May 5th – 7th,2022

#GlobalAzure

BY COMMUNITY • FOR COMMUNITY •

SPAIN #GLOBALAZURE

Nuestro patrocinadores

avanade

bravent
By veryTI group

clevertask

encamina
PIENSA EN COLORES

NTT DaTa

ilitia

Insight

intelequia

KaBel

myCloudDoor

nunsys
Tu socio tecnológico

plain concepts

prodware

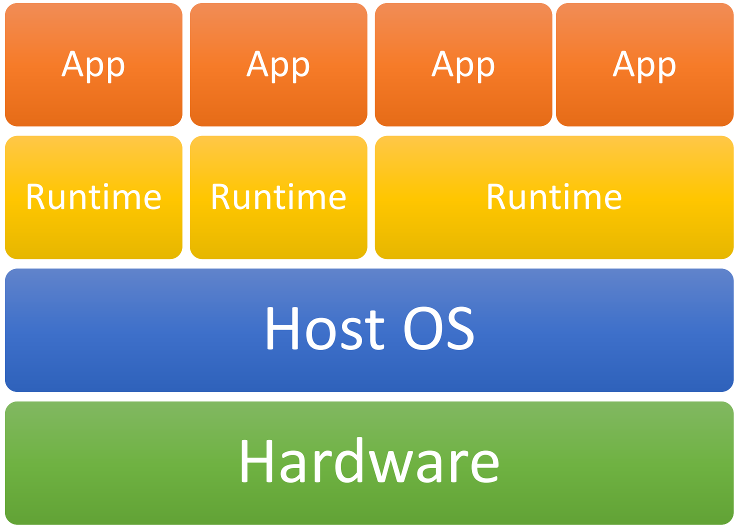
Colabora

Microsoft

· BY COMMUNITY · FOR COMMUNITY ·
SPAIN
#GLOBALAZURE

ALL COMMANDS AND
EVENTS IN THIS SHOW --
EVEN THOSE BASED ON REAL
ONES -- ARE ENTIRELY FICTIONAL.
ALL CRAPY SLIDES ARE
EXPLAINED…..POORLY. THE
FOLLOWING SESSION CONTAINS
COARSE LANGUAGE AND DUE TO
ITS CONTENT IT SHOULD NOT BE
VIEWED BY ANYONE

Traditional environment



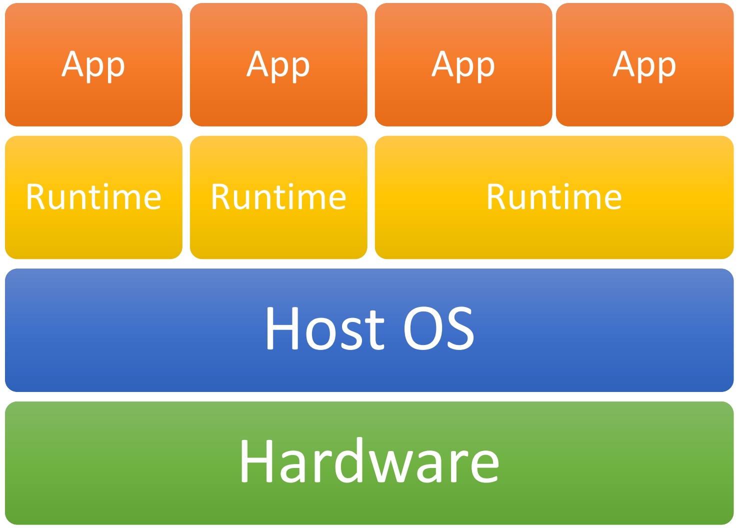
App App App App

Runtime Runtime Runtime

Host OS

Hardware

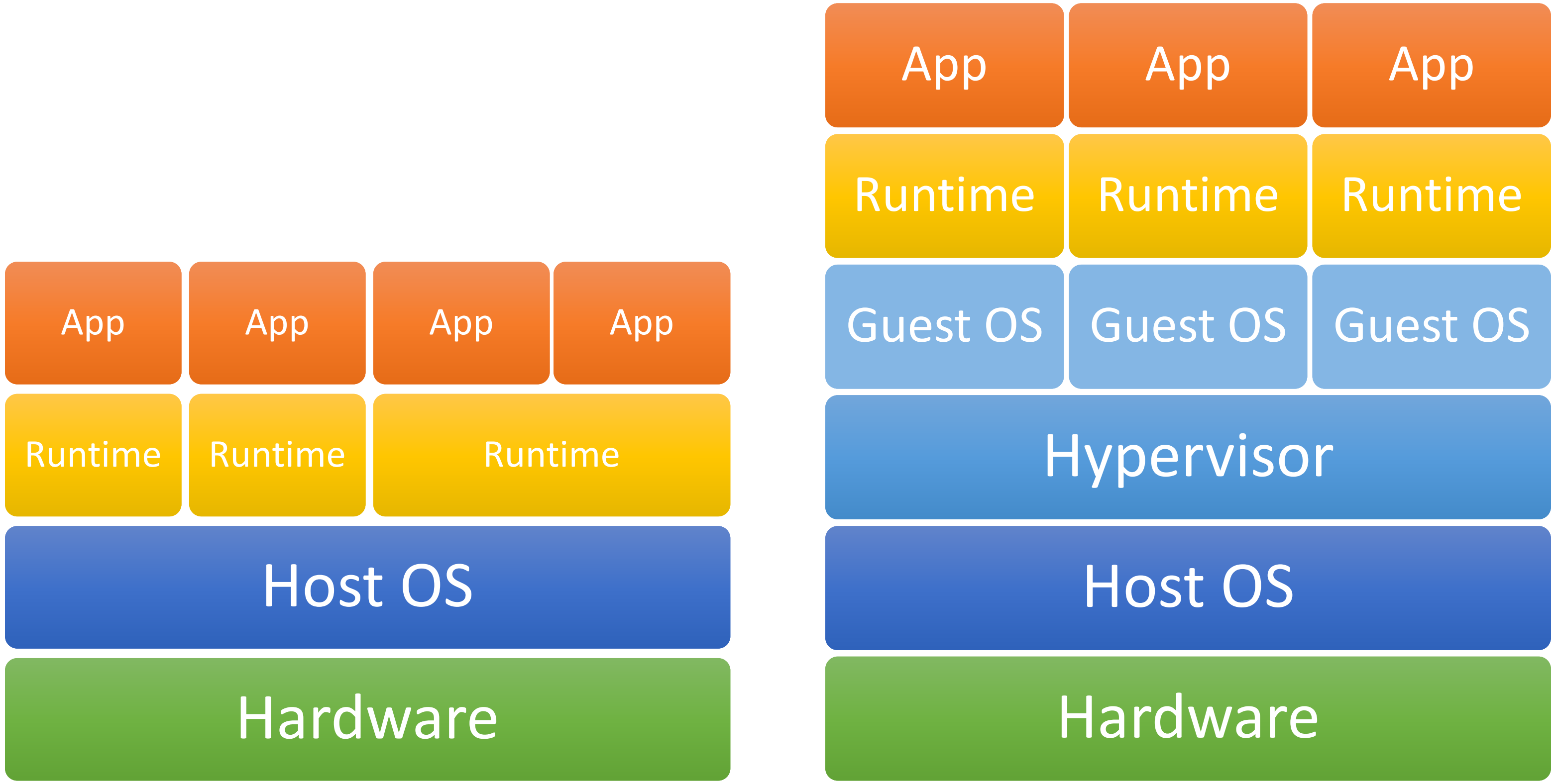
Traditional environment



Problems:

- Shared environment variables
- Can not have different versions of the same runtime
- Can not scale applications automatically as the resource is static

Virtual environment



App App App App

Runtime Runtime Runtime

Host OS

Hardware

App App App

Runtime Runtime Runtime

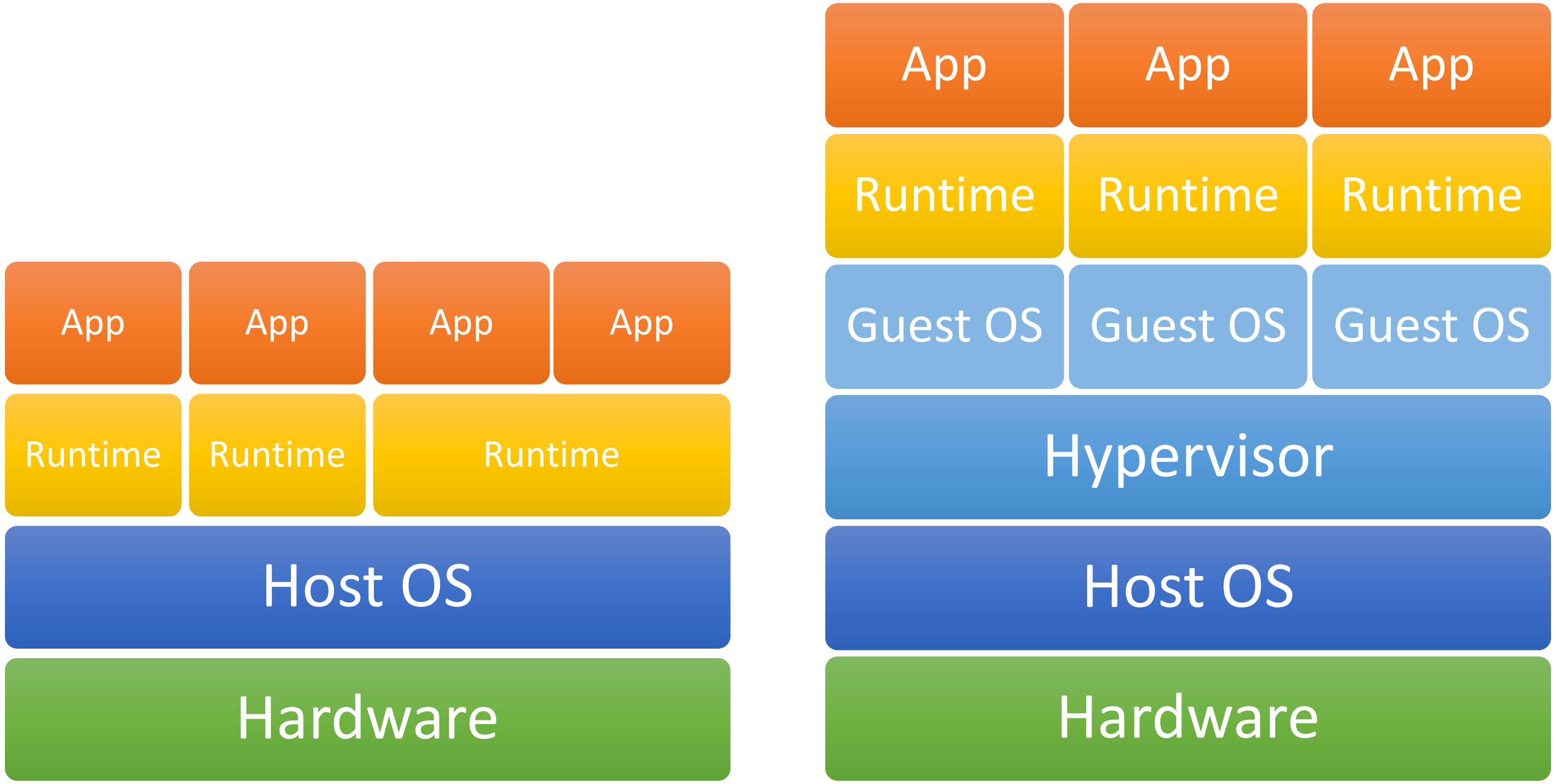
Guest OS Guest OS Guest OS

Hypervisor

Host OS

Hardware

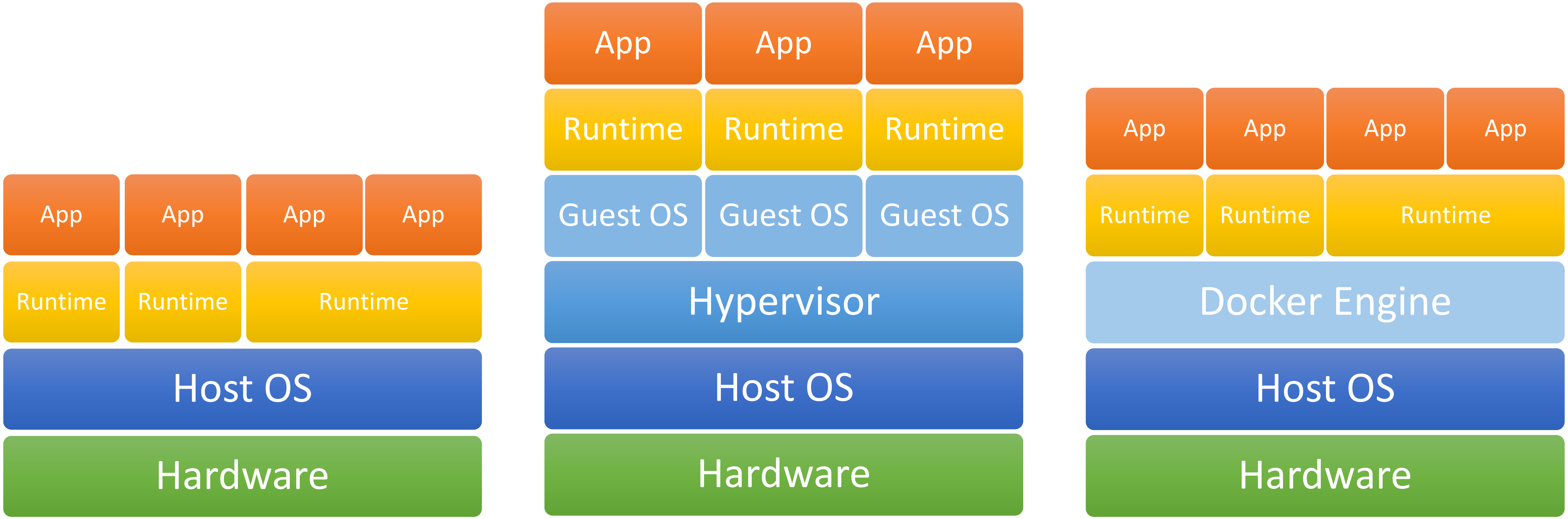
Virtual environment



Problems:

- 1 VM per App = overkill
- Shared environments variables in the same VM
- Slow scalation as the VM resources are static
- Resources are wasted

Containers environment

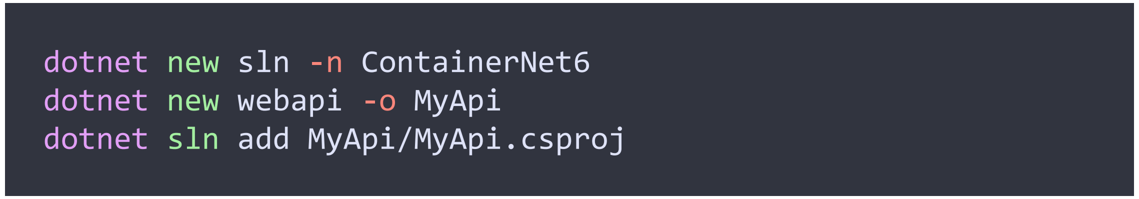


App App App App
Runtime Runtime Runtime
Host OS
Hardware

App App App
Runtime Runtime Runtime
Guest OS Guest OS Guest OS
Hypervisor
Host OS
Hardware

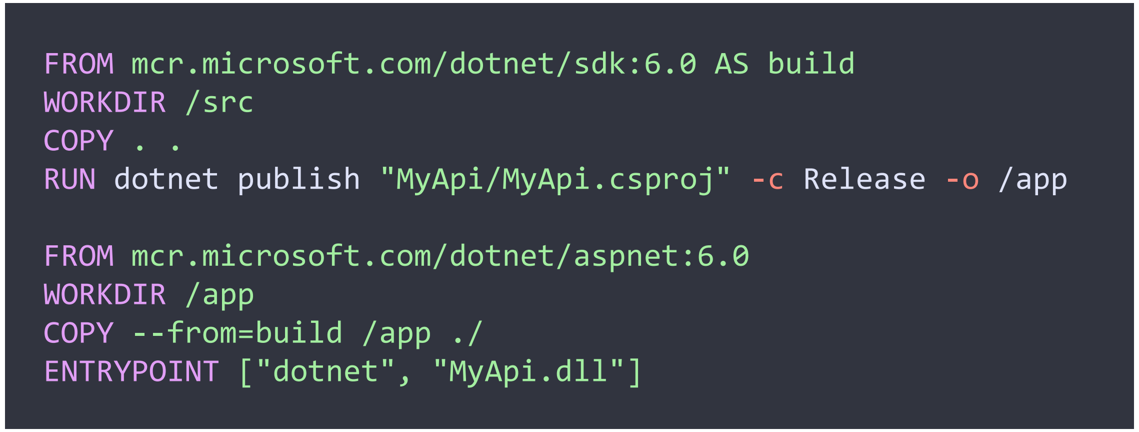
App App App App
Runtime Runtime Runtime
Docker Engine
Host OS
Hardware

demo



```
dotnet new sln -n ContainerNet6
dotnet new webapi -o MyApi
dotnet sln add MyApi/MyApi.csproj
```

Dockerfile

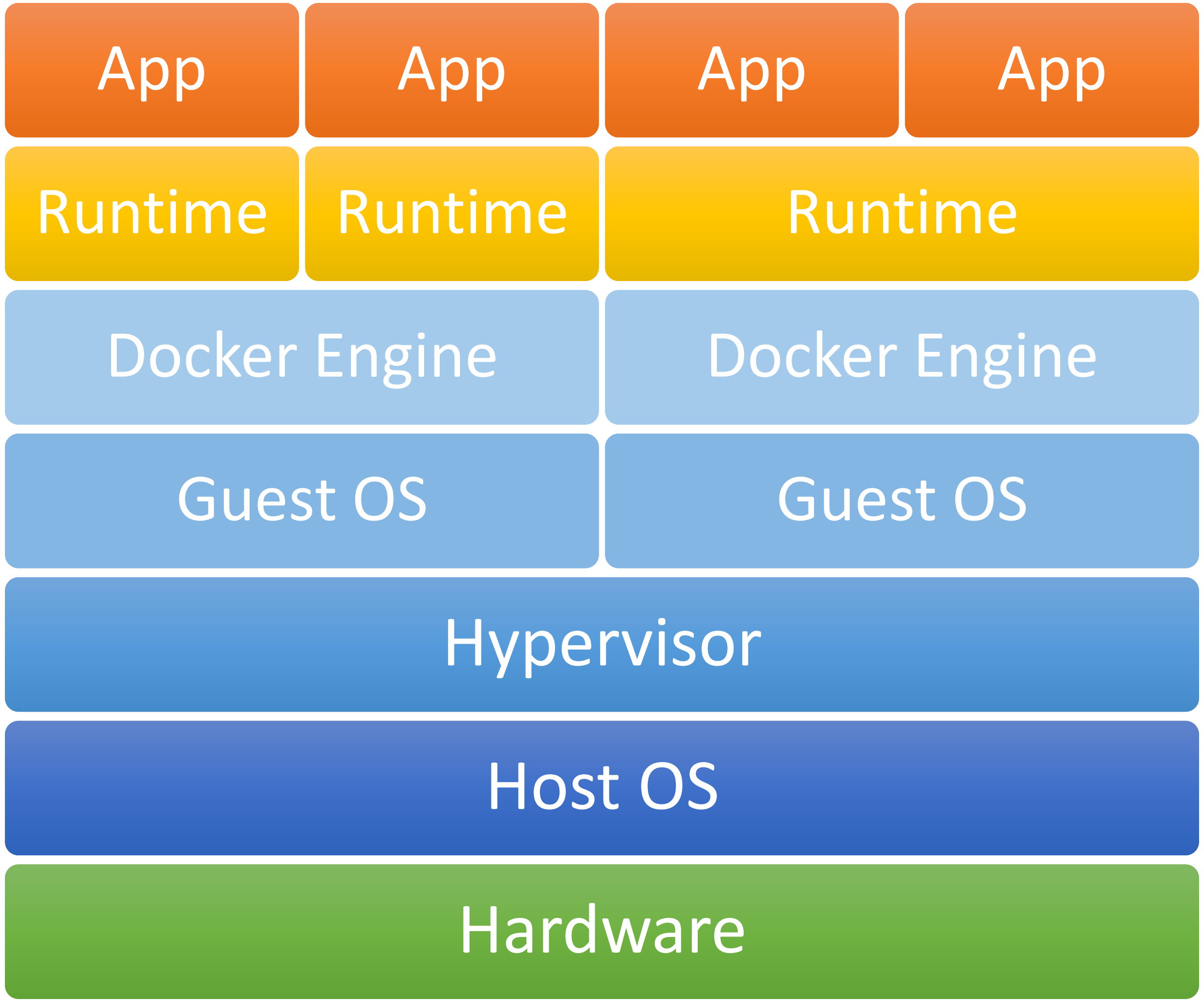


```
FROM mcr.microsoft.com/dotnet/sdk:6.0 AS build
WORKDIR /src
COPY . .
RUN dotnet publish "MyApi/MyApi.csproj" -c Release -o /app

FROM mcr.microsoft.com/dotnet/aspnet:6.0
WORKDIR /app
COPY --from=build /app ./
ENTRYPOINT ["dotnet", "MyApi.dll"]
```

demo

```
docker build -t myapi:0.1 .
docker run -p 8080:80 --rm myapi
curl http://localhost:8080/WeatherForecast
```

Actual environment



App App App App

Runtime Runtime Runtime

Docker Engine Docker Engine

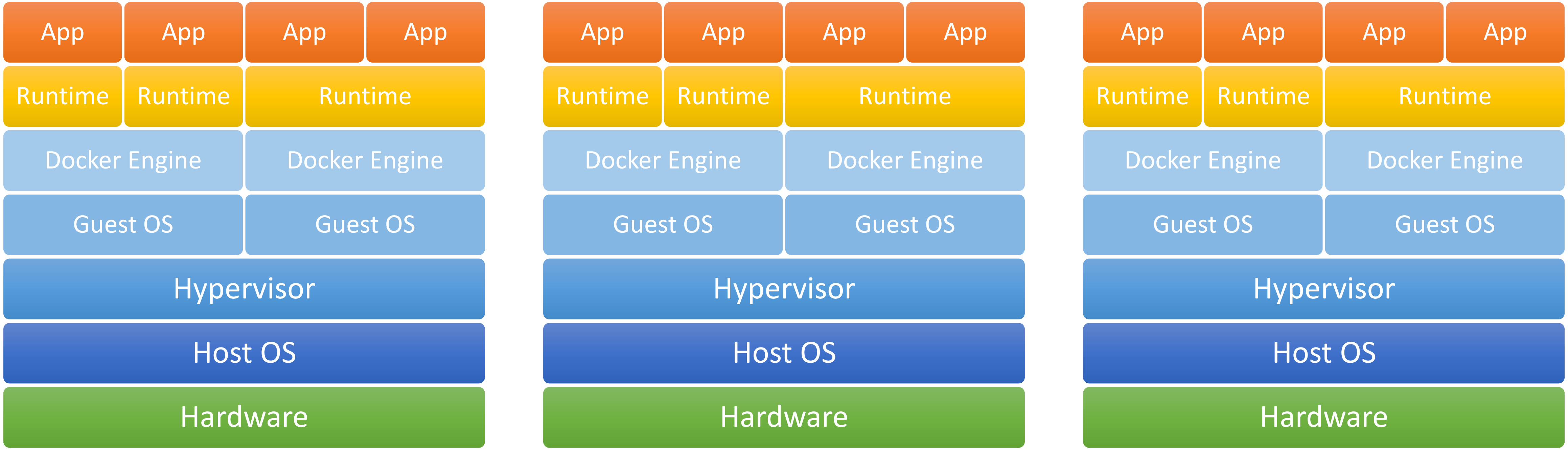
Guest OS Guest OS

Hypervisor

Host OS

Hardware

Actual environment



App App App App
Runtime Runtime Runtime
Docker Engine Docker Engine
Guest OS Guest OS
Hypervisor
Host OS
Hardware

App App App App
Runtime Runtime Runtime
Docker Engine Docker Engine
Guest OS Guest OS
Hypervisor
Host OS
Hardware

App App App App
Runtime Runtime Runtime
Docker Engine Docker Engine
Guest OS Guest OS
Hypervisor
Host OS
Hardware

Kubernetes = K8S

- **Service Discovery** and **Load Balancing**: Kubernetes can expose a container using the DNS name or using their own IP address.

- **Self-healing**: Auto healing is a great feature that Kubernetes provides — it restarts, kills, and replaces containers that fail.

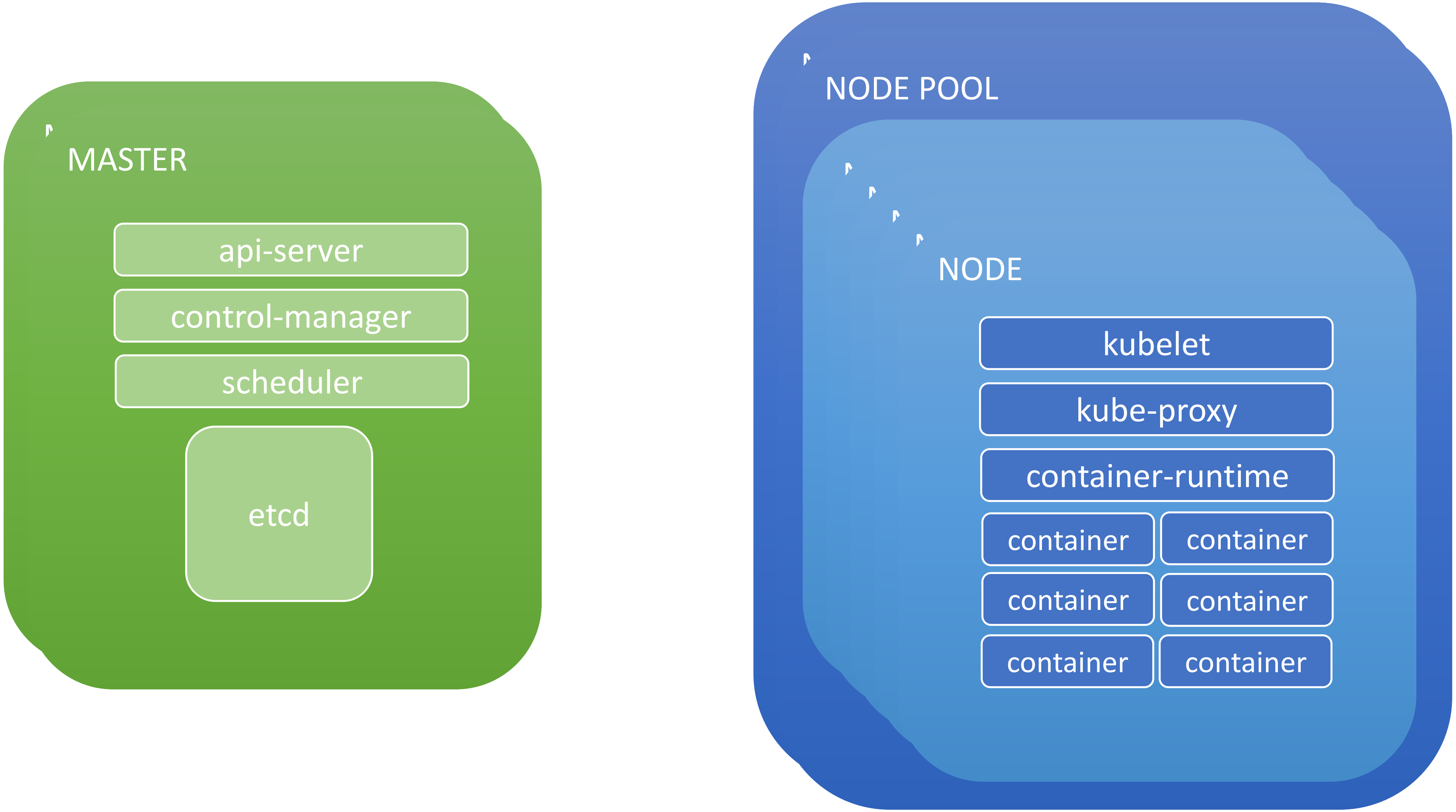
- Automated **Roll outs** and **Rollbacks**: Micro service systems could include hundreds, if not thousands, of services which can be hard or impossible to spin up manually. With this feature, you're able to specify the desired state of a given application (deployment) and Kubernetes will do the work to make sure to achieve this state.

- **Secret/Config** Management: This allows you to store config and sensitive data like passwords, tokens and SSH keys.

- Auto **Resource Management**: Specify the resource, RAM and CPU, needed for your deployments, and Kubernetes will distribute containers to relevant nodes, and fit them for optimal use of machine resources.

- **Storage Orchestration**: Kubernetes allows you to automatically mount a storage system of your choice, such as local storage or from public cloud providers.

K8S


MASTER

api-server

control-manager

scheduler

etcd

NODE POOL

NODE

kubelet

kube-proxy

container-runtime

container

container

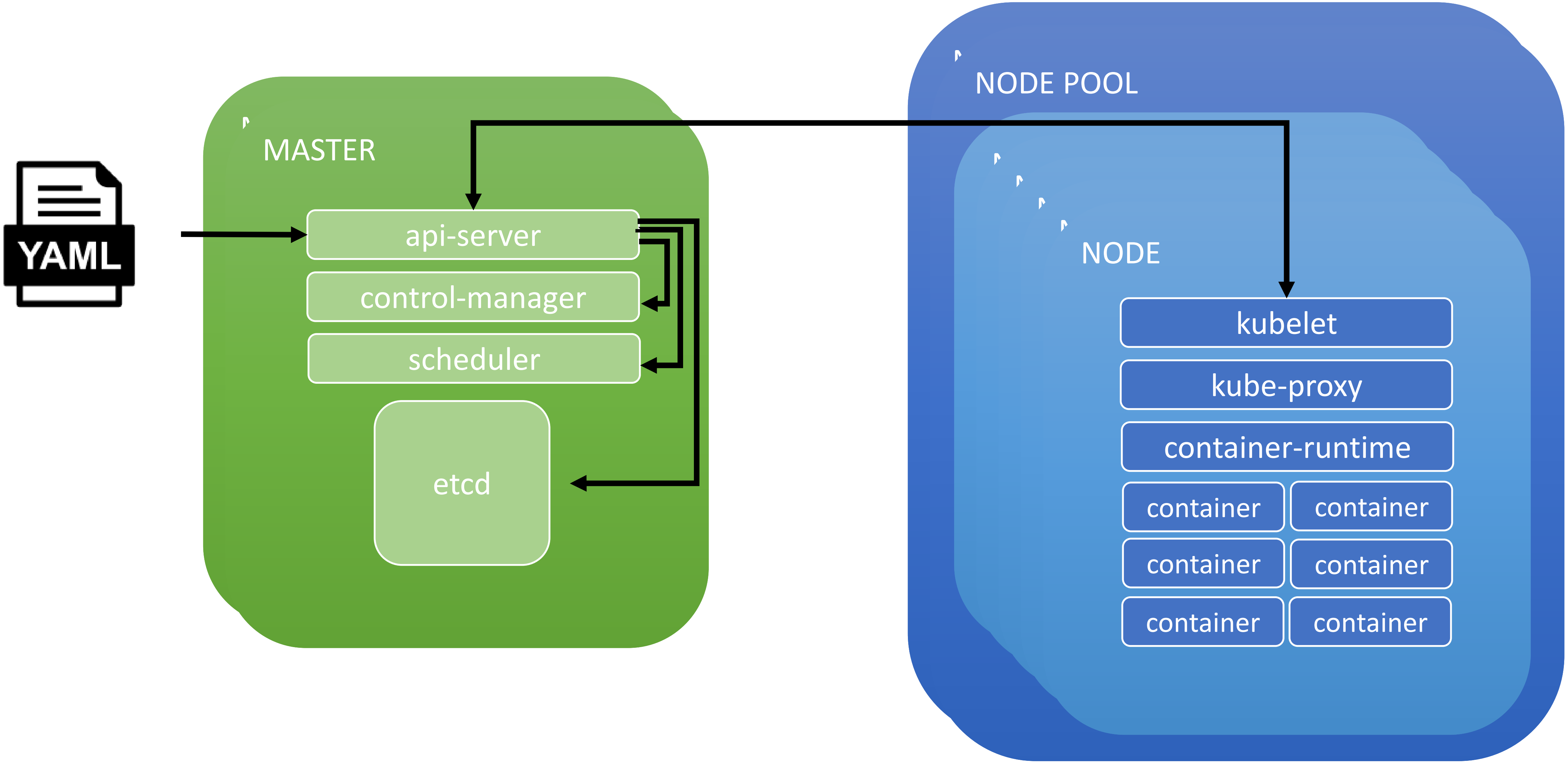
container

container

container

container

K8S


YAML
MASTER
api-server
control-manager
scheduler
etcd
NODE POOL
NODE
kubelet
kube-proxy
container-runtime
container
container
container
container
container
container

Azure Kubernetes Service = AKS

- **Managed Kubernetes Cluster:** Azure Kubernetes Service (AKS) offers serverless Kubernetes, an integrated continuous integration and continuous delivery (CI/CD) experience, and enterprise-grade security and governance. Unite your development and operations teams on a single platform to rapidly build, deliver, and scale applications with confidence.

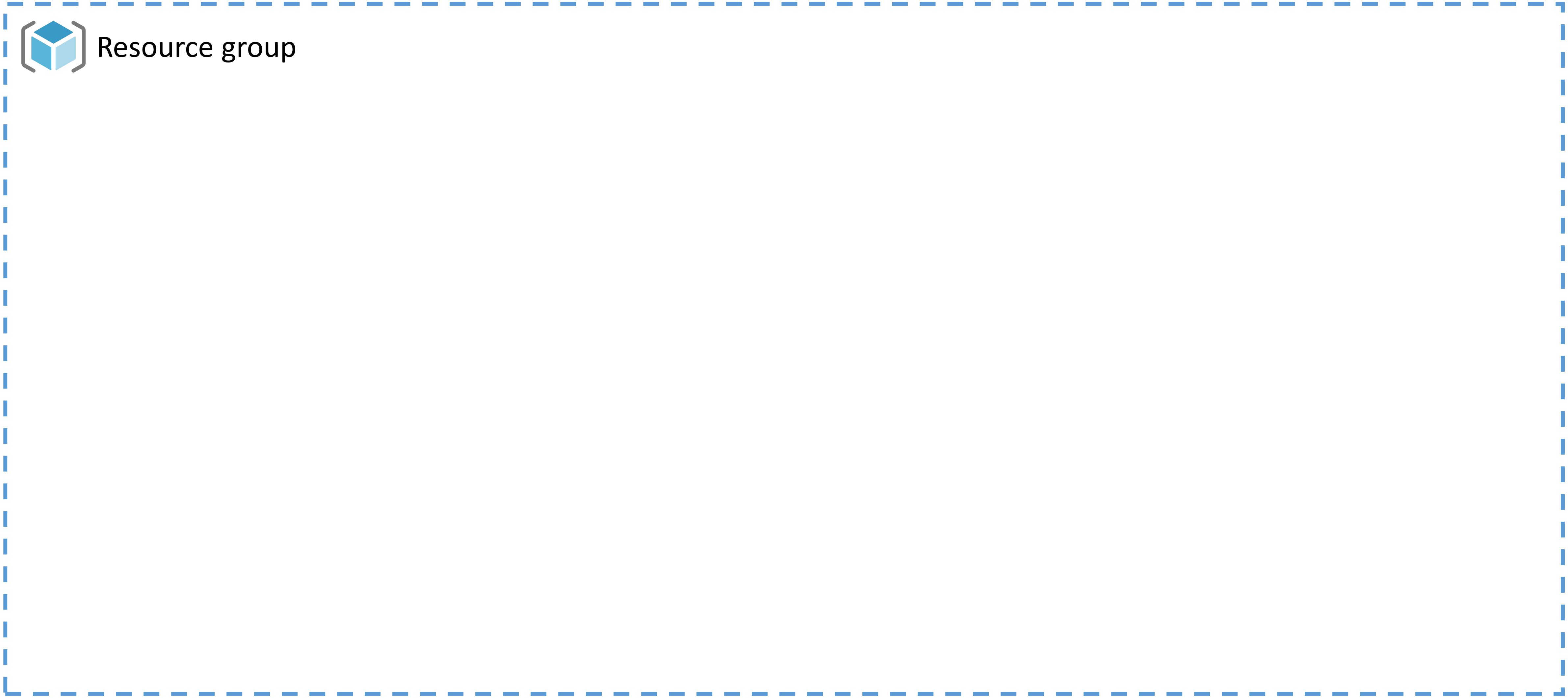
- **Elastic provisioning** of capacity without the need to manage the infrastructure and with the ability to add event-driven autoscaling and triggers.

- Faster end-to-end development experience through **Azure Kubernetes tools**.

- **Most comprehensive authentication and authorization** capabilities using Azure Active Directory, and dynamic rules enforcement across multiple clusters with Azure Policy.

- Availability in **more regions** than any other cloud provider.

AKS



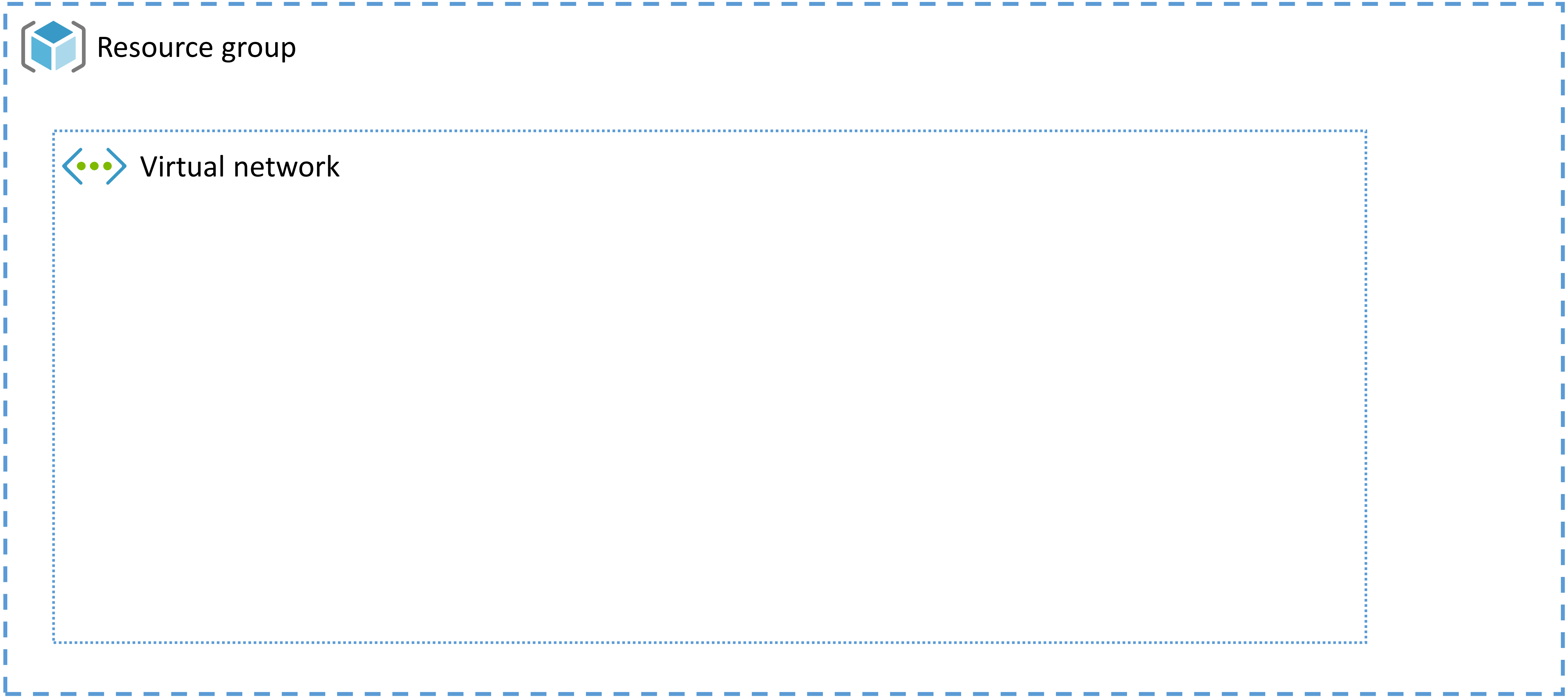
Resource group

AKS

Resource group

```
rg="fer-globalazure-2022"

n="fergab22"

az group create \
  --name $rg \
  --location westeurope
```

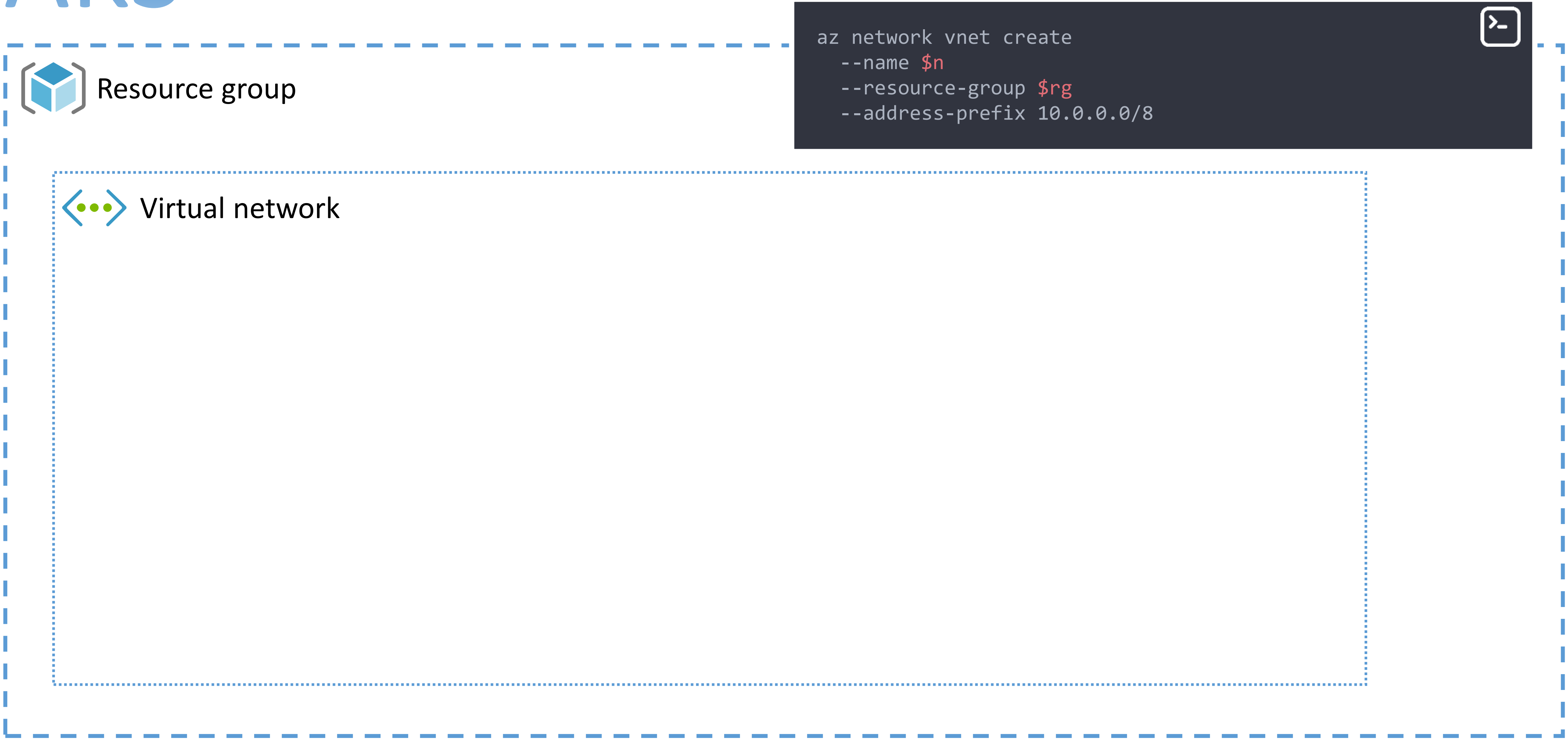
AKS



Resource group

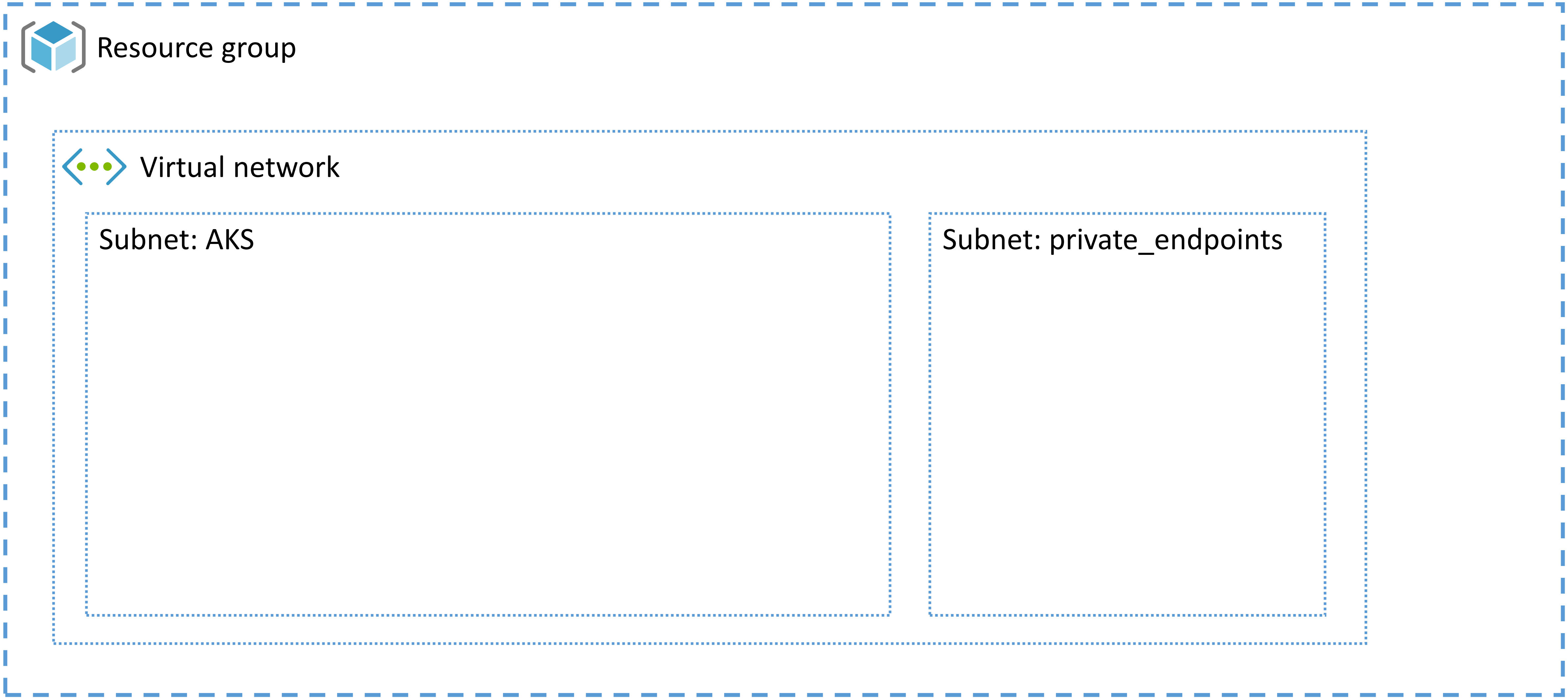
Virtual network

AKS



```
az network vnet create
  --name $n
  --resource-group $rg
  --address-prefix 10.0.0.0/8
```

AKS



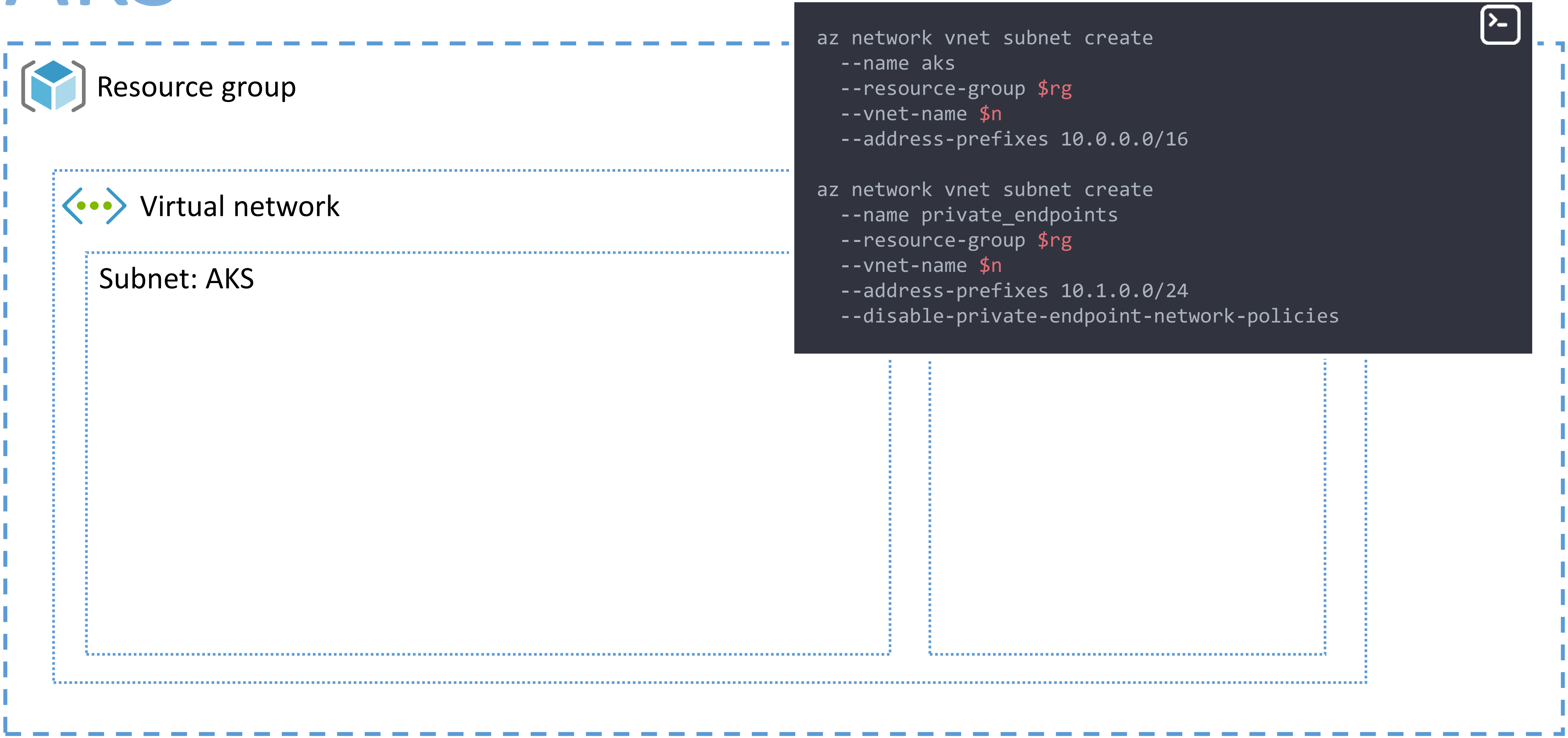
Resource group

Virtual network

Subnet: AKS

Subnet: private_endpoints

AKS



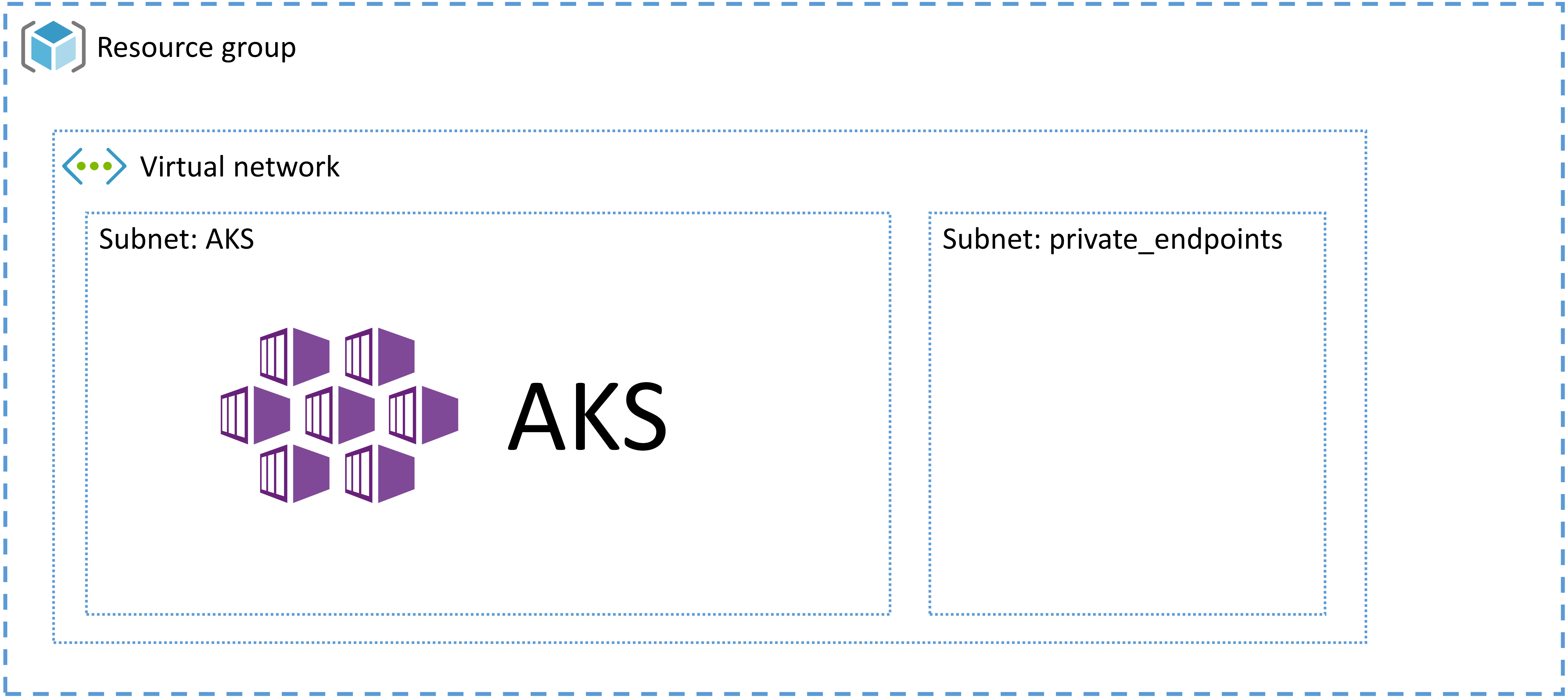
Resource group

Virtual network

Subnet: AKS

```
az network vnet subnet create
  --name aks
  --resource-group $rg
  --vnet-name $n
  --address-prefixes 10.0.0.0/16

az network vnet subnet create
  --name private_endpoints
  --resource-group $rg
  --vnet-name $n
  --address-prefixes 10.1.0.0/24
  --disable-private-endpoint-network-policies
```

AKS



Resource group

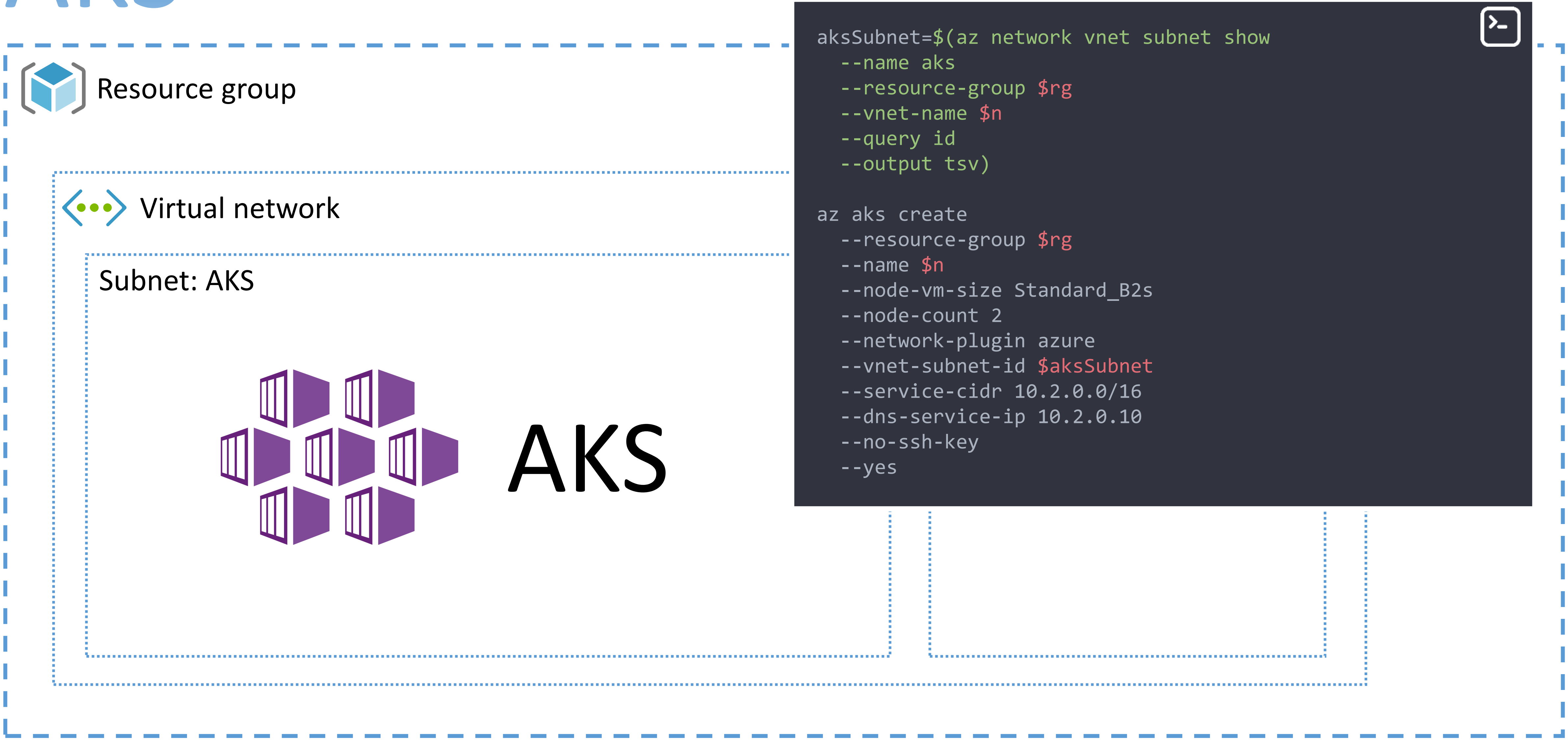
Virtual network

Subnet: AKS

AKS

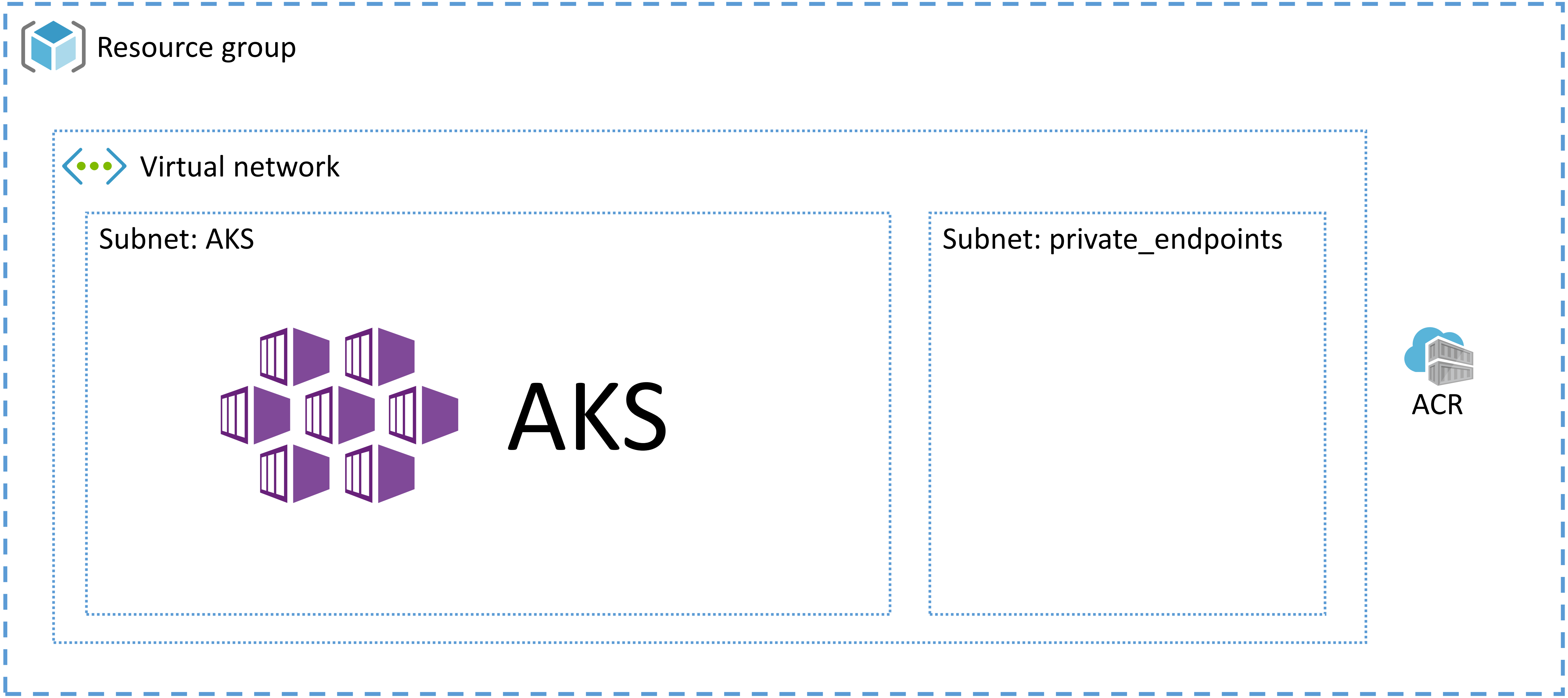
Subnet: private_endpoints

AKS



```
aksSubnet=$(az network vnet subnet show
  --name aks
  --resource-group $rg
  --vnet-name $n
  --query id
  --output tsv)

az aks create
  --resource-group $rg
  --name $n
  --node-vm-size Standard_B2s
  --node-count 2
  --network-plugin azure
  --vnet-subnet-id $aksSubnet
  --service-cidr 10.2.0.0/16
  --dns-service-ip 10.2.0.10
  --no-ssh-key
  --yes
```

AKS



Resource group

Virtual network

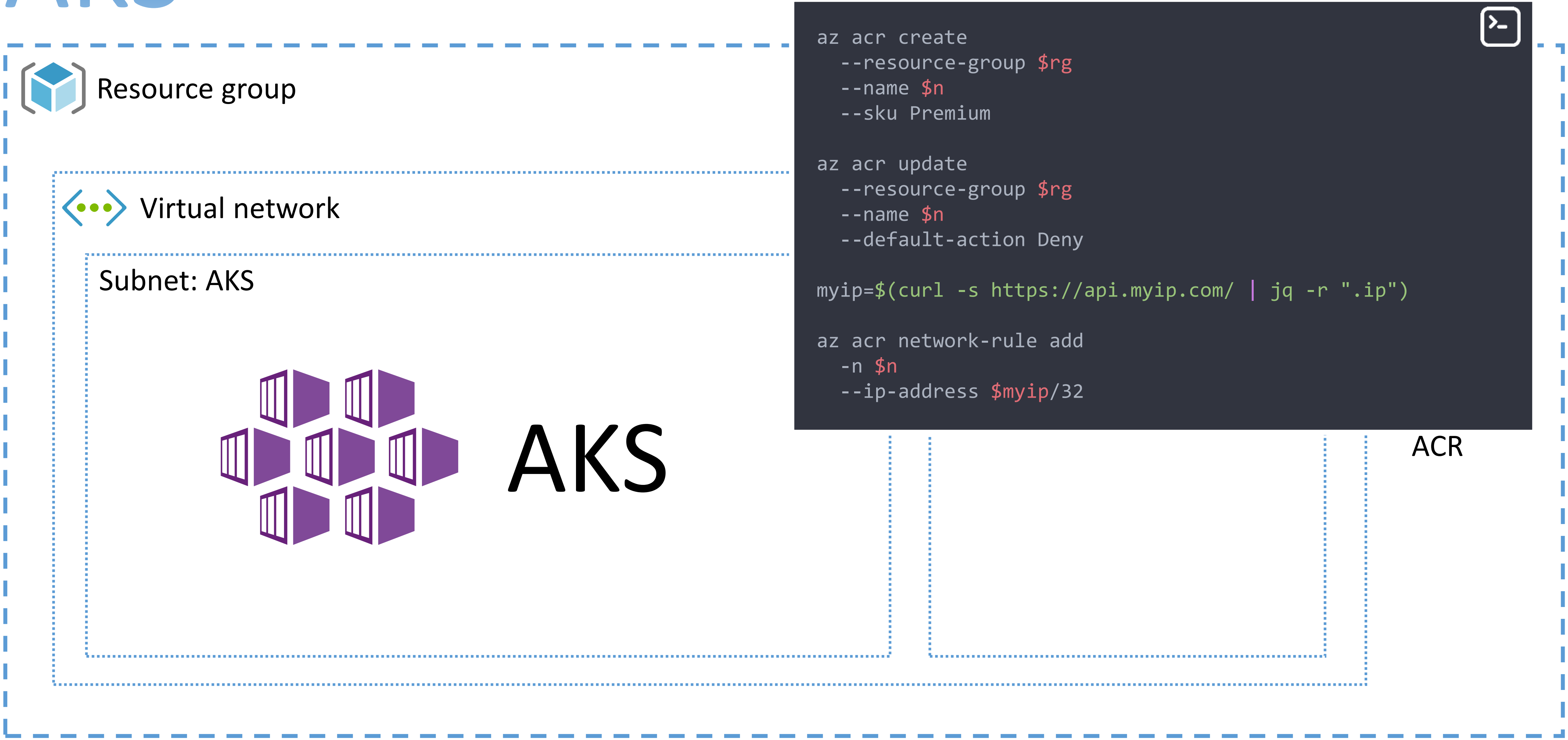
Subnet: AKS

AKS

Subnet: private_endpoints

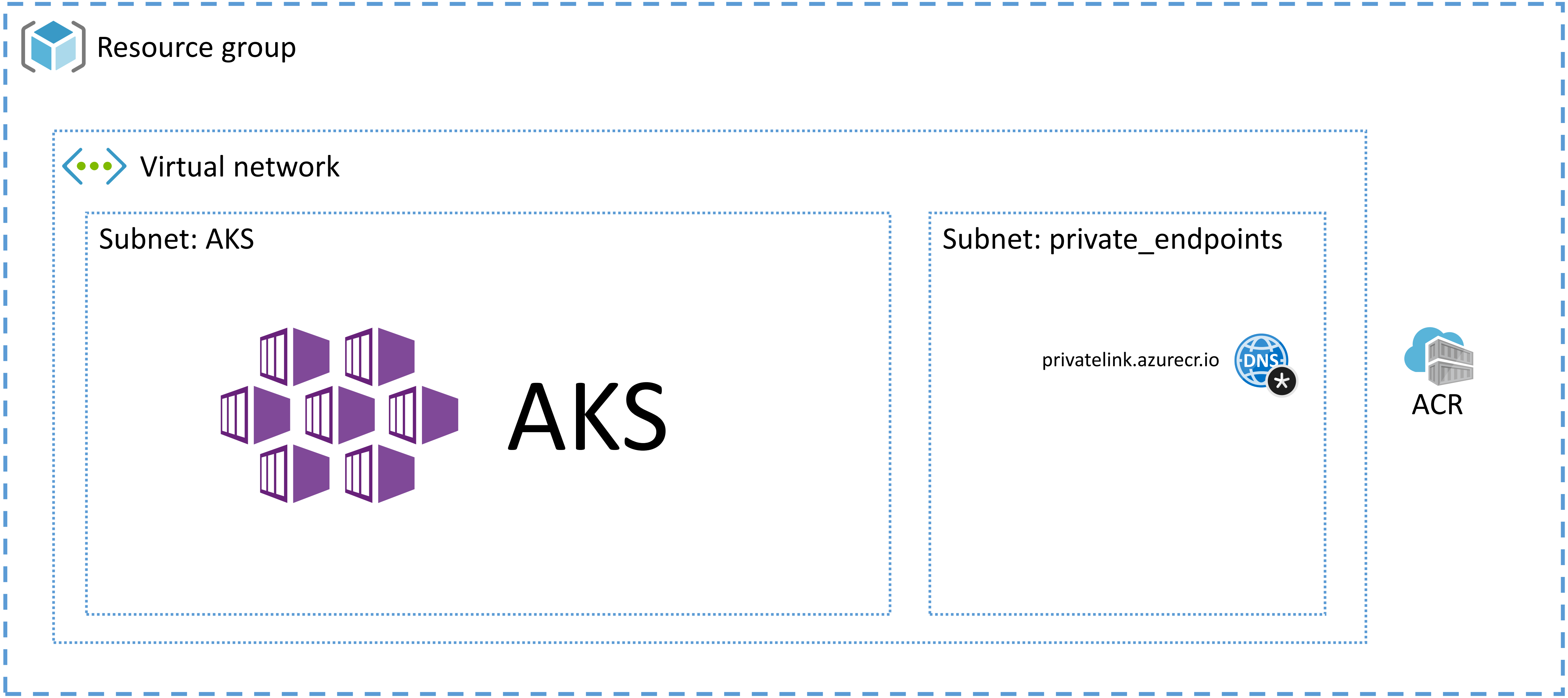
ACR

AKS



```
az acr create
  --resource-group $rg
  --name $n
  --sku Premium

az acr update
  --resource-group $rg
  --name $n
  --default-action Deny

myip=$(curl -s https://api.myip.com/ | jq -r ".ip")

az acr network-rule add
  -n $n
  --ip-address $myip/32
```

AKS



Resource group

Virtual network

Subnet: AKS

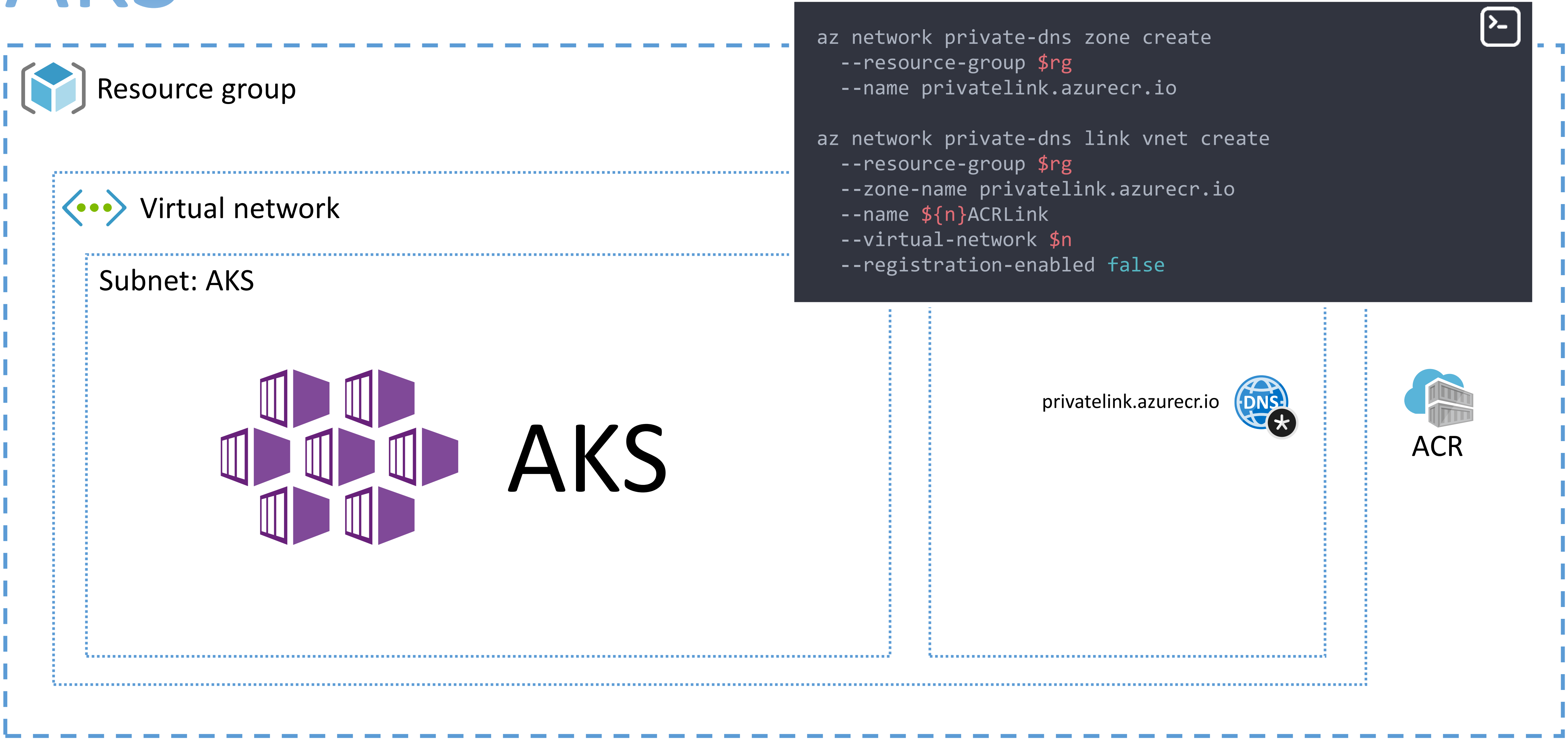
AKS

Subnet: private_endpoints

privatelink.azurecr.io

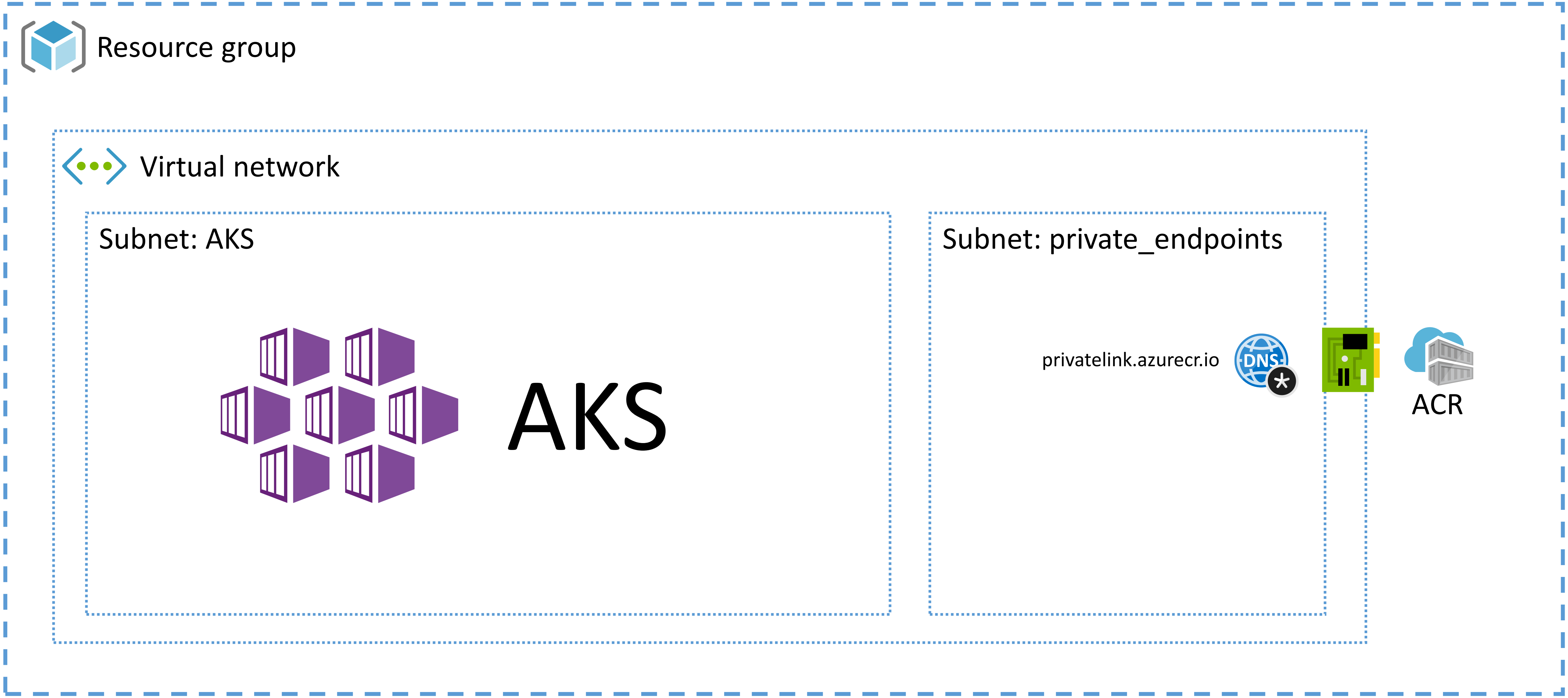
ACR

AKS

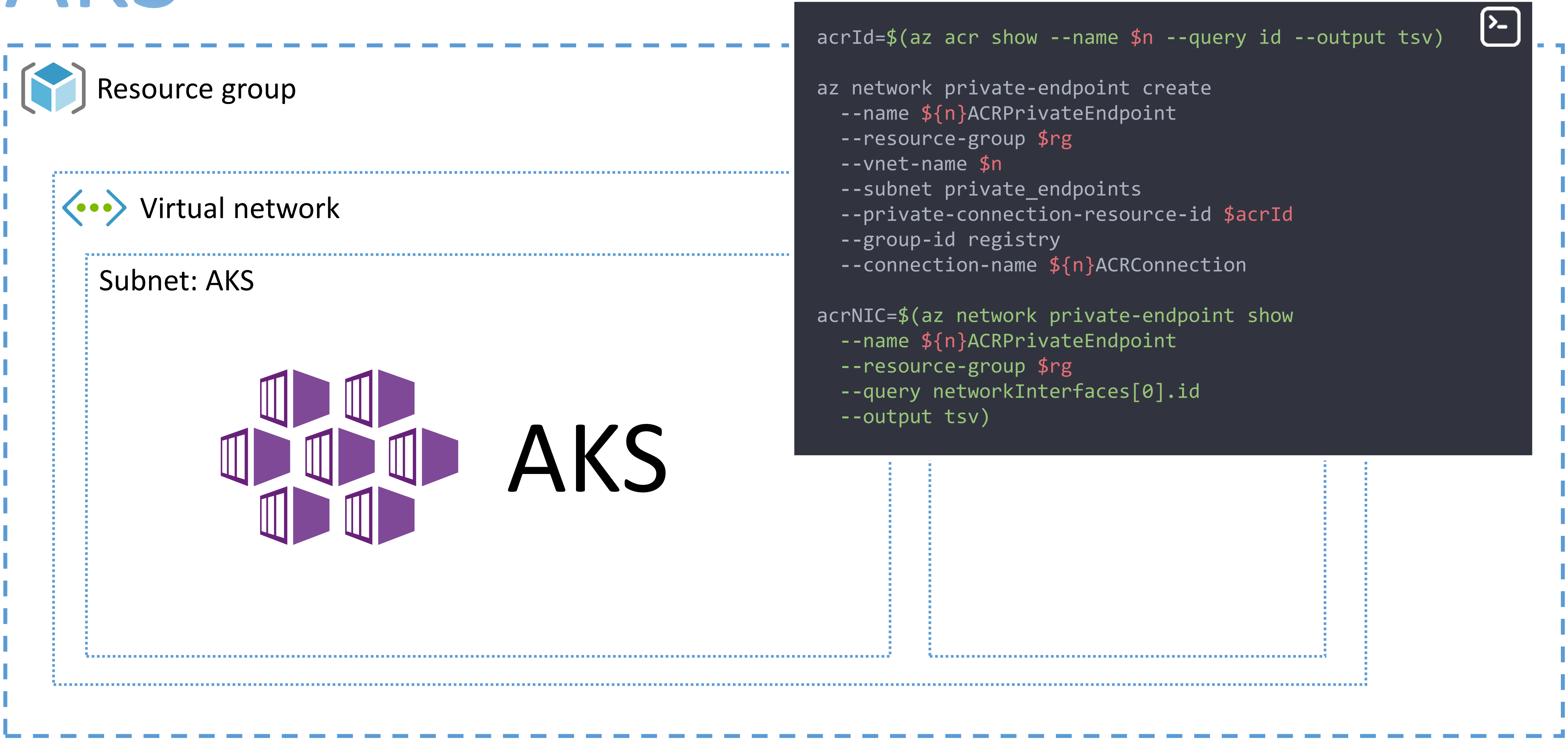


```
az network private-dns zone create
  --resource-group $rg
  --name privatelink.azurecr.io

az network private-dns link vnet create
  --resource-group $rg
  --zone-name privatelink.azurecr.io
  --name ${n}ACRLink
  --virtual-network $n
  --registration-enabled false
```

AKS

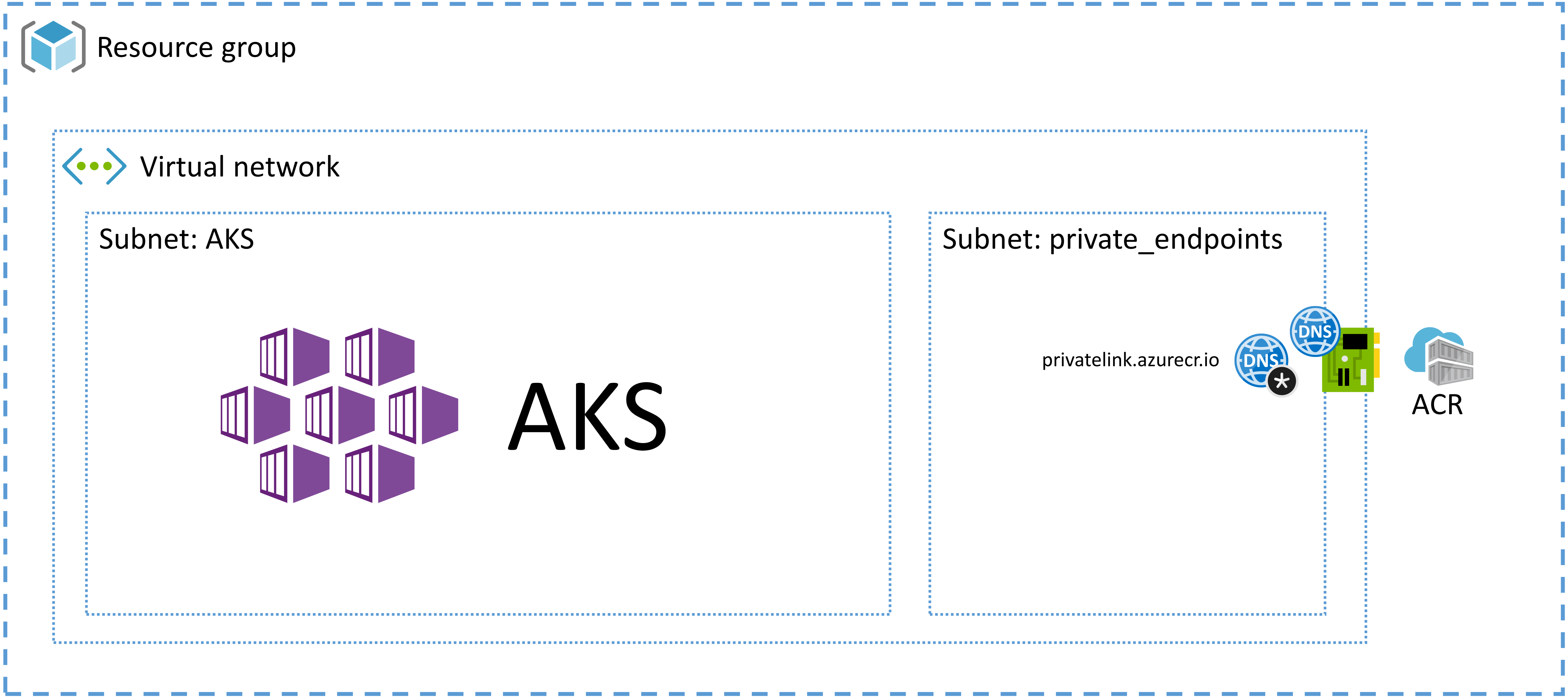

Resource group
Virtual network
Subnet: AKS
AKS
Subnet: private_endpoints
privatelink.azurecr.io
DNS
ACR

AKS

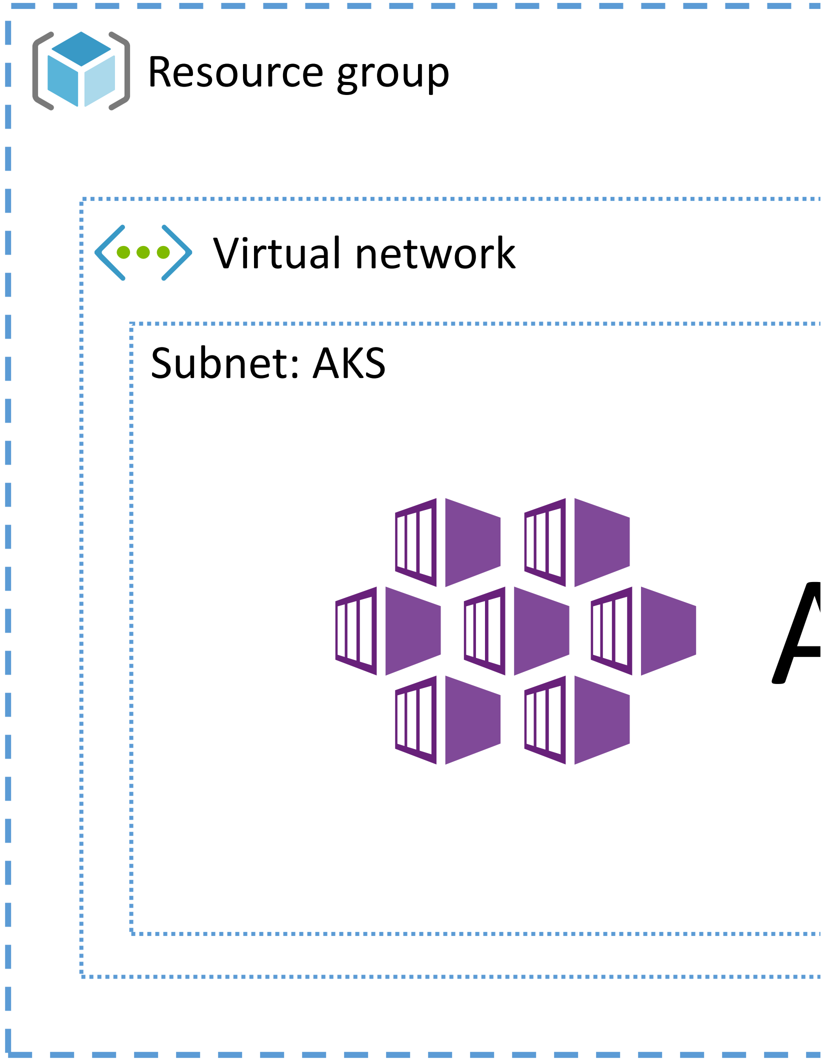


```
acrId=$(az acr show --name $n --query id --output tsv)

az network private-endpoint create
  --name ${n}ACRPrivateEndpoint
  --resource-group $rg
  --vnet-name $n
  --subnet private_endpoints
  --private-connection-resource-id $acrId
  --group-id registry
  --connection-name ${n}ACRConnection

acrNIC=$(az network private-endpoint show
  --name ${n}ACRPrivateEndpoint
  --resource-group $rg
  --query networkInterfaces[0].id
  --output tsv)
```

AKS

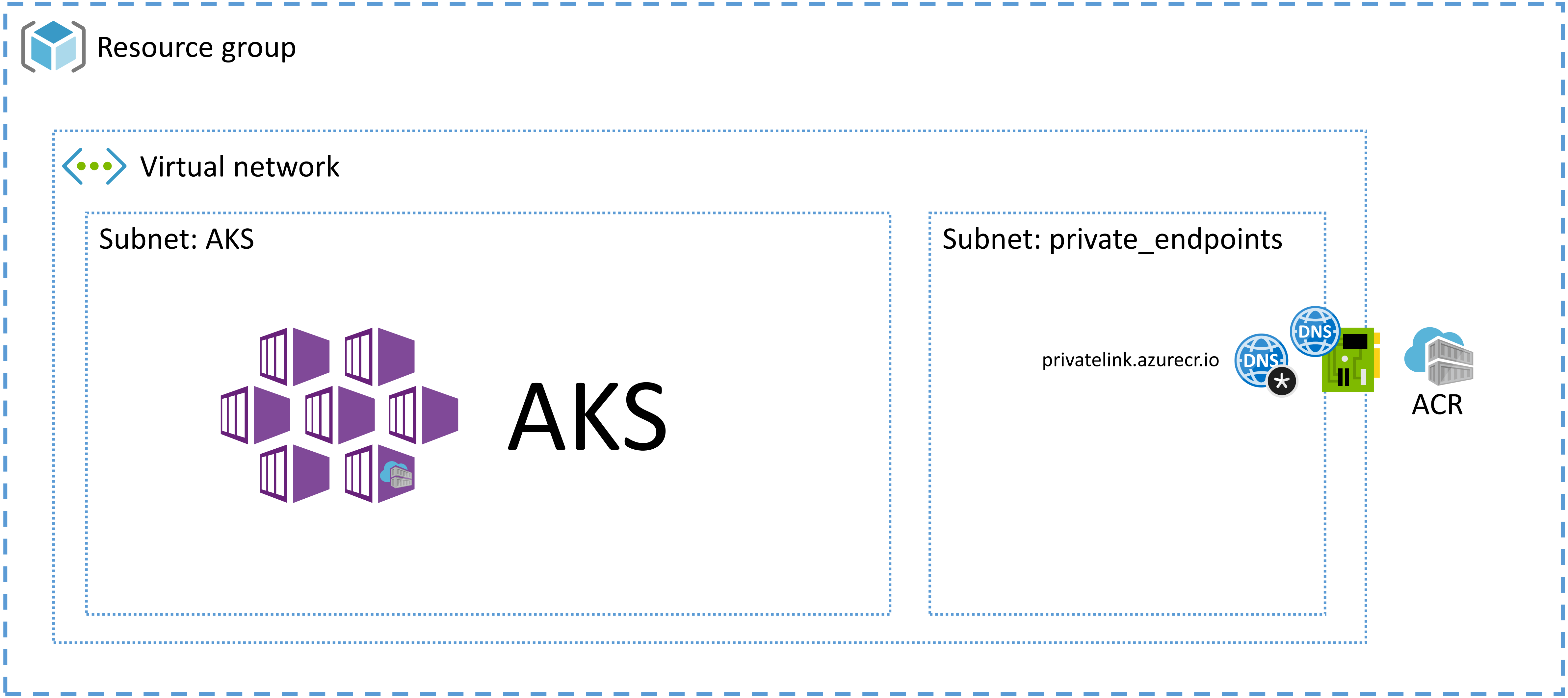

Resource group
Virtual network
Subnet: AKS
AKS
Subnet: private_endpoints
privatelink.azurecr.io
DNS
DNS
ACR

AKS



```
acrIP1=$(az resource show
  --ids $acrNIC
  --api-version 2019-04-01
  --query properties.ipConfigurations[1].properties.privateIPAddress
  --output tsv)

acrIP2=$(az resource show
  --ids $acrNIC
  --api-version 2019-04-01
  --query properties.ipConfigurations[0].properties.privateIPAddress
  --output tsv)

az network private-dns record-set a create
  --name $n
  --zone-name privatelink.azurecr.io
  --resource-group $rg

az network private-dns record-set a create
  --name $n.westeurope.data
  --zone-name privatelink.azurecr.io
  --resource-group $rg

az network private-dns record-set a add-record
  --record-set-name $n
  --zone-name privatelink.azurecr.io
  --resource-group $rg
  --ipv4-address $acrIP1

az network private-dns record-set a add-record
  --record-set-name $n.westeurope.data
  --zone-name privatelink.azurecr.io
  --resource-group $rg
  --ipv4-address $acrIP2
```

AKS



Resource group

Virtual network

Subnet: AKS

AKS

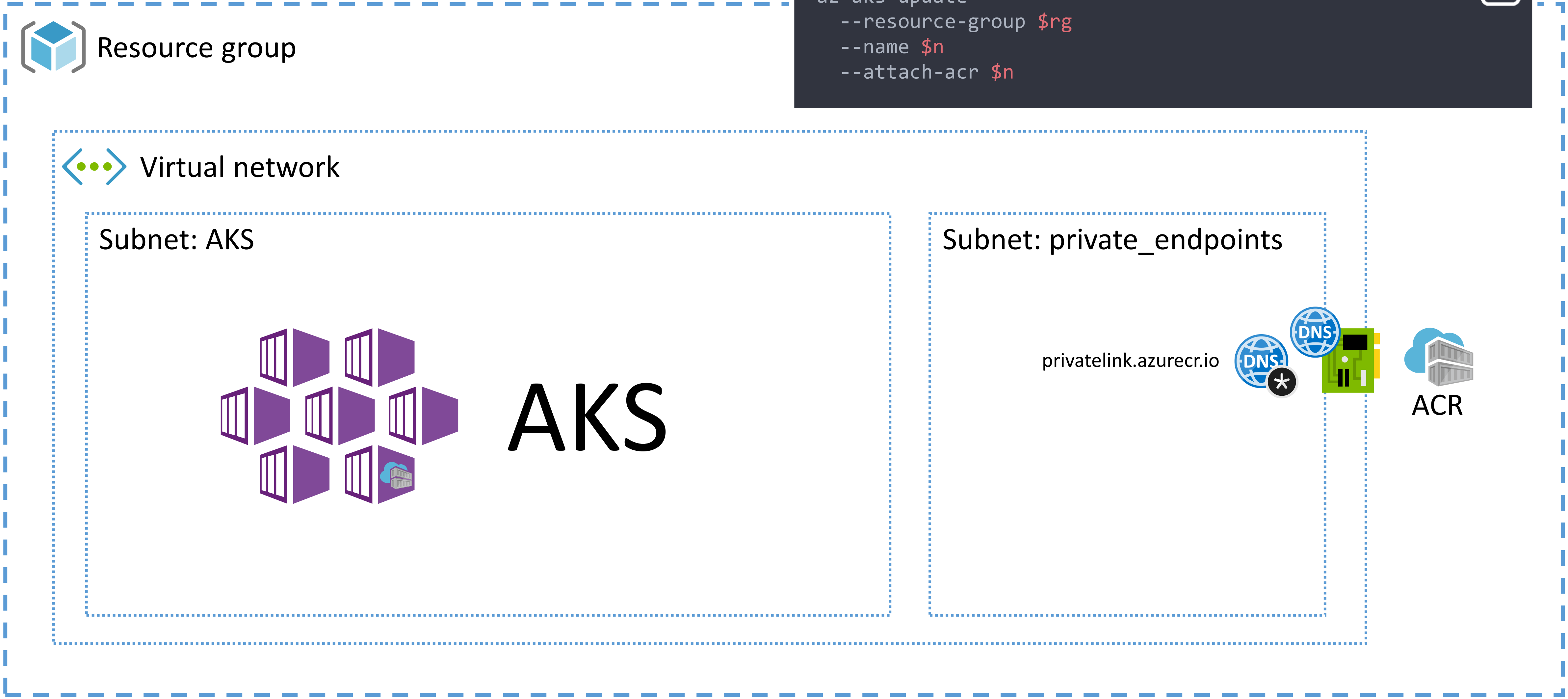
Subnet: private_endpoints

privatelink.azurecr.io

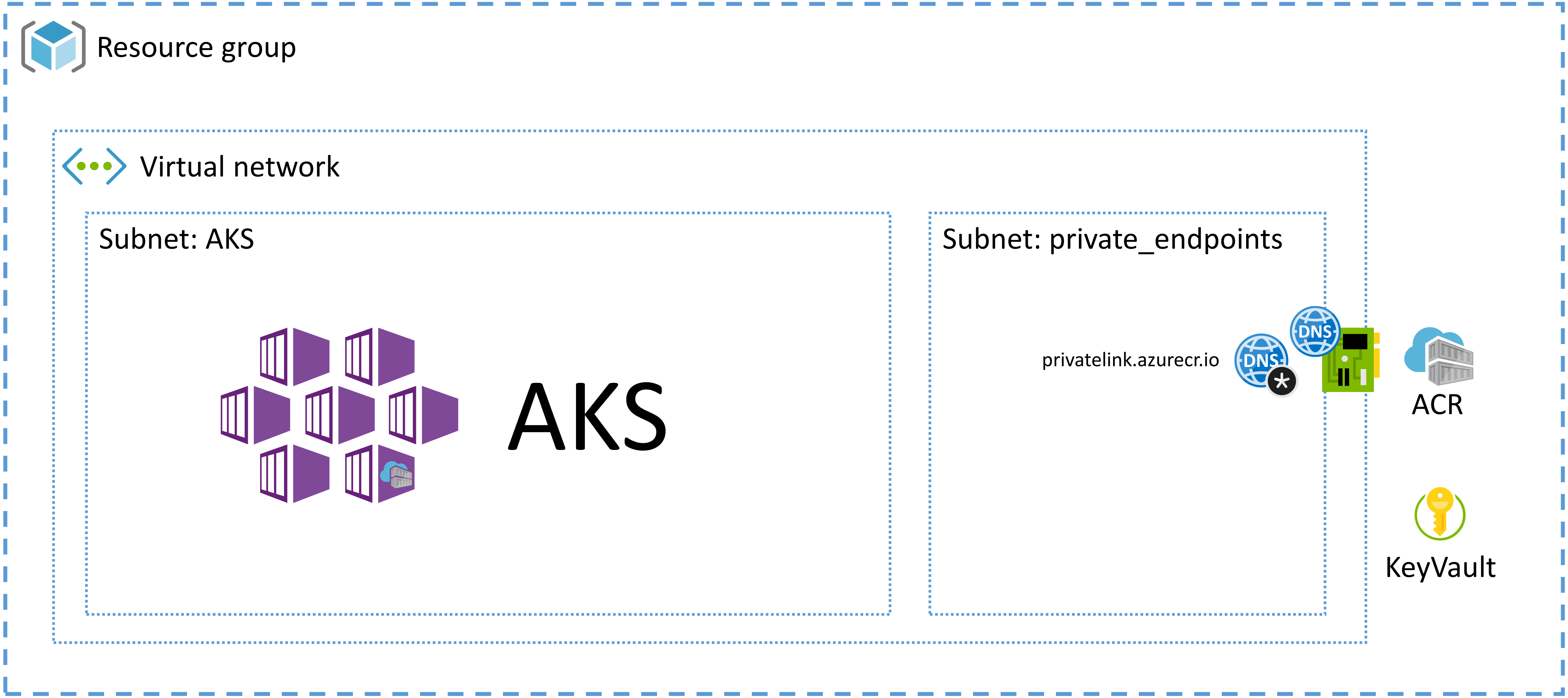
ACR

AKS

```
az aks update
  --resource-group $rg
  --name $n
  --attach-acr $n
```



AKS



Resource group

Virtual network

Subnet: AKS

AKS

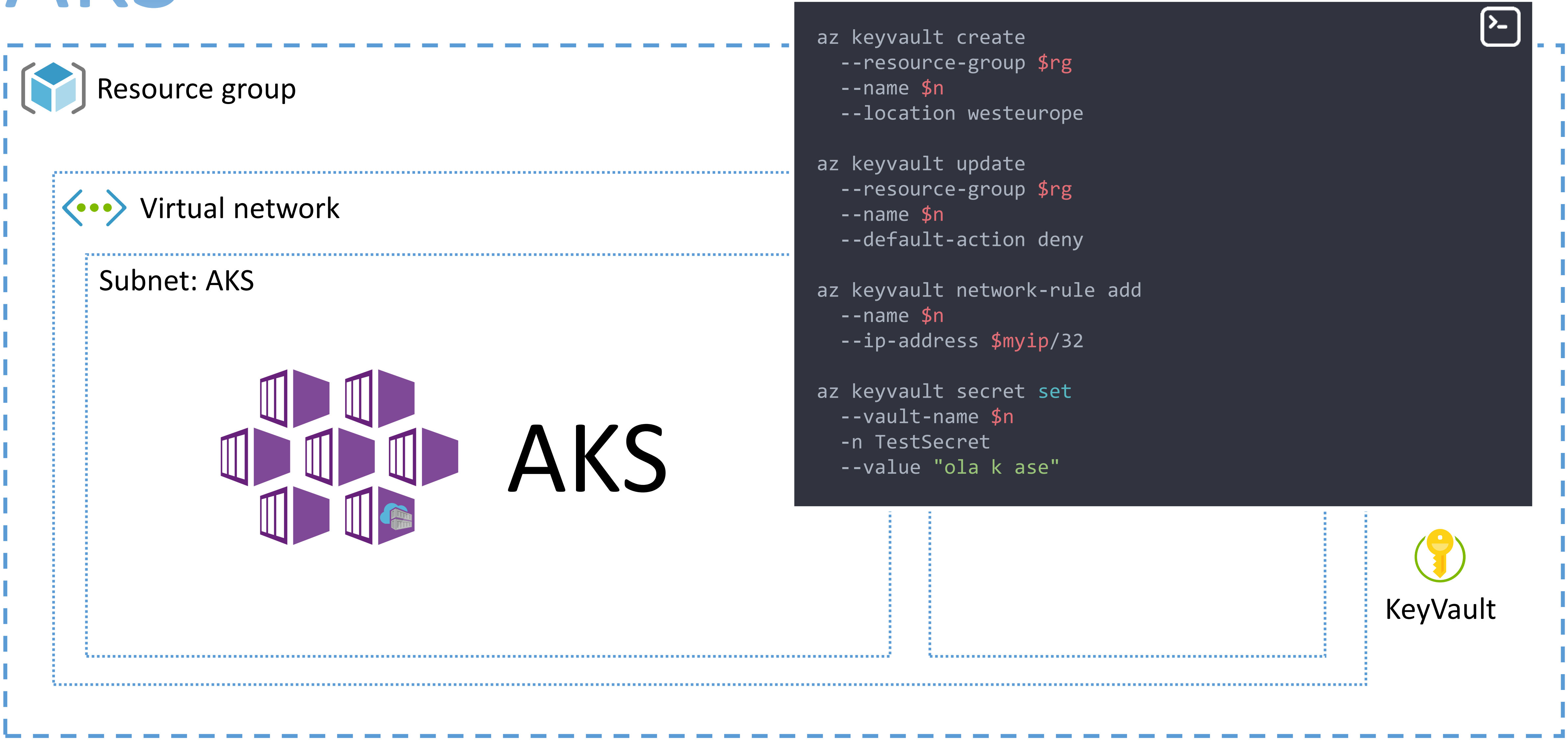
Subnet: private_endpoints

privatelink.azurecr.io

ACR

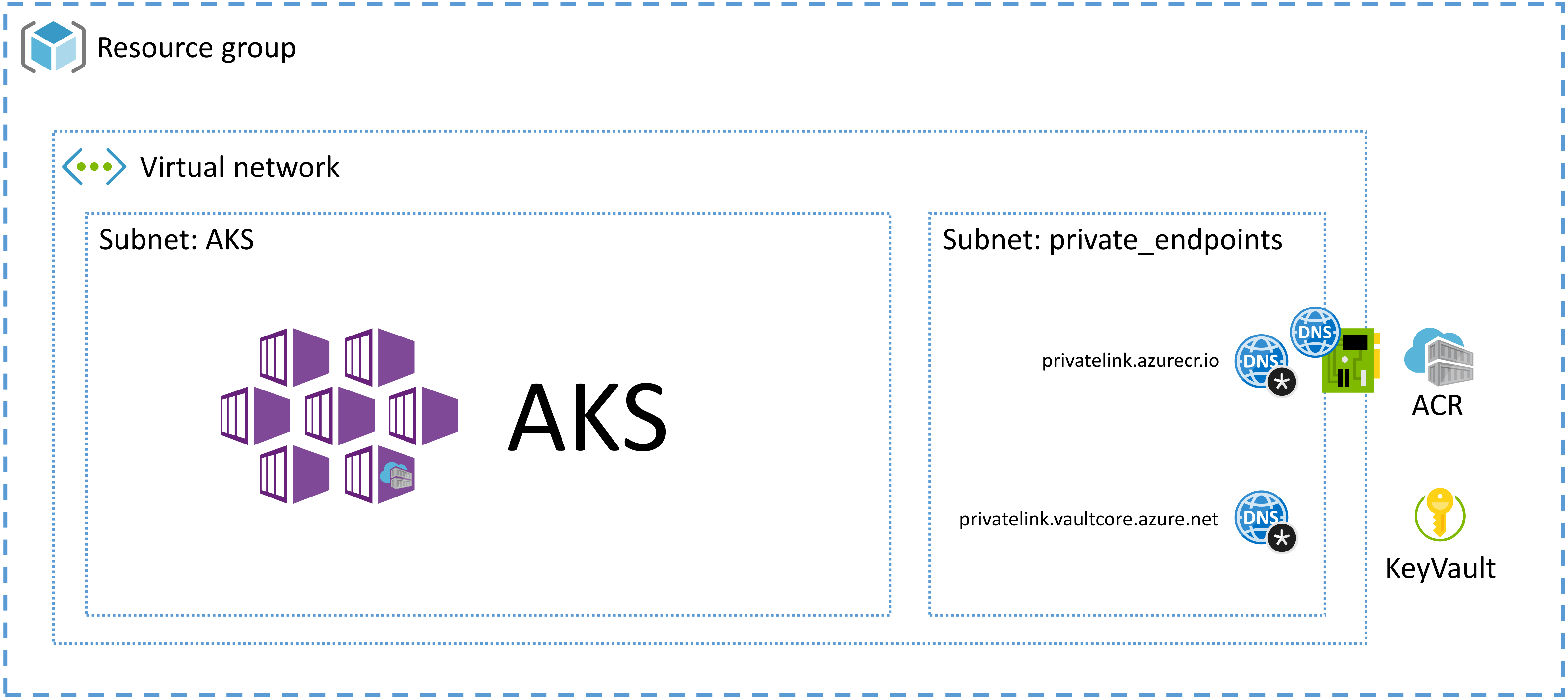
KeyVault

AKS



```
az keyvault create
  --resource-group $rg
  --name $n
  --location westeurope

az keyvault update
  --resource-group $rg
  --name $n
  --default-action deny

az keyvault network-rule add
  --name $n
  --ip-address $myip/32

az keyvault secret set
  --vault-name $n
  -n TestSecret
  --value "ola k ase"
```

AKS



Resource group

Virtual network

Subnet: AKS

AKS

Subnet: private_endpoints

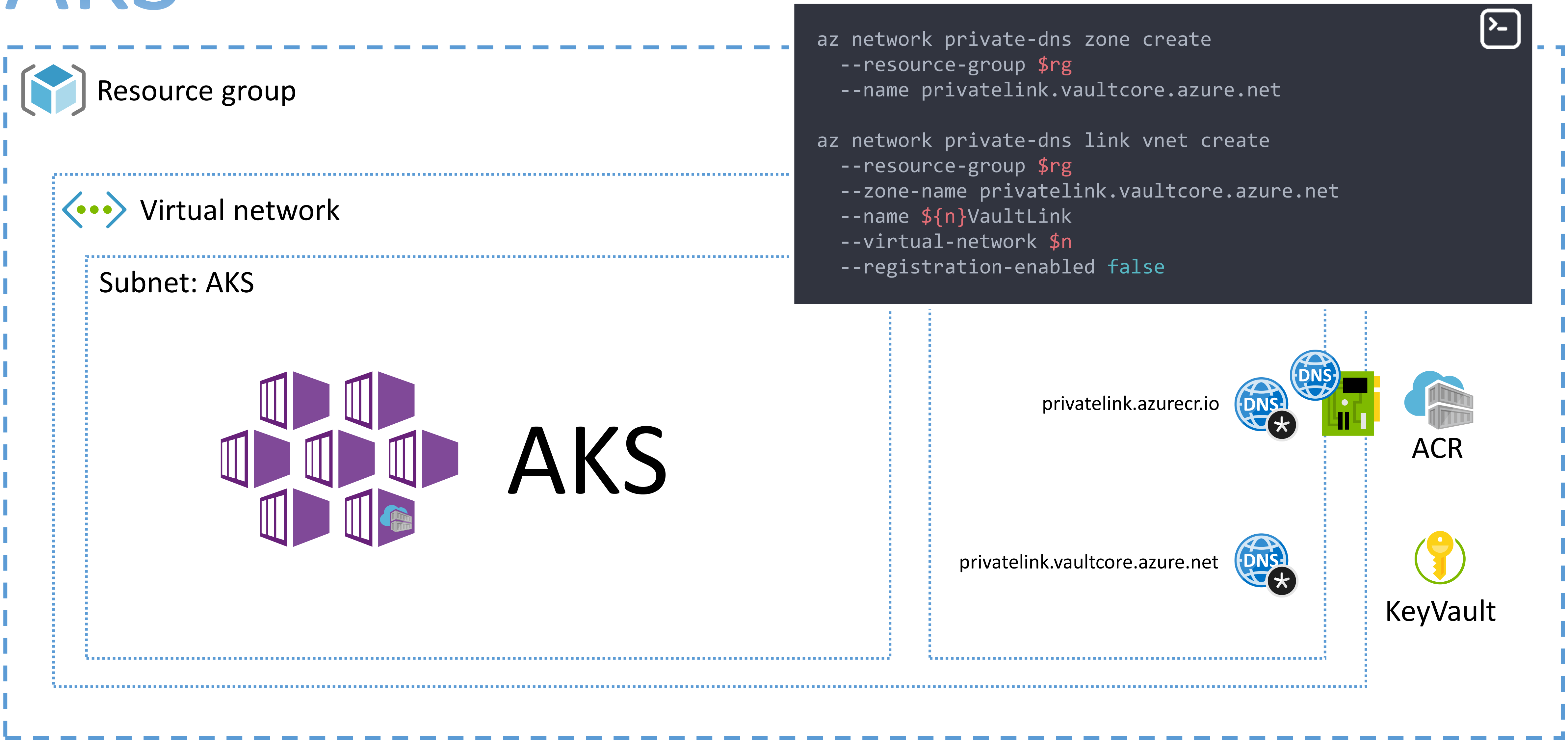
privatelink.azurecr.io

privatelink.vaultcore.azure.net

ACR

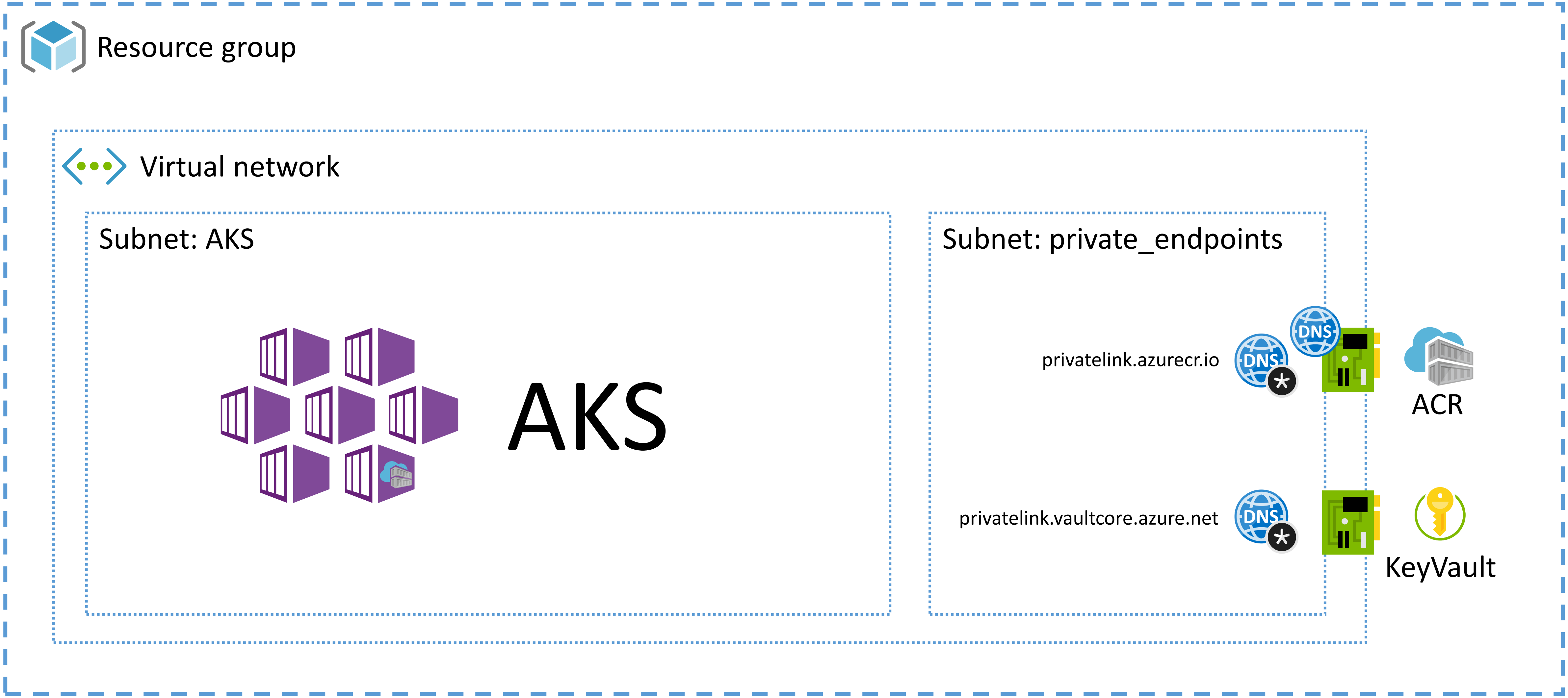
KeyVault

AKS

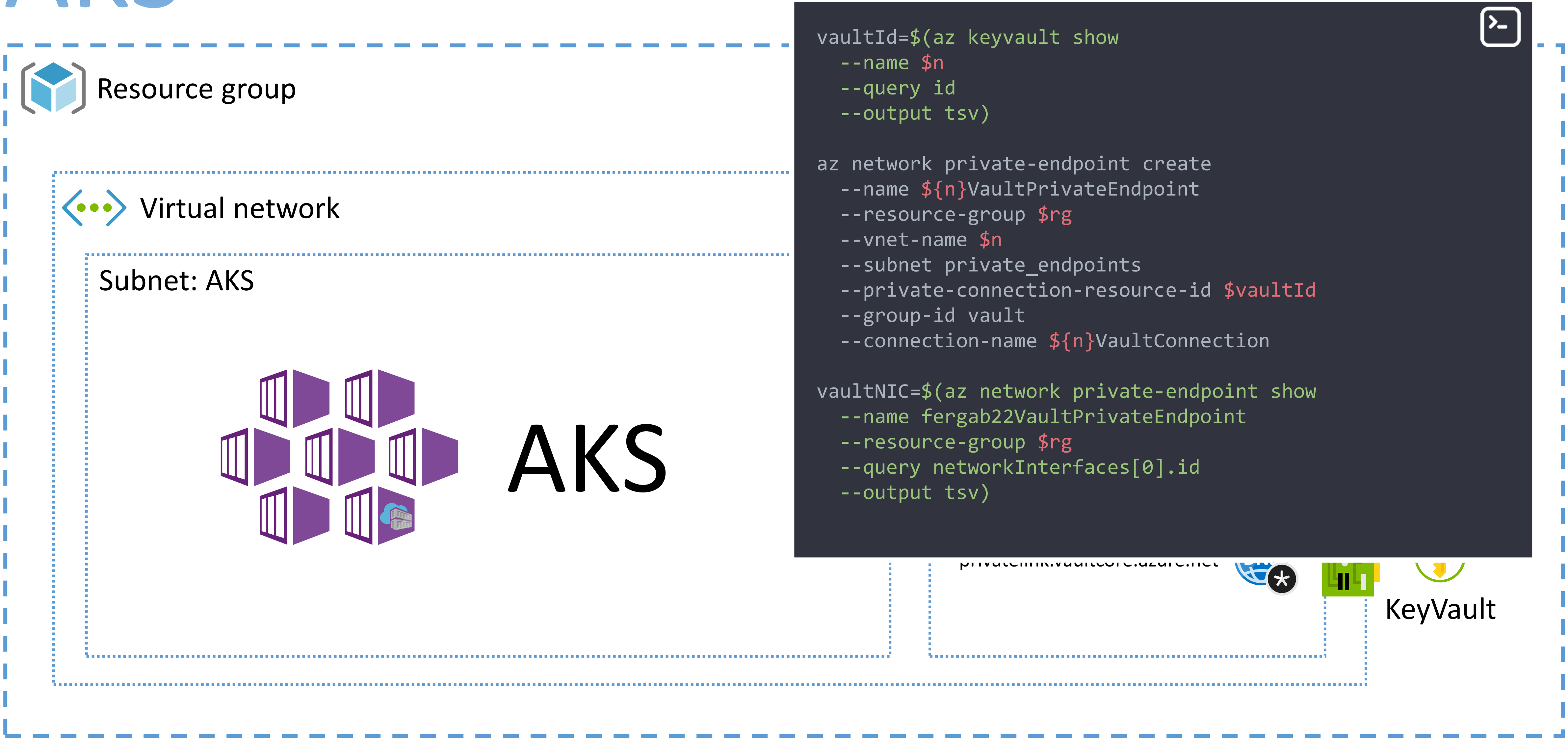


```
az network private-dns zone create
  --resource-group $rg
  --name privatelink.vaultcore.azure.net

az network private-dns link vnet create
  --resource-group $rg
  --zone-name privatelink.vaultcore.azure.net
  --name ${n}VaultLink
  --virtual-network $n
  --registration-enabled false
```

AKS

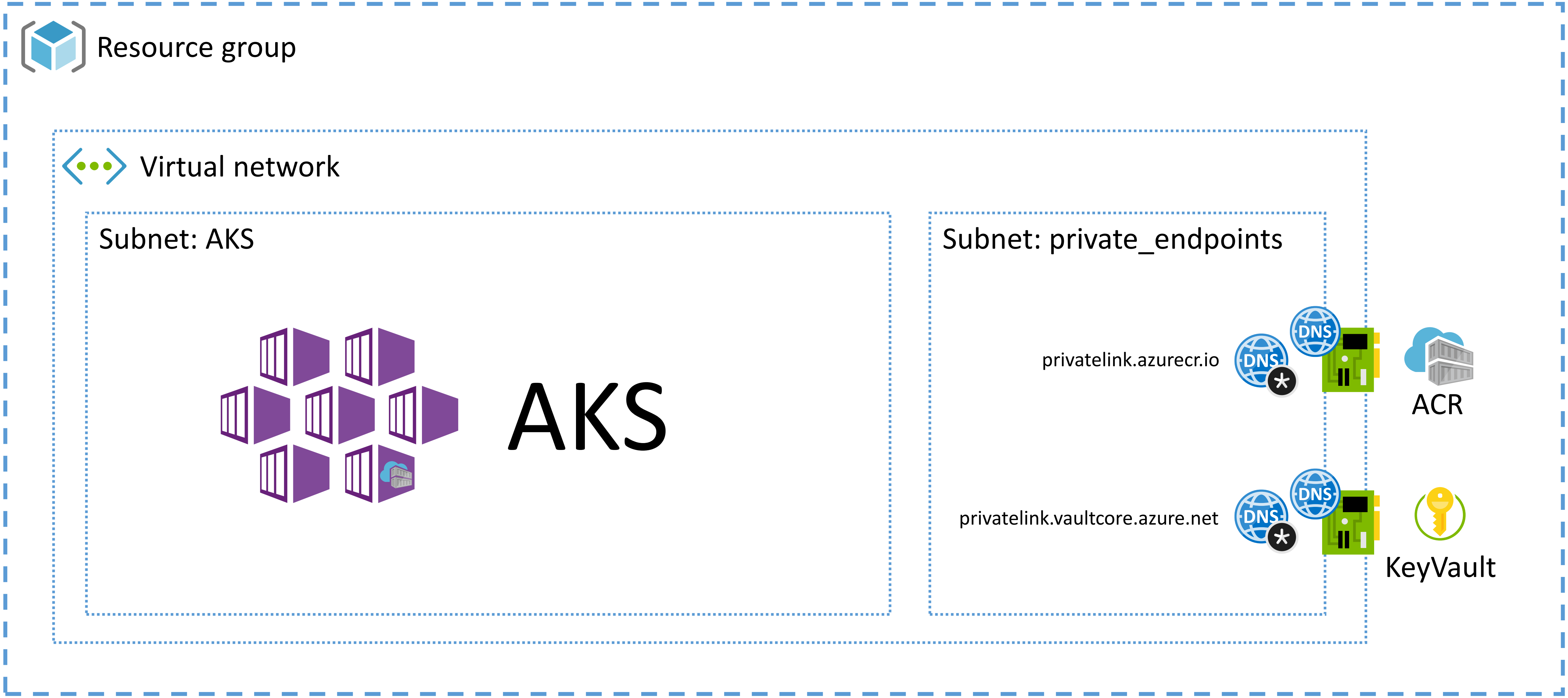

Resource group
Virtual network
Subnet: AKS
AKS
Subnet: private_endpoints
privatelink.azurecr.io
DNS
DNS
ACR
privatelink.vaultcore.azure.net
DNS
KeyVault

AKS

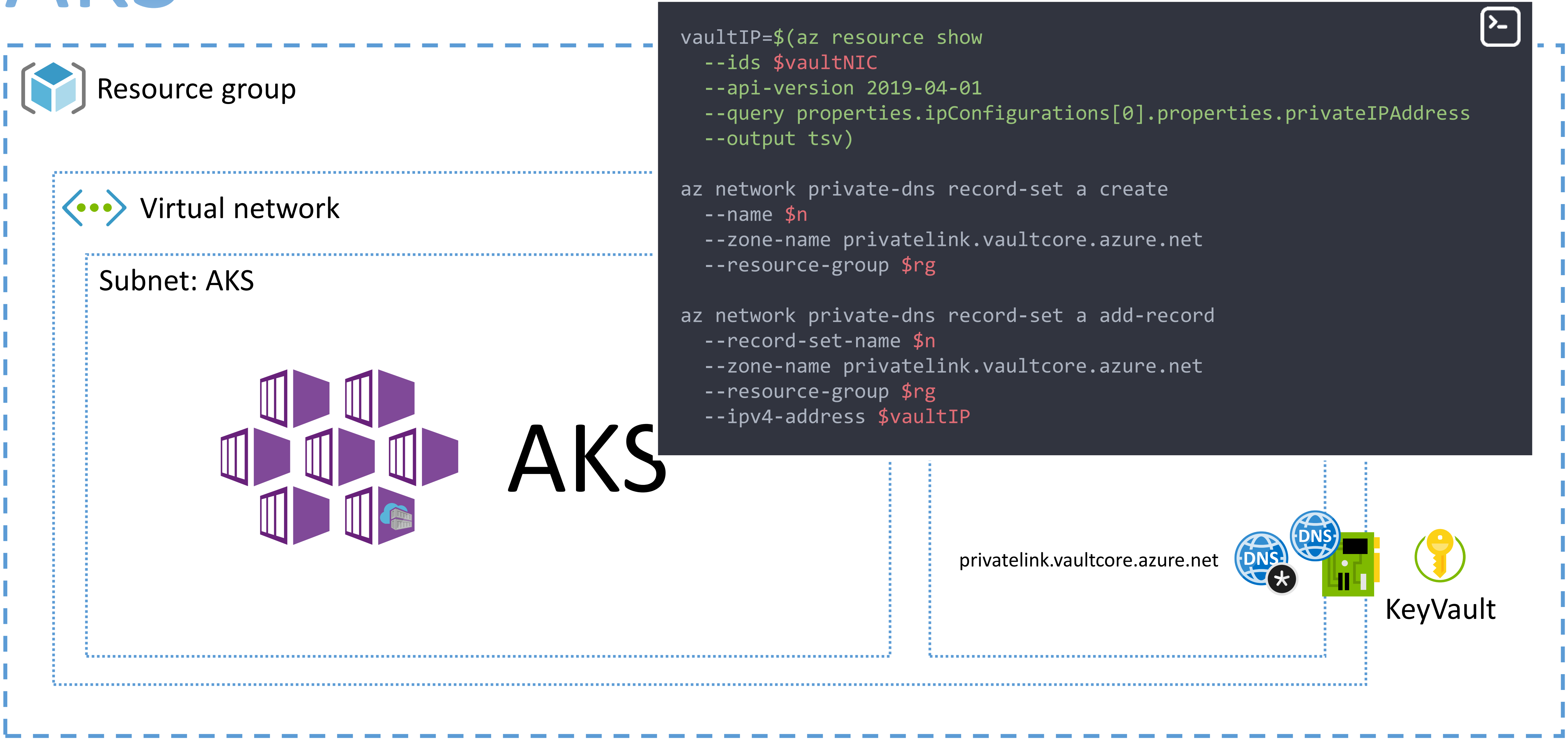


```
vaultId=$(az keyvault show
  --name $n
  --query id
  --output tsv)

az network private-endpoint create
  --name ${n}VaultPrivateEndpoint
  --resource-group $rg
  --vnet-name $n
  --subnet private_endpoints
  --private-connection-resource-id $vaultId
  --group-id vault
  --connection-name ${n}VaultConnection

vaultNIC=$(az network private-endpoint show
  --name fergab22VaultPrivateEndpoint
  --resource-group $rg
  --query networkInterfaces[0].id
  --output tsv)
```

AKS

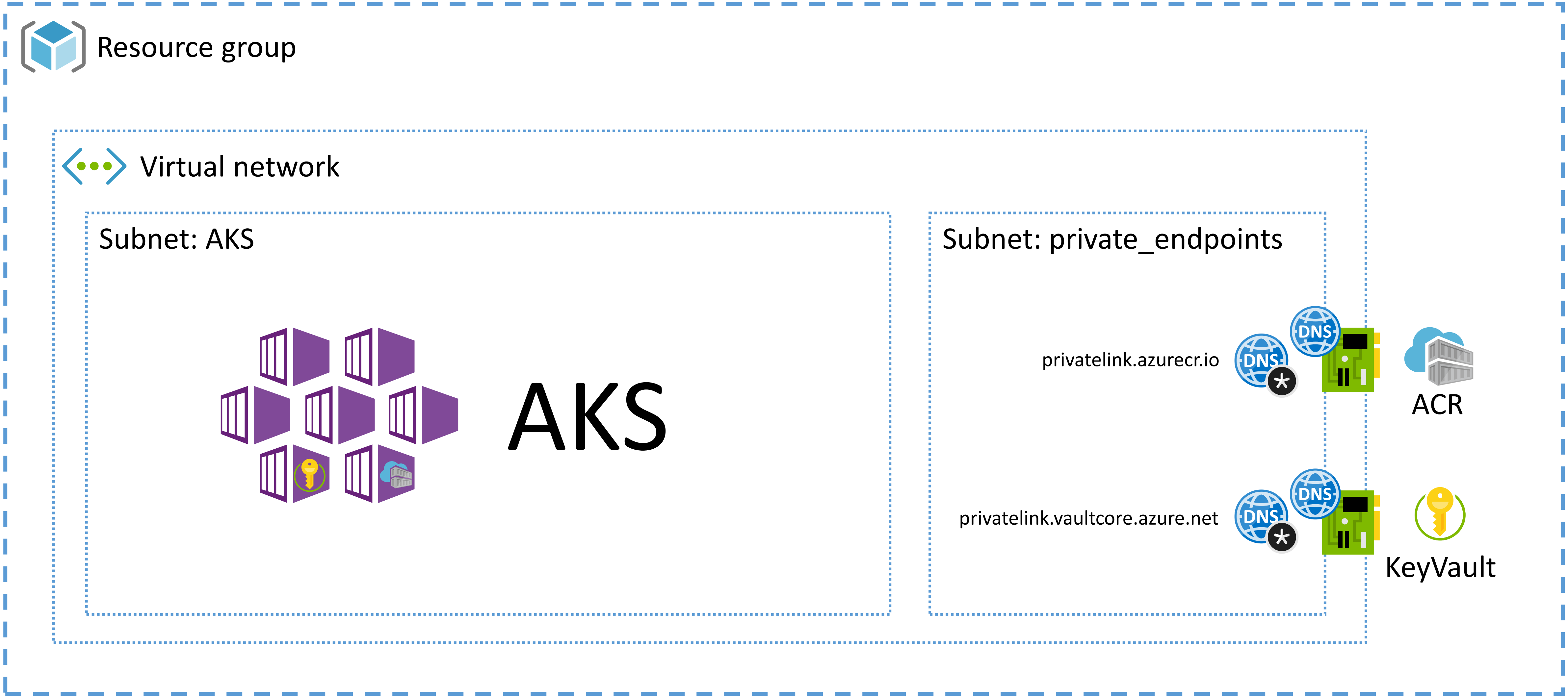

Resource group
Virtual network
Subnet: AKS
AKS
Subnet: private_endpoints
privatelink.azurecr.io
DNS
DNS
ACR
privatelink.vaultcore.azure.net
DNS
DNS
KeyVault

AKS



```
vaultIP=$(az resource show
  --ids $vaultNIC
  --api-version 2019-04-01
  --query properties.ipConfigurations[0].properties.privateIPAddress
  --output tsv)

az network private-dns record-set a create
  --name $n
  --zone-name privatelink.vaultcore.azure.net
  --resource-group $rg

az network private-dns record-set a add-record
  --record-set-name $n
  --zone-name privatelink.vaultcore.azure.net
  --resource-group $rg
  --ipv4-address $vaultIP
```

AKS



Resource group

Virtual network

Subnet: AKS

AKS

Subnet: private_endpoints

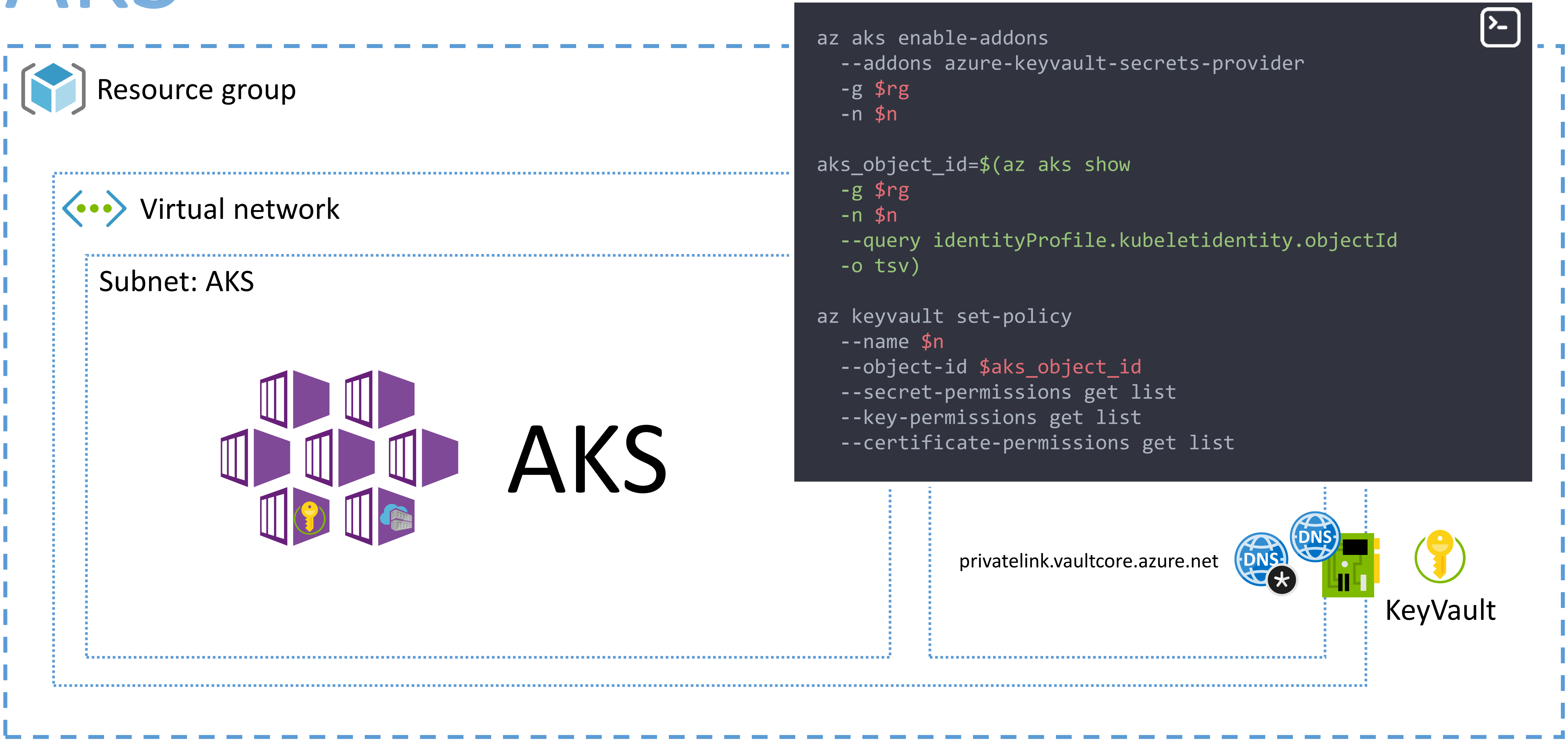
privatelink.azurecr.io

ACR

privatelink.vaultcore.azure.net

KeyVault

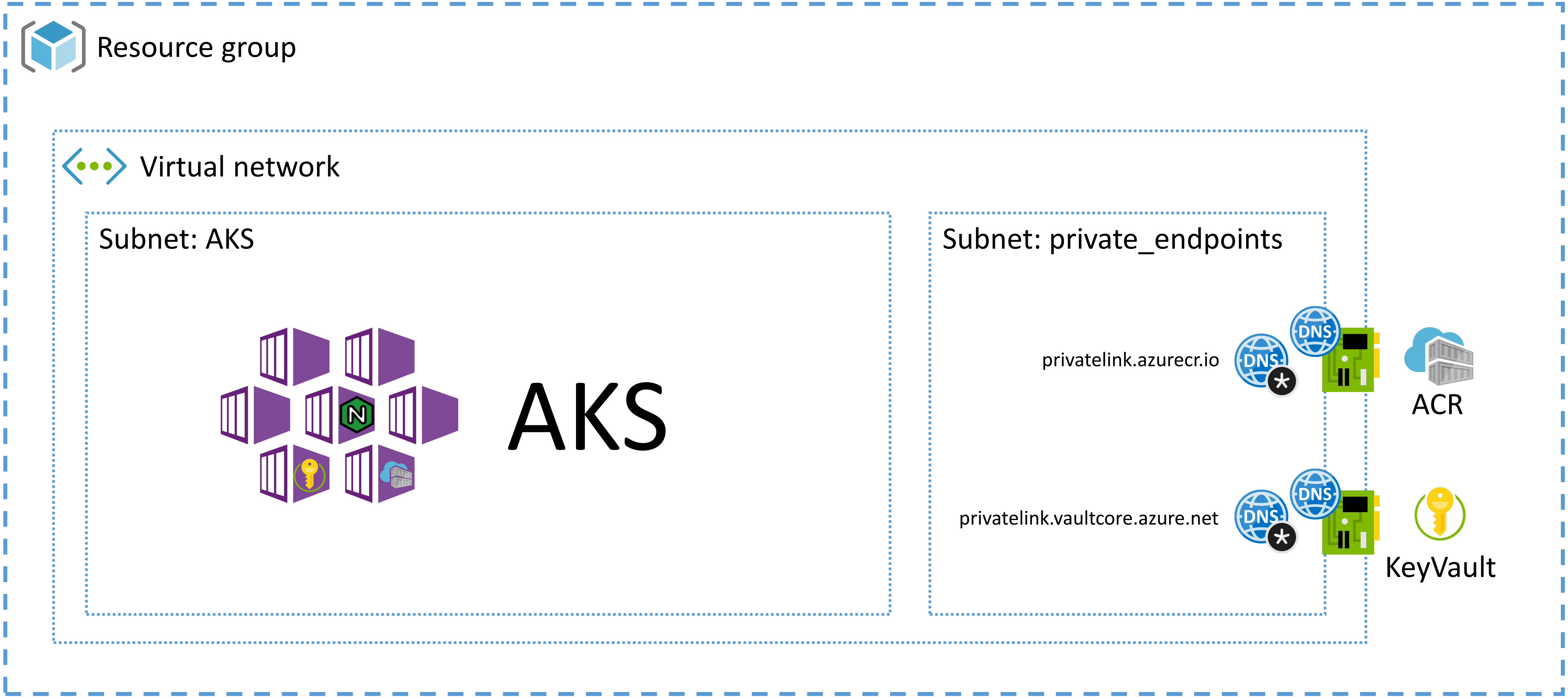
AKS



```
az aks enable-addons
  --addons azure-keyvault-secrets-provider
  -g $rg
  -n $n

aks_object_id=$(az aks show
  -g $rg
  -n $n
  --query identityProfile.kubeletidentity.objectId
  -o tsv)

az keyvault set-policy
  --name $n
  --object-id $aks_object_id
  --secret-permissions get list
  --key-permissions get list
  --certificate-permissions get list
```

```
az aks get-credentials -n $n -g $rg
```

AKS



Resource group

Virtual network

Subnet: AKS

AKS

Subnet: private_endpoints

privatelink.azurecr.io

DNS

DNS

ACR

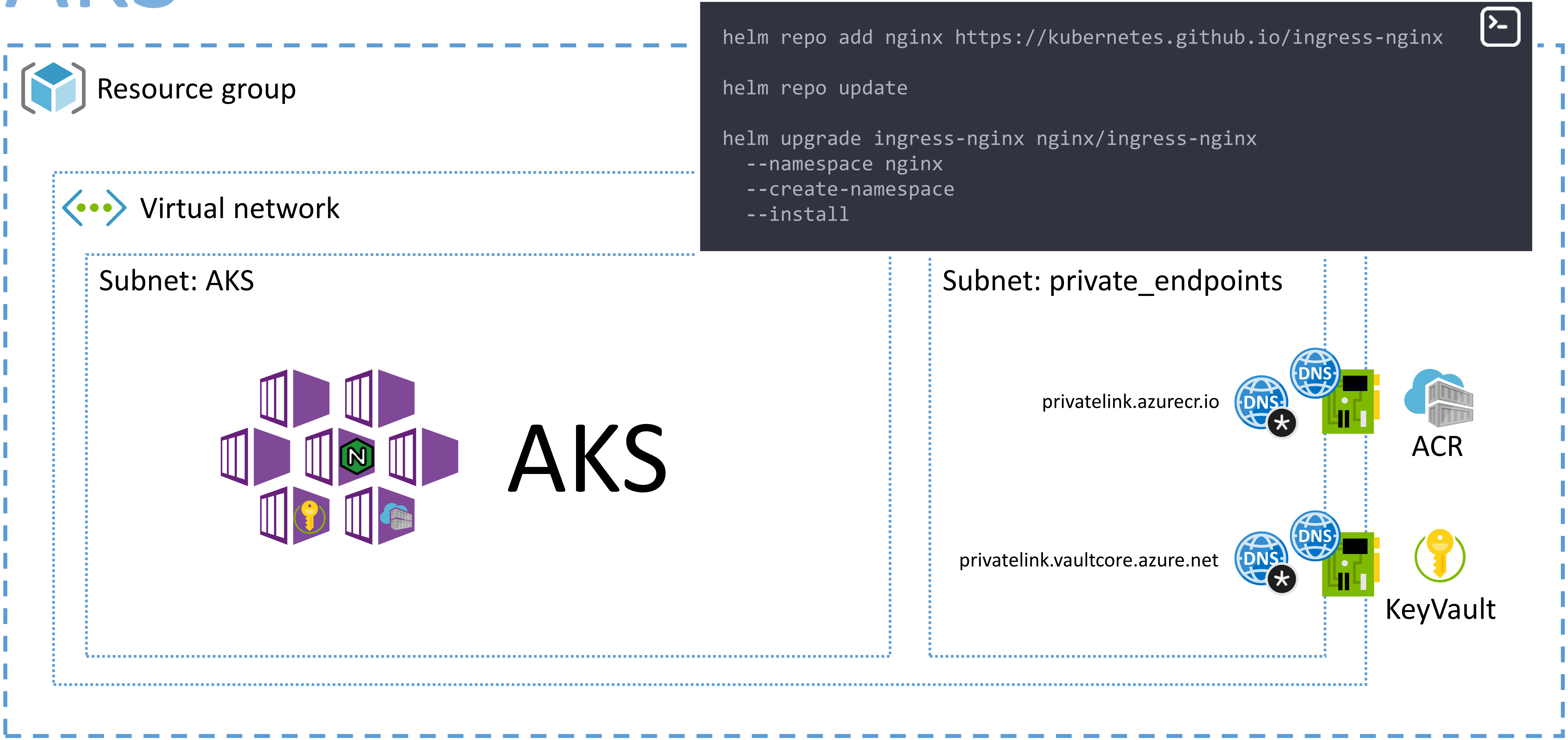
privatelink.vaultcore.azure.net

DNS

DNS

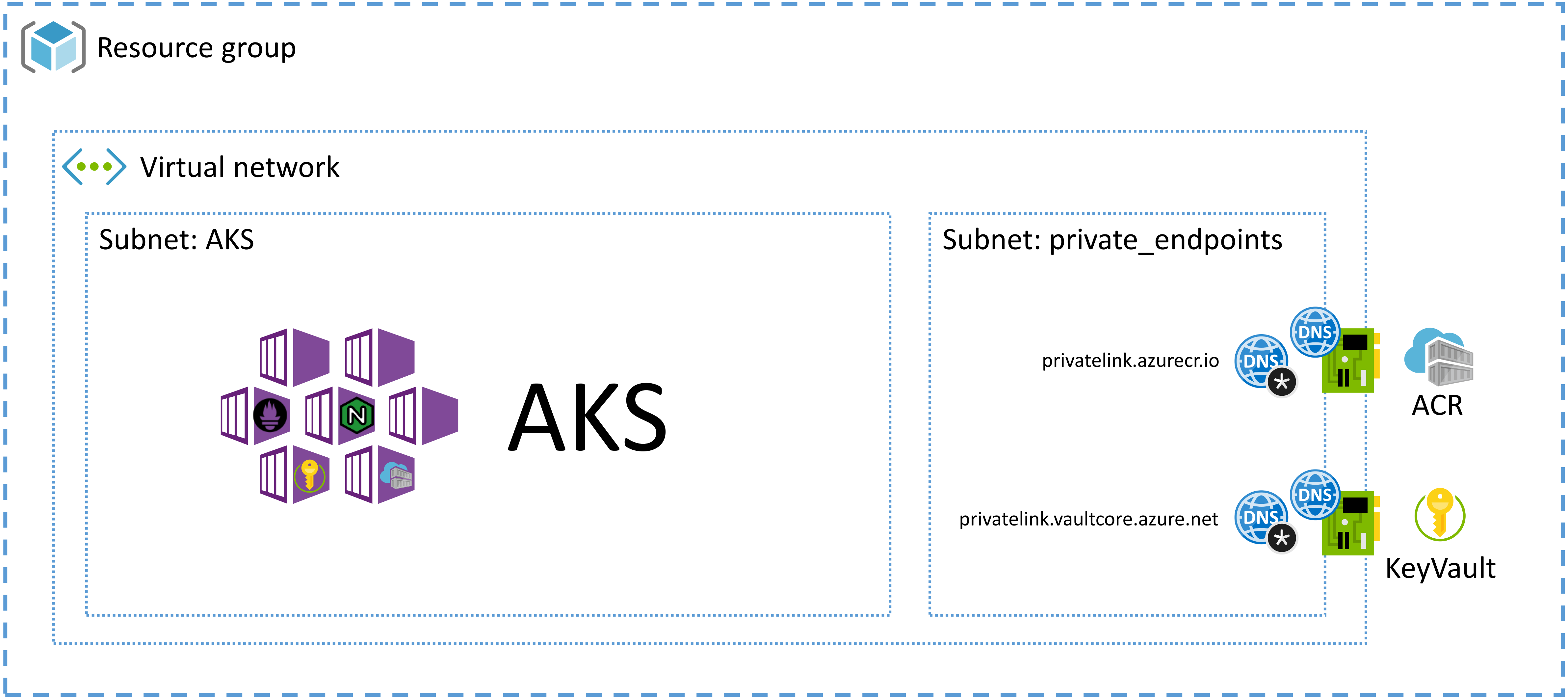
KeyVault

AKS



```
helm repo add nginx https://kubernetes.github.io/ingress-nginx

helm repo update

helm upgrade ingress-nginx nginx/ingress-nginx
  --namespace nginx
  --create-namespace
  --install
```

AKS



Resource group

Virtual network

Subnet: AKS

AKS

Subnet: private_endpoints

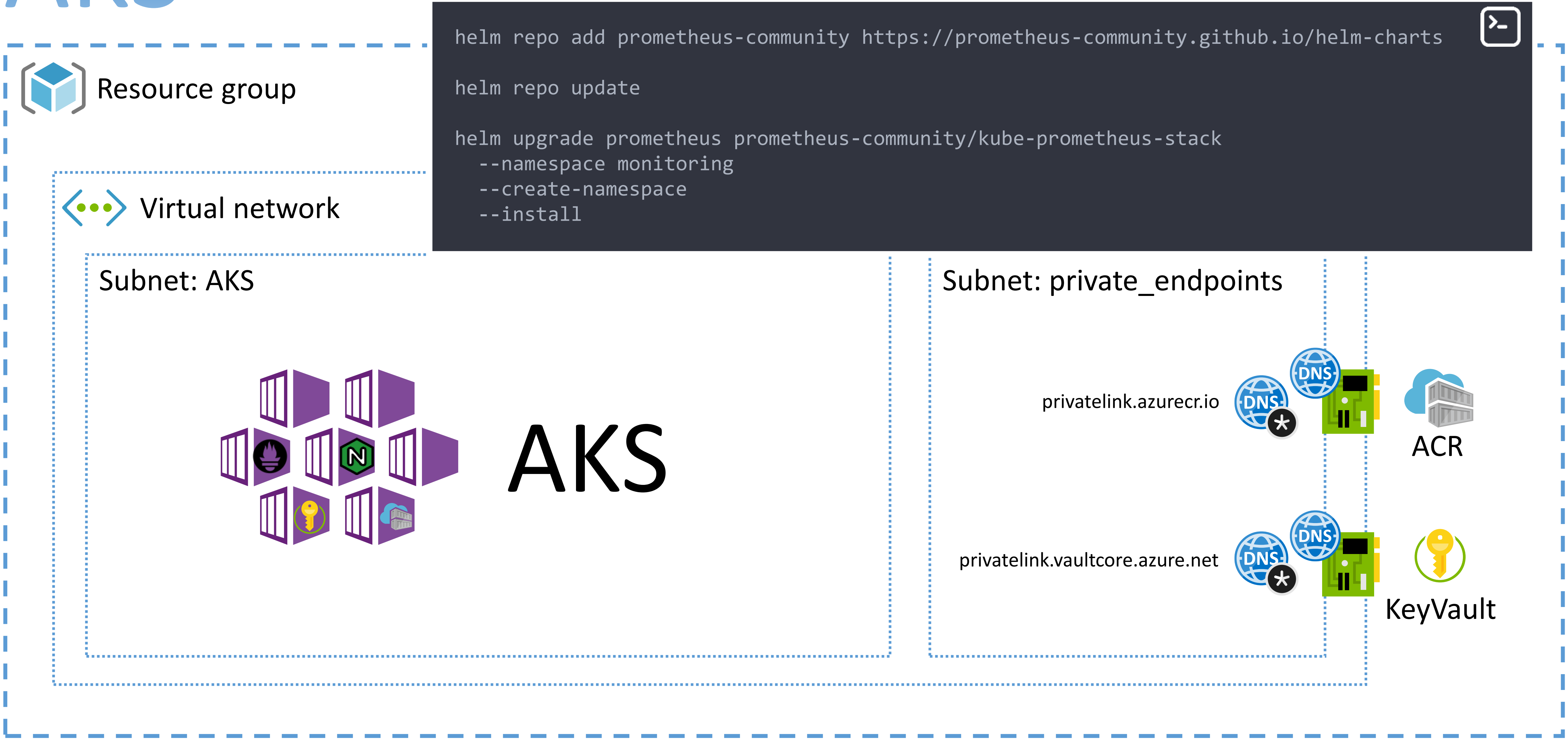
privatelink.azurecr.io

ACR

privatelink.vaultcore.azure.net

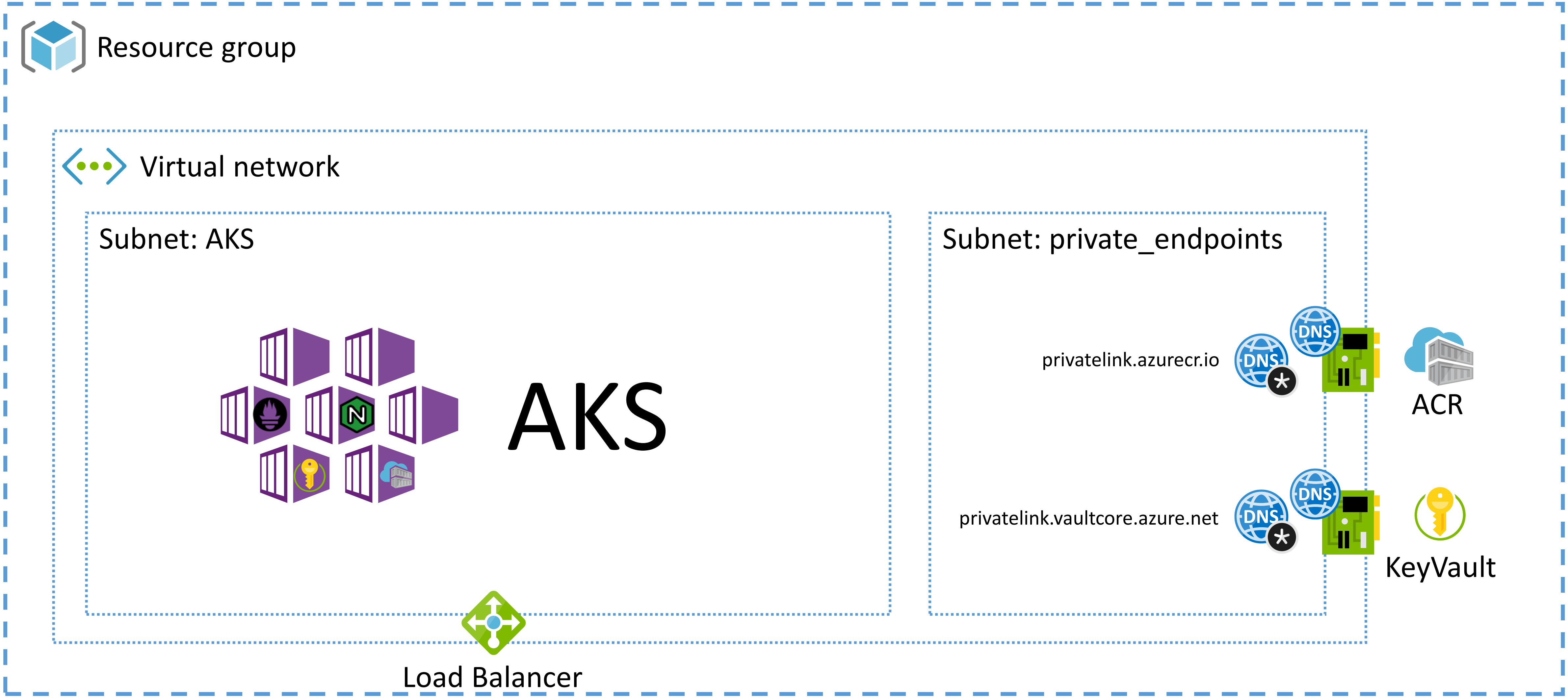
KeyVault

AKS



```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts

helm repo update

helm upgrade prometheus prometheus-community/kube-prometheus-stack
  --namespace monitoring
  --create-namespace
  --install
```

AKS



Resource group

Virtual network

Subnet: AKS

AKS

Subnet: private_endpoints

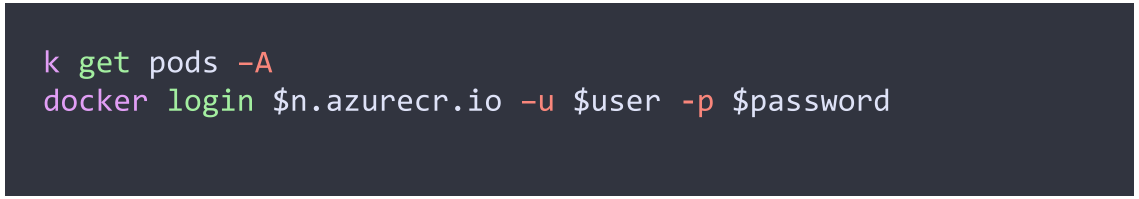
privatelink.azurecr.io

ACR

privatelink.vaultcore.azure.net

KeyVault

Load Balancer



```
k get pods -A
docker login $n.azurecr.io -u $user -p $password
```

K8S Deployment

Container
Port 80

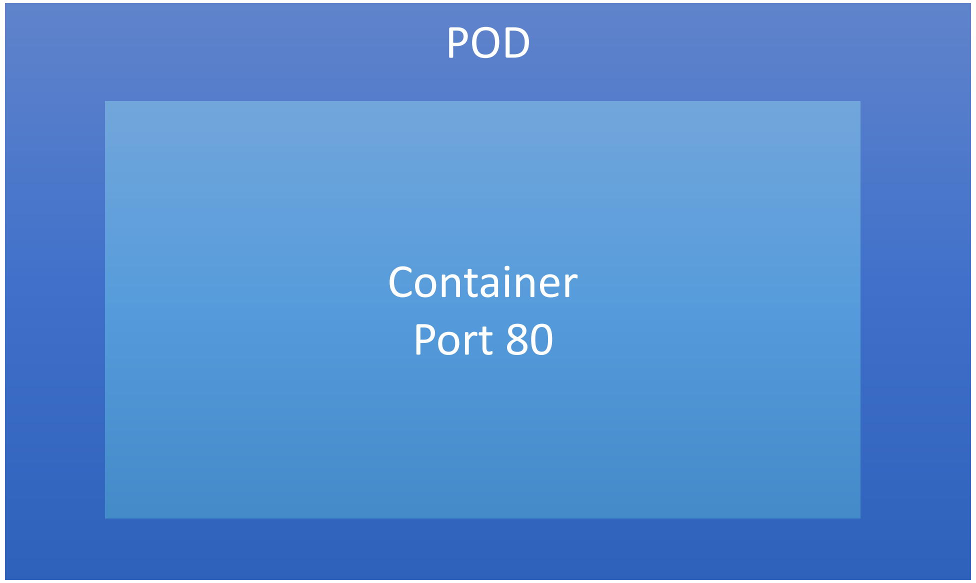
K8S Deployment

demo

```
docker tag myapi:0.1 fergab22.azurecr.io/myapi:0.1
docker push fergab22.azurecr.io/myapi:0.1
```

Container
Port 80

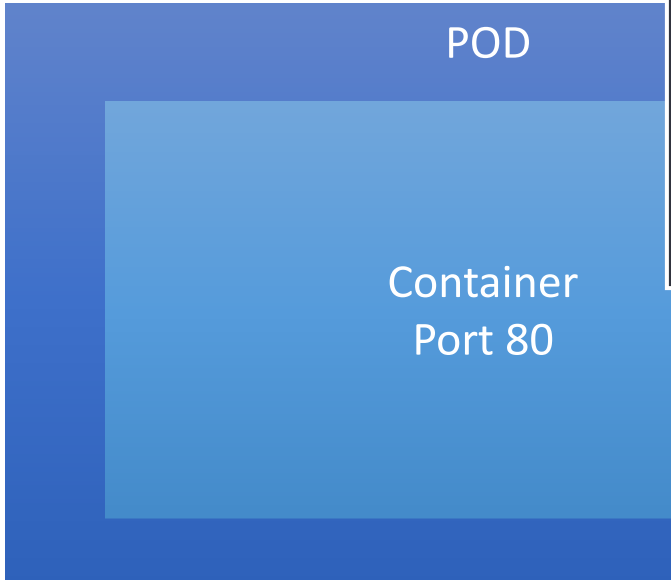
K8S Deployment



POD

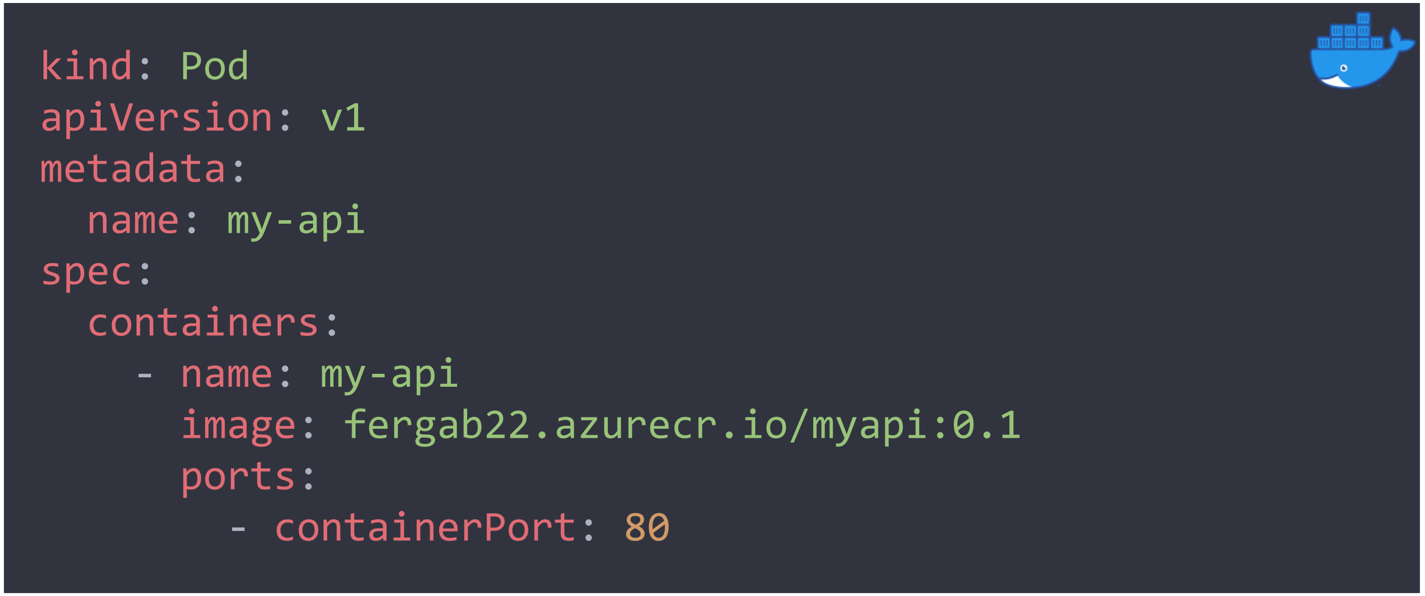
Container
Port 80

K8S Deployment



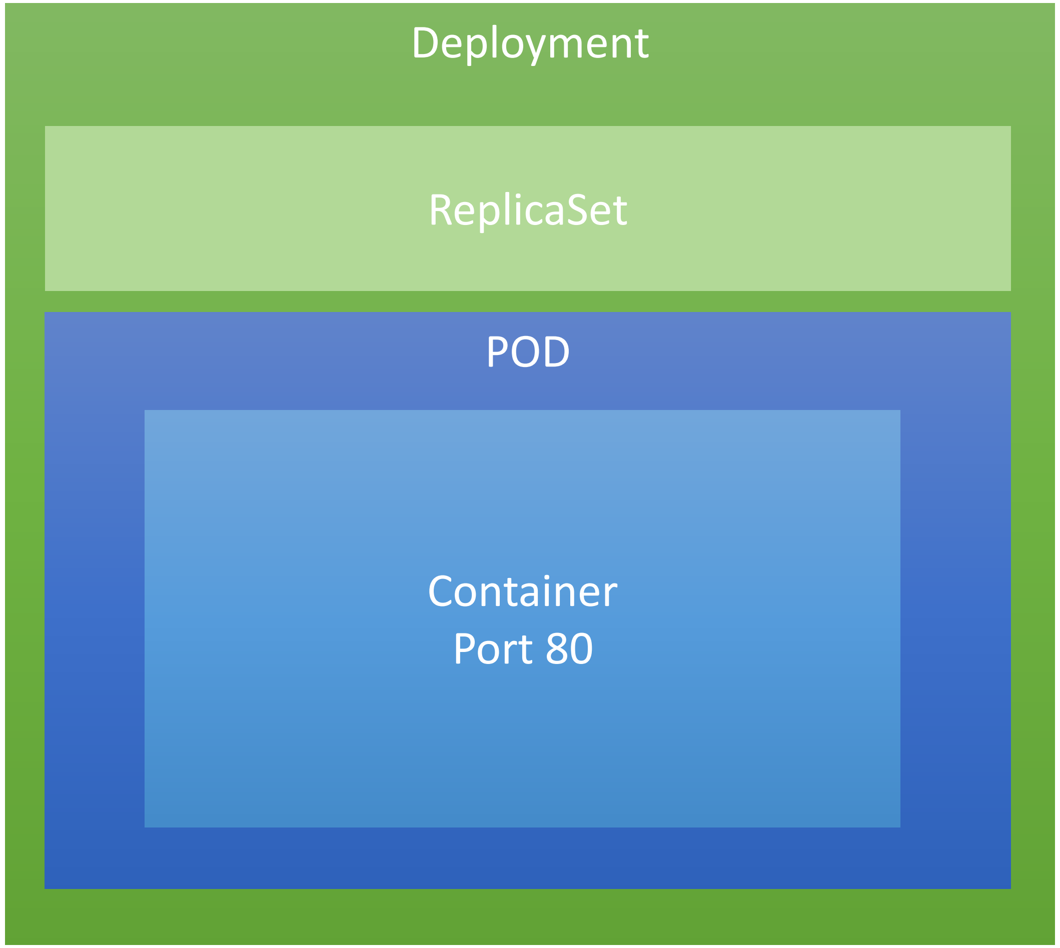
POD

Container
Port 80

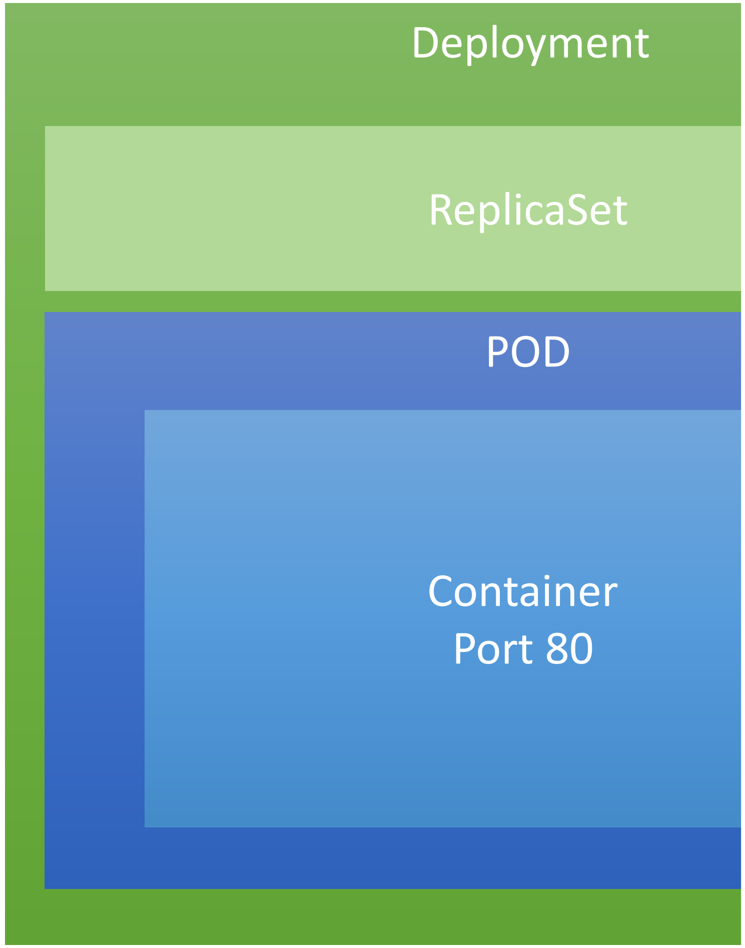


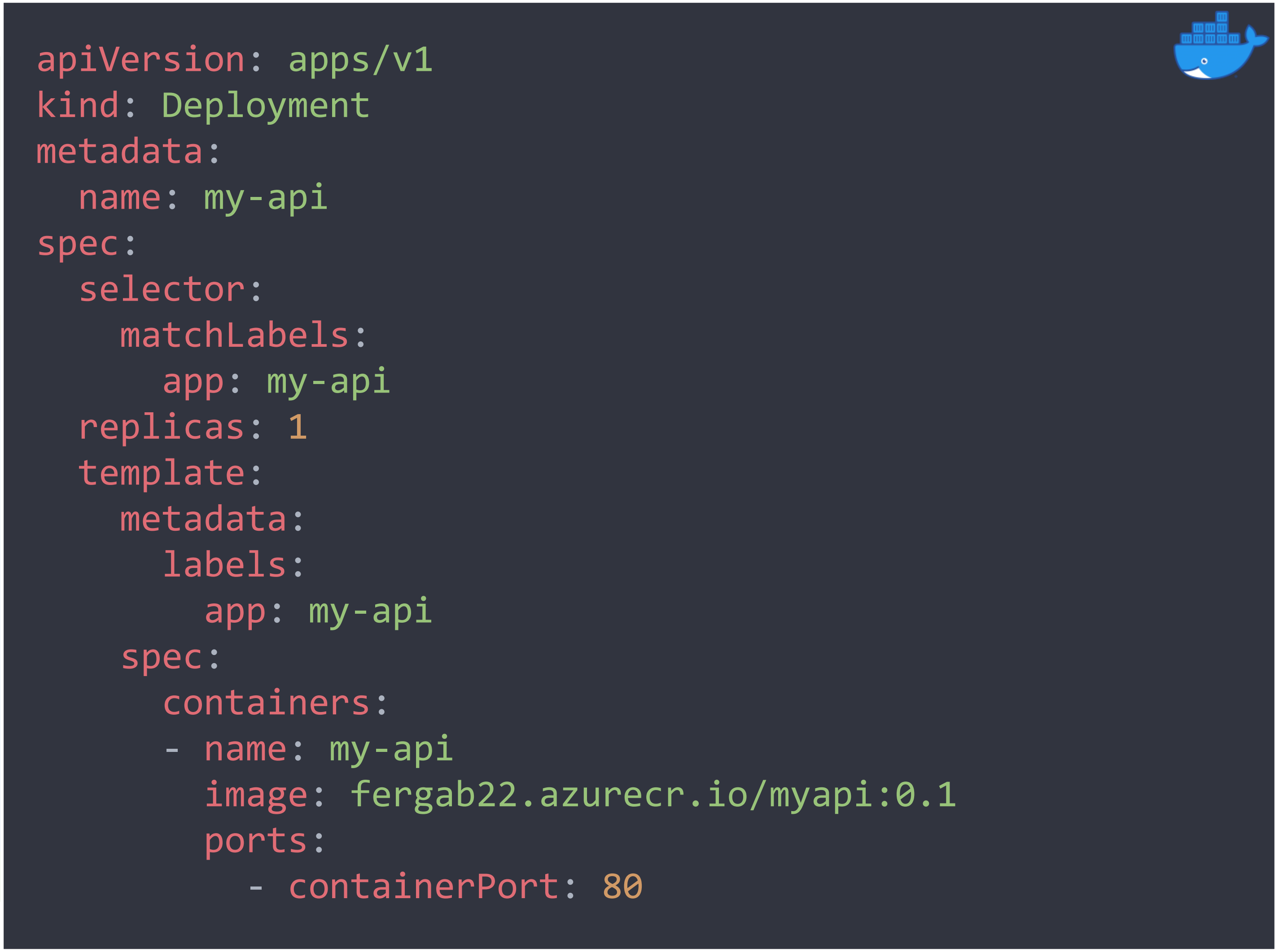
```yaml
kind: Pod
apiVersion: v1
metadata:
  name: my-api
spec:
  containers:
    - name: my-api
      image: fergab22.azurecr.io/myapi:0.1
      ports:
        - containerPort: 80
```

K8S Deployment



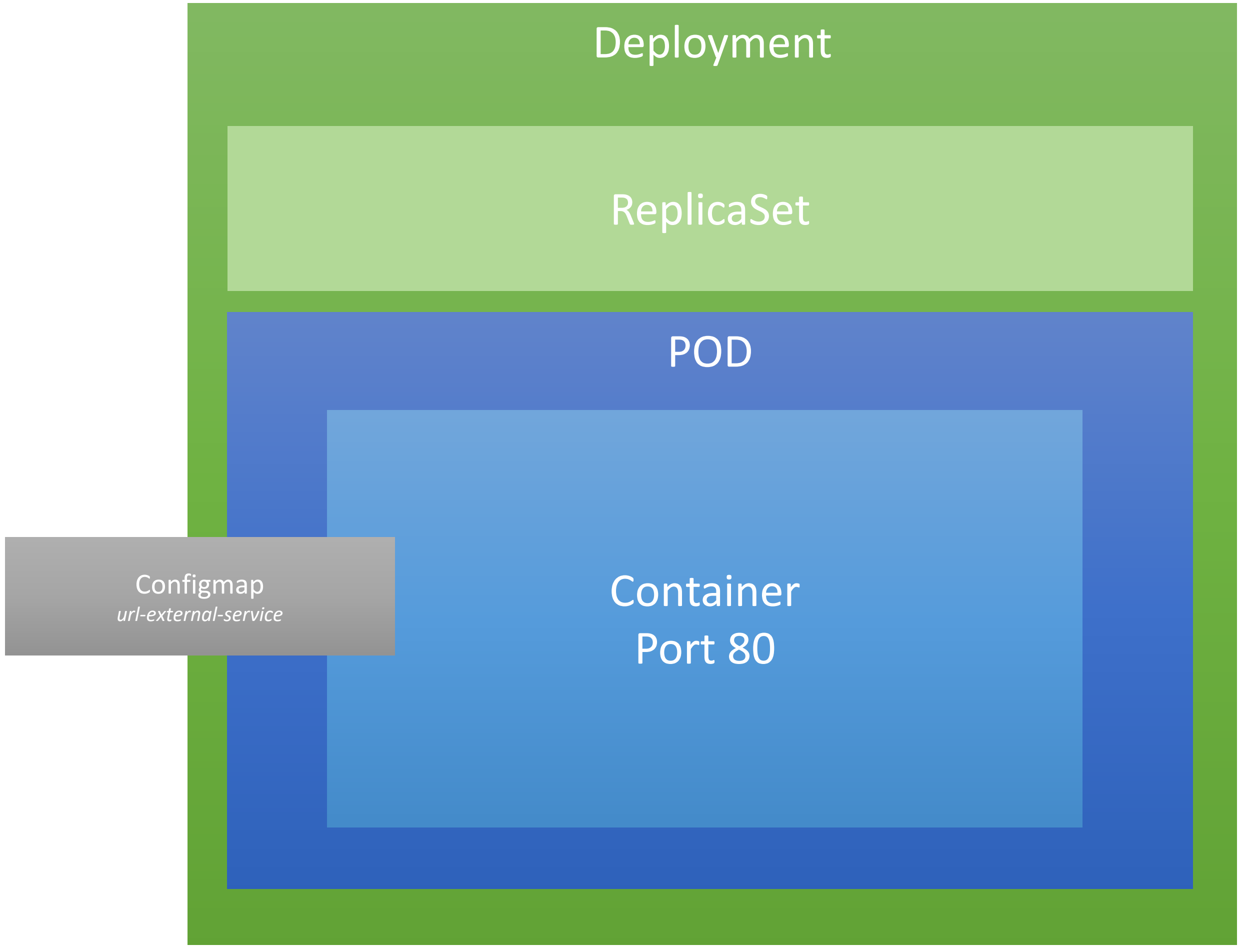
K8S Deployment



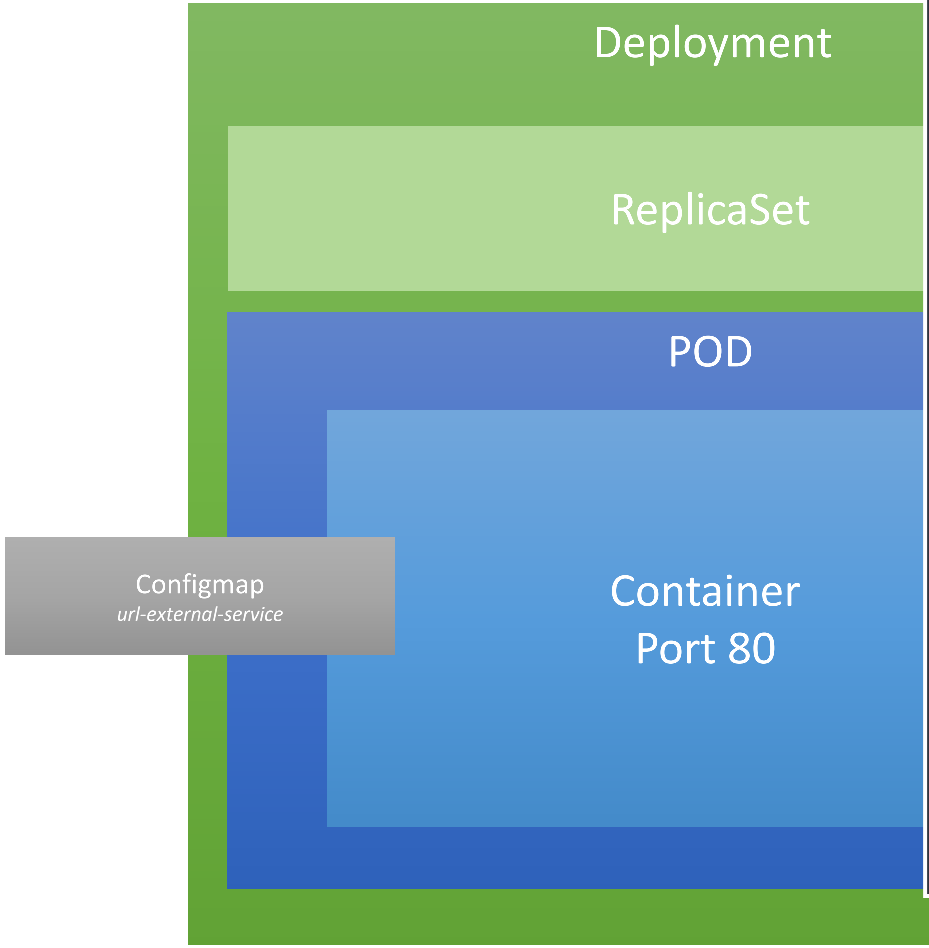


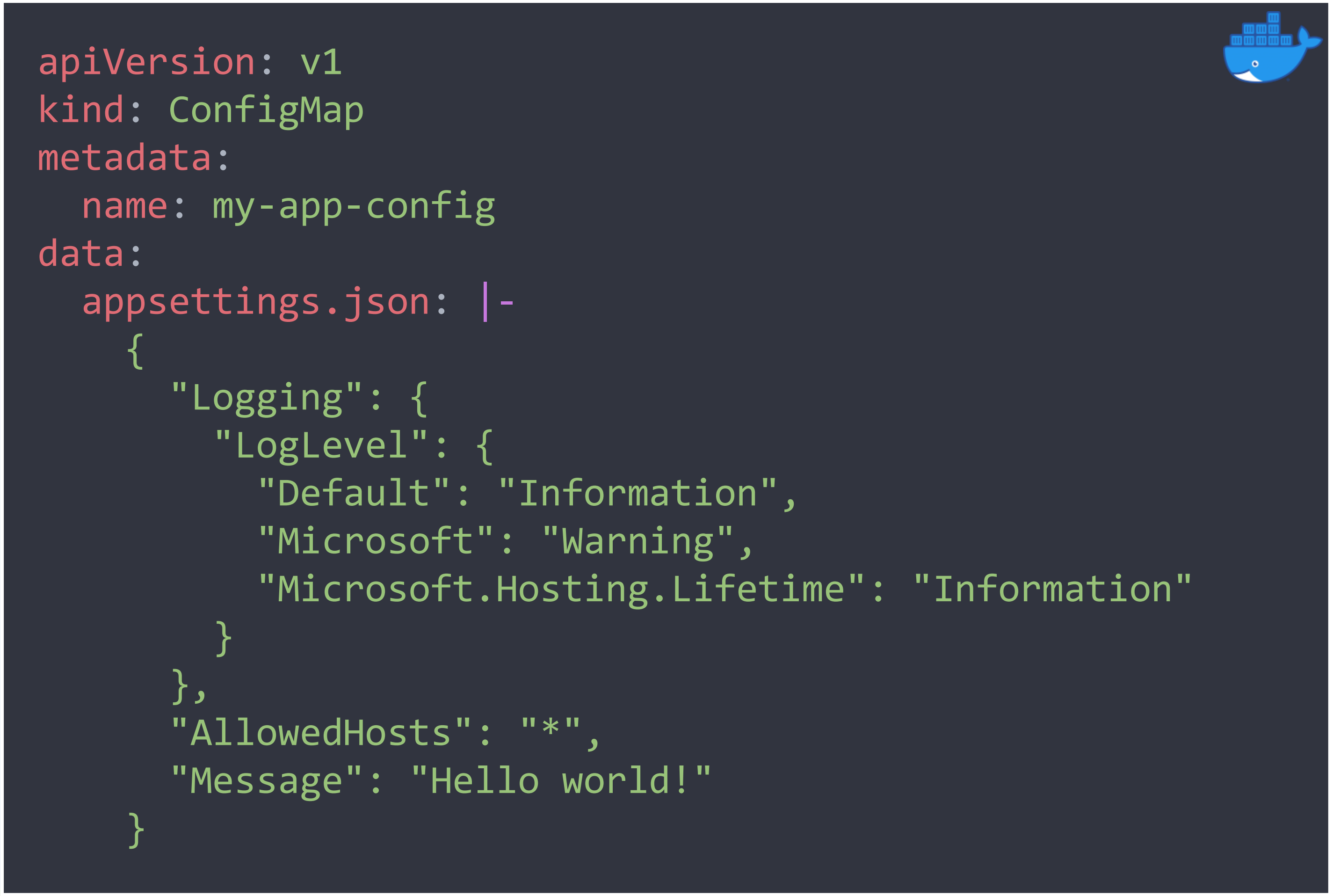
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-api
spec:
  selector:
    matchLabels:
      app: my-api
  replicas: 1
  template:
    metadata:
      labels:
        app: my-api
    spec:
      containers:
      - name: my-api
        image: fergab22.azurecr.io/myapi:0.1
        ports:
          - containerPort: 80
```

K8S Deployment



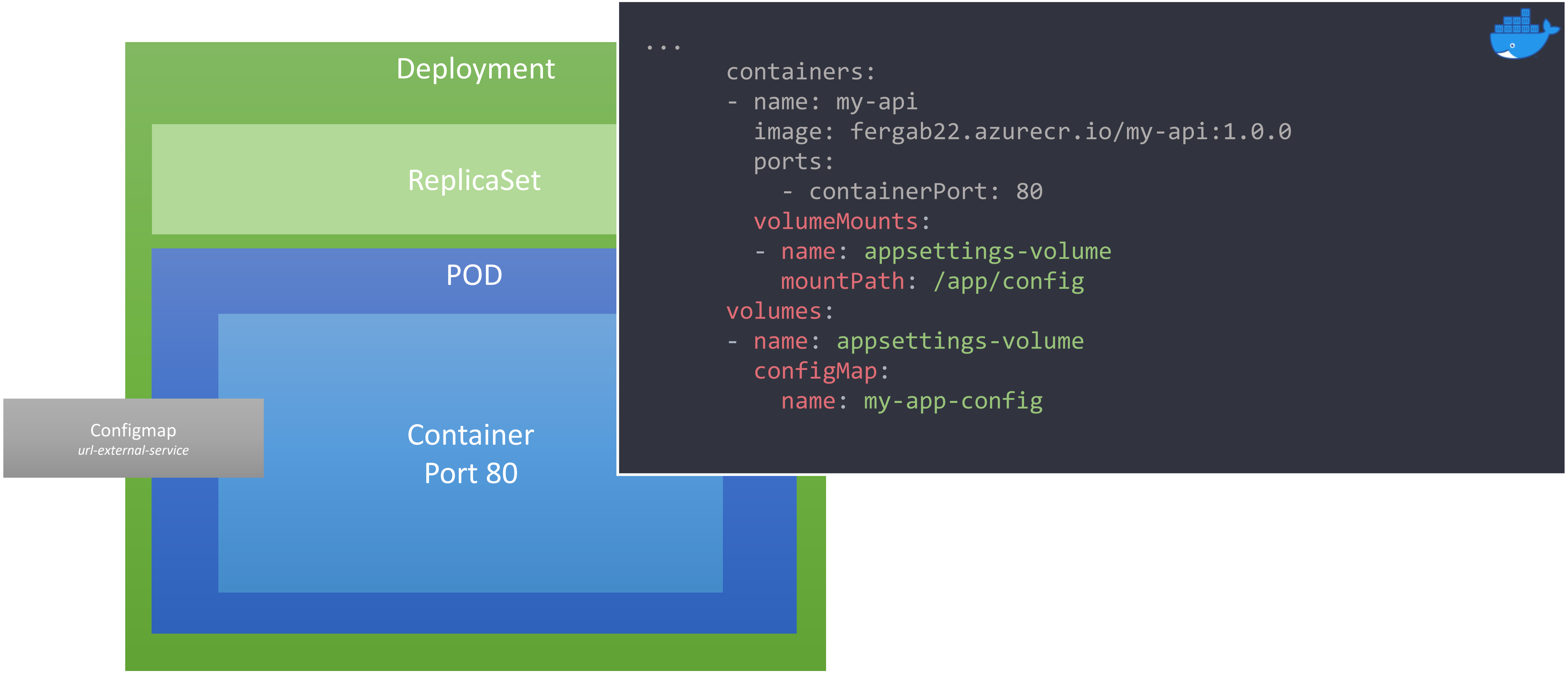
K8S Deployment





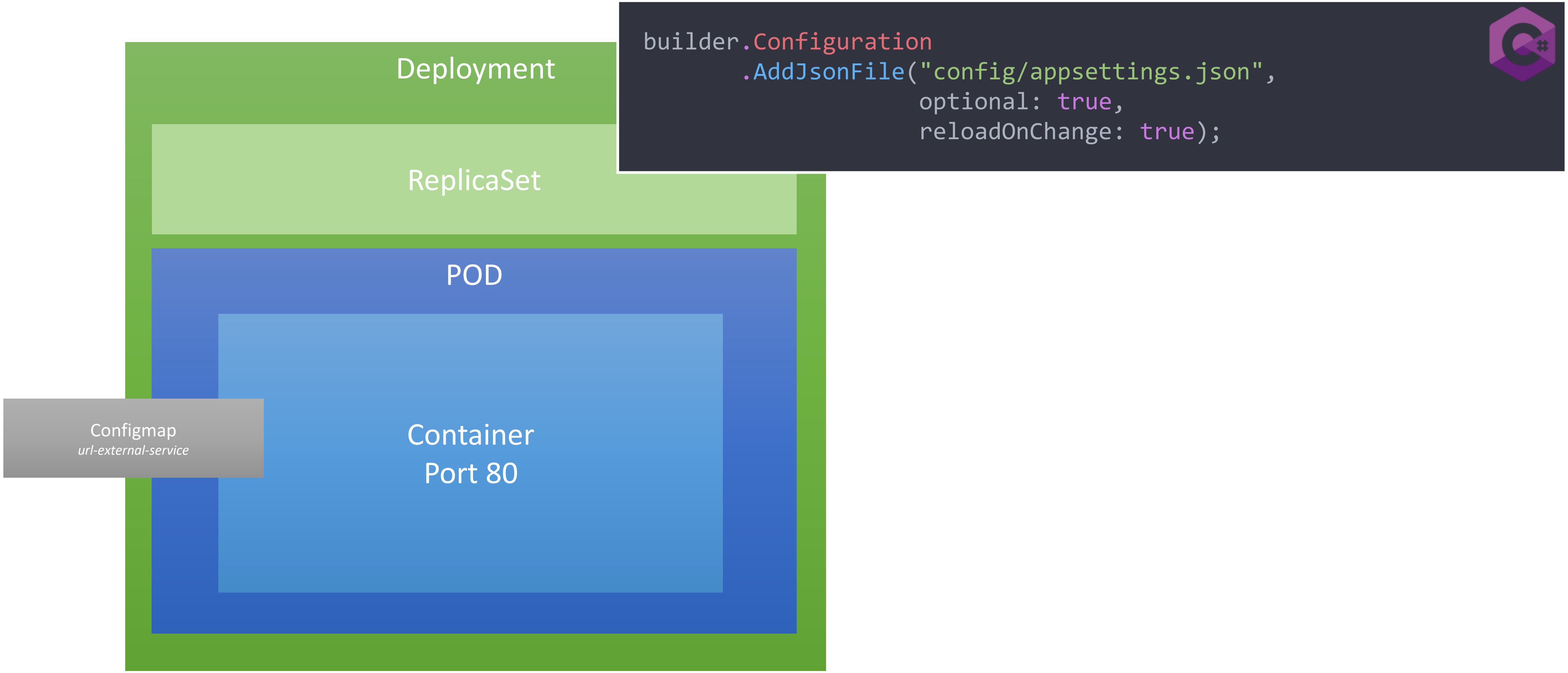
```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-app-config
data:
  appsettings.json: |-
    {
      "Logging": {
        "LogLevel": {
          "Default": "Information",
          "Microsoft": "Warning",
          "Microsoft.Hosting.Lifetime": "Information"
        }
      },
      "AllowedHosts": "*",
      "Message": "Hello world!"
    }
```

K8S Deployment



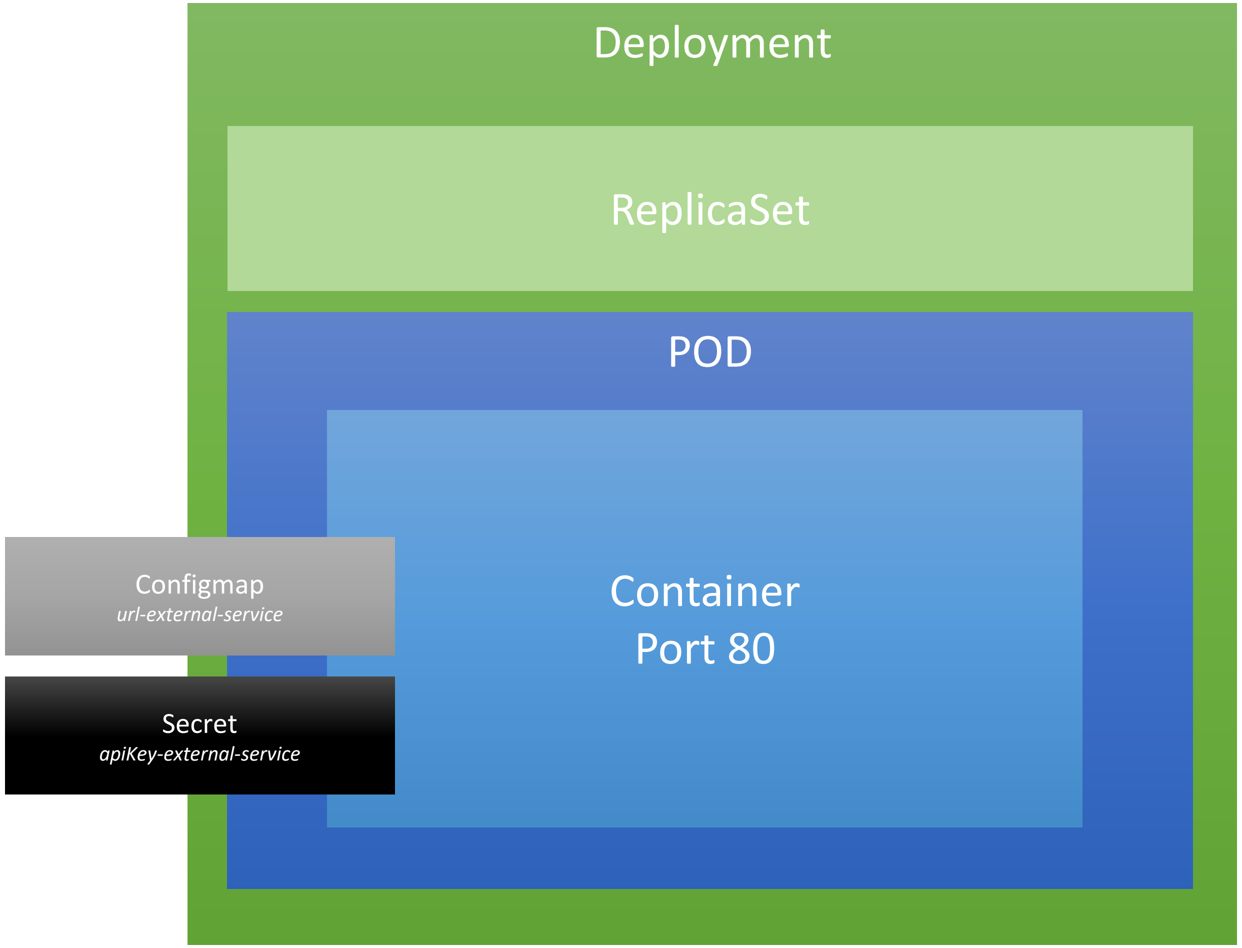
```
...
    containers:
    - name: my-api
      image: fergab22.azurecr.io/my-api:1.0.0
      ports:
        - containerPort: 80
      volumeMounts:
      - name: appsettings-volume
        mountPath: /app/config
    volumes:
    - name: appsettings-volume
      configMap:
        name: my-app-config
```

K8S Deployment

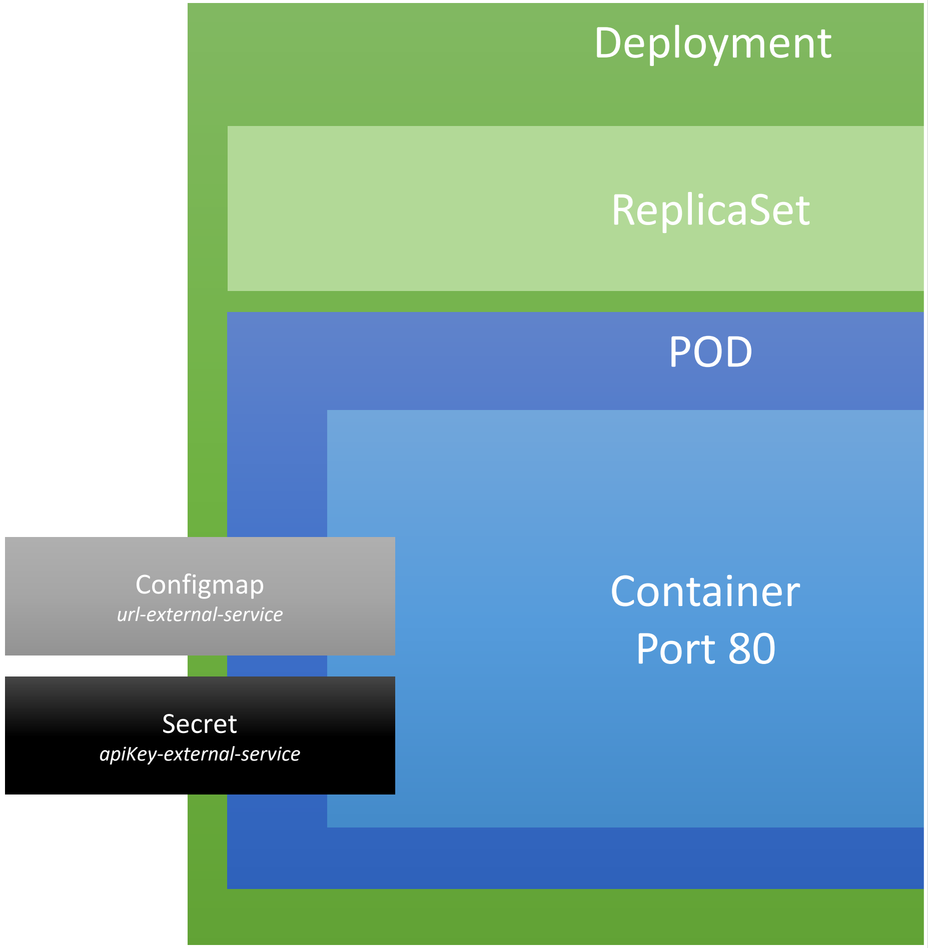


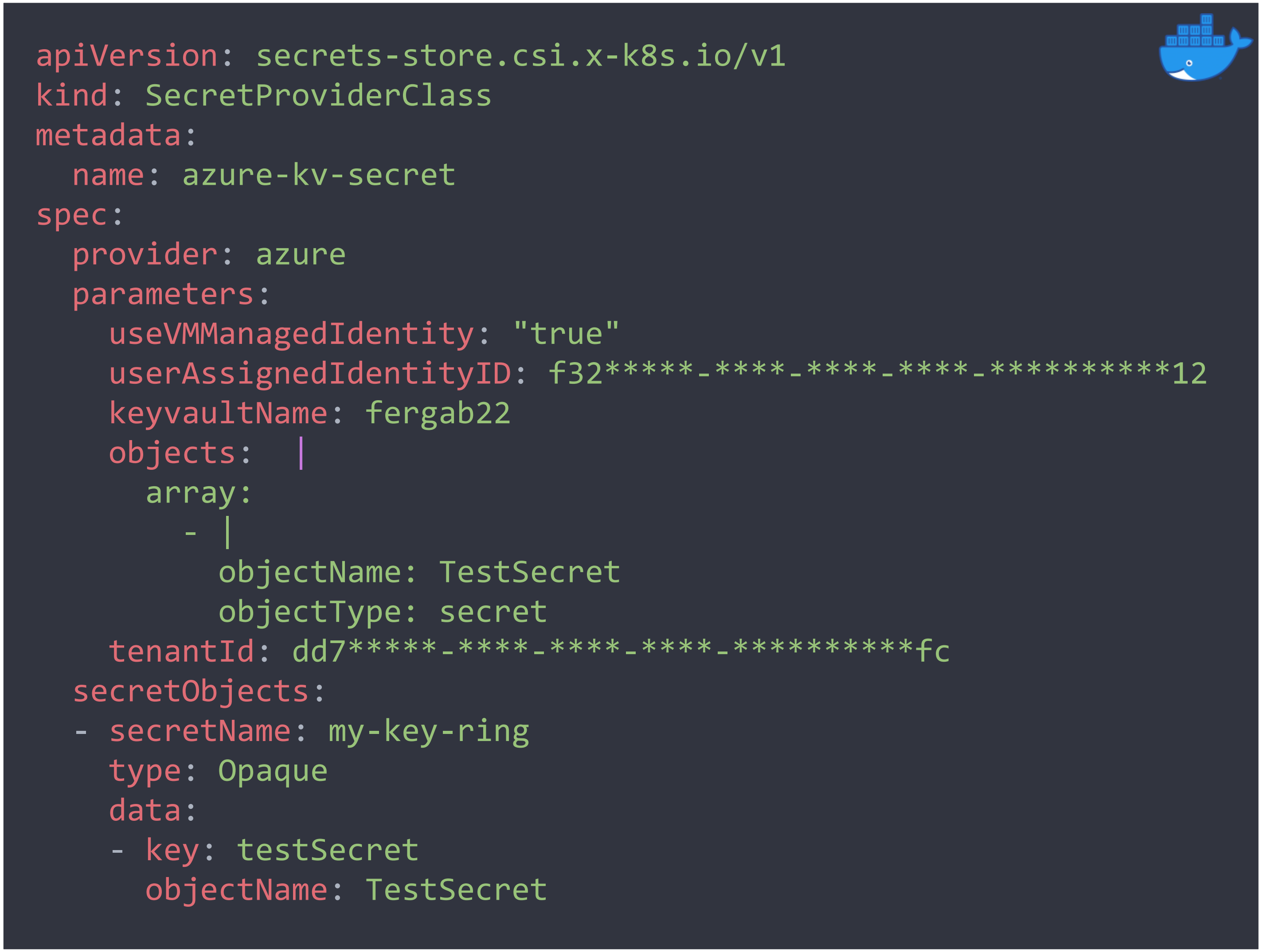
```
builder.Configuration
       .AddJsonFile("config/appsettings.json",
                    optional: true,
                    reloadOnChange: true);
```

K8S Deployment



K8S Deployment



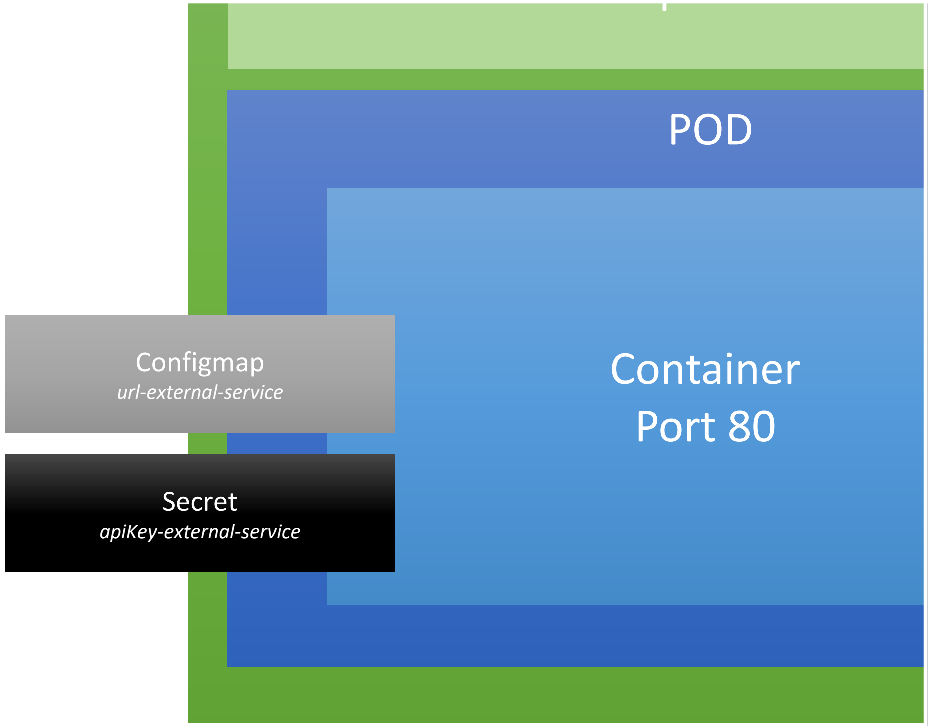


```yaml
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: azure-kv-secret
spec:
  provider: azure
  parameters:
    useVMManagedIdentity: "true"
    userAssignedIdentityID: f32*****-****-****-****-**********12
    keyvaultName: fergab22
    objects:  |
      array:
        - |
          objectName: TestSecret
          objectType: secret
    tenantId: dd7*****-****-****-****-**********fc
  secretObjects:
  - secretName: my-key-ring
    type: Opaque
    data:
    - key: testSecret
      objectName: TestSecret
```

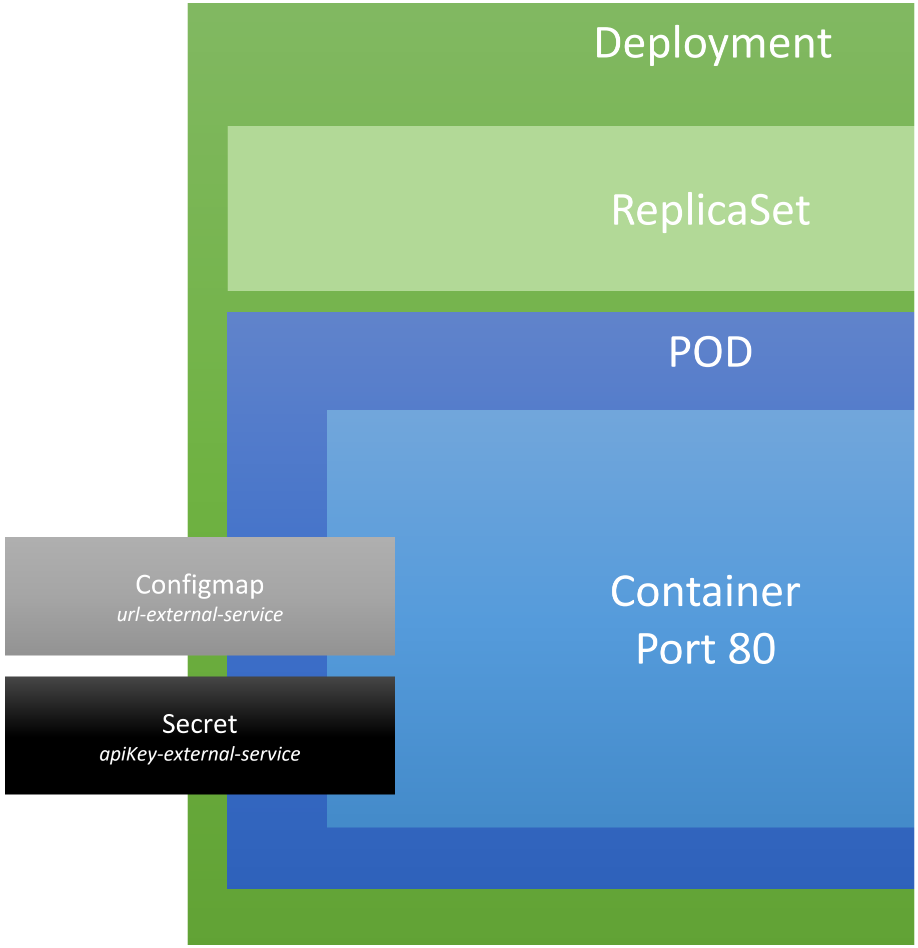
K8S Deployment

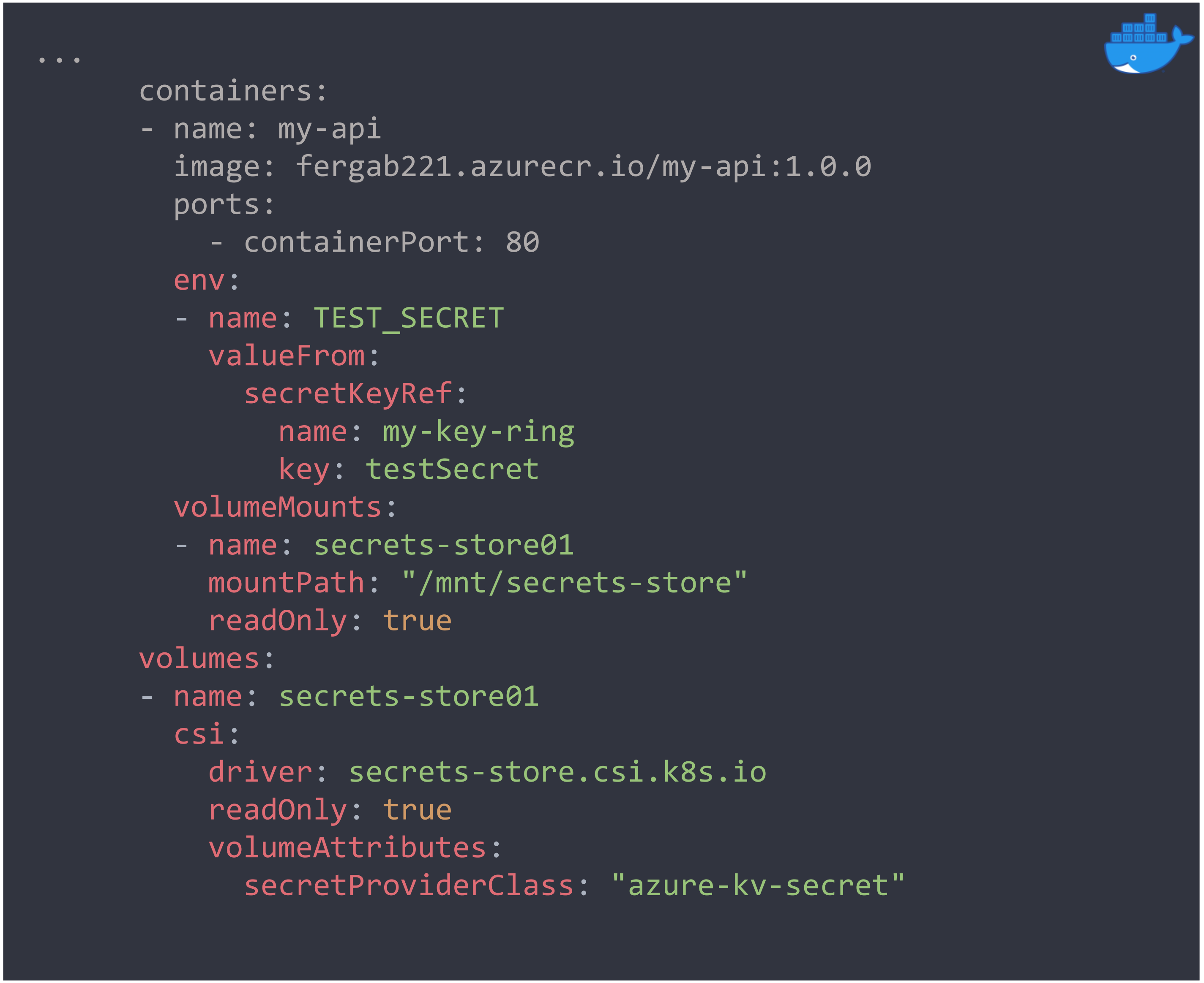
```
az aks show \
  -g $rg \
  -n $n \
  --query identityProfile.kubeletidentity.clientId \
  -o tsv
```

```
ersion: secrets-store.csi.x-k8s.io/v1
: SecretProviderClass
data:
me: azure-kv-secret
:
provider: azure
parameters:
  useVMManagedIdentity: "true"
  userAssignedIdentityID: f32*****-****-****-****-*********12
  keyvaultName: fergab22
  objects:  |
    array:
      - |
        objectName: TestSecret
        objectType: secret
  tenantId: dd7*****-****-****-****-*********fc
secretObjects:
- secretName: my-key-ring
  type: Opaque
  data:
  - key: testSecret
    objectName: TestSecret
```



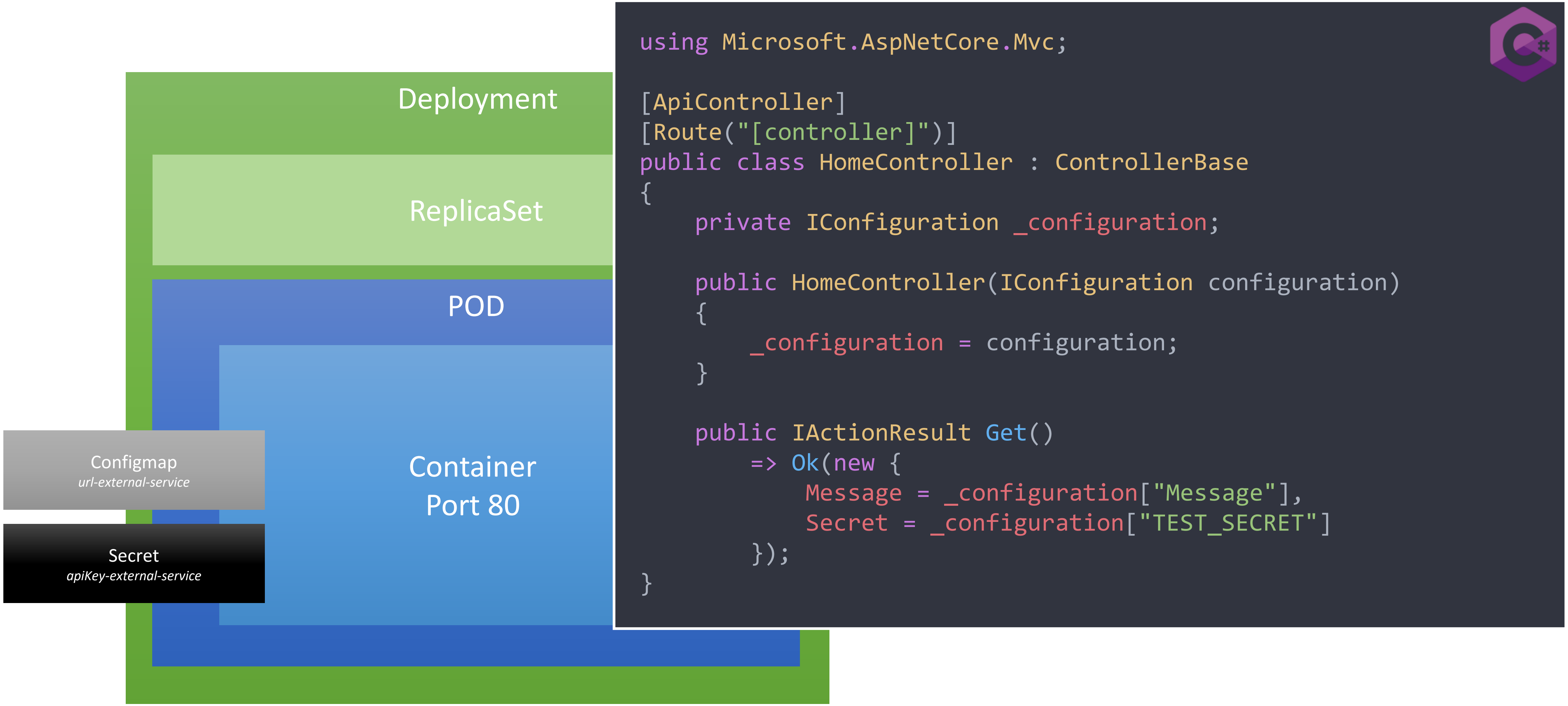
K8S Deployment





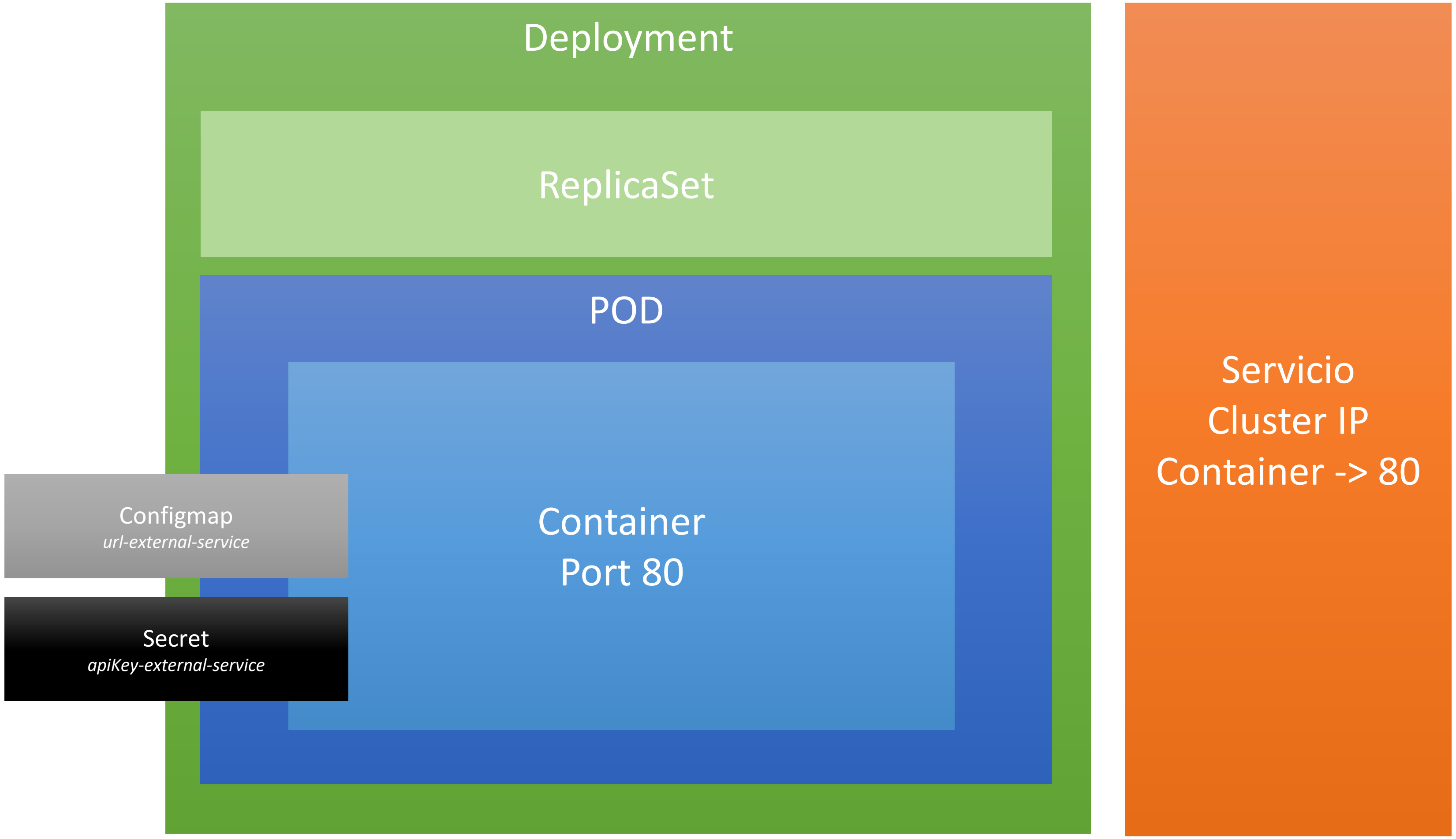
```
...
      containers:
      - name: my-api
        image: fergab221.azurecr.io/my-api:1.0.0
        ports:
          - containerPort: 80
        env:
        - name: TEST_SECRET
          valueFrom:
            secretKeyRef:
              name: my-key-ring
              key: testSecret
        volumeMounts:
        - name: secrets-store01
          mountPath: "/mnt/secrets-store"
          readOnly: true
      volumes:
      - name: secrets-store01
        csi:
          driver: secrets-store.csi.k8s.io
          readOnly: true
          volumeAttributes:
            secretProviderClass: "azure-kv-secret"
```

K8S Deployment

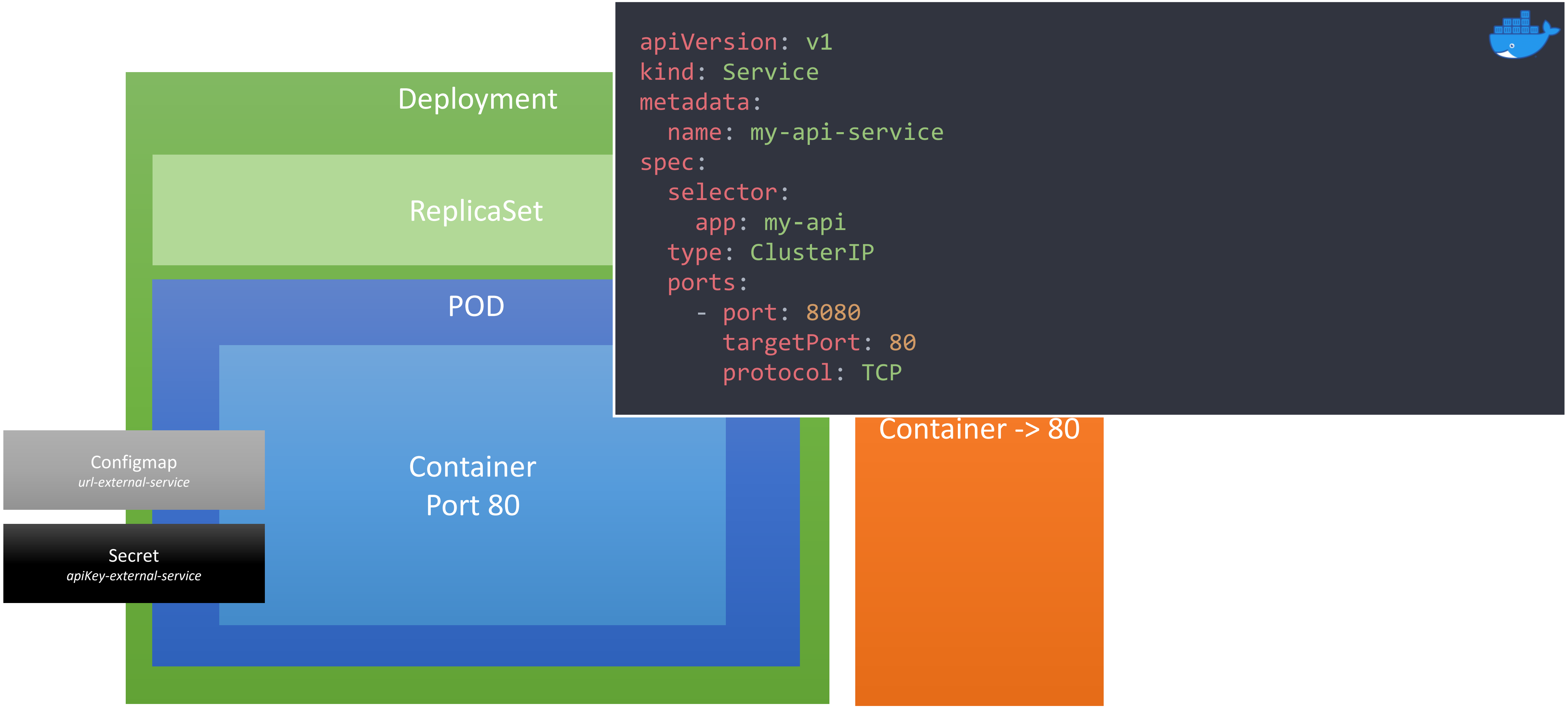


```csharp
using Microsoft.AspNetCore.Mvc;

[ApiController]
[Route("[controller]")]
public class HomeController : ControllerBase
{
    private IConfiguration _configuration;

    public HomeController(IConfiguration configuration)
    {
        _configuration = configuration;
    }

    public IActionResult Get()
        => Ok(new {
            Message = _configuration["Message"],
            Secret = _configuration["TEST_SECRET"]
        });
}
```

K8S Deployment

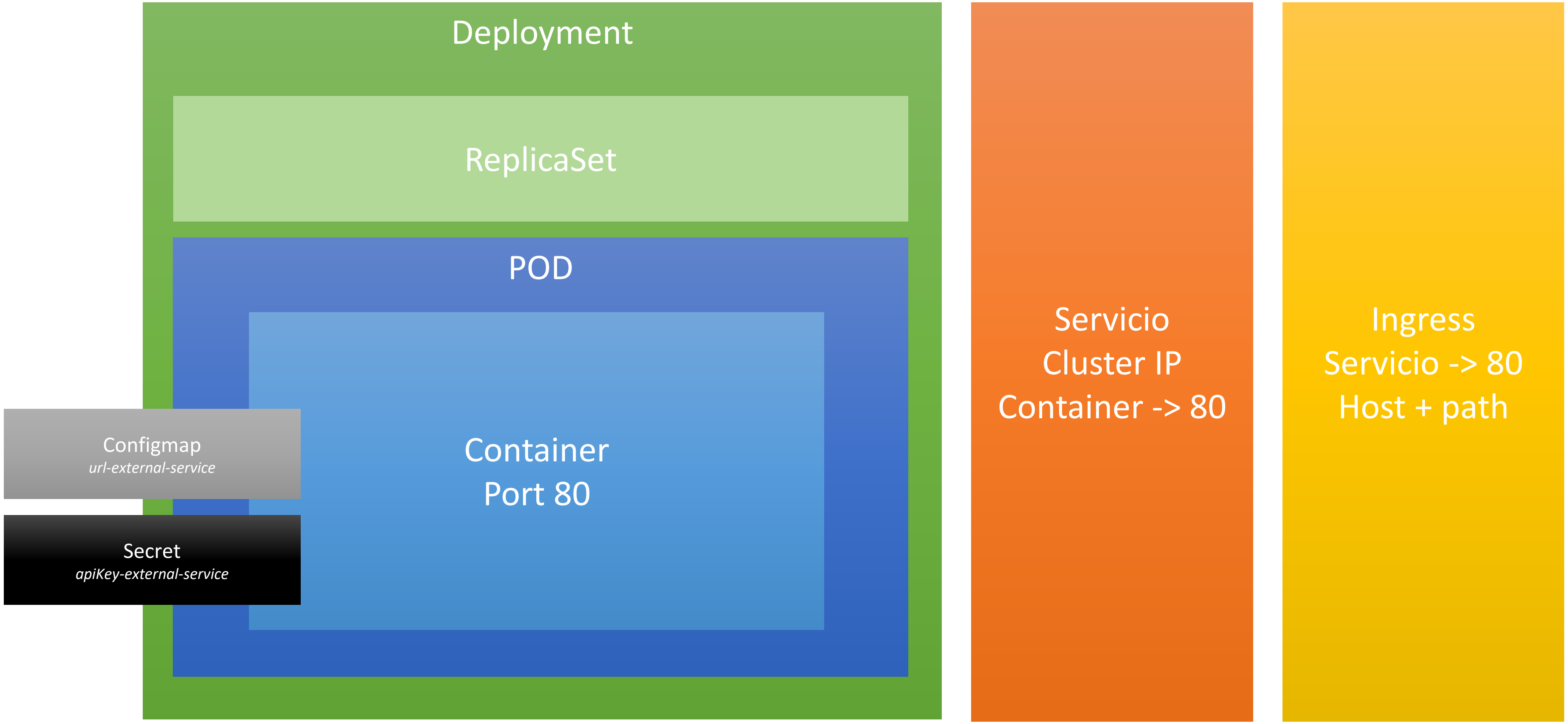


K8S Deployment



```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-api-service
spec:
  selector:
    app: my-api
  type: ClusterIP
  ports:
    - port: 8080
      targetPort: 80
      protocol: TCP
```

K8S Deployment



Deployment

ReplicaSet

POD

Container
Port 80

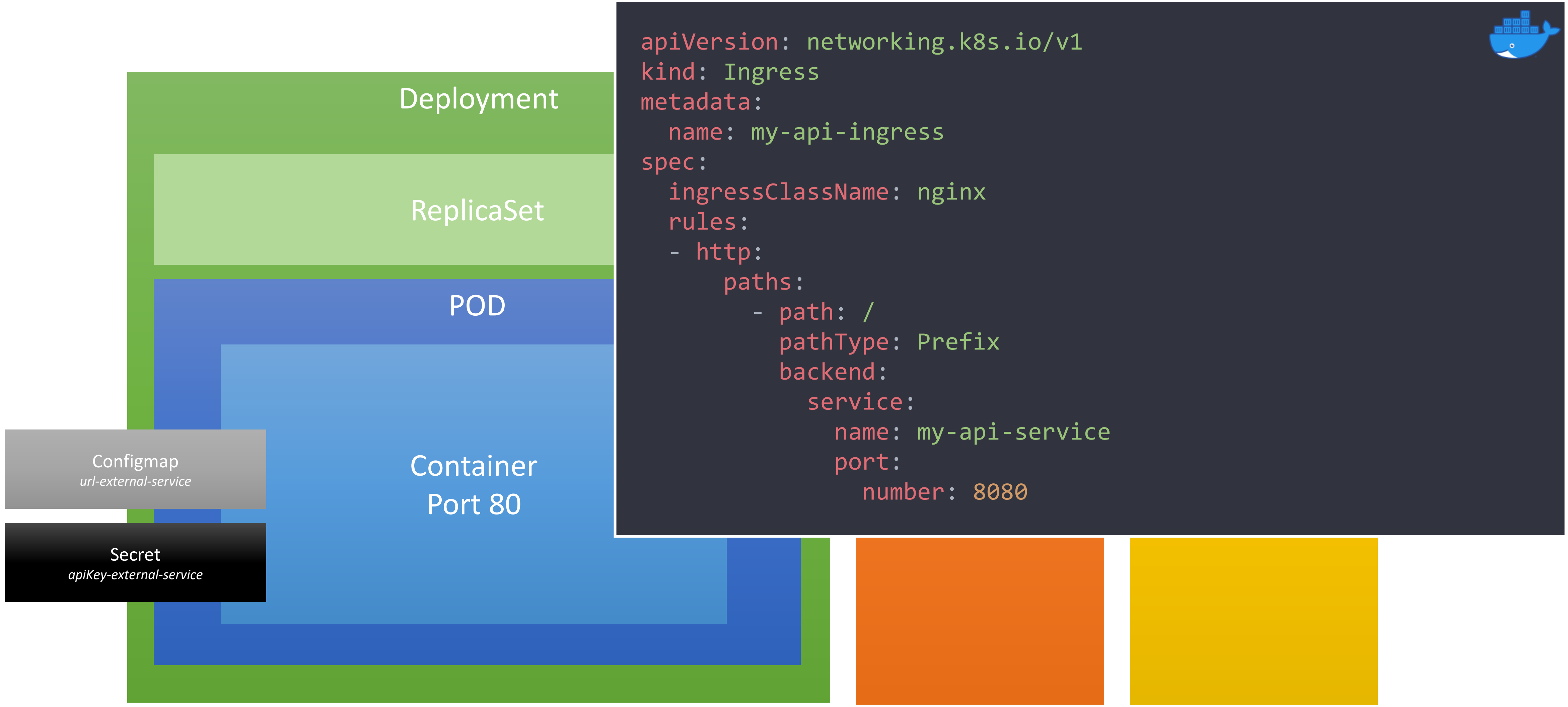
Configmap
url-external-service

Secret
apiKey-external-service

Servicio
Cluster IP
Container -> 80

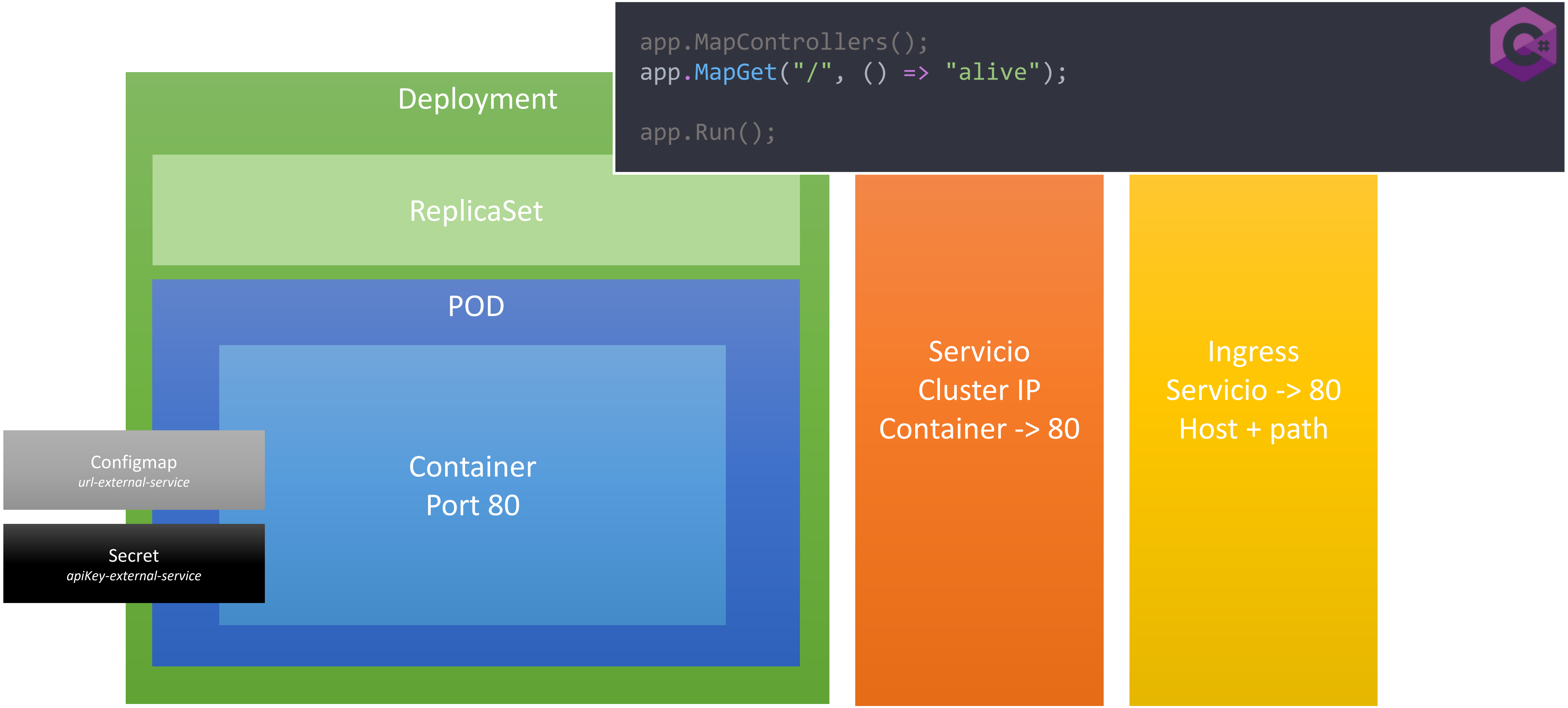
Ingress
Servicio -> 80
Host + path

K8S Deployment



```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-api-ingress
spec:
  ingressClassName: nginx
  rules:
  - http:
      paths:
        - path: /
          pathType: Prefix
          backend:
            service:
              name: my-api-service
              port:
                number: 8080
```

K8S Deployment



```
app.MapControllers();
app.MapGet("/", () => "alive");

app.Run();
```

```
docker build -t fergab22.azurecr.io/myapi:1.0 .
docker push fergab22.azurecr.io/myapi:1.0
k create namespace my-api
k apply -f deployment.yaml -n my-api
```

```
az group delete -n $n
```

```
thank you!
```

Nuestro patrocinadores

avanade

bravent
By veryTI group

clevertask

encamina
PIENSA EN COLORES

NTT DaTa

ilitia

Insight

intelequia

KaBel

myCloudDoor

nunsys
Tu socio tecnológico

plain concepts

prodware

Colabora

Microsoft

BY COMMUNITY • FOR COMMUNITY •
SPAIN
#GLOBALAZURE

• BY COMMUNITY • FOR COMMUNITY •
SPAIN
#GLOBALAZURE