Create an App for Azure (part 5)

Cloud, CSS, Docker, HTML, PHP

Language Skills : 5

The digital transition has pushed software companies and enterprises to migrate their applications to the web. Azure cloud services have provided the ideal platform, the right services, and the tools to go digital while keeping full control over the data. This publication proposes the creation of a simple application by discovering some of these services and tools.

Published Wednesday June 22nd 2022, 18:14
Modified Monday August 26th 2024, 10:04

By Olivier Paudex

Security on Azure

Unlike virtual machines (VMs), services in Azure are public. That is, they are shared by multiple people. They are also called **PaaS**, for **"Platform as a Service"**, or **"tenants"**, because they are just an application instance of a global server.

> This publication is composed of several parts. This being the 5th part.

However, there are opportunities to make these services private. This publication will talk about several things:

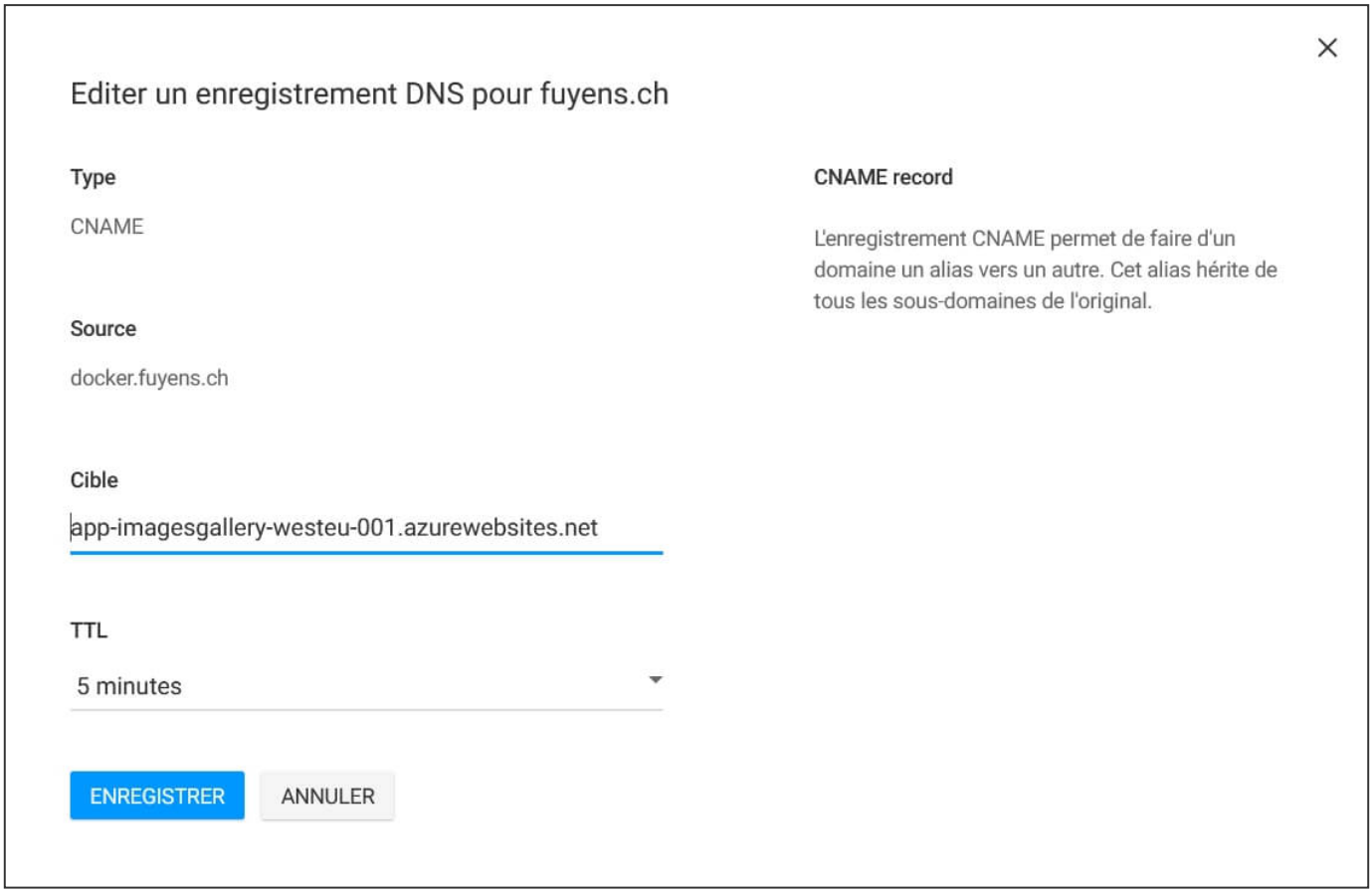
- The DNS.
- SSL certificates.
- Virtual networks.
- Private endpoints.
- The **"Application Gateway"** service.

DNS

So far, the application responds to the following URL: **"https://[app name].azurewebsites.net/"**.

This is not very practical. Most of the time, an application should respond to a custom domain name. To achieve this, nothing could be easier.

- Reserve a domain name
- Create a record in your domain of type **CNAME**
- The application, in this example, will take the name **"docker.fuyens.ch"**.
- His target is the URL given by Azure.



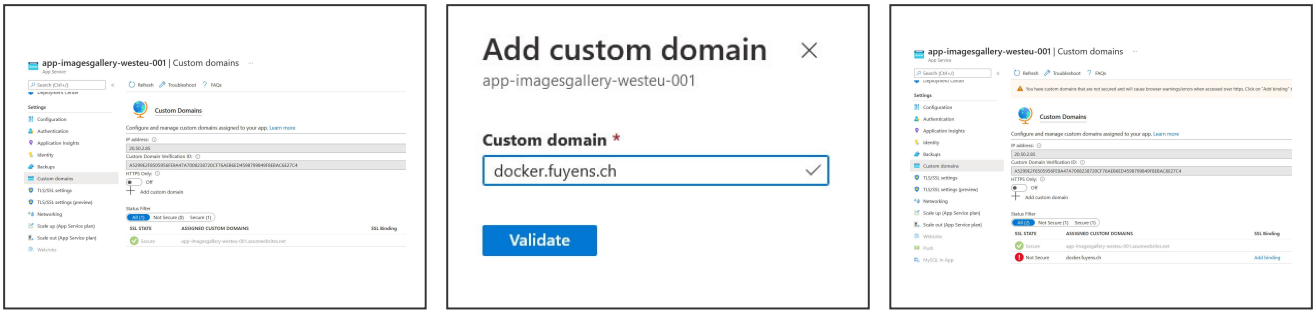
The CNAME registration

Return to the application and add your new custom domain name.

- Click on the **"Custom Domains"** tab.
- Add your custom domain name.

The name is now bound to the application, but is not yet secure. You can see the**"Not Secure"** warning in red.

You can try entering the URL, in my example: **http://docker.fuyens.ch**. This works, but the browser will warn you that the site is not secure.

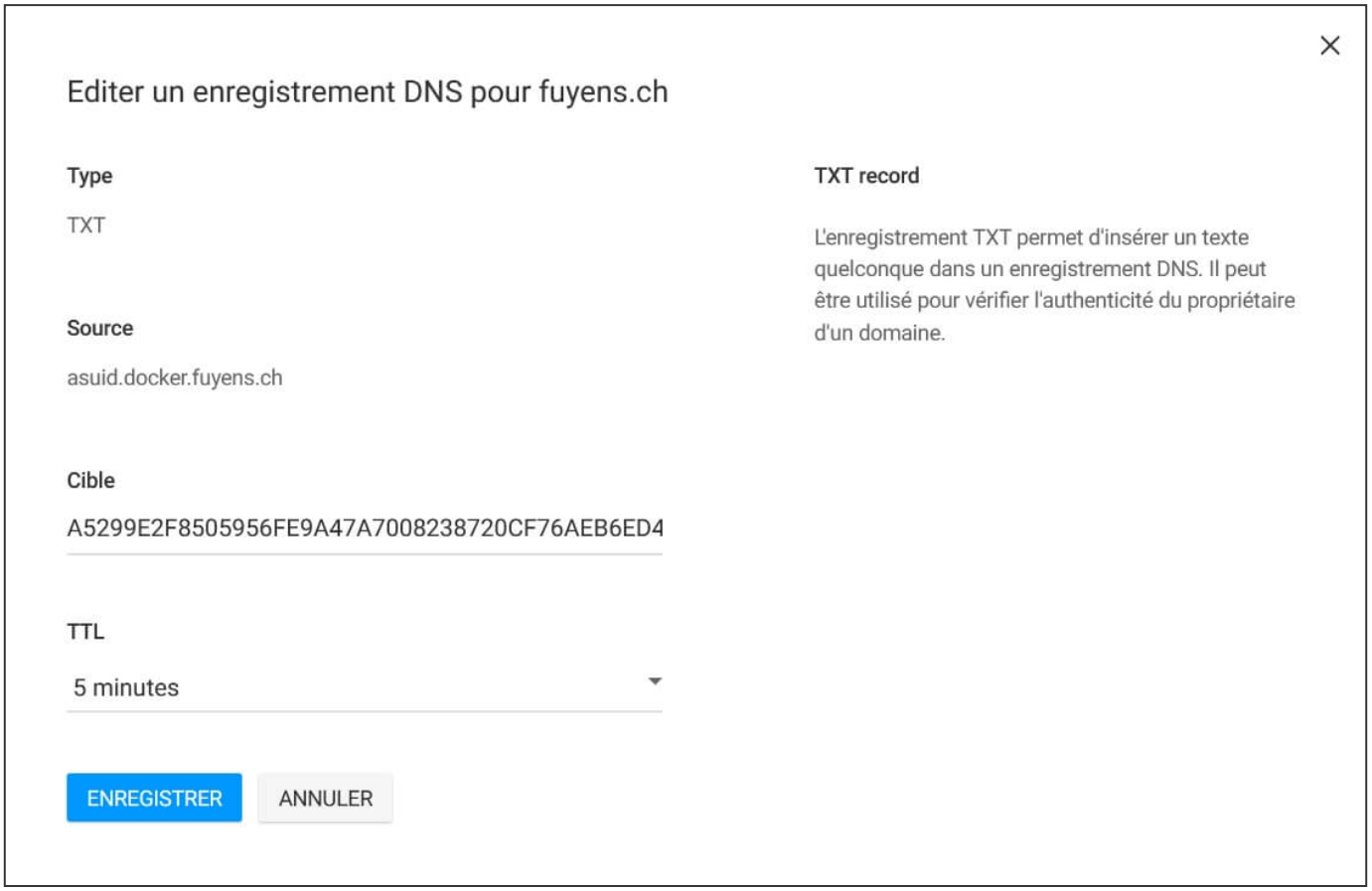


Add a custom domain name

The SSL certificate

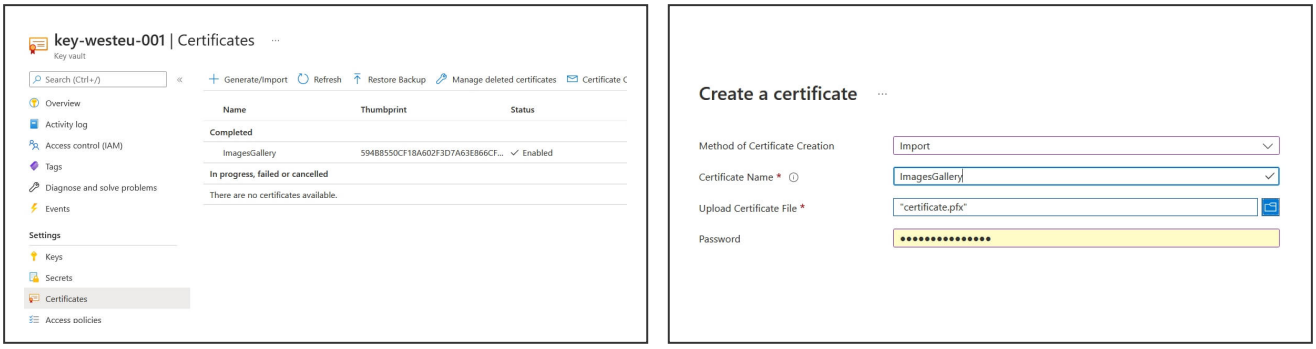
To remedy this problem, you need to subscribe to an SSL certificate.

- Copy the IP address of the application (see image above).
- Copy the verification ID of the custom domain.
- Return to your DNS.
- Create a **TXT** record and name it **"asuid.docker.fuyens.ch"**, in my example.
- Paste the verification ID into it.



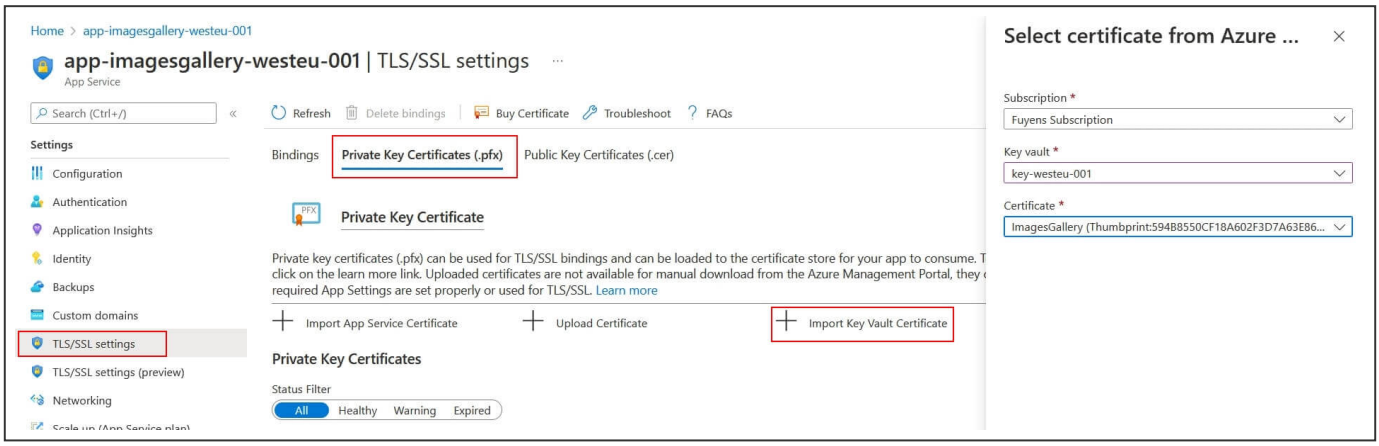
The TXT record for custom domain verification

- Create a certificate by choosing one of the many vendors on the market.
- Convert your certificate to **PFX** format, if necessary (with SSLShopper or OpenSSL, if you're more command-line savvy).
- Return to the KeyVault.
- Click on the **"Certificates"** tab.
- Click on **"Generate/Import"**.
- Select **"Import"**.
- Enter a name for the certificate.
- Select your PFX-formatted certificate.
- Enter the password you gave when creating the certificate.



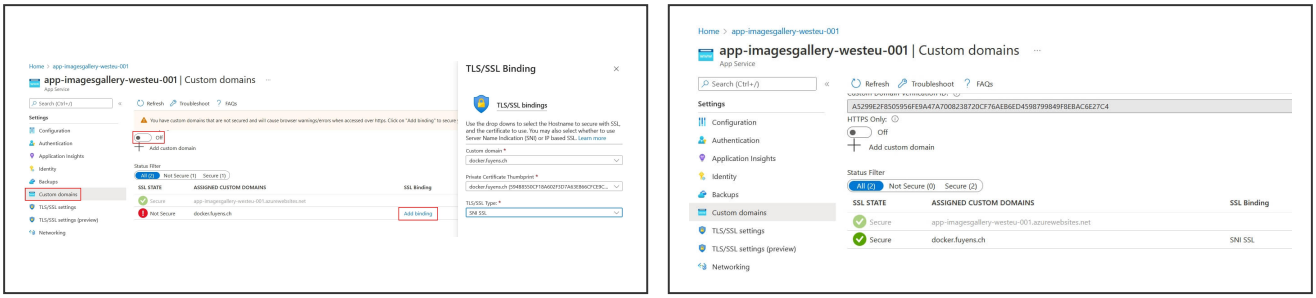
Importing the certificate into the KeyVault

- Return to the application.
- Click on the **"TLS/SSL settings"** tab.
- Click on **"Private Key Certificates (.pfx)**.
- Click on **"Import Key Vault Certificate"**.
- Select the certificate.



Importing a certificate from KeyVault

- Click on the **"Custom Domains"** tab.
- Click on **"Add binding"**.
- Select the certificate.



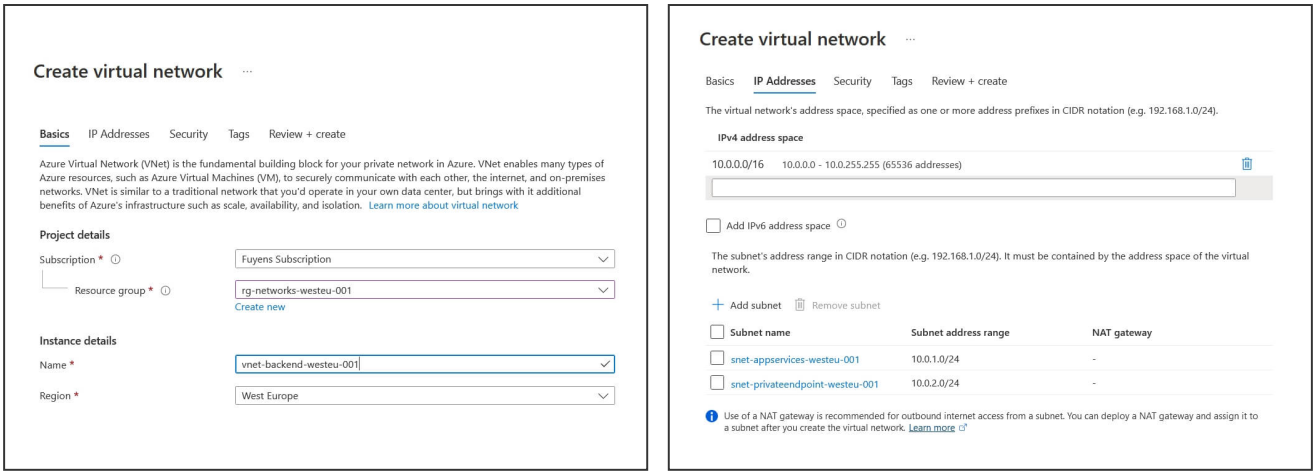
Binding the certificate to the application

This time, our application is validated by a certificate.

Virtual networks

While our application is public, the same is not true for all other services (database, storage, keyvault). We'll remedy this by creating virtual networks and privatized access.

- Create a **"Virtual network (Vnet)".**
- Select the network resource group.
- Enter a name like **"vnet-backend-westeu-001"**.
- Add the subnet **"snet-appservices-westeu-001"** and apply the IP network **"10.0.1.0/24"** to it.
- Add the subnet **"snet-privateendpoint-westeu-001"** and apply the IP network **"10.0.2.0/24"** to it.



Creating the "Backend "Vnet

- Start with another Vnet.
- Enter a name like **"vnet-frontend-westeu-001"**.
- Add the subnet **"snet-appgateway-westeu-001"** and apply the IP network **"10. 1.1.0/24"**.

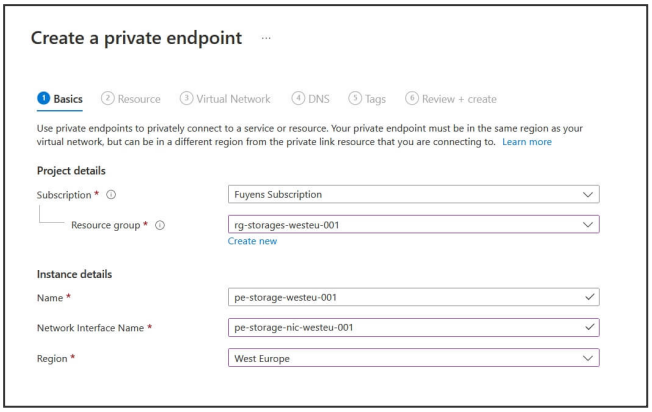
- Add the subnet **“snet-privateendpoint-westeu-001”** and apply the IP network **“10.1.2.0/24”** to it.

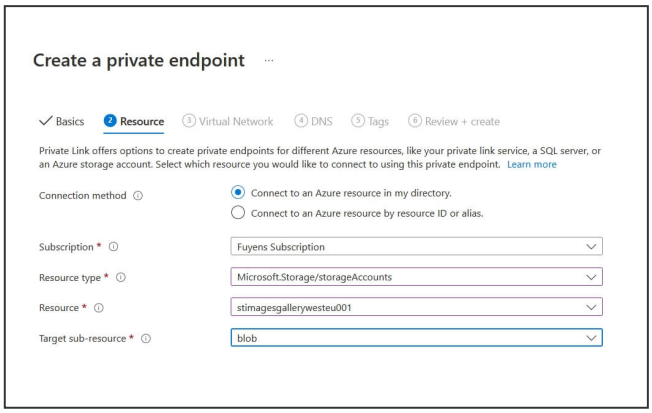
Private endpoints

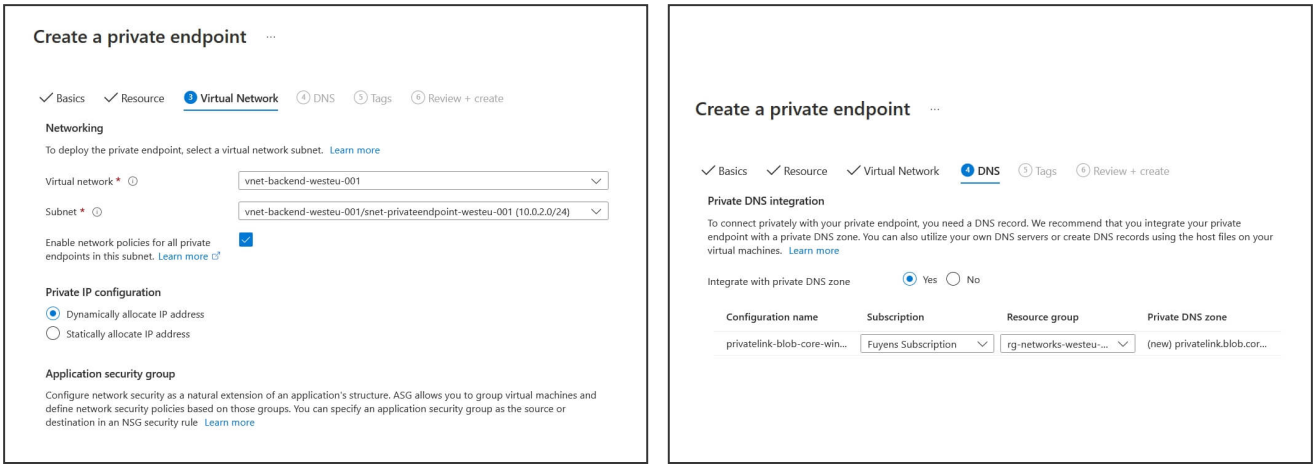
Private endpoints allow you to add a private IP address to a public service, such as storage or database.

Storage

- Add an endpoint (private endpoint).
- Select the storage resource group.
- Enter a name like **“pe-storage-westeu-001”**.
- Enter a name for the network card like **“pe-storage-nic-westeu-001”**.
- Select the resource type **“Microsoft.Storage/storageAccounts”**.
- Select **“blob”** for the sub-resource.
- Select the subnet **“Private endpoint”**, previously created.
- Select integration with a private DNS zone.







Configuring a private endpoint for the storage service

The database

- Start for the database by selecting the database resource group and resource type **"Microsoft.Sql/servers"**.
- Select the subnet **"Private endpoint"**, previously created.

The safe

The safe is doing very well!

It has two private endpoints. One for the **"backend vnet"**, so the application can retrieve secrets. The other for the **"frontend vnet"** so that the **"application gateway"** service can retrieve the SSL certificate there.

- Recommence a third time for the KeyVault.
- Select the resource type **"Microsoft.KeyVault/vaults"**.
- Select the vnet **"backend"** once.
- Recommence one last time for the Keyvault.
- Select the resource type **"Microsoft.KeyVault/vaults"**.
- Select the virtual network **"frontend"**and the subnet **"private endpoint"** once.

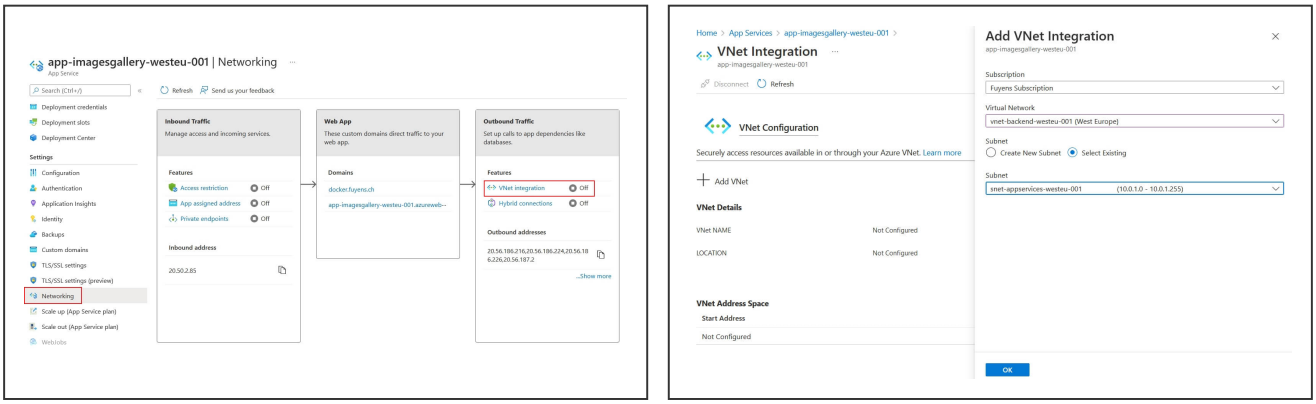
The three services are now linked to the virtual network **"Backend"**.

The vault service is also linked to the virtual network **"Frontend"**.

Integration of the application into the virtual network

To integrate an application into a virtual network, simply select it.

- Return to the application.
- Click the **"Networking"** tab.
- Click **"Vnet Integration"**.
- Click **"Add Vnet"**.
- Select the vnet **"backend"**.
- Select the subnet **"appservices"**.

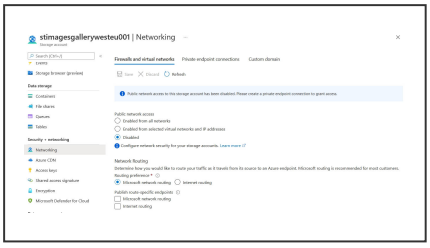
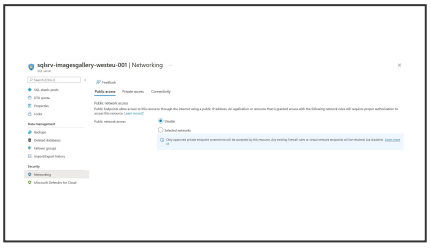
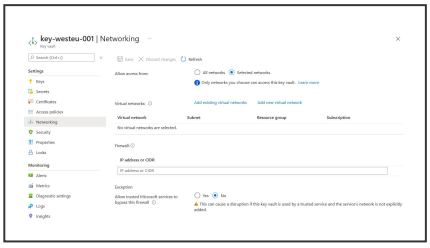


Integrating an application into a virtual network

Disabling access rights to different services

To secure the infrastructure, go around the three services and disable access to the public network.

- In the storage account, click **"Networking"**, then select **"Disable public network access"**.
- Idem for the database server.
- In the vault, there is no disabling possible, so select **"Selected networks"** and in the exceptions, select **"No"** for **"Allow trusted Microsoft services to bypass this firewall"**.

Disabling public network access

Verifying access

Verify that the application is still working

- Try to connect to the database using **Azure Data Studio**, there should be a connection error.
- Idem for storage with **Azure Storage Explorer**.

The Application Gateway

The Application Gateway allows you to create, as the name implies, a single gateway at the entrance to the infrastructure. It filters authorized protocols, and redirects traffic to one or more applications or services. Among other things, it allows:

- Add a single public IP address as input to our infrastructure.
- Add security by creating listening rules on a particular protocol.
- Add a firewall option to our infrastructure.
- Balance loads across multiple resources, redirecting traffic to a resource based on the URL.

> Security resources are among the most expensive elements of the Azure cloud. A front-end gateway like the one used for this demo app can cost about **$6 per day**. So be careful not to leave it running longer than necessary.

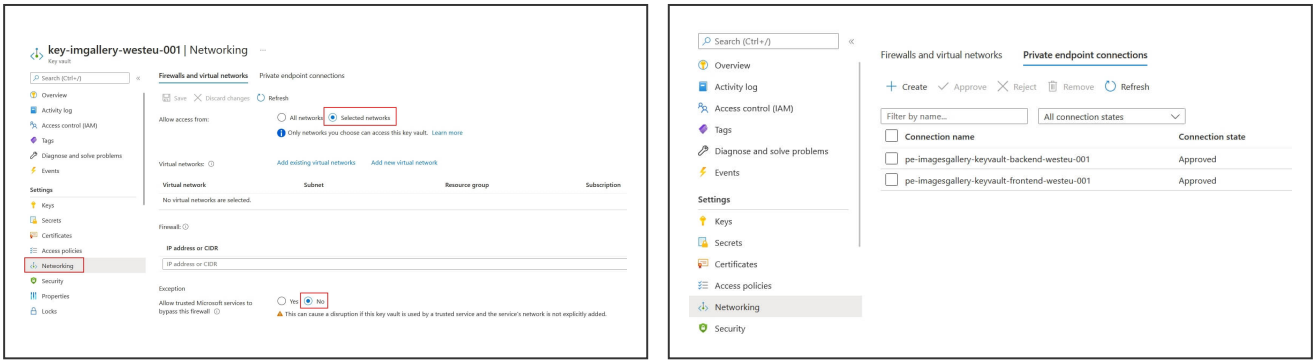
Azure Managed Identity (AMI)

The gateway application will use the SSL certificate in place of the application. To access the **"Keyvault"** containing the certificate, it must register with the **"Keyvault"** service. Unlike the **"Apps service"** which uses a **"System assigned"** identity, the **"Application gateway"** service will have to use a **"User assigned"** identity. To accomplish this, it must go through an **"Azure Managed Identity"** or AMI.

- Create an Azure User Managed Identity.
- Copy its ID from the **"Overview"** page.
- Go back to the Keyvault.
- Add an access rule.
- Add the **"GET"** right for the **"Secret permissions"**.
- Click on **"Select principal"**.
- Paste in the AMI ID.
- Add the keyvault access for the AMI.

Accessing the Keyvault from the Application Gateway

- Click on the **"Networking"** tab of the Keyvault.
- Select "Selected networks".
- Do not add any virtual networks.
- Select **"No"** for the exception **"Allow trusted Microsoft services to bypass this firewall"**.
- Be sure to verify that you have configured two private endpoints.



Keyvault configuration with two private endpoints

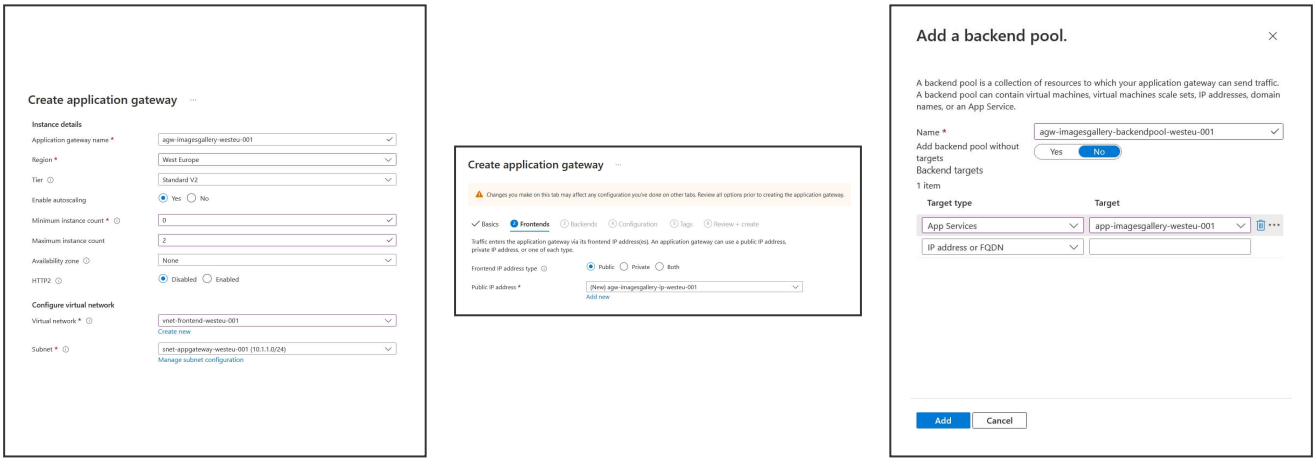
Application Gateway

Basics tab

- Create an Application Gateway.
- Give it a name.
- Leave the minimum number of instances at 0.
- Add the maximum number of instances at 2.
- For the virtual network configuration, add the virtual network **"frontend"** and the subnet **"app gateway"**.

"Frontends and Backends" tab

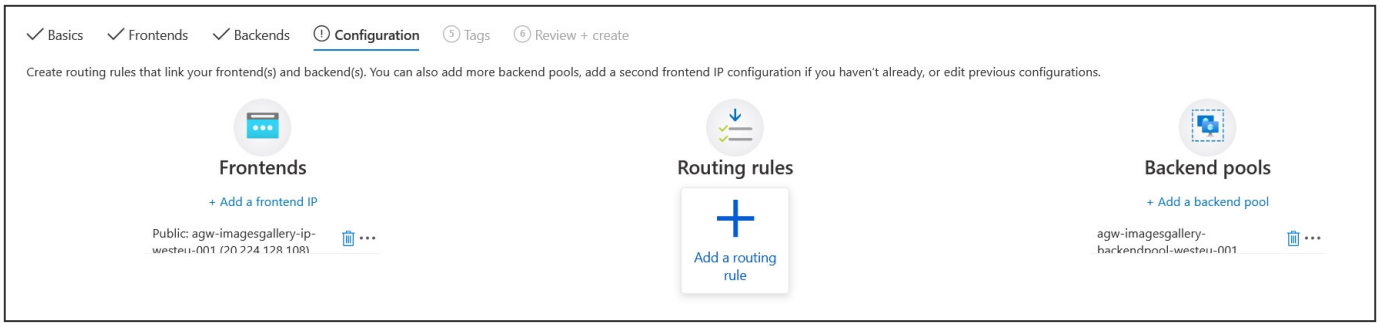
- For the **"Frontends"** tab, add a new public IP address.
- For the **"Backends"** tab, add a backend pool and associate the application with it as the target.



The creation of the "Frontend" and "Backend"

Configuration tab

Once the **"Frontend"** and the **"Backend"** are added, the Application Gateway needs to connect them through routing rules that will filter the allowed protocols. This is called a**"listener"**.



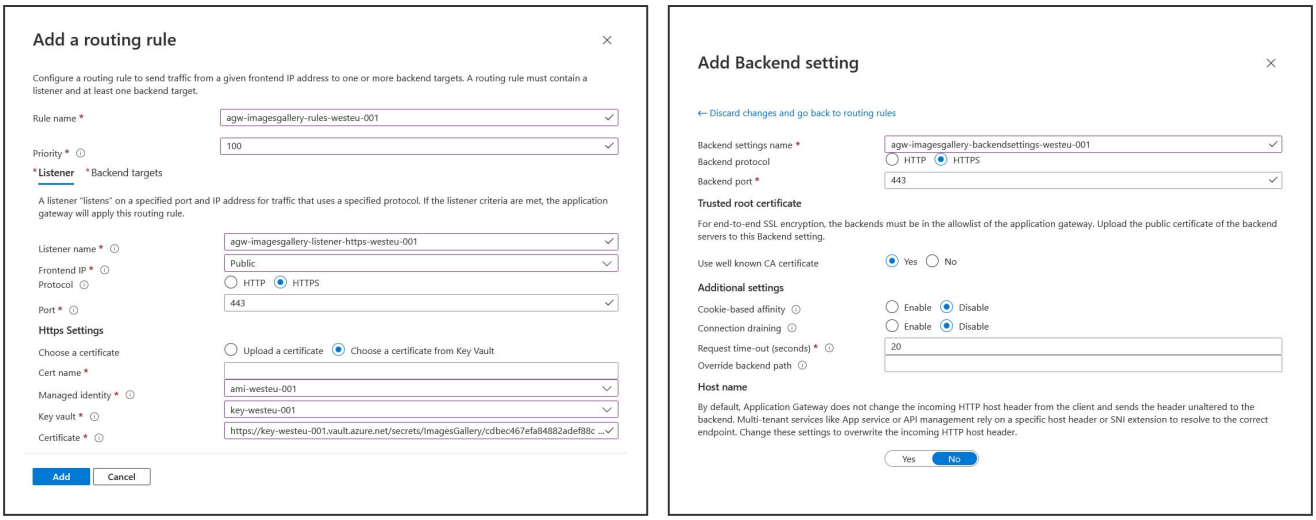
Routing rules

- Click **"Add a routing rule"**.
- Name it.
- Give it a priority of 100.

- Add a **"Listener"** and associate HTTPS protocol and port 443 with it.
- Select **"Choose a certificate from Key Vault"**.
- Select the AMI and Keyvault.
- Paste the certificate's secret ID URL.

> Microsoft recommends that in this regard, only the path to the certificate should be given, without its ID. This makes it easier to renew the certificate.

- Create a new **"Backend setting"**.
- Name it.
- Select HTTPS protocol and port 443.
- Select **"Yes"** the **"Use well known CA certificate"**.



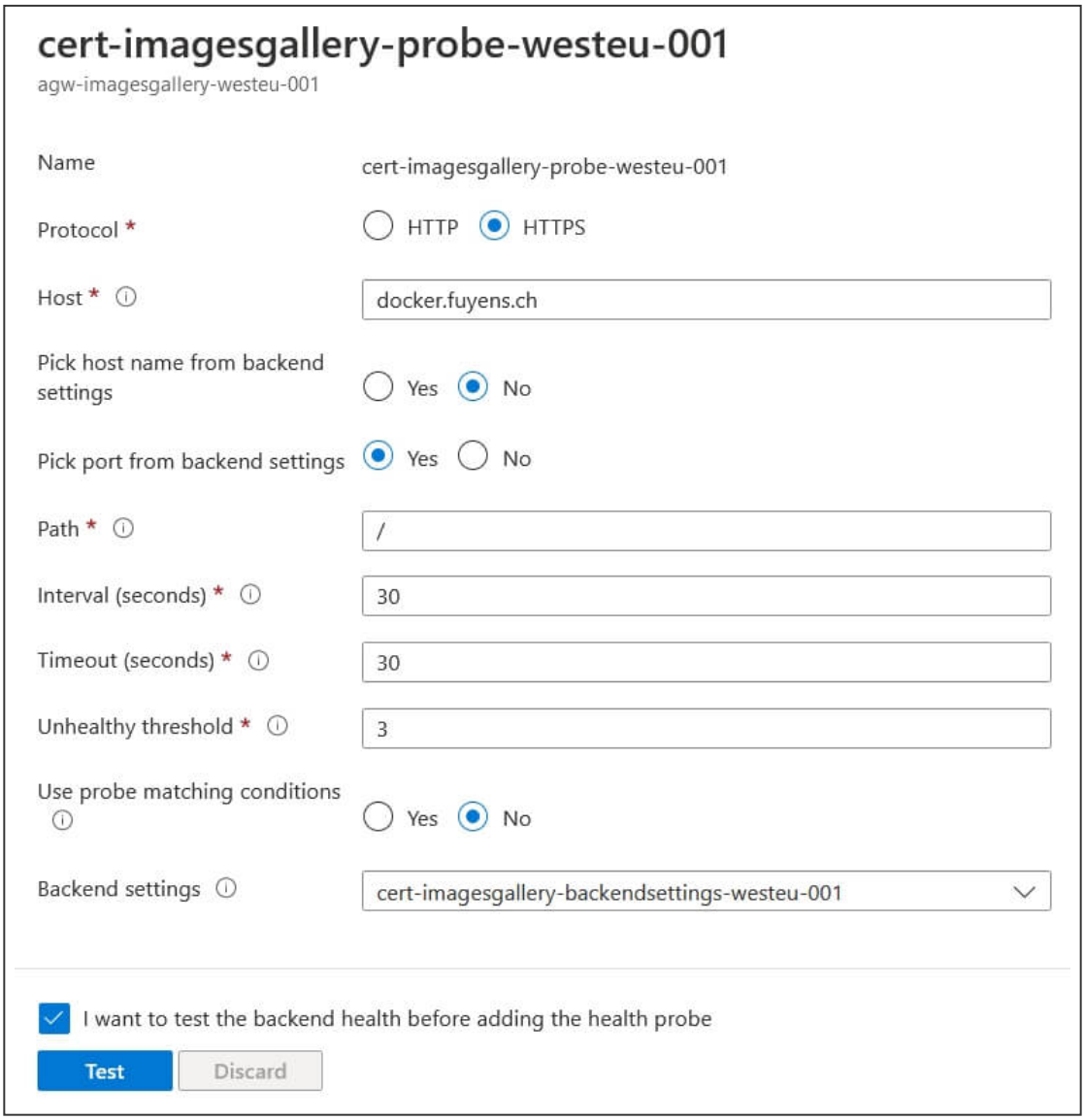
The routing rule and its parameters

The health probe

The health probe is used to monitor and test that the gateway application is working properly.

- Create a health probe (**"probe-health"**).
- Name it.
- **Very important!** Give the **"host"**, the name of your application on the certificate. In my case, this one is **"docker.fuyens.ch"**.

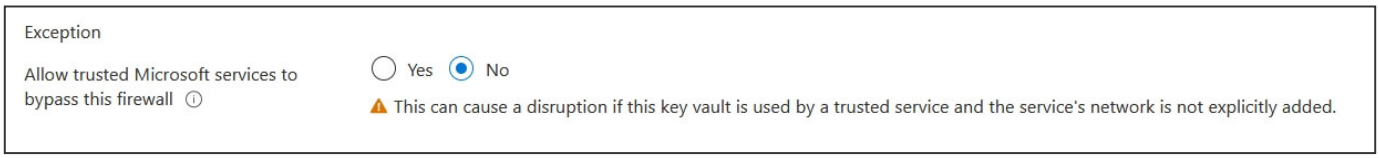
- Select **"No"** for the **"Pick host name from backend settings"**.
- Select **"Yes"** for the **"Pick port from backend settings"** option.
- Enter **"/"** for the **("path")**.
- Select the backend settings.
- Leave the other options as default.
- Check **"I want to test the backend health before adding the health probe"**.
- Click **"Test"**.



Health Probe Creation

The result should be a green check mark.

> If not, check your certificate name, which should match the probe. If it still doesn't, go back to the Keyvault and temporarily allow Microsoft services to access it.



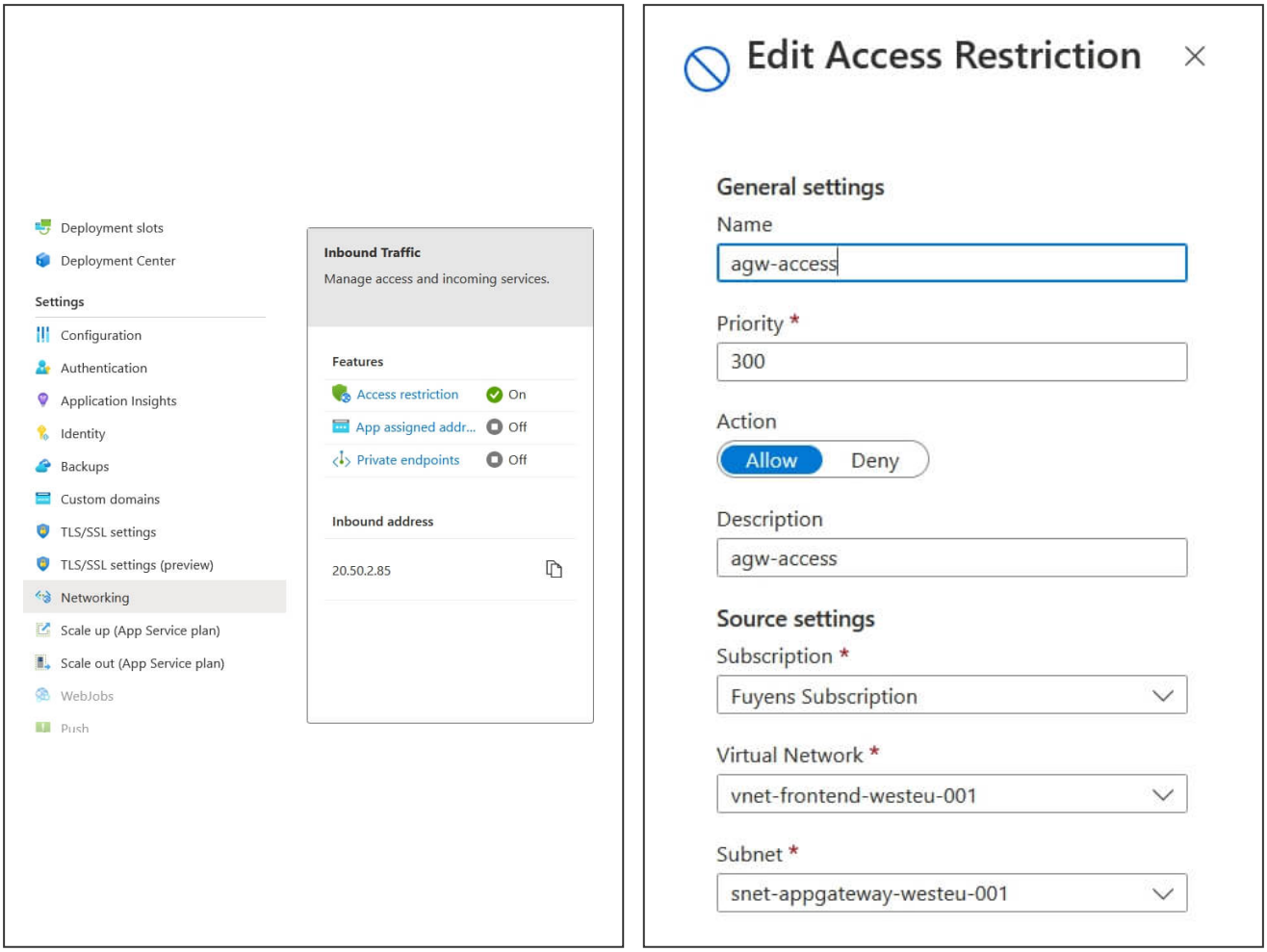
Security exception should always be on "No" when using a private endpoint

- Test the probe one last time by clicking on **"backend health"** and make sure you have the green check mark.

Restrict access to the application.

Now that the infrastructure has an **"application gateway"**, we can close public access to the application.

- Return to **"App services"**.
- Click **"Networking"**.
- Click **"Access restriction"**.
- Add a rule that only allows the application gateway subnet to access the application.



Application-level access restriction

Conclusion

Here it is, this long publication is now complete. It validates all of my research and various tests I undertook to build a secure infrastructure around an application that uses **"PaaS"** services on Microsoft's Azure cloud. I also covered the use of containers with Docker and the use of SSL certificates.