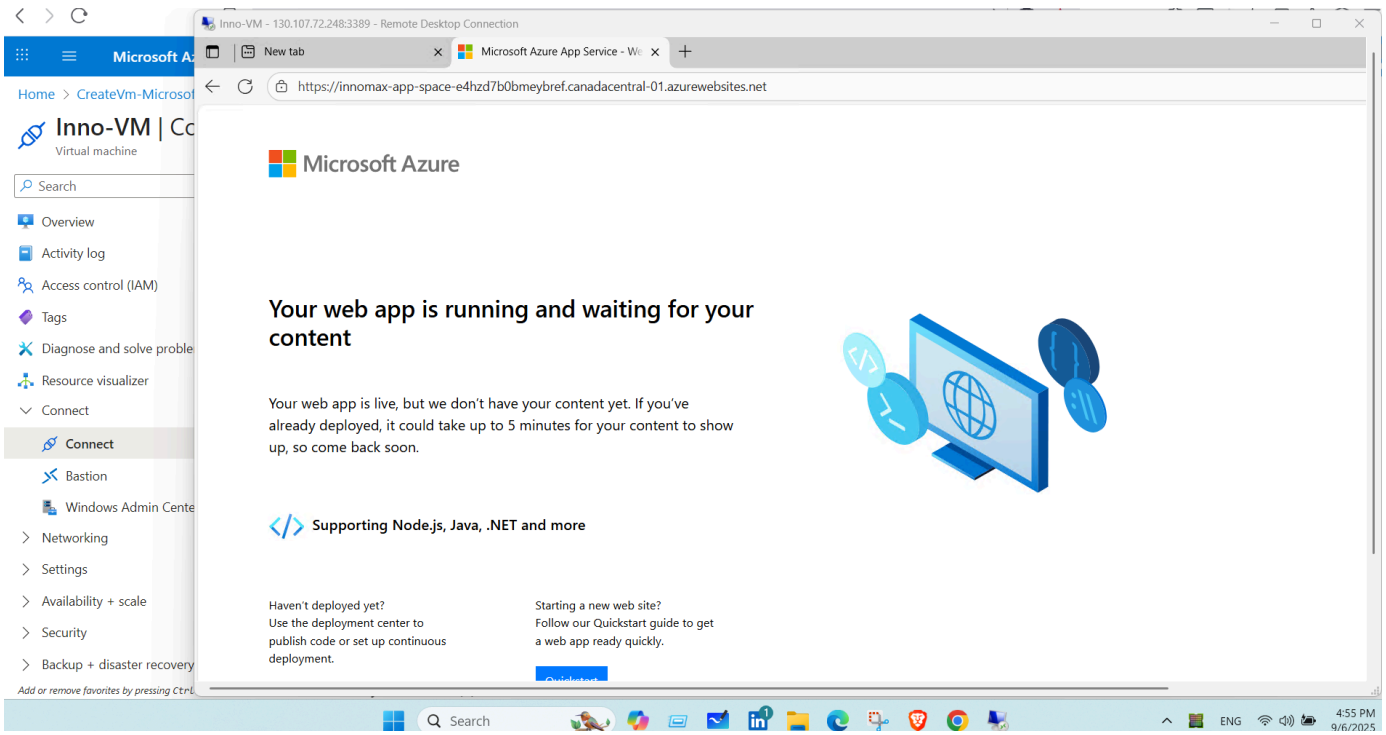



Figure 2-10

---

## 5. Conclusion

This lab successfully demonstrated a core Azure security pattern. We transformed a publicly accessible Azure Web App into a privately accessible service, shielded from the public internet.

**Key takeaways implemented:**

**Principle of Least Privilege:** The web app is now accessible only from authorized resources within the specified Virtual Network.

**Zero-Trust Network Security:** Public access was explicitly denied after establishing a secure private access method.