



Google Cloud

Automatización de DevOps

Priyanka Vergadia

Developer Advocate, Google Cloud

En este módulo, se presenta la automatización de DevOps, que es un factor clave para asegurar la coherencia, la confiabilidad y la velocidad en las implementaciones.

Objetivos de aprendizaje

- Automatizar la implementación de servicios con canalizaciones de CI/CD
- Usar Cloud Source Repositories para el control de versión y de código fuente
- Automatizar compilaciones con Cloud Build y activadores de compilación
- Administrar imágenes de contenedor con Container Registry
- Investigar la infraestructura con código mediante Terraform

Abordaremos específicamente los servicios que son compatibles con las prácticas de integración y entrega continuas, que forman parte del marco de trabajo de las DevOps.

Para emplear las DevOps y los microservicios, se necesitan canalizaciones automatizadas para integrar, entregar y, potencialmente, implementar código. Lo ideal es que estas canalizaciones se ejecuten en recursos aprovisionados a pedido. En este módulo, se presentan las herramientas de Google Cloud para desarrollar código y crear canalizaciones de entrega automatizadas que se aprovisionan a pedido.

Hablaremos sobre cómo usar Cloud Source Repositories para controlar versiones y código fuente, Cloud Build (incluidos los activadores de compilación) para automatizar compilaciones y Container Registry para administrar contenedores.

Por último, revisaremos herramientas de infraestructura como código, como Terraform.

Objetivos de aprendizaje

- Automatizar la implementación de servicios con canalizaciones de CI/CD
- Usar Cloud Source Repositories para el control de versión y de código fuente
- Automatizar compilaciones con Cloud Build y activadores de compilación
- Administrar imágenes de contenedor con Container Registry
- Investigar la infraestructura con código mediante Terraform

Abordaremos específicamente los servicios que son compatibles con las prácticas de integración y entrega continuas, que forman parte del marco de trabajo de las DevOps.

Para emplear las DevOps y los microservicios, se necesitan canalizaciones automatizadas para integrar, entregar y, potencialmente, implementar código. Lo ideal es que estas canalizaciones se ejecuten en recursos aprovisionados a pedido. En este módulo, se presentan las herramientas de Google Cloud para desarrollar código y crear canalizaciones de entrega automatizadas que se aprovisionan a pedido.

Objetivos de aprendizaje

- Automatizar la implementación de servicios con canalizaciones de CI/CD
- Usar Cloud Source Repositories para el control de versión y de código fuente
- Automatizar compilaciones con Cloud Build y activadores de compilación
- Administrar imágenes de contenedor con Container Registry
- Investigar la infraestructura con código mediante Terraform

Hablaremos sobre cómo usar Cloud Source Repositories para las fuentes y el control de versión...

Objetivos de aprendizaje

- Automatizar la implementación de servicios con canalizaciones de CI/CD
- Usar Cloud Source Repositories para el control de versión y de código fuente
- Automatizar compilaciones con Cloud Build y activadores de compilación
- Administrar imágenes de contenedor con Container Registry
- Investigar la infraestructura con código mediante Terraform

... Cloud Build, incluidos los activadores de compilación para automatizar compilaciones...

Objetivos de aprendizaje

- Automatizar la implementación de servicios con canalizaciones de CI/CD
- Usar Cloud Source Repositories para el control de versión y de código fuente
- Automatizar compilaciones con Cloud Build y activadores de compilación
- Administrar imágenes de contenedor con Container Registry
- Investigar la infraestructura con código mediante Terraform

... y cómo administrar contenedores con Container Registry.

Objetivos de aprendizaje

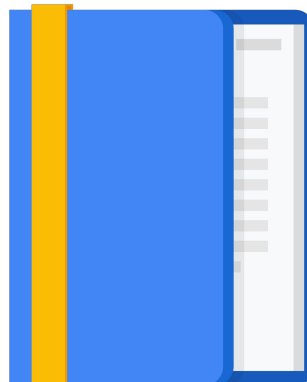
- Automatizar la implementación de servicios con canalizaciones de CI/CD
- Usar Cloud Source Repositories para el control de versión y de código fuente
- Automatizar compilaciones con Cloud Build y activadores de compilación
- Administrar imágenes de contenedor con Container Registry
- Investigar la infraestructura con código mediante Terraform

Por último, revisaremos herramientas de infraestructura como código, **como** **Deployment Manager y** Terraform.

Temario

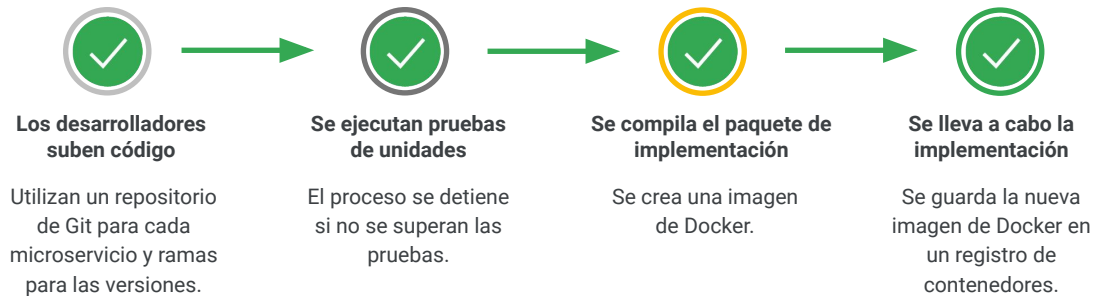
Canalizaciones de integración continua

Infraestructura como código



Comencemos por hablar de las canalizaciones de integración continua.

Las canalizaciones de integración continua automatizan la compilación de aplicaciones



Las canalizaciones de integración continua automatizan la compilación de aplicaciones. En este gráfico, se muestra una vista simplificada de una canalización, que se debería personalizar para que cumpla con sus requisitos. Este proceso comienza cuando el código se sube al repositorio en el que se ejecutan todas las pruebas de unidades. Si se superan todas las pruebas, se compila un paquete de implementación como una imagen de Docker. Luego, la imagen se guarda en un registro de contenedores a partir del cual se puede implementar. Cada microservicio debe tener su propio repositorio.

Entre los pasos adicionales típicos, se incluyen el análisis del código con lint, el análisis de calidad con herramientas como SonarQube, las pruebas de integración, la generación de informes de pruebas y el análisis de imágenes.

Google proporciona los componentes necesarios para una canalización de integración continua

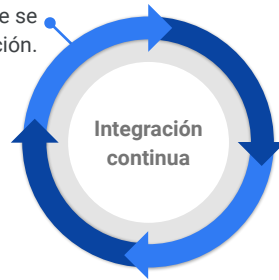


Google Cloud proporciona los componentes necesarios para crear una canalización de integración continua. Veamos cuáles son.

Google proporciona los componentes necesarios para una canalización de integración continua

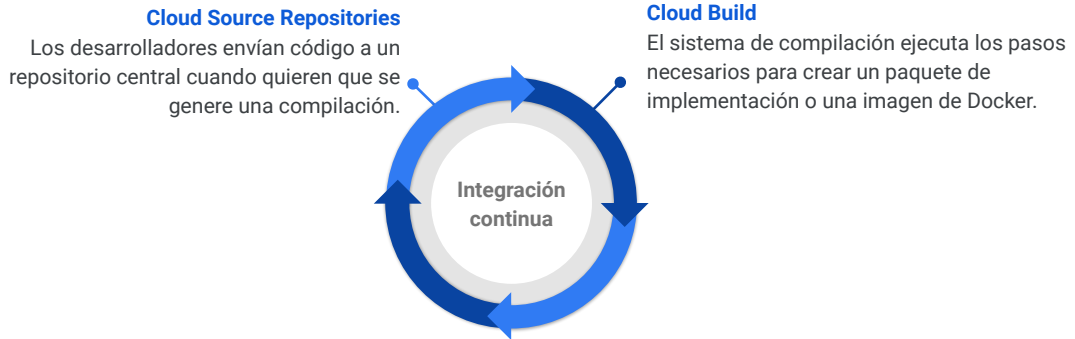
Cloud Source Repositories

Los desarrolladores envían código a un repositorio central cuando quieren que se genere una compilación.



El servicio de **Cloud Source Repositories** proporciona repositorios de Git privados alojados en Google Cloud. Estos permiten desarrollar e implementar una app o un servicio en un espacio que ofrece funciones de colaboración y control de versiones para su código. Cloud Source Repositories se integra en Google Cloud, así que proporciona una experiencia fluida a los desarrolladores.

Google proporciona los componentes necesarios para una canalización de integración continua



Cloud Build ejecuta sus compilaciones en la infraestructura de Google Cloud. Puede importar código fuente de Cloud Storage, Cloud Source Repositories, GitHub o Bitbucket; ejecutar una compilación según sus especificaciones, y producir artefactos como contenedores de Docker o archivos de Java. Además, ejecuta su compilación como una serie de pasos que se realizan en un contenedor de Docker. Un paso de la compilación puede hacer todo lo que se puede realizar en un contenedor, independientemente del entorno. Existen pasos estándares, pero también puede definir los que desee.

Google proporciona los componentes necesarios para una canalización de integración continua



Un **activador de Cloud Build** inicia una compilación de forma automática cada vez que realiza algún cambio en el código fuente. Puede configurar el activador para que compile el código a partir de cualquier cambio que se realice en el repositorio de código fuente o solo a partir de los cambios que coincidan con ciertos criterios.

Google proporciona los componentes necesarios para una canalización de integración continua

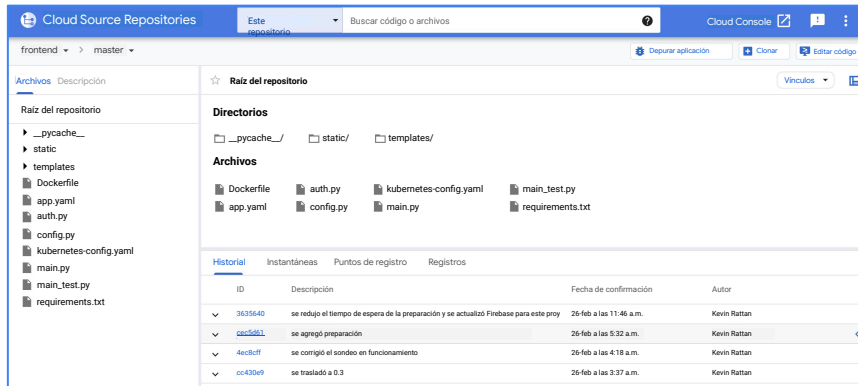


Con Container Registry, su equipo puede administrar las imágenes de Docker o paquetes de implementación, llevar a cabo análisis de vulnerabilidades y decidir quién accede a qué recursos con un control de acceso preciso, todo en un solo lugar.

Revisemos cada uno de estos servicios con más detalle.

Cloud Source Repositories proporciona repositorios de Git administrados

Controle el acceso a los repositorios usando IAM en sus proyectos de Google Cloud.



Cloud Source Repositories proporciona repositorios de Git administrados.

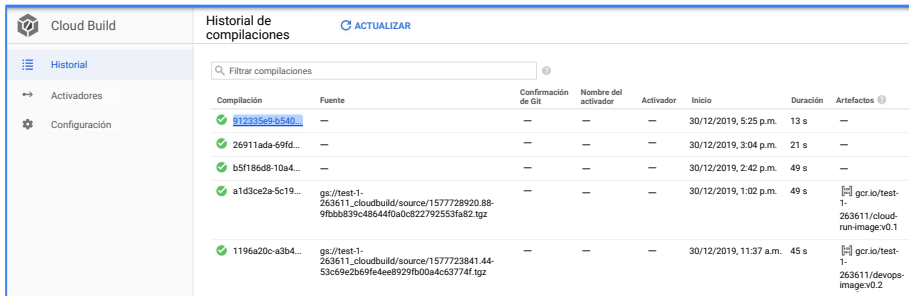
Puede usar IAM para agregar miembros del equipo a su proyecto y otorgarles permisos a fin de que puedan crear, ver y actualizar repositorios.



Los repositorios se pueden configurar para publicar mensajes en un tema de Pub/Sub específico. Los mensajes se pueden publicar cuando un usuario crea o borra un repositorio, o envía una confirmación.

Otras funciones de Cloud Source Repositories incluyen la capacidad de depurar en la etapa de producción con Cloud Debugger; generar registros de auditoría para proporcionar información sobre las acciones y el momento y lugar en que se realizaron, y realizar implementaciones directas en App Engine. También se puede conectar un repositorio existente de GitHub o Bitbucket a Cloud Source Repositories. Los repositorios conectados se sincronizan automáticamente con esta plataforma.

Cloud Build le permite compilar software con rapidez en todos los lenguajes

- Servicio de compilación de Docker alojado por Google
 - Es una alternativa al uso del comando de compilación de Docker.
- Se usa la CLI para enviar una compilación
`gcloud builds submit --tag gcr.io/your-project-id/image-name .`



Cloud Build								
Historial de compilaciones ACTUALIZAR								
Filtrar compilaciones								
Compilación	Fuente	Confirmación de Git	Nombre del activador	Activador	Inicio	Duración	Artefactos	
912335e9-b540	—	—	—	—	30/12/2019, 5:25 p.m.	13 s	—	
26911ada-69fd...	—	—	—	—	30/12/2019, 3:04 p.m.	21 s	—	
b5f186d8-10a4...	—	—	—	—	30/12/2019, 2:42 p.m.	49 s	—	
a1d3ce2a-5c19...	gs://test-1-263611_cloudbuild/source/1577728920.88-9fbb839c48644f0a0c822792553fa82.tgz	—	—	—	30/12/2019, 1:02 p.m.	49 s	 gcr.io/test-1-263611/cloud-run-image.v0.1	
1196a20c-a3b4...	gs://test-1-263611_cloudbuild/source/157772841.44-53c9e2b69fe4ee8929fb00a4c63774f.tgz	—	—	—	30/12/2019, 11:37 a.m.	45 s	 gcr.io/test-1-263611/devops-image.v0.2	

Cloud Build permite compilar software con rapidez en todos los lenguajes. Es un servicio de compilación de Docker alojado por Google y es una alternativa al uso del comando de compilación de Docker. Además, se puede utilizar la CLI para enviar una compilación usando gcloud. Se muestra un ejemplo en esta diapositiva.

Con “gcloud build submits”, la compilación se envía y ejecuta como un elemento remoto. “--tag” es la etiqueta que se debe utilizar cuando se crea la imagen. Esta etiqueta debe usar el espacio de nombres gcr.io o *.gcr.io.*. Si la utiliza, su fuente debe contener un Dockerfile.

Por último, el “.” representa la ubicación de la fuente que se compilará.

Los activadores de compilación supervisan un repositorio y crean un contenedor cuando se envía código

Son compatibles con Maven, compilaciones personalizadas y Docker

Crear activador

1 Seleccionar origen 2 Seleccionar un repositorio

Selecciona el origen

Elige una opción de hosting para el repositorio

☒ Cloud Source Repository
☐ GitHub
☐ Bitbucket

[Continuar](#) [Cancelar](#)

Crear activador

✓ Selecciona el origen 2 Seleccionar un repositorio 3 Config

Selecciona un repositorio

Fuente: Cloud Source Repository

Filtrar repositorios

☒ default
Cloud Source Repository

[Continuar](#) [Cancelar](#)

Configuración del activador

Fuente: Cloud Source Repository Repositorio: <https://source.developers.google.com/p/rem>

Nombre (opcional)

Tipo de activador

☒ Rama
☐ Etiqueta

Rama (regex)
Coincide con la rama: principal

Configuración de compilación

☒ Dockerfile
Especifica la ruta de acceso dentro del repositorio de

☐ cloudbuild.yaml
Especifica la ruta de acceso de un archivo de configuración de
Cloud Build en el repositorio de Git [Más información](#)

Directorio de Dockerfile
Este directorio también se usará como contexto de la compilación de Docker

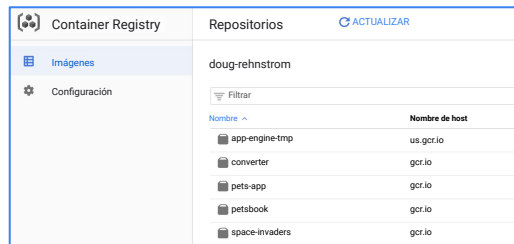
Los activadores de compilación supervisan un repositorio y crean un contenedor cuando se envía código. Son compatibles con Maven, compilaciones personalizadas y Docker.

Un activador de Cloud Build inicia una compilación de forma automática cada vez que se modifica el código fuente. Puede configurarse para iniciar una compilación basada en confirmaciones de una rama específica o en confirmaciones que contengan una etiqueta determinada. Puede especificar una expresión regular que incluya el valor de la rama o la etiqueta con el que debe coincidir.

Se puede especificar la configuración de compilación en un Dockerfile o un archivo de Cloud Build. En esta diapositiva, se muestra la configuración requerida. En primer lugar, se selecciona una fuente, que puede corresponder a Cloud Source Repositories, GitHub o Bitbucket. En la siguiente etapa, se selecciona un repositorio de código fuente y, luego, la configuración del activador. Esta última incluye información como la rama o etiqueta que se utilizará para la activación y la configuración de compilación, por ejemplo, el Dockerfile o el archivo de Cloud Build.

Container Registry es un repositorio de Docker alojado en Google Cloud

- Las imágenes compiladas con Cloud Build se guardan automáticamente en Container Registry.
 - Las imágenes se etiquetan con el prefijo `gcr.io/your-project-id/image-name`.
- Puede utilizar comandos de envío y recuperación de Docker con Container Registry.
 - `docker push gcr.io/your-project-id/image-name`
 - `docker pull gcr.io/your-project-id/image-name`



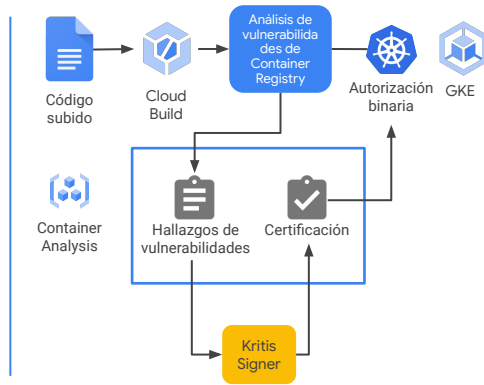
Container Registry es un repositorio de Docker alojado en Google Cloud. Las imágenes compiladas con Cloud Build se guardan automáticamente en Container Registry. Las imágenes se etiquetan con un prefijo, como se muestra en esta diapositiva.

También se pueden enviar y recuperar imágenes con los comandos estándar de Docker. Para enviar una imagen, use el siguiente comando:
`docker push gcr.io/your-project-id/image-name`

Para recuperar una imagen, use el siguiente comando:
`docker pull gcr.io/your-project-id/image-name`

La autorización binaria permite exigir que solo contenedores de confianza se implementen en GKE

- Habilite la autorización binaria en el clúster de GKE.
- Agregue una política que requiera imágenes firmadas.
- Cuando se compila una imagen con Cloud Build, un “certificador” verifica que provenga de un repositorio de confianza (por ejemplo, Source Repositories).
- Container Registry incluye análisis de vulnerabilidades para los contenedores.



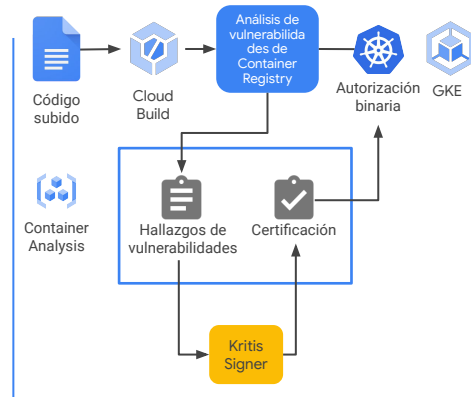
La autorización binaria permite exigir que solo contenedores de confianza se implementen en GKE. Esta función es un servicio de Google Cloud basado en la especificación de Kritis. Para que funcione, debe habilitarla en el clúster de GKE donde se realizará la implementación. Se necesita una política para firmar las imágenes. Cuando se compila una imagen con Cloud Build, un certificador verifica que provenga de un repositorio de confianza, como Source Repositories. Container Registry incluye análisis de vulnerabilidades para los contenedores. En el diagrama, se muestra un flujo de trabajo típico.

Cuando se sube código, se activa la compilación con Cloud Build. Como parte de esta, Container Registry realizará un análisis de vulnerabilidades cuando se suba la nueva imagen. La herramienta de análisis publica mensajes en Pub/Sub. Kritis Signer escucha las notificaciones de Pub/Sub provenientes del análisis de vulnerabilidades del registro de contenedores y aplica una certificación si la imagen pasa el análisis. Después, el servicio de autorización binaria de Google Cloud aplica la política que requiere certificaciones de Kritis Signer a fin de implementar la imagen de contenedor.

Con este flujo, se impide la implementación de imágenes con vulnerabilidades que estén por debajo de cierto umbral.

La autorización binaria permite exigir que solo contenedores de confianza se implementen en GKE

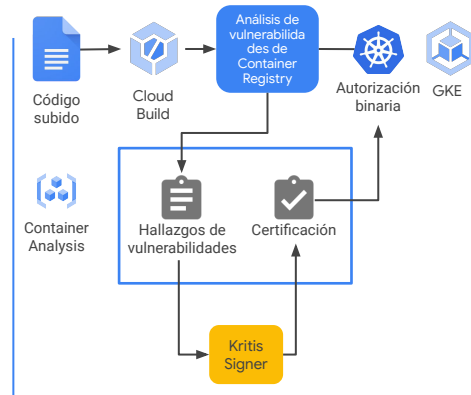
- Habilite la autorización binaria en el clúster de GKE.
- Agregue una política que requiera imágenes firmadas.
- Cuando se compila una imagen con Cloud Build, un "certificador" verifica que provenga de un repositorio de confianza (por ejemplo, Source Repositories).
- Container Registry incluye análisis de vulnerabilidades para los contenedores.



La autorización binaria permite exigir que solo contenedores de confianza se implementen en GKE. Esta función es un servicio de Google Cloud basado en la especificación de Kritis.

La autorización binaria permite exigir que solo contenedores de confianza se implementen en GKE

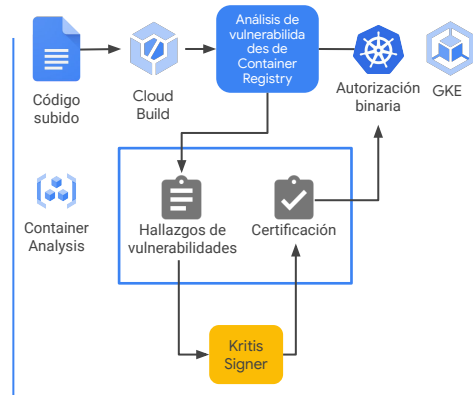
- Habilite la autorización binaria en el clúster de GKE.
- Agregue una política que requiera imágenes firmadas.
- Cuando se compila una imagen con Cloud Build, un "certificador" verifica que provenga de un repositorio de confianza (por ejemplo, Source Repositories).
- Container Registry incluye análisis de vulnerabilidades para los contenedores.



Para que funcione, debe habilitar la autorización binaria en el clúster de GKE donde se realizará la implementación.

La autorización binaria permite exigir que solo contenedores de confianza se implementen en GKE

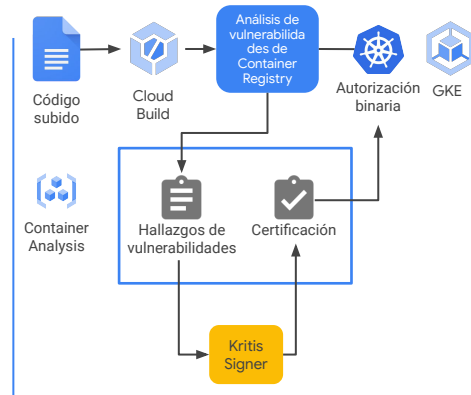
- Habilite la autorización binaria en el clúster de GKE.
- Agregue una política que requiera imágenes firmadas.
- Cuando se compila una imagen con Cloud Build, un "certificador" verifica que provenga de un repositorio de confianza (por ejemplo, Source Repositories).
- Container Registry incluye análisis de vulnerabilidades para los contenedores.



Se necesita una política para firmar las imágenes.

La autorización binaria permite exigir que solo contenedores de confianza se implementen en GKE

- Habilite la autorización binaria en el clúster de GKE.
- Agregue una política que requiera imágenes firmadas.
- Cuando se compila una imagen con Cloud Build, un "certificador" verifica que provenga de un repositorio de confianza (por ejemplo, Source Repositories).
- Container Registry incluye análisis de vulnerabilidades para los contenedores.



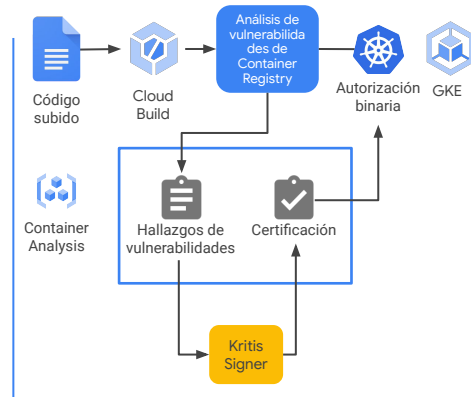
Cuando se compila una imagen con Cloud Build, un certificador verifica que provenga de un repositorio de confianza, como Source Repositories. Container Registry incluye análisis de vulnerabilidades para los contenedores. En el diagrama, se muestra un flujo de trabajo típico.

Cuando se sube código, se activa la compilación con Cloud Build. Como parte de esta, Container Registry realizará un análisis de vulnerabilidades cuando se suba la nueva imagen. La herramienta de análisis publica mensajes en Pub/Sub.

La autorización binaria permite exigir que solo contenedores de confianza se implementen en GKE

- Habilite la autorización binaria en el clúster de GKE.
- Agregue una política que requiera imágenes firmadas.
- Cuando se compila una imagen con Cloud Build, un "certificador" verifica que provenga de un repositorio de confianza (por ejemplo, Source Repositories).

● Container Registry incluye análisis de vulnerabilidades para los contenedores.



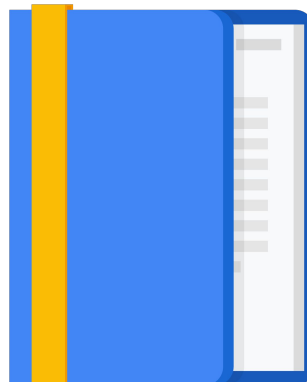
Kritis Signer escucha las notificaciones de Pub/Sub provenientes del análisis de vulnerabilidades del registro de contenedores y aplica una certificación si la imagen pasa el análisis. Después, el servicio de autorización binaria de Google Cloud aplica la política que requiere certificaciones de Kritis Signer a fin de implementar la imagen de contenedor.

Con este flujo, se impide la implementación de imágenes con vulnerabilidades que estén por debajo de cierto umbral.

Temario

Canalizaciones de integración continua

Infraestructura como código



Ahora, consideremos la infraestructura como código.

Migrar a la nube implica un cambio de mentalidad

Infraestructura local

- Se compran máquinas.
- Las máquinas se mantienen en ejecución por varios años.
- Se prefiere una pequeña cantidad de máquinas grandes.
- Las máquinas representan gastos de capital.

Computación en la nube

- Se alquilan máquinas.
- Las máquinas se desactivan en cuanto sea posible.
- Se prefiere una gran cantidad de máquinas pequeñas.
- Las máquinas representan gastos mensuales.

Google Cloud

Migrar a la nube implica un cambio de mentalidad. El modelo a pedido y de pago por uso de la computación en la nube es distinto al de aprovisionamiento de infraestructura local. Un modelo local tradicional consistiría en comprar máquinas y ejecutarlas de forma continua. La infraestructura de procesamiento suele componerse de una cantidad más pequeña de máquinas grandes. Desde el punto de vista de la contabilidad, las máquinas representan un gasto de capital que se devalúa con el tiempo.

Cuando se utiliza la nube, los recursos se alquilan en lugar de comprarse y, como resultado, procuramos desactivar las máquinas en cuanto sea posible para reducir los costos. El enfoque suele consistir en tener muchas máquinas pequeñas, escalar horizontalmente en lugar de verticalmente y operar preparándose para las fallas. Desde el punto de vista de la contabilidad, las máquinas representan un gasto operativo mensual.

Migrar a la nube implica un cambio de mentalidad

Infraestructura local

- Se compran máquinas.
- Las máquinas se mantienen en ejecución por varios años.
- Se prefiere una pequeña cantidad de máquinas grandes.
- Las máquinas representan gastos de capital.

Migrar a la nube implica un cambio de mentalidad. El modelo a pedido y de pago por uso de la computación en la nube es distinto al de aprovisionamiento de infraestructura local. Un modelo local tradicional consistiría en comprar máquinas y ejecutarlas de forma continua. La infraestructura de procesamiento suele componerse de una cantidad más pequeña de máquinas grandes. Desde el punto de vista de la contabilidad, las máquinas representan un gasto de capital que se devalúa con el tiempo.

Migrar a la nube implica un cambio de mentalidad

Infraestructura local

- Se compran máquinas.
- Las máquinas se mantienen en ejecución por varios años.
- Se prefiere una pequeña cantidad de máquinas grandes.
- Las máquinas representan gastos de capital.

Computación en la nube

- Se alquilan máquinas.
- Las máquinas se desactivan en cuanto sea posible.
- Se prefiere una gran cantidad de máquinas pequeñas.
- Las máquinas representan gastos mensuales.

Google Cloud

Cuando se utiliza la nube, los recursos se alquilan en lugar de comprarse y, como resultado, procuramos desactivar las máquinas en cuanto sea posible para reducir los costos. El enfoque suele consistir en tener muchas máquinas pequeñas, escalar horizontalmente en lugar de verticalmente y operar preparándose para las fallas. Desde el punto de vista de la contabilidad, las máquinas representan un gasto operativo mensual.

En la nube, toda la infraestructura debe ser desechable

- No se reparan las máquinas con errores.
- No se instalan parches.
- No se actualizan las máquinas.
- Si necesita reparar una máquina, bórrala y cree una nueva.
- Para que la infraestructura sea desechable, automatice todo con código:
 - Puede automatizar con secuencias de comandos.
 - Puede utilizar herramientas declarativas para definir la infraestructura.

En otras palabras, en la nube, toda la infraestructura debe ser desechable. La clave para ello es la infraestructura como código (laC), que permite automatizar las actividades de aprovisionamiento, implementación y configuración.

De esta forma, se minimizan los riesgos; se eliminan los errores manuales, y se admiten las implementaciones repetibles, el escalamiento y una operación más rápida. Implementar una o cien máquinas requiere el mismo esfuerzo. La automatización puede alcanzarse usando secuencias de comandos o herramientas declarativas, como Terraform, que analizaremos después.

Es muy importante no dedicar tiempo a reparar máquinas con errores ni a instalar parches o actualizaciones. Estas actividades producirán problemas cuando se recreen los entornos más adelante. Si una máquina necesita mantenimiento, quítela y cree una nueva.

Se pueden reducir los costos aprovisionando entornos efímeros, como los de prueba, que replican el entorno de producción.

La infraestructura como código (IaC) permite aprovisionar o quitar infraestructuras con rapidez

- Cree una infraestructura cuando sea necesario.
- Destruya la infraestructura cuando no la utilice.
- Cree infraestructuras idénticas para desarrollo, pruebas y producción.
- Puede formar parte de una canalización de CI/CD.
- Las plantillas son los componentes básicos de los procedimientos de recuperación ante desastres.
- Administre las dependencias y la complejidad de los recursos.
- Google Cloud admite varias herramientas de IaC.



Google Cloud

Terraform es una herramienta usada en infraestructura como código (IaC). Antes de analizar Terraform, veamos qué es lo que hace la infraestructura como código. En términos simples, permite aprovisionar y quitar rápidamente las infraestructuras.

El aprovisionamiento bajo demanda de una implementación es extremadamente potente. Puede integrarse en una canalización de integración continua que facilite la ruta hacia la implementación continua.

El aprovisionamiento automatizado de la infraestructura significa que la infraestructura se puede aprovisionar a pedido y que las complejidades de la implementación se administran en el código. Esto proporciona la flexibilidad necesaria para modificar la infraestructura a medida que cambien las necesidades. Además, todos los cambios ocurren en un lugar. Ahora, la infraestructura para entornos como el de desarrollo y el de pruebas puede replicar fácilmente el entorno de producción y puede borrarse de inmediato cuando no se utilice. Todo esto se debe a la infraestructura como código.

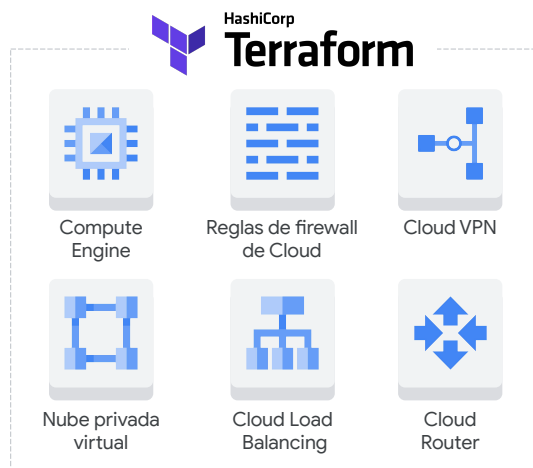
Existen varias herramientas que se pueden usar para la IaC. Google Cloud admite Terraform, en la que las implementaciones se describen en un archivo conocido como una configuración. En él, se detallan todos los recursos que se deben aprovisionar. Los parámetros de configuración pueden modularse con plantillas, lo que permite la abstracción de los recursos en componentes reutilizables entre las implementaciones.

Además de Terraform, Google Cloud admite otras herramientas de IaC, como las siguientes:

- Chef
- Puppet
- Ansible
- Packer

Terraform es una herramienta de automatización de la infraestructura

- Proceso de implementación repetible
- Lenguaje declarativo
- Enfoque en la aplicación
- Implementación en paralelo
- Basado en plantillas



Google Cloud

Terraform es una herramienta de código abierto que permite aprovisionar recursos de Google Cloud.

Terraform permite aprovisionar recursos de Google Cloud, como máquinas virtuales, contenedores, almacenamiento y redes, con archivos de configuración declarativos. Solo debe especificar todos los recursos necesarios para su aplicación en un formato declarativo y, luego, implementar la configuración. El lenguaje de configuración de HashiCorp (HCL) permite realizar descripciones concisas de los recursos mediante bloques, argumentos y expresiones.

Esta implementación puede repetirse una y otra vez con resultados coherentes y usted puede quitar una implementación completa con un solo comando o clic. La ventaja de un enfoque declarativo es que le permite especificar cuál debe ser la configuración y dejar que el sistema decida los pasos siguientes.

En vez de implementar cada recurso por separado, se especifica el conjunto de recursos que componen la aplicación o el servicio, lo que permite enfocarse en la aplicación. A diferencia de Cloud Shell, Terraform implementará recursos en paralelo.

Terraform utiliza las API subyacentes de cada servicio de Google Cloud para implementar sus recursos, lo cual permite implementar casi todos los recursos que hemos visto hasta ahora, desde instancias, plantillas de instancias y grupos hasta redes de VPC, reglas de firewall, túneles VPN, Cloud Routers y balanceadores de cargas.

[Documentación: [Usa Terraform con Google Cloud](#)]

Lenguaje de Terraform

- El lenguaje de Terraform es la interfaz que permite declarar los recursos.
- Los recursos son objetos de infraestructura.
- El archivo de configuración guía la administración de estos recursos.

```
resource "google_compute_network" "default" {  
  name = "${var.network_name}"  
  auto_create_subnetworks = false  
}  
  
<BLOCK TYPE> "<BLOCK LABEL>" "<BLOCK LABEL>" {  
  # Block body  
  <IDENTIFIER> = <EXPRESSION> # Argument  
}
```

El lenguaje de Terraform es la interfaz de usuario que se usa para declarar recursos. Estos son objetos de infraestructura, como máquinas virtuales de Compute Engine, buckets de almacenamiento, contenedores o redes. Una configuración de Terraform es un documento completo en el lenguaje de Terraform que le dice al servicio cómo administrar una determinada colección de infraestructura. Una configuración puede constar de varios archivos y directorios.

La sintaxis del lenguaje de Terraform incluye los siguientes aspectos:

- Bloques que representan objetos y pueden tener cero o más etiquetas (los bloques tienen un cuerpo que permite declarar argumentos y bloques anidados)
- Argumentos que se utilizan para asignar un valor a un nombre
- Una expresión que representa un valor que puede asignarse a un identificador

Terraform puede utilizarse en múltiples nubes públicas y privadas

- Se considera una herramienta de primera clase en Google Cloud.
- Viene instalada en Cloud Shell.

```
provider "google" {  
  region = "us-central1"  
}  
  
resource "google_compute_instance" {  
  name         = "instance name"  
  machine_type = "n1-standard-1"  
  zone         = "us-central1-f"  
  
  disk {  
    image = "image to build instance"  
  }  
}  
  
output "instance_ip" {  
  value = "${google_compute.instance_ip_address}"  
}
```

Google Cloud

Terraform puede utilizarse en múltiples nubes públicas y privadas, y viene instalada en Cloud Shell.

El ejemplo de archivo de configuración de Terraform que se muestra comienza con un bloque de proveedor que indica que Google Cloud es el proveedor. La región para la implementación se especifica en el bloque del proveedor.

El bloque de recursos especifica una instancia de Google Cloud Compute Engine o una máquina virtual. Los detalles de la instancia que se creará se especifican en el bloque de recursos.

El bloque de salida especifica una variable de salida para el módulo de Terraform. En este caso, se asignará un valor a la variable de salida "instance_ip".

Lab

Cómo compilar una canalización de DevOps

En este lab, compilará una canalización de DevOps con Cloud Source Repositories, Cloud Build y Container Registry. Primero, creará un repositorio de Git. Luego, escribirá una aplicación de Python sencilla y la agregará al repositorio. Después, probará su aplicación web en Cloud Shell y, posteriormente, definirá una compilación de Docker.

Cuando la defina, utilizará Cloud Build para crear una imagen de Docker y almacenarla en Container Registry. Luego, aprenderá a automatizar compilaciones con activadores. Después de definir un activador, lo probará modificando su programa y enviando el cambio al repositorio de Git.

Revisión del lab

Cómo compilar una canalización de DevOps

En este lab, aprendió a usar herramientas de Google Cloud para crear una canalización automatizada sencilla de integración continua. Utilizó Cloud Source Repositories para crear un repositorio de Git y, luego, usó Cloud Build y algunos activadores para automatizar la creación de las imágenes de Docker después de asignar código al repositorio.

Cuando Cloud Build creó sus imágenes de Docker, las almacenó en Container Registry. Usted aprendió a acceder a esas imágenes y probarlas en una VM de Compute Engine.

Puede continuar con un recorrido por el lab, pero recuerde que la interfaz de usuario de Google Cloud puede cambiar, por lo que su entorno podría ser un poco diferente.

Repaso

Automatización de DevOps

En este módulo, aprendió sobre los servicios que puede utilizar para automatizar la implementación de sus recursos y servicios de nube. Utilizó Cloud Source Repositories, Cloud Build, activadores y Container Registry para crear canalizaciones de integración continua. Una canalización de este tipo automatiza la creación de paquetes de implementación como las imágenes de Docker cuando se modifica su código fuente.

También aprendió a automatizar la creación de la infraestructura mediante Terraform, una herramienta de infraestructura como código.

Cuestionario

¿Qué herramientas de Google Cloud se pueden utilizar para crear una canalización de integración continua?

- A. Cloud Source Repositories
- B. Cloud Build
- C. Container Registry
- D. Todas las opciones anteriores

¿Qué herramientas de Google Cloud se pueden utilizar para crear una canalización de integración continua?

- A. Cloud Source Repositories
- B. Cloud Build
- C. Container Registry
- D. Todas las opciones anteriores

Cuestionario

¿Qué herramientas de Google Cloud se pueden utilizar para crear una canalización de integración continua?

- A. Cloud Source Repositories
- B. Cloud Build
- C. Container Registry
- D. Todas las opciones anteriores

- A. Si bien se puede usar Cloud Source Repositories para crear una canalización de integración continua, esta no es la opción correcta.
- B. Si bien se puede usar Cloud Build para crear una canalización de integración continua, esta no es la opción correcta.
- C. Si bien se puede usar Container Registry para crear una canalización de integración continua, esta no es la opción correcta.
- D. Todas las opciones son correctas. Source Repositories proporciona un repositorio de Git privado, Cloud Build crea contenedores y Container Registry es un repositorio de imágenes de Docker que realiza análisis de vulnerabilidades. Estos tres componentes suelen usarse en las canalizaciones de integración continua. Cuando se recibe una confirmación, se compila y prueba el código para crear una imagen que se publica en un registro.

Cuestionario

Mencione algunos de los motivos para automatizar la creación de la infraestructura usando herramientas de código como Deployment Manager y Terraform.

Mencione algunos de los motivos para automatizar la creación de la infraestructura usando herramientas de código como Deployment Manager y Terraform.

Cuestionario

Mencione algunos de los motivos para automatizar la creación de la infraestructura usando herramientas de código como Deployment Manager y Terraform.

Es más fácil crear entornos de desarrollo, pruebas y producción con el mismo esfuerzo.

Es más fácil modificar y corregir la infraestructura a lo largo del tiempo.

Simplifica la administración.

Se automatizan el aprovisionamiento y el retiro de servicio ordenado.

Ahorra dinero.

Los entornos de pruebas y de implementación deben ser idénticos o tan parecidos como sea posible. Automatizar su aprovisionamiento reduce los errores que podrían producirse con los procesos manuales, lo que facilita modificar la infraestructura a medida que los requisitos y las tecnologías cambian. Con la automatización, también es mucho más fácil manejar la administración y los permisos mediante IAM. Cuando se automatiza el aprovisionamiento, este se simplifica y se vuelve repetible, y también se automatiza el retiro de servicio ordenado, por lo que solo se generan costos de infraestructura cuando esta se utiliza.

Los entornos de pruebas y de implementación deben ser idénticos o tan parecidos como sea posible.

Automatizar su aprovisionamiento reduce los errores que podrían producirse con los procesos manuales, lo que facilita modificar la infraestructura a medida que los requisitos y las tecnologías cambian.

Con la automatización, también es mucho más fácil manejar la administración y los permisos mediante IAM.

Cuando se automatiza el aprovisionamiento, este se simplifica y se vuelve repetible, y también se automatiza el retiro de servicio ordenado, por lo que solo se generan costos de infraestructura cuando esta se utiliza.

Cuestionario

¿Qué función de Google Cloud sería la más fácil de utilizar para automatizar una compilación cuando se asigna código al repositorio de su código fuente?

- A. Activadores de Cloud Build
- B. Cloud Functions
- C. App Engine
- D. Cloud Scheduler

¿Qué función de Google Cloud sería la más fácil de utilizar para automatizar una compilación cuando se asigna código al repositorio de su código fuente?

- A. Activadores de Cloud Build
- B. Cloud Functions
- C. App Engine
- D. Cloud Scheduler

Cuestionario

¿Qué función de Google Cloud sería la más fácil de utilizar para automatizar una compilación cuando se asigna código al repositorio de su código fuente?

A. Activadores de Cloud Build

B. Cloud Functions

C. App Engine

D. Cloud Scheduler

- A. Respuesta correcta. Los activadores de Cloud Build están diseñados para activar automáticamente una compilación cuando se modifica el código fuente.
- B. Respuesta incorrecta. Si bien se podrían usar para compilar una solución, por ejemplo, con webhooks, este no es el caso de uso de Cloud Functions, que se emplea para backends sin servidores y cuenta con compatibilidad y escalabilidad integradas.
- C. Respuesta incorrecta. App Engine es una plataforma que sirve para implementar aplicaciones que suelen emplearse como microservicios.
- D. Respuesta incorrecta. Cloud Scheduler es un programador de clase empresarial. Automatizar una compilación es un proceso activado por un evento, no un proceso programado.

La respuesta A es correcta. Los activadores de Cloud Build están diseñados para activar automáticamente una compilación cuando se modifica el código fuente.

La respuesta B no es correcta. Si bien se podrían usar para compilar una solución, por ejemplo, con webhooks, este no es el caso de uso de Cloud Functions, que se emplea para backends sin servidores y cuenta con compatibilidad y escalabilidad

integradas.

La respuesta C no es correcta. App Engine es una plataforma que sirve para implementar aplicaciones que suelen emplearse como microservicios.

La respuesta D no es correcta. Cloud Scheduler es un programador de clase empresarial. Automatizar una compilación es un proceso activado por un evento, no un proceso programado.

Más recursos

Soluciones de Google Cloud para DevOps

<https://cloud.google.com/devops/>

Deployment Manager

<https://cloud.google.com/deployment-manager/>

Terraform en Google Cloud

<https://cloud.google.com/community/tutorials/getting-started-on-gcp-with-terraform>

Esta es una lista de recursos útiles para obtener más detalles sobre temas analizados en esta sección.