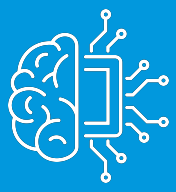

» CUSTOMER CASE STUDY

Deploying AWS EKS to enable Machine Learning (ML) Applications

Deploying AWS EKS from scratch following best practices with the entire environment completely defined and managed using Infrastructure-as-Code to enable ML Applications

NEBULAWORKS

Executive Summary

Nebulaworks was engaged by a leading global security services company with expertise in diverse industries. They specialize in providing integrated security solutions in North America, Europe, Asia-Pacific and the Middle East.

In an attempt to create an automated shift scheduling system for their security personnel that leverages direct messaging technology, engaging with Nebulaworks provided the ability to migrate to AWS Secrets Manager from their previous solution and provided Infrastructure as Code (IAC) to deploy and manage their EKS clusters. The solution provided a stable secrets management solution increasing environment resiliency, enabled the ability to perform rolling updates for their clusters and the ability to leverage GPU based EKS nodes to help the client Engineering team explore the use of Machine Learning (ML) applications to assist the Natural Language Processing (NLP) components of their automated shift scheduler. Nebulaworks provided the ability to use AWS Secrets Manager and Infrastructure as Code (IAC) to manage EKS clusters. The solution created increased resiliency, ability to perform updates to clusters and use GPU based EKS nodes to explore Machine Learning to assist parts of the automated shift scheduler.

About the Client

From positioning security guards to help protect people, to innovative security technologies using cloud platforms, the global security services company aims to deliver comprehensive solutions to help clients mitigate risks and create secure environments. With strong client relationships being a priority to assist clients, the security services involve video surveillance and alarm monitoring as well as risk assessment and consulting. Their consulting services provide guidance on regulatory compliance, crisis management, emergency response and fire safety services. With a global presence, the client has a vast network of security professionals and resources to provide consistent security services around the world.

The Challenge

The client's goal was to make shift scheduling for security personnel automated, efficient, and ultimately assure that shifts were filled in a timely manner to create a better customer experience. The call center that was previously handling the shift scheduling was expensive to operate, and incapable of providing the shift coverage that the company was comfortable seeking. The first iteration of the shift scheduler leveraged a Proof of Concept (PoC) AWS EKS, and the engineering team was considered an internal startup, tasked with building the system quickly and at a low

cost. In a staggered manner, the new platform owned 20% of the shift scheduling, while the call center still handled the balance of shift scheduling. This initial version of the system had issues with an unreliable secrets management, as well as an inconsistent way to provision and manage their PoC EKS clusters. The original secrets management solution resulted in failed application deployments due to it crashing, and it was considered a black box since the original implementers were no longer part of the company. Additionally, there was limited release management process due to the lack of diverse environments and reproducibility. With hard deadlines and a strong desire to decommission the call center and allow the new system to be 100% responsible for shift scheduling, this engineering problem required a partner that understood the pain points, accelerating the development and operationalizing a production platform in order to reduce cost for the business. Leveraging the call center was a risk, since it often resulted in missed shifts, and incurred a high cost to keep it operational.

Why AWS

AWS was the platform of choice due to the low barrier to entry, pay as you go model, and scalability. The team engineering the shift scheduler system operated as a startup, and was a small division crafted from the larger IT team. With the managed services in AWS, the team was able to begin iterating using the AWS EKS managed service to begin deploying their container based applications in rapid and cost-effective fashion. In addition, the AWS solution provided a reliable and proven method to scale the new service into production.

Why Nebulaworks?

The client required an embedded DevOps team that had a breadth of experience with various cloud native technologies, and operated like a modern software engineering team. Nebulaworks has expertise in building cloud native applications using Infrastructure as Code (IAC), extensive experience with container based workloads and orchestrators, Continuous Integration/Continuous Delivery/Continuous Deployment (CI/CD1/CD2), and their existing secrets management solution which was Hashicorp Vault. Aligning with their existing software/infrastructure stack and culture, Nebulaworks provided a team that was able to hit the ground running, providing consultative services to determine the best course of action to improve and enhance cloud native applications. Ultimately this allowed the client to leverage the AWS Cloud in a way to optimize their investment in building the text message based scheduling solution their business demanded.

The Solution

A critical part of engineering effort was to create a new EKS development environment from scratch following best practices, and have the entire environment be completely defined and deployed using Infrastructure-as-Code. This allowed for the new platform to be self-documenting, as well as being able to be fully reproducible. The Nebulaworks team chose Terraform as the Infrastructure-as-Code platform due to the team's extensive experience with it, as well as its ability to not only define the AWS infrastructure, but also define the applications that would be deployed to the new cluster.

The team decided to leverage the module capabilities of Terraform, knowing that even though the team was focused on the development environment, it would be composed of components that could be reused to create brand-new environments for staging and production. One module exists for the VPC, which defines the VPC, subnets, NAT gateways, and routing rules, as well as annotations on the subnets for EKS. Another module creates the EKS cluster itself, as well as required IAM permissions for various applications within the cluster. Finally, a node group module manages a set of EKS nodes for a given instance type.

With multiple node group modules per cluster, it became possible to accommodate applications with widely varying requirements and use cases. One of which was building a Natural Language Processing (NLP) feature as part of this system. The text-message based chat bot that was built required the ability to communicate with employees via SMS messaging. Machine learning training was necessary in order to obtain coherent and relevant responses. Working alongside a team of machine learning engineers, the Nebulaworks engineering team provided the platform capabilities to leverage GPU-Based Kubernetes nodes in the cluster. Leveraging node affinity rules, the machine learning model training was executed by placing the python model training program on the correct GPU nodes. Since GitLab's shared runners were not able to meet this requirement, the Nebulaworks team setup custom GitLab runners on GPU-powered instances for model building and training, as well as a unique node group specifically designed to run that application.

As part of the migration process to new EKS clusters, the team needed to consider how to migrate application secrets to the new environment. The previous secret management solution was manually installed and could not be accurately reproduced, creating the opportunity to explore a different secrets management solution. Since the Nebulaworks team already leveraged an AWS-native solution for Kubernetes, they decided to use AWS Secrets Manager. Secrets Manager proved to be easy to integrate into all the applications, as well as providing a central point of management for the secrets themselves, and the ability to restrict visibility of secrets through IAM permissions.

EKS also provides a straightforward path toward upgrading the Kubernetes version of a cluster. By utilizing the modular Terraform approach toward building a cluster and node group, the team

can easily build an isolated environment to test future versions of Kubernetes before deciding to roll out an upgrade. The highly-available control plane also ensures that cluster upgrades are vetted and pre-tested before rolling out, otherwise the cluster is restored without leaving it unrecoverable.

Results and Benefits

The solution provided a robust EKS platform and therefore more resilience, stability and consistency in their EKS application deployments. In total there were 3 points of failures removed after engagement completion.

1.) Transitioned to AWS Secrets Manager (SM). AWS SM scaled to their needs, provided a reliable service with an acceptable SLA, and provided a straightforward way to upload and retrieve secrets and sensitive data for their applications.

2.) Nebulaworks assisted in the use of self-managed Gitlab CI Runners. Self managed CI Runners resulted in more control of availability and performance for their CI/CD1/CD2 processes, and the ability to customize the nodes to execute tests against.

3.) Gitlab AutoDevops features were abandoned due to its inability to allow the EKS Clusters and integrations to be codified and managed via code and a change management process. Nebulaworks introduced rigor around building, testing, and deploying EKS clusters using Terraform, which resulted in high control of the state of their clusters.

From the time that Nebulaworks started, the company doubled the workload handled by the call center and was continuing to rise. Migrating to AWS Secrets Manager provided the level of reliability that the engineering team and key stakeholders desired for this critical initiative, removing up to 5 vault failures a week. Kubernetes clusters were codified using Terraform, providing more control on the characteristics of the cluster as well as a path to consistently create clusters quickly for new feature testing. Python based Machine Learning model development was possible by leveraging node affinity rules on the Kubernetes cluster. Along with the codification of the cluster, Nebulaworks provided a standard way to introduce new Kubernetes+Helm container based applications, accelerating the time to deploy a new service, and assisted in onboarding over 20 new container based applications, and contributed to over 80 production deployments, increasing their frequency of production application deployments by 30%.