Plano de control de IA de NetApp

NetApp Solutions

NetApp
April 18, 2024

This PDF was generated from https://docs.netapp.com/es-es/netapp-solutions/ai/aicp_introduction.html on April 18, 2024. Always check docs.netapp.com for the latest.

Tabla de contenidos

Plano de control de IA de NetApp . . . 1
TR-4798: Plano de control de IA de NetApp . . . 1
Conceptos y componentes . . . 2
Requisitos de hardware y software . . . 11
Puesta en marcha de Kubernetes . . . 12
Implementación y configuración de Trident de NetApp . . . 13
Despliegue de Kubeflow . . . 21
Ejemplo de operaciones y tareas de Kubeflow . . . 30
Distribución del flujo de aire de Apache . . . 38
Ejemplo de flujos de trabajo de flujo de aire de Apache . . . 42
Ejemplo de operaciones de Trident . . . 42
Ejemplo de tareas de alto rendimiento para puestas en marcha de ONTAP AI . . . 46
Pruebas de rendimiento . . . 58
Conclusión . . . 58

Plano de control de IA de NetApp

TR-4798: Plano de control de IA de NetApp

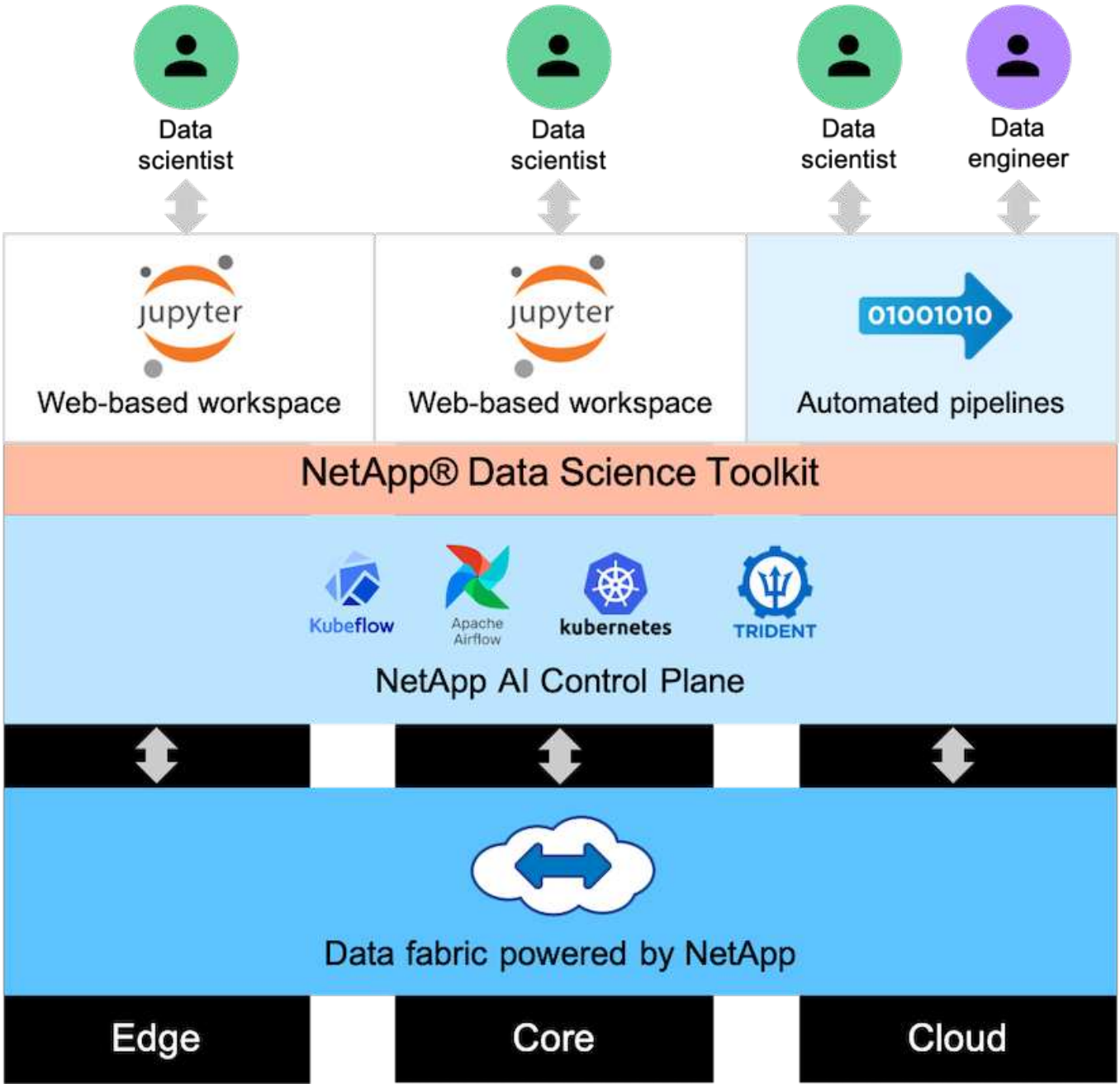
Mike Oglesby, NetApp

Empresas y organizaciones de todos los tamaños y sectores se están decantando por la inteligencia artificial (IA), el aprendizaje automático (ML) y el aprendizaje profundo (DL) para resolver problemas reales, ofrecer productos y servicios innovadores y obtener una ventaja en un mercado cada vez más competitivo. A medida que las organizaciones aumentan el uso de la IA, EL ML y el AP, deben hacer frente a numerosos retos, como la escalabilidad de la carga de trabajo y la disponibilidad de los datos. En este documento se demuestra cómo puede hacer frente a estos retos usando el plano de control de IA de NetApp, una solución que empareja las funcionalidades de gestión de datos de NetApp con marcos y herramientas de código abierto más populares.

Este informe le muestra cómo clonar rápidamente un espacio de nombres de datos. También le muestra cómo replicar datos sin problemas en sitios y regiones para crear una canalización de datos coherente y unificada de IA/ML/DL. Además, le guía por la definición e implementación de flujos de trabajo de formación de IA, ML y DL que incorporan la creación casi instantánea de datos y líneas de base de modelos para su trazabilidad y versionado. Con esta solución, puede rastrear cada entrenamiento de modelo que se ejecuta en el conjunto de datos exacto que se utilizó para entrenar o validar el modelo. Por último, este documento le muestra cómo aprovisionar rápidamente espacios de trabajo de Juppyter Notebook con acceso a conjuntos de datos masivos.

Nota: Para entrenamiento distribuido al estilo HPC a escala que implique un gran número de servidores GPU que requieran acceso compartido al mismo conjunto de datos o, si necesita o prefiere un sistema de archivos paralelo, consulte el documento "TR-4890". En este informe técnico se describe cómo incluir "Solución de sistema de archivos en paralelo totalmente compatible de NetApp BeeGFS" Como parte del plano de control de IA de NetApp. Esta solución está diseñada para escalar a partir de un puñado de sistemas NVIDIA DGX A100, hasta un SuperPOD de 140 nodos completo.

El plano de control de IA de NetApp se dirige a los científicos e ingenieros de datos y, por lo tanto, se necesita experiencia mínima en NetApp o en ONTAP® de NetApp. Con esta solución, las funciones de gestión de datos se pueden ejecutar utilizando interfaces y herramientas sencillas y conocidas. Si ya dispone de almacenamiento de NetApp en su entorno, puede probar el plano de control de IA de NetApp hoy mismo. Si desea probar la solución pero no dispone de almacenamiento de NetApp, visite "cloud.netapp.com"Y puede volver a tener una solución de almacenamiento de NetApp basada en cloud en cuestión de minutos. La siguiente figura ofrece una visualización de la solución.



Conceptos y componentes

Inteligencia artificial

La IA es una disciplina informática en la que las computadoras están entrenadas para imitar las funciones cognitivas de la mente humana. Los desarrolladores de IA entrenan computadoras para aprender y resolver problemas de una manera similar, o incluso superior a, humanos. El aprendizaje profundo y el aprendizaje automático son subcampos de la IA. Las organizaciones adoptan cada vez más IA, ML y DL para dar soporte a sus necesidades empresariales cruciales. Algunos ejemplos son los siguientes:

- Analizar grandes cantidades de datos para desconocer información empresarial anteriormente desconocida
- Interacción directa con los clientes mediante el procesamiento de lenguaje natural

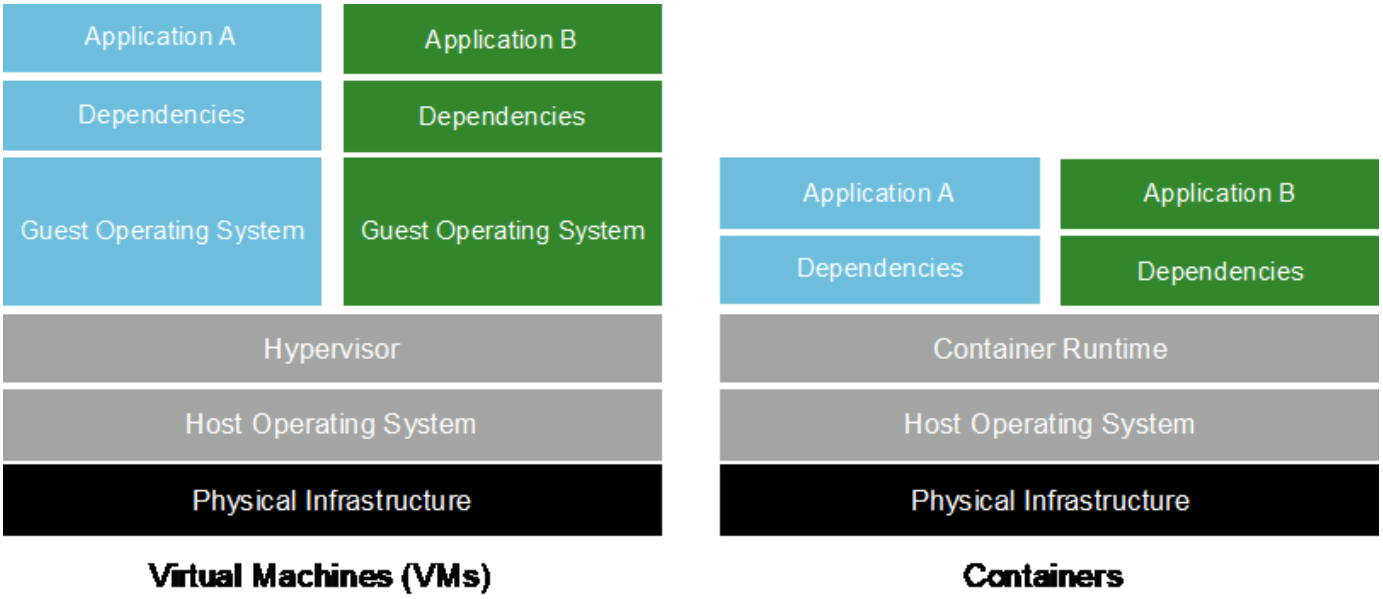
- Automatización de diversos procesos y funciones empresariales

La formación de IA moderna y las cargas de trabajo de inferencia requieren de funcionalidades de computación en paralelo masivas. Por lo tanto, se están utilizando cada vez más GPU para ejecutar operaciones de IA, ya que las capacidades de procesamiento paralelo de las GPU son muy superiores a las de las CPU de uso general.

Contenedores

Los contenedores son instancias aisladas del espacio de usuario que se ejecutan sobre un kernel de sistema operativo host compartido. La adopción de contenedores aumenta rápidamente. Los contenedores ofrecen muchos de los mismos beneficios de uso de pruebas de espacio que las máquinas virtuales (VM). Sin embargo, debido a que se eliminan las capas de hipervisor y de sistema operativo «guest» de las que dependen las máquinas virtuales, los contenedores son mucho más ligeros. En la siguiente figura, se muestra una visualización de las máquinas virtuales en comparación con los contenedores.

Los contenedores también permiten el paquete eficiente de dependencias de aplicaciones, tiempos de ejecución, etc., directamente con una aplicación. El formato de embalaje de contenedor más utilizado es el contenedor Docker. Una aplicación que se haya contenedor en el formato de contenedor Docker se puede ejecutar en cualquier máquina que pueda ejecutar contenedores Docker. Esto es cierto incluso si las dependencias de la aplicación no están presentes en la máquina porque todas las dependencias están empaquetadas en el propio contenedor. Para obtener más información, visite la "Sitio web de Docker".



Kubernetes

Kubernetes es una plataforma de orquestación de contenedores distribuida de código abierto que originalmente diseñada por Google y que ahora se mantiene mediante Cloud Native Computing Foundation (CNCF). Kubernetes permite automatizar las funciones de puesta en marcha, gestión y escalado para aplicaciones en contenedores. En los últimos años, Kubernetes se ha convertido en la plataforma de orquestación de contenedores dominante. Aunque son compatibles con otros formatos de empaquetado y tiempos de ejecución, Kubernetes se utiliza con más frecuencia como un sistema de orquestación para contenedores Docker. Para obtener más información, visite la "Sitio web de Kubernetes".

Trident de NetApp

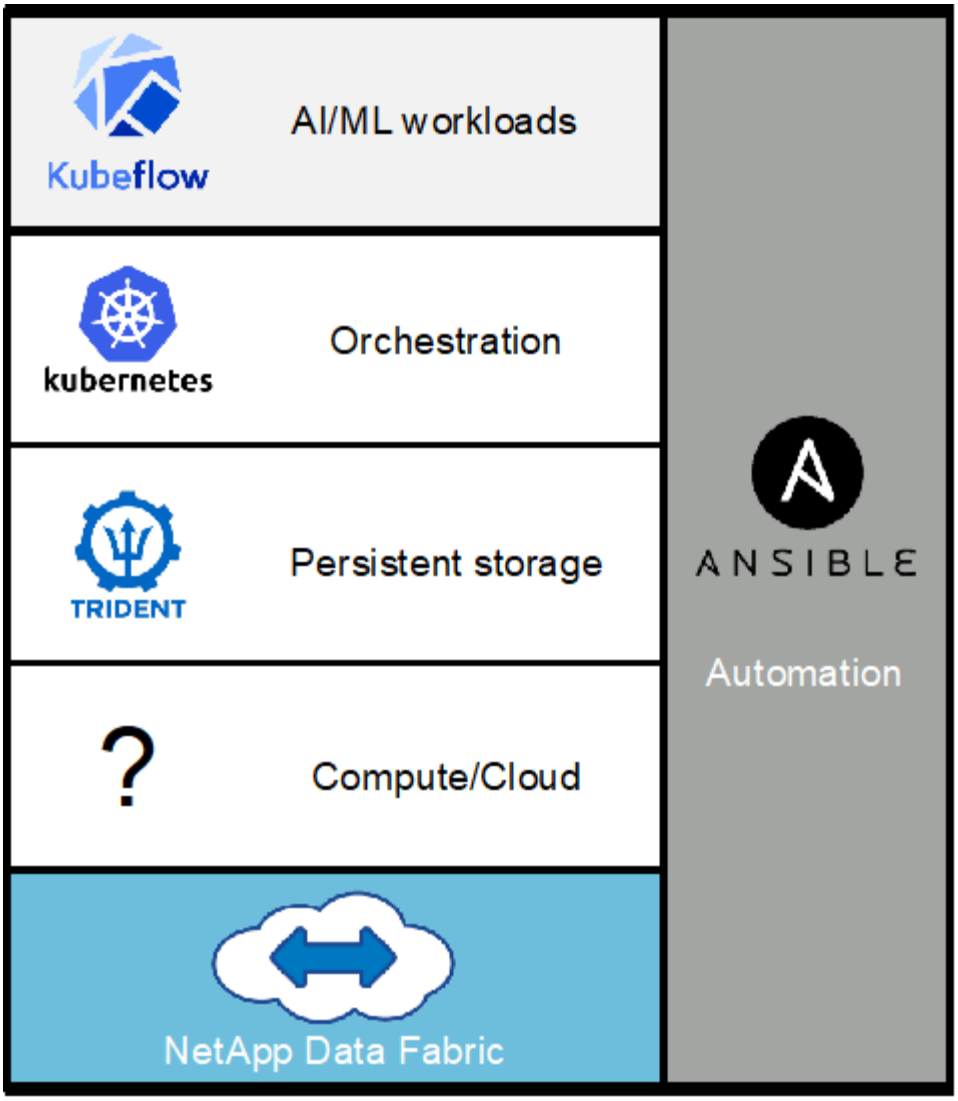
Trident es un orquestador de almacenamiento de código abierto desarrollado y mantenido por NetApp que simplifica en gran medida la creación, la gestión y el consumo de almacenamiento persistente para cargas de trabajo de Kubernetes. Trident, en sí misma una aplicación nativa de Kubernetes, se ejecuta directamente en un clúster de Kubernetes. Con Trident, los usuarios de Kubernetes (desarrolladores, científicos de datos, administradores de Kubernetes, etc.) pueden crear, gestionar e interactuar con volúmenes de almacenamiento persistente en el formato Kubernetes estándar, con el que ya están familiarizados. Al mismo tiempo, pueden aprovechar las funciones avanzadas de gestión de datos de NetApp y un Data Fabric con tecnología de NetApp. Trident elimina las complejidades del almacenamiento persistente y facilita el consumo. Para obtener más información, visite la "Sitio web de Trident".

DeepOps de NVIDIA

DeepOps es un proyecto de código abierto de NVIDIA que, con Ansible, automatiza la puesta en marcha de clústeres de servidores de GPU de acuerdo con las prácticas recomendadas. DeepOps es modular y se puede utilizar para realizar varias tareas de puesta en marcha. En este documento y en el ejercicio de validación descrito, DeepOps se utiliza para poner en marcha un clúster de Kubernetes que consta de nodos de trabajo de servidor GPU. Para obtener más información, visite la "Sitio web DeepOps".

Kubeflow

Kubeflow es un kit de herramientas DE IA Y ML de código abierto para Kubernetes que fue desarrollado originalmente por Google. El proyecto Kubeflow hace que la puesta en marcha de flujos de trabajo de IA y ML en Kubernetes sea sencilla, portátil y escalable. —Kubeflow elimina las complejidades de Kubernetes, lo que permite a los científicos de datos centrarse en lo que saben más ciencia de datos. Consulte la siguiente figura para ver una visualización. Kubeflow ha ido ganando terreno a medida que los departamentos DE TI de las empresas se han estandarizado cada vez más en Kubernetes. Para obtener más información, visite la "Sitio web de Kubeflow".



Tuberías de Kubeflow

Los oleoductos de Kubeflow son un componente clave de Kubeflow. Las canalizaciones de Kubeflow son una plataforma y un estándar para definir y poner en marcha flujos de trabajo DE IA Y ML escalables y portátiles. Para obtener más información, consulte "Documentación oficial de Kubeflow".

Servidor de portátiles Juppyter

Un servidor Juppyter Notebook es una aplicación web de código abierto que permite a los científicos de datos crear documentos similares a wiki llamados portátiles Juppyter que contienen código en vivo así como pruebas descriptivas. Los portátiles Juppyter se utilizan ampliamente en la comunidad de AI Y ML como medio para documentar, almacenar y compartir proyectos de IA y ML. Kubeflow simplifica el aprovisionamiento y la puesta en marcha de servidores para portátiles Juppyter en Kubernetes. Para obtener más información sobre los Cuadernos Jupyter, visite "Sitio Web de Juppyter". Para obtener más información acerca de Jupyter Notebooks en el contexto de Kubeflow, consulte "Documentación oficial de Kubeflow".

Flujo de aire Apache

Apache Airflow es una plataforma de gestión de flujos de trabajo de código abierto que permite la creación, programación y supervisión de programas para flujos de trabajo empresariales complejos. A menudo se utiliza para automatizar los flujos de trabajo de ETL y de canalización de datos, pero estos tipos de flujos de trabajo

no se limitan a ellos. El proyecto de flujo de aire fue iniciado por Airbnb, pero desde entonces se ha vuelto muy popular en la industria y ahora está bajo los auspicios de la Apache Software Foundation. El flujo de aire se escribe en Python, los flujos de trabajo del flujo de aire se crean a través de scripts Python y el flujo de aire está diseñado según el principio de "configuración como código". Muchos usuarios de flujo de aire empresarial ahora ejecutan el flujo de aire sobre Kubernetes.

Gráficos de Acíclicos dirigidos (DAG)

En el flujo de aire, los flujos de trabajo se denominan gráficos Acíclicos dirigidos (DAG). Los DAG se componen de tareas que se ejecutan en secuencia, en paralelo o en una combinación de las dos, dependiendo de la definición DAG. El programador de flujo de aire ejecuta tareas individuales en una matriz de trabajadores y cumple con las dependencias a nivel de tarea especificadas en la definición DAG. Los DAG se definen y crean a través de scripts Python.

ONTAP 9 de NetApp

ONTAP 9 de NetApp es la última generación del software de gestión del almacenamiento de NetApp y permite a empresas como la suya modernizar su infraestructura y realizar la transición a un centro de datos preparado para el cloud. Gracias a las capacidades de gestión de datos líderes del sector, ONTAP le permite gestionar y proteger sus datos con un solo conjunto de herramientas sin importar dónde residan. También puede mover los datos libremente a donde necesite: El extremo, el núcleo o el cloud. ONTAP 9 incluye numerosas funciones que simplifican la gestión de datos, aceleran y protegen sus datos esenciales y preparan su infraestructura para el futuro con arquitecturas de cloud híbrido.

Simplifique la gestión de los datos

La gestión de datos es vital para las operaciones TECNOLÓGICAS de su empresa, de modo que pueda usar los recursos adecuados para sus aplicaciones y conjuntos de datos. ONTAP incluye las siguientes funciones para facilitar y simplificar las operaciones, y reducir el coste total de funcionamiento:

- **Compactación de datos inline y deduplicación expandida.** la compactación de datos reduce el espacio perdido dentro de los bloques de almacenamiento, y la deduplicación aumenta significativamente la capacidad efectiva.
- **Calidad de servicio (QoS) mínima, máxima y adaptativa.** los controles granulares de QoS ayudan a mantener los niveles de rendimiento para aplicaciones críticas en entornos altamente compartidos.
- **ONTAP FabricPool.** esta función ofrece una organización automática en niveles de datos inactivos para opciones de almacenamiento en cloud público y privado, incluidos Amazon Web Services (AWS), Azure y almacenamiento basado en objetos StorageGRID de NetApp.

Acelere y proteja sus datos

ONTAP no solo ofrece niveles de rendimiento y protección de datos superiores, sino que amplía estas funcionalidades con las siguientes funciones:

- **Alto rendimiento y baja latencia.** ONTAP ofrece el rendimiento más alto posible con la menor latencia posible.
- **La tecnología ONTAP FlexGroup de NetApp.** Un volumen FlexGroup es un contenedor de datos de alto rendimiento que se puede escalar linealmente hasta 20 PB y 400 000 millones de archivos, lo que proporciona un espacio de nombres único que simplifica la gestión de datos.
- **Protección de datos.** ONTAP ofrece capacidades integradas de protección de datos con administración común en todas las plataformas.
- **Cifrado de volumen de NetApp.** ONTAP ofrece cifrado nativo a nivel de volumen con compatibilidad para

la gestión de claves incorporada y externa.

Infraestructura preparada para futuros retos

ONTAP 9 le ayuda a satisfacer las exigentes y siempre cambiantes necesidades de su empresa:

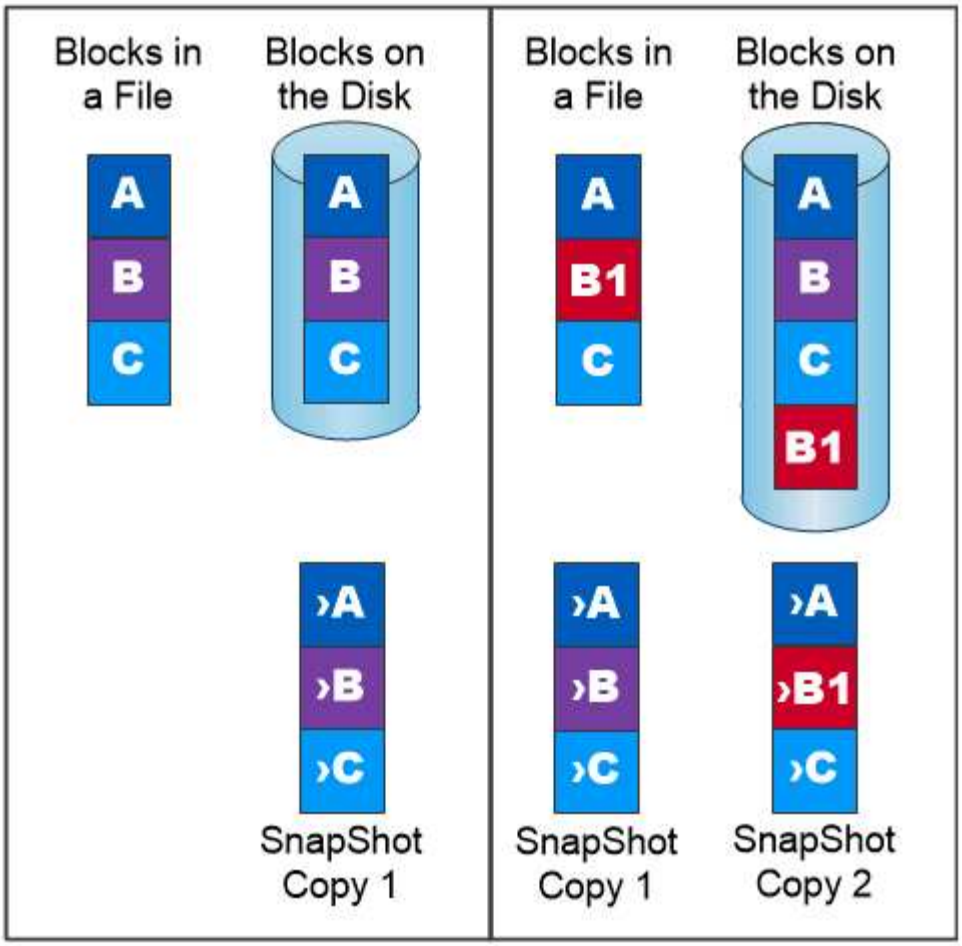
- **Escalado sencillo y operaciones no disruptivas.** ONTAP admite la adición sin interrupciones de capacidad a las controladoras existentes y a los clústeres de escalado horizontal. Puede empezar a utilizar tecnologías punteras como NVMe y FC 32 GB, sin necesidad de realizar costosas migraciones de datos y sin cortes.
- **Conexión al cloud.** ONTAP es uno de los programas de gestión del almacenamiento con mejor conexión al cloud e incluye opciones de almacenamiento definido por software (ONTAP Select) e instancias nativas del cloud (Cloud Volumes Service de NetApp) en todos los clouds públicos.
- **Integración con aplicaciones emergentes.** al utilizar la misma infraestructura que soporta las aplicaciones empresariales existentes, ONTAP ofrece servicios de datos de clase empresarial para plataformas y aplicaciones de próxima generación como OpenStack, Hadoop y MongoDB.

Copias Snapshot de NetApp

Una copia Snapshot de NetApp es una imagen puntual de solo lectura de un volumen. La imagen consume un espacio de almacenamiento mínimo y tiene una sobrecarga del rendimiento mínima, ya que solo registra los cambios que se han realizado en los archivos creados desde que se realizó la última copia Snapshot, como se muestra en la siguiente figura.

Las copias Snapshot deben su eficiencia a la tecnología de virtualización del almacenamiento central de ONTAP, el sistema de archivos de escritura en cualquier lugar (WAFL). Al igual que una base de datos, WAFL utiliza metadatos para apuntar a los bloques de datos reales en el disco. Sin embargo, a diferencia de una base de datos, WAFL no sobrescribe los bloques existentes. Escribe los datos actualizados en un bloque nuevo y cambia los metadatos. Porque ONTAP hace referencia a los metadatos cuando crea una copia Snapshot, en lugar de copiar bloques de datos, es tan eficiente que las copias Snapshot. Al hacerlo, se elimina el tiempo de búsqueda que otros sistemas incurren en la localización de los bloques a copiar, así como el costo de hacer la copia misma.

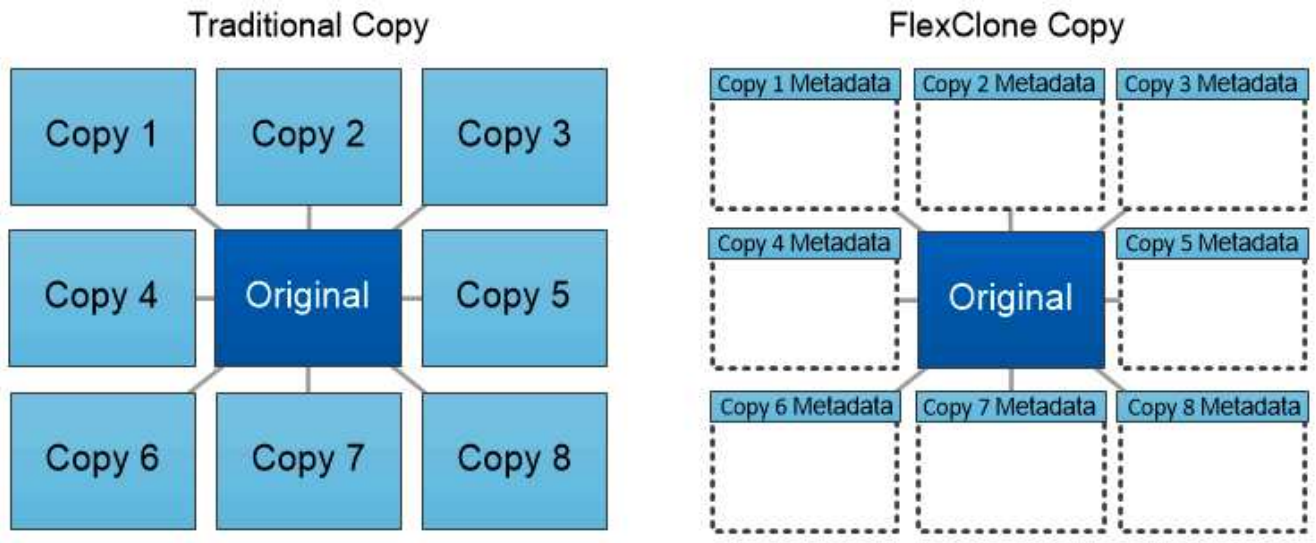
Puede utilizar una copia Snapshot para recuperar archivos o LUN individuales o para restaurar el contenido completo de un volumen. ONTAP compara la información de punteros de la copia Snapshot con los datos del disco para reconstruir el objeto faltante o dañado, sin tiempo de inactividad ni un coste de rendimiento significativo.



A Snapshot copy records only changes to the active file system since the last Snapshot copy.

Tecnología FlexClone de NetApp

La tecnología FlexClone de NetApp hace referencia a los metadatos de Snapshot para crear copias puntuales editables de un volumen. Las copias comparten bloques de datos con sus padres, sin consumir almacenamiento excepto lo que se necesita para los metadatos hasta que se escriben los cambios en la copia, como se muestra en la siguiente figura. Cuando se pueden crear copias tradicionales en minutos o incluso horas, el software FlexClone le permite copiar incluso los conjuntos de datos más grandes de forma casi instantánea. Esto lo convierte en la opción ideal para las situaciones en las que necesita varias copias de conjuntos de datos idénticos (un espacio de trabajo de desarrollo, por ejemplo) o copias temporales de un conjunto de datos (probar una aplicación contra un conjunto de datos de producción).

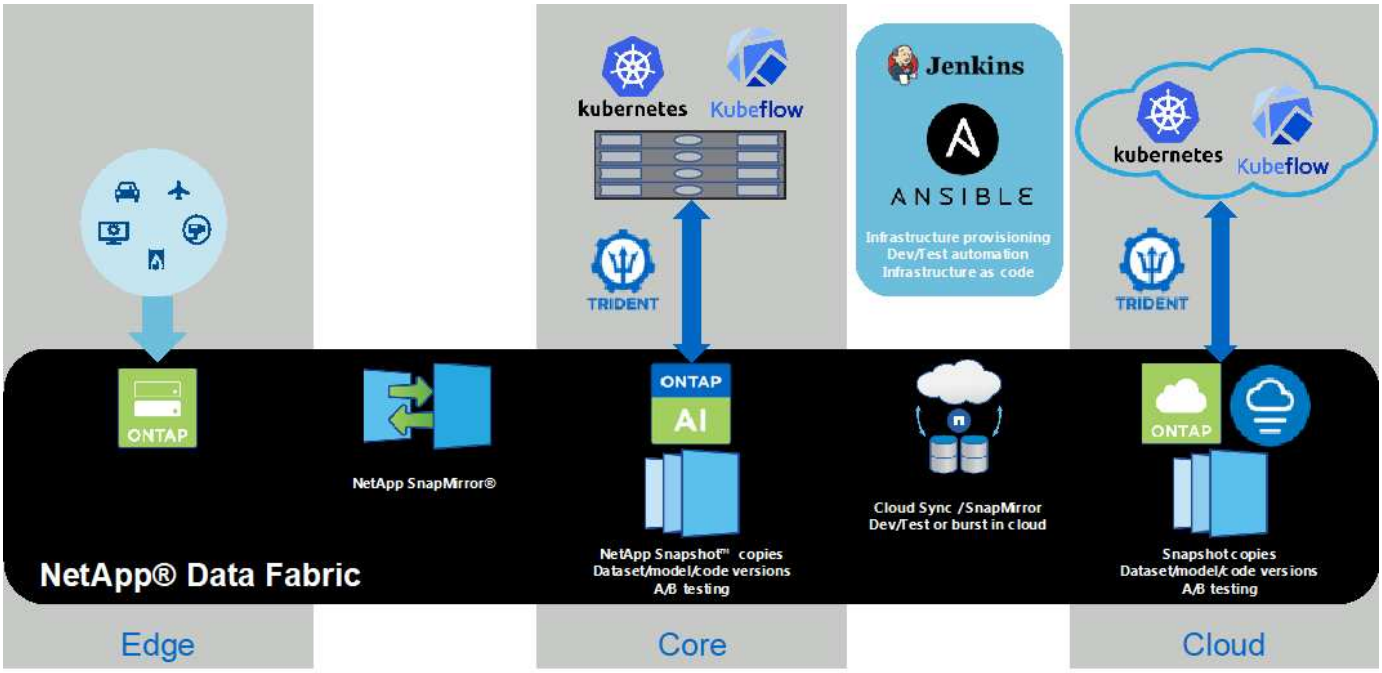


FlexClone copies share data blocks with their parents, consuming no storage except what is required for metadata.

Tecnología de replicación de datos de SnapMirror de NetApp

El software SnapMirror de NetApp es una solución de replicación unificada rentable y fácil de usar para todo Data Fabric. Replica datos a altas velocidades mediante LAN o WAN. Le proporciona una alta disponibilidad de datos y una rápida replicación de datos para todo tipo de aplicaciones, incluidas aplicaciones vitales para el negocio en entornos tanto virtuales como tradicionales. Al replicar datos en uno o varios sistemas de almacenamiento de NetApp y actualizar continuamente los datos secundarios, estos están siempre al día y disponibles cuando los necesite. No se requieren servidores de replicación externos. Consulte la figura siguiente para ver un ejemplo de una arquitectura que aprovecha la tecnología SnapMirror.

El software SnapMirror aprovecha las eficiencias del almacenamiento de ONTAP de NetApp y envía únicamente los bloques cambiados a través de la red. El software SnapMirror también usa la compresión de red incorporada para acelerar las transferencias de datos y reducir la utilización de ancho de banda hasta un 70 %. Con la tecnología SnapMirror, puede aprovechar un flujo de datos de thin replication para crear un único almacén que mantenga los reflejos activos y las copias de momentos específicos anteriores, lo que reduce el tráfico de red hasta un 50 %.



Copia y sincronización de NetApp BlueXP

La copia y sincronización de BlueXP es un servicio de NetApp que ofrece una sincronización de datos rápida y segura. Ya tenga que transferir archivos entre recursos compartidos de archivos NFS o SMB en las instalaciones, NetApp StorageGRID, NetApp ONTAP S3, NetApp Cloud Volumes Service, Azure NetApp Files, AWS S3, AWS EFS, Azure Blob, Google Cloud Storage, o IBM Cloud Object Storage, BlueXP Copy and Sync mueve los archivos a donde los necesitas de forma rápida y segura.

Una vez transferidos los datos, estarán completamente disponibles para su uso tanto en origen como en destino. BlueXP Copy and Sync puede sincronizar los datos bajo demanda al activar una actualización o sincronizar continuamente los datos en función de una programación predefinida. Independientemente de ello, BlueXP Copy y Sync solo mueve los diferenciales, por lo que se reducen al mínimo el tiempo y el dinero que se invierten en la replicación de datos.

Copia y sincronización de BlueXP es una herramienta de software como servicio (SaaS) extremadamente sencilla de configurar y usar. Las transferencias de datos activadas por BlueXP Copy and Sync se llevan a cabo por agentes de datos. Los agentes de datos de BlueXP Copy y Sync se pueden poner en marcha en AWS, Azure, Google Cloud Platform o en las instalaciones.

XCP de NetApp

XCP de NetApp es el software basado en cliente para migraciones de datos y análisis del sistema de archivos entre NetApp y NetApp. XCP se ha diseñado para escalar y lograr el máximo rendimiento utilizando todos los recursos del sistema disponibles para gestionar conjuntos de datos de gran volumen y migraciones de alto rendimiento. XCP le ayuda a obtener una visibilidad completa del sistema de archivos con la opción de generar informes.

XCP de NetApp está disponible en un único paquete compatible con los protocolos NFS y SMB. XCP incluye un binario de Linux para conjuntos de datos NFS y un ejecutable de Windows para conjuntos de datos SMB.

XCP File Analytics de NetApp es un software basado en host que detecta recursos compartidos de archivos, ejecuta análisis en el sistema de archivos y proporciona una consola para el análisis de archivos. XCP File Analytics es compatible con los sistemas NetApp y de otros proveedores, y se ejecuta en hosts Linux o

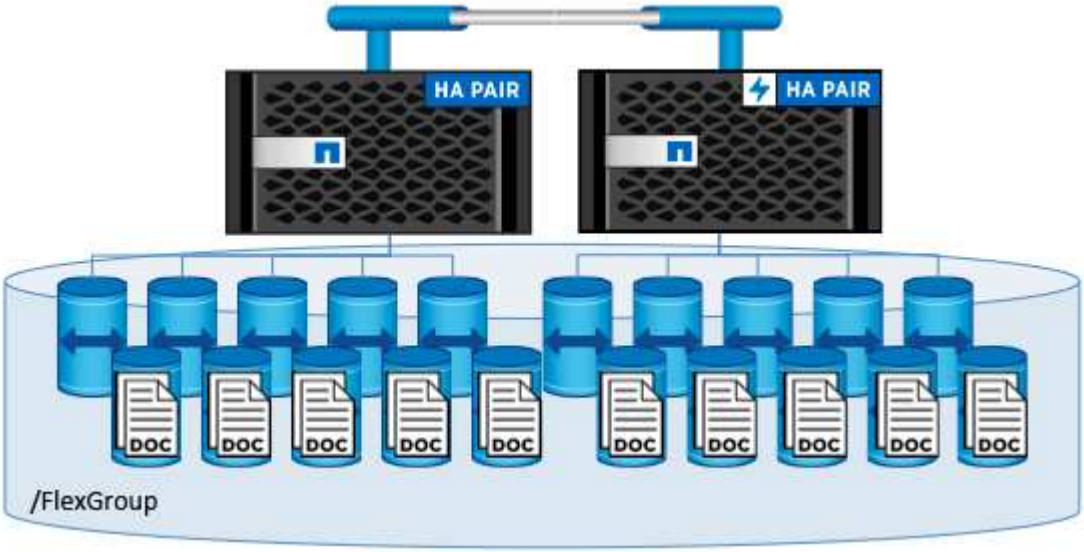
Windows para proporcionar análisis en sistemas de archivos NFS y exportados SMB.

ONTAP FlexGroup Volumes de NetApp

Un conjunto de datos de entrenamiento puede ser una colección con hasta miles de millones de archivos. Pueden ser archivos de texto, de audio, de vídeo o cualquier otra forma de datos no estructurados que deban almacenarse y procesarse para su lectura en paralelo. El sistema de almacenamiento debe almacenar un gran número de archivos pequeños y debe leerlos en paralelo, con una entrada y salida secuencial o aleatoria

Un volumen FlexGroup es un espacio de nombres único que comprende varios volúmenes miembro constituyentes, tal y como se muestra en la siguiente figura. Desde el punto de vista de un administrador de almacenamiento, un volumen FlexGroup se gestiona y actúa como un volumen FlexVol de NetApp. Los archivos de un volumen de FlexGroup se asignan a volúmenes miembro individuales y no están repartidos en volúmenes o nodos. Ofrecen las siguientes capacidades:

- Los volúmenes FlexGroup proporcionan varios petabytes de capacidad y una baja latencia predecible para cargas de trabajo con una gran cantidad de metadatos.
- Permiten un máximo de 400 000 millones de archivos en un mismo espacio de nombres.
- Admiten operaciones en paralelo para cargas de trabajo NAS entre varias CPU, nodos, agregados y volúmenes FlexVol constituyentes.



Requisitos de hardware y software

La solución de plano de control de IA de NetApp no depende de este hardware específico. La solución es compatible con cualquier dispositivo de almacenamiento físico, instancia definida por software o servicio cloud de NetApp compatible con Trident. Entre los ejemplos se incluyen un sistema de almacenamiento AFF de NetApp, Azure NetApp Files, Cloud Volumes Service de NetApp, una instancia de almacenamiento definido por software ONTAP Select de NetApp o una instancia de Cloud Volumes ONTAP de NetApp. Además, la solución puede implementarse en cualquier clúster de Kubernetes, siempre que la versión de Kubernetes utilizada sea compatible con Kubeflow y Trident de NetApp. Si desea ver una lista de las versiones de Kubernetes compatibles con ubeflow,

consulte la "Documentación oficial de Kubeflow". Si desea ver una lista de las versiones de Kubernetes compatibles con Trident, consulte "Documentación de Trident". Consulte las siguientes tablas para obtener información detallada sobre el entorno que se utilizó para validar la solución.

Componente de la infraestructura	Cantidad	Detalles	Sistema operativo
Host de salto de implementación	1	MÁQUINA VIRTUAL	Sistema operativo Ubuntu 20.04.2 LTS
Nodos maestros de Kubernetes	1	MÁQUINA VIRTUAL	Sistema operativo Ubuntu 20.04.2 LTS
Nodos de trabajo de Kubernetes	2	MÁQUINA VIRTUAL	Sistema operativo Ubuntu 20.04.2 LTS
Nodos de trabajo de GPU de Kubernetes	2	NVIDIA DGX-1 (nativo)	NVIDIA DGX OS 4.0.5 (basado en Ubuntu 18.04.2 LTS)
Reducida	1 par de alta disponibilidad	AFF A220 de NetApp	ONTAP 9.7 P6 de NetApp

Componente de software	Versión
Flujo de aire Apache	2.0.1
Tabla de Helm para flujo de aire de Apache	8.0.8
Docker	19.03.12
Kubeflow	1.2
Kubernetes	1.18.9
Trident de NetApp	21.01.2
DeepOps de NVIDIA	Funcionalidad de implementación de Trident desde la rama maestra hasta el encargo "61898cdfda"; Todas las demás funciones de la versión 21.03

Soporte técnico

NetApp no ofrece compatibilidad empresarial con Apache Airflow, Docker, Kubeflow, Kubernetes u NVIDIA DeepOps. Si está interesado en una solución totalmente compatible con funcionalidades similares a la solución del plano de control de IA de NetApp, "Póngase en contacto con NetApp" Acerca de soluciones de IA/ML totalmente compatibles que NetApp ofrece en colaboración con partners.

Puesta en marcha de Kubernetes

En esta sección se describen las tareas que debe completar para poner en marcha un clúster de Kubernetes en el que desea implementar la solución de plano de control de IA de NetApp. Si ya tiene un clúster de Kubernetes, puede omitir esta sección siempre que ejecute una versión de Kubernetes compatible con Kubeflow y NetApp Trident. Si desea ver una lista de las versiones de Kubernetes compatibles con ubeflow, consulte la

"Documentación oficial de Kubeflow". Si desea ver una lista de las versiones de Kubernetes compatibles con Trident, consulte "Documentación de Trident".

En el caso de puestas en marcha de Kubernetes en las instalaciones que incorporan nodos de reconstrucción completa con GPU de NVIDIA, NetApp recomienda usar la herramienta de puesta en marcha de Kubernetes DeepOps de NVIDIA. En esta sección, se describe la puesta en marcha de un clúster de Kubernetes con DeepOps.

Requisitos previos

Antes de realizar el ejercicio de implementación descrito en esta sección, asumimos que ya ha realizado las siguientes tareas:

1. Ya ha configurado todos los nodos Kubernetes con configuración básica (por ejemplo, un sistema NVIDIA DGX que forma parte de un pod ONTAP AI), de acuerdo con instrucciones de configuración estándar.
2. Ha instalado un sistema operativo compatible en todos los nodos maestro y de trabajo de Kubernetes y en un host de salto de implementación. Para obtener una lista de los sistemas operativos compatibles con DeepOps, consulte "Sitio DeepOps GitHub".

Utilice NVIDIA DeepOps para instalar y configurar Kubernetes

Para poner en marcha y configurar su clúster de Kubernetes con NVIDIA DeepOPS, realice las siguientes tareas desde un host de salto de implementación:

1. Descargue NVIDIA DeepOps siguiendo las instrucciones de "Página de Inicio" En el sitio de NVIDIA DeepOps GitHub.
2. Implemente Kubernetes en su clúster siguiendo las instrucciones que se indican en "Página de la Guía de implementación de Kubernetes" En el sitio de NVIDIA DeepOps GitHub.

Implementación y configuración de Trident de NetApp

Implementación y configuración de Trident de NetApp

En esta sección se describen las tareas que debe completar para instalar y configurar NetApp Trident en su clúster Kubernetes.

Requisitos previos

Antes de realizar el ejercicio de implementación descrito en esta sección, asumimos que ya ha realizado las siguientes tareas:

1. Ya existe un clúster de Kubernetes en funcionamiento y ejecuta una versión de Kubernetes compatible con Trident. Para obtener una lista de las versiones compatibles, consulte "Documentación de Trident".
2. Ya existe un dispositivo de almacenamiento de NetApp, una instancia definida por software o un servicio de almacenamiento en cloud que Trident admite.

Instale Trident

Para instalar y configurar NetApp Trident en su clúster de Kubernetes, realice las siguientes tareas desde el host de salto de implementación:

1. Implemente Trident mediante uno de los siguientes métodos:
   - Si utilizó NVIDIA DeepOps para poner en marcha su clúster de Kubernetes, también puede usar NVIDIA DeepOps para implementar Trident en el clúster de Kubernetes. Para poner en marcha Trident con DeepOps, siga "Instrucciones de puesta en marcha de Trident" En el sitio de NVIDIA DeepOps GitHub.
   - Si no utilizó NVIDIA DeepOPS para poner en marcha su clúster de Kubernetes, o si simplemente prefiere poner en marcha Trident manualmente, puede implementar Trident siguiendo el "instrucciones de puesta en funcionamiento" En la documentación de Trident. Asegúrese de crear al menos un back-end de Trident y al menos un Kubernetes StorageClass para obtener más información acerca de cómo configurar "Back-ends" y.. "Clases de almacenamiento" Consulte los subapartados vinculados en los documentos de NetApp.

> Si va a poner en marcha la solución del plano de control de IA de NetApp en un POD de IA de ONTAP, consulte "Ejemplo de Back-ends de Trident para puestas en marcha de ONTAP AI" Para algunos ejemplos de diferentes Back-ends de Trident que es posible que desee crear y. "Ejemplo de una historia Kubernetes para las puestas en marcha de IA de ONTAP" Puede que desee crear algunos ejemplos de diferentes clases de almacenamiento de Kubernetes.

Ejemplo de Back-ends de Trident para puestas en marcha de ONTAP AI

Antes de poder usar Trident para aprovisionar recursos de almacenamiento de forma dinámica dentro de su clúster de Kubernetes, debe crear uno o varios Back-ends de Trident. Los siguientes ejemplos representan diferentes tipos de backends que puede crear si va a implementar la solución de plano de control de IA de NetApp en un pod ONTAP AI. Para obtener más información acerca de backends, consulte "Documentación de Trident".

1. NetApp recomienda crear un Back-end de Trident habilitado para FlexGroup para cada LIF de datos (interfaz de red lógica que proporciona acceso a los datos) que desee utilizar en su sistema AFF de NetApp. Esto le permitirá equilibrar los montajes de volumen entre LIF

   Los siguientes comandos de ejemplo muestran la creación de dos Back-ends de Trident habilitados para FlexGroup para dos LIF de datos diferentes asociadas con la misma máquina virtual de almacenamiento (SVM) de ONTAP. Estas backends utilizan la `ontap-nas-flexgroup` controlador de almacenamiento. ONTAP admite dos tipos de volúmenes de datos principales: FlexVol y FlexGroup. Los volúmenes FlexVol tienen un tamaño limitado (a partir de la escritura, el tamaño máximo depende de la implementación específica). Por otro lado, los volúmenes FlexGroup se pueden escalar de forma lineal hasta 20 PB y 400 000 millones de archivos y, además, ofrecen un espacio de nombres único que simplifica enormemente la gestión de los datos. Por lo tanto, los volúmenes FlexGroup son óptimos para cargas de trabajo de IA y ML que dependen de grandes cantidades de datos.

   Si está trabajando con una pequeña cantidad de datos y desea usar volúmenes de FlexVol en lugar de volúmenes de FlexGroup, puede crear Back-ends de Trident que utilizan `ontap-nas` controlador de almacenamiento en lugar de `ontap-nas-flexgroup` controlador de almacenamiento.

```
$ cat << EOF > ./trident-backend-ontap-ai-flexgroups-iface1.json
{
    "version": 1,
```

```
    "storageDriverName": "ontap-nas-flexgroup",
    "backendName": "ontap-ai-flexgroups-iface1",
    "managementLIF": "10.61.218.100",
    "dataLIF": "192.168.11.11",
    "svm": "ontapai_nfs",
    "username": "admin",
    "password": "ontapai"
}
EOF
$ tridentctl create backend -f ./trident-backend-ontap-ai-flexgroups-
iface1.json -n trident
+----------------------------+---------------------
+--------------------------------------+--------+---------+
|            NAME            |   STORAGE DRIVER    |
UUID                 | STATE  | VOLUMES |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
| ontap-ai-flexgroups-iface1 | ontap-nas-flexgroup | b74cbddb-e0b8-40b7-
b263-b6da6dec0bdd | online |       0 |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
$ cat << EOF > ./trident-backend-ontap-ai-flexgroups-iface2.json
{
    "version": 1,
    "storageDriverName": "ontap-nas-flexgroup",
    "backendName": "ontap-ai-flexgroups-iface2",
    "managementLIF": "10.61.218.100",
    "dataLIF": "192.168.12.12",
    "svm": "ontapai_nfs",
    "username": "admin",
    "password": "ontapai"
}
EOF
$ tridentctl create backend -f ./trident-backend-ontap-ai-flexgroups-
iface2.json -n trident
+----------------------------+---------------------
+--------------------------------------+--------+---------+
|            NAME            |   STORAGE DRIVER    |
UUID                 | STATE  | VOLUMES |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
| ontap-ai-flexgroups-iface2 | ontap-nas-flexgroup | 61814d48-c770-436b-
9cb4-cf7ee661274d | online |       0 |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
$ tridentctl get backend -n trident
```

```
+----------------------------+---------------------
+--------------------------------------+--------+---------+
|            NAME            |   STORAGE DRIVER    |
UUID                 | STATE  | VOLUMES |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
| ontap-ai-flexgroups-iface1 | ontap-nas-flexgroup | b74cbddb-e0b8-40b7-
b263-b6da6dec0bdd | online |       0 |
| ontap-ai-flexgroups-iface2 | ontap-nas-flexgroup | 61814d48-c770-436b-
9cb4-cf7ee661274d | online |       0 |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
```

2. NetApp también recomienda crear uno o varios Back-ends de Trident habilitados para FlexVol. Si se utilizan FlexGroup Volumes para el entrenamiento del almacenamiento de conjuntos de datos, es posible que se desee usar volúmenes de FlexVol para almacenar resultados, resultados, información de depuración, etc. Si se desean usar volúmenes de FlexVol, se deben crear uno o varios Back-ends de Trident habilitados para FlexVol. Los siguientes comandos de ejemplo muestran la creación de una única estructura de fondo Trident habilitada para FlexVol que utiliza una sola LIF de datos.

```
$ cat << EOF > ./trident-backend-ontap-ai-flexvols.json
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "ontap-ai-flexvols",
    "managementLIF": "10.61.218.100",
    "dataLIF": "192.168.11.11",
    "svm": "ontapai_nfs",
    "username": "admin",
    "password": "ontapai"
}
EOF
$ tridentctl create backend -f ./trident-backend-ontap-ai-flexvols.json -n
trident
+----------------------------+---------------------
+--------------------------------------+--------+---------+
|            NAME            |   STORAGE DRIVER    |                 UUID
| STATE  | VOLUMES |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
| ontap-ai-flexvols          | ontap-nas           | 52bdb3b1-13a5-4513-
a9c1-52a69657fabe | online |       0 |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
$ tridentctl get backend -n trident
+----------------------------+---------------------
+--------------------------------------+--------+---------+
|            NAME            |   STORAGE DRIVER    |                 UUID
| STATE  | VOLUMES |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
| ontap-ai-flexvols          | ontap-nas           | 52bdb3b1-13a5-4513-
a9c1-52a69657fabe | online |       0 |
| ontap-ai-flexgroups-iface1 | ontap-nas-flexgroup | b74cbddb-e0b8-40b7-
b263-b6da6dec0bdd | online |       0 |
| ontap-ai-flexgroups-iface2 | ontap-nas-flexgroup | 61814d48-c770-436b-
9cb4-cf7ee661274d | online |       0 |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
```

Ejemplo de clases de almacenamiento Kubernetes para puestas en marcha de IA en ONTAP

Antes de poder usar Trident para aprovisionar recursos de almacenamiento de forma dinámica dentro del clúster de Kubernetes, debe crear una o varias clases de

almacenamiento de Kubernetes. Los siguientes ejemplos representan diferentes tipos de clases de almacenamiento que puede que desee crear si pone en marcha la solución de plano de control de IA de NetApp en un pool de IA de ONTAP. Si desea obtener más información sobre las clases de almacenamiento, consulte "Documentación de Trident".

1. NetApp recomienda crear un tipo de almacenamiento aparte para cada back-end de Trident habilitado para FlexGroup que haya creado en la sección "Ejemplo de Back-ends de Trident para puestas en marcha de ONTAP AI", paso 1. Estas clases de almacenamiento granular le permiten añadir montajes NFS que corresponden a LIF específicas (los LIF especificados cuando se crearon las back-ends de Trident) como un back-end concreto especificado en el archivo SPEC de StorageClass. Los comandos de ejemplo siguientes muestran la creación de dos StorageClasses que corresponden a los dos backends de ejemplo que se crearon en la sección "Ejemplo de Back-ends de Trident para puestas en marcha de ONTAP AI", paso 1. Si desea obtener más información sobre las clases de almacenamiento, consulte "Documentación de Trident".

   Para que no se elimine un volumen persistente cuando se elimine la reclamación de volumen persistente (RVP) correspondiente, en el siguiente ejemplo se utiliza un `reclaimPolicy` valor de `Retain`. Para obtener más información acerca de `reclaimPolicy` consulte el funcionario "Documentación de Kubernetes".

```
$ cat << EOF > ./storage-class-ontap-ai-flexgroups-retain-iface1.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-ai-flexgroups-retain-iface1
provisioner: netapp.io/trident
parameters:
  backendType: "ontap-nas-flexgroup"
  storagePools: "ontap-ai-flexgroups-iface1:.*"
reclaimPolicy: Retain
EOF
$ kubectl create -f ./storage-class-ontap-ai-flexgroups-retain-
iface1.yaml
storageclass.storage.k8s.io/ontap-ai-flexgroups-retain-iface1 created
$ cat << EOF > ./storage-class-ontap-ai-flexgroups-retain-iface2.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-ai-flexgroups-retain-iface2
provisioner: netapp.io/trident
parameters:
  backendType: "ontap-nas-flexgroup"
  storagePools: "ontap-ai-flexgroups-iface2:.*"
reclaimPolicy: Retain
EOF
$ kubectl create -f ./storage-class-ontap-ai-flexgroups-retain-
iface2.yaml
storageclass.storage.k8s.io/ontap-ai-flexgroups-retain-iface2 created
$ kubectl get storageclass
NAME                                PROVISIONER         AGE
ontap-ai-flexgroups-retain-iface1   netapp.io/trident   0m
ontap-ai-flexgroups-retain-iface2   netapp.io/trident   0m
```

2. NetApp también recomienda crear un StorageClass que se corresponda con el back-end Trident habilitado para FlexVol que ha creado en la sección "Ejemplo de Back-ends de Trident para puestas en marcha de ONTAP AI", paso 2. Los comandos de ejemplo siguientes muestran la creación de un solo tipo de almacenamiento para volúmenes FlexVol.

   En el siguiente ejemplo, no se ha especificado un back-end determinado en el archivo de definición StorageClass porque solo se creó un back-end Trident habilitado para FlexVol. Cuando se usa Kubernetes para administrar volúmenes que usan este clase de almacenamiento, Trident intenta usar cualquier back-end disponible que utilice `ontap-nas` controlador.

```
$ cat << EOF > ./storage-class-ontap-ai-flexvols-retain.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-ai-flexvols-retain
provisioner: netapp.io/trident
parameters:
  backendType: "ontap-nas"
reclaimPolicy: Retain
EOF
$ kubectl create -f ./storage-class-ontap-ai-flexvols-retain.yaml
storageclass.storage.k8s.io/ontap-ai-flexvols-retain created
$ kubectl get storageclass
NAME                                PROVISIONER         AGE
ontap-ai-flexgroups-retain-iface1   netapp.io/trident   1m
ontap-ai-flexgroups-retain-iface2   netapp.io/trident   1m
ontap-ai-flexvols-retain            netapp.io/trident   0m
```

3. NetApp también recomienda crear un tipo de almacenamiento genérico para volúmenes FlexGroup. En los siguientes comandos de ejemplo, se muestra la creación de un solo tipo de almacenamiento genérico para volúmenes FlexGroup.

   Tenga en cuenta que no se ha especificado un back-end determinado en el archivo de definición StorageClass. Por lo tanto, cuando se usa Kubernetes para administrar volúmenes que usan esta clase de almacenamiento, Trident intenta usar cualquier back-end disponible que utilice `ontap-nas-flexgroup` controlador.

```
$ cat << EOF > ./storage-class-ontap-ai-flexgroups-retain.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-ai-flexgroups-retain
provisioner: netapp.io/trident
parameters:
  backendType: "ontap-nas-flexgroup"
reclaimPolicy: Retain
EOF
$ kubectl create -f ./storage-class-ontap-ai-flexgroups-retain.yaml
storageclass.storage.k8s.io/ontap-ai-flexgroups-retain created
$ kubectl get storageclass
NAME                                PROVISIONER         AGE
ontap-ai-flexgroups-retain          netapp.io/trident   0m
ontap-ai-flexgroups-retain-iface1   netapp.io/trident   2m
ontap-ai-flexgroups-retain-iface2   netapp.io/trident   2m
ontap-ai-flexvols-retain            netapp.io/trident   1m
```

Despliegue de Kubeflow

En esta sección se describen las tareas que debe completar para poner en marcha Kubeflow en su clúster de Kubernetes.

Requisitos previos

Antes de realizar el ejercicio de implementación descrito en esta sección, asumimos que ya ha realizado las siguientes tareas:

1. Ya tiene un clúster de Kubernetes en funcionamiento y ejecuta una versión de Kubernetes que admite ubeflow. Para obtener una lista de las versiones compatibles, consulte "Documentación oficial de Kubeflow".
2. Ya ha instalado y configurado NetApp Trident en su clúster de Kubernetes, como se indica en "Implementación y configuración de Trident".

Establezca el tipo de almacenamiento de Kubernetes predeterminado

Antes de poner en marcha Kubeflow, debe designar un clase de almacenamiento predeterminado dentro del clúster de Kubernetes. El proceso de implementación de Kubeflow intenta aprovisionar nuevos volúmenes persistentes mediante el tipo de almacenamiento predeterminado. Si no se designa StorageClass como clase de almacenamiento predeterminado, la implementación falla. Para designar un StorageClass predeterminado en el clúster, realice la siguiente tarea desde el host de salto de implementación. Si ya ha designado un tipo de almacenamiento predeterminado en el clúster, puede omitir este paso.

1. Designe una de las clases de almacenamiento existentes como clase de almacenamiento predeterminada. Los comandos de ejemplo siguientes muestran la designación de un StorageClass llamado `ontap-ai-flexvols-retain` Como el tipo de almacenamiento predeterminado.

La `ontap-nas-flexgroup` El tipo de backend de Trident tiene un tamaño de RVP mínimo que es bastante grande. De manera predeterminada, Kubeflow intenta suministrar EVs que son sólo unos pocos GBS en tamaño. Por lo tanto, no debe designar un StorageClass que utilice `ontap-nas-flexgroup` Tipo back-end como StorageClass predeterminado para la implementación de Kubeflow.

```
$ kubectl get sc
NAME                                PROVISIONER             AGE
ontap-ai-flexgroups-retain          csi.trident.netapp.io   25h
ontap-ai-flexgroups-retain-iface1   csi.trident.netapp.io   25h
ontap-ai-flexgroups-retain-iface2   csi.trident.netapp.io   25h
ontap-ai-flexvols-retain            csi.trident.netapp.io   3s
$ kubectl patch storageclass ontap-ai-flexvols-retain -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
storageclass.storage.k8s.io/ontap-ai-flexvols-retain patched
$ kubectl get sc
NAME                                 PROVISIONER             AGE
ontap-ai-flexgroups-retain           csi.trident.netapp.io   25h
ontap-ai-flexgroups-retain-iface1    csi.trident.netapp.io   25h
ontap-ai-flexgroups-retain-iface2    csi.trident.netapp.io   25h
ontap-ai-flexvols-retain (default)   csi.trident.netapp.io   54s
```

Utilice NVIDIA DeepOps para poner en marcha Kubeflow

NetApp recomienda usar la herramienta de puesta en marcha de Kubeflow que proporciona NVIDIA DeepOps. Para poner en marcha Kubeflow en su clúster de Kubernetes con la herramienta de puesta en marcha DeepOps, siga estas tareas desde el host de salto de implementación.

Como alternativa, puede implementar Kubeflow manualmente siguiendo la "instrucciones de instalación" En la documentación oficial de Kubeflow

1. Implemente Kubeflow en su clúster siguiendo el "Instrucciones de despliegue de Kubeflow" En el sitio de NVIDIA DeepOps GitHub.
2. Tenga en cuenta la URL del panel de Kubeflow que genera la herramienta de puesta en marcha de DeepOps Kubeflow.

```
$ ./scripts/k8s/deploy_kubeflow.sh -x
…
INFO[0007] Applied the configuration Successfully!
filename="cmd/apply.go:72"
Kubeflow app installed to: /home/ai/kubeflow
It may take several minutes for all services to start. Run 'kubectl get
pods -n kubeflow' to verify
To remove (excluding CRDs, istio, auth, and cert-manager), run:
./scripts/k8s_deploy_kubeflow.sh -d
To perform a full uninstall : ./scripts/k8s_deploy_kubeflow.sh -D
Kubeflow Dashboard (HTTP NodePort): http://10.61.188.111:31380
```

3. Confirmar que todos los POD implementados en el espacio de nombres Kubeflow muestran un `STATUS` de `Running` y confirmar que ningún componente puesto en marcha en el espacio de nombres se encuentra en un estado de error. El inicio de todos los pods puede tardar varios minutos.

```
$ kubectl get all -n kubeflow
NAME                                                          READY
STATUS    RESTARTS   AGE
pod/admission-webhook-bootstrap-stateful-set-0                1/1
Running   0          95s
pod/admission-webhook-deployment-6b89c84c98-vrtbh             1/1
Running   0          91s
pod/application-controller-stateful-set-0                     1/1
Running   0          98s
pod/argo-ui-5dcf5d8b4f-m2wn4                                  1/1
Running   0          97s
pod/centraldashboard-cf4874ddc-7hcr8                          1/1
Running   0          97s
pod/jupyter-web-app-deployment-685b455447-gjhh7               1/1
Running   0          96s
pod/katib-controller-88c97d85c-kgq66                          1/1
Running   1          95s
pod/katib-db-8598468fd8-5jw2c                                 1/1
Running   0          95s
pod/katib-manager-574c8c67f9-wtrf5                            1/1
Running   1          95s
pod/katib-manager-rest-778857c989-fjbzn                       1/1
Running   0          95s
pod/katib-suggestion-bayesianoptimization-65df4d7455-qthmw    1/1
Running   0          94s
pod/katib-suggestion-grid-56bf69f597-98vwn                    1/1
Running   0          94s
pod/katib-suggestion-hyperband-7777b76cb9-9v6dq               1/1
Running   0          93s
```

```
pod/katib-suggestion-nasrl-77f6f9458c-2qzxq                    1/1
Running   0          93s
pod/katib-suggestion-random-77b88b5c79-l64j9                   1/1
Running   0          93s
pod/katib-ui-7587c5b967-nd629                                  1/1
Running   0          95s
pod/metacontroller-0                                           1/1
Running   0          96s
pod/metadata-db-5dd459cc-swzkm                                 1/1
Running   0          94s
pod/metadata-deployment-6cf77db994-69fk7                       1/1
Running   3          93s
pod/metadata-deployment-6cf77db994-mpbjt                       1/1
Running   3          93s
pod/metadata-deployment-6cf77db994-xg7tz                       1/1
Running   3          94s
pod/metadata-ui-78f5b59b56-qb6kr                               1/1
Running   0          94s
pod/minio-758b769d67-llvdr                                     1/1
Running   0          91s
pod/ml-pipeline-5875b9db95-g8t2k                               1/1
Running   0          91s
pod/ml-pipeline-persistenceagent-9b69ddd46-bt9r9               1/1
Running   0          90s
pod/ml-pipeline-scheduledworkflow-7b8d756c76-7x56s             1/1
Running   0          90s
pod/ml-pipeline-ui-79ffd9c76-fcwpd                             1/1
Running   0          90s
pod/ml-pipeline-viewer-controller-deployment-5fdc87f58-b2t9r   1/1
Running   0          90s
pod/mysql-657f87857d-l5k9z                                     1/1
Running   0          91s
pod/notebook-controller-deployment-56b4f59bbf-8bvnr            1/1
Running   0          92s
pod/profiles-deployment-6bc745947-mrdkh                        2/2
Running   0          90s
pod/pytorch-operator-77c97f4879-hmlrv                          1/1
Running   0          92s
pod/seldon-operator-controller-manager-0                       1/1
Running   1          91s
pod/spartakus-volunteer-5fdfddb779-l7qkm                       1/1
Running   0          92s
pod/tensorboard-6544748d94-nh8b2                               1/1
Running   0          92s
pod/tf-job-dashboard-56f79c59dd-6w59t                          1/1
Running   0          92s
```

```
pod/tf-job-operator-79cbfd6dbc-rb58c                          1/1
Running   0          91s
pod/workflow-controller-db644d554-cwrnb                       1/1
Running   0          97s
NAME                                                     TYPE
CLUSTER-IP      EXTERNAL-IP   PORT(S)             AGE
service/admission-webhook-service                        ClusterIP
10.233.51.169   <none>        443/TCP             97s
service/application-controller-service                   ClusterIP
10.233.4.54     <none>        443/TCP             98s
service/argo-ui                                          NodePort
10.233.47.191   <none>        80:31799/TCP        97s
service/centraldashboard                                 ClusterIP
10.233.8.36     <none>        80/TCP              97s
service/jupyter-web-app-service                          ClusterIP
10.233.1.42     <none>        80/TCP              97s
service/katib-controller                                 ClusterIP
10.233.25.226   <none>        443/TCP             96s
service/katib-db                                         ClusterIP
10.233.33.151   <none>        3306/TCP            97s
service/katib-manager                                    ClusterIP
10.233.46.239   <none>        6789/TCP            96s
service/katib-manager-rest                               ClusterIP
10.233.55.32    <none>        80/TCP              96s
service/katib-suggestion-bayesianoptimization            ClusterIP
10.233.49.191   <none>        6789/TCP            95s
service/katib-suggestion-grid                            ClusterIP
10.233.9.105    <none>        6789/TCP            95s
service/katib-suggestion-hyperband                       ClusterIP
10.233.22.2     <none>        6789/TCP            95s
service/katib-suggestion-nasrl                           ClusterIP
10.233.63.73    <none>        6789/TCP            95s
service/katib-suggestion-random                          ClusterIP
10.233.57.210   <none>        6789/TCP            95s
service/katib-ui                                         ClusterIP
10.233.6.116    <none>        80/TCP              96s
service/metadata-db                                      ClusterIP
10.233.31.2     <none>        3306/TCP            96s
service/metadata-service                                 ClusterIP
10.233.27.104   <none>        8080/TCP            96s
service/metadata-ui                                      ClusterIP
10.233.57.177   <none>        80/TCP              96s
service/minio-service                                    ClusterIP
10.233.44.90    <none>        9000/TCP            94s
service/ml-pipeline                                      ClusterIP
10.233.41.201   <none>        8888/TCP,8887/TCP   94s
```

```
service/ml-pipeline-tensorboard-ui                   ClusterIP
10.233.36.207   <none>        80/TCP              93s
service/ml-pipeline-ui                               ClusterIP
10.233.61.150   <none>        80/TCP              93s
service/mysql                                        ClusterIP
10.233.55.117   <none>        3306/TCP            94s
service/notebook-controller-service                  ClusterIP
10.233.10.166   <none>        443/TCP             95s
service/profiles-kfam                                ClusterIP
10.233.33.79    <none>        8081/TCP            92s
service/pytorch-operator                             ClusterIP
10.233.37.112   <none>        8443/TCP            95s
service/seldon-operator-controller-manager-service   ClusterIP
10.233.30.178   <none>        443/TCP             92s
service/tensorboard                                  ClusterIP
10.233.58.151   <none>        9000/TCP            94s
service/tf-job-dashboard                             ClusterIP
10.233.4.17     <none>        80/TCP              94s
service/tf-job-operator                              ClusterIP
10.233.60.32    <none>        8443/TCP            94s
service/webhook-server-service                       ClusterIP
10.233.32.167   <none>        443/TCP             87s
NAME                                                      READY   UP-
TO-DATE   AVAILABLE   AGE
deployment.apps/admission-webhook-deployment              1/1     1
1           97s
deployment.apps/argo-ui                                   1/1     1
1           97s
deployment.apps/centraldashboard                          1/1     1
1           97s
deployment.apps/jupyter-web-app-deployment                1/1     1
1           97s
deployment.apps/katib-controller                          1/1     1
1           96s
deployment.apps/katib-db                                  1/1     1
1           97s
deployment.apps/katib-manager                             1/1     1
1           96s
deployment.apps/katib-manager-rest                        1/1     1
1           96s
deployment.apps/katib-suggestion-bayesianoptimization     1/1     1
1           95s
deployment.apps/katib-suggestion-grid                     1/1     1
1           95s
deployment.apps/katib-suggestion-hyperband                1/1     1
1           95s
```

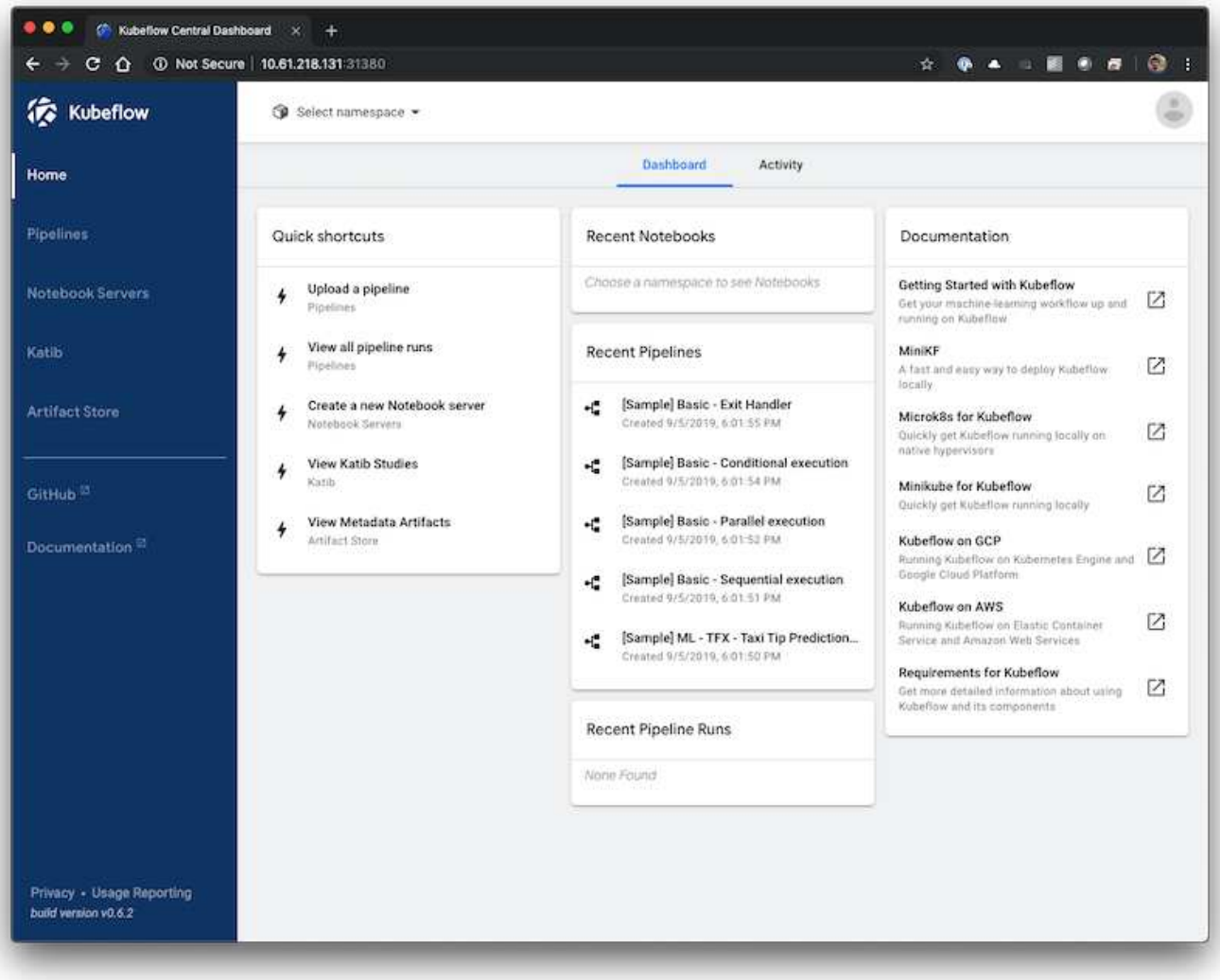
```
deployment.apps/katib-suggestion-nasrl                     1/1     1
1           95s
deployment.apps/katib-suggestion-random                    1/1     1
1           95s
deployment.apps/katib-ui                                   1/1     1
1           96s
deployment.apps/metadata-db                                1/1     1
1           96s
deployment.apps/metadata-deployment                        3/3     3
3           96s
deployment.apps/metadata-ui                                1/1     1
1           96s
deployment.apps/minio                                      1/1     1
1           94s
deployment.apps/ml-pipeline                                1/1     1
1           94s
deployment.apps/ml-pipeline-persistenceagent               1/1     1
1           93s
deployment.apps/ml-pipeline-scheduledworkflow              1/1     1
1           93s
deployment.apps/ml-pipeline-ui                             1/1     1
1           93s
deployment.apps/ml-pipeline-viewer-controller-deployment   1/1     1
1           93s
deployment.apps/mysql                                      1/1     1
1           94s
deployment.apps/notebook-controller-deployment             1/1     1
1           95s
deployment.apps/profiles-deployment                        1/1     1
1           92s
deployment.apps/pytorch-operator                           1/1     1
1           95s
deployment.apps/spartakus-volunteer                        1/1     1
1           94s
deployment.apps/tensorboard                                1/1     1
1           94s
deployment.apps/tf-job-dashboard                           1/1     1
1           94s
deployment.apps/tf-job-operator                            1/1     1
1           94s
deployment.apps/workflow-controller                        1/1     1
1           97s
NAME
DESIRED   CURRENT   READY   AGE
replicaset.apps/admission-webhook-deployment-6b89c84c98              1
1         1       97s
```

```
replicaset.apps/argo-ui-5dcf5d8b4f                                  1
1         1       97s
replicaset.apps/centraldashboard-cf4874ddc                          1
1         1       97s
replicaset.apps/jupyter-web-app-deployment-685b455447               1
1         1       97s
replicaset.apps/katib-controller-88c97d85c                          1
1         1       96s
replicaset.apps/katib-db-8598468fd8                                 1
1         1       97s
replicaset.apps/katib-manager-574c8c67f9                            1
1         1       96s
replicaset.apps/katib-manager-rest-778857c989                       1
1         1       96s
replicaset.apps/katib-suggestion-bayesianoptimization-65df4d7455    1
1         1       95s
replicaset.apps/katib-suggestion-grid-56bf69f597                    1
1         1       95s
replicaset.apps/katib-suggestion-hyperband-7777b76cb9               1
1         1       95s
replicaset.apps/katib-suggestion-nasrl-77f6f9458c                   1
1         1       95s
replicaset.apps/katib-suggestion-random-77b88b5c79                  1
1         1       95s
replicaset.apps/katib-ui-7587c5b967                                 1
1         1       96s
replicaset.apps/metadata-db-5dd459cc                                1
1         1       96s
replicaset.apps/metadata-deployment-6cf77db994                      3
3         3       96s
replicaset.apps/metadata-ui-78f5b59b56                              1
1         1       96s
replicaset.apps/minio-758b769d67                                    1
1         1       93s
replicaset.apps/ml-pipeline-5875b9db95                              1
1         1       93s
replicaset.apps/ml-pipeline-persistenceagent-9b69ddd46              1
1         1       92s
replicaset.apps/ml-pipeline-scheduledworkflow-7b8d756c76            1
1         1       91s
replicaset.apps/ml-pipeline-ui-79ffd9c76                            1
1         1       91s
replicaset.apps/ml-pipeline-viewer-controller-deployment-5fdc87f58  1
1         1       91s
replicaset.apps/mysql-657f87857d                                    1
1         1       92s
```

```
replicaset.apps/notebook-controller-deployment-56b4f59bbf            1
1         1       94s
replicaset.apps/profiles-deployment-6bc745947                        1
1         1       91s
replicaset.apps/pytorch-operator-77c97f4879                          1
1         1       94s
replicaset.apps/spartakus-volunteer-5fdfddb779                       1
1         1       94s
replicaset.apps/tensorboard-6544748d94                               1
1         1       93s
replicaset.apps/tf-job-dashboard-56f79c59dd                          1
1         1       93s
replicaset.apps/tf-job-operator-79cbfd6dbc                           1
1         1       93s
replicaset.apps/workflow-controller-db644d554                        1
1         1       97s
NAME                                                        READY   AGE
statefulset.apps/admission-webhook-bootstrap-stateful-set   1/1     97s
statefulset.apps/application-controller-stateful-set        1/1     98s
statefulset.apps/metacontroller                             1/1     98s
statefulset.apps/seldon-operator-controller-manager         1/1     92s
$ kubectl get pvc -n kubeflow
NAME             STATUS   VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS               AGE
katib-mysql      Bound    pvc-b07f293e-d028-11e9-9b9d-00505681a82d
10Gi       RWO            ontap-ai-flexvols-retain   27m
metadata-mysql   Bound    pvc-b0f3f032-d028-11e9-9b9d-00505681a82d
10Gi       RWO            ontap-ai-flexvols-retain   27m
minio-pv-claim   Bound    pvc-b22727ee-d028-11e9-9b9d-00505681a82d
20Gi       RWO            ontap-ai-flexvols-retain   27m
mysql-pv-claim   Bound    pvc-b2429afd-d028-11e9-9b9d-00505681a82d
20Gi       RWO            ontap-ai-flexvols-retain   27m
```

4. En su navegador web, acceda al panel central de Kubeflow navegando hasta la URL que anotó en el paso 2.

   El nombre de usuario predeterminado es `admin@kubeflow.org`, y la contraseña predeterminada es `12341234`. Para crear usuarios adicionales, siga las instrucciones de "Documentación oficial de Kubeflow".



Ejemplo de operaciones y tareas de Kubeflow

Esta sección incluye ejemplos de diversas operaciones y tareas que puede que desee realizar utilizando Kubeflow.

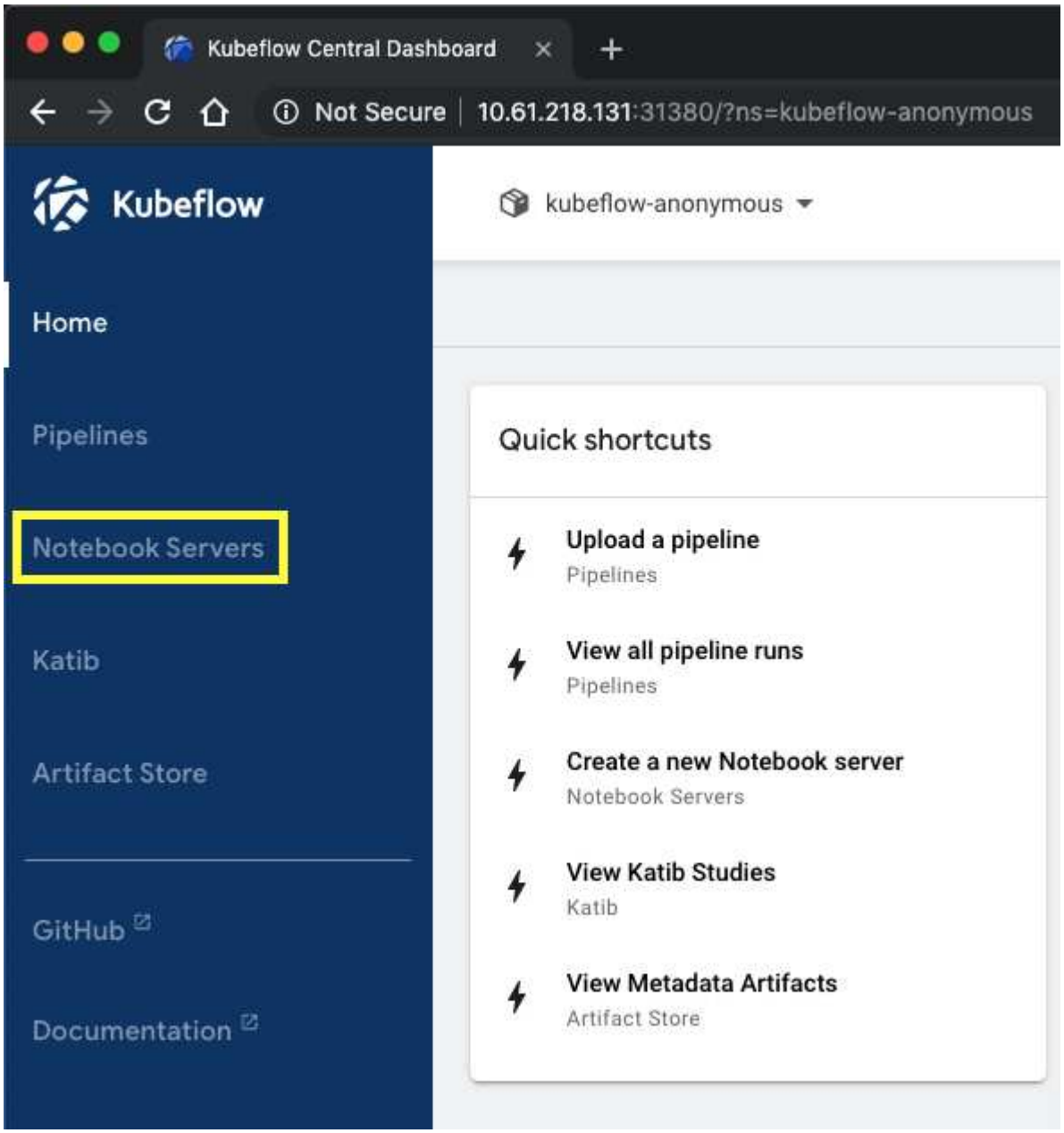
Ejemplo de operaciones y tareas de Kubeflow

Esta sección incluye ejemplos de diversas operaciones y tareas que puede que desee realizar utilizando Kubeflow.

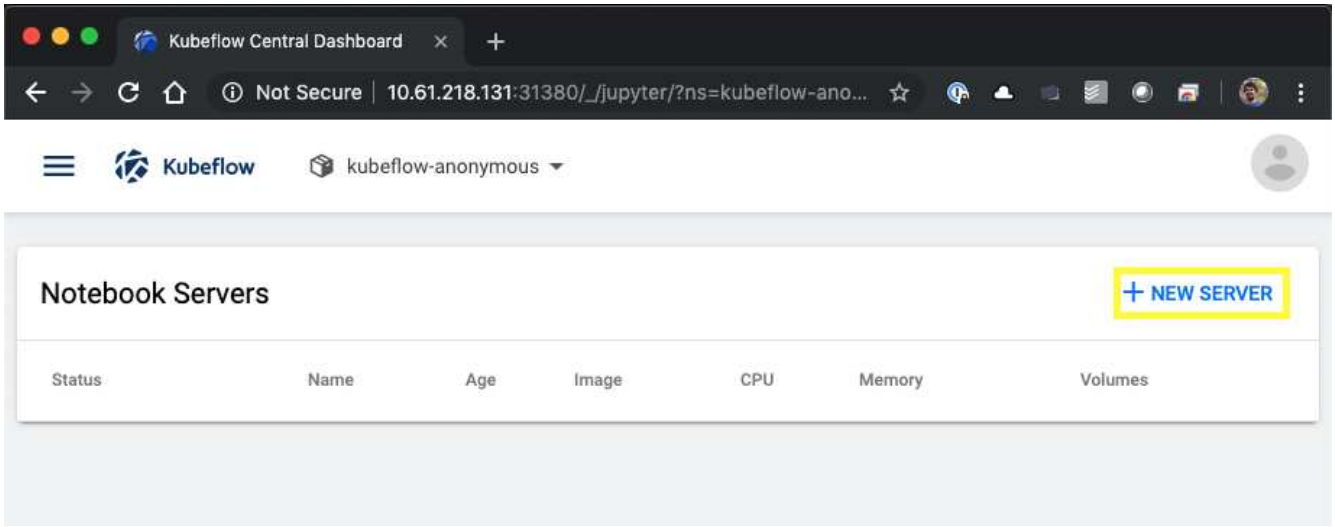
Aprovisione un espacio de trabajo para portátiles Juppyter para uso científico de datos o desarrollador

Kubeflow es capaz de suministrar rápidamente nuevos servidores Jupyter Notebook para actuar como espacios de trabajo de científicos de datos. Para aprovisionar un nuevo servidor Juppyter Notebook con Kubeflow, realice las siguientes tareas. Para obtener más información acerca de Jupyter Notebooks dentro del contexto de Kubeflow, consulte "Documentación oficial de Kubeflow".

1. Desde el panel central de Kubeflow, haga clic en Notebook Servers en el menú principal para acceder a la página de administración del servidor Jupyter Notebook.

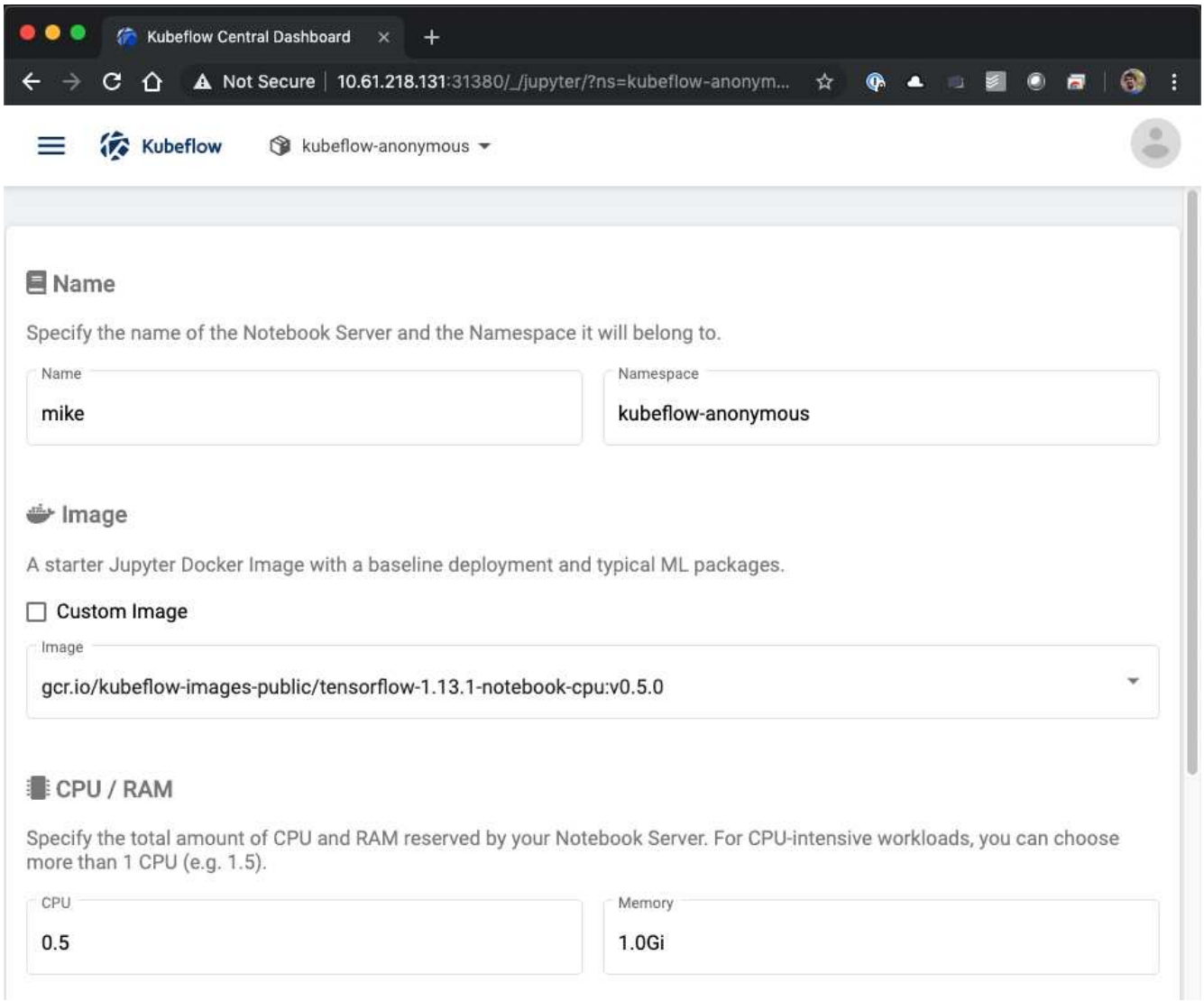


2. Haga clic en Nuevo servidor para aprovisionar un nuevo servidor Juppyter Notebook.

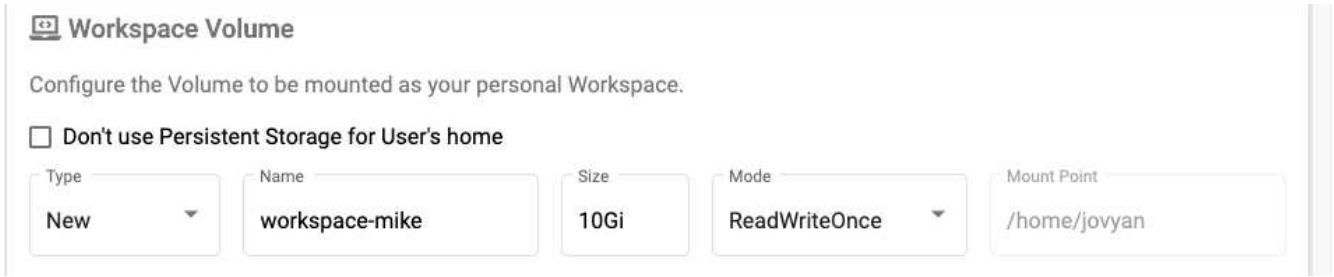


3. Asigne un nombre a su nuevo servidor, elija la imagen de Docker en la que desea que se base su servidor y especifique la cantidad de CPU y RAM que debe reservar su servidor. Si el campo espacio de nombres está en blanco, utilice el menú Seleccionar espacio de nombres en el encabezado de la página para elegir un espacio de nombres. El campo de espacio de nombres se rellena automáticamente con el espacio de nombres elegido.

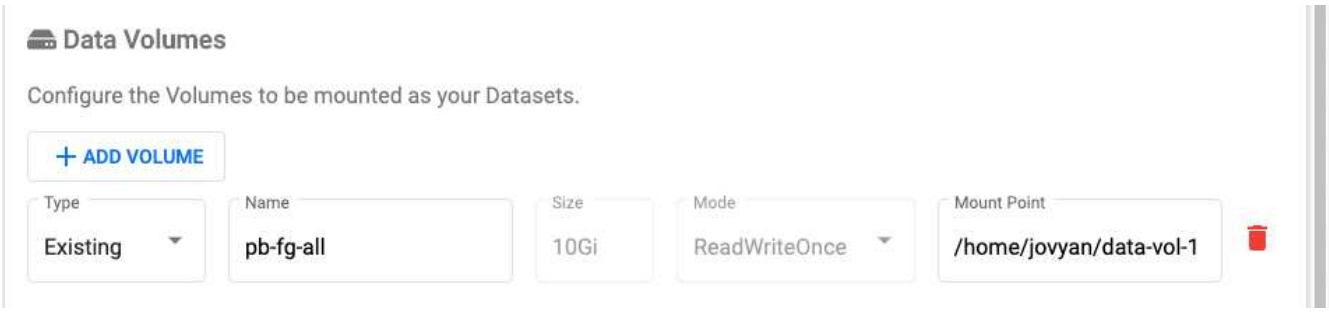
   En el siguiente ejemplo, la `kubeflow-anonymous` se elige espacio de nombres. Además, se aceptan los valores predeterminados de imagen Docker, CPU y RAM.



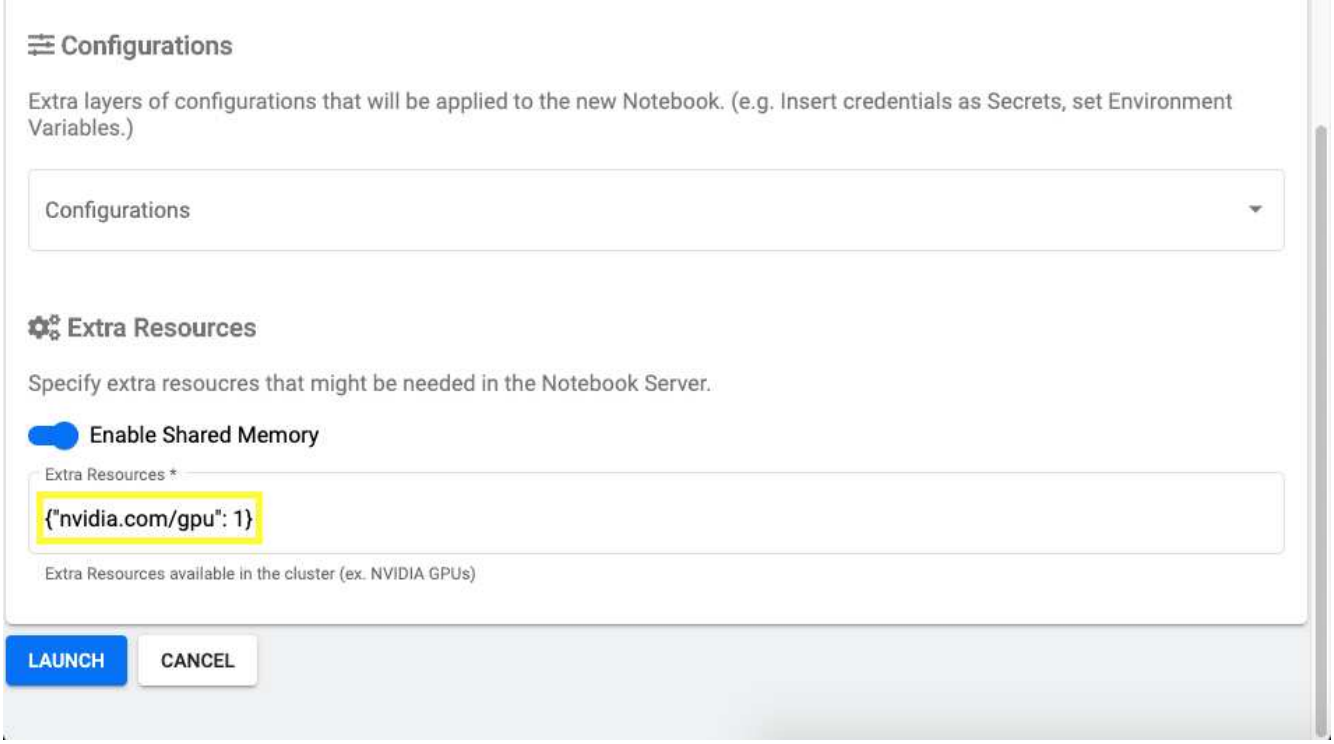
4. Especifique los detalles del volumen del área de trabajo. Si decide crear un nuevo volumen, ese volumen o RVP se aprovisionan con el tipo de almacenamiento predeterminado. Dado que se designó un clase de almacenamiento con Trident como el clase de almacenamiento predeterminado de la sección "Despliegue de Kubeflow", El volumen o PVC se aprovisiona con Trident. Este volumen se monta automáticamente como espacio de trabajo predeterminado dentro del contenedor servidor del portátil Jupyter. Los portátiles que cree un usuario en el servidor que no se guarden en un volumen de datos independiente se guardarán automáticamente en este volumen de área de trabajo. Por lo tanto, los portátiles se conservan entre reinicios.



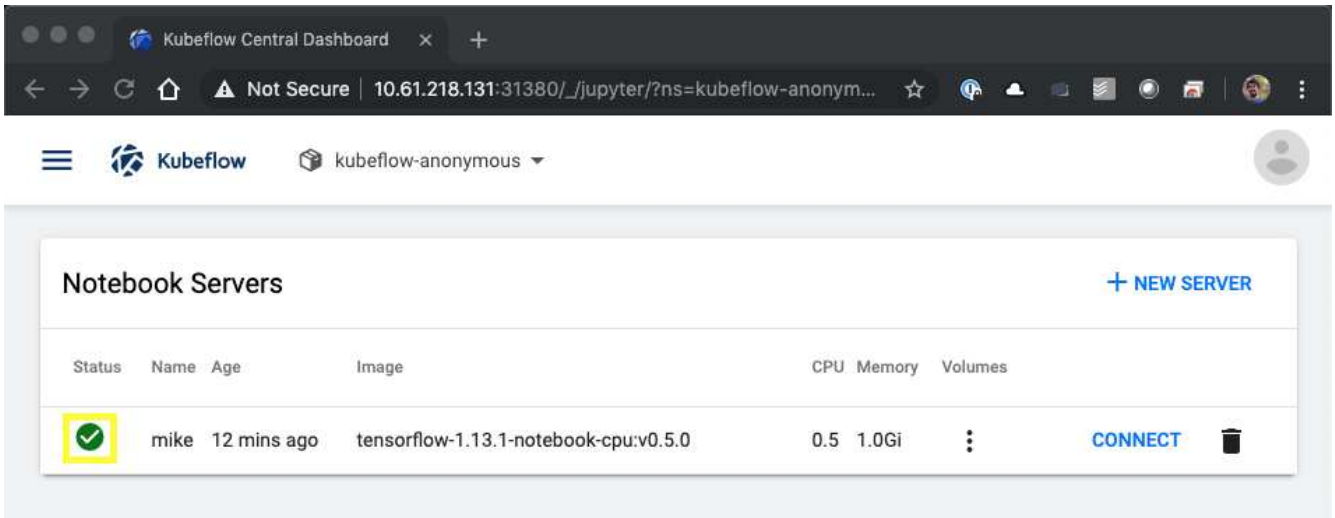
5. Añadir volúmenes de datos. En el siguiente ejemplo, se especifica un RVP existente llamado 'pb-fg-All' y acepta el punto de montaje predeterminado.



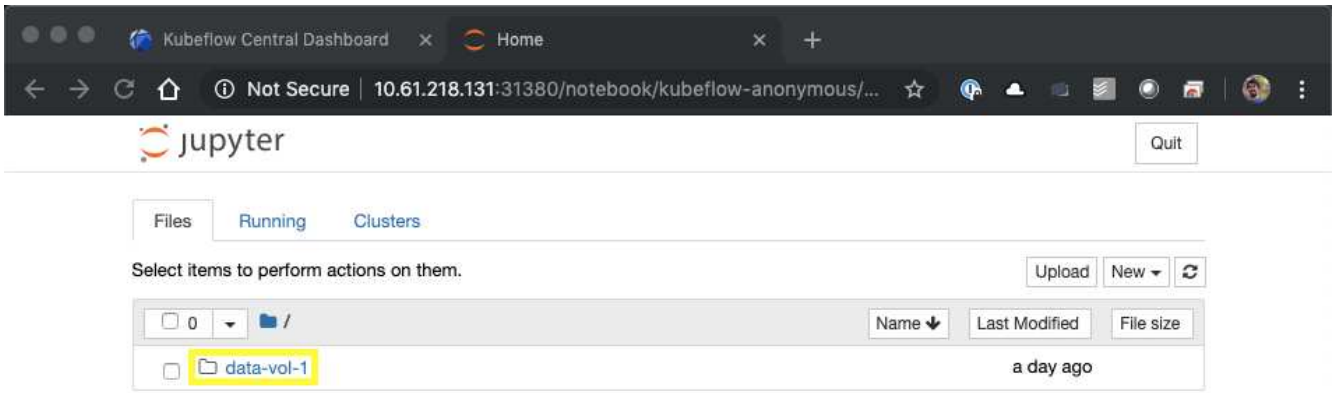
6. **Opcional:** Solicite que se asigne el número deseado de GPU al servidor de su portátil. En el siguiente ejemplo, se solicita una GPU.

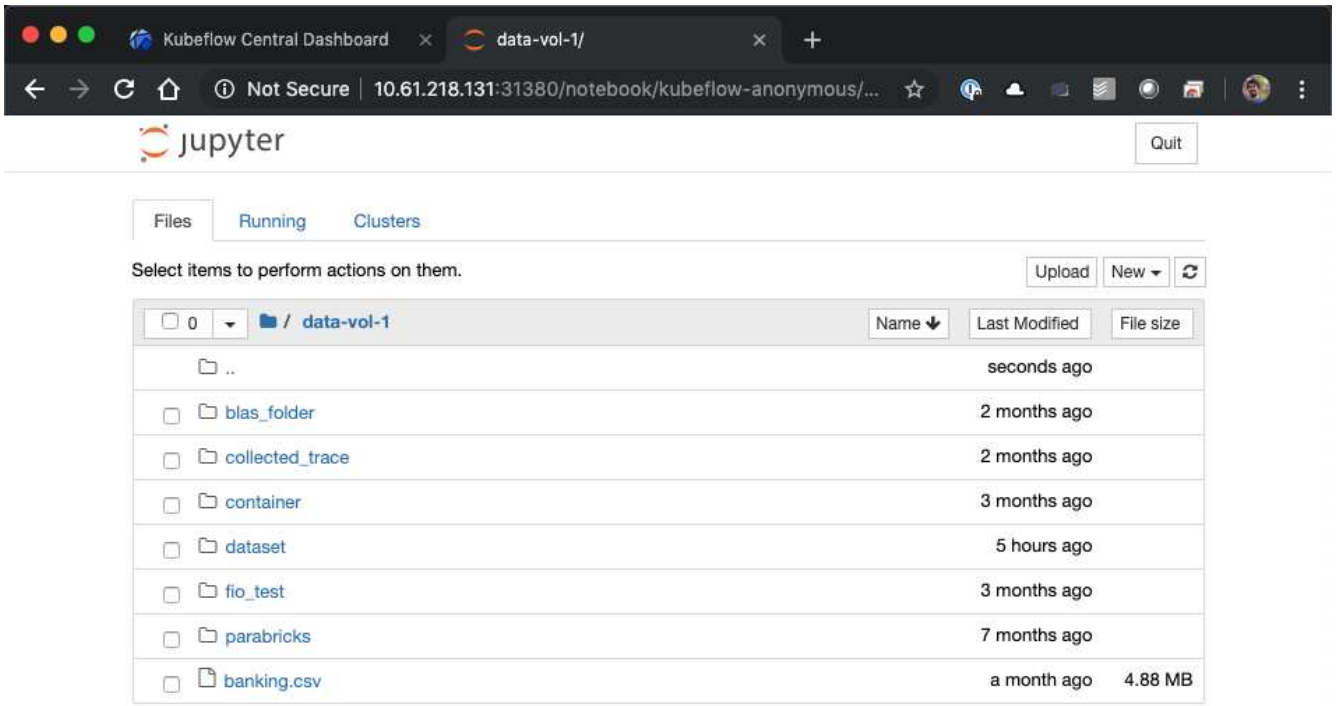


7. Haga clic en Iniciar para aprovisionar su nuevo servidor de notebook.
8. Espere a que el servidor de su portátil esté completamente aprovisionado. Esto puede tardar varios minutos si nunca ha aprovisionado un servidor con la imagen Docker que ha especificado porque es necesario descargar la imagen. Una vez que el servidor ha sido completamente aprovisionado, verá una Marca de verificación verde en la columna Estado de la página de administración del servidor Jupyter Notebook.



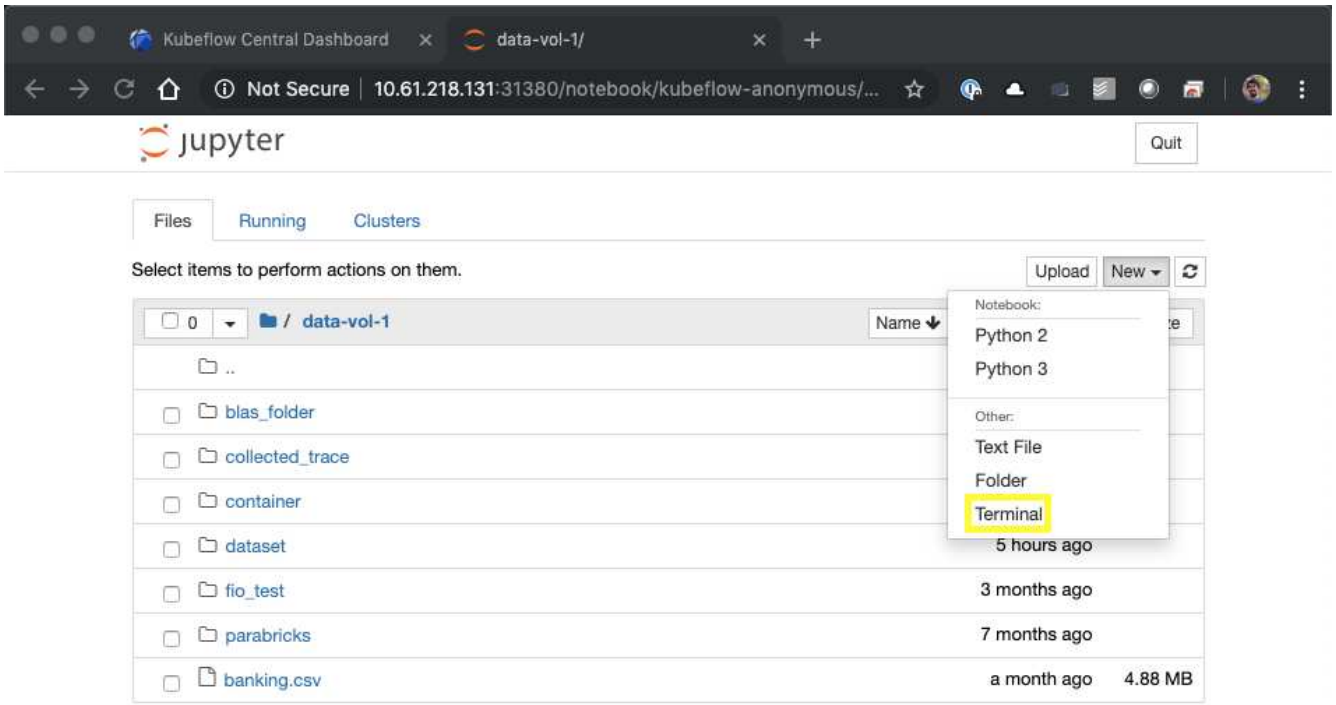
9. Haga clic en conectar para conectarse a la nueva interfaz web del servidor.
10. Confirme que el volumen del conjunto de datos especificado en el paso 6 está montado en el servidor. Tenga en cuenta que este volumen se monta en el espacio de trabajo predeterminado de forma predeterminada. Desde la perspectiva del usuario, ésta es sólo otra carpeta dentro del área de trabajo. El usuario, que es probable que un científico de datos y no sea un experto en infraestructura, no necesita tener experiencia en almacenamiento para utilizar este volumen.

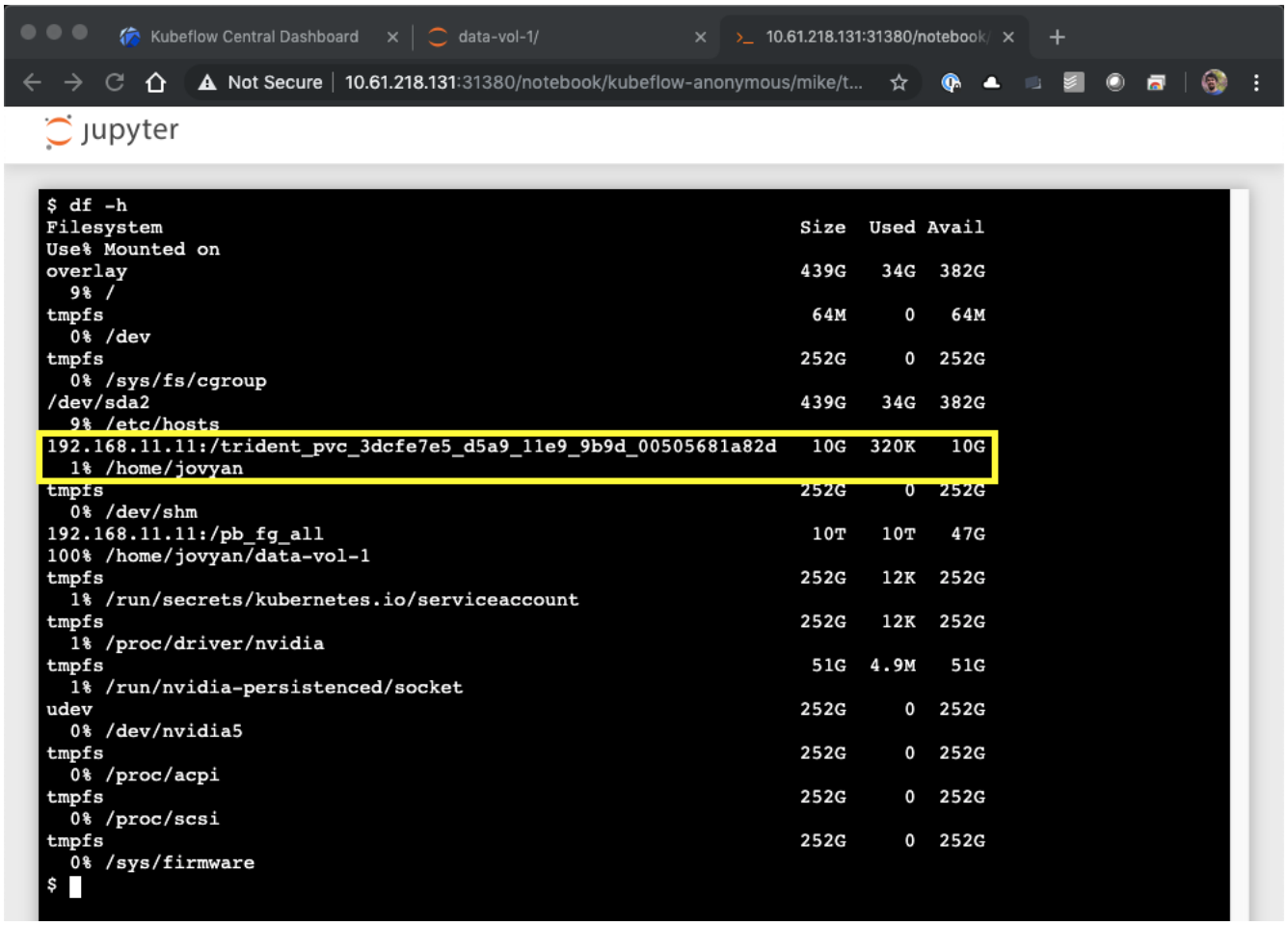




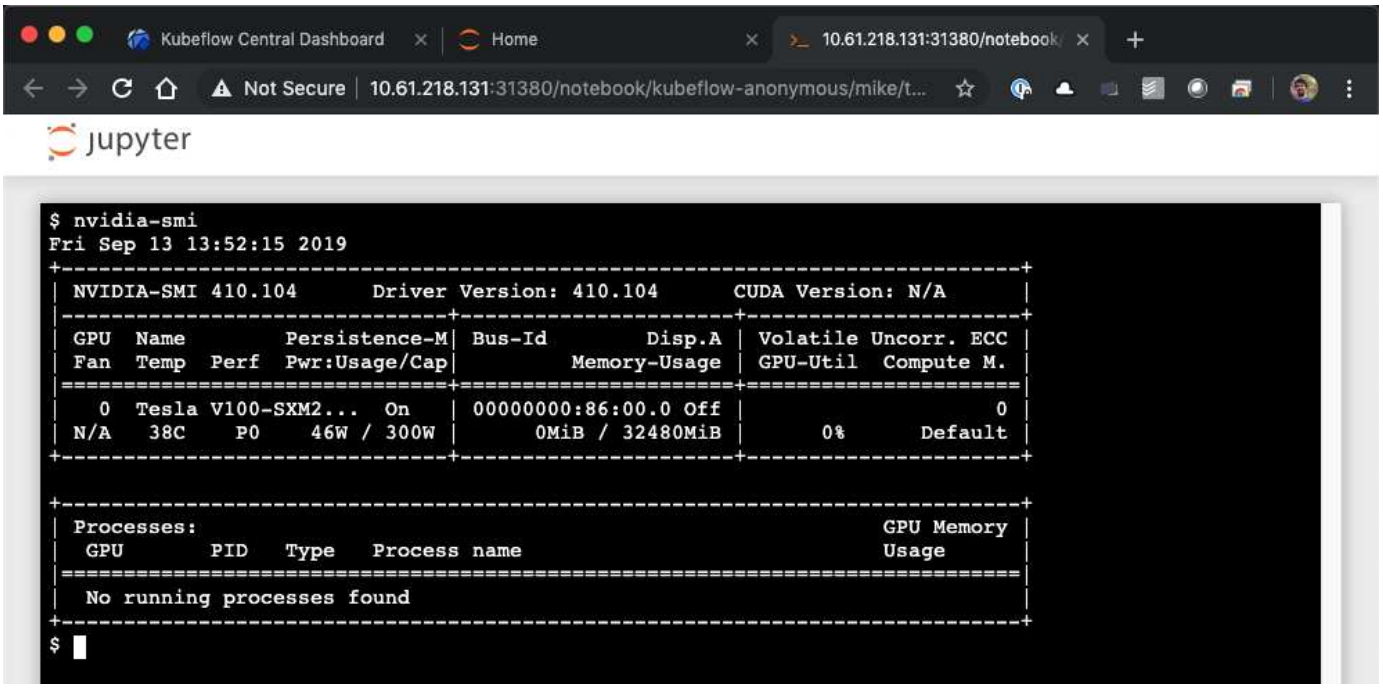
11. Abra un terminal y, suponiendo que se haya solicitado un nuevo volumen en el paso 5, ejecute `df -h` Para confirmar que un nuevo volumen persistente aprovisionado por Trident se monta como espacio de trabajo predeterminado.

    El directorio de área de trabajo predeterminado es el directorio base con el que se presenta cuando se accede por primera vez a la interfaz web del servidor. Por lo tanto, cualquier artefacto que cree mediante la interfaz web se almacena en este volumen persistente aprovisionado por Trident.





12. Con el terminal, ejecute `nvidia-smi` Para confirmar que se ha asignado el número correcto de GPU al servidor del portátil. En el siguiente ejemplo, se ha asignado una GPU al servidor de portátiles como se solicitó en el paso 7.



Ejemplos de cuadernos y canalizaciones

La "Kit de herramientas para la ciencia de datos de NetApp para Kubernetes" Se puede utilizar junto con Kubeflow. El uso del kit de herramientas para la ciencia de datos de NetApp con Kubeflow ofrece las siguientes ventajas:

- Los científicos de datos pueden llevar a cabo operaciones avanzadas de gestión de datos de NetApp directamente desde un portátil Juppyter.
- Las operaciones avanzadas de gestión de datos de NetApp pueden incorporarse en flujos de trabajo automatizados mediante el marco de canalizaciones de Kubeflow.

Consulte la "Ejemplos de Kubeflow" Sección dentro del repositorio de Data Science Toolkit de NetApp, GitHub para obtener información sobre el uso del kit de herramientas con Kubeflow.

Distribución del flujo de aire de Apache

NetApp recomienda ejecutar el flujo de aire de Apache sobre Kubernetes. En esta sección se describen las tareas que debe completar para poner en marcha el flujo de aire en el clúster de Kubernetes.

> Es posible poner en marcha un flujo de aire en plataformas distintas a Kubernetes. Esta solución no cubre la posibilidad de poner en marcha un flujo de aire en plataformas distintas a Kubernetes.

Requisitos previos

Antes de realizar el ejercicio de implementación descrito en esta sección, asumimos que ya ha realizado las siguientes tareas:

1. Ya tiene un clúster de Kubernetes en funcionamiento.
2. Ya ha instalado y configurado NetApp Trident en su clúster de Kubernetes como se indica en la sección “NetApp Trident Deployment and Configuration”.

Instale el Helm

El flujo de aire se pone en marcha con Helm, un conocido administrador de paquetes para Kubernetes. Antes de implementar el flujo de aire, debe instalar Helm en el host de salto de la implementación. Para instalar Helm en el host de salto de despliegue, siga la "instrucciones de instalación" En la documentación oficial de Helm.

Establezca el tipo de almacenamiento de Kubernetes predeterminado

Antes de implementar el flujo de aire, debe designar un tipo de almacenamiento predeterminado en el clúster de Kubernetes. El proceso de implementación de flujo de aire intenta aprovisionar nuevos volúmenes persistentes mediante el tipo de almacenamiento predeterminado. Si no se designa StorageClass como clase de almacenamiento predeterminado, la implementación falla. Para designar un tipo de almacenamiento predeterminado en el clúster, siga las instrucciones que se describen en la sección "Despliegue de Kubeflow". Si ya ha designado un tipo de almacenamiento predeterminado en el clúster, puede omitir este paso.

Utilice Helm para desplegar el flujo de aire

Para poner en marcha el flujo de aire en su clúster de Kubernetes con Helm, realice las siguientes tareas desde el host de salto de implementación:

1. Despliegue el flujo de aire con Helm siguiendo la "instrucciones de puesta en funcionamiento" Para el diagrama de flujo de aire oficial en el Hub de artefactos. Los comandos de ejemplo siguientes muestran la implementación del flujo de aire con Helm. Modifique, agregue o elimine valores en la `custom-values.yaml` fichero según sea necesario en función de su entorno y de la configuración deseada.

```
$ cat << EOF > custom-values.yaml
###################################
# Airflow - Common Configs
###################################
airflow:
  ## the airflow executor type to use
  ##
  executor: "CeleryExecutor"
  ## environment variables for the web/scheduler/worker Pods (for
airflow configs)
  ##
  #
###################################
# Airflow - WebUI Configs
###################################
web:
  ## configs for the Service of the web Pods
  ##
  service:
    type: NodePort
###################################
# Airflow - Logs Configs
###################################
logs:
  persistence:
    enabled: true
###################################
# Airflow - DAGs Configs
###################################
dags:
  ## configs for the DAG git repository & sync container
  ##
  gitSync:
    enabled: true
    ## url of the git repository
    ##
    repo: "git@github.com:mboglesby/airflow-dev.git"
```

```
    ## the branch/tag/sha1 which we clone
    ##
    branch: master
    revision: HEAD
    ## the name of a pre-created secret containing files for ~/.ssh/
    ##
    ## NOTE:
    ## - this is ONLY RELEVANT for SSH git repos
    ## - the secret commonly includes files: id_rsa, id_rsa.pub,
known_hosts
    ## - known_hosts is NOT NEEDED if `git.sshKeyscan` is true
    ##
    sshSecret: "airflow-ssh-git-secret"
    ## the name of the private key file in your `git.secret`
    ##
    ## NOTE:
    ## - this is ONLY RELEVANT for PRIVATE SSH git repos
    ##
    sshSecretKey: id_rsa
    ## the git sync interval in seconds
    ##
    syncWait: 60
EOF
$ helm install airflow airflow-stable/airflow -n airflow --version 8.0.8
--values ./custom-values.yaml
...
Congratulations. You have just deployed Apache Airflow!
1. Get the Airflow Service URL by running these commands:
   export NODE_PORT=$(kubectl get --namespace airflow -o
jsonpath="{.spec.ports[0].nodePort}" services airflow-web)
   export NODE_IP=$(kubectl get nodes --namespace airflow -o
jsonpath="{.items[0].status.addresses[0].address}")
   echo http://$NODE_IP:$NODE_PORT/
2. Open Airflow in your web browser
```

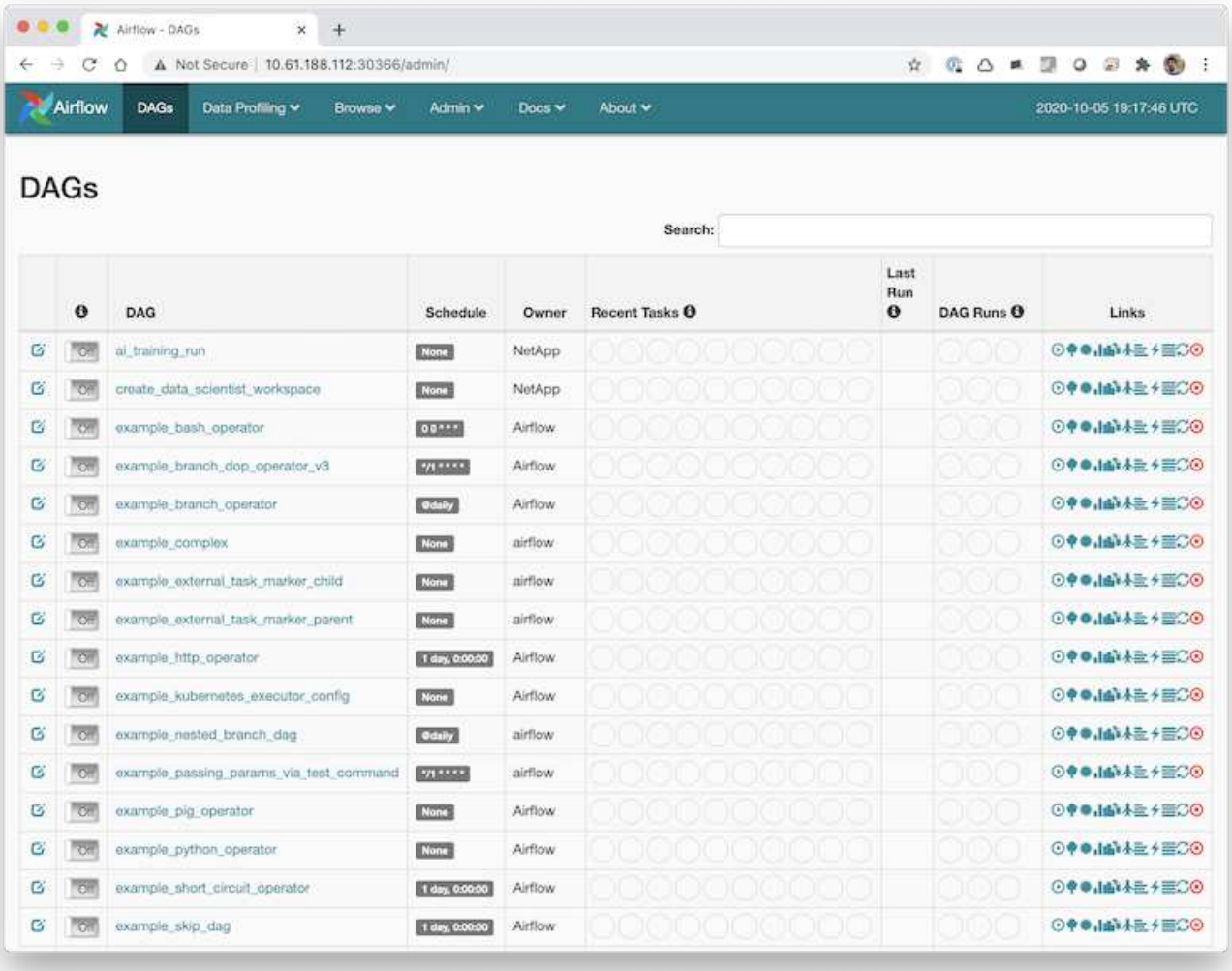
2. Confirme que todos los pods de flujo de aire estén activos y en funcionamiento. Puede tardar varios minutos en comenzar todos los pods.

```
$ kubectl -n airflow get pod
NAME                                READY   STATUS    RESTARTS   AGE
airflow-flower-b5656d44f-h8qjk      1/1     Running   0          2h
airflow-postgresql-0                1/1     Running   0          2h
airflow-redis-master-0              1/1     Running   0          2h
airflow-scheduler-9d95fcdf9-clf4b   2/2     Running   2          2h
airflow-web-59c94db9c5-z7rg4        1/1     Running   0          2h
airflow-worker-0                    2/2     Running   2          2h
```

3. Obtenga la URL del servicio web de flujo de aire siguiendo las instrucciones que se imprimieron en la consola cuando implementó el flujo de aire con Helm en el paso 1.

```
$ export NODE_PORT=$(kubectl get --namespace airflow -o
jsonpath="{.spec.ports[0].nodePort}" services airflow-web)
$ export NODE_IP=$(kubectl get nodes --namespace airflow -o
jsonpath="{.items[0].status.addresses[0].address}")
$ echo http://$NODE_IP:$NODE_PORT/
```

4. Confirme que puede acceder al servicio web de flujo de aire.



Ejemplo de flujos de trabajo de flujo de aire de Apache

La "Kit de herramientas para la ciencia de datos de NetApp para Kubernetes" Se puede utilizar junto con el flujo de aire. El uso del kit de herramientas para la ciencia de datos de NetApp con flujo de aire le permite incorporar operaciones de gestión de datos de NetApp en flujos de trabajo automatizados orquestados por flujo de aire.

Consulte la "Ejemplos de flujo de aire" Sección dentro del repositorio de Data Science Toolkit de NetApp, GitHub para obtener información sobre el uso del kit de herramientas con flujo de aire.

Ejemplo de operaciones de Trident

En esta sección se incluyen ejemplos de diversas operaciones que quizás desee realizar con Trident.

Importe un volumen existente

Si hay volúmenes existentes en su sistema/plataforma de almacenamiento de NetApp que desea montar en contenedores dentro de su clúster de Kubernetes, pero que no están ligados a las RVP en el clúster, debe

importar estos volúmenes. Es posible usar la funcionalidad de importación de volúmenes de Trident para importar estos volúmenes.

Los comandos de ejemplo siguientes muestran la importación del mismo volumen, denominado `pb_fg_all`, Dos veces, una para cada backend de Trident que se creó en el ejemplo de la sección "Ejemplo de Back-ends de Trident para puestas en marcha de ONTAP AI", paso 1. Importar el mismo volumen dos veces de esta manera le permite montar el volumen (un volumen FlexGroup existente) varias veces en diferentes LIF, como se describe en la sección "Ejemplo de Back-ends de Trident para puestas en marcha de ONTAP AI", paso 1. Para obtener más información acerca de las EVs, consulte "Documentación oficial sobre Kubernetes". Para obtener más información sobre la funcionalidad de importación de volúmenes, consulte "Documentación de Trident".

An `accessModes` valor de `ReadOnlyMany` Se especifica en los archivos de especificaciones de PVC de ejemplo. Para obtener más información acerca de `accessMode` consulte "Documentación oficial sobre Kubernetes".

> Los nombres de backend especificados en los siguientes comandos de importación de ejemplo corresponden a las backends que se crearon en el ejemplo de la sección "Ejemplo de Back-ends de Trident para puestas en marcha de ONTAP AI", paso 1. Los nombres de StorageClass especificados en el siguiente ejemplo de archivos de definición PVC corresponden a las clases de almacenamiento que se crearon en el ejemplo de la sección "Ejemplo de clases de almacenamiento Kubernetes para puestas en marcha de IA en ONTAP", paso 1.

```
$ cat << EOF > ./pvc-import-pb_fg_all-iface1.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pb-fg-all-iface1
  namespace: default
spec:
  accessModes:
    - ReadOnlyMany
  storageClassName: ontap-ai-flexgroups-retain-iface1
EOF
$ tridentctl import volume ontap-ai-flexgroups-iface1 pb_fg_all -f ./pvc-
import-pb_fg_all-iface1.yaml -n trident
+--------------------------------+--------
+-----------------------------------+----------
+-----------------------------------------------+--------+---------+
|          NAME                  |  SIZE  |       STORAGE CLASS
| PROTOCOL |             BACKEND UUID                          | STATE  |
MANAGED |
+--------------------------------+--------
+-----------------------------------+----------
+-----------------------------------------------+--------+---------+
| default-pb-fg-all-iface1-7d9f1 | 10 TiB | ontap-ai-flexgroups-retain-
iface1 | file     | b74cbddb-e0b8-40b7-b263-b6da6dec0bdd | online | true
|
+--------------------------------+--------
```

```
+-----------------------------------+----------
+-------------------------------------------+--------+---------+
$ cat << EOF > ./pvc-import-pb_fg_all-iface2.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pb-fg-all-iface2
  namespace: default
spec:
  accessModes:
    - ReadOnlyMany
  storageClassName: ontap-ai-flexgroups-retain-iface2
EOF
$ tridentctl import volume ontap-ai-flexgroups-iface2 pb_fg_all -f ./pvc-
import-pb_fg_all-iface2.yaml -n trident
+--------------------------------+--------
+-----------------------------------+----------
+-------------------------------------------+--------+---------+
|          NAME                  |  SIZE  |       STORAGE CLASS
| PROTOCOL |             BACKEND UUID                        | STATE  |
MANAGED |
+--------------------------------+--------
+-----------------------------------+----------
+-------------------------------------------+--------+---------+
| default-pb-fg-all-iface2-85aee | 10 TiB | ontap-ai-flexgroups-retain-
iface2 | file     | 61814d48-c770-436b-9cb4-cf7ee661274d | online | true
|
+--------------------------------+--------
+-----------------------------------+----------
+-------------------------------------------+--------+---------+
$ tridentctl get volume -n trident
+-----------------------------------+---------
+-----------------------------------+----------
+--------------------------------------+--------+---------+
|               NAME               |  SIZE   |           STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+-----------------------------------+---------
+-----------------------------------+----------
+--------------------------------------+--------+---------+
| default-pb-fg-all-iface1-7d9f1   | 10 TiB  | ontap-ai-flexgroups-retain-
iface1 | file     | b74cbddb-e0b8-40b7-b263-b6da6dec0bdd | online | true
|
| default-pb-fg-all-iface2-85aee   | 10 TiB  | ontap-ai-flexgroups-retain-
iface2 | file     | 61814d48-c770-436b-9cb4-cf7ee661274d | online | true
|
+-----------------------------------+---------
```

```
+-----------------------------------+----------
+--------------------------------------+--------+---------+
$ kubectl get pvc
NAME                  STATUS   VOLUME                                   CAPACITY
ACCESS MODES   STORAGECLASS                          AGE
pb-fg-all-iface1      Bound    default-pb-fg-all-iface1-7d9f1
10995116277760   ROX            ontap-ai-flexgroups-retain-iface1   25h
pb-fg-all-iface2      Bound    default-pb-fg-all-iface2-85aee
10995116277760   ROX            ontap-ai-flexgroups-retain-iface2   25h
```

Aprovisione un nuevo volumen

Puede usar Trident para aprovisionar un nuevo volumen en su plataforma o sistema de almacenamiento de NetApp. Los siguientes comandos de ejemplo muestran el aprovisionamiento de un volumen FlexVol nuevo. En este ejemplo, se aprovisiona el volumen con el tipo de almacenamiento que se creó en el ejemplo de la sección "Ejemplo de clases de almacenamiento Kubernetes para puestas en marcha de IA en ONTAP", paso 2.

An `accessModes` valor de `ReadWriteMany` Se especifica en el siguiente archivo de definición de PVC de ejemplo. Para obtener más información acerca de `accessMode` consulte "Documentación oficial sobre Kubernetes".

```
$ cat << EOF > ./pvc-tensorflow-results.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: tensorflow-results
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-ai-flexvols-retain
EOF
$ kubectl create -f ./pvc-tensorflow-results.yaml
persistentvolumeclaim/tensorflow-results created
$ kubectl get pvc
NAME                                STATUS    VOLUME
CAPACITY         ACCESS MODES   STORAGECLASS                        AGE
pb-fg-all-iface1                    Bound     default-pb-fg-all-iface1-7d9f1
10995116277760   ROX            ontap-ai-flexgroups-retain-iface1   26h
pb-fg-all-iface2                    Bound     default-pb-fg-all-iface2-85aee
10995116277760   ROX            ontap-ai-flexgroups-retain-iface2   26h
tensorflow-results                  Bound     default-tensorflow-results-
2fd60   1073741824       RWX            ontap-ai-flexvols-retain
25h
```

Ejemplo de tareas de alto rendimiento para puestas en marcha de ONTAP AI

En esta sección, se incluyen ejemplos de varias tareas de alto rendimiento que se pueden ejecutar cuando se pone en marcha Kubernetes en un pod ONTAP AI.

Ejemplo de tareas de alto rendimiento para puestas en marcha de ONTAP AI

En esta sección, se incluyen ejemplos de varias tareas de alto rendimiento que se pueden ejecutar cuando se pone en marcha Kubernetes en un pod ONTAP AI.

Ejecute una carga de trabajo de IA de un solo nodo

Para ejecutar una tarea DE IA y ML de un solo nodo en el clúster de Kubernetes, realice las siguientes tareas desde el host de puesta en marcha. Con Trident, puede crear de forma rápida y sencilla un volumen de datos, con potencialmente petabytes de datos al que se puede acceder una carga de trabajo de Kubernetes. Para que un volumen de datos de este tipo sea accesible desde un pod de Kubernetes, solo tiene que especificar

una RVP en la definición del pod. Este paso es una operación nativa de Kubernetes, no se necesita experiencia en NetApp.

> En esta sección se supone que ya ha realizado un contenedor (en el formato de contenedor de Docker) con la carga de trabajo específica DE IA y ML que intenta ejecutar en su clúster de Kubernetes.

1. Los siguientes comandos de ejemplo muestran la creación de un trabajo de Kubernetes para una carga de trabajo de prueba de ImageNET que utiliza el conjunto de datos de TensorFlow. Para obtener más información acerca del conjunto de datos ImageNET, consulte "Sitio web de ImageNET".

   Este trabajo de ejemplo solicita ocho GPU y, por lo tanto, puede ejecutarse en un solo nodo de trabajo de GPU con ocho o más GPU. Este trabajo de ejemplo se puede enviar en un clúster para el que no hay un nodo de trabajo con ocho o más GPU o esté ocupado actualmente con otra carga de trabajo. Si es así, el trabajo permanece en estado pendiente hasta que dicho nodo de trabajo esté disponible.

   Además, para maximizar el ancho de banda de almacenamiento, el volumen que contiene los datos de entrenamiento necesarios se monta dos veces en el pod que crea este trabajo. Otro volumen también se monta en el pod. Este segundo volumen se utilizará para almacenar resultados y métricas. Estos volúmenes se hacen referencia en la definición de trabajo utilizando los nombres de las RVP. Para obtener más información sobre los trabajos de Kubernetes, consulte "Documentación oficial sobre Kubernetes".

   An `emptyDir` volumen con un `medium` valor de `Memory` está montado en `/dev/shm` en el pod que crea este trabajo de ejemplo. El tamaño predeterminado de `/dev/shm` El volumen virtual que se crea automáticamente mediante el tiempo de ejecución del contenedor Docker puede en ocasiones ser insuficiente para las necesidades de TensorFlow. Montaje de un `emptyDir` volumen como en el ejemplo siguiente proporciona un tamaño suficiente `/dev/shm` volumen virtual. Para obtener más información acerca de `emptyDir` volúmenes, consulte "Documentación oficial sobre Kubernetes".

   El contenedor único que se especifica en esta definición de trabajo de ejemplo se proporciona un `securityContext > privileged` valor de `true`. Este valor significa que el contenedor tiene acceso raíz en el host de forma efectiva. Esta anotación se utiliza en este caso porque la carga de trabajo específica que se está ejecutando requiere acceso raíz. Específicamente, una operación de caché clara que ejecuta la carga de trabajo requiere acceso raíz. Si esto o no `privileged: true` la anotación es necesaria depende de los requisitos de la carga de trabajo específica que se esté ejecutando.

```
$ cat << EOF > ./netapp-tensorflow-single-imagenet.yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: netapp-tensorflow-single-imagenet
spec:
  backoffLimit: 5
  template:
    spec:
      volumes:
      - name: dshm
        emptyDir:
          medium: Memory
      - name: testdata-iface1
```

```
        persistentVolumeClaim:
          claimName: pb-fg-all-iface1
      - name: testdata-iface2
        persistentVolumeClaim:
          claimName: pb-fg-all-iface2
      - name: results
        persistentVolumeClaim:
          claimName: tensorflow-results
      containers:
      - name: netapp-tensorflow-py2
        image: netapp/tensorflow-py2:19.03.0
        command: ["python", "/netapp/scripts/run.py", "--
dataset_dir=/mnt/mount_0/dataset/imagenet", "--dgx_version=dgx1", "--
num_devices=8"]
        resources:
          limits:
            nvidia.com/gpu: 8
        volumeMounts:
        - mountPath: /dev/shm
          name: dshm
        - mountPath: /mnt/mount_0
          name: testdata-iface1
        - mountPath: /mnt/mount_1
          name: testdata-iface2
        - mountPath: /tmp
          name: results
        securityContext:
          privileged: true
      restartPolicy: Never
EOF
$ kubectl create -f ./netapp-tensorflow-single-imagenet.yaml
job.batch/netapp-tensorflow-single-imagenet created
$ kubectl get jobs
NAME                                       COMPLETIONS   DURATION   AGE
netapp-tensorflow-single-imagenet          0/1           24s        24s
```

2. Confirme que el trabajo que ha creado en el paso 1 se está ejecutando correctamente. El siguiente comando de ejemplo confirma que se creó un solo pod para el trabajo, tal como se especifica en la definición de trabajos, y que este pod se ejecuta actualmente en uno de los nodos de trabajo de la GPU.

```
$ kubectl get pods -o wide
NAME                                                READY   STATUS
RESTARTS   AGE
IP              NODE            NOMINATED NODE
netapp-tensorflow-single-imagenet-m7x92          1/1     Running     0
3m    10.233.68.61    10.61.218.154   <none>
```

3. Confirme que el trabajo que ha creado en el paso 1 se ha completado correctamente. Los siguientes comandos de ejemplo confirman que el trabajo se ha completado correctamente.

```
$ kubectl get jobs
NAME                                          COMPLETIONS   DURATION
AGE
netapp-tensorflow-single-imagenet             1/1           5m42s
10m
$ kubectl get pods
NAME                                                  READY   STATUS
RESTARTS   AGE
netapp-tensorflow-single-imagenet-m7x92               0/1     Completed
0          11m
$ kubectl logs netapp-tensorflow-single-imagenet-m7x92
[netapp-tensorflow-single-imagenet-m7x92:00008] PMIX ERROR: NO-
PERMISSIONS in file gds_dstore.c at line 702
[netapp-tensorflow-single-imagenet-m7x92:00008] PMIX ERROR: NO-
PERMISSIONS in file gds_dstore.c at line 711
Total images/sec = 6530.59125
================ Clean Cache !!! ==================
mpirun -allow-run-as-root -np 1 -H localhost:1 bash -c 'sync; echo 1 >
/proc/sys/vm/drop_caches'
==========================================
mpirun -allow-run-as-root -np 8 -H localhost:8 -bind-to none -map-by
slot -x NCCL_DEBUG=INFO -x LD_LIBRARY_PATH -x PATH python
/netapp/tensorflow/benchmarks_190205/scripts/tf_cnn_benchmarks/tf_cnn_be
nchmarks.py --model=resnet50 --batch_size=256 --device=gpu
--force_gpu_compatible=True --num_intra_threads=1 --num_inter_threads=48
--variable_update=horovod --batch_group_size=20 --num_batches=500
--nodistortions --num_gpus=1 --data_format=NCHW --use_fp16=True
--use_tf_layers=False --data_name=imagenet --use_datasets=True
--data_dir=/mnt/mount_0/dataset/imagenet
--datasets_parallel_interleave_cycle_length=10
--datasets_sloppy_parallel_interleave=False --num_mounts=2
--mount_prefix=/mnt/mount_%d --datasets_prefetch_buffer_size=2000
--datasets_use_prefetch=True --datasets_num_private_threads=4
--horovod_device=gpu >
/tmp/20190814_105450_tensorflow_horovod_rdma_resnet50_gpu_8_256_b500_ima
genet_nodistort_fp16_r10_m2_nockpt.txt 2>&1
```

4. **Opcional:** limpiar artefactos de trabajo. Los siguientes comandos de ejemplo muestran la eliminación del objeto de trabajo creado en el paso 1.

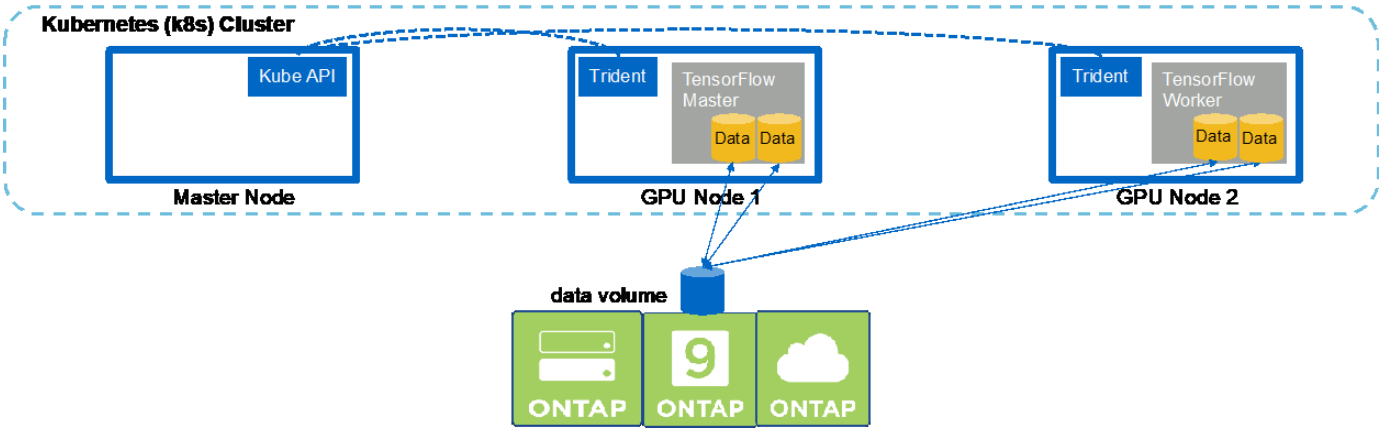
   Cuando se elimina el objeto de trabajo, Kubernetes elimina automáticamente todos los pods asociados.

```
$ kubectl get jobs
NAME                                                COMPLETIONS   DURATION
AGE
netapp-tensorflow-single-imagenet                   1/1           5m42s
10m
$ kubectl get pods
NAME                                                      READY   STATUS
RESTARTS   AGE
netapp-tensorflow-single-imagenet-m7x92                   0/1     Completed
0          11m
$ kubectl delete job netapp-tensorflow-single-imagenet
job.batch "netapp-tensorflow-single-imagenet" deleted
$ kubectl get jobs
No resources found.
$ kubectl get pods
No resources found.
```

Ejecute una carga de trabajo de IA distribuida síncrona

Para ejecutar un trabajo de IA y ML multinodo síncrono en un clúster de Kubernetes, lleve a cabo las siguientes tareas en el host de saltos de la puesta en marcha. Este proceso le permite aprovechar los datos almacenados en un volumen de NetApp y utilizar más GPU de las que puede proporcionar un único nodo de trabajo. Consulte la siguiente figura para obtener una descripción de un trabajo de IA distribuido sincrónica.

> Los trabajos distribuidos síncronos pueden ayudar a aumentar el rendimiento y la precisión de la formación en comparación con los trabajos distribuidos de manera asíncrona. Un análisis de los pros y los contras de los trabajos síncronos frente a los trabajos asincrónicos está fuera del alcance de este documento.



1. En los siguientes comandos de ejemplo, se muestra la creación de un trabajador que participa en la ejecución síncrona y distribuida de la misma tarea de prueba de rendimiento TensorFlow que se ejecutó en un solo nodo en el ejemplo de la sección "Ejecute una carga de trabajo de IA de un solo nodo". En este ejemplo específico, sólo se implementa un único trabajador porque el trabajo se ejecuta en dos nodos de

trabajo.

Esta puesta en marcha de trabajo de ejemplo solicita ocho GPU y, por lo tanto, puede ejecutarse en un único nodo de trabajo de GPU con ocho o más GPU. Si los nodos de trabajo de la GPU tienen más de ocho GPU, para maximizar el rendimiento, es posible que desee aumentar este número para ser igual al número de GPU que disponen los nodos de trabajo. Para obtener más información sobre las puestas en marcha de Kubernetes, consulte "Documentación oficial sobre Kubernetes".

En este ejemplo se crea una puesta en marcha de Kubernetes, ya que este trabajador en contenedor específico nunca lo completaría por sí solo. Por lo tanto, no tiene sentido implementarlo usando la construcción de trabajos de Kubernetes. Si su trabajador está diseñado o escrito para completar por sí solo, entonces podría tener sentido utilizar la construcción del trabajo para desplegar a su trabajador.

El POD especificado en esta especificación de implementación de ejemplo recibe una `hostNetwork` valor de `true`. Este valor significa que el pod utiliza la pila de red del nodo de trabajo del host en lugar de la pila de red virtual que Kubernetes suele crear para cada pod. Esta anotación se utiliza en este caso porque la carga de trabajo específica depende de Open MPI, NCCL y Horovod para ejecutar la carga de trabajo de forma síncrona distribuida. Por lo tanto, requiere acceso a la pila de red del host. Un debate sobre Open MPI, NCCL y Horovod está fuera del alcance de este documento. Si esto o no `hostNetwork: true` la anotación es necesaria depende de los requisitos de la carga de trabajo específica que se esté ejecutando. Para obtener más información acerca de `hostNetwork` consulte "Documentación oficial sobre Kubernetes".

```
$ cat << EOF > ./netapp-tensorflow-multi-imagenet-worker.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: netapp-tensorflow-multi-imagenet-worker
spec:
  replicas: 1
  selector:
    matchLabels:
      app: netapp-tensorflow-multi-imagenet-worker
  template:
    metadata:
      labels:
        app: netapp-tensorflow-multi-imagenet-worker
    spec:
      hostNetwork: true
      volumes:
      - name: dshm
        emptyDir:
          medium: Memory
      - name: testdata-iface1
        persistentVolumeClaim:
          claimName: pb-fg-all-iface1
      - name: testdata-iface2
        persistentVolumeClaim:
          claimName: pb-fg-all-iface2
```

```
      - name: results
        persistentVolumeClaim:
          claimName: tensorflow-results
      containers:
      - name: netapp-tensorflow-py2
        image: netapp/tensorflow-py2:19.03.0
        command: ["bash", "/netapp/scripts/start-slave-multi.sh",
"22122"]
        resources:
          limits:
            nvidia.com/gpu: 8
        volumeMounts:
        - mountPath: /dev/shm
          name: dshm
        - mountPath: /mnt/mount_0
          name: testdata-iface1
        - mountPath: /mnt/mount_1
          name: testdata-iface2
        - mountPath: /tmp
          name: results
        securityContext:
          privileged: true
EOF
$ kubectl create -f ./netapp-tensorflow-multi-imagenet-worker.yaml
deployment.apps/netapp-tensorflow-multi-imagenet-worker created
$ kubectl get deployments
NAME                                      DESIRED   CURRENT   UP-TO-DATE
AVAILABLE   AGE
netapp-tensorflow-multi-imagenet-worker   1         1         1
1           4s
```

2. Confirme que el despliegue del trabajador que creó en el paso 1 se inició correctamente. Los siguientes comandos de ejemplo confirman que se creó un solo pod de trabajadores para la implementación, tal y como se indica en la definición de la puesta en marcha, y que este pod se ejecuta actualmente en uno de los nodos de trabajo de la GPU.

```
$ kubectl get pods -o wide
NAME                                                      READY
STATUS    RESTARTS   AGE
IP              NODE            NOMINATED NODE
netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725   1/1
Running   0          60s   10.61.218.154   10.61.218.154   <none>
$ kubectl logs netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725
22122
```

3. Cree un trabajo de Kubernetes en un maestro que se inicia, participe y realice un seguimiento de la ejecución de un trabajo de varios nodos síncronos. Los siguientes comandos de ejemplo crean un maestro que inicia sesión, participa en y realiza un seguimiento de la ejecución síncrona distribuida de la misma tarea de prueba de rendimiento TensorFlow que se ejecutó en un solo nodo del ejemplo de la sección "Ejecute una carga de trabajo de IA de un solo nodo".

   Este trabajo maestro de ejemplo solicita ocho GPU y, por lo tanto, puede ejecutarse en un único nodo de trabajo de GPU con ocho o más GPU. Si los nodos de trabajo de la GPU tienen más de ocho GPU, para maximizar el rendimiento, es posible que desee aumentar este número para ser igual al número de GPU que disponen los nodos de trabajo.

   El POD maestro especificado en esta definición de trabajo de ejemplo recibe una `hostNetwork` valor de `true`, así como se le dio a la cápsula de trabajo un `hostNetwork` valor de `true` en el paso 1. Consulte el paso 1 para obtener más información acerca de por qué es necesario este valor.

```
$ cat << EOF > ./netapp-tensorflow-multi-imagenet-master.yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: netapp-tensorflow-multi-imagenet-master
spec:
  backoffLimit: 5
  template:
    spec:
      hostNetwork: true
      volumes:
      - name: dshm
        emptyDir:
          medium: Memory
      - name: testdata-iface1
        persistentVolumeClaim:
          claimName: pb-fg-all-iface1
      - name: testdata-iface2
        persistentVolumeClaim:
          claimName: pb-fg-all-iface2
      - name: results
        persistentVolumeClaim:
          claimName: tensorflow-results
      containers:
      - name: netapp-tensorflow-py2
        image: netapp/tensorflow-py2:19.03.0
        command: ["python", "/netapp/scripts/run.py", "--
dataset_dir=/mnt/mount_0/dataset/imagenet", "--port=22122", "--
num_devices=16", "--dgx_version=dgx1", "--
nodes=10.61.218.152,10.61.218.154"]
        resources:
          limits:
            nvidia.com/gpu: 8
```

```
        volumeMounts:
        - mountPath: /dev/shm
          name: dshm
        - mountPath: /mnt/mount_0
          name: testdata-iface1
        - mountPath: /mnt/mount_1
          name: testdata-iface2
        - mountPath: /tmp
          name: results
        securityContext:
          privileged: true
      restartPolicy: Never
EOF
$ kubectl create -f ./netapp-tensorflow-multi-imagenet-master.yaml
job.batch/netapp-tensorflow-multi-imagenet-master created
$ kubectl get jobs
NAME                                        COMPLETIONS   DURATION   AGE
netapp-tensorflow-multi-imagenet-master   0/1           25s        25s
```

4. Confirme que el trabajo maestro que creó en el paso 3 se está ejecutando correctamente. El siguiente comando de ejemplo confirma que se creó un único pod maestro para el trabajo, tal como se indica en la definición de trabajos, y que este pod se ejecuta actualmente en uno de los nodos de trabajo de la GPU. También debe ver que el pod de trabajo que originalmente vio en el paso 1 sigue en ejecución y que los pods maestro y trabajador se ejecutan en diferentes nodos.

```
$ kubectl get pods -o wide
NAME                                                              READY
STATUS    RESTARTS   AGE
IP              NODE            NOMINATED NODE
netapp-tensorflow-multi-imagenet-master-ppwwj                     1/1
Running   0          45s   10.61.218.152   10.61.218.152   <none>
netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725   1/1
Running   0          26m   10.61.218.154   10.61.218.154   <none>
```

5. Confirme que el trabajo maestro que ha creado en el paso 3 se ha completado correctamente. Los siguientes comandos de ejemplo confirman que el trabajo se ha completado correctamente.

```
$ kubectl get jobs
NAME                                        COMPLETIONS   DURATION   AGE
netapp-tensorflow-multi-imagenet-master   1/1           5m50s      9m18s
$ kubectl get pods
NAME                                                              READY
STATUS      RESTARTS   AGE
netapp-tensorflow-multi-imagenet-master-ppwwj                     0/1
Completed   0          9m38s
```

```
netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725   1/1
Running     0          35m
$ kubectl logs netapp-tensorflow-multi-imagenet-master-ppwwj
[10.61.218.152:00008] WARNING: local probe returned unhandled
shell:unknown assuming bash
rm: cannot remove '/lib': Is a directory
[10.61.218.154:00033] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at
line 702
[10.61.218.154:00033] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at
line 711
[10.61.218.152:00008] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at
line 702
[10.61.218.152:00008] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at
line 711
Total images/sec = 12881.33875
================ Clean Cache !!! ==================
mpirun -allow-run-as-root -np 2 -H 10.61.218.152:1,10.61.218.154:1 -mca
pml ob1 -mca btl ^openib -mca btl_tcp_if_include enp1s0f0 -mca
plm_rsh_agent ssh -mca plm_rsh_args "-p 22122" bash -c 'sync; echo 1 >
/proc/sys/vm/drop_caches'
==========================================
mpirun -allow-run-as-root -np 16 -H 10.61.218.152:8,10.61.218.154:8
-bind-to none -map-by slot -x NCCL_DEBUG=INFO -x LD_LIBRARY_PATH -x PATH
-mca pml ob1 -mca btl ^openib -mca btl_tcp_if_include enp1s0f0 -x
NCCL_IB_HCA=mlx5 -x NCCL_NET_GDR_READ=1 -x NCCL_IB_SL=3 -x
NCCL_IB_GID_INDEX=3 -x
NCCL_SOCKET_IFNAME=enp5s0.3091,enp12s0.3092,enp132s0.3093,enp139s0.3094
-x NCCL_IB_CUDA_SUPPORT=1 -mca orte_base_help_aggregate 0 -mca
plm_rsh_agent ssh -mca plm_rsh_args "-p 22122" python
/netapp/tensorflow/benchmarks_190205/scripts/tf_cnn_benchmarks/tf_cnn_be
nchmarks.py --model=resnet50 --batch_size=256 --device=gpu
--force_gpu_compatible=True --num_intra_threads=1 --num_inter_threads=48
--variable_update=horovod --batch_group_size=20 --num_batches=500
--nodistortions --num_gpus=1 --data_format=NCHW --use_fp16=True
--use_tf_layers=False --data_name=imagenet --use_datasets=True
--data_dir=/mnt/mount_0/dataset/imagenet
--datasets_parallel_interleave_cycle_length=10
--datasets_sloppy_parallel_interleave=False --num_mounts=2
--mount_prefix=/mnt/mount_%d --datasets_prefetch_buffer_size=2000 --
datasets_use_prefetch=True --datasets_num_private_threads=4
--horovod_device=gpu >
/tmp/20190814_161609_tensorflow_horovod_rdma_resnet50_gpu_16_256_b500_im
agenet_nodistort_fp16_r10_m2_nockpt.txt 2>&1
```

6. Elimine la implementación del trabajador cuando ya no la necesite. Los siguientes comandos de ejemplo muestran la eliminación del objeto de implementación de trabajo que se creó en el paso 1.

Cuando se elimina el objeto de implementación de trabajo, Kubernetes elimina automáticamente todos los pods de trabajador asociados.

```
$ kubectl get deployments
NAME                                        DESIRED   CURRENT   UP-TO-DATE
AVAILABLE   AGE
netapp-tensorflow-multi-imagenet-worker   1         1         1
1           43m
$ kubectl get pods
NAME                                                        READY
STATUS      RESTARTS   AGE
netapp-tensorflow-multi-imagenet-master-ppwwj              0/1
Completed   0          17m
netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725   1/1
Running     0          43m
$ kubectl delete deployment netapp-tensorflow-multi-imagenet-worker
deployment.extensions "netapp-tensorflow-multi-imagenet-worker" deleted
$ kubectl get deployments
No resources found.
$ kubectl get pods
NAME                                            READY   STATUS
RESTARTS   AGE
netapp-tensorflow-multi-imagenet-master-ppwwj   0/1     Completed   0
18m
```

7. **Opcional:** Limpie los artefactos del trabajo maestro. Los siguientes comandos de ejemplo muestran la eliminación del objeto de trabajo maestro que se creó en el paso 3.

   Cuando se elimina el objeto de trabajo maestro, Kubernetes elimina automáticamente todos los pods maestros asociados.

```
$ kubectl get jobs
NAME                                        COMPLETIONS   DURATION   AGE
netapp-tensorflow-multi-imagenet-master   1/1           5m50s      19m
$ kubectl get pods
NAME                                            READY   STATUS
RESTARTS   AGE
netapp-tensorflow-multi-imagenet-master-ppwwj   0/1     Completed   0
19m
$ kubectl delete job netapp-tensorflow-multi-imagenet-master
job.batch "netapp-tensorflow-multi-imagenet-master" deleted
$ kubectl get jobs
No resources found.
$ kubectl get pods
No resources found.
```

Pruebas de rendimiento

Hemos realizado una comparativa simple del rendimiento como parte de la creación de esta solución. Hemos ejecutado varias tareas estándar de prueba de rendimiento de IA de NetApp mediante Kubernetes, y comparamos los resultados de las pruebas de rendimiento con ejecuciones realizadas mediante un sencillo comando Docker run. No observamos diferencias notables en cuanto al rendimiento. Por lo tanto, llegamos a la conclusión de que el uso de Kubernetes para orquestar trabajos de entrenamiento de IA en contenedores no afecta negativamente al rendimiento. Consulte la siguiente tabla para ver los resultados de nuestra comparativa de rendimiento.

Prueba de rendimiento	Conjunto de datos	Docker Run (imágenes/s)	Kubernetes (imágenes/s)
TensorFlow de un solo nodo	Datos sintéticos	6,667.2475	6,661.93125
TensorFlow de un solo nodo	ImageNET	6,570.2025	6,530.59125
TensorFlow distribuido síncrono de dos nodos	Datos sintéticos	13,213.70625	13,218.288125
TensorFlow distribuido síncrono de dos nodos	ImageNET	12,941.69125	12,881.33875

Conclusión

Empresas y organizaciones de todos los tamaños y sectores se están decantando por la inteligencia artificial (IA), el aprendizaje automático (ML) y el aprendizaje profundo (DL) para solucionar problemas del mundo real, ofrecer productos y servicios innovadores y obtener una ventaja en un mercado cada vez más competitivo. A medida que las organizaciones aumentan el uso de la IA, EL ML y el AP, deben hacer frente a numerosos retos, como la escalabilidad de la carga de trabajo y la disponibilidad de los datos. Estos desafíos se pueden abordar mediante el uso de la solución de plano de control de IA de NetApp.

Esta solución le permite clonar rápidamente un espacio de nombres de los datos. Además, le permite definir e implementar flujos de trabajo de entrenamiento de IA, ML y DL que incorporan la creación casi instantánea de bases de datos y modelos para su trazabilidad y versionado. Con esta solución, puede rastrear cada entrenamiento de modelo único que se ejecuta en los conjuntos de datos exactos con los que se ha entrenado el modelo y/o validado. Por último, esta solución le permite aprovisionar rápidamente espacios de trabajo de los portátiles Juppyter con acceso a conjuntos de datos masivos.

Dado que esta solución está dirigida a científicos e ingenieros de datos, se requiere una experiencia mínima en NetApp o ONTAP de NetApp. Con esta solución, las funciones de gestión de datos se pueden ejecutar utilizando interfaces y herramientas sencillas y conocidas. Además, esta solución utiliza componentes totalmente de código abierto y libres. Por lo tanto, si ya dispone de almacenamiento de NetApp en su entorno, puede implantar esta solución hoy mismo. Si desea probar esta solución pero no dispone de almacenamiento de NetApp, visite "cloud.netapp.com"Y puede tener una solución de almacenamiento de NetApp basada en cloud en un instante.

Información de copyright

Copyright © 2024 NetApp, Inc. Todos los derechos reservados. Imprimido en EE. UU. No se puede reproducir este documento protegido por copyright ni parte del mismo de ninguna forma ni por ningún medio (gráfico, electrónico o mecánico, incluidas fotocopias, grabaciones o almacenamiento en un sistema de recuperación electrónico) sin la autorización previa y por escrito del propietario del copyright.

El software derivado del material de NetApp con copyright está sujeto a la siguiente licencia y exención de responsabilidad:

ESTE SOFTWARE LO PROPORCIONA NETAPP «TAL CUAL» Y SIN NINGUNA GARANTÍA EXPRESA O IMPLÍCITA, INCLUYENDO, SIN LIMITAR, LAS GARANTÍAS IMPLÍCITAS DE COMERCIALIZACIÓN O IDONEIDAD PARA UN FIN CONCRETO, CUYA RESPONSABILIDAD QUEDA EXIMIDA POR EL PRESENTE DOCUMENTO. EN NINGÚN CASO NETAPP SERÁ RESPONSABLE DE NINGÚN DAÑO DIRECTO, INDIRECTO, ESPECIAL, EJEMPLAR O RESULTANTE (INCLUYENDO, ENTRE OTROS, LA OBTENCIÓN DE BIENES O SERVICIOS SUSTITUTIVOS, PÉRDIDA DE USO, DE DATOS O DE BENEFICIOS, O INTERRUPCIÓN DE LA ACTIVIDAD EMPRESARIAL) CUALQUIERA SEA EL MODO EN EL QUE SE PRODUJERON Y LA TEORÍA DE RESPONSABILIDAD QUE SE APLIQUE, YA SEA EN CONTRATO, RESPONSABILIDAD OBJETIVA O AGRAVIO (INCLUIDA LA NEGLIGENCIA U OTRO TIPO), QUE SURJAN DE ALGÚN MODO DEL USO DE ESTE SOFTWARE, INCLUSO SI HUBIEREN SIDO ADVERTIDOS DE LA POSIBILIDAD DE TALES DAÑOS.

NetApp se reserva el derecho de modificar cualquiera de los productos aquí descritos en cualquier momento y sin aviso previo. NetApp no asume ningún tipo de responsabilidad que surja del uso de los productos aquí descritos, excepto aquello expresamente acordado por escrito por parte de NetApp. El uso o adquisición de este producto no lleva implícita ninguna licencia con derechos de patente, de marcas comerciales o cualquier otro derecho de propiedad intelectual de NetApp.

Es posible que el producto que se describe en este manual esté protegido por una o más patentes de EE. UU., patentes extranjeras o solicitudes pendientes.

LEYENDA DE DERECHOS LIMITADOS: el uso, la copia o la divulgación por parte del gobierno están sujetos a las restricciones establecidas en el subpárrafo (b)(3) de los derechos de datos técnicos y productos no comerciales de DFARS 252.227-7013 (FEB de 2014) y FAR 52.227-19 (DIC de 2007).

Los datos aquí contenidos pertenecen a un producto comercial o servicio comercial (como se define en FAR 2.101) y son propiedad de NetApp, Inc. Todos los datos técnicos y el software informático de NetApp que se proporcionan en este Acuerdo tienen una naturaleza comercial y se han desarrollado exclusivamente con fondos privados. El Gobierno de EE. UU. tiene una licencia limitada, irrevocable, no exclusiva, no transferible, no sublicenciable y de alcance mundial para utilizar los Datos en relación con el contrato del Gobierno de los Estados Unidos bajo el cual se proporcionaron los Datos. Excepto que aquí se disponga lo contrario, los Datos no se pueden utilizar, desvelar, reproducir, modificar, interpretar o mostrar sin la previa aprobación por escrito de NetApp, Inc. Los derechos de licencia del Gobierno de los Estados Unidos de América y su Departamento de Defensa se limitan a los derechos identificados en la cláusula 252.227-7015(b) de la sección DFARS (FEB de 2014).

Información de la marca comercial

NETAPP, el logotipo de NETAPP y las marcas que constan en http://www.netapp.com/TM son marcas comerciales de NetApp, Inc. El resto de nombres de empresa y de producto pueden ser marcas comerciales de sus respectivos propietarios.