Private GPT LLMs
Azure OpenAI Service sicher deployen mit Terraform

27.02.2024

Kenny Pflug
@feO2x
Consultant @Thinktecture

Kenny Pflug

Consultant Software Architect @ Thinktecture AG

- Distributed Systems with ASP.NET Core
- .NET internals
- Cloud-native

kenny.pflug@thinktecture.com @feO2x https://www.thinktecture.com

Thanks, Thorsten Hans 🙋

Today's menu

Main Course

- What is Azure OpenAI service?
- How can I integrate it with Azure private networking?
- How can I integrate it into apps available via the internet? (with Terraform Demo)
- How can I integrate it into my on-premise apps?

On the side

- How can I stream responses from Azure OpenAI Service to my frontend?

What is Azure OpenAI Service?

- Platform as a Service (PaaS) offer from Microsoft Azure
- Run and interact one or more Large Language Models (LLMs) in one service instance
- The underlying Cloud infrastructure is shared with other Azure customers
- Built on top of Azure Resource Manager (ARM) and can be automated by Terraform, Pulumi, or Bicep
- Currently, you need to apply so that your Azure Subscription can use OpenAI Service

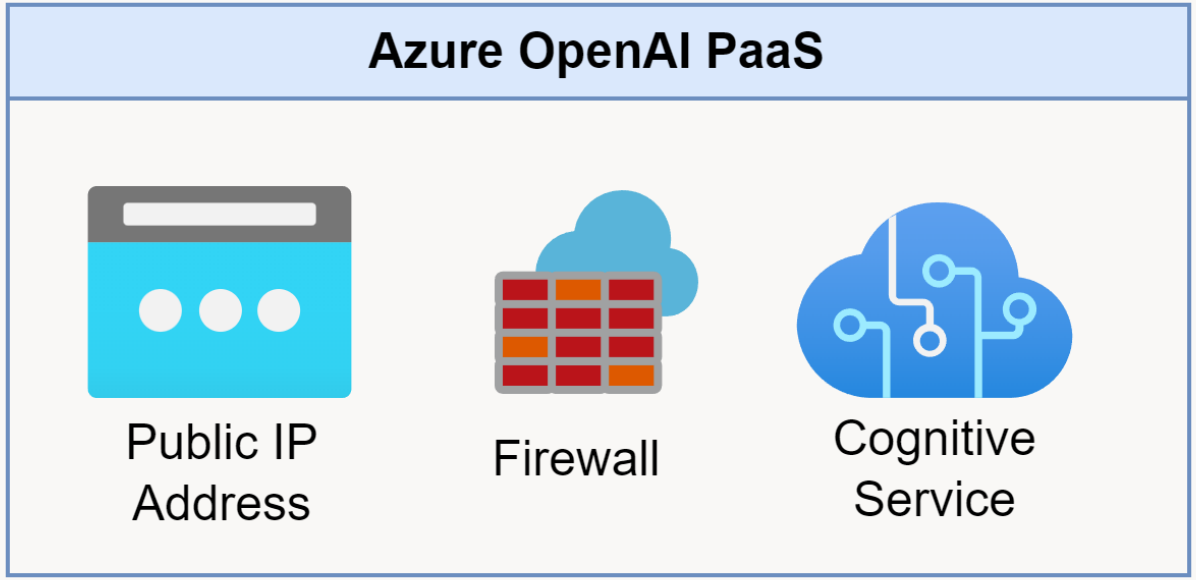
Demo Time

A quick look at the final product

OpenAI Service PaaS – The Defaults (1)

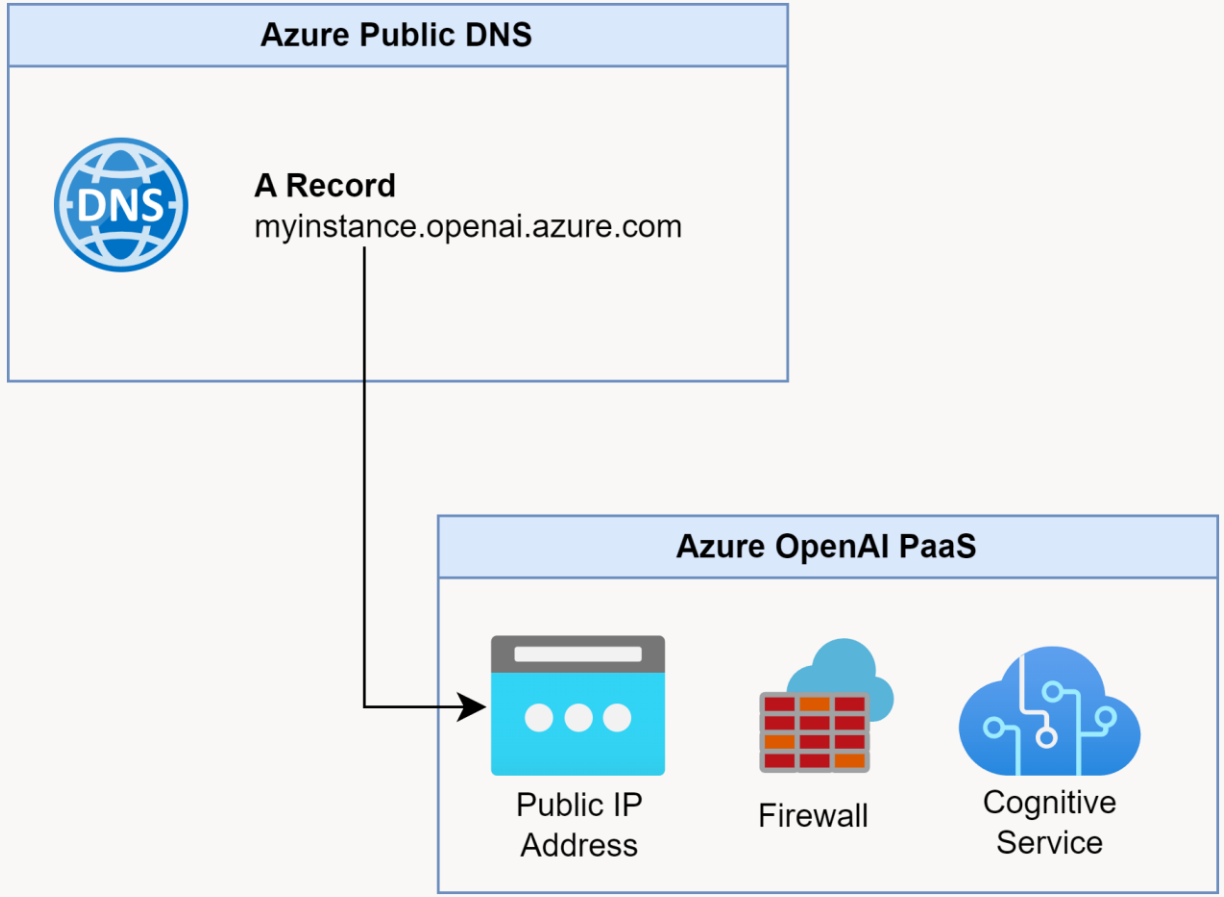
When creating an Azure OpenAI Service resource, you get the following:

- A cognitive service instance, with the type of “OpenAI”
- A firewall for traffic management, as with almost every Azure PaaS service
- A public IP address which is routable via the internet
- Access to the service instance is authenticated via an API key



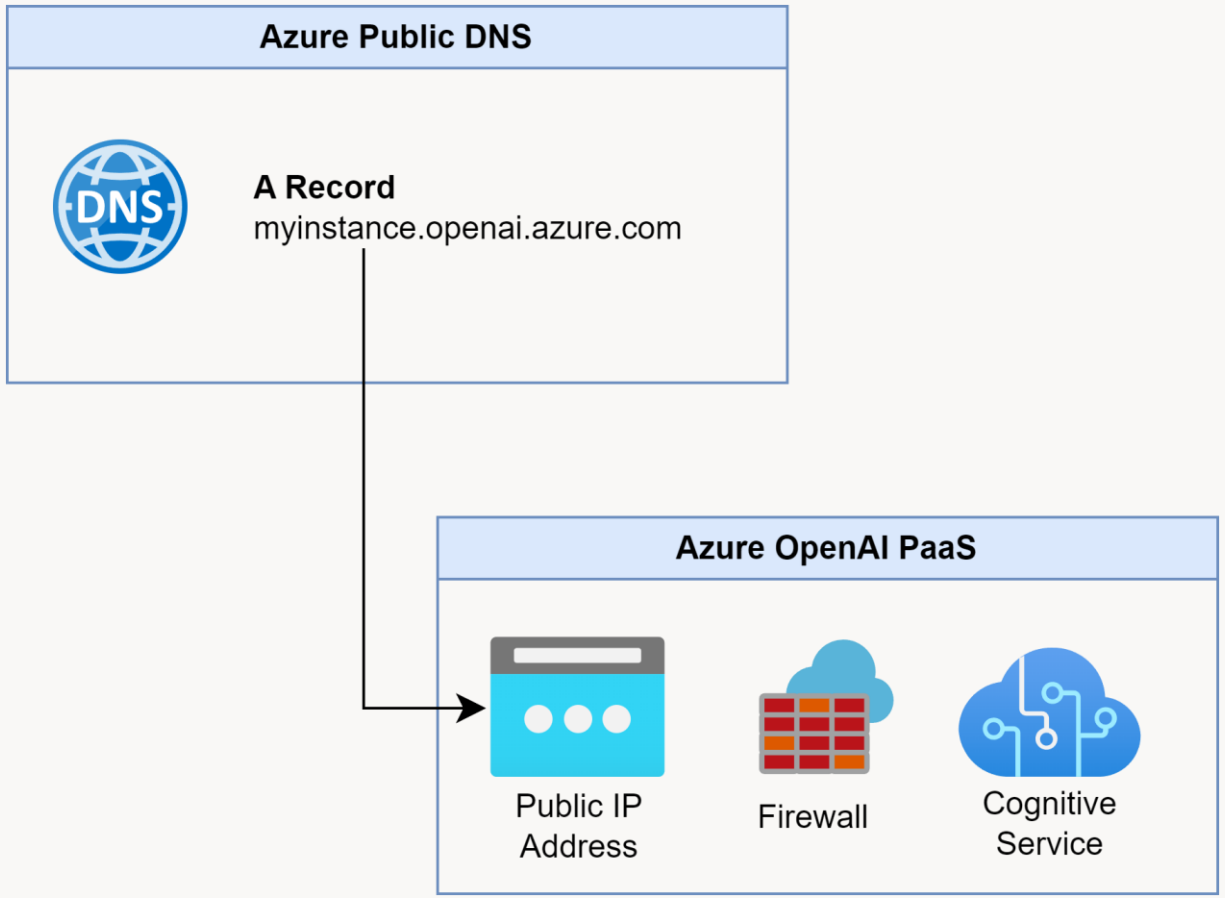
OpenAI Service PaaS – The Defaults (2)

For the public IP Address, an A record is created in Azure's Public DNS zone, which is used for IP Address resolution and TLS connections.



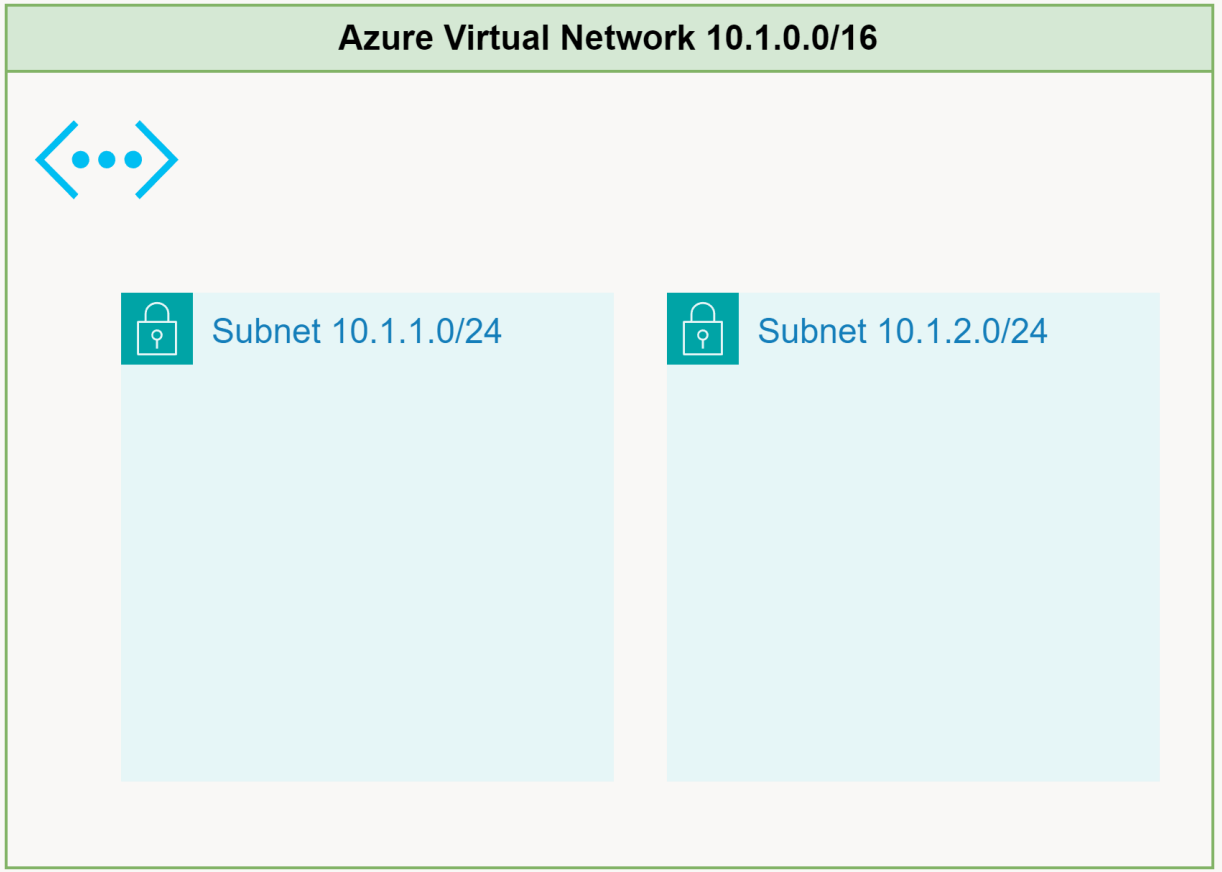
Why protect Azure OpenAI Service?

- By default, instances are available via the internet
- The service is secured via client access keys – this of course can become troublesome when one of these keys leak
- This becomes outright dangerous when an LLM has access to company data via data storage, indexes, or in Retrieval Augmented Generation (RAG) scenarios



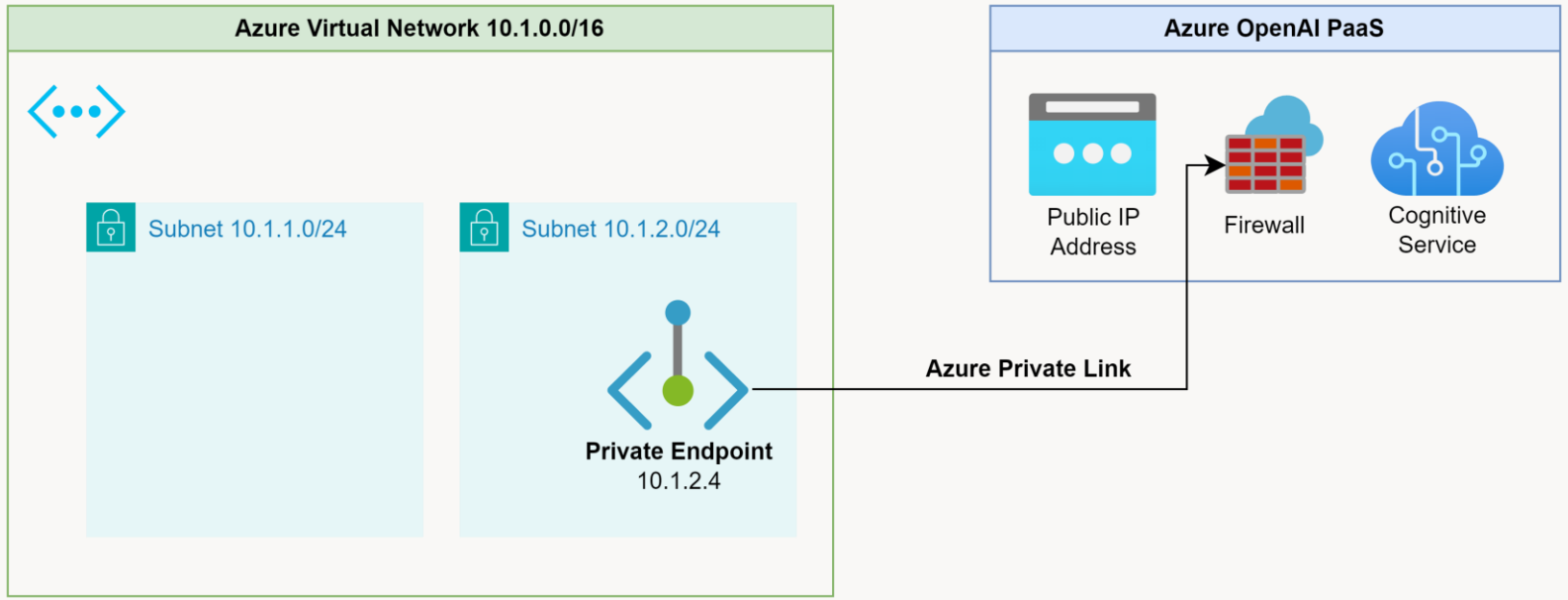
Let's start with Virtual Networks…

- In Azure, Virtual Networks (VNETs) represent the core building block for private networking
- A VNET lives in a single subscription and single region, but can span multiple Availability Zones (AZs)
- A VNET can be divided into subnets
- Typically, RFC1918 CIDR ranges are used within VNETs



…and add Private Link

- Azure Private Link is one way to connect a PaaS service to a VNET
- From a subnet, an IP address will be allocated that becomes a read-only Network Interface Card (NIC)
- This NIC routes packets to the OpenAI service instance within Azure
- This works across Azure regions

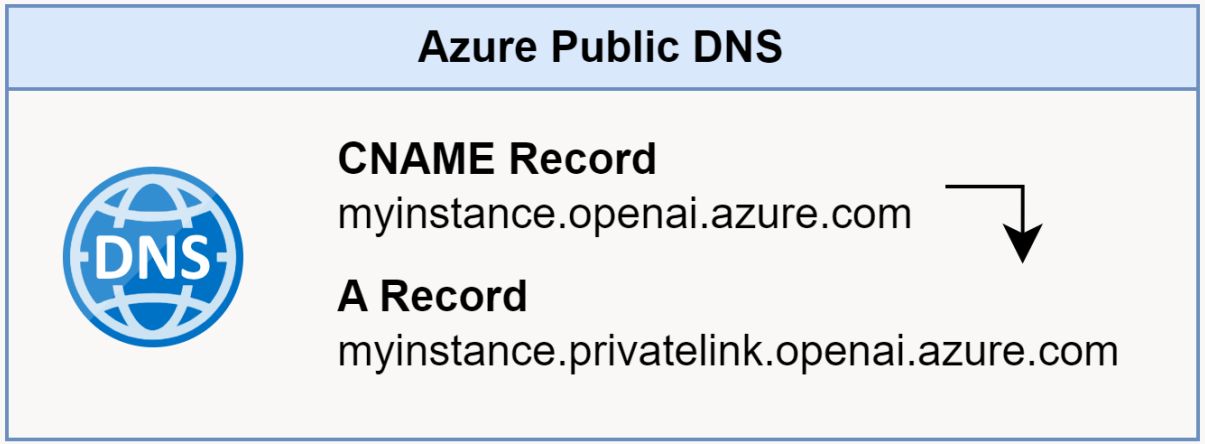


Public DNS and Private Endpoints

When creating a Private Link, records in the Azure Public DNS will change:

- A new A record for myinstance.**privatelink**.openai.azure.com will be created which points to the public IP address of our OpenAI service instance
- the previous A record for myinstance.openai.azure.com will be turned into a CNAME record which points to the new A record

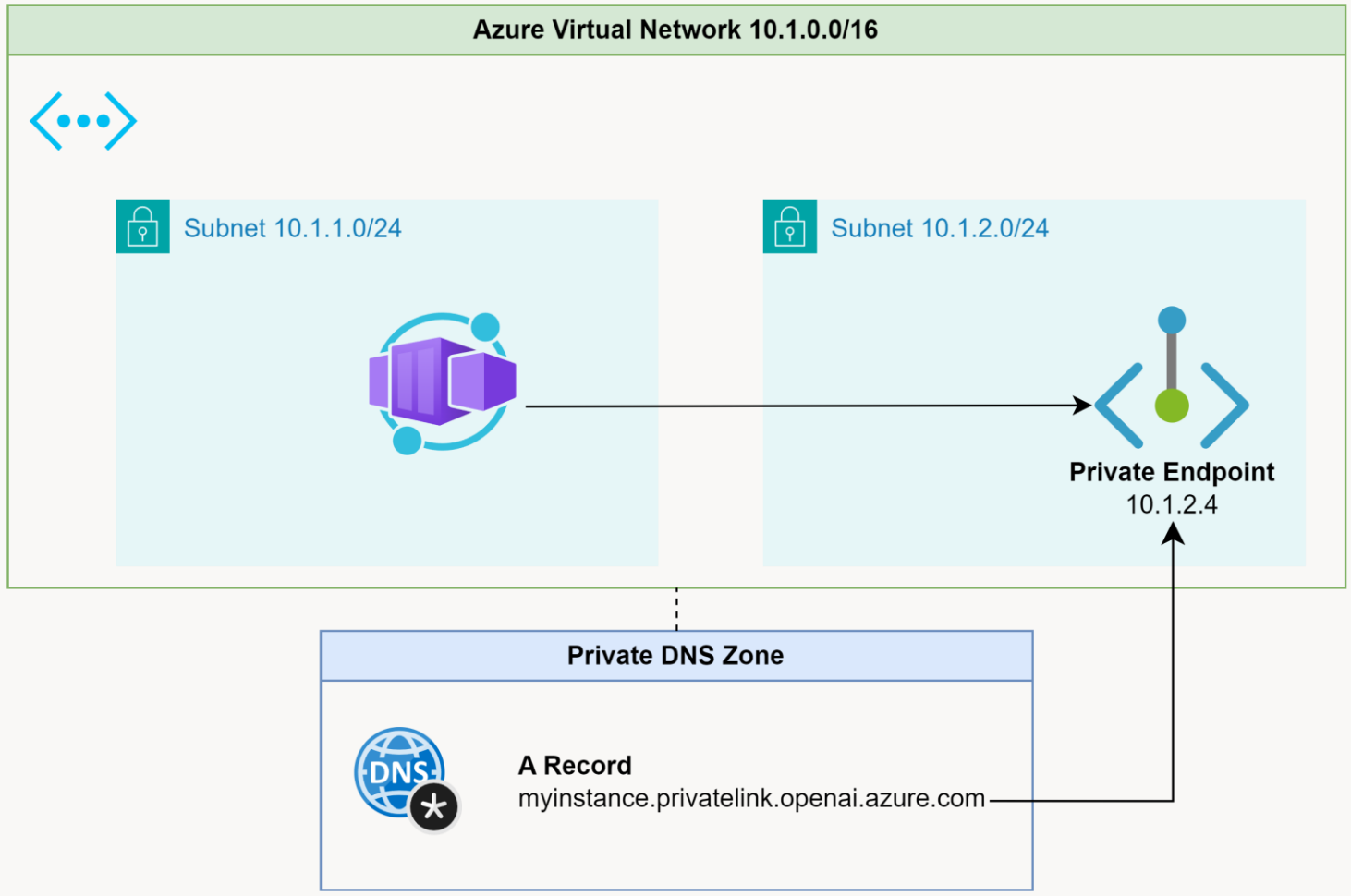
Important: our OpenAI Service instance is still available via the internet!



Private DNS Zone

Services in our VNET want to access the private endpoint via its domain name.

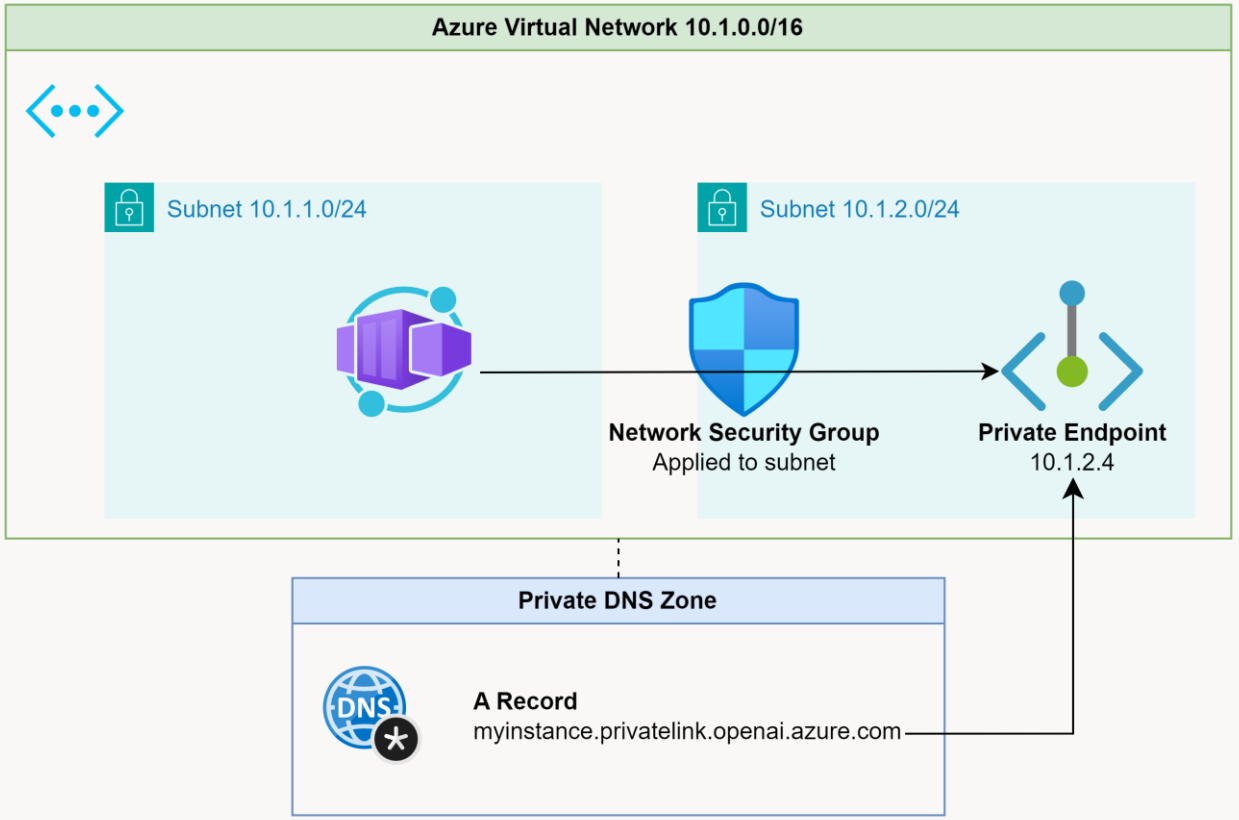
- A private DNS zone encapsulates a domain and provides records for it to linked VNETs
- In our private DNS zone, we need to define an A record which points to the IP of the Private Endpoint



Introducing Network Security Groups

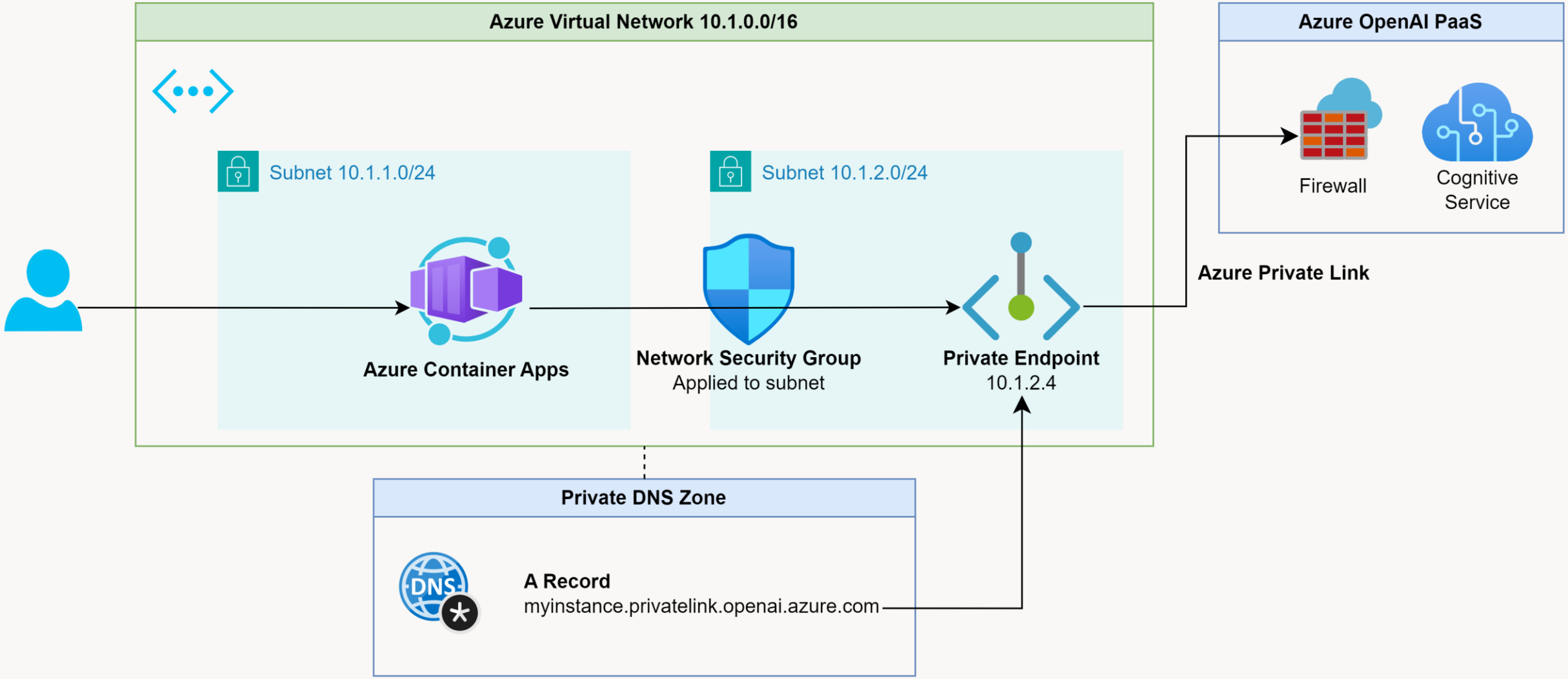
Network Security Groups (NSGs) can be used to allow and deny traffic

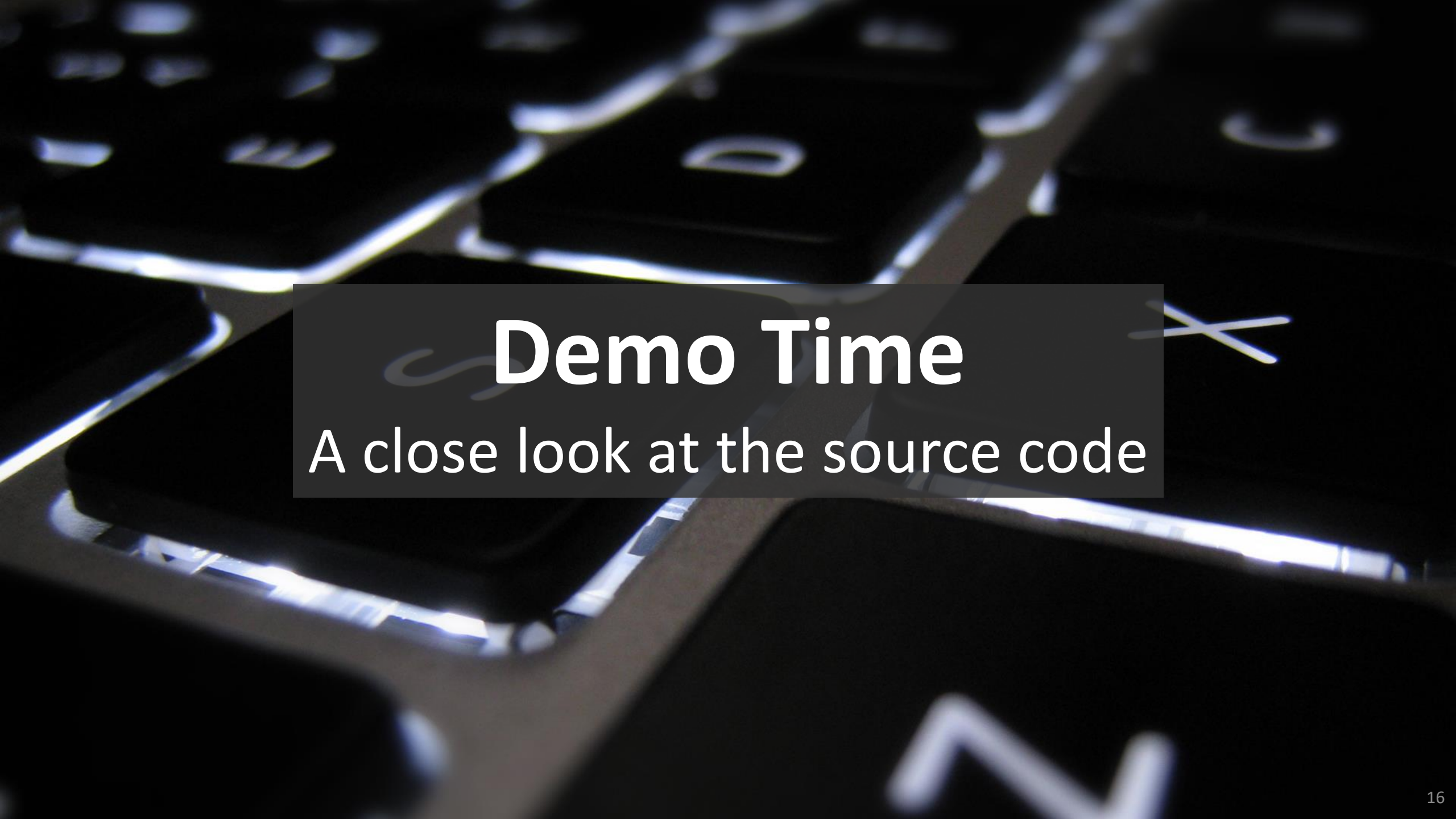
- They can be applied to VNET subnets or individual NICs (prefer the former)
- NSGs consist of rules that allow or deny traffic
- The sources and targets can be IPs, IP ranges, service tags, protocols, ports, or actions



Let's put it all together

Demo Time




Demo Time
A close look at the source code

Stream data from Open AI to your frontend with C#

- The Azure.AI.OpenAI NuGet package can be used to access an (Azure) OpenAI service from C#.
- The key class here is OpenAIClient: the DTO you pass in are the messages and instructions for the AI assistant
- The call to the OpenAI will return an IAsyncEnumerable<T> which unfortunately is hard to use with JS/TS-based clients
- This is why we simply write plain text asynchronously to the response (long-living HTTP request)
- An alternative would be using SignalR

```
private static async Task StreamChatResponse(OpenAIClient openAiClient,
                                             OpenAiAccessOptions openAiAccessOptions,
                                             HttpResponse response,
                                             ChatDto dto,
                                             ILogger logger,
                                             CancellationToken cancellationToken)
{
    response.StatusCode = StatusCodes.Status200OK;
    response.ContentType = "text/plain";
    var messages = new List<ChatRequestMessage>(dto.Messages.Count + 1)
    {
        new ChatRequestSystemMessage(
            "Please keep your answers short, they should be two paragraphs long or shorter.")
    };
    messages.AddRange(
        dto.Messages.Select(
            m => m.Originator == Originators.Ai ?
                (ChatRequestMessage) new ChatRequestAssistantMessage(m.Text) :
                new ChatRequestUserMessage(m.Text)
        )
    );

    var options = new ChatCompletionsOptions(openAiAccessOptions.ModelName, messages)
    {
        ChoiceCount = 1,
        MaxTokens = 1000
    };
    var streamingResponse = await openAiClient.GetChatCompletionsStreamingAsync(options, cancellationToken);
    await foreach (var item in streamingResponse)
    {
        logger.Debug("{@StreamItem}", item);
        if (!item.ContentUpdate.IsNullOrEmpty()) // Do not ignore white space responses
            await response.WriteAsync(item.ContentUpdate, cancellationToken);
    }
}
```

Receiving streams in Angular clients

- You can simply use the regular Angular HttpClient
- You need to report Progress, observe events, and the response type must be text
- The HTTP response will then not be buffered, but you will receive data from the observable every time the backend writes a chunk to the response body
- Depending on the HTTP event type, you can aggregate the data in a different way and provide a unified structure to the calling component

```
getAiResponse(chatMessages: ChatMessage[]): Observable<AiResponseDto> {

  const url = environment.apiBaseAddress + '/api/chat';
  const dto: ChatDto = { messages: chatMessages };
  return this.httpClient
    .post(
      url,
      dto,
      {
        reportProgress: true,
        observe: 'events',
        responseType: 'text'
      }
    )
    .pipe(
      filter(event => event.type === HttpEventType.Response || event.type === HttpEventType.DownloadProgress),
      map(event => {
        let message: string = '';
        let isFinsihed: boolean = false;
        if (event.type === HttpEventType.DownloadProgress) {
          const progressEvent = event as HttpDownloadProgressEvent;
          if (progressEvent.partialText) {
            message = progressEvent.partialText;
          }
        }
        else if (event.type === HttpEventType.Response) {
          isFinsihed = true;
          if (event.body) {
            message = event.body;
          }
        }

        const responseDto: AiResponseDto = {
          message,
          isFinsihed
        };
        return responseDto;
      })
    );
}
```

How can we further improve network security?

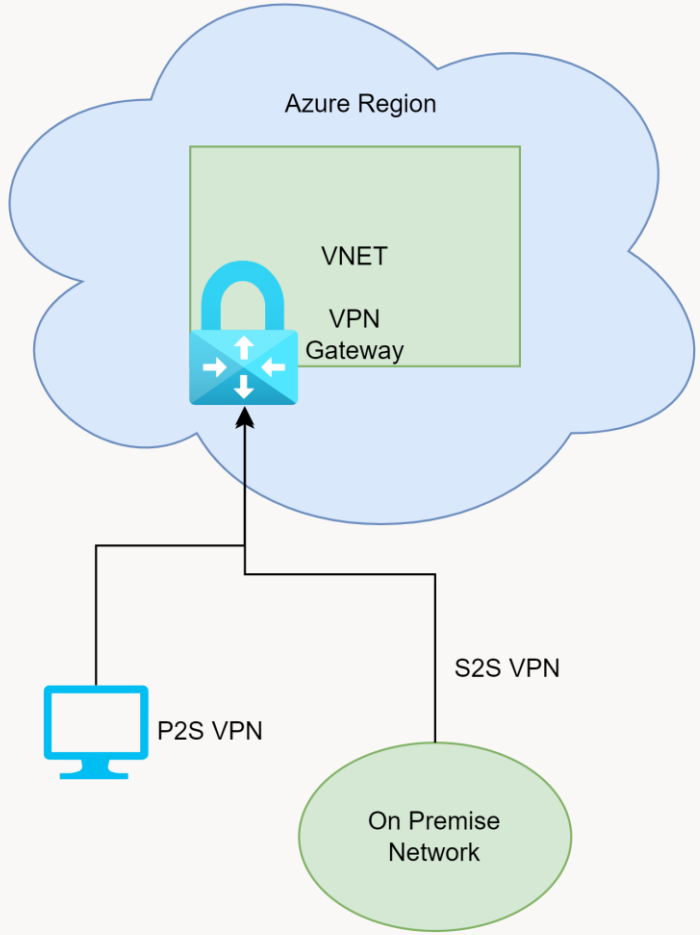
- Secure the web app with proper authentication (IDP, MFA)
- Implement a Hub-and-Spoke network topology with an Azure Firewall
- Secret management, for example via Azure Key Vault
- Save messages and responses
- Use advanced features that might require Sematic Kernel or LangChain support

How to connect to company networks to Azure VNETs?

There are several easy ways to connect your on-premise network to your Azure VNET:

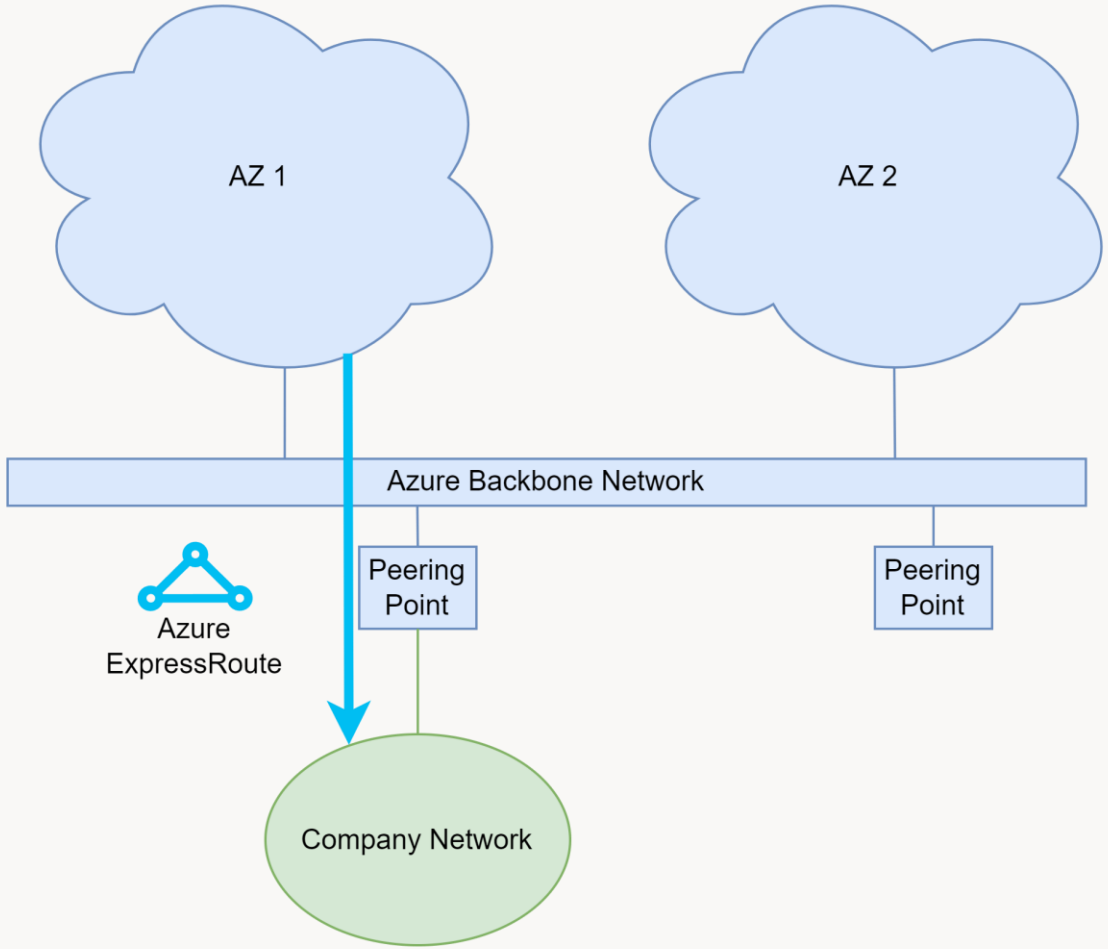
- Point-to-Site (P2S): connects a single computer to the VNET – Azure Bastion is a managed solution
- Site-to-Site (S2S): connects a single on-premise network to a VNET

In both circumstances, an Azure VPN Gateway must be set up that allows access to the VNET.



Azure ExpressRoute

- VPN over the Internet might come with network fluctuations
- You can directly connect to Azure's Backbone network via ExpressRoute
- In a Peering Point facility, an ISP can connect your company network with Microsoft routers
- The "last mile" (from PP facility to your company network) is also usually provided by the ISP



Sources

- Security Best Practices for GenAI Applications (OpenAI) in Azure
- Azure OpenAI Service models
- Terraform Azure RM Cognitive Account Docs
- Angular HttpClient track and show request progress

Danke schön!

Demos aus der Session:
https://github.com/thinktecture-labs/dcn-host-azure-openai-service-securely

https://www.thinktecture.com/wissen/
https://labs.thinktecture.com/

https://www.thinktecture.com/ueber-uns/karriere/

Kenny Pflug
https://thinktecture.com/kenny-pflug