Daten-Pipelines, Data Lakes und Management

NetApp Solutions

NetApp
April 17, 2024

This PDF was generated from https://docs.netapp.com/de-de/netapp-solutions/ai/mlops_fsxn_s3_integration.html on April 17, 2024. Always check docs.netapp.com for the latest.

Inhalt

Daten-Pipelines, Data Lakes und Management . . . 1
AWS FSX for NetApp ONTAP (FSxN) für MLOps . . . 1
MLOps für die Hybrid-Multi-Cloud mit Domino Data Lab und NetApp . . . 36
NVIDIA AI Enterprise with NetApp and VMware . . . 51
TR-4851: NetApp StorageGRID Data Lake für Workloads beim autonomen Fahren – Lösungsdesign . . . 61
NetApp AI Control Plane . . . 61
MLRun-Pipeline mit Iguazio . . . 118
TR-4915: Datenverschiebung mit E-Series und BeeGFS für KI- und Analyse-Workflows . . . 146

Daten-Pipelines, Data Lakes und Management

AWS FSX for NetApp ONTAP (FSxN) für MLOps

Autor(en):
Jian Jian (Ken), Senior Data & Applied Scientist, NetApp

In diesem Abschnitt wird die praktische Anwendung der Entwicklung von KI-Infrastrukturen erläutert und der Aufbau einer MLOps-Pipeline mithilfe von FSxN durchgängig dargestellt. Sie umfasst drei umfassende Beispiele und hilft Ihnen, Ihre MLOps-Anforderungen über diese leistungsstarke Datenmanagement-Plattform zu erfüllen.

Die Schwerpunkte dieser Artikel liegen auf:

1. "Teil 1 – Integration von AWS FSX for NetApp ONTAP (FSxN) als privater S3-Bucket in AWS SageMaker"
2. "Teil 2 – Nutzung von AWS FSX for NetApp ONTAP (FSxN) als Datenquelle für das Modelltraining in SageMaker"
3. "Teil 3 – Aufbau Einer vereinfachten MLOPS-Pipeline (CI/CT/CD)"

Am Ende dieses Abschnitts haben Sie ein fundiertes Verständnis dafür gewonnen, wie Sie FSxN zur Optimierung von MLOPS-Prozessen nutzen können.

Teil 1 – Integration von AWS FSX for NetApp ONTAP (FSxN) als privater S3-Bucket in AWS SageMaker

Autor(en):
Jian Jian (Ken), Senior Data & Applied Scientist, NetApp

Einführung

Mithilfe von SageMaker als Beispiel bietet diese Seite Anleitungen zur Konfiguration von FSxN als privaten S3-Bucket.

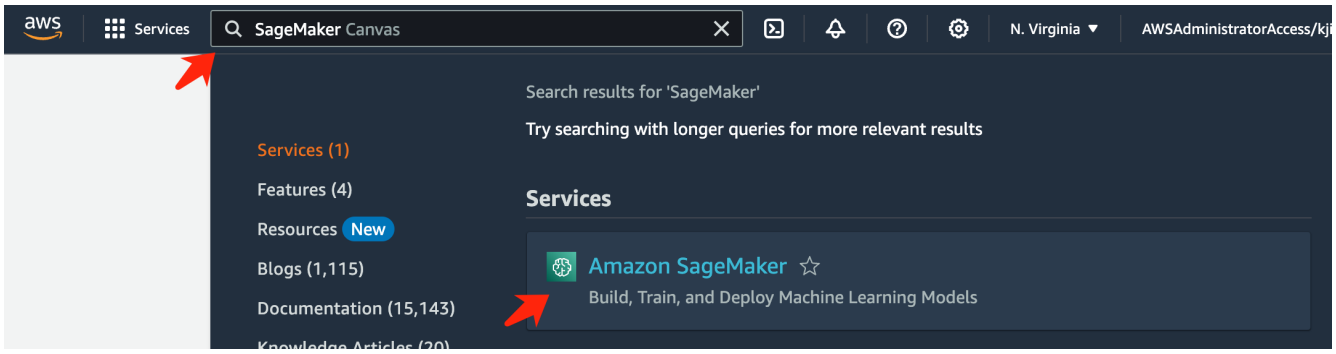
Weitere Informationen über FSxN finden Sie in dieser Präsentation ("Video-Link")

Benutzerhandbuch

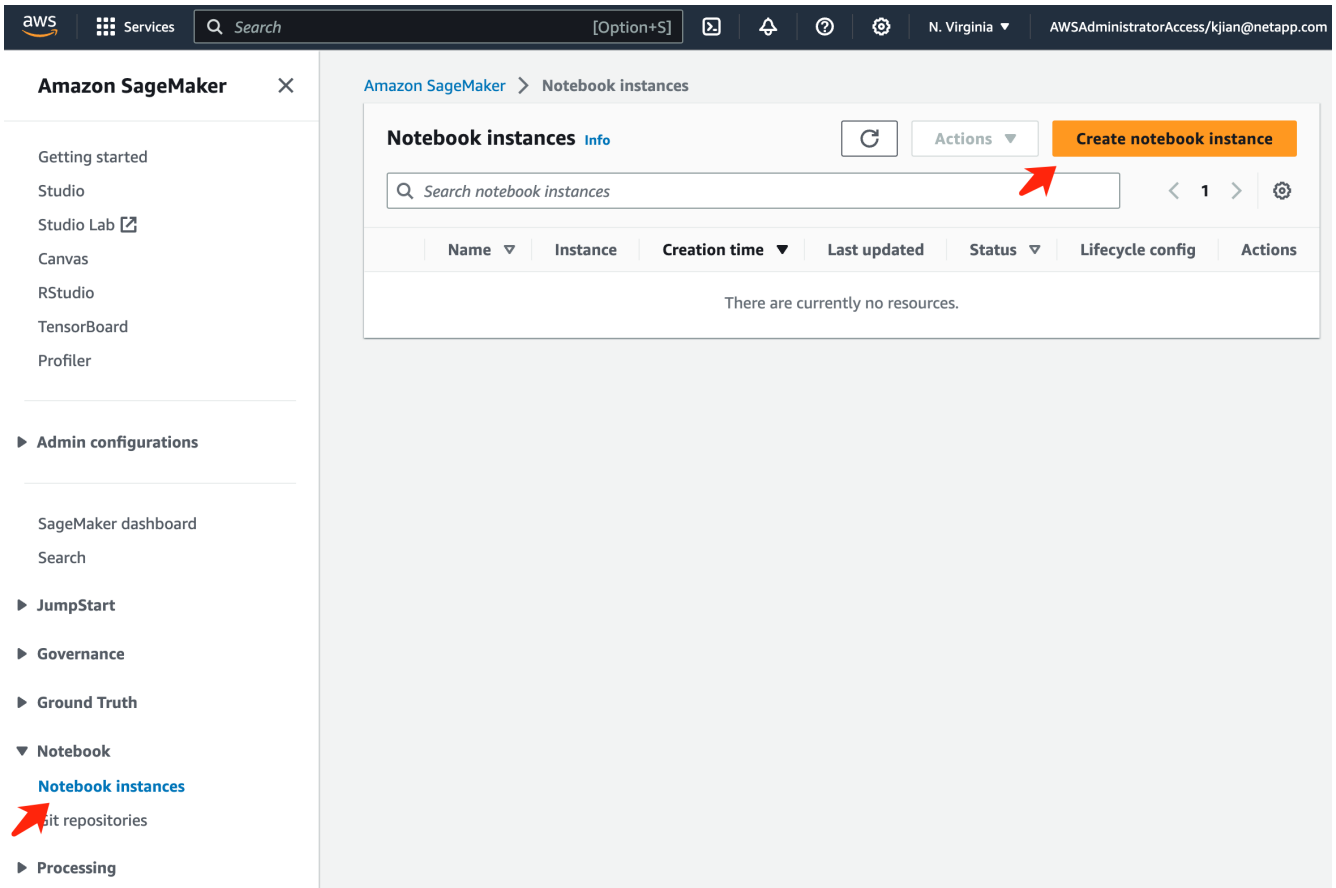
Server-Erstellung

Erstellen Sie eine SageMaker-Notebook-Instanz

1. Öffnen Sie die AWS-Konsole. Suchen Sie im Suchfeld nach SageMaker und klicken Sie auf den Dienst **Amazon SageMaker**.



2. Öffnen Sie die **Notizbuchinstanzen** unter der Registerkarte Notizbuch, klicken Sie auf die orangefarbene Schaltfläche **Notizbuchinstanz erstellen**.



3. Auf der Seite „Erstellung“
   Geben Sie den Namen der **Notebook-Instanz** ein
   Erweitern Sie das Fenster **Netzwerk**
   Lassen Sie andere Einträge standardmäßig und wählen Sie eine **VPC**, **Subnetz** und **Sicherheitsgruppe(n)** aus. (Diese **VPC** und **Subnetz** werden später zum Erstellen des FSxN-Dateisystems verwendet)
   Klicken Sie unten rechts auf den orangefarbenen Button **Notizbuchinstanz erstellen**.

Amazon SageMaker > Notebook instances > Create notebook instance

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more

Notebook instance settings

Notebook instance name

fsxn-demo

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type

ml.t3.medium

Elastic Inference Learn more

none

Platform identifier Learn more

Amazon Linux 2, Jupyter Lab 3

▶ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the **AmazonSageMakerFullAccess** IAM policy attached.

AmazonSageMakerServiceCatalogProductsUseRole

Create role using the role creation wizard

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook
  Lifecycle configurations always have root access

Encryption key - *optional*

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

▼ Network - *optional*

VPC - *optional*

Default vpc-0df3956ab1fca2ec9 (172.31.0.0/16)

Subnet

Choose a subnet in an availability zone supported by Amazon SageMaker.

subnet-00060df0d0f562672 (172.31.16.0/20) | us-east-1a

Security group(s)

sg-0a39b3985770e9256 (default) ✕

Direct internet access

- (●) Enable — Access the internet directly through Amazon SageMaker
- ( ) Disable — Access the internet through a VPC
  To train or host models from a notebook, you need internet access. To enable internet access, make sure that your VPC has a NAT gateway and your security group allows outbound connections. Learn more

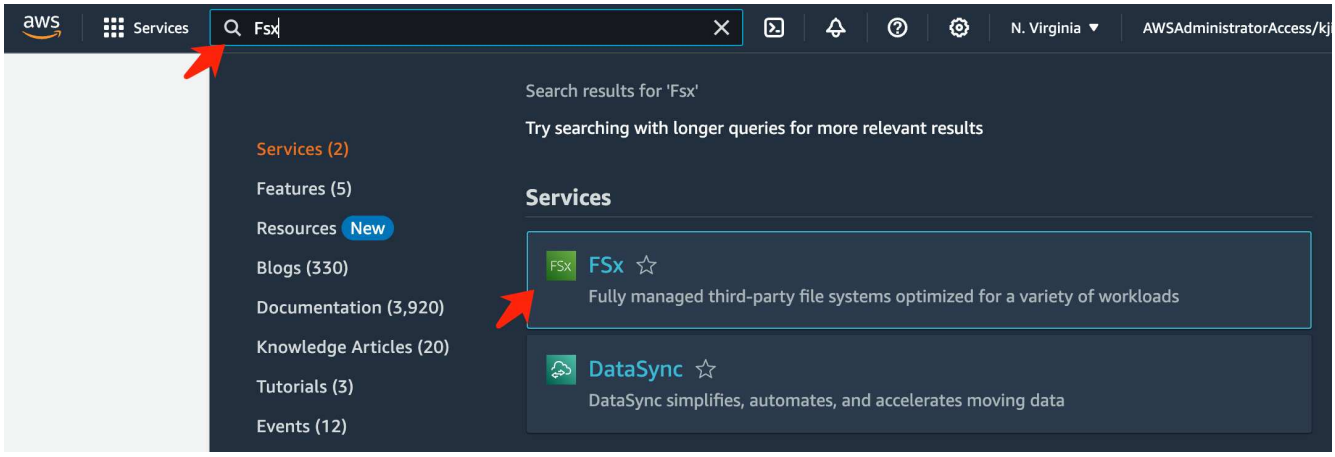
▶ Git repositories- *optional*

▶ Tags - *optional*

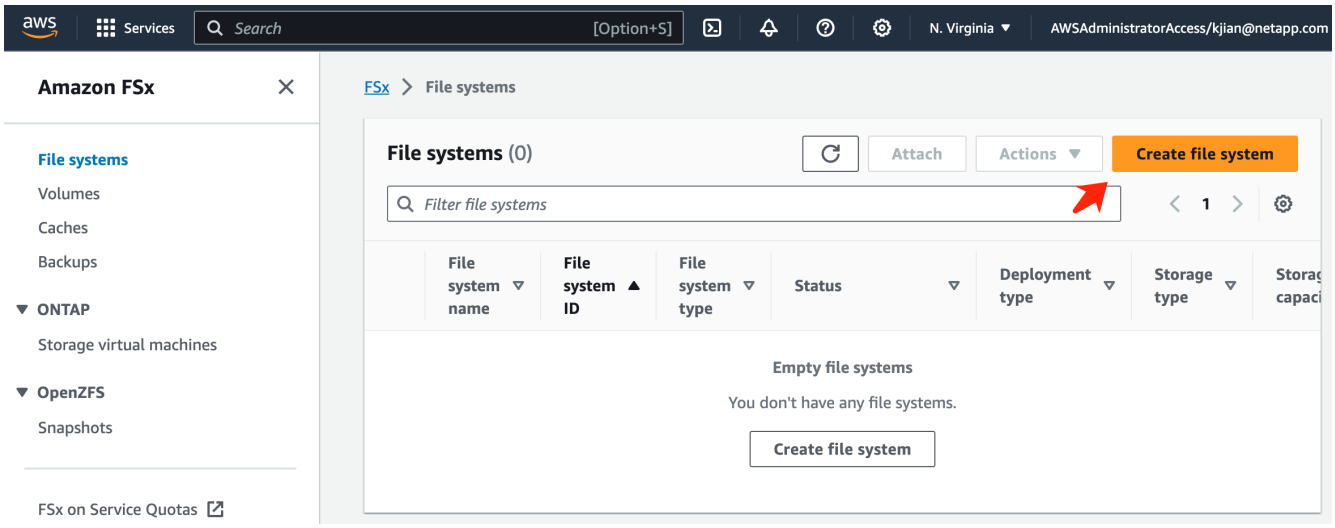
Cancel | Create notebook instance

Erstellen Sie ein FSxN-Dateisystem

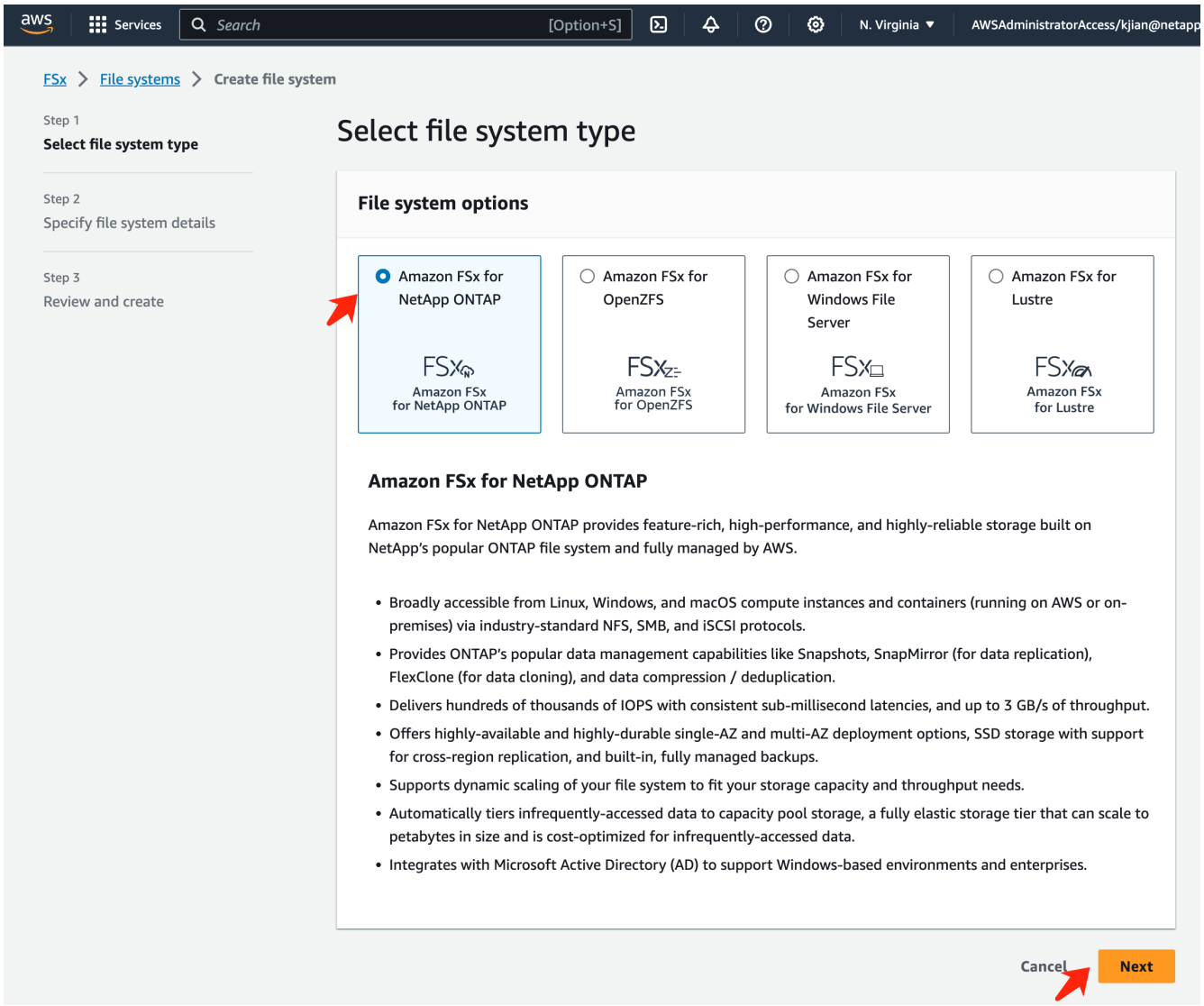
1. Öffnen Sie die AWS-Konsole. Suchen Sie im Suchfeld FSX und klicken Sie auf den Dienst **FSX**.



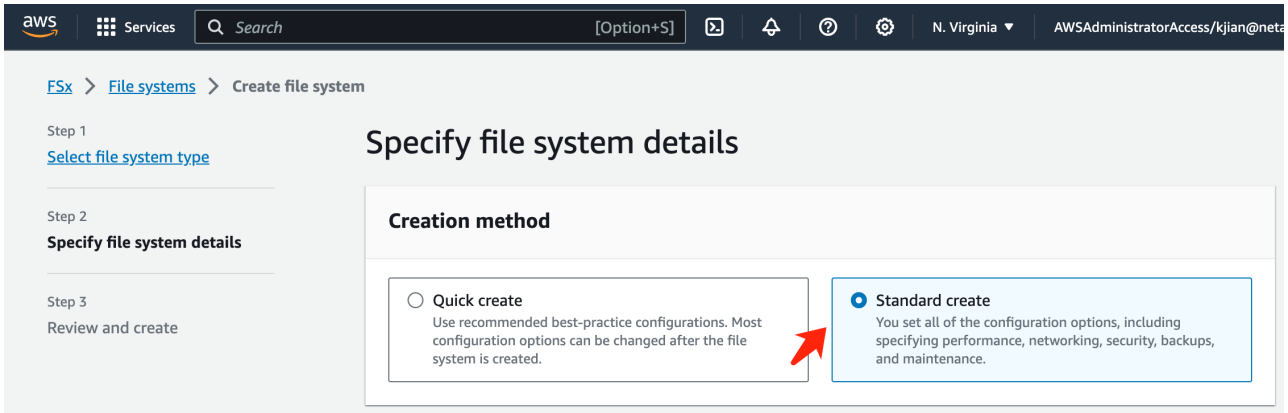
2. Klicken Sie auf **Dateisystem erstellen**.



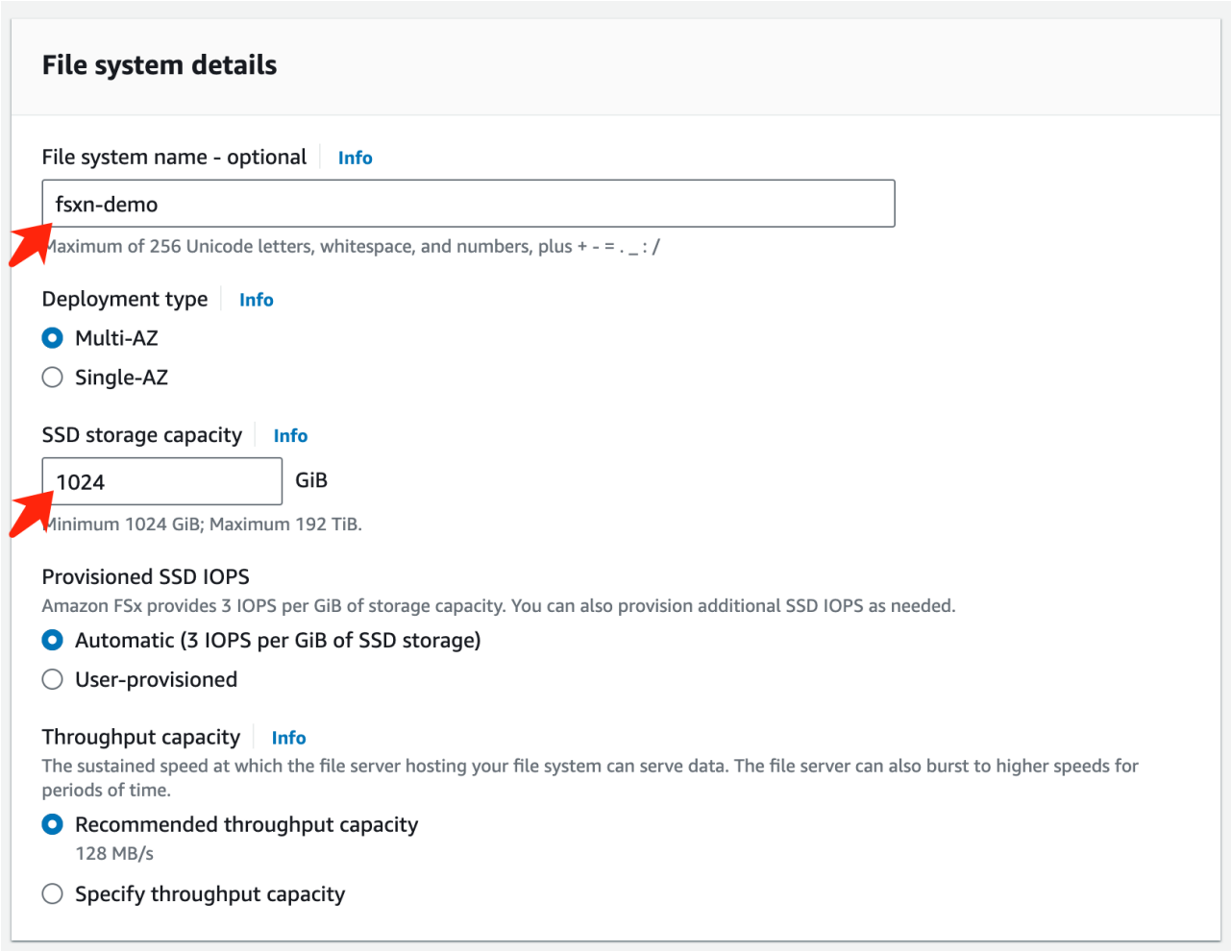
3. Wählen Sie die erste Karte **FSX für NetApp ONTAP** und klicken Sie auf **Weiter**.



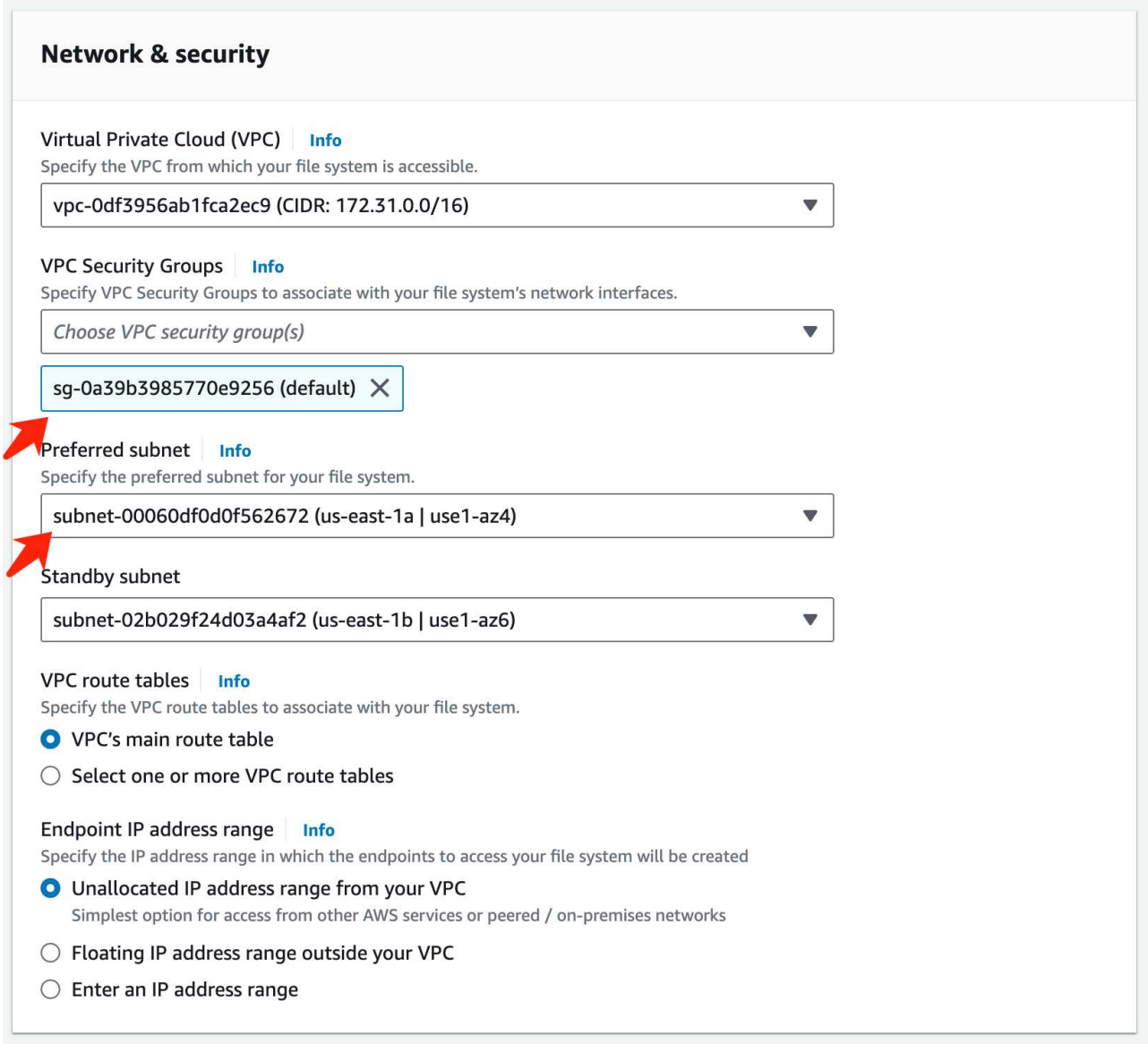
4. Auf der Konfigurationsseite für Details.
   a. Wählen Sie die Option **Standard create**.



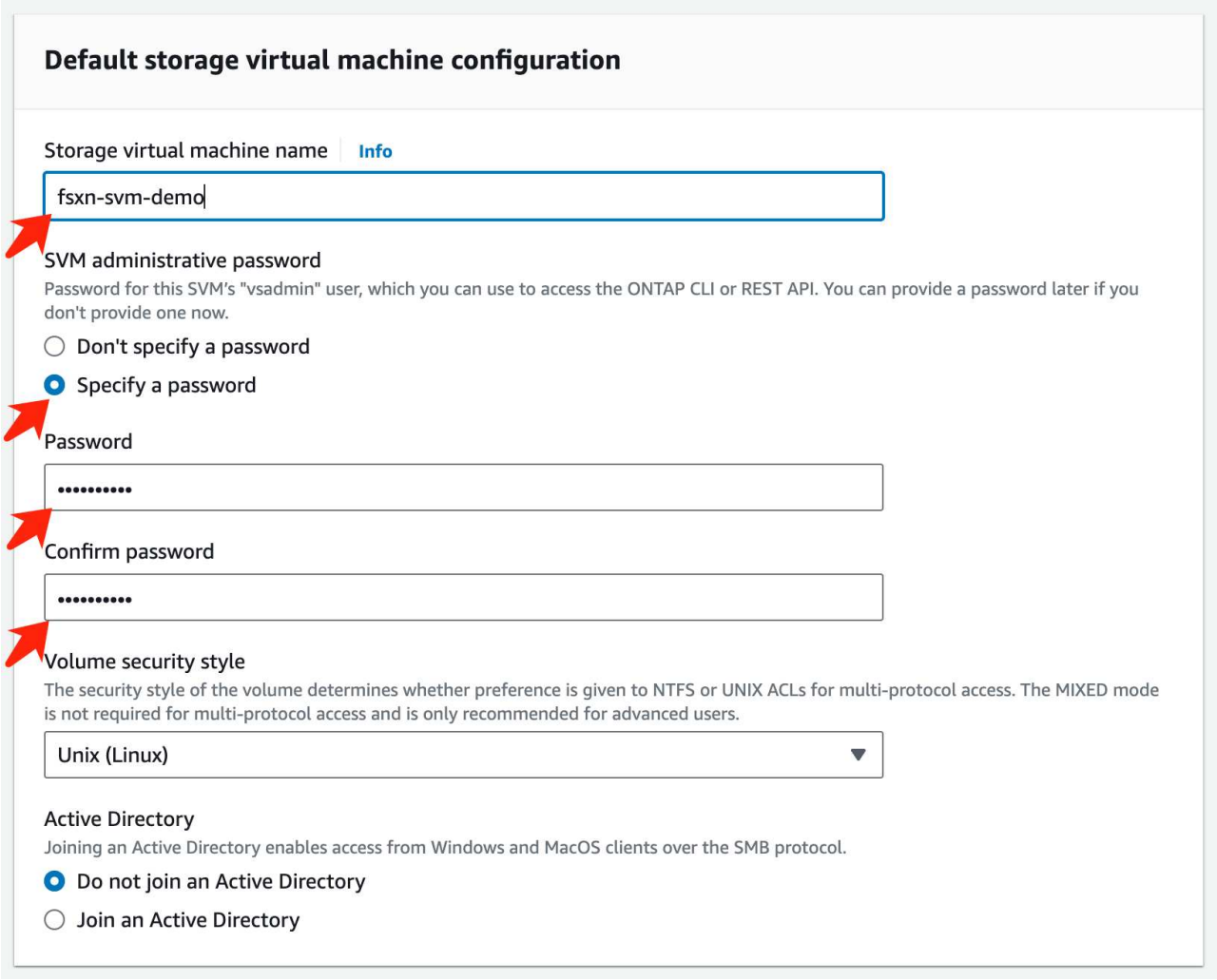
   b. Geben Sie den Namen des **Dateisystems** und die Kapazität des **SSD-Speichers** ein.



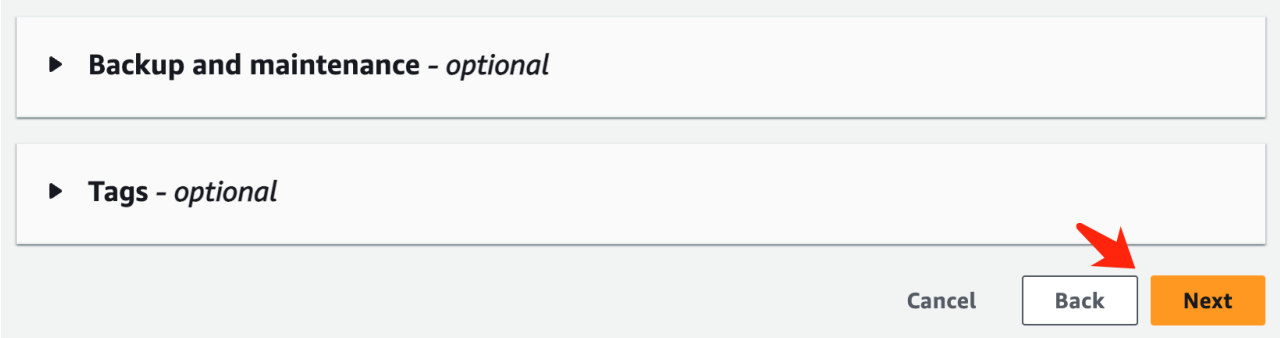
c. Stellen Sie sicher, dass Sie die **VPC** und **Subnetz** mit der **SageMaker Notebook**-Instanz identisch verwenden.



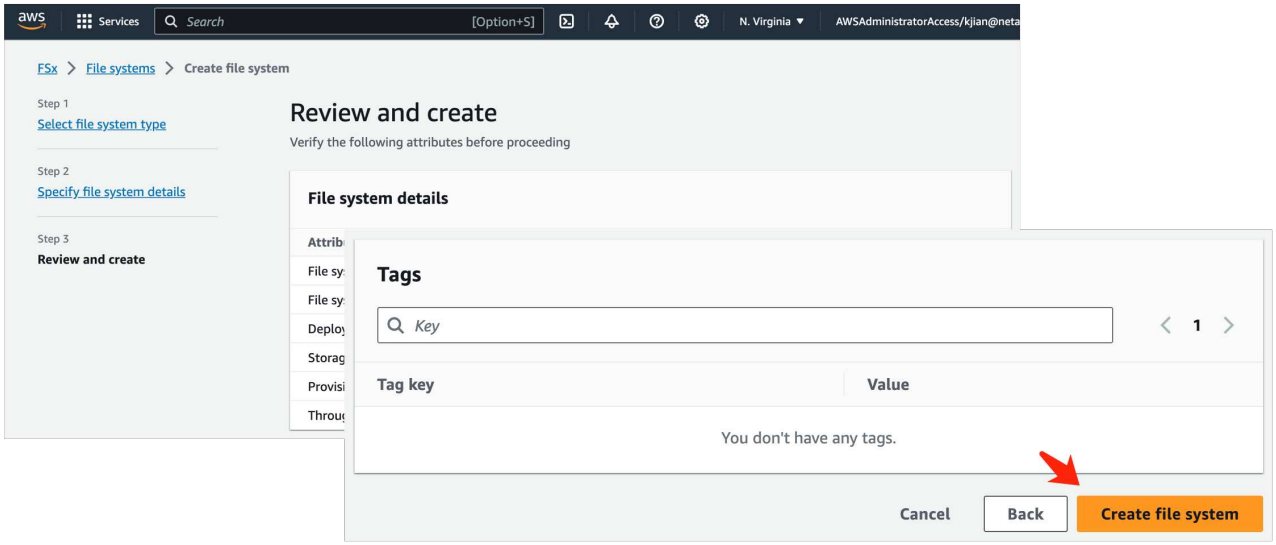
d. Geben Sie den Namen der **Storage Virtual Machine** und **Geben Sie ein Passwort** für Ihre SVM (Storage Virtual Machine) ein.



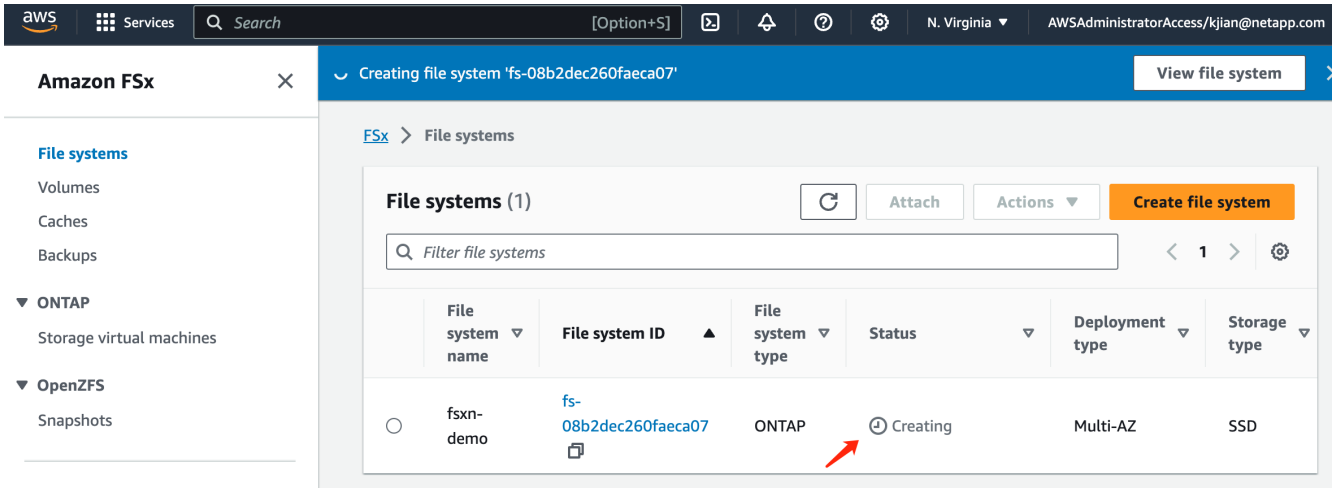
e. Lassen Sie andere Einträge standardmäßig und klicken Sie auf die orangefarbene Schaltfläche **Weiter** unten rechts.



f. Klicken Sie auf die orangefarbene Schaltfläche **Dateisystem erstellen** unten rechts auf der Überprüfungsseite.



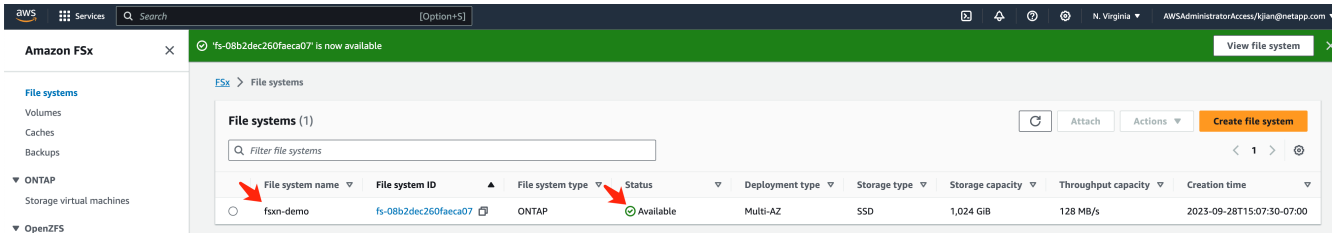
5. Es kann ungefähr 20-40 Minuten* dauern, um das FSX Dateisystem hochzudrehen.



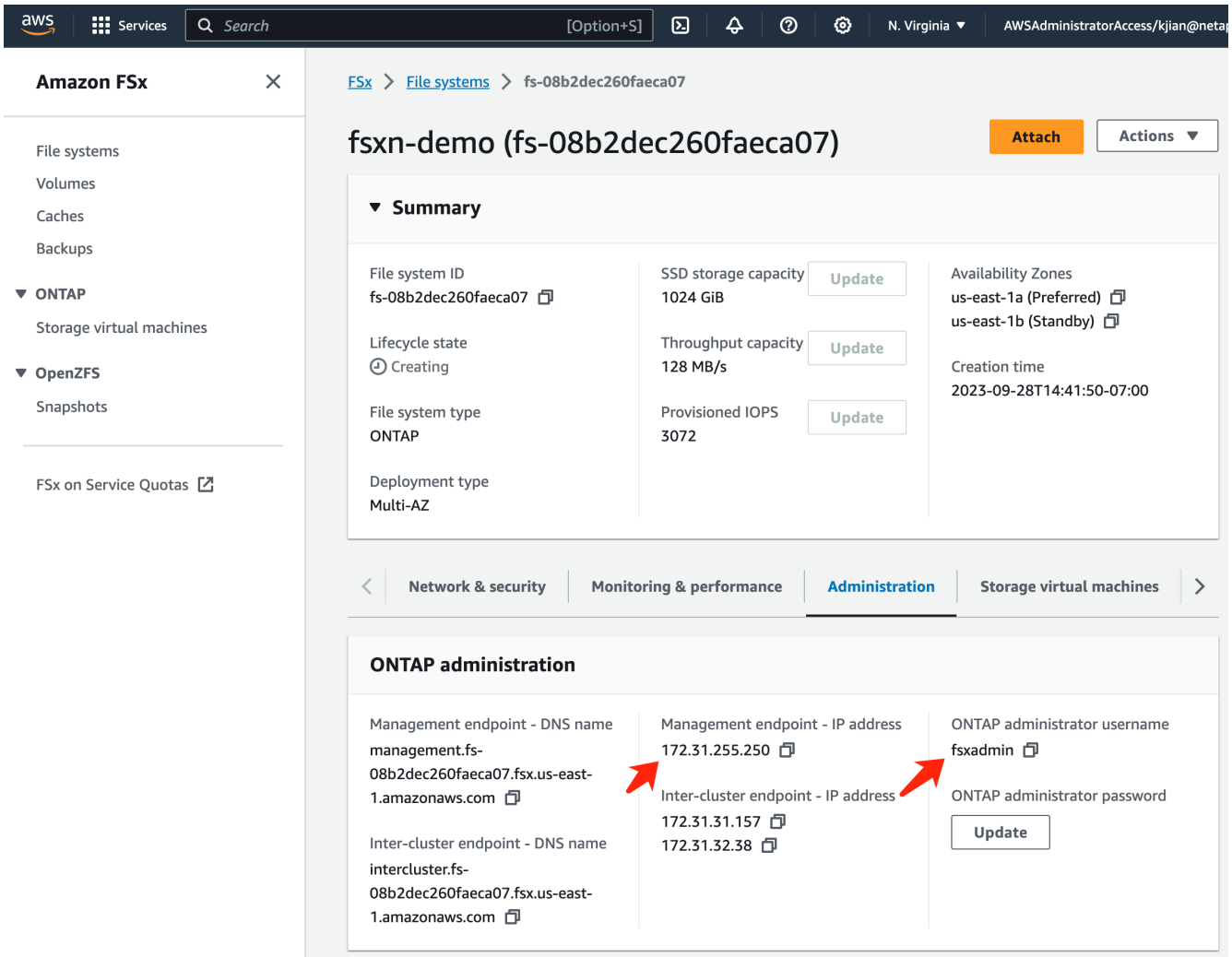
Serverkonfiguration

ONTAP-Konfiguration

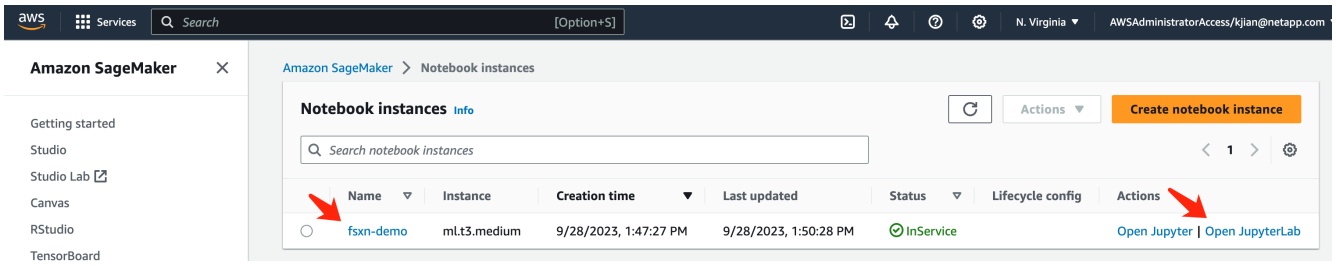
1. Öffnen Sie das erstellte FSX Dateisystem. Bitte stellen Sie sicher, dass der Status **verfügbar** ist.



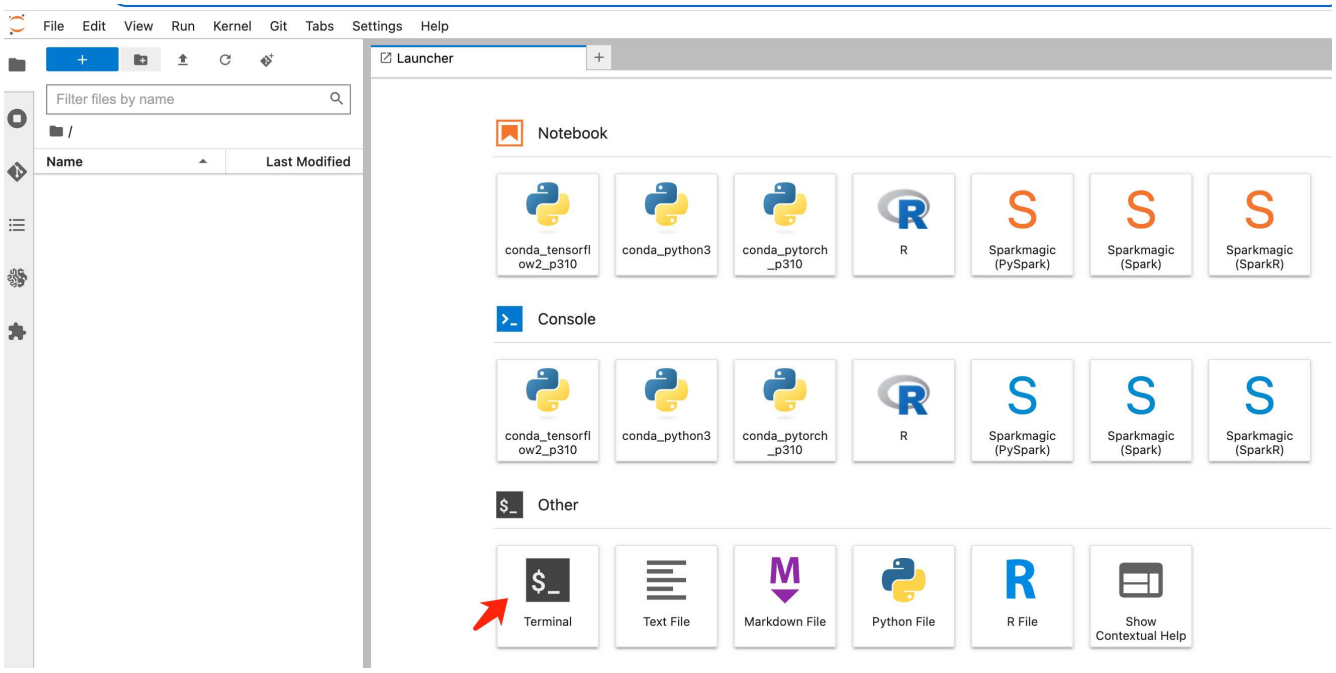
2. Wählen Sie die Registerkarte **Verwaltung** aus und behalten Sie die Optionen **Verwaltungsendpunkt - IP-Adresse** und **ONTAP-Administratorbenutzername** bei.



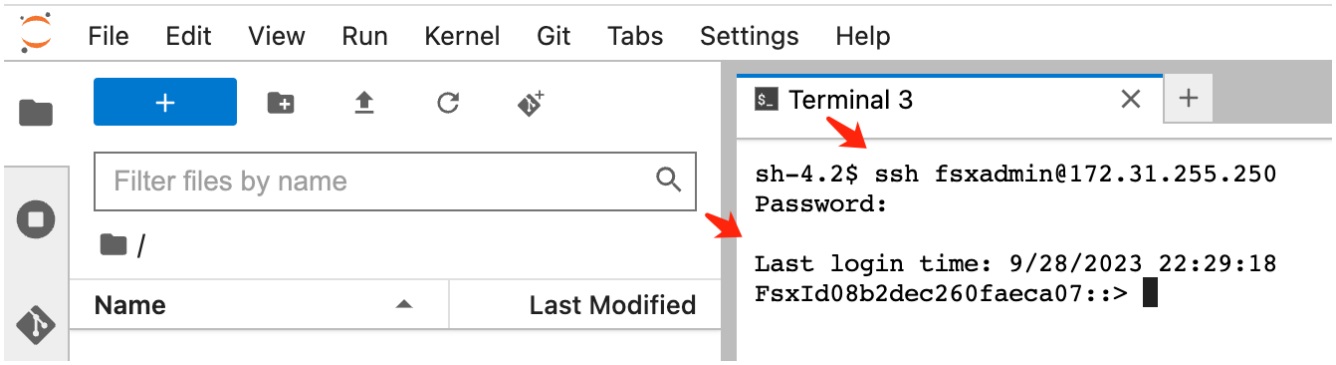
3. Öffnen Sie die erstellte **SageMaker Notebook-Instanz** und klicken Sie auf **JupyterLab öffnen**.



4. Öffnen Sie auf der Seite Jupyter Lab ein neues **Terminal**.

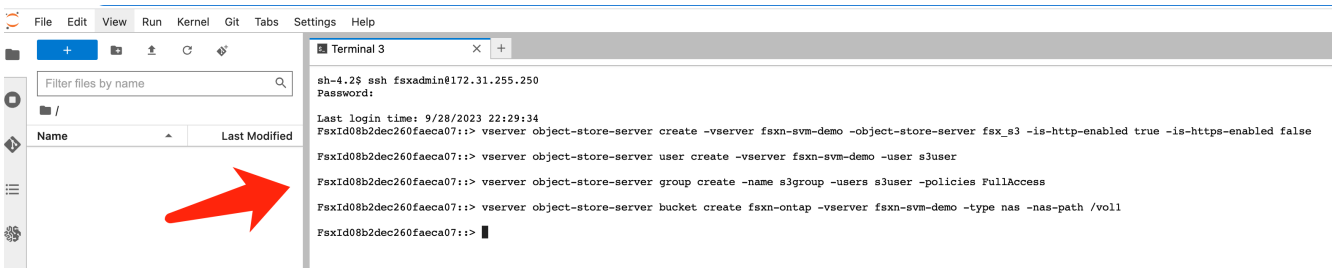


5. Geben Sie zum Anmelden beim FSxN ONTAP-Dateisystem den ssh-Befehl ssh@<admin user Name> <ONTAP Server IP> ein. (Benutzername und IP-Adresse werden aus Schritt 2 abgerufen)
   Bitte verwenden Sie das Passwort, das beim Erstellen der **Storage Virtual Machine** verwendet wird.



6. Führen Sie die Befehle in der folgenden Reihenfolge aus.
   Wir verwenden **fsxn-ontap** als Namen für den **FSxN privaten S3-Bucket-Namen**.
   Bitte verwenden Sie für das Argument **-vserver** den Namen der virtuellen Speichermaschine*.

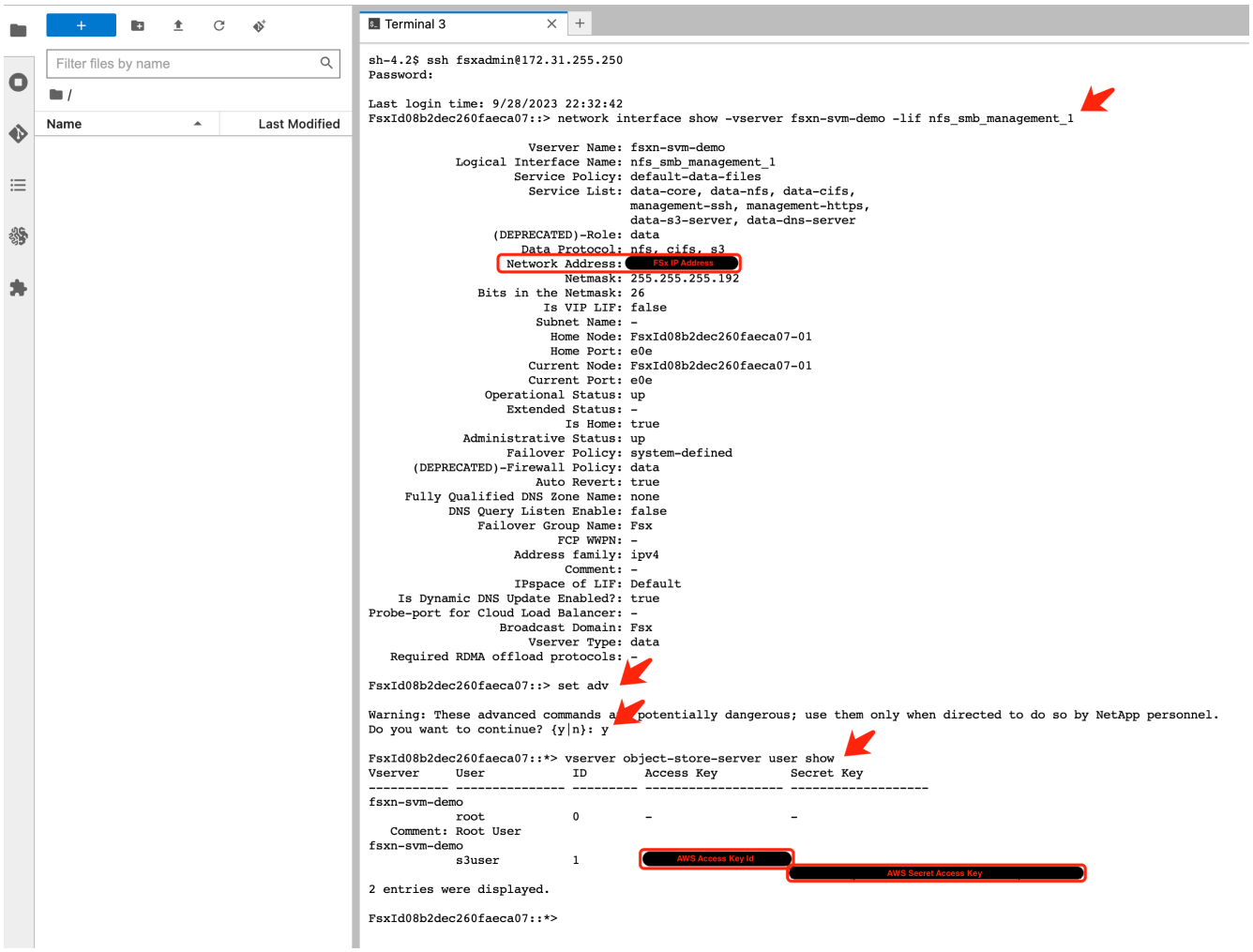
```
vserver object-store-server create -vserver fsxn-svm-demo -object-store
-server fsx_s3 -is-http-enabled true -is-https-enabled false

vserver object-store-server user create -vserver fsxn-svm-demo -user
s3user

vserver object-store-server group create -name s3group -users s3user
-policies FullAccess

vserver object-store-server bucket create fsxn-ontap -vserver fsxn-svm-
demo -type nas -nas-path /vol1
```



7. Führen Sie die folgenden Befehle aus, um die Endpunkt-IP und die Zugangsdaten für FSxN private S3 abzurufen.

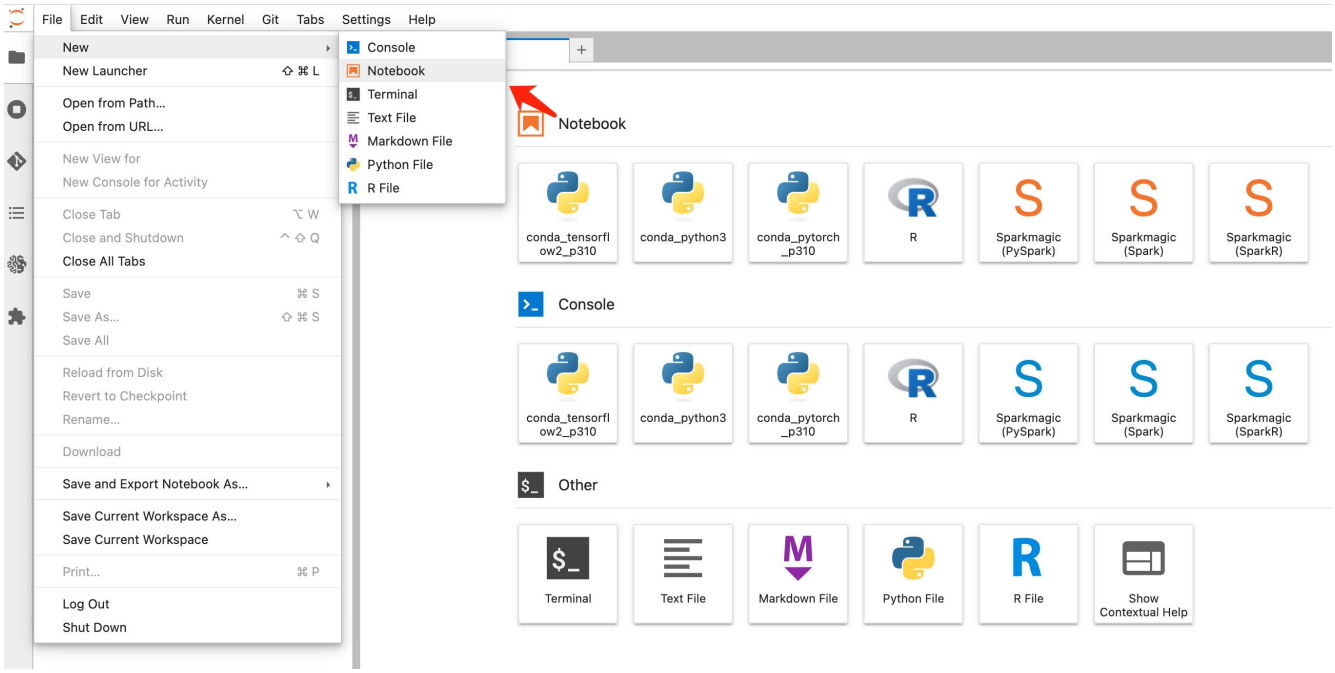
```
network interface show -vserver fsxn-svm-demo -lif nfs_smb_management_1

set adv

vserver object-store-server user show
```

8. Die IP-Adresse des Endpunkts und die Anmeldeinformationen bleiben für die zukünftige Verwendung erhalten.



Client-Konfiguration

1. Erstellen Sie in der SageMaker-Notebook-Instanz ein neues Jupyter-Notebook.



2. Verwenden Sie den unten stehenden Code als Umgehung, um Dateien in einen FSxN privaten S3-Bucket hochzuladen.
   Ein umfangreiches Codebeispiel finden Sie in diesem Notizbuch.
   "Fsxn_Demo.ipynb"

```
# Setup configurations
# -------- Manual configurations --------
seed: int = 77                                                  # Random
seed
bucket_name: str = 'fsxn-ontap'                                 # The bucket
name in ONTAP
aws_access_key_id = '<Your ONTAP bucket key id>'                # Please get
this credential from ONTAP
aws_secret_access_key = '<Your ONTAP bucket access key>'        # Please get
this credential from ONTAP
fsx_endpoint_ip: str = '<Your FSxN IP address>'                 # Please get
this IP address from FSXN
# -------- Manual configurations --------

# Workaround
## Permission patch
!mkdir -p vol1
!sudo mount -t nfs $fsx_endpoint_ip:/vol1 /home/ec2-user/SageMaker/vol1
!sudo chmod 777 /home/ec2-user/SageMaker/vol1

## Authentication for FSxN as a Private S3 Bucket
```

```
!aws configure set aws_access_key_id $aws_access_key_id
!aws configure set aws_secret_access_key $aws_secret_access_key

## Upload file to the FSxN Private S3 Bucket
%%capture
local_file_path: str = <Your local file path>

!aws s3 cp --endpoint-url http://$fsx_endpoint_ip /home/ec2-user
/SageMaker/$local_file_path  s3://$bucket_name/$local_file_path

# Read data from FSxN Private S3 bucket
## Initialize a s3 resource client
import boto3

# Get session info
region_name = boto3.session.Session().region_name

# Initialize Fsxn S3 bucket object
# --- Start integrating SageMaker with FSXN ---
# This is the only code change we need to incorporate SageMaker with
FSXN
s3_client: boto3.client = boto3.resource(
    's3',
    region_name=region_name,
    aws_access_key_id=aws_access_key_id,
    aws_secret_access_key=aws_secret_access_key,
    use_ssl=False,
    endpoint_url=f'http://{fsx_endpoint_ip}',
    config=boto3.session.Config(
        signature_version='s3v4',
        s3={'addressing_style': 'path'}
    )
)
# --- End integrating SageMaker with FSXN ---

## Read file byte content
bucket = s3_client.Bucket(bucket_name)

binary_data = bucket.Object(data.filename).get()['Body']
```

Damit ist die Integration zwischen FSxN und der SageMaker-Instanz abgeschlossen.

Nützliche Debugging-Checkliste

- Stellen Sie sicher, dass sich die SageMaker-Notebook-Instanz und das FSxN-Dateisystem im gleichen VPC befinden.

- Denken Sie daran, den Befehl **set dev** auf ONTAP auszuführen, um die Berechtigungsebene auf **dev** zu setzen.

FAQ (Stand 27. September 2023)

F: Warum erhalte ich den Fehler "**ein Fehler ist aufgetreten (NotImplementiert) beim Aufruf der CreateMultipartUpload Operation: Der von Ihnen angeforderte s3 Befehl ist nicht implementiert**" beim Hochladen von Dateien in FSxN?

A: Als privater S3-Bucket unterstützt FSxN das Hochladen von Dateien mit bis zu 100 MB. Bei Verwendung des S3-Protokolls werden Dateien mit einer Größe von mehr als 100 MB in 100-MB-Blöcke unterteilt, und die Funktion „CreateMultipartUpload“ wird aufgerufen. Die aktuelle Implementierung von FSxN private S3 unterstützt diese Funktion jedoch nicht.

F: Warum erhalte ich den Fehler "**ein Fehler ist aufgetreten (AccessDenied) beim Aufruf der PutObject-Operationen: Zugriff verweigert**" beim Hochladen von Dateien in FSxN?

A: Um von einer SageMaker-Notebook-Instanz auf den privaten FSxN-S3-Bucket zuzugreifen, wechseln Sie die AWS-Anmeldeinformationen zu den FSxN-Anmeldeinformationen. Die Gewährung von Schreibberechtigungen für die Instanz erfordert jedoch eine Problemumgehungslösung, bei der der Bucket gemountet und der Shell-Befehl 'chmod' ausgeführt wird, um die Berechtigungen zu ändern.

F: Wie kann ich den FSxN privaten S3-Eimer mit anderen SageMaker ML-Diensten integrieren?

A: Leider bietet das SageMaker Services SDK keine Möglichkeit, den Endpunkt für den privaten S3-Bucket anzugeben. Daher ist FSxN S3 nicht kompatibel mit SageMaker-Diensten wie SageMaker Data Wrangler, SageMaker Clarify, SageMaker Glue, SageMaker Athena, SageMaker AutoML, und andere.

Teil 2 – Nutzung von AWS FSX for NetApp ONTAP (FSxN) als Datenquelle für das Modelltraining in SageMaker

Autor(en):
Jian Jian (Ken), Senior Data & Applied Scientist, NetApp

Einführung

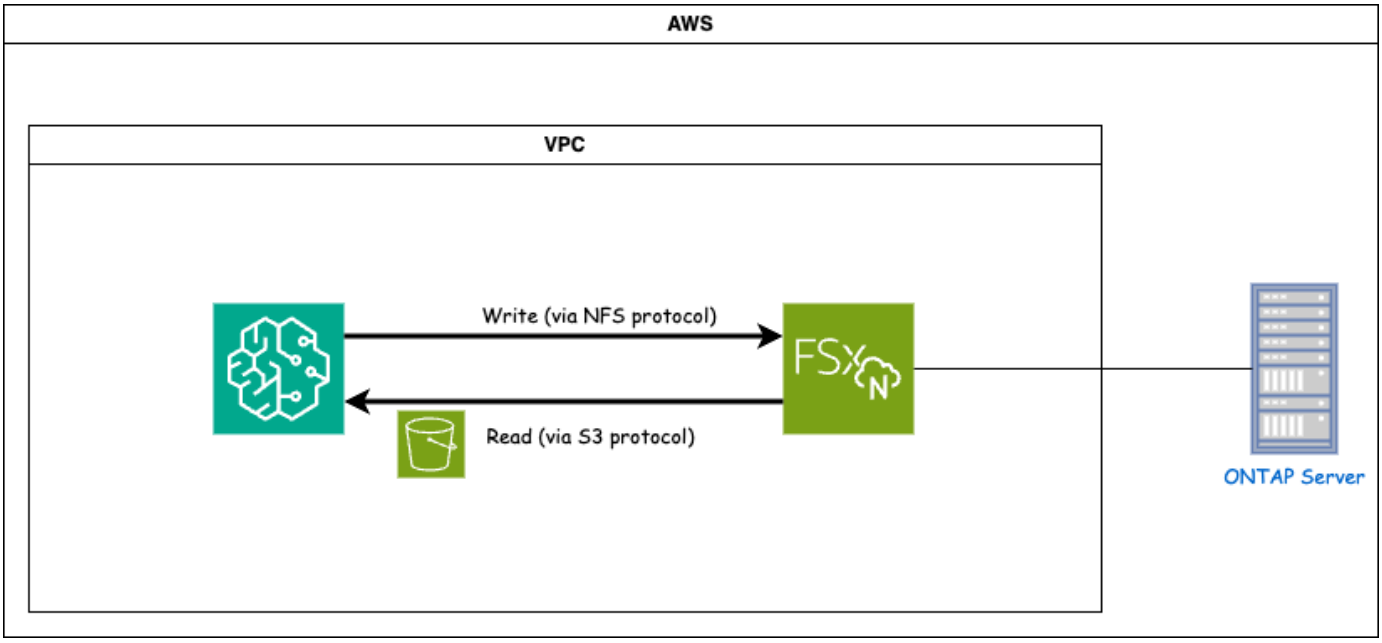
Dieses Tutorial bietet ein praktisches Beispiel für ein Computer-Vision-Klassifizierungsprojekt und bietet praktische Erfahrungen beim Aufbau von ML-Modellen, die FSxN als Datenquelle in der SageMaker-Umgebung verwenden. Der Schwerpunkt des Projekts liegt auf der Verwendung von PyTorch, einem Deep-Learning-Framework, um die Reifenqualität anhand von Reifenbildern zu klassifizieren. Er betont die Entwicklung von Machine-Learning-Modellen, die FSxN als Datenquelle in Amazon SageMaker verwenden.

Was ist FSxN

Amazon FSX for NetApp ONTAP ist eine wirklich vollständig gemanagte Storage-Lösung von AWS. Es nutzt das ONTAP Filesystem von NetApp, um zuverlässigen und hochperformanten Storage bereitzustellen. Durch die Unterstützung von Protokollen wie NFS, SMB und iSCSI ermöglicht sie einen nahtlosen Zugriff von verschiedenen Computing-Instanzen und Containern. Der Service bietet eine außergewöhnliche Performance, die schnelle und effiziente Datenvorgänge ermöglicht. Es bietet zudem eine hohe Verfügbarkeit und Langlebigkeit, wodurch sichergestellt wird, dass die Daten zugänglich und geschützt bleiben. Darüber hinaus ist die Storage-Kapazität von Amazon FSX for NetApp ONTAP skalierbar, sodass Sie sie einfach an Ihre Anforderungen anpassen können.

Voraussetzung

Netzwerkumgebung



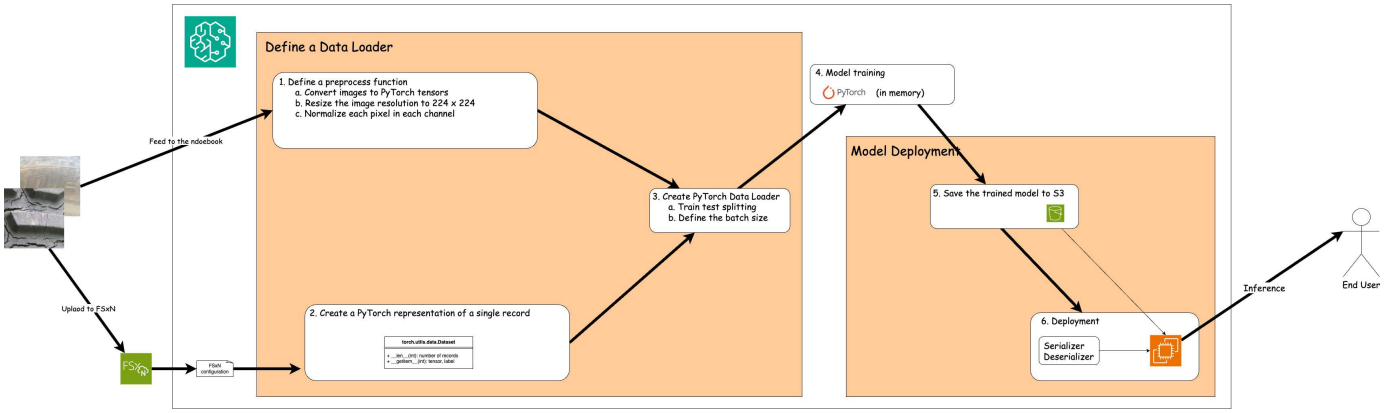
FSxN (Amazon FSX for NetApp ONTAP) ist ein AWS Storage-Service. Sie umfasst ein Filesystem, das auf dem NetApp ONTAP System ausgeführt wird, und eine AWS-gemanagte System Virtual Machine (SVM), die eine Verbindung mit ihr herstellt. Im angegebenen Diagramm befindet sich der von AWS gemanagte NetApp ONTAP-Server außerhalb der VPC. Die SVM dient als Vermittler zwischen SageMaker und dem NetApp ONTAP-System, empfängt Betriebsanforderungen von SageMaker und leitet sie an den zugrunde liegenden Speicher weiter. Für den Zugriff auf FSxN muss SageMaker innerhalb derselben VPC wie die FSxN-Bereitstellung platziert werden. Diese Konfiguration gewährleistet die Kommunikation und den Datenzugriff zwischen SageMaker und FSxN.

Datenzugriff

In realen Szenarien verwenden Data Scientists in der Regel die in FSxN gespeicherten Daten, um ihre Modelle für maschinelles Lernen zu erstellen. Da das FSxN-Dateisystem nach der Erstellung jedoch zunächst leer ist, müssen die Trainingsdaten für Demonstrationszwecke manuell hochgeladen werden. Dies kann erreicht werden, indem FSxN als Volumen auf SageMaker montiert wird. Nachdem das Dateisystem erfolgreich gemountet wurde, können Sie den Datensatz an den gemounteten Speicherort hochladen, sodass er für das Training Ihrer Modelle in der SageMaker-Umgebung verfügbar ist. Dieser Ansatz ermöglicht es Ihnen, die Speicherkapazität und Funktionen von FSxN zu nutzen, während Sie mit SageMaker für die Modellentwicklung und das Training arbeiten.

Beim Lesen von Daten wird FSxN als privater S3-Bucket konfiguriert. Weitere Informationen zur Konfiguration finden Sie unter "Teil 1 – Integration von AWS FSX for NetApp ONTAP (FSxN) als privater S3-Bucket in AWS SageMaker"

Integrationsübersicht



Der Workflow, in dem Trainingsdaten in FSxN für den Aufbau eines Deep-Learning-Modells in SageMaker verwendet werden, lässt sich in drei Hauptschritte zusammenfassen: Definition des Datenladers, Modelltraining und Bereitstellung. Auf hoher Ebene bilden diese Schritte die Grundlage einer MLOPS-Pipeline. Jeder Schritt umfasst jedoch mehrere detaillierte Teilschritte für eine umfassende Implementierung. Diese Teilschritte umfassen verschiedene Aufgaben wie Datenvorverarbeitung, Dataset-Splitting, Modellkonfiguration, Hyperparameter-Tuning, Modellbewertung, und Modellimplementierung. Diese Schritte gewährleisten einen gründlichen und effektiven Prozess für die Erstellung und Implementierung von Deep-Learning-Modellen unter Verwendung von Trainingsdaten von FSxN in der SageMaker-Umgebung.

Schritt-für-Schritt-Integration

Datenlader

Um ein PyTorch Deep-Learning-Netzwerk mit Daten zu trainieren, wird ein Datenlader erstellt, um die Dateneinspeisung zu erleichtern. Der Data Loader definiert nicht nur die Batch-Größe, sondern bestimmt auch den Ablauf zum Lesen und Vorverarbeiten jedes Datensatzes innerhalb des Stapels. Durch die Konfiguration des Data Loader können wir die Verarbeitung von Daten in Batches übernehmen und so das Training des Deep-Learning-Netzwerks ermöglichen.

Der Datenlader besteht aus 3 Teilen.

Vorverarbeitungsfunktion

```
from torchvision import transforms

preprocess = transforms.Compose([
    transforms.ToTensor(),
    transforms.Resize((224,224)),
    transforms.Normalize(
        mean=[0.485, 0.456, 0.406],
        std=[0.229, 0.224, 0.225]
    )
])
```

Der oben genannte Code-Snippet demonstriert die Definition von Bildvorverarbeitungstransformationen mit dem Modul **torchvision.transforms**. In dieser turktoriellen Funktion wird das vorprozessende Objekt erstellt, um eine Reihe von Transformationen anzuwenden. Erstens wandelt die **ToTensor()** Transformation das Bild in eine Tensor-Darstellung um. Anschließend passt die Umwandlung **Resize 224,224** das Bild auf eine feste

Größe von 24x224 Pixeln an. Schließlich normalisiert die **Normalisieren()** Transformation die Tensor-Werte, indem sie den Mittelwert subtrahiert und durch die Standardabweichung entlang jedes Kanals dividiert. Die für die Normalisierung verwendeten Werte für Mittelwert und Standardabweichung werden häufig in vortrainierten neuronalen Netzmodellen verwendet. Insgesamt bereitet dieser Code die Bilddaten für die weitere Verarbeitung oder Eingabe in ein vortrainiertes Modell vor, indem er sie in einen Tensor konvertiert, die Größe verändert und die Pixelwerte normalisiert.

Die PyTorch-Dataset-Klasse

```
import torch
from io import BytesIO
from PIL import Image


class FSxNImageDataset(torch.utils.data.Dataset):
    def __init__(self, bucket, prefix='', preprocess=None):
        self.image_keys = [
            s3_obj.key
            for s3_obj in list(bucket.objects.filter(Prefix=prefix).all())
        ]
        self.preprocess = preprocess

    def __len__(self):
        return len(self.image_keys)

    def __getitem__(self, index):
        key = self.image_keys[index]
        response = bucket.Object(key)

        label = 1 if key[13:].startswith('defective') else 0

        image_bytes = response.get()['Body'].read()
        image = Image.open(BytesIO(image_bytes))
        if image.mode == 'L':
            image = image.convert('RGB')

        if self.preprocess is not None:
            image = self.preprocess(image)
        return image, label
```

Diese Klasse bietet Funktionen zum Abrufen der Gesamtzahl der Datensätze im Datensatz und definiert die Methode zum Lesen von Daten für jeden Datensatz. Innerhalb der Funktion ***getitem*** verwendet der Code das Bucket-Objekt boto3 S3, um die Binärdaten aus FSxN abzurufen. Der Code-Stil für den Zugriff auf Daten aus FSxN ähnelt dem Lesen von Daten aus Amazon S3. Die nachfolgende Erklärung geht auf den Erstellungsprozess des privaten S3-Objekts **bucket** ein.

FSxN als privates S3-Repository

```
seed = 77                                                    # Random seed
bucket_name = '<Your ONTAP bucket name>'                     # The bucket
name in ONTAP
aws_access_key_id = '<Your ONTAP bucket key id>'             # Please get
this credential from ONTAP
aws_secret_access_key = '<Your ONTAP bucket access key>'     # Please get
this credential from ONTAP
fsx_endpoint_ip = '<Your FSxN IP address>'                   # Please get
this IP address from FSXN
```

```
import boto3

# Get session info
region_name = boto3.session.Session().region_name

# Initialize Fsxn S3 bucket object
# --- Start integrating SageMaker with FSXN ---
# This is the only code change we need to incorporate SageMaker with FSXN
s3_client: boto3.client = boto3.resource(
    's3',
    region_name=region_name,
    aws_access_key_id=aws_access_key_id,
    aws_secret_access_key=aws_secret_access_key,
    use_ssl=False,
    endpoint_url=f'http://{fsx_endpoint_ip}',
    config=boto3.session.Config(
        signature_version='s3v4',
        s3={'addressing_style': 'path'}
    )
)
# s3_client = boto3.resource('s3')
bucket = s3_client.Bucket(bucket_name)
# --- End integrating SageMaker with FSXN ---
```

Um Daten aus FSxN in SageMaker zu lesen, wird ein Handler erstellt, der auf den FSxN-Speicher mit dem S3-Protokoll verweist. Auf diese Weise kann FSxN als privater S3-Bucket behandelt werden. Die Handler-Konfiguration umfasst die Angabe der IP-Adresse der FSxN SVM, des Bucket-Namens und der erforderlichen Anmeldedaten. Eine umfassende Erklärung zum Bezug dieser Konfigurationselemente finden Sie in dem Dokument unter "Teil 1 – Integration von AWS FSX for NetApp ONTAP (FSxN) als privater S3-Bucket in AWS SageMaker".

In dem oben genannten Beispiel wird das Bucket-Objekt verwendet, um das PyTorch-Datensatzobjekt zu instanziieren. Das Datensatzobjekt wird im nachfolgenden Abschnitt näher erläutert.

Der PyTorch Data Loader

```
from torch.utils.data import DataLoader
torch.manual_seed(seed)

# 1. Hyperparameters
batch_size = 64

# 2. Preparing for the dataset
dataset = FSxNImageDataset(bucket, 'dataset/tyre', preprocess=preprocess)

train, test = torch.utils.data.random_split(dataset, [1500, 356])

data_loader = DataLoader(dataset, batch_size=batch_size, shuffle=True)
```

Im angegebenen Beispiel wird eine Batch-Größe von 64 angegeben, was darauf hinweist, dass jeder Batch 64 Datensätze enthält. Durch die Kombination der PyTorch **Datensatz** Klasse, der Vorverarbeitungsfunktion und der Training Batch Größe erhalten wir den Data Loader für das Training. Dieser Daten-Loader erleichtert den Prozess, den Datensatz während der Trainingsphase in Batches zu durchlaufen.

Modelltraining

```
from torch import nn


class TyreQualityClassifier(nn.Module):
    def __init__(self):
        super().__init__()
        self.model = nn.Sequential(
            nn.Conv2d(3,32,(3,3)),
            nn.ReLU(),
            nn.Conv2d(32,32,(3,3)),
            nn.ReLU(),
            nn.Conv2d(32,64,(3,3)),
            nn.ReLU(),
            nn.Flatten(),
            nn.Linear(64*(224-6)*(224-6),2)
        )
    def forward(self, x):
        return self.model(x)
```

```
import datetime

num_epochs = 2
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

model = TyreQualityClassifier()
fn_loss = torch.nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=1e-3)


model.to(device)
for epoch in range(num_epochs):
    for idx, (X, y) in enumerate(data_loader):
        X = X.to(device)
        y = y.to(device)

        y_hat = model(X)

        loss = fn_loss(y_hat, y)
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
        current_time = datetime.datetime.now().strftime("%Y-%m-%d
%H:%M:%S")
        print(f"Current Time: {current_time} - Epoch [{epoch+1}/
{num_epochs}]- Batch [{idx + 1}] - Loss: {loss}", end='\r')
```

Dieser Code implementiert einen standardmäßigen PyTorch-Trainingsprozess. Es definiert ein neuronales Netzmodell mit dem Namen **TireQualityClassifier**, das konvolutionelle Schichten und eine lineare Schicht verwendet, um die Reifenqualität zu klassifizieren. Die Trainingsschleife iteriert Daten-Batches, berechnet den Verlust und aktualisiert die Parameter des Modells mittels Rückverbreitung und Optimierung. Außerdem werden die aktuelle Zeit, die aktuelle Epoche, der Stapel und der Verlust für Überwachungszwecke gedruckt.

Modellbereitstellung

Einsatz

```
import io
import os
import tarfile
import sagemaker

# 1. Save the PyTorch model to memory
buffer_model = io.BytesIO()
traced_model = torch.jit.script(model)
torch.jit.save(traced_model, buffer_model)

# 2. Upload to AWS S3
sagemaker_session = sagemaker.Session()
bucket_name_default = sagemaker_session.default_bucket()
model_name = f'tyre_quality_classifier.pth'

# 2.1. Zip PyTorch model into tar.gz file
buffer_zip = io.BytesIO()
with tarfile.open(fileobj=buffer_zip, mode="w:gz") as tar:
    # Add PyTorch pt file
    file_name = os.path.basename(model_name)
    file_name_with_extension = os.path.split(file_name)[-1]
    tarinfo = tarfile.TarInfo(file_name_with_extension)
    tarinfo.size = len(buffer_model.getbuffer())
    buffer_model.seek(0)
    tar.addfile(tarinfo, buffer_model)

# 2.2. Upload the tar.gz file to S3 bucket
buffer_zip.seek(0)
boto3.resource('s3') \
    .Bucket(bucket_name_default) \
    .Object(f'pytorch/{model_name}.tar.gz') \
    .put(Body=buffer_zip.getvalue())
```

Der Code speichert das PyTorch-Modell in **Amazon S3**, da SageMaker das Modell für die Bereitstellung in S3 speichern muss. Durch das Hochladen des Modells auf **Amazon S3** wird es für SageMaker zugänglich, was die Bereitstellung und Inferenz auf dem bereitgestellten Modell ermöglicht.

```
import time
from sagemaker.pytorch import PyTorchModel
from sagemaker.predictor import Predictor
from sagemaker.serializers import IdentitySerializer
from sagemaker.deserializers import JSONDeserializer


class TyreQualitySerializer(IdentitySerializer):
```

```
    CONTENT_TYPE = 'application/x-torch'

    def serialize(self, data):
        transformed_image = preprocess(data)
        tensor_image = torch.Tensor(transformed_image)

        serialized_data = io.BytesIO()
        torch.save(tensor_image, serialized_data)
        serialized_data.seek(0)
        serialized_data = serialized_data.read()

        return serialized_data


class TyreQualityPredictor(Predictor):
    def __init__(self, endpoint_name, sagemaker_session):
        super().__init__(
            endpoint_name,
            sagemaker_session=sagemaker_session,
            serializer=TyreQualitySerializer(),
            deserializer=JSONDeserializer(),
        )

sagemaker_model = PyTorchModel(
    model_data=f's3://{bucket_name_default}/pytorch/{model_name}.tar.gz',
    role=sagemaker.get_execution_role(),
    framework_version='2.0.1',
    py_version='py310',
    predictor_cls=TyreQualityPredictor,
    entry_point='inference.py',
    source_dir='code',
)

timestamp = int(time.time())
pytorch_endpoint_name = '{}-{}-{}'.format('tyre-quality-classifier', 'pt',
timestamp)
sagemaker_predictor = sagemaker_model.deploy(
    initial_instance_count=1,
    instance_type='ml.p3.2xlarge',
    endpoint_name=pytorch_endpoint_name
)
```

Dieser Code erleichtert die Bereitstellung eines PyTorch-Modells auf SageMaker. Es definiert einen benutzerdefinierten Serialisator, **TyreQualitySerializer**, der Eingabedaten als PyTorch-Tensor vorverarbeitet und serialisiert. Die Klasse **TireQualityPredictor** ist ein benutzerdefinierter Prädiktor, der den definierten Serialisator und einen **JSONDeserializer** verwendet. Der Code erstellt außerdem ein **PyTorchModel**-Objekt, um den S3-Standort des Modells, die IAM-Rolle, die Framework-Version und den Eintrittspunkt für die Inferenz

festzulegen. Der Code generiert einen Zeitstempel und erstellt einen Endpunktnamen basierend auf dem Modell und dem Zeitstempel. Schließlich wird das Modell mithilfe der Bereitstellungsmethode bereitgestellt, wobei die Anzahl der Instanzen, der Instanztyp und der Name des generierten Endpunkts angegeben werden. Dadurch kann das PyTorch-Modell auf SageMaker bereitgestellt und für Inferenz zugänglich sein.

Inferenz

```
image_object = list(bucket.objects.filter('dataset/tyre'))[0].get()
image_bytes = image_object['Body'].read()

with Image.open(with Image.open(BytesIO(image_bytes)) as image:
    predicted_classes = sagemaker_predictor.predict(image)

    print(predicted_classes)
```

Dies ist das Beispiel für die Verwendung des implementierten Endpunkts zur Inferenz.

Teil 3 – Aufbau Einer vereinfachten MLOPS-Pipeline (CI/CT/CD)

Autor(en):
Jian Jian (Ken), Senior Data & Applied Scientist, NetApp

Einführung

In diesem Tutorial erfahren Sie, wie Sie verschiedene AWS-Services für die Erstellung einer einfachen MLOPS-Pipeline nutzen können, die Continuous Integration (CI), Continuous Training (CT) und Continuous Deployment (CD) umfasst. Im Gegensatz zu herkömmlichen DevOps-Pipelines erfordert MLOps beim Abschluss des Betriebszyklus zusätzliche Überlegungen. Wenn Sie dieses Tutorial befolgen, erhalten Sie Einblicke in die Integration von CT in die MLOPS-Schleife, was ein kontinuierliches Training Ihrer Modelle und eine nahtlose Bereitstellung für die Inferenz ermöglicht. Das Tutorial führt Sie durch die Nutzung von AWS-Services zur Einrichtung dieser End-to-End-MLOPS-Pipeline.

Manifest

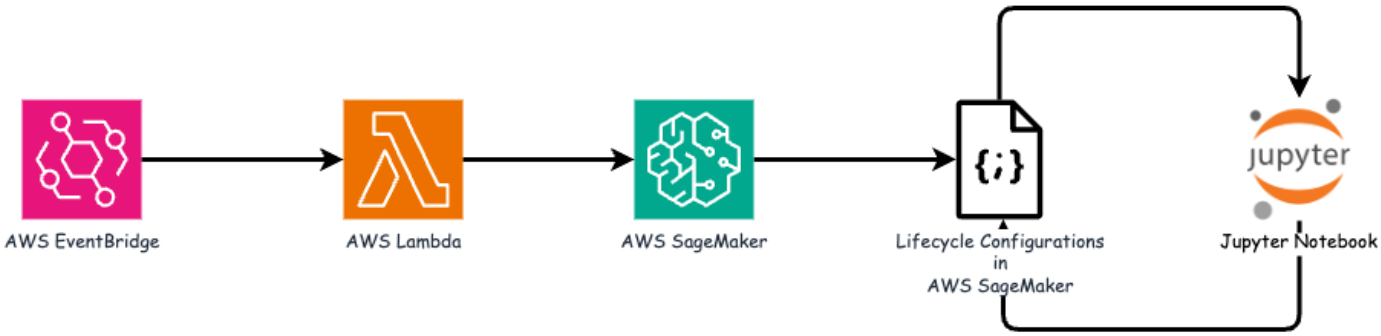
Funktionalität	Name	Kommentar
Datenspeicher	AWS FSxN	Siehe "Teil 1 – Integration von AWS FSX for NetApp ONTAP (FSxN) als privater S3-Bucket in AWS SageMaker".
Data Science IDE	AWS SageMaker	Dieses Tutorial basiert auf dem in vorgestellten Jupyter Notebook "Teil 2 – Nutzung von AWS FSX for NetApp ONTAP (FSxN) als Datenquelle für das Modelltraining in SageMaker".
Funktion zum Auslösen der MLOPS-Pipeline	AWS Lambda-Funktion	-
Cron-Job-Trigger	AWS EventBridge	-

Funktionalität	Name	Kommentar
Deep-Learning-Framework	PyTorch	-
AWS Python SDK	Boto3	-
Programmiersprache	Python	v3.10

Voraussetzung

- Ein vorkonfiguriertes FSxN-Dateisystem. In diesem Tutorial werden die in FSxN gespeicherten Daten für den Trainingsprozess verwendet.
- Eine **SageMaker Notebook-Instanz**, die so konfiguriert ist, dass sie dieselbe VPC wie das oben erwähnte FSxN-Dateisystem verwendet.
- Bevor Sie die **AWS Lambda-Funktion** aktivieren, stellen Sie sicher, dass die **SageMaker Notebook-Instanz** den Status **angehalten** hat.
- Der Instanztyp **ml.g4dn.xlarge** wird benötigt, um die GPU-Beschleunigung zu nutzen, die für die Berechnung tiefer neuronaler Netzwerke notwendig ist.

Der Netapp Architektur Sind



Diese MLOPS-Pipeline ist eine praktische Implementierung, die mithilfe eines Cron-Jobs eine serverlose Funktion auslöst, die wiederum einen AWS-Service ausführt, der mit einer Lifecycle-Callback-Funktion registriert ist. Die **AWS EventBridge** fungiert als Cron-Job. Es ruft regelmäßig eine **AWS Lambda-Funktion** auf, die für die Umschulung und Neuimplementierung des Modells verantwortlich ist. Bei diesem Vorgang wird die **AWS SageMaker Notebook**-Instanz hochgefahren, um die erforderlichen Aufgaben auszuführen.

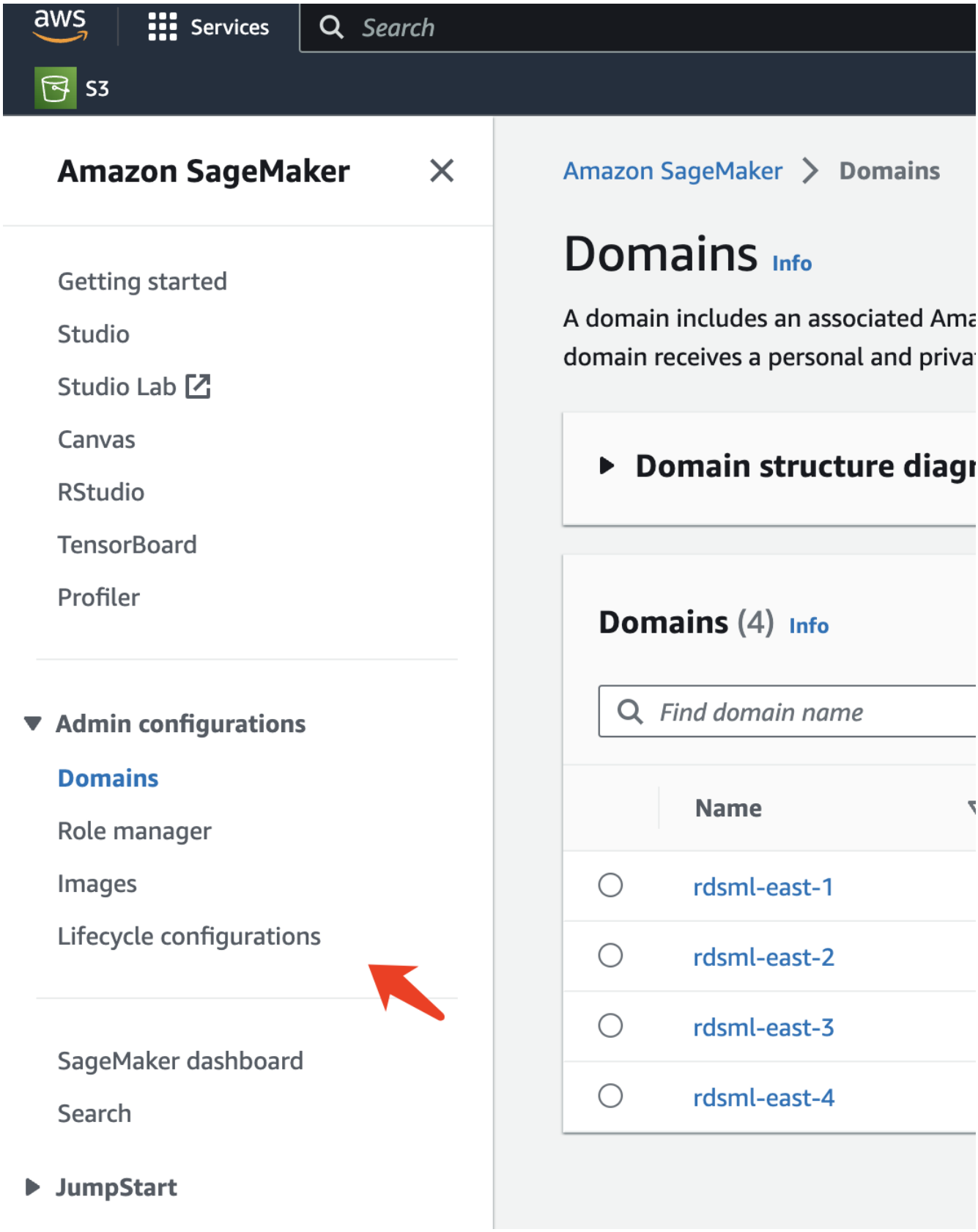
Schritt-für-Schritt-Konfiguration

Lebenszykluskonfigurationen

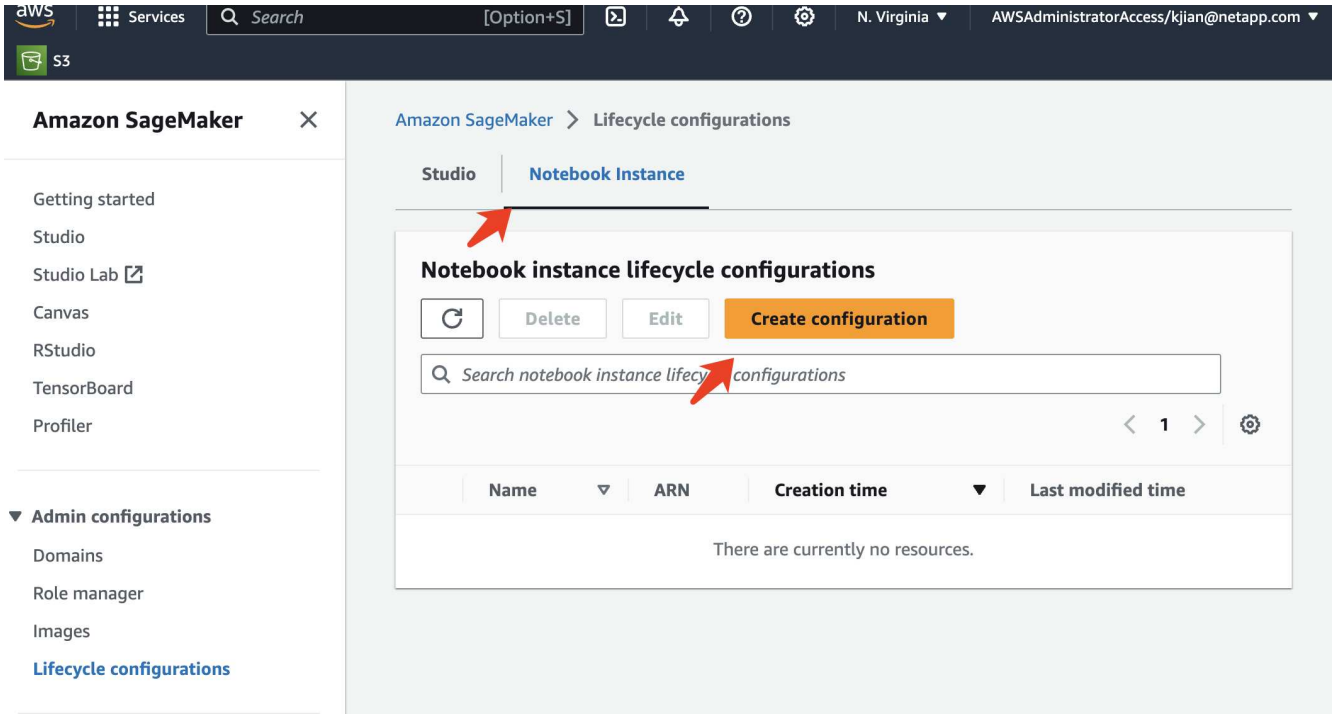
Um die Lifecycle-Callback-Funktion für die AWS SageMaker Notebook-Instanz zu konfigurieren, würden Sie **Lifecycle-Konfigurationen** verwenden. Mit diesem Service können Sie die erforderlichen Aktionen definieren, die beim Starten der Notizbuchinstanz ausgeführt werden müssen. Konkret kann ein Shell-Skript innerhalb der **Lifecycle-Konfigurationen** implementiert werden, um die Notebook-Instanz nach Abschluss der Trainings- und Bereitstellungsprozesse automatisch herunterzufahren. Dies ist eine erforderliche Konfiguration, da die Kosten eine der wichtigsten Überlegungen bei MLOPS sind.

Es ist wichtig zu beachten, dass die Konfiguration für **Lifecycle-Konfigurationen** im Voraus eingerichtet werden muss. Daher wird empfohlen, die Konfiguration dieses Aspekts zu priorisieren, bevor mit dem Setup der anderen MLOPS-Pipeline fortgefahren wird.

1. Um eine Lifecycle-Konfiguration einzurichten, öffnen Sie das Fenster **SageMaker** und navigieren Sie zu **Lifecycle-Konfigurationen** unter dem Abschnitt **Admin-Konfigurationen**.



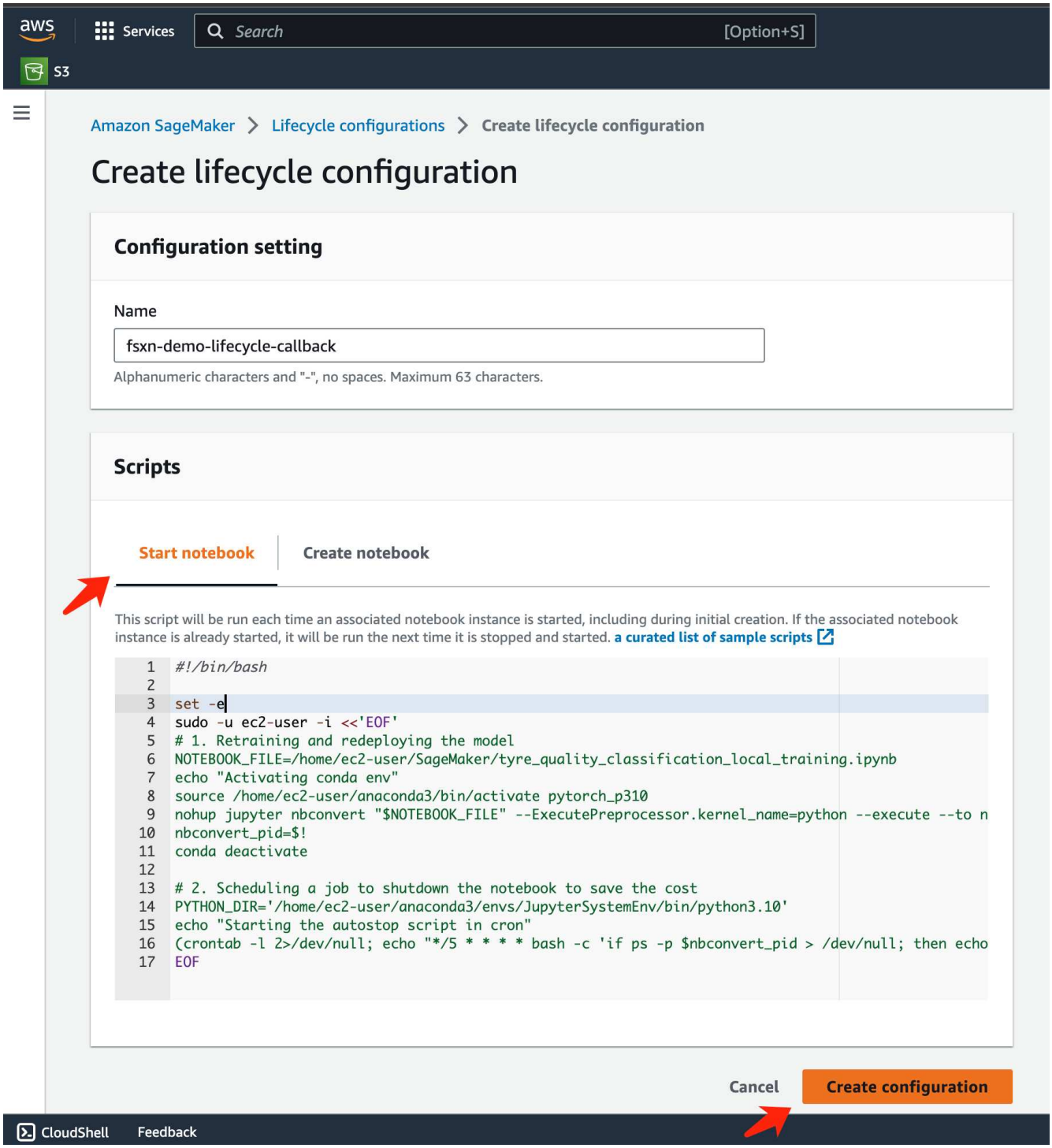
2. Wählen Sie die Registerkarte **Notebook-Instanz** aus und klicken Sie auf die Schaltfläche **Konfiguration erstellen**



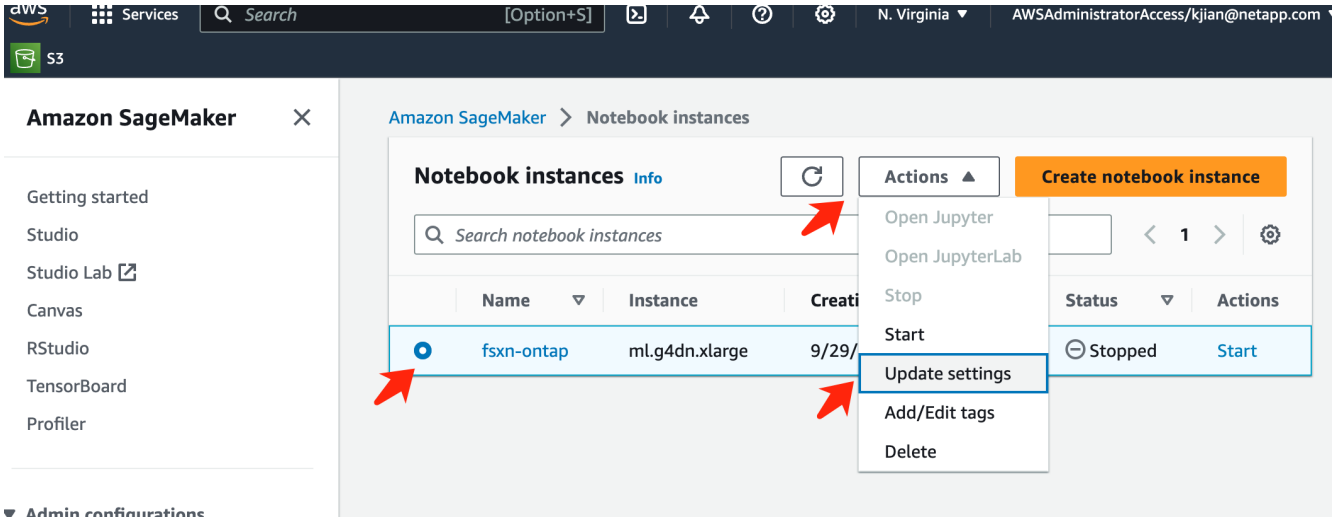
3. Fügen Sie den unten stehenden Code in den Eingabebereich ein.

```
#!/bin/bash

set -e
sudo -u ec2-user -i <<'EOF'
# 1. Retraining and redeploying the model
NOTEBOOK_FILE=/home/ec2-
user/SageMaker/tyre_quality_classification_local_training.ipynb
echo "Activating conda env"
source /home/ec2-user/anaconda3/bin/activate pytorch_p310
nohup jupyter nbconvert "$NOTEBOOK_FILE"
--ExecutePreprocessor.kernel_name=python --execute --to notebook &
nbconvert_pid=$!
conda deactivate

# 2. Scheduling a job to shutdown the notebook to save the cost
PYTHON_DIR='/home/ec2-
user/anaconda3/envs/JupyterSystemEnv/bin/python3.10'
echo "Starting the autostop script in cron"
(crontab -l 2>/dev/null; echo "*/5 * * * * bash -c 'if ps -p
$nbconvert_pid > /dev/null; then echo \"Notebook is still running.\" >>
/var/log/jupyter.log; else echo \"Notebook execution completed.\" >>
/var/log/jupyter.log; $PYTHON_DIR -c \"import boto3;boto3.client(
\'sagemaker\').stop_notebook_instance(NotebookInstanceName=get_notebook_
name())\" >> /var/log/jupyter.log; fi'") | crontab -
EOF
```

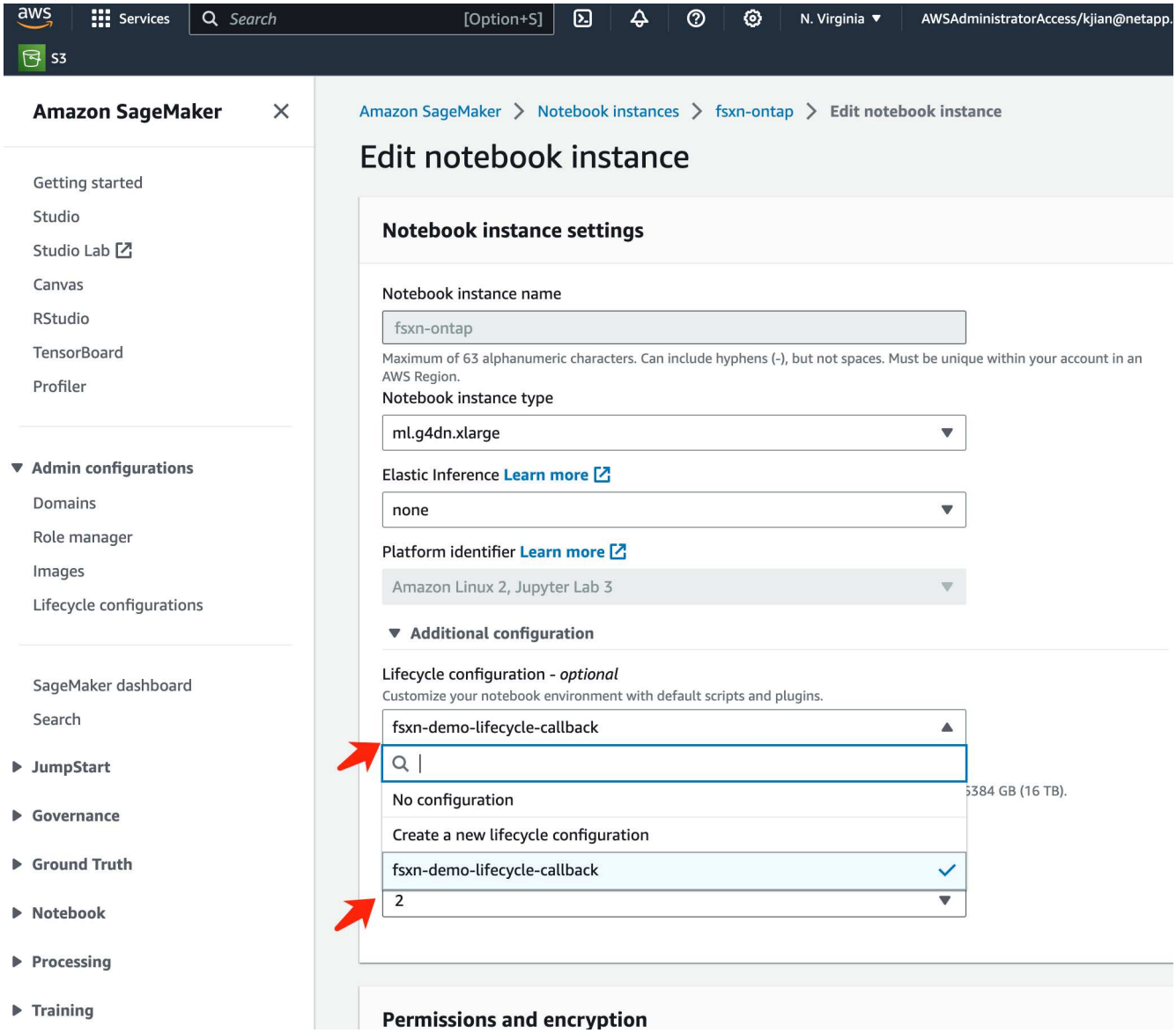
4. Dieses Skript führt das Jupyter Notebook aus, das das erneute Training und die erneute Bereitstellung des Modells für die Inferenz übernimmt. Nach Abschluss der Ausführung wird das Notebook automatisch innerhalb von 5 Minuten heruntergefahren. Weitere Informationen zur Problembeschreibung und zur Codeimplementierung finden Sie unter "Teil 2 – Nutzung von AWS FSX for NetApp ONTAP (FSxN) als Datenquelle für das Modelltraining in SageMaker".



5. Navigieren Sie nach der Erstellung zu Notizbuchinstanzen, wählen Sie die Zielinstanz aus, und klicken Sie im Dropdown-Menü Aktionen auf **Einstellungen aktualisieren**.



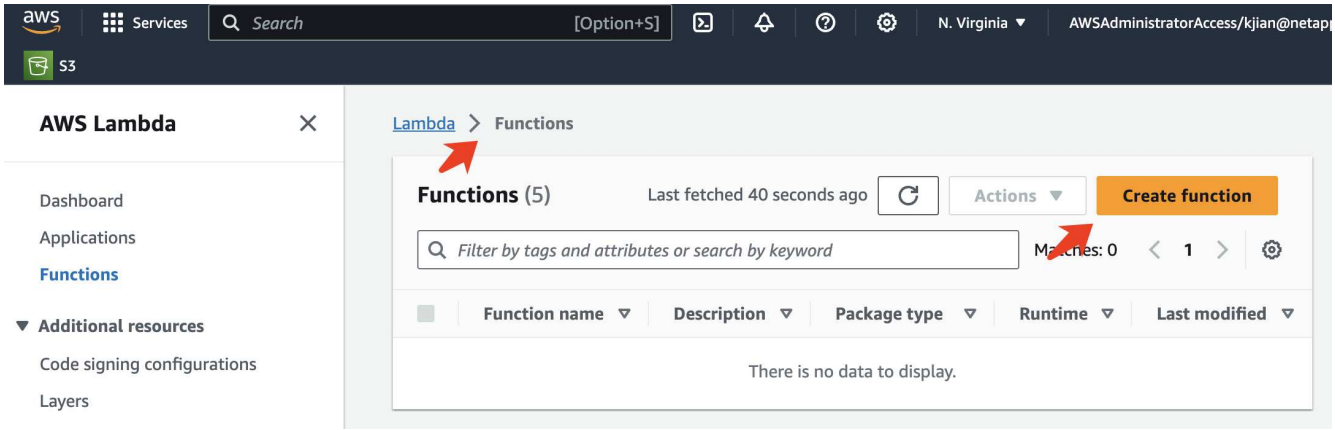
6. Wählen Sie die erstellte **Lifecycle-Konfiguration** aus und klicken Sie auf **Notebook-Instanz aktualisieren**.



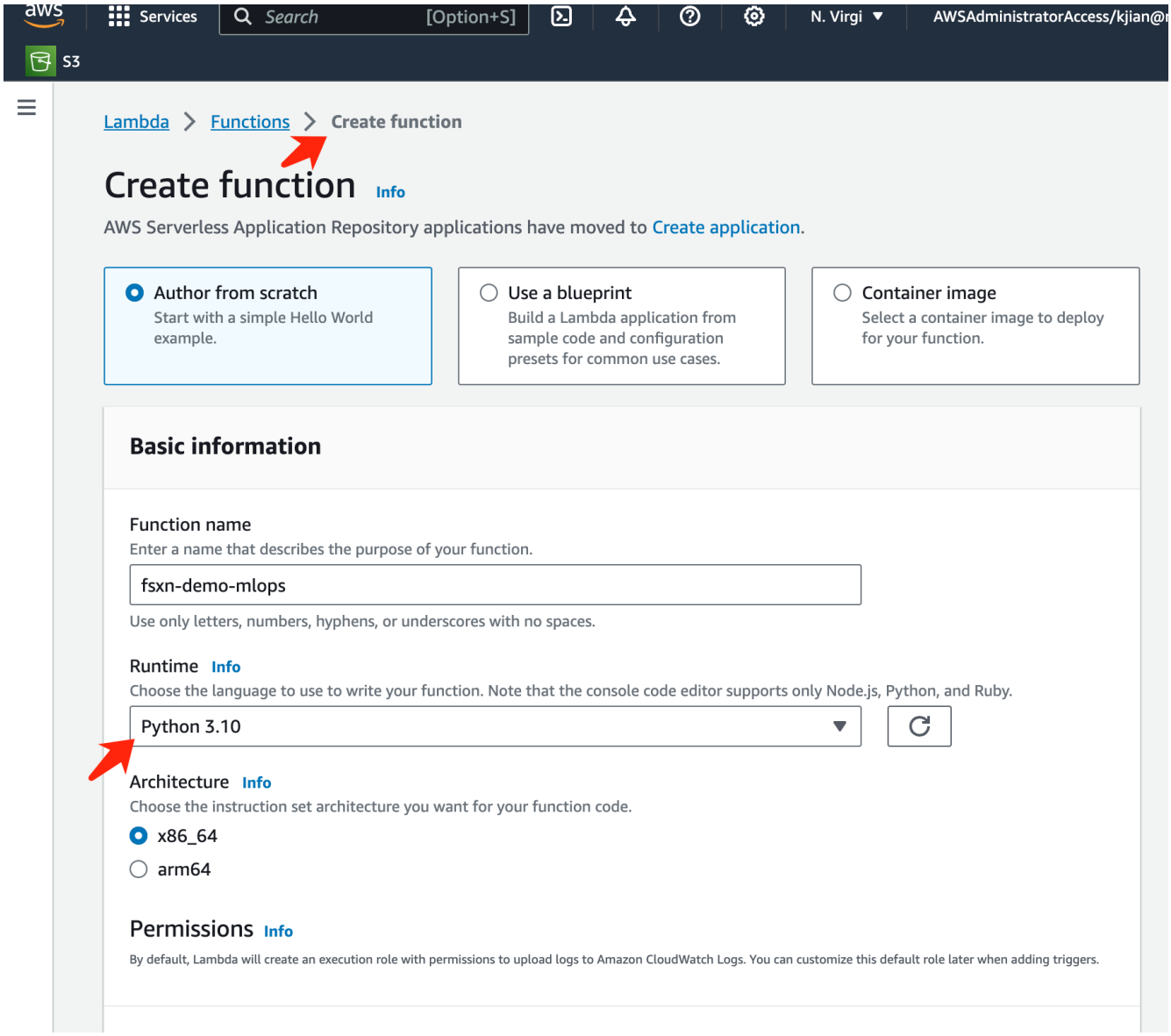
AWS Lambda serverlose Funktion

Wie bereits erwähnt, ist die **AWS Lambda-Funktion** für die Einrichtung der **AWS SageMaker Notebook-Instanz** zuständig.

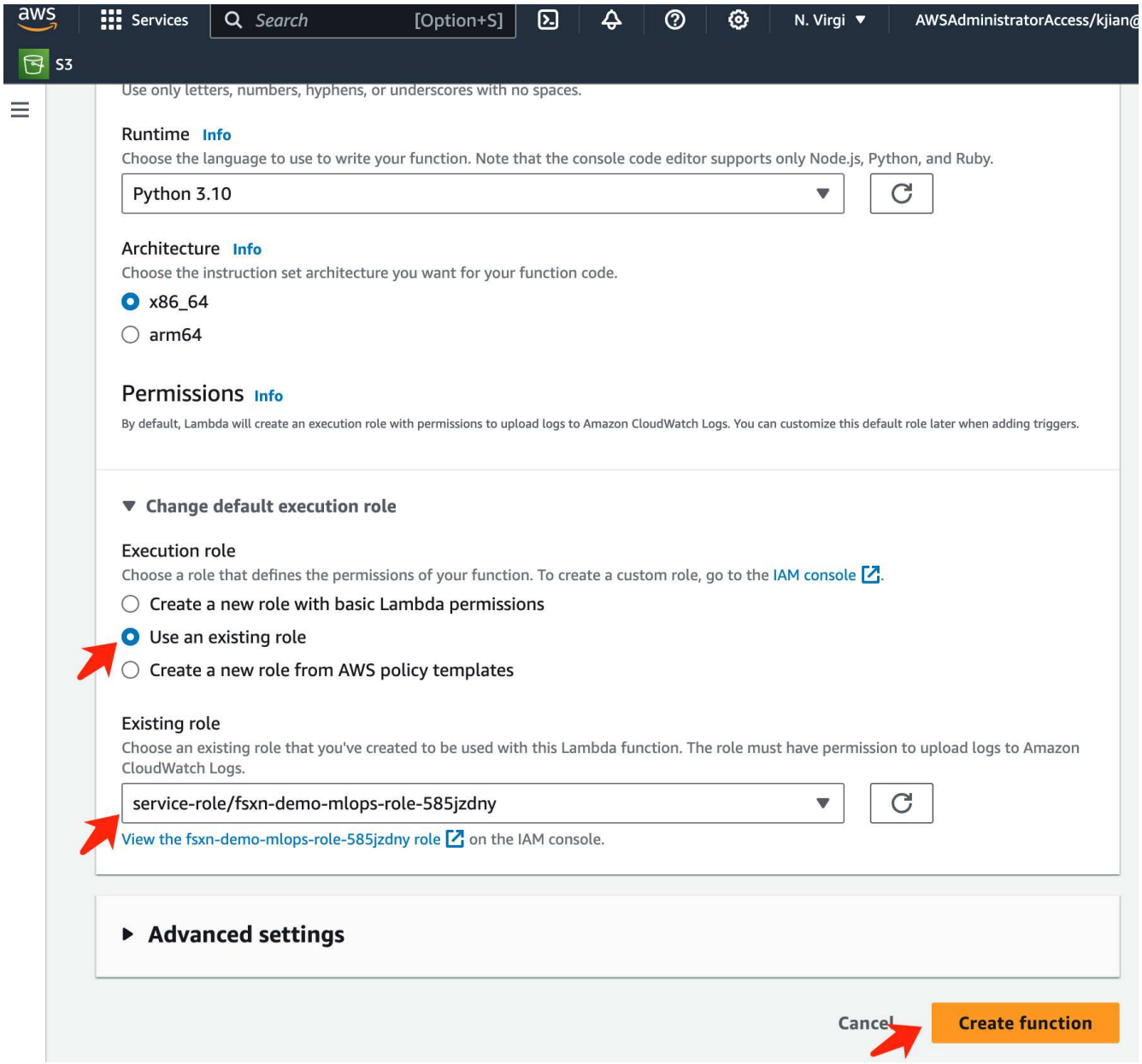
1. Um eine **AWS Lambda-Funktion** zu erstellen, navigieren Sie zum entsprechenden Panel, wechseln Sie zum Reiter **Funktionen** und klicken Sie auf **Create Function**.



2. Bitte legen Sie alle erforderlichen Einträge auf der Seite ab und denken Sie daran, die Runtime auf **Python 3.10** umzuschalten.



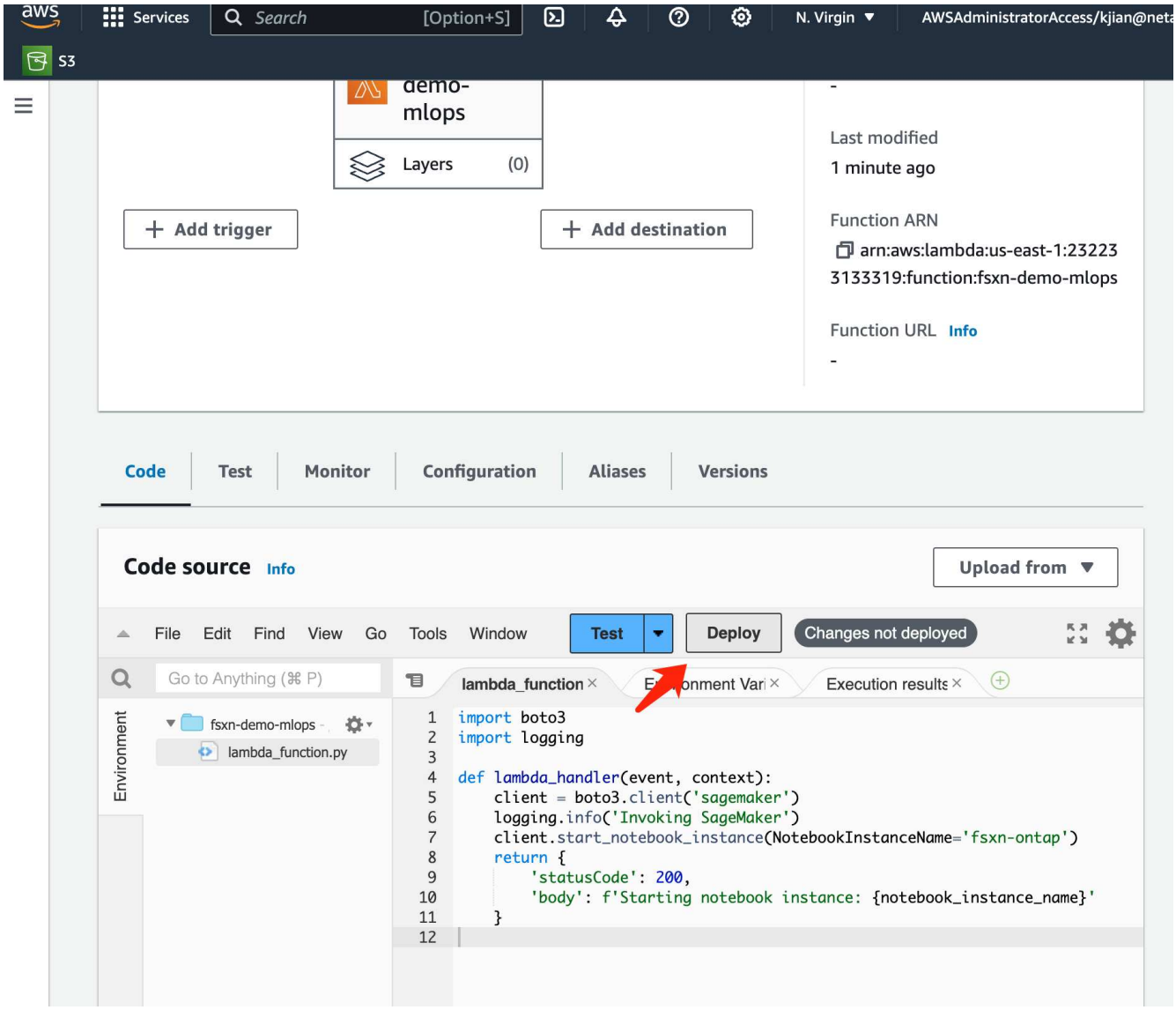
3. Bitte überprüfen Sie, ob die vorgesehene Rolle die erforderliche Berechtigung hat **AmazonSageMakerFullAccess** und klicken Sie auf den Button **Funktion erstellen**.



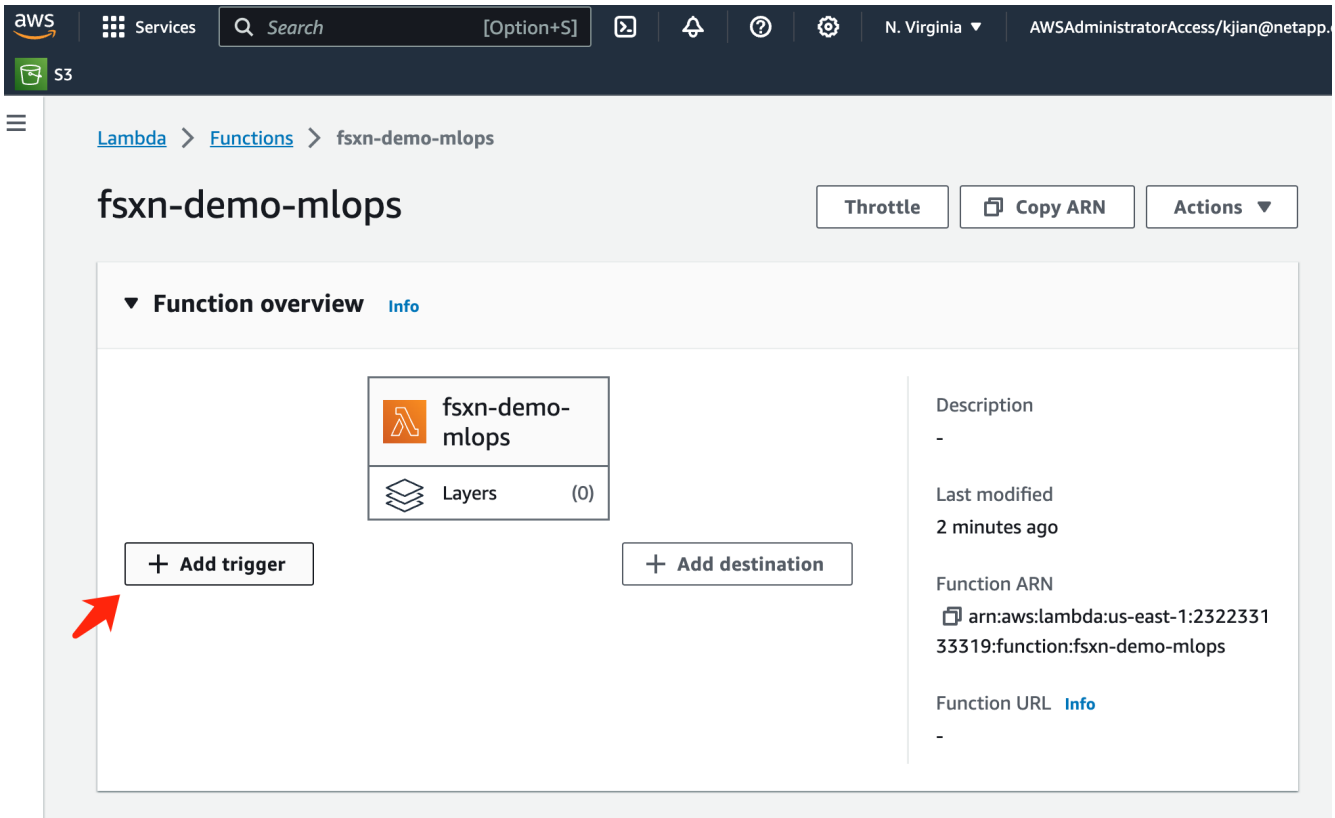
4. Wählen Sie die erstellte Lambda-Funktion aus. Kopieren Sie auf der Registerkarte Code den folgenden Code, und fügen Sie ihn in den Textbereich ein. Dieser Code startet die Notebook-Instanz mit dem Namen **fsxn-ontap**.

```
import boto3
import logging

def lambda_handler(event, context):
    client = boto3.client('sagemaker')
    logging.info('Invoking SageMaker')
    client.start_notebook_instance(NotebookInstanceName='fsxn-ontap')
    return {
        'statusCode': 200,
        'body': f'Starting notebook instance: {notebook_instance_name}'
    }
```

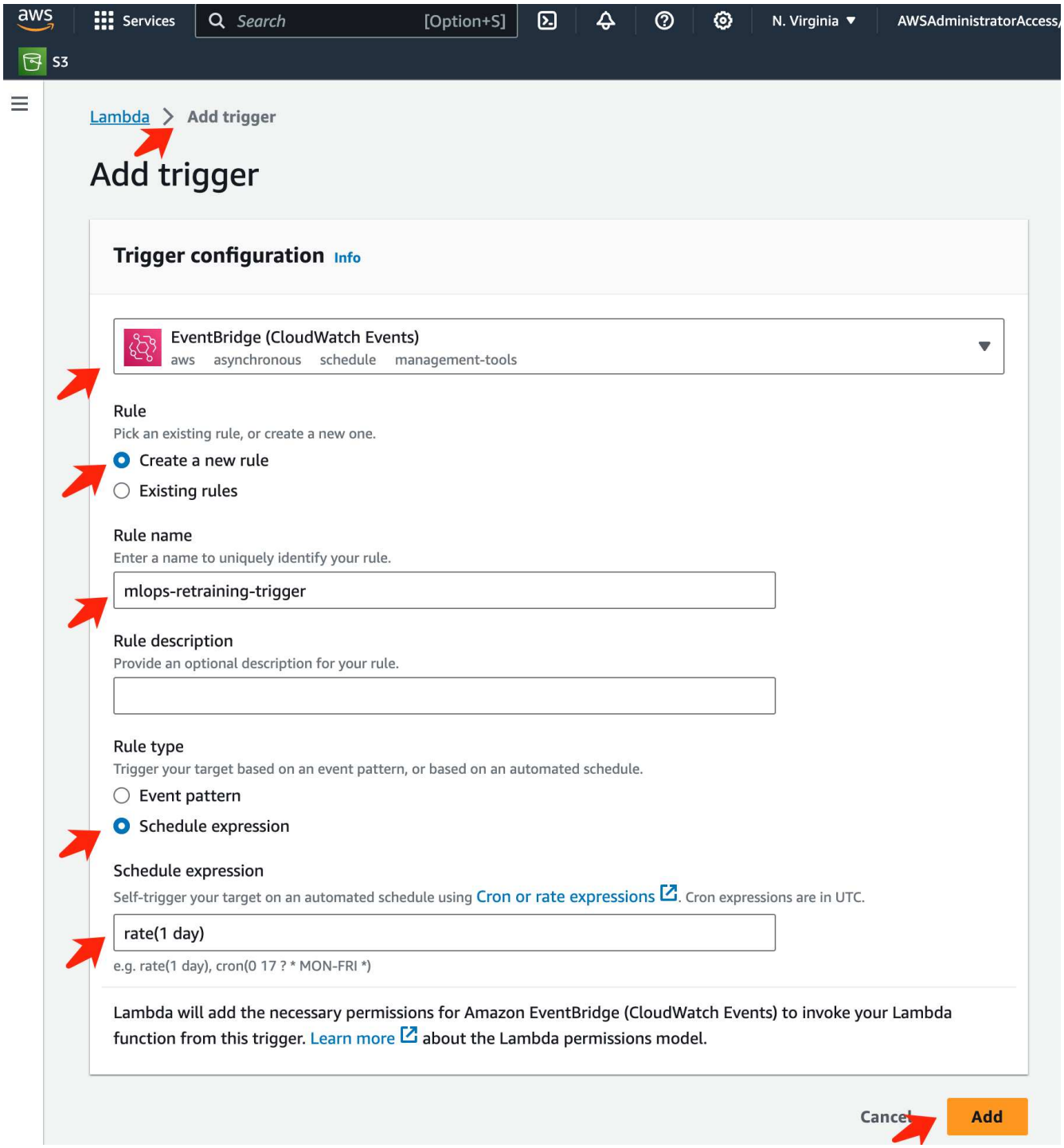
5. Klicken Sie auf die Schaltfläche **deploy**, um diese Codeänderung anzuwenden.



6. Um anzugeben, wie diese AWS Lambda-Funktion ausgelöst werden soll, klicken Sie auf die Schaltfläche Add Trigger.



7. Wählen Sie EventBridge aus dem Dropdown-Menü aus, und klicken Sie dann auf das Optionsfeld Neue Regel erstellen. Geben Sie im Feld Ausdruck Zeitplan ein `rate(1 day)`, Und klicken Sie auf die Schaltfläche Hinzufügen, um diese neue Cron-Job-Regel auf die AWS Lambda-Funktion zu erstellen und anzuwenden.



Nach Abschluss der zweistufigen Konfiguration startet die **AWS Lambda-Funktion** täglich das **SageMaker Notebook**, führt Modellumschulungen mit den Daten aus dem **FSxN**-Repository durch, stellt das aktualisierte Modell in der Produktionsumgebung wieder bereit und fährt die **SageMaker Notebook-Instanz** automatisch herunter, um die Kosten zu optimieren. Damit bleibt das Modell auf dem neuesten Stand.

Damit ist das Tutorial zur Entwicklung einer MLOPS-Pipeline abgeschlossen.

MLOps für die Hybrid-Multi-Cloud mit Domino Data Lab und NetApp

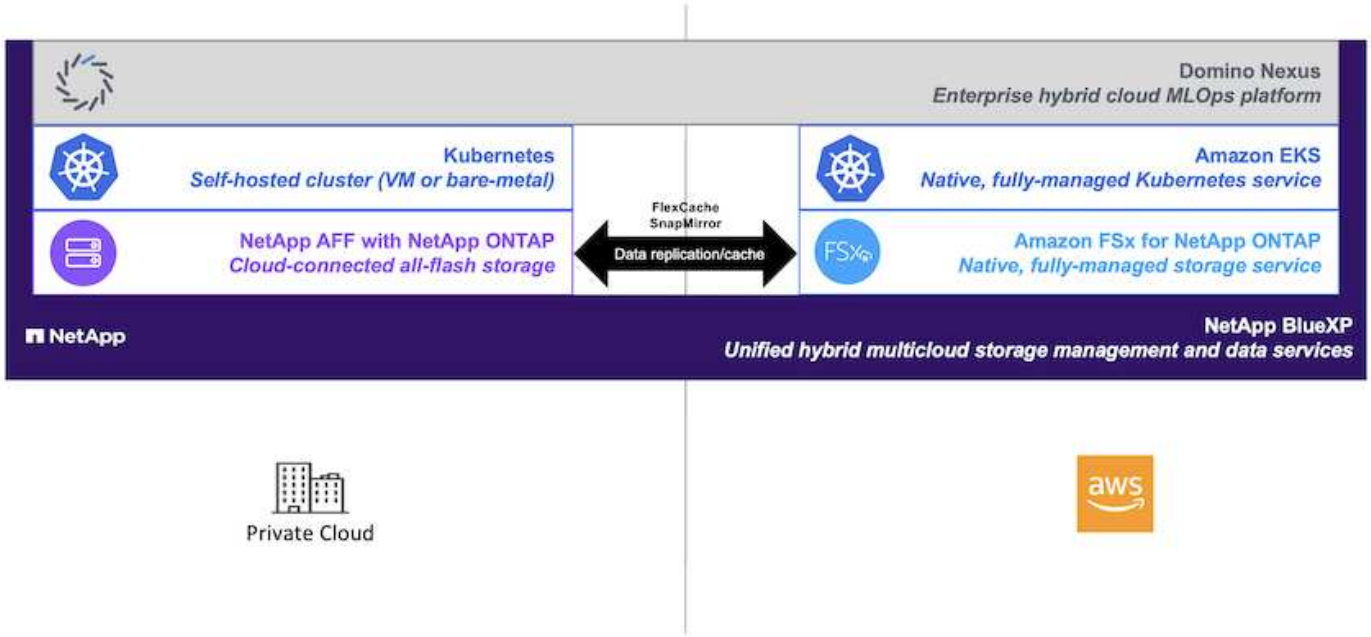
MLOps für die Hybrid-Multi-Cloud mit Domino Data Lab und NetApp

Mike Oglesby, NetApp

Weltweit setzen Unternehmen derzeit KI ein, um ihre Geschäfte und Prozesse zu transformieren. Aus diesem Grund ist die KI-fähige Computing-Infrastruktur oft knapp. Um die Vorteile verfügbarer Computing-Umgebungen in verschiedenen Regionen, Datacentern und Clouds zu nutzen und ein ausgewogenes Verhältnis zwischen Kosten, Verfügbarkeit und Performance zu erzielen, setzen Unternehmen zunehmend auf MLOps-Architekturen für Hybrid-Multi-Cloud.

Domino Nexus von Domino Data Lab ist eine einheitliche MLOps-Kontrollebene, mit der Sie Data Science- und Machine-Learning-Workloads in jedem beliebigen Compute-Cluster ausführen können – in jeder Cloud, Region oder lokal. Sie vereinheitlicht Data-Science-Silos im gesamten Unternehmen, sodass Sie an einem Ort Modelle erstellen, implementieren und überwachen können. Ebenso können Sie mit den Hybrid-Cloud-Datenmanagementfunktionen von NetApp Ihre Daten an Ihre Jobs und Workspaces übertragen – unabhängig davon, wo sie ausgeführt werden. Das Paar von Domino Nexus und NetApp ermöglicht die Flexibilität, Workloads über verschiedene Umgebungen hinweg zu planen, ohne sich Gedanken über die Datenverfügbarkeit machen zu müssen. Mit anderen Worten: Sie können Ihre Workloads und Ihre Daten an die entsprechende Computing-Umgebung senden. So können Sie Ihre KI-Implementierungen beschleunigen und gleichzeitig die Vorschriften zu Datenschutz und -Hoheit einhalten.

In dieser Lösung wird die Implementierung einer einheitlichen MLOps-Kontrollebene demonstriert, die ein Kubernetes-Cluster vor Ort und einen EKS-Cluster (Elastic Kubernetes Service) in Amazon Web Services (AWS) umfasst.



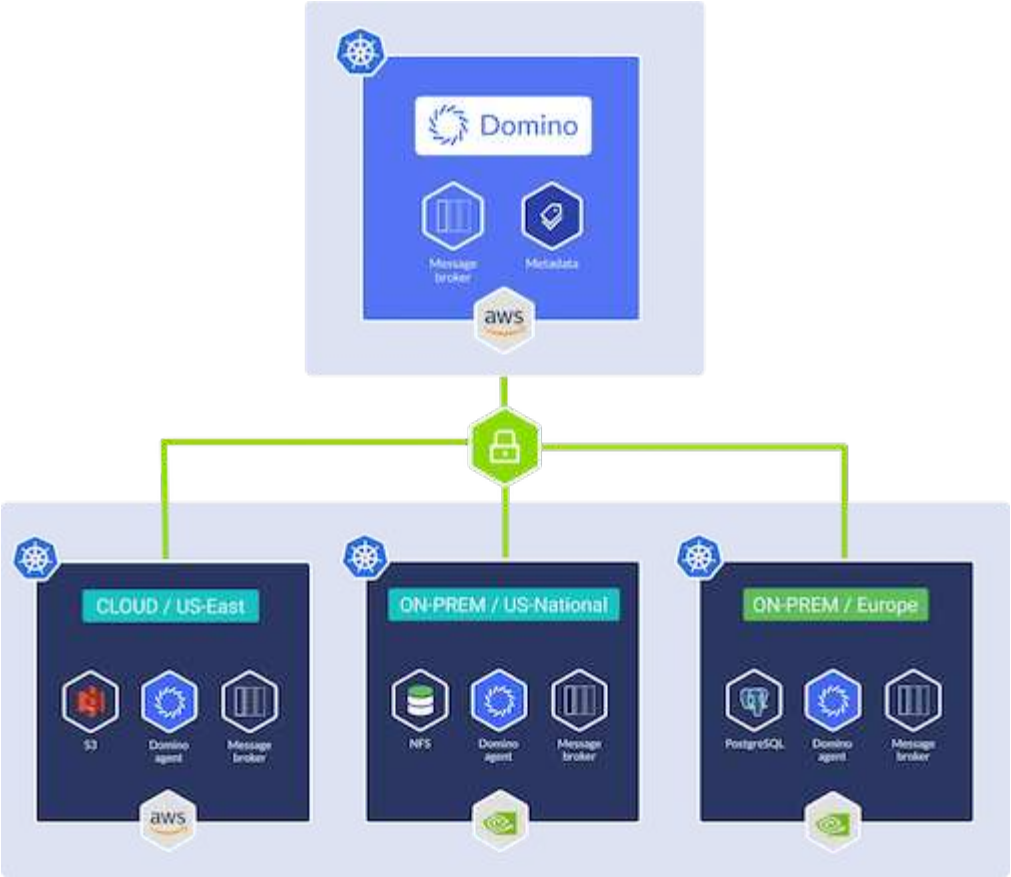
Technologischer Überblick

Domino Data Lab

Domino Data Lab unterstützt modellgetriebene Unternehmen mit seiner führenden Enterprise-KI-Plattform, der mehr als 20 % der Fortune 100 Unternehmen vertrauen. Domino beschleunigt die Entwicklung und Implementierung von Data Science-Arbeiten bei gleichzeitiger Verbesserung von Zusammenarbeit und Governance. Mit Domino können Unternehmen weltweit bessere Medikamente entwickeln, ihre Produktivität steigern, bessere Autos bauen und vieles mehr. Domino wurde 2013 gegründet und wird von Coatue Management, Great Hill Partners, Highland Capital, Sequoia Capital und anderen führenden Investoren unterstützt.

Mit Domino können Unternehmen und ihre Data Scientists KI auf einer einheitlichen End-to-End-Plattform erstellen, implementieren und managen – schnell, verantwortungsvoll und kostengünstig. Teams können in allen Umgebungen auf sämtliche Daten, Tools, Computing, Modelle und Projekte zugreifen, die sie benötigen. So können sie zusammenarbeiten, vergangene Arbeiten wiederverwenden, Modelle in der Produktion nachverfolgen und die Genauigkeit verbessern, mit Best Practices standardisieren und KI verantwortungsbewusst und kontrolliert machen.

- **Open and Flexible:** Zugriff auf das größte Ökosystem von Open Source- und kommerziellen Tools und Infrastrukturen, für die besten Innovationen und ohne Anbieterbindung.
- **System of Record:** zentraler Hub für KI-Abläufe und Wissen im gesamten Unternehmen, der Best Practices, funktionsübergreifende Zusammenarbeit, schnellere Innovation und Effizienz ermöglicht.
- **Integriert:** integrierte Workflows und Automatisierung – entwickelt für Unternehmensprozesse, Kontrollen und Governance – erfüllen Ihre Compliance- und gesetzlichen Anforderungen.
- **Hybrid-Multi-Cloud:** führen Sie KI-Workloads in der Nähe Ihrer Daten aus – On-Premises, Hybrid, beliebige Clouds oder Multi-Cloud – für geringere Kosten, optimale Performance und Compliance.



Domino Nexus

Domino Nexus ist eine zentrale Konsole für die Ausführung von Data Science- und Machine Learning-Workloads in jedem beliebigen Compute-Cluster – in jeder Cloud, Region oder lokal. Sie vereinheitlicht Data-Science-Silos im gesamten Unternehmen, sodass Sie an einem Ort Modelle erstellen, implementieren und überwachen können.

NetApp BlueXP

NetApp BlueXP vereint alle Storage- und Datenservices von NetApp in einem einzigen Tool zum Erstellen, Schützen und Überwachen Ihrer Hybrid-Multi-Cloud-Daten. Sie bietet eine einheitliche Arbeitsumgebung für Storage- und Datenservices lokal und in Cloud-Umgebungen und ermöglicht durch AIOps eine einfache Betriebsabläufe. Zudem bieten sie flexible Nutzungsmodelle und integrierten Schutz, die in der heutigen Cloud-orientierten Welt erforderlich sind.

NetApp ONTAP

ONTAP 9, die jüngste Generation der Storage-Managementsoftware von NetApp, ermöglicht Unternehmen eine Modernisierung der Infrastruktur und den Übergang zu einem Cloud-fähigen Datacenter. Dank der erstklassigen Datenmanagementfunktionen lassen sich mit ONTAP sämtliche Daten mit einem einzigen Toolset managen und schützen, ganz gleich, wo sich diese Daten befinden. Zudem können Sie die Daten problemlos dorthin verschieben, wo sie benötigt werden: Zwischen Edge, Core und Cloud. ONTAP 9 umfasst zahlreiche Funktionen, die das Datenmanagement vereinfachen, geschäftskritische Daten beschleunigen und schützen und Infrastrukturfunktionen der nächsten Generation über Hybrid-Cloud-Architekturen hinweg ermöglichen.

Vereinfachtes Datenmanagement

Für den Enterprise IT-Betrieb und die Data Scientists spielt Datenmanagement eine zentrale Rolle, damit für KI-Applikationen die entsprechenden Ressourcen zum Training von KI/ML-Datensätzen verwendet werden. Die folgenden zusätzlichen Informationen über NetApp Technologien sind bei dieser Validierung nicht im Umfang enthalten, können jedoch je nach Ihrer Implementierung relevant sein.

Die ONTAP Datenmanagement-Software umfasst die folgenden Funktionen, um den Betrieb zu optimieren und zu vereinfachen und damit Ihre Gesamtbetriebskosten zu senken:

- Inline-Data-Compaction und erweiterte Deduplizierung: Data-Compaction reduziert den ungenutzten Speicherplatz in Storage-Blöcken, während Deduplizierung die effektive Kapazität deutlich steigert. Dies gilt für lokal gespeicherte Daten und für Daten-Tiering in die Cloud.
- Minimale, maximale und adaptive Quality of Service (AQoS): Durch granulare QoS-Einstellungen (Quality of Service) können Unternehmen ihre Performance-Level für kritische Applikationen auch in Umgebungen mit vielen unterschiedlichen Workloads garantieren.
- NetApp FabricPool: Bietet automatisches Tiering von „kalten“ Daten in Private- und Public-Cloud-Storage-Optionen, einschließlich Amazon Web Services (AWS), Azure und NetApp StorageGRID Storage-Lösung. Weitere Informationen zu FabricPool finden Sie unter "TR-4598: FabricPool Best Practices".

Beschleunigung und Sicherung von Daten

ONTAP bietet überdurchschnittliche Performance und Datensicherung, erweitert diese Funktionen auf folgende Weise:

- Performance und niedrige Latenz: ONTAP bietet höchstmöglichen Durchsatz bei geringstmöglicher Latenz.
- Datensicherung ONTAP verfügt über integrierte Funktionen für die Datensicherung mit zentralem Management über alle Plattformen hinweg.
- NetApp Volume Encryption (NVE) ONTAP bietet native Verschlüsselung auf Volume-Ebene und unterstützt sowohl Onboard- als auch externes Verschlüsselungsmanagement.
- Multi-Faktor- und Multi-Faktor-Authentifizierung – ONTAP ermöglicht die gemeinsame Nutzung von Infrastrukturressourcen mit höchstmöglicher Sicherheit.

Zukunftssichere Infrastruktur

ONTAP bietet folgende Funktionen, um anspruchsvolle und sich ständig ändernde Geschäftsanforderungen zu erfüllen:

- Nahtlose Skalierung und unterbrechungsfreier Betrieb. Mit ONTAP sind das Hinzufügen von Kapazitäten zu bestehenden Controllern und das Scale-out von Clustern unterbrechungsfrei möglich. Kunden können Upgrades auf die neuesten Technologien wie NVMe und 32 GB FC ohne teure Datenmigrationen oder Ausfälle durchführen.
- Cloud-Anbindung: ONTAP ist die Storage-Managementsoftware mit der umfassendsten Cloud-Integration und bietet Optionen für softwaredefinierten Storage und Cloud-native Instanzen in allen Public Clouds.
- Integration in moderne Applikationen: ONTAP bietet Datenservices der Enterprise-Klasse für Plattformen und Applikationen der neuesten Generation, wie autonome Fahrzeuge, Smart Citys und Industrie 4.0, auf derselben Infrastruktur, die bereits vorhandene Unternehmensanwendungen unterstützt.

Amazon FSX für NetApp ONTAP

Amazon FSX for NetApp ONTAP ist ein vollständig gemanagter AWS-Service direkt vom Erstanbieter, der äußerst zuverlässigen, skalierbaren, hochperformanten und funktionsreichen File-Storage auf Basis des

beliebten ONTAP-Filesystems von NetApp bietet. FSX für ONTAP kombiniert die bekannten Funktionen, Performance, Funktionen und API-Vorgänge von NetApp Filesystemen mit der Agilität, Skalierbarkeit und Einfachheit eines vollständig gemanagten AWS Service.

NetApp Astra Trident

Astra Trident ermöglicht die Nutzung und das Management von Storage-Ressourcen über alle gängigen NetApp Storage-Plattformen hinweg, in der Public Cloud oder lokal, einschließlich ONTAP (AFF, FAS, Select, Cloud, Amazon FSX for NetApp ONTAP), Element Software (NetApp HCI, SolidFire), Azure NetApp Files Service und Cloud Volumes Service auf Google Cloud. Astra Trident ist ein CSI-konformer dynamischer Storage-Orchestrator, der sich nativ in Kubernetes integrieren lässt.

Kubernetes

Kubernetes ist eine ursprünglich von Google entwickelte Open-Source-Plattform zur Container-Orchestrierung, die jetzt von der Cloud Native Computing Foundation (CNCF) verwaltet wird. Kubernetes unterstützt die Automatisierung von Implementierungs-, Management- und Skalierungsfunktionen für Container-Applikationen und ist die dominierende Plattform für die Container-Orchestrierung in Enterprise-Umgebungen.

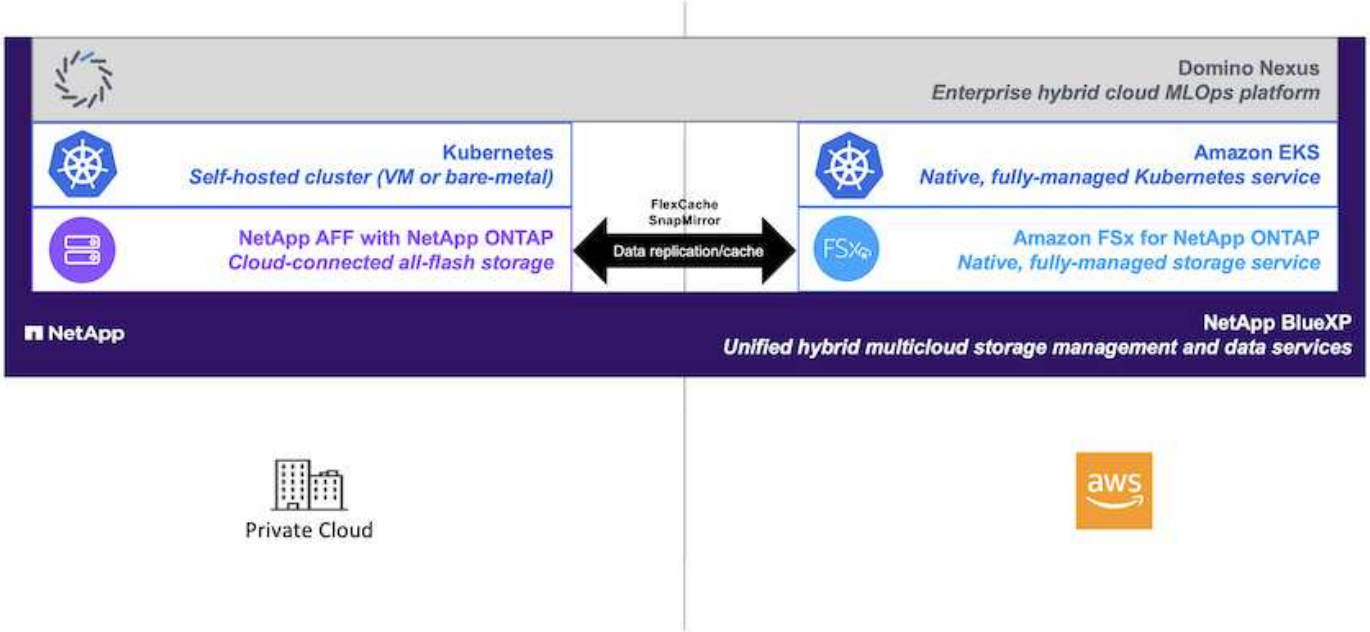
Amazon Elastic Kubernetes Service (EKS)

Amazon Elastic Kubernetes Service (Amazon EKS) ist ein gemanagter Kubernetes Service in der AWS Cloud. Amazon EKS managt automatisch die Verfügbarkeit und Skalierbarkeit der Kubernetes-Kontrollebene Nodes, die für die Planung von Containern, das Management der Applikationsverfügbarkeit, das Speichern von Cluster-Daten und andere wichtige Aufgaben zuständig sind. Mit Amazon EKS können Sie von der Performance, Skalierbarkeit, Zuverlässigkeit und Verfügbarkeit einer AWS-Infrastruktur profitieren und auch in die Netzwerk- und Sicherheitsservices von AWS integrieren.

Der Netapp Architektur Sind

Diese Lösung kombiniert die Planungsfunktionen für Hybrid-Multi-Cloud-Workloads von Domino Nexus mit NetApp Datenservices in einer einheitlichen Hybrid-Cloud-MLOps-Plattform. Einzelheiten hierzu finden Sie in der folgenden Tabelle.

Komponente	Name	Umgebung
MLOPS-Kontrollebene	"Domino Enterprise AI Platform mit Domino Nexus"	AWS
MLOps Platform Computing-Umgebungen beschrieben	"Domino Nexus Datenebenen"	AWS, lokales Datacenter
On-Premises-Computing-Plattform	"Kubernetes" Mit "NetApp Astra Trident"	Lokalen Datacenter
Cloud-Computing-Plattform	"Amazon Elastic Kubernetes Service (EKS)" Mit "NetApp Astra Trident"	AWS
On-Premises-Datenplattform	"NetApp Storage Appliance" Betrieben von "NetApp ONTAP"	Lokalen Datacenter
Cloud-Datenplattform	"Amazon FSX für NetApp ONTAP"	AWS



Ersteinrichtung

In diesem Abschnitt werden die ersten Einrichtungsaufgaben beschrieben, die für den Einsatz von Domino Nexus mit NetApp Datenservices in einer hybriden Umgebung mit einem On-Premises-Datacenter und AWS durchgeführt werden müssen.

Voraussetzungen

Bevor Sie die in diesem Abschnitt beschriebenen Schritte ausführen, gehen wir davon aus, dass Sie bereits die folgenden Aufgaben ausgeführt haben:

- Sie haben Ihre lokale NetApp ONTAP Storage-Plattform bereits implementiert und konfiguriert. Weitere Informationen finden Sie im "NetApp Produktdokumentation".
- Sie haben bereits eine Amazon FSX for NetApp ONTAP-Instanz in AWS bereitgestellt. Weitere Informationen finden Sie im "Amazon FSX for NetApp ONTAP Produktseite".
- Sie haben bereits ein Kubernetes-Cluster in Ihrem lokalen Datacenter bereitgestellt. Weitere Informationen finden Sie im "Domino Administratorhandbuch".
- Sie haben bereits einen Amazon EKS-Cluster in AWS bereitgestellt. Weitere Informationen finden Sie im "Domino Administratorhandbuch".
- Sie haben NetApp Astra Trident in Ihrem Kubernetes-Cluster vor Ort installiert. Außerdem haben Sie diese Trident Instanz zur Verwendung Ihrer lokalen NetApp ONTAP Storage-Plattform beim Bereitstellen und Managen von Storage-Ressourcen konfiguriert. Weitere Informationen finden Sie im "NetApp Astra Trident – Dokumentation".
- Sie haben NetApp Astra Trident in Ihrem Amazon EKS Cluster installiert. Außerdem haben Sie diese Trident Instanz konfiguriert, um Ihre Amazon FSX for NetApp ONTAP Instanz beim Bereitstellen und Managen von Storage-Ressourcen zu verwenden. Weitere Informationen finden Sie im "NetApp Astra Trident – Dokumentation".
- Zwischen dem On-Premises-Datacenter und der Virtual Private Cloud (VPC) in AWS muss eine bidirektionale Netzwerkverbindung hergestellt werden. Weitere Informationen zu den verschiedenen Optionen zur Implementierung finden Sie im "Amazon Virtual Private Network (VPN)-Dokumentation".

Domino Enterprise AI Platform in AWS installieren

Befolgen Sie die Anweisungen unter, um die Domino Enterprise MLOPS Platform in AWS zu installieren "Domino Administratorhandbuch". Sie müssen Domino in demselben Amazon EKS-Cluster bereitstellen, den Sie zuvor bereitgestellt haben. Außerdem muss NetApp Astra Trident bereits in diesem EKS-Cluster installiert und konfiguriert sein. Sie müssen in der Installationsdatei von Domino.yml eine von Trident gemanagte Storage-Klasse als Shared Storage-Klasse angeben.

> Siehe "Referenzhandbuch für die Domino-Installation" Für Details, wie Sie eine freigegebene Speicherklasse in Ihrer domino.yml-Installationsdatei angeben.

> "Technischer Bericht TR-4952" Erklärt die Implementierung von Domino in AWS mit Amazon FSX für NetApp ONTAP und kann als nützliche Referenz bei der Fehlerbehebung dienen.

Aktivieren Sie Domino Nexus

Als Nächstes müssen Sie Domino Nexus aktivieren. Siehe "Domino Administratorhandbuch" Entsprechende Details.

Implementieren Sie eine Domino Datenebene in Ihrem lokalen Datacenter

Als Nächstes müssen Sie eine Domino Datenebene in Ihrem lokalen Datacenter implementieren. Sie müssen diese Datenebene im zuvor bereitgestellten On-Premises-Kubernetes-Cluster implementieren. Außerdem muss NetApp Astra Trident bereits in diesem Kubernetes-Cluster installiert und konfiguriert sein. Siehe "Domino Administratorhandbuch" Entsprechende Details.

Offenlegung vorhandener NetApp Volumes für Domino

In diesem Abschnitt werden die Aufgaben beschrieben, die ausgeführt werden müssen, damit vorhandene NetApp ONTAP NFS-Volumes für die Domino MLOPS Plattform zur Verfügung gestellt werden können. Diese Schritte gelten sowohl für On-Premises- als auch für AWS.

Warum NetApp ONTAP Volumes für Domino offenlegen?

Der Einsatz von NetApp Volumes in Verbindung mit Domino bietet folgende Vorteile:

- Dank der Skalierungsfunktionen von NetApp ONTAP können Sie Workloads mit extrem großen Datensätzen ausführen.
- Sie können Workloads über mehrere Computing-Nodes hinweg ausführen, ohne Daten auf einzelne Nodes kopieren zu müssen.
- Sie können die NetApp Funktionen zum Verschieben und Synchronisieren von Daten in Hybrid-Multi-Clouds nutzen, um über diverse Datacenter und/oder Clouds hinweg auf Ihre Daten zuzugreifen.
- Es soll möglich sein, schnell und einfach einen Cache Ihrer Daten in einem anderen Rechenzentrum oder einer anderen Cloud zu erstellen.

Vorhandene NFS-Volumes, die nicht von Astra Trident bereitgestellt wurden, können offengelegt werden

Wenn Ihr vorhandenes NetApp ONTAP-NFS-Volume nicht mit Astra Trident bereitgestellt wurde, befolgen Sie die in diesem Unterabschnitt beschriebenen Schritte.

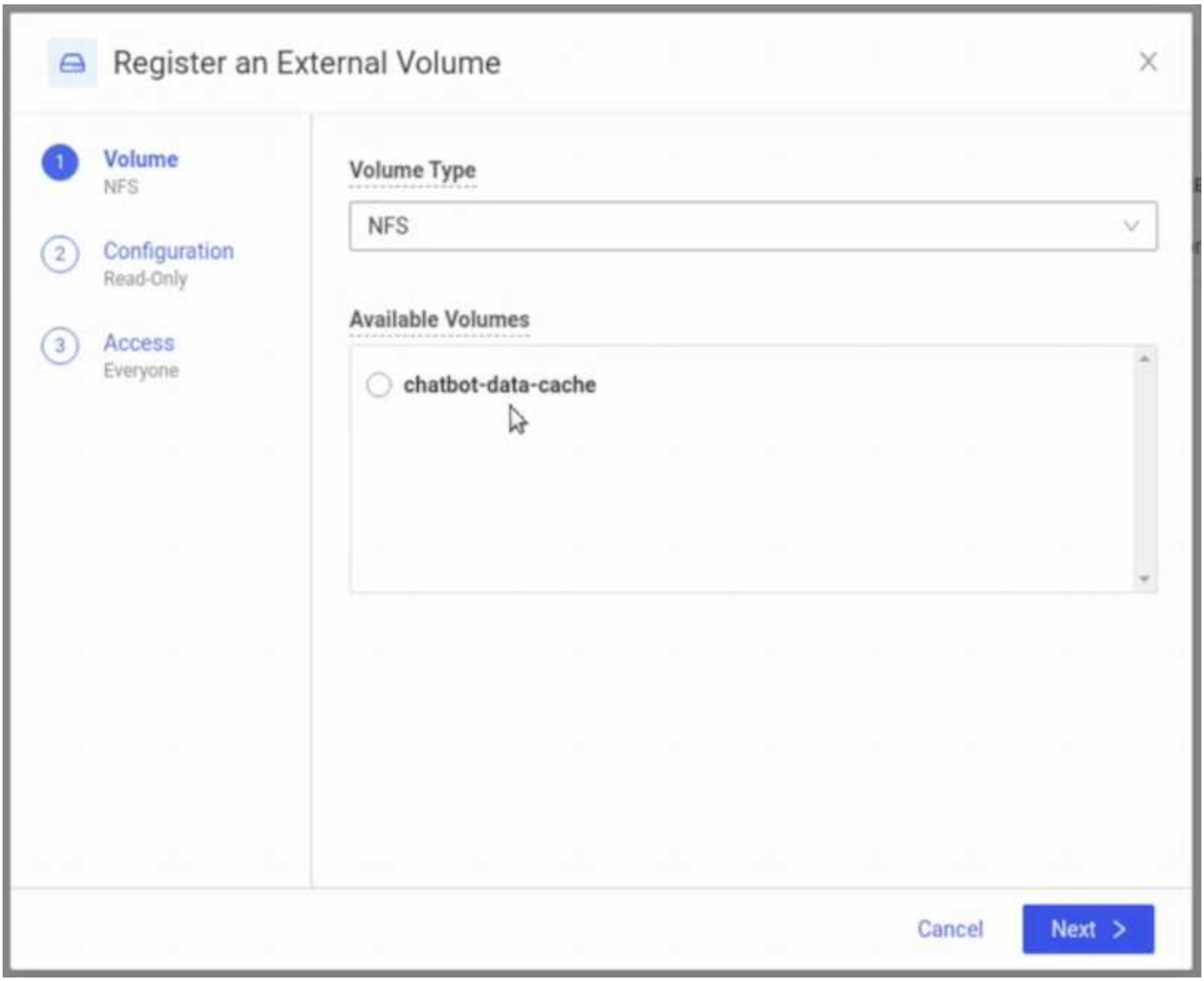
Erstellen Sie PV und PVC in Kubernetes

> Erstellen Sie für lokale Volumes das PV und die PVC in Ihrem lokalen Kubernetes-Cluster. Erstellen Sie für Amazon FSX for NetApp ONTAP Volumes das PV und die PVC in Amazon EKS.

Zuerst müssen Sie ein Persistent Volume (PV) und eine Persistent Volume Claim (PVC) in Ihrem Kubernetes-Cluster erstellen. Um das PV und die PVC zu erstellen, verwenden Sie die "BEISPIEL FÜR NFS PV/PVC" Aus dem Domino Admin Guide und aktualisieren Sie die Werte, um Ihre Umgebung zu reflektieren. Stellen Sie sicher, dass Sie die richtigen Werte für das angeben `namespace`, `nfs.path`, und `nfs.server` Felder. Darüber hinaus empfehlen wir, Ihren PV- und PVC-eindeutigen Namen anzugeben, die die Art der Daten darstellen, die auf dem entsprechenden ONTAP-NFS-Volume gespeichert sind. Wenn das Volumen beispielsweise Bilder von Herstellungsfehlern enthält, können Sie das PV nennen, `pv-mfg-defect-images`, Und die PVC, `pvc-mfg-defect-images`.

Externes Datenvolumen in Domino registrieren

Als Nächstes müssen Sie ein externes Datenvolumen in Domino registrieren. Informationen zur Registrierung eines externen Datenvolumens finden Sie im "Anweisungen" Im Domino-Administratorhandbuch. Wenn Sie das Volume registrieren, stellen Sie sicher, dass Sie „NFS“ aus dem Drop-Down-Menü „Volume Type“ auswählen. Nach der Auswahl von „NFS“ sollten Sie Ihre PVC in der Liste „Verfügbare Volumes“ sehen.



Legen Sie vorhandene Volumes offen, die von Astra Trident bereitgestellt wurden

Wenn Ihr vorhandenes Volume durch Astra Trident bereitgestellt wurde, befolgen Sie die in diesem Unterabschnitt beschriebenen Schritte.

Vorhandene PVC bearbeiten

Wenn Ihr Volume durch Astra Trident bereitgestellt wurde, verfügen Sie bereits über eine Persistent Volume Claim (PVC), die Ihrem Volume entspricht. Um dieses Volume Domino zur Verfügung zu stellen, müssen Sie die PVC bearbeiten und der Liste der Etiketten in die folgende Beschriftung hinzufügen `metadata.labels` Feld:

```
"dominodatalab.com/external-data-volume": "Generic"
```

Externes Datenvolumen in Domino registrieren

Als Nächstes müssen Sie ein externes Datenvolumen in Domino registrieren. Informationen zur Registrierung eines externen Datenvolumens finden Sie im "Anweisungen" Im Domino-Administratorhandbuch. Achten Sie beim Registrieren des Volumes darauf, im Dropdown-Menü „Volume Type“ die Option „Generic“ auszuwählen. Nachdem Sie „Allgemein“ ausgewählt haben, sollten Sie Ihre PVC in der Liste „Verfügbare Volumes“ sehen.

Zugriff auf dieselben Daten in unterschiedlichen Umgebungen

In diesem Abschnitt werden die Aufgaben beschrieben, die ausgeführt werden müssen, damit in verschiedenen Computing-Umgebungen auf dieselben Daten zugegriffen werden kann. In der Domino MLOPS Plattform werden Computing-Umgebungen als „Datenebenen“ bezeichnet. Befolgen Sie die in diesem Abschnitt beschriebenen Aufgaben, wenn sich Ihre Daten auf einem NetApp Volume in einer Datenebene befinden, aber in einer anderen Datenebene darauf zugreifen müssen. Diese Art von Szenario wird oft als „Bursting“ oder, wenn die Zielumgebung die Cloud ist, als „Cloud Bursting“ bezeichnet. Diese Funktion ist häufig erforderlich, wenn Computing-Ressourcen mit begrenzten oder überbelegten Ressourcen genutzt werden. Wenn Ihr On-Premises-Compute-Cluster beispielsweise überzeichnet ist, können Sie Workloads in die Cloud planen, wo sie sofort gestartet werden können.

Es gibt zwei empfohlene Optionen für den Zugriff auf ein NetApp Volume, das sich in einer anderen Datenebene befindet. Diese Optionen werden in den Unterabschnitten unten erläutert. Wählen Sie eine dieser Optionen je nach Ihren spezifischen Anforderungen aus. Die vor- und Nachteile der beiden Optionen werden in der folgenden Tabelle beschrieben.

Option	Vorteile	Nachteile
Option 1 - Cache	- Vereinfachter Workflow - Möglichkeit, eine Teilmenge von Daten auf der Grundlage der Anforderungen zwischenzuspeichern - Fähigkeit, Daten zurück zur Quelle zu schreiben - Keine Fernkopie zu verwalten	- Erhöhte Latenz beim ersten Datenzugriff als Cache hydratisiert wird.

Option	Vorteile	Nachteile
Option 2 - Spiegeln	- Vollständige Kopie des Quellvolumens - Keine erhöhte Latenz durch Cachehydratation (nach dem Spiegelvorgang abgeschlossen ist)	- Muss warten, bis der Spiegelvorgang abgeschlossen ist, bevor der Zugriff auf Daten erfolgt - Muss eine Remote-Kopie verwalten - Keine Fähigkeit, zurück zu Quelle zu schreiben

Option 1 - Erstellen Sie einen Cache eines Volumes, das sich in einer anderen Datenebene befindet

Mit "NetApp FlexCache Technologie", Sie können einen Cache eines NetApp-Volumes erstellen, das sich in einer anderen Datenebene befindet. Wenn Sie beispielsweise ein NetApp-Volume in Ihrer lokalen Datenebene haben und auf dieses Volume in der AWS-Datenebene zugreifen müssen, können Sie einen Cache des Volumes in AWS erstellen. In diesem Abschnitt werden die Aufgaben beschrieben, die beim Erstellen eines Cache eines NetApp Volumes in einer anderen Datenebene ausgeführt werden müssen.

Erstellen Sie ein FlexCache-Volume in der Zielumgebung

> Wenn die Zielumgebung Ihr On-Premises-Datacenter ist, erstellen Sie das FlexCache Volume in Ihrem lokalen ONTAP System. Wenn es sich bei der Zielumgebung um AWS handelt, erstellen Sie das FlexCache-Volume auf Ihrer Amazon FSX für NetApp ONTAP Instanz.

Zuerst müssen Sie ein FlexCache Volume in der Zielumgebung erstellen.

Wir empfehlen die Verwendung von BlueXP zur Erstellung des FlexCache Volumes. Folgen Sie den Anweisungen in der, um mit BlueXP ein FlexCache Volume zu erstellen "Dokumentation zum BlueXP Volume Caching".

Falls Sie BlueXP nicht verwenden möchten, können Sie das FlexCache Volume mithilfe von ONTAP System Manager oder der ONTAP CLI erstellen. Informationen zur Erstellung eines FlexCache Volume mit System Manager finden Sie in der Anleitung unter "ONTAP-Dokumentation". Informationen zum Erstellen eines FlexCache Volume mit der ONTAP CLI finden Sie in den Anweisungen in "ONTAP-Dokumentation".

Wenn Sie diesen Prozess automatisieren möchten, können Sie die verwenden "BlueXP API", Das "ONTAP REST API", Oder die "ONTAP Ansible Sammlung".

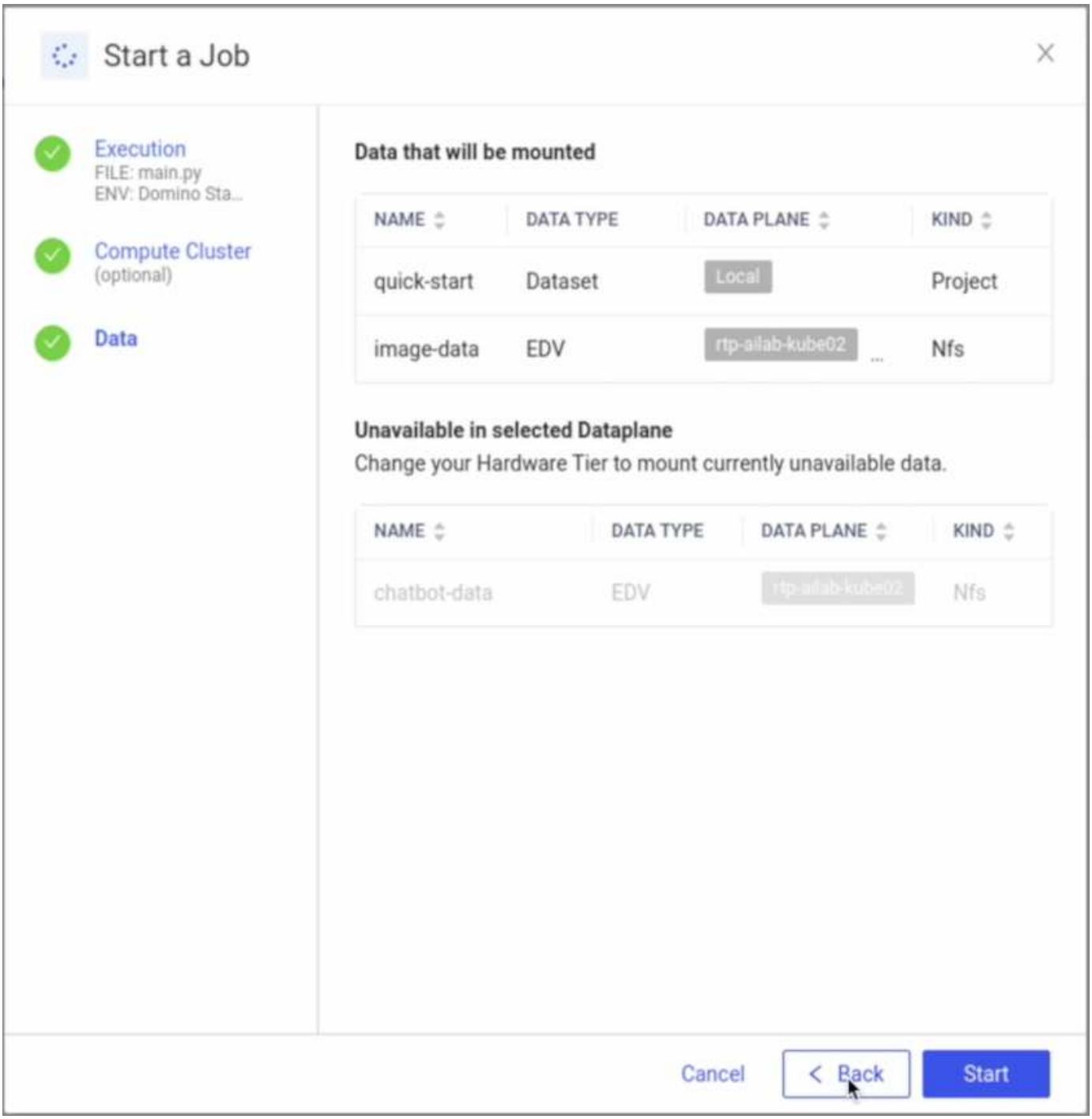
> System Manager ist in Amazon FSX for NetApp ONTAP nicht verfügbar.

FlexCache-Volume für Domino verfügbar machen

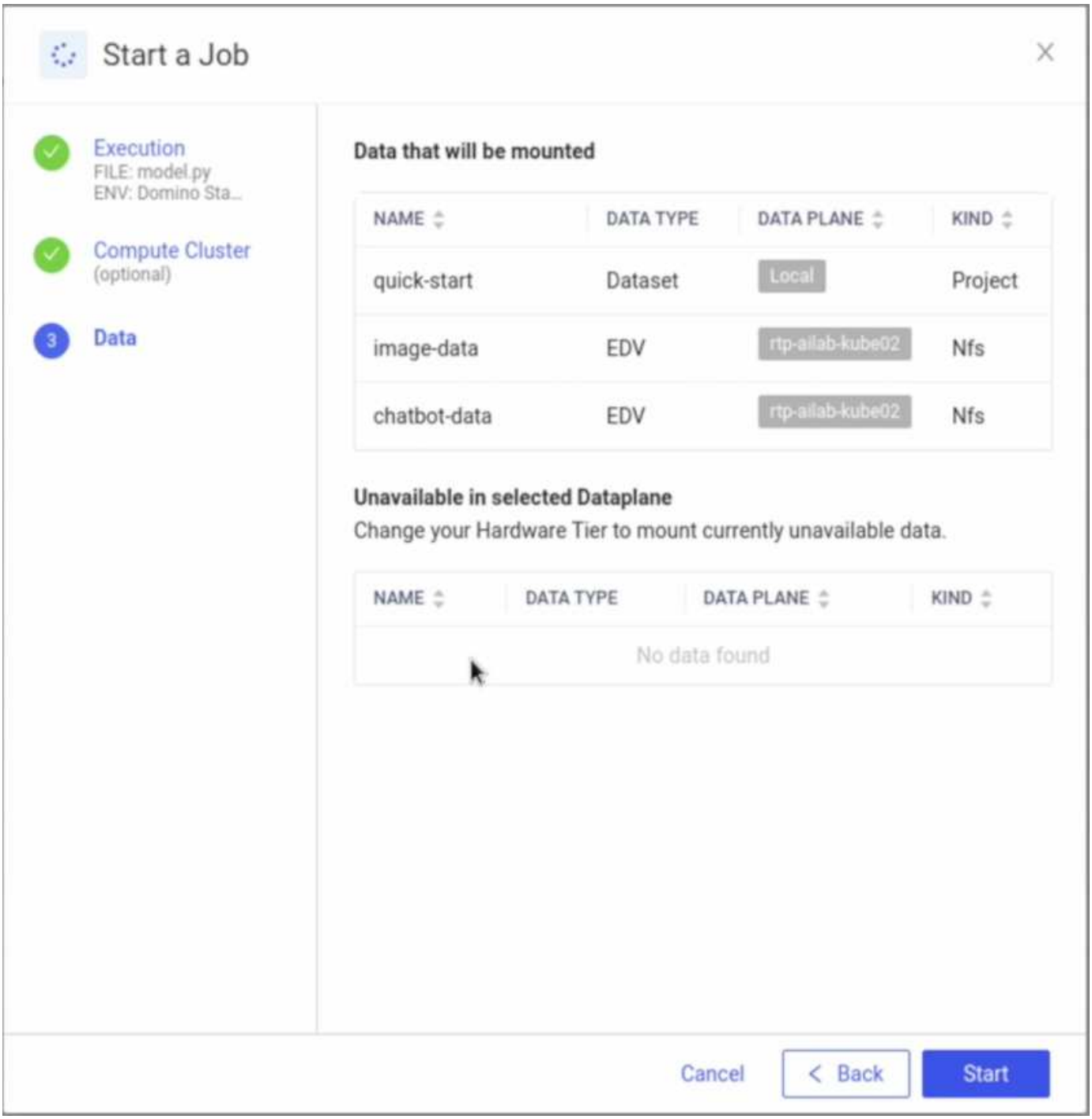
Als Nächstes müssen Sie das FlexCache-Volume der Domino MLOPS Plattform offenlegen. Um das FlexCache Volume für Domino offenzulegen, folgen Sie den Anweisungen im Unterabschnitt „Expose existing NFS Volumes that were not provisioned by Astra Trident“ des "Abschnitt „Expose Existing NetApp Volumes to Domino““ Dieser Lösung einsetzen.

Jetzt können Sie das FlexCache-Volume mounten, wenn Sie Jobs und Arbeitsbereiche auf der Zieldatenebene starten, wie in den folgenden Screenshots gezeigt.

Bevor Sie ein FlexCache-Volume erstellen



Nach der Offenlegung des FlexCache-Volumes gegenüber Domino



Option 2 - Replizieren Sie ein Volume, das sich in einer anderen Datenebene befindet

Mit "NetApp SnapMirror Datenreplizierungstechnologie", Sie können eine Kopie eines NetApp-Volumes erstellen, das sich in einer anderen Datenebene befindet. Wenn Sie beispielsweise ein NetApp-Volume in Ihrer lokalen Datenebene haben und auf dieses Volume in der AWS-Datenebene zugreifen müssen, können Sie eine Kopie des Volumes in AWS erstellen. In diesem Abschnitt werden die Aufgaben beschrieben, die beim Erstellen einer Kopie eines NetApp Volumes in einer anderen Datenebene ausgeführt werden müssen.

SnapMirror Beziehung erstellen

Zunächst müssen Sie eine SnapMirror Beziehung zwischen Ihrem Quell-Volume und einem neuen Ziel-Volume in der Zielumgebung erstellen. Beachten Sie, dass das Ziel-Volume als Teil des Prozesses zur Erstellung der SnapMirror Beziehung erstellt wird.

Wir empfehlen den Einsatz von BlueXP zum Aufbau der SnapMirror Beziehung. Folgen Sie den Anweisungen in der, um eine SnapMirror Beziehung mit BlueXP zu erstellen "Dokumentation zur BlueXP Replizierung".

Falls Sie BlueXP nicht nutzen möchten, können Sie die SnapMirror Beziehung mithilfe von ONTAP System Manager oder der ONTAP CLI erstellen. Informationen zur Erstellung einer SnapMirror Beziehung mit System Manager finden Sie in der Anleitung in "ONTAP-Dokumentation". Informationen zum Erstellen einer SnapMirror Beziehung mit der ONTAP CLI finden Sie in den Anweisungen in "ONTAP-Dokumentation".

Wenn Sie diesen Prozess automatisieren möchten, können Sie die verwenden "BlueXP API", Das "ONTAP REST API", Oder die "ONTAP Ansible Sammlung".

System Manager ist in Amazon FSX for NetApp ONTAP nicht verfügbar.

SnapMirror Beziehung unterbrechen

Als nächstes müssen Sie die SnapMirror Beziehung unterbrechen, um das Ziel-Volume für den Datenzugriff zu aktivieren. Warten Sie, bis die erste Replikation abgeschlossen ist, bevor Sie diesen Schritt durchführen.

Sie können feststellen, ob die Replizierung abgeschlossen ist, indem Sie den Spiegelungsstatus in BlueXP, ONTAP System Manager oder der ONTAP CLI überprüfen. Wenn die Replikation abgeschlossen ist, wird der Spiegelungsstatus „snapmirrored“ sein.

Wir empfehlen den Einsatz von BlueXP, um die Beziehung zu SnapMirror zu zerbrechen. Folgen Sie den Anweisungen in der, um eine SnapMirror Beziehung mit BlueXP zu unterbrechen "Dokumentation zur BlueXP Replizierung".

Falls Sie BlueXP nicht nutzen möchten, können Sie die SnapMirror Beziehung mit ONTAP System Manager oder der ONTAP CLI unterbrechen. Anweisungen zum Aufbrechen einer SnapMirror Beziehung mit System Manager finden Sie in der "ONTAP-Dokumentation". Informationen zum Aufbrechen einer SnapMirror Beziehung mit der ONTAP CLI finden Sie in den Anweisungen in "ONTAP-Dokumentation".

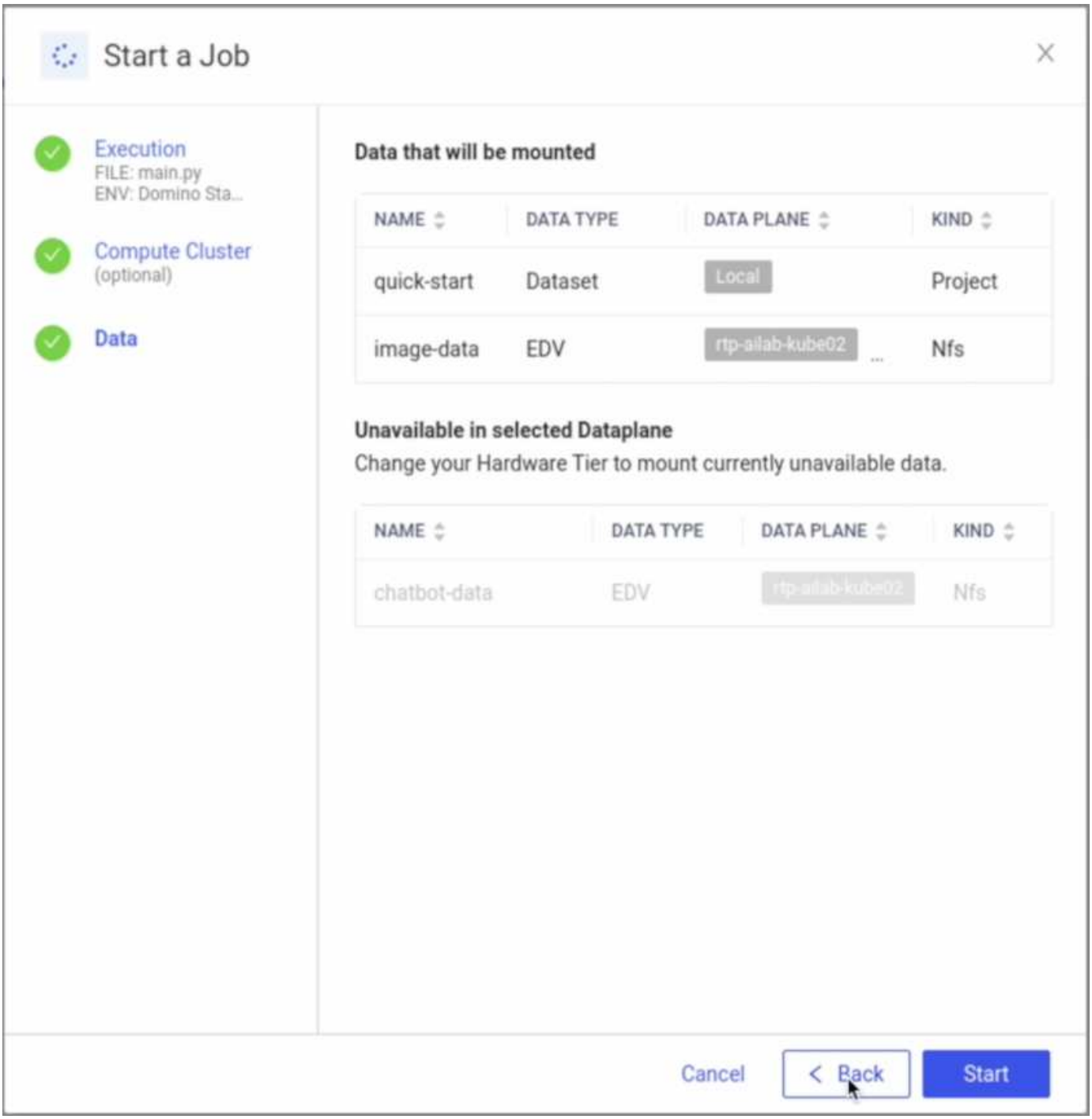
Wenn Sie diesen Prozess automatisieren möchten, können Sie die verwenden "BlueXP API", Das "ONTAP REST API", Oder die "ONTAP Ansible Sammlung".

Ziel-Volume für Domino verfügbar machen

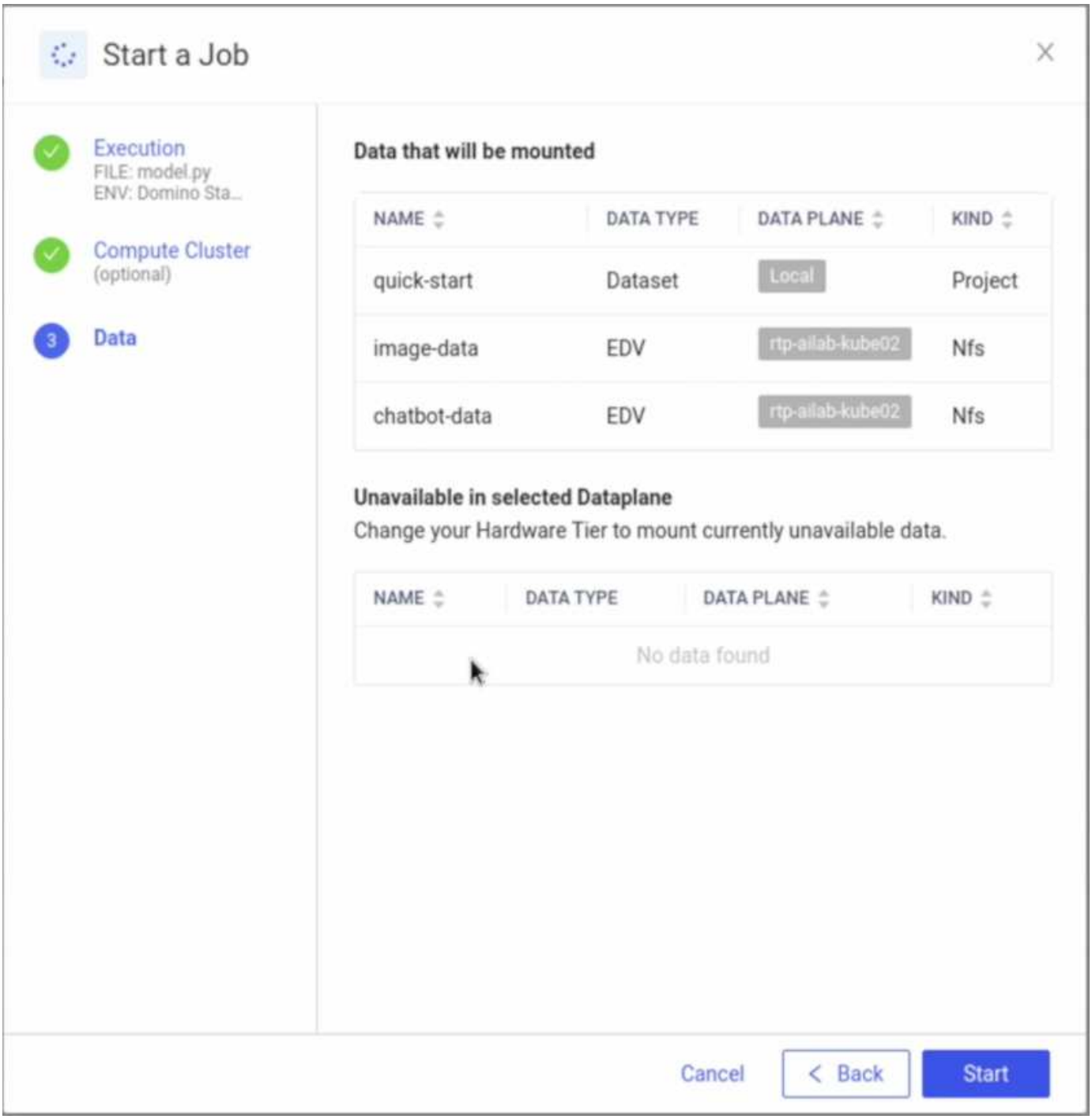
Als Nächstes müssen Sie das Zielvolume der Domino MLOps-Plattform zugänglich machen. Um das Ziel-Volume für Domino offenzulegen, folgen Sie den Anweisungen im Unterabschnitt „Expose existing NFS Volumes that were not provisioned by Astra Trident“ des "Abschnitt „Expose Existing NetApp Volumes to Domino““ Dieser Lösung einsetzen.

Jetzt können Sie das Zielvolume mounten, wenn Sie Jobs und Arbeitsbereiche auf der Zieldatenebene starten, wie in den folgenden Screenshots gezeigt.

Vor dem Erstellen der SnapMirror Beziehung



Nach der Belichtung des Zielvolume für Domino



Weitere Informationen

Weitere Informationen zu den in diesem Dokument beschriebenen Daten finden Sie in den folgenden Dokumenten bzw. auf den folgenden Websites:

- Domino Data Lab

  "https://domino.ai"

- Domino Nexus

  "https://domino.ai/platform/nexus"

- NetApp BlueXP

  "https://bluexp.netapp.com"

- NetApp ONTAP Datenmanagement-Software

  "https://www.netapp.com/data-management/ontap-data-management-software/"

- Mit den NetApp AI Lösungen

  "https://www.netapp.com/artificial-intelligence/"

Danksagungen

- Josh Mineroff, Director of SA, Tech Alliances, Domino Data Lab
- Nicholas Jablonski, Field CTO, Domino Data Lab
- Prabu Arjunan, Solution Architect bei NetApp
- Brian Young, Global Alliance Director, Technology Alliance Partners, NetApp

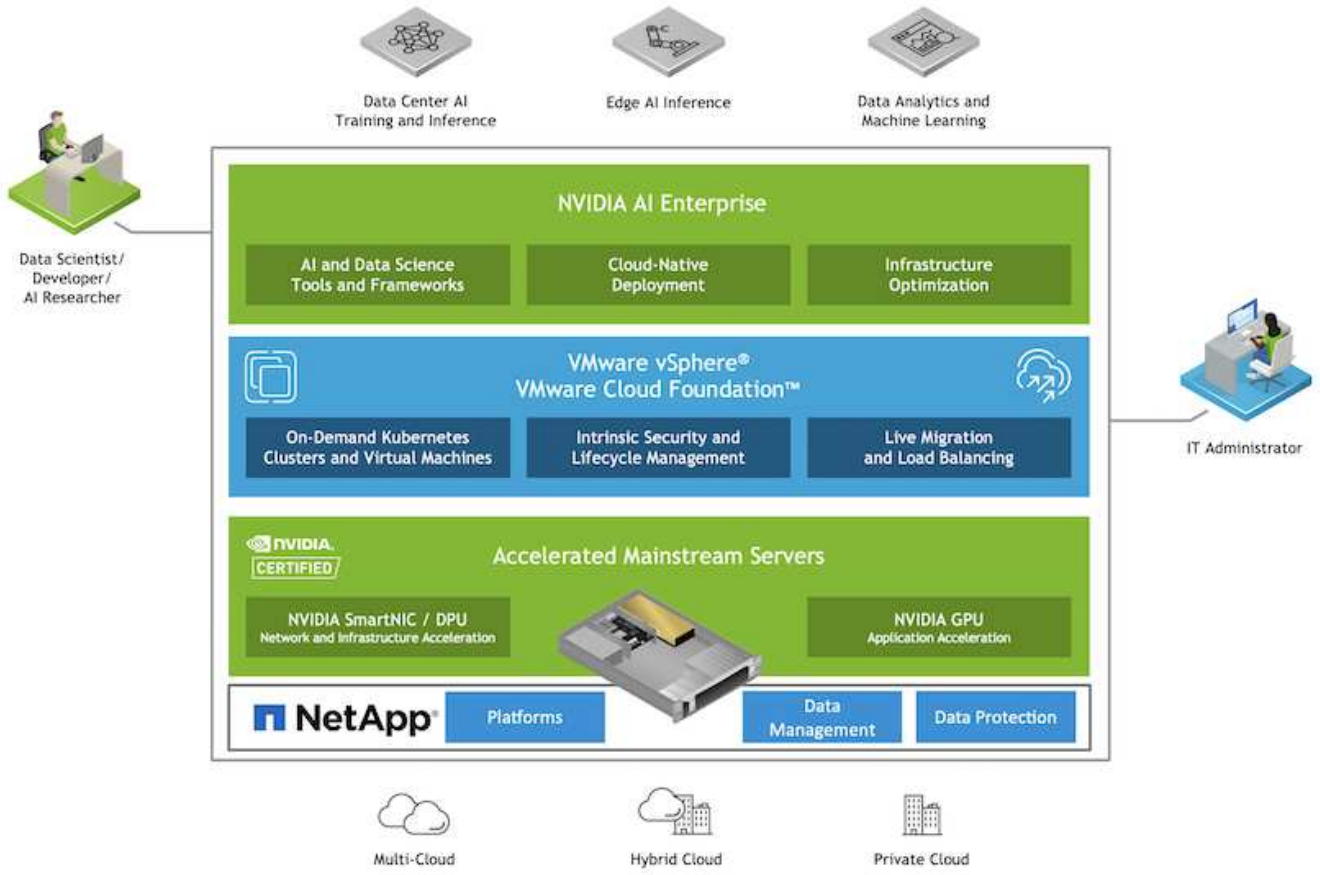
NVIDIA AI Enterprise with NetApp and VMware

NVIDIA AI Enterprise with NetApp and VMware

Mike Oglesby, NetApp

Für IT-Architekten und -Administratoren sind KI-Tools häufig kompliziert und können nicht kenne. Viele KI-Plattformen sind darüber hinaus nicht für die Enterprise-Klasse ausgelegt. NVIDIA AI Enterprise mit Unterstützung von NetApp und VMware wurde entwickelt, um eine optimierte KI-Architektur der Enterprise-Klasse bereitzustellen.

NVIDIA AI Enterprise ist eine End-to-End-Suite mit Cloud-nativer KI- und Datenanalyse-Software, die von NVIDIA für die Ausführung auf VMware vSphere mit NVIDIA-zertifizierten Systemen optimiert, zertifiziert und unterstützt wird. Diese Software vereinfacht die einfache und schnelle Implementierung, das einfache Management und die Skalierung von KI-Workloads in modernen Hybrid-Cloud-Umgebungen. NVIDIA AI Enterprise mit NetApp und VMware bietet KI-Workload und Datenmanagement der Enterprise-Klasse in einem einfachen, vertrauten Paket.



Technologischer Überblick

NVIDIA AI Enterprise

NVIDIA AI Enterprise ist eine End-to-End-Suite mit Cloud-nativer KI- und Datenanalyse-Software, die von NVIDIA für die Ausführung auf VMware vSphere mit NVIDIA-zertifizierten Systemen optimiert, zertifiziert und unterstützt wird. Diese Software vereinfacht die einfache und schnelle Implementierung, das einfache Management und die Skalierung von KI-Workloads in modernen Hybrid-Cloud-Umgebungen.

NVIDIA GPU CLOUD (NGC)

NVIDIA NGC hostet einen Katalog GPU-optimierter Software für KI-Fachleute, um ihre KI-Lösungen zu entwickeln. Zudem bietet die Lösung Zugriff auf verschiedene KI-Services wie NVIDIA Base Command für das Modelltraining, NVIDIA Fleet Command für die Implementierung und das Monitoring von Modellen sowie die NGC Private Registry, die einen sicheren Zugriff auf und Management proprietärer KI-Software ermöglicht. Zudem können NVIDIA KI Enterprise-Kunden Support über das NGC-Portal anfordern.

VMware vSphere

VMware vSphere ist die Virtualisierungsplattform von VMware, die Datacenter in aggregierte Computing-Infrastrukturen umwandelt, und zwar mit CPU-, Storage- und Netzwerkressourcen. VSphere verwaltet diese Infrastrukturen als einheitliche Betriebsumgebung und stellt Administratoren die Tools zum Management der an dieser Umgebung teilnehmenden Datacenter bereit.

Die beiden Kernkomponenten von vSphere sind ESXi und vCenter Server. ESXi ist die Virtualisierungsplattform, auf der Administratoren Virtual Machines und virtuelle Appliances erstellen und

ausführen. VCenter Server ist der Service, über den Administratoren mehrere Hosts managen können, die in einem Netzwerk und einem Pool von Host-Ressourcen verbunden sind.

NetApp ONTAP

ONTAP 9, die jüngste Generation der Storage-Managementsoftware von NetApp, ermöglicht Unternehmen eine Modernisierung der Infrastruktur und den Übergang zu einem Cloud-fähigen Datacenter. Dank der erstklassigen Datenmanagementfunktionen lassen sich mit ONTAP sämtliche Daten mit einem einzigen Toolset managen und schützen, ganz gleich, wo sich diese Daten befinden. Zudem können Sie die Daten problemlos dorthin verschieben, wo sie benötigt werden: Zwischen Edge, Core und Cloud. ONTAP 9 umfasst zahlreiche Funktionen, die das Datenmanagement vereinfachen, geschäftskritische Daten beschleunigen und schützen und Infrastrukturfunktionen der nächsten Generation über Hybrid-Cloud-Architekturen hinweg ermöglichen.

Vereinfachtes Datenmanagement

Für den Enterprise IT-Betrieb und die Data Scientists spielt Datenmanagement eine zentrale Rolle, damit für KI-Applikationen die entsprechenden Ressourcen zum Training von KI/ML-Datensätzen verwendet werden. Die folgenden zusätzlichen Informationen über NetApp Technologien sind bei dieser Validierung nicht im Umfang enthalten, können jedoch je nach Ihrer Implementierung relevant sein.

Die ONTAP Datenmanagement-Software umfasst die folgenden Funktionen, um den Betrieb zu optimieren und zu vereinfachen und damit Ihre Gesamtbetriebskosten zu senken:

- Inline-Data-Compaction und erweiterte Deduplizierung: Data-Compaction reduziert den ungenutzten Speicherplatz in Storage-Blöcken, während Deduplizierung die effektive Kapazität deutlich steigert. Dies gilt für lokal gespeicherte Daten und für Daten-Tiering in die Cloud.
- Minimale, maximale und adaptive Quality of Service (AQoS): Durch granulare QoS-Einstellungen (Quality of Service) können Unternehmen ihre Performance-Level für kritische Applikationen auch in Umgebungen mit vielen unterschiedlichen Workloads garantieren.
- NetApp FabricPool: Bietet automatisches Tiering von „kalten“ Daten in Private- und Public-Cloud-Storage-Optionen, einschließlich Amazon Web Services (AWS), Azure und NetApp StorageGRID Storage-Lösung. Weitere Informationen zu FabricPool finden Sie unter "TR-4598: FabricPool Best Practices".

Beschleunigung und Sicherung von Daten

ONTAP bietet überdurchschnittliche Performance und Datensicherung, erweitert diese Funktionen auf folgende Weise:

- Performance und niedrige Latenz: ONTAP bietet höchstmöglichen Durchsatz bei geringstmöglicher Latenz.
- Datensicherung ONTAP verfügt über integrierte Funktionen für die Datensicherung mit zentralem Management über alle Plattformen hinweg.
- NetApp Volume Encryption (NVE) ONTAP bietet native Verschlüsselung auf Volume-Ebene und unterstützt sowohl Onboard- als auch externes Verschlüsselungsmanagement.
- Multi-Faktor- und Multi-Faktor-Authentifizierung – ONTAP ermöglicht die gemeinsame Nutzung von Infrastrukturressourcen mit höchstmöglicher Sicherheit.

Zukunftssichere Infrastruktur

ONTAP bietet folgende Funktionen, um anspruchsvolle und sich ständig ändernde Geschäftsanforderungen zu erfüllen:

- Nahtlose Skalierung und unterbrechungsfreier Betrieb. Mit ONTAP sind das Hinzufügen von Kapazitäten

zu bestehenden Controllern und das Scale-out von Clustern unterbrechungsfrei möglich. Kunden können Upgrades auf die neuesten Technologien wie NVMe und 32 GB FC ohne teure Datenmigrationen oder Ausfälle durchführen.

- Cloud-Anbindung: ONTAP ist die Storage-Managementsoftware mit der umfassendsten Cloud-Integration und bietet Optionen für softwaredefinierten Storage (ONTAP Select) und Cloud-native Instanzen (NetApp Cloud Volumes Service) in allen Public Clouds.
- Integration in moderne Applikationen: ONTAP bietet Datenservices der Enterprise-Klasse für Plattformen und Applikationen der neuesten Generation, wie autonome Fahrzeuge, Smart Citys und Industrie 4.0, auf derselben Infrastruktur, die bereits vorhandene Unternehmensanwendungen unterstützt.

NetApp DataOps Toolkit

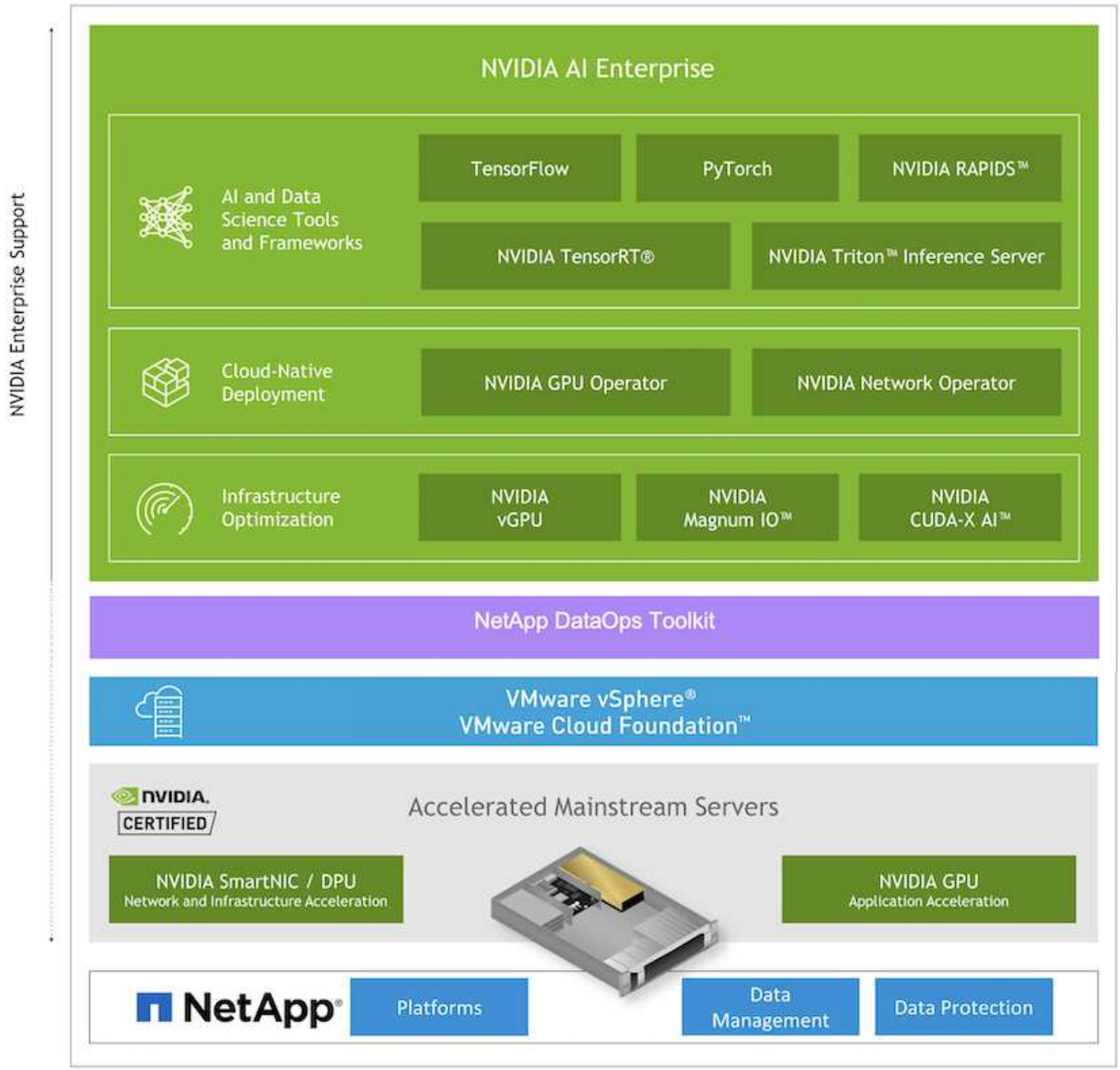
Das NetApp DataOps Toolkit ist ein Python-basiertes Tool zur Vereinfachung des Managements von Entwicklungs-/Trainings-Workspaces und Inferenzservern, die durch hochleistungsfähigen, horizontal skalierbaren NetApp Storage gesichert werden. Die wichtigsten Funktionen:

- Schnelle Bereitstellung neuer JupyterLab Workspaces mit hoher Kapazität, die durch hochperformanten horizontal skalierbaren NetApp Storage unterstützt werden
- Schnelle Bereitstellung neuer NVIDIA Triton Inferenz Server Instanzen, die durch NetApp Storage der Enterprise-Klasse unterstützt werden
- Klonen Sie JupyterLab Workspaces mit hoher Kapazität in nahezu instandem Tempo, um Experimente oder schnelle Iterationen zu ermöglichen.
- Speichern Sie nahezu instandlich Snapshots von JupyterLab Workspaces mit hoher Kapazität für Backup und/oder Rückverfolgbarkeit/Baselining.
- Provisionieren, Klonen und Snapshot High-Capacity-High-Performance-Daten-Volumes gleichzeitig

Der Netapp Architektur Sind

Die Lösung baut auf einer bewährten, vertrauten Architektur auf, die NetApp, VMware und NVIDIA-zertifizierte Systeme umfasst. Einzelheiten hierzu finden Sie in der folgenden Tabelle.

Komponente	Details
KI- und Data-Analytics-Software	"NVIDIA AI Enterprise für VMware"
Virtualisierungsplattform	"VMware vSphere"
Computing-Plattform	"Von NVIDIA zertifizierte Systeme"
Datenmanagementplattform	"NetApp ONTAP"



Ersteinrichtung

In diesem Abschnitt werden die anfänglichen Einrichtungsaufgaben beschrieben, die durchgeführt werden müssen, um NVIDIA AI Enterprise mit NetApp und VMware zu nutzen.

Voraussetzungen

Bevor Sie die in diesem Abschnitt beschriebenen Schritte durchführen, gehen wir davon aus, dass Sie VMware vSphere und NetApp ONTAP bereits implementiert haben. Siehe "NVIDIA AI Enterprise Product Support Matrix" Weitere Informationen zu unterstützten vSphere Versionen finden Sie unter. Siehe "Dokumentation der Lösungen von NetApp und VMware" Finden Sie Einzelheiten zur Implementierung von VMware vSphere mit NetApp ONTAP.

NVIDIA AI Enterprise Host Software installieren

Befolgen Sie zum Installieren der NVIDIA AI Enterprise Host-Software die Anweisungen in Abschnitt 1-4 in "NVIDIA AI Enterprise Quick Start Guide".

NVIDIA NGC Software

In diesem Abschnitt werden die Aufgaben beschrieben, die ausgeführt werden müssen, um NVIDIA NGC Enterprise-Software in einer NVIDIA KI Enterprise-Umgebung zu verwenden.

Einrichtung

In diesem Abschnitt werden die anfänglichen Einrichtungsaufgaben beschrieben, die durchgeführt werden müssen, um NVIDIA NGC Enterprise-Software in einer NVIDIA KI Enterprise-Umgebung zu nutzen.

Voraussetzungen

Bevor Sie die in diesem Abschnitt beschriebenen Schritte ausführen, gehen wir davon aus, dass Sie die NVIDIA AI Enterprise Host-Software bereits implementiert haben. Befolgen Sie dazu die Anweisungen auf der "Ersteinrichtung" Seite.

Erstellen einer Ubuntu Gast-VM mit vGPU

Zunächst müssen Sie eine Ubuntu 20.04 Gast-VM mit vGPU erstellen. Um eine Ubuntu 20.04 Gast-VM mit vGPU zu erstellen, befolgen Sie die Anweisungen in der "NVIDIA AI Enterprise Deployment Guide".

Laden Sie die NVIDIA Gast-Software herunter und installieren Sie sie

Als Nächstes müssen Sie die erforderliche NVIDIA-Gastsoftware auf der Gast-VM installieren, die Sie im vorherigen Schritt erstellt haben. Befolgen Sie zum Herunterladen und Installieren der erforderlichen NVIDIA-Gastsoftware auf der Gast-VM die Anweisungen in den Abschnitten 5.1-5.4 in "NVIDIA AI Enterprise Quick Start Guide".

> Wenn Sie die in Abschnitt 5.4 beschriebenen Verifizierungsaufgaben durchführen, müssen Sie möglicherweise ein anderes CUDA Container Image Version-Tag verwenden, da das CUDA Container-Image seit dem Schreiben des Handbuchs aktualisiert wurde. In unserer Validierung haben wir 'nvidia/cuda:11.0.3-base-ubuntu20.04' verwendet.

KI-/Analytics-Framework-Container herunterladen

Als Nächstes müssen Sie benötigte KI- oder Analyse-Framework-Container-Images von NVIDIA NGC herunterladen, damit diese in Ihrer Gast-VM verfügbar sein können. Befolgen Sie zum Herunterladen von Framework-Containern innerhalb der Gast-VM die Anweisungen im "NVIDIA AI Enterprise Deployment Guide".

Installation und Konfiguration des NetApp DataOps Toolkit

Als Nächstes müssen Sie das NetApp DataOps Toolkit für herkömmliche Umgebungen innerhalb der Gast-VM installieren. Das NetApp DataOps Toolkit kann zur Verwaltung von horizontal skalierbaren Daten-Volumes auf Ihrem ONTAP System direkt aus dem Terminal innerhalb der Gast-VM verwendet werden. Führen Sie die folgenden Aufgaben aus, um das NetApp DataOps Toolkit in der Gast-VM zu installieren.

1. Installieren Sie pip.

```
$ sudo apt update
$ sudo apt install python3-pip
$ python3 -m pip install netapp-dataops-traditional
```

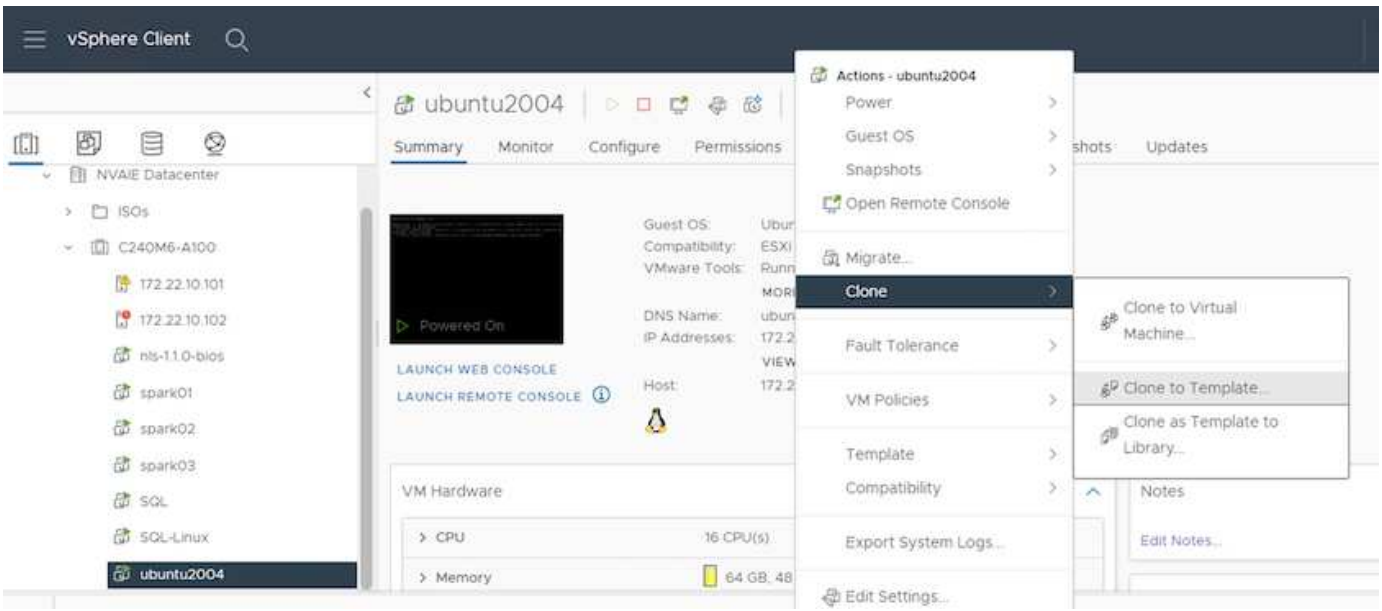
2. Melden Sie sich vom Gast-VM-Terminal ab und melden Sie sich dann erneut an.
3. Konfigurieren Sie das NetApp DataOps Toolkit. Um diesen Schritt abzuschließen, benötigen Sie API-Zugriffsdetails für Ihr ONTAP-System. Möglicherweise müssen Sie diese Informationen von Ihrem Storage-Administrator beziehen.

```
$ netapp_dataops_cli.py config

Enter ONTAP management LIF hostname or IP address (Recommendation: Use
SVM management interface): 172.22.10.10
Enter SVM (Storage VM) name: NVAIE-client
Enter SVM NFS data LIF hostname or IP address: 172.22.13.151
Enter default volume type to use when creating new volumes
(flexgroup/flexvol) [flexgroup]:
Enter export policy to use by default when creating new volumes
[default]:
Enter snapshot policy to use by default when creating new volumes
[none]:
Enter unix filesystem user id (uid) to apply by default when creating
new volumes (ex. '0' for root user) [0]:
Enter unix filesystem group id (gid) to apply by default when creating
new volumes (ex. '0' for root group) [0]:
Enter unix filesystem permissions to apply by default when creating new
volumes (ex. '0777' for full read/write permissions for all users and
groups) [0777]:
Enter aggregate to use by default when creating new FlexVol volumes:
aff_a400_01_NVME_SSD_1
Enter ONTAP API username (Recommendation: Use SVM account): admin
Enter ONTAP API password (Recommendation: Use SVM account):
Verify SSL certificate when calling ONTAP API (true/false): false
Do you intend to use this toolkit to trigger BlueXP Copy and Sync
operations? (yes/no): no
Do you intend to use this toolkit to push/pull from S3? (yes/no): no
Created config file: '/home/user/.netapp_dataops/config.json'.
```

Erstellen einer Gast-VM-Vorlage

Und schließlich müssen Sie eine VM-Vorlage auf der Grundlage Ihrer Gast-VM erstellen. Sie können diese Vorlage verwenden, um schnell Gast-VMs zur Nutzung der NVIDIA NGC Software zu erstellen.

Um eine VM-Vorlage auf der Grundlage Ihrer Gast-VM zu erstellen, melden Sie sich bei VMware vSphere an, klicken Sie einfach auf den Namen der Gast-VM, wählen Sie „Klonen“, wählen Sie „Vorlage klonen…“ und folgen Sie dann dem Assistenten.



Beispiel: Anwendungsfall – TensorFlow Training Job

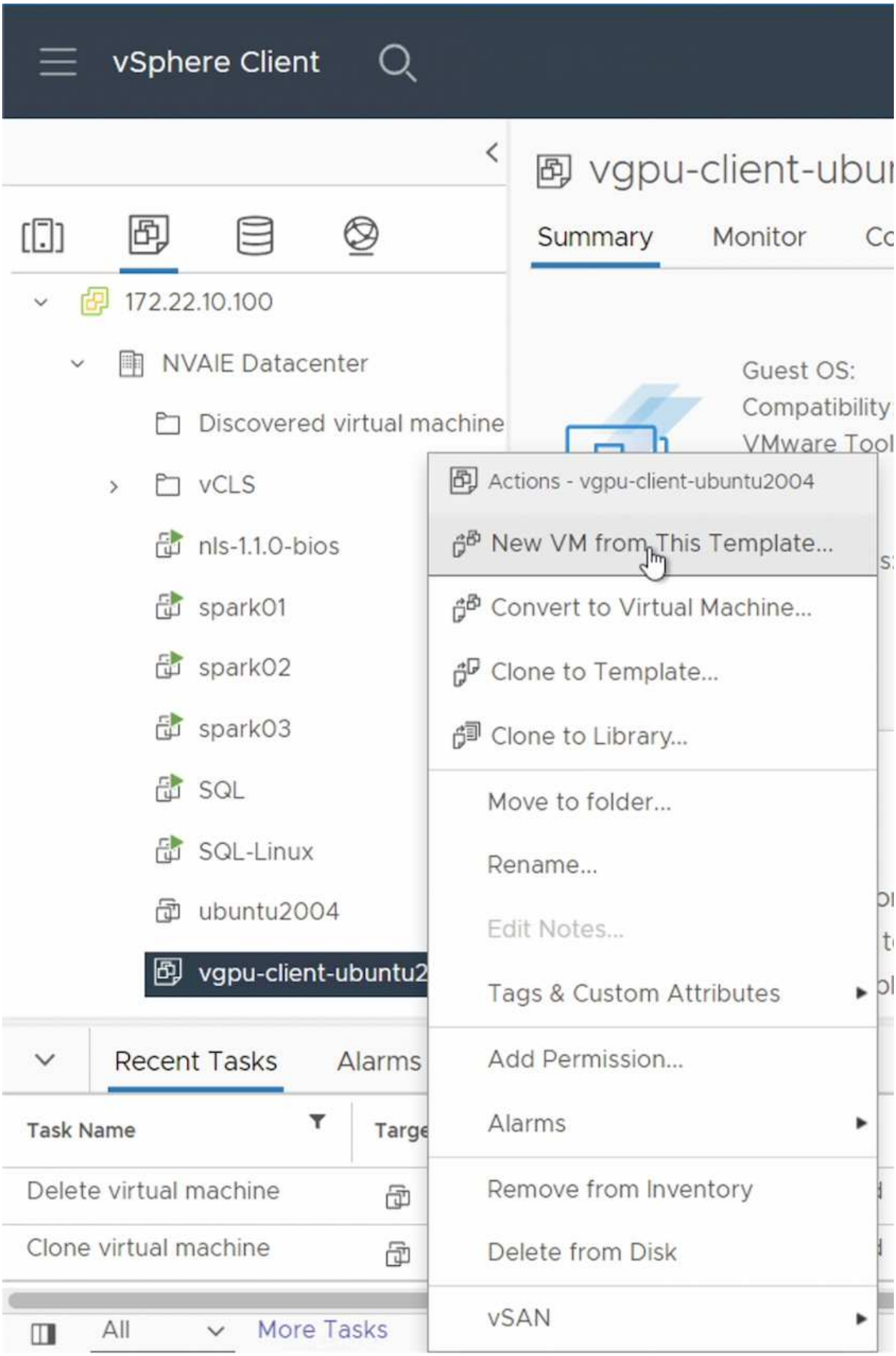
In diesem Abschnitt werden die Aufgaben beschrieben, die ausgeführt werden müssen, um einen TensorFlow-Trainingsjob in einer NVIDIA AI Enterprise-Umgebung auszuführen.

Voraussetzungen

Bevor Sie die in diesem Abschnitt beschriebenen Schritte durchführen, gehen wir davon aus, dass Sie bereits eine Gast-VM-Vorlage erstellt haben, und befolgen Sie die Anweisungen auf dem "Einrichtung" Seite.

Gast-VM aus Vorlage erstellen

Zunächst müssen Sie eine neue Gast-VM auf der Vorlage erstellen, die Sie im vorherigen Abschnitt erstellt haben. Um eine neue Gast-VM von Ihrer Vorlage zu erstellen, melden Sie sich bei VMware vSphere an, klicken Sie einfach auf den Vorlagennamen, wählen Sie „Neue VM aus dieser Vorlage…“ und folgen Sie dann dem Assistenten.


vSphere Client
vgpu-client-ubu
Summary
Monitor
172.22.10.100
NVAIE Datacenter
Discovered virtual machine
vCLS
nls-1.1.0-bios
spark01
spark02
spark03
SQL
SQL-Linux
ubuntu2004
vgpu-client-ubuntu2
Guest OS:
Compatibility
VMware Tool
Actions - vgpu-client-ubuntu2004
New VM from This Template...
Convert to Virtual Machine...
Clone to Template...
Clone to Library...
Move to folder...
Rename...
Edit Notes...
Tags & Custom Attributes
Add Permission...
Alarms
Remove from Inventory
Delete from Disk
vSAN
Recent Tasks
Alarms
Task Name
Delete virtual machine
Clone virtual machine
All
More Tasks

Erstellen und Mounten des Daten-Volumes

Als Nächstes müssen Sie ein neues Datenvolumen erstellen, auf dem Sie Ihren Trainingsdatensatz speichern können. Mit dem NetApp DataOps Toolkit können Sie schnell ein neues Datenvolumen erstellen. Der folgende Beispiel-Befehl zeigt die Erstellung eines Volumes mit dem Namen „imagenet“ mit einer Kapazität von 2 TB.

```
$ netapp_dataops_cli.py create vol -n imagenet -s 2TB
```

Bevor Sie Ihr Daten-Volume mit den Daten füllen können, müssen Sie es innerhalb der Gast-VM mounten. Mit dem NetApp DataOps Toolkit können Sie ein Datenvolumen schnell einbinden. Der folgende Beispielbefehl zeigt das Mounten des Volumes, das im vorherigen Schritt erstellt wurde.

```
$ sudo -E netapp_dataops_cli.py mount vol -n imagenet -m ~/imagenet
```

Daten-Volume Füllen

Nachdem das neue Volume bereitgestellt und angehängt wurde, kann der Trainingsdatensatz vom Quellspeicherort abgerufen und auf dem neuen Volume abgelegt werden. Dabei geht es in der Regel um das Abrufen der Daten von einem S3- oder Hadoop-Data Lake, was gelegentlich auch von einem Data Engineer ermöglicht wird.

Führen Sie TensorFlow Trainingsjob aus

Sie sind jetzt bereit, Ihre TensorFlow-Schulungsaufgabe auszuführen. Führen Sie die folgenden Aufgaben aus, um Ihren TensorFlow-Schulungsjob auszuführen.

1. Rufen Sie das Image des NVIDIA NGC Enterprise TensorFlow Containers auf.

   ```
   $ sudo docker pull nvcr.io/nvaie/tensorflow-2-1:22.05-tf1-nvaie-2.1-py3
   ```

2. Starten Sie eine Instanz des NVIDIA NGC Enterprise TensorFlow Containers. Verwenden Sie die Option '-V', um Ihr Datenvolumen an den Container anzuhängen.

   ```
   $ sudo docker run --gpus all -v ~/imagenet:/imagenet -it --rm
   nvcr.io/nvaie/tensorflow-2-1:22.05-tf1-nvaie-2.1-py3
   ```

3. Führen Sie Ihr TensorFlow-Trainingsprogramm im Container aus. Der folgende Beispielbefehl zeigt die Ausführung eines Beispielprogramms ResNet-50, das im Container-Image enthalten ist.

   ```
   $ python ./nvidia-examples/cnn/resnet.py --layers 50 -b 64 -i 200 -u
   batch --precision fp16 --data_dir /imagenet/data
   ```

Weitere Informationen

Weitere Informationen zu den in diesem Dokument beschriebenen Daten finden Sie in

den folgenden Dokumenten bzw. auf den folgenden Websites:

- NetApp ONTAP Datenmanagement-Software – ONTAP Informationsbibliothek

  http://mysupport.netapp.com/documentation/productlibrary/index.html?productID=62286

- NetApp DataOps Toolkit

  "https://github.com/NetApp/netapp-dataops-toolkit"

- NVIDIA AI Enterprise with VMware

  https://www.nvidia.com/en-us/data-center/products/ai-enterprise/vmware/^]

Danksagungen

- Bobby Oommen, Sr Manager, NetApp
- Ramesh Isaac, Systemadministrator bei NetApp
- Roney Daniel, Technical Marketing Engineer, NetApp

TR-4851: NetApp StorageGRID Data Lake für Workloads beim autonomen Fahren – Lösungsdesign

David Arnette, NetApp

TR-4851 veranschaulicht die Verwendung von NetApp StorageGRID Objekt-Storage als Daten-Repository und Managementsystem für die Softwareentwicklung Machine Learning (ML) und Deep Learning (DL). In diesem Whitepaper werden der Datenfluss und die Anforderungen bei der Entwicklung der Software für autonome Fahrzeuge sowie die Funktionen von StorageGRID zur Optimierung des Daten-Lebenszyklus beschrieben. Diese Lösung gilt für alle mehrstufigen Daten-Pipeline-Workflows, die in DEN ML- und DL-Entwicklungsprozessen typisch sind.

"TR-4851: NetApp StorageGRID Data Lake für Workloads beim autonomen Fahren – Lösungsdesign"

NetApp AI Control Plane

TR-4798: NetApp AI Control Plane

Mike Oglesby, NetApp

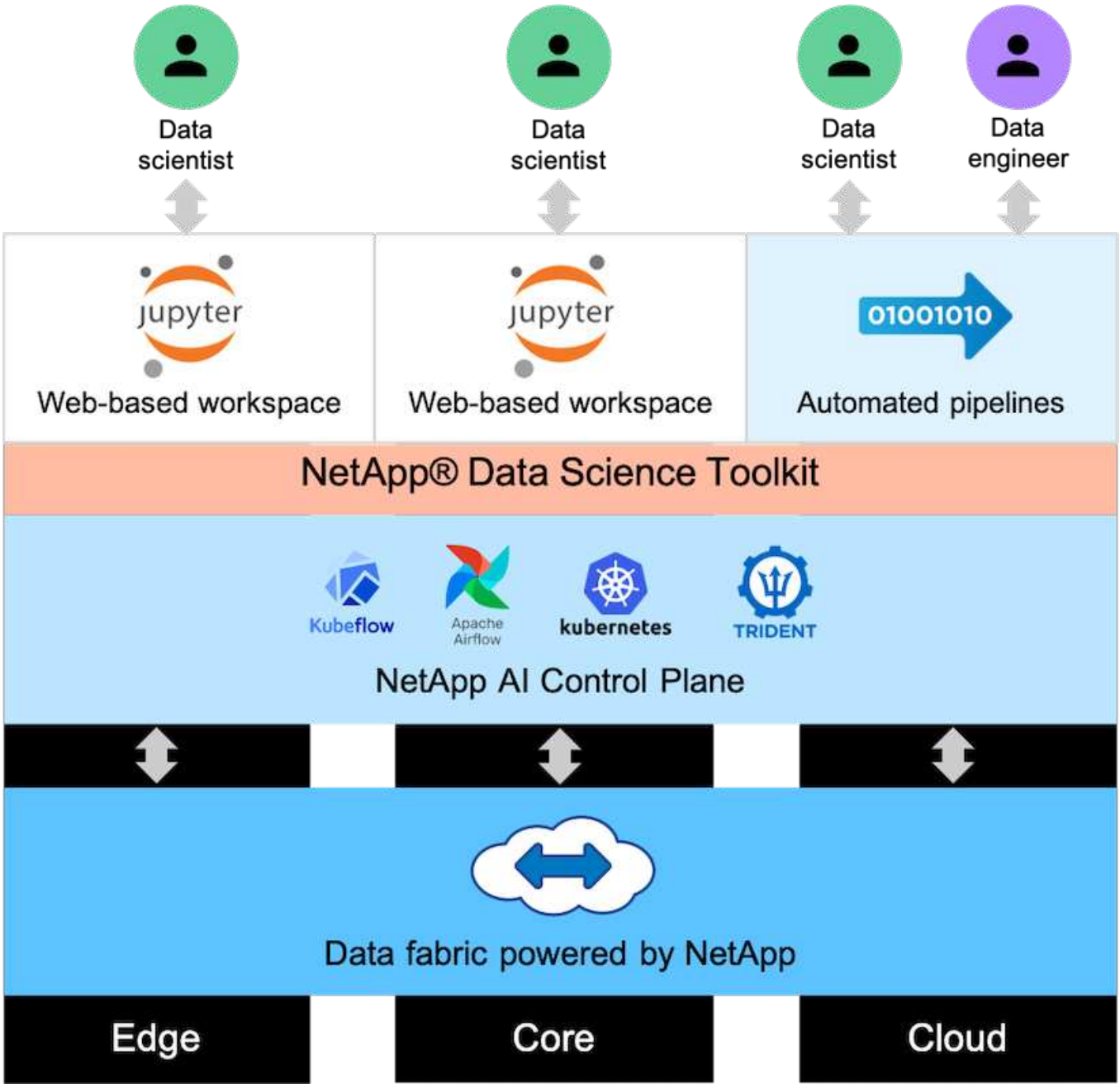
Unternehmen aller Größen und Branchen setzen zunehmend auf künstliche Intelligenz (KI), maschinelles Lernen (ML) und Deep Learning (DL), um Probleme aus der Praxis zu lösen, innovative Produkte und Services bereitzustellen und sich in einem immer härter umkämpften Markt einen Schritt voraus zu sein. Beim verstärkten Einsatz von KI, ML und DL müssen Unternehmen mit vielen Herausforderungen konfrontiert werden, darunter Workload-Skalierbarkeit und Datenverfügbarkeit. In diesem Dokument wird veranschaulicht, wie Sie diese Herausforderungen mit der NetApp AI Control Plane

bewältigen können. Diese Lösung vereint die Datenmanagement-Funktionen von NetApp mit gängigen Open-Source-Tools und -Frameworks.

Dieser Bericht zeigt Ihnen, wie Sie einen Daten-Namespace schnell klonen können. Außerdem zeigt sie, wie Daten nahtlos über Standorte und Regionen hinweg repliziert werden, um eine zusammenhängende und einheitliche KI/ML/DL-Datenpipeline zu erstellen. Darüber hinaus werden die Trainings-Workflows für AI, ML und DL definiert und implementiert, die die nahezu sofortige Erstellung von Daten und Modellbasiden für Rückverfolgbarkeit und Versionierung enthalten. Mit dieser Lösung können Sie verfolgen, welches Modell-Training zurück zu dem genauen Datensatz führt, der zum Training und/oder zur Validierung des Modells verwendet wurde. Abschließend erfahren Sie in diesem Dokument, wie Sie Jupyter Notebook-Workspaces mit Zugriff auf riesige Datensätze schnell bereitstellen können.

Hinweis: Für verteilte Schulungen im HPC-Stil mit einer großen Anzahl von GPU-Servern, die einen gemeinsamen Zugriff auf denselben Datensatz benötigen oder wenn Sie ein paralleles Dateisystem benötigen/bevorzugen, schauen Sie sich das hier an "TR-4890". In diesem technischen Bericht wird die Vorgehensweise erläutert "Die vollständig unterstützte parallele NetApp Lösung BeeGFS von NetApp" Teil der NetApp KI-Kontrollebene. Diese Lösung wurde entwickelt, um von einer Handvoll NVIDIA DGX A100-Systemen bis zu einem voll aufgeblasen SuperPOD mit 140 Nodes skalieren zu können.

Die NetApp AI Control Plane ist auf Data Scientists und Data Engineers ausgerichtet und benötigt daher nur eine minimale NetApp oder NetApp ONTAP® Expertise. Bei dieser Lösung lassen sich Datenmanagementfunktionen mit einfachen und vertrauten Tools und Schnittstellen ausführen. Wenn Sie bereits NetApp Storage in Ihrer Umgebung haben, können Sie noch heute die NetApp AI Control Plane testen. Wenn Sie die Lösung testen möchten, aber noch nicht bereits über NetApp Storage verfügen, besuchen Sie "cloud.netapp.com", Und Sie sind innerhalb von Minuten mit einer Cloud-basierten NetApp Storage-Lösung einsatzbereit. Die folgende Abbildung bietet eine Visualisierung der Lösung.



Konzepte und Komponenten

Künstliche Intelligenz

KI ist eine Informatik-Disziplin, in der Computer trainiert werden, um kognitive Funktionen des menschlichen Verstandes nachzuahmen. KI-Entwickler Schulen Computer, um Probleme so zu lernen oder zu lösen, dass sie dem Menschen ähneln oder sogar überlegen sind. Deep Learning und Machine Learning sind Unterfelder der KI. Unternehmen setzen zunehmend auf KI, ML und DL, um ihre kritischen Geschäftsanforderungen zu unterstützen. Dies sind einige Beispiele:

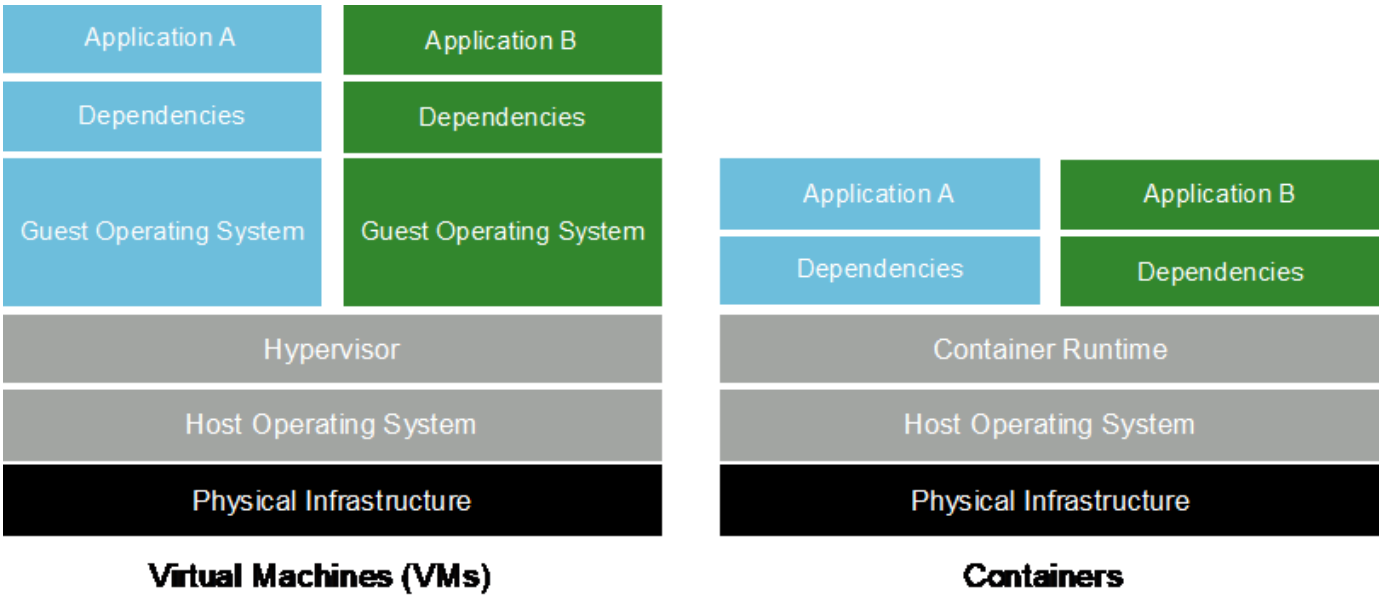
- Analyse großer Datenmengen für bislang unbekannte geschäftliche Einblicke
- Direkte Interaktion mit Kunden über natürliche Sprachverarbeitung
- Automatisierung verschiedener Geschäftsprozesse und Funktionen

Moderne KI-Trainings- und Inferenz-Workloads erfordern extrem hohe parallele Computing-Funktionen. Deshalb werden GPUs zunehmend zur Ausführung von KI-Operationen eingesetzt, da die Funktionen der parallelen Verarbeitung von GPUs denen allgemeiner CPUs überlegen sind.

Container

Container sind isolierte Instanzen von Benutzerspeicherplatz, die auf einem Kernel des Shared-Host-Betriebssystems laufen. Die Einführung von Containern nimmt immer schneller zu. Container bieten viele der gleichen Vorteile von Applikationen im Sandbox-Bereich, die Virtual Machines (VMs) bieten. Da jedoch der Hypervisor und das Gastbetriebssystem die Anzahl der VMs beseitigen, sind die Container viel schlanker. Die folgende Abbildung zeigt eine Visualisierung von Virtual Machines gegenüber Containern.

Container erlauben außerdem die effiziente Bündelung von Applikationsabhängigkeiten, Laufzeiten usw. und damit direkt mit einer Applikation. Das am häufigsten verwendete Format für Containerverpackungen ist der Docker Container. Eine Applikation, die im Docker-Container-Format gesichert wurde, kann auf jeder Maschine ausgeführt werden, die Docker Container ausführen kann. Dies gilt auch dann, wenn die Abhängigkeiten der Anwendung nicht auf der Maschine vorhanden sind, weil alle Abhängigkeiten im Container selbst verpackt sind. Weitere Informationen finden Sie auf der "Docker-Website".



Kubernetes

Kubernetes ist eine ursprünglich von Google entwickelte Open-Source-Plattform zur Container-Orchestrierung, die jetzt von der Cloud Native Computing Foundation (CNCF) verwaltet wird. Kubernetes ermöglicht die Automatisierung von Implementierungs-, Management- und Skalierungsfunktionen für Container-Applikationen. In den letzten Jahren hat sich Kubernetes zur führenden Plattform für die Container-Orchestrierung entwickelt. Obwohl andere Container-Verpackungsformate und Laufzeiten unterstützt werden, wird Kubernetes am häufigsten als Orchestrierungssystem für Docker Container verwendet. Weitere Informationen finden Sie auf der "Kubernetes-Website".

NetApp Trident

Trident ist ein Open-Source-Storage-Orchestrator, der von NetApp entwickelt und gewartet wird. Damit wird die Erstellung, das Management und die Nutzung von persistentem Storage für Kubernetes-Workloads erheblich vereinfacht. Trident, selbst eine native Kubernetes-Applikation, wird direkt in einem Kubernetes Cluster ausgeführt. Mit Trident können Kubernetes-Benutzer (Entwickler, Data Scientists, Kubernetes Administratoren usw.) persistente Storage-Volumes im gewohnten Kubernetes-Standardformat erstellen, managen und

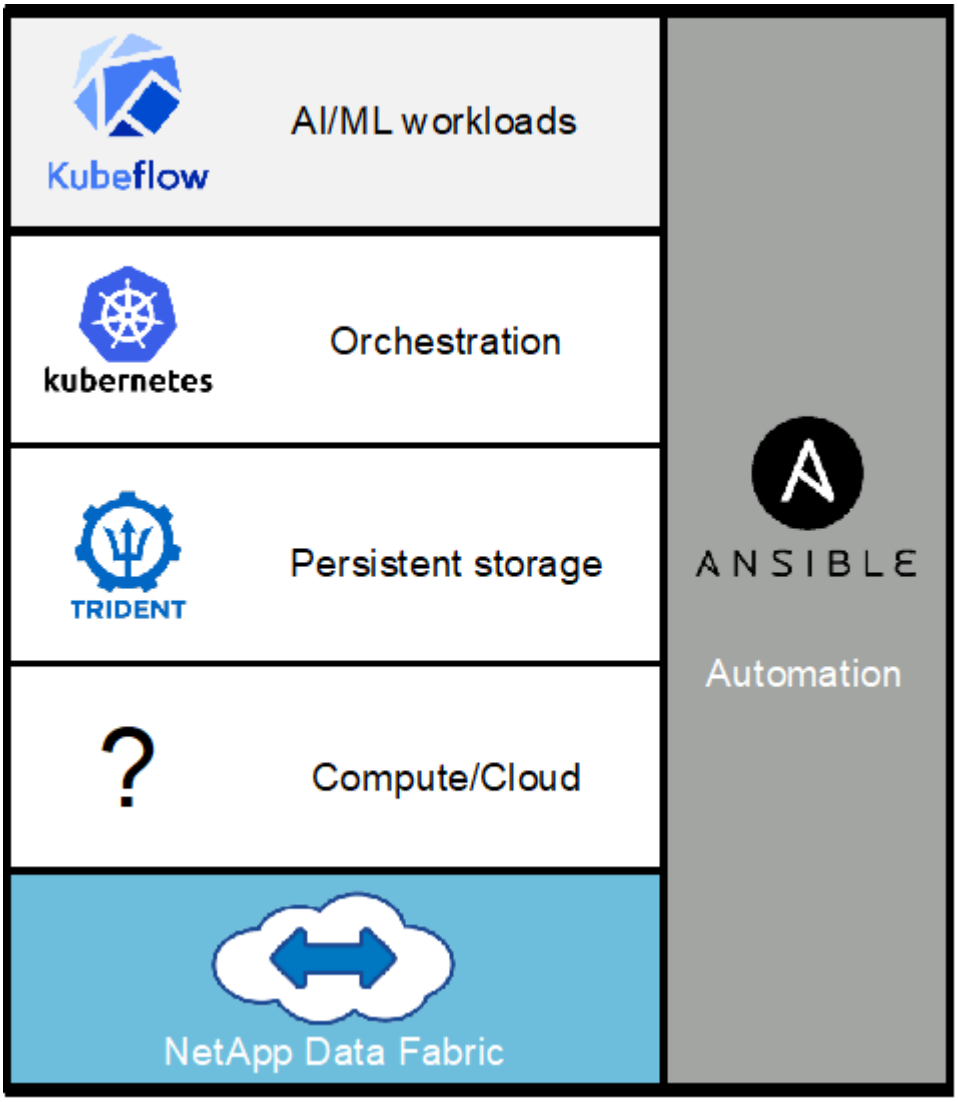
interagieren. Gleichzeitig können sie von den erweiterten Datenmanagement-Funktionen von NetApp und der Data Fabric Strategie von NetApp profitieren. Trident abstrahiert die Komplexität von persistentem Storage und vereinfacht die Nutzung. Weitere Informationen finden Sie auf der "Trident Website".

NVIDIA DeepOps

DeepOps ist ein Open-Source-Projekt von NVIDIA, das mithilfe von Ansible die Implementierung von GPU-Server-Clustern gemäß Best Practices automatisiert. DeepOps ist modular aufgebaut und kann für verschiedene Implementierungsaufgaben verwendet werden. DeepOps wird in diesem Dokument und der hier beschriebene Validierungsübung verwendet, um ein Kubernetes-Cluster zu implementieren, der aus GPU-Server-Worker-Nodes besteht. Weitere Informationen finden Sie auf der "DeepOps-Website".

Kubeflow

Kubeflow ist ein Open Source AI und ML Toolkit für Kubernetes und wurde ursprünglich von Google entwickelt. Das Kubeflow-Projekt macht Implementierungen von KI- und ML-Workflows auf Kubernetes einfach, tragbar und skalierbar. Kubeflow abstrahiert die Besonderheiten von Kubernetes und ermöglicht Data Scientists, sich auf das zu konzentrieren, was sie am besten kennen—Datenwissenschaft. Eine Visualisierung finden Sie in der folgenden Abbildung. Kubeflow hat seine Bedeutung gewonnen, da DIE IT-Abteilungen in Unternehmen zunehmend auf Kubernetes standardisiert haben. Weitere Informationen finden Sie auf der "Kubeflow-Website".



Kubeflow-Pipelines

Kubeflow Pipelines sind ein Schlüsselbestandteil von Kubeflow. Kubeflow Pipelines sind eine Plattform und ein Standard für die Definition und Implementierung portabler und skalierbarer KI- und ML-Workflows. Weitere Informationen finden Sie im "Offizielle Dokumentation von Kubeflow".

Jupyter Notebook Server

Ein Jupyter Notebook Server ist eine Open Source Web-Anwendung, mit der Data Scientists Wiki-ähnliche Dokumente erstellen können, genannt Jupyter Notebooks, die sowohl Live-Code als auch einen beschreibenden Test enthalten. Jupyter Notebooks werden in der KI- und ML-Community häufig eingesetzt, um KI- und ML-Projekte zu dokumentieren, zu speichern und gemeinsam zu nutzen. Kubeflow vereinfacht die Bereitstellung und Bereitstellung von Jupyter Notebook-Servern auf Kubernetes. Weitere Informationen zu Jupyter Notebooks finden Sie auf der "Jupyter-Website". Weitere Informationen zu Jupyter Notebooks im Kontext von Kubeflow finden Sie im "Offizielle Dokumentation von Kubeflow".

Apache Airflow

Apache Airflow ist eine Open-Source-Workflow-Managementplattform, die programmatisches Authoring, Scheduling und Monitoring für komplexe Unternehmens-Workflows ermöglicht. Sie wird häufig zur Automatisierung von ETL- und Daten-Pipeline-Workflows verwendet, beschränkt sich jedoch nicht auf diese Arten von Workflows. Das Airflow-Projekt wurde von Airbnb gestartet, ist aber inzwischen sehr populär in der Branche und fällt nun unter die Schirmherrschaft der Apache Software Foundation. Der Luftstrom wird in Python geschrieben, Airflow-Workflows werden über Python-Skripte erstellt und Airflow wird nach dem Prinzip „Configuration as Code“ entworfen. Viele Benutzer von Airflow setzen nun Airflow auf Kubernetes aus.

Gesteuerte Acyclic-Grafiken (DAGs)

In Airflow werden Workflows als gesteuerte Acyclic Grafs (DAGs) bezeichnet. DAGs bestehen aus Aufgaben, die je nach DAG-Definition nacheinander, parallel oder kombiniert ausgeführt werden. Der Airflow Scheduler führt individuelle Aufgaben auf einem Array von Mitarbeitern aus und erfüllt dabei die in der DAG-Definition festgelegten Abhängigkeiten auf Aufgabenebene. DAGs werden über Python Skripte definiert und erstellt.

NetApp ONTAP 9

NetApp ONTAP 9 ist die neueste Generation der Storage-Managementsoftware von NetApp. Mit ihr können Unternehmen wie Ihres ihre Infrastruktur modernisieren und auf ein Cloud-fähiges Datacenter umstellen. Dank der erstklassigen Datenmanagementfunktionen lassen sich mit ONTAP Ihre Daten mit einem einzigen Toolset managen und schützen, ganz gleich, wo sich diese Daten befinden. Zudem können Sie die Daten problemlos dorthin verschieben, wo Sie sie benötigen: Zwischen Edge, Core und Cloud. ONTAP 9 umfasst zahlreiche Funktionen, die das Datenmanagement vereinfachen, geschäftskritische Daten beschleunigen und schützen und die Infrastruktur über Hybrid-Cloud-Architekturen hinweg zukunftssicher machen.

Datenmanagement Vereinfachen

Für Ihren Enterprise-IT-Betrieb spielt Datenmanagement eine zentrale Rolle, da es die richtigen Ressourcen für Ihre Applikationen und Datensätze verwenden kann. ONTAP umfasst die folgenden Funktionen, um den Betrieb zu optimieren und zu vereinfachen und damit die Gesamtbetriebskosten zu senken:

- **Inline-Data-Compaction und erweiterte Deduplizierung.** Data-Compaction reduziert den ungenutzten Speicherplatz in Storage-Blöcken, während Deduplizierung die effektive Kapazität deutlich steigert.
- **Minimum, Maximum und anpassungsfähige Quality of Service (QoS)** durch granulare QoS-Kontrollen können geschäftskritische Applikationen auch in hochgemeinsam genutzten Umgebungen weiterhin Performance-Level aufrechterhalten.

- **ONTAP FabricPool.** Diese Funktion bietet automatisches Tiering von „kalten“ Daten in Private- und Public-Cloud-Storage-Optionen, einschließlich Amazon Web Services (AWS), Azure und objektbasiertem NetApp StorageGRID Storage.

Daten beschleunigen und sichern

ONTAP bietet überdurchschnittliche Performance und Datensicherung, erweitert durch folgende Funktionen:

- **Hohe Performance und niedrige Latenz.** ONTAP bietet den höchstmöglichen Durchsatz bei geringstmöglicher Latenz.
- **NetApp ONTAP FlexGroup-Technologie.** Ein FlexGroup Volume ist ein hochperformanter Daten-Container, der sich linear auf bis zu 20 PB und 400 Milliarden Dateien skalieren lässt und über einen Single Namespace das Datenmanagement vereinfacht.
- **Datensicherung.** ONTAP bietet integrierte Datensicherungsfunktionen mit einem einheitlichen Management über alle Plattformen hinweg.
- **NetApp Volume Encryption.** ONTAP bietet native Verschlüsselung auf Volume-Ebene mit integriertem und externem Verschlüsselungsmanagement.

Zukunftssichere Infrastruktur

ONTAP 9 hilft Unternehmen dabei, flexibel auf ständig wechselnde Geschäftsanforderungen zu reagieren:

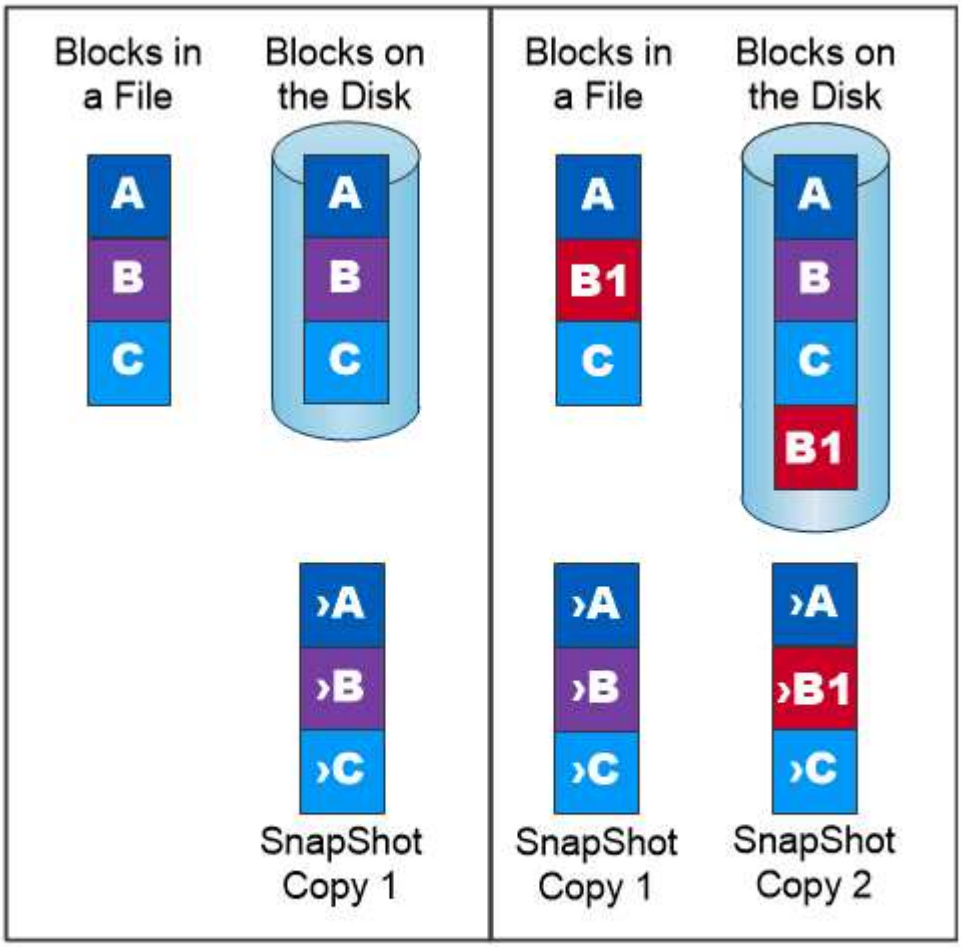
- **Nahtlose Skalierung und unterbrechungsfreier Betrieb.** ONTAP unterstützt das unterbrechungsfreie Hinzufügen von Kapazitäten zu bestehenden Controllern sowie das Scale-out von Clustern. Damit lassen sich Upgrades auf die neuesten Technologien wie NVMe und 32 GB FC ohne teure Datenmigrationen oder Ausfälle durchführen.
- **Cloud Connection.** ONTAP ist eine der Storage-Managementsoftware mit der umfassendsten Cloud-Integration und bietet Optionen für softwaredefinierten Storage (ONTAP Select) und Cloud-native Instanzen (NetApp Cloud Volumes Service) in allen Public Clouds.
- **Integration in moderne Applikationen.** durch den Einsatz derselben Infrastruktur, die bereits vorhandene Enterprise-Applikationen unterstützt, bietet ONTAP Datenservices der Enterprise-Klasse für Plattformen und Applikationen der neuesten Generation, wie OpenStack, Hadoop und MongoDB.

NetApp Snapshot Kopien

Eine NetApp Snapshot Kopie ist ein schreibgeschütztes, zeitpunktgenaues Image eines Volumes. Das Image verbraucht nur minimalen Speicherplatz und beeinträchtigt den Performance-Overhead, da nur Änderungen an Dateien aufgezeichnet werden, die seit der letzten Snapshot Kopie erstellt wurden, wie in der folgenden Abbildung dargestellt.

Snapshot Kopien sind der zentralen ONTAP Storage-Virtualisierungstechnologie, dem Write Anywhere File Layout (WAFL), verdanken sie ihre Effizienz. Wie eine Datenbank verwendet WAFL Metadaten, um auf die tatsächlichen Datenblöcke auf der Festplatte zu verweisen. Im Gegensatz zu einer Datenbank überschreiben WAFL jedoch keine vorhandenen Blöcke. Aktualisierte Daten werden in einen neuen Block geschrieben und die Metadaten geändert. Der Grund dafür ist, dass ONTAP bei der Erstellung einer Snapshot Kopie Metadaten referenziert, statt Datenblöcke zu kopieren. Somit sind die Snapshot Kopien so effizient. So entfallen die Suchzeit, die andere Systeme beim Auffinden der zu kopierenden Blöcke sowie die Kosten für die Erstellung der Kopie selbst tragen.

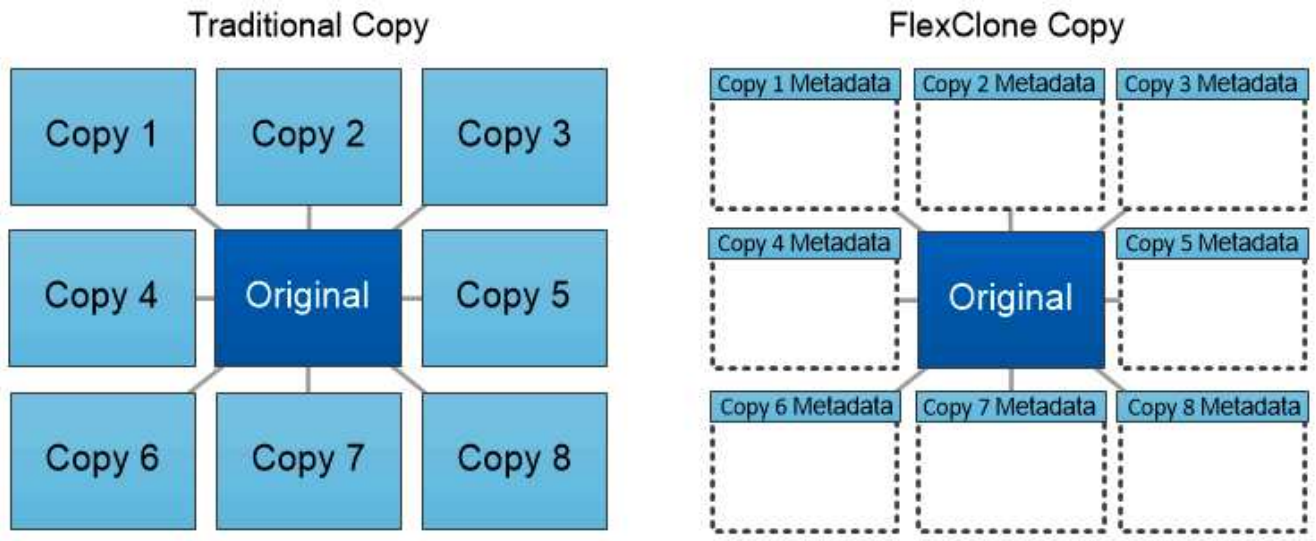
Sie können eine Snapshot Kopie verwenden, um einzelne Dateien oder LUNs wiederherzustellen oder den gesamten Inhalt eines Volume wiederherzustellen. ONTAP vergleicht Zeigerinformationen in der Snapshot-Kopie mit Daten auf der Festplatte, um das fehlende oder beschädigte Objekt ohne Ausfallzeiten und hohe Performance-Kosten zu rekonstruieren.



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

NetApp FlexClone Technologie

Die NetApp FlexClone Technologie referenziert Snapshot Metadaten, um beschreibbare, zeitpunktgenaue Kopien eines Volumes zu erstellen. Kopien verwenden Datenblöcke gemeinsam mit ihren Eltern und verbrauchen somit keinen Storage, außer was für Metadaten erforderlich ist, bis Änderungen in die Kopie geschrieben werden, wie in der folgenden Abbildung dargestellt. Bei der Erstellung herkömmlicher Kopien dauert die Erstellung von Minuten oder gar Stunden, mit FlexClone können Sie selbst die größten Datensätze nahezu sofort kopieren. Daher eignet sie sich besonders für Situationen, in denen mehrere Kopien identischer Datensätze (z. B. ein Entwicklungs-Workspace) oder temporäre Kopien eines Datensatzes benötigt werden (d. h. eine Applikation gegen einen Produktionsdatensatz testen).

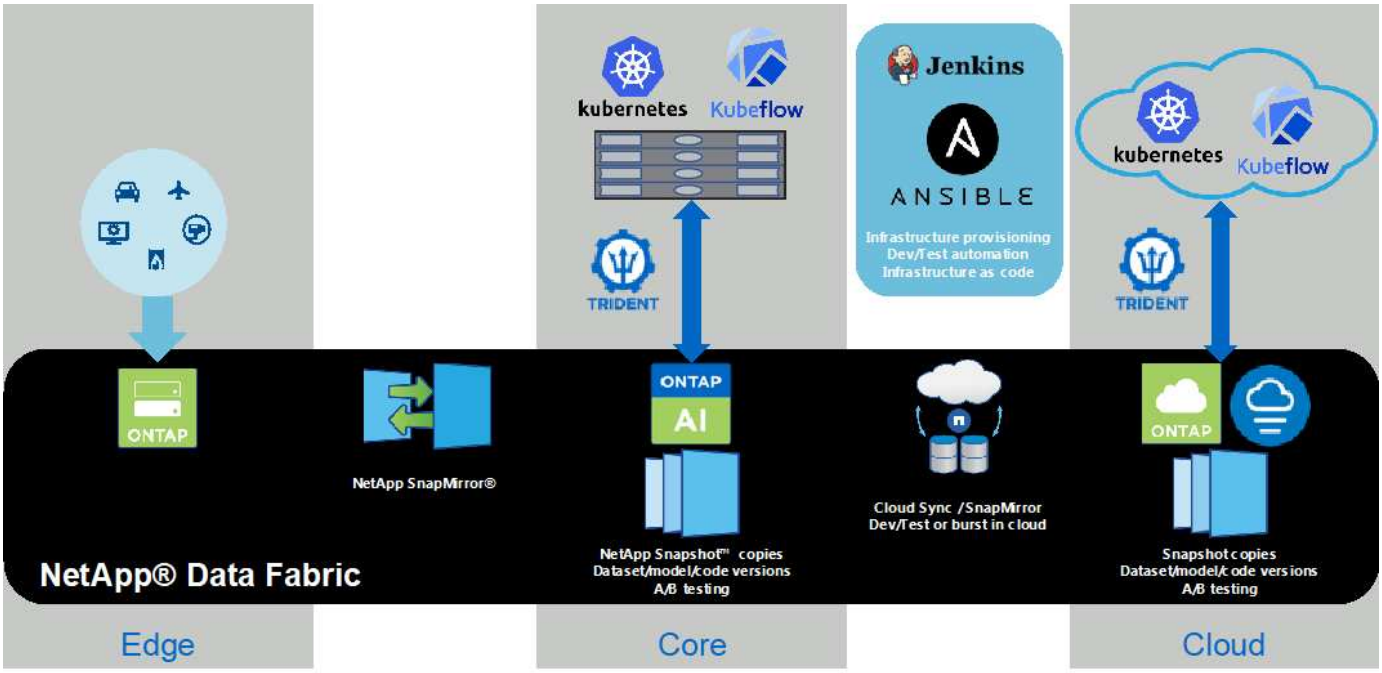


FlexClone copies share data blocks with their parents, consuming no storage except what is required for metadata.

NetApp SnapMirror Datenreplizierung

NetApp SnapMirror ist eine kostengünstige, benutzerfreundliche und einheitliche Replizierungslösung für die gesamte Data-Fabric-Strategie. Sie repliziert Daten mit hoher Geschwindigkeit über LAN oder WAN. Sie bietet hohe Datenverfügbarkeit und schnelle Datenreplizierung für alle Arten von Applikationen, einschließlich geschäftskritischer Applikationen in virtuellen und herkömmlichen Umgebungen. Durch das Replizieren und ständige Aktualisieren der sekundären Daten auf einem Storage-System von NetApp sind die Daten immer aktuell und verfügbar. Es sind keine externen Replizierungsserver erforderlich. In der folgenden Abbildung finden Sie ein Beispiel für eine Architektur, die die SnapMirror Technologie nutzt.

SnapMirror Software nutzt NetApp ONTAP Storage-Effizienzfunktionen, indem nur geänderte Datenblöcke im Netzwerk verschoben werden. Außerdem verwendet SnapMirror Software eine integrierte Netzwerkkomprimierung, um die Datenübertragung zu beschleunigen und die Auslastung der Netzwerkbandbreite um bis zu 70 % zu reduzieren. Mit der SnapMirror Technologie lässt sich ein Thin-Replication-Datenstrom erstellen, um ein einzelnes Repository zu erstellen, das sowohl den aktiven Spiegel als auch die zeitpunktgenau Kopien enthält. Auf diese Weise verringert sich der Datenverkehr im Netzwerk um bis zu 50 %.



NetApp BlueXP Kopie und Synchronisierung

BlueXP Copy and Sync ist ein NetApp Service für schnelle und sichere Datensynchronisierung. Unabhängig davon, ob Sie Dateien zwischen On-Premises-NFS- oder SMB-Dateifreigaben, NetApp StorageGRID, NetApp ONTAP S3, NetApp Cloud Volumes Service, Azure NetApp Files, AWS S3, AWS EFS, Azure Blob mit Google Cloud Storage oder IBM Cloud Object Storage verschiebt BlueXP Copy and Sync die Dateien schnell und sicher an den gewünschten Speicherort.

Nach der Übertragung stehen die Daten an der Quelle und am Ziel vollständig zur Verfügung. BlueXP Copy and Sync kann Daten nach Bedarf synchronisieren, wenn ein Update ausgelöst wird oder Daten kontinuierlich anhand eines vordefinierten Zeitplans synchronisiert werden. Trotzdem werden mit BlueXP Copy and Sync nur die Deltas verschoben, sodass Zeit und Kosten für die Datenreplizierung minimiert werden.

BlueXP Copy and Sync ist ein Software-as-a-Service-Tool (SaaS), das sich äußerst einfach einrichten und verwenden lässt. Datentransfers, die durch BlueXP Copy und Sync ausgelöst werden, erfolgen durch Datenmanager. Die Datenmanager von BlueXP Copy und Sync können in AWS, Azure, Google Cloud Platform oder lokal implementiert werden.

NetApp XCP

Der Client-basierte NetApp XCP Software ermöglicht Datenmigrationen zwischen beliebigen Systemen von NetApp und NetApp zu NetApp sowie Einblicke in das Filesystem. XCP ist für Skalierung ausgelegt und erreicht maximale Performance, indem alle verfügbaren Systemressourcen für umfangreiche Datensätze und hochperformante Migrationen genutzt werden. Mit XCP erhalten Sie eine vollständige Übersicht über das Dateisystem und können Berichte generieren.

NetApp XCP ist in einem einzigen Paket erhältlich, das NFS- und SMB-Protokolle unterstützt. XCP enthält eine Linux-Binärdatei für NFS-Datensätze und ein Windows Executable für SMB-Datensätze.

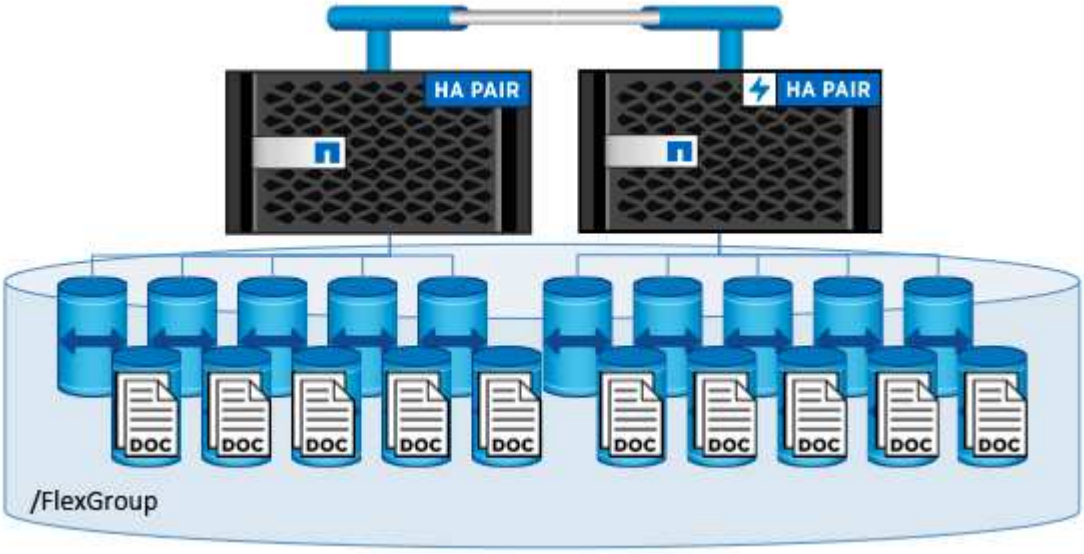
Die hostbasierte Software NetApp XCP File Analytics erkennt Dateifreigaben, führt Scans im Filesystem aus und bietet ein Dashboard für Dateianalysen. XCP File Analytics ist sowohl mit Systemen von NetApp als auch mit Systemen anderer Hersteller kompatibel und wird auf Linux- oder Windows-Hosts ausgeführt, um Analysen für NFS- und SMB-exportierte Filesysteme zu ermöglichen.

NetApp ONTAP FlexGroup Volumes

Ein Trainingsdatensatz kann eine Sammlung von möglicherweise Milliarden von Dateien sein. Dateien können Text, Audio, Video und andere Formen unstrukturierter Daten enthalten, die gespeichert und verarbeitet werden müssen, damit sie gleichzeitig gelesen werden können. Das Storage-System muss eine große Anzahl an kleinen Dateien speichern und diese parallel für sequenzielle und zufällige I/O lesen

Ein FlexGroup Volume ist ein einziger Namespace, der aus mehreren zusammengehörigen Member Volumes besteht, wie in der folgenden Abbildung dargestellt. Aus Sicht eines Storage-Administrators wird ein FlexGroup Volume wie ein NetApp FlexVol Volume gemanagt und verhält sich so wie ein NetApp Volume. Dateien in einem FlexGroup Volume werden Volumes einzelner Mitglieder zugewiesen und nicht über Volumes oder Nodes verteilt. Sie bieten folgende Möglichkeiten:

- FlexGroup Volumes bieten eine Kapazität im Petabyte-Bereich und eine planbare niedrige Latenz für Workloads mit vielen Metadaten.
- Sie unterstützen bis zu 400 Milliarden Dateien im selben Namespace
- Sowie parallelisierte Vorgänge bei NAS-Workloads über CPUs, Nodes, Aggregate und zusammengehörige FlexVol Volumes hinweg.



Hardware- und Software-Anforderungen

Die NetApp AI Control Plane Lösung ist nicht von dieser spezifischen Hardware abhängig. Die Lösung ist mit jeder physischen NetApp Storage Appliance, jeder softwaredefinierten Instanz oder jedem Cloud-Service kompatibel, der von Trident unterstützt wird. Hierzu zählen beispielsweise ein NetApp AFF Storage-System, Azure NetApp Files, NetApp Cloud Volumes Service, eine softwaredefinierte Storage-Instanz von NetApp ONTAP Select oder eine NetApp Cloud Volumes ONTAP Instanz. Außerdem kann die Lösung auf jedem beliebigen Kubernetes Cluster implementiert werden, solange die verwendete Kubernetes-Version von Kubeflow und NetApp Trident unterstützt wird. Eine Liste der von Kubeflow unterstützten Kubernetes-Versionen finden Sie im "Offizielle Dokumentation von Kubeflow". Eine Liste der von Trident unterstützten Kubernetes-Versionen finden Sie im "Trident Dokumentation". In den folgenden Tabellen finden Sie

Einzelheiten zur Umgebung, die zur Validierung der Lösung verwendet wurde.

Infrastrukturkomponente	Menge	Details	Betriebssystem
Bereitstellung Jump-Host	1	VM	Ubuntu 20.04.2 LTS
Kubernetes Master-Nodes	1	VM	Ubuntu 20.04.2 LTS
Kubernetes-Worker-Nodes	2	VM	Ubuntu 20.04.2 LTS
Kubernetes-GPU-Worker-Nodes	2	NVIDIA DGX-1 (Bare-Metal)	NVIDIA DGX OS 4.0.5 (basierend auf Ubuntu 18.04.2 LTS)
Storage	1 HA-Paar	NetApp AFF A220	NetApp ONTAP 9.7 P6

Software-Komponente	Version
Apache Airflow	2.0.1
Apache Airflow Helm Chart	8.0.8
Docker	19.03.12
Kubeflow	1.2
Kubernetes	1.18.9
NetApp Trident	21.01.2
NVIDIA DeepOps	Trident-Implementierungsfunktionen von der Master-Zweigstelle bei der Übertragung "61898cdfda"; Alle anderen Funktionen von Version 21.03

Unterstützung

NetApp bietet keine Enterprise-Unterstützung für Apache Airflow, Docker, Kubeflow, Kubernetes oder NVIDIA DeepOps. Wenn Sie an einer vollständig unterstützten Lösung mit Funktionen interessiert sind, die ähnlich der NetApp KI Control Plane Lösung sind, "Kontakt zu NetApp" Über vollständig unterstützte KI/ML-Lösungen, die NetApp gemeinsam mit Partnern anbietet.

Kubernetes Deployment

In diesem Abschnitt werden die Aufgaben beschrieben, die zur Implementierung eines Kubernetes-Clusters erforderlich sind, in dem die NetApp AI Control Plane Lösung implementiert werden kann. Wenn Sie bereits einen Kubernetes Cluster besitzen, können Sie diesen Abschnitt überspringen, solange Sie eine von Kubeflow und NetApp Trident unterstützte Version von Kubernetes ausführen. Eine Liste der von Kubeflow unterstützten Kubernetes-Versionen finden Sie im "Offizielle Dokumentation von Kubeflow". Eine Liste der von Trident unterstützten Kubernetes-Versionen finden Sie im "Trident Dokumentation".

Für lokale Kubernetes-Implementierungen, die Bare-Metal-Nodes mit NVIDIA-GPU(s) einbinden, empfiehlt NetApp die Verwendung des NVIDIA DeepOps Kubernetes-Implementierungstools. In diesem Abschnitt wird

die Bereitstellung eines Kubernetes-Clusters mit DeepOps beschrieben.

Voraussetzungen

Bevor Sie die in diesem Abschnitt beschriebenen Bereitstellungsaufgaben ausführen, gehen wir davon aus, dass Sie bereits die folgenden Aufgaben ausgeführt haben:

1. Sie haben bereits alle Bare-Metal-Kubernetes-Nodes (z. B. ein NVIDIA DGX-System, das Teil eines ONTAP AI Pods ist) gemäß den Standardkonfigurationsanweisungen konfiguriert.
2. Sie haben ein unterstütztes Betriebssystem auf allen Kubernetes Master- und Worker-Nodes und auf einem Deployment Jump-Host installiert. Eine Liste der von DeepOps unterstützten Betriebssysteme finden Sie im "DeepOps GitHub-Website".

Verwenden Sie NVIDIA DeepOps zum Installieren und Konfigurieren von Kubernetes

Um Ihren Kubernetes-Cluster mit NVIDIA DeepOps zu implementieren und zu konfigurieren, führen Sie die folgenden Aufgaben über einen Bereitstellungs-Jump-Host aus:

1. Laden Sie NVIDIA DeepOps herunter, indem Sie den Anweisungen auf der folgen "Erste Schritte" Auf der NVIDIA DeepOps GitHub Website.
2. Implementieren Sie Kubernetes in Ihrem Cluster, indem Sie die Anweisungen auf dem befolgen "Seite „Kubernetes Deployment Guide“" Auf der NVIDIA DeepOps GitHub Website.

Implementierung und Konfiguration von NetApp Trident

Implementierung und Konfiguration von NetApp Trident

In diesem Abschnitt werden die Aufgaben beschrieben, die Sie abschließen müssen, um NetApp Trident in Ihrem Kubernetes Cluster zu installieren und zu konfigurieren.

Voraussetzungen

Bevor Sie die in diesem Abschnitt beschriebenen Bereitstellungsaufgaben ausführen, gehen wir davon aus, dass Sie bereits die folgenden Aufgaben ausgeführt haben:

1. Sie verfügen bereits über ein funktionierendes Kubernetes Cluster und führen eine von Trident unterstützte Version von Kubernetes aus. Eine Liste der unterstützten Versionen finden Sie im "Trident Dokumentation".
2. Sie verfügen bereits über eine funktionierende NetApp Storage Appliance, softwaredefinierte Instanz oder Cloud-Storage-Service, die von Trident unterstützt wird.

Installation Von Trident

Um NetApp Trident in Ihrem Kubernetes-Cluster zu installieren und zu konfigurieren, führen Sie die folgenden Aufgaben über den Jump-Host der Implementierung aus:

1. Trident lässt sich mit einer der folgenden Methoden implementieren:
    - Wenn Sie zur Implementierung Ihres Kubernetes-Clusters NVIDIA DeepOps verwendet haben, können Sie Trident auch mit NVIDIA DeepOps in Ihrem Kubernetes-Cluster implementieren. Folgen Sie zur Implementierung von Trident mit DeepOps dem "Trident Implementierungsanleitungen" Auf der NVIDIA DeepOps GitHub Website.
    - Wenn Sie NVIDIA DeepOps nicht zur Implementierung Ihres Kubernetes-Clusters verwendet haben oder Trident einfach manuell implementieren möchten, können Sie Trident anhand der implementieren

"Implementierungsanleitungen" In der Trident Dokumentation. Erstellen Sie mindestens ein Trident Back-End und mindestens eine Kubernetes StorageClass. Sie erhalten weitere Informationen zur Konfiguration "Back-Ends" Und "StorageClasses" Siehe verlinkte Unterabschnitte bei NetApp Docs.

> Wenn Sie die NetApp AI Control Plane Lösung auf einem ONTAP AI POD implementieren, lesen Sie "Beispiel Trident Back-Ends für ONTAP KI-Implementierungen" Beispiele für verschiedene Trident Back-Ends, die Sie möglicherweise erstellen möchten, und "Beispiel: Kubernetes Storageclasses für ONTAP AI Implementierungen" Für einige Beispiele für verschiedene Kubernetes StorageClasses, die Sie vielleicht erstellen möchten.

Beispiel Trident Back-Ends für ONTAP KI-Implementierungen

Bevor Sie Trident zur dynamischen Bereitstellung von Storage-Ressourcen innerhalb Ihres Kubernetes-Clusters verwenden können, müssen Sie ein oder mehrere Trident Back-Ends erstellen. Die folgenden Beispiele stellen unterschiedliche Arten von Backend dar, die erstellt werden sollten, wenn Sie die NetApp AI Control Plane Lösung auf einem ONTAP AI POD implementieren. Weitere Informationen zu Backends finden Sie im "Trident Dokumentation".

1. NetApp empfiehlt die Erstellung eines FlexGroup-fähigen Trident-Back-End für jede Daten-LIF (logische Netzwerkschnittstelle mit Datenzugriff), die Sie auf Ihrem NetApp AFF System verwenden möchten. So können Sie einen Ausgleich von Volume-Mounts über LIFs hinweg erzielen

   Die folgenden Beispielbefehle zeigen die Erstellung von zwei FlexGroup-fähigen Trident Back-Ends für zwei unterschiedliche Daten-LIFs, die mit der gleichen ONTAP Storage Virtual Machine (SVM) verbunden sind. Diese Backends verwenden den `ontap-nas-flexgroup` Storage-Treiber: ONTAP unterstützt zwei wesentliche Daten-Volume-Typen: FlexVol und FlexGroup. Die Größe von FlexVol-Volumes ist begrenzt (ab diesem Schreibvorgang hängt die maximale Größe von der spezifischen Implementierung ab). FlexGroup Volumes hingegen lassen sich linear auf bis zu 20 PB und 400 Milliarden Dateien skalieren und sorgen in einem Single Namespace für eine erhebliche Vereinfachung des Datenmanagements. Daher sind FlexGroup-Volumes optimal für AI- und ML-Workloads, die auf große Datenmengen basieren.

   Wenn Sie mit einer geringen Menge an Daten arbeiten und statt FlexGroup Volumes FlexVol Volumes verwenden möchten, können Sie Trident Back-Ends erstellen, die den verwenden `ontap-nas` Storage-Treiber statt des `ontap-nas-flexgroup` Storage-Treiber:

```
$ 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 empfiehlt außerdem, ein oder mehrere FlexVol-fähige Trident Back-Ends zu erstellen. Wenn Sie FlexGroup Volumes für das Training von Datensatz-Storage verwenden, könnten Sie FlexVol Volumes für die Speicherung von Ergebnissen, Ausgabe, Debug-Informationen usw. verwenden. Falls Sie FlexVol Volumes verwenden möchten, müssen Sie ein oder mehrere FlexVol-aktivierte Trident-Backends erstellen. Die folgenden Beispielbefehle zeigen die Erstellung eines einzelnen FlexVol-fähigen Trident Back-End, das eine einzelne Daten-LIF verwendet.

```
$ 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 |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
```

Beispiel: Kubernetes StorageClasses für ONTAP KI-Implementierungen

Bevor Trident zur dynamischen Provisionierung von Storage-Ressourcen innerhalb Ihres Kubernetes-Clusters verwendet werden kann, müssen Sie einen oder mehrere Kubernetes StorageClasses erstellen. Die folgenden Beispiele stellen verschiedene Arten

von StorageClasses dar, die Sie möglicherweise erstellen möchten, wenn Sie die NetApp AI Control Plane Lösung auf einem ONTAP AI POD implementieren. Weitere Informationen zu StorageClasses finden Sie im "Trident Dokumentation".

1. NetApp empfiehlt, für jedes FlexGroup-fähige Trident Back-End, das Sie im Abschnitt erstellt haben, eine separate StorageClass zu erstellen "Beispiel Trident Back-Ends für ONTAP KI-Implementierungen", Schritt 1. Mit diesen granularen StorageClasses können Sie NFS-Mounts hinzufügen, die bestimmten LIFs entsprechen (die LIFs, die Sie beim Erstellen der Trident Back-Ends angegeben haben), als ein bestimmtes Back-End, das in der StorageClass Spec-Datei angegeben ist. Die folgenden Beispielbefehle zeigen die Erstellung von zwei StorageClasses, die den beiden Beispiel Backends entsprechen, die im Abschnitt erstellt wurden "Beispiel Trident Back-Ends für ONTAP KI-Implementierungen", Schritt 1. Weitere Informationen zu StorageClasses finden Sie im "Trident Dokumentation".

   Damit ein anhaltendes Volume nicht gelöscht wird, wenn das entsprechende PersistenzVolumeClaim (PVC) gelöscht wird, verwendet das folgende Beispiel ein `reclaimPolicy` Der Wert von `Retain`. Weitere Informationen zum `reclaimPolicy` Feld, siehe den offiziellen "Kubernetes-Dokumentation".

```
$ 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 empfiehlt außerdem die Erstellung einer StorageClass, die dem FlexVol-fähigen Trident Back-End entspricht, das Sie im Abschnitt erstellt haben "Beispiel Trident Back-Ends für ONTAP KI-Implementierungen", Schritt 2. Die folgenden Beispielbefehle zeigen die Erstellung einer einzelnen StorageClass für FlexVol Volumes.

   Im folgenden Beispiel wird in der StorageClass-Definitionsdatei kein bestimmtes Back-End angegeben, da nur ein FlexVol-fähiges Trident-Backend erstellt wurde. Wenn Sie Volumes mit Kubernetes verwalten, die diese StorageClass verwenden, versucht Trident, jedes verfügbare Backend zu verwenden, das den verwendet `ontap-nas` Treiber.

```
$ 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 empfiehlt außerdem, eine generische StorageClass für FlexGroup Volumes zu erstellen. Die folgenden Beispielbefehle zeigen die Erstellung eines einzelnen allgemeinen StorageClass für FlexGroup Volumes.

   Beachten Sie, dass in der StorageClass-Definitionsdatei kein bestimmtes Backend angegeben ist. Wenn Sie daher Kubernetes für die Administration von Volumes verwenden, die diese StorageClass verwenden, versucht Trident, jedes verfügbare Back-End zu verwenden, das die verwendet `ontap-nas-flexgroup` Treiber.

```
$ 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
```

Kubeflow Deployment

In diesem Abschnitt werden die Aufgaben beschrieben, die Sie zur Bereitstellung von Kubeflow in Ihrem Kubernetes-Cluster abschließen müssen.

Voraussetzungen

Bevor Sie die in diesem Abschnitt beschriebenen Bereitstellungsaufgaben ausführen, gehen wir davon aus, dass Sie bereits die folgenden Aufgaben ausgeführt haben:

1. Sie haben bereits einen funktionierenden Kubernetes-Cluster und führen eine Version von Kubernetes aus, die von Kubeflow unterstützt wird. Eine Liste der unterstützten Versionen finden Sie im "Offizielle Dokumentation von Kubeflow".
2. Sie haben NetApp Trident bereits in Ihrem Kubernetes Cluster installiert und konfiguriert, wie in beschrieben "Implementierung und Konfiguration von Trident".

Standard-Kubernetes StorageClass festlegen

Bevor Sie Kubeflow implementieren, müssen Sie eine Standard-StorageClass in Ihrem Kubernetes Cluster zuweisen. Der Kubeflow Implementierungsprozess versucht, neue persistente Volumes mit der Standard-StorageClass bereitzustellen. Wenn keine StorageClass als Standard-StorageClass festgelegt ist, schlägt die Bereitstellung fehl. Um eine Standard-StorageClass innerhalb des Clusters festzulegen, führen Sie die folgende Aufgabe vom Sprunghost für die Bereitstellung aus. Wenn Sie bereits eine Standard-StorageClass innerhalb Ihres Clusters festgelegt haben, können Sie diesen Schritt überspringen.

1. Weisen Sie einen Ihrer vorhandenen StorageClasses als Standard-StorageClass zu. Die folgenden Beispielbefehle zeigen die Bezeichnung einer StorageClass mit dem Namen an `ontap-ai- flexvols-retain` Als Standard-StorageClass.

Der `ontap-nas-flexgroup` Der Trident Back-End-Typ hat eine ziemlich große PVC-Mindestgröße. Standardmäßig versucht Kubeflow, PVCs bereitzustellen, die nur wenige GB groß sind. Daher sollten Sie keine StorageClass angeben, die den verwendet `ontap-nas-flexgroup` Back-End-Typ als Standard StorageClass für die Zwecke der Kubeflow-Bereitstellung.

```
$ 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
```

Verwenden Sie NVIDIA DeepOps zur Implementierung von Kubeflow

NetApp empfiehlt, das von NVIDIA DeepOps zur Verfügung gestellte Kubeflow Implementierungs-Tool zu verwenden. Um Kubeflow in Ihren Kubernetes-Cluster mit dem DeepOps-Implementierungstool zu implementieren, führen Sie die folgenden Aufgaben auf dem Sprunghost der Bereitstellung aus.

Alternativ können Sie Kubeflow manuell bereitstellen, indem Sie dem folgen "Installationsanweisungen" In der offiziellen Kubeflow Dokumentation

1. Implementieren Sie Kubeflow in Ihrem Cluster, indem Sie den folgen "Anweisungen zur Bereitstellung von Kubeflow" Auf der NVIDIA DeepOps GitHub Website.
2. Beachten Sie unten die URL des Kubeflow Dashboard, die vom DeepOps Kubeflow Bereitstellungs-Tool ausgegeben wird.

```
$ ./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. Vergewissern Sie sich, dass alle im Kubeflow Namespace bereitgestellten Pods A zeigen `STATUS` Von `Running` Und vergewissern Sie sich, dass sich keine im Namespace bereitgestellten Komponenten in einem Fehlerzustand befinden. Es kann mehrere Minuten dauern, bis alle Pods beginnen.

```
$ 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
```

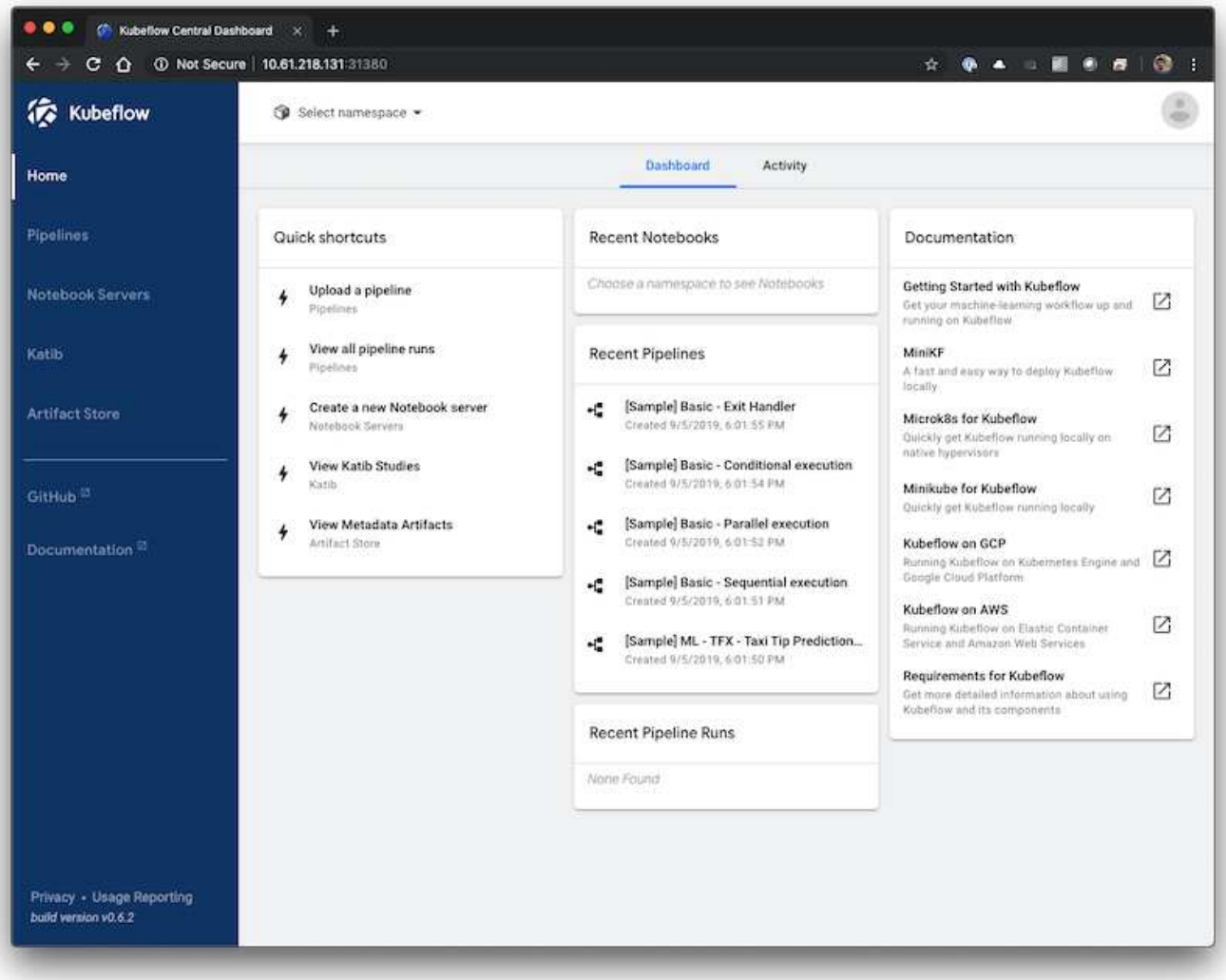
```
deployment.apps/katib-suggestion-nasrl                     1/1     1
1           95s
deployment.apps/katib-suggestion-random                    1/1     1
1           95s
deployment.apps/katib-ui                                   1/1     1
1           96s
deployment.apps/metadata-db                                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. Rufen Sie in Ihrem Webbrowser das zentrale Kubeflow Dashboard auf, indem Sie zur URL navigieren, die Sie in Schritt 2 angegeben haben.

   Der Standardbenutzername lautet `admin@kubeflow.org`, Und das Standardpasswort lautet `12341234`. Um weitere Benutzer zu erstellen, befolgen Sie die Anweisungen im "Offizielle Dokumentation von Kubeflow".



Beispiel Kubeflow Operations und Tasks

Dieser Abschnitt enthält Beispiele für verschiedene Vorgänge und Aufgaben, die Sie mit Kubeflow ausführen möchten.

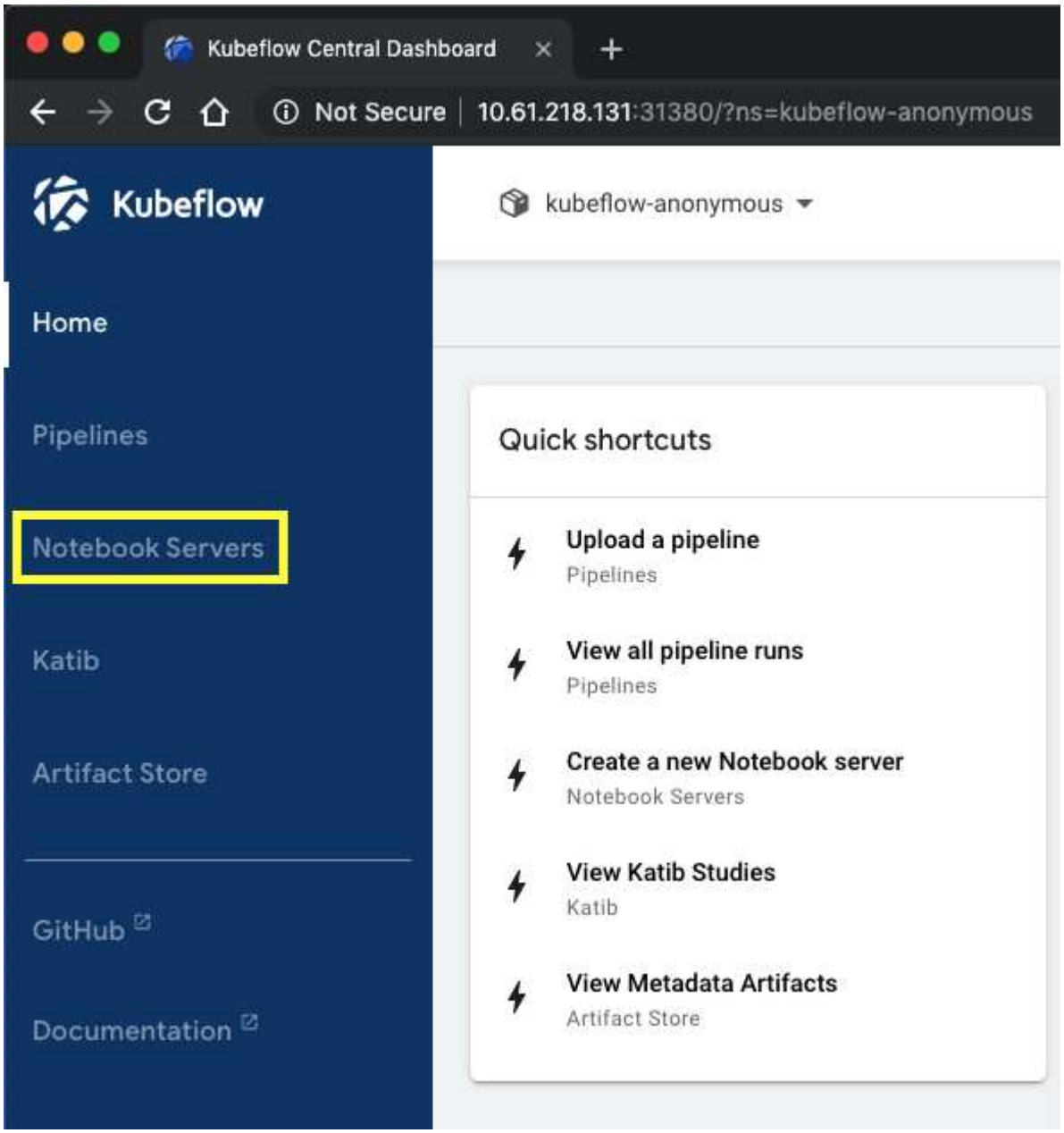
Beispiel Kubeflow Operations und Tasks

Dieser Abschnitt enthält Beispiele für verschiedene Vorgänge und Aufgaben, die Sie mit Kubeflow ausführen möchten.

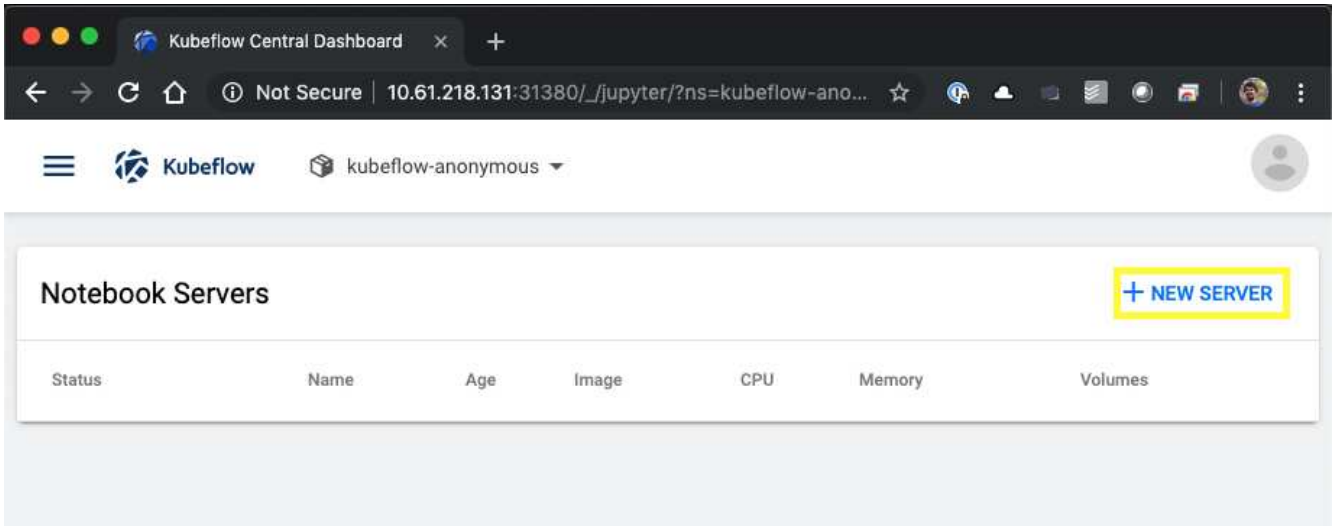
Bereitstellen eines Jupyter Notebook Workspace für Data Scientist oder Entwickler

Kubeflow ist in der Lage, neue Jupyter Notebook-Server schnell als Data Scientist-Workspaces bereitzustellen. Um einen neuen Jupyter Notebook-Server mit Kubeflow bereitzustellen, führen Sie die folgenden Aufgaben aus. Weitere Informationen zu Jupyter Notebooks im Kubeflow-Kontext finden Sie im "Offizielle Dokumentation von Kubeflow".

1. Klicken Sie im zentralen Kubeflow Dashboard im Hauptmenü auf Notebook-Server, um zur Jupyter Notebook-Serververwaltungsseite zu navigieren.

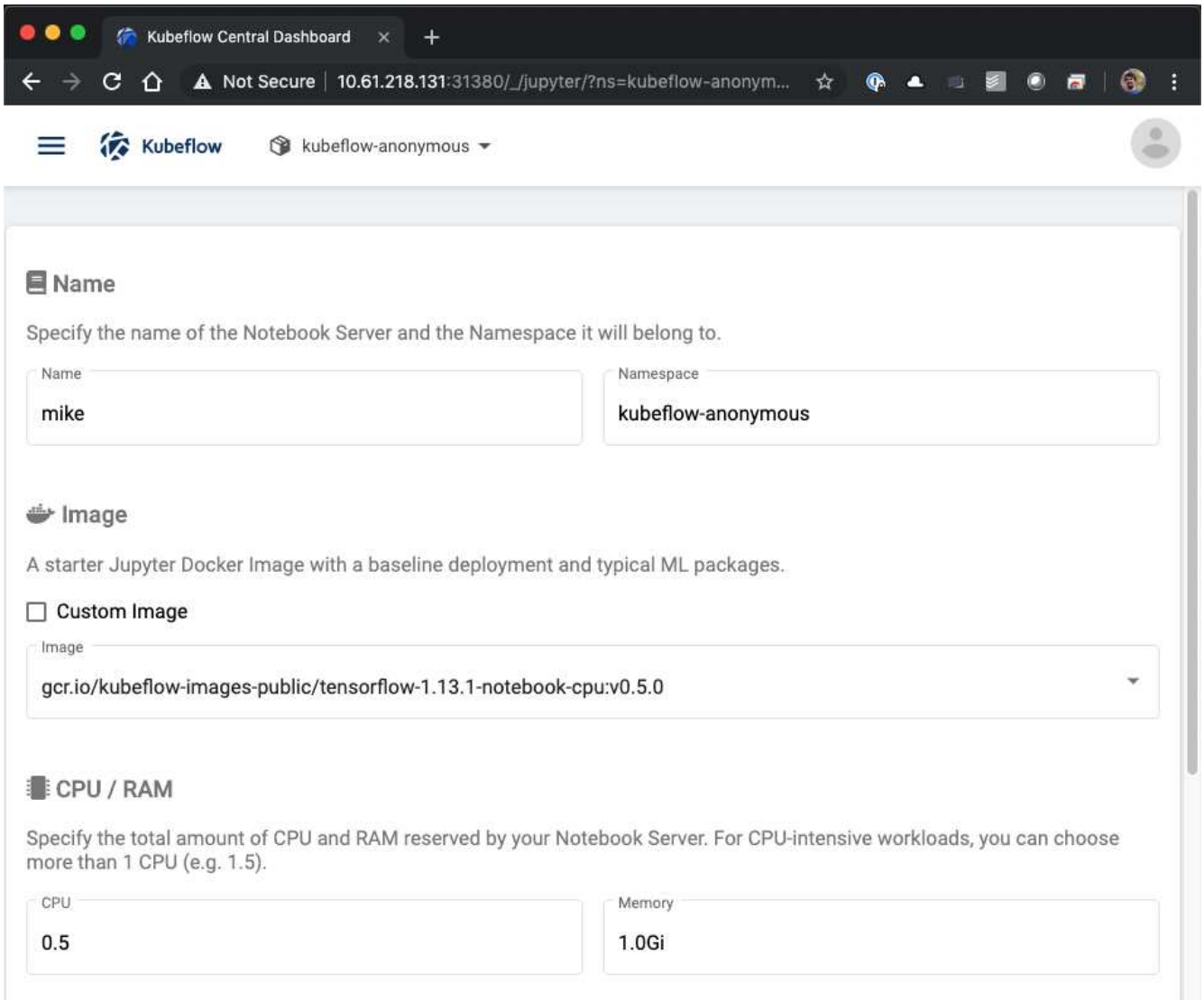


2. Klicken Sie auf Neuer Server, um einen neuen Jupyter Notebook-Server bereitzustellen.

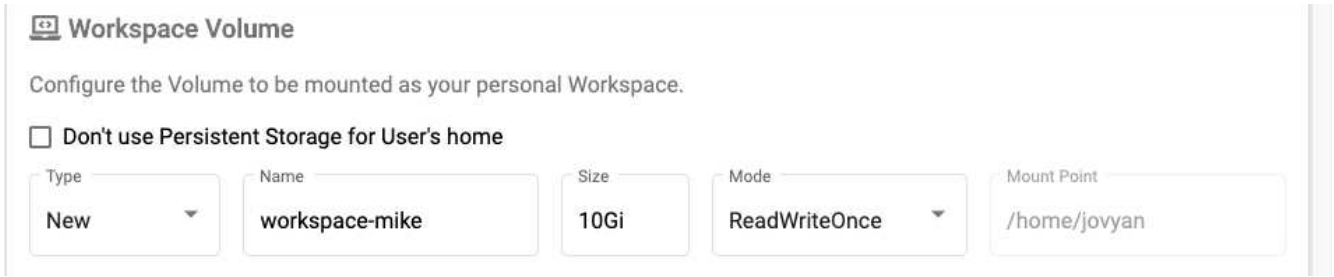


3. Geben Sie Ihrem neuen Server einen Namen, wählen Sie das Docker-Image aus, auf dem Ihr Server basieren soll, und geben Sie die CPU- und RAM-Größe an, die von Ihrem Server reserviert werden soll. Wenn das Namespace-Feld leer ist, wählen Sie über das Menü Namespace auswählen in der Seitenüberschrift einen Namespace aus. Das Namespace-Feld wird dann automatisch mit dem ausgewählten Namespace gefüllt.

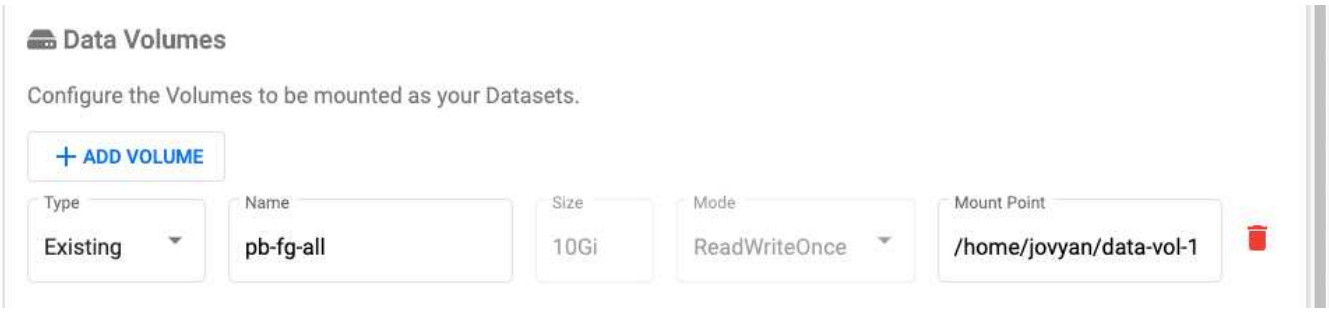
   Im folgenden Beispiel wird der verwendet `kubeflow-anonymous` Namespace ausgewählt. Darüber hinaus werden die Standardwerte für das Docker Image, die CPU und den RAM akzeptiert.



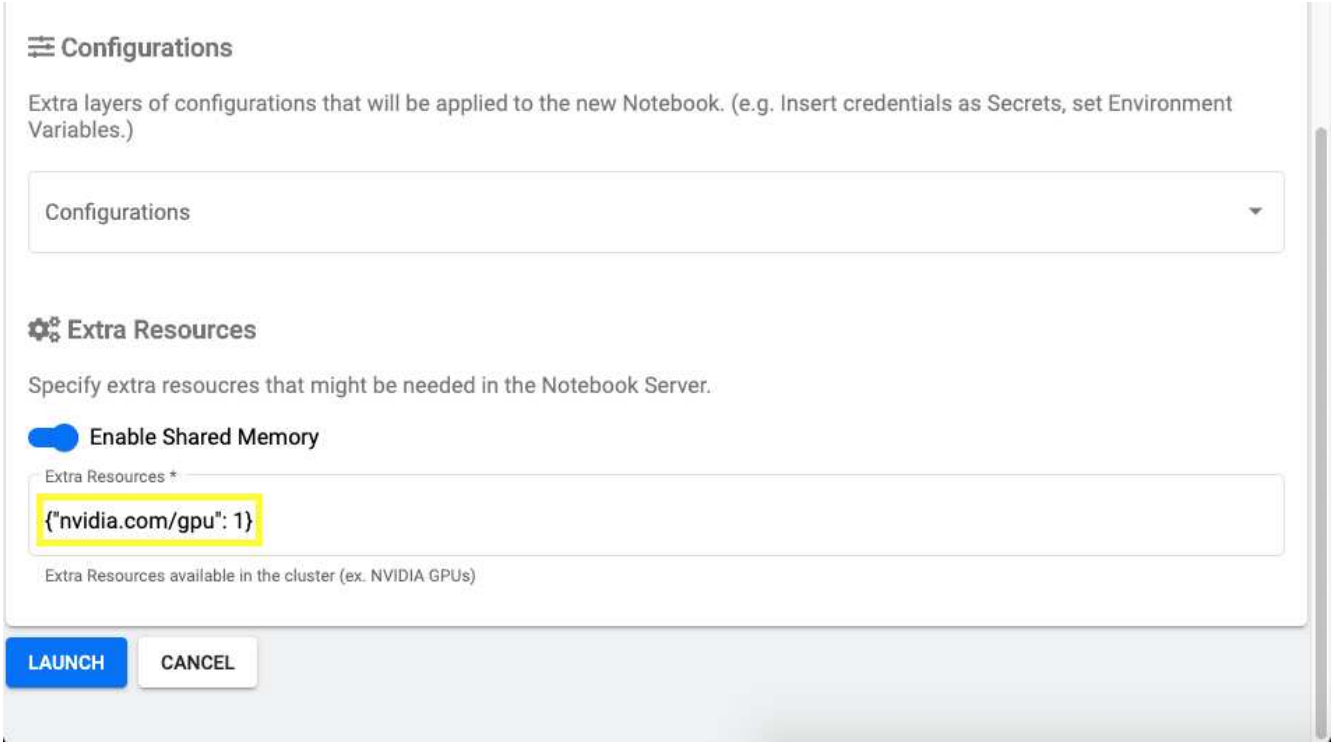
4. Geben Sie die Einzelheiten zum Workspace-Volumen an. Wenn Sie ein neues Volume erstellen möchten, wird dieses Volume oder PVC mithilfe der Standard-StorageClass bereitgestellt. Da ein StorageClass mit Trident als Standard-StorageClass in diesem Abschnitt festgelegt wurde "Kubeflow Deployment", Das Volume oder die PVC wird mit Trident bereitgestellt. Dieses Volume wird automatisch als Standardarbeitsbereich im Jupyter Notebook Server-Container bereitgestellt. Alle Notebooks, die ein Benutzer auf dem Server erstellt, die nicht in einem separaten Datenvolumen gespeichert werden, werden automatisch in diesem Workspace-Volume gespeichert. Daher bleiben die Notebooks auch bei Neustarts erhalten.



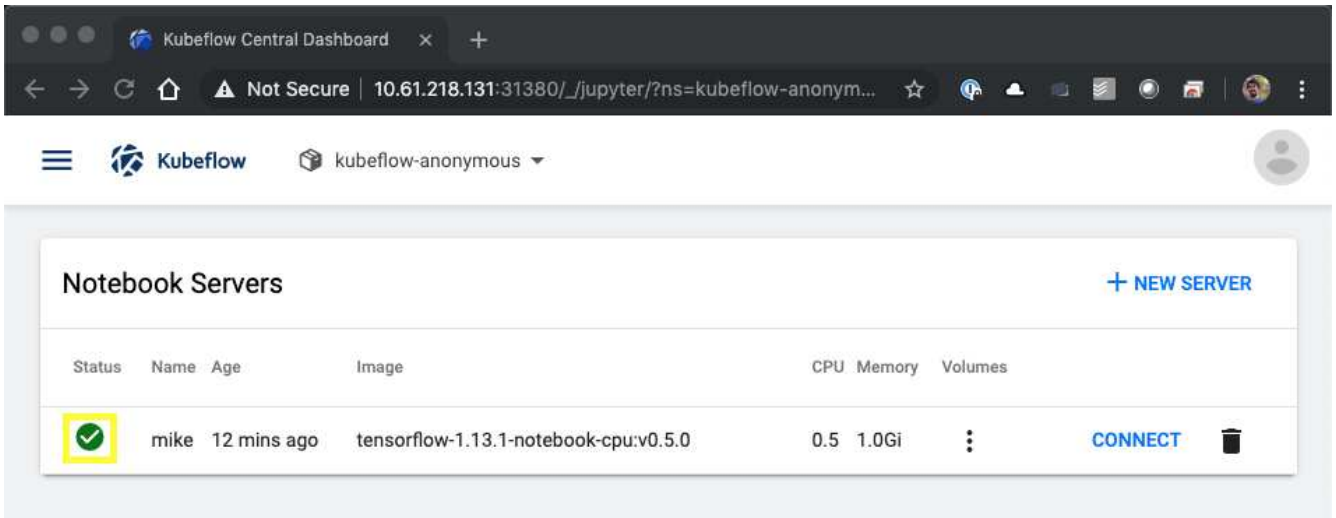
5. Hinzufügen von Daten-Volumes Im folgenden Beispiel wird ein vorhandenes PVC mit dem Namen „pb-fg-all“ angegeben und der Standard-Bereitstellungspunkt akzeptiert.



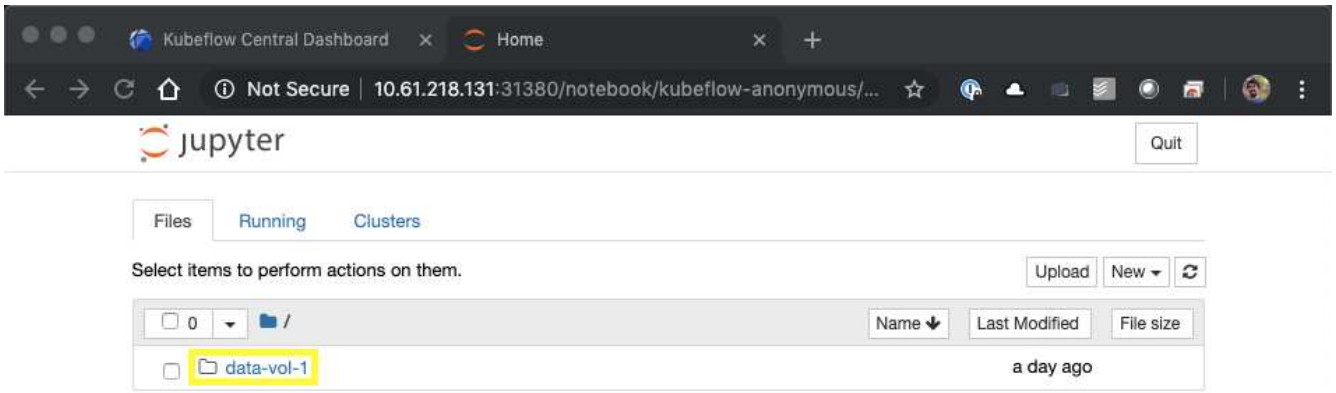
6. **Optional:** Anfrage, dass die gewünschte Anzahl von GPUs Ihrem Notebook-Server zugewiesen wird. Im folgenden Beispiel wird eine GPU angefordert.

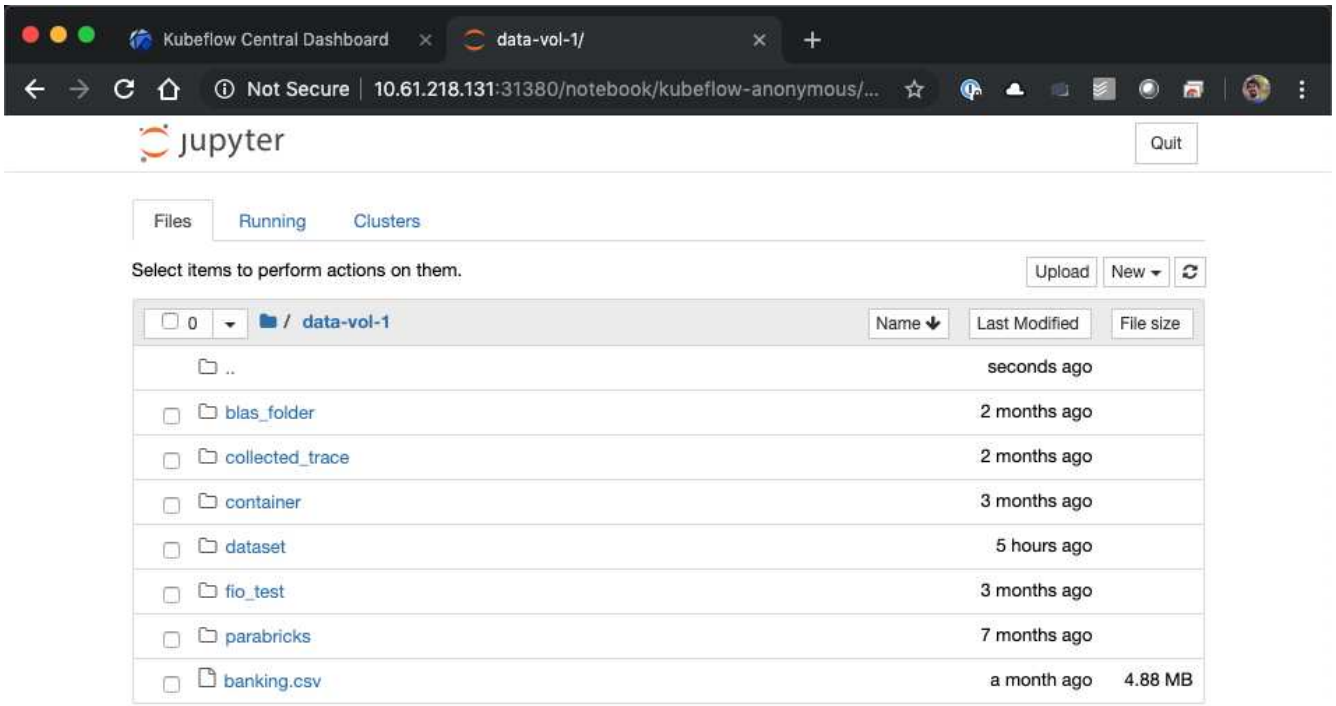


7. Klicken Sie auf Starten, um Ihren neuen Notebook-Server bereitzustellen.
8. Warten Sie, bis Ihr Notebook-Server vollständig bereitgestellt ist. Dies kann einige Minuten dauern, wenn Sie noch keinen Server mit dem von Ihnen angegebenen Docker-Image bereitgestellt haben, da das Image heruntergeladen werden muss. Wenn Ihr Server vollständig bereitgestellt wurde, wird in der Spalte Status auf der Seite Jupyter Notebook Server Administration ein grünes Häkchen angezeigt.



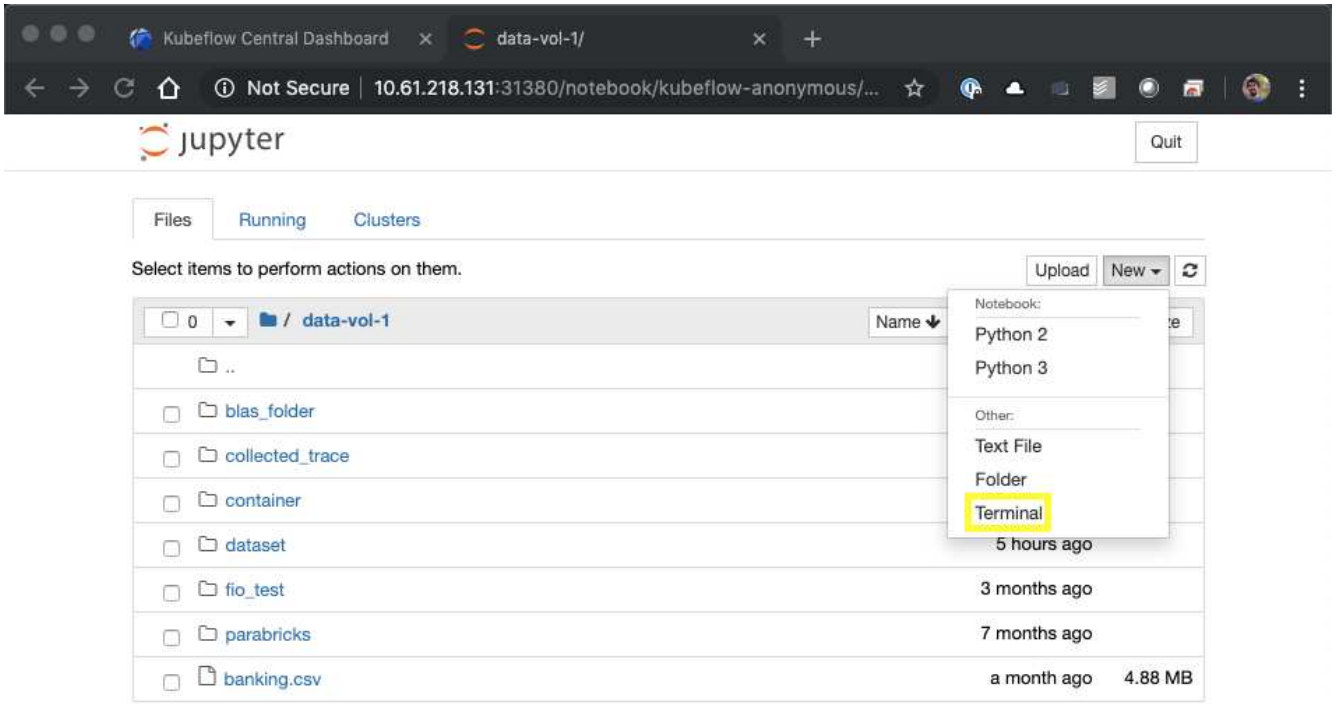
9. Klicken Sie auf Verbinden, um eine Verbindung mit der neuen Server-Webschnittstelle herzustellen.
10. Vergewissern Sie sich, dass das in Schritt 6 angegebene Datensatz-Volume auf dem Server installiert ist. Beachten Sie, dass dieses Volume standardmäßig im Standardarbeitsbereich bereitgestellt wird. Aus der Perspektive des Benutzers ist dies nur ein weiterer Ordner im Arbeitsbereich. Der Anwender, der wahrscheinlich ein Data Scientist ist und nicht ein Infrastruktur-Experte ist, muss nicht über Storage-Fachkenntnisse verfügen, um dieses Volumen zu nutzen.

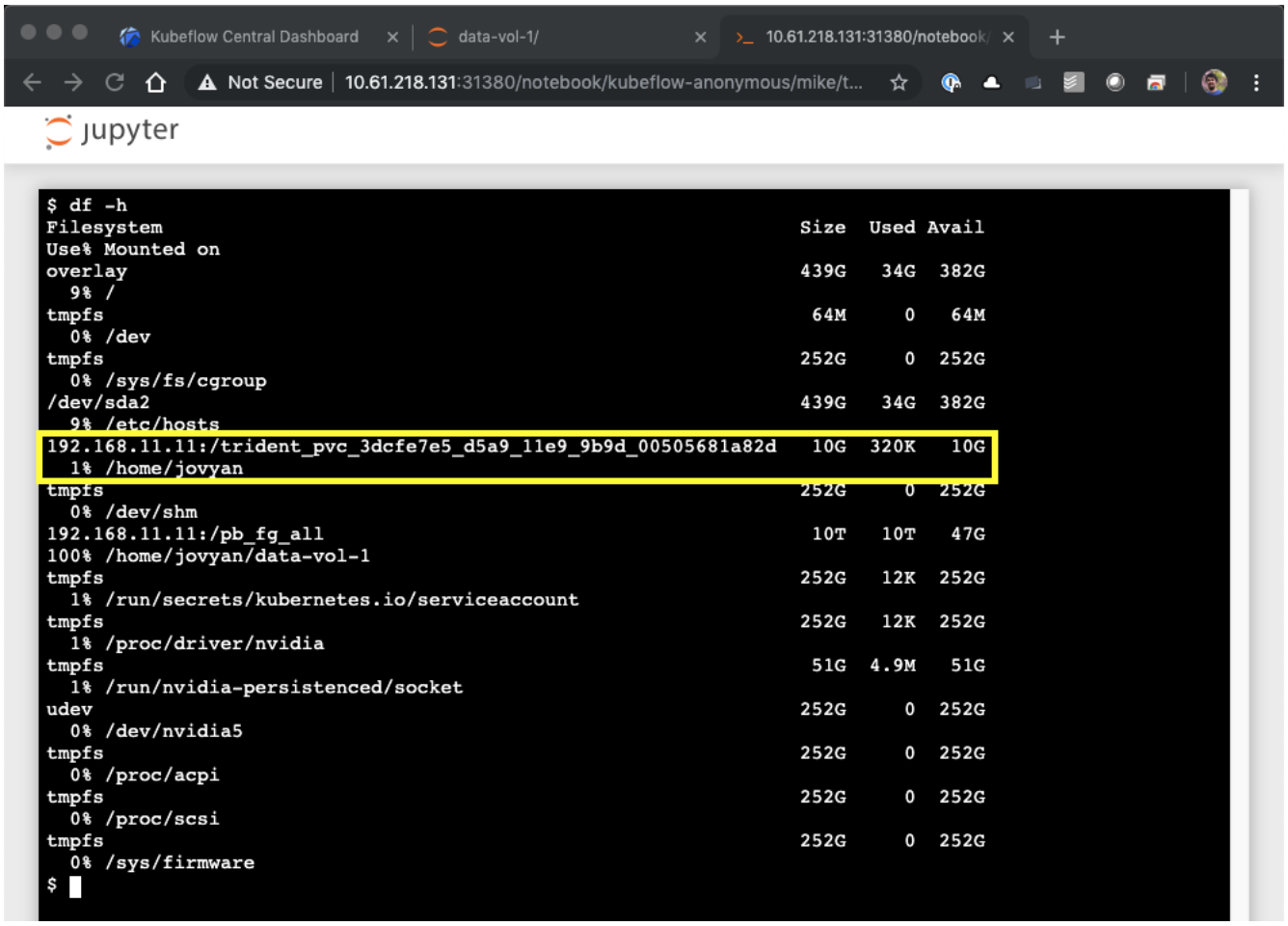




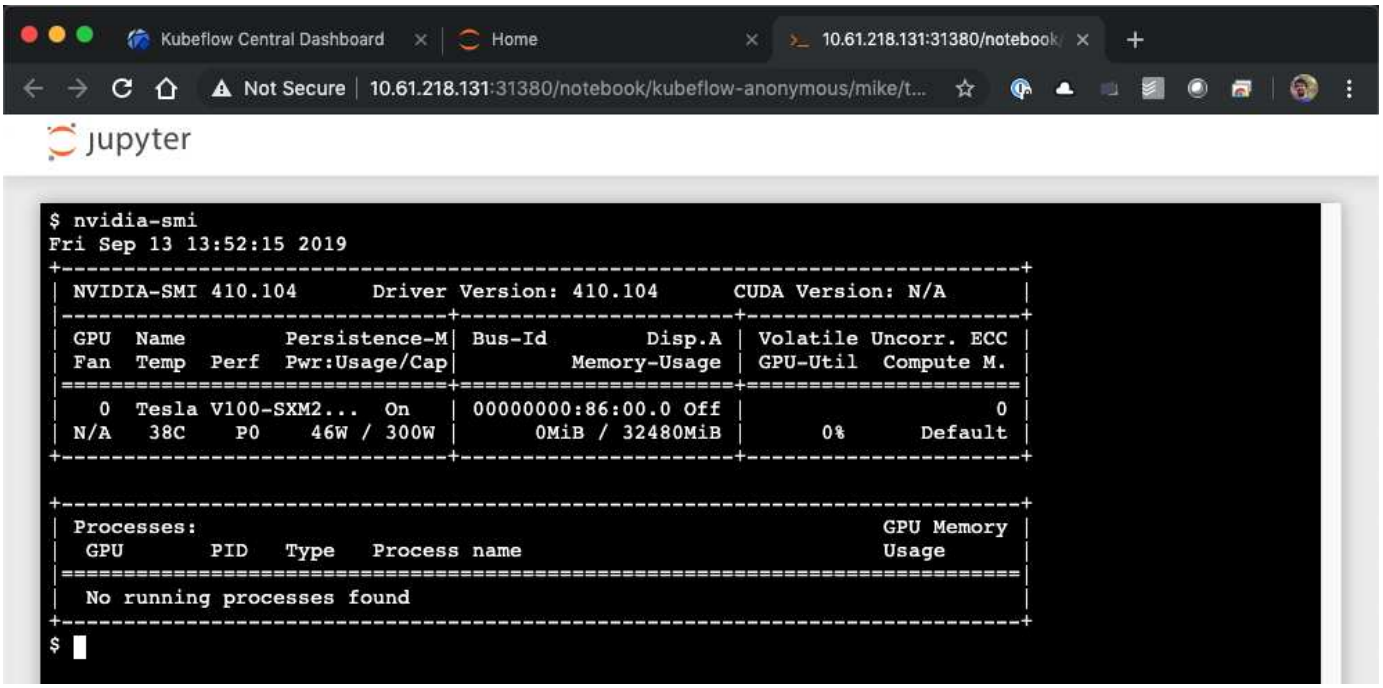
11. Öffnen Sie ein Terminal und, vorausgesetzt, dass ein neues Volume in Schritt 5 angefordert wurde, führen Sie aus `df -h` So bestätigen Sie, dass ein neues persistentes Volume mit Trident-Bereitstellung als Standardarbeitsbereich gemountet wird.

    Das Standard-Workspace-Verzeichnis ist das Basisverzeichnis, mit dem Sie angezeigt werden, wenn Sie zum ersten Mal auf die Web-Schnittstelle des Servers zugreifen. Auf diesem persistenten Volume durch Trident werden daher alle mit der Web-Schnittstelle erstellten Artefakte gespeichert.





12. Führen Sie mit dem Terminal aus `nvidia-smi` Um zu bestätigen, dass dem Notebook-Server die korrekte Anzahl an GPUs zugewiesen wurde. Im folgenden Beispiel wurde dem Notebook-Server eine GPU zugewiesen, wie in Schritt 7 angefordert.



Beispiel für Notebooks und Pipelines

Der "Das NetApp Data Science Toolkit für Kubernetes" Kann in Verbindung mit Kubeflow verwendet werden. Die Verwendung des NetApp Data Science Toolkit und Kubeflow bietet folgende Vorteile:

- Data Scientists können erweiterte NetApp Datenmanagement-Vorgänge direkt über ein Jupyter Notebook ausführen.
- Erweiterte Datenmanagement-Vorgänge von NetApp können mithilfe des Kubeflow Pipelines in automatisierte Workflows integriert werden.

Siehe "Kubeflow Beispiele" Im Abschnitt zum NetApp Data Science Toolkit GitHub Repository finden Sie weitere Informationen zur Verwendung des Toolkit mit Kubeflow.

Apache Airflow Deployment

NetApp empfiehlt, Apache Airflow zusätzlich zu Kubernetes auszuführen. In diesem Abschnitt werden die Aufgaben beschrieben, die Sie zur Implementierung von Airflow in Ihrem Kubernetes-Cluster ausführen müssen.

> Es ist möglich, Airflow auf anderen Plattformen als Kubernetes bereitzustellen. Die Implementierung von Airflow auf anderen Plattformen als Kubernetes ist nicht im Umfang dieser Lösung enthalten.

Voraussetzungen

Bevor Sie die in diesem Abschnitt beschriebenen Bereitstellungsaufgaben ausführen, gehen wir davon aus, dass Sie bereits die folgenden Aufgaben ausgeführt haben:

1. Sie verfügen bereits über einen funktionierenden Kubernetes-Cluster.
2. Sie haben NetApp Trident bereits in Ihrem Kubernetes Cluster installiert und konfiguriert, wie im Abschnitt „NetApp Trident Deployment and Configuration“ beschrieben.

Installieren Sie Helm

Der Luftstrom wird über Helm, einen beliebten Paketmanager für Kubernetes, implementiert. Bevor Sie Airflow bereitstellen, müssen Sie Helm auf dem Bereitstellungs-Jump-Host installieren. Um Helm auf dem Sprunghost für die Bereitstellung zu installieren, folgen Sie dem "Installationsanweisungen" In der offiziellen Helm-Dokumentation.

Standard-Kubernetes StorageClass festlegen

Bevor Sie Airflow bereitstellen, müssen Sie eine Standard-StorageClass in Ihrem Kubernetes-Cluster zuweisen. Der Airflow-Implementierungsprozess versucht, mithilfe der Standard-StorageClass neue persistente Volumes bereitzustellen. Wenn keine StorageClass als Standard-StorageClass festgelegt ist, schlägt die Bereitstellung fehl. Befolgen Sie die Anweisungen im Abschnitt, um eine Standard-StorageClass in Ihrem Cluster festzulegen "Kubeflow Deployment". Wenn Sie bereits eine Standard-StorageClass innerhalb Ihres Clusters festgelegt haben, können Sie diesen Schritt überspringen.

Verwenden Sie Helm zum Bereitstellen des Luftstroms

Um Airflow mithilfe von Helm in Ihren Kubernetes-Cluster zu implementieren, führen Sie die folgenden Aufgaben vom Bereitstellungs-Jump-Host aus:

1. Setzen Sie den Luftstrom mithilfe von Helm ein, indem Sie den folgen "Implementierungsanleitungen" Für das offizielle Airflow-Diagramm auf dem Artefakt-Hub. Die folgenden Beispielbefehle zeigen die Bereitstellung von Airflow mit Helm. Ändern, Hinzufügen und/oder Entfernen von Werten im `custom-values.yaml` Datei nach Bedarf, abhängig von Ihrer Umgebung und der gewünschten Konfiguration.

```
$ 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
```

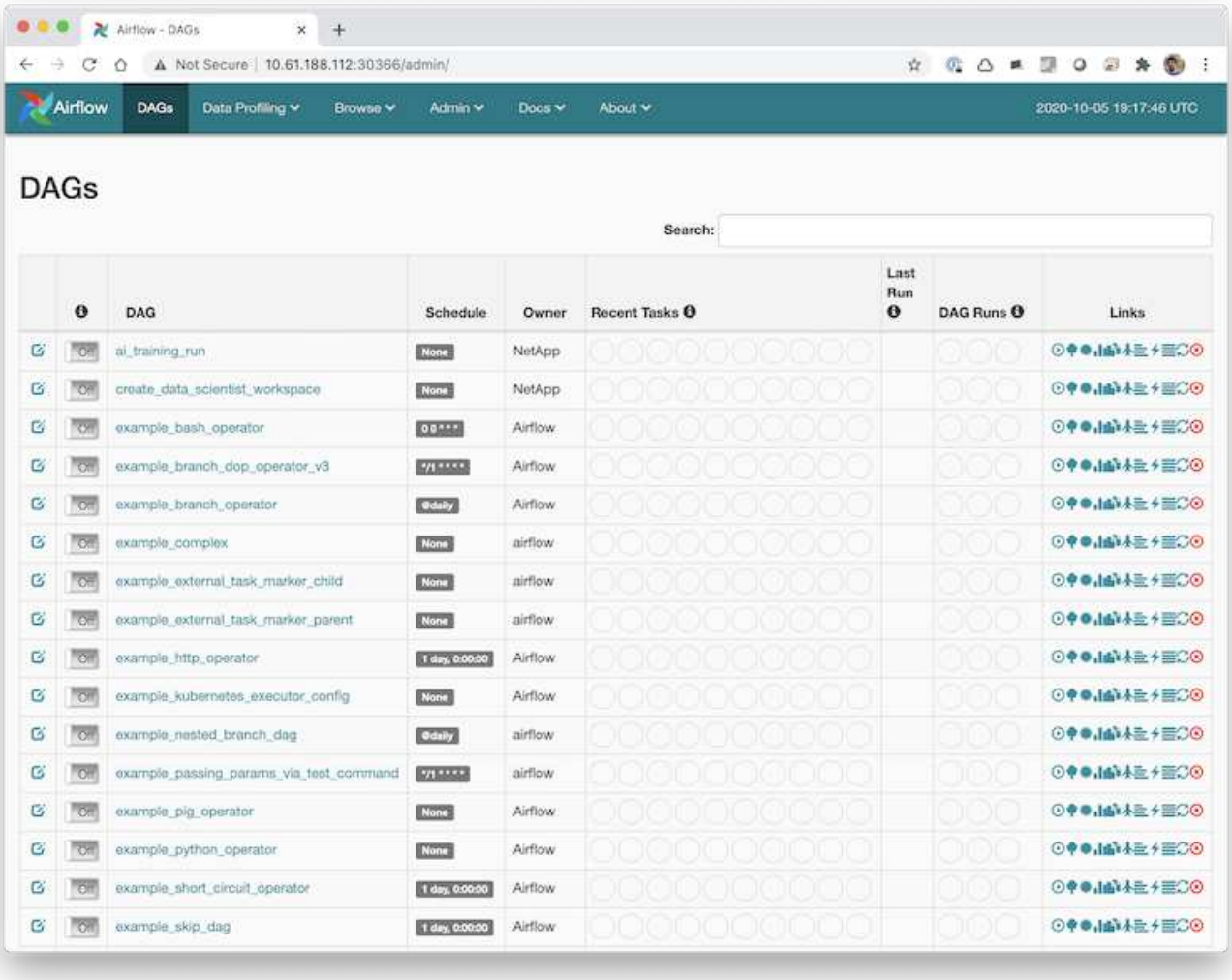
2. Vergewissern Sie sich, dass alle Airflow-Pods betriebsbereit sind. Es kann ein paar Minuten dauern, bis alle Pods beginnen.

```
$ 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. Rufen Sie die URL des Airflow Webservice ab, indem Sie die Anweisungen befolgen, die bei der Bereitstellung von Airflow mit Hilfe von Helm in Schritt 1 an der Konsole gedruckt wurden.

```
$ 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. Vergewissern Sie sich, dass Sie auf den Airflow Webservice zugreifen können.



Beispiel Für Apache Airflow Workflows

Der "Das NetApp Data Science Toolkit für Kubernetes" Kann in Verbindung mit Airflow verwendet werden. Mit dem NetApp Data Science Toolkit in Airflow können Sie NetApp Datenmanagement-Operationen in automatisierte Workflows einbinden, die mit Airflow orchestriert sind.

Siehe "Beispiele Für Luftströmungen" Im Abschnitt zum NetApp Data Science Toolkit GitHub Repository finden Sie weitere Informationen zur Verwendung des Toolkit mit Airflow.

Beispiel Für Trident-Vorgänge

Dieser Abschnitt enthält Beispiele für verschiedene Vorgänge, die Sie mit Trident durchführen können.

Importieren eines vorhandenen Volumes

Wenn bereits vorhandene Volumes auf dem NetApp Storage-System/auf der Plattform vorhanden sind, die Sie in Containern innerhalb des Kubernetes Clusters einbinden möchten, die jedoch nicht an PVCs im Cluster gebunden sind, müssen Sie diese Volumes importieren. Sie können diese Volumes mit der Trident-Funktion für

den Volume-Import importieren.

Die folgenden Beispielbefehle zeigen den Import des gleichen Volumes mit dem Namen `pb_fg_all`, Zweimal einmal pro Trident Back-End, das im Beispiel im Abschnitt erstellt wurde "Beispiel Trident Back-Ends für ONTAP KI-Implementierungen", Schritt 1. Wenn Sie dasselbe Volume zweimal auf diese Weise importieren, können Sie das Volume (ein vorhandenes FlexGroup Volume) mehrmals zwischen verschiedenen LIFs mounten, wie im Abschnitt beschrieben "Beispiel Trident Back-Ends für ONTAP KI-Implementierungen", Schritt 1. Weitere Informationen zu PVCs finden Sie im "Offizielle Kubernetes-Dokumentation". Weitere Informationen zur Importfunktion des Volumes finden Sie im "Trident Dokumentation".

An `accessModes` Der Wert von `ReadOnlyMany` Ist in den Beispiel-PVC-Spezifikationsdateien angegeben. Weitere Informationen zum `accessMode` Feld, siehe "Offizielle Kubernetes-Dokumentation".

> Die Backend-Namen, die in den folgenden Beispiel-Importbefehlen angegeben sind, entsprechen den Backends, die im Beispiel im Abschnitt erstellt wurden "Beispiel Trident Back-Ends für ONTAP KI-Implementierungen", Schritt 1. Die StorageClass-Namen, die im folgenden Beispiel PVC-Definitionsdateien angegeben sind, entsprechen den StorageClasses, die im Beispiel im Abschnitt erstellt wurden "Beispiel: Kubernetes StorageClasses für ONTAP KI-Implementierungen", Schritt 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
```

Bereitstellung eines neuen Volumes

Mit Trident können Sie ein neues Volume auf Ihrem NetApp Storage-System oder Ihrer NetApp Plattform bereitstellen. Mit den folgenden Beispielbefehlen wird die Bereitstellung eines neuen FlexVol Volumes angezeigt. In diesem Beispiel wird das Volume mit der StorageClass bereitgestellt, die im Beispiel im Abschnitt erstellt wurde "Beispiel: Kubernetes StorageClasses für ONTAP KI-Implementierungen", Schritt 2.

An `accessModes` Der Wert von `ReadWriteMany` Wird in der folgenden Beispiel-PVC-Definitionsdatei angegeben. Weitere Informationen zum `accessMode` Feld, siehe "Offizielle Kubernetes-Dokumentation".

```
$ 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
```

Beispiele für hochperformante Jobs für ONTAP KI-Implementierungen

Dieser Abschnitt enthält Beispiele für verschiedene hochperformante Jobs, die ausgeführt werden können, wenn Kubernetes auf einem ONTAP AI POD implementiert wird.

Beispiele für hochperformante Jobs für ONTAP KI-Implementierungen

Dieser Abschnitt enthält Beispiele für verschiedene hochperformante Jobs, die ausgeführt werden können, wenn Kubernetes auf einem ONTAP AI POD implementiert wird.

Single-Node-KI-Workload ausführen

Um einen Single-Node-KI- und -ML-Job in Ihrem Kubernetes-Cluster auszuführen, führen Sie die folgenden Aufgaben vom Bereitstellungs-Jump-Host aus: Mit Trident lässt sich ein Daten-Volume schnell und einfach erstellen, das möglicherweise mehrere Petabyte an Daten enthält und damit für Kubernetes-Workloads zugänglich ist. Wenn ein solches Daten-Volume über einen Kubernetes Pod zugänglich sein soll, geben Sie in der Pod-Definition einfach ein PVC an. Dieser Schritt ist ein nativer Kubernetes-Betrieb. Es ist kein NetApp Fachwissen erforderlich.

> In diesem Abschnitt wird vorausgesetzt, dass Sie bereits den spezifischen KI- und ML-Workload in einem Container (im Docker Container-Format) bereitstellen, den Sie in Ihrem Kubernetes-Cluster ausführen möchten.

1. Die folgenden Beispielbefehle zeigen die Erstellung eines Kubernetes-Jobs für einen TensorFlow Benchmark-Workload, bei dem der ImageNet-Datensatz verwendet wird. Weitere Informationen zum ImageNet-Datensatz finden Sie im "ImageNet-Website".

   Dieses Beispieljob fordert acht GPUs an und kann daher auf einem einzelnen GPU-Worker-Node mit acht oder mehr GPUs ausgeführt werden. Dieser Beispieljob kann in einem Cluster eingereicht werden, für den ein Worker-Node mit acht oder mehr GPUs nicht vorhanden ist oder derzeit mit einem anderen Workload belegt ist. Wenn dies der Fall ist, bleibt der Job in einem ausstehenden Status, bis ein solcher Worker-Node verfügbar ist.

   Zusätzlich wird das Volume, das die erforderlichen Trainingsdaten enthält, zur Maximierung der Storage-Bandbreite zweimal innerhalb des POD angehängt, das diesen Job erstellt. Ein weiteres Volume wird ebenfalls im POD gemountet. Dieses zweite Volume wird zur Speicherung der Ergebnisse und Kennzahlen verwendet. Diese Volumes werden in der Jobdefinition unter Verwendung der Namen der VES referenziert. Weitere Informationen zu Kubernetes-Jobs finden Sie im "Offizielle Kubernetes-Dokumentation".

   An `emptyDir` Volumen mit einem `medium` Der Wert von `Memory` Ist in angehängt `/dev/shm` In dem POD, den dieser Beispieljob erzeugt. Die Standardgröße des `/dev/shm` Das virtuelle Volume, das automatisch durch die Docker Container-Laufzeit erstellt wird, kann manchmal nicht ausreichen, um die Anforderungen von TensorFlow zu erfüllen. Montage an `emptyDir` Das Volumen wie im folgenden Beispiel bietet eine ausreichend große Menge `/dev/shm` Virtuelles Volume: Finden Sie weitere Informationen zu `emptyDir` Volumes, siehe "Offizielle Kubernetes-Dokumentation".

   Der einzelne Container, der in dieser Beispieljobdefinition angegeben wird, wird als angegeben `securityContext > privileged` Der Wert von `true`. Dieser Wert bedeutet, dass der Container

effektiv Root-Zugriff auf dem Host hat. Diese Annotation wird in diesem Fall verwendet, da der spezifische Workload ausgeführt wird und den Root-Zugriff erfordert. Insbesondere für einen Vorgang mit klarem Cache, der von dem Workload ausgeführt wird, ist ein Root-Zugriff erforderlich. Ob oder nicht `privileged: true` Eine Anmerkung ist erforderlich, abhängig von den Anforderungen des spezifischen Workloads, die Sie ausführen.

```
$ 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. Vergewissern Sie sich, dass der in Schritt 1 erstellte Job korrekt ausgeführt wird. Der folgende Beispielbefehl bestätigt, dass für den Job ein einzelner Pod erstellt wurde, wie in der Jobdefinition angegeben, und dass dieser Pod derzeit auf einem der GPU-Worker-Nodes ausgeführt wird.

```
$ 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. Bestätigen Sie, dass der in Schritt 1 erstellte Job erfolgreich abgeschlossen wurde. Mit den folgenden Beispielbefehlen wird bestätigt, dass der Job erfolgreich abgeschlossen wurde.

```
$ 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. **Optional:** Aufräumen von Auftragsartefakten. Die folgenden Beispielbefehle zeigen das Löschen des in Schritt 1 erstellten Jobobjekts.

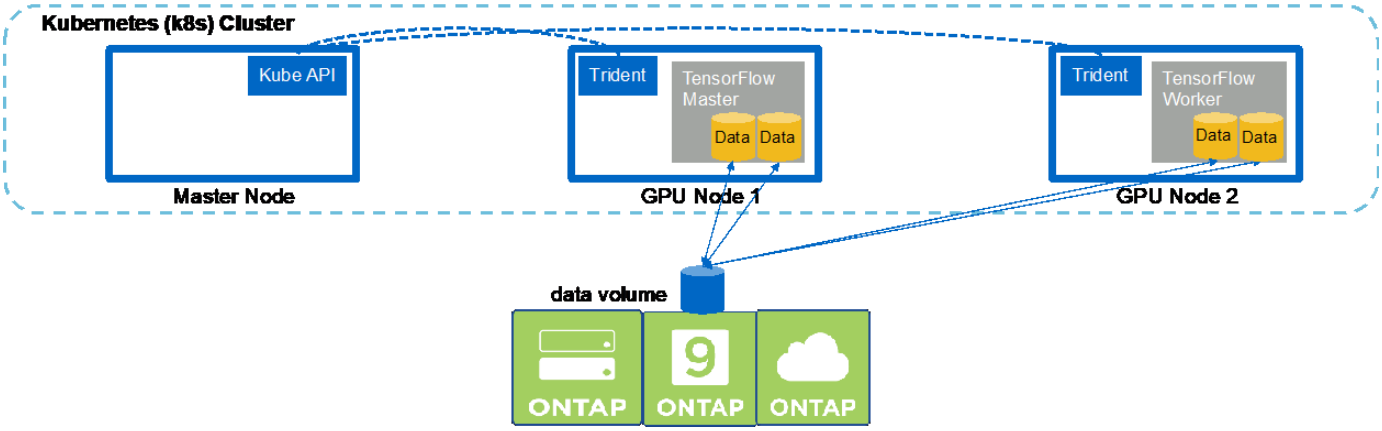
   Wenn Sie das Jobobjekt löschen, löscht Kubernetes automatisch alle zugehörigen Pods.

```
$ 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.
```

Ausführung eines synchronen, verteilten KI-Workloads

Um einen synchronen AI- und ML-Job mit mehreren Nodes in Ihrem Kubernetes-Cluster auszuführen, führen Sie die folgenden Aufgaben auf dem Jump-Host der Implementierung durch. Mit diesem Prozess können Sie auf einem NetApp Volume gespeicherte Daten nutzen und mehr GPUs verwenden, als ein einzelner Worker-Node bieten kann. Die folgende Abbildung zeigt einen synchronen, verteilten KI-Job.

> Synchrone, verteilte Jobs können dazu beitragen, die Performance und die Trainingsgenauigkeit im Vergleich zu asynchronen, verteilten Jobs zu steigern. Eine Diskussion über die vor- und Nachteile von synchronen Jobs gegenüber asynchronen Jobs geht nicht in dieses Dokument über.



1. Die folgenden Beispielbefehle zeigen die Erstellung eines Workers, der an der synchronen, verteilten Ausführung desselben TensorFlow Benchmark-Jobs beteiligt ist, der auf einem einzelnen Node im Beispiel im Abschnitt ausgeführt wurde "Single-Node-KI-Workload ausführen". In diesem speziellen Beispiel wird nur ein einziger Worker bereitgestellt, da der Job über zwei Worker-Nodes ausgeführt wird.

Dieses Beispiel für die Implementierung eines Mitarbeiters fordert acht GPUs an und kann so auf einem einzelnen GPU-Worker-Node mit acht oder mehr GPUs ausgeführt werden. Wenn Ihre GPU-Worker-Nodes mehr als acht GPUs aufweisen, um die Performance zu maximieren, könnte die Anzahl dieser GPUs erhöht werden, die der Funktion der Worker-Nodes entsprechen. Weitere Informationen über Kubernetes-Implementierungen finden Sie im "Offizielle Kubernetes-Dokumentation".

In diesem Beispiel wird eine Kubernetes-Implementierung erstellt, da dieser spezifische Container-Worker niemals eigenständig abgeschlossen sein würde. Daher macht es keinen Sinn, es durch die Verwendung des Kubernetes Job-Konstrukts zu implementieren. Wenn Ihr Mitarbeiter für sich selbst konzipiert oder geschrieben wurde, ist es möglicherweise sinnvoll, das Job-Konstrukt für die Bereitstellung Ihres Mitarbeiters zu verwenden.

Dem POD, der in diesem Beispiel der Implementierungsspezifikation angegeben ist, wird eine zugewiesen `hostNetwork` Der Wert von `true`. Dieser Wert bedeutet, dass der Pod den Netzwerk-Stack des Host-Mitarbeiters-Nodes anstelle des virtuellen Netzwerk-Stacks verwendet, den Kubernetes normalerweise für jeden Pod erstellt. Diese Annotation wird in diesem Fall verwendet, da der spezifische Workload auf Open MPI, NCCL und Horovod angewiesen ist, um den Workload in einer synchronen, verteilten Art und Weise auszuführen. Daher ist für diese Lösung der Zugriff auf den Host-Netzwerk-Stack erforderlich. Eine Diskussion über Open MPI, NCCL und Horovod geht nicht in dieses Dokument über. Ob oder nicht `hostNetwork: true` Eine Anmerkung ist erforderlich, abhängig von den Anforderungen des spezifischen Workloads, die Sie ausführen. Weitere Informationen zum `hostNetwork` Feld, siehe "Offizielle Kubernetes-Dokumentation".

```
$ 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. Bestätigen Sie, dass die in Schritt 1 erstellte Workers-Bereitstellung erfolgreich gestartet wurde. Die folgenden Beispielbefehle bestätigen, dass ein Pod für einzelne Mitarbeiter für die Implementierung erstellt wurde, wie in der Implementierungsdefinition angegeben, und dass dieser Pod derzeit auf einem der GPU-Worker-Nodes ausgeführt wird.

```
$ 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. Kubernetes-Job für einen Master erstellen, der zu Beginn startet, an dem er teilnimmt und die Ausführung des synchronen Jobs mit mehreren Nodes verfolgt. Die folgenden Beispielbefehle erzeugen einen Master, der abstartet, an dem teilnimmt und die synchrone, verteilte Ausführung desselben TensorFlow-Benchmark-Jobs verfolgt, der auf einem einzelnen Node im Beispiel im Abschnitt ausgeführt wurde "Single-Node-KI-Workload ausführen".

   Dieses Beispiel-Master-Job fordert acht GPUs an, wodurch auf einem einzelnen GPU-Worker-Node mit acht oder mehr GPUs ausgeführt werden kann. Wenn Ihre GPU-Worker-Nodes mehr als acht GPUs aufweisen, um die Performance zu maximieren, könnte die Anzahl dieser GPUs erhöht werden, die der Funktion der Worker-Nodes entsprechen.

   Der Master-Pod, der in dieser Beispiel-Jobdefinition angegeben wird, wird A zugewiesen `hostNetwork` Der Wert von `true`, So wie der Arbeiter POD wurde ein `hostNetwork` Der Wert von `true` In Schritt 1. Weitere Informationen dazu, warum dieser Wert notwendig ist, finden Sie in Schritt 1.

```
$ 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. Vergewissern Sie sich, dass der in Schritt 3 erstellte Master-Job korrekt ausgeführt wird. Der folgende Beispielbefehl bestätigt, dass für den Job ein einzelner Master-Pod erstellt wurde, wie in der Jobdefinition angegeben, und dass dieser Pod derzeit auf einem der GPU-Worker-Nodes ausgeführt wird. Sie sollten auch sehen, dass der Worker Pod, den Sie ursprünglich in Schritt 1 gesehen haben, noch läuft und dass die Master- und Worker-Pods auf unterschiedlichen Nodes ausgeführt werden.

```
$ 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. Vergewissern Sie sich, dass der in Schritt 3 erstellte Masterjob erfolgreich abgeschlossen wurde. Mit den folgenden Beispielbefehlen wird bestätigt, dass der Job erfolgreich abgeschlossen wurde.

```
$ 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. Löschen Sie die Mitarbeiterbereitstellung, wenn Sie sie nicht mehr benötigen. Die folgenden Beispielbefehle zeigen das Löschen des in Schritt 1 erstellten Workers Deployment-Objekts.

Wenn Sie das Bereitstellungsobjekt für Mitarbeiter löschen, löscht Kubernetes automatisch alle zugehörigen „Worker“-Pods.

```
$ 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. **Optional:** Säubern Sie die Master Job Artefakte. Die folgenden Beispielbefehle zeigen das Löschen des in Schritt 3 erstellten Master-Jobobjekts.

   Wenn Sie das Master-Job-Objekt löschen, löscht Kubernetes automatisch alle zugehörigen Master-Pods.

```
$ 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.
```

Performance-Tests

Bei der Erstellung dieser Lösung haben wir einen einfachen Performance-Vergleich durchgeführt. Wir haben mithilfe von Kubernetes mehrere standardmäßige NetApp KI-Benchmark-Jobs ausgeführt. Die Benchmark-Ergebnisse wurden mit Ausführungsvorgängen verglichen, die über einen einfachen Docker Run-Befehl ausgeführt wurden. Wir haben keine spürbaren Unterschiede in der Performance gesehen. Daher kamen wir zu dem Schluss, dass die Verwendung von Kubernetes zur Orchestrierung von Training-Jobs in Container-KI keine negative Auswirkung auf die Performance hat. Die Ergebnisse unseres Performance-Vergleichs finden Sie in der folgenden Tabelle.

Benchmark	Datensatz	Docker Run (Images/Sek.)	Kubernetes (Bilder/Sek.)
TensorFlow mit einem Node	Synthetische Daten	6,667.2475	6,661.93125
TensorFlow mit einem Node	ImageNet	6,570.2025	6,530.59125
Synchrones verteiltes TensorFlow mit zwei Knoten	Synthetische Daten	13,213.70625	13,218.288125
Synchrones verteiltes TensorFlow mit zwei Knoten	ImageNet	12,941.69125	12,881.33875

Schlussfolgerung

Unternehmen und Unternehmen aller Größen und Branchen setzen zunehmend auf künstliche Intelligenz (KI), Machine Learning (ML) und Deep Learning (DL), um Probleme aus der Praxis zu lösen, innovative Produkte und Services zu entwickeln und sich in einem immer härter umkämpften Markt einen Schritt voraus zu sein. Beim verstärkten Einsatz von KI, ML und DL müssen Unternehmen mit vielen Herausforderungen konfrontiert werden, darunter Workload-Skalierbarkeit und Datenverfügbarkeit. Diese Herausforderungen können durch den Einsatz der NetApp AI Control Plane Lösung bewältigt werden.

Mit dieser Lösung können Sie einen Daten-Namespace schnell klonen. Darüber hinaus können Kunden AI-, ML- und DL-Training-Workflows definieren und implementieren, die die nahezu sofortige Erstellung von Daten und Modellbasiden für Rückverfolgbarkeit und Versionierung enthalten. Mit dieser Lösung können Sie hinter jedem einzelnen Modell-Training zurück zu den genauen Datensätzen führen, mit denen das Modell geschult und/oder validiert wurde. Und schließlich können Sie mit dieser Lösung schnell Jupyter Notebook-Workspaces mit Zugriff auf riesige Datensätze bereitstellen.

Da diese Lösung auf Data Scientists und Data Engineers ausgerichtet ist, sind minimale Fachkenntnisse von NetApp oder NetApp ONTAP erforderlich. Bei dieser Lösung lassen sich Datenmanagementfunktionen mit einfachen und vertrauten Tools und Schnittstellen ausführen. Darüber hinaus nutzt diese Lösung vollständig Open-Source- und freie Komponenten. Wenn Sie also bereits NetApp Storage in Ihrer Umgebung haben,

können Sie diese Lösung heute implementieren. Wenn Sie diese Lösung testen möchten, aber noch nicht bereits über NetApp Storage verfügen, besuchen Sie "cloud.netapp.com", Und Sie sind innerhalb kürzester Zeit mit einer Cloud-basierten NetApp Storage-Lösung einsatzbereit.

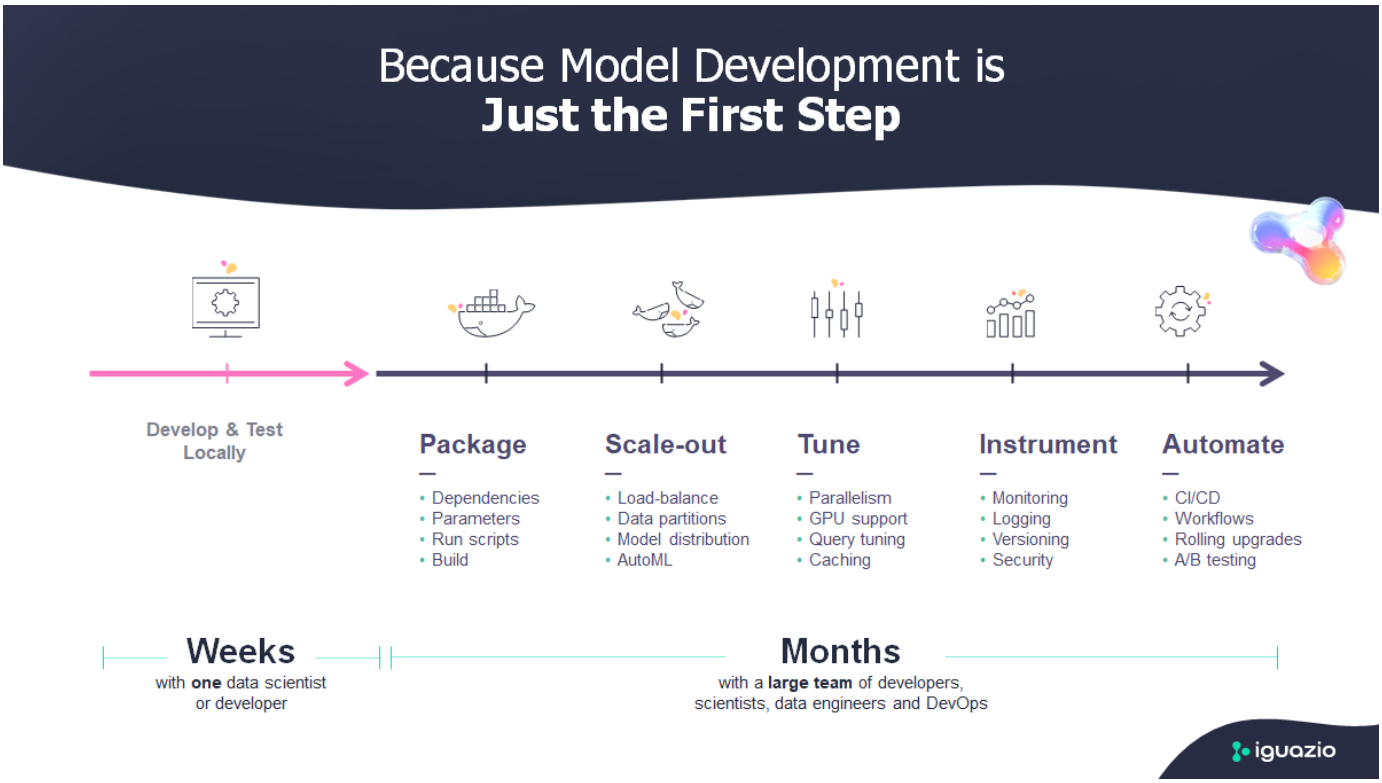
MLRun-Pipeline mit Iguazio

TR-4834: NetApp und Iguazio für MLRun Pipeline

Rick Huang, David Arnette, NetApp Marcelo Litovsky, Iguazio

In diesem Dokument werden die Details der MLRun-Pipeline unter Verwendung von NetApp ONTAP AI, der NetApp AI Control Plane, der NetApp Cloud Volumes Software und der Iguazio Data Science Platform behandelt. Wir verwendeten Nuclio serverlose Funktion, Kubernetes Persistent Volumes, NetApp Cloud Volumes, NetApp Snapshot Kopien, Grafana Dashboard, Und andere Dienste auf der Iguazio-Plattform, um eine End-to-End-Daten-Pipeline für die Simulation von Netzwerkausfall Erkennung zu bauen. Die Technologien von Iguazio und NetApp wurden integriert, um schnelle Modellbereitstellung, Datenreplizierung und Produktionsüberwachungsfunktionen vor Ort und in der Cloud zu ermöglichen.

Der Fokus der Arbeit eines Datenwissenschaftlers sollte auf das Training und Tuning von ml- (Machine Learning) und künstlicher Intelligenz (KI)-Modellen liegen. Allerdings verbringen Data Scientists nach Untersuchungen von Google ~80 % ihrer Zeit damit, herauszufinden, wie ihre Modelle mit Enterprise-Applikationen funktionieren und im Maßstab laufen lassen können, wie in der folgenden Abbildung dargestellt, die Modellentwicklung im AI/ML-Workflow.



Für das Management von End-to-End-KI/ML-Projekten ist ein umfassenderes Verständnis von Unternehmenskomponenten erforderlich. Obwohl DevOps die Definition, Integration und Implementierung

dieser Typen von Komponenten übernommen hat, zielen Machine-Learning-Vorgänge auf einen ähnlichen Flow ab, der KI/ML-Projekte umfasst. Eine Vorstellung der Merkmale einer End-to-End-KI/ML-Pipeline im Unternehmen erhalten Sie in der folgenden Liste der erforderlichen Komponenten:

- Storage
- Netzwerkbetrieb
- Datenbanken
- File-Systeme
- Container
- CI/CD-Pipeline (Continuous Integration und Continuous Deployment
- Integrierte Entwicklungsumgebung (IDE)
- Sicherheit
- Datenzugriffsrichtlinien
- Trennt
- Cloud
- Einheitliche
- Toolsets und Bibliotheken für Data Science

In diesem Whitepaper zeigen wir, wie die Partnerschaft zwischen NetApp und Iguazio die Entwicklung einer End-to-End-KI/ML-Pipeline deutlich vereinfacht. Dadurch wird die Markteinführungszeit für alle Ihre KI/ML-Applikationen verkürzt.

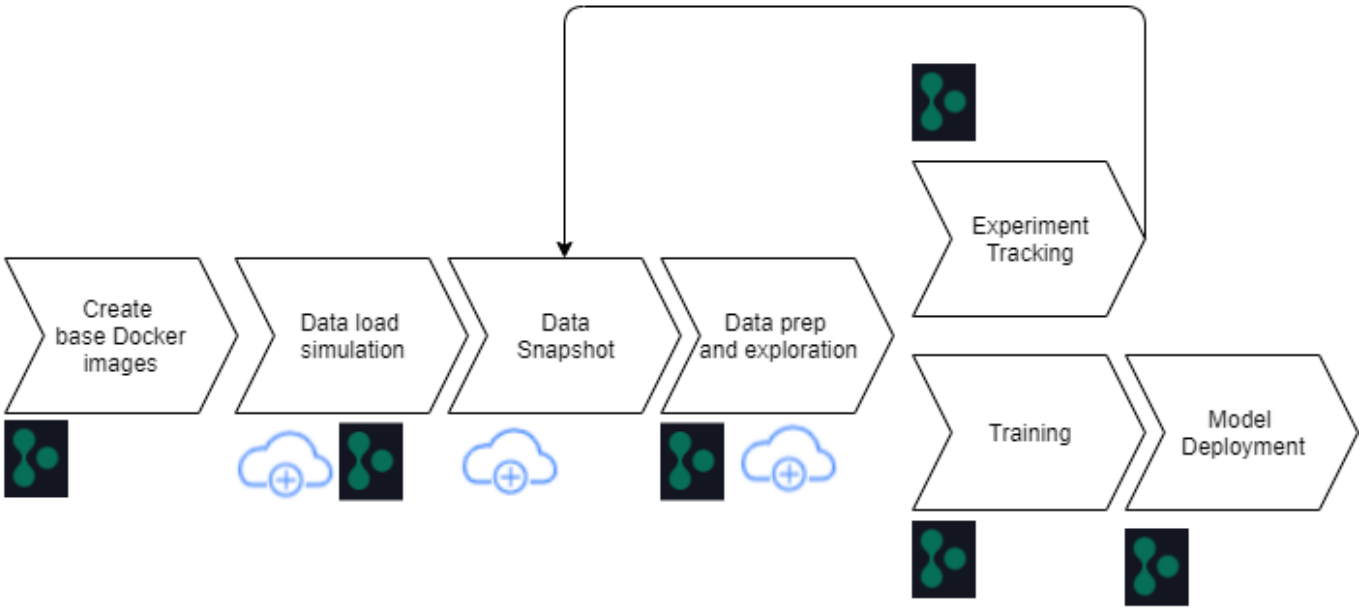
Zielgruppe

Die Welt der Datenwissenschaft berührt mehrere Disziplinen in der Informationstechnologie und im Business.

- Der Data Scientist benötigt die Flexibilität, ihre Tools und Bibliotheken einzusetzen.
- Der Data Engineer muss wissen, wie die Daten fließen und wo sie sich befinden.
- DevOps-Engineers benötigen die Tools, um neue KI/ML-Applikationen in ihre CI/CD-Pipelines zu integrieren.
- Geschäftsanwender möchten auf KI-/ML-Applikationen zugreifen können. Wir beschreiben, wie NetApp und Iguazio jede dieser Rollen unterstützen, mit unseren Plattformen einen Mehrwert für das Unternehmen zu schaffen.

Lösungsüberblick

Diese Lösung folgt dem Lebenszyklus einer KI/ML-Applikation. Zunächst legen wir die Arbeit der Datenanalysten fest, welche Schritte für die Datenvorbereitung und das Training und die Implementierung der Modelle erforderlich sind. Wir befolgen die erforderlichen Arbeiten, um eine vollständige Pipeline zu erstellen mit der Möglichkeit, Artefakte zu verfolgen, mit der Durchführung zu experimentieren und sich nach Kubeflow zu implementieren. Um den gesamten Zyklus abzuschließen, integrieren wir die Pipeline mit NetApp Cloud Volumes, um die Datenversionierung zu ermöglichen, wie im folgenden Bild zu sehen ist.



iguazio

Cloud Volumes ONTAP

Technologischer Überblick

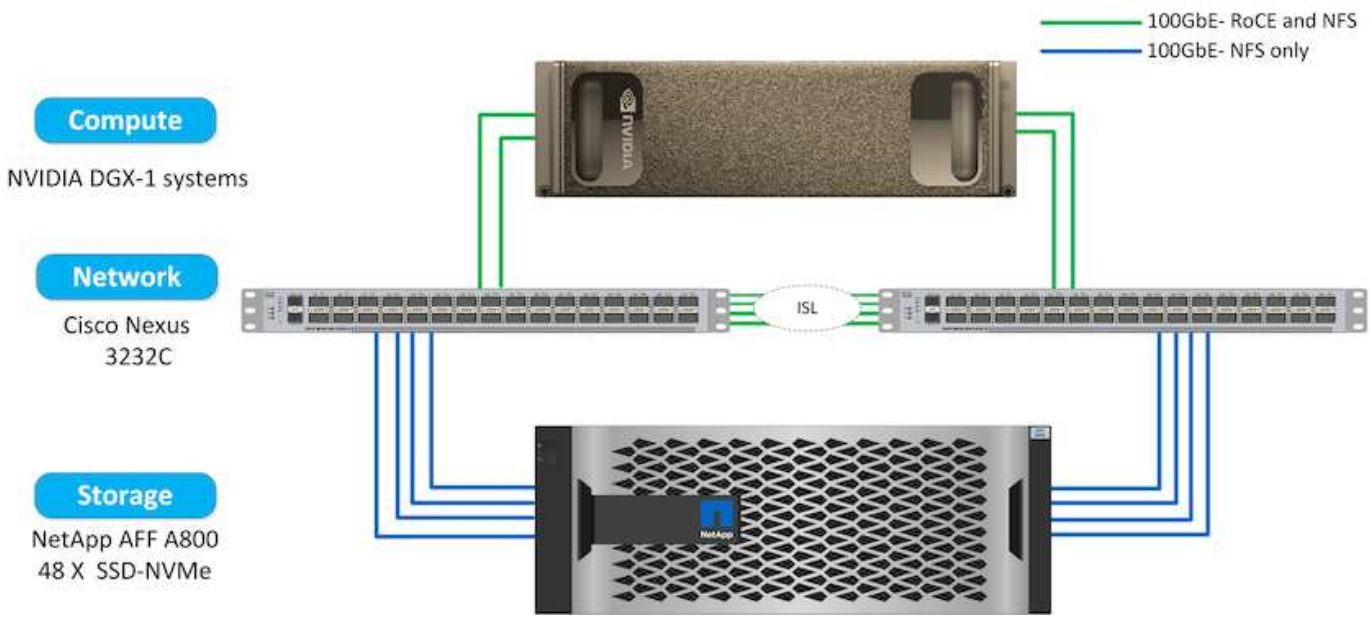
Übersicht über NetApp

NetApp ist die Instanz für Daten in der Hybrid Cloud. Mit einem Portfolio an Hybrid Cloud Data Services, die das Management von Applikationen und Daten über Cloud- und On-Premises-Umgebungen hinweg vereinfachen, beschleunigt NetApp den digitalen Wandel. Gemeinsam mit Partnern hilft NetApp Unternehmen weltweit, das volle Potenzial ihrer Daten auszuschöpfen und so Touchpoints zu Kunden auszubauen, Innovationen voranzutreiben und Betriebsabläufe zu optimieren.

NetApp ONTAP AI

Mit NetApp ONTAP AI mit Unterstützung von NVIDIA DGX Systemen und NetApp All-Flash-Storage mit Cloud-Integration optimieren Sie den Datenfluss zuverlässig und beschleunigen Analysen, Training und Inferenz über Ihre Data-Fabric-Architektur im gesamten Datacenter (Edge-, Core- und Cloud-Bereich). Sie bietet IT-Abteilungen eine Architektur mit folgenden Vorteilen:

- Sie vereinfacht das Design
- Computing und Storage können unabhängig voneinander skaliert werden
- Kunden können mit einer kleinen Konfiguration starten und nahtlos skalieren
- Bietet eine Reihe von Storage-Optionen für verschiedene Performance- und KostennutzungNetApp ONTAP AI bietet konvergente Infrastruktur-Stacks mit NVIDIA DGX-1, einem KI-System im Petaflop-Bereich und NVIDIA Mellanox High-Performance-Ethernet-Switches, um KI-Workloads zu vereinheitlichen, die Implementierung zu vereinfachen und den ROI zu beschleunigen. Für diesen technischen Bericht haben wir ONTAP AI mit einem DGX-1 und einem NetApp AFF A800 Storage-System genutzt. Das folgende Bild zeigt die Topologie von ONTAP AI mit dem in dieser Validierung verwendeten DGX-1-System.



NetApp AI Control Plane

Die NetApp AI Control Plane ermöglicht Ihnen die optimale Nutzung von KI- und ML-Applikationen mit einer Lösung, die höchste Skalierbarkeit, optimierte Implementierung und ununterbrochene Datenverfügbarkeit bietet. Die AI Control Plane Lösung integriert Kubernetes und Kubeflow in die Data-Fabric-Strategie von NetApp. Kubernetes, die standardmäßige Container-Orchestrierungsplattform der Branche für Cloud-native Bereitstellungen, bietet Skalierbarkeit und Portabilität von Workloads. Kubeflow ist eine Open-Source Machine Learning-Plattform, die Management und Bereitstellung vereinfacht und Entwicklern ermöglicht, mehr Datenwissenschaft in weniger Zeit zu leisten. Eine Data-Fabric-Strategie von NetApp bietet Datenverfügbarkeit und -Portabilität ohne Kompromisse. So wird sichergestellt, dass Ihre Daten in der gesamten Pipeline (Edge, Core, Cloud) aufgerufen werden können. Dieser technische Bericht verwendet die NetApp AI Control Plane in einer MLRun-Pipeline. Das folgende Bild zeigt die Kubernetes-Cluster-Managementseite, auf der Sie unterschiedliche Endpunkte für jedes Cluster haben können. Wir haben persistente NFS-Volumes mit dem Kubernetes-Cluster verbunden. Die folgenden Images zeigen ein persistentes Volume, das mit dem Cluster verbunden ist "NetApp Trident" Unterstützung für persistenten Storage und Datenmanagementfunktionen

Kubernetes Clusters

Discover Cluster

4 Kubernetes Clusters

kubernetes

https://3.20.111.39:6443
Cluster Endpoint

v1.15.5
Cluster Version

19.07.1
Trident Version

0
Working Environments

kubernetes

https://172.31.14.31:6443
Cluster Endpoint

v1.15.5
Cluster Version

19.07.1
Trident Version

1
Working Environments

Persistent Volumes for Kubernetes

Connected with Kubernetes Cluster

Cloud Volumes ONTAP is connected to 1 Kubernetes cluster. View Cluster

You can connect another Kubernetes cluster to this Cloud Volumes ONTAP system. If the Kubernetes cluster is in a different network than Cloud Volumes ONTAP, specify a custom export policy to provide access to clients.

Kubernetes Cluster

Select Kubernetes Cluster

kubernetes

Custom Export Policy *(Optional)*

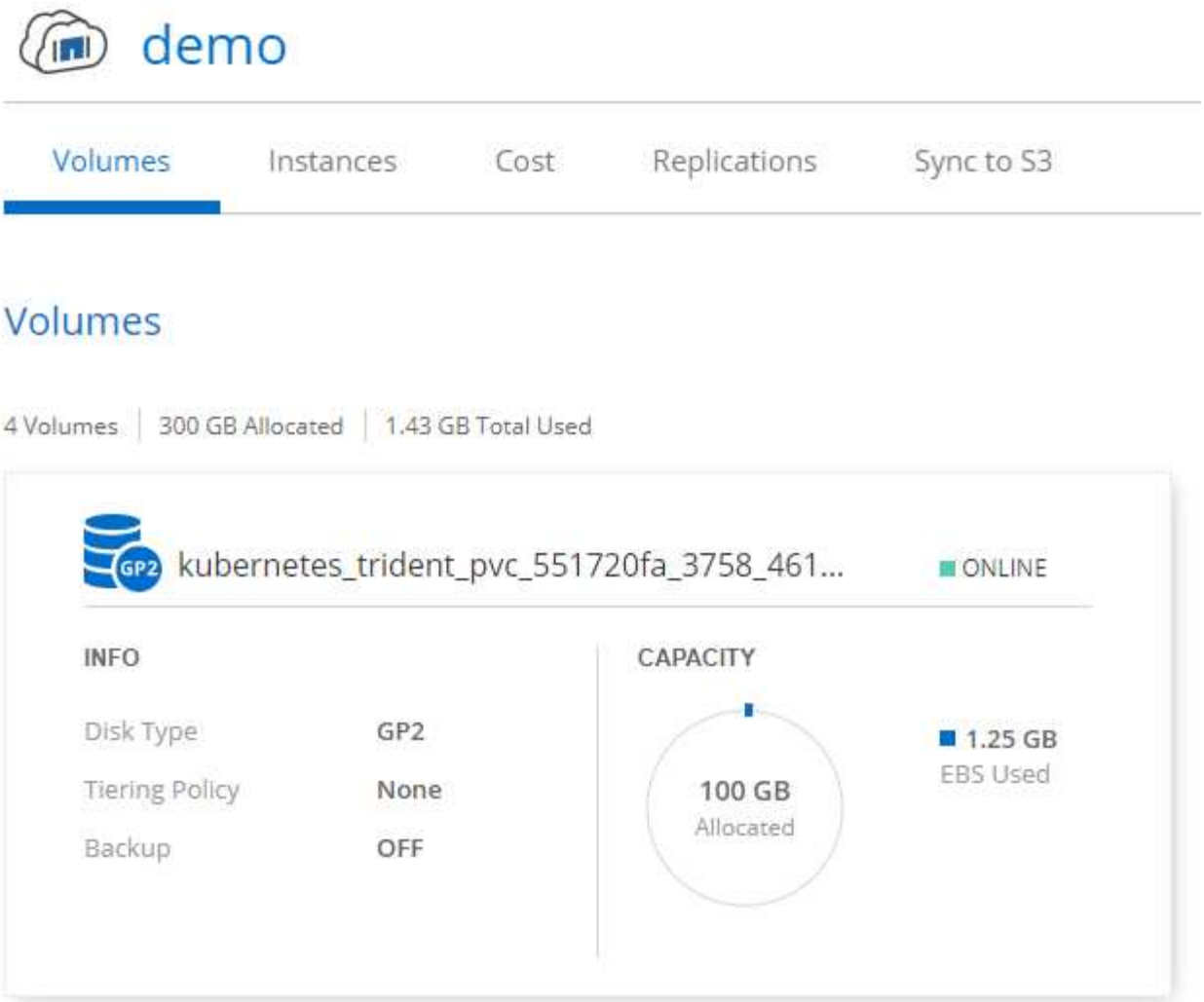
Custom Export Policy

172.31.0.0/16

☑ Set as default storage class

◉ NFS ○ iSCSI

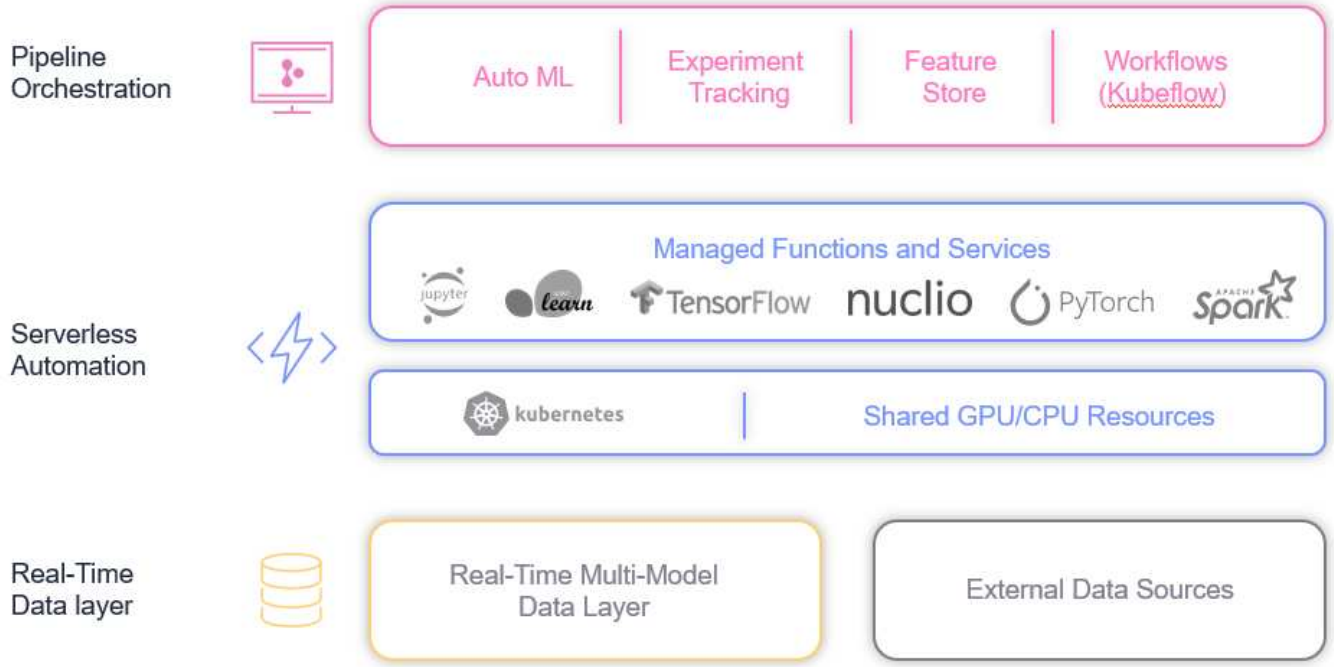
Connect | Cancel



Iguazio Übersicht

Die Iguazio Data Science Platform ist eine vollständig integrierte und sichere Plattform für Data-Science-Dienste (PaaS), die die Entwicklung vereinfacht, die Performance beschleunigt, die Zusammenarbeit erleichtert und betriebliche Herausforderungen bewältigt. Diese Plattform umfasst die folgenden Komponenten, und die Iguazio Data Science Platform wird im folgenden Bild dargestellt:

- Eine Data-Science-Umgebung mit Jupyter Notebooks, integrierten Analyse-Engines und Python-Paketen
- Modellmanagement mit Experimentierverfolgung und automatisierten Pipeline-Funktionen
- Gemanagt von Daten und ML-Services über ein skalierbares Kubernetes-Cluster
- Nuclio, ein Framework ohne Server in Echtzeit
- Eine extrem schnelle und sichere Datenschicht, die SQL, NoSQL, Zeitreihen-Datenbanken, Dateien (einfache Objekte) und Streaming unterstützt
- Integration mit Datenquellen von Drittanbietern wie NetApp, Amazon S3, HDFS, SQL Datenbanken sowie Streaming- und Messaging-Protokollen
- Echtzeit-Dashboards auf Basis von Grafana



Software- und Hardware-Anforderungen erfüllt

Netzwerkkonfiguration

Die folgende Netzwerkkonfigurationsanforderung für die Einrichtung in der Cloud:

- Das Iguazio Cluster und NetApp Cloud Volumes müssen sich in der gleichen virtuellen Private Cloud befinden.
- Der Cloud-Manager muss Zugriff auf Port 6443 auf den Iguazio-App-Nodes haben.
- In diesem technischen Bericht haben wir Amazon Web Services genutzt. Anwender haben jedoch die Möglichkeit, die Lösung bei jedem Cloud-Provider zu implementieren.für lokale Tests in ONTAP AI mit NVIDIA DGX-1 haben wir den gehosteten DNS-Service von Iguazio verwendet.

Clients müssen auf dynamisch erstellte DNS-Domänen zugreifen können. Auf Wunsch können Kunden ihr eigenes DNS verwenden.

Hardwareanforderungen

Sie können Iguazio vor Ort in Ihrem eigenen Cluster installieren. Wir haben die Lösung in NetApp ONTAP AI mit einem NVIDIA DGX-1-System verifiziert. In der folgenden Tabelle ist die Hardware aufgeführt, die zum Testen dieser Lösung verwendet wird.

Trennt	Menge
DGX-1-Systeme	1
NetApp AFF A800 System	1 Hochverfügbarkeitspaar (HA), umfasst 2 Controller und 48 NVMe-SSDs (3,8 TB oder höher)
Netzwerk-Switches der Cisco Nexus 3232C-Serie	2

In der folgenden Tabelle sind die erforderlichen Softwarekomponenten für lokale Tests aufgeführt:

Software	Version oder sonstige Informationen
NetApp ONTAP Datenmanagement-Software	9.7
Cisco NX-OS Switch-Firmware	7.0(3)I6(1)
NVIDIA DGX-BETRIEBSSYSTEM	4.4 – Ubuntu 18.04 LTS
Docker-Container-Plattform	19.03.5
Container-Version	20.01-tf1-py2
Framework für Machine Learning	TensorFlow 1.15.0
Iguazio	Version 2.8 Oder Höher
ESX Server	6.5

Die Lösung wurde mit Iguazio Version 2.5 und NetApp Cloud Volumes ONTAP für AWS vollständig getestet. Das Iguazio Cluster und die NetApp Software werden beide auf AWS ausgeführt.

Software	Version oder Typ
Iguazio	Version 2.8 Oder Höher
App-Node	M5.4xlarge
Daten-Node	I3.4xlarge

Übersicht Über Die Fehlervorhersage Für Netzwerkgeräte: Anwendungsbeispiel

Dieser Anwendungsfall basiert auf einem Iguazio-Kunden im Telekommunikationsbereich in Asien. Da 100.000 Enterprise-Kunden und jedes Jahr 125 Netzwerkausfälle verursacht haben, war es entscheidend, dass sie proaktiv Maßnahmen ergreifen und verhindern müssen, dass Netzwerkausfälle die Kunden beeinträchtigen. Diese Lösung bot den Kunden folgende Vorteile:

- Prädiktive Analysen bei Netzwerkausfällen
- Integration in ein Ticketsystem
- Proaktive Maßnahmen zur Verhinderung von Netzwerkfehlern infolge dieser Implementierung von Iguazio wurden 60 % der Ausfälle proaktiv verhindert.

Setup-Übersicht

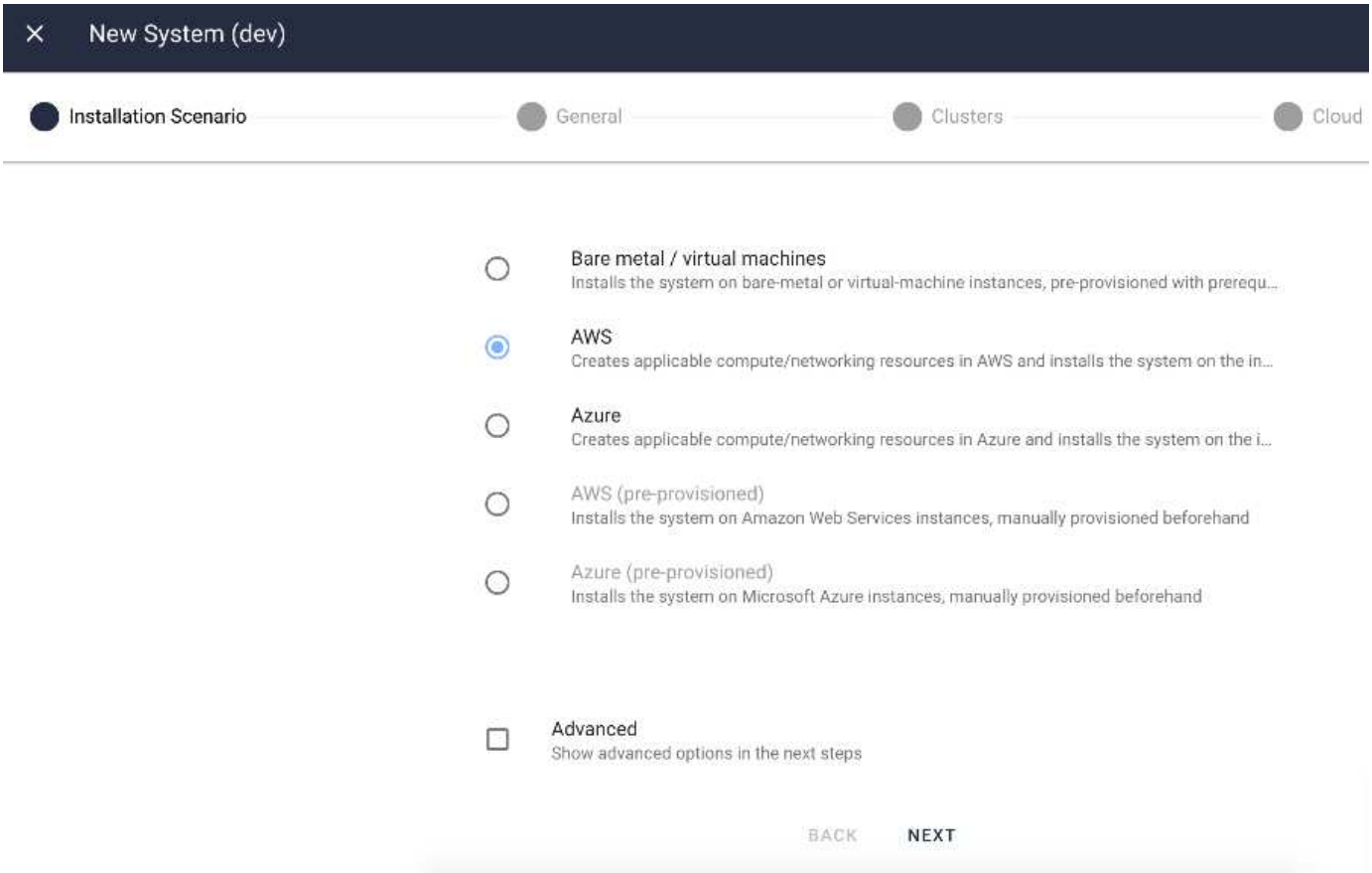
Iguazio kann lokal oder bei einem Cloud-Provider installiert werden.

Iguazio-Installation

Die Bereitstellung kann als Service ausgeführt und von Iguazio oder vom Kunden gemanagt werden. In beiden Fällen bietet Iguazio eine Implementierungsapplikation (Provazio) für die Bereitstellung und das Management von Clustern.

Informationen zur vor-Ort-Installation finden Sie unter "NVA-1121" Für Computing-, Netzwerk- und Storage-Einrichtung. Der vor-Ort-Einsatz von Iguazio wird von Iguazio ohne zusätzliche Kosten für den Kunden bereitgestellt. Siehe "Auf dieser Seite" Für DNS- und SMTP-Serverkonfigurationen. Die Seite für die

Installation von Provazio wird wie folgt angezeigt.



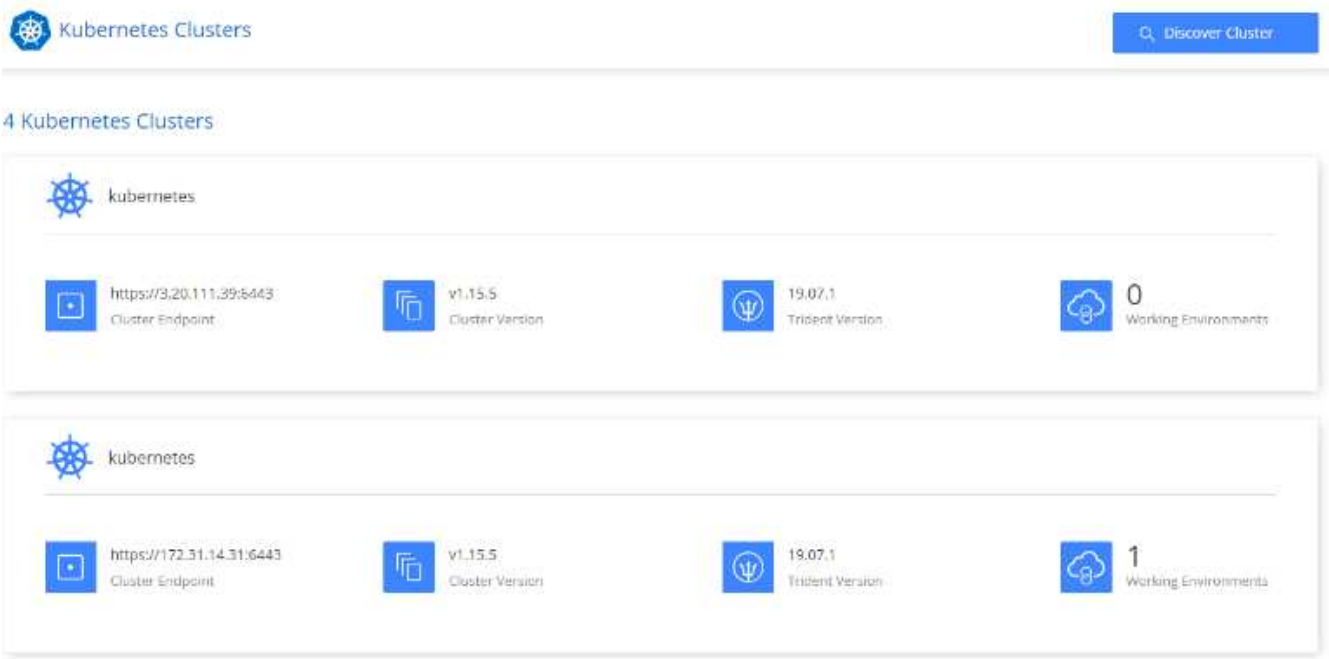
Kubernetes-Cluster Konfigurieren

Dieser Abschnitt ist in zwei Teile für die Cloud und lokale Implementierung unterteilt.

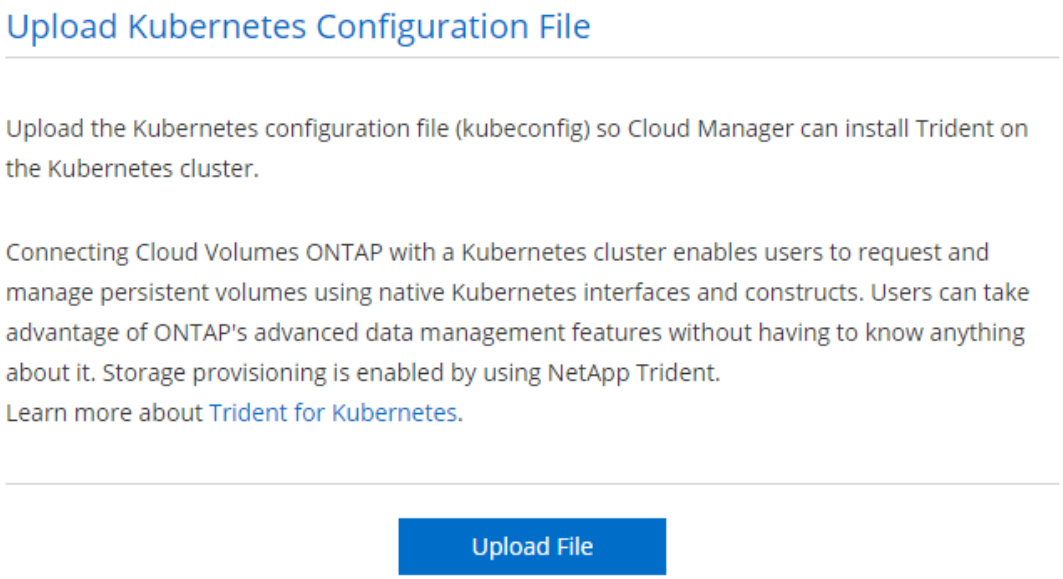
Konfiguration Von Cloud Deployment Kubernetes

Über NetApp Cloud Manager können Sie die Verbindung zum Iguazio Kubernetes Cluster definieren. Für Trident ist Zugriff auf mehrere Ressourcen im Cluster erforderlich, damit das Volume verfügbar wird.

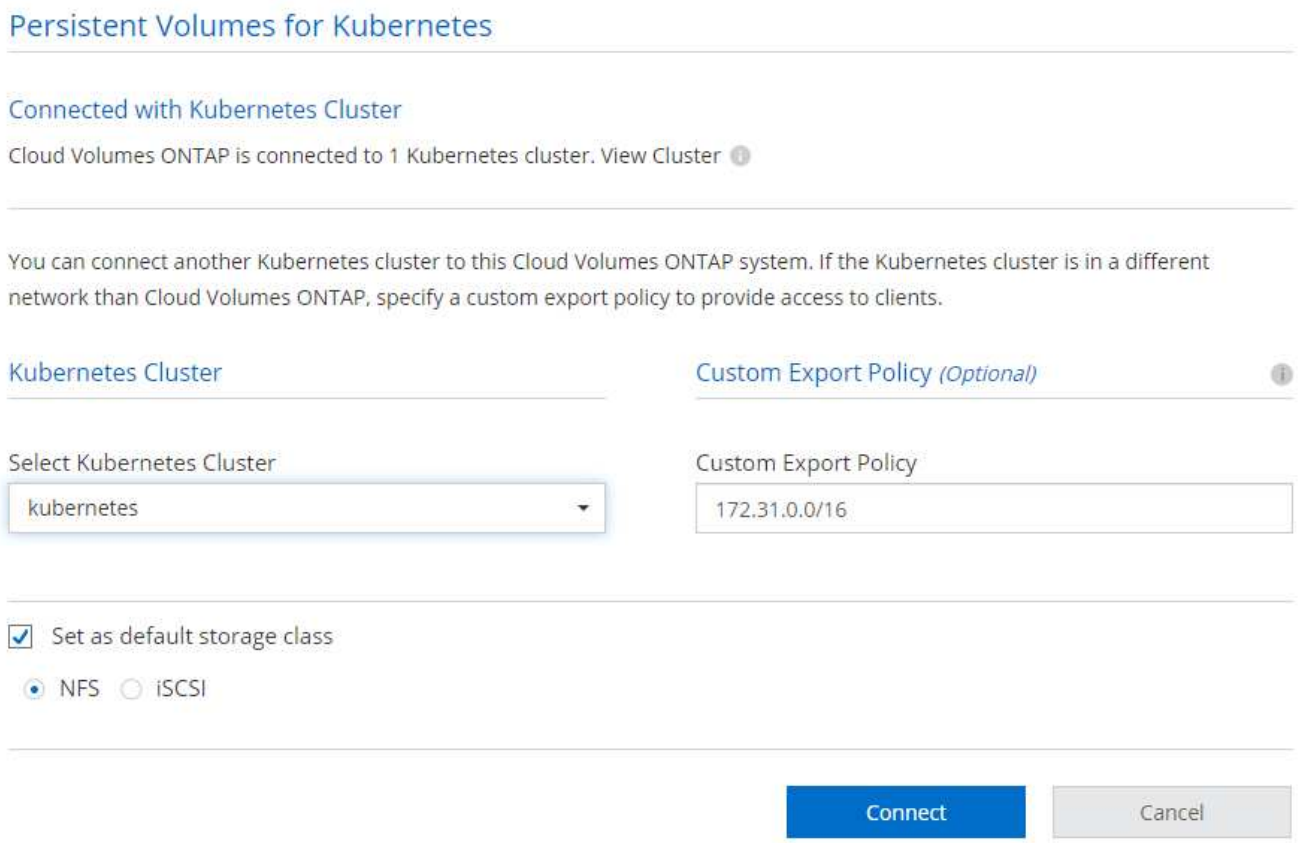
1. Um den Zugriff zu aktivieren, rufen Sie die Kubernetes-Konfigurationsdatei von einem der Iguazio-Knoten ab. Die Datei befindet sich unter `/home/Iguazio/.kube/config`. Laden Sie diese Datei auf Ihren Desktop herunter.
2. Gehen Sie zu Discover Cluster, um zu konfigurieren.



3. Laden Sie die Kubernetes-Konfigurationsdatei hoch. Siehe folgendes Bild.



4. Trident implementieren und ein Volume dem Cluster zuweisen. Sehen Sie das folgende Bild zum Definieren und Zuweisen eines Persistent Volumes zum Iguazio-Cluster.dieser Prozess erstellt ein Persistent Volume (PV) im Iguazio-Cluster. Bevor Sie ihn verwenden können, müssen Sie einen Persistent Volume Claim (PVC) definieren.



On-Premises-Implementierung Der Kubernetes-Konfiguration

Eine lokale Installation von NetApp Trident finden Sie unter "TR-4798" Entsprechende Details. Nach der Konfiguration des Kubernetes-Clusters und der Installation von NetApp Trident können Sie Trident mit dem Iguazio Cluster verbinden, um NetApp Datenmanagementfunktionen zu aktivieren, wie beispielsweise die Erstellung von Snapshot Kopien Ihrer Daten und Modelle.

Persistent Volume Claim Definieren

1. Speichern Sie die folgende YAML in einer Datei, um ein PVC vom Typ Basic zu erstellen.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: basic
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
  storageClassName: netapp-file
```

2. Wenden Sie die YAML-Datei auf Ihr Iguazio Kubernetes-Cluster an.

```
Kubectl -n default-tenant apply -f <your yaml file>
```

Weisen Sie NetApp Volume dem Jupyter Notebook zu

Iguazio bietet verschiedene Managed Services, um Data Scientists einen vollständigen End-to-End-Stack für die Entwicklung und Implementierung von KI/ML-Applikationen bereitzustellen. Weitere Informationen zu diesen Komponenten finden Sie unter "Iguazio Übersicht über Anwendungsdienste und -Tools".

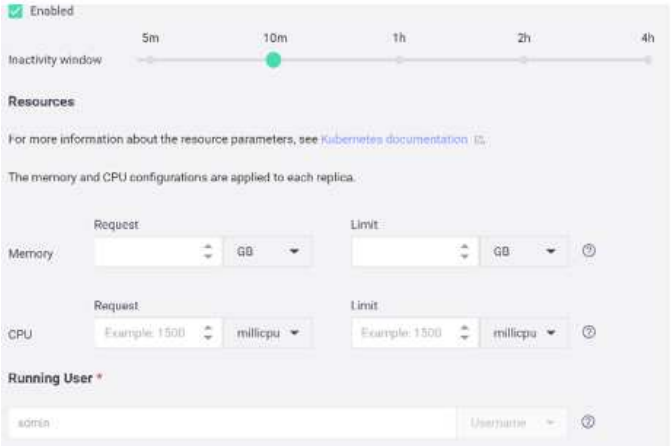
Einer der Managed Services ist Jupyter Notebook. Jeder Entwickler erhält seinen eigenen Einsatz eines Notebook-Containers mit den nötigen Ressourcen für die Entwicklung. Um ihnen Zugriff auf das NetApp Cloud Volume zu geben, können Sie das Volume ihrem Container zuweisen und die Ressourcenzuweisung, den laufenden Benutzer sowie die Einstellungen für Umgebungsvariablen für Persistent Volume Claims nutzen. Das folgende Bild ist dargestellt:

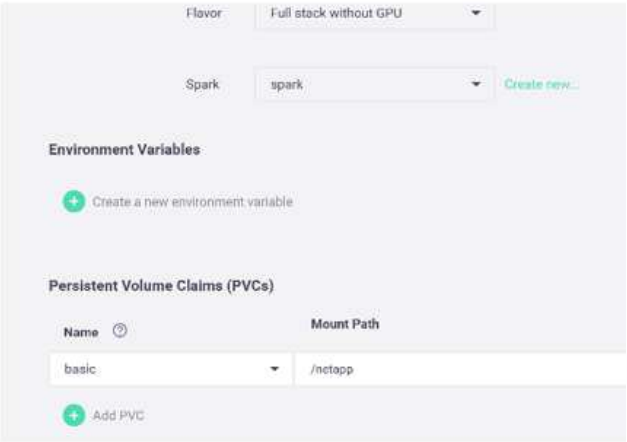
Informationen zu einer On-Premises-Konfiguration finden Sie unter "TR-4798" Bei dem Setup von Trident sind die Datenmanagementfunktionen von NetApp ONTAP aktiviert, wie z. B. die Erstellung von Snapshot Kopien Ihrer Daten oder des Modells zur Versionierung. Fügen Sie in Ihrer Trident Back-End-Konfigurationsdatei die folgende Zeile ein, um Snapshot Verzeichnisse sichtbar zu machen:

```
{
    …
    "defaults": {
        "snapshotDir": "true"
    }
}
```

Sie müssen eine Back-End-Konfigurationsdatei von Trident im JSON-Format erstellen und dann Folgendes ausführen "Trident-Befehl" So verwenden Sie sie:

```
tridentctl create backend -f <backend-file>
```





Bereitstellen der Anwendung

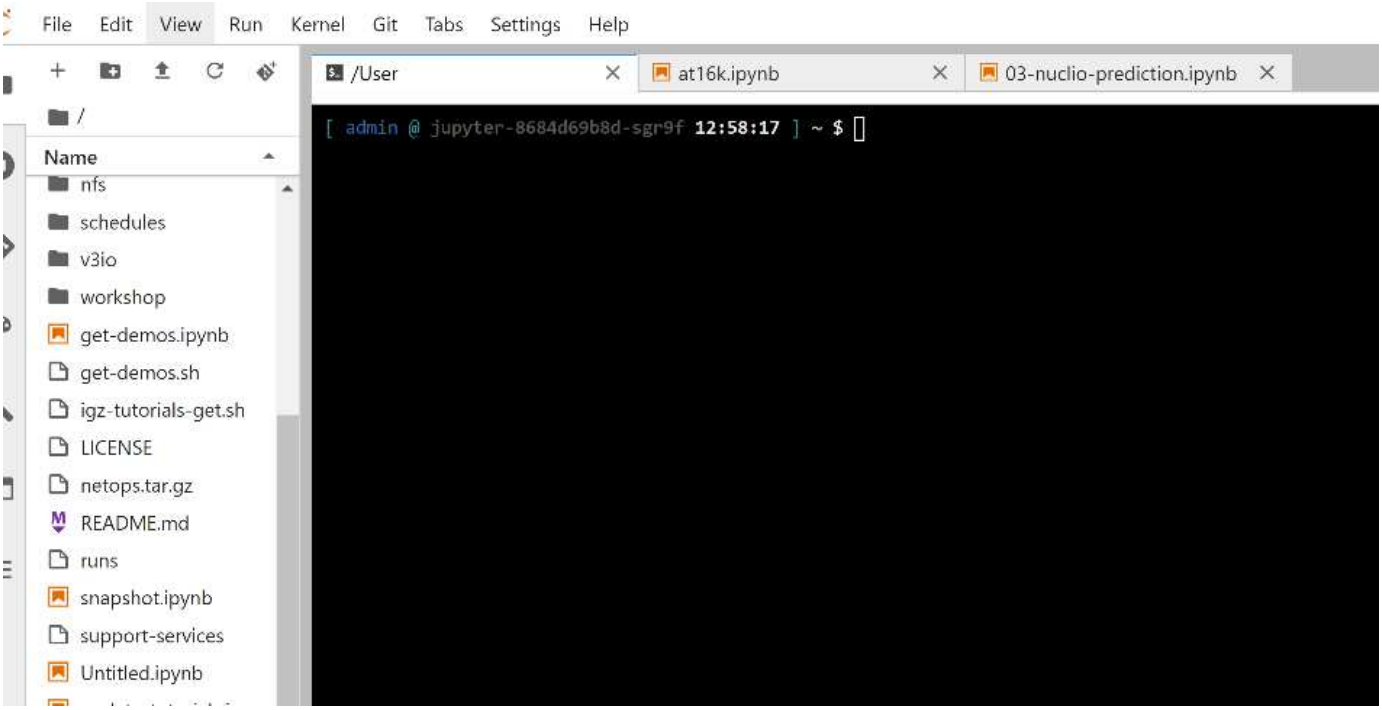
In den folgenden Abschnitten wird die Installation und Bereitstellung der Anwendung beschrieben.

Code von GitHub

Nachdem das NetApp Cloud Volume oder NetApp Trident Volume nun für den Iguazio Cluster und die Entwicklungsumgebung zur Verfügung steht, können Sie nun mit der Überprüfung der Applikation beginnen.

Benutzer haben ihren eigenen Arbeitsbereich (Verzeichnis). Auf jedem Notebook befindet sich der Pfad zum Benutzerverzeichnis `/User`. Die Iguazio-Plattform verwaltet das Verzeichnis. Wenn Sie die obigen Anweisungen befolgen, steht das NetApp Cloud Volume im zur Verfügung `/netapp` Verzeichnis.

Holen Sie sich den Code von GitHub mit einem Jupyter Terminal.



Klont das Projekt in der Jupyter-Terminalaufforderung.

```
cd /User
git clone .
```

Sie sollten jetzt sehen, die `netops- netapp` Ordner in der Dateistruktur im Jupyter-Arbeitsbereich.

Konfigurieren Sie Die Arbeitsumgebung

Kopieren Sie die `Notebook set_env-Example.ipynb` Als `set_env.ipynb`. Öffnen und bearbeiten `set_env.ipynb`. Dieses Notizbuch legt Variablen für Anmeldeinformationen, Dateipositionen und Ausführungstreiber fest.

Wenn Sie die obigen Anweisungen befolgen, sind die folgenden Schritte die einzigen Änderungen, die vorgenommen werden müssen:

1. Erhalten Sie diesen Wert über das Iguazio Services Dashboard: `docker_registry`

   Beispiel: `docker-registry.default-tenant.app.clusterq.iguaziodev.com:80`

2. Ändern `admin` An Ihren Iguazio-Benutzernamen:

   `IGZ_CONTAINER_PATH = '/users/admin'`

   Nachstehend finden Sie Einzelheiten zur Systemverbindung mit ONTAP. Geben Sie den Volume-Namen an, der bei der Installation von Trident generiert wurde. Die folgende Einstellung gilt für ein ONTAP Cluster vor Ort:

   ```
   ontapClusterMgmtHostname = '0.0.0.0'
   ontapClusterAdminUsername = 'USER'
   ontapClusterAdminPassword = 'PASSWORD'
   sourceVolumeName = 'SOURCE VOLUME'
   ```

   Die folgende Einstellung gilt für Cloud Volumes ONTAP:

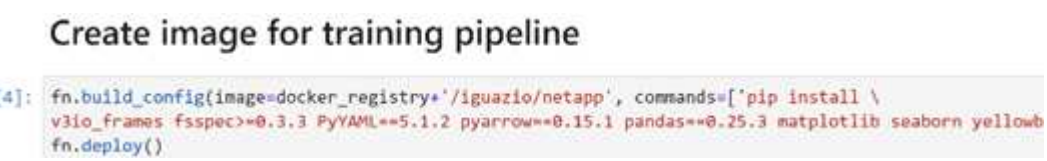
```
MANAGER=ontapClusterMgmtHostname
svm='svm'
email='email'
password=ontapClusterAdminPassword
weid="weid"
volume=sourceVolumeName
```

Basis-Docker-Images Erstellen

Alles, was Sie zum Aufbau einer ML-Pipeline benötigen, ist in der Iguazio-Plattform enthalten. Der Entwickler kann die Spezifikationen der Docker-Images festlegen, die für die Ausführung der Pipeline erforderlich sind, und die Image-Erstellung aus Jupyter Notebook ausführen. Öffnen Sie das Notebook `create-images.ipynb` Und alle Zellen ausführen.

Dieses Notizbuch erstellt zwei Bilder, die wir in der Pipeline verwenden.

- `iguazio/netapp.` Zum behandeln VON ML-Aufgaben.



- `netapp/pipeline`. Enthält Dienstprogramme zur Bearbeitung von NetApp Snapshot Kopien.

Create image for Ontap utilitites

```
[0]: fn.build_config(image=docker_registry + '/netapp/pipeline:latest',commands=['apt -y update','pip install v3io_frames netapp_ontap'
     fn.deploy()
```

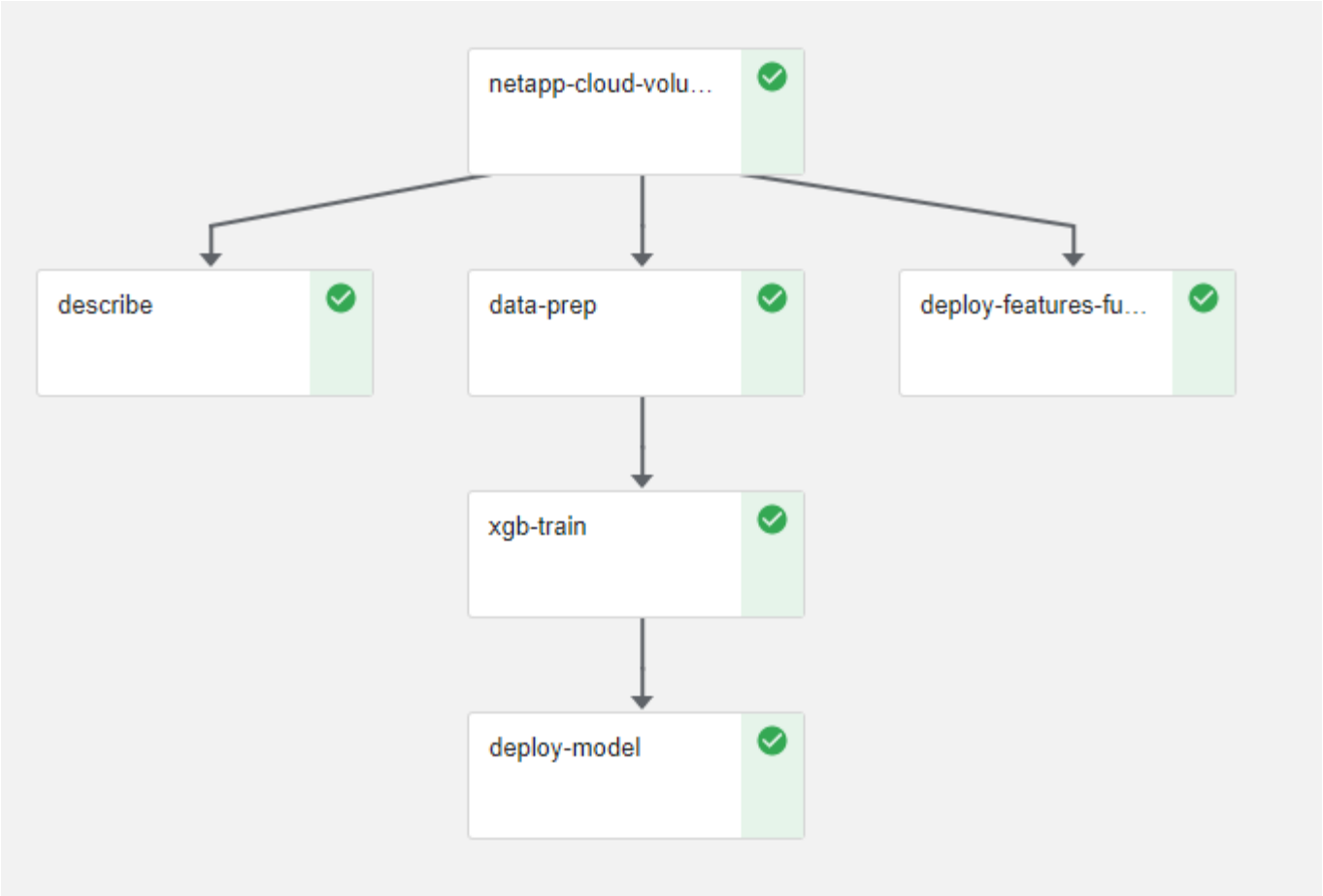
Überprüfen Sie Einzelne Jupyter Notebooks

In der folgenden Tabelle sind die Bibliotheken und Frameworks aufgeführt, mit denen wir diese Aufgabe erstellt haben. All diese Komponenten wurden vollständig in die rollenbasierten Zugriffs- und Sicherheitskontrollen von Iguazio integriert.

Bibliotheken/Framework	Beschreibung
MLRun	Ein von Iguazio verwaltet, um die Montage, Ausführung und Überwachung einer ML/AI-Pipeline zu ermöglichen.
Nuclio	Ein serverloses Funktionsgerüst, das mit Iguazio integriert ist. Auch als Open-Source-Projekt von Iguazio verfügbar.
Kubeflow	Kubernetes-basiertes Framework für die Implementierung der Pipeline Dies ist auch ein Open-Source-Projekt, zu dem Iguazio beiträgt. Es ist in Iguazio integriert für zusätzliche Sicherheit und Integration in den Rest der Infrastruktur.
Docker	Eine Docker-Registry wird als Service in der Iguazio-Plattform ausgeführt. Sie können dies auch ändern, um eine Verbindung zu Ihrer Registrierung herzustellen.
NetApp Cloud Volumes	Cloud Volumes auf AWS ermöglichen uns Zugriff auf große Datenmengen und die Möglichkeit, Snapshot Kopien mit den für das Training verwendeten Datensätzen zu erstellen.
Trident	Trident ist ein Open-Source-Projekt, das von NetApp gemanagt wird. Sie vereinfacht die Integration in Storage- und Computing-Ressourcen in Kubernetes.

Für die Erstellung der ML-Pipeline haben wir mehrere Notebooks verwendet. Jedes Notebook kann einzeln getestet werden, bevor es in die Pipeline gebracht wird. Wir decken jedes Notebook individuell nach dem Bereitstellungsablauf dieser Demonstrationsanwendung ab.

Das gewünschte Ergebnis ist eine Pipeline, mit der ein Modell auf Basis einer Snapshot-Kopie der Daten trainiert und das Modell zur Inferenz implementiert wird. Im folgenden Bild ist ein Blockdiagramm einer abgeschlossenen MLRun-Pipeline dargestellt.



Implementierung Der Funktion Zur Datengenerierung

In diesem Abschnitt wird beschrieben, wie wir Nuclio serverlose Funktionen zur Generierung von Netzwerkgerätedaten verwendet haben. Der Anwendungsfall wird von einem Iguazio-Client aus angepasst, der die Pipeline implementierte und Iguazio-Services zur Überwachung und Vorhersage von Fehlern bei Netzwerkgeräten verwendete.

Wir haben Daten von Netzwerkgeräten simuliert. Ausführen des Jupyter-Notebooks `data-generator.ipynb` Erstellt eine serverlose Funktion, die alle 10 Minuten läuft und eine Parkett-Datei mit neuen Daten erzeugt. Um die Funktion bereitzustellen, führen Sie alle Zellen in diesem Notebook aus. Siehe "Nuclio Website" Um unbekannte Komponenten in diesem Notebook zu überprüfen.

Beim Generieren der Funktion wird eine Zelle mit dem folgenden Kommentar ignoriert. Jede Zelle im Notebook wird als Teil der Funktion angenommen. Importieren Sie das Nuclio-Modul, um es zu aktivieren `%nuclio magic`.

```
# nuclio: ignore
import nuclio
```

In der Spezifikation für die Funktion definieren wir die Umgebung, in der die Funktion ausgeführt wird, wie sie ausgelöst wird und welche Ressourcen sie verbraucht.

```
spec = nuclio.ConfigSpec(config={"spec.triggers.inference.kind":"cron",

"spec.triggers.inference.attributes.interval" :"10m",
                                "spec.readinessTimeoutSeconds" : 60,
                                "spec.minReplicas" : 1},……
```

Der `init_context` Die Funktion wird beim Initialisieren der Funktion vom Nuclio Framework aufgerufen.

```
def init_context(context):
    ….
```

Jeder Code, der nicht in einer Funktion enthalten ist, wird aufgerufen, wenn die Funktion initialisiert wird. Wenn Sie ihn aufrufen, wird eine Handler-Funktion ausgeführt. Sie können den Namen des Handlers ändern und in der Funktionungsspezifikation angeben.

```
def handler(context, event):
            …
```

Sie können die Funktion vor der Bereitstellung vom Notebook aus testen.

```
%%time
# nuclio: ignore
init_context(context)
event = nuclio.Event(body='')
output = handler(context, event)
output
```

Die Funktion kann über das Notebook bereitgestellt oder über eine CI/CD-Pipeline (Anpassung dieses Codes) bereitgestellt werden.

```
addr = nuclio.deploy_file(name='generator',project='netops',spec=spec,
tag='v1.1')
```

Pipeline-Notebooks

Diese Notizbücher sollen für dieses Setup nicht einzeln ausgeführt werden. Dies ist nur ein Bericht über jedes Notizbuch. Wir haben sie als Teil der Pipeline aufgerufen. Um sie einzeln auszuführen, lesen Sie die MLRun Dokumentation durch, um sie als Kubernetes Jobs auszuführen.

Snap_cv.ipynb

Dieses Notebook wickelt die Snapshot-Kopien des Cloud Volumes am Anfang der Pipeline ab. Er übergibt den Namen des Volumes an den Pipeline-Kontext. Dieses Notebook stößt ein Shell-Skript an, um die Snapshot

Kopie zu bearbeiten. Während der Ausführung in der Pipeline enthält der Ausführungskontext Variablen, um alle Dateien zu finden, die für die Ausführung benötigt werden. Beim Schreiben dieses Codes muss sich der Entwickler nicht um den Speicherort der Datei im Container kümmern, der ihn ausführt. Wie später beschrieben, wird diese Applikation mit allen Abhängigkeiten implementiert und ist die Definition der Pipeline-Parameter, die den Kontext der Ausführung liefern.

```
command = os.path.join(context.get_param('APP_DIR'),"snap_cv.sh")
```

Der Speicherort der erstellten Snapshot-Kopie wird im MLRun-Kontext platziert, der von Schritten in der Pipeline verwendet werden soll.

```
context.log_result('snapVolumeDetails',snap_path)
```

Die nächsten drei Notebooks werden parallel ausgeführt.

Data-prep.ipynb

RAW-Metriken müssen in Funktionen für das Modelltraining umgewandelt werden. Dieses Notebook liest die RAW-Kennzahlen aus dem Snapshot Verzeichnis und schreibt die Funktionen für das Modelltraining auf das NetApp Volume.

Bei Ausführung im Kontext der Pipeline ist der Input `DATA_DIR` Enthält den Speicherort der Snapshot Kopie.

```
metrics_table = os.path.join(str(mlruncontext.get_input('DATA_DIR',
os.getenv('DATA_DIR','/netpp'))),
                            mlruncontext.get_param('metrics_table',
os.getenv('metrics_table','netops_metrics_parquet')))
```

Beschreiben.ipynb

Um die eingehenden Kennzahlen zu visualisieren, stellen wir einen Pipeline-Schritt bereit, der über die Kubeflow und MLRun UIs verfügbare Plots und Diagramme bereitstellt. Jede Ausführung hat eine eigene Version dieses Visualisierungstools.

```
ax.set_title("features correlation")
plt.savefig(os.path.join(base_path, "plots/corr.png"))
context.log_artifact(PlotArtifact("correlation",  body=plt.gcf()),
local_path="plots/corr.html")
```

Deploy-Feature-function.ipynb

Wir überwachen kontinuierlich die Metriken, die nach Anomalien suchen. Dieses Notizbuch erstellt eine serverlose Funktion, die die Funktionen generiert, die für die Vorhersage von eingehenden Metriken erforderlich sind. Dieses Notizbuch ruft die Erstellung der Funktion auf. Der Funktionscode befindet sich im Notebook `data- prep.ipynb`. Beachten Sie, dass wir für diesen Zweck dasselbe Notebook wie einen Schritt in der Pipeline verwenden.

Training.ipynb

Nachdem wir die Funktionen erstellt haben, lösen wir das Modelltraining aus. Das Ergebnis dieses Schritts ist das Modell, das zur Inferenz verwendet wird. Wir sammeln auch Statistiken, um jede Ausführung (Experiment) im Auge zu behalten.

Beispielsweise gibt der folgende Befehl die Genauigkeitbewertung in den Kontext des entsprechenden Experiments ein. Dieser Wert ist in Kubeflow und MLRun sichtbar.

```
context.log_result('accuracy',score)
```

Bereitstellen-Inferenz-Funktion.ipynb

Der letzte Schritt in der Pipeline ist die Implementierung des Modells als serverlose Funktion für kontinuierliche Inferenz. Dieses Notebook ruft die Erstellung der serverlosen Funktion auf, die in definiert ist `nuclio-inference- function.ipynb`.

Pipeline prüfen und aufbauen

Die Kombination der Ausführung aller Notebooks in einer Pipeline ermöglicht es, kontinuierlich Experimente durchzuführen, um die Genauigkeit des Modells anhand neuer Metriken neu zu bewerten. Öffnen Sie zuerst das `pipeline.ipynb` Notebook. Wir erläutern Ihnen die Details, die zeigen, wie NetApp und Iguazio die Implementierung dieser ML-Pipeline vereinfachen.

Wir verwenden MLRun, um jedem Schritt der Pipeline Kontext bereitzustellen und die Ressourcenzuordnung zu bearbeiten. Der MLRun API-Service wird auf der Iguazio-Plattform ausgeführt und dient als Interaktionsstelle mit Kubernetes-Ressourcen. Jeder Entwickler kann keine Ressourcen direkt anfordern. Die API verarbeitet die Anforderungen und ermöglicht Zugriffskontrollen.

```
# MLRun API connection definition
mlconf.dbpath = 'http://mlrun-api:8080'
```

Diese Pipeline kann mit NetApp Cloud Volumes und On-Premises-Volumes eingesetzt werden. Wir haben diese Demo für die Verwendung von Cloud Volumes erstellt. Im Code sehen Sie jedoch die Option zur Ausführung vor Ort.

```
# Initialize the NetApp snap fucntion once for all functions in a notebook
if [ NETAPP_CLOUD_VOLUME ]:
    snapfn =
code_to_function('snap',project='NetApp',kind='job',filename="snap_cv.ipyn
b").apply(mount_v3io())
    snap_params = {
    "metrics_table" : metrics_table,
    "NETAPP_MOUNT_PATH" : NETAPP_MOUNT_PATH,
    'MANAGER' : MANAGER,
    'svm' : svm,
    'email': email,
    'password': password ,
    'weid': weid,
    'volume': volume,
    "APP_DIR" : APP_DIR
       }
else:
    snapfn =
code_to_function('snap',project='NetApp',kind='job',filename="snapshot.ipy
nb").apply(mount_v3io())
….
snapfn.spec.image = docker_registry + '/netapp/pipeline:latest'
snapfn.spec.volume_mounts =
[snapfn.spec.volume_mounts[0],netapp_volume_mounts]
      snapfn.spec.volumes = [ snapfn.spec.volumes[0],netapp_volumes]
```

Die erste Aktion, die erforderlich ist, um ein Jupyter-Notebook in einen Kubeflow-Schritt zu verwandeln, ist, den Code in eine Funktion zu verwandeln. Eine Funktion verfügt über alle Spezifikationen, die zum Ausführen dieses Notebooks erforderlich sind. Wenn Sie das Notebook nach unten scrollen, sehen Sie, dass wir für jeden Schritt in der Pipeline eine Funktion definieren.

Teil des Notebooks	Beschreibung
<Code_to_Function> (Teil des MLRun-Moduls)	Name der Funktion: Projektname. Dient zur Organisation aller Projektartefakte. Dies ist in der MLRun UI sichtbar. Freundlich. In diesem Fall ist Kubernetes-Job. Dies könnte DASK, mpi, funkk8s und mehr sein. Weitere Informationen finden Sie in der MLRun-Dokumentation. Datei: Der Name des Notebooks. Dies kann auch ein Speicherort in Git (HTTP) sein.
Bild	Der Name des Docker Images, das wir für diesen Schritt verwenden. Das haben wir früher mit dem Notebook create-image.ipynb erstellt.
Volume_Mounts und Volumes	Einzelheiten zum Mounten des NetApp Cloud Volume zur Laufzeit.

Außerdem definieren wir Parameter für die Schritte.

```
params={   "FEATURES_TABLE":FEATURES_TABLE,
           "SAVE_TO" : SAVE_TO,
           "metrics_table" : metrics_table,
           'FROM_TSDB': 0,
           'PREDICTIONS_TABLE': PREDICTIONS_TABLE,
           'TRAIN_ON_LAST': '1d',
           'TRAIN_SIZE':0.7,
           'NUMBER_OF_SHARDS' : 4,
           'MODEL_FILENAME' : 'netops.v3.model.pickle',
           'APP_DIR' : APP_DIR,
           'FUNCTION_NAME' : 'netops-inference',
           'PROJECT_NAME' : 'netops',
           'NETAPP_SIM' : NETAPP_SIM,
           'NETAPP_MOUNT_PATH': NETAPP_MOUNT_PATH,
           'NETAPP_PVC_CLAIM' : NETAPP_PVC_CLAIM,
           'IGZ_CONTAINER_PATH' : IGZ_CONTAINER_PATH,
           'IGZ_MOUNT_PATH' : IGZ_MOUNT_PATH
            }
```

Nachdem Sie die Funktionsdefinition für alle Schritte erstellt haben, können Sie die Pipeline erstellen. Wir verwenden den `kfp` Modul, um diese Definition zu erstellen. Der Unterschied zwischen der Verwendung von MLRun und dem Selbstaufbau besteht in der Vereinfachung und Verkürzung der Codierung.

Die von uns definierten Funktionen werden mit dem in Schrittkomponenten umgewandelt `as_step` Funktion von MLRun.

Definition Von Snapshot-Schritten

Initiieren einer Snapshot-Funktion, -Ausgabe und Mounten von v3io als Quelle:

```
snap = snapfn.as_step(NewTask(handler='handler',params=snap_params),
name='NetApp_Cloud_Volume_Snapshot',outputs=['snapVolumeDetails','training
_parquet_file']).apply(mount_v3io())
```

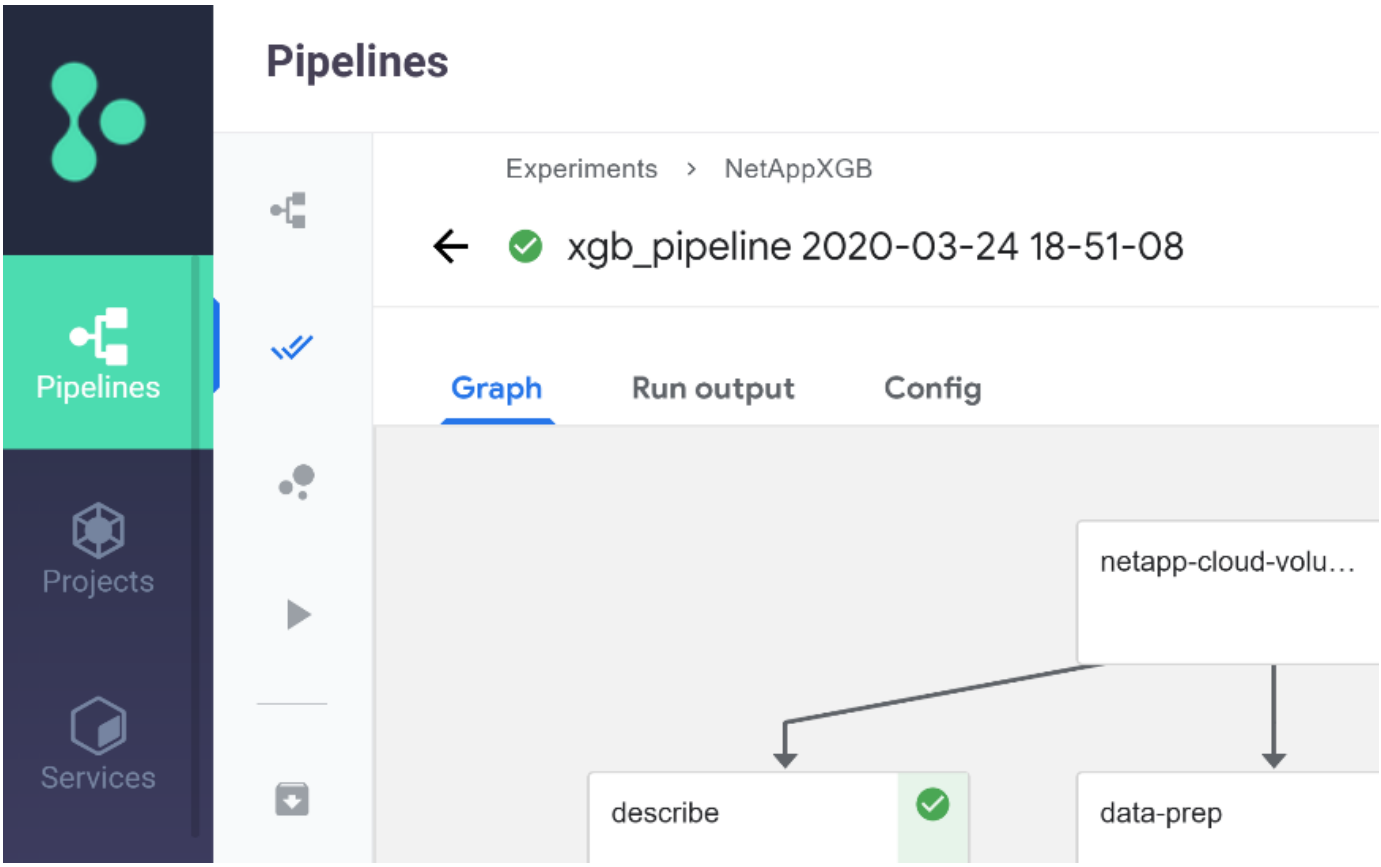
Parameter	Details
NewTask	NewTask ist die Definition der Funktion Run.
(MLRun-Modul)	Handler. Name der zu aufrufe Python-Funktion. Wir haben den Namen Handler im Notebook verwendet, aber er ist nicht erforderlich. Param. Die Parameter, die wir an die Ausführung übergeben haben. Innerhalb unseres Codes verwenden wir context.get_param (‘PARAMETER’), um die Werte zu erhalten.

Parameter	Details
As_Step	Name: Name des Kubeflow-Pipeline-Schritts. Ausgänge. Dies sind die Werte, die der Schritt dem Wörterbuch nach Abschluss hinzugefügt hat. Werfen Sie einen Blick auf das Snap_cv.ipynb Notebook. Mount_v3io(). Hiermit wird der Schritt zum Mounten von /Benutzer für den Benutzer konfiguriert, der die Pipeline ausführt.

```
prep = data_prep.as_step(name='data-prep',
handler='handler',params=params,
                          inputs = {'DATA_DIR':
snap.outputs['snapVolumeDetails']} ,

out_path=artifacts_path).apply(mount_v3io()).after(snap)
```

Parameter	Details
Eingänge	Sie können die Ausgänge eines vorherigen Schritts an einen Schritt übergeben. In diesem Fall ist Snap.Outputs ['snapVolumeDetails'] der Name der Snapshot Kopie, die wir im Snap Schritt erstellt haben.
Out_PATH	Ein Speicherort für Artefakte, die mithilfe des MLRun-Moduls log_Artefakte erzeugt werden.

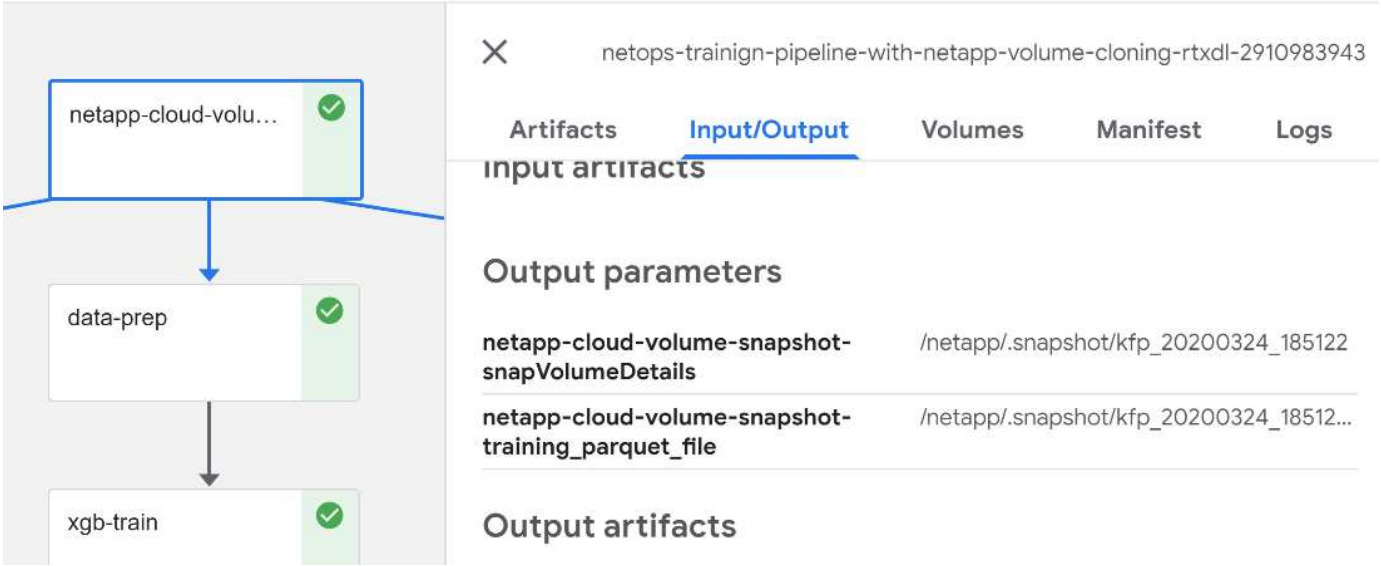
Sie können laufen `pipeline.ipynb` Von oben nach unten. Anschließend können Sie im Iguazio-Dashboard zur Registerkarte Pipelines wechseln, um den Fortschritt zu überwachen, wie auf der Registerkarte Iguazio-Dashboard-Pipelines zu sehen ist.



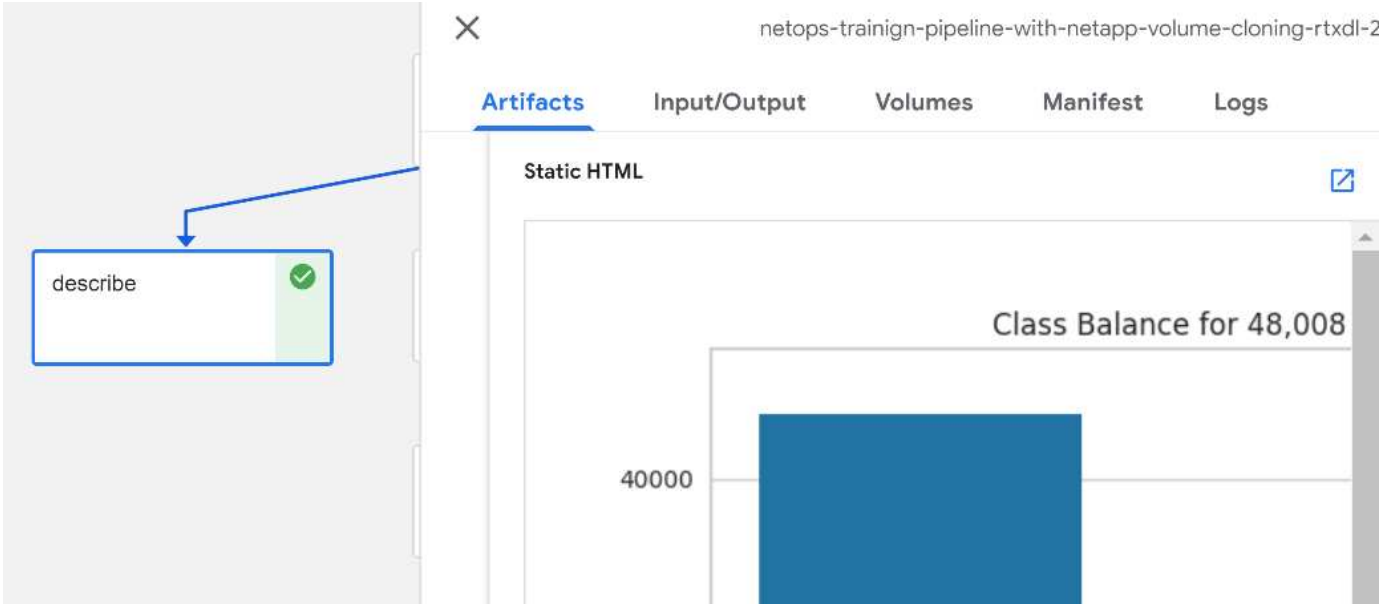
Da wir die Genauigkeit des Trainingsschritts in jedem Lauf protokolliert haben, haben wir für jedes Experiment eine Aufzeichnung der Genauigkeit, wie in der Aufzeichnung der Trainingsgenauigkeit zu sehen ist.

Run name	Status	Duration	Pipeline Version	Recurring ...	Start time	accuracy
xgb_pipeline 2020-03-24 18-51-...	✓	0:08:43	[View pipeline]	-	3/24/2020, 2:51:09 PM	0.985
xgb_pipeline 2020-03-19 13-31-...	✓	0:08:14	[View pipeline]	-	3/19/2020, 9:31:19 AM	0.980
xgb_pipeline 2020-03-18 12-56-...	✓	0:08:11	[View pipeline]	-	3/18/2020, 8:56:08 AM	0.990
xgb_pipeline 2020-03-17 19-49-...	✓	0:08:03	[View pipeline]	-	3/17/2020, 3:49:31 PM	0.985
xgb_pipeline 2020-03-17 18-34-...	✓	0:05:54	[View pipeline]	-	3/17/2020, 2:34:56 PM	0.980
xgb_pipeline 2020-03-17 17-34-...	✓	0:04:48	[View pipeline]	-	3/17/2020, 1:34:16 PM	0.982
xgb_pipeline 2020-03-17 17-01-...	✓	0:05:25	[View pipeline]	-	3/17/2020, 1:01:58 PM	0.987
xgb_pipeline 2020-03-16 16-47-...	✓	0:06:08	[View pipeline]	-	3/16/2020, 12:47:19 ...	0.983
xgb_pipeline 2020-03-16 13-57-...	✓	0:05:18	[View pipeline]	-	3/16/2020, 9:57:03 AM	0.980

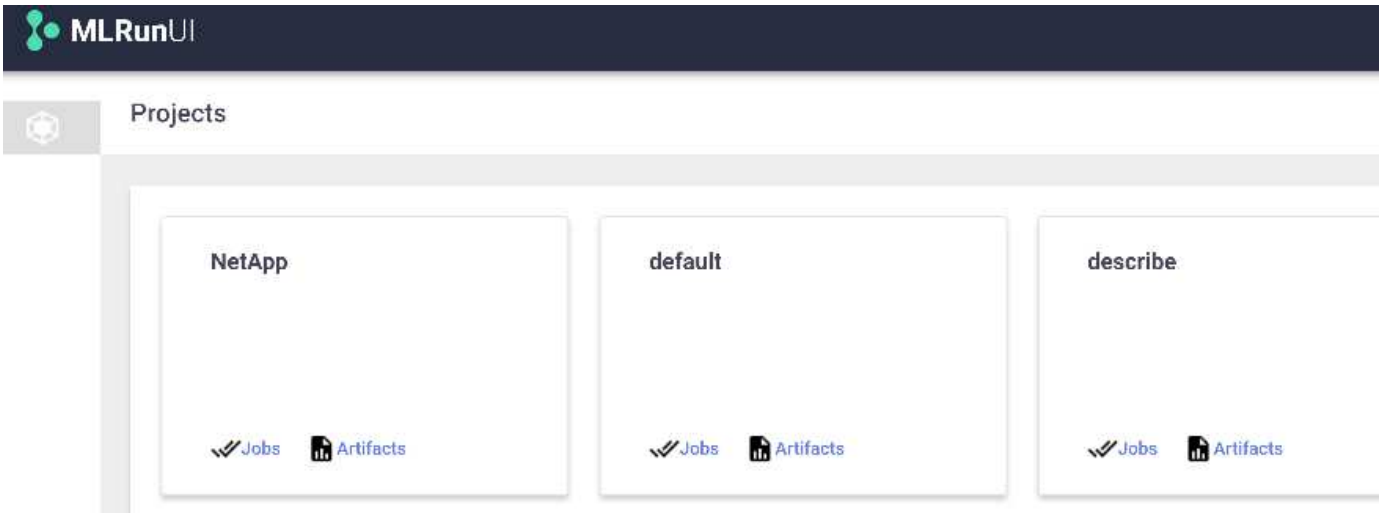
Wenn Sie den Schritt Snapshot auswählen, wird der Name der Snapshot Kopie angezeigt, die zur Ausführung dieses Experiments verwendet wurde.



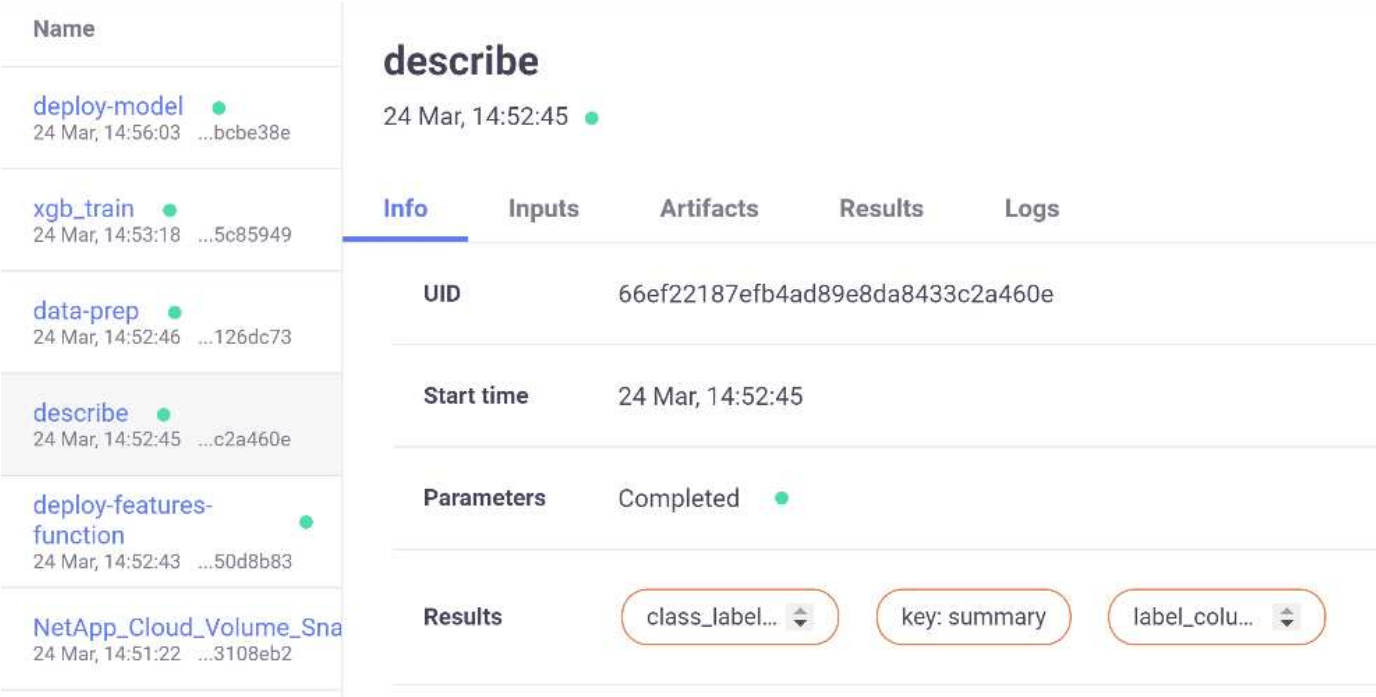
Der beschriebene Schritt hat visuelle Artefakte, um die von uns verwendeten Metriken zu untersuchen. Sie können erweitern, um die vollständige Darstellung wie im folgenden Bild zu sehen.



Die MLRun API-Datenbank verfolgt auch Eingänge, Ausgänge und Artefakte für jeden nach Projekt organisierten Durchlauf. Ein Beispiel für Eingänge, Ausgänge und Artefakte für jeden Durchlauf ist im folgenden Bild zu sehen.



Für jede Aufgabe werden zusätzliche Details gespeichert.



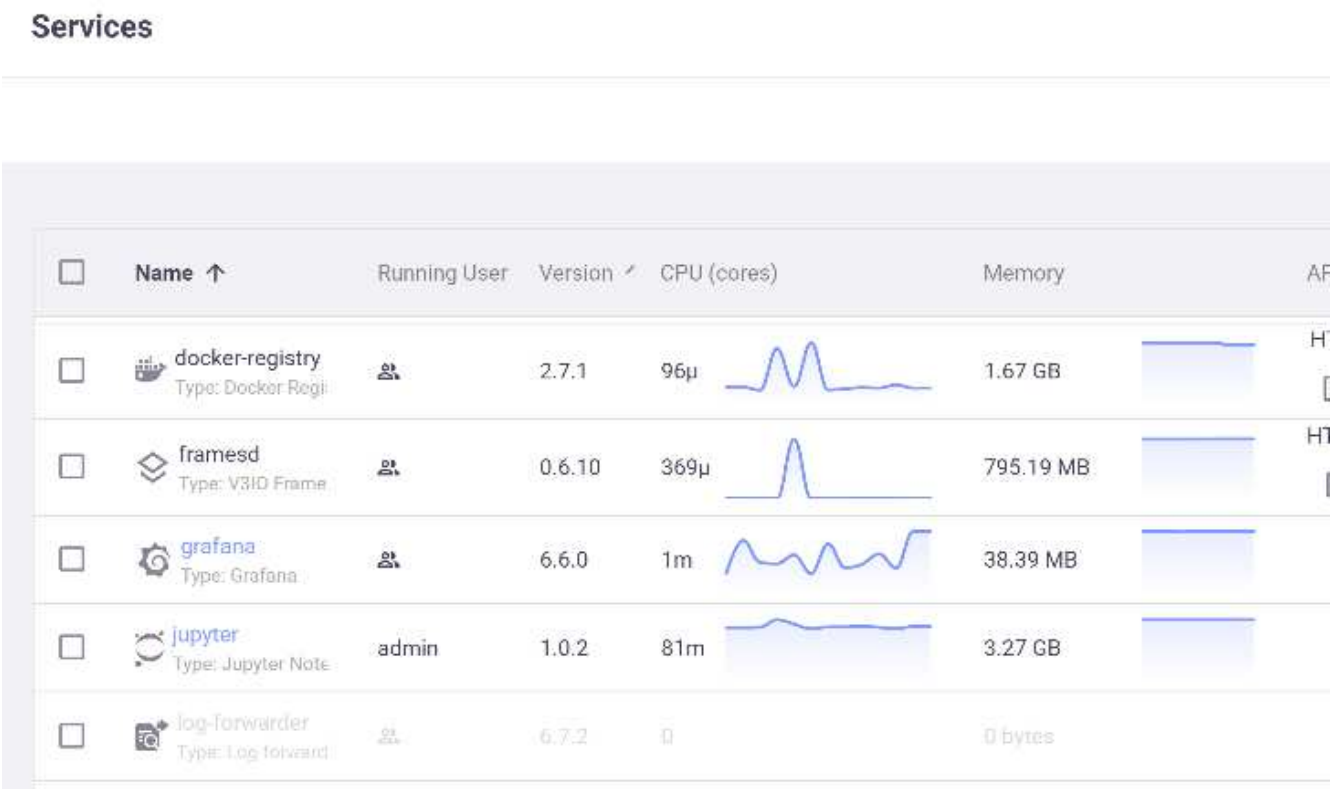
Es gibt mehr Informationen über MLRun, als wir in diesem Dokument abdecken können. AI Artefakte, einschließlich der Definition der Schritte und Funktionen, können in der API-Datenbank gespeichert, versioniert und einzeln oder als volles Projekt aufgerufen werden. Projekte können auch gespeichert und zur späteren Verwendung an Git geschoben werden. Wir empfehlen Ihnen, weitere Informationen im zu erhalten "MLRun GitHub-Website".

Implementierung Von Grafana Dashboard

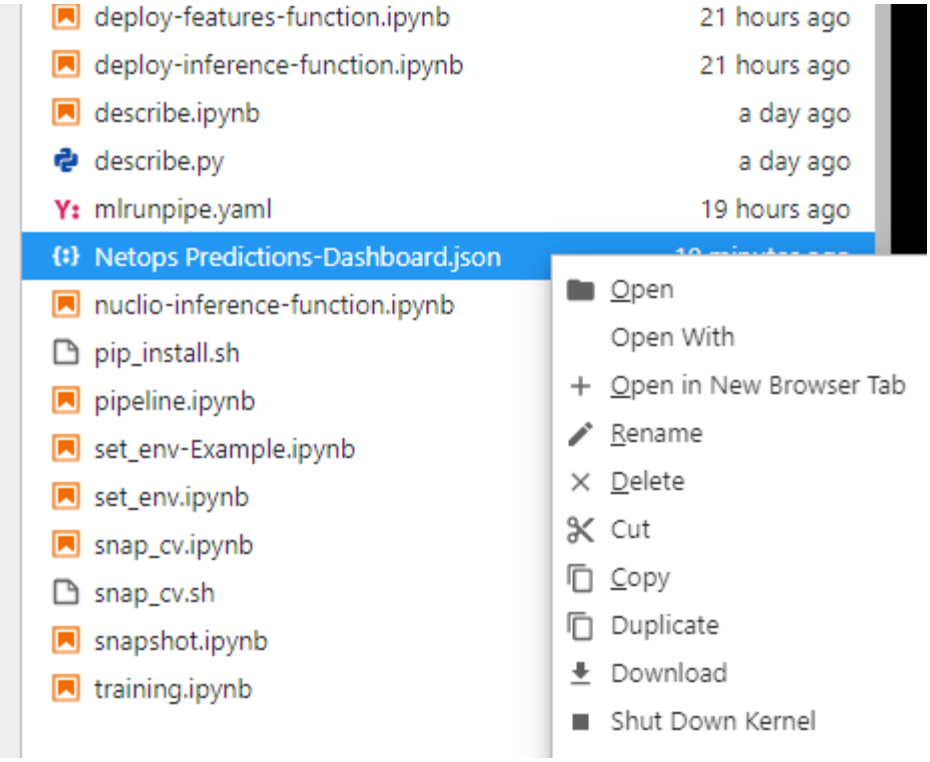
Wenn alles bereitgestellt ist, kommen wir Rückschlüsse auf neue Daten. Die Modelle prognostizieren den Ausfall der Netzwerkgeräte. Die Ergebnisse der Vorhersage werden in einer Iguazio-Timeseries-Tabelle gespeichert. Sie können die Ergebnisse mit Grafana in der in die Sicherheits- und Datenzugriffsrichtlinie von Iguazio integrierten Plattform visualisieren.

Sie können das Dashboard bereitstellen, indem Sie die bereitgestellte JSON-Datei in die Grafana-Schnittstellen im Cluster importieren.

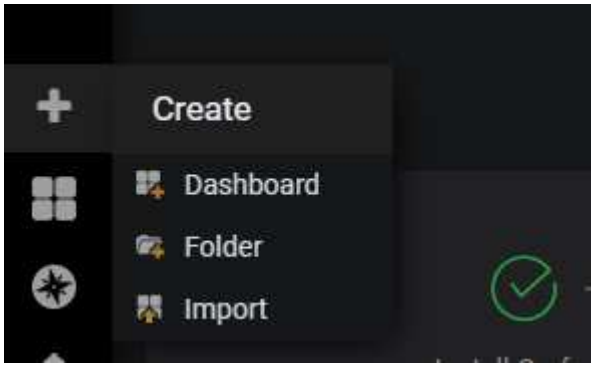
1. Um zu überprüfen, ob der Grafana-Service ausgeführt wird, suchen Sie unter Services.



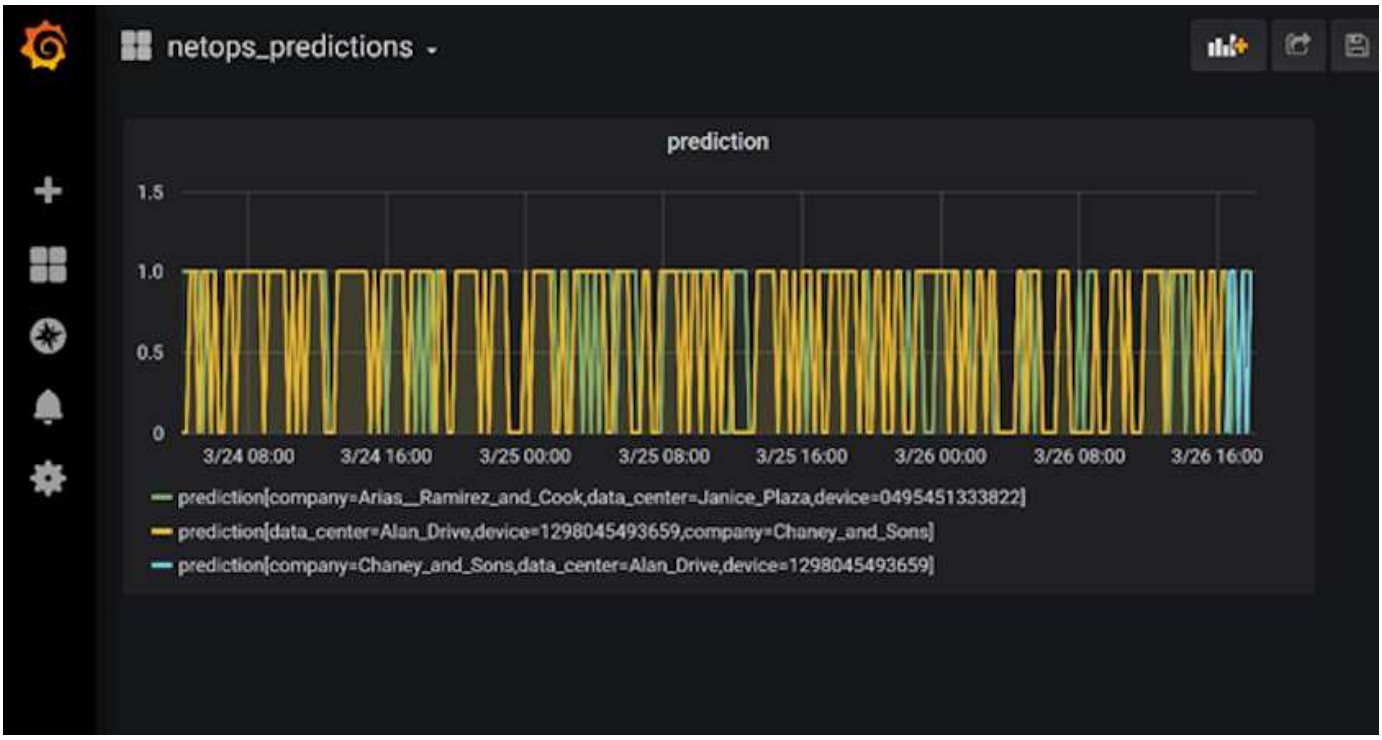
2. Wenn diese nicht vorhanden ist, stellen Sie eine Instanz im Abschnitt „Services“ bereit:
   a. Klicken Sie Auf Neuer Dienst.
   b. Wählen Sie Grafana aus der Liste aus.
   c. Akzeptieren Sie die Standardeinstellungen.
   d. Klicken Sie Auf Weiter.
   e. Geben Sie Ihre Benutzer-ID ein.
   f. Klicken Sie Auf Dienst Speichern.
   g. Klicken Sie oben auf Änderungen übernehmen.
3. Laden Sie die Datei herunter, um das Dashboard bereitzustellen `NetopsPredictions-Dashboard.json` Über die Jupyter-Schnittstelle.



4. Öffnen Sie Grafana im Abschnitt Services, und importieren Sie das Dashboard.



5. Klicken Sie Auf Hochladen `*.json` Datei und wählen Sie die Datei aus, die Sie zuvor heruntergeladen haben (`NetopsPredictions-Dashboard.json`). Das Dashboard wird nach Abschluss des Uploads angezeigt.



Funktion „Bereinigen“ Bereitstellen

Wenn Sie eine Menge Daten generieren, ist es wichtig, die Dinge sauber und organisiert zu halten. Stellen Sie dazu die Bereinigungsfunktion mit dem bereit `cleanup.ipynb` Notebook.

Vorteile

NetApp und Iguazio beschleunigen und vereinfachen die Implementierung von KI- und ML-Applikationen. Dazu bauen sie in wichtigen Frameworks wie Kubeflow, Apache Spark und TensorFlow sowie Orchestrierungs-Tools wie Docker und Kubernetes auf. Durch die Vereinheitlichung der End-to-End-Datenpipeline reduzieren NetApp und Iguazio die Latenz und Komplexität moderner Computing-Workloads, wodurch die Lücke zwischen Entwicklung und Betrieb geschlossen wird. Data Scientists können Abfragen zu großen Datensätzen durchführen und Daten und algorithmische Modelle während der Trainingsphase sicher mit autorisierten Benutzern teilen. Sobald die Container-Modelle für die Produktion bereit sind, können Sie sie ganz einfach von Entwicklungsumgebungen in betriebliche Umgebungen verschieben.

Schlussfolgerung

Bei der Erstellung eigener KI/ML-Pipelines ist die Konfiguration der Integration, Verwaltung, Sicherheit und Zugänglichkeit der Komponenten einer Architektur eine schwierige Aufgabe. Der Zugriff auf und die Kontrolle über ihre Umgebung für Entwickler stellt eine weitere Herausforderung dar.

Die Kombination aus NetApp und Iguazio vereint diese Technologien als Managed Services, um die Technologieeinführung zu beschleunigen und die Markteinführungszeit für neue KI/ML-Applikationen zu verkürzen.

TR-4915: Datenverschiebung mit E-Series und BeeGFS für KI- und Analyse-Workflows

Cody Harryman und Ryan Rodine, NetApp

TR-4915 beschreibt, wie Daten aus einem beliebigen Daten-Repository in ein BeeGFS Filesystem verschoben werden, das durch SAN-Storage der NetApp E-Series unterstützt wird. Bei Applikationen für künstliche Intelligenz (KI) und Machine Learning (ML) müssen Kunden daher möglicherweise routinemäßig große Datensätze von mehr als vielen Petabyte an Daten in ihre BeeGFS Cluster verschieben, um Modelle zu entwickeln. In diesem Dokument erfahren Sie, wie dies mithilfe von NetApp XCP und NetApp BlueXP Kopier- und Synchronisierungstools erreicht werden kann.

"TR-4915: Datenverschiebung mit E-Series und BeeGFS für KI- und Analyse-Workflows"

Copyright-Informationen

Copyright © 2024 NetApp. Alle Rechte vorbehalten. Gedruckt in den USA. Dieses urheberrechtlich geschützte Dokument darf ohne die vorherige schriftliche Genehmigung des Urheberrechtsinhabers in keiner Form und durch keine Mittel – weder grafische noch elektronische oder mechanische, einschließlich Fotokopieren, Aufnehmen oder Speichern in einem elektronischen Abrufsystem – auch nicht in Teilen, vervielfältigt werden.

Software, die von urheberrechtlich geschütztem NetApp Material abgeleitet wird, unterliegt der folgenden Lizenz und dem folgenden Haftungsausschluss:

DIE VORLIEGENDE SOFTWARE WIRD IN DER VORLIEGENDEN FORM VON NETAPP ZUR VERFÜGUNG GESTELLT, D. H. OHNE JEGLICHE EXPLIZITE ODER IMPLIZITE GEWÄHRLEISTUNG, EINSCHLIESSLICH, JEDOCH NICHT BESCHRÄNKT AUF DIE STILLSCHWEIGENDE GEWÄHRLEISTUNG DER MARKTGÄNGIGKEIT UND EIGNUNG FÜR EINEN BESTIMMTEN ZWECK, DIE HIERMIT AUSGESCHLOSSEN WERDEN. NETAPP ÜBERNIMMT KEINERLEI HAFTUNG FÜR DIREKTE, INDIREKTE, ZUFÄLLIGE, BESONDERE, BEISPIELHAFTE SCHÄDEN ODER FOLGESCHÄDEN (EINSCHLIESSLICH, JEDOCH NICHT BESCHRÄNKT AUF DIE BESCHAFFUNG VON ERSATZWAREN ODER -DIENSTLEISTUNGEN, NUTZUNGS-, DATEN- ODER GEWINNVERLUSTE ODER UNTERBRECHUNG DES GESCHÄFTSBETRIEBS), UNABHÄNGIG DAVON, WIE SIE VERURSACHT WURDEN UND AUF WELCHER HAFTUNGSTHEORIE SIE BERUHEN, OB AUS VERTRAGLICH FESTGELEGTER HAFTUNG, VERSCHULDENSUNABHÄNGIGER HAFTUNG ODER DELIKTSHAFTUNG (EINSCHLIESSLICH FAHRLÄSSIGKEIT ODER AUF ANDEREM WEGE), DIE IN IRGENDEINER WEISE AUS DER NUTZUNG DIESER SOFTWARE RESULTIEREN, SELBST WENN AUF DIE MÖGLICHKEIT DERARTIGER SCHÄDEN HINGEWIESEN WURDE.

NetApp behält sich das Recht vor, die hierin beschriebenen Produkte jederzeit und ohne Vorankündigung zu ändern. NetApp übernimmt keine Verantwortung oder Haftung, die sich aus der Verwendung der hier beschriebenen Produkte ergibt, es sei denn, NetApp hat dem ausdrücklich in schriftlicher Form zugestimmt. Die Verwendung oder der Erwerb dieses Produkts stellt keine Lizenzierung im Rahmen eines Patentrechts, Markenrechts oder eines anderen Rechts an geistigem Eigentum von NetApp dar.

Das in diesem Dokument beschriebene Produkt kann durch ein oder mehrere US-amerikanische Patente, ausländische Patente oder anhängige Patentanmeldungen geschützt sein.

ERLÄUTERUNG ZU „RESTRICTED RIGHTS“: Nutzung, Vervielfältigung oder Offenlegung durch die US-Regierung unterliegt den Einschränkungen gemäß Unterabschnitt (b)(3) der Klausel „Rights in Technical Data – Noncommercial Items“ in DFARS 252.227-7013 (Februar 2014) und FAR 52.227-19 (Dezember 2007).

Die hierin enthaltenen Daten beziehen sich auf ein kommerzielles Produkt und/oder einen kommerziellen Service (wie in FAR 2.101 definiert) und sind Eigentum von NetApp, Inc. Alle technischen Daten und die Computersoftware von NetApp, die unter diesem Vertrag bereitgestellt werden, sind gewerblicher Natur und wurden ausschließlich unter Verwendung privater Mittel entwickelt. Die US-Regierung besitzt eine nicht ausschließliche, nicht übertragbare, nicht unterlizenzierbare, weltweite, limitierte unwiderrufliche Lizenz zur Nutzung der Daten nur in Verbindung mit und zur Unterstützung des Vertrags der US-Regierung, unter dem die Daten bereitgestellt wurden. Sofern in den vorliegenden Bedingungen nicht anders angegeben, dürfen die Daten ohne vorherige schriftliche Genehmigung von NetApp, Inc. nicht verwendet, offengelegt, vervielfältigt, geändert, aufgeführt oder angezeigt werden. Die Lizenzrechte der US-Regierung für das US-Verteidigungsministerium sind auf die in DFARS-Klausel 252.227-7015(b) (Februar 2014) genannten Rechte beschränkt.

Markeninformationen

NETAPP, das NETAPP Logo und die unter http://www.netapp.com/TM aufgeführten Marken sind Marken von NetApp, Inc. Andere Firmen und Produktnamen können Marken der jeweiligen Eigentümer sein.