turbonomic

CHECKLIST

Enterprise K8s Readiness Checklist

While Kubernetes may have graduated to enterprise-ready, there are a number of things to consider before you can claim enterprise readiness within your organization. Perhaps it seems like a lifetime ago since you stood up your first Kubernetes cluster. The first application has been rolled out and that light house line-of-business is happy. Congratulations! Your success has opened the floodgates to more lines of business wanting the speed, agility, and elasticity that Kubernetes promises. It's now on you to deliver these benefits at scale. Here are the top 3 things you need to think about…

#1 Culture

As we discovered in our 2020 State of Multicloud Report, you have to tackle culture before you can tackle anything else. At the end of the day it's people that make or break the success modernizing your applications and platforms. And, the rapid, iterative development of cloud native, in particular, demands DevOps mindsets and methodologies. **What are you doing to bridge the all too pervasive gap between application and infrastructure teams?**

Every team has their domain expertise, but the silos in skill sets can foster siloed teams, especially if you're relying on disparate, disconnected tools. Instead, consider platforms and solutions that bridge your applications and infrastructure teams. For example, Turbonomic Application Resource Management continuously ensures that applications get the resources they need to perform. A side benefit is that it provides complete visibility into the application stack—from the service to the node to the host. Alternative solutions can only provide views into certain parts of the stack. If teams have no single source of truth, it's nearly impossible to effectively collaborate.

Ask yourself:

- ☐ Do my teams have complete visibility into how our applications get the resources they need to perform?
- ☐ Can my Application and Dev teams use the same platform as my DevOps, SRE, Operations, and Infrastructure teams?

turbonomic

#2 Complexity

Complexity takes many forms when you're operating at scale.

A. Multitenancy

The best practices around managing resource requests provide some manageability, but at scale there are challenges. Because resource requests specified for a service are estimates, it is very easy for a cluster to get into the following situations:

- **Overestimation of resources**, where estimated resource usage is significantly higher than the actual usage. Understandably, Developers tend to be overly conservative about their resource needs as a way to ensure that their service will not suffer.
- **Underestimation of resources**, where the actual resource usage is higher than the requested usage. It can result in noisy neighbor problems, or even worse overcommitted nodes, where pods may be evicted due to resource pressure (for incompressible resources like memory). This resource contention results in poor application performance and poor experiences for customers.

The crux of the issue is that Developers only see and have responsibility for their service. A multitenant cluster will host many services, each with their own specifications that do not account for the compound effect of running together. It's a very difficult challenge. Both DevOps and SRE teams worry about it. It's hard to see the impact of overallocation and where to optimize when searching across hundreds, or even just 50 different services running in different projects. DevOps and SREs BOTH need to understand whether 1) Services are over allocated, 2) The project/namespace is under-allocated, 3) The cluster is undersized. Requests are that silent "killer" to elasticity and efficiency. Learn more about Best Practices for Managing Multitenancy.

Ask yourself:

- ☐ How am I ensuring that application services are being allocated only what they really need?
- ☐ Does this approach scale? In other words, can I apply this approach across 50 different services? 100? 1000?

B. Multicloud

One of the key advantages of Kubernetes is its portability; it can run on any cloud or infrastructure. However, organizations are finding that even though Kubernetes can run anywhere, the management tools are not always built to operate anywhere. Each cloud offering has their own dashboards and views. It's useful to the teams directly using those tools, but doesn't provide a global view of all your applications across all your infrastructure.

Ask yourself:

- ☐ Do my teams have a global view of all our applications across all our infrastructure and clouds?
- ☐ Do my teams understand how every application gets the resources it needs to perform?
- ☐ If not, how many war room meetings could we avoid if we did?

C. More speed, more change

Containerization has enabled developers to rapidly and continuously deploy new features and updates to their applications. Smaller, but more frequent development releases lead to more changes to environments than ever before. Kubernetes will do initial placement--there a pod lives until it dies. Managing constant change in production requires continuous analysis of application demand and redistribution of load to ensure that services continuously perform.

Ask yourself:

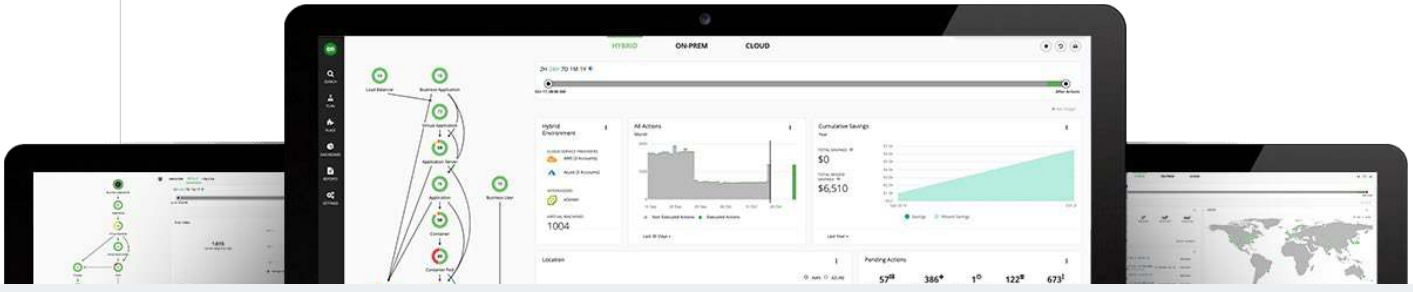
- ☐ How are we ensuring that newly deployed services always perform?
- ☐ How are we ensuring those same services don't disrupt existing services running in the environment?
- ☐ Do we have a “feedback loop” between how services run in production and how they are configured at deployment?

#3 Skills & Training

With the cloud native landscape rapidly evolving, it's no surprise that the skills gap has consistently come up as a top challenge for organizations. Leaders looking to drive change must invest in training their teams or at least creating an environment where dedicating time to self-education is encouraged and valued. It also helps to invest in the types of automation that free teams from manual labor. When it comes to managing application resources, orchestration, scheduling, and even autoscaling require some level of decision-making from people.

Ask yourself:

- ☐ How am I ensuring that my teams get the proper skills training to excel in Kubernetes management?
- ☐ Can we leverage solutions that take the burden of resource decision-making off people?



Bonus Step: Automate with Turbonomic

See how you can rely on AI-powered software to automatically allocate resources continuously in real-time. Visit turbonomic.com/download to start automating today.