Containerization

Case Study

gslab

Reducing costs by 40% and improving scalability by moving to a Kubernetes environment

A WAN management product company migrates from a virtual machine based microservice environment to Kubernetes

Executive Summary

In today's modern tech stack, the move from Virtual Machines (VMs) to Containers is an obvious first step. It is crucial to adopt the right strategies while moving towards microservice based architectures. This success story covers one such modernization from VMs to K8s.

Overview

Our customer offers a software-defined solution for managing and configuring WANs using a centralized cloud-based interface. They offer a hybrid solution with minimal hardware setup on the customer premises. Most of the infrastructure is hosted on AWS cloud.

As their user base scaled up, they faced significant challenges in handling, sustaining and expanding VM instances. Hence, they decided to migrate their VM-based microservices to Docker and K8s based Architecture.

GS Lab's rich experience in the networking & cloud domains, helped the customer to prepare, strategize and mitigate the risks of migrating the large legacy VMs. We successfully migrated five clusters in one year, each consisting of more than 25 microservices hosted by more than 50 AWS instances.

Challenge

Rapid growth in user base in different parts of world led to extensive growth of cloud instances. It was hard to scale with the existing set up. These were some of the most prominent challenges we faced:

1. Manual provisioning of infrastructure

Manual provisioning introduced human errors. Manual ad-hoc changes over time, led to differences in intermediate environments. This in turn caused unexpected issues and failed deployments. Bringing up a new cluster was a nightmare due to the lack of a fully automated system for handling infrastructure, provisioning and configuration management.

2. High infrastructure cost

There were many more AWS instances on a cluster than the number of microservices. As the number of users increased, the cost of buying and managing these instances spiraled out of control. There were two primary reasons for high infrastructure costs:

- **AWS instance count:** A high number of unmanaged instances led to high costs for the customer. This also increased the cost spent on human capital maintaining these instances.

- **Underutilization of instances:** The microservices were not utilizing the AWS instances optimally a which led to the customer paying extra for unutilized resources.

3. High downtime

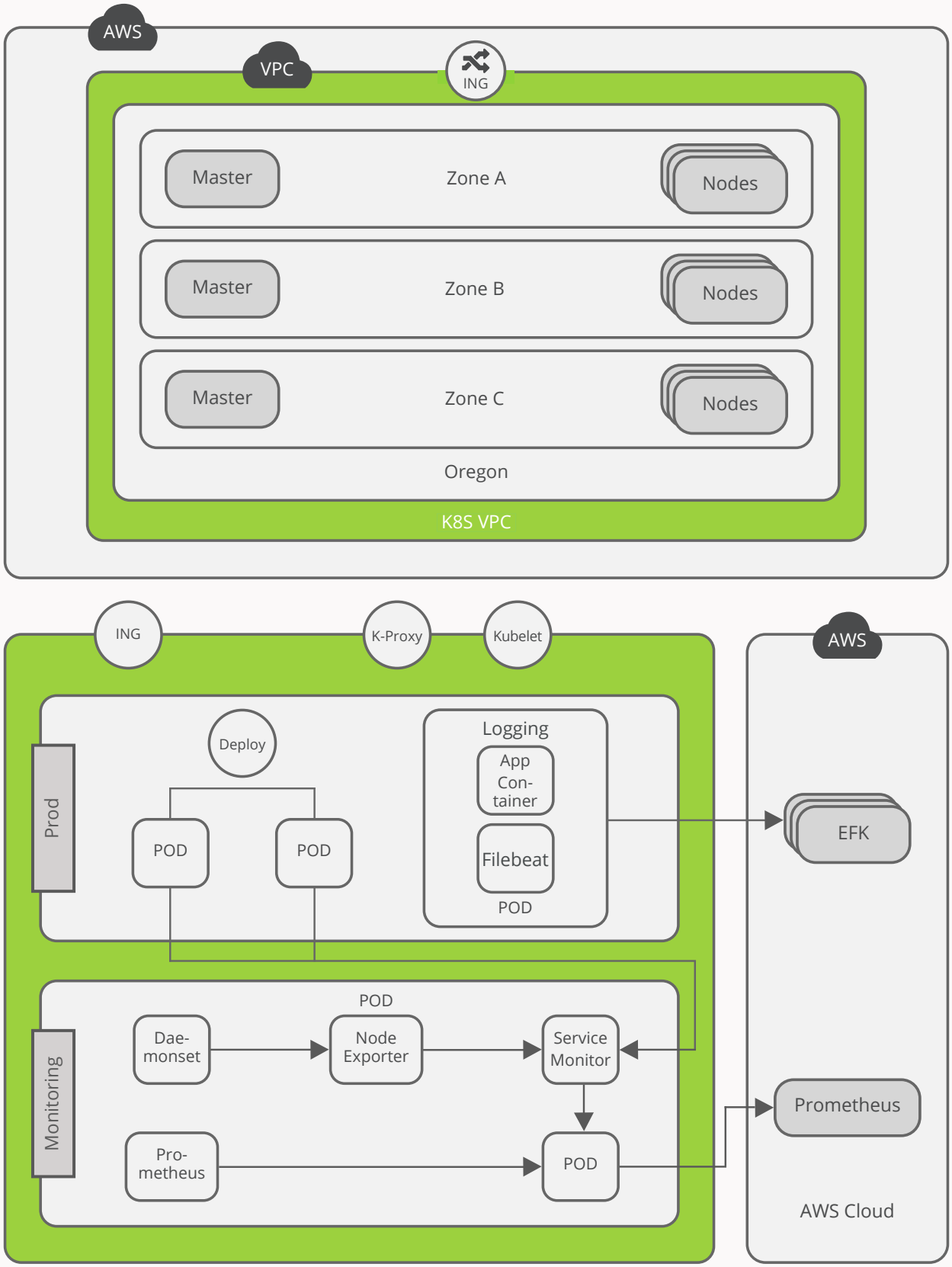
It took hours to deploy the product. The rollback process was complicated and time-consuming, which added to even loner downtimes.

4. Tiresome manual scaling (elasticity)

Autoscaling (horizontal/vertical) was not implemented in the solution. This led to manual scaling of instances which was tedious and unnecessary. Controller services were needed to grow and shrink the solution based on auto configurable parameters.

GS Lab has a rich history of helping networking product companies in developing SDN, SD-WAN, and NFV based high-performing, scalable solutions. Our networking, cloud migration and DevOps experience helped our customer in creating a much simpler and easy-to-use solution.

Considering the various challenges faced by our customer, GS Lab proposed and implemented the following solution:



Solution architecture

Our solution covered the following requirements:

Dockerizing the individual applications: We wrote Docker files for all microservices. Services were installed in docker images as Debian packages using APT.

Bringing up K8s clusters: We used Kubernetes Operations (Kops) to bring the K8s cluster into AWS cloud. Kops is an open-source project that enables fast and swift setup of Kubernetes clusters.

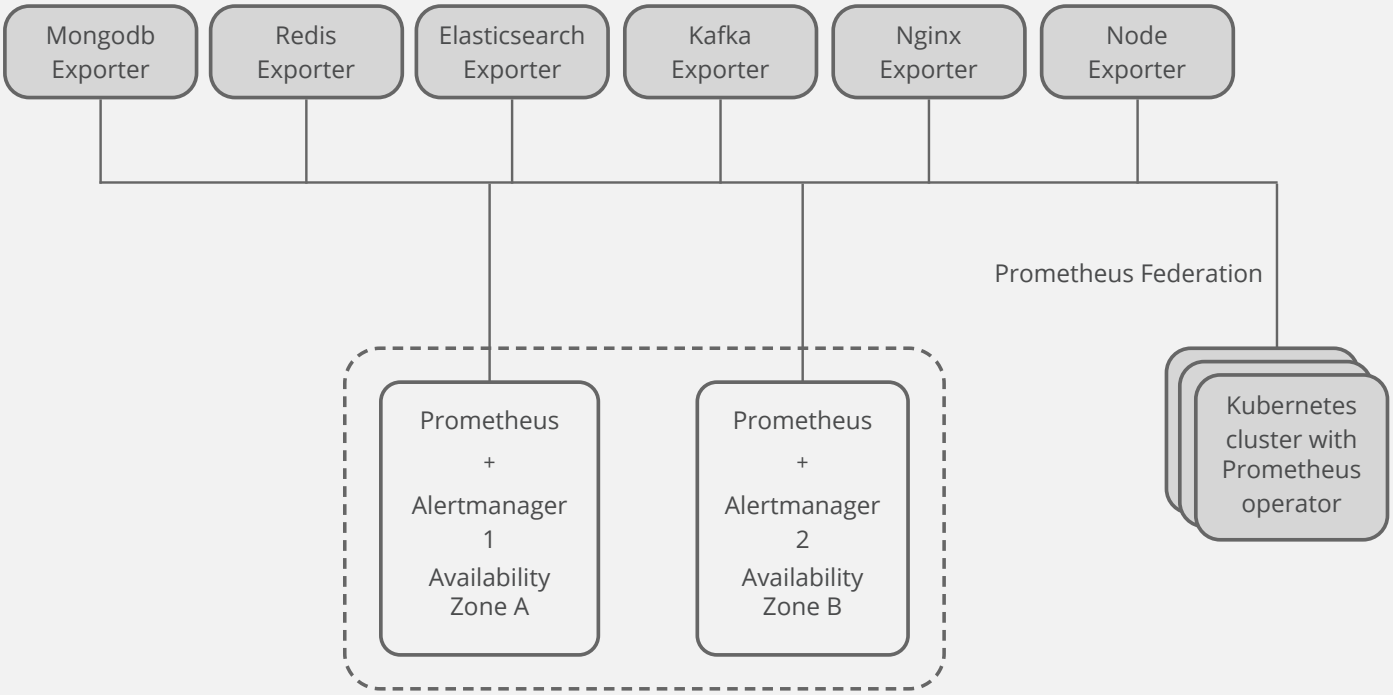
Application connection: We used K8s service-types such as NodePort, LoadBalancer, ExternalName and ClusterIP to connect the applications internally and to the database.

K8s networking: We used Weave networking solutions in our environment. Using Weave, we created a mesh overlay network between each of the nodes in the cluster and enabled versatile routing between the participants.

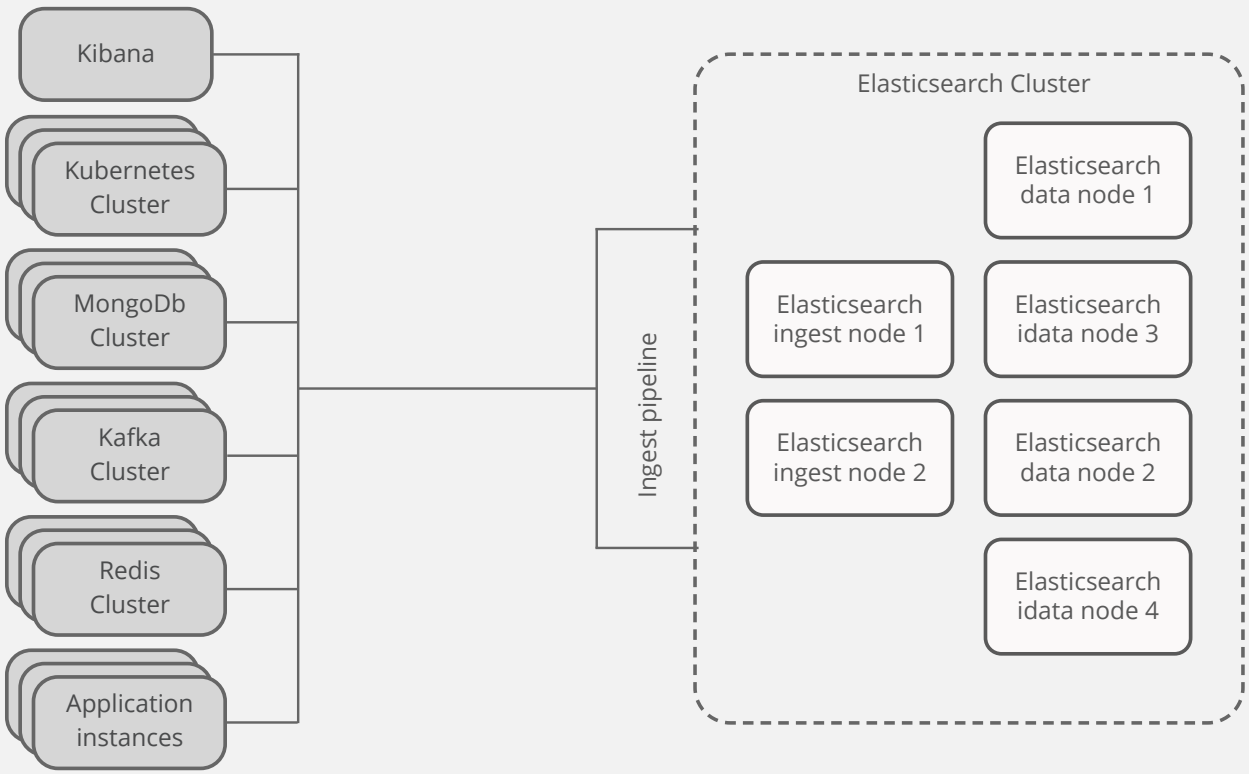
Logging and monitoring: We implemented Sidecar utility containers with individual applications that facilitate log collection and monitoring. Filebeat was used as a log shipper for a centralized Elasticsearch cluster. Prometheus Operator was used to collect service metrics which was federated to a centralized Prometheus server.

Autoscaling: We integrated two types of autoscaling; Horizontal Pod Autoscaler (HPA) and Cluster Autoscaler (node autoscaling). The horizontal pod autoscaler (HPA) automatically scales the number of Pods deployed based on observed CPU or Memory consumption. The Cluster autoscaler dynamically scales the number of nodes to match current cluster utilization and controls the cost of operating Kubernetes clusters on a cloud provider platform.

Deployment Framework: We built a custom framework using docker and Ansible. We used shell scripts to dockerize and deploy applications.



Prometheus HA



Filebeat as a log shipper

Impact

40% Reduction in costs

Improved flexibility

Improved monitoring

Once the solution was implemented, our customer's challenges plunged dramatically.

1. **Reduced AWS cost:** It only takes seven nodes to manage 30+ microservices. 50+ AWS instances hosting these microservices dropped down to only 10+ AWS instances.
2. **Decreased downtime:** We have achieved zero downtime using Rolling Update for kubernetes. The system goes down only during database updates, which happens rarely.
3. **Auto scaling:** Auto-scaling automatically increases or decreases pod or node counts depending on application load.
4. **Reduced deployment time:** Infrastructure provisioning, deployment, and configuration management of applications occurs with just one click. Additionally, multi-region deployments are seamless and hassle-free.
5. **Improved monitoring and logging:** Monitoring & logging is now enhanced with more detailed and meaningful dashboards on Grafana and Kibana.

Great Software Laboratory (GS Lab) has been the technology partner of choice to 100+ organizations across North America, Europe and Asia-Pacific for over 17 years. Leveraging our expertise in 130+ tools & technologies, we have created 300+ 'first-of-its-kind' solutions to real-world problems. Our 'Beyond code' philosophy ensures that we not only push boundaries of existing technologies but also try out newer problem solving approaches to keep our customers one step ahead of their competitors. Our global team of 1200+ employees is adept at creating 'real value' at each stage of the customer growth journey, right from proof-of-concepts to completely scaled up products. For more information about our solutions & offerings, please visit www.gslab.com