**Terraform para Arquitectura Segura con LB Público, NAT Gateway y Monitoreo**

Voy a proporcionarte una estructura modular de Terraform para implementar tu arquitectura con todas las funcionalidades de seguridad solicitadas. Organizaré el código en módulos lógicos.

**Estructura de Archivos**

modules/

├── network/

│ ├── main.tf # VNet, subnets, NSGs

│ ├── nat-gateway.tf # NAT Gateway y IP pública

├── load-balancer/

│ ├── main.tf # LB público con DDoS Protection

├── monitoring/

│ ├── main.tf # Log Analytics (qa/prod)

├── variables.tf # Variables comunes

├── outputs.tf # Outputs útiles

main.tf # Configuración principal

**1. Módulo de Red (**modules/network/main.tf**)**

resource "azurerm\_virtual\_network" "main" {

name = "vnet-main"

address\_space = ["10.0.0.0/16"]

location = var.location

resource\_group\_name = var.resource\_group\_name

}

resource "azurerm\_subnet" "backend" {

name = "backend-subnet"

resource\_group\_name = var.resource\_group\_name

virtual\_network\_name = azurerm\_virtual\_network.main.name

address\_prefixes = ["10.0.1.0/24"]

}

resource "azurerm\_subnet" "private\_endpoint" {

name = "privateendpoint-subnet"

resource\_group\_name = var.resource\_group\_name

virtual\_network\_name = azurerm\_virtual\_network.main.name

address\_prefixes = ["10.0.2.0/28"]

enforce\_private\_link\_endpoint\_network\_policies = true

}

resource "azurerm\_network\_security\_group" "backend" {

name = "nsg-backend"

location = var.location

resource\_group\_name = var.resource\_group\_name

security\_rule {

name = "Allow-LB"

priority = 100

direction = "Inbound"

access = "Allow"

protocol = "Tcp"

source\_port\_range = "\*"

destination\_port\_ranges = [80, 443]

source\_address\_prefix = "AzureLoadBalancer"

destination\_address\_prefix = "\*"

}

security\_rule {

name = "Deny-All-Inbound"

priority = 4000

direction = "Inbound"

access = "Deny"

protocol = "\*"

source\_port\_range = "\*"

destination\_port\_range = "\*"

source\_address\_prefix = "\*"

destination\_address\_prefix = "\*"

}

}

**2. Módulo de NAT Gateway (**modules/network/nat-gateway.tf**)**

resource "azurerm\_public\_ip" "nat" {

name = "nat-gw-ip"

location = var.location

resource\_group\_name = var.resource\_group\_name

allocation\_method = "Static"

sku = "Standard"

}

resource "azurerm\_nat\_gateway" "main" {

name = "nat-gateway"

location = var.location

resource\_group\_name = var.resource\_group\_name

sku\_name = "Standard"

}

resource "azurerm\_nat\_gateway\_public\_ip\_association" "main" {

nat\_gateway\_id = azurerm\_nat\_gateway.main.id

public\_ip\_address\_id = azurerm\_public\_ip.nat.id

}

resource "azurerm\_subnet\_nat\_gateway\_association" "backend" {

subnet\_id = azurerm\_subnet.backend.id

nat\_gateway\_id = azurerm\_nat\_gateway.main.id

}

**3. Módulo de Load Balancer (**modules/load-balancer/main.tf**)**

resource "azurerm\_public\_ip" "lb" {

name = "lb-public-ip"

location = var.location

resource\_group\_name = var.resource\_group\_name

allocation\_method = "Static"

sku = "Standard"

ddos\_protection\_mode = "Enabled"

}

resource "azurerm\_lb" "main" {

name = "public-lb"

location = var.location

resource\_group\_name = var.resource\_group\_name

sku = "Standard"

frontend\_ip\_configuration {

name = "PublicIPAddress"

public\_ip\_address\_id = azurerm\_public\_ip.lb.id

}

}

resource "azurerm\_lb\_backend\_address\_pool" "main" {

loadbalancer\_id = azurerm\_lb.main.id

name = "BackEndAddressPool"

}

**4. Módulo de Monitoreo (**modules/monitoring/main.tf**)**

resource "azurerm\_log\_analytics\_workspace" "qa" {

name = "logs-qa"

location = var.location

resource\_group\_name = var.resource\_group\_name

sku = "PerGB2018"

retention\_in\_days = 30

}

resource "azurerm\_log\_analytics\_workspace" "prod" {

name = "logs-prod"

location = var.location

resource\_group\_name = var.resource\_group\_name

sku = "PerGB2018"

retention\_in\_days = 30

}

resource "azurerm\_monitor\_diagnostic\_setting" "lb" {

name = "lb-monitoring"

target\_resource\_id = azurerm\_lb.main.id

log\_analytics\_workspace\_id = var.environment == "prod" ? azurerm\_log\_analytics\_workspace.prod.id : azurerm\_log\_analytics\_workspace.qa.id

metric {

category = "AllMetrics"

}

}

**Archivo Principal (**main.tf**)**

module "network" {

source = "./modules/network"

resource\_group\_name = azurerm\_resource\_group.main.name

location = var.location

}

module "nat\_gateway" {

source = "./modules/network/nat-gateway"

resource\_group\_name = azurerm\_resource\_group.main.name

location = var.location

subnet\_id = module.network.backend\_subnet\_id

}

module "load\_balancer" {

source = "./modules/load-balancer"

resource\_group\_name = azurerm\_resource\_group.main.name

location = var.location

}

module "monitoring" {

source = "./modules/monitoring"

resource\_group\_name = azurerm\_resource\_group.main.name

location = var.location

environment = var.environment

}

**Variables (**variables.tf**)**

variable "location" {

description = "Azure region"

type = string

default = "eastus"

}

variable "resource\_group\_name" {

description = "Name of the resource group"

type = string

default = "rg-movie-analyst"

}

variable "environment" {

description = "Environment (qa/prod)"

type = string

validation {

condition = contains(["qa", "prod"], var.environment)

error\_message = "Environment must be qa or prod."

}

}

**Implementación**

1. Inicializa Terraform:

terraform init

1. Revisa el plan:

terraform plan -var="environment=prod"

1. Aplica los cambios:

terraform apply -var="environment=prod"

**Características Clave Implementadas**

* **DDoS Protection Standard** en la IP del LB
* **NSGs restrictivos** (solo puertos 80/443 desde el LB)
* **NAT Gateway para tráfico saliente seguro**
* **Monitoreo separado para QA/PROD** con Log Analytics
* **Arquitectura modular** para fácil mantenimiento

**Terraform para Frontend con App Service en la Arquitectura**

Agregaré el módulo de App Service al diseño existente, integrando el frontend (movie-analyst-ui) con las configuraciones de seguridad y conectividad definidas.

**Estructura Actualizada**

text

Copy

Download

modules/

├── app-service/ # Nuevo módulo para el frontend

│ ├── main.tf

│ ├── variables.tf

├── network/

├── load-balancer/

├── monitoring/

main.tf # Actualizado para incluir App Service

**1. Módulo de App Service (**modules/app-service/main.tf**)**

hcl

Copy

Download

resource "azurerm\_app\_service\_plan" "main" {

name = "asp-${var.app\_name}-${var.environment}"

location = var.location

resource\_group\_name = var.resource\_group\_name

kind = "Linux"

reserved = true *# Necesario para Linux*

sku {

tier = var.sku\_tier

size = var.sku\_size

}

}

resource "azurerm\_app\_service" "main" {

name = "app-${var.app\_name}-${var.environment}"

location = var.location

resource\_group\_name = var.resource\_group\_name

app\_service\_plan\_id = azurerm\_app\_service\_plan.main.id

site\_config {

linux\_fx\_version = "NODE|14-lts" *# Ajusta según tu runtime*

always\_on = true

}

app\_settings = {

"API\_BASE\_URL" = "http://${var.lb\_public\_ip}/api" *# Usa la IP del LB*

"NODE\_ENV" = var.environment

"WEBSITES\_PORT" = "3000" *# Puerto que usa tu app Node.js*

}

identity {

type = "SystemAssigned"

}

}

*# Integración con VNet (opcional - solo si necesitas acceso a recursos privados)*

resource "azurerm\_app\_service\_virtual\_network\_swift\_connection" "vnet\_integration" {

count = var.enable\_vnet\_integration ? 1 : 0

app\_service\_id = azurerm\_app\_service.main.id

subnet\_id = var.integration\_subnet\_id

}

**2. Variables del Módulo (**modules/app-service/variables.tf**)**

hcl

Copy

Download

variable "app\_name" {

description = "Nombre de la aplicación (ej: movie-analyst-ui)"

type = string

}

variable "environment" {

description = "Entorno (qa/prod)"

type = string

}

variable "location" {

description = "Región de Azure"

type = string

}

variable "resource\_group\_name" {

description = "Nombre del resource group"

type = string

}

variable "sku\_tier" {

description = "Tier del App Service Plan"

type = string

default = "Basic"

}

variable "sku\_size" {

description = "Tamaño del SKU"

type = string

default = "B1"

}

variable "lb\_public\_ip" {

description = "IP pública del Load Balancer para conectar al backend"

type = string

}

variable "enable\_vnet\_integration" {

description = "Habilita integración con VNet"

type = bool

default = false

}

variable "integration\_subnet\_id" {

description = "ID de la subred de integración (si enable\_vnet\_integration=true)"

type = string

default = null

}

**3. Actualización del**main.tf**Principal**

hcl

Copy

Download

module "app\_service" {

source = "./modules/app-service"

app\_name = "movie-analyst-ui"

environment = var.environment

location = var.location

resource\_group\_name = azurerm\_resource\_group.main.name

lb\_public\_ip = module.load\_balancer.lb\_public\_ip

sku\_tier = var.environment == "prod" ? "Standard" : "Basic"

*# Opcional: Integración con VNet si el frontend necesita recursos privados*

enable\_vnet\_integration = false

integration\_subnet\_id = module.network.integration\_subnet\_id

}

**4. Outputs Adicionales (**outputs.tf**)**

hcl

Copy

Download

output "app\_service\_url" {

value = module.app\_service.app\_service\_default\_hostname

}

output "app\_service\_principal\_id" {

value = module.app\_service.app\_service\_principal\_id

description = "ID de la identidad gestionada para integración con otros servicios"

}

**Configuraciones Clave**

**Seguridad del App Service**

* **Sin exposición directa a la VNet**: Por defecto, la app se comunica con el backend via LB público.
* **Identidad gestionada**: Para acceder de forma segura a Key Vault u otros servicios.
* **Solo puerto HTTP/S**: Configurado en site\_config.

**Integración con la Arquitectura Existente**

1. **Conexión al Backend**:
   * La variable lb\_public\_ip inyecta la IP del LB en las variables de entorno.
   * La app usa API\_BASE\_URL para llamar al backend.
2. **Monitoreo**:
   * Agrega esta configuración al módulo de monitoreo:

hcl

Copy

Download

resource "azurerm\_monitor\_diagnostic\_setting" "app\_service" {

name = "app-service-logs"

target\_resource\_id = module.app\_service.app\_service\_id

log\_analytics\_workspace\_id = var.environment == "prod" ? azurerm\_log\_analytics\_workspace.prod.id : azurerm\_log\_analytics\_workspace.qa.id

metric {

category = "AllMetrics"

}

log {

category = "AppServiceHTTPLogs"

enabled = true

}

}

**Implementación**

bash

Copy

Download

terraform init

terraform plan -var="environment=prod"

terraform apply -var="environment=prod"

**Características Implementadas**

* **Frontend aislado**: Corre en App Service sin requerir VNet.
* **Escalabilidad**: Diferentes SKUs para QA (B1) y PROD (S1).
* **Conectividad segura al backend**: A través del LB público con DDoS Protection.
* **Monitoreo integrado**: Logs en el workspace correspondiente.