Documento técnico de AWS

Integración y entrega continuas para redes 5G en AWS

Integración y entrega continuas para redes 5G en AWS: Documento técnico de AWS

Table of Contents

Integración y entrega continuas para redes 5G en AWS

Fecha de publicación: 08 de marzo de 2021 (Revisiones del documento)

Resumen

Este documento técnico presenta la integración y la entrega continuas (CI/CD) para las redes 5G y cómo las herramientas y servicios de Amazon Web Services (AWS) pueden utilizarse para automatizar por completo la implementación y las actualizaciones de las funciones de dichas redes. El documento ofrece una descripción detallada de las distintas etapas de CI/CD para las funciones de red 5G, incluida la configuración de la red, la implementación de la infraestructura, la implementación de funciones de red nativas en la nube y las actualizaciones continuas de las funciones de red. También proporciona detalles sobre la integración con herramientas de código abierto y de terceros con fines de pruebas, observabilidad y orquestación.

Este documento técnico está dirigido a los proveedores de servicios de comunicación (CSP) y a los proveedores de software independientes (ISV).

Introducción

Tradicionalmente, el desarrollo, las pruebas de integración de laboratorio y campo, así como la implementación en producción de nuevos nodos de red o nuevas características en una red celular tardaban semanas o incluso meses en garantizar la estabilidad de los servicios de telecomunicaciones críticos para la misión y para el negocio. El largo ciclo de implementación se debía a la arquitectura monolítica de nodos de red tradicionales, un entorno de múltiples proveedores y muchas interfaces de punto a punto entre las entidades de red de las redes móviles 2G, 3G y 4G.

Tal y como se describía en el documento técnico de Evolución de la red 5G con AWS, las redes móviles 5G, estandarizadas por 3GPP, admiten ahora una arquitectura nativa en la nube que se habilita mediante la virtualización y la creación de contenedores. Más concretamente, las redes 5G introducen y respaldan un nuevo paradigma de la arquitectura de microservicios, sin estado y basada en servicios.

Esta arquitectura 5G implica que distintas funciones de red puedan funcionar como servicios independientes levemente acoplados que se comunican entre sí a través de interfaces y API bien definidas. Lo más importante es que cada una de las funciones de red puede actualizarse de forma independiente. Este cambio de arquitectura en 5G permite a los CSP lograr una mayor agilidad y eficacia operativa al facilitar una implementación más frecuente de actualizaciones de las funciones de red, a la vez que se mantienen las pruebas, los requisitos de seguridad y los estándares a través de la automatización.

La integración y la implementación de características nuevas para un CSP suelen comenzar cuando el proveedor de funciones de red publica un nuevo paquete de software de funciones de red, como una imagen de Docker en una función de red basada en contenedores, o un nuevo archivo de configuración, como un gráfico de Helm en el caso de una aplicación de Kubernetes. (Un gráfico de Helm es una colección de archivos que describen un conjunto de recursos de Kubernetes relacionados).

La idea de usar el paradigma de CI/CD para la implementación de funciones de red 5G está ganando terreno, pero la materialización de esta idea ha supuesto un desafío en el sector de las telecomunicaciones.

AWS ha sido pionero en el desarrollo de nuevas herramientas de CI/CD para la entrega de software destinadas a ayudar a una gran variedad de industrias a desarrollar e implementar cambios de software de forma rápida, sin por ello dejar de lado la estabilidad y la seguridad de los sistemas.

Estas herramientas incluyen un conjunto de servicios de desarrollo de software y operaciones (DevOps), como AWS CodeStar, CodeCommit, CodePipeline, CodeBuild y CodeDeploy.

AWS también prioriza la idea de la infraestructura como código (IaC) mediante el kit de desarrollo en la nube de AWS (AWS CDK), AWS CloudFormation y herramientas de terceros basadas en API, como Terraform. Con estas herramientas, AWS puede almacenar los procesos de implementación de una función de red en AWS como código fuente y mantener este código fuente de IaC en la canalización de CI/CD para realizar una entrega continua.

Este documento técnico describe los procesos detallados para aprovechar las herramientas de CI/CD e IaC de AWS para la implementación y actualización de la función de red 5G. Además, el documento trata la integración con herramientas de terceros para pruebas, observabilidad y orquestación.

Las herramientas de CI/CD de AWS no se limitan a las funciones de red 5G. También se emplean para automatizar la implementación de redes 4G, lo que permite a los CSP implementar y actualizar las funciones de red 4G de forma rápida y eficaz. La mayoría de las funciones de red 4G se basan en la función de red virtual (VNF). Los conjuntos de herramientas de CI/CD de AWS, como AWS CloudFormation, se pueden usar para automatizar la implementación de funciones de red virtual 4G, lo que mejora la escalabilidad y la agilidad en las implementaciones de redes 4G.

Integración y entrega continuas

Integración continua

La integración continua (CI) es un proceso de software en el que los desarrolladores insertan su código con regularidad en un repositorio central, como AWS CodeCommit o GitHub. Cada inserción de código desencadena una compilación automatizada, seguida de la ejecución de pruebas. El objetivo principal de CI es detectar los problemas del código en una etapa inicial, mejorar la calidad de este y reducir el tiempo que se tarda en validar y en publicar nuevas actualizaciones de software.

Entrega e implementación continuas

La entrega continua (CD) es un proceso de software en el que se implementan artefactos en el entorno de prueba, el entorno de ensayo y el entorno de producción. La entrega continua puede automatizarse por completo o bien contar con fases de aprobación en puntos críticos. Esto garantiza que todas las aprobaciones necesarias antes de la implementación, como la aprobación de administración de versiones, se hayan aplicado. Cuando la entrega continua se implementa correctamente, los desarrolladores dispondrán siempre de un artefacto listo para su implementación que se ha sometido a un proceso de pruebas estandarizado.

Con la implementación continua, las revisiones se implementan en un entorno de producción de forma automática sin la aprobación explícita de un desarrollador, con lo que se automatiza todo el proceso de publicación de software. Esto permite un bucle de retroalimentación de los clientes en las primeras etapas del ciclo de vida del producto.

Con la implementación continua, cada cambio que se confirma y que supera las pruebas automatizadas se envía a producción de forma automática. La entrega continua no tiene como objetivo publicar todos los cambios que se confirmen y pasen las pruebas automatizadas a producción de forma inmediata, sino garantizar que todos los cambios estén listos para pasar a producción.

Infraestructura como código

Tal y como se detalla en el documento técnico Evolución de la red 5G con AWS, la infraestructura como código (IaC) es un factor clave para automatizar el proceso de aprovisionamiento y la administración del ciclo de vida tanto de la aplicación como de su entorno. En lugar de basarse en pasos de realización manual, tanto los desarrolladores como los administradores de redes

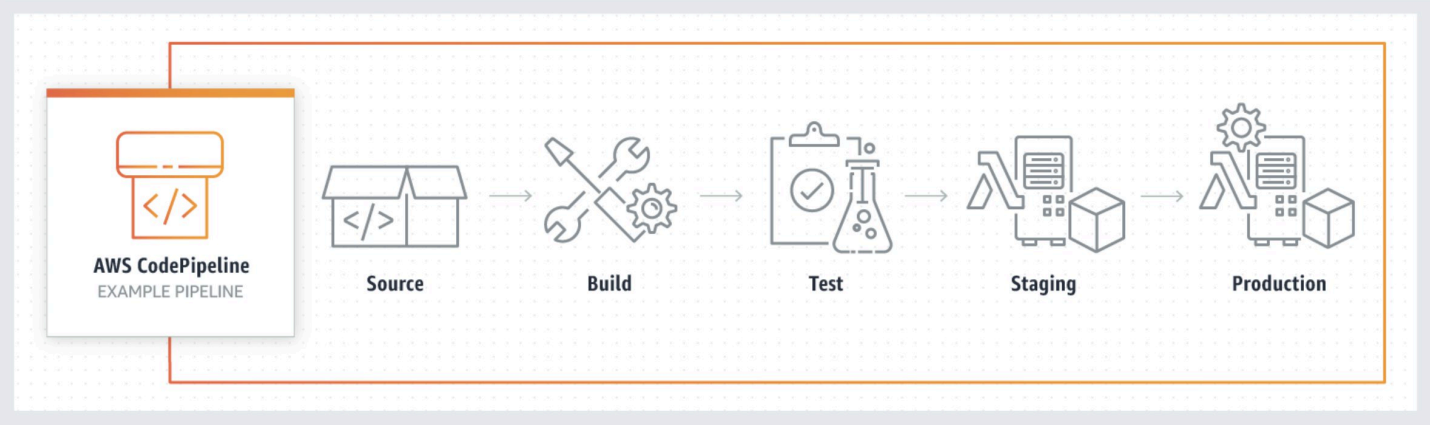
y TI pueden crear instancias de la infraestructura mediante archivos de configuración. IaC trata esos archivos de configuración como código de software. Estos archivos pueden utilizarse para generar un conjunto de artefactos; en concreto, los servicios de computación, almacenamiento, red y aplicaciones que componen un entorno operativo. IaC elimina las desviaciones de la configuración a través de la automatización, lo que aumenta la velocidad y la agilidad de las implementaciones de infraestructura.

En el caso de la implementación de virtualización de funciones de red (NFV) en AWS, este marco de IaC aporta valor desde el punto de vista de la orquestación. Cada uno de los pasos, desde la creación de la nube virtual privada (VPC) hasta la implementación de funciones de red, se puede programar, administrar como código fuente y mantener con el control de versiones en AWS CodeCommit.

Este marco de IaC para funciones de red da como resultado la creación e implementación de una infraestructura y de funciones de red repetibles y fiables, que pueden extenderse a la automatización integral (E2E) de la administración de segmentos de red y de la administración del ciclo de vida de los servicios. AWS proporciona un completo conjunto de herramientas para crear, mantener e implementar una infraestructura mediante programación, de forma descriptiva y declarativa, con servicios como AWS CloudFormation, AWS CDK, AWS CDK for Kubernetes y la exposición de la API de todos los servicios de AWS.

CI/CD en AWS

CI/CD se puede representar como una canalización, donde el código nuevo se envía en un extremo, se prueba en una serie de etapas (origen, creación, prueba, ensayo y producción) y, a continuación, se publica como código listo para producción.



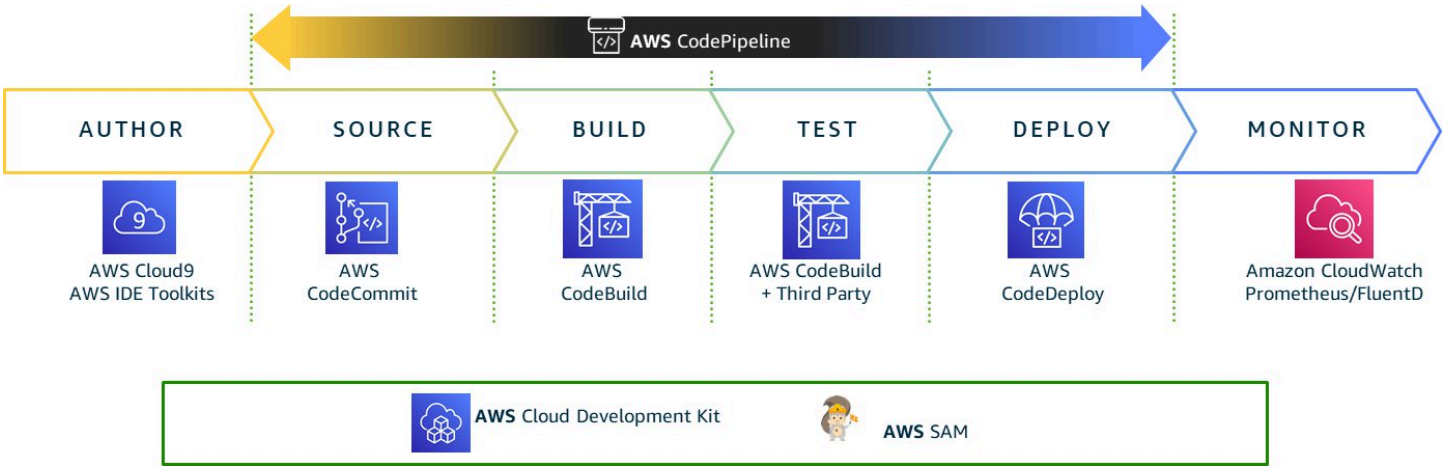
Información general de la canalización de CI/CD

Cada etapa de la canalización de CI/CD se estructura como unidad lógica en el proceso de entrega. Cada etapa actúa como una puerta que examina un aspecto determinado del código. Se supone que, a medida que el código avanza a través de la canalización, su calidad va mejorando en las etapas posteriores, ya que se van verificando más aspectos de este. Los problemas que se descubren en una etapa temprana impiden que el código avance a través de la canalización. Los resultados de las pruebas se envían inmediatamente al equipo y todas las compilaciones y versiones posteriores se detienen si el software no supera la etapa.

AWS aporta un completo conjunto de herramientas para desarrolladores de CI/CD a fin de acelerar los ciclos de desarrollo y lanzamiento de software. AWS CodePipeline automatiza las fases de compilación, prueba e implementación del proceso de lanzamiento cada vez que se produce un cambio de código, en función del modelo de lanzamiento definido. Esto permite una entrega rápida y fiable de las características y las actualizaciones.

Las canalizaciones de código pueden integrarse con otros servicios. Estos pueden ser servicios de AWS, como Amazon Simple Storage Service (Amazon S3), o productos de terceros, como GitHub. AWS CodePipeline puede abordar diversos casos de uso de desarrollo y operación, entre los que se incluyen los siguientes:

- Compilación, creación y prueba de código con AWS CodeBuild
- Entrega continua de aplicaciones basadas en contenedores a la nube
- Validación previa a la implementación de los artefactos (como descriptores e imágenes de contenedor) requerida para las funciones de servicio de red o funciones de red nativas en la nube concretas
- Pruebas funcionales, de integración y de rendimiento para una función de red en contenedores o una función de red virtual (CNF/VNF), incluidas las pruebas de base de referencia y de regresión
- Pruebas de fiabilidad y recuperación de desastres (DR).



Componentes de la canalización de CI/CD de AWS

AWS puede configurar canalizaciones de CI/CD con las herramientas para desarrolladores de AWS siguientes:

- AWS CodeCommit
- AWS CodeBuild
- AWS CodePipeline
- AWS CodeDeploy
- Amazon Elastic Container Registry
- AWS CodeStar

La creación de canalizaciones de CI/CD puede automatizarse con AWS CDK y AWS CloudFormation. En el dominio de virtualización de las funciones de redes (NFV), esta automatización nativa de AWS puede integrarse en un marco de administración y orquestación (MANO) y en el marco de orquestación de servicios del CSP.

El proceso de CI/CD incluye los pasos siguientes:

- Configuración de red: AWS CDK y AWS CloudFormation inician la creación de los requisitos previos de red:
    - Pila de redes (VPC, subredes, puerta de enlace de NAT, tabla de enrutamiento y puerta de enlace de Internet)

- Implementación de la infraestructura: AWS CDK y AWS CloudFormation inician la creación de las pilas de recursos siguientes:

    - Pila de computación (creación de clústeres de Amazon Elastic Kubernetes Service (Amazon EKS), nodos de trabajo de EKS, AWS Lambda)
    - Pila de almacenamiento (buckets de Amazon S3, volúmenes de Amazon Elastic Block Store (Amazon EBS) y Amazon Elastic File System (Amazon EFS)
    - Pila de supervisión (CloudWatch , Amazon OpenSearch Service (OpenSearch Service)
    - Pila de seguridad (AWS Identity and Access Management (AWS IAM), grupos de seguridad de Amazon Elastic Compute Cloud (Amazon EC2) y listas de control de acceso a la red (NACL) de VPC

- Implementación de funciones de red en la nube (CNF): en esta etapa, se implementan funciones CNF en clústeres de EKS mediante Kubectl y las herramientas de gráficos de Helm. En esta etapa también se implementa cualquier aplicación o herramienta específica que las CNF necesiten para funcionar de forma eficaz (como Prometheus o Fluentd). Las CNF pueden implementarse mediante funciones Lambda o con AWS CodeBuild.
- Implementación y actualizaciones continuas: son una secuencia de pasos que se llevan a cabo de forma iterativa para implementar cambios que forman parte de los cambios de contenedor o de configuración que dan lugar a actualizaciones. Al igual que en el caso de la implementación de CNF, la implementación y las actualizaciones continuas pueden automatizarse mediante los servicios de AWS, con el desencadenador de AWS CodeCommit, Amazon Elastic Container Registry (Amazon ECR) o un sistema de origen de terceros, como GitLab Webhooks.

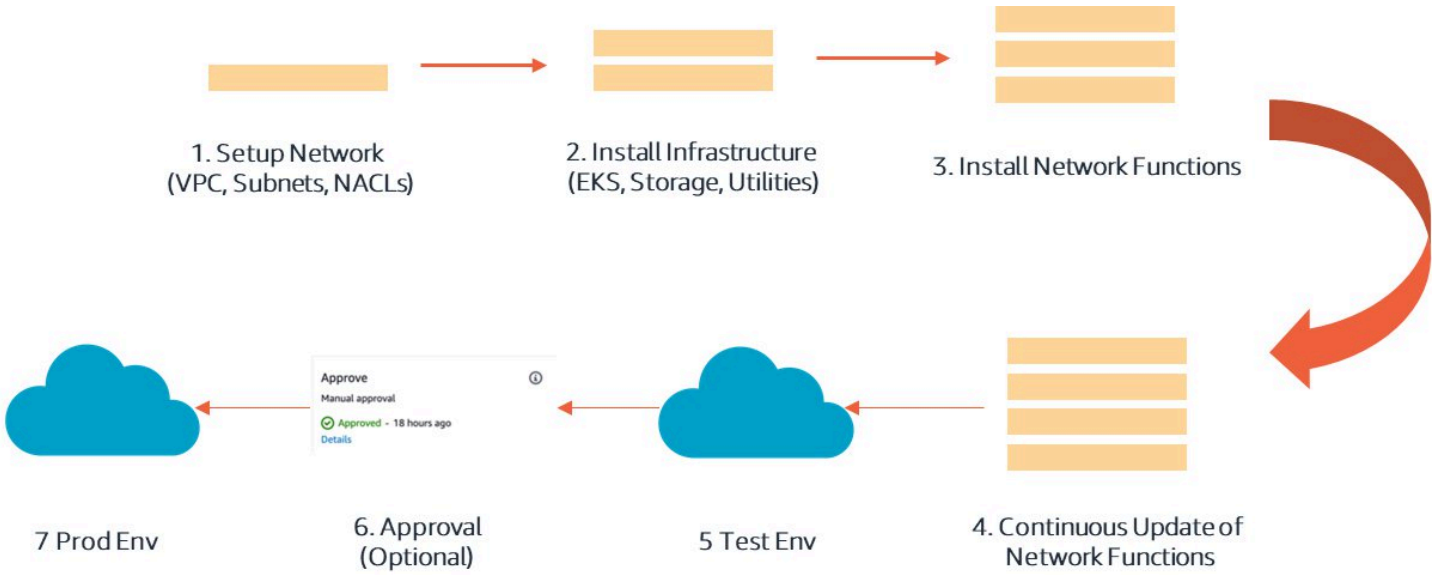


Diagrama del flujo de canalización de CI/CD de AWS

La canalización de CI/CD se crea con AWS CodePipeline y usa un servicio de entrega continua que modela, visualiza y automatiza los pasos necesarios para publicar el software. Al definir etapas en una canalización, puede recuperar código de un repositorio de código fuente, integrar ese código fuente en un artefacto divulgable, probar el artefacto e implementarlo en producción. Solo se implementará el código que haya superado correctamente todas estas etapas. Si lo desea, puede añadir otros requisitos a la canalización (como las aprobaciones manuales) como ayuda para garantizar que solo se implementen en producción los cambios aprobados.

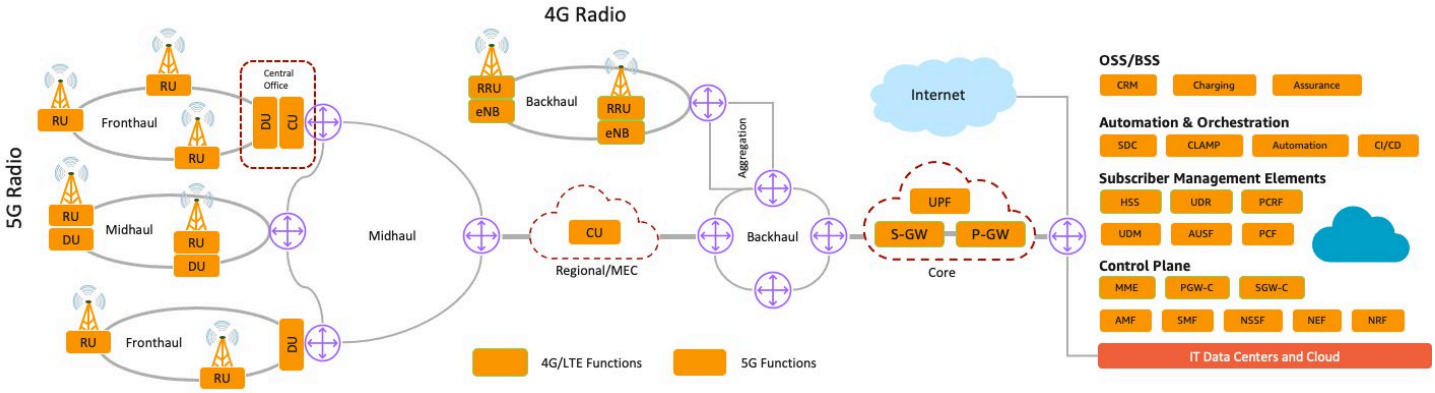
Redes 5G en AWS

El modelo típico de infraestructura de red 5G se compone de un sitio de radio 4G o 5G; una red fronthaul, midhaul o backhaul; un sitio de red central y un centro de datos de telecomunicaciones o TI. Los CSP pueden usar los servicios de AWS para crear una infraestructura de red 5G escalable y flexible y, a la vez, reducir el coste de inversión inicial. AWS se puede usar para implementar el centro de operaciones de red (NOC) virtual en la región que aloja el sistema de soporte de operaciones/sistema de soporte empresarial (OSS/BSS) y la mayoría de las funciones de red principales del plano de control.

AWS también puede aprovecharse para implementar la oficina central (CO) local o el centro de datos distribuido con una flota de instancias de AWS Outposts que alojen principalmente funciones del plano de usuario, como UPF (función de plano de usuario), unidad central (CU) de RAN y computación de borde multiacceso (MEC). En el documento técnico Evolución de la red 5G con

AWS se ofrece una explicación más detallada de la arquitectura de referencia y de las ventajas de la implementación de redes 5G en AWS.

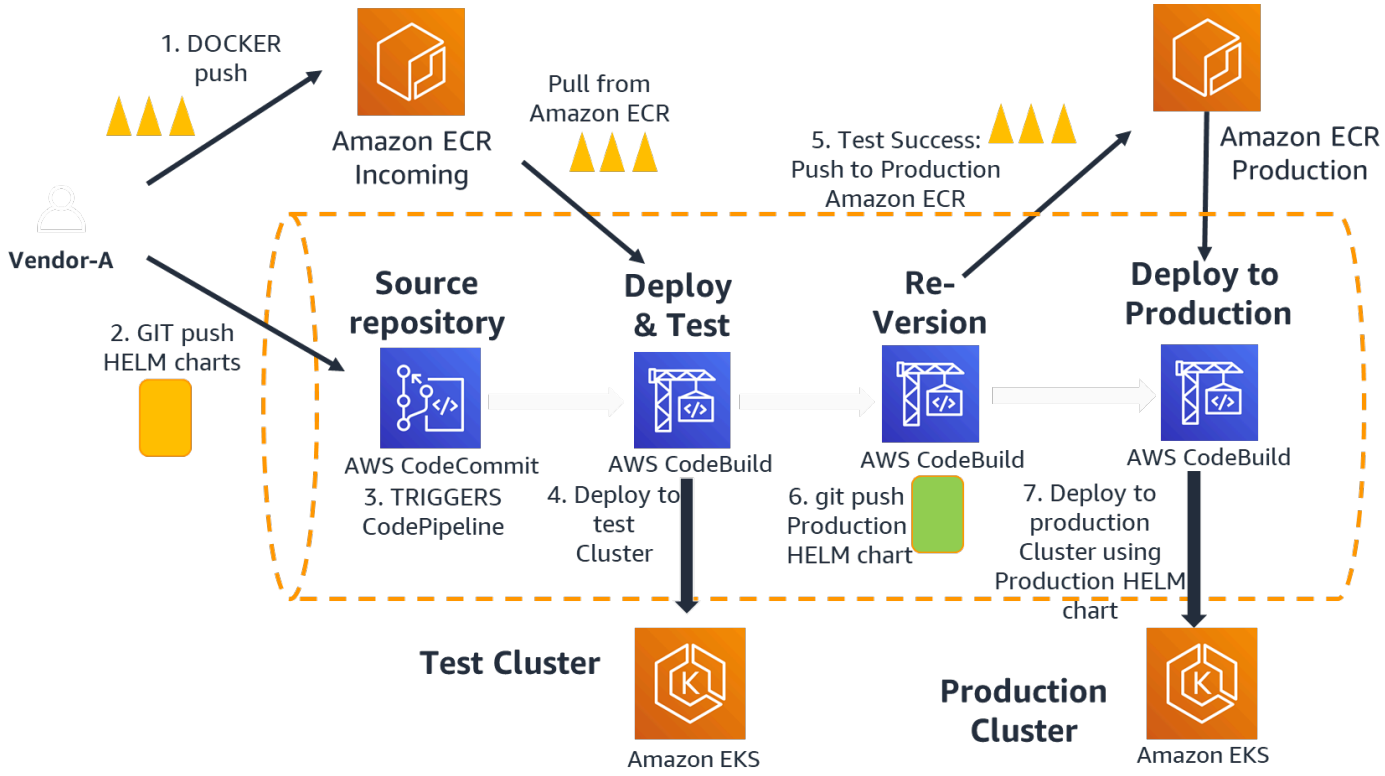
A la hora de implementar la red 5G en AWS, las herramientas de CI/CD de AWS (que se presentan en las secciones siguientes de este documento técnico) pueden facilitar la automatización total de la implementación, la actualización y la administración del ciclo de vida de las funciones de red 5G.



Arquitectura E2E de la red 5G

CI/CD en las redes 5G

La construcción de diseño de la infraestructura se almacena en forma de código con lenguaje declarativo. Esto permite que el CSP cuente con una reproducción repetible de la infraestructura con el mismo comportamiento esperado que se requiere. El código se mantiene en el repositorio de código y se configura una canalización para orquestar las actualizaciones en las pilas implementadas (por ejemplo, AWS CDK y AWS CloudFormation). AWS puede ayudar a crear una infraestructura como código (IaC) para la rápida incorporación de las funciones de los proveedores de software independientes (ISV).



Flujo de canalización de código

Los cambios en las configuraciones de las funciones de red nativas en la nube a través de gráficos de Helm se consideran desencadenadores de la ejecución de una canalización de CI/CD automática para las funciones de red.

AWS CodeCommit puede utilizarse para mantener los archivos de configuración y Amazon ECR se puede usar para conservar las imágenes de los contenedores.

Tal y como se muestra en la ilustración del flujo de canalización de código, cuando el ISV inserta nuevos cambios de código en el repositorio correspondiente (un gráfico de Helm, archivos de configuración o un archivo de propiedades), se desencadena la canalización de código. La canalización de código extrae la imagen de ECR y usa el gráfico de Helm para implementar la aplicación. Las pruebas de la aplicación nueva pueden integrarse en el marco de automatización de pruebas de terceros. En función del resultado, los CSP pueden dar su aprobación para la implementación de producción.

En la fase de origen de CodePipeline se buscan cambios en los archivos de configuración. Los proveedores válidos para la fase de origen son CodeCommit, Amazon S3, GitHub o AWS CloudFormation. Pueden integrarse sistemas de origen alternativos mediante el uso de funciones

Lambda para implementar Webhooks, lo que permite la integración basada en eventos entre Gitlab y AWS CodePipeline. Vea los enlaces siguientes para obtener una guía de implementación detallada.

- Webhooks con GitLab
- Integraciones de registro de contenedores

El diseño de la canalización de CI/CD debe tener en cuenta pasos de implementación críticos, como la implementación inicial, las pruebas y su promoción a producción una vez que los resultados de las pruebas concuerden con las expectativas y se hayan verificado con la base de referencia. Cada etapa del proceso de canalización proporciona artefactos de datos, que permiten la comparación y la toma de decisiones controlada por datos.

Pasos de la canalización de CI/CD de la aplicación

Cada etapa puede considerarse como una tarea independiente, lo que permite la incorporación de flujos de trabajo de validación e implementación adecuados para admitir el servicio de red y las funciones de red nativas en la nube. Las tareas de ejecución pueden incorporar herramientas de terceros adicionales, como generadores y simuladores de tráfico, lo que permite la validación integral de los servicios de red.

AWS proporciona un servicio sofisticado de AWS Step Functions (máquina de estado nativa en la nube) que se integra de forma nativa con otros servicios de AWS y también tiene la capacidad de integrarse con sistemas externos, como Jira o un marco de automatización de pruebas.

Pasos detallados de CI/CD

CI/CD se puede representar como una canalización, donde el código nuevo se envía en un extremo, se prueba en una serie de etapas (origen, creación, prueba, ensayo y producción) y, a continuación, se publica como código listo para producción.

A continuación, se indican los pasos para la implementación y las pruebas. La implementación y la configuración se dividen principalmente en cuatro secciones básicas:

- Configuración de red

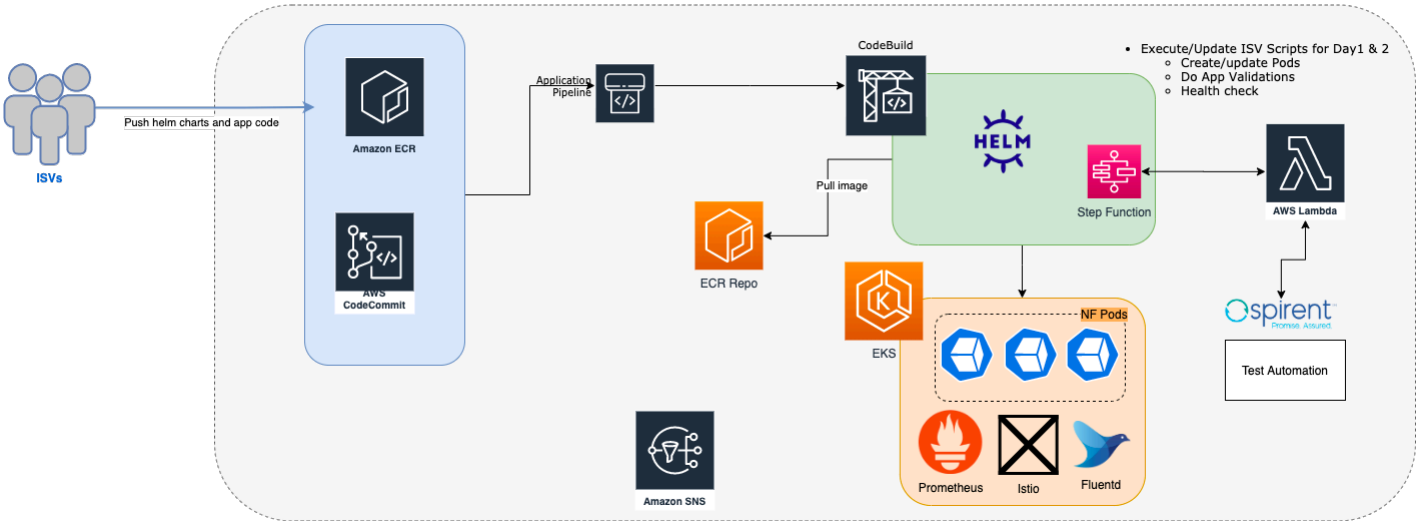
- Implementación de la infraestructura
- Implementación de funciones de redes nativas en la nube
- Entrega continua de CNF

Configuración de red

Concéntrese en la configuración de los requisitos previos de la infraestructura. Esto incluye la creación de la VPC, las redes, las subredes y las NACL, entre otros elementos. Diseñe el plan de la red IP; para ello, tenga en cuenta el plan de implementación de los ISV y del cliente (como las asignaciones estáticas frente a dinámicas y multitenencia). Este plan puede codificarse en AWS CDK o AWS CloudFormation. La ejecución de este código implementa los requisitos previos de la red de infraestructura en la nube .

Implementación de la infraestructura

La implementación de la infraestructura aprovisiona cualquier componente de la infraestructura. Esto incluye la generación del clúster de EKS y la infraestructura de respaldo, como EFS, nodos de trabajo de EKS, instancias de ELB y la configuración del clúster de acuerdo con los requisitos de la función de red nativa en la nube. Según los requisitos de CNF, AWS también implementa interfaces de red adicionales para los nodos, incluidas las interfaces de Multus. La mayoría de los pasos de implementación y configuración deben realizarse solo una vez para la aplicación y se actualizan solo cuando es necesario, como actualización de la aplicación.

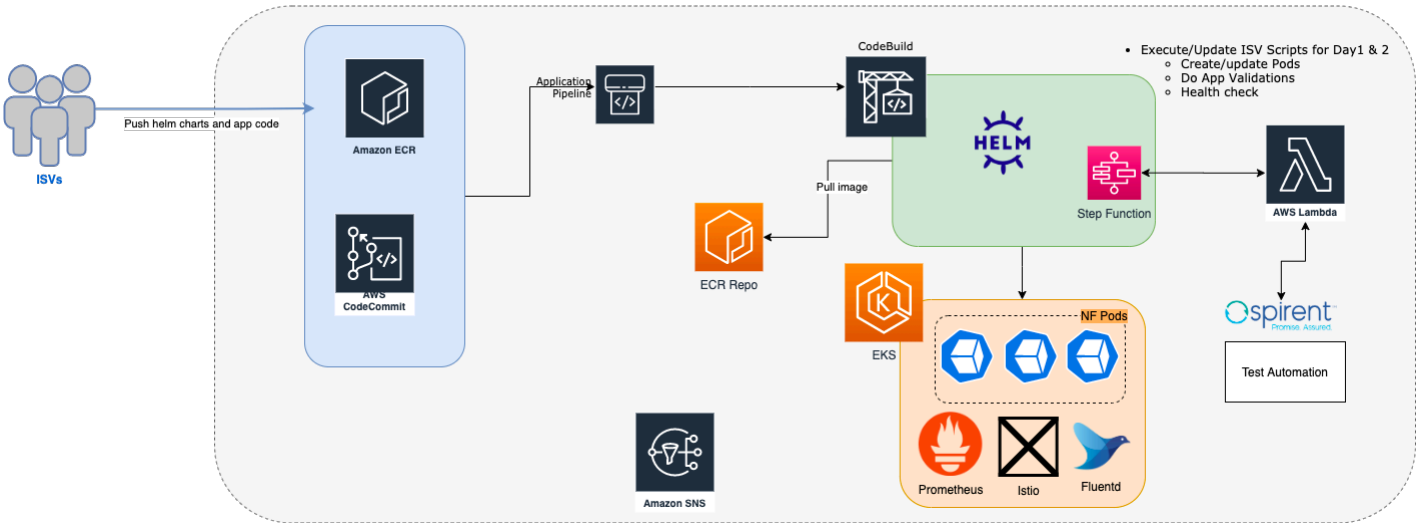


Implementación de la infraestructura con CDK

Implementación de funciones de redes nativas en la nube

La implementación de CNF está relacionada con la implementación de la aplicación. Como parte de la implementación de CNF, los gráficos de Helm de la aplicación se implementan a través de la canalización de código de CI/CD. Se incorpora la devolución de llamada para ejecutar scripts individuales específicos de la aplicación que implican principalmente comprobaciones previas y posteriores. Los gráficos de Helm se implementan en secuencia en función de las necesidades de la aplicación y comprueban el estado de los POD de Kubernetes antes de avanzar al paso siguiente de la implementación. A menudo, los ISV proporcionan un script encapsulador para ejecutar los gráficos de Helm y las comprobaciones de estado. Estos scripts de ISV se invocan desde AWS CodePipeline. Como parte de esta fase, se implementan los agentes de registro y supervisión (como Prometheus y Fluentd), además de Amazon CloudWatch, que registra y supervisa la infraestructura en la nube de la aplicación.

La canalización de código se integra en el marco de automatización de pruebas de terceros. La canalización de código puede llamar directamente a las API del marco de automatización de pruebas para ejecutar la prueba en la aplicación implementada, consultar los resultados de la prueba y analizar el resultado. Esto simplifica la implementación y las pruebas de la aplicación.

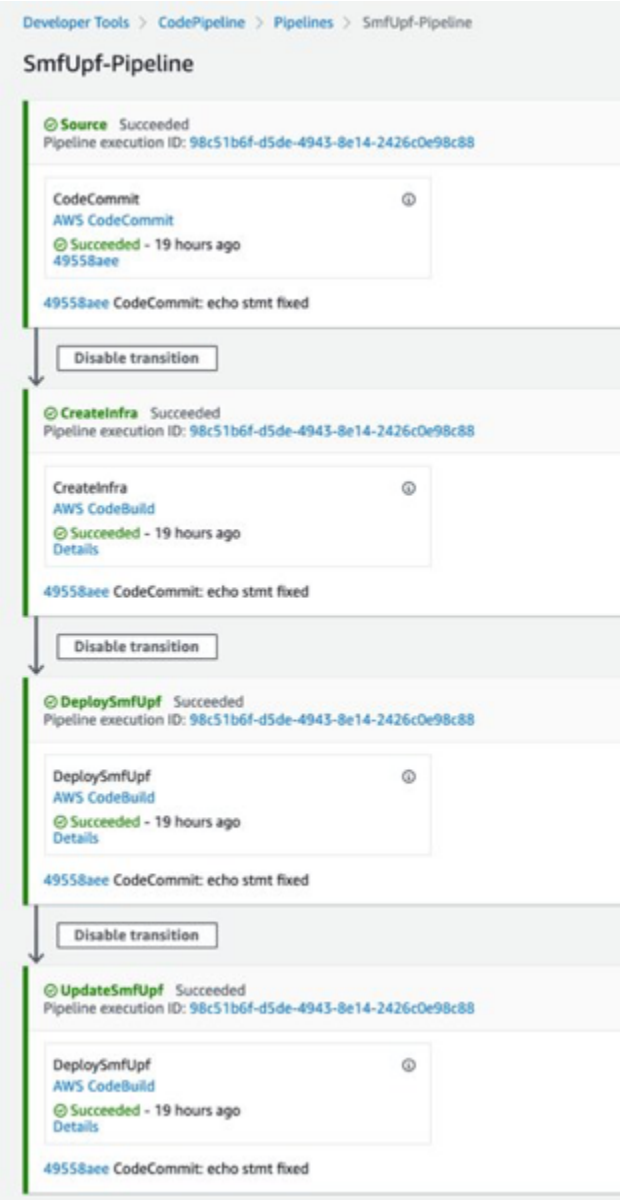


Implementación y actualización de una aplicación

A continuación, se muestra un ejemplo de la implementación de CNF de la función del plano de usuario/función de administración de sesiones (UPF/SMF) a través de AWS CodePipeline.

- Automatice todo el proceso de CI/CD con CodeCommit, CodeBuild y CodePipeline.

- Las tareas de creación de infraestructura e instalación de aplicaciones se integran como parte de la canalización.
- Los agentes de FluentD y Prometheus se instalan y se crean en paneles de Amazon CloudWatch.



Ejemplo de implementación de CNF de UPF/SMF

Entrega continua de CNF

Este paso se compone de una secuencia de pasos que se llevan a cabo de forma iterativa para implementar cambios que forman parte de los cambios de contenedor o de configuración que dan

lugar a actualizaciones. La entrega continua de CNF se automatiza a través de canalizaciones y es específica de las aplicaciones individuales. AWS usa gráficos estándar de Helm para actualizar instancias de CNF específicas. La canalización de código cuenta con comprobaciones previas y posteriores del estado de actualización de la aplicación. La canalización de CI/CD actualizada también se integra en un marco de automatización de pruebas para ejecutar pruebas automatizadas. Esta abstracción permite una implementación limpia de las funciones de red.

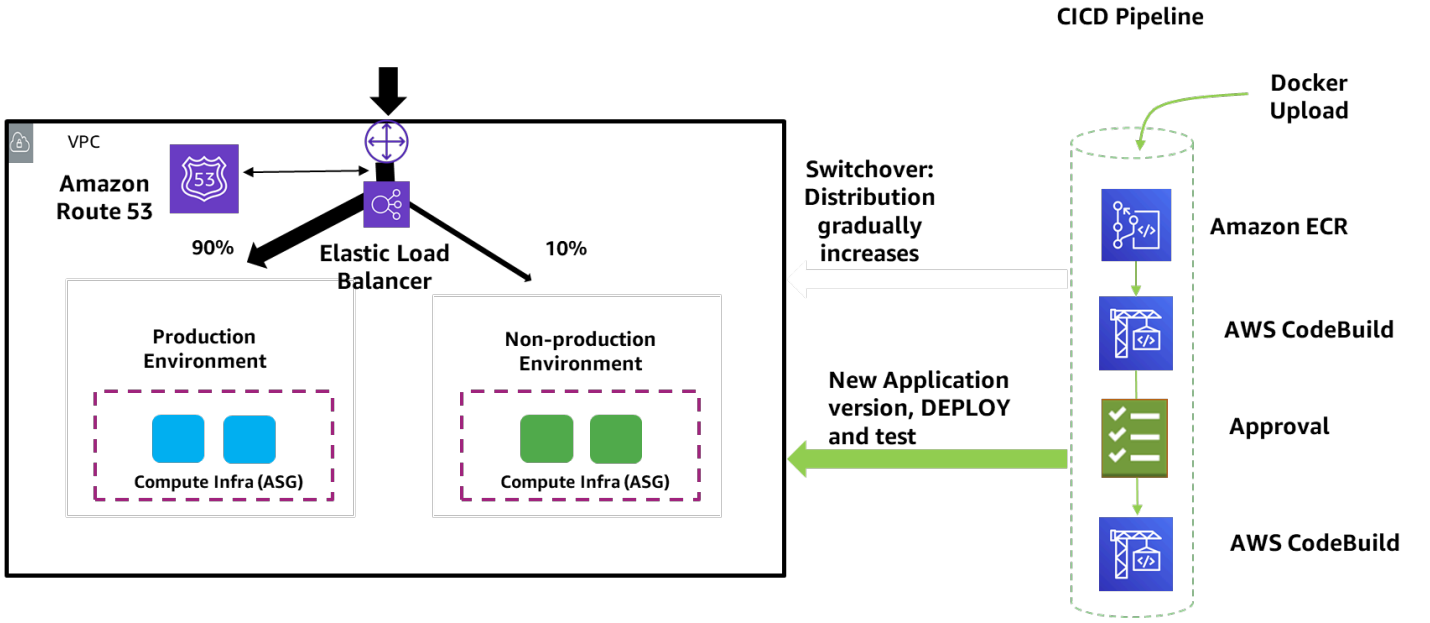
La entrega e implementación continuas de CNF pueden clasificarse en términos generales en las categorías siguientes:

- Actualizaciones de aplicaciones: la mayoría de las actualizaciones de las aplicaciones son cambios en los POD de aplicaciones de Kuberbetes. Estas actualizaciones pueden aplicarse de forma automática a través de la canalización de código. La mayoría de las CNF admiten actualizaciones in situ al proporcionar varias instancias de POD de aplicaciones. La existencia de instancias múltiples permite un enfoque de actualización gradual. No todos los cambios de POD de las aplicaciones admiten la actualización de Helm. Las canalizaciones tienen en cuenta estas variaciones y usan la instalación o eliminación de Helm según sea necesario.
- Actualizaciones principales: las actualizaciones principales son sobre todos cambios de esquema de base de datos. Este cambio no se puede aplicar sin causar cierto tiempo de inactividad. El enfoque estándar para estos cambios consiste en eliminar la aplicación y volver a crear los pods relevantes. Durante el proceso, puede que la aplicación no esté disponible. Estas son las herramientas que se usan para las actualizaciones:

    - AWS CloudFormation permite que los clientes describan y aprovisionen todos los recursos de la infraestructura en plantillas de JSON o YAML. AWS CloudFormation proporciona un poderoso mecanismo de extensión a través de recursos personalizados con respaldo de Lambda. Los clientes pueden ampliar AWS CloudFormation más allá de los recursos de AWS y aprovisionar el recurso requerido en otros entornos; por ejemplo, recursos locales en entornos híbridos. AWS CDK ofrece a los desarrolladores la posibilidad de crear código con lenguajes de programación familiares de mayor nivel, como Python, TypeScript, JavaScript, Java y C# para, después, compilar el código en un formato JSON de AWS CloudFormation de nivel inferior, que puede implementarse a continuación.
    - Implementación azul/verde: AWS admite y recomienda las implementaciones de tipo azul/verde y basadas en valores controlados tanto en entornos de prueba como de producción. Las implementaciones azul/verde permiten a los clientes probar una nueva versión de la aplicación en un entorno contenido. Proporcionan un método

fácil y elegante de transferir el tráfico de producción. Las implementaciones basadas en valores controlados amplían este concepto, ya que permiten que un entorno verde que no sea de producción se pruebe con una pequeña proporción del tráfico de producción para descubrir cualquier problema causado por este tipo de tráfico. La nueva versión de la aplicación se prueba con tráfico de prueba interno simulado y con pequeñas cantidades de tráfico de producción, lo que aporta confianza al usuario antes de transferir el tráfico de producción. El tráfico de producción se incrementa de forma gradual hasta que se completa la transición. La implementación implica grupos de destino de ELB ponderados y DNS ponderados también.

- La automatización se puede lograr mediante la configuración de AWS CodePipeline con las etapas de implementaciones de tipo azul/verde y basadas en valores controlados. La etapa de aprobación puede llevarse a cabo de forma manual inicialmente durante el aprovisionamiento, pero más tarde debe automatizarse por completo. En los entornos de prueba, se recomienda probar siempre con una acción de reversión para validar la compatibilidad con versiones anteriores y posteriores antes de implementar en producción. La implementación azul/verde en clústeres con malla de servicios depende del soporte que proporcione la aplicación final y la puerta de enlace de enrutamiento para que la malla de servicios logre una transición sin problemas.
- AWS Systems Manager proporciona una interfaz de usuario unificada para poder ver datos operativos de diferentes servicios de AWS que las funciones de red implementadas por CI/CD utilizan. Systems Manager le permite automatizar las tareas operativas en sus recursos de AWS.



Implementación de valores controlados

Seguridad

La seguridad es un elemento clave. A continuación, se muestra una lista de los pasos de seguridad que el proceso de CI/CD de AWS tiene en cuenta al implementar una aplicación.

- Origen: el repositorio de ECR asignado al proveedor está configurado con la marca para escanear al insertar habilitada, de modo que cualquier carga de imágenes de Docker se someterá inmediatamente a un análisis de seguridad. Todas las vulnerabilidades y exposiciones comunes (CVE) conocidas se marcarán con notificaciones. Aparte de ECR, cuando los proveedores colocan gráficos en el repositorio de AWS CodeCommit, se les pide que cifren las contraseñas que se usan con Secrets Manager, en lugar de hacerlo con texto sin formato.
- Integridad de los artefactos: los artefactos que se usan en la canalización se cifran en reposo (con claves administradas de AWS) o en tránsito (con SSL/TLS).
- Usuarios y roles de IAM: los permisos que se proporcionan a los usuarios o a los recursos se basan en el principio de privilegios mínimos. Debería haber una relación de confianza entre los roles de IAM que puede ser necesario configurar si opera en varios recursos de diferentes servicios. Por ejemplo, AWS CodeBuild necesita permiso para ejecutar comandos en un clúster de Amazon EKS.

- Auditoría: la capacidad de auditoría que ofrece AWS CloudTrail realiza un seguimiento de cada una de las llamadas a la API entre los distintos servicios y operaciones del usuario y permite la evaluación de eventos anteriores.
- Análisis de vulnerabilidades de imágenes: las imágenes de CNF que se cargan en Amazon ECR se analizan de forma automática para detectar vulnerabilidades de seguridad. Hay un informe de los resultados del análisis disponible en la AWS Management Console y también puede recuperarse a través de la API. Las conclusiones pueden enviarse posteriormente a los operadores del CSP para que tomen medidas correctivas, incluido el reemplazo de la imagen de CNF.

Se realizan comprobaciones de seguridad en varias etapas de la canalización para garantizar que la imagen recién cargada es segura y que satisface las comprobaciones de cumplimiento deseadas, de modo que puede enviarse una notificación a los CSP para su aprobación:

- El registro de contenedores busca cualquier vulnerabilidad de CVE abierta.
- La configuración se analiza durante la etapa de prueba para detectar posibles filtraciones de información o patrones conocidos de información de identificación personal (PII), lo que desencadena reglas de comprobación de cumplimiento para problemas como puertos TCP/UDP que se hayan abierto de forma inesperada y vulnerabilidades de DOS.
- Se comprueba la compatibilidad con versiones anteriores y posteriores para garantizar la seguridad de las actualizaciones o reversiones.

Además de la aplicación, es fundamental aprovisionar seguridad de la canalización al garantizar la transferencia cifrada de los artefactos en todas las etapas, ya sea en reposo o en tránsito.

Observabilidad

AWS posibilita la observabilidad de las funciones CNF de 5G implementadas en AWS de forma predeterminada. Amazon CloudWatch habilita esta característica. CloudWatch proporciona la visibilidad total de sus recursos y aplicaciones en la nube.

Amazon CloudWatch aplica cuatro pasos principales durante este proceso:

1. Recopilar: recopile las métricas y los registros de todos sus recursos, aplicaciones y servicios de AWS que se ejecutan en servidores locales y de AWS.

2. Supervisar: visualice las aplicaciones y la infraestructura con los paneles de CloudWatch, correlacione los registros y las métricas en paralelo para solucionar problemas y establezca alertas con las alarmas de CloudWatch.
3. Actuar: automatice la respuesta ante los cambios operativos con CloudWatch Events y AWS Auto Scaling.
4. Analizar: métricas de hasta un segundo, retención de datos ampliada (15 meses) y análisis en tiempo real con CloudWatch Metric Math.

El agente de Amazon CloudWatch se instala en el clúster de Kubernetes del cliente. Dicho agente admite las características de configuración, detección y extracción de métricas de Prometheus y enriquece y publica todas las métricas y los metadatos de alta fidelidad de Prometheus con formato de métricas integradas (EMF) para CloudWatch Logs.

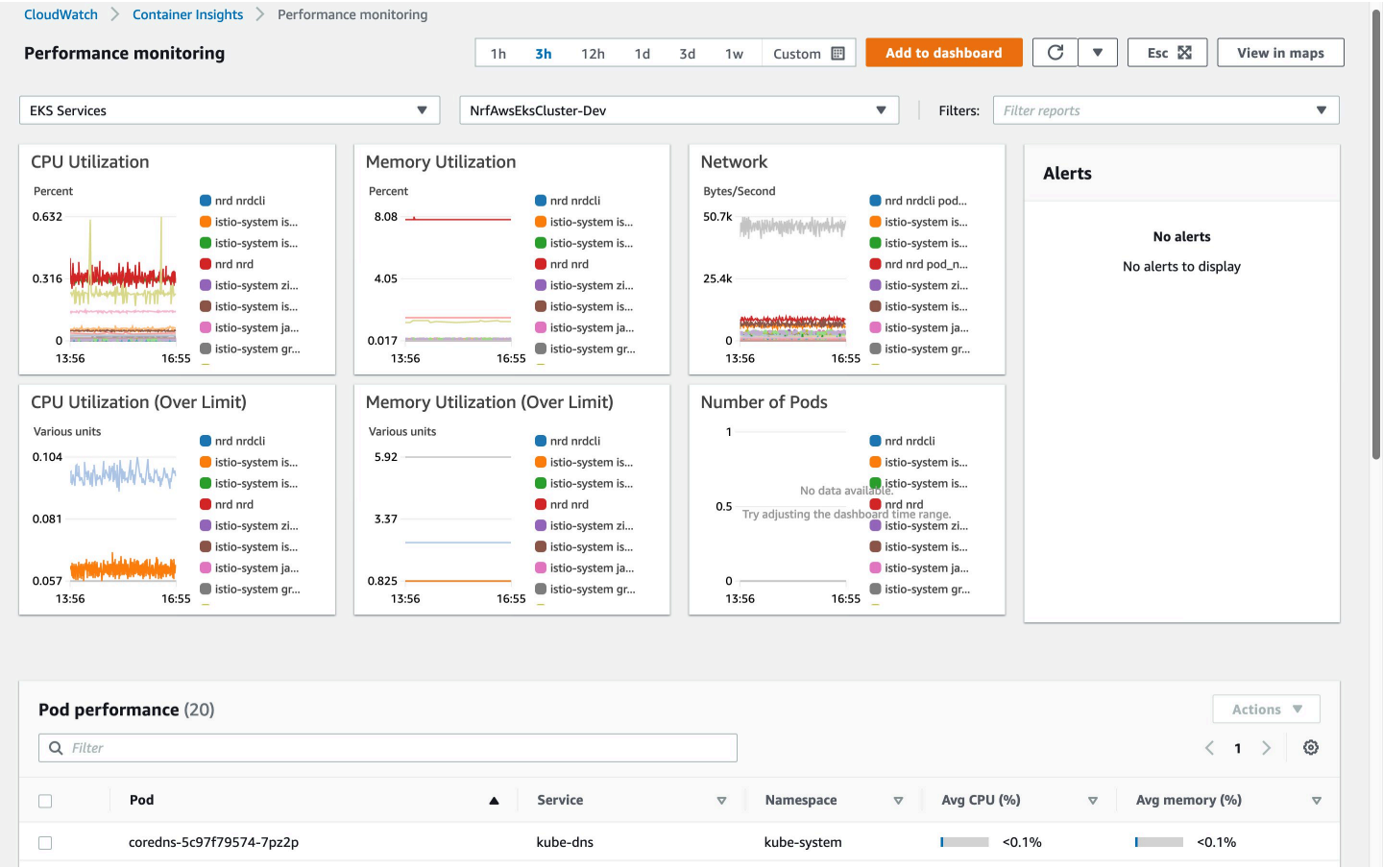
Amazon CloudWatch Container Insights automatiza la detección y la recopilación de métricas de Prometheus de aplicaciones en contenedores. Recopila, filtra y crea de forma automática métricas agregadas personalizadas de CloudWatch que se visualizan en paneles.

Cada evento crea puntos de datos de métricas como métricas personalizadas de CloudWatch para un conjunto seleccionado de dimensiones de métricas plenamente configurable. La publicación de métricas agregadas de Prometheus como estadísticas de métricas personalizadas de CloudWatch reduce la cantidad de métricas necesarias para monitorear, generar alarmas y solucionar problemas y errores de rendimiento. También puede analizar las métricas de alta fidelidad de Prometheus con el lenguaje de consulta de CloudWatch Logs Insights para aislar pods y etiquetas específicos que afectan al estado y al rendimiento de sus entornos en contenedores.

AWS CloudTrail ofrece esta visibilidad y registra todas las llamadas a la API en todos los servicios. AWS Config ofrece la capacidad de validación del cumplimiento. AWS ofrece a los clientes opciones de supervisión adicionales de las métricas, los registros, los eventos para la aplicación, la infraestructura y las canalizaciones; para ello, usa varios servicios, como AWS X-Ray y AWS CloudTrail.

- AWS puede integrar de forma nativa herramientas de métricas de código abierto, como Prometheus y Fluentd, entre otras.
- Las métricas de Prometheus se pueden incorporar además en Amazon CloudWatch o en OpenSearch Service para realizar un análisis más detallado.
- AWS usa FluentD como mecanismo estándar para la recopilación de registros de varios sistemas. Ese mismo mecanismo se usa y se configura para este proyecto.

Para obtener más detalles sobre cómo configurar este mecanismo, consulte Configure FluentD como DaemonSet para que envíe registros a CloudWatch Logs.



Ejemplo de métricas monitorizadas de Amazon CloudWatch

Orquestación de CI/CD con herramientas de terceros y de código abierto

La capa de orquestación usa la infraestructura como código (IaC) para implementar y configurar la infraestructura subyacente requerida para ejecutar las funciones de red 5G. Esta capa debe diseñarse de forma que sea modular, portable y reutilizable.

La infraestructura sigue las prácticas recomendadas nativas en la nube de alta disponibilidad, redundancia y escalabilidad.

Como se demostró en las secciones anteriores, la implementación de la infraestructura subyacente se puede lograr con el kit de desarrollo en la nube de AWS. Para conseguirlo, use Terraform de Hashicorp.

Terraform

Terraform es una herramienta de software IaC de código abierto que proporciona un flujo de trabajo de interfaz de línea de comandos (CLI) coherente para administrar cientos de servicios en la nube. Terraform codifica las API de la nube en archivos de configuración declarativos.

Para realizar una implementación con Terraform, aplique los mismos principios que se usan en CDK. El código se estructura en módulos que permiten que los componentes de red se personalicen y se reutilicen en función de los requisitos del proveedor.

La configuración está totalmente parametrizada, lo que permite que las implementaciones se adapten por completo en función de las recomendaciones de los proveedores y los ISV.

La implementación de funciones de red se divide en dos fases:

- La infraestructura de AWS requerida se crea y se administra a través de un repositorio central.
- La configuración y el código se almacenan de forma centralizada en un repositorio GitHub.

Una vez creados los requisitos previos, la función de red estará lista para implementarse mediante una canalización de aplicaciones que se estableció en la etapa anterior.

Implementación de la infraestructura

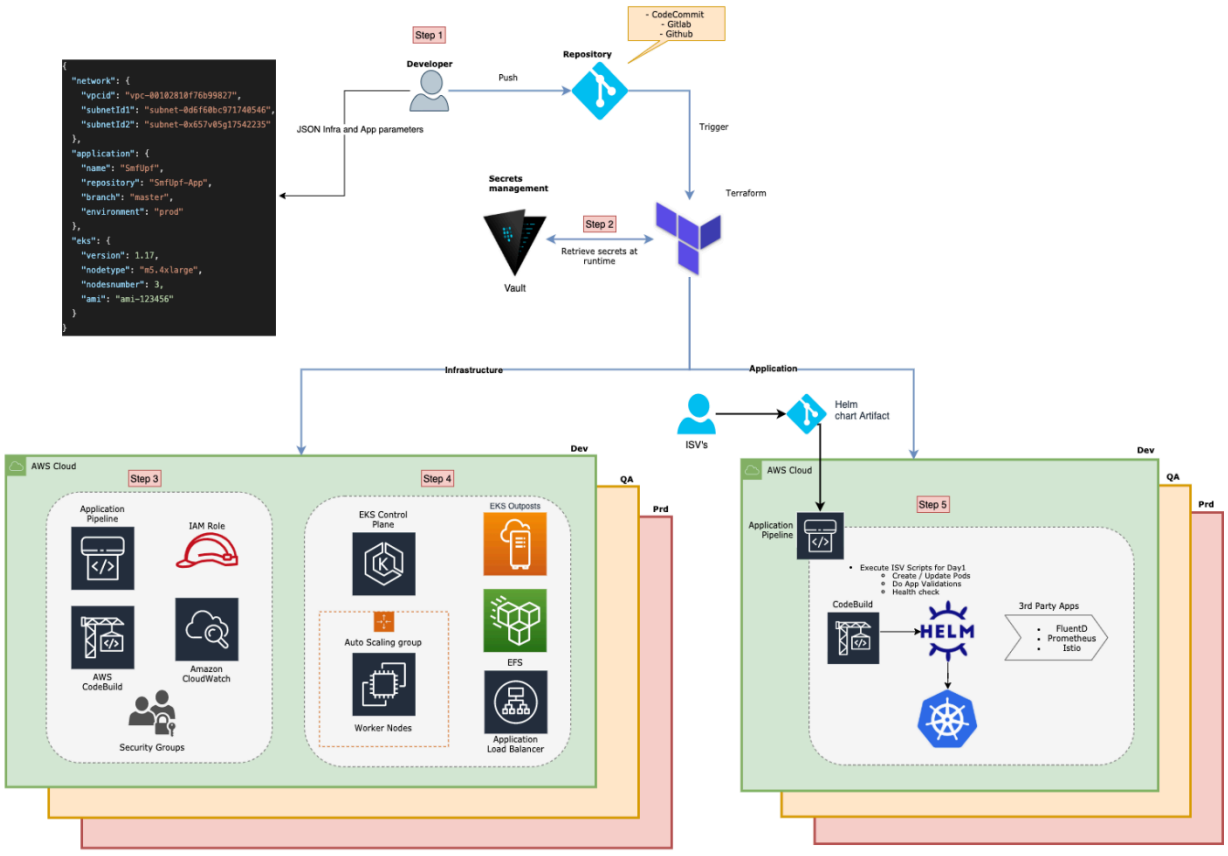
La implementación de la infraestructura incluye todos los requisitos previos para que la función de red se implemente y se configure correctamente.

Algunos de los componentes que se crean como parte de esta fase son los siguientes:

- Redes: VPC, subredes públicas y privadas, rutas y equilibradores de carga
- Computación: Kubernetes (Vmware Tanzu, Amazon EKS o AWS Outposts), nodos principales y de trabajo de las instancias de Amazon EC2, grupo de Auto Scaling
- Almacenamiento: Amazon EFS, Amazon EBS, bucket de Amazon S3
- Seguridad: roles de IAM, grupos de seguridad
- Canalización: CodePipeline, CodeBuild
- Observabilidad: CloudWatch, Prometheus, FluentD

A continuación, se describe la secuencia de la infraestructura orquestada por Terraform que se explica en la ilustración siguiente:

1. Un desarrollador rellena un archivo JSON que se almacena en un repositorio central con el código IaC. El archivo contiene información sobre la configuración de infraestructura deseada, como el tamaño de las instancias, la versión de Kubernetes, la información de red y los detalles del repositorio de aplicaciones.
2. Se recuperan secretos de HashiCorp Vault o AWS Secrets Manager en tiempo de ejecución.
3. Se implementan y configuran los componentes de la infraestructura (redes, computación, almacenamiento y seguridad).
4. Se implementa un clúster de Amazon EKS con nodos de trabajo que alojan los pods de funciones de red. Amazon EKS también puede implementarse en AWS Outposts para admitir cargas de trabajo que requieran proximidad con un centro de datos.
5. Se crea y se configura una canalización de aplicaciones para escuchar los cambios en el repositorio de funciones de red. Cada vez que se inserte código en la rama del repositorio que se haya configurado, la canalización desencadena de forma automática la creación, la prueba y la implementación de la función de red.
6. Las herramientas de observabilidad que recopilan y centralizan los registros y las métricas se implementan como servicios en todos los nodos y proporcionan datos prácticamente en tiempo real que se pueden visualizar en paneles de Grafana o de OpenSearch.



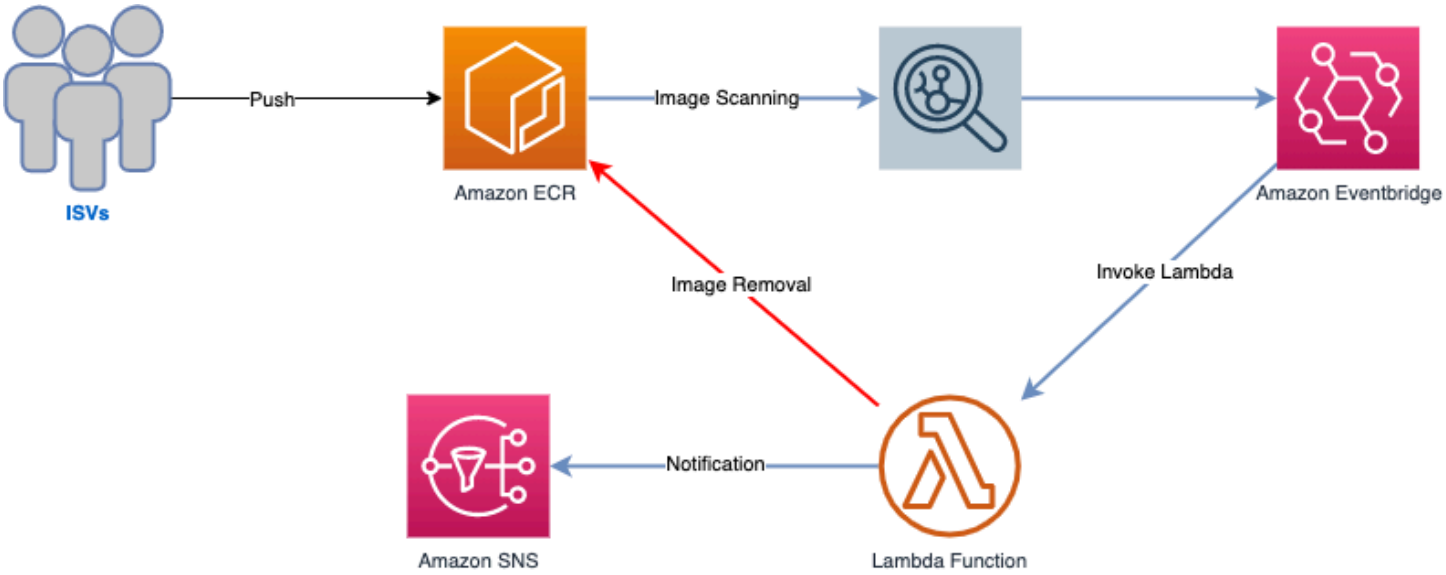
Implementación y configuración de funciones de red

Implementación y configuración de funciones de red

La canalización creada en la fase anterior permite que tanto los ISV como los proveedores descentralicen y optimicen la implementación de las funciones de red. La canalización está conectada y escucha los cambios del repositorio de aplicaciones, que se configuró en el archivo JSON, en el paso 1 de la ilustración anterior.

Para examinar las imágenes publicadas por terceros, se implementa y se configura una solución de análisis de vulnerabilidades que ayuda a identificar vulnerabilidades del software en las imágenes de contenedores. La solución de análisis comprueba automáticamente todas las imágenes nuevas insertadas en Amazon ECR. Para obtener más información sobre el análisis de imágenes para ECR, consulte Escaneo de imágenes.

La ilustración siguiente muestra la arquitectura de la solución de análisis de vulnerabilidades de imágenes.



Arquitectura de la solución de análisis de vulnerabilidades de imágenes

La canalización de aplicaciones puede configurarse de forma que se desencadene al haber cambios en la imagen después del resultado del análisis o cambios directos en el repositorio. Por ejemplo, cuando se cree una imagen de Helm.

La lista que se muestra a continuación es la secuencia que crea o actualiza la función de red, tal y como se representa en la ilustración siguiente:

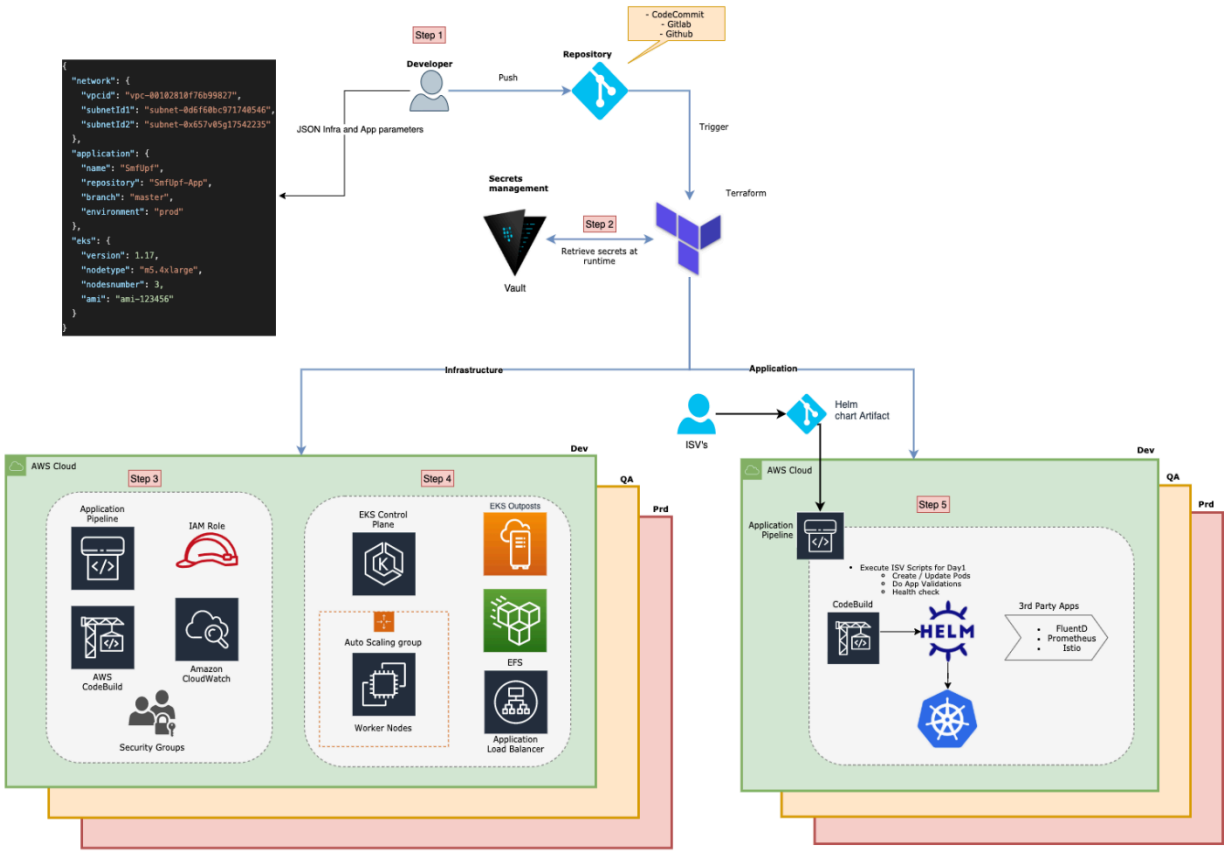
Los ISV publican nuevas imágenes en Amazon ECR. (Si la imagen se aprueba, se desencadena la canalización de aplicaciones).

CodePipeline extrae la nueva imagen de Amazon ECR y usa CodeBuild para implementar la imagen en Kubernetes. Se pueden usar comandos de Helm para actualizar la función de red.

Después de implementar la imagen, se desencadena la prueba como servicio (TaS). La prueba como servicio valida la nueva implementación y centraliza los datos y las métricas sobre el rendimiento de las funciones de red en situaciones de estrés.

Los registros y las métricas se recopilan y se centralizan en OpenSearch y Grafana. También pueden configurarse soluciones de terceros como Datadog, Istio y Prometheus para proporcionar mayor observabilidad.

Además, puede implementarse e integrarse en la solución un marco de administración y orquestación (MANO) para coordinar los recursos de red. Este usa los datos recopilados para llevar a cabo acciones automatizadas, como la segmentación de la red y el escalado automático de la calidad del servicio (QoS).



Canalización de aplicaciones

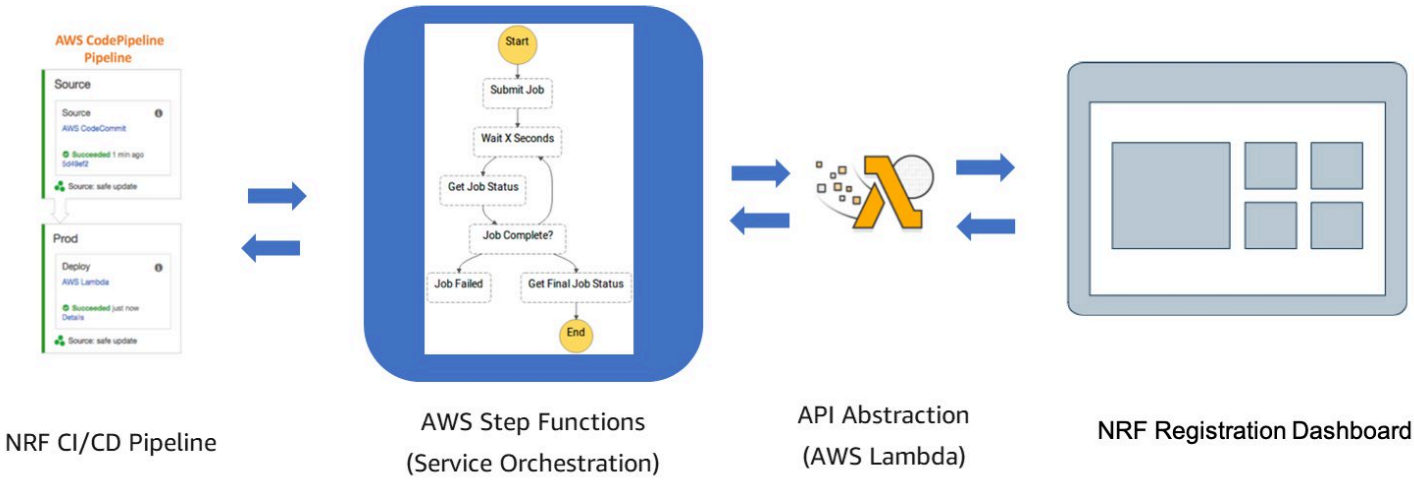
Pruebas

El marco de automatización de pruebas específico de las telecomunicaciones puede integrarse en la canalización de código. La canalización de código se integra en Step Functions para orquestar la integración con un marco de automatización de pruebas. AWS Step Functions usa varias funciones Lambda para invocar el marco de automatización de pruebas (herramientas de terceros) a través de llamadas a la API. En principio, la instancia de Step Functions obtiene el ID de prueba, ejecuta la prueba y obtiene los resultados del marco de automatización de pruebas. A continuación, analiza los resultados y los pasa a la canalización de código, a fin de aprobarse la aplicación para la implementación de producción. La aprobación puede automatizarse o mantenerse de forma manual en la canalización de código, según sea necesario. Este es un paso importante para que los CSP promuevan la implementación desde el entorno de prueba al de producción. Las API de alto nivel necesarias para la integración se clasifican de la manera siguiente:

- Obtención de contexto

- Ejecución de un caso de prueba específico
- Detención de un caso de prueba
- Obtención de resultados

La complejidad de llamar a API REST externas se modela mediante AWS Step Functions, que permite que las construcciones estándar invoquen flujos paralelos, esperen los resultados, se ramifiquen en función de las condiciones e integren la API REST con AWS CodePipeline.



Flujo de pruebas

CI/CD y orquestación

La integración y entrega continuas forman parte de una filosofía de automatización global que incluye la arquitectura nativa en la nube y la forma en que se aplica a 5G. La orquestación es otra faceta de esta filosofía de la que se espera que sea dinámica y reactiva a cualquier cambio que ocurra en la red. Las funciones de orquestación y CI/CD deben estar estrechamente acopladas para garantizar un servicio correcto y minimizar las interrupciones de este. Además, se espera que la integración entre CI/CD y la orquestación tenga lugar en dos frentes:

- La aplicación de revisiones y actualizaciones en el sistema debe administrarse y orquestarse de forma que se minimice la interrupción de cualquier servicio activo. Por ejemplo, la orquestación puede determinar de forma dinámica el mejor momento para implementar una actualización.
- La orquestación que tiene en cuenta los procesos de CI/CD permite desviar el tráfico durante la implementación de actualizaciones en función de la estrategia de modelo de implementación adoptada (valores controlados, lineal o todo a la vez).

Normalmente, las soluciones de orquestación se ejecutan sobre las canalizaciones de CI/CD para permitir que la orquestación introduzca fases de gobernanza en esas canalizaciones y quede expuesta en los ciclos de actualización en curso.

Conclusión

CI/CD proporciona una ruta clara y eficaz que permite a los desarrolladores y a los equipos de aplicaciones implementar código de aplicación nuevo en cuestión de minutos. AWS cuenta con una amplia gama de herramientas que pueden ayudar a los desarrolladores en la integración, las pruebas y la implementación de código nuevo, entre las que se incluyen AWS CodePipeline, AWS CodeCommit, AWS CodeBuild, AWS CodeDeploy y muchas más. En este documento se ha analizado cómo pueden utilizarse los servicios de AWS a fin de crear un proceso de CI/CD para implementar una función de red 5G de forma totalmente automatizada, incluidos los distintos pasos necesarios para completar la implementación de nuevo código. También se ha analizado cómo integrar un marco de automatización de pruebas de terceros en el proceso de CI/CD, además de cómo usar herramientas de terceros como Terraform.

Colaboradores

Entre los colaboradores de este documento, están las siguientes personas:

- Hisham Elshaer, consultor sénior, AWS Telecom, Amazon Web Services
- Vara Prasad Talari, consultor principal, AWS Telecom, Amazon Web Services
- Rabi Abdel, consultor principal, AWS Telecom, Amazon Web Services
- Franco Bontorin, consultor sénior, Entrega compartida, Amazon Web Services
- Pragtideep Singh, consultor, Entrega compartida, Amazon Web Services
- Subbarao Duggisetty, arquitecto de infraestructuras en la nube, Cuentas globales, Amazon Web Services
- Young Jung, arquitecto sénior de soluciones de socios, AWS Telecom, Amazon Web Services

Revisiones del documento

Para recibir notificaciones sobre las actualizaciones de este documento técnico, suscríbase a la fuente RSS.

update-history-change	update-history-description	update-history-date
Publicación inicial	Documento técnico publicado por primera vez	8 de marzo de 2021

Documentación adicional

Para obtener información adicional, consulte la siguiente documentación:

- Práctica de integración y entrega continuas en AWS (documento técnico)
- Red principal de paquetes móviles de grado empresarial para operadores de telefonía en AWS (documento técnico)
- Evolución de la red 5G con AWS (documento técnico)

Siglas

- AMF: función de administración de acceso y movilidad
- API: interfaz de programación de aplicaciones
- AUSF: función del servidor de autenticación
- BSS: sistema de soporte empresarial
- CDK: kit de desarrollo en la nube
- CI/CD: integración continua y entrega continua
- CLI: interfaz de línea de comandos
- CNF: función de red nativa en la nube o en contenedores
- CSP: proveedor de servicios de comunicación
- CU: unidad central de RAN
- CVE: vulnerabilidades y exposiciones comunes
- DoS: denegación del servicio
- DR: recuperación de desastres
- DU: unidad distribuida de RAN
- E2E: integral
- ECR: Elastic Container Registry
- EFS: Elastic File System
- EKS: Elastic Kubernetes Service
- EPC: núcleo de paquetes evolucionado
- IaC: infraestructura como código
- ISV: proveedor de software independiente
- MANO: administración y orquestación
- MEC: computación de borde de acceso múltiple
- NACL: lista de control de acceso de red
- NAT: traducción de direcciones de red
- NF: función de red
- NFV: virtualización de funciones de red
- NFVO: orquestador de virtualización de funciones de red

- NOC: centro de operaciones de red
- NRF: función de repositorio de red
- OSS: sistema de soporte de operaciones
- PII: información de identificación personal
- QoS: calidad del servicio
- RAN: red de acceso por radio
- SBI: interfaz basada en servicios
- SMF: función de administración de sesiones
- SSL: capa de sockets seguros
- TaS: prueba como servicio
- TCP: protocolo de control de transmisión
- TLS: seguridad de la capa de transporte
- UDM: administración de datos unificada
- UDP: protocolo de datagramas de usuario
- UPF: función de plano de usuario
- VIM: administrador de infraestructura virtualizada
- VNF: función de red virtual
- VPC: nube virtual privada

Avisos

Los clientes son responsables de realizar sus propias evaluaciones de la información contenida en este documento. Este documento: (a) solo tiene fines informativos, (b) representa las prácticas y las ofertas de productos vigentes de AWS, que están sujetas a cambios sin previo aviso, y (c) no crea ningún compromiso ni garantía de AWS y sus empresas afiliadas, proveedores o concesionarios de licencias. Los productos o servicios de AWS se proporcionan “tal cual”, sin garantías, representaciones ni condiciones de ningún tipo, ya sean explícitas o implícitas. Las responsabilidades y obligaciones de AWS en relación con sus clientes se rigen por los acuerdos de AWS, y este documento no modifica ni forma parte de ningún acuerdo entre AWS y sus clientes.