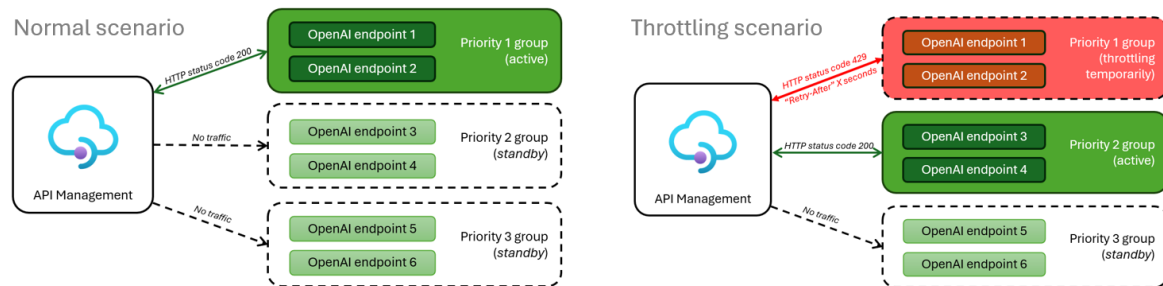


Smart Load-Balancing for Azure OpenAI with Azure API Management



APIM Load Balancing Architecture

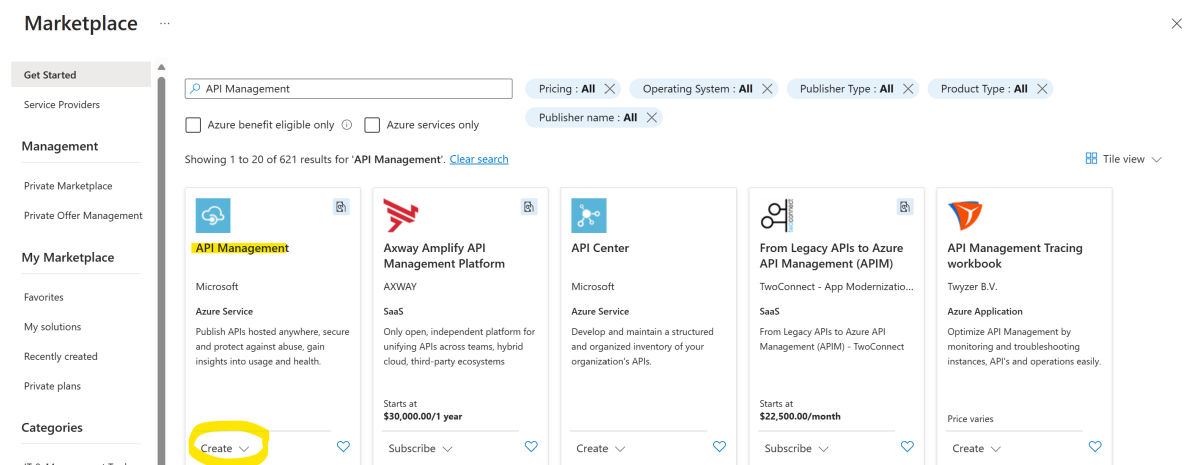
The groups of resources can contain one or more Azure OpenAI endpoints, with the key being to assign a priority to each group. This approach allows us to first utilize committed capacity, subsequently transitioning to Pay-as-you-go resources if needed. Alternatively, it offers the flexibility to prioritize resources within the region before doing cross-region API calls. In the event of all resources being exhausted, we will revert back to the initial resource and relay its response to the API caller.

Tutorial

Provision Azure API Management

Firstly, we want to provision an [Azure API Management instance](#) and ensure that we enable `Managed Identity` during provisioning. It'll make most sense to put our APIM instance in the same region where our primary Azure OpenAI instance lives in.

To get started, use the Azure Portal to provision an API Management instance:



Provision APIM instance

For the [pricing tier](#), use the tier that suits our needs, I've tested with `Developer` and `Basic` tiers.


Home > [redacted] > Marketplace > API Management >


Create API Management service

API Management service


Basics Monitoring Scale Managed identity Virtual network Protocol settings Tags Review + install


Project details
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.


Subscription * ⓘ [redacted] 


Resource group * ⓘ [redacted] 
[Create new](#)

Instance details


Region * ⓘ (Europe) Sweden Central 


Resource name * apim-demo-blog 

Organization name * ⓘ Clemens 

Administrator email * ⓘ [redacted] 

Pricing tier
API Management pricing tiers vary in computing capacity per unit and the offered feature set - for example, support for virtual networks, multi-regional deployments, or self-hosted gateways. To accommodate more API requests, consider adding API Management service units instead. [Learn more](#)

 The Developer tier of API Management does not include SLA and should not be used for production purposes. Your service may experience intermittent outages, for example during upgrades. [Learn more](#)

Pricing tier ⓘ Developer (no SLA) 
[See all pricing tiers](#)

Create your APIM instance

Lastly, we shouldn't forget to enable Managed Identity during provisioning:


Create API Management service

API Management service

Basics Monitoring Scale **Managed identity** Virtual network Protocol settings Tags Review + install

A system assigned managed identity enables Azure resources to authenticate to cloud services (e.g., Azure Key Vault) without storing credentials in code. Once enabled, all necessary permissions can be granted via Azure role-based access control. The lifecycle of this type of managed identity is tied to the lifecycle of this resource. Additionally, each resource (e.g., Virtual Machine) can only have one system assigned managed identity. [Learn more](#)

System assigned managed identity
Enable system assigned identity to grant the resource access to other existing resources.

Status 

Enable Managed Identity during provisioning

Provision Azure OpenAI and assign Managed Identity

Next, ensure that your Azure OpenAI resources are prepared. Since we will be directing API calls through APIM to multiple resources, it is essential to confirm that each Azure OpenAI resource has the same models (type and version) deployed, and, crucially, with identical names. For instance:

- Azure OpenAI resource 1 -> Model deployment with name `gpt-4-8k-0613` (model type: `gpt-4-8k`, version: `0613`)
- Azure OpenAI resource 2 -> Model deployment with name `gpt-4-8k-0613` (model type: `gpt-4-8k`, version: `0613`)

I recommend adopting a uniform naming scheme for each deployment by encoding the model type and version in its name (e.g., `gpt-4-turbo-1106`).

To complete this step, we must provide API Management with access to our Azure OpenAI resources. However, for security considerations, let's refrain from using API keys. Instead, we should add the **Managed Identity** of the APIM to each Azure OpenAI resource, granting it the **Cognitive Services openAI User** permission. We may perform this step while the APIM instance is still in the provisioning stage.

So for each Azure OpenAI Service instance, we need to add the Managed Identity of the API Management. For this, goto each Azure OpenAI instance in the Azure Portal, click **Access control (IAM)**, click **+ Add**, click **Add role assignment**, select the role **Cognitive Services openAI User**:

Add role assignment

RoleMembersReview + assign

A role definition is a collection of permissions. You can use the built-in roles or you can create your own custom roles. [Learn more](#)

Assignment type

Job function rolesPrivileged administrator roles

Grant access to Azure resources based on job function, such as the ability to create virtual machines.

openai

Type: AllCategory: All

Name	Description	Type	Category	Details
Azure AI Developer	Can perform all actions within an Azure AI resource besides managing the resource itself.	BuiltInRole	None	View
Cognitive Services Data Reader (Preview)	Lets you read Cognitive Services data.	BuiltInRole	Preview	View
Cognitive Services OpenAI Contributor	Full access including the ability to fine-tune, deploy and generate text	BuiltInRole	None	View
Cognitive Services OpenAI User	Ability to view files, models, deployments. Readers are able to call inference operations such as chat completio...	BuiltInRole	None	View
Cognitive Services User	Lets you read and list keys of Cognitive Services.	BuiltInRole	AI + Machine Learning	View

Showing 1 - 5 of 5 results.

Select Cognitive Services OpenAI User role

Click Next, select **Managed Identity** under **Assign access to**, then **+ select Members**, and select the Managed Identity of your API Management instance.

Home > azure-openai-latency-testing > aoai-latency-testing-sweden | Access control (IAM)

Add role assignment

RoleMembersReview + assign

Selected role

Cognitive Services OpenAI User

Assign access to

User, group, or service principal

Managed identity

Members

+ Select members

Name	Object ID	Type
No members selected		

Description

Optional

Review + assign

Previous

Next

Select managed identities

Some results might be hidden due to your ABAC condition.

Subscription

Managed identity

API Management service (3)

Select

Search by name

Selected members:

apim-demo-blog

Remove

Select

Close

Feedback

Select your APIM Managed Identity

Import API schema into API Management

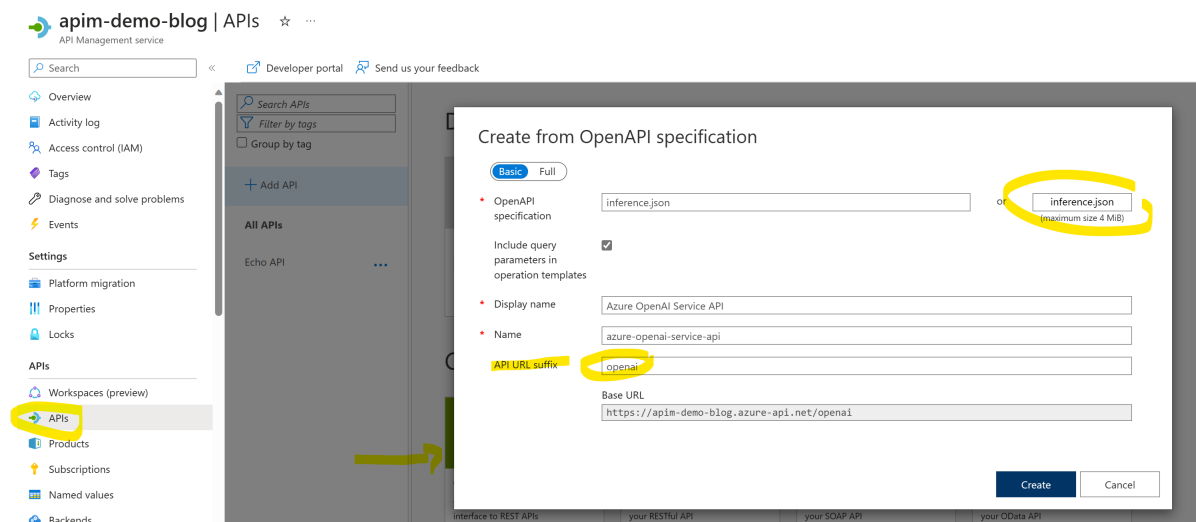
Next, we need to add Azure OpenAI's API schema into our APIM instance. For this, download the desired API schema for Azure OpenAI Service for the [schema repo](#). In this engagement, we'll be using version [2024-03-01-preview](#).

Once downloaded, open `inference.json` in the editor of your choice and update the `servers` section:

```
"servers": [  
  {  
    "url": "https://{AOAI Endpoint URL}.openai.azure.com/openai"  
  }  
],
```

We won't use this, but in order to import the file into API Management, we need to a correct URL there.

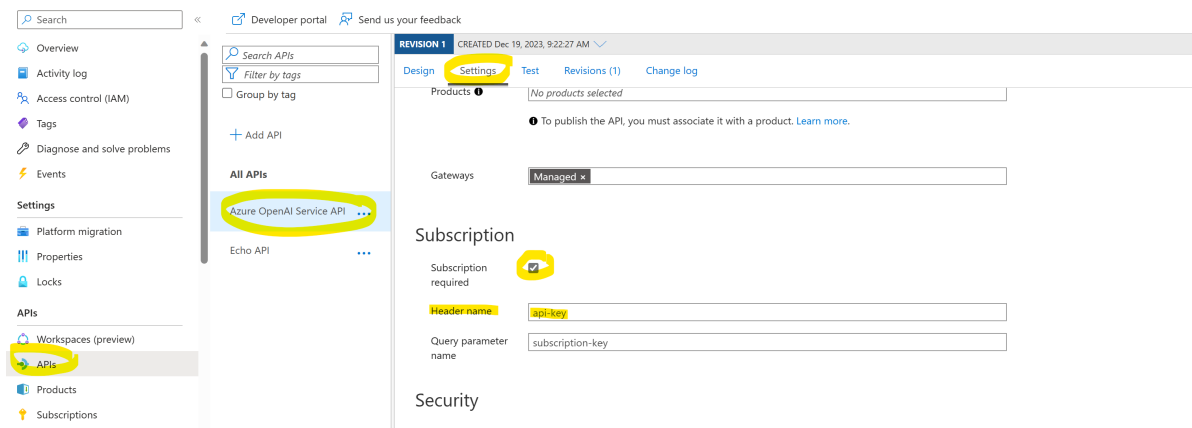
Now, let's go to our API Management instance in the Azure Portal, then select `API` on the left side, click `+ Add API` and select `openAPI`. In the dialog, load our `inference.json` and make sure to set `API URL suffix` to `openai`. Then click `Create`.



Create API definition

Configure API Management

Next, let's fully configure API Management. So select the new API, goto `Settings`, then go to `Subscription` and ensure `subscription required` is checked and `Header name` is set to `api-key`. This is important to ensure compatibility with the OpenAI SDK.



Set Header name to api-key

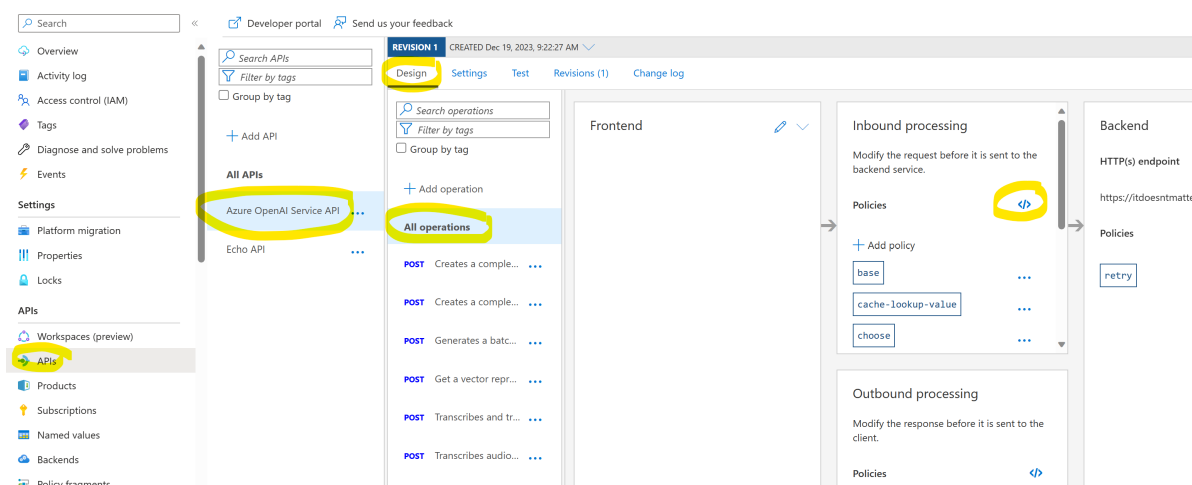
Also validate that `API URL suffix` is set to `openai`:

Now, download the `apim-policy.xml` and edit the backends section as needed:

```
backends.Add(new JObject()
{
    { "url", "https://openai-eastus.openai.azure.com/" },
    { "priority", 1 },
    { "isThrottling", false },
    { "retryAfter", DateTime.MinValue }
});
...
```

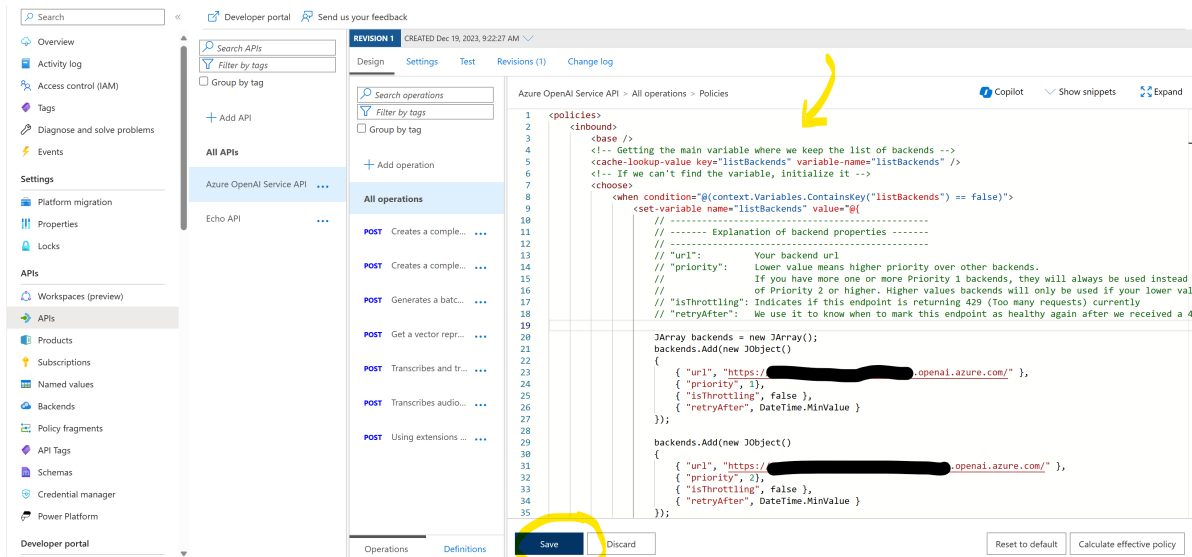
Make sure you add all the Azure OpenAI instances you want to use and assign them the desired priority.

To put this policy into effect, go back to API Management, select `Design`, select `All operations` and click the `</>` icon in inbound processing.



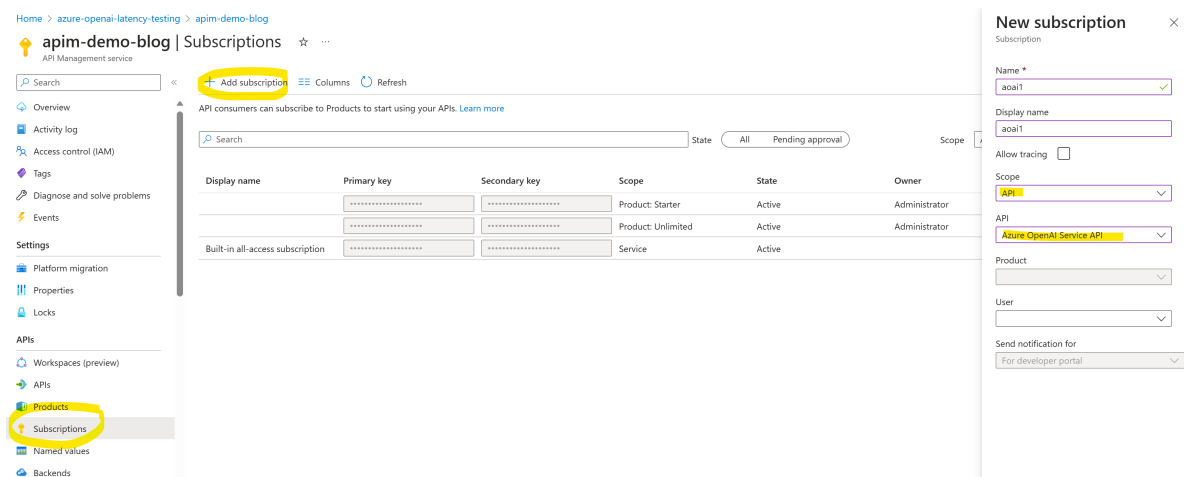
Select inbound processing

Replace the code with the contents of your `apim-policy.xml`, then hit `Save`:



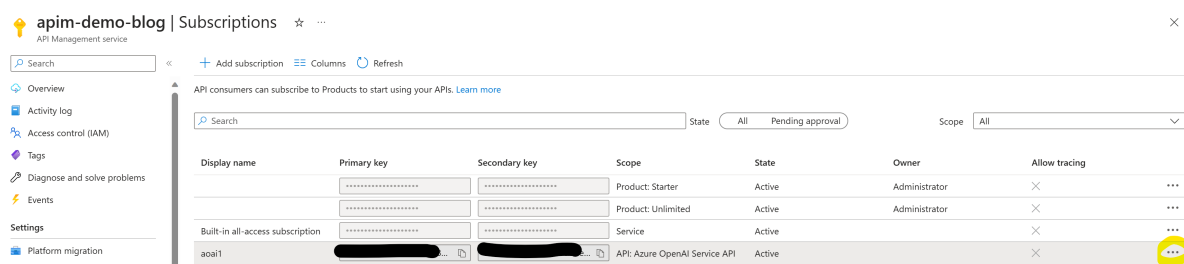
Select APIM policies

Lastly, goto **Subscriptions** in API Management, select **+ Add subscription**, give it a name and scope it **API** and select your **Azure OpenAI Service API**, click **Create**.



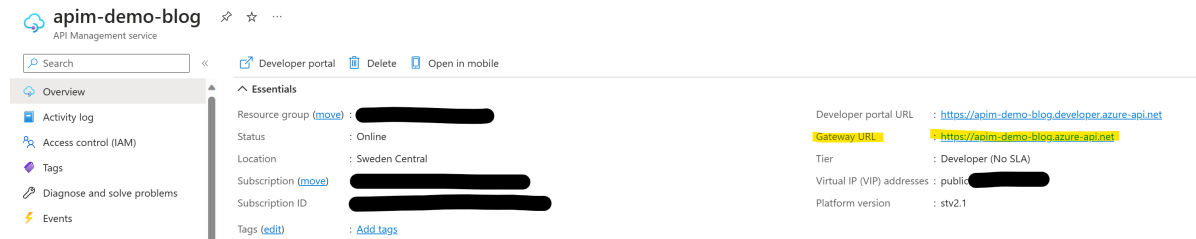
Create new subscription keys in APIM

Then get the primary subscription key via the **...** on the right side, we need this for the next step:



Get new subscription keys

Lastly, we also need the endpoint URL, which we can find on the overview page:



Get the APIM endpoint

Test it

Finally, we can test if everything works by running some code of your choice, e.g., this code with OpenAI Python SDK (v1.x):

```
from openai import AzureOpenAI

client = AzureOpenAI(
    azure_endpoint="https://<your APIM endpoint>.azure-api.net/",
    api_key="<your APIM subscription key>",
    api_version="2024-02-15-preview"
)

message_text = [{"role": "system", "content": "You are an AI assistant that helps people find information."}]

completion = client.chat.completions.create(
    model="gpt-4", # model = "deployment_name"
    messages = message_text,
    temperature=0.7,
    max_tokens=800,
    top_p=0.95,
    frequency_penalty=0,
    presence_penalty=0,
    stop=None
)

print(completion)
```

Our response looks good:

```
ChatCompletion(id='chatcmpl-9BjsSWOWXmsidRRr1jP306G7L2gam', choices=[Choice(finish_reason='stop', index=0, logprobs=None, message=ChatCompletionMessage(content='Great! How can I assist you today?', role='assistant', function_call=None, tool_calls=None), content_filter_results={'hate': {'filtered': False, 'severity': 'safe'}, 'self_harm': {'filtered': False, 'severity': 'safe'}, 'sexual': {'filtered': False, 'severity': 'safe'}, 'violence': {'filtered': False, 'severity': 'safe'}}]), created=1712584988, model='gpt-4', object='chat.completion', system_fingerprint='fp_1402c60a5a', usage=CompletionUsage(completion_tokens=9, prompt_tokens=18, total_tokens=27), prompt_filter_results=[{'prompt_index': 0, 'content_filter_results': {'hate': {'filtered': False, 'severity': 'safe'}, 'self_harm': {'filtered': False, 'severity': 'safe'}, 'sexual': {'filtered': False, 'severity': 'safe'}, 'violence': {'filtered': False, 'severity': 'safe'}}}])
```