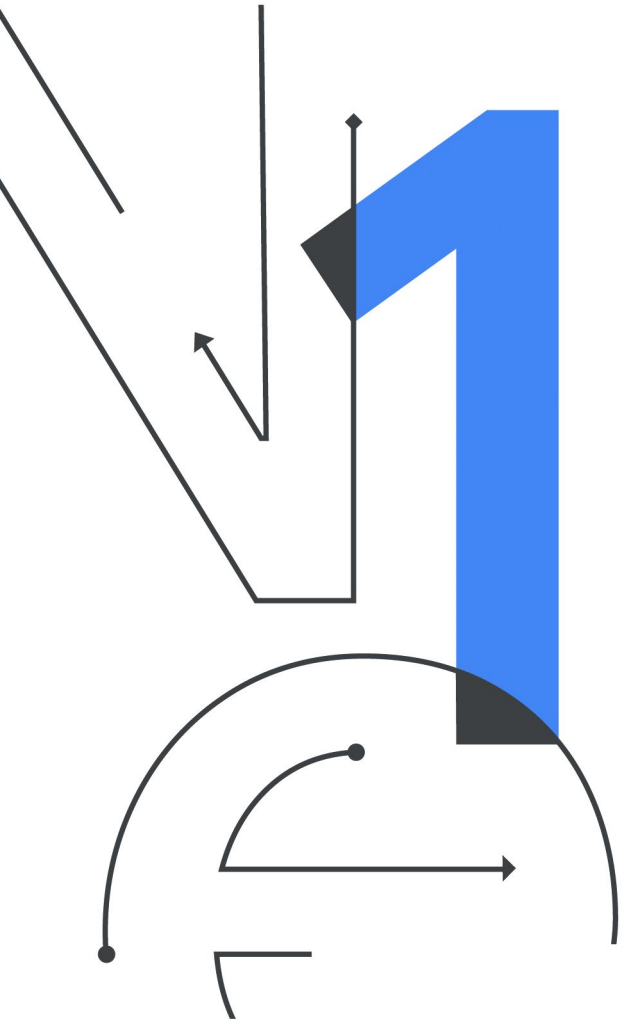
High Performance Computing on GKE

Cristian Mezzanotte
Google Cloud Customer Engineer

Contents



GKE Overview

Why do customers choose Google Kubernetes Engine (GKE)?

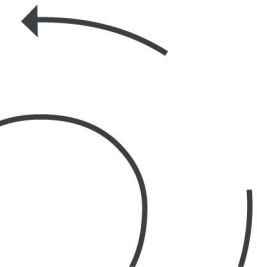
Quick Up and Running

Simplified Management

Optimized workloads

Scale to any work load

Open



Not all Kubernetes are created equal

Highly managed, Low customer effort

GKE

Customers who **don't want to worry about the complexity** of kubernetes and want Google to handle deployment, scaling and management

Unmanaged, High customer effort

OSS K8s on GCP

OSS K8s on your own

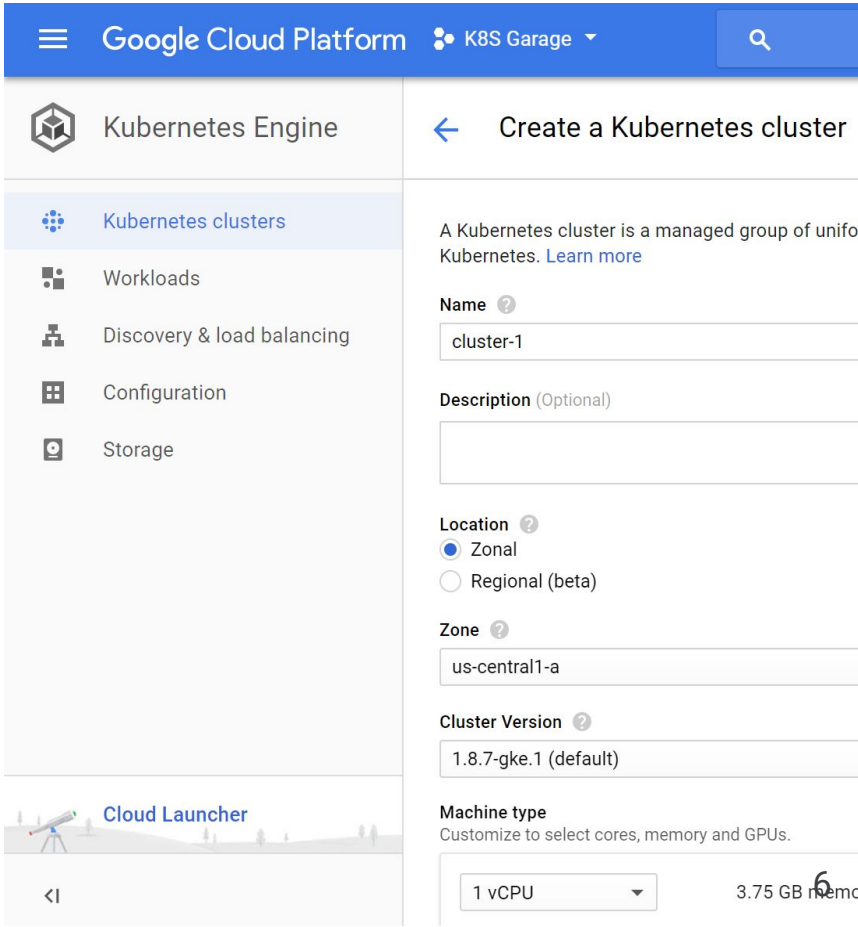
Expert Kubernetes users who have **time to manage and configure various settings and deployment**, and are not looking to leverage any integrations nor take advantage of Google SRE

Kubernetes the Easy Way

Start a cluster with one-click

View your clusters and workloads in a single pane of glass

Google keeps your cluster up and running



GKE makes scaling easy

GKE simplifies automated scaling with multiple offerings

Vertical Pod Autoscaling

Watch resource utilization of your deployments and adjust requested CPU and RAM to stabilize the workloads

Node Auto-Provisioning

Optimizes cluster resources with an enhanced version of Cluster Autoscaling

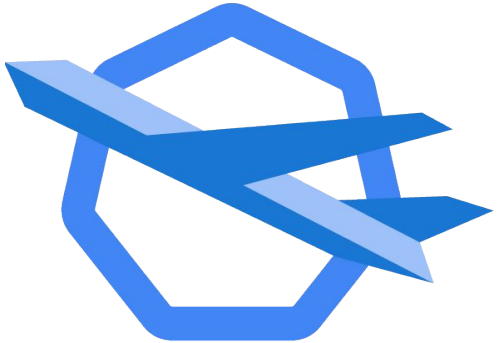
15K+ nodes on GKE

Scale to meet the needs of any workload

GKE Autopilot

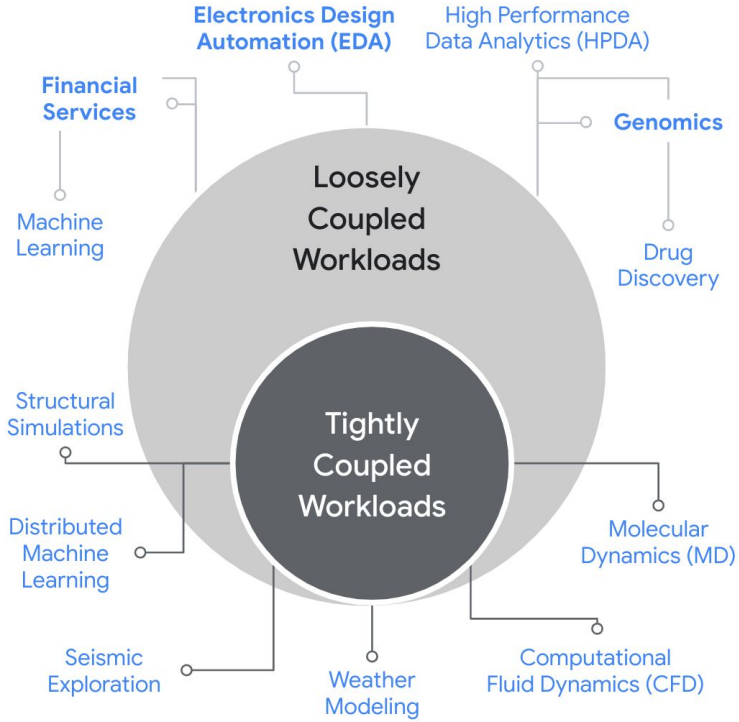
Fully Managed and Optimized for Production

- Optimized for production by K8s experts
- **SLA** on control plane, nodes and **Pods** (all monitored by Google)
- Secure by default with hardening guidelines implemented
- Resources provisioned based on workload
- It’s still Kubernetes, still GKE

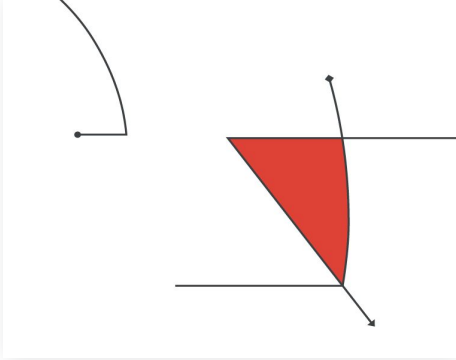


HPC

High Performance Computing



Does HPC for GKE make sense?



- Yes! :)
- Multi-tenant, secure, cost efficient, scalable, open source
- It’s a $XXX M business already
- Growing +70% YoY
- But…
- It’s not for everyone and all use cases
- Only for containerized workloads
- Rich networking tools for microservices lead to performance challenges in MPI

Scalability and performance

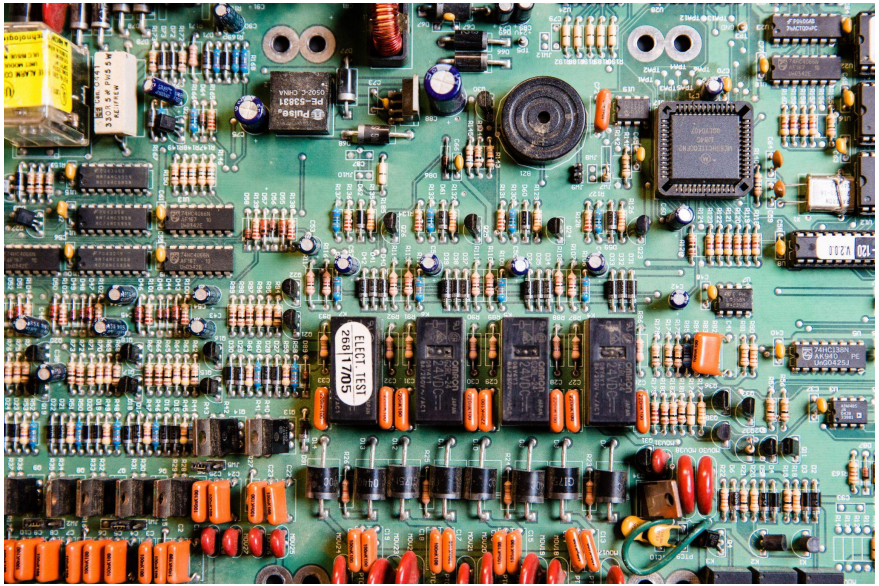
Mega-clusters - 15 000 nodes

Use smaller VMs - 1-2 worker pods

100 pods/s peak throughput

30-50 pods/s as average*

Image Streaming



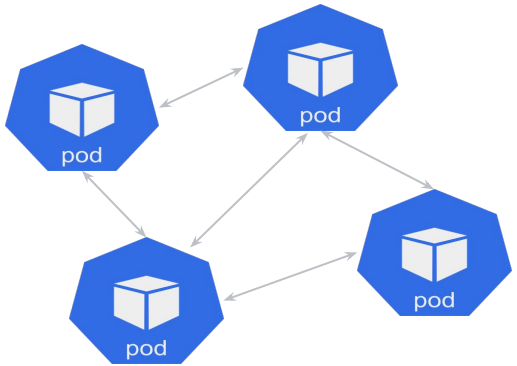
https://cloud.google.com/kubernetes-engine/docs/best-practices/scalability#gke-limits

Tightly coupled workloads

Use key settings - optimized network latency

- Compact placement
- Host networking
- Busy polling
- Increase TCP memory
- gVNIC

Kubeflow's MPI Operator V2



Operation modes

Autopilot

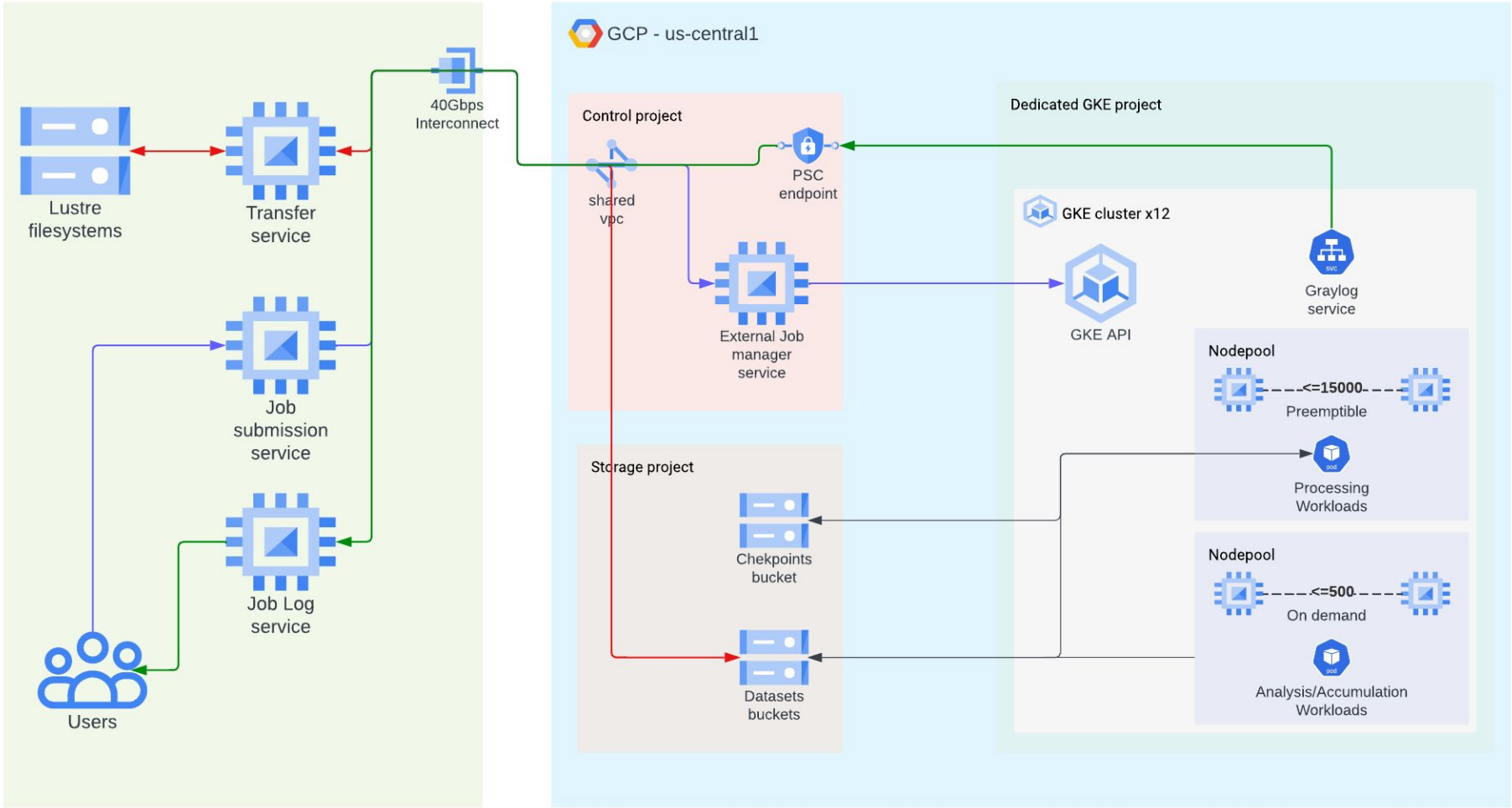
Google manages cluster configuration, including nodes, scaling, security.

Less Management Overhead

Standard

User determines cluster configuration

Custom configs


GCP - us-central1
Lustre filesystems
Transfer service
40Gbps Interconnect
Job submission service
Job Log service
Users
Control project
shared vpc
PSC endpoint
External Job manager service
Dedicated GKE project
GKE cluster x12
GKE API
Graylog service
Nodepool
<=15000
Preemptible
Processing Workloads
Storage project
Chekpoints bucket
Datasets buckets
Nodepool
<=500
On demand
Analysis/Accumulation Workloads

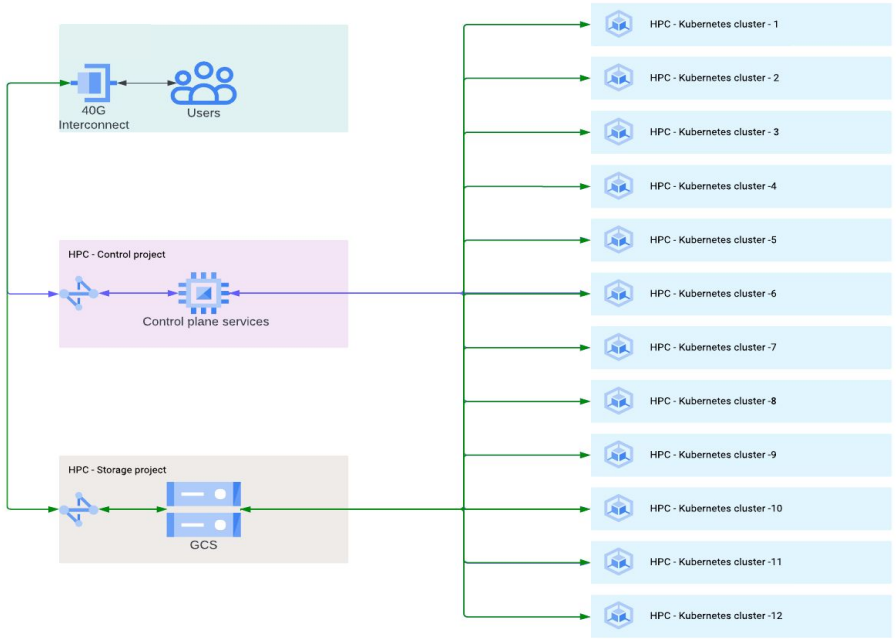
GKE as a supercomputer

Up to 2.8 Millions vcpu

A 7X capacity increase

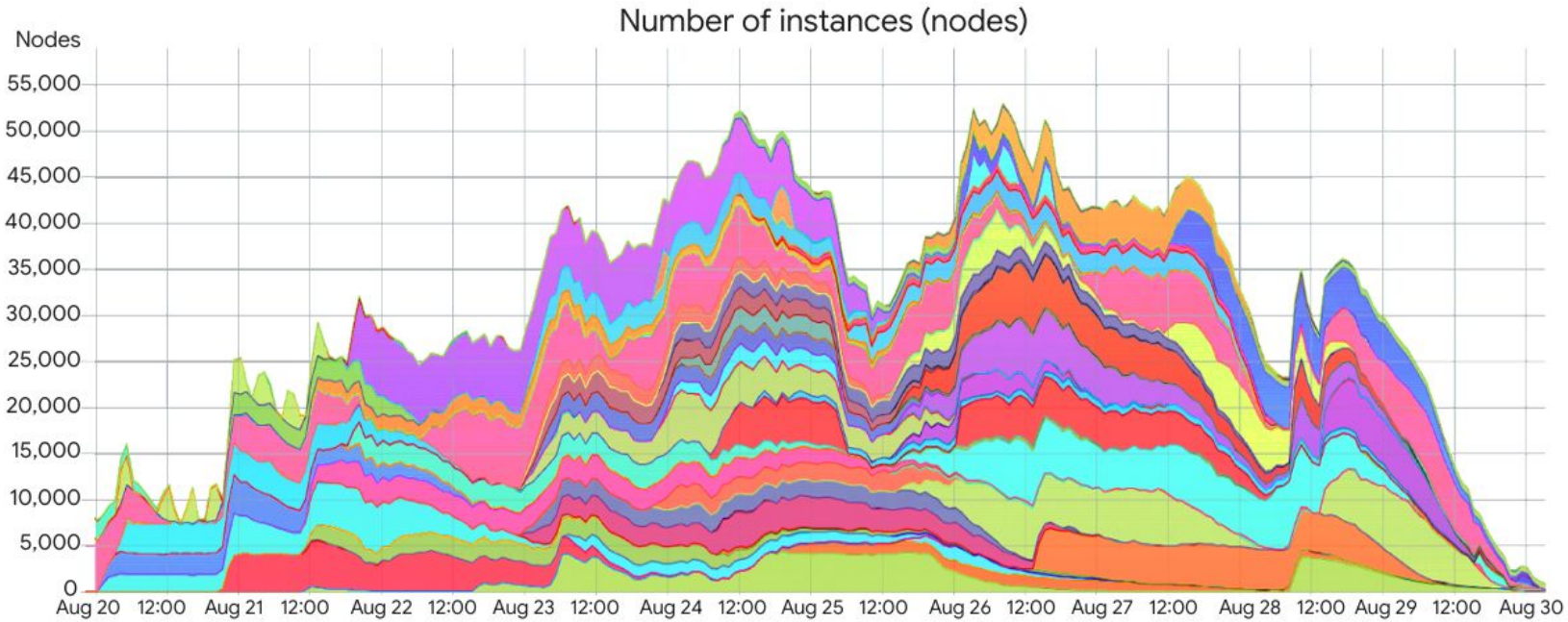
	Maximum	Peak
GKE nodes	180000	53417
Total vcpu	2.88 Millions*	855536*
Performance	72.02PFlops*	21.14PFlops*
Top500 (June 22)	Top 7	Top 24
GCS bandwidth	1.2 TBps	503 GBps

* using n2d-standard-16



Scale up to 53,417 instances

855,536 vCPUs



Bioinformatics

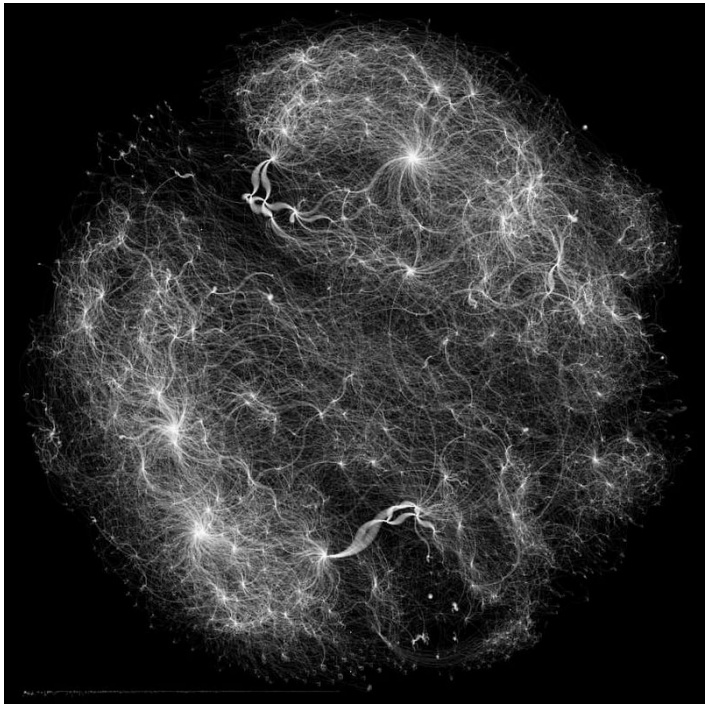
Bayer Crop Science runs a 240k vCPU cluster

Used GKE to build a single cluster with 15000 nodes and 240k CPU

~⅔ in Spot VMs for cost efficiency

Integrated with Pub/Sub, Spanner and GCS

Blog post ->



Tools

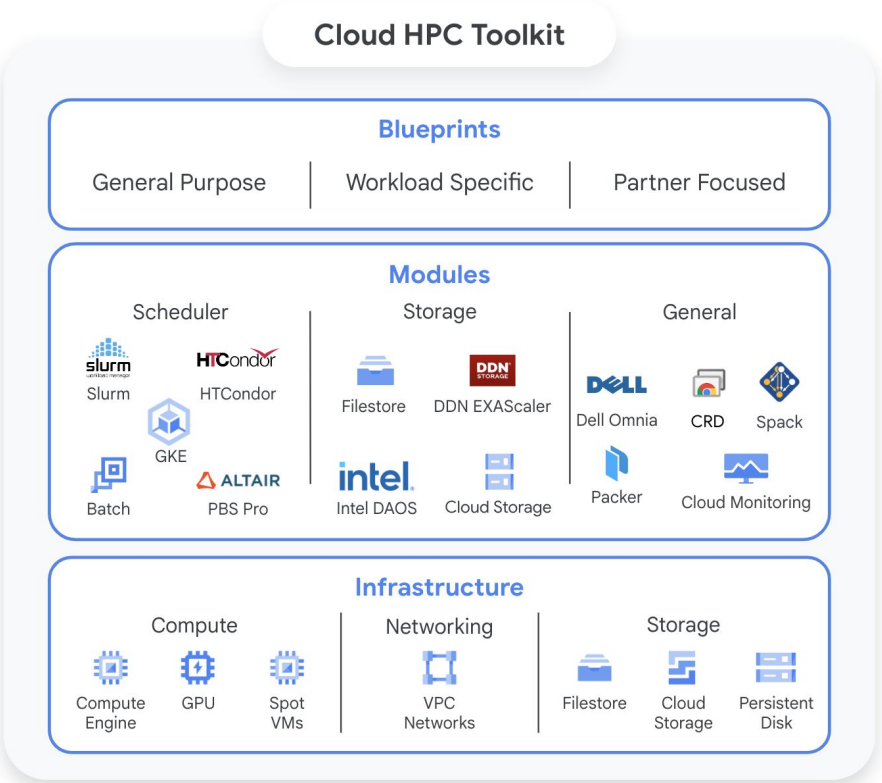
Cloud HPC Toolkit supports GKE

HPC Best Practice

The Cloud HPC 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 Cloud HPC Toolkit supports the core Google Cloud services and features that are required for HPC.



Example: Blueprint

```yaml
vars:
  project_id:  ## Set GCP Project ID Here ##
  deployment_name: hpc-small
  region: us-west4
  zone: us-west4-c

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

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

 - id: debug_node_group
    source: community/modules/compute/schedmd-slurm-gcp-v5-node-group
    settings:
      node_count_dynamic_max: 4
      machine_type: n2-standard-2

- id: compute_node_group
    source: community/modules/compute/schedmd-slurm-gcp-v5-node-group
    settings:
      node_count_dynamic_max: 20
      machine_type: c2-standard-60
```

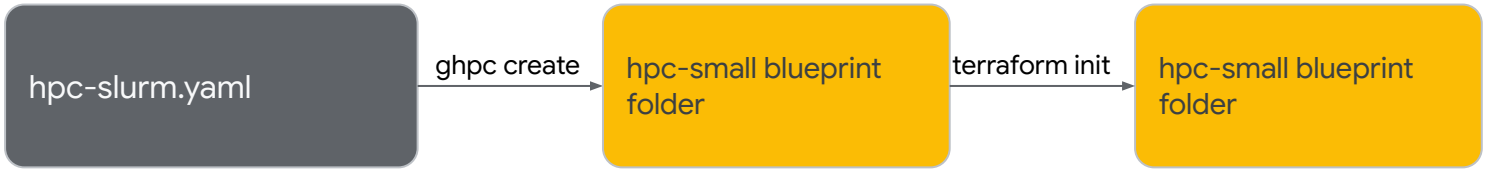
```yaml
- id: debug_partition
    source: community/modules/compute/schedmd-slurm-gcp-v5-partition
    Use: [network1, homefs, debug_node_group]
    settings:
      partition_name: debug
      enable_placement: false
      is_default: true

- id: compute_partition
    Source: community/modules/compute/schedmd-slurm-gcp-v5-partition
    use: [network1, homefs, compute_node_group]
    settings:
      partition_name: compute

  - id: slurm_controller
    source: community/modules/scheduler/schedmd-slurm-gcp-v5-controller
    use: [network1, homefs, debug_partition, compute_partition]
    settings:
      disable_controller_public_ips: false

  - id: slurm_login
    source: community/modules/scheduler/schedmd-slurm-gcp-v5-login
    use: [network1, slurm_controller]
    settings:
      machine_type: n2-standard-4
      disable_login_public_ips: false
```

Simplifying HPC Deployments on GCP

HPC Toolkit File
creates 1 file system a slurm cluster and 2 partitions

Generated folder before terraform init
(before external dependencies)

Generated folder after terraform init
(with external dependencies)



62 lines of YAML

What customers have to understand with the toolkit

2397 lines of HCL
895 lines of Markdown
267 lines of YAML
67 lines of shell, python, etc
3626 lines total

19160 lines of HCL
9591 lines of Markdown
5160 lines of Python
3309 lines of YAML
3049 lines of shell, python, etc
40269 lines total

What customers have to understand without the toolkit

How do we **map a Batch system** components to **GKE** and Kubernetes?

Dynamic Workload Scheduler (DWS)

New obtainability capabilities for accelerators

Works across GCP

- Managed Instance Groups on GCE
- Batch on GCE
- GKE
- Vertex AI

Calendar Mode:
Job start times assurance with Future Reservations

Use Cases:
(re)training, recurring fine-tuning

GPUs

Flex Start Mode:
Optimized economics and higher obtainability for on-demand resources

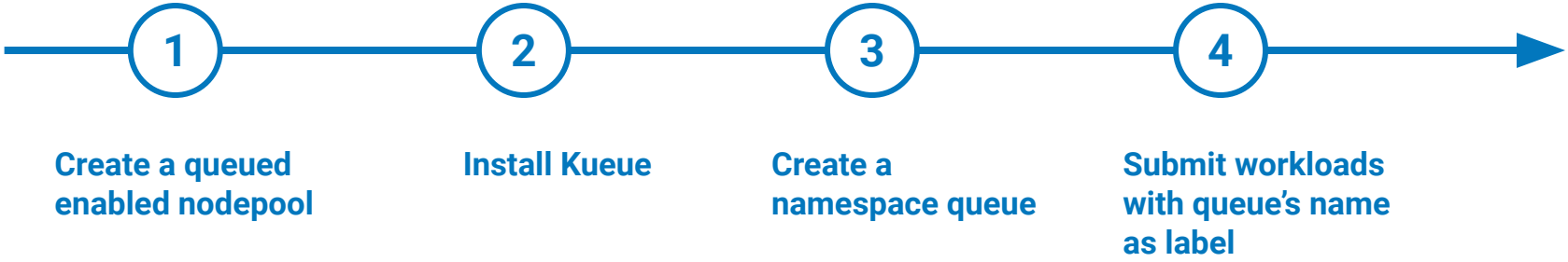
Use Cases:
time flexible experiments, fine tuning, batch inference

GPUs & TPUs

> *"The new DWS scheduling capabilities have been a game-changer in procuring sufficient GPU capacity for our training runs. We didn't have to worry about wasting money on idle GPUs while refreshing the page hoping for sufficient compute resources to become available."*
>
> **- Sahil Chopra, Co-Founder & CEO, Linum AI**

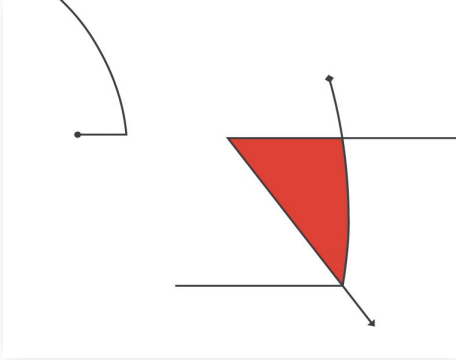
Kueue will orchestrate all interactions with underlying APIs

Just install and run workloads with a special label



- ✓ The easiest way to consume DWS on GKE is via Kueue
- ✓ All resources provisioned at once - no incomplete provisioning
- ✓ Pay only for resources consumed - no cost for queueing
- ✓ All orchestration automated - retries, interactions with GCE APIs, monitoring releasing workloads, VM provisioning

File Storage : Filestore



Filestore Basic	Filestore High Scale[Beta]
Designed for common enterprise storage needs	Designed For HTC & batch compute where high capacity and performance are critical
Enables File sharing, Software Dev, Web Hosting, Basic AI	Enables Electronic Design Automation (EDA), Advanced AI, Media rendering & transcoding

AI/ML

Open and Scalable AI Infrastructure

to cost-effectively run AI workloads at scale

Flexible Hardware

Support diverse ML workloads with varying hardware needs. Flexibility to use CPUs, GPUs, or TPUs. Strong partnership with Intel, AMD, & NVIDIA.

OSS Software Ecosystem

Make the most of ML with an open software ecosystem. Easily use open source tools, frameworks, & APIs. Run workloads via VMs, containers, or managed services.

Speed & Performance

Deliver results faster. Set up ML environments quickly, automate orchestration, manage large clusters, and set up low latency applications.

Workbench

Training

Serving

State of the Art AI

Manage complexity and innovate. Unlock more value from infrastructure with state of the art AI.

Reduction Server (70% increased throughput)

Optimized TensorFlow Runtime

Matching Engine (few millisecond ANN)

Vizier Hyperparameter Tuning

Compute Engine

Kubernetes Engine

BigQuery

Bigtable

Dataflow

Spark

Vertex AI

Unified AI/ML Platform on GKE

Team 1 | Team 2 | Team 3 | Team 4 | Team 5

Build | Train | Deploy

Tools and Libraries

jupyter | TensorFlow | PyTorch | NVIDIA CUDA | XGBoost | NVIDIA | DASK

Distributed Computing Frameworks

RAY | NCCL

Workflow and Data Processing

Kubeflow | Spark | beam | Apache Airflow

Custom Frameworks

GKE

Kueue: Kubernetes-native Job queuing

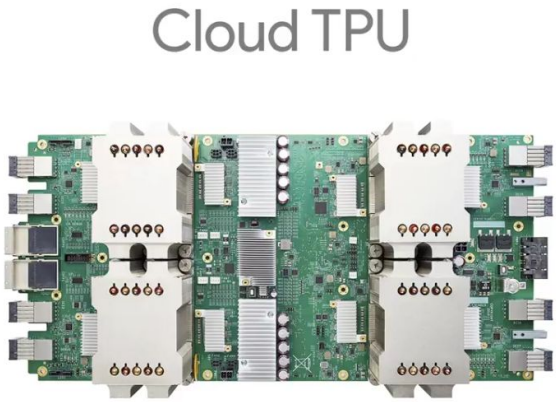
Autoscaling | Placement | Provisioning

Multi-Instance | TimeSharing | Local SSD | GCS Fuse | Fast Socket | gVNIC

Compute | GPU | TPU | Storage | Network

Cloud TPUs

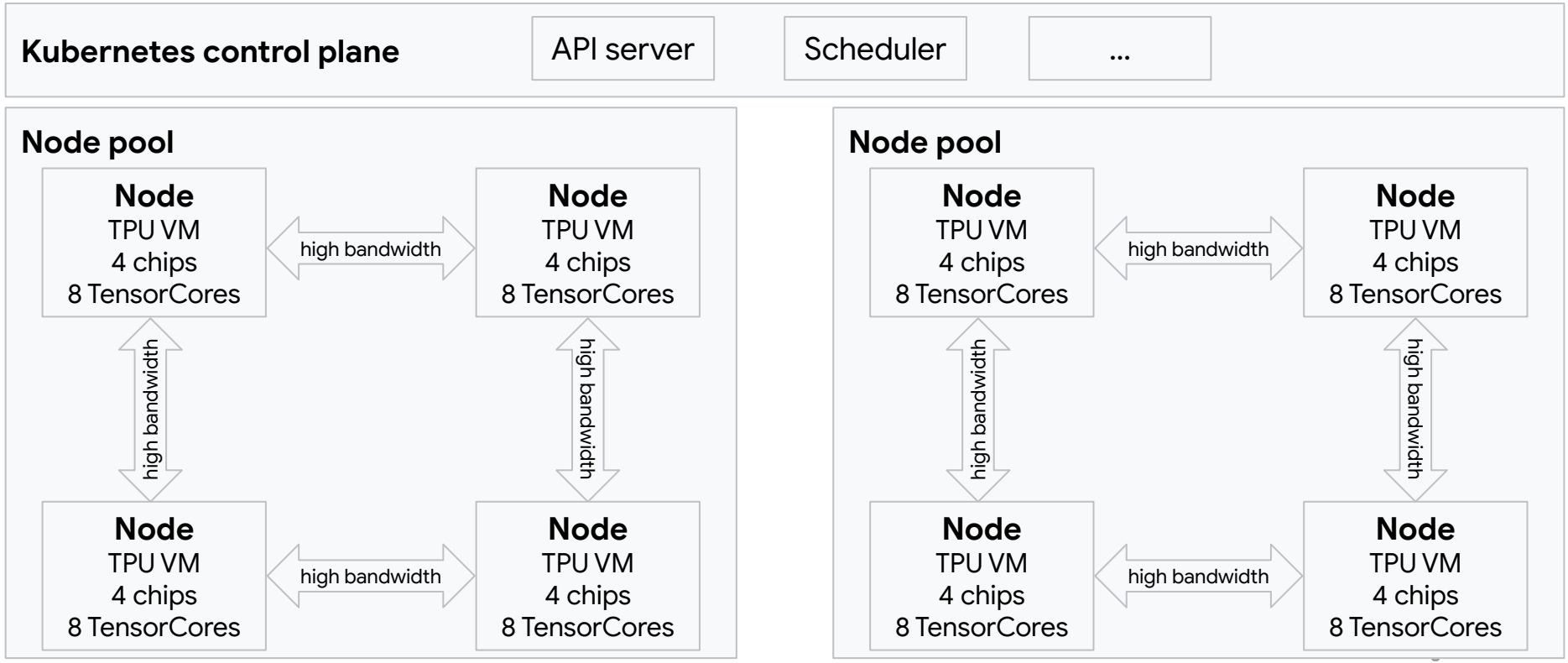
- **Faster training:** Minimize time-to-accuracy[1]
- **Industry-leading interconnect:** 6Tbps per host interconnect to speed up training
- **Flop-per-dollar gains:** 2.2x more peak FLOPs and ~1.4x more peak FLOPs per dollar vs TPU v3
- **Exceptionally high utilization** of these FLOPs at scale up through thousands of TPU v4 chips
- **Sustainable:** ~90% zero-carbon energy footprint in our us-central2 data center, containing TPU v4



[1] Source: https://cloud.google.com/blog/products/ai-machine-learning/cloud-tpu-v4-mlperf-2-0-results

Use GKE Node Auto-provisioning to dynamically spin up new TPUs

Example: If another workload needs a TPU v4 with 2x2x4 topology, GKE will spin up a second node pool.



New A3 VM with NVIDIA H100 GPU

VM Overview

2x Intel Sapphire Rapids CPUs
Up to 208 vCPUs
2TB system memory, 2x4TiB SSDs
8x NVIDIA H100 GPUs w/ NVLink
8x 200Gbps NICs

	NVIDIA A100	NVIDIA H100
GPU Architecture	NVIDIA Ampere	NVIDIA Hopper
FP32 Peak	19.5 TFLOPS	60 TFLOPS
FP16 Tensor Peak	312 TFLOPS	1000 TFLOPS
Memory	80GB HBM2e	80GB HBM3
Memory BW	2TB/s	3TB/s
NVLink BW	600GB/s	900GB/s

Sharing GPUs with GKE

Multi-instance GPUs

- Partition one A100 GPUs into up to 7 slices
- Hardware isolation from other containers on the same physical GPU
- Predictable throughput and latency for parallel workloads

Multi-process Service (MPS)

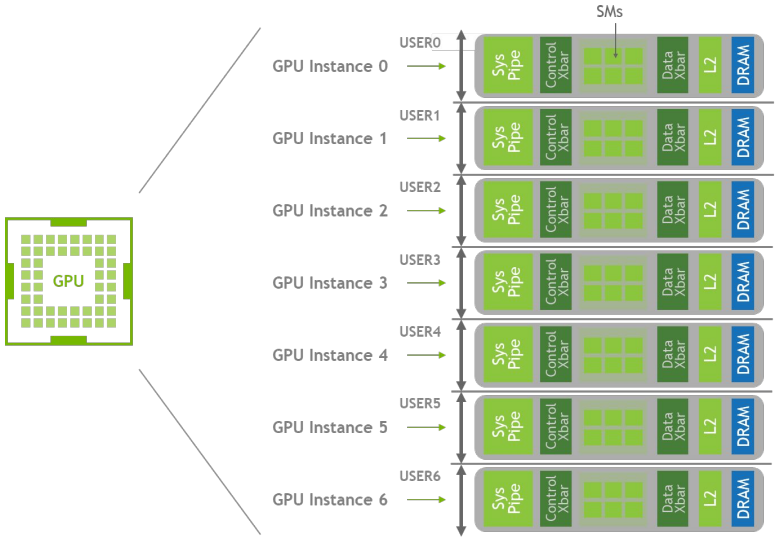
- Up to 48 parallelized partitions
- Software based isolation works on any NVIDIA GPU
- Predictable throughput and latency for parallel workloads

GPU Time-Sharing

- Up to 48 fractional GPU units with time-context shifting
- Workloads with low GPU requests
- Burstable GPU workloads

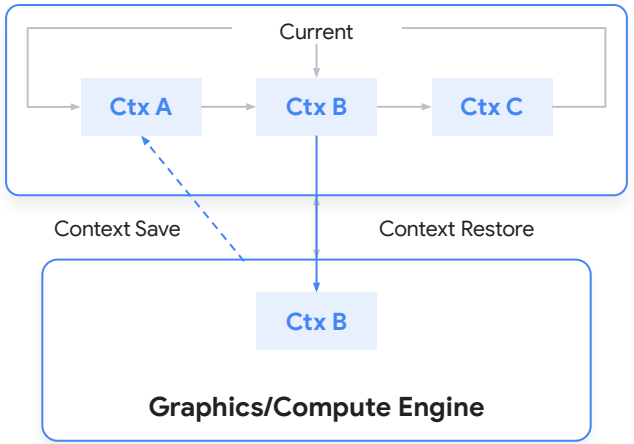
Multi-Instance GPUs

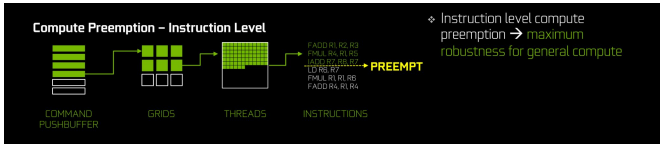
- Create multiple “GPU instances” on a single GPU
- Dedicated compute and memory for each GPU instance offers consistent QoS, throughput and latency
- Superior isolation compared to software based solutions
- Higher utilization of NVIDIA GPUs



GPU Time-sharing

- Cost-efficient utilization of resources as multiple containers can now share a GPU, eliminating over-provisioning and reducing total cost.
- GPU is allocated fairly to all containers with each container gets a timeslice.
- Unique to GKE as OSS Kubernetes does not allow fractional requests for GPUs.
- Works with all available NVidia GPUs

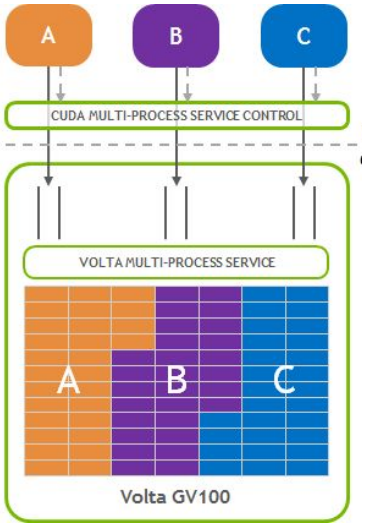




Private Preview

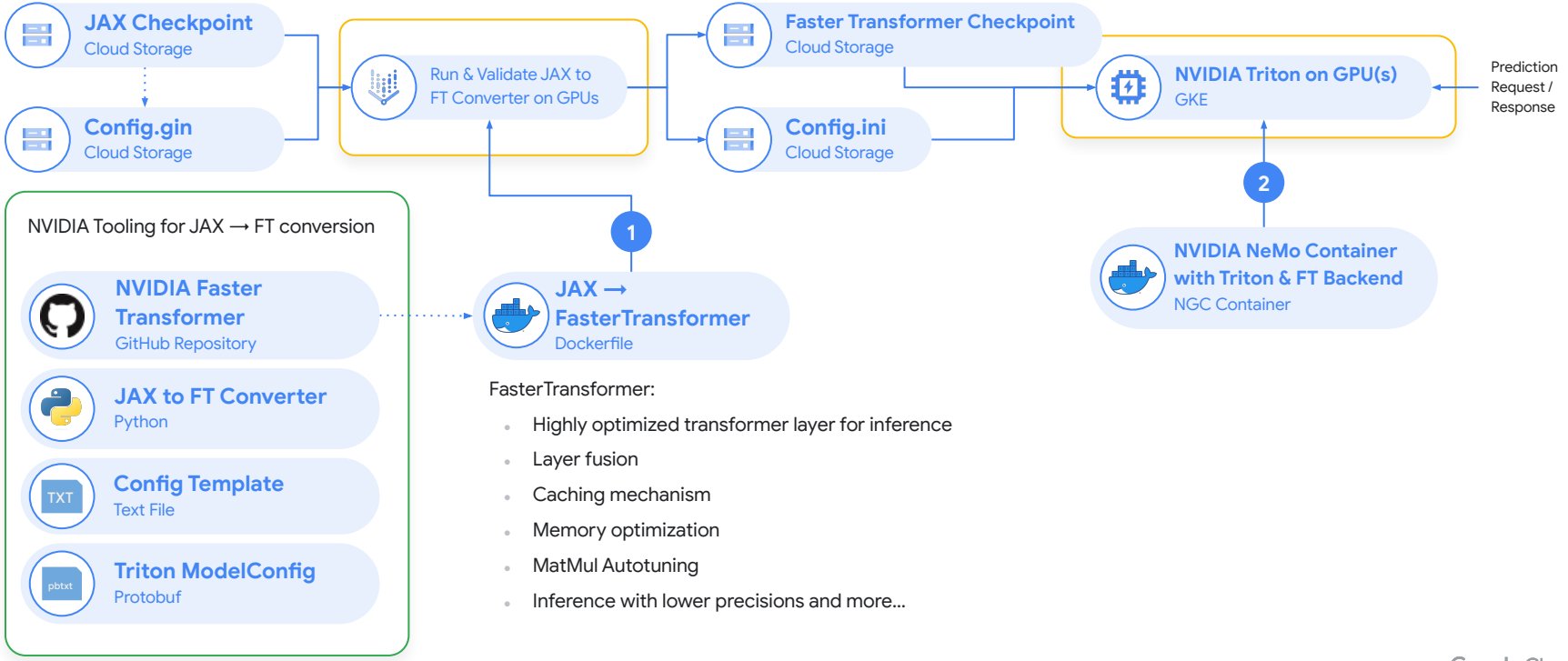
GPU Multi-Process Service

- Create multiple GPU partitions on a single GPU
- Workloads execute in parallel
- Dedicated compute and memory for each GPU partition offers consistent throughput
- Achieve higher utilization
- Reduced context switching
- Works with all NVIDIA GPUs



Large Language Models Serving

JAX → NVIDIA FasterTransformer (FT) and Serve with **Triton Inference Server**



FasterTransformer:

- Highly optimized transformer layer for inference
- Layer fusion
- Caching mechanism
- Memory optimization
- MatMul Autotuning
- Inference with lower precisions and more...

Use Spot VMs

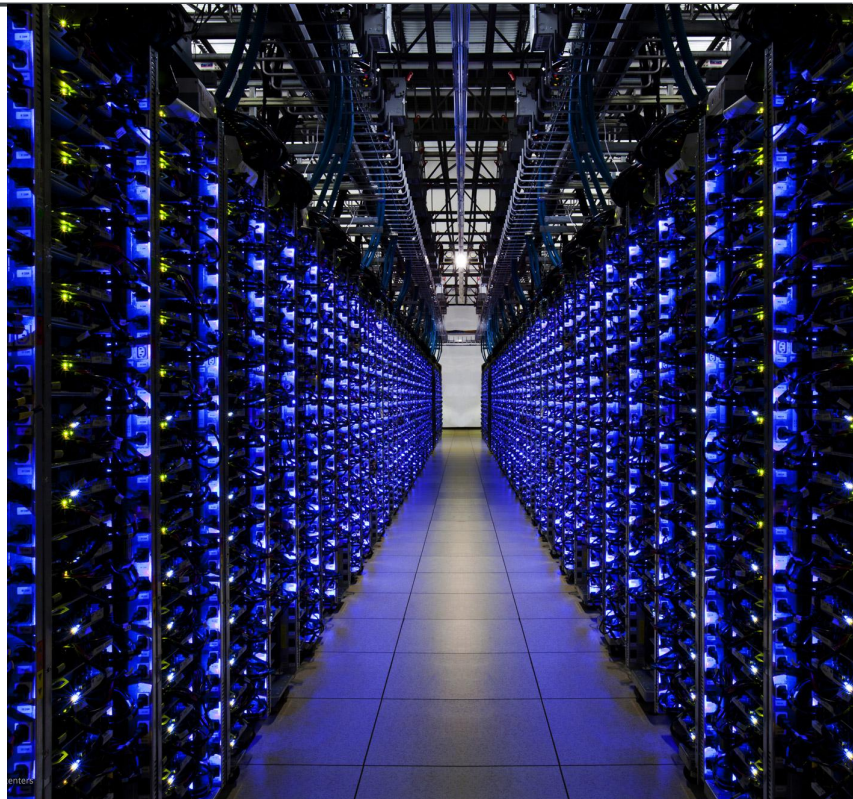
Up to 91% discount*

Checkpointing & graceful termination

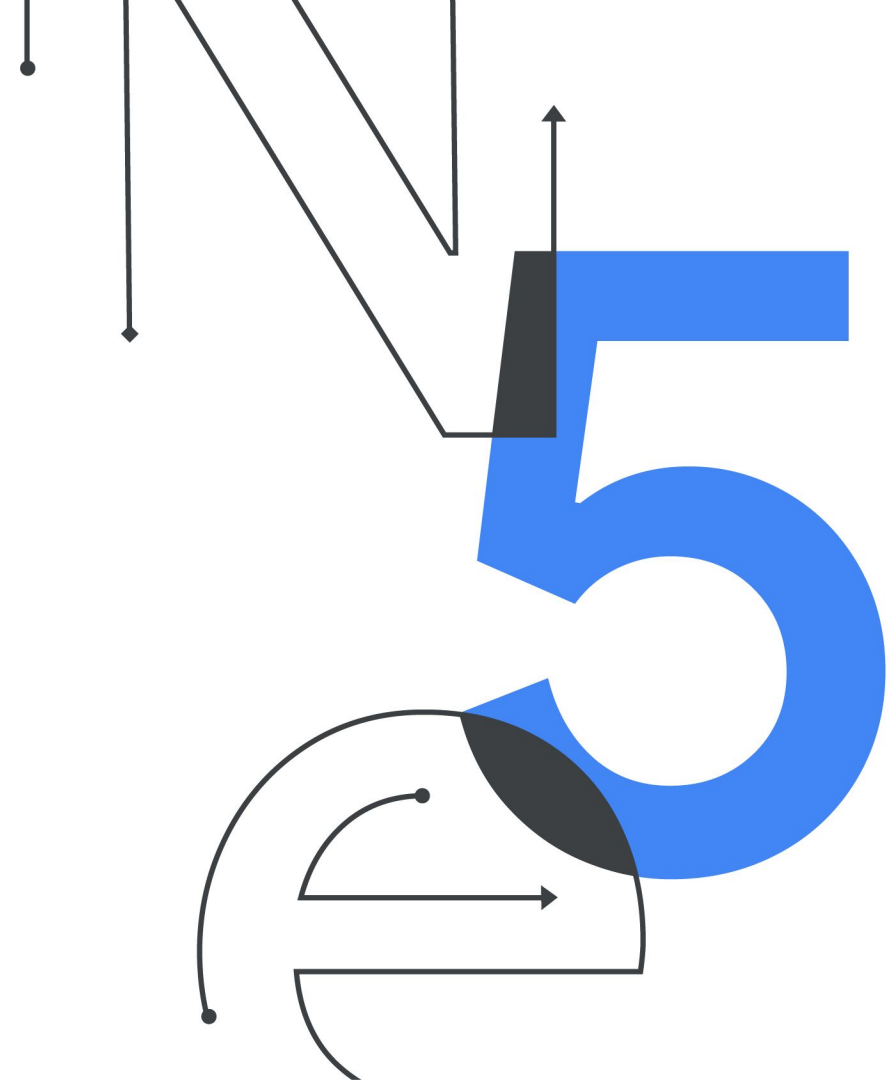
GKE automatically heals failed VMs

GKE seeks the cheapest VMs

*https://cloud.google.com/spot-vms



What’s next



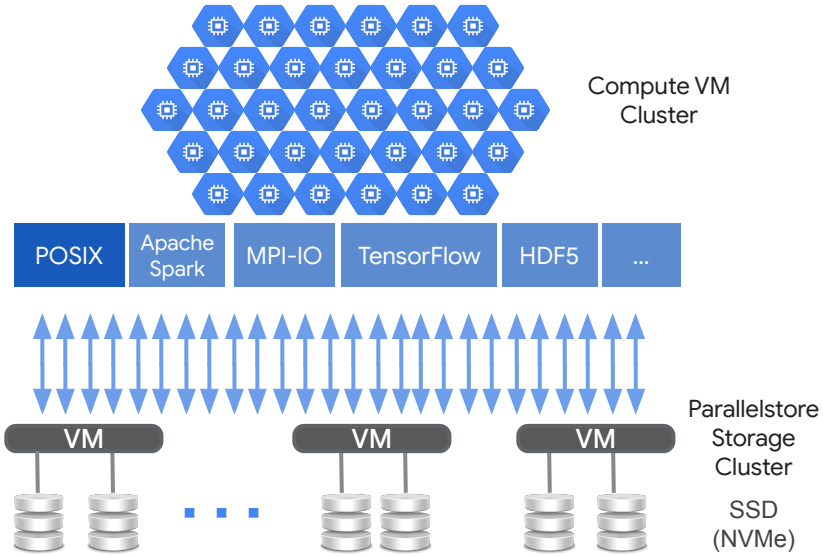
Parallelstore

Next Gen HPC Storage System
Now in Private Preview!

Accelerate HPC and AI workloads that require extreme scale and/or low latency I/O operations

Key Advantages

- Unique open-source DAOS storage architecture improves performance over existing POSIX storage options
- Standalone mode, or accelerate access to Cloud Storage
- Well-aligned to emerging patterns in AI workloads with distributed metadata, extreme IOPS, and K/V architecture
- Demonstrated >1GB/s per TiB, >1.5M create/s with open-source on GCP
    - Only limited by available resources, DAOS has demonstrated >8TB/s, 145M create/s in the real world!



Roadmap

- GKE support for A3 VM featuring NVIDIA H100 (GA)
- GKE supports Filestore Enterprise Backup (GA)
- Support for C3D VM (AMD EPYC GENOA)
- RDMA
- Parallelstore (Intel DAOS)

Thank you

Google Cloud
Next '22