Deploying Cloud Native Qumulo (CNQ) on GCP Using Terraform release no releases or repo not found

This repository contains comprehensive Terraform modules that let you deploy Google Cloud Storage buckets and then create a CNQ cluster with 1 or 3 to 24 VMs that are well-architected and have fully elastic compute and persistent object storage.

Getting Started with Cloud Native Qumulo (CNQ)

For an overview of CNQ, its reference architecture, and limits, see How Cloud Native Qumulo Works and for prerequisites and detailed instructions, see Deploying Cloud Native Qumulo on GCP with Terraform on the Documentation Portal.

Qumulo Core >= 7.6.0 is required for this Terraform

> ✅ **Tip:** For help with deployment, configuration, updates, scaling out your cluster, and best practices for high performance, message us on Slack.

Description of Modules in This Repository

There are two modules in this repository *persistent-storage* and *compute*. The *persistent-storage* module creates the GCS buckets for all data the Qumulo cluster stores. The *compute* module creates the GCE VMs and the Qumulo NeuralCache which form the compute resources for the cluster.

Terraform deployment

You may choose to deploy each module using the `terraform.tfvars` file with your specific variable values. Alternatively, if you are integrating into a CI/CD pipeline, you may opt to call the module directly passing in the variables for the module.

Terraform state management

By default the `provider.tf` files in each module are configured to write state to GCS. It is absolutely critical that the state for each module is in a separate Terraform state file. If deploying the modules as packaged and using the `terraform.tfvars` file you need only configure the bucket for GCS state storage in each `provider.tf`.
Terraform will then create a 'tfstate' folder with unique subfolders for the state files for persistent-storage and compute. This must be done prior to `terraform init`.
If you are integrating this into your own CI/CD pipeline and calling the modules directly ensure that state is managed independently for both modules.

About This Repository

For more information about this repository, contact Dack Busch and Gokul Kupparaj.