# What is Karpenter?

Karpenter is an open-source Kubernetes cluster autoscaler developed by AWS, designed to enable rapid and efficient scaling of infrastructure based on real-time demands. Initially created for AWS, Karpenter integrates seamlessly with EC2 instances, supporting both On-Demand and Spot Instances. However, Karpenter is not limited to AWS. As of November 2023, it expanded to support Azure, making it a versatile tool for multicloud environments.

Karpenter’s architecture is highly flexible, featuring a pluggable design that allows for future expansion to additional cloud providers. This architecture is actively maintained by the Kubernetes community under the repository kubernetes-sigs/karpenter, ensuring continuous development and the futureproofing of its core features.

## Key Capabilities of Karpenter:

* **Rapid Node Provisioning**: Karpenter provisions new nodes as quickly as needed, reducing the wait time for unscheduled pods to acquire resources.
* **Optimal Resource Allocation**: It ensures nodes are appropriately sized for the workload, preventing over-provisioning and contributing to cost efficiency.
* **Automatic Node Termination**: Karpenter automatically terminates nodes when they are no longer required, avoiding the cost of unused resources.

## Karpenter vs. Cluster Autoscaler (CAS)

Kubernetes users have relied on the Cluster Autoscaler (CAS) for scaling, but it presents limitations in handling dynamic, fast-changing cloud environments.

Here’s why Karpenter outperforms CAS:

* **Faster Scaling**: By leveraging AWS’s Fleet API, Karpenter bypasses the delays caused by EC2 Auto Scaling Groups, significantly improving response times.
* **Flexible Node Management**: Unlike CAS’s rigid Node Group configurations, Karpenter allows for dynamic adjustment of instance types and sizes, offering better resource utilization.
* **Simplified Configurations**: Karpenter reduces the need for manually defined Node Groups and updates, making scaling simpler and more efficient.

## Karpenter’s Cost Optimization Strategies

When it comes to cost optimization, Karpenter enables two main strategies: optimizing for price and optimizing for efficiency.

## Optimizing for Price: Leveraging EC2 Pricing Models

AWS offers multiple pricing strategies for EC2 instances, and Karpenter helps switch between them to optimize costs. Here’s how you can leverage these models:

* **On-Demand Instances**: This is the simplest pricing model, offering flexibility but also being the most expensive. On-Demand instances are ideal for short-term, unpredictable workloads. However, they should be avoided for long-term or stable workloads due to their high cost. Karpenter can provision resources from On-Demand instances when needed but should prioritize other, more cost-effective options first.
* **Spot Instances**: Spot Instances provide up to 90% savings compared to On-Demand instances. These instances are ideal for workloads that can handle interruptions, as AWS can reclaim the capacity with short notice. Karpenter is natively integrated with AWS Spot, automatically handling node terminations by draining affected nodes and launching new instances. This ensures that workloads remain uninterrupted while keeping costs significantly lower.
* **Reserved Instances & Savings Plans**: Karpenter can also work with Reserved Instances (RIs) and Savings Plans to maximize cost savings. RIs offer significant discounts in exchange for a long-term commitment to specific instance types, while Savings Plans provide flexibility across instance types and AWS services in exchange for a monetary commitment.

By configuring NodePools in Karpenter, workloads can be directed to use RIs or Savings Plans first. This helps ensure that reserved capacity is fully utilized before provisioning more expensive On-Demand instances. For example, if your organization has RIs for certain EC2 instance types, Karpenter can prioritize using these for workload scaling, thereby avoiding unnecessary On-Demand costs. Similarly, Savings Plans can be leveraged to dynamically adjust workloads while staying within your committed spend, maximizing flexibility and cost efficiency.

## Optimizing for Efficiency: Reducing Waste

Cost optimization isn’t only about reducing what you pay; it’s also about using fewer resources efficiently. Karpenter ensures that your infrastructure is right-sized to meet workload demands without over-provisioning, preventing unnecessary resource usage. This is achieved through Disruption Controllers, which automatically manage node usage based on real-time data.

Here’s how Karpenter’s Disruption Controllers help optimize costs:

* Expiration: Nodes are deleted after they have reached the end of a predefined lifecycle or lease period. This prevents unnecessary costs by automatically removing infrastructure that is no longer needed.
* Consolidation: Karpenter can analyze workloads running on nodes and consolidate them onto fewer, more optimized nodes. This process helps eliminate underutilized resources, shifting workloads to other nodes and shutting down those that aren’t fully utilized. For example, if multiple nodes are running at low capacity, Karpenter can consolidate those workloads onto fewer nodes, freeing up resources and shutting down the unused infrastructure.
* Drift Management: Sometimes, a node’s configuration may no longer match the desired state of the cluster due to changes in the NodePool settings or updates to machine images. Karpenter detects this drift and replaces non-compliant nodes with ones that match the current desired configuration, ensuring that your cluster remains optimized without manual intervention.
* Handling Spot Instance Interruptions: Spot Instances can be interrupted when AWS reclaims capacity. Karpenter is equipped with a built-in mechanism to gracefully handle these interruptions. When an interruption notice is received (usually with a 2-minute warning), Karpenter reallocates workloads to new Spot or On-Demand instances, ensuring minimal disruption to your applications while optimizing costs.

## Best Practices for Karpenter in Cost Optimization

To maximize cost savings with Karpenter, follow these best practices:

1. Optimize Node Pools: Configure NodePools to support multiple instance types and sizes, allowing Karpenter to allocate the best-fit resources dynamically. This flexibility helps prevent over-provisioning and ensures resource optimization.
2. Leverage Spot Instances: Whenever possible, use Spot Instances for interruptible workloads, but balance this with stability by enabling Karpenter’s consolidation features to minimize downtime.
3. Monitor Cluster Efficiency: Regularly review cluster metrics to identify underutilized resources. Tools like AWS Cost and Usage Reports (CUR) can provide real-time cost data, helping to ensure that you’re not overpaying for resources you don’t need.