

IaC - DevOps з використанням Terraform

Мета: Отримати практичний досвід у DevOps та підході "Інфраструктура як код" (Infrastructure-as-Code) через автоматизоване створення та розгортання хмарної інфраструктури в **Microsoft Azure** за допомогою фреймворку **Terraform**.

Завдання:

1. Описати віртуальну інфраструктуру (Група ресурсів, Мережа, Група безпеки, VM) у коді.
2. Розгорнути віртуальну інфраструктуру в Azure.
3. Перевірити роботу веб-сервера.
4. Видалити ресурси після завершення.

Лабораторне середовище:

- Visual Studio Code з встановленим плагіном *HashiCorp Terraform*.
- Встановлений **Terraform**.
- Встановлений **Azure CLI**.

Крок 1: Підготовка проекту

1. Створіть на своєму комп'ютері папку для лабораторної роботи, наприклад `Terraform-Azure-Lab`.
2. Всередині цієї папки створіть підпапку `files`.
3. Створіть 4 текстові файли з розширеннями `.tf` та `.tpl` і вставте в них наведений нижче код.

1.1. Файл `provider.tf`

Цей файл вказує Terraform, що ми будемо працювати з хмарою Azure.

```
terraform {  
    required_providers {  
        azurerm = {  
            source  = "hashicorp/azurerm"  
            version = "=3.0.0"  
        }  
    }  
}
```

```
}
```

```
provider "azurerm" {
  features {}
}
```

1.2. Файл vars-common.tf (Змінні)

Тут ми визначаємо змінні, щоб легко змінювати налаштування в одному місці.

```
variable "resource_group_name" {
  default = "Terraform-Lab-RG"
}

variable "location" {
  default = "West Europe"
}

variable "vm_name" {
  default = "my-linux-vm"
}
```

1.3. Файл files/template.tpl (Скрипт запуску)

Створіть цей файл у папці `files`. Це bash-скрипт, який автоматично встановить веб-сервер Apache при першому запуску віртуальної машини (зверніть увагу: для Azure/Ubuntu ми використовуємо `apt`, а не `yum`).

```
#!/bin/bash
# Оновлюємо пакети
sudo apt-get update
# Встановлюємо Apache, PHP та MySQL клієнт
sudo apt-get install -y apache2 php libapache2-mod-php php-mysql

# Вмикаємо та запускаємо веб-сервер
sudo systemctl enable apache2
sudo systemctl start apache2

# Завантажуємо тестовий сайт (як в оригінальній лабораторній)
if [ ! -f /var/www/html/bootcamp-app.tar.gz ]; then
  cd /var/www/html
  sudo rm index.html
  sudo wget [http://fgcom.org.uk/wp-content/uploads/2020/08/bootcamp-
app.tar](http://fgcom.org.uk/wp-content/uploads/2020/08/bootcamp-app.tar)
```

```
sudo tar xvf bootcamp-app.tar
sudo chown www-data:www-data /var/www/html/rds.conf.php
fi
```

1.4. Файл main.tf (Головна конфігурація)

Це основний файл, де описується вся інфраструктура. В Azure вона складається з багатьох пов'язаних компонентів.

```
# 1. Створюємо групу ресурсів
resource "azurerm_resource_group" "rg" {
    name      = var.resource_group_name
    location  = var.location
}

# 2. Створюємо віртуальну мережу (VNet)
resource "azurerm_virtual_network" "vnet" {
    name          = "my-vnet"
    address_space = ["10.0.0.0/16"]
    location      = azurerm_resource_group.rg.location
    resource_group_name = azurerm_resource_group.rg.name
}

# 3. Створюємо підмережу (Subnet)
resource "azurerm_subnet" "subnet" {
    name          = "internal"
    resource_group_name = azurerm_resource_group.rg.name
    virtual_network_name = azurerm_virtual_network.vnet.name
    address_prefixes = ["10.0.1.0/24"]
}

# 4. Створюємо публічну IP-адресу (Standard SKU)
resource "azurerm_public_ip" "pip" {
    name          = "my-public-ip"
    resource_group_name = azurerm_resource_group.rg.name
    location      = azurerm_resource_group.rg.location
    allocation_method = "Static" # Standard SKU вимагає Static
    sku           = "Standard" # Змінюємо Basic на Standard
}

# 5. Створюємо групу безпеки (Firewall)
resource "azurerm_network_security_group" "nsg" {
    name          = "my-nsg"
    location      = azurerm_resource_group.rg.location
    resource_group_name = azurerm_resource_group.rg.name
```

```

# Дозволяємо SSH (порт 22)
security_rule {
    name          = "SSH"
    priority      = 1001
    direction     = "Inbound"
    access        = "Allow"
    protocol      = "Tcp"
    source_port_range = "*"
    destination_port_range = "22"
    source_address_prefix = "*"
    destination_address_prefix = "*"
}

# Дозволяємо HTTP (порт 80) для веб-сайту
security_rule {
    name          = "HTTP"
    priority      = 1002
    direction     = "Inbound"
    access        = "Allow"
    protocol      = "Tcp"
    source_port_range = "*"
    destination_port_range = "80"
    source_address_prefix = "*"
    destination_address_prefix = "*"
}

}

}

# 6. Створюємо мережевий інтерфейс (NIC)
resource "azurerm_network_interface" "nic" {
    name          = "my-nic"
    location      = azurerm_resource_group.rg.location
    resource_group_name = azurerm_resource_group.rg.name

    ip_configuration {
        name          = "internal"
        subnet_id      = azurerm_subnet.subnet.id
        private_ip_address_allocation = "Dynamic"
        public_ip_address_id       = azurerm_public_ip.pip.id
    }
}

# Приєднуємо Firewall до інтерфейсу
resource "azurerm_network_interface_security_group_association" "example" {
    network_interface_id      = azurerm_network_interface.nic.id
    network_security_group_id = azurerm_network_security_group.nsg.id
}

```

```

}

# 7. Створюємо Віртуальну Машину (Ubuntu Linux)
resource "azurerm_linux_virtual_machine" "vm" {
    name          = var.vm_name
    resource_group_name = azurerm_resource_group.rg.name
    location      = azurerm_resource_group.rg.location
    size          = "Standard_B1s" # Економний розмір
    admin_username = "azureuser"

    # Передаємо скрипт запуску (встановлення Apache)
    custom_data = filebase64("./files/template.tpl")

    network_interface_ids = [
        azurerm_network_interface.nic.id,
    ]

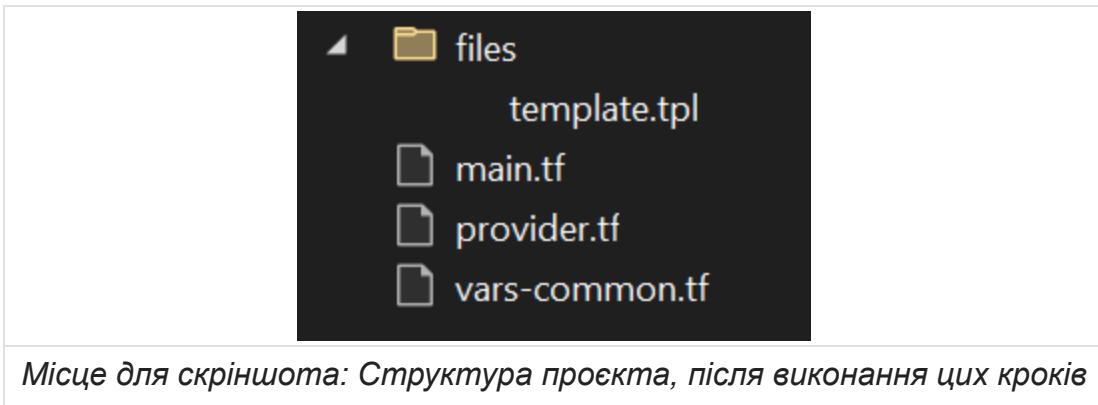
    # Вказуємо шлях до вашого SSH-ключа (створимо його на наступному кроці)
    admin_ssh_key {
        username  = "azureuser"
        public_key = file("./cloud_key.pub")
    }

    os_disk {
        caching           = "ReadWrite"
        storage_account_type = "Standard_LRS"
    }

    source_image_reference {
        publisher = "Canonical"
        offer     = "0001-com-ubuntu-server-jammy"
        sku       = "22_04-lts"
        version   = "latest"
    }
}

# Виводимо IP-адресу після створення
output "public_ip_address" {
    value = azurerm_public_ip.pip.ip_address
}

```



Крок 2: Генерація SSH-ключів

Azure Linux VM вимагають SSH-ключі для безпечноного доступу.

1. Відкрийте термінал у VS Code (Ctrl + `).
2. Переконайтесь, що ви знаходитесь у папці проекту.
3. Виконайте команду для створення ключів (натискайте Enter на всі питання):

```
ssh-keygen -t rsa -f cloud_key
```

4. Це створить два файли у вашій папці: `cloud_key` (приватний ключ) та `cloud_key.pub` (публічний ключ, який Terraform завантажить в Azure).

```
PS C:\lab5\Terraform> ssh-keygen -t rsa -f cloud_key
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in cloud_key
Your public key has been saved in cloud_key.pub
The key's randomart image is:
+---[RSA 3072]---+
| ..+*=o*..|
| ..= o .++==E|
| o .oB+=+.*|
| o.====,+o|
| S ...o o|
| . ..o |
| . +. |
| . + ..|
| .....|
+---[SHA256]---+
```

Mісце для скріншота: Виконання команди в терміналі

Крок 3: Встановлення та налаштування Azure CLI

Для того, щоб Terraform міг створювати ресурси у вашому акаунті Azure, на комп'ютері має бути встановлений та налаштований інструмент **Azure CLI**.

1. Перевірка наявності:

Відкрийте термінал у VS Code і введіть команду:

```
az --version
```

Якщо ви бачите помилку (наприклад, "*The term 'az' is not recognized*"), перейдіть до пункту 2. Якщо виводиться номер версії — переходьте одразу до пункту 4.

2. Встановлення:

Найшвидший спосіб встановити Azure CLI на Windows — виконати наступну команду в терміналі:

```
winget install -e --id Microsoft.AzureCLI
```

(Якщо система запитає підтвердження ліцензійної угоди, натисніть клавішу *Y* та *Enter*).

Альтернативний варіант: Якщо `winget` не спрацював, завантажте та запустіть MSI-інсталятор з [офіційного сайту Microsoft](#).

3. Перезапуск середовища (Важливо!):

Після завершення встановлення **повністю закрийте Visual Studio Code** і відкрийте його знову. Це необхідно, щоб термінал оновив змінні середовища і "побачив" нову команду `az`.

4. Авторизація:

Тепер авторизуйтесь у своєму акаунті, щоб надати доступ Terraform:

```
az login
```

5. Відкриється браузер. Увійдіть у свій обліковий запис Microsoft Azure (використовуйте студентський акаунт, якщо є).
6. Після успішного входу закрийте браузер і поверніться до терміналу. Ви побачите список ваших підписок.

```
Select the account you want to log in with. For more information on login with Azure CLI, see https://go.microsoft.com/fwlink/?linkid= [REDACTED]
Retrieving subscriptions for the selection...
[Tenant and subscription selection]
No Subscription name Subscription ID Tenant
--- -----
[1] * Azure subscription 1 [REDACTED]
The default is marked with an *; the default tenant is '[REDACTED]' and subscription is 'Azure subscription 1' ([REDACTED]).
Select a subscription and tenant (Type a number or Enter for no changes):
Tenant: [REDACTED]
Subscription: Azure subscription 1 ([REDACTED])
[Announcements]
With the new Azure CLI login experience, you can select the subscription you want to use more easily. Learn more about it and its configuration at https://go.microsoft.com/fwlink/?linkid=[REDACTED]
If you encounter any problem, please open an issue at https://aka.ms/azclibug
[Warning] The login output has been updated. Please be aware that it no longer displays the full list of available subscriptions by default.
```

Mісце для скріншота: Результат виконання команди az login у терміналі (спісок підписок)

Крок 4: Розгортання інфраструктури

Встановлення Terraform (якщо не встановлено)

1. Відкрийте термінал і перевірте наявність Terraform:

```
terraform -version
```

2. Якщо ви отримали помилку, встановіть його командою:

```
winget install HashiCorp.Terraform
```

3. Після встановлення **перезапустіть термінал/VS Code**.

Тепер ми готові запустити Terraform.

4. **Ініціалізація:** Завантажте необхідні модулі для Azure.

```
terraform init
```

(Має з'явитися повідомлення "Terraform has been successfully initialized!")

```
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/azurerm versions matching "3.0.0"...
- Installing hashicorp/azurerm v3.0.0...
- Installed hashicorp/azurerm v3.0.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

Місце для скріншота: Результат виконання команди terraform init у терміналі

4. Планування: Перевірте, що саме Terraform збирається створити.

```
terraform plan
```

(Ви побачите список ресурсів з плюсиками +, що означає "буде створено").

5. Застосування: Розгорніть ресурси в хмарі.

```
terraform apply
```

- Коли система запитає підтвердження Enter a value:, введіть yes і натисніть Enter.
- Чекайте завершення (це може зайняти 2-5 хвилин).

6. У кінці ви побачите повідомлення Apply complete! і вашу IP-адресу:

Outputs:

```
public_ip_address = "20.x.x.x"
```

```
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

azurerm_public_ip.pip: Creating...
azurerm_public_ip.pip: Creation complete after 3s [id=/subscr
azurerm_network_interface.nic: Creating...
azurerm_network_interface.nic: Creation complete after 3s [id
azurerm_network_interface_security_group_association.example:
azurerm_linux_virtual_machine.vm: Creating...
azurerm_network_interface_security_group_association.example:
[redacted] /resourceGroups/Terraform-Lab-RG/providers/Microso
azurerm_linux_virtual_machine.vm: Still creating... [00m10s e
azurerm_linux_virtual_machine.vm: Creation complete after 18s

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.

Outputs:

public_ip_address = "51.██████████"
```

Місце для скріншота: Результат виконання команди `terraform apply` у терміналі

Крок 5: Перевірка результатів

1. Перевірка в порталі Azure:

- Зайдіть на portal.azure.com.
- Перейдіть у "Resource groups".
- Знайдіть групу Terraform-Lab-RG . В ній ви побачите всі створені ресурси (VM, Network, Disk тощо).

Mісце для скріншота: Список ресурсів у створеній групі

2. Підключення через SSH (опціонально):

- У терміналі VS Code введіть:

```
ssh -i cloud_key azureuser@<ВАША_ІР_АДРЕСА>
```

- Ви потрапите в консоль вашої віддаленої Linux машини.

```
ssh -i cloud_key azureuser@51.101.127.222
The authenticity of host '51.101.127.222 (51.101.127.222)' can't be established.
ED25519 key fingerprint is SHA256: [REDACTED].
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '51.101.127.222' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-1041-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro
```

Mісце для скріншота: Виконання команди підключення

3. Перевірка веб-сайту:

- Скопіюйте отриману public_ip_address .
- Відкрийте браузер і вставте адресу: http://<ВАША_ІР_АДРЕСА> .
- Ви маєте побачити сторінку "Welcome to the Cloud Computing Development Module!".

(Примітка: Якщо сайт не відкривається одразу, зажекайте 1-2 хвилини, поки скрипт template.tpl завершить встановлення Apache).

Налагодження веб-сервера (Troubleshooting)

Якщо після розгортання ви відкрили IP-адресу в браузері, але отримали помилку з'єднання (наприклад, `ERR_CONNECTION_REFUSED`), це означає, що скрипт автоматичного встановлення (`template.tpl`) не спрацював або ще не завершився. У цьому випадку налаштуйте сервер вручну.

1. Підключіться до віртуальної машини через SSH:

У терміналі VS Code виконайте команду (замініть `<IP_АДРЕСА>` на вашу):

```
ssh -i cloud_key azureuser@<IP_АДРЕСА>
```

(Якщо запитає підтвердження *Are you sure... , введіть yes*).

2. Встановіть та запустіть веб-сервер вручну:

Скопіюйте та вставте наступні команди у термінал SSH (по черзі або блоком):

```
# Оновлення списків та встановлення Apache
sudo apt-get update
sudo apt-get install -y apache2 php libapache2-mod-php php-mysql

# Запуск служби
sudo systemctl enable apache2
sudo systemctl start apache2
```

3. Налаштуйте контент сайту:

Очистіть папку та створіть просту сторінку для перевірки:

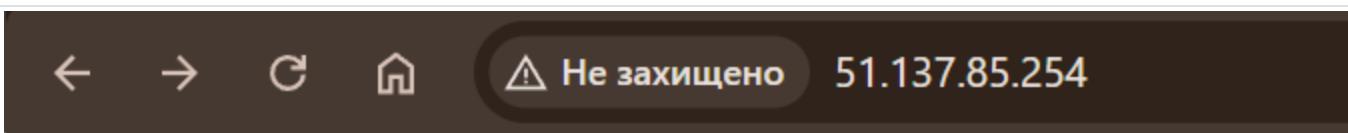
```
# Очищення папки
cd /var/www/html
sudo rm -rf *

# Створення тестової сторінки
echo '<html><head><title>Apache Default</title></head><body><h1>Apache is
running correctly!</h1><p>Manual setup successful.</p></body></html>' | 
sudo tee index.html

# Перезапуск Apache
sudo systemctl restart apache2
```

4. Перевірка:

Знову відкрийте вашу IP-адресу у браузері. Ви маєте побачити сторінку з текстом **"Apache is running correctly!"**.



Apache is running correctly!

This is a default page.

Mісце для скріншота: Відкритий веб-сайт

Крок 6: Видалення ресурсів

ВАЖЛИВО: Після завершення роботи обов'язково видаліть ресурси, щоб не витрачати кошти. Terraform робить це однією командою.

0. Для виходу з `ssh` введіть у терміналі:

```
exit
```

1. У терміналі введіть:

```
terraform destroy
```

2. Підтвердіть дію, ввівши `yes`.
3. Terraform автоматично видалить усі ресурси, які він створив (VM, IP, мережі тощо).

```
azurerm_public_ip.pip: Destroying... [id=/subscrip
azurerm_subnet.subnet: Still destroying... [id=/su
azurerm_public_ip.pip: Still destroying... [id=/su
azurerm_public_ip.pip: Destruction complete after
azurerm_subnet.subnet: Destruction complete after
azurerm_virtual_network.vnet: Destroying... [id=/s
azurerm_virtual_network.vnet: Still destroying...
azurerm_virtual_network.vnet: Destruction complete
azurerm_resource_group.rg: Destroying... [id=/subs
azurerm_resource_group.rg: Still destroying... [id=
azurerm_resource_group.rg: Destruction complete af
```

Destroy complete! Resources: 8 destroyed.

Місце для скріншота: Підтвердження видалення групи ресурсів