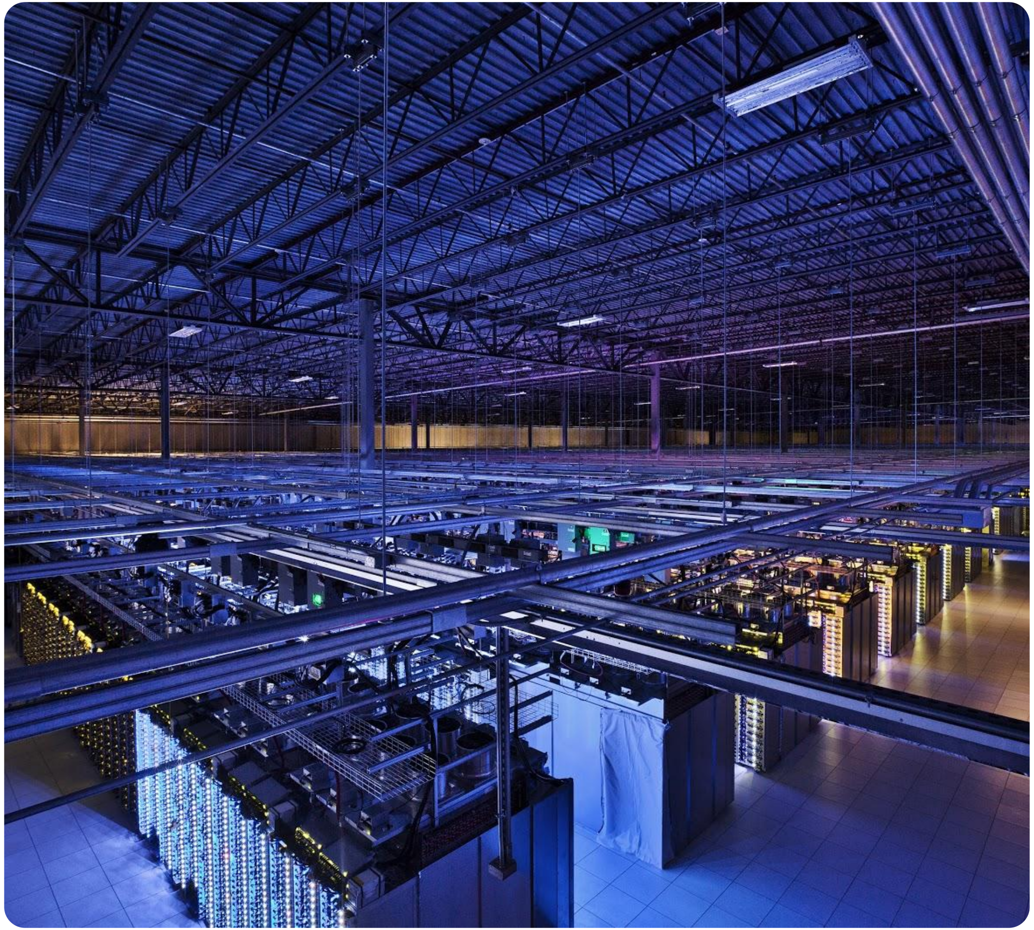
Google Cloud Cluster Toolkit

Carlos Boneti
Sr. Staff Software Engineer
Google Cloud

Google Cloud



Cluster Toolkit Objective

"Make it **easy** for customers and partners to deploy **repeatable turnkey** HPC environments following Google Cloud's **HPC best practices**"

Benefits of the Cluster Toolkit

Easily create turnkey HPC environments

- Easily create turnkey HPC environments and get the best performance out-of-the-box
- Start with verified cluster blueprints and stay up to date with GCP best practices

Configurable, extensible and open-source

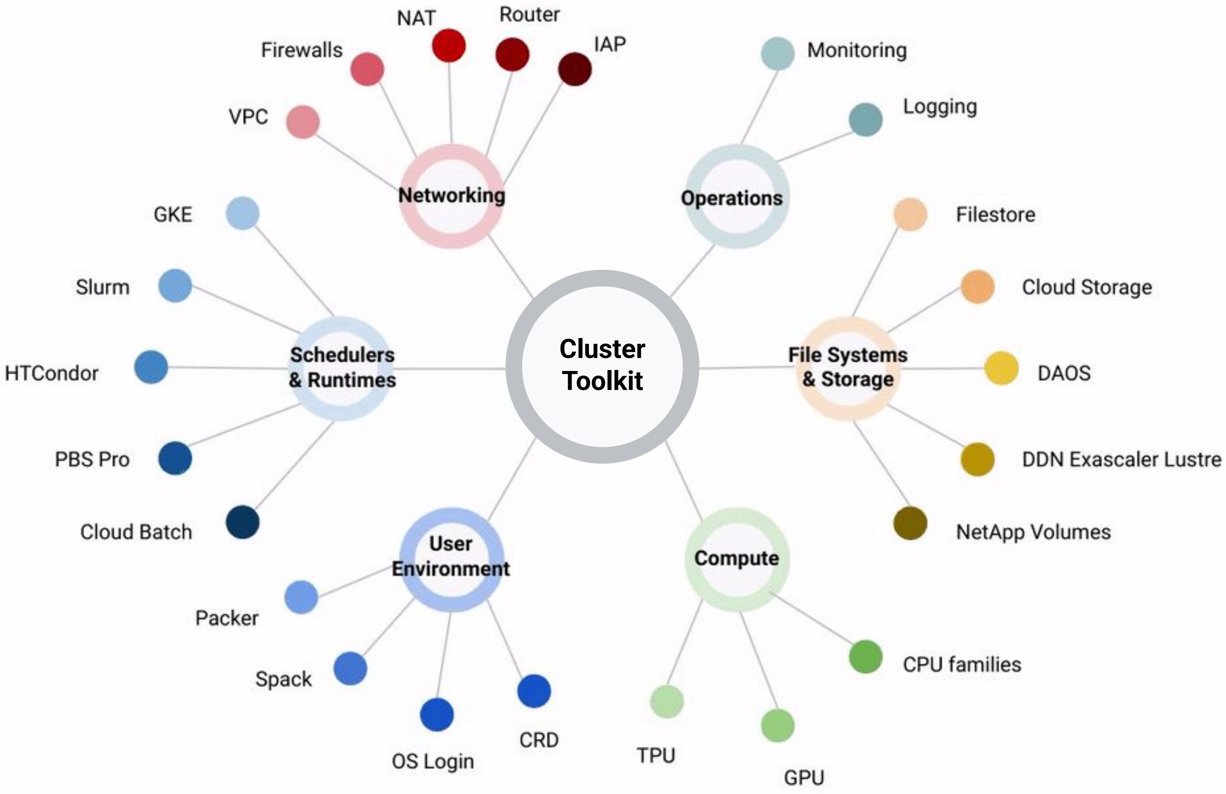
- Based on standard tech stack, and allows a broad set of cluster customizations: YAML, Terraform, Packer, Ansible and Shell
- Open source - available for customers and solution providers to add new features

Supports analytics via Cloud Monitoring

- Built-in labeling functionality makes it easy to track resources with Cloud Monitoring
- Optional custom HPC labeling available to get insights on cluster performance

A vast range of services

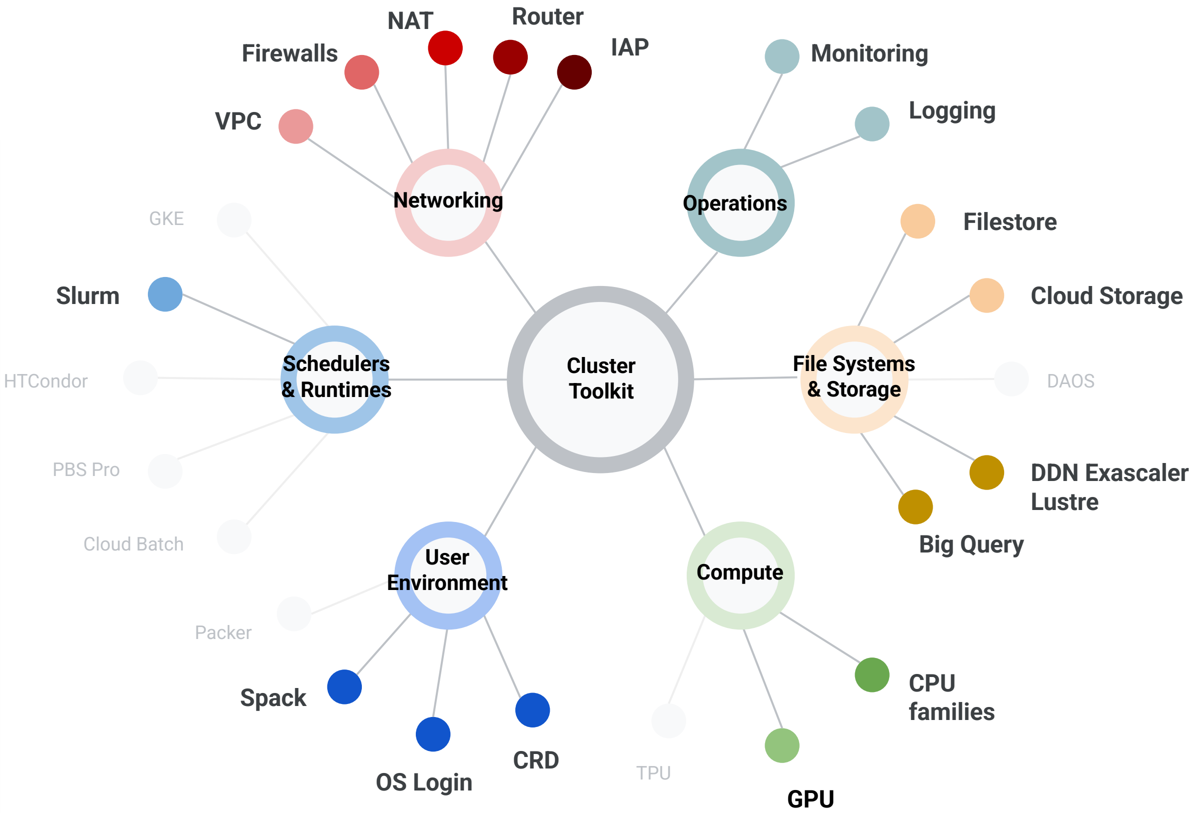
Deploying cluster computing environments often requires leveraging technologies from various domains.



UC Riverside

Ex: UCR

"Through the Scale and Innovation of Google Cloud Platform and their Toolkit we revolutionized research at UCR. We achieve goals once deemed impossible by our researchers, in extraordinary timeframes."
– Chuck Forsyth, Director Research Computing, UCR.

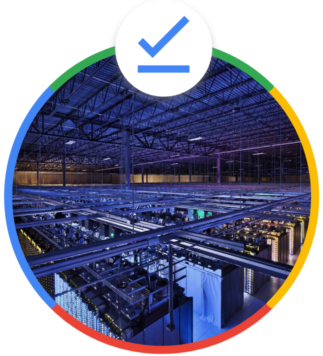


Journey to a working environment
From 37K lines to 80

HPC Environment Configuration
Environments are extended by editing a simple text file in YAML format (**80 lines**)

Optional: Create deployment artifacts
Based on the toolkit modules and the config file, generates the deployment folder containing code for infrastructure provisioning and configuration (**~20K lines**)

Deploy environment
Reads the deployment folder and provisions all infrastructure and software (**~37K lines**)

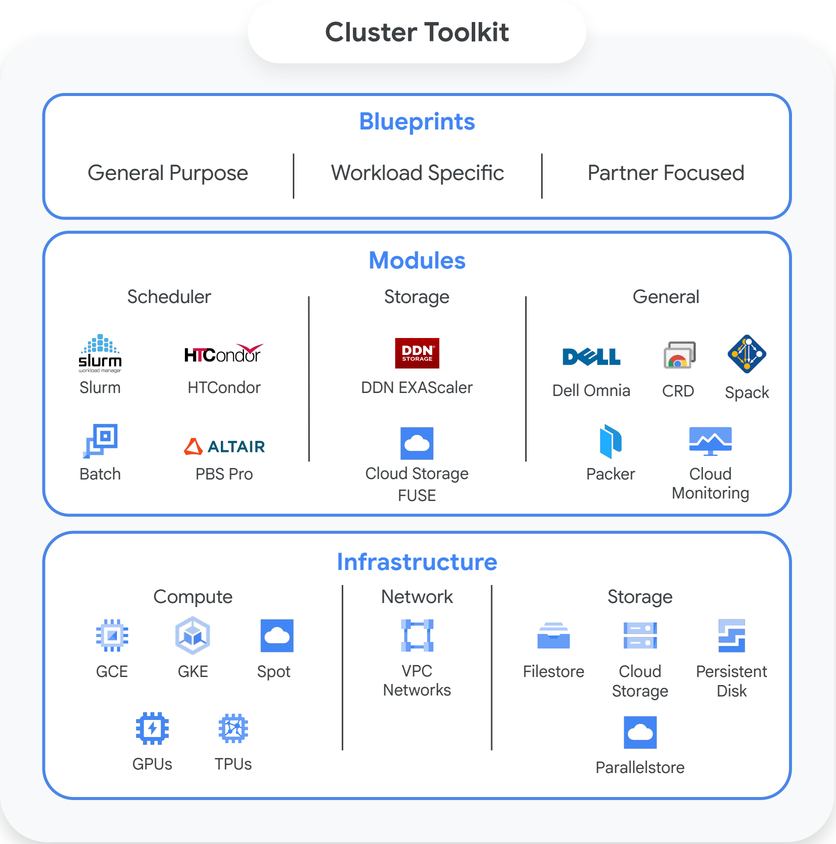


Cluster Toolkit

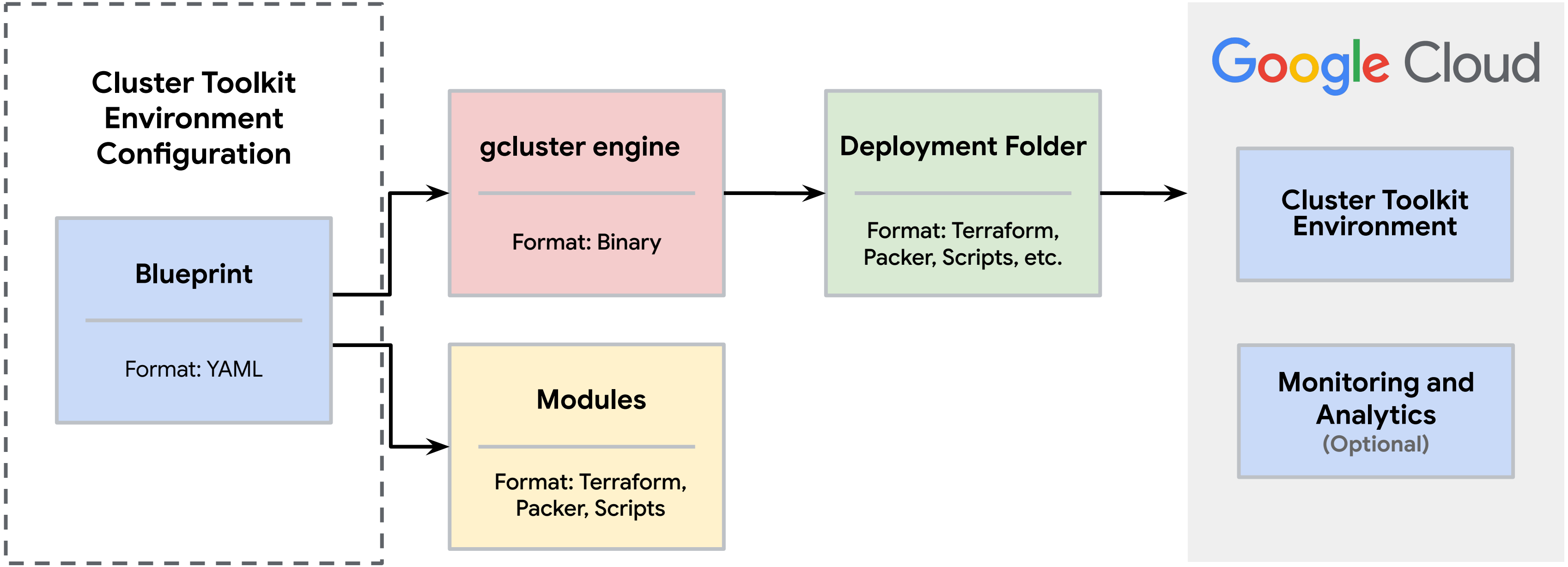
The Cluster Toolkit is a modular, composable, terraform-based toolkit designed to make it easy to deploy repeatable, turnkey HPC environments that follow Google Cloud's HPC best practices.

Key components:

- **Blueprints** defines an HPC environment. They reference individual modules which they use to compose the desired system.
- **Modules** are code to deploy specific components of an HPC system, such as a cluster's partition, a storage system, or the network. Either imported from public sources (Github), or hosted privately.
- **Infrastructure** will host the HPC system that is built, and the Cluster Toolkit supports the core Google Cloud services and features that are required for HPC.



Cluster Toolkit Deployment



The YAML blueprint is a higher level abstraction of the cluster

Modules are building blocks

- Abstracted, loosely defined interfaces for storage, network, etc.
- We curate a list of modules implementing known best practices
- Can point to arbitrary terraform or packer code

Dependency is injected between modules

- Composing becomes easier if modules focus on specific responsibilities
- ex: A scheduler should not create networks, a database should not create storage systems, a storage should not create SAs.

Before and after terraform

- Can build images and handle software installation, initialization beyond terraform
- The toolkit generates terraform, which makes it familiar to customers who already have pipelines or TF expertise

Why not just use terraform?

Easier:

- Compact YAML blueprint syntax is easy to explain to customers and does not require previous terraform knowledge
- Single file represents the entire deployed environment and can include various aspects like project creation, service enablement, network, **image creation,** etc.
- Users define high-level dependencies (via **use** clause). The toolkit handles the complex variable passing.
- **Easy to share best practices, tutorials, guides**
- Allows for fast and easy modular development. **Iterative modifications are incredibly simple** to implement. Switch from filestore to lustre by swapping out 1 line. New slurm partition in 6 lines. Monitoring dashboard: 4 lines…
- Benefit from continued investment in making blueprints simpler, improved usability and deployment (deploy command coming soon)

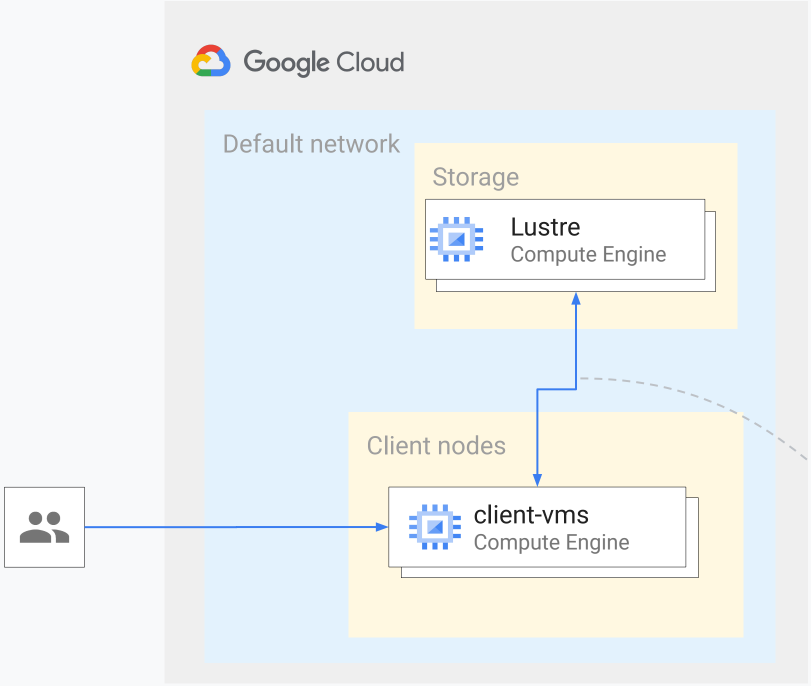
Increased supportability:

- Customers' entire environment is described
- Easy to reproduce and test

Expandable:

- Large ecosystem of pre-existing interoperable modules (several file systems, schedulers, etc.) that are tested together
- Addresses Terraform's last mile problem (Provisioners are a Last Resort). The toolkit integrates image building, infrastructure provisioning, software installation and configuration.

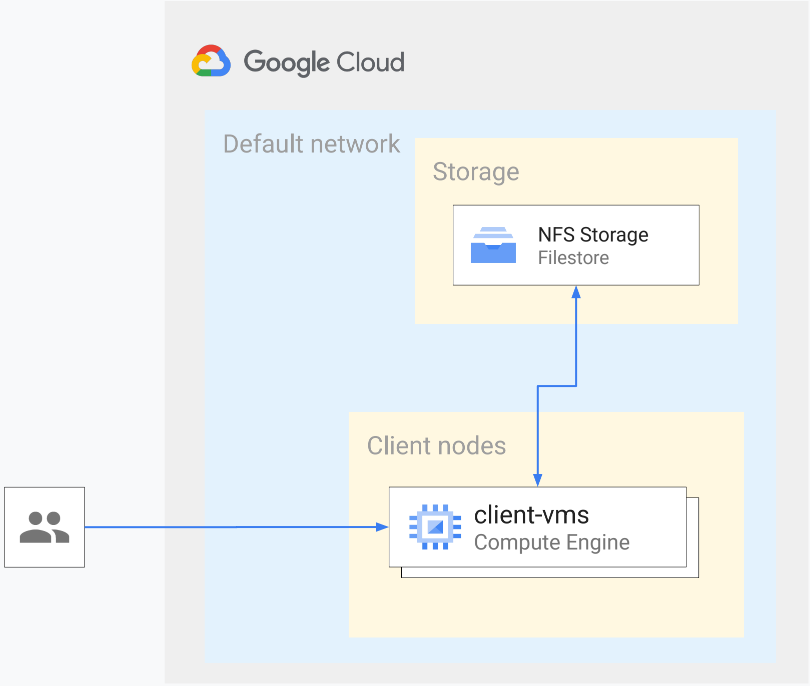
YAML as higher level abstraction




```yaml
blueprint_name: pfs

vars:
  project_id:  ## Set GCP Project ID Here ##
  deployment_name: small-pfs
  region: us-central1
  zone: us-central1-c

deployment_groups:
- group: primary
  modules:
  - id: network1
    source: modules/network/pre-existing-vpc

  - id: home
    source: community/modules/file-system/DDN-EXAScaler
    use: [network1]
    settings:
      local_mount: /home

  - source: modules/compute/vm-instance
    id: compute_instances
    use: [network1, home]
    settings:
      name_prefix: client-vm
      add_deployment_name_before_prefix: true
      instance_count: 2
      machine_type: n2-standard-2
```

dependency expressed easily, including drivers and os configuration

YAML as higher level abstraction




```yaml
blueprint_name: pfs

vars:
  project_id:  ## Set GCP Project ID Here ##
  deployment_name: small-pfs
  region: us-central1
  zone: us-central1-c

deployment_groups:
- group: primary
  modules:
  - id: network1
    source: modules/network/pre-existing-vpc

  - id: home
    source: modules/file-system/filestore
    use: [network1]
    settings:
      local_mount: /home

  - source: modules/compute/vm-instance
    id: compute_instances
    use: [network1, home]
    settings:
      name_prefix: client-vm
      add_deployment_name_before_prefix: true
      instance_count: 2
      machine_type: n2-standard-2
```

Elements of a blueprint: header

Name of the blueprint. It can be used to filter billing, monitoring, etc. (think of this as a “class” name)

```yaml
---

blueprint_name: hpc-slurm

vars:
  project_id:  ## Set GCP Project ID Here ##
  deployment_name: hpc-small
  region: us-central1
  zone: us-central1-a
```

Deployment variables are available to all modules, as long as they have an input variable that matches these names. They can be used to change instances of the blueprints and can be overridden via `--vars` flag (think object properties). Deployment name will determine the name of the deployment folder (output of `ghpc create`).

These four variables are almost always present, but only deployment_name is required..

```yaml
# TF state can also be set with  `--backend-config "bucket=${GCS_BUCKET}"`
terraform_backend_defaults:
  type: gcs
  configuration:
    bucket: your-tf-state-bucket-here
```

It is possible to set the terraform state of a deployment in a blueprint or with the `backend-config` flag.

https://cloud.google.com/cluster-toolkit/docs/setup/hpc-blueprint

Elements of a blueprint: Deployment groups

```yaml
# Documentation for each of the modules used below can be found at
# https://github.com/GoogleCloudPlatform/hpc-toolkit/blob/main/modules/README.md
deployment_groups:
- group: network-and-storage
  modules:
  # Source is an embedded resource, denoted by "resources/*" without ./, ../, /
  # as a prefix. To refer to a local resource, prefix with ./, ../ or /
  # Example - ./resources/network/vpc
  - id: network1
    source: modules/network/vpc

  - id: homefs
    source: modules/file-system/filestore
    use: [network1]
    settings:
      local_mount: /home
```

Groups define top-level terraform folders (and tf state)

Groups have a list of modules, which are building blocks of a deployment.

Modules can `use` other modules. This means: assign any input variables of “homefs” with the value of matching outputs of “network”

https://cloud.google.com/cluster-toolkit/docs/setup/hpc-blueprint

A note about modules

- Modules can be anywhere: the toolkit repo, any github repo, a local folder...
- `gcluster` does not need to know anything about the modules, we often call Cloud Foundation Toolkit terraform modules directly.
- Default values simplify configuration and promote consistency.
- We provide over 50 modules implementing schedulers, storage options, etc.
- It is easy to write your own modules. For best results, follow our guidelines.

https://github.com/GoogleCloudPlatform/cluster-toolkit/tree/main/modules#module-fields

https://github.com/GoogleCloudPlatform/cluster-toolkit/blob/main/docs/module-guidelines.md

HPC Deployment folder

- Generated by the gcluster command.
- Contains all the required artifacts for deploying an environment: terraform modules, scripts, etc.
- Self contained – can be tracked like source code and distributed
- A top-level folder for each deployment-group. One top-level main.tf for each deployment group.

```
hpc-cluster-simple/
├── packer
│   └── image100
│       ├── example.yaml
│       ├── image.pkr.hcl
│       ├── module.json
│       ├── README.md
│       └── variables.pkr.hcl
└── infrastructure
    ├── main.tf
    ├── modules
    │   ├── filestore
    │   ├── pre-existing-vpc
    │   ├── simple_instance
    │   └── startup-script
    ├── terraform.tfvars
    └── variables.tf
```

HPC Blueprints Catalog

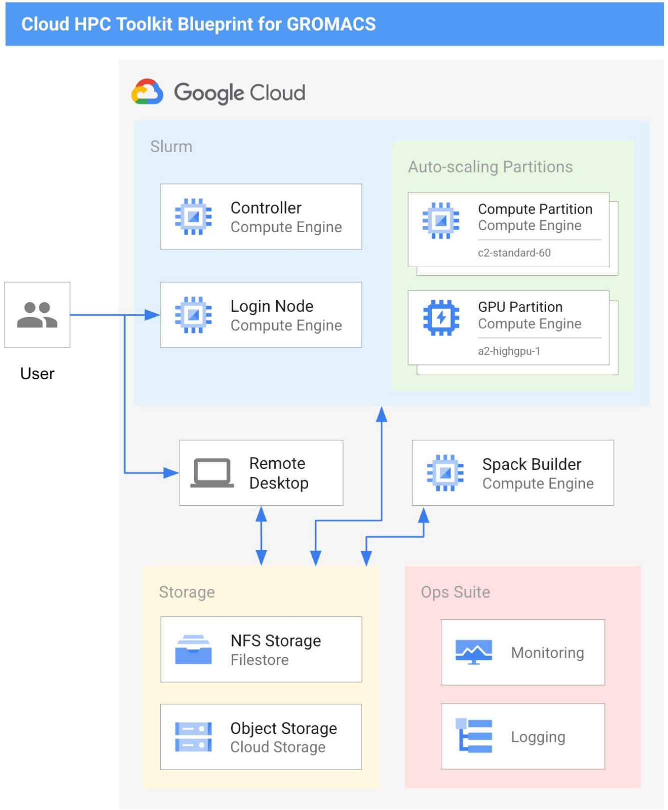
- Cluster Toolkit provides a list of curated and tested example blueprints
- Blueprint Catalog makes it easy to filter by
  - Software & Applications: Fluent, WRF, OpenFoam, Gromacs, QSim
  - Scheduler
  - Storage Type
  - Machine Type
  - Operating System
- Examples are ready to use as is or can be a jumping off point for building a custom environment
- Most examples are backed by nightly integration tests that ensure they will be functional out of the box.
- g.co/cloud/cluster-toolkit/docs/setup/hpc-blueprint-catalog
  - or google *"Cluster Toolkit blueprint catalog"*

HPC blueprint	Scheduler	Storage	Machine types	Base operating system	Featured software and compute resources	Collection
client-google-cloud-storage	None	Cloud Storage	E2	`hpc-centos-7`		Community, Experimental
hpc-amd-slurm	Slurm	Filestore	C2D	`hpc-centos-7*`	• AMD • AMD Optimizing C/C++ and Fortran Compilers (AOCC) • OpenFoam • Spack	Community
hpc-enterprise-slurm	Slurm	Filestore, Filestore High Scale, DDN-EXAScaler Lustre	N2, C2, C3, A2	`hpc-centos-7*`	• Google Virtual NIC (gVNIC) • GPUs • Performance persistent disks (pd-ssd) • Tier_1 networking	Core
hpc-gke	GKE	None	N2, C2	Containers		Community, Experimental
hpc-intel-select-slurm	Slurm	Filestore	C2, C3	`hpc-centos-7*`	• Intel Select Solutions	Community
hpc-slurm	Slurm	Filestore	N2, C2	`hpc-centos-7*`		Core
hpc-slurm-daos	Slurm	Intel DAOS	N2, C2	`hpc-centos-7*`	• DAOS file system	Community
hpc-slurm-gromacs	Slurm	Filestore	C2	`hpc-centos-7*`	• GROMACS • Spack	Community, Experimental
hpc-slurm-hcls	Slurm	Cloud Storage,	A2, C2	`hpc-centos-7*`	• GPUs	Community

Cluster Toolkit Blueprint for GROMACS

What was deployed?

- APIs enabled, VPC network created
- Storage
  - GCS Buckets, Filestore (NFS)
- Spack
  - Builder installs Intel MPI, GCC, and GROMACS to Filestore
- Slurm Cluster
  - VMs: Login Node, Controller
  - Auto-scaling Partitions: CPU, GPU (A2 + NVIDIA A100)
  - Storage Mounted
- Remote Desktop VMs with GPU Acceleration
  - Chrome Remote Desktop, VMD
  - Storage Mounted
- HPC Monitoring Dashboard



Cluster Toolkit GROMACS Blueprint

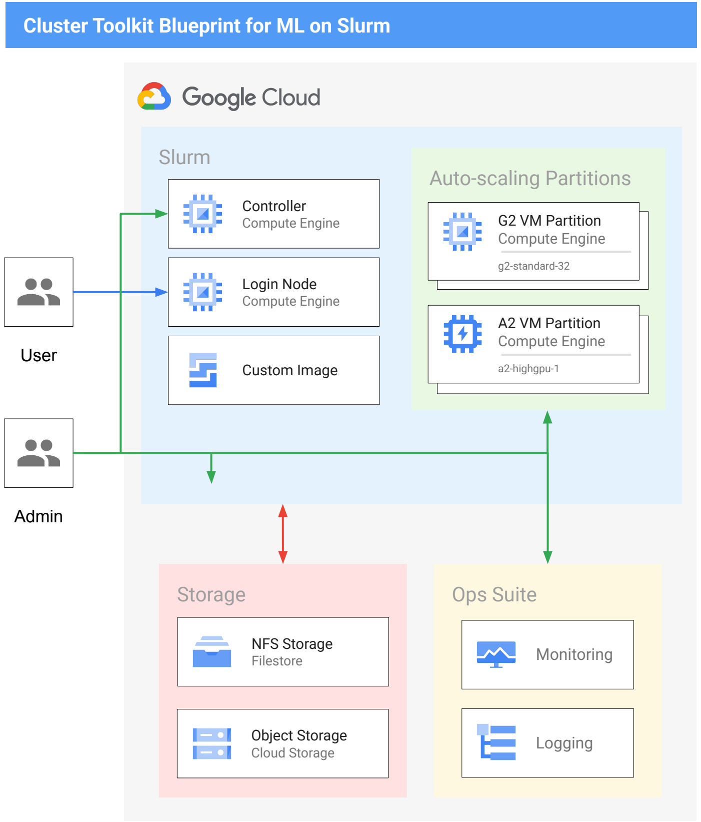
Cluster Toolkit GROMACS Demo Video

Cluster Toolkit Blueprint for EDA

Blueprint Components

- **Network** - Existing VPC network leveraged, gVNIC installed
- **Storage** - New Filestore (NFS) with 2TB of SSD storage
- **Cluster**
  - **Platform** - Google Compute Engine (GCE)
  - **Scheduler** - Slurm Workload Manager
  - **Environment** - Automatically installs Conda, Tensorflow, NVIDIA Drivers, CUDA, NCCL, TensorRT, Pytorch, and more
  - **Image** - Custom Debian image created by packer
  - **Management VMs** - Login Node, Controller
  - **Auto-scaling Partitions**
    - A2 VMs (1 x NVIDIA A100 GPUs), Compact Placement
    - G2 VMs (1 x NVIDIA L4 GPUs), Compact Placement

Cluster Toolkit
ML on Slurm Blueprint



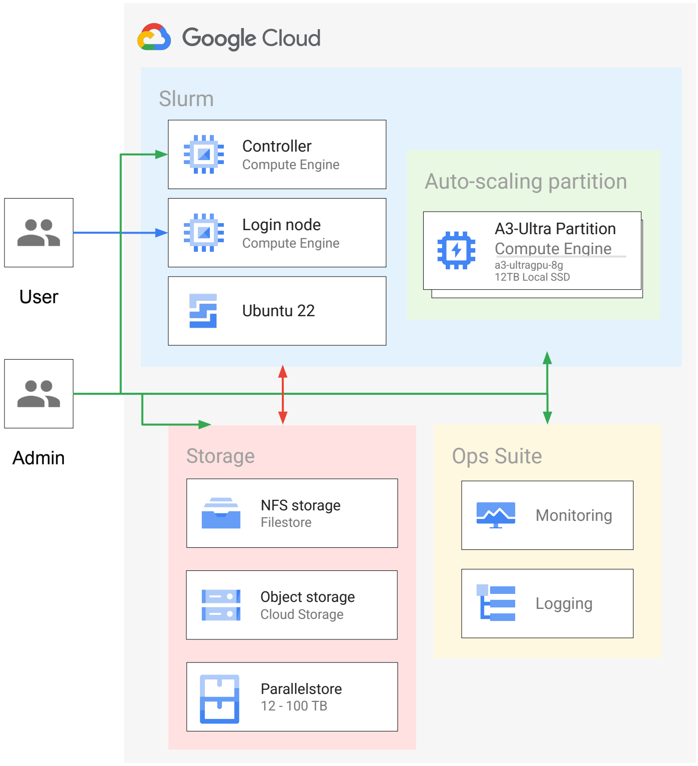
Blueprint for NeMo on A3-Ultra VMs with Slurm

Blueprint Components

- **Network** - 2 x VPCs for Host + 8x GPU NICS, gVNIC installed
- **Shared Storage** - Filestore (NFS)
- **Object Storage** - Google Cloud Storage via Cloud Storage FUSE
- **Parallel Filesystem -** Parallelstore (Intel DAOS) 12-100TB
- **Cluster**
  - **Scheduler** - Slurm Workload Manager
  - **Environment** - Automatically installs
  - NVIDIA Drivers, CUDA, Nvidia Enroot / Pyxis
  - **Image** - Ubuntu 22 available; Rocky 8 in development
  - **Management VMs** - Login node, Controller node
  - **Compute partition**
    - **A3-Ultra VMs**
      - 8 x NVIDIA H200 GPUs
      - 3600 Gbps total bandwidth
      - 12TB Local SSD per VM
    - Compact Placement
    - Automatic topology-awareness scheduling

A3-Ultra Cluster Toolkit Blueprint

Cluster Toolkit blueprint for A3-Ultra on Slurm



How software is configured

- **Startup scripts**: Shell / Data or ansible, they automate specialization after boot. Many playbooks available as flags: driver installation, docker configuration, monitoring, disk setup, etc.
  - Updates happen at redeployment. Best for things that don't change or ephemeral systems.
- **Image building**: support for packer modules to build images as part of blueprint. Leverages startup script to make "runtime or image building interchangeable".
  - Since compute nodes are basically ephemeral, rolling update of compute images is easy.
- **Spack**: spack modules leverage startup scripts to automate common spack and Ramble tasks.
- **Containers**: Slurm + Enroot has surged in popularity and most AI workloads now leverage containers.
  - Mostly unmanaged, most users pull / build / run their own. Best quality of living for the user.

“Using the Cluster Toolkit, we can now create an HPC cluster in Google Cloud in a matter of minutes.”

Adrian Tate, CEO, *NAG*

“The Cluster Toolkit reduces complexities and improves automation while mitigating errors for HPC in the cloud.”

Suresh Andani, Director, Cloud Business Development, *AMD*

Try it yourself

Cluster Toolkit > Documentation

Deploy an HPC cluster with Slurm

This document describes how to deploy an HPC cluster with Slurm in the Google Cloud console.

To follow step-by-step guidance for this task directly in the Google Cloud console, click **Guide me**:

Guide me

---

https://cloud.google.com/cluster-toolkit/docs/quickstarts/slurm-cluster

Feedback and Questions?