lastsheet9 lastpage7

Cloud computing solutions like on-demand virtual machines or online software services continue to grow in popularity. Deploying their service on cloud infrastructure offers various benefits to software providers. These software providers then face the challenge to minimize the amount of resources used while still meeting their customer's requirements. Several frameworks to manage resources and applications in a distributed environment are available, but their development is still ongoing and the state of the art is rapidly evolving, making it a challenge to use such frameworks and their features effectively in practice.

The goal of this thesis is to research how applications can be enhanced with adaptive performance management by relying on the capabilities of Kubernetes, the de-facto standard platform for container orchestration. In particular, the Metrics API for monitoring of resource usage metrics and horizontal as well as vertical scaling concepts of Kubernetes may prove useful to support adaptive resource allocation. Moreover, concepts for oversubscription as a way to simulate vertical scaling without having to reschedule pods, will be evaluated.

Through a series of experiments involving multiple applications and workloads, the effects of different configurations and combinations of horizontal and vertical scaling in Kubernetes are explored. Both the resource utilization of the nodes and the applications' performance are taken into account. First, the effect of different oversubscription configurations is described. Second, the impact of scaling applications using the default Kubernetes autoscaler is analyzed. Finally, the impact of using the default Kubernetes horizontal autoscaler when multiple pods are co-located, is evaluated.

The results show that providing appropriate pod resource configurations greatly impacts an application's performance. Moreover, the type of test application is shown to have an impact on the performance gain of scaling. A log management application using Cassandra is shown to not scale well using the default Kubernetes horizontal autoscaler. A significant Kubernetes-specific overhead decreases the application's performance when replicas are added. For another application, the autoscaler is proven to be effective when the user workload is seasonal. When the workload is bursty, however, the autoscaler is shown to be inconsistent and ineffective. Furthermore, co-locating a low and a high priority application has a minimal impact on the high priority application's performance, while increasing resource utilization in general. This has been confirmed for two types of applications. In summary, providing suitable resource configurations increases cost-efficiency without any major downsides, but the effects of using the HPA depend on the type of application and the user workload.