Technical Validation

StormForge: Optimizing Kubernetes Application and Resource Efficiency with Machine Learning

Exceed Business Outcomes for Cloud-native Apps while Managing Runaway Cloud Costs

By Kerry Dolan, Senior IT Validation Analyst

October 2021

This ESG Technical Validation was commissioned by StormForge and is distributed under license from ESG.

Introduction

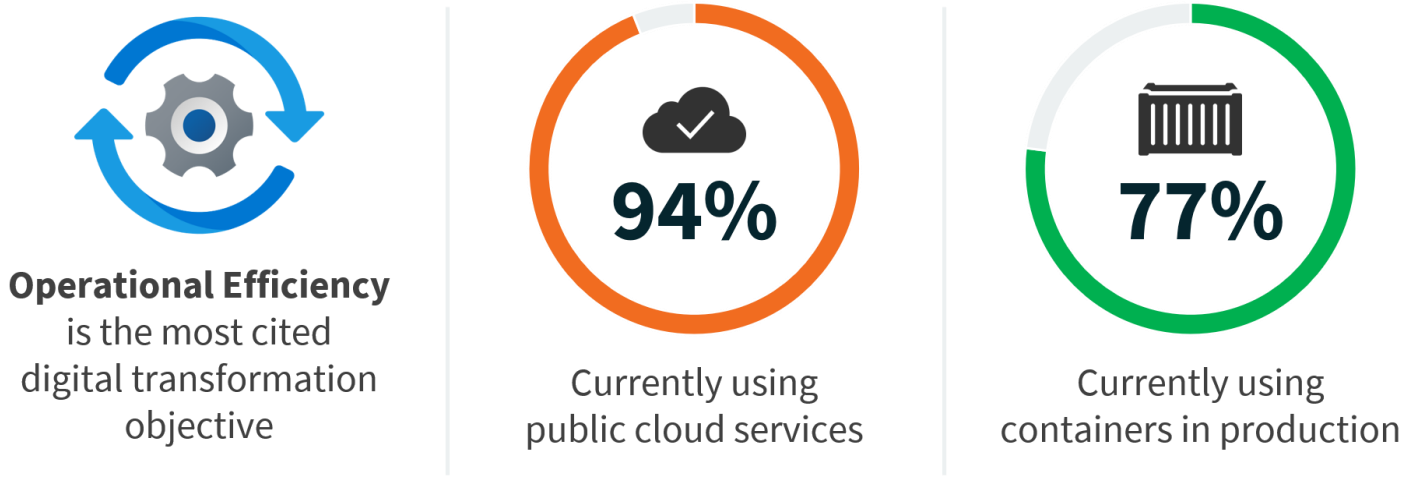
This report documents ESG's technical validation of StormForge for automating Kubernetes resource allocation to optimize business objectives.

Background

According to ESG research, the most-cited objective for digital transformation was to become more operationally efficient.[1] For many, leveraging cloud resources is part of this transformation; it is faster and easier to spin up compute instances, storage, and applications in the cloud than to buy, build, and manage them on-premises. In addition, with cloud computing, organizations pay only for what they use. ESG research demonstrates that public cloud infrastructure adoption has almost doubled in the past five years, with 94% of respondents currently using public cloud services.[2]

Leveraging containers orchestrated by Kubernetes is a continuation of that theme, enabling engineers to speed development and add features to applications more quickly. In ESG research, 77% of respondents are currently using containers in production applications, with another 21% planning to in the next year.[3]

Figure 1. Efficiency, Public Cloud Services, and Containers



Source: Enterprise Strategy Group

However, the reality of cost reduction from these operational improvements often trails expectations. Most organizations are acutely aware of the difficulty of managing cloud and container infrastructure costs. It is much too easy to ensure scalability by overprovisioning, resulting in wasted resources, but it is much too difficult to really understand how effectively resources are being used.

- Are you spending too much for what you are getting?
- If you spend more, will you get the performance, availability, or other business objectives you want?

[1] Source: ESG Research Report, *2021 IT Spending Intentions Survey*, January 2021.
[2] Ibid.
[3] Source: ESG Master Survey Results: *The Maturation of Cloud-native Security: Securing Modern Applications and Infrastructure*, June 2021.

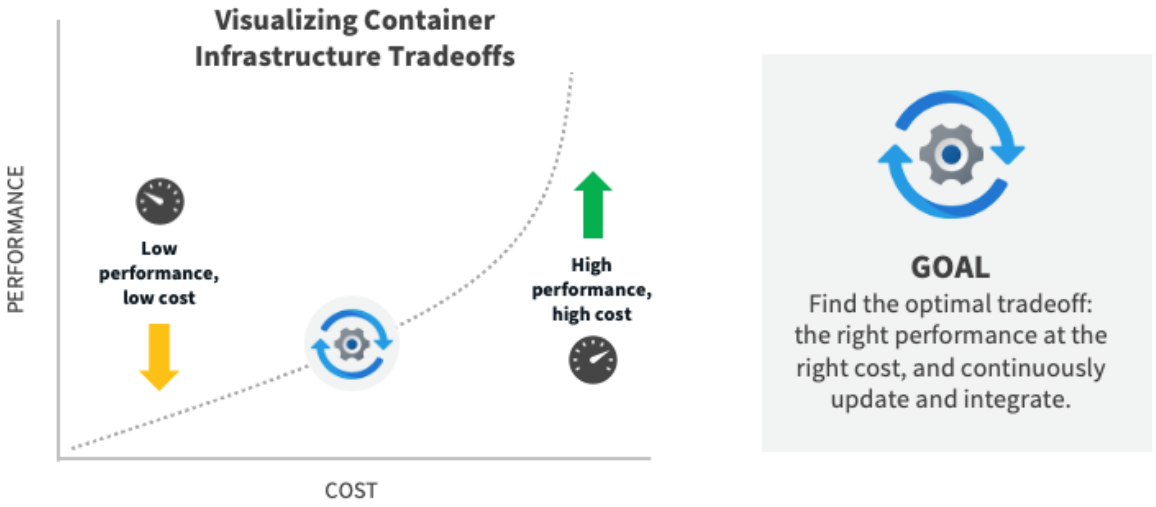
While there are tools to measure performance, application availability, and costs, it is up to administrators to correlate this information and make decisions about infrastructure spending. This is mostly guesswork. Most default to overspending—whether they realize it or not—to protect against application performance and availability lapses.

Many have turned to containers for agility and speed, but managing containers and Kubernetes orchestration is complex, and IT administrators struggle to efficiently manage container environments. When asked about their biggest challenges managing container-based application environments, the top responses included managing in a hybrid cloud, costs, lack of skills, troubleshooting, and orchestration.[4]

Challenges of Kubernetes Deployments

Container deployments are complex. For each container, IT must weigh multiple interrelated factors, such as the right memory requests and limits, CPU requests and limits, replicas, and myriad application-specific settings such as JVM heap size, garbage collection, file buffers, etc. Using the manual trial-and-error method—test, change the code, test again—this task is virtually impossible to do well across thousands of containers. The impact of misconfiguration can be serious, impacting users with poor performance and availability. Many organizations default to overprovisioning because it is so difficult to understand the impact of changing resource allocations. Developers end up in reactive mode, having to troubleshoot container infrastructure instead of focusing on application innovation.

Managing container infrastructure is a game of cost tradeoffs, made even more challenging by having multiple objectives. Take an example with just a single metric: performance. To ensure performance, IT can provision extra resources for every container. This may optimize the end-user experience but often results in idle extra resources that increase costs. Or IT can maintain a cost threshold but fail to deliver an appropriate user experience.



The simplified graph at right demonstrates IT's objective: find the right tradeoff of cost versus performance. This seems simple enough, *but in a Kubernetes deployment there are as many graphs as there are metrics*. A complete picture must combine performance, utilization, memory, CPU, replicas, network bandwidth, application-specific metrics, *ad infinitum*. And this is only for a single container; when you scale to thousands of containers for different applications and services, it is an impossible task for humans to manage optimally by testing individual metrics. And even if you get close, you still need to continually optimize as your application changes. In complex Kubernetes environments, developers often end up spending time doing manual, iterative testing, one parameter at a time, and then take their best guesses about the optimal configuration.

Organizations need better visibility into Kubernetes infrastructure and interdependencies underlying their applications and a systematic, scalable way to implement changes to assure the achievement of business goals. They need a fast, easy way

[4] Source: ESG Master Survey Results, *Trends in Modern Application Environments*, December 2019.

to intelligently allocate Kubernetes resources to deliver on their performance, availability, and other goals, while keeping costs down. And they need this not just once, but continually as applications are reworked and improved.

StormForge: Performance Testing and Resource Optimization

StormForge is a cloud-based solution that uses machine learning and integrated performance testing to recommend optimizations for your Kubernetes application deployments. There are two key parts of the StormForge solution: the performance engine and the optimization engine, both of which run in AWS. StormForge delivers iterative experiments to help organizations understand and visualize the tradeoffs between multiple metrics—such as performance, availability, time, and costs—so they can make the best choices for their business needs. Once an experiment completes, customers choose the data point that best meets their goals and can then easily export the optimal configuration from StormForge. StormForge is priced by the number of namespaces it supports.

StormForge can eliminate time-consuming, manual testing and tuning, helping organizations make better Kubernetes resource decisions, achieve their business objectives, reduce costs, and keep developers focused on innovation instead of manual tuning.

Figure 1. StormForge

Source: Enterprise Strategy Group

StormForge is:

- *Proactive*. Experiments are done in a preproduction cluster, so IT can identify the optimal configuration before implementing it. Organizations gain insight into how applications will perform with various resource allocations and how much those allocations cost. Building this into the CI/CD process ensures optimization as applications change.
- *Single platform*. Load testing and optimization are built into a single platform, so no integration is needed. StormForge also supports importing an organization's current load testing into the optimization engine.
- *Automated and machine-learning-driven*. Machine learning drives rapid load test experiments across multiple parameters. Tests are created in minutes and scale to hundreds of thousands of requests and millions of concurrent users; results of each trial inform the next one.

- *Visual*. StormForge creates the right test, automates as it learns from each iteration, and delivers a visual representation of the optimal configuration as well as numerous options. Users click on each data point to identify and export the configuration.
- *Purpose-built for Kubernetes.* StormForge works on any CNCF-certified Kubernetes distribution.

ESG Technical Validation

ESG viewed a demo that was run on a standard Google Kubernetes Engine (GKE) cluster composed of six instances, each with 8 vCPUs and 8GB RAM located in the US Central region. The demo focused on the ease of experiment setup and the results that can be obtained.

Setting Up and Running Experiments

Installing StormForge involves a simple download of a CLI tool to a system with network access to a pre-production Kubernetes cluster on premises or in the cloud. Once credentials are added, the StormForge controller is authorized and installs on the cluster to orchestrate experiment trials. A web-based GUI can be used to easily manage the optimization function.

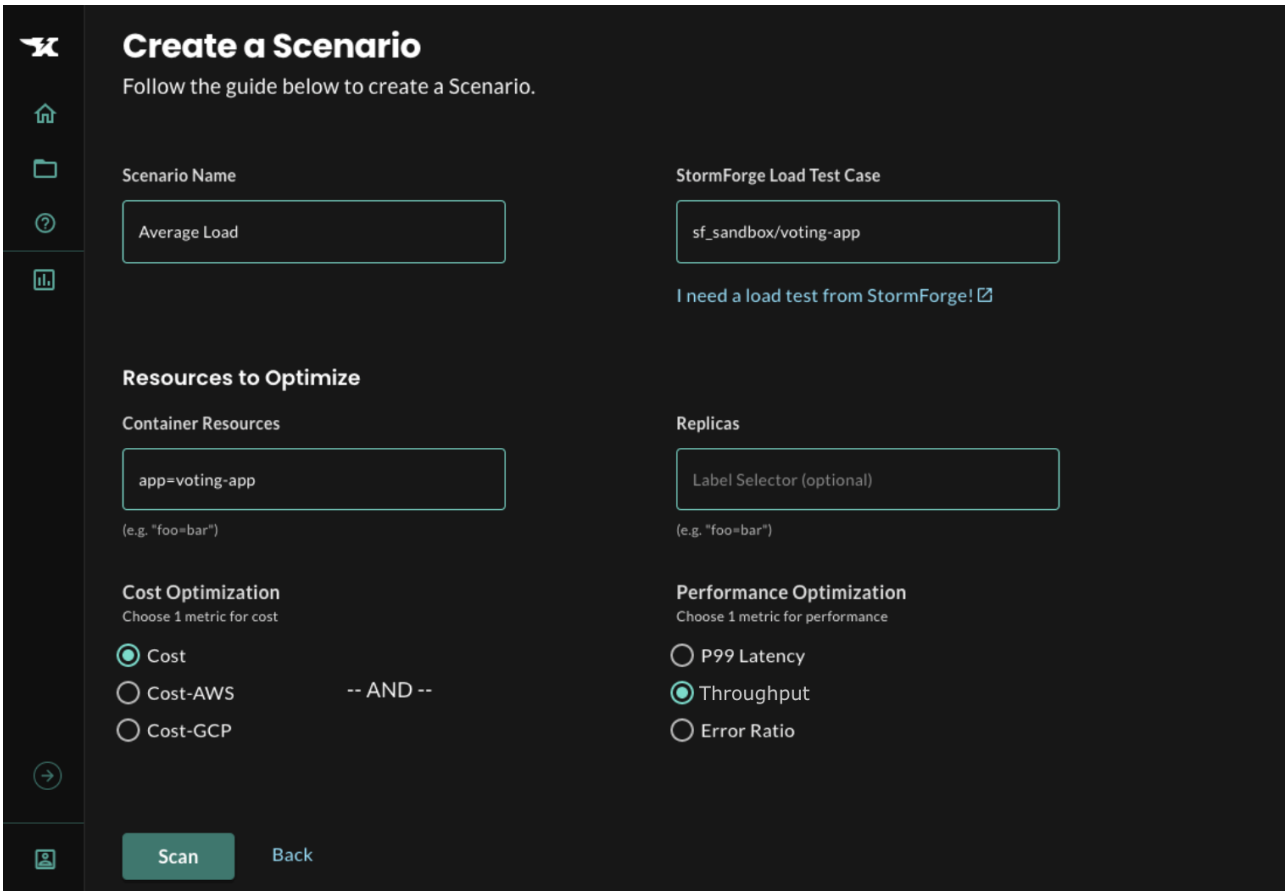
The StormForge wizard helps users set up performance testing and optimization. First, users define where the load testing will come from: StormForge, Locust config files, or APIs to customer performance testing that can be integrated using the Pod template option. StormForge testing as-a-service is located in AWS data centers and can scale to a million transactions per second.

ESG Testing

ESG viewed a demo showing creation of an experiment on a voting web app that utilized numerous microservices, including Redis, Postgres, and .NET. The performance testing created 300 clients executing a workflow on the application over two minutes, with two voting boxes: one for cats and one for dogs. After selecting the Vote namespace to test, we completed test set up.

Users have the choice to test all microservices making up the application or only selected ones; we selected all the microservices in the app. Figure 3 shows the completed UI screen to set up this scenario, sending an average load to the voting app, and optimizing for cost (calculated as a function of CPU and memory utilization) and throughput. Note that this doesn't just tune a single microservice such as Redis; StormForge creates traffic on the front end to test and tune all the "knobs" down the chain.

Figure 3. StormForge Experiment Setup



Source: Enterprise Strategy Group

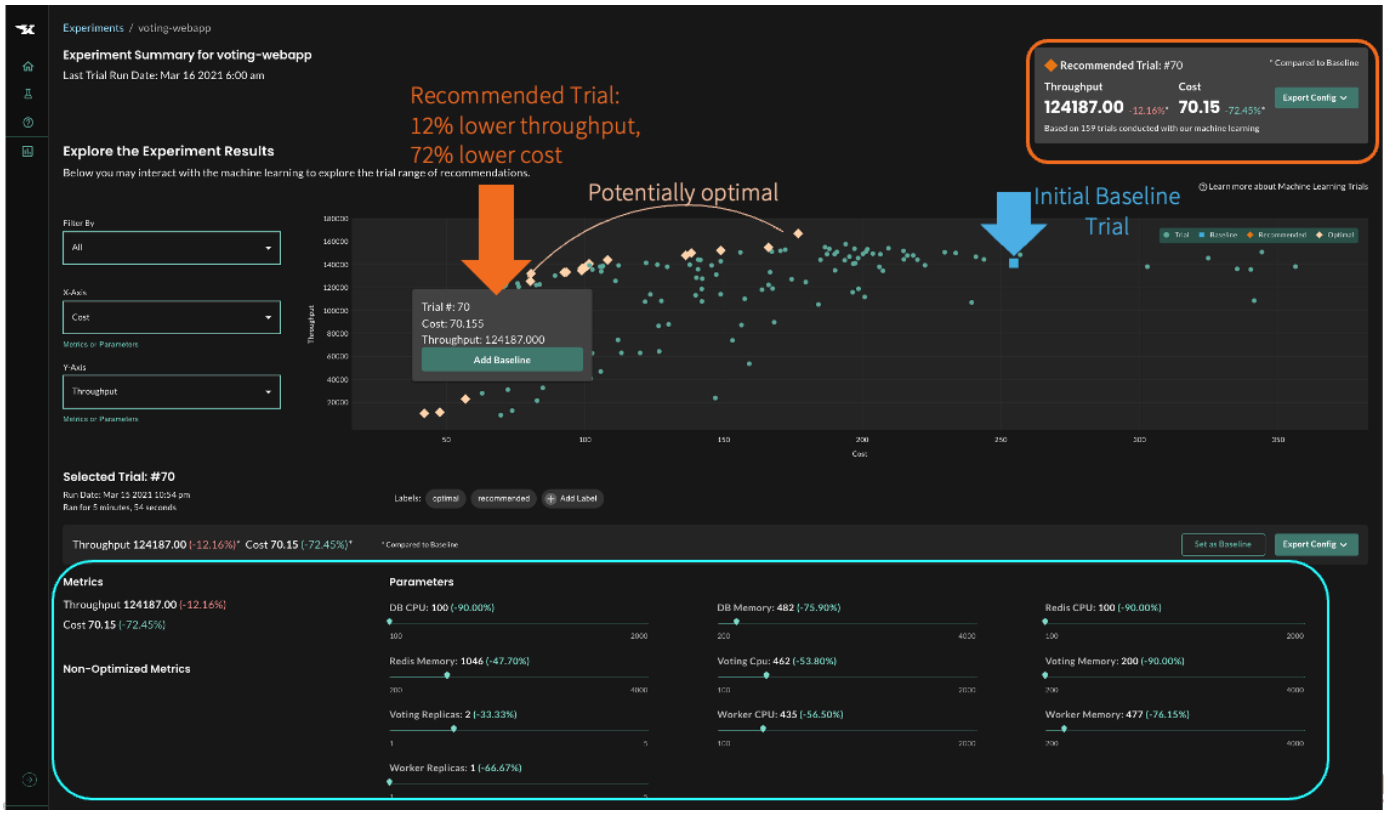
Users may also tune other baseline parameters such as CPU, memory, replicas, etc. We executed the test with the click of a button, and StormForge patched the configuration in place, stabilized, and then began to run, scraping metrics from the cluster. We watched the CLI as metrics were captured, and the machine learning engine used those results to create another series of parameters for the next trial. This is the experiment process that happens iteratively over the number of trials desired.

Figure 4 shows the results that StormForge delivered. The graph shows cost and throughput plotted, with each point representing a trial—that is, a set of parameters imposed on the application by the machine learning. The blue square indicates the initial baseline test run; this is the comparison configuration and often represents the over-provisioned config in use. The green dots represent different trials.

Light orange diamonds indicate trials that are potentially optimal; this pareto curve, as it is called, is a feature of multi-objective optimization that machine learning can generate. Those diamonds display the most and least of the two parameters plotted: cost and throughput. The dark orange diamond (hidden in the figure) shows the recommended configuration, which is defined as the middle of that pareto curve. The details of its cost and throughput are displayed in the upper right corner. In this case, the recommended configuration was in trial #70, which delivered 12% lower throughput but 72% lower cost, when compared to the baseline configuration. If this trial is chosen, the user simply clicks the green ***Export Config*** button in the upper right corner to deliver the recommended changes. Also, settings for the non-optimized metrics for Trial #70, which could be tuned in a different experiment, are displayed below the graph.

One of the key features of StormForge is the ability to select other trial results as needed. In this example, the recommended configuration reduced throughput slightly but generated the greatest cost reduction. But what if any throughput degradation is unacceptable? The user can simply pick a different trial, one of the light orange diamonds. For example, trial #63 boosts throughput 2% over the baseline but still reduces costs 57%. This could be the best option if throughput cannot be reduced and would still significantly reduce costs.

Figure 4. StormForge Voting App Experiment Summary



Source: Enterprise Strategy Group

Notes:

- The filters and both axes can be changed using the left drop-down arrows. For example, we could change the X axis to look at the number of replicas, Redis CPU, database memory, or worker CPU in these trials with a single click.
- The time savings are enormous with StormForge. To test 200 trials, changing the code and pushing it out daily, would take 200 days. With StormForge, that might take hours.
- Any application-specific parameters such as heap size or file buffers can also be included for testing.
- A close look at StormForge results can provide unexpected insights. For example, StormForge can point out parameters that make no difference in what users are trying to optimize, so those parameters can be removed from consideration. StormForge can also identify unstable configurations that would likely fail in production.
- Machine learning includes no implicit bias, and therefore, developers are sometimes surprised by results. For example, a developer may know that giving Redis more memory will always improve performance. That may be true,

but the developer does not know how much value that extra cost is adding—if it is adding 1% to performance but 30% to cost, that is a decision best made knowingly.

- StormForge mentioned a customer whose nine-person development team spent more than a year of repetitive trial-and-error testing to tune its Kubernetes environment. This customer ran StormForge for two days and found 50% reduction in cost for the same performance, in part because their implicit bias neglected to review certain parameters.

Why This Matters

Optimizing Kubernetes resources can save companies millions, but the manual method of trial-and-error wastes time, uses engineering expertise that could be used more wisely, and fails to adequately take business objectives into account.

StormForge helps organizations with "what if" analyses–how will my application behave if I increase or decrease Kubernetes resources? How can I optimize Kubernetes resources to achieve my cost, performance, and other business goals?

ESG validated the ease and speed of experiment set up, load testing, and resource optimization with StormForge. StormForge delivers the analysis; then with a click of a button, IT can export the configuration that will deliver the desired results.

StormForge eliminates the blind spots and highlights solutions to Kubernetes issues. This lets organizations make truly data-driven decisions. While 2% better performance might mean little to one company's batch process, it could mean a significant revenue increase for another company's multi-user application.

Customer Story: Trilio Saves Time and Money with StormForge

ESG spoke with Prashanto Kochavara, the Director of Product at Trilio, a company that offers data management and data protection for cloud-native applications. With a Kubernetes-centric architecture, Trilio provides services including backup, recovery, data mobility, migration, and ransomware protection for customers. Trilio's solution has a subscription-based licensing model based on the number of compute nodes, VMs, vCPUs, or clusters.

Trilio implemented StormForge to right-size its Kubernetes deployment and ensure that development moves quickly. Using StormForge for regression, scalability, and stress testing helps Trilio to be sure its services are hyper-scalable for customers. Kochavara estimates that StormForge with machine learning delivers 3-4x in time savings compared with the company's standard process. Said Kochavara, "Anywhere we can save time, there is a direct correlation to lower cost. StormForge saves us time and money. In my opinion, any development team should be using it." Trilio includes StormForge in the development process to ensure that new versions continue to achieve the same efficiency. He also commented that developers are much happier since they spend less time executing cumbersome, repetitive optimization tasks.

"Anywhere we can save time, there is a direct correlation to lower cost. StormForge saves us time and money. In my opinion, any development team should be using it."

P. Kochavara, Director of Product, Trilio

Trilio saw another potential benefit that StormForge could deliver to customers. Trilio sees StormForge as a way to analyze its customers' environments and provide guidance for maximizing efficiency for cloud backups, restores, captures, and mobility activities. Using StormForge, Trilio provides multiple cost/resource options and lets its customers discover the optimal crossover point between resource allocation and benefits. With StormForge, Trilio can help customers optimize their recovery point and recovery time objectives to get the most value from their data management and protection policies.

In summary, Kochavara said, "Faster development—that's why Kubernetes is here. StormForge definitely aligns with that overarching philosophy—and complements it."

The Bigger Truth

Kubernetes at scale is extremely complicated. Because of that, organizations waste millions on inefficient deployments and still don't get the performance and availability they want. While organizations use multiple performance and cost tools, they don't really know how tweaking the levers will impact results. A common response to problems is overprovisioning, adding cost without necessarily fixing anything. As a result, cloud costs continue to rise and take up an increasing portion of the cost of revenue.

A smarter way is to make data-driven decisions—but simply looking at a dashboard to see that performance is low doesn't help. You need to know why it is low, but most importantly, you need to know *what you can change to deliver the performance you need, how that might impact other parameters, and what the overall cost will be.*

The purpose of Kubernetes is to speed development and increase business agility. But, like any objective, there is always a cost/benefit analysis in play. The challenge is to figure out what matters most to your business—performance, availability, speed of updates, etc.—and be able to automatically tune multiple inputs to make sure you achieve those objectives at the lowest possible cost.

"Faster development—that's why Kubernetes is here. StormForge definitely aligns with that overarching philosophy—and complements it."

P. Kochavara, Director of Product, Trilio

ESG validated that StormForge offers an advanced tool for this job, leveraging machine learning to rapidly analyze multiple dimensions. StormForge can:

- Test myriad configurations, analyze them, and find the optimal cost and performance profile for your Kubernetes deployment.
- Display this profile visually along with other configuration options, so you can make an informed decision.
- Export the desired configuration with the click of a button.

Of course, any potential solution must be tested in your environment with your applications to see if it services your needs. But if your organization wants to see how Kubernetes applications behave under load—before putting new versions into production—then ESG recommends a close look at StormForge. Instead of waiting until a threshold is tripped and end-users are experiencing problems with your platform, you can ensure the right Kubernetes resources while eliminating waste. And, once you start looking at your applications this way, you might just want StormForge to be the last quality gate before each new code release.

The goal of ESG Validation reports is to educate IT professionals about information technology solutions for companies of all types and sizes. ESG Validation reports are not meant to replace the evaluation process that should be conducted before making purchasing decisions, but rather to provide insight into these emerging technologies. Our objectives are to explore some of the more valuable features and functions of IT solutions, show how they can be used to solve real customer problems, and identify any areas needing improvement. The ESG Validation Team's expert third-party perspective is based on our own hands-on testing as well as on interviews with customers who use these products in production environments.

Enterprise Strategy Group is an IT analyst, research, validation, and strategy firm that provides market intelligence and actionable insight to the global IT community.

 www.esg-global.com

 contact@esg-global.com

 508.482.0188