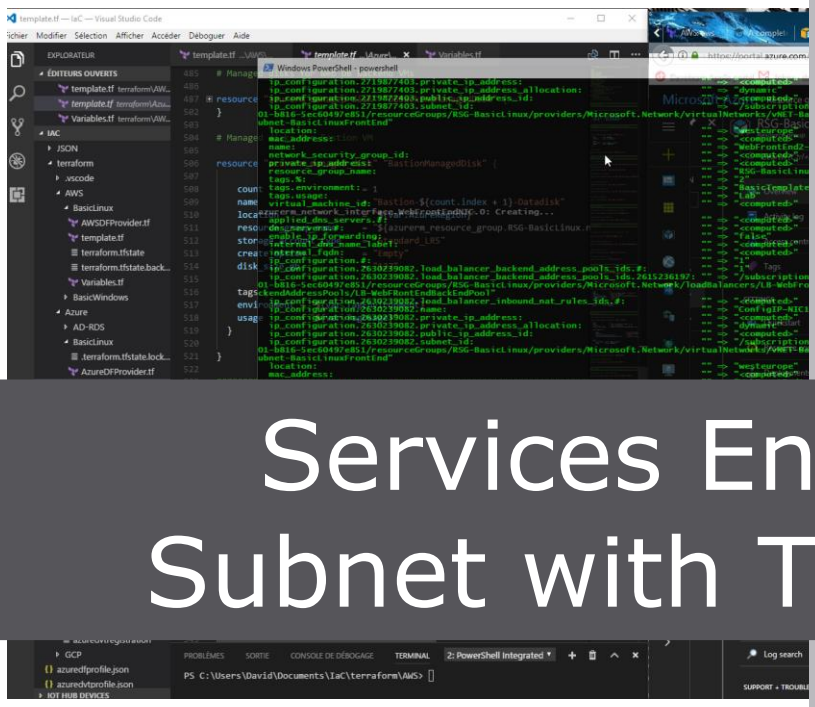


Teknews.cloud

# Services EndPoint in Subnet with Terraform

David Frappart

21/01/2018

## Contributor

| Name | Title | Email |
|---|---|---|
| David Frappart | Cloud Architect | david@teknews.cloud |
| | | |
| | | |

## Version Control

| Version | Date | State |
|---|---|---|
| 1.0 | 2018/07/16 | Final |
| | | |
| | | |

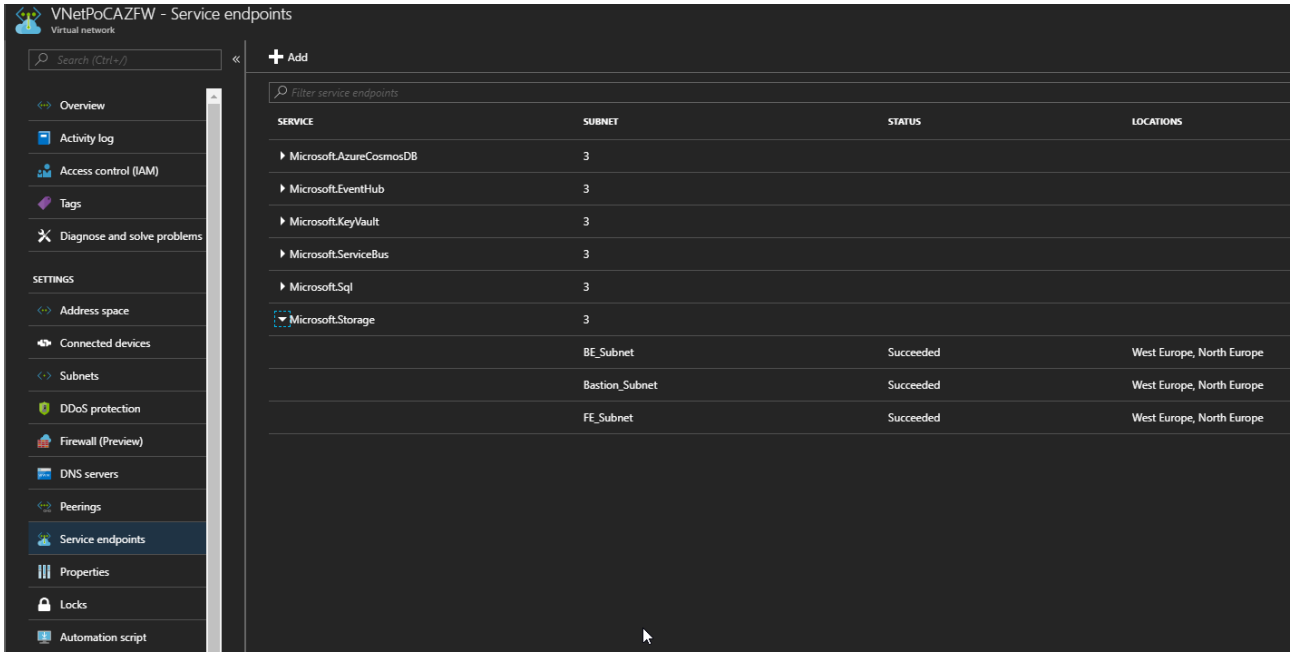
## Table of Contents

# 1 Introduction

VNet have been around since quite some time. As we already know, those allow the Network isolation of IaaS Resources, mainly Azure virtual machine.
On the other hand, PaaS service such as Azure storage or Azure SQL have long been available but only through public IP addresses. While the traffic originating from a IaaS VM in a subnet was not really going on the Internet, this justification was often considered as not enough for Securty guys. So Cloud providers listened to customers and Microsoft came over with the Service Enpoint in Azure subnet.

# 2 The service Endpoint in Azure Subnet – a short reminder

Answering the requirement of its customers regarding access to its PaaS offer, Microsoft added the Services Endpoints from virtual Network, more specifically from subnet in Azure virtual Network.
The concept is quite easy: following interrogations from Security teams, Microsoft updated its PaaS services so that they could be reached from Private IPs in VNet without requiring the addition of a public IP on an Azure VM initiating traffic to let's say Azure storage or Azure SQL.
The solution allows specific subnet to access a choice of PaaS Service simply by adding the chosen Endpoint in the portal:




Currently, the following services are available through Endpoints:

- AzureCosmosDB,
- KeyVault,

- Azure Sql,
- Azure Storage,
- Azure ServiceBus,
- AzureEventHub

# 3 A look at the ARM JSON Config

As mentioned, the service endpoint is added on a subnet. The following is an extract of a subnet created through JSON, for which the services endpoints are added:

```
{
    "$schema": "http://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "name": {
            "type": "String"
        },
        "location": {
            "type": "String"
        },
        "addressPrefix": {
            "type": "String"
        },
        "subnetName": {
            "type": "String"
        },
        "subnetAddressPrefix": {
            "type": "String"
        }
    },
    "resources": [
        {
            "type": "Microsoft.Network/virtualNetworks",
            "name": "[parameters('name')]",
            "apiVersion": "2018-02-01",
            "location": "[parameters('location')]",
            "properties": {
                "addressSpace": {
                    "addressPrefixes": [
                        "[parameters('addressPrefix')]"
                    ]
                },
```

```
                "subnets": [
                    {
                        "name": "[parameters('subnetName')]",
                        "properties": {
                            "addressPrefix": "[parameters('subnetAddressPrefix')]",
                            "serviceEndpoints": [
                                {
                                    "service": "Microsoft.AzureCosmosDB"
                                },
                                {
                                    "service": "Microsoft.KeyVault"
                                },
                                {
                                    "service": "Microsoft.Sql"
                                },
                                {
                                    "service": "Microsoft.Storage"
                                },
                                {
                                    "service": "Microsoft.ServiceBus"
                                },
                                {
                                    "service": "Microsoft.EventHub"
                                }
                            ]
                        }
                    }
                ]
            }
        }
    ]
}
```

As one can read, the service endpoints are added as a list in the JSON with the property servicEndpoints

# 4 Updating the Subnet Module in Terraform

So now, people will wonder. Indeed, the Endpoints have been available since a few month now. And tha's true. On the other hand, it was not available through terraform at the same time as it was either in the portal are in ARM JSON.

That being said, it is now available (but i did not check for which 0.11.x relase it became available). So in the following part i display my own version of a Terraform module which i modified to take Endpoints into account.

## 4.1 Terraform config

On Terraform documentation, one can see discreet (in the social meaning, not the mathematical one) property in the subnet. This property is simply called **service_endpoints**. Following is my own module code:

```
################################################################
#This module allows the creation of a Subnet
################################################################

#Variable declaration for Module

variable "SubnetName" {
  type    = "string"
  default = "DefaultSubnet"
}

variable "RGName" {
  type    = "string"
  default = "DefaultRSG"
}

variable "vNetName" {
  type = "string"
}

variable "Subnetaddressprefix" {
  type = "string"
}

variable "NSGid" {
  type = "string"
}

variable "SVCEP" {
  type    = "list"
  default = ["Microsoft.AzureCosmosDB", "Microsoft.KeyVault", "Microsoft.Sql",
"Microsoft.Storage", "Microsoft.ServiceBus", "Microsoft.EventHub"]
}

variable "EnvironmentTag" {
```

```
  type    = "string"
  default = "Poc"
}

variable "EnvironmentUsageTag" {
  type    = "string"
  default = "Poc usage only"
}

#Creation fo the subnet

resource "azurerm_subnet" "TerraSubnet" {
  name                      = "${var.SubnetName}"
  resource_group_name       = "${var.RGName}"
  virtual_network_name      = "${var.vNetName}"
  address_prefix            = "${var.Subnetaddressprefix}"
  network_security_group_id = "${var.NSGid}"
  service_endpoints         = "${var.SVCEP}"
}

#Output

output "Name" {
  value = "${azurerm_subnet.TerraSubnet.name}"
}

output "Id" {
  value = "${azurerm_subnet.TerraSubnet.id}"
}

output "AddressPrefix" {
  value = "${azurerm_subnet.TerraSubnet.address_prefix}"
}

output "RGName" {
  value = "${azurerm_subnet.TerraSubnet.resource_group_name}"
}
```

I put a default value for the SVEP (Service Endpoint) variable with a list of all available endpoint. Now it does not mean that we have to activate all the endpoints on a subnet created with this module but rather that it is possible to bypass the variable if we agree with this default value.

## 4.2 Testing and checking the deployment

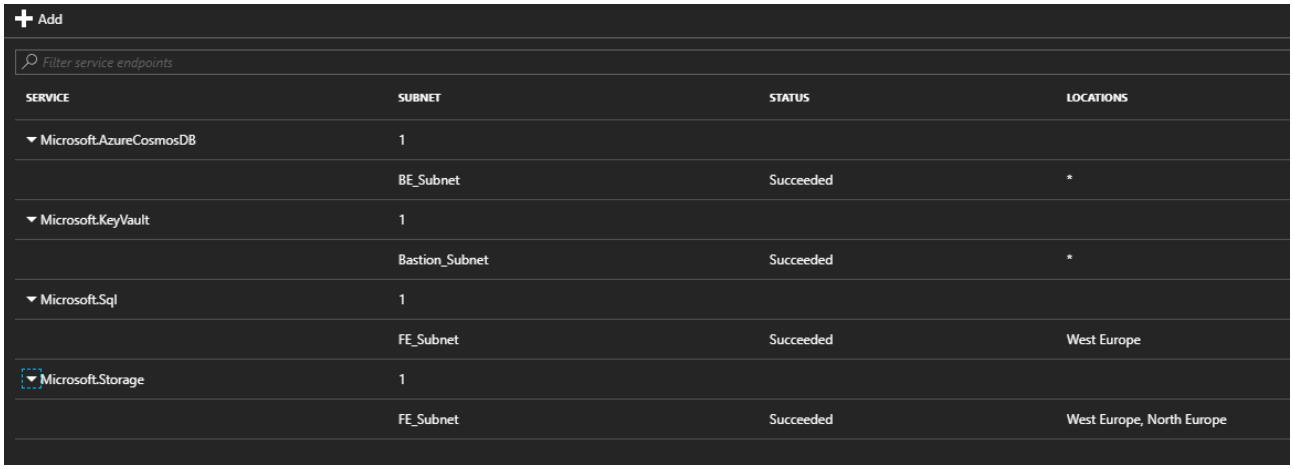
Now that we had a look on the module, let's trythe following: deploy a simple VNet with 3 Subnet. The Front End subnet access Azure storage and Azure SQL. The backend subnet access only the CosmosDB Endpoint while the Bastion Subnet access only the Keyvault. We get the following config:

```
#########################################################
# This file deploys the subnet and NSG for
#Basic linux architecture Architecture
#########################################################

#########################################################################
# Subnet and NSG
#########################################################################

#########################################################################
# Bastion zone
#########################################################################

#Bastion_Subnet NSG

module "NSG_Bastion_Subnet" {
  #Module location
  source = "./Modules/07 NSG"

  #Module variable
  NSGName             = "NSG_${lookup(var.SubnetName, 2)}"
  RGName              = "${module.ResourceGroupInfra.Name}"
  NSGLocation         = "${var.AzureRegion}"
  EnvironmentTag      = "${var.EnvironmentTag}"
  EnvironmentUsageTag = "${var.EnvironmentUsageTag}"
}

#Bastion_Subnet

module "Bastion_Subnet" {
  #Module location
  source = "./Modules/06 Subnet"

  #Module variable
  SubnetName          = "${lookup(var.SubnetName, 2)}"
  RGName              = "${module.ResourceGroupInfra.Name}"
  vNetName            = "${module.SampleArchi_vNet.Name}"
  Subnetaddressprefix = "${lookup(var.SubnetAddressRange, 2)}"
```

```
  NSGid               = "${module.NSG_Bastion_Subnet.Id}"
  SVCEP               = ["Microsoft.KeyVault"]
  EnvironmentTag      = "${var.EnvironmentTag}"
  EnvironmentUsageTag = "${var.EnvironmentUsageTag}"
}

#############################################################################
# FE zone
#############################################################################

#FE_Subnet NSG

module "NSG_FE_Subnet" {
  #Module location
  source = "./Modules/07 NSG"

  #Module variable
  NSGName             = "NSG_${lookup(var.SubnetName, 0)}"
  RGName              = "${module.ResourceGroupInfra.Name}"
  NSGLocation         = "${var.AzureRegion}"
  EnvironmentTag      = "${var.EnvironmentTag}"
  EnvironmentUsageTag = "${var.EnvironmentUsageTag}"
}

#FE_Subnet

module "FE_Subnet" {
  #Module location
  source = "./Modules/06 Subnet"

  #Module variable
  SubnetName          = "${lookup(var.SubnetName, 0)}"
  RGName              = "${module.ResourceGroupInfra.Name}"
  vNetName            = "${module.SampleArchi_vNet.Name}"
  Subnetaddressprefix = "${lookup(var.SubnetAddressRange, 0)}"
  NSGid               = "${module.NSG_FE_Subnet.Id}"
  SVCEP               = ["Microsoft.Storage", "Microsoft.Sql"]
  EnvironmentTag      = "${var.EnvironmentTag}"
  EnvironmentUsageTag = "${var.EnvironmentUsageTag}"
}

#############################################################################
# BE zone
#############################################################################
```

```
#BE_Subnet NSG

module "NSG_BE_Subnet" {
  #Module location
  source = "./Modules/07 NSG"

  #Module variable
  NSGName             = "NSG_${lookup(var.SubnetName, 1)}"
  RGName              = "${module.ResourceGroupInfra.Name}"
  NSGLocation         = "${var.AzureRegion}"
  EnvironmentTag      = "${var.EnvironmentTag}"
  EnvironmentUsageTag = "${var.EnvironmentUsageTag}"
}

#FE_Subnet

module "BE_Subnet" {
  #Module location
  source = "./Modules/06 Subnet"

  #Module variable
  SubnetName          = "${lookup(var.SubnetName, 1)}"
  RGName              = "${module.ResourceGroupInfra.Name}"
  vNetName            = "${module.SampleArchi_vNet.Name}"
  Subnetaddressprefix = "${lookup(var.SubnetAddressRange, 1)}"
  NSGid               = "${module.NSG_BE_Subnet.Id}"
  SVCEP               = ["Microsoft.AzureCosmosDB"]
  EnvironmentTag      = "${var.EnvironmentTag}"
  EnvironmentUsageTag = "${var.EnvironmentUsageTag}"
}
```

Using this config, I did get the expected result as displayed in the screenshot below:




# 5 Conclusion

In this article, we had a look at the service endpoint on the subnet side, and how to configure it using terraform. In a future article, we will look on the other side of the IaaS / PaaS private access with the storage account and Azure SQL.
For those interested in the module update, it is available on my github here.

# Another Tech blog

My thoughts and experiences on Cloud related tek