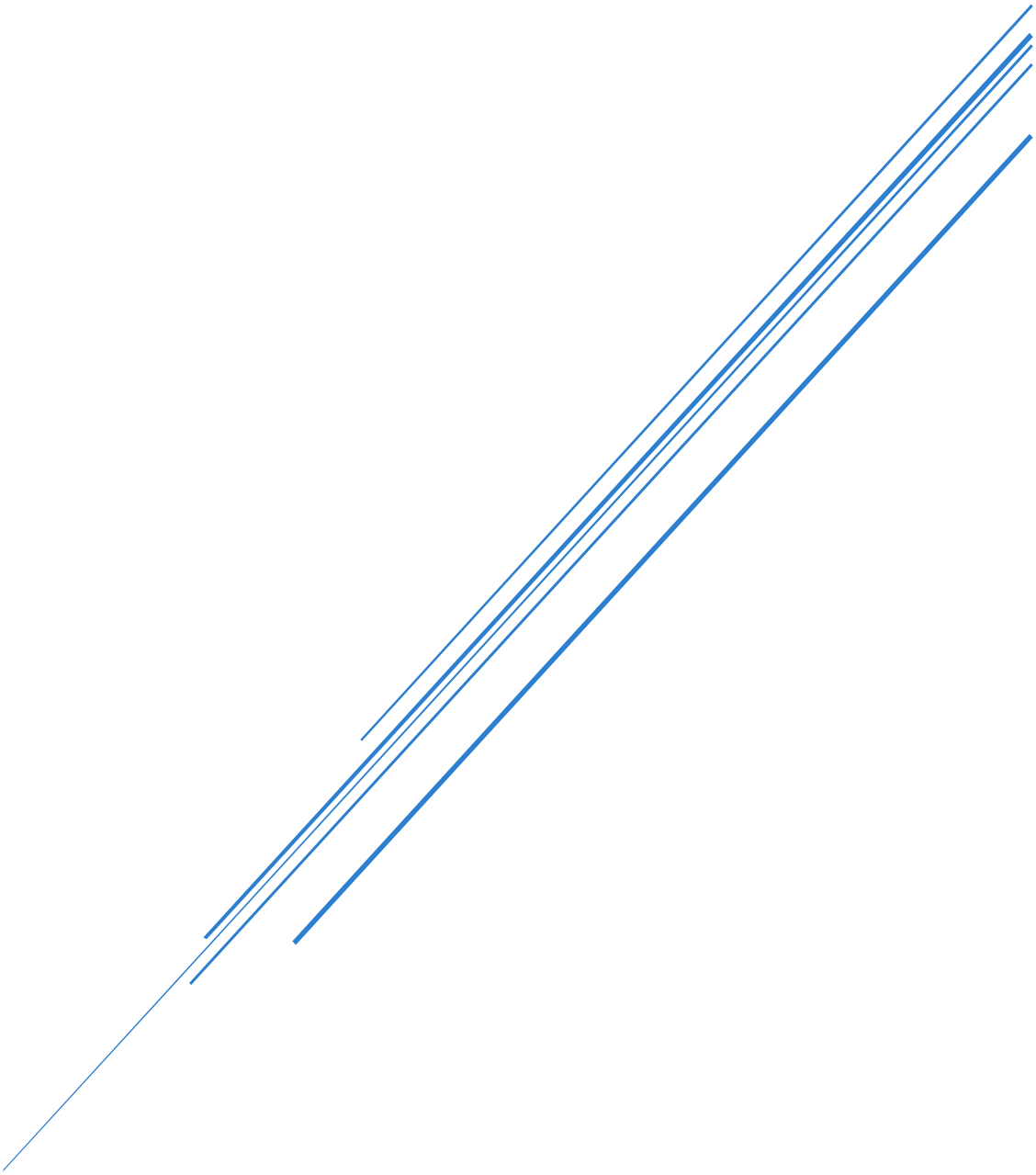
RESEARCH ANALYSIS

Opimizing GPU Costs on Amazon EKS with GPU Slicing and Karpenter AutoScaler



Eugene Parker
DevOps Engineer

Table of Contents

Introduction

The client operates GPU-intensive AI workloads on Amazon EKS (Elastic Kubernetes Service) and seeks to reduce GPU costs. The CTO has identified GPU slicing as a potential solution. Additionally, some clusters use Karpenter Autoscaler, and they want to explore GPU slicing within these autoscaling environments.

This research analysis provides a detailed approach to implementing GPU slicing on EKS and integrating it with Karpenter Autoscaler for cost-efficient and optimized GPU utilization.

To optimize GPU utilization and reduce costs for your AI workloads on Amazon EKS, I propose the implementation of GPU slicing, which allows multiple pods to share a single GPU through time-slicing. This approach is particularly beneficial for workloads that do not require exclusive access to a GPU, thereby maximizing resource efficiency.

Problem Statement

Challenges Faced by the Client:

- High GPU Costs: GPU resources are underutilized due to workloads that do not fully saturate an entire GPU.
- Inefficient GPU Allocation: AI workloads running on Kubernetes may not efficiently share GPU resources, leading to fragmentation and waste.
- Dynamic Scaling Complexity: Clusters using Karpenter autoscaler need GPU-aware autoscaling, which is not natively supported.
- Performance Considerations: GPU slicing must be implemented without significant performance loss for AI workloads.

Goal:

- Implement GPU slicing to share GPUs among multiple workloads, thereby reducing costs.
- Ensure compatibility with Karpenter to enable dynamic scaling of GPU nodes while maintaining optimal performance.

Research on GPU Slicing Technologies

What is GPU Slicing?

GPU slicing allows multiple Kubernetes workloads (pods) to share a single GPU, improving **utilization** and **reducing waste**. There are two primary approaches:

Method	Description	Use Case
Time-Slicing	Multiple containers share a single GPU by allocating fixed time slots for execution.	Best for inference workloads with intermittent GPU use.
Multi-Instance GPU (MIG)	A GPU is divided into **hardware partitions**, each assigned to separate workloads.	Best for workloads requiring **dedicated** GPU memory & compute.

For EKS, NVIDIA Time-Slicing is the most practical approach, as it works across all NVIDIA GPUs without requiring A100-specific MIG hardware support.

NVIDIA GPU Operator and Device Plugin for GPU Sharing

To enable GPU slicing in Kubernetes (EKS), we need to use:

(a) NVIDIA GPU Operator

- Automates installation and management of NVIDIA GPU drivers, device plugin, and CUDA runtime in Kubernetes.
- Deploys NVIDIA DCGM Exporter for GPU monitoring.

(b) NVIDIA Device Plugin

- Enables fine-grained GPU allocation within Kubernetes.
- Supports time-slicing, allowing multiple pods to request fractions of a GPU.

Enabling GPU Slicing on EKS Clusters

Deploy NVIDIA GPU Operator:

- The NVIDIA GPU Operator simplifies the management of GPU resources in Kubernetes environments. It automates the deployment and management of NVIDIA drivers and Kubernetes plugins.

- Source: https://docs.nvidia.com/datacenter/cloud-native/gpu-operator/latest/amazon-eks.html

Configure GPU Time-Slicing:

- Time-slicing allows multiple processes to share a GPU by allocating time slots for each process. This method is effective for workloads with intermittent GPU usage.
- To implement time-slicing, configure the NVIDIA device plugin with the appropriate settings.

Integrating GPU Slicing with Karpenter Autoscaler:

Karpenter is a flexible and high-performance Kubernetes cluster autoscaler that can be configured to work with GPU instances. By integrating GPU slicing with Karpenter, you can dynamically scale your GPU resources based on workload demands.

Install Karpenter:

- Ensure Karpenter is installed and properly configured in your EKS cluster. AWS provides a comprehensive guide on setting up Karpenter
- Source: https://docs.aws.amazon.com/eks/latest/best-practices/karpenter.html

Configure Karpenter for GPU Instances:

- Define node templates in Karpenter that specify GPU instance types suitable for your workloads. This ensures that Karpenter provisions nodes with the necessary GPU resources when scaling.
- Source : https://aws.amazon.com/blogs/containers/delivering-video-content-with-fractional-gpus-in-containers-on-amazon-eks/

Deploy Workloads with GPU Requests:

- When deploying your AI workloads, specify the GPU resource requests and limits in your pod specifications. This ensures that Karpenter schedules the pods on nodes with adequate GPU resources.
- Source: https://aws.amazon.com/blogs/containers/gpu-sharing-on-amazon-eks-with-nvidia-time-slicing-and-accelerated-ec2-instances/

By following these steps, you can effectively implement GPU slicing on your EKS clusters and leverage Karpenter for dynamic scaling, leading to optimized GPU utilization and cost savings.

Implementing GPU Slicing on Amazon EKS

Prerequisites

- Amazon EKS cluster running Kubernetes v1.22+
- GPU-enabled EC2 instances (e.g., G4dn, G5, P4 instances)
- Helm installed on local machine
- Karpenter installed (if applicable)
- 4.2 Deploy NVIDIA GPU Operator on EKS

Deploy NVIDIA GPU Operator on EKS

- This installs the **GPU Operator**, which deploys **NVIDIA drivers, CUDA runtime, and the device plugin**.

Enable GPU Slicing with Time-Slicing (Run Config Map Script)

- Modify the **NVIDIA device plugin configuration** to enable **time-slicing**:
- Apply the ConfigMap
- Restart the NVIDIA Device Plugin

Deploy AI Workloads with Fractional GPU Allocation

- Modify AI workloads to request **partial GPUs** instead of full GPUs:
    - This allows multiple AI workloads to share a single GPU instance

Integrating GPU Slicing with Karpenter Autoscaler (Run Sample AI Script to test)

- Configure Karpenter to Support GPU Nodes
- Create a **Karpenter NodePool** that provisions GPU instances **dynamically**:
    - Karpenter provisions GPU nodes dynamically, reducing costs by scaling nodes only when needed.

Deploy GPU Workloads with Karpenter Autoscaling

- Ensure AI workloads request **fractional GPUs** to take advantage of slicing.
- Karpenter will **automatically scale GPU nodes** when demand increases.
- Idle GPU nodes will **terminate automatically**, reducing costs.

Monitoring and Optimization

- Monitor GPU Usage in Kubernetes
- Optimize Time-Slicing for Performance
- Adjust timeSlicing.period in nvidia-device-plugin-config.yaml for performance tuning:

By implementing **GPU Slicing with NVIDIA Time-Slicing** and integrating it with **Karpenter Autoscaler**, the cluster should be able to autoscale and dynamically provision/terminate pods which is in a nutshell:

- **Reduces GPU costs** by maximizing resource utilization.
- **Enables dynamic scaling** of GPU nodes.
- **Optimizes AI workload scheduling** for better performance.

Reference Lists

- GPU sharing on Amazon EKS with NVIDIA time-slicing and accelerated EC2 instances
  - Source : https://aws.amazon.com/blogs/containers/gpu-sharing-on-amazon-eks-with-nvidia-time-slicing-and-accelerated-ec2-instances/
- Installing and Configuring Karpenter on Fargate for Autoscaling in Amazon EKS
  - Source : https://community.aws/content/2drAPvul6M6GL27Vq6mJcWjLvBR/navigating-amazon-eks-eks-karpenter-fargate
- Scaling a Large Language Model with NVIDIA NIM on Amazon EKS with Karpenter
  - Source Link : https://aws.amazon.com/blogs/containers/scaling-a-large-language-model-with-nvidia-nim-on-amazon-eks-with-karpenter/
- Delivering video content with fractional GPUs in containers on Amazon EKS
  - Source: https://aws.amazon.com/blogs/containers/delivering-video-content-with-fractional-gpus-in-containers-on-amazon-eks
- Karpenter
  - Source : https://docs.aws.amazon.com/eks/latest/best-practices/karpenter.html
  - GPU sharing on Amazon EKS with NVIDIA time-slicing and accelerated EC2 instances
- GPU sharing on Amazon EKS with NVIDIA time-slicing and accelerated EC2 instances
  - Source : https://aws.amazon.com/blogs/containers/gpu-sharing-on-amazon-eks-with-nvidia-time-slicing-and-accelerated-ec2-instances
- NVIDIA GPU Operator with Amazon EKS
  - Source : https://docs.nvidia.com/datacenter/cloud-native/gpu-operator/latest/amazon-eks.html
  -