

### REPASO

1. DevOps es...

- a) una herramienta
- b) una práctica
- c) una filosofía
- d) una combinación de herramientas, prácticas y filosofía

2. ¿De qué herramientas se vale un DevOps?

- a) Gestión del código
- b) Orquestadores
- c) Virtualizadores
- d) Gestores de recursos

3. La Infraestructura como código nos permite...

- a) Gestionar la configuración de la infraestructura.
- b) Implementar servidores virtuales más fácilmente.
- c) Publicar una aplicación de manera automática.

4. ¿Cuales son los lenguajes de marcado más usados para los archivos de IaC?

- a) TXT
- b) JSON
- c) XML
- d) YAML

5. Ansible es una herramienta IaC que funciona...

- a) solo en AWS
- b) solo en Infraestructura on premise
- c) solo en máquinas virtuales
- d) todas las anteriores

6. Si utilizo AWS y Azure, ¿que herramienta me recomendás?

- a) CloudFormation
- b) Ansible
- c) Terraform

7. Explicá el principio de idempotencia. Ejemplificá.

El principio de idempotencia define la propiedad de usar una automatización x cantidad de veces, y sin importar las veces lo hagamos, el resultado siempre será el mismo. Un ejemplo podría ser ejecutar con terraform alguna configuración declarada para instanciar recursos, no importa cuantas veces lo ejecutemos, siempre va a generar siempre lo mismo.

8. Describí paso a paso cómo funciona Ansible.

Para empezar a usar Ansible hay una serie de pasos que debemos seguir:

1. Instalar (si es que no lo hemos hecho aún) python3, boto3, ansible y el cli de aws
2. Entrar desde el navegador a la consola de aws y crear un par de claves. Esto es para asociar a las instancias y poder conectarnos una vez ejecutada la IaC
3. Descargar el archivo .pem correspondiente
4. Crear un archivo con extensión .yaml o .json que será el playbook a ejecutar a. Declaramos el host, que sería dónde vamos a ejecutar el playbook. En nuestro caso usamos localhost, pero podría ser cualquier otro

## Recopilación Preguntas INFRAESTRUCTURA II

y ejecutarse remotamente b. Declaramos las tareas que vamos a ejecutar. En este paso cada tarea va a tener un nombre (representativo), un módulo (que podría ser aws\_s3 o ec2) y luego los parámetros necesarios para configurar los requerimientos. c. Declaramos un grupo de seguridad para poder acceder a través de SSH utilizando las claves generadas en el punto 2 d. Declaramos la instancia del recurso que vamos a crear y sus propiedades

5. En nuestro caso al utilizar cuentas educate deberíamos ir a la consola de AWS para obtener nuestras credenciales

6. Una vez obtenidas, las configuramos ya sea creando el archivo .aws/credentials manualmente, o a través del comando aws configure. Luego debemos agregarlas al PATH, cosa que en linux se hace con el comando export.

7. Ejecutamos nuestro playbook con el comando ansible-playbook <ruta al archivo>.yaml (o .json si es que utilizamos ese formato)

### 9. Describí paso a paso cómo funciona Terraform

Para automatizar los despliegues y cambios Terraform tiene dos entradas en su arquitectura. Una es la configuración concreta que hayamos escrito en un archivo tf, la otra es el estado que almacena Terraform con la infraestructura ya desplegada. Con eso es que puede crear el plan comparando el estado actual con la configuración.

1. Instalamos (si es que aún no lo hemos hecho) terraform

2. Creamos un archivo con extensión .tf. En este archivo vamos a escribir de manera declarativa utilizando HCL lo que queremos configurar. a. Indicamos el proveedor que vamos a usar (nosotros venimos utilizando AWS) y declaramos las configuraciones que queramos hacerle. Algunas básicas podrían ser indicar la ubicación de nuestras credenciales o la región que vamos a usar b. Declaramos los módulos con sus respectivas configuraciones

3. En nuestro caso al utilizar cuentas de aws educate deberíamos ir a la consola de AWS para obtener nuestras credenciales. Una vez obtenidas, las configuramos: esto lo podemos hacer creando el archivo .aws/credentials manualmente, o a través del comando aws configure. Si en el item 2a pusimos otra ubicación para las credenciales, debemos asegurarnos de crear ahí el archivo. Luego debemos agregarlas al PATH, cosa que en linux se hace con el comando export.

4. Con el archivo y las credenciales listas, nos ubicamos en la carpeta donde esté nuestro archivo .tf y ejecutamos el comando terraform init. Este comando va a descargarnos todos los módulos que hayamos indicado en nuestro archivo

5. Ejecutamos el comando terraform plan para ver el plan de acción de terraform

6. Si el plan es correcto, hacemos los cambios efectivos con terraform apply

## PARCIAL

1. Indicá etapas correctas del ciclo de vida de DevOps

Pruebas

Lanzamiento

Debug

Desplazamiento

2. ¿Qué perfiles encontramos en ecosistema DevOps?

Analista de hardware

Desarrolladores de aplicaciones

Analistas de CI/CD

Experto en telecomunicaciones

## Recopilación Preguntas INFRAESTRUCTURA II

3. La infraestructura como código nos propone...  
**Gestionar la configuración de la infraestructura**  
Implementar servidores virtuales más fácilmente  
Publicar una aplicación de manera automática
4. ¿Cuáles son los beneficios de la IaC?  
Reducir el error de los sistemas  
**Reducción de costos**  
**Estandarización de la configuración**  
Reducción de los tests
5. Ansible es una herramienta IaC que funciona...  
Solo en AWS  
Solo en infraestructura on premise  
Solo en máquinas virtuales  
**Todas las anteriores**
6. Si utilizo AWS ¿qué herramienta me recomendas?  
CloudFormation  
Ansible  
**Terraform**
7. El principio de idempotencia se define como  
Posibilidad de hacer despliegues continuos, obteniendo cambios permanentes en la infraestructura.  
Posibilidad de reducción de costos, manteniendo la infraestructura al mínimo.  
**Automatización "n" cantidad de veces obteniendo siempre y en todos los casos el mismo resultado.**
8. Ansible nació como una herramienta de Configuration Management y luego se extendió a la IaC  
**Verdadero**  
Falso
9. ¿Qué características define a los módulos de Terraform?  
Son estáticos  
**Son reutilizables**  
Son inmutables
10. Terraform y Ansible destruyen la infraestructura utilizando el mismo código con la que se creó  
Sí, ambas herramientas lo hacen  
**Solo lo hace Terraform**  
Solo lo hace Ansible
11. Describí paso a paso cómo funciona Terraform.
12. Describí algunas características del tipo de analista que usa Ansible.
13. Describí con tus palabras las ventajas de usar IaC frente al enfoque tradicional de la creación / administración de infraestructura.

## Recopilación Preguntas INFRAESTRUCTURA II

**14.** Asumiendo que tengo una llave .pen denominada “millave”, una vpc Id:vpc-7a117c07, una subnet Id:subnet-82bceedd, quiero crear una instancia Linux y poder conectarme a la misma vía SSH, revisamos y corregimos el siguiente playbook. Debes marcar errores/faltantes y proponer la solución al mismo.

### Código del Tema 1:

```
---
- hosts: localhost
  connection: local
  tasks:
    - name: Crear grupo de seguridad con HTTPS, HTTP y SSH
      ec2_group:
        name: sg_profe
        vpc_id: vpc-7a117c07
        description: sg con las reglas
        region: us-west-1
        rules:
          - proto: tcp
            ports:
              - 443
              - 80
              - 23
              - 8080
            cidr_ip: 0.0.0.0/0
            rule_desc: Acepto todo el trafico
    - name: Creamos nuestro servidor
      ec2:
        region: us-east-1
        instance_type: t2.micro
        image: ami-0c2b8caldad447f8a
        instance_tags:
          Name: Instancia_Profe
        wait: yes
        wait_timeout: 500
        group: grupo_creado
        volumes:
          - device_name: /dev/xvda
            volume_type: gp2
            volume_size: 8
        vpc_subnet_id: subnet-82bceedd
        assign_public_ip: no
        key_name: millave
        register: info
    - name: DNS Publico de nuestro servidor
      debug:
        msg: "La ip publica es {{ info.instances[0].public_ip }} y su DNS es {{
info.instances[0].public_dns_name }}"
```

### Código del Tema 2:

```
---
- hosts: localhost
  connection: local
  tasks:
    - name: Crear grupo de seguridad con HTTPS, HTTP y SSH
      ec2_group:
        name: sg_profe
        vpc_id: vpc-7a117c07
        description: sg con las reglas
        region: us-east-1
        rules:
          - proto: tcp
            ports:
              - 443
              - 80
              - 22
```

## Recopilación Preguntas INFRAESTRUCTURA II

```
- 8080
  cidr_ip: 0.0.0.0/0
  rule_desc: Acepto todo el trafico
- name: Creamos nuestro servidor
  ec2:
    region: us-east-1
    instance_type: t2.micro
    image: ami-0c2b8ca1dad447f8a
    instance_tags:
      Name: Instancia_Profe
    wait: yes
    wait_timeout: 500
    group: grupo_creado
    volumes:
      - device_name: /dev/xvda
        volume_type: gp2
        volume_size: 8
    vpc_subnet_id: subnet-82bceedd
    assign_public_ip: no

  register: info
- name: DNS Publico de nuestro servidor
  debug:
    msg: "La ip publica es {{ info.instances[0].public_ip }} y su DNS es {{
info.instances[0].public_dns_name }}"
```

### PREGUNTAS DE KAHOOT

1. El eje principal de la materia es infraestructura como código: **FALSO**

Durante la cursada vamos a estar trabajando a partir de tres grandes ejes que organizan la materia:  
Infraestructura como código, pipelines de CI/CD y Monitoreo

2. El enfoque de infraestructura como código combina las prácticas de:

Desarrollo y análisis funcional.

**Desarrollo e infraestructura.**

Infraestructura y análisis de monitoreo.

Ningún opción es correcta

3. La implementación de pipelines se lo conoce como "CI/CD". **VERDADERO**

Pipeline: tubería. Es un flujo que tiene un principio y un fin.

Integración continua (CI) y despliegue continuo (CD).

4. Docker es una herramienta que nos permite:

**Construir nuestras aplicaciones en un entorno portable.** Este es el verdadero objetivo de Docker

Eliminar la cantidad de errores al compilar.

Compilar nuestras aplicaciones más rápido.

Testear nuestras aplicaciones.

Docker permite compilar las aplicaciones más rápido y testearlas pero no es su objetivo principal.

## Recopilación Preguntas INFRAESTRUCTURA II

1. ¿DevOps es un nuevo rol de Infraestructura? **FALSO**, DevOps es un movimiento cultural.
2. Dos aspectos fundamentales de la cultura DevOps son:
  - Coordinación de Proyectos. Esto está a cargo de las metodologías ágiles.
  - **Monitoreo**. No monitoreo si el servidor está funcionando sino monitoreo el time to market, cuánto tarda el proceso de testing, cuánto tarda el proceso de despliegue. Más que monitoreo es el proceso de medir, monitoreo nuestro proceso.  
El SRE es el que se encarga del monitoreo de aplicativos y disponibilidad de los servicios; para Google el SRE es un DevOps al cuadrado.
  - **Automatización**. Esto está exclusivamente a cargo del DevOps.
  - Desarrollo. Uso herramientas de desarrollo para automatizar, pero no hago desarrollo de aplicativos.
3. El rol de Site Reliability Engineer (SRE) consiste en:
  - **Diseñar y monitorear los sistemas**. Cuando decimos sistemas nos referimos a un load balancer, por ejemplo.
  - Coordinar los sprints del proyecto.
  - Implementar nuevas versiones manualmente.
  - Ninguna opción es correcta.

### 1. AWS recomienda que los templates se versionen en...

GitHub

**S3**

En nuestra PC/Nube personal

Ninguna es correcta

Lo puedo agregar donde quiero pero AWS entiende S3 y lo integra de forma nativa.

### 2. ¿Cuál es el orden de ejecución de un template?

1. Plantilla
2. Pila
3. Cambios
4. Crear recurso

### 3. CloudFormation no forma parte de las certificaciones oficiales de AWS. **FALSO**

### 4. ¿Qué extensión de archivo es válida para un template?

JSON

YAML

TXT

**Todas las anteriores.**

**Ojo! La extensión solo me da una pauta de lo que tiene el archivo. Lo importante es el FORMATO, la estructura interna, que en caso de CloudFormation permite JSON y YAML. El formato más potente y legible es YAML.**

### 1. ¿Qué es ANSIBLE?

Una herramienta utilizada comúnmente para implementar DevOps.

Una herramienta para aprovisionar servicios en la nube.

Una herramienta para aplicar configuraciones a los servidores.

**Todas las anteriores.**

## Recopilación Preguntas INFRAESTRUCTURA II

2. Ansible es una herramienta open source. **Verdadero.**
3. ¿Cuál es el formato más común para escribir nuestro código con el objetivo de ser ejecutado por Ansible?  
.json  
**.yaml**  
.jar  
.py
4. ¿Cómo ejecuto un playbook?  
ansible playbook mi\_ec2.yml  
ansible run playbook mi\_ec2.yml  
ansible deploy playbook mi\_ec2.yml  
**ansible-playbook mi\_ec2.yml**
5. ¿Qué es un inventario?  
Una lista de controladores que Ansible dispone para hacer los despliegues.  
Una lista de recursos en archivos txt listos para utilizar.  
Una lista de los proveedores cloud donde puedo desplegar mis aplicaciones.  
**Una lista de nodos a la que Ansible puede acceder.**
6. ¿Cuáles son los casos de uso de Ansible?  
Orquestación  
Gestionar servidores  
Aprovisionamiento  
**Todas las anteriores**

### PREGUNTAS DE PG

**1** ¿Qué estrategia poseemos para reutilizar nuestro propio código?

☐ Funciones.

☒ Roles.

✓ Esta opción es correcta: los roles nos permiten reutilizar la lógica para un problema similar al que nos enfrentamos.

☐ Métodos.

## Recopilación Preguntas INFRAESTRUCTURA II

2

¿Qué tipo de agente es necesario instalar en los servidores de destino para administrar nuestra infraestructura?

☐ Ansible Agent.

☐ Red Hat Ansible Client.

☒ No es necesario instalar un agente.

✓ ¡Es la opción correcta! No es necesario instalar ningún tipo de agente para utilizar Ansible.

3

¿Qué protocolos utiliza Ansible para conectarse en servidores Linux y Windows?

☐ HTTPS / WinHTTP.

☐ TCP / UDP.

☒ SSH / WINRM.

✓ ¡Correcto! Ansible utiliza el protocolo SSH para conectarse a servidores Linux y WinRM para servidores Windows. Antes de usarlos no olvides configurar tus credenciales.



## Recopilación Preguntas INFRAESTRUCTURA II

1

Si quiero ejecutar un template de Terraform lo hago con el comando...

☐ terraform create.

☐ terraform -c.

☒ terraform apply.

✓ ¡Es la opción correcta!  
Siempre que tengas dudas  
podés ejecutar "terraform -  
help" para ver la lista de  
opciones.

2

¿Cuál es la palabra reservada para hacer referencia a un proveedor de infraestructura dentro  
nuestros templates?

☒ Provider.

✓ ¡Correcto! En nuestros  
templates indicamos qué  
tipo de infraestructura  
vamos a utilizar con esta  
palabra reservada.

☐ Aws.

☐ Workspace.

## Recopilación Preguntas INFRAESTRUCTURA II

### 3 ¿Por qué decimos que Terraform es código declarativo?

☐ Por ser multi-cloud.

☒ Por el uso de módulos.

✓ ¡Correcto! Cuando utilizamos código haciendo foco en el resultado y utilizamos una gran cantidad de módulos estamos hablando de código declarativo.

☐ Por la extensión de los archivos de código.

Vamos ahora a realizar una práctica distinta. En este ejemplo tenemos un template con errores de sintaxis. ¿Los podemos identificar? De forma individual deben realizarlo en una copia del archivo para comparar el antes y después de sus modificaciones. ¿Qué recursos se están creando?

#### Template con errores – CloudFormation

```
# Guardar el archivo como parameters.yml
Parameters:
  # Parametros para el Security Group
  SGDescription:
    Description: Security Group Description
    Type: Int
  SGPort:
    Description: Simple Description with MinValue and MaxValue
    Type: Number
    MinValue: 22
    MaxValue: 65535
  SGIngressCIDR:
    Description: The IP address range can be used to communicate to the EC2
instances
    Type: Int
    MinLength: '9'
    MaxLength: '18'
    Default: 0.0.0.0/0
    AllowedPattern: (\d{1,3})\.\(\d{1,3})\.\(\d{1,3})\.\(\d{1,3})/(\d{1,2})
    ConstraintDescription: Use a valid IP CIDR range with format x.x.x.x/x.
  # Parametros para la instancia ec2
  InstanceType:
    Description: EC2 instance type
    Type: Int
```

## Recopilación Preguntas INFRAESTRUCTURA II

Default: t2.small.

AllowedValues:

- t1.micro
- t2.nano
- t2.micro
- t2.small
- m1.small

ConstraintDescription: Use a valid EC2 instance type.

KeyPairName:

Description: EC2 KeyPair to enable SSH access to the instances.

Type: AWS::EC2::KeyPair::KeyName

ConstraintDescription: Use the name of an existing EC2 KeyPair

# Parametros para redes

VPC:

Description: VPC to operate in

Type: AWS::EC2::VPC::Id

SubnetIDs:

Description: Subnet IDs that is a List of Subnet Id

Type: "List<AWS::EC2::Subnet::Id>"

DbSubnetIpBlocks:

Description: "Comma delimited list of CIDR blocks"

Type: CommaDelimitedList

Default: "10.0.48.0/24, 10.0.112.0/24, 10.0.176.0/24"

# Password para un db

DBPassword:

NoEcho: true

Description: Account password for the database

Type: Int

Resources:

# Creacion de una instancia EC2

MyInstance:

Type: "AWS::EC2::Instance"

Properties:

InstanceType: !Ref InstanceType

KeyName: !Ref KeyPairName

ImageId: "ami-a4c7edb2"

# Creacion de un security group

MySecurityGroup:

Type: "AWS::EC2::SecurityGroup"

Properties:

GroupDescription: !Ref SGDescription

SecurityGroupIngress:

- CidrIp: !Ref SGIngressCIDR

## Recopilación Preguntas INFRAESTRUCTURA II

```
FromPort: !Ref SGPort
ToPort: !Ref SGPort
IpProtocol: tcp
VpcId: !Ref VPC
```

### Template con errores – Ansible

```
---
- name: test raw module
  hosts: all host
  tasks:
    - name: run ipconfig

      raw: ipconfig
      register: ipconfig
    - debug: var=ipconfig
- name: test stat module
  hosts: windows
  tasks:
    - names: test stat module on file
      win_stat: path="C:/Windows/win.ini"
      register: stat_file
    - debugging: var=stat_file
    - names: check stat_file result
      assert:
        that:
          * "stat_file.stat.exists"
          * "not stat_file.stat.isdir"
          * "stat_file.stat.size > 0"
          * "stat_file.stat.md5"
```

### Template con errores – Terraform

```
provider cloud "aws" {
  alias : "us_east_1"
  region : "us-east-1"
}
module mi sitio "mi_sitio" {
- source =
"https://raw.githubusercontent.com/cloudacademy/static-website-example/master/index.html"
- dominio= "hello.example.com"
}
resource "aws_s3_bucket_object" "my_index" {
  bucket = "${module.mi_sitio.bucket_name}"
  key = "index.html"
  content = "<pre>Hello World!</pre>"
  content_type = "text/html; charset=utf-8"
}
output "bucket_name" {
  description = "El nombre del bucket S3 es usado para alojar el contenido."
  value = "${module.mi_sitio.bucket_name}"
}
```