Architecture Decision Record: Kubernetes container orchestration

Problem Statement

We need to select a container orchestration platform for our growing cloud-native application portfolio. Our current legacy platform deployment is too slow, and not agile enough to keep up with our growing needs. We're looking for a system that will allow us to scale our services in the most efficient way possible without compromising on agility or ease of use.

Considered Alternatives

1. Docker Swarm
2. Kubernetes
3. Apache Mesos

Decision Made

After conducting a thorough analysis of each container orchestration platform, we have decided to adopt Kubernetes as the best option for our enterprise needs. Our reasons for choosing Kubernetes are as follows:

1. **Scalability:** Kubernetes's unique design is perfect for scaling applications, and as our requirements for scalability evolve over time, Kubernetes has the built-in ability to meet these changes without any issues.
2. **Decentralized Architecture:** Kubernetes' master-worker topology ensures a decentralized architecture which ensures that there is no single point of failure.
3. **Community Support:** Kubernetes has the largest and most active open-source community, which means it has a large number of contributors, developers, and vendors making it easier for us to get help and find resources.
4. **Ecosystem Support:** Kubernetes has a growing ecosystem with a variety of third-party tools, integrations with container registries, CI/CD pipelines, data storage, and more.

Therefore, we have decided to adopt Kubernetes as our container orchestration platform for the present and the immediate future.

Credit: this page is generated by ChatGPT, then edited for clarity and format.