

# How OpenTable Optimized Kubernetes Troubleshooting with Komodor



**Company Size:** Over 2000 employees

**Industry:** Restaurant Tech

**Komodor Installation:** Over 300 product engineers

## About OpenTable

OpenTable is a leading provider of online restaurant reservations, part of Booking Holdings. They connect diners with restaurants, helping to fill seats and provide a seamless reservation experience. OpenTable operates a large and complex infrastructure, heavily reliant on technology to support its platform and services.

The engineering teams, including a sizable product engineering team of around 300 members, work to maintain and enhance the platform. Michael B., Staff Site Reliability Engineer and manager of two teams, including Build and Release Engineering, has been with OpenTable for five years. His teams manage infrastructure tooling and the deployment pipelines, heavily leveraging Kubernetes for their infrastructure.

## A Lack of Effective Troubleshooting in K8s

OpenTable was migrating from Singularity and Mesos to Kubernetes to enhance security for its engineers. While transitioning to Kubernetes, they lost a vital feature: the real time visibility into the operational state of deployments that Singularity's UI provided.

Engineers could no longer easily troubleshoot deployments in real-time. Deployments would often fail with little to no feedback, leaving engineers in the dark about the root cause. The only information they received was a "succeeded" or "failed" message, which was insufficient for effective troubleshooting.

Furthermore, upon failure, the system automatically cleaned up, erasing crucial evidence needed to diagnose the issue. The Kubernetes Dashboard offered visibility into the raw state of Kubernetes but fell short in key areas—historical logs and investigative analysis for troubleshooting were notably absent. Even for highly skilled engineers, the lack of these critical features made resolving issues more challenging and less efficient.

## Reduce Reliance on Platform Team

- Engineers needed to understand the interim phase, especially during deployment failures or pending deployments.
- Existing metrics and logging solutions were ineffective during the application's initial bootstrap phase.
- OpenTable aimed to empower engineers with self-service troubleshooting and reduce platform team support tickets.
- There was a need to minimize Mean Time to Recovery (MTTR) and enhance the developer experience.
- The platform team was overwhelmed with questions like, "Why is this not working?"
- Finding a tool to make deployment troubleshooting more accessible and user-friendly was crucial.

## Achieving Visibility and Self Service Troubleshooting

OpenTable adopted Komodor to address these challenges. Michael B. conducted the initial proof of concept and heavily advocated for its implementation due to its ability to fill the observability gap. Key benefits included:

- **Enhanced Post-Deployment Visibility:** Komodor provided the necessary insight into the deployment process, particularly during the critical phase between deployment and live application. This allowed engineers to quickly identify and diagnose failures and pending issues.
- **Developer Empowerment:** Komodor simplified Kubernetes troubleshooting, making it accessible to engineers without requiring them to be Kubernetes experts. The intuitive interface and detailed event tracking significantly reduced the burden on the platform team, leading to increased self-service capabilities.
- **Faster Troubleshooting with Git Annotations:** Komodor's annotation of deployments with Git hashes for the application source code, deployment values, and base Docker images proved invaluable. This feature allowed Michael to quickly resolve a two-hour long deployment issue in under five minutes by pinpointing a problematic base image patch version.

## Improving Efficiency with Faster Incident Resolutions

- **Faster Troubleshooting with Git Annotations:** Komodor's annotation of deployments with Git hashes for the application source code, deployment values, and base Docker images proved invaluable. This feature allowed Michael to quickly resolve a two-hour long deployment issue in under five minutes by pinpointing a problematic base image patch version.
- **Reduced MTTR:** While precise metrics weren't tracked, the time taken to resolve deployment issues drastically decreased due to Komodor's clear and comprehensive event visibility. The platform team actively shared Komodor links when assisting engineers, further evangelizing the tool and promoting its use.
- **Improved User Experience:** Engineers were genuinely pleased with Komodor, finding it much more user-friendly and helpful compared to the Kubernetes dashboard. The pictorial representation of events and co-related deployments made it easy to understand and comprehend what was happening in the environment.
- **Overall Efficiency:** With Komodor, OpenTable saw a reduction in the volume of support questions related to deployment issues. The tool enabled quicker problem identification and resolution, thus streamlining the workflow and improving overall efficiency.

Komodor became a crucial tool for OpenTable, significantly enhancing their Kubernetes adoption and operational efficiency by providing robust troubleshooting and observability capabilities.



**Michael B.**  
OpenTable

*"Komodor has improved the user experience for engineers, who were previously relying on the Kubernetes dashboard. After Komodor was introduced, we (the platform team) started providing links to Komodor when helping engineers, which led to a reduction in the number of we received, as the engineers were able to self-serve more using Komodor."*