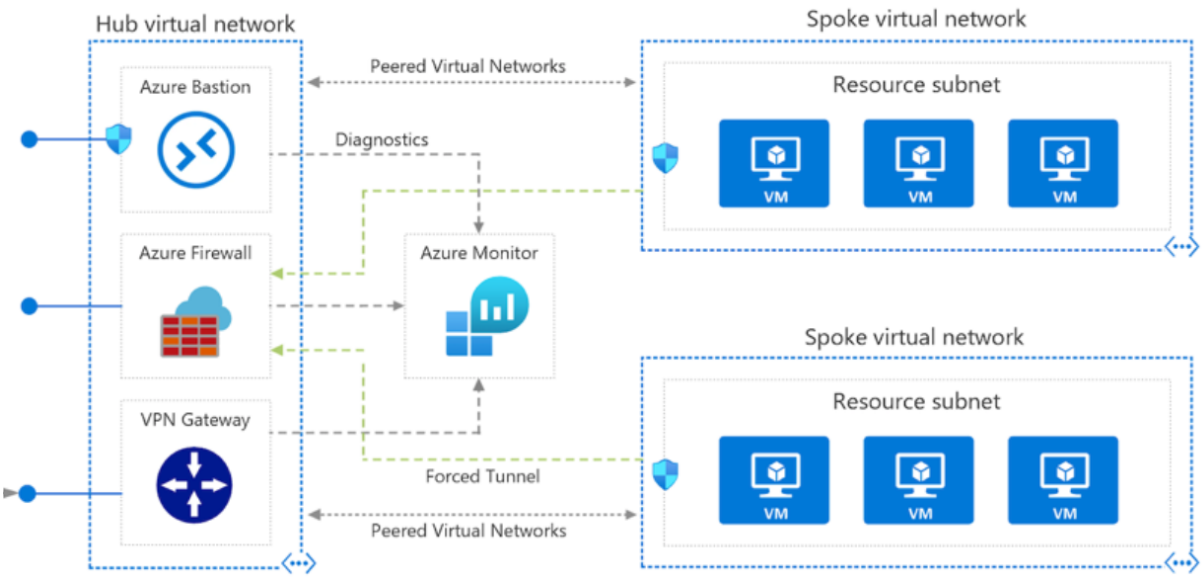
Chmurowisko

Szkolenie Terraform: Dzień 3

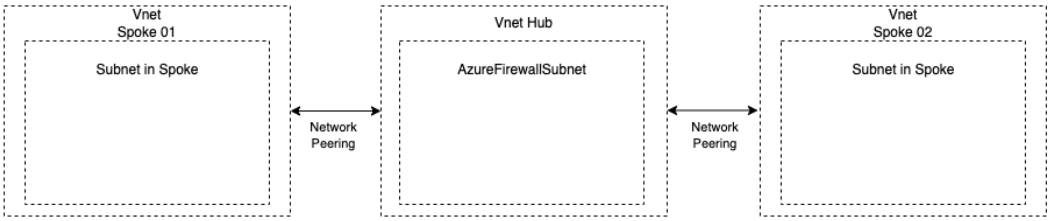
1. Zadanie: Sieć Hub and Spoke

Przykładowa architektura rozwiązania zbudowanego w sieci Hub & Spoke widoczna jest na Rys. *1*.



Rys. 1 Przykładowa architektura sieci Hub & Spoke

Architektura sieciowa oraz zasoby tworzone w ramach zadania 1 są widoczne na Rys. 2.



Rys. 2 Architektura sieciowa Zad 1.

Opis zadania: Należy zbudować strukturę sieciową przedstawioną na Rys. 2.

Należy utworzyć trzy sieci wirtualne (nie kolidujące adresacją) oraz w każdej z nich utworzyć 1 subnet po czym utworzyć peering między sieciami spoke a siecią hub zgodnie z rysunkiem.

Peering należy utworzyć w obie strony (w stronę sieci Spoke i w stronę sieci Hub). Podczas tworzenia peeringu należy nadać wartość „true" w parametrach: allow_virtual_network_access oraz allow_forwarded_traffic.

UWAGA: W podsieci w sieci VNet Hub, będzie tworzony Azure Firewall więc musi nazywać się „AzureFirewallSubnet" oraz posiadać maskę /26 np. 10.0.0.0/26.

Przykładowa adresacja sieci:

- Vnet Hub – 10.0.0.0/16
- Vnet Spoke 1 – 10.1.0.0/16
- Vnet Spoke 2 – 10.2.0.0/16
- Subnet in Hub – 10.0.0.0/26 (musi nazywać się „AzureFirewallSubnet”).
- Subnet in Spoke 1 – 10.1.0.0/24
- Subnet in Spoke 2 – 10.2.0.0/24

Zasoby które należy utworzyć:

- Sieć wirtualna Vnet Hub
- Sieć wirtualna Vnet Spoke 1
- Sieć wirtualna Vnet Spoke 2
- Subnet w sieci wirtualnej Vnet Hub
- Subnet w sieci wirtualnej Vnet Spoke 1
- Subnet w sieci wirtualnej Vnet Spoke 2
- Peering z sieci Vnet Hub do sieci Vnet Spoke 1
- Peering z sieci Vnet Spoke 1 do sieci Vnet Hub
- Peering z sieci Vnet Hub do sieci Vnet Spoke 2
- Peering z sieci Vnet Spoke 2 do sieci Vnet Hub

Przydatne linki:
Terraform dokumentacja sieci wirtualnych
https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/virtual_network

Terraform dokumentacja subnetów
https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/subnet

Terraform dokumentacja peeringu
https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/virtual_network_peering

Przykładowe rozwiązanie znajduje się w folderze „1-HubAndSpoke”

2. Zadanie: Aliasowanie providerów

Korzystając z kodu z poprzedniego zadania, przerób go tak, by wszystkie zasoby dotyczące „Hub" były utworzone w jednej subskrypcji, a wszystkie dotyczące „Spoke" w drugiej subskrypcji.

W celu utworzenia zasobów na różnych subskrypcjach, musisz wykorzystać kilka providerów (jeden provider auzrerm działa w zakresie jednej subskrypcji). Aby móc korzystać z kilku providerów tego samego rodzaju, musisz nadać im aliasy np. hub, spoke.

W tym zadaniu będziesz musiał dodać także, kolejny data source dla kolejnej grupy zasobów. Przy każdym aktualnie zdefiniowanym zasobie, będziesz musiał podać wskazanie z jakiego providera ma korzystać ten zasób. Zwróć uwagę, by poprawnie wskazać provider.

Po zmianach, przetestuj czy Twój kod nadal działa i wszystko rozstawia się poprawnie.

Pomocne linki:
Terraform dokumentacja na temat aliasowania providerów
https://developer.hashicorp.com/terraform/language/providers/configuration#alias-multiple-provider-configurations

3. Zadanie: Provisioners

W tym zadaniu wypróbujesz działanie provisioners w Terraform, poznasz działanie file, local-exec i remote-exec provisioner. Każdy z dostępnych provisioner zdefiniuj w zasobie wirtualnej maszyny z publicznym adresem IP.

Provisioner file – Utwórz plik z wiadomością .md, która chcesz przesłać na maszynę wirtualną po utworzeniu i skonfiguruj provisioner tak, by przesłać go na maszynę.

Provisioner remote-exec – Przy jego pomocy spróbuj zainstalować NGINX na maszynie wirtualnej.

Provisioner local-exec – Wykorzystaj go tak, by zapisać w pliku na lokalnej maszynie publiczny adres IP utworzonej maszyny.

Przydatne linki:
Terraform dokumentacja provisioners
https://www.terraform.io/language/resources/provisioners/syntax

Terraform dokumetnacja provisioner file
https://www.terraform.io/language/resources/provisioners/file

Terraform dokumentacja provisioner local-exec
https://www.terraform.io/language/resources/provisioners/local-exec

Terraform dokumentacja provisoner remote-exec
https://www.terraform.io/language/resources/provisioners/remote-exec

4. Zadanie: Wykorzystywanie gotowych modułów

Przygotuj nowy projekt terraform, skonfiguruj provider Azurerm i przygotuj data source pobierający Twoją grupę zasobów.

W następnym kroku utwórz sieć wirtualną i trzy podsieci z wykorzystaniem modułu z terraform registry: https://registry.terraform.io/modules/Azure/network/azurerm/latest

Po utworzeniu sieci, utwórz maszynę wirtualną (Ubuntu) wykorzystując moduł: https://registry.terraform.io/modules/Azure/compute/azurerm/latest. Podczas tworzenia wskaż id podsieci stworzonej w module 1 (sprawdź jakie wyjścia są w module tworzącym sieć).

Podczas tworzenia VM wykorzystaj autentykacji przy pomocy hasła i ustaw rozmiar „Standard_D2as_v5". Wyszukaj w dokumentacji jak ustawić te parametry.

Sprawdź utworzone moduły z poziomu portalu Azure.

5. Zadanie: Moduł Storage Account

Przygotuj moduł, który tworzący Storage Account.

Moduł powinien tworzyć takie zasoby:

- Storage Account,
- Storage Account Container (z wykorzystaniem for_each).

Moduł powinien przyjmować takie wejścia (wraz z odpowiednim typowaniem):

- Storage_account_name
- Resource_group_name
- Location
- Containers (mapa obiektów)

Zmienne wejściowe powinny być przekazany do odpowiednich argumentów w zasobach. Pozostałe wymagane pola możesz uzupełnić dowolnie.

Moduł powinien posiadać takie wyjścia (ouputs):

- Id storage account

Po przygotowaniu modułu, spróbuj przy jego pomocy utworzyć dwa storage account.

6. Zadanie: Moduł VM

Przygotuj moduł, który tworzy maszynę wirtualną, interfejs sieciowy (wraz z opcjonalnym publicznym adresem ip). Dodaj odpowiedni alias dla providera azure w konfiguracji modułu np. „virtual-machine”.

Moduł powinien tworzyć takie zasoby:

- Publiczny adres ip (jeśli odpowiednia zmienna będzie przyjmować wartość true),
- Interfejs sieciowy
- Maszyna wirtualna linux (logowanie przy pomocy loginu i hasła)

Moduł powinien przyjmować takie wejścia (wraz z odpowiednim typowaniem):

- Use_public_ip
- Resource_group_name
- Location
- Admin_username
- Admin_password
- Virtual_machine_name
- Subnet_id
- Source_image (obiekt zawierający dane na temat wykorzystywanego obrazu systemu).

Na wyjściu z modułu ustaw:

- ID Maszyny wirtualnej
- Publiczny adres IP maszyny
- Prywatny adres IP maszyny

W przypadku zmiennej wejściowej przechowującej dane na temat systemu operacyjnego ustaw domyślną wartość z Ubuntu 16.04.

Jeśli jakieś zmienne nie są podane, przyjmij stałe wartości np. rozmiar maszyny.

Po przygotowaniu modułu, przetestuj go tworząc dwie maszyny wirtualne z jego wykorzystaniem. Tylko jedna maszyna powinna posiadać publiczny adres IP.

7. Zadanie: Moduł rozstawiający sieć Spoke

W ramach tego zadania, należy wydzielić kod rozstawiający sieć spoke do osobnego modułu. Moduł powinien także tworzyć dwustronny peering pomiędzy siecią spoke, a hub. (Będzie wymagało to zastosowania dwóch providerów w ramach jednego modułu.

W module powinny znaleźć się cztery zasoby:

- Sieć wirtualna
- Podsieć (powinna być tworzona z wykorzystaniem for_each)
- Peering z sieci hub do sieci spoke
- Peering z sieci spoke do sieci hub

Moduł powinien przyjmować takie zmienne (wraz z odpowiednim typowaniem):

- Spoke_resource_group_name
- Spoke_number (wykorzystywany w nazwach)
- Location
- Address_prefix
- Subnets (Mapa wykorzystywana w for_each przy tworzeniu podsieci)
- Hub_virtual_network_id
- Hub_virtual_network_name
- Hub_resource_group_name

Zmienne związane z siecią hub są potrzebne do peeringu z hub do spoke.

Na wyjściu z modułu ustaw (będą one potrzebne w kolejnym zadaniu):

- Id utworzonej sieci spoke
- Nazwę utworzonej sieci spoke
- Mapę utworzonych podsieci

Zmienne związane z siecią hub są potrzebne do peeringu z hub do spoke.

Po przygotowaniu modułu zastąp aktualne tworzenie sieci spoke w konfiguracji z wykorzystaniem nowo utworzonego modułu. Przetestuj, czy zasoby są poprawnie tworzone.

Przydatne linki:
Terraform dokumentacja aliasowanie providerów
https://www.terraform.io/language/providers/configuration#alias-multiple-provider-configurations