PLATFORM9

Case study

How a SaaS data management company slashed AWS EKS cost by 58%

Platform9's Elastic Machine Pool (EMP) delivers continuous and intelligent optimization to maximize Kubernetes utilization and cut public cloud cost.

Summary

A rapidly expanding SaaS data management firm, experiencing a surge in service adoption, adopted an aggressive cloud-native approach by transitioning their main SaaS application to run on Kubernetes in Amazon EKS. This move not only accelerated feature development but also increased cloud expenses, with Kubernetes constituting about 50% of their multimillion-dollar AWS costs. Even with the surge in usage, the budget remained unchanged due to economic constraints.

Challenges

The Operations team struggled with unchecked cluster deployments and poor utilization. Despite the use of tools like AWS CloudWatch and Kubecost offering insights, true optimization was elusive. Expenses increased as attempts to optimize clusters produced minimal outcomes.

- Inability to persuade developers to prioritize cost savings
- Inefficient Bin Packing
- Lack of granular visibility of utilization and costs
- Manual optimization wasn't working

Results

- In just four months, the Ops team was able to cut Kubernetes costs by 58%, from $19 million to $8 million per year.
- In this time period, EMP was able to deliver these cost savings by automatically increasing average cluster memory utilization from 25% to an acceptable 60%.

“Our Kubernetes costs were multiplying out of control. With EMP, we finally achieved automated optimization. In months, we cut spending by 58%, and our cluster memory utilization improved from 25% to 60%, helping us avoid a budget crisis."

VP Engineering,
SaaS Data Management company

Situation

As the adoption of its service skyrocketed, a SaaS data management technology company rapidly expanded its cloud consumption. The company's primary revenue-generating product is deployed as SaaS on AWS. With more than 50% of their workloads containerized and running on Elastic Kubernetes Service (EKS) – AWS's native Kubernetes offering – Kubernetes quickly became the largest consumer of cloud infrastructure.

Their primary application running microservices was deployed on hundreds of EKS clusters across development, staging, and production environments. While this allowed the company to accelerate feature development, it also drove up cloud costs. Kubernetes accounted for almost 50% of their multimillion-dollar AWS bills. Despite this enormous increase in usage, budgets remained flat due to economic conditions.

Challenges

The company's VP of Reliability Engineering was under pressure from the CFO to keep cloud spend under control. However, she and her DevOps architects faced a number of challenges:

Inability to persuade developers to prioritize cost savings:

Developers sized and allocated the main application resources based on peak usage requirements. However, for the majority of the time, the application used only a small percentage of the resources reserved for it, resulting in a lot of waste. The developers preferred this approach because it minimizes the risk of costly downtime. Specifically, they wanted to avoid situations where Kubernetes can't allocate the necessary resources, which can cause the application to shut down.

Inefficient Bin Packing:

Workloads on Kubernetes are placed on available nodes based on the workload's resource requirements, as well as a set of plugins or 'sidecars' that must be installed on each node. Since the instance sizes on AWS are not always configurable to your workload needs, there were often cases where there were more resources available on a given node than what was allocated to the workload plus all the plugin components deployed on it. This inefficient bin packing resulted in a lot of capacity that was not used but still incurred expenses for the company.

Lack of granular visibility of utilization and costs:

While the DevOps teams had good visibility into their overall AWS and Kubernetes spend, they found it very difficult to identify the exact workload that was contributing to the majority of costs. The tools from AWS Cost Explorer were not granular enough to help them identify the spikes in usage that contributed to the majority of the costs. Another challenge was having visibility into the actual utilization of their EKS clusters. While they could see how many resources were allocated to their workloads, they could not see what percentage of those resources were actually used on average over a given period of time.

Manual optimization wasn't working:

On top of the visibility and control challenges, keeping clusters optimized was a losing battle. By the time the Ops team tuned utilization on one cluster, two more were deployed and wasting resources. The exponential growth overwhelmed any manual efforts. The Kubernetes sprawl was unstoppable.

The Ops team was drowning, trying to keep up with the sprawling growth. Engineers continued deploying new clusters with minimal oversight. As a result, their overall memory utilization of Kubernetes nodes was 25% on average.

Alternatives tried and failed

The Ops team tried several alternatives to get costs under control:

Spot instances were not a fit for all workloads

The team utilized spot instances as a way to optimize Kubernetes costs. However, spot instances were not a fit for all their workloads, because several components of their primary application could not tolerate interrupts well. As a result, they had to create affinity rules to map the right workloads to the right instances, adding to the complexity. In the end, spot instances could only account for 20% of their infrastructure needs at most, and the remaining 80% had to be deployed using on-demand instances.

CloudWatch rules

The Ops team tried setting CloudWatch alarms to notify when CPU or memory utilization dropped below 50% on a cluster. This required manually configuring per-cluster alarms across all of their EKS environments. The overhead of managing the alarms at scale was untenable. Engineers also tended to ignore or delete the alarms to avoid dealing with justification paperwork.

Reserved Instances & Savings Plan

The company negotiated substantial discounts with AWS by committing to 1 and 3-year Reserved Instance purchases. However, because Reserve Instance (RI) commitments are instance type specific and can not be easily transferred to different instance types, their dynamic workloads and unpredictable capacity needs made the RI model too rigid. Switching to a Savings Plan instead of Reserved Instances helped, but the long-term commitments locked them into significant excess capacity as needs changed over time.

Cluster downsizing

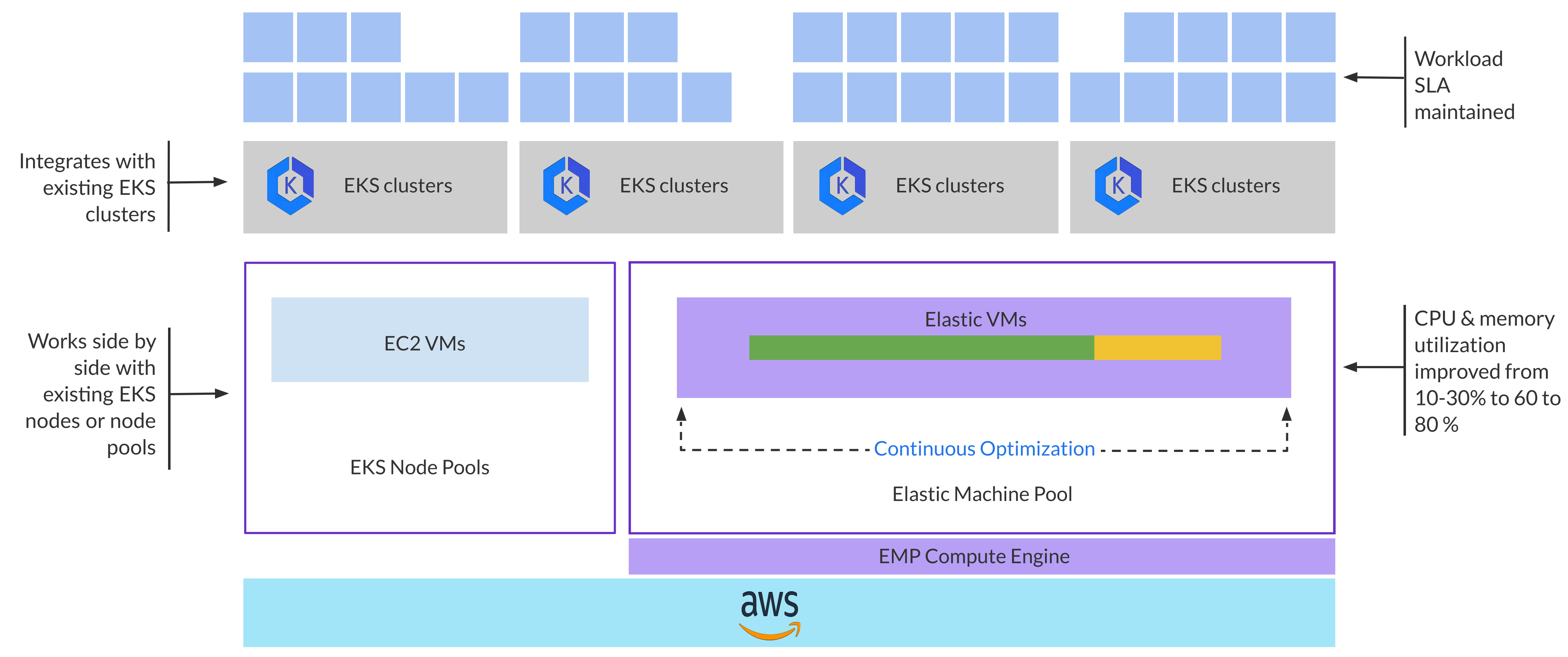
As a brute force tactic, the Ops team manually consolidated clusters and aggressively scaled nodes down. However, this caused performance issues and outage risks. The engineers rebelled against this aggressive downsizing. The operational burden of constant downsizing was also unsustainable for the small Ops team.

Solution

On the verge of a budget crisis, the VP of Reliability Engineering began looking for alternative solutions and discovered Platform9's Elastic Machine Pool (EMP). The idea that this solution could dynamically optimize Kubernetes through intelligent oversubscription and bin packing, and do this all behind the scenes with no changes to their applications was really appealing to her.

EMP leverages patent-pending software to consolidate Kubernetes deployments into the densest possible configuration within a cluster. Using technology such as dynamic rebalancing of workloads across available compute resources, EMP is able to make sure that the application SLA is never compromised, while still getting the best utilization from the underlying CPU and Memory resources. This allows EMP to achieve the best of both worlds - cut down wasted resources by half while keeping the application SLA. All of this is beyond native Kubernetes capabilities.

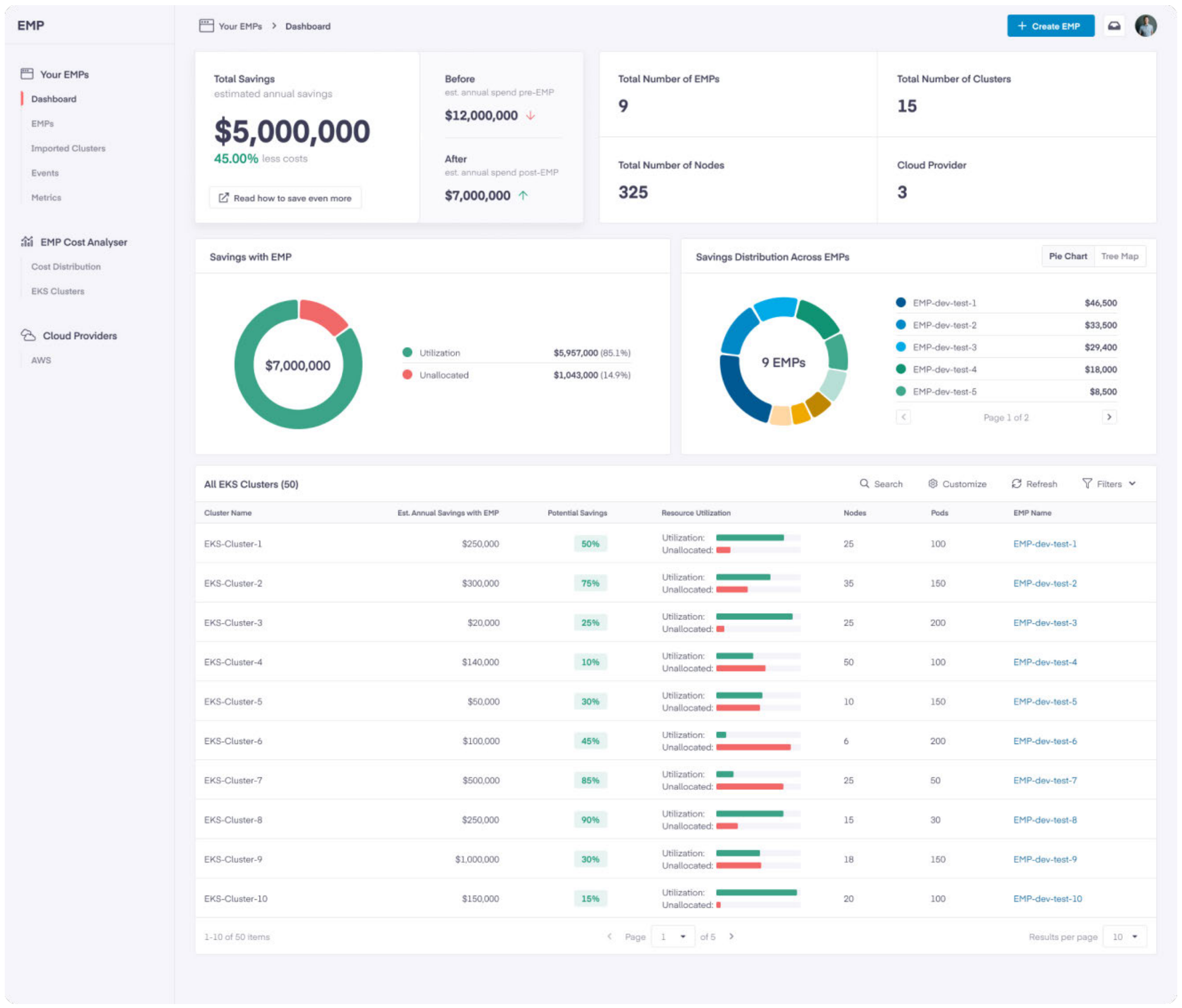
EMP continuously monitors resource utilization across all clusters to identify optimization opportunities. The algorithm predicts workload needs and simulates scheduling scenarios to model the optimal distribution for maximizing utilization.



EMP Architecture

Based on these predictions, EMP elastically scales cluster resources up and down in real-time to maintain peak efficiency. When utilization spikes occur that require additional capacity, EMP seamlessly scales out the cluster and reschedules pods onto new nodes without any downtime.

All of this occurs automatically behind the scenes, without any manual intervention required by engineering teams. Engineers do not need to change their resource requirement settings for their applications, thus reducing the constant friction between Engineering and Ops around resource optimization.



Total Savings
estimated annual savings
$5,000,000
45.00% less costs

Before
est. annual spend pre-EMP
$12,000,000

After
est. annual spend post-EMP
$7,000,000

Total Number of EMPs: 9
Total Number of Clusters: 15
Total Number of Nodes: 325
Cloud Provider: 3

Savings with EMP

$7,000,000

● Utilization	$5,957,000 (85.1%)
● Unallocated	$1,043,000 (14.9%)

Savings Distribution Across EMPs

9 EMPs

● EMP-dev-test-1	$46,500
● EMP-dev-test-2	$33,500
● EMP-dev-test-3	$29,400
● EMP-dev-test-4	$18,000
● EMP-dev-test-5	$8,500

Page 1 of 2

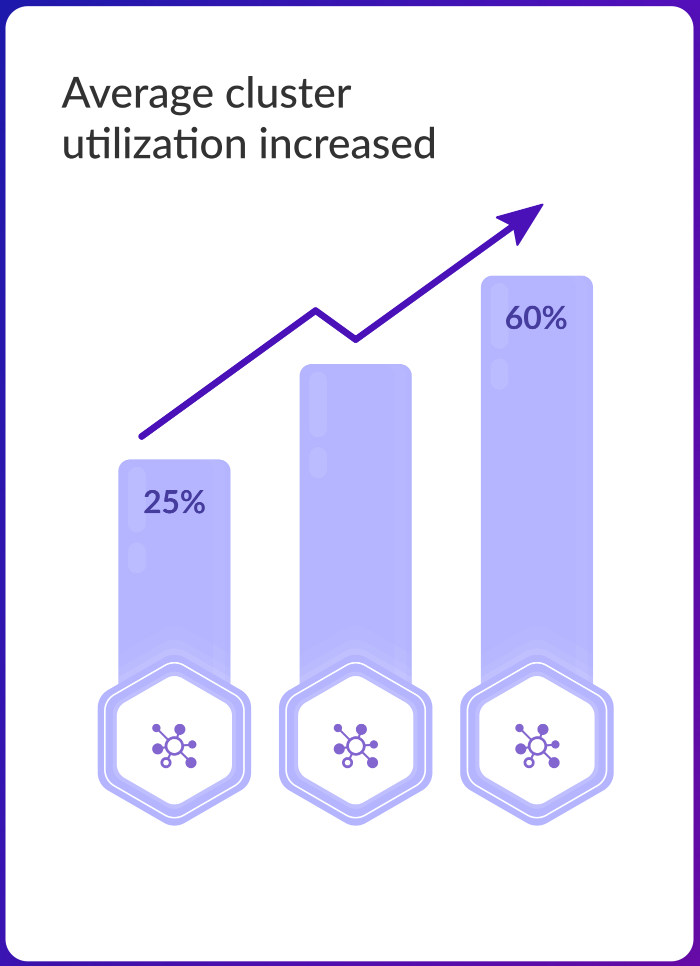
All EKS Clusters (50)

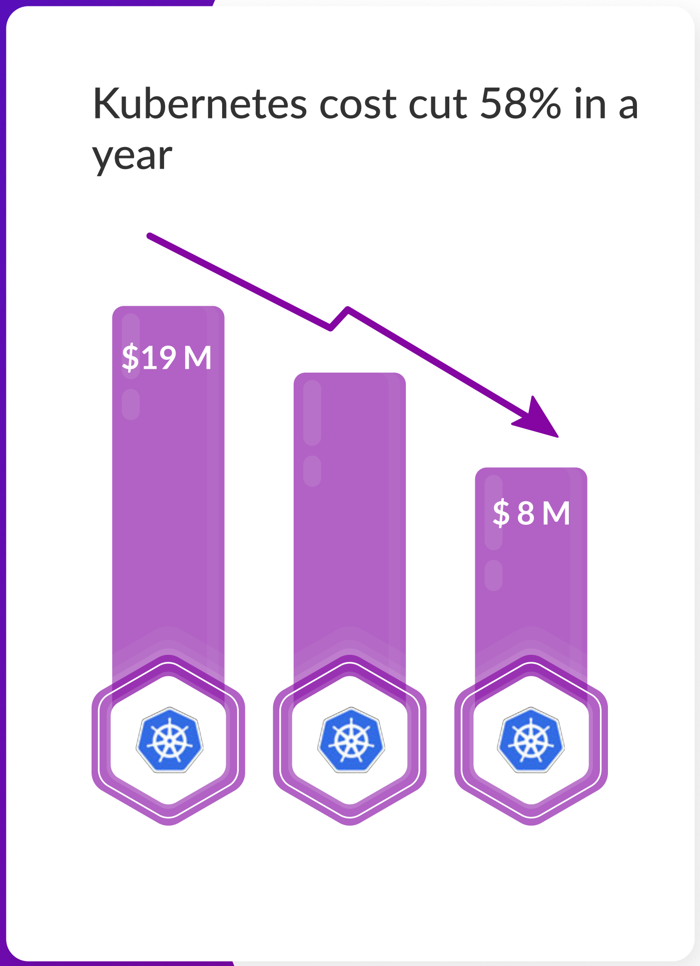
Cluster Name	Est. Annual Savings with EMP	Potential Savings	Resource Utilization	Nodes	Pods	EMP Name
EKS-Cluster-1	$250,000	50%	Utilization: Unallocated:	25	100	EMP-dev-test-1
EKS-Cluster-2	$300,000	75%	Utilization: Unallocated:	35	150	EMP-dev-test-2
EKS-Cluster-3	$20,000	25%	Utilization: Unallocated:	25	200	EMP-dev-test-3
EKS-Cluster-4	$140,000	10%	Utilization: Unallocated:	50	100	EMP-dev-test-4
EKS-Cluster-5	$50,000	30%	Utilization: Unallocated:	10	150	EMP-dev-test-5
EKS-Cluster-6	$100,000	45%	Utilization: Unallocated:	6	200	EMP-dev-test-6
EKS-Cluster-7	$500,000	85%	Utilization: Unallocated:	25	50	EMP-dev-test-7
EKS-Cluster-8	$250,000	90%	Utilization: Unallocated:	15	30	EMP-dev-test-8
EKS-Cluster-9	$1,000,000	30%	Utilization: Unallocated:	18	150	EMP-dev-test-9
EKS-Cluster-10	$150,000	15%	Utilization: Unallocated:	20	100	EMP-dev-test-10

1-10 of 50 items   Page 1 of 5   Results per page 10

EMP Dashboard

Deploying EMP into their existing EKS environment required no infrastructure or tooling changes for the company. EMP seamlessly integrated with their existing AWS EKS environment and allowed them to easily migrate workloads, starting with Dev and QA workloads first, then moving to staging and production as the Ops teams built more confidence with the product.





Results

The company first rolled out EMP for its QA environment. Within weeks of deployment, EMP started delivering significant improvements in Kubernetes utilization. As EMP optimized the infrastructure footprint, the company's Kubernetes spending decreased steadily month after month.

In just four months, the Ops team was able to cut Kubernetes costs by 58%, from $19 million to $8 million per year. In this time period, EMP was able to deliver these cost savings by automatically increasing average cluster memory utilization from 25% to an acceptable 60%.

The company regained visibility and control over its out-of-control Kubernetes costs. The VP of Reliability Engineering was able to avoid a budget crisis while continuing to meet their scaling demands, thanks to Platform9. EMP enabled her small team to intelligently tame Kubernetes cost overruns without imposing rigid development constraints or impacting developer productivity.

EMP provided invaluable reassurance. The team confidently informed the CFO about the cost savings. Evenings and weekends previously spent on manual adjustments or coordinating with internal teams were now liberated due to the reduced cloud expenses. Instead of financial fire drills, the Ops team could finally focus on enabling innovation for the company's developers.

About us

Platform9 empowers enterprises with a faster, better, and more cost-effective way to go cloud native. Its fully automated container management and orchestration solution delivers cost control, resource reduction, and speed of application deployment. Its unique always-on assurance™ technology ensures 24/7 non-stop operations through remote monitoring, automated upgrades, and proactive problem resolution. Innovative enterprises like Juniper, Kingfisher Plc, Mavenir, Redfin, and Cloudera achieve 4x faster time-to-market, up to 90% reduction in operational costs, and 99.99% uptime. Platform9 is an inclusive, globally distributed company backed by leading investors.

Follow us on

Headquarter: 84W Santa Clara St Suite 800, San Jose, CA 95113

India office: 7th Floor, Smartworks M Agile Building,Pan Card Club Road, Baner Pune, 411045 Maharashtra, India

Phone: +1 650-898-7369 | Email: info@platform9.com | Website: https://platform9.com/contact/