Terraform, variables, interpolation and networking

variables

- tipos nativos: strings, maps (key=value), list ([])
  - boolean no es nativo
- localización:
  - en el cuerpo de script
  - por línea de comando o entono
  - en un fichero adicional de variables
  - más información: terraform variables

- ```
  variable "nombre_variable" {
    default = "valor por defecto"
  }

  variable "localizaciones" {
    type = map
    default = {
      location1 = "xxxx"
      location2 = "yyyy"
    }
  }
  ```

- testeando variables y su uso:

  ```
  variable "server_name" {
      default = "web-server"
  }

  variable "locations" {
      type = "map"
      default = {
          location1 = "xxx"
          location2 = "yyy"
      }
  }

  variable "subnets" {
      type = "list"
      default = ["10.0.1.10","10.0.1.11"]
  }

  variable "live" {
    type = "string"
    default = true
  }
  ```

  - `terraform console`
    - var.server_name
    - var.locations[«location1»]
    - var.locations.location1
  ```

    - var.subnets[0]
    - var.live

credentials

- uso variables [de entorno] para almacenar credenciales
- **TF_VAR_<NOMBRE>** definida como variable de entorno del sistema

interpolation

- variables
    - `var.<nombre>`
- resources
    - `<resource>.<nombre>.<atributo>`
- data: recoger un dato del estado
    - `data.<resource>.<nombre>.<atributo>`

-
```
variable "subscription_id" {

}

#suponiendo que está en el entorno
provider "azurerm" {
    version         = ">=1.27"
    client_id       = "6c609608-a350-476f-85ff-eaff7e82f981"
    client_secret   = "qW0S]tToc]=]ub8HM7yciLSnwOWh1emD"
    tenant_id       = "c8537169-fe81-494a-a2dd-b4ea067073a1"
    subscription_id = var.subscription_id
    features {}
}
```

azure locations

- https://azure.microsoft.com/en-us/global-infrastructure/regions/
- https://azure.microsoft.com/en-us/global-infrastructure/geographies/
- https://azure.microsoft.com/en-us/global-infrastructure/services/
- `az login –service-principal -u $TF_VAR_client_id -p $TF_VAR_client_secret -t $TF_VAR_tenant_id`
- `az account list-locations -o table`
- `az login [–username | -u] <username> [–password | -p ] <password>`

azure resource groups

- agrupación lógica de recursos bajo tu criterio
- cada **resource* ha de ir en** resource group

- terraform.tfvars

```
web_server_location = "westus2"
web_server_rg = "web-rg"
```

main.tf

```
variable web_server_location {}
variable web_server_rg {}

resource "azurerm_resource_group" "web_server_rg" {
  name = var.web_server_rg
  location = var.web_server_location
}
```

azure VNET

- logical isolates network
- espacio IP
- creació de subnets
- conexión con otras VNET, VPN o Endpoints
- NSG = Network Security Groups (firewall básico)

```
resource_prefix = "web-server"
web_server_address_space = "1.0.0.0/22"
```

```
variable "resource_prefix" {}
variable "web_server_address_space" {}

resource "azurerm_virtual_network" "web_server_vnet" {
  name = "${var.resource_prefix}-vnet"
  location = var.web_server_location
  resource_group_name = azurerm_resource_group.web_server_rg.name
  address_space = [var.web_server_address_space]
}
```

dependencies

- decirle a Terraform en que orden se deben crear los recursos
- al crear elementos que dependen (o no) de otros, se pueden dar los siguientes casos:
  - que no haya dependencia
  - dependencia indirecta: Terraform la asume o las dedice de las variables
  - dependencia directa (o explícita): usando la propiedad `depends_on`

azure subnet

- subnetwork en nuestra VNET
- address space

- segmentación
- NSG = Network Security Groups

- terraform.tfvars

```
web_server_address_prefix = "1.0.1.0/24
```

subnet.tf

```
variable web_server_address_prefix {}

resource "axzurerm_subnet" "web_server_subnet"{
  name = "${var.resource_prefix}-subnet"
  resource_group_name = azurerm_resource_group.web_server_rg.name
  virtual_network_name = azurerm_virtual_network.web_server_vnet.name
  address_prefix = var.web_server_address_prefix
}
```

azure Network Interface

- equivalencia con el mundo real
- atachar a Subnets/VNET
- IP públicas o privadas
- Estáticas o dinámicas
- DNS Settings
- NSG

- terraform.tfvars

```
web_server_name = "web-01"
```

nic.tf

```
  variable web_server_name {}
  resource "azurerm_network_interface" "web_server_nic" {
    name = "${var.web_server_name}-nic"
    location = var.web_server_location
    resource_group_name = azurerm_resource_group.web_server_rg.name

    ip_configuration {
      name = "${var.web_server_name}-ip"
      subnet_id = azurerm_subnet.web_server_subnet.id
      private_ip_address_allocation = "dynamic"
    }
  }
```