Kubernetes Cloud Cost Monitoring with OpenCost & Optimization Strategies

Matt Ray
Senior Community Manager for OpenCost
mattray@kubecost.com

SCaLE 20x
Who am I?

Matt Ray

- Senior Community Manager for OpenCost at Kubecost.
- Co-host of the Software Defined Talk podcast for 8+ years.
- Living in Sydney, Australia for 6+ years after relocating from Austin, Texas.
- Active in Open Source for much, much longer.
- mattray@kubecost.com
- https://www.linkedin.com/in/mhray/
- @mattray on GitHub, Mastodon, too many Slacks
The complexity of operating Kubernetes efficiently is real.

**Technical COMPLEXITY**

Higher-level abstractions, more shared resources, and increasingly dynamic

**Behavioral COMPLEXITY**

Decentralized releases means any engineer can increase spend quickly

Any engineer → Any org → Any time

$ $ $
OpenCost

Open source Kubernetes cost monitoring

Specification and Implementation

Cloud Native Computing Foundation Sandbox Project

FinOps Certified Solution

- https://opencost.io
- https://github.com/opencost
- https://www.cncf.io/projects/opencost/
OpenCost Specification

Created by a community of Kubernetes practitioners
https://github.com/opencost/opencost/blob/develop/spec/opencost-specv01.md

- Adobe
- Armory
- AWS
- D2IQ
- Google Cloud
- Kubecost
- Mindcurv
- New Relic
- Pixie
- Red Hat
- SUSE
Kubernetes workloads are often transient and vary in the resources they consume.

How do we measure who is responsible for what and how much?

- Management fees
- Expenses from nodes
- Persistent volumes
- Attached disks
- Load balancers
- Network ingress/egress
Total Cluster Costs

Total Cluster Costs = Cluster Asset Costs + Cluster Overhead Costs
Cluster Asset Costs

Cluster Asset Costs = Resource Allocation Costs + Resource Usage Costs
Cluster Asset Costs: Node Costs

Cluster Asset Costs = Resource Allocation Costs + Resource Usage Costs

- Cluster Overhead Costs
  - Resource Allocation Costs
  - Resource Usage Costs

= Cluster Management Fees
  - Node (CPU, RAM, GPU)
  - Persistent Volume
  - Load Balancer
  - Network Egress
We’ve got the cost of our Kubernetes assets

Now let's distribute them across Workloads
Workload Costs

Inside the Kubernetes Cluster

What is Measured

• CPU
• Memory
• GPU
• Storage Volume
• Load Balancer

Aggregations

• Container
• Pod
• Deployment
• StatefulSet
• Job
• Controller Name
• Controller Kind
• Label
• Annotation
• Namespace
• Node
• Cluster
Total Cluster Costs = Workloads + Cluster Idle Costs + Cluster Overhead Costs
# Workload Costs + Cluster Idle Costs

Total Cluster Costs = Workloads + Cluster Idle Costs + Cluster Overhead Costs

<table>
<thead>
<tr>
<th>Cluster Overhead Costs</th>
<th>Resource Allocation Costs</th>
<th>Resource Usage Costs</th>
<th>Cluster Idle Costs</th>
<th>Workload Costs</th>
</tr>
</thead>
</table>

[Diagram]
### Workload Costs + Cluster Idle Costs

Total Cluster Costs = Workloads + Cluster Idle Costs + Cluster Overhead Costs

<table>
<thead>
<tr>
<th>Resource Allocation Costs</th>
<th>Resource Usage Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster Overhead Costs</td>
<td></td>
</tr>
</tbody>
</table>

=  

<table>
<thead>
<tr>
<th>Cluster Idle Costs</th>
<th>Allocated Costs</th>
<th>Usage Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster Overhead Costs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Cluster Idle Cost

Cluster Idle Cost = Cluster Asset Costs - Workload Costs

Idle costs can be calculated at the Asset/Resource and at the Workload level.

```
<table>
<thead>
<tr>
<th>Allocated</th>
<th>Idle</th>
</tr>
</thead>
</table>
```

Total CPU Capacity
I understand what we’re measuring

How do I get at the metrics?
OpenCost Architecture

- k8s-api
- billing data

OpenCost

Prometheus

ksm
node-exporter
cadvisor
Deploying OpenCost

https://www.opencost.io/docs/install

Prometheus
• Prom community Helm chart the default

OpenCost Manifest
• kubectl apply --namespace opencost -f https://raw.githubusercontent.com/opencost/opencost/develop/kubernetes/opencost.yaml

OpenCost Helm Chart
• https://github.com/opencost/opencost-helm-chart/
• Configurable settings (Prometheus, namespaces, etc.)
## Accessing OpenCost

- **API**
- **Web UI**
- **kubectl cost**
- **Prometheus**

<table>
<thead>
<tr>
<th>CLUSTER</th>
<th>NAMESPACE</th>
<th>MONTHLY RATE (ALL)</th>
<th>COST EFFICIENCY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>opencost</td>
<td>18.295200</td>
<td>0.231010</td>
</tr>
<tr>
<td></td>
<td>prometheus</td>
<td>17.992800</td>
<td>0.000000</td>
</tr>
<tr>
<td></td>
<td>kube-system</td>
<td>11.383200</td>
<td>0.033410</td>
</tr>
<tr>
<td>SUMMED</td>
<td></td>
<td>47.671200</td>
<td></td>
</tr>
</tbody>
</table>
## OpenCost

### Last 7 days by controller daily

25 January 2023 through now by Controller

<table>
<thead>
<tr>
<th>Name</th>
<th>CPU</th>
<th>RAM</th>
<th>PV</th>
<th>Efficiency</th>
<th>Total cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Totals</td>
<td>$1.28</td>
<td>$0.09</td>
<td>$0.00</td>
<td>15.4%</td>
<td>$1.37</td>
</tr>
<tr>
<td>deployment:coredns</td>
<td>$0.55</td>
<td>$0.05</td>
<td>$0.00</td>
<td>3.2%</td>
<td>$0.60</td>
</tr>
<tr>
<td>daemonset:kube-proxy</td>
<td>$0.55</td>
<td>$0.00</td>
<td>$0.00</td>
<td>0.3%</td>
<td>$0.55</td>
</tr>
<tr>
<td>daemonset:aws-node</td>
<td>$0.14</td>
<td>$0.00</td>
<td>$0.00</td>
<td>12.8%</td>
<td>$0.14</td>
</tr>
<tr>
<td>deployment:opencost</td>
<td>$0.05</td>
<td>$0.04</td>
<td>$0.00</td>
<td>18.3%</td>
<td>$0.09</td>
</tr>
</tbody>
</table>
What’s the Future of OpenCost?
What’s the Future of OpenCost?

What do you want it to be?
Near-Term Roadmap

- External Asset Costs
- Backstage integration
- More Clouds
- More Documentation
- More Integrations
Get Involved with OpenCost

https://www.opencost.io

Slack
- https://slack.cncf.io/ #opencost

GitHub
- https://github.com/opencost/opencost
- https://github.com/opencost/opencost-helm-chart
- https://github.com/opencost/opencost-website

OpenCost Working Group

LinkedIn
- https://www.linkedin.com/showcase/opencost/
Kubernetes Optimization Strategies

We’ve got the numbers, now what do we do?
The FinOps Foundation provides guidance on cloud financial management through best practices, education, and standards.

Establish a FinOps practice within your organization.
FinOps is an evolving cloud financial management discipline and cultural practice that enables organizations to get maximum business value by helping engineering, finance & business teams to collaborate on data-driven spending decision.

**Principles**
- Teams need to collaborate
- Everyone takes ownership for their cloud usage
- A centralized team drives FinOps
- Reports should be accessible and timely
- Decisions are driven by business value of cloud
- Take advantage of the variable cost model of the cloud

**Personas**
- FinOps Practitioner
- Executive
- Business/Product Owner
- Finance/Procurement
- Engineering/Operations

**Domains**
- Understanding Cloud Usage and Cost
- Performance Tracking & Benchmarking
- Real-Time Decision Making
- Cloud Rate Optimization
- Cloud Usage Optimization
- Organizational Alignment

**Phases**
- Crawl
- Walk
- Run

Slide provided by the FinOps Foundation under the CC BY 4.0
Optimization Strategy

Start at the top
Efficiencies and cost savings compound
Coordinate your savings across the org
This is an iterative process
Workload Strategies

Applications Running on Kubernetes

**Abandoned Workloads**
- Deleting abandoned pods, controllers, or even entire namespaces

**Right Sizing Containers**
- Updating pod manifests to reflect observed usage
- Providing requests and possibly LimitRanges for default resource allocations
- Always provide CPU requests, probably do not use CPU limits
- Always use memory requests and make limits equivalent

**Managing Unclaimed Volumes**
- Delete volumes that are unused by any pods or move them to a cheaper storage tier
Kubernetes Strategies

Cluster Configurations

**Right Sizing Cluster Nodes**
- Adjust the number and size of your cluster's nodes to stop overspending on unused capacity
- AMD CPUs may be less expensive than Intel for some workloads

**Underutilized Nodes**
- Adjust the number and type of your cluster's nodes to stop overspending on unused capacity
- Check CPU, memory, storage class, and network requirements

**Managing Unclaimed Volumes**
- Delete volumes that are unused by any pods or move them to a cheaper storage tier
Operating System Optimizations

Under the Kubernetes Clusters

Delete Unassigned Resources
  • Disks and IP addresses that are not being used by any clusters may continue to incur charges

Resize Local Disks
  • Resize local disks with low utilization

Switch to Arm architecture
  • Arm CPUs are generally less expensive than Intel across cloud providers for similar performance
Cloud Infrastructure Optimizations

Cloud FinOps

**Reserved Instances**
- Consider purchasing reserved instances based on historical resource usage patterns

**Spot Instances**
- Identify workloads ready for spot (preemptible) nodes and resize your cluster to realize the savings of migrating workloads to spot

**Savings Plans**
- Talk to your cloud vendor about all your options

This is why you have a FinOps team.
Optimization Strategy

Start at the top
Efficiencies and cost savings compound
Coordinate your savings across the org
This is an iterative process
Thanks!
opencost@kubecost.com