Parcial Infraestructura II - TEMA 2

dayana.otagri0902@gmail.com (no compartidos)
Cambiar de cuenta

Preguntas a desarrollar

Describí paso a paso cómo funciona Terraform.

1. Se instala Terraform (en caso de no haberlo hecho)
2. Se crea un archivo con extensión .tf (extensión propia de Terraform). Este archivo contiene la configuración, esta escrito de forma declarativa, es decir, sobre la lógica que se va a ejecutar sin indicar los detalles de cómo se va desarrollar.
- Se le indica el provider, proveedor que se va a utilizar
- shared_credentials_fire: “donde estan las credenciales de nuestra cuenta” region : más cercana a utilizar.
- Módulos con sus configuraciones
3. El proveedor en este caso es AWS, aquí se encuentran dos alternativas para declarar las credenciales de acceso: 1. crear un archivo .aws/credentials o 2. Por medio del comando aws configure
4. Dentro de la carpeta donde se encuentra el .ft ejecutamos el comando: terraform init que trabaja sobre el provider y los módulos
5. Una vez finalizada la ejecución del comando anterior se realiza el siguiente comando: terraform plan, que muestra todo lo que se va a crear
6. Terminado el paso anteriror ejecutamos terraform apply para crear los recursos, este nos pide una confirmación llegado a cierto punto de la ejecución, en ese momento escribimos yes y enter
7. Finalmente Terraform destroy para destruir los recursos

Describi algunas caracteristicas del tipo de Analista que usa Ansible

El tipo de analista que utiliza Ansible es aquel que:

1. Le gusta el software open source, es decir código abierto.
2. Primero gestiona servidores y luego gestionar la infraestructura
3. Le gusta pensar de forma descriptiva lo que le permite una buena configuración de los archivos
4. Tiene un buen conocimiento de yaml, ya que, Ansible utiliza este como lenguaje para los playbooks

Describi con tus palabras las ventajas de usar IaC frente al enfoque tradicional de la creacion / administracion de Infraestructura

1. Reducción del error humano: Porque permite seguir una serie de pasos o procedimientos bien definidos y ordenados evitando una mala configuración. Esto permite que aumente la confianza en la infraestructura.
2. Repetibilidad y predictibilidad: Al saber el contexto en el que funciona una aplicación, es posible repetir los pasos que sean necesarios y tener la posibilidad de saber cuáles serán los resultados, en este sentido, esto nos permite que la infraestructura sea más testeable y estable.
3. Tiempos y reducción de desperdicios: Gracias a los scripts de código el tiempo de ejecución de la infraestructura será reducido y tampoco se necesitaran componentes de más.
4. Control de versiones: Debido a que la infraestructura es definida en archivos estos pueden ser versionados en templates, que nos permiten utilizar parámetros para tener un código más genérico.
5. Reducción de costos: La automatización permite hacer énfasis en otras tareas lo que aporta una flexibilización en los equipos encargados de la infraestructura permitiendo desarrollar más tareas.
6. Testeos: La infraestructura como código permite realizar pruebas de las aplicaciones en cualquier entorno desde el inicio del ciclo desarrollo.
7. Entornos estables y escalables: Dejar de lado las configuraciones manuales y reducir las dependencias, permite generar entornos más estables y escalables.
8. Estandarización de la configuración: Evita problemas de incompatibilidad con la infraestructura y hace que las aplicaciones se ejecuten con mejor rendimiento.
9. Documentación: Documentar cada cambio y registrado por el usuario que lo desarrolla y permite tener rollback en caso de encontrar errores en los despliegues y necesitar versiones anteriores.
10. Rapidez y seguridad: Nos permite estandarizar los grupos de seguridad con los permisos mínimos aunque necesarios.