AGIC & Kubenet – Tips and tricks to make it works

David Frappart

About me

- Still exploring the Cloud platform capabilities (which get new stuff all the time)
- Breath IaC and Automation (but more Hashicorp stuff than other ^^)
- Still struggles in the K8S landscape
- MVP Azure since 2019
- Huge Final Fantasy fan

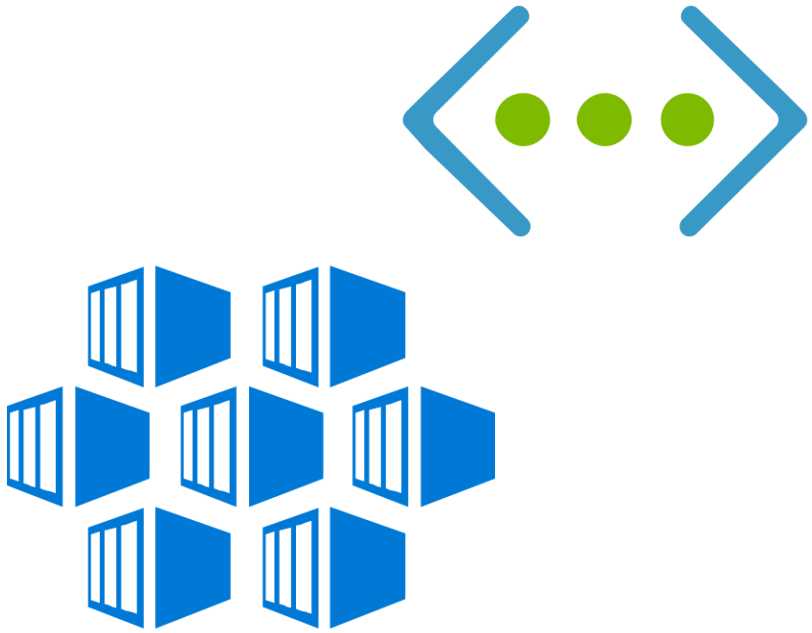
agenda.

1. Concepts Review
2. The simple(st) way - AGIC Add-on
3. The control freak way – helm charts, AGIC OSS and Pod Identity OSS

1

Concepts Review

AKS Networking Model Choice

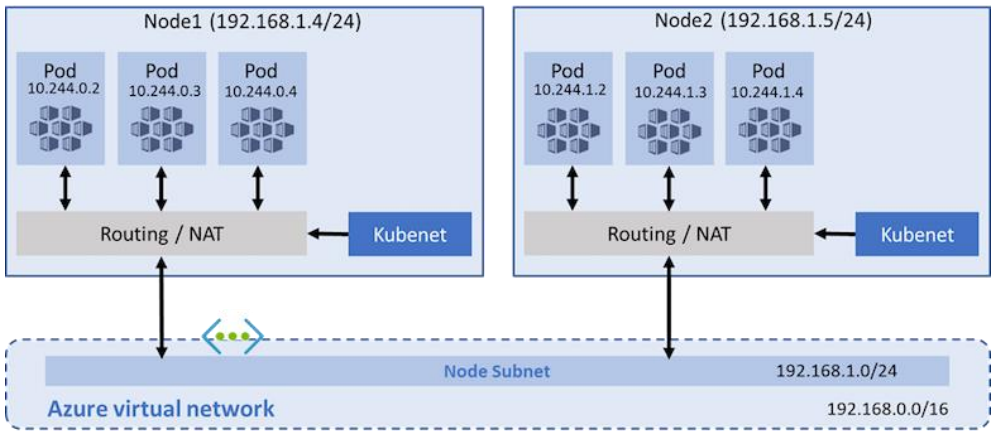


2 choices when building AKS

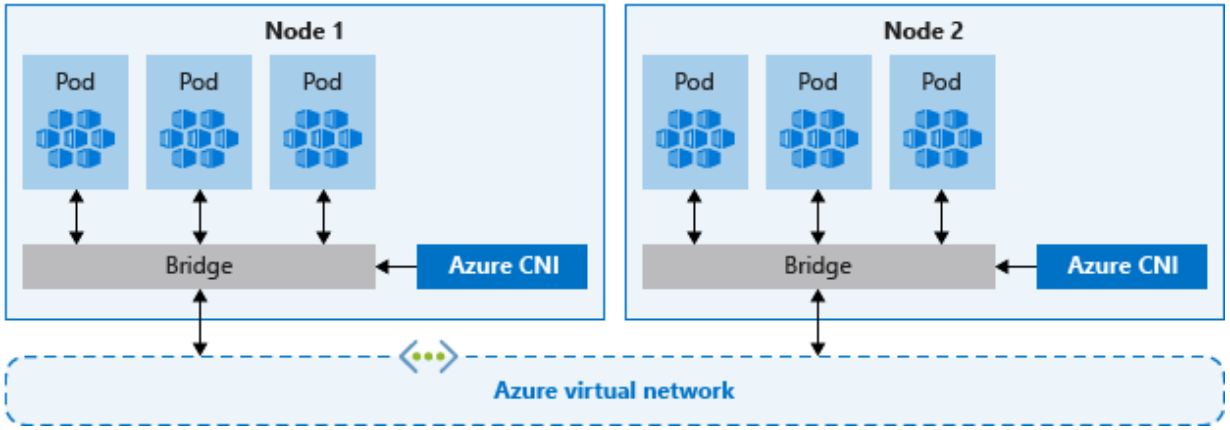
- Azure CNI
- Rich integration with Azure Vnet
- More features
- Everything get an IP from the Vnet Range, Node, Pod…

- Kubenet
- Less integration and features
- Only Nodes consume IP from Vnet Range

Kubenet vs Azure CNI

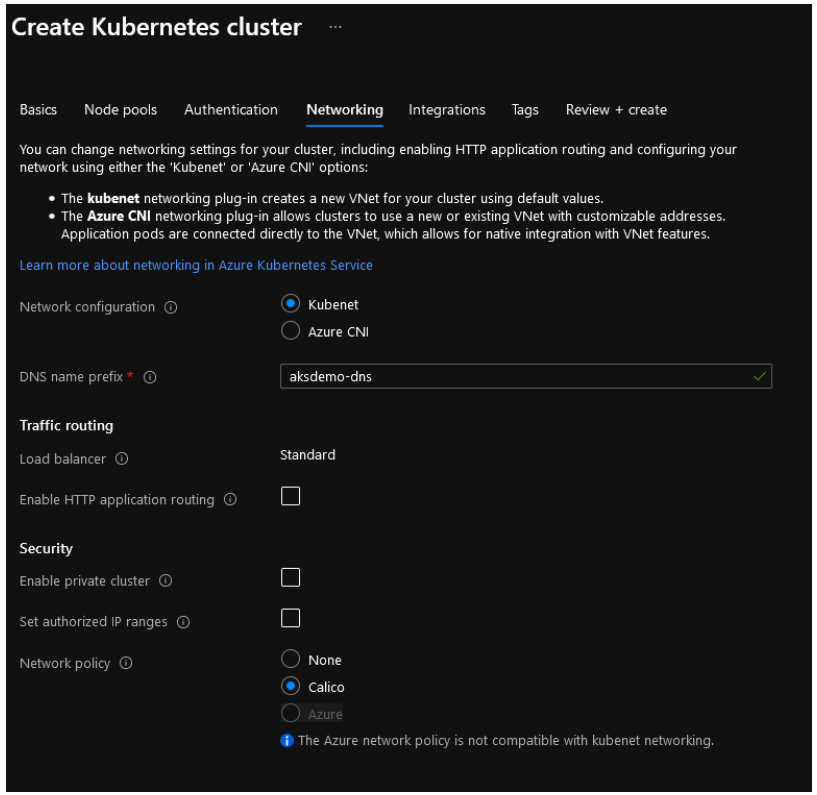


Kubenet

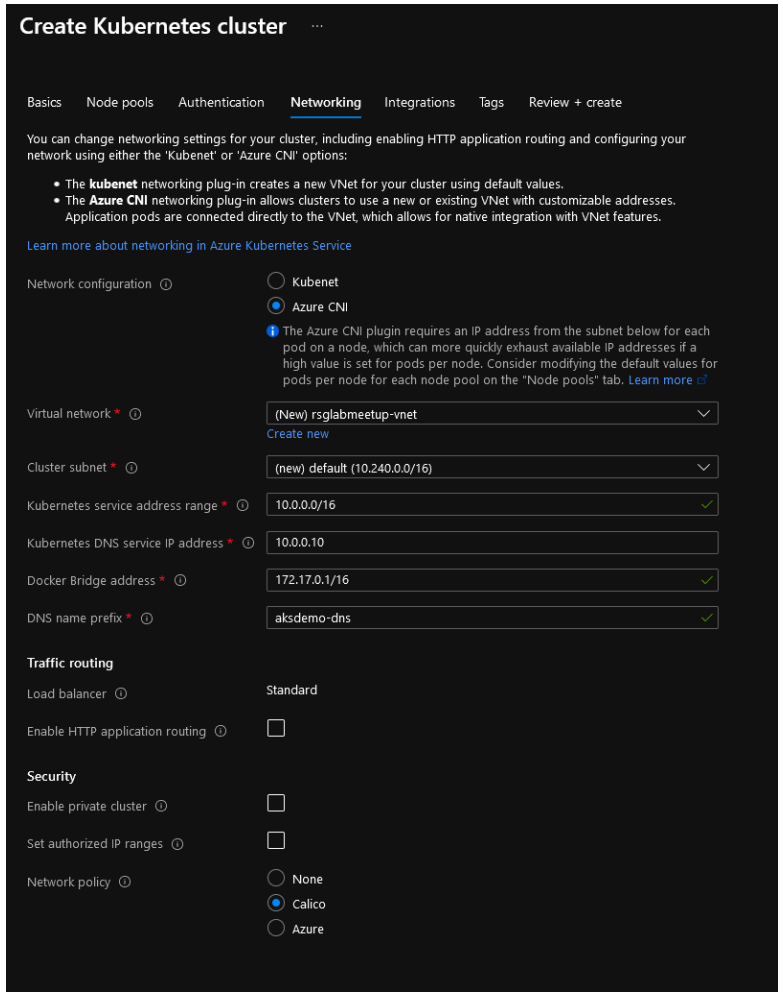


Azure CNI

Kubenet vs Azure CNI

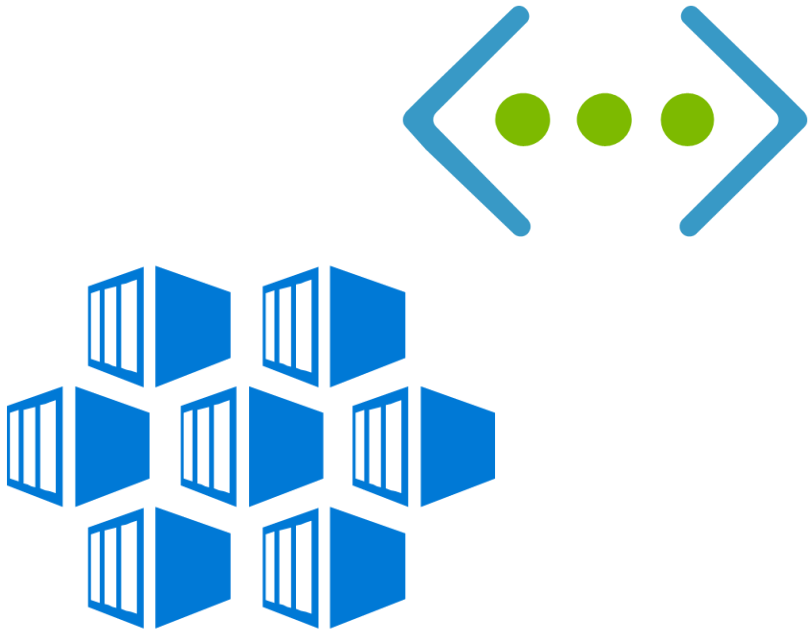


Kubenet



Azure CNI

The specificities of Kubenet and why you may choose it



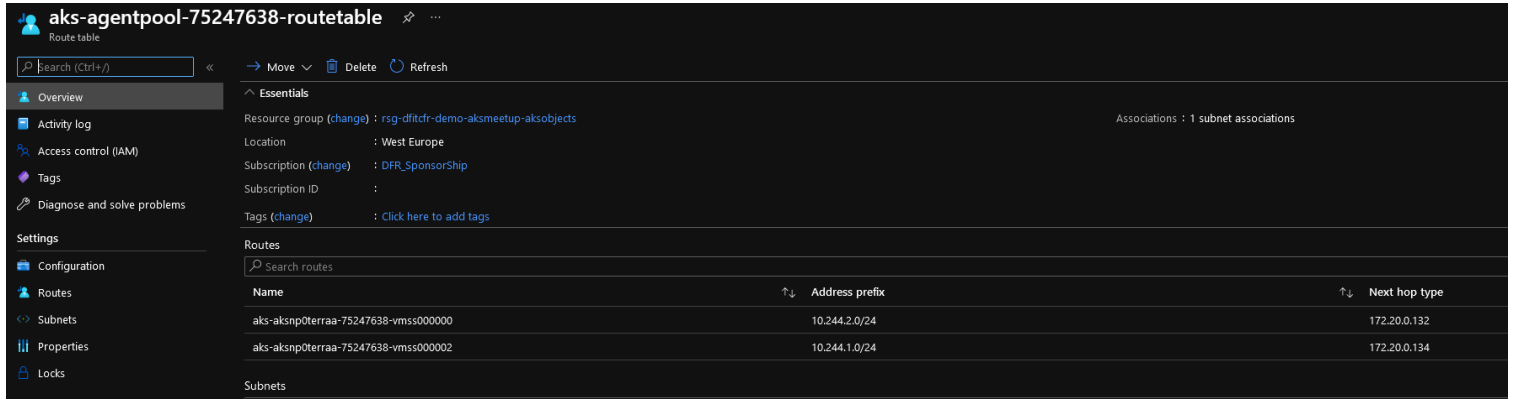
Kubenet

- Isolate pod addressing from the VNet
- Avoid claiming huge RFC 1918 range for AKS
- Pods are reachable from the outside only through exposed Kubernetes Services
- Limited to Linux node pool only

Keep in mind

- Du to the isolation, an UDR is required
- Still consumes 3 big ranges that should not be used anywhere else (apart on other AKS kubenet cluster

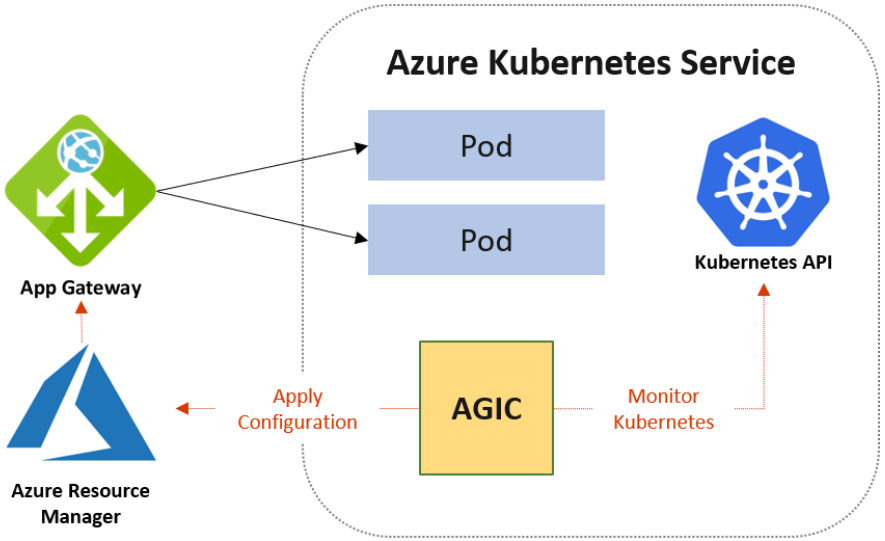
The specificities of Kubenet and why you may choose it



CIDR to be cautious of	Default value
The *--service-cidr* is used to assign internal services in the AKS cluster an IP address.	10.0.0.0/16
The *--pod-cidr* should be a large address space that isn't in use elsewhere in your network environment.	10.244.0.0/16
The *--docker-bridge-address* lets the AKS nodes communicate with the underlying management platform.	172.17.0.1/16

- Each node is assigned his own /24 in the service CIDR subnet
- Limitation of 400 nodes (due to the UDR limitation

AGIC TLDR



- Provide an Ingress Controller in AKS based on Application Gateway
- Network layer Lives in the Azure Control Plane as the Application Gateway

Just a reminder on AGIC Add-on vs OSS Project

- Simply add the feature through cli
- Benefit from the support of Microsoft

Add-on

- Get more control on the deployment and features
- Use community tools for deployment such as Helm

Open Source Project

2

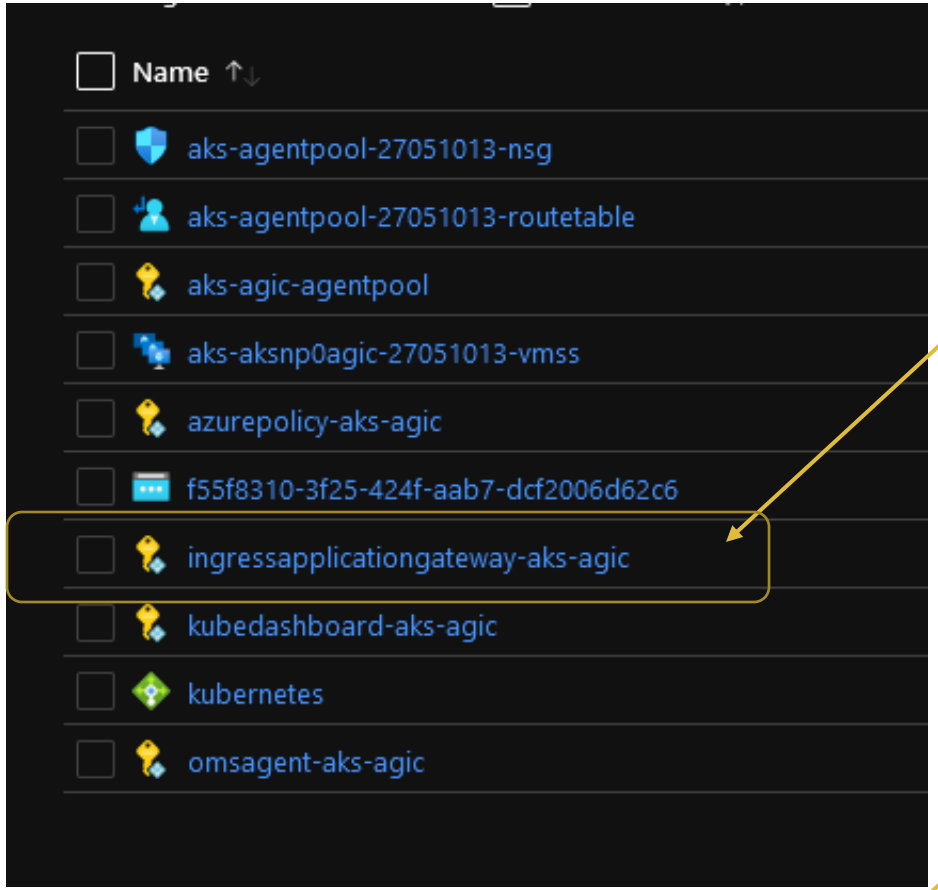
AGIC Add-on - Making it work

One ~~ring~~command to ~~rule~~configure them all

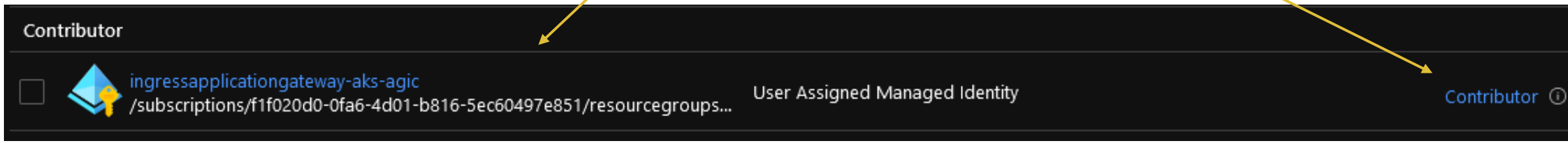
- Application Gateway Id
- AKS Cluster Id
- AKS Cluster resource group

```
$agwid = (az network application-gateway show -n agw-1 -g rsgagicmeetup -o tsv --query id)
az aks enable-addons -n aks-agic -g rsgagicmeetup -a ingress-appgw --appgw-id $agwid
 AAD role propagation done[##############################################]  100.0000%{
  "aadProfile": {…},
  "addonProfiles": {
==============================Truncated==================================
    "ingressApplicationGateway": {
      "config": {
        "applicationGatewayId": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/…",
        "effectiveApplicationGatewayId": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/…"
      },
      "enabled": true,
      "identity": {
        "clientId": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
        "objectId": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
        "resourceId": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/…
      }
    },
==============================Truncated==================================
 }
```

Display config



- New UAI on environment
- Contributor role on the applcation Gateway



2.5

What about using terraform ?

Since the GA, Terraform provider allows deployment with a few options

- Enabled or disabled
- Green field or brown field deployment
- Auto create the agw subnet (?)

```
addon_profile {

   azure_policy {…}

   oms_agent {…}

   ingress_application_gateway {
     enabled                                = var.IsAGICEnabled
     gateway_id                             = var.AGWId
     gateway_name                           = var.AGWName
     subnet_cidr                            = var.AGWSubnetCidr
     subnet_id                              = var.AGWSubnetId
   }
```

3

The control freak way: helm charts, AGIC OSS and Pod Identity OSS

Following the documentation

(The one on github btw)

https://azure.github.io/application-gateway-kubernetes-ingress/

Outline

- Set up pod identity
- Deploy AGIC with Helm chart

And also, what was done automatically with the Add-on

- RBAC Assignments for proper Identity
- UDR on required Subnet

Using Terraform for

RBAC assignments:

- Managed Identity (created by Azure) requirement for Pod Identity
- Managed Identity for AGIC

UDR config

- Associate agw subnet with AKS UDR

User Assign Identity and RBAC

```
#####################################################################
# Creating a UAI for AGIC with contributor role on RG

module "UAIAGIC" {

  #Module Location
  source                                = "github.com/dfrappart/Terra-AZModuletest//Custom_Modules/Kube_UAI/"

  #Module variable
  UAISuffix                             = "agic"
  TargetLocation                        = data.azurerm_resource_group.AKSRG.location
  TargetRG                              = data.azurerm_resource_group.AKSRG.name
  RBACScope                             = data.azurerm_resource_group.AKSRG.id
  BuiltinRoleName                       = "Contributor"
  ResourceOwnerTag                      = var.ResourceOwnerTag
  CountryTag                            = var.CountryTag
  CostCenterTag                         = var.CostCenterTag
  Environment                           = var.Environment
  Project                               = var.Project


}
```

UDR Association

```
data "azurerm_subnet" "AGWSubnet" {
  name                 = data.terraform_remote_state.AKSClus1.outputs.AGWSubnetName
  virtual_network_name  = data.terraform_remote_state.AKSClus1.outputs.VNetName
  resource_group_name   = data.azurerm_resource_group.AKSRG.name
}


#########################################################################
# Associating route to agw subnet

resource "azurerm_subnet_route_table_association" "example" {
  subnet_id                                 = data.azurerm_subnet.AGWSubnet.id
  route_table_id                            = data.azurerm_subnet.AKSSubnet.route_table_id
}
```

Use helm for

- Install Pod Identity

  https://github.com/Azure/aad-pod-identity/blob/master/charts/aad-pod-identity/README.md

- Install AGIC

  https://github.com/Azure/application-gateway-kubernetes-ingress/blob/master/docs/helm-values-documenation.md

Pod Identity - parameters

```
variable "HelmPodIdentityParam" {
  type                  = map
  description            = "A map used to feed the dynamic blocks of the pod identity helm chart"
  default                = {

      "set1" = {
        ParamName             = "nmi.allowNetworkPluginKubenet"
        ParamValue            = "true"

    }
      "set2" = {
        ParamName             = "installCRDs"
        ParamValue            = "true"

    }

  }

}
```

Pod Identity - config

```
#####################################################################
# installing pod identity from helm

resource "helm_release" "podidentity" {
  name                                = "podidentity"
  repository                          = "https://raw.githubusercontent.com/Azure/aad-pod-identity/master/charts"
  chart                               = "aad-pod-identity"
  version                             = var.PodIdChartVer


  dynamic "set" {
    for_each                          = var.HelmPodIdentityParam
    iterator                          = each
    content {
      name                            = each.value.ParamName
      value                           = each.value.ParamValue
    }

  }


}
```

AGIC – config in yaml file

```
verbosityLevel: 3

################################################################################
# Specify which application gateway the ingress controller will manage
appgw:
    subscriptionId: ${subid}
    resourceGroup: ${rgname}
    name: ${agicname}
    usePrivateIP: false
    shared: false
################################################################################
# Specify the authentication with Azure Resource Manager
armAuth:
    type: aadPodIdentity
    identityResourceID: ${PodIdentityId}
    identityClientID:  ${PodIdentityclientId}

################################################################################
# Specify if the cluster is RBAC enabled or not
rbac:
    enabled: ${IsRBACEnabled}
```

AGIC – rendering the yaml config

```
#####################################################################
# installing AGIC from helm

data "template_file" "agicyamlconfig" {
  template                                  = file("./template/agicyamlconfig.yaml")
  vars = {
    subid                                    = data.azurerm_subscription.current.subscription_id
    rgname                                   = data.azurerm_resource_group.AKSRG.name
    agicname                                 = data.terraform_remote_state.AKSClus1.outputs.AGWName
    PodIdentityId                            = module.UAIAGIC.FullUAIOutput.id
    PodIdentityclientId                      = module.UAIAGIC.FullUAIOutput.client_id
    IsRBACEnabled                            = true
  }

}
```

AGIC - config

```
resource "helm_release" "agic" {
  name                                    = "agic"
  repository                              = "https://appgwingress.blob.core.windows.
net/ingress-azure-helm-package/"
  chart                                   = "ingress-azure"
  version                                 = var.AgicChartVer
  namespace                               = "agic"
  create_namespace                        = true

  values = [data.template_file.agicyamlconfig.rendered]



}
```

AGIC - Test

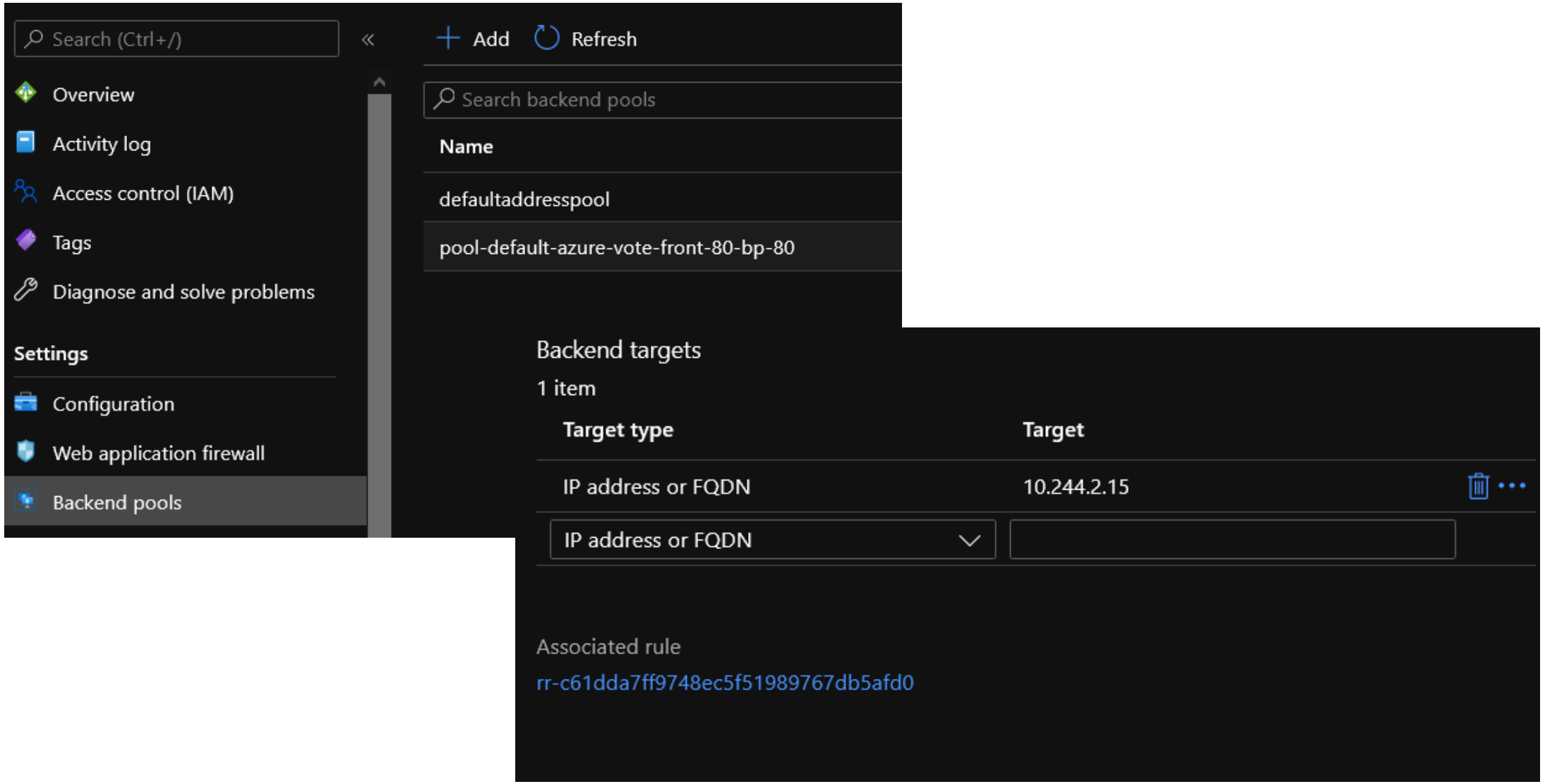
```yaml
---
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: azurevoteingress
  annotations:
    kubernetes.io/ingress.class: azure/application-gateway
    appgw.ingress.kubernetes.io/appgw-ssl-certificate: self-signed-aks-teknews-cloud
spec:
  rules:
  - http:
      paths:
      - backend:
          serviceName: azure-vote-front
          servicePort: 80
```

AGIC - Test

```
PS C:\Users\AKSAGICMeetup\03_TFForKube_helm> kubectl get service
NAME               TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
azure-vote-back    ClusterIP   10.0.132.226   <none>        6379/TCP       2m37s
azure-vote-front   NodePort    10.0.157.38    <none>        80:31851/TCP   2m37s
kubernetes         ClusterIP   10.0.0.1       <none>        443/TCP        69m

PS C:\Users\AKSAGICMeetup\03_TFForKube_helm> kubectl get ingress
NAME               CLASS    HOSTS   ADDRESS        PORTS   AGE
azurevoteingress   <none>   *       51.138.81.42   80      2m46s
PS C:\Users\davidfrappart\Documents\IaC\Azure\AKSAGICMeetup\03_TFForKube_helm> kubectl describe ingress
azurevoteingress
Name:             azurevoteingress
Namespace:        default
Address:          51.138.81.42
Default backend:  default-http-backend:80 (<error: endpoints "default-http-backend" not found>)
Rules:
  Host        Path  Backends
  ----        ----  --------
  *
                    azure-vote-front:80 (10.244.2.15:80)
Annotations:  appgw.ingress.kubernetes.io/appgw-ssl-certificate: self-signed-aks-teknews-cloud
              kubernetes.io/ingress.class: azure/application-gateway
Events:       <none>
```

AGIC - Test


Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Settings
Configuration
Web application firewall
Backend pools
Add
Refresh
Search backend pools
Name
defaultaddresspool
pool-default-azure-vote-front-80-bp-80
Backend targets
1 item
Target type
Target
IP address or FQDN
10.244.2.15
IP address or FQDN
Associated rule
rr-c61dda7ff9748ec5f51989767db5afd0

Thank you for attention!