Architecting Global GPU Availability with Multikueue

Speakers

Kishore Jagannath
Researcher, Blogger, Senior Customer Solutions Engineer at Google

Ram J A
Problem Solver, Solutions Architect at Google

KubeCon

Agenda

Architecting Global GPU Availability with Multikueue

01 **Introduction to Kueues & MultiKueue**
Foundation of job queueing and cluster-wide resource management.

02 **The Admission Process**
Deep dive into how MultiKueue dispatches workloads to worker clusters.

03 **The GPU Scarcity Problem**
Analyzing regional silos and the impact of fragmented utilization.

04 **The Solution & Live Demo**
Bridging regional scarcity through federation and real-time scaling.


GET IT! WE NEED COMPUTE!
GRUNT!
?!
Shhh! They don't see us!
Stay hidden!
A100
H100
TPUv4
A4000
GPU
TPU

Efficient Resource Management with Kueue

Optimizing Batch & Training Workloads

Kueue manages job queuing and resource quotas for complex AI/ML environments, ensuring fair access to high-demand hardware.

Multi-Team Collaboration

Enables multiple teams to share a common pool of GPUs and compute nodes without interference.

AI/ML Training

Orchestrates large-scale model training jobs that require dedicated, high-performance resources.

Batch Inference

Handles high-throughput inference tasks efficiently by managing job priority and allocation.

Resource Optimization

Prevents resource starvation and maximizes the utilization of limited datacenter GPUs.

Kueue Resources & Architecture

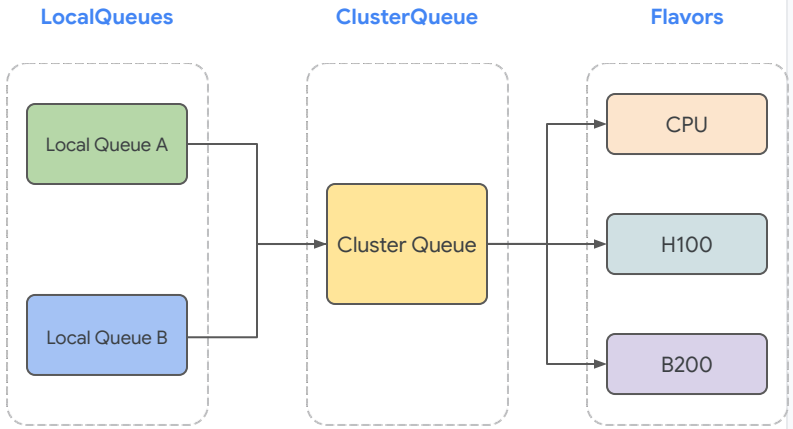
Hierarchical resource management for Kubernetes clusters

ResourceFlavor
Defines physical characteristics of resources (e.g., GPU models, spot vs. on-demand).

ClusterQueue
A cluster-scoped resource that manages a global pool of ResourceFlavors and sets fair-sharing rules.

LocalQueue
A namespace-scoped entry point for users to submit jobs, mapped to a specific ClusterQueue.

Resource Quota
Integrated mechanism to enforce limits on consumption across different teams and flavors.



Validation Flow:

Jobs enter through **LocalQueues** (Namespace scoped), pass through **ClusterQueues** (Global rules, cluster scoped), and consume specific **Resource Flavors** (Compute/ GPU Flavors)

Multi-Team Quota Enforcement with Kueue

Throttling resource usage across namespaces via ClusterQueue and LocalQueue

ResourceFlavor: n1-standard-8

Optimized for development costs. Features **N1 GPU** for general compute and model prototyping.

ClusterQueue: Development

Global Quota: **Shared Resource Pool**

Team Research

LQ: **research-lq**

Points to Dev CQ

Team Engineering

LQ: **eng-lq**

Points to Dev CQ

Quota: CPU 60, GPU 8, Mem 126Gi

Multiple teams share a limited quota defined at the cluster level, preventing resource hogging in development.

ResourceFlavor: a4-highgpu-8g

Mission-critical performance. Leverages **A4 series GPU** for high-throughput production inference.

ClusterQueue: Production

Dedicated Resource Quotas

- Strictly enforced limits for live traffic.
- Priority-based preemption disabled.
- Guaranteed GPU availability.
- **CPU: 64 | GPU: 8 | Mem: 256Gi**

LocalQueue: prod-lq

Single entry point for production workloads.

MultiKueue: Beyond Local Limits

MultiKueue unlocks global orchestration for Kubernetes batch and live workloads.

Global Accelerator Chasing

CROSS-REGION HARDWARE HUNTING

Dynamically dispatches batch AI/ML jobs to worker clusters across geographic regions the moment GPU/TPU capacity becomes available globally.

Unified Global Quotas

FAIR-SHARE & BORROWING LIMITS

Enforces hierarchical organizational quotas across a fleet of clusters, preventing any single team from monopolizing multi-region environments.

Topology Abstraction

SINGLE ENTRY POINT API

Researchers submit jobs to one centralized Manager Cluster. MultiKueue abstracts away complex topologies and manual targeting.

Global Spot Harvesting

AUTOMATED COST OPTIMIZATION

Opportunistically routes flexible batch workloads to whichever worker cluster currently possesses the cheapest preemptible compute capacity.

Scaling with MultiKueue Architecture

Extending job dispatching across multiple clusters via Manager-Worker topology

Kueue vs. MultiKueue

Kueue (Single Cluster)

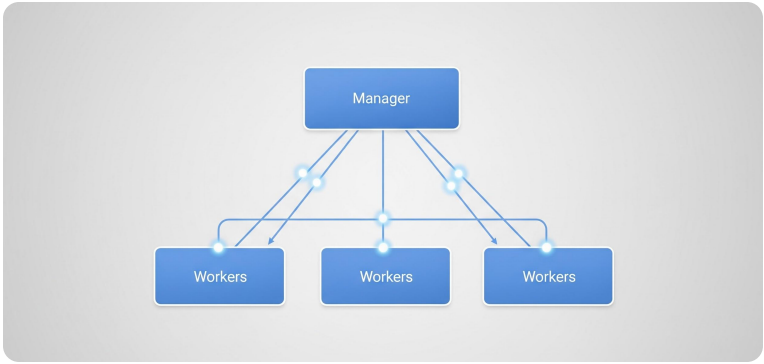
Manages local resource quotas and job queuing within a single Kubernetes control plane.

MultiKueue (Multi-Cluster)

Acts as a meta-orchestrator. It dispatches jobs from a central manager cluster to one or more worker clusters based on availability.

- **Centralized Quota:** Manage global limits while workers execute.
- **Job Offloading:** Automatically burst workloads to remote clusters.
- **Transparent to Users:** Submit to one LocalQueue; run anywhere.

Manager-Worker Topology



Manager: Hosts MultiKueue controller. Watches LocalQueues and creates 'remote' jobs.

Worker: Standard Kueue clusters. They receive the job, execute it within their local ClusterQueues, and report status back.

Workload Admission & Provisioning

Defining admission criteria and provisioning signaling via Admission Controllers

Multikueue

The central mechanism for managing and dispatching workloads across multiple distinct Kubernetes clusters.

- Evaluates workloads created on the Management Cluster to determine which remote Worker Clusters are suitable.

ProvisioningRequest

Bridges logical Kueue quotas with physical infrastructure. Coordinates Cluster Autoscaler APIs to guarantee node provisioning.

- **Best-effort-atomic-scaleup:** Fulfills batch demands by scaling hardware groups atomically.
- **Check-capacity-autoscaling:** Checks real-time physical resource and cloud capacity quotas.
- **Queue-dws-provisioning:** Coordinates pipelines natively with GKE's Dynamic Workload Scheduling API.

MultiKueue Deployment in Action

Dispatching workloads to regional worker clusters

Manager Cluster

ClusterQueue: prod-main

Global Quota Management

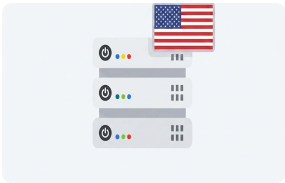
MultiKueue Controller

Admission Checks

Functionality:

- Aggregates regional demand
- Offloads to available workers
- Maintains state consistency

Worker Cluster 01: North America

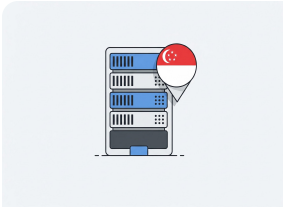


Region: us-central1

Local Cluster Queue

Admission Checks

Worker Cluster 02: Asia Pacific



Region: asia-southeast1 (Singapore)

Local Cluster Queue

Admission Checks

Problem Statement and Demo

The Regional Silo: GPU Scarcity

Single-region dependencies cripple AI availability and user experience

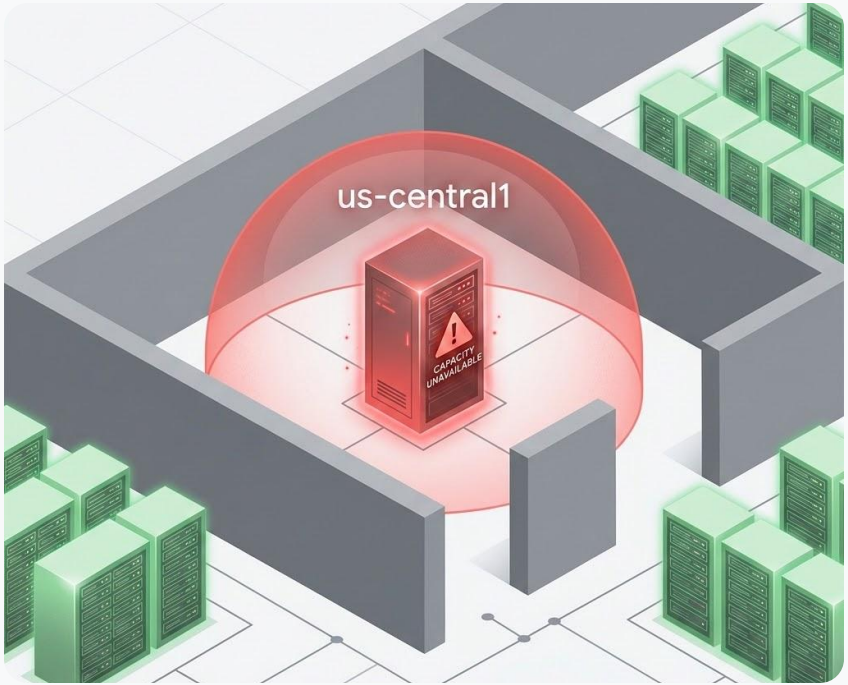
Critical Availability Bottlenecks

Regional Capacity Constraints:

- GPU capacity is locked to **local regional availability**.
- Clusters/VMs cannot tap into idle resources in adjacent REGIONS.

Impact on Operations:

- **Disrupted Jobs:** Training and inference jobs fail or stall due to "Insufficient Regional Capacity."
- **Poor CX:** High latency and service timeouts lead to degraded customer trust.
- **Innovation Block:** Teams wait weeks for hardware expansions instead of iterating.



Non Optimal Solutions

Traditional provisioning models struggle with the volatile nature of GenAI workloads

Inefficient Resource Management Strategies

1. Upfront Provisioning (High OpEx):

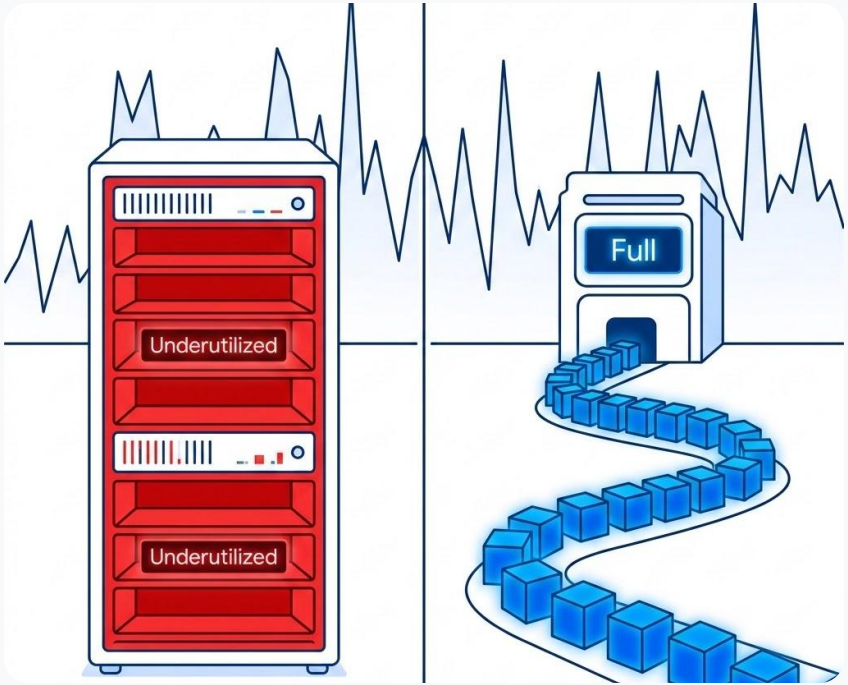
- Organizations reserve capacity to meet peak demands.
- Result: **Low utilization** and wasted spend during idle periods.

2. Queued Provisioning (High Latency):

- Workloads wait for regional availability.
- Result: **Time-to-market delays** and stalled innovation.

The Core Failure:

- Static models cannot adapt to **uneven traffic patterns**.
- Regional spikes cause local failures while global capacity sits idle.



Beyond Quotas: Hunting for Global GPU Capacity

Leveraging MultiKueue to bridge GenAI scarcity through regional federation

The Enterprise Challenge & Solution

The Scarcity Bottleneck: GenAI demand creates regional GPU fluctuations (us-central1 vs. asia-se1).

Traditional Trade-off: High latency (waiting) vs. High OpEx (overprovisioning).

The MultiKueue Objective: Grant batch inference workloads near real-time access to global GPUs during scale-out.

Architecture Approach:

- GKE Worker Clusters with GPU nodepools.
- Scale-from-zero logic enabled.
- Cluster Autoscaler for dynamic provisioning.
- Best-effort-atomic-class autiscaling was chosen as the admission controller for worker clusters

Regional Dispatch Failure: The SchedulingGated Issue

Manager commits to exhausted region while valid capacity remains in secondary worker with **Best-effort-atomic-scaleup** provisioning class

Manager Cluster

Dispatch Logic:

1. Observes global workload
2. Selects **us-central1** (Stale view)
3. **Job Pinned to Cluster A**

Worker Cluster A: us-central1 (North America)

Admission Controller
scale-up

Best-effort-atomic-scaleup

NO CAPACITY

Provisioning Status ACCEPTED

Result: Pod enters SchedulingGated

Stuck: No automated failover possible

Worker Cluster B: asia-southeast1 (Singapore)

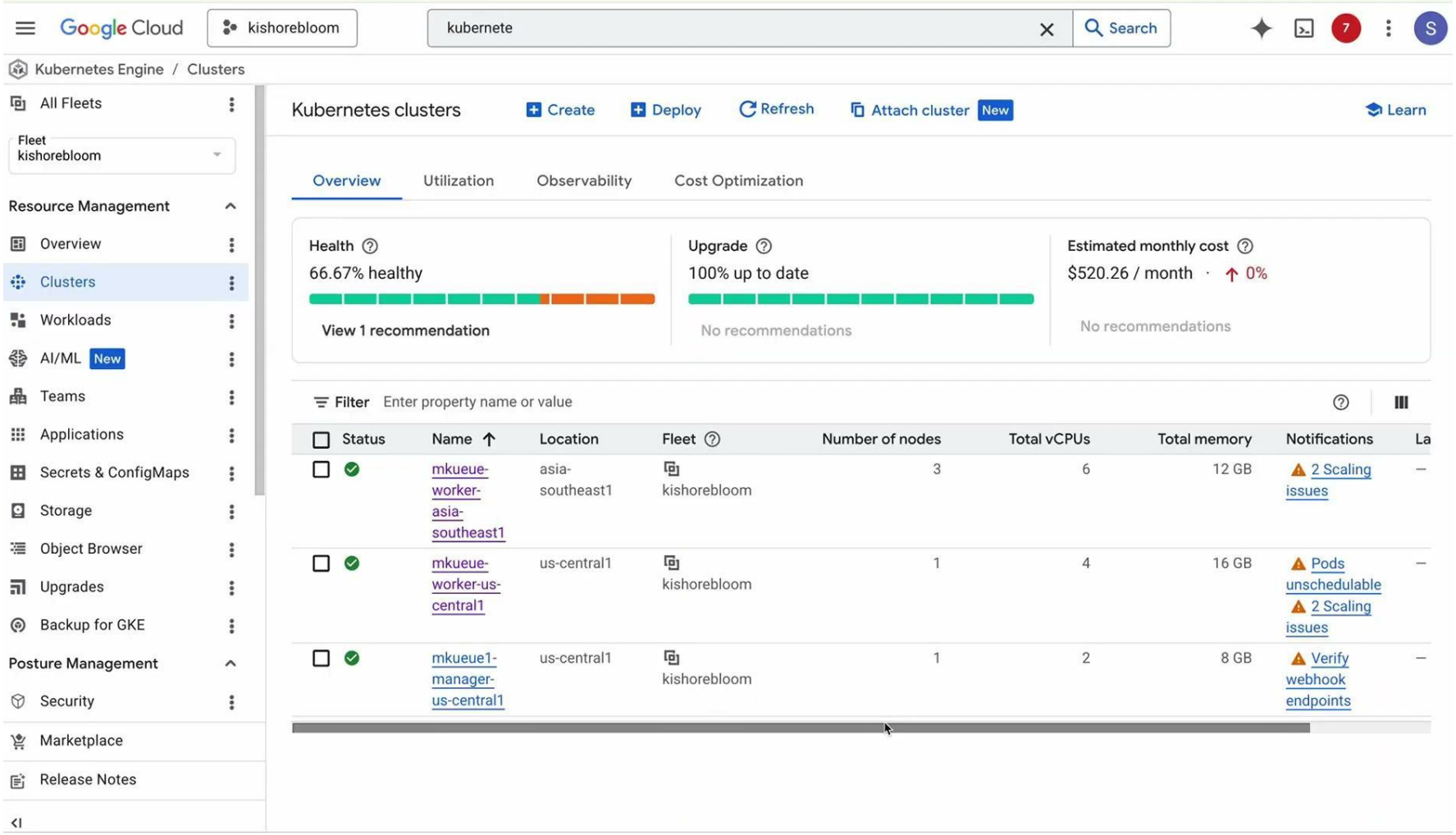
Admission Controller
scale-up

Best-effort-atomic-scaleup

Status: IDLE CAPACITY

Available for workload execution.

Bypassed due to Manager's regional pinning.


Google Cloud
kishorebloom
kubernete
Search
Kubernetes Engine / Clusters
All Fleets
Fleet
kishorebloom
Resource Management
Overview
Clusters
Workloads
AI/ML New
Teams
Applications
Secrets & ConfigMaps
Storage
Object Browser
Upgrades
Backup for GKE
Posture Management
Security
Marketplace
Release Notes
Kubernetes clusters
Create
Deploy
Refresh
Attach cluster New
Learn
Overview
Utilization
Observability
Cost Optimization
Health
66.67% healthy
View 1 recommendation
Upgrade
100% up to date
No recommendations
Estimated monthly cost
$520.26 / month · ↑ 0%
No recommendations
Filter
Enter property name or value
Status
Name
Location
Fleet
Number of nodes
Total vCPUs
Total memory
Notifications
mkueue-worker-asia-southeast1
asia-southeast1
kishorebloom
3
6
12 GB
2 Scaling issues
mkueue-worker-us-central1
us-central1
kishorebloom
1
4
16 GB
Pods unschedulable
2 Scaling issues
mkueue1-manager-us-central1
us-central1
kishorebloom
1
2
8 GB
Verify webhook endpoints


Success!
Full rack
of compute!
The tribes will
calculate!
GPU
A100
H100
TPUv4
A40B0
TPU