Hewlett Packard Enterprise | NVIDIA

Technical white paper

Accelerating speech AI with HPE and NVIDIA

An end-to-end solution for building advanced speech recognition and text-to-speech systems

Contents

1.0 Introduction

Voice or speech artificial intelligence (AI) is making its way into ever more types of applications, such as customer support virtual assistants, chatbots, video conferencing systems, drive-through convenience food orders, retail by phone, augmented reality, and media and entertainment. This increase is due to the benefits that customized voice applications offer businesses of any size, in almost every industry—from global enterprises to original equipment manufacturers delivering speech AI–based systems and cloud services to systems integrators and independent software vendors.

More people are working and learning from home, shopping online, and seeking remote customer support, which strains contact centers and pushes voice applications to their limits. Customer service wait times have recently tripled as staffing shortages have hit contact centers hard, according to a 2022 Bloomberg report.

While AI for voice services has been in high demand, development tools have lagged. Enterprises want low code development tools to facilitate building speech AI applications that feature high accuracy as well as ways to customize voice experiences. Enterprises need a solution stack that can optimize the collection, processing, and analysis of audio data to capitalize on the AI speech opportunity. Enterprises will face challenges, like deploying speech AI–based applications at scale while ensuring high accuracy, addressing data security and privacy concerns, enabling interactions in many languages, translating industry-specific jargon, and enabling real-time responses.

That's where the collaboration of HPE and NVIDIA® comes in. A joint solution powered by NVIDIA Riva, HPE Ezmeral, NVIDIA-Certified Systems by HPE, and HPE GreenLake allow enterprises to accelerate the deployment of speech AI applications. This joint solution enables enterprises to deploy powerful speech AI technologies with the highest possible accuracy at the lowest possible cost.

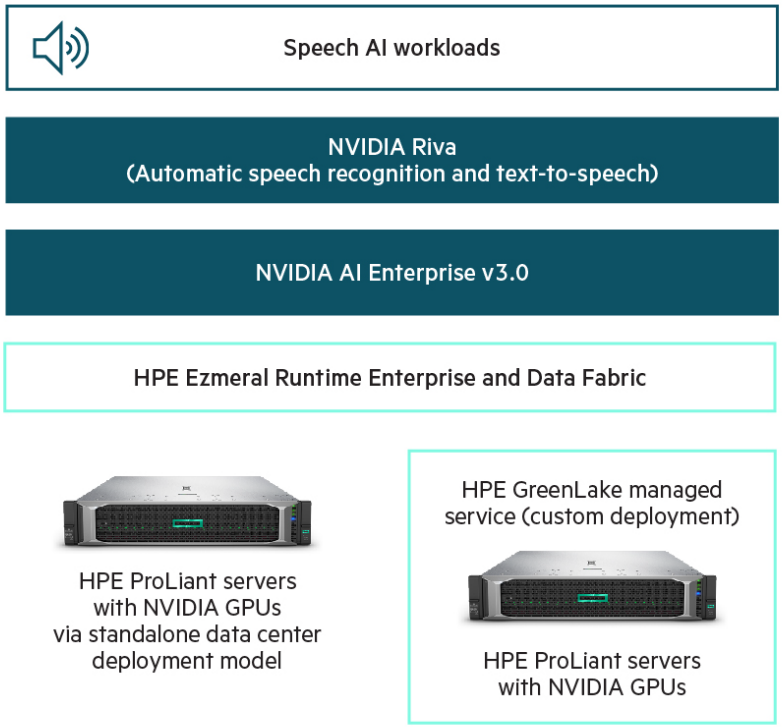

Figure 1. NVIDIA Riva on HPE

2.0 NVIDIA Riva

NVIDIA Riva is a GPU-accelerated speech AI—automatic speech recognition (ASR) and text-to-speech (TTS)—SDK for building fully customizable, real-time conversational AI pipelines and deploying them in clouds, on-premises in data centers, at the edge, or on embedded devices. Riva is fully customizable and optimized for maximum performance on NVIDIA-certified servers. NVIDIA Riva is available as a set of containers and pretrained models.

2.1 Riva ASR

ASR, also known as speech-to-text, refers to the task of getting a program to automatically transcribe spoken language to text. It takes human voice as input and converts it into readable text with minimal transcription errors in near real-time performance. The NVIDIA Riva skill provides high-quality, pretrained, out-of-the-box ASR models across 15 languages: English (U.S./UK), Spanish (LATAM/Spain), Mandarin, Hindi, Russian, Arabic, Japanese, Korean, German, Portuguese Brazilian, French, and Italian.

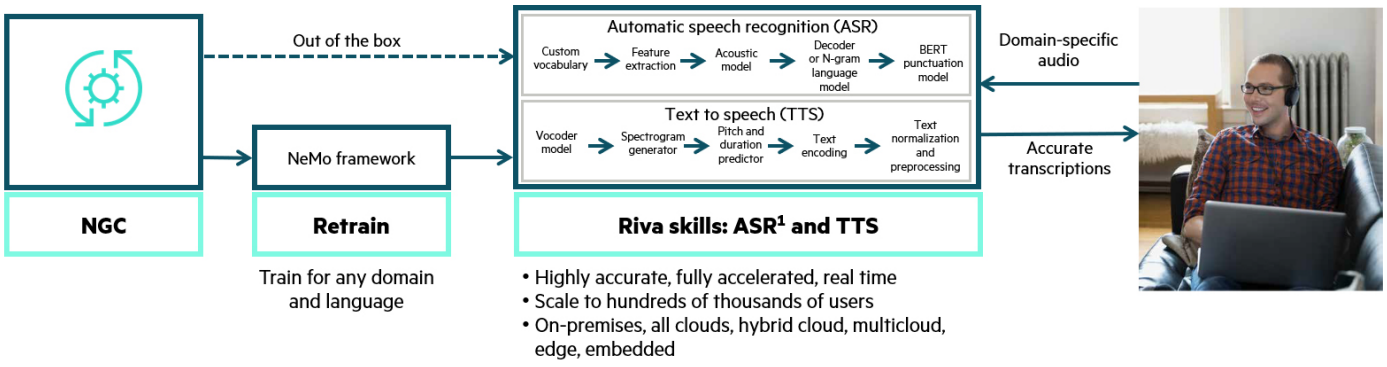

Figure 2. NVIDIA Riva—Accelerated SDK for real-time speech AI

2.2 Riva TTS

The TTS pipeline implemented for the Riva TTS service is based on a two-stage pipeline. Riva first generates a mel spectrogram using the first model, and then generates speech using the second model. This pipeline forms a TTS system that enables you to synthesize natural sounding speech from raw transcripts without any additional information such as patterns or rhythms of speech. Riva TTS currently supports two high-quality male and female voices for English.

Riva is a containerized application and can be deployed using Kubernetes and Helm charts. Kubernetes, also known as K8s, is an open source platform for automating deployment, scaling, and managing containerized applications. Kubernetes includes support for GPUs, which enables enterprises to scale up training and inference deployment to multicloud GPU clusters seamlessly. Helm is an application package manager running on top of Kubernetes. It lets you create Helm charts where you can define, install, and upgrade Kubernetes applications such as Riva.

3.0 Riva on HPE

HPE Ezmeral Runtime Enterprise provides enterprise-grade container management for Kubernetes-based language processing. Designed to run both cloud-native and non-cloud-native applications at scale with persistent data, HPE Ezmeral helps data scientists support multiple enterprise-grade containerized AI/machine learning (ML) applications via multitenancy. This enables organizations to deploy speech AI ASR workloads built using Riva faster and greatly accelerates time to value.

HPE Ezmeral also features multiple levels of built-in security controls to integrate with identity providers such as AD/LDAP, single sign-on, and SAML integration. Additionally, clusters and applications can be further secured by using strongly attested identities provided by SPIRE for authentication. Users can enjoy intelligent traffic shaping, load balancing, canary rollouts, and A/B testing of AI speech application–based microservices through HPE Ezmeral built-in service mesh.

HPE ProLiant servers enable the high-performance computing applications required to turn audio into insights. HPE systems that are NVIDIA certified bring together HPE servers and NVIDIA GPUs in optimized configurations that are validated for performance, manageability, security, and scalability and back by enterprise-grade support. IT professionals can even get help architecting, training, and project managing ASR solutions with HPE Pointnext Services.

HPE Ezmeral Data Fabric supports IT analytics and data science initiatives by providing a fully integrated data storage and analytics platform. It supports a wide variety of data types and formats, reducing the time-consuming task of negotiating access to geographic speech data silos. HPE Ezmeral Data Fabric provides a unified view, and data access point simplifies speech and on-speech data management and reduces the overhead necessary to access Riva data that might be stored in hybrid and multicloud. Enterprise customers can use the service to create a unified data repository for data scientists, developers, and IT to access and use, with control of how it is used and shared.

HPE GreenLake is a scalable IT infrastructure service that provides a usage-based IT platform and vertical-based workloads that are aligned to capacity usage. IT organizations can easily scale up and down to handle fluctuations in demand and receive personalized HPE support to augment their IT teams. This flexible hybrid cloud model provides enterprises with the agility to scale their Riva environment without the usual delays associated with procuring and managing new infrastructure. Enterprise can better align their spending with capacity needs for their speech AI projects. This will lead to a reduction in overprovisioning and ancillary support spend.

The following sections discuss how Riva can be deployed on HPE using a combination of HPE Ezmeral software, HPE servers, and HPE storage arrays. Both an entry-level and scalable production-level configuration will be highlighted. Please see Appendix A for an outline of deployment steps of Riva software on HPE Ezmeral Runtime Enterprise.

4.0 Riva setup basics

NVIDIA Riva is set up as a client-server infrastructure. On the client side, we have the ASR/TTS applications that capture input audio/text streams or batches and create gRPC messages with protocol buffer details outlined here. The gRPC messages are sent to the Riva server where all the ASR/TTS AI processing occurs. Therefore, the Riva server is the brain of the system, which requires GPUs, whereas the client needs only CPUs. Client to server follows a many-to-one architecture that one Riva server can handle multiple and/or concurrent requests from a single or many client apps.

While Riva is highly computationally optimized, the capacity of the Riva server is a function of several software and hardware configurations that can be found here for ASR and here for TTS.

In the context of Kubernetes deployment, Riva would have at least two types of worker nodes:

1. CPU Linux® worker nodes—For client application pods
    a. An ASR/TTS apps could be written by a programming language that gRPC supports. Riva comes with CLI and Python wheel clients. Here are some of the sample apps built based on Riva.
2. GPU Linux worker nodes—For Riva server pods
    a. The best practice is to maintain a 1:1:1 mapping among Riva server container: Riva server pod: GPU card.
    b. One Riva server container can be configured in a way that can run multiple ASR or TTS pipelines with different speech AI models concurrently. In that case, the client app can choose to which Riva server pipeline it sends the gRPC messages. The number of concurrent pipelines is a function of the complexity and configuration of the pipeline, GPU memory, and compute capacity. Multiple pipelines can share some of the components, hence saving GPU memory (more).

5.0 Riva entry-level configuration on HPE

This entry-level configuration is designed for small-scale proof-of-concept (POC) and testing environments. It minimizes the number of compute servers for cost reasons and is not designed with high availability (HA) in mind. The HPE Ezmeral Runtime Enterprise and Kubernetes control planes are deployed on virtual machines (VMs) to conserve host count as they don't typically need a lot of compute resources, especially for a POC/entry-level deployment.

The Kubernetes worker nodes are deployed on bare metal (see Figure 3) to provide the best performance possible over two server hosts: one server host for the NVIDIA Riva server and the other host for running a single ASR or TTS application. For storage back end, any existing network file system (NFS)–based storage can be used including HPE and non-HPE storage options. The performance of this configuration should be similar to the A30 results outlined in the NVIDIA Riva user guide.

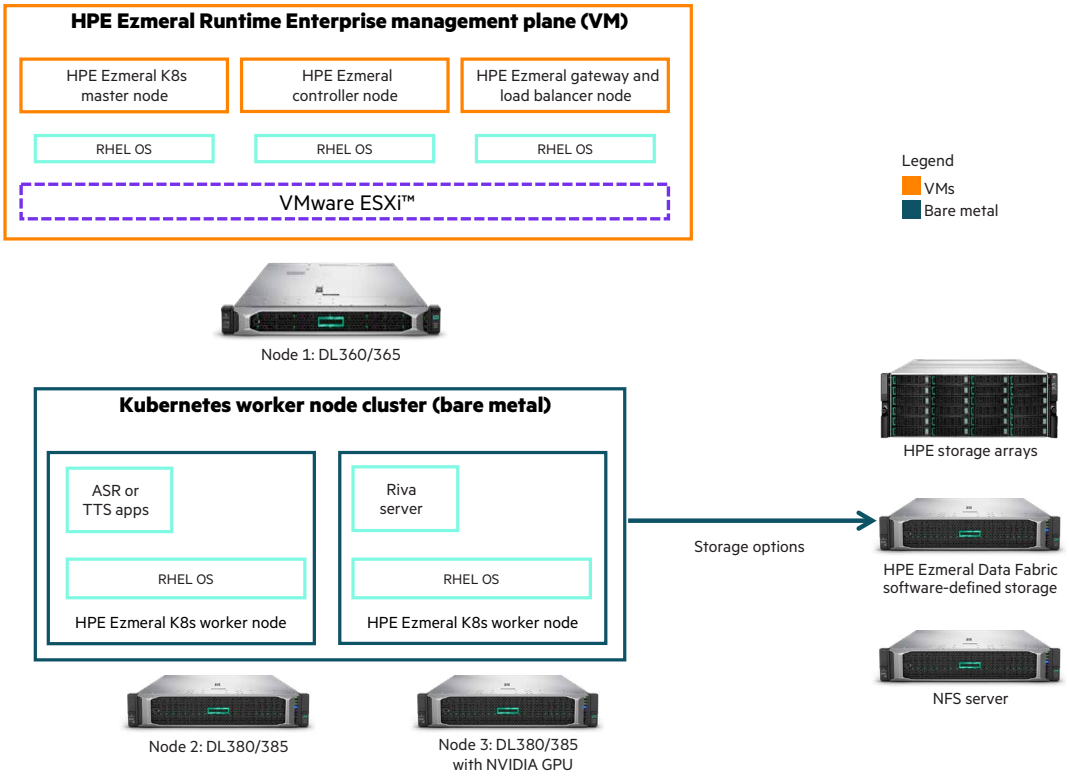

Figure 3. Suggested entry-level configuration for NVIDIA Riva on HPE

If multiple ASR or TTS apps or containers are being deployed, then Node 2 should be switched from bare metal to a VM environment to provide better isolation and resource allocation for each of the applications/containers (see Figure 4). A single NVIDIA Riva server would be deployed on a bare metal server for performance reasons for servicing the incoming speech streams.

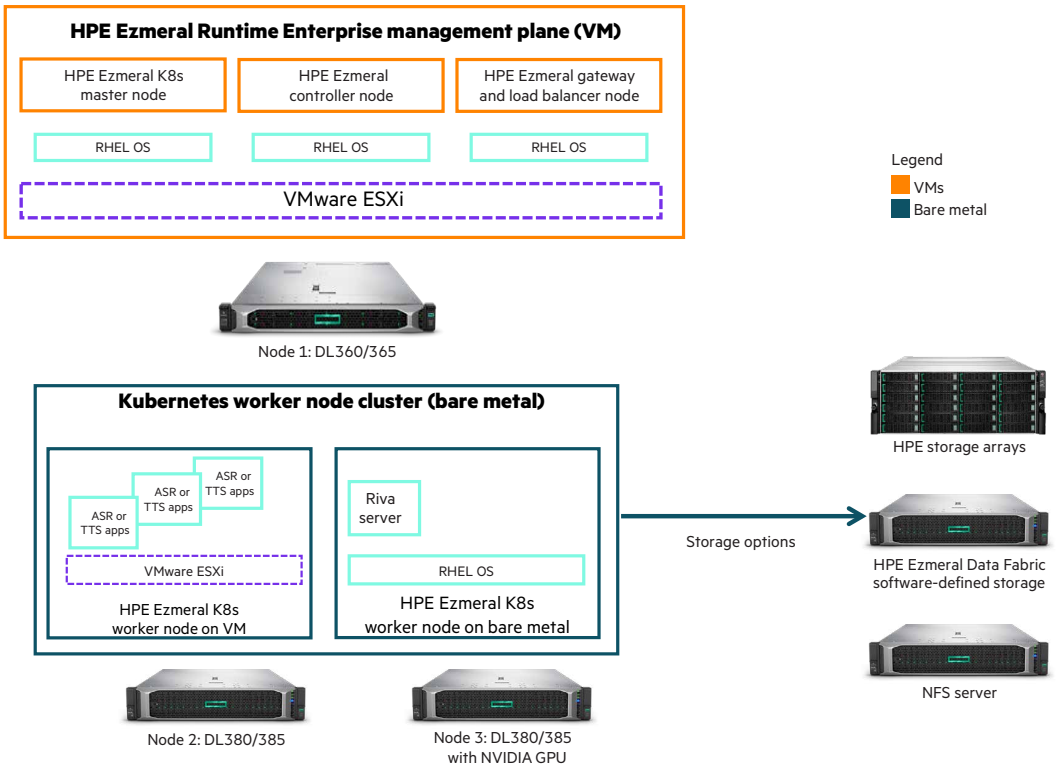

Figure 4. Suggested entry-level configuration for NVIDIA Riva on HPE (multiple ASR or TTS apps)

Table 1. Suggested BOM for entry and POC configuration

Host type	HPE server model
HPE Ezmeral Runtime Enterprise primary controller, gateway, load balancer Kubernetes master	HPE ProLiant DL360/365 Gen10 Plus or Gen11 with minimal 24 cores per socket, 256 GB RAM, 1x 10 GB NIC, 1x 1.2 TB SATA SSD disk for OS 2x 500 GB SATA SSD drive for local ephemeral storage
Kubernetes worker for ASR or TTS apps	HPE ProLiant DL380/385 Gen10 Plus or Gen11 with minimal 16 cores per socket, 256 GB RAM, 1x 10 GB NIC, 1x 1.2 TB SATA SSD disk 1x 1.2 TB SATA SSD disk for OS 2x 500 GB SATA SSD drive for local ephemeral storage
Kubernetes worker for Riva server	HPE ProLiant DL380/385 Gen10 Plus or Gen11 with minimal 24 cores per socket, 256 GB RAM, 1x 10 GB NIC, 1 NVIDIA A30 1x 1.2 TB SATA SSD disk for OS 2x 500 GB SATA SSD drive for local ephemeral storage
Back-end storage	Any HPE or non-HPE storage arrays or NFS on a single-node server or 4 nodes HPE Ezmeral Data Fabric cluster (docs.datafabric.hpe.com/71/AdvancedInstallation/MinimumClusterSize.html?hl=four)

6.0 Riva production-level configuration on HPE

The following Riva on HPE configuration is recommended for production-level deployment as highlighted in Figure 4. This production design uses a combination of VMs and bare metal servers for cost/performance optimization. The HPE Ezmeral control plane and K8s master are deployed on multiple VM instances to provide HA. VMs are used to conserve server host count as the HPE Ezmeral control plane and K8s master don't typically need a lot of compute resources.

The Kubernetes worker nodes are deployed on bare metal to provide the best performance and HA possible over a minimum of four server hosts. A minimum of two server hosts to provide HA and horizontal pod scaling for the ASR or TTS application. A similar design also applies for the NVIDIA Riva server pod where each pod is assigned a dedicated NVIDIA GPU.

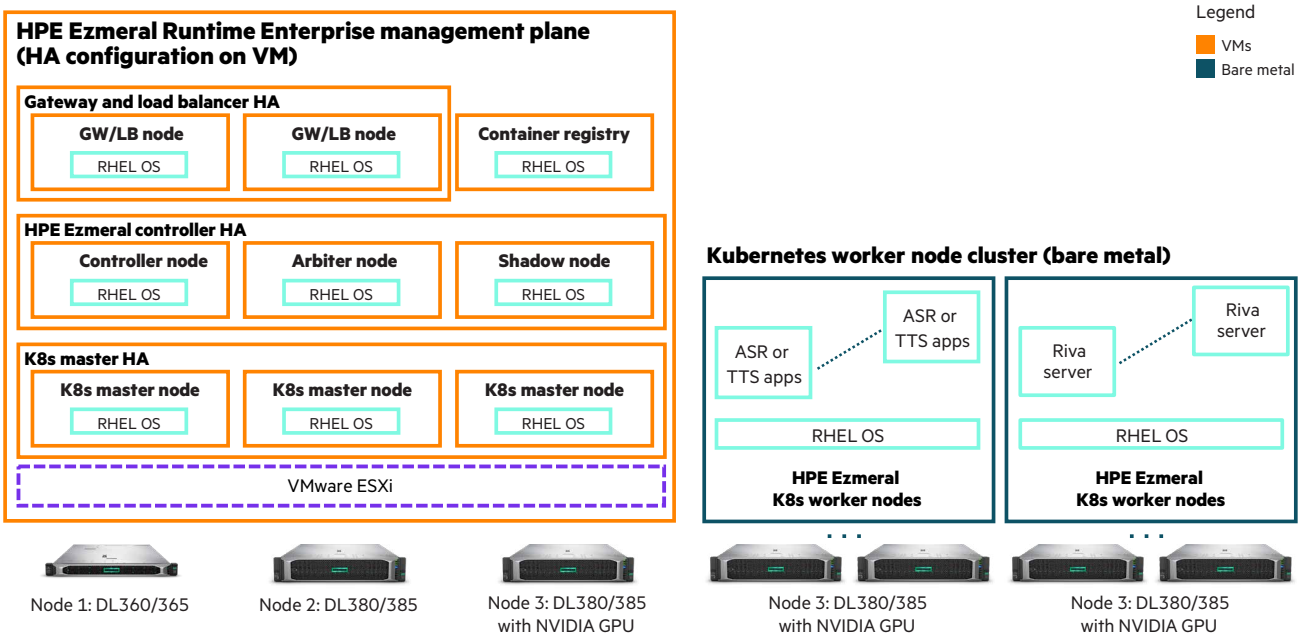

Figure 5. Suggested production configuration for NVIDIA Riva on HPE

Table 2. Suggested BOM for production configuration

Host type	HPE server model
HPE Ezmeral Runtime Enterprise primary controller, gateway, load balancer Kubernetes master	HPE ProLiant DL360/365 Gen10 Plus or Gen11 with minimal 24 cores, 256 GB RAM, 2x 10 GB NIC 1x 1.2 TB SSD disk for OS 2x 500 GB SSD drives for local ephemeral storage
Kubernetes worker for ASR or TTS apps	HPE ProLiant DL380/385 Gen10 Plus or Gen11 with 18 to 32 CPU cores per socket, 512 GB RAM, 2x 10 GB NIC 1x 1.2 TB SSD disk for OS 2x 500 GB NVMe drives for local ephemeral storage
Kubernetes worker for Riva server	HPE ProLiant DL380/385 Gen10 Plus or Gen11 minimal 64 cores per socket, with 1024 GB RAM, 2x 10 GB NIC 1x 1.2 TB SSD disk for OS 2x 500 GB NVMe drives for local ephemeral storage 1 to 3 NVIDIA A100 GPUs per HPE ProLiant depending on number of Riva server pods on this HPE ProLiant
Back-end storage options	HPE Nimble Storage or HPE Alletra storage arrays or 6 nodes HPE Ezmeral Data Fabric cluster (docs.datafabric.hpe.com/71/AdvancedInstallation/PlanningtheCluster-examples.html)

Conclusion

Speech AI has become an integral and essential part of consumers' everyday lives. Enterprises are discovering new ways of bringing great value to their products by incorporating speech AI services. NVIDIA and HPE can help enterprises adopt speech AI with an end-to-end solution. The combined offering of NVIDIA Riva, HPE Ezmeral, HPE certified NVIDIA servers, and HPE GreenLake enable enterprises to quickly build and deploy ASR and speech synthesis services-based applications.

Appendix A: NVIDIA Riva on HPE Ezmeral Runtime Enterprise deployment workflow

The deployment steps for NVIDIA Riva on HPE Ezmeral Runtime Enterprise (ERE) will mostly follow the steps from the NVIDIA Riva user guide and some modifications that are unique for HPE Ezmeral.

This is an example of deploying and scaling Riva Speech Skills on HPE ERE. It includes the following steps:

1. Creating the ERE cluster
2. Deploying the Riva API service
3. Deploying a sample client
4. Scaling the cluster

Prerequisites
Before continuing, ensure you have:

- An ERE account with the appropriate user/role privileges to manage ERE
- The ERE command-line tool, configured for your account
- Access to NVIDIA NGC and the associated command-line interface
- Cluster management tools helm and kubectl

The deployment described in this paper has been tested with helm (v3.9.4) and kubectl (v1.21.10).

Step 1: Creating the HPE ERE cluster
The cluster contains three separate node groups:

- Compute worker node(s): A GPU-equipped node where the main Riva service is running on a NVIDIA Riva certified GPU provides good value and sufficient capacity for validating applications. This node group can be used to scale large number of nodes.
- Master/control plane nodes: Formerly known as master node. In HA, it requires to have a minimum of three nodes. It can be used to stage as a client node accessing the Riva service. The node is used for benchmarking in this example. These nodes are not considered part of the HPE ERE control plane.
- Data fabric storage nodes: These nodes have data fabric tag enabled. These hosts can become the data fabric worker storage nodes if we implement HPE Ezmeral Data Fabric on Kubernetes.

1. Deploy the HPE ERE v5.5 cluster using these installation steps.
2. Verify that the nodes now appear in Kubernetes. If so, the cluster was successfully created.

```
cat .kube/config

kubectl get pods-A
kubectl get nodes--show-labels
kubectl get nodes--selector role=workers
kubectl get nodes--selector role=clients
kubectl get nodes--selector role=loadbalancers
```

Step 2: Deploying the NVIDIA Riva API server
The NVIDIA Riva Speech Skills Helm chart is designed to automate deployment to a Kubernetes cluster. After downloading the Helm chart, please theses following changes specific to HPE Ezmeral.

1. Download and untar the Riva API Helm chart. Replace VERSION_TAG with the specific version needed.

```
export NGC_CLI_API_KEY=<your NGC API key>

export VERSION_TAG="2.5.0"

helm fetch helm.ngc.nvidia.com/nvidia/riva/charts/riva-api-${VERSION_TAG}.tgz --
username='$oauthtoken' --password=$NGC_CLI_API_KEY

tar -xvzf riva-api-${VERSION_TAG}.tgz
```

2. In the `riva-api` folder, modify the following files:
   a. values.yaml
      i. Set riva.speechServices.[asr,nlp,tts] to true or false as needed to enable or disable these services. For example, if only ASR is needed, then set ASR value to true and the NLP and TTS values to false.
      ii. In modelRepoGenerator.ngcModelConfigs.[asr,nlp,tts], comment or uncomment specific models or languages as needed.
      iii. Change service.type from LoadBalancer to ClusterIP. This directly exposes the service only to other services within the cluster, such as the proxy service to be installed below.
   b. templates/deployment.yaml
      i. **Skip this section** on adding a node selector constraint to ensure that NVIDIA Riva is only deployed on the correct GPU resources. In spec.template.spec, add:
      ii. **nodeSelector:**
      iii. **ERE.amazonERE.com/nodegroup**: gpu-linux-workers
3. **From the NVIDIA Riva deployment guide, please skip Step 3 of the section Deploy the Riva API. Instead, we will use the HPE ERE NVIDIA device plug-in that HPE provides as a part of the ERE installation. This device plug-in will be used instead of the one provided by NVIDIA.** The ERE cluster is running as described in the section Creating ERE cluster above built using the installation guide already offers a ERE customized NVIDIA device plug-in from HPE as shown in the example below. No need to use the device plug-in from NVIDIA.

```
kubectl get nodes "-o=custom-
columns=NAME:.metadata.name,GPU:.status.allocatable.nvidia\.com/gpu"

NAME                    GPU
ezam-03.perflab.hp.com  <none>
ezam-15.perflab.hp.com  1
kubectl get pods -n kube-system | grep nvidia
nvidia-device-plugin-mixed-hw97r 1/1   Running       0      4d
nvidiagpubeat-mhxbt      1/1    Running       0      4d
```

4. Ensure you are in a working directory with riva-api as a subdirectory, and then install the Riva Helm chart. You can explicitly override variables from the values.yaml file, such as the riva.speechServices.[asr,nlp,tts] settings.

```
helm install riva-api \
--set ngcCredentials.password=`echo -n $NGC_CLI_API_KEY | base64-w0`\
--set modelRepoGenerator.modelDeployKey=`echo -n tlt_encode | base64-w0`\
--set riva.speechServices.asr=true\
--set riva.speechServices.nlp=false\
--set riva.speechServices.tts=false--generate-name
```

5. The Helm chart runs two containers in order: a riva-model-init container that downloads and deploys the models, followed by a riva-speech-api container to start the speech service API. Depending on the number of models, the initial model deployment could take an hour or more. To monitor the deployment, use kubectl to describe the riva-api pod and to watch the container logs.

```
export pod=`kubectl get pods | cut -d " " -f 1 | grep riva-api`
kubectl describe pod $pod
kubectl logs -f $pod -c riva-model-init
kubectl logs -f $pod -c riva-speech-api
```

Step 3: Deploying a sample client

Riva provides a container with a set of pre-built sample clients to test the NVIDIA Riva services. These clients are also available on GitHub for those interested in using them.

1. Create the client-deployment.yaml file that defines the deployment, containing the following:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: riva-client
  labels:
    app: "rivaasrclient"
  spec:
  replicas: 1
  selector:
    matchLabels:
      app: "rivaasrclient"
  template:
    metadata:
      labels:
        app: "rivaasrclient"
    spec:
      nodeSelector:
          ERE.amazonERE.com/nodegroup: cpu-linux-clients
      imagePullSecrets:
      - name: imagepullsecret
      containers:
      - name: riva-client
        image: "nvcr.io/nvidia/riva/riva-speech-client:2.5.0"
        command: ["/bin/bash"]
        args: ["-c", "while true; do sleep 5; done"]
```

2. Deploy the client service.

```
kubectl apply -f client-deployment.yaml
kubectl get pods
NAME                              READY   STATUS    RESTARTS   AGE
riva-api-154821-6458fcfdc8-m5hqs  1/1     Running   0          46h
riva-client-8f8759-8pd6f          1/1     Running   0          26h
```

3. Connect to the client pod.

```
export cpod=`kubectl get pods | cut -d " " -f 1 | grep riva-client`
kubectl exec --stdin --tty $cpod /bin/bash
```

4. From inside the shell of the client pod, run the sample ASR client on an example .wav file. Specify the endpoint, with port 50051, as the service address. Below are examples of one .wav file or multiple .wav files.

```
kubectl get service
NAME              TYPE      CLUSTER-IP    EXTERNAL-IP PORT(S)    AGE
kubernetes        ClusterIP 10.96.0.1     <none>      443/TCP    4d
riva-api-1654821  ClusterIP 10.106.92.243 <none>      8000/TCP,8001/TCP,8002/TCP,50051/TCP 46h
riva_streaming_asr_client \
 --audio_file=/opt/riva/wav/en-US_sample.wav \
 --automatic_punctuation=true \
 --riva_uri=10.106.92.243:50051
```

For a in file1.wav file2.wav file3.wav file4.wav file5.wav; do riva_asr_client --audio_file=/opt/riva/wav/${a} --automatic_punctuation=true --riva_uri=10.106.92.243:50051; done

5. If the test .wav files are successfully processed by NVIDIA Riva, you should see the transcription texts associated with these .wav files.

Learn more at

hpe.com/us/en/software.html
hpe.com/us/en/greenlake.html
nvidia.com/en-us/ai-data-science/products/riva/
hpe.com/us/en/solutions/artificial-intelligence/nvidia-collaboration.html
hpe.com/us/en/software/marketplace.html

a50007987ENW

Hewlett Packard Enterprise