**Azure CNI network**

In AKS we have two types of networking which is default AKS Kubernetes network plugin “kubenet” and more advanced “azure” network plugin. The main differences between them according MS docs are:

* With kubenet, nodes get an IP address from a virtual network subnet. Network address translation (NAT) is then configured on the nodes, and pods receive an IP address “hidden” behind the node IP. This approach reduces the number of IP addresses that we need to reserve in your network space for pods to use.
* With Azure Container Networking Interface (CNI), every pod gets an IP address from the subnet and can be accessed directly. These IP addresses must be unique across our network space and must be planned in advanced.

Each node has a configuration parameter for the maximum number of pods that it supports. The equivalent number of IP addresses per node are then reserved up front for that node. This approach requires more planning, and often leads to IP address exhaustion or the need to rebuild clusters in a larger subnet as your application demands grow.

**Note:** **For initial deployment it is very important to choose appropriate VM size for your cluster nodes because you can’t change size after the deployment.**

The good thing is that already now have AKS multiple node pools feature in preview. This will allow us to add additional node pools to our cluster and when we are adding a new pool, we can choose some other VM size for that pool nodes and then we can deploy our pods accordingly on those nodes which will suit our application requirements.

**We can use these CLI commands to get info about available VM sizes:**

**#Use this command to list all available VM sizes in a selected location:**

az vm list-skus --location westeurope --query "[].{Name:name, Size:size, Tier:tier}" --output table

**#Use this command to get detailed info of VM size:**

az vm list-skus --location westeurope --query "[?name=='Standard\_D2\_v2']"

**Note:** If you will not specify a –node-vm-size for your cluster then the default size will be used which is **“Standard\_D2\_v2”.**

As we already mentioned some az aks create options like: –vm-set-type or –load-balancer-sku are only become available after the **aks-preview** extension installation.

We are almost ready to start an AKS cluster deployment and run our Azure CLI script. This is a script which you can use to deploy and AKS from scratch:

The formula below provides a good estimate on how large your subnet must be, in order to accommodate your AKS resources.

Subnet size = (number of nodes + 1) + ((number of nodes + 1) \* maximum number of pods per node that you configure)

When using Azure CNI, by default each node is setup to run 30 pods. If you need to change this limit, you will have to deploy your AKS cluster using Azure CLI or Azure Resource Manager templates.

Just as an example, for a default AKS cluster deployment, using Azure CNI with 4 nodes, the subnet size at a minimum must be

IPs required = (4 + 1) + ((4+ 1) \* (30 pods per node)) = 5 + (5 \* 30) = 155

This means that the subnet must be at least a /24.

Mostly, we have used the default Kubenet network, but the [best practice guidance](https://docs.microsoft.com/ja-jp/azure/aks/operator-best-practices-network#choose-the-appropriate-network-model) provides some of the shortcomings of Kubenet and [guidance](https://docs.microsoft.com/ja-jp/azure/aks/operator-best-practices-network#choose-the-appropriate-network-model) that production environments should use Azure CNI. The Application Gateway used this time is based on Azure CNI.

With Azure CNI, you can:

* Assign IP to nodes and pods from Azure virtual network.
* Everything is in the Azure virtual network, so it can be used and routed to other Azure and on-premises resources.

Indeed, in Kubenet, pods received addresses from IP ranges managed in the cluster, and the IP ranges with the nodes were different.

The AKS network concept is more detailed in [Azure Kubernetes Service (AKS) Network Concepts for Applications](https://docs.microsoft.com/ja-jp/azure/aks/concepts-network)(<https://docs.microsoft.com/en-us/azure/aks/concepts-network>).

**Step1: Creating a resource group:**

az group create --name microservicerg --location eastus

**Step2: Creating a container registry (ACR):**

az acr create --resource-group microservicerg --name microserviceacr --sku basic

**Step3****: Creating a Service Principal (SP):**

az ad sp create-for-rbac --skip-assignment

or

az ad sp create-for-rbac --name spmicroservice

{

"appId": "6d7a6de8-8f3c-4ba4-b323-a061953c0fa4",

"displayName": "spmicroservice",

"name": "http://spmicroservice",

"password": "3a3ea7e1-26ee-453d-a959-07ecaa860bb6",

"tenant": "214afa4d-d9b0-4cc4-9497-06abd7cbc5ea"

}

Now we need to grant authority to ACR:

**Step4: Resource name of (ACR):**

az acr show --resource-group microservicerg --name microserviceacr --query "id" --output tsv

/subscriptions/a2ca6b0c-2592-420a-be30-b1cbe1da7646/resourceGroups/microservicerg/providers/Microsoft.ContainerRegistry/registries/microserviceacr

**Step5: Grant ACR authority to service principal and assign an acrpull role to pull images from ACR:**

az role assignment create --assignee <appId> --scope <acrId> --role acrpull

az role assignment create --assignee 6d7a6de8-8f3c-4ba4-b323-a061953c0fa4 --scope /subscriptions/a2ca6b0c-2592-420a-be30-b1cbe1da7646/resourceGroups/microservicerg/providers/Microsoft.ContainerRegistry/registries/microserviceacr --role acrpull

appId: Check it from result of service principal.

acrId: Resource name obtained above.

## **Virtual network size**

With Azure CNI, all IP addresses are allocated from the virtual network, so you need a subnet that can provide a sufficient number of IPs. See [Planning your cluster for IP addressing](https://docs.microsoft.com/ja-jp/azure/aks/configure-azure-cni?view=azure-cli-latest#plan-ip-addressing-for-your-cluster)(<https://docs.microsoft.com/en-us/azure/aks/configure-azure-cni>).

**Step6: Create a virtual network to build an AKS cluster. Change the size as needed.**

az network vnet create -g microservice -n microserviceaksvnet --address-prefix 10.0.0.0/16 --subnet- name microserviceakssubnet --subnet-prefix 10.0.0.0/24

**Step7: Get the resource name of the virtual network subnet.**

az network vnet subnet list -g microservice --vnet-name microserviceaksvnet --query "[0].id" --output tsv

**Step8: Give the created service principal network contributor rights for the virtual network subnet.**

az role assignment create --assignee <appId> --scope <subnet-id> --role "network contributor"

Step9: