mitrais

continuous commitment

MITRAIS WHITE PAPER
COMPARISON OF
KUBERNETES SERVICE OFFERING FROM
AWS, AZURE AND GOOGLE

VER.0.1

Table of Contents

Table of Figures

1. Overview

Many organizations today have opted to adopt container-based architecture in their application development lifecycle. [Forester Study] found that many organizations that adopted containers in their development experienced an increased efficiency within the development process and a more consistent deployment process.

Benefits for an organization adopting container-based architecture include:

- Reduces hardware cost

  Container-based architecture reduces the number of virtual machines required and leads to a reduction in the number of resources used by the operating system and the application itself.

- Reduces the licensing and maintenance costs

  No need for the additional license cost each guest OS or the hypervisor.

- Portability

  Containers can run in any environment consistently. Once deployed, a container can be ported into any environment that supports it and runs with similar behaviours without any modification. Applications can easily be ported from on-premise infrastructure into a cloud-ready environment or vice versa.

- Increased Productivity

  Containers can be easily integrated into a Continuous Delivery pipeline, allowing easy delivery of an updated application into multiple environments, and eventually to the production environment. If any issue arises, rolling back to the previous version can be done quickly by reverting to a previous container image build. Creating multiple environments is quick without having to deal with multiple complicated build and deployment scripts.

Previously the adoption of the container-based architecture was clouded by concerns over security and the immature of the technology, especially in production environments.

While containers run exactly the same on-premise as on cloud, it is not an easy task to set up and to configure environments capable of orchestrating and managing multiple containers at once.

With Kubernetes, it is easy to perform container orchestration such as automatic scaling, rolling updates, and automated deployment (without downtime). However, creating a Kubernetes cluster in an on-premise environment is not an easy task. This is how the managed services from a major Cloud provider can save time.

In this document, we evaluate the managed container service from 3 major cloud platform providers.

2. Evaluated Provider

	PROVIDER	SERVICE NAME	Started From
1	AWS	Elastic Kubernetes Service	2018
2	Azure	Azure Kubernetes Service	2017
3	Google	Google Kubernetes Engine	2014

Table 1 – Evaluated Platform

2.1. AWS Elastic Kubernetes Service

The AWS Elastic Kubernetes Service (EKS) is a Kubernetes managed service from AWS. The EKS Cluster consists of two main components – the Control Plane and Worker Nodes.

The Control Plane consists of 3 Kubernetes Master Nodes and runs on 3 different availability zones. Any incoming traffic is directed through the AWS Network Load Balancer. The Control Plane cannot be managed by the user - it is automatically managed by AWS.

Worker Nodes run on an AWS EC2 instance inside a VPC and can be managed by the user.

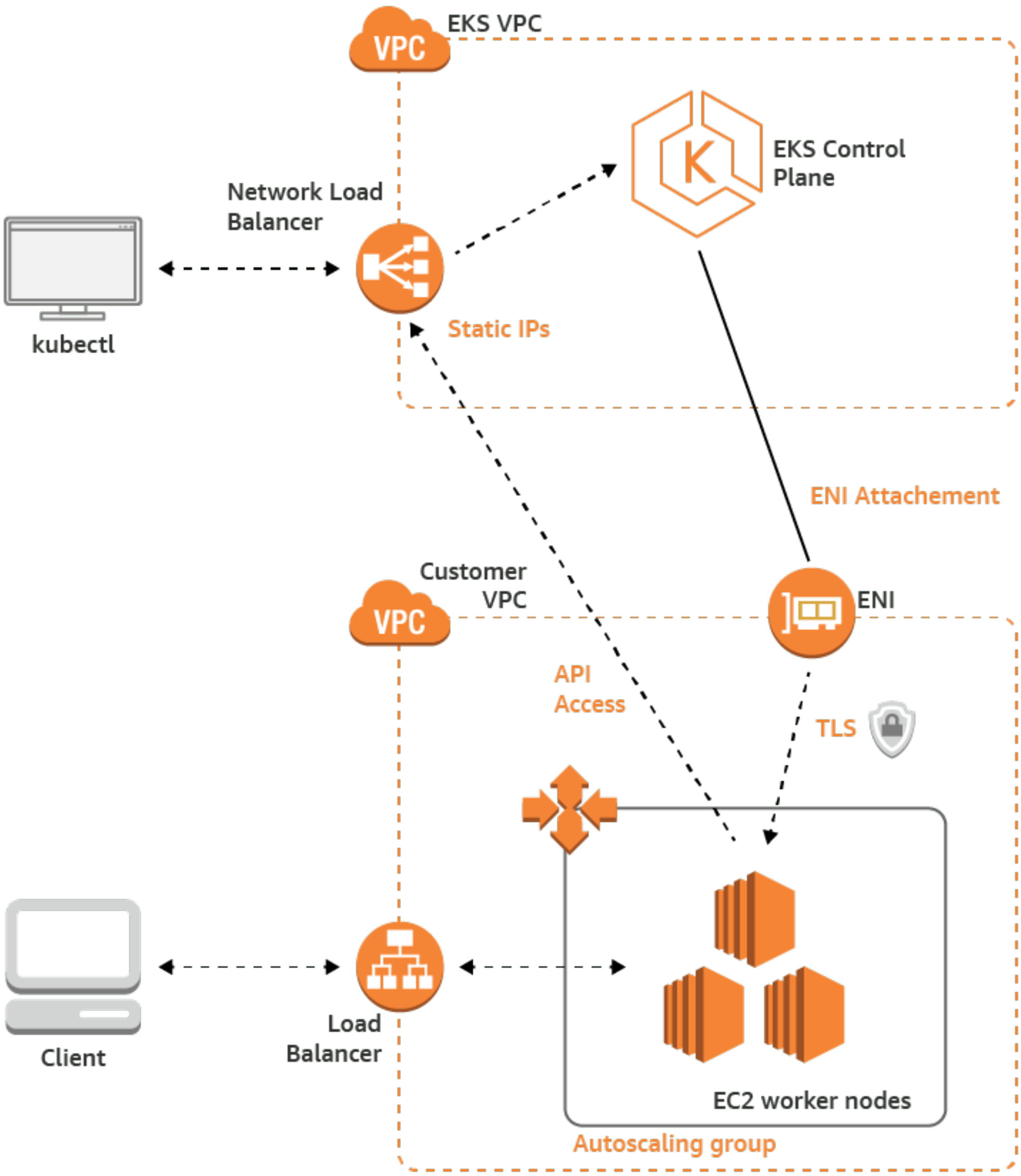

Figure 1 AWS EKS Architecture

2.2. Azure Kubernetes Service

Azure Kubernetes service (AKS) is the managed Kubernetes service from Azure. AKS is built on top of the open source Azure Kubernetes Service Engine (AKS-engine). AKS consists of a Cluster Master and Nodes. The Cluster Master is managed by the Azure platform. Nodes are where the application workload resides and are managed by the user.

In AKS, the Cluster Master is not charged to the user. Users are only charged for the VM infrastructure that is used by the Kubernetes Nodes. The worker Node in AKS supports Linux and Windows machines.

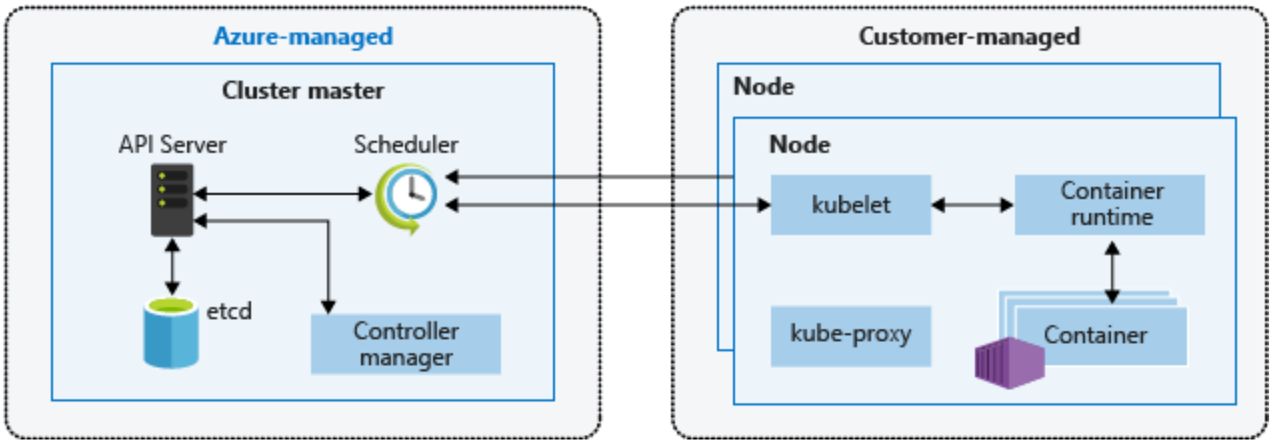

Figure 2 Azure Kubernetes Service Architecture

2.3. Google Kubernetes Engine

Google Kubernetes Engine is a managed Kubernetes environment provided by Google. The Kubernetes Engine consists of one Cluster Master and multiple worker nodes. The Cluster Master is free of charge, and users are only charged for the infrastructure used by the worker nodes.

3. Other Platforms

In this paper, we have only assessed the Kubernetes services from the leading three cloud platforms. Other platforms not yet assessed include:

- DigitalOcean Kubernetes
- IBM Cloud Kubernetes Service
- Alibaba Container Service for Kubernetes (ACK)

Mitrais may update this evaluation to include these platforms in the future.

4. Evaluation Criteria

CRITERIA	MITRAIS WEIGHTING
Cost	1
Utilities	1
Scalability	1
Security	1
Availability	1

Table 2 – Evaluation Criteria

Criteria used:

1. Cost

   The total cost to run a cluster and single nodes is assessed. The parameter is the total cost of running a single cluster in the smallest possible virtual machine instance and the smallest single-node instance.

2. Utilities

   The platform should offer various utilities either based on CLI or a web/desktop interface.

3. Scalability
   The capability of the platform to automatically scale up when maximum utilization is reached.

4. Security

   Support for additional security mechanisms other than default framework that they based on (docker or Kubernetes).

5. Availability

   Ability to handle conditions such as downtime and connectivity issues and mitigate them as necessary.

5. Platform Evaluation

PLATFORM	EVALUATION METHOD	1	2	3	4	5	6	7	TOTAL SCORE
AWS EKS	Product Overview	2	5	4	5	5			**21**
Azure AKS	Product Overview	4	5	3	5	3			**20**
Google Kubernetes Engine	Product Overview	5	5	4	4	5			**23**

Table 3 – Platform Evaluation

6. Platform Recommendation

6.1. Cost

AWS EKS costs $0.20 per hour for each EKS cluster. Additional EC2 and EBS volumes are applied for each worker node. The monthly cost estimation for AWS EKS with a single cluster and a single t2.micro (smallest instance) worker node is as follows:

Service Name	**Total Monthly Cost (30 Days)**
Kubernetes Cluster	$144
Worker Node T2.Micro 8 GB EBS Volume	$11.65
TOTAL	$155.65

Azure AKS does not charge for the Kubernetes master node, but only for the volume used by the virtual machine for each worker node. The monthly cost estimation for Azure AKS with single cluster and an A1 (smallest instance) worker node is as follows:

Service Name	**Total Monthly Cost (30 Days)**
Kubernetes Cluster	Free
Worker Node A1 32 GB Volume	$43.80
TOTAL	$43.80

Google Kubernetes Engine does not charge for the Kubernetes master node. It only charges for the virtual machine and volume for each worker node. Monthly cost estimation for Google Kubernetes engine with a single cluster and n1-standard-1 (smallest instance) worker is as follows:

Service Name	**Total Monthly Cost (30 Days)**
Kubernetes Cluster	Free
Worker Node n1-standard-1 375 GB Volume	$37.70
TOTAL	$37.70

6.2. Utilities

AWS EKS tools to help manage and configure EKS resources:

- AWS CLI: tools to configure various AWS resources, including EKS clusters.
- CloudFormation: Template-based json file to automate worker node creation. AWS also provides support with a template that is compatible with AWS EKS.
- eksctl: To manage cluster creation. This utility also ships its own version of a kubectl utility to communicate with the Kubernetes API Server.
- Kubectl: To communicate with the Kubernetes API server.

Azure AKS tools to help manage and configure AKS resources:

- Azure CLI: tools to configure various Azure resources, including AKS clusters.
- Resource Manager Template: JSON files that automate resource creation in the Azure platform. The templates support EKS Cluster creation.
- Kubectl: Command line tools to communicate with Kubernetes API server.

Google Kubernetes Engine tools to help configure the cluster and its resources:

- gcloud: command line utility to configure various GCP resources, including Kubernetes.
- Google cloud shell: an online shell utility that comes preinstalled with gcloud and kubctl utility.
- Kubectl: Kubernetes command line utility to communicate with Kubernetes API server.

6.3. Scalability

AWS EKS supports automatic scaling of the application. Native Kubernetes API Horizontal Pod Autoscaler and Kubernetes Metric Server are supported by EKS starting from version 2.

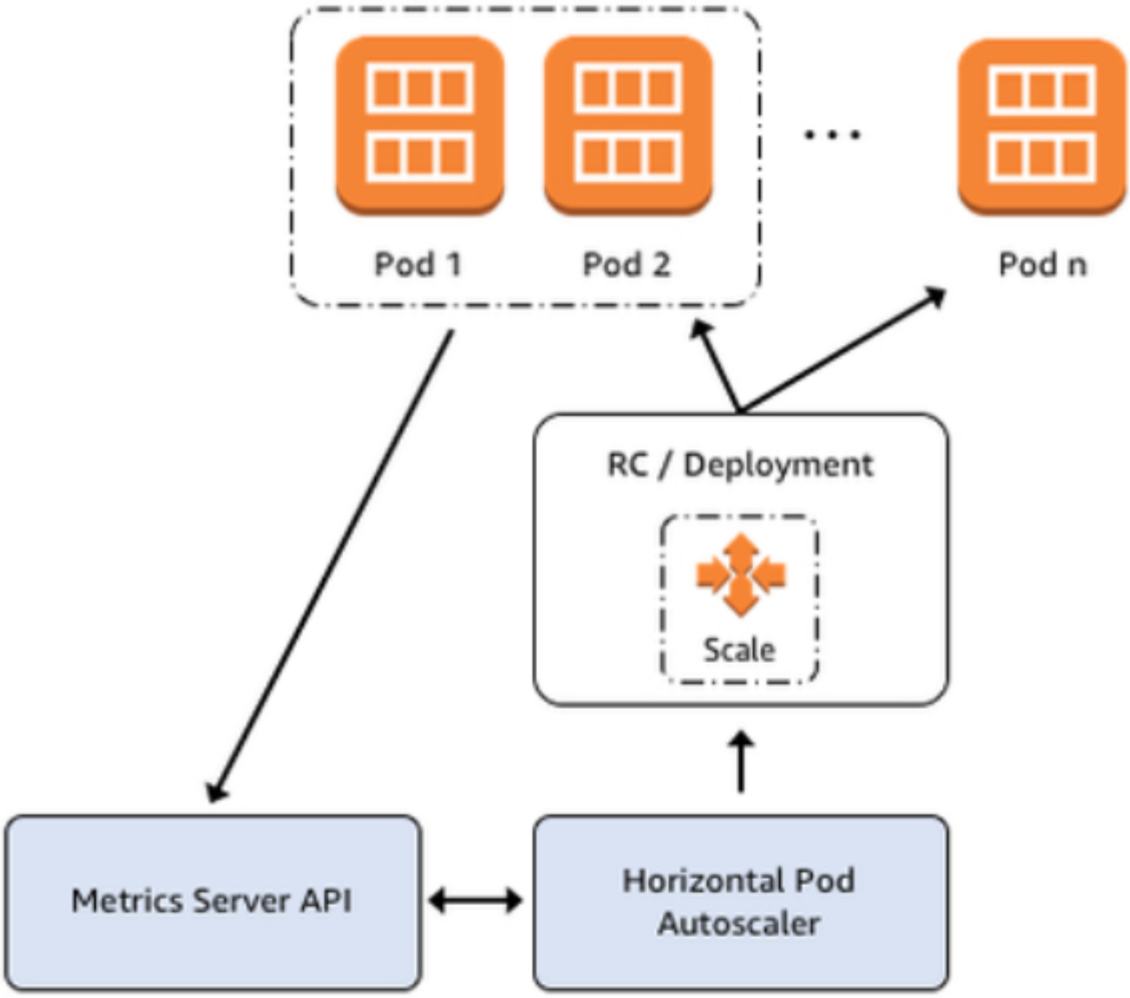

Figure 3 AWS EKS Automatic Scaling

Azure AKS supports automatic scaling via Cluster Autoscaler or the Horizontal Pod Autoscaler API. Cluster Autoscaler works by watching whether there are any pods that cannot be scheduled because all resources have been consumed. If so, the number of nodes is automatically increased. The Horizontal Pod Autoscaler works by using the

Kubernetes Metrics Server. If services need more resources, it will automatically increase the number of pods to meet the demand. Both Cluster Autoscaler and Horizontal Pod Autoscaler can reduce the number of nodes/pods automatically if demand decreases.

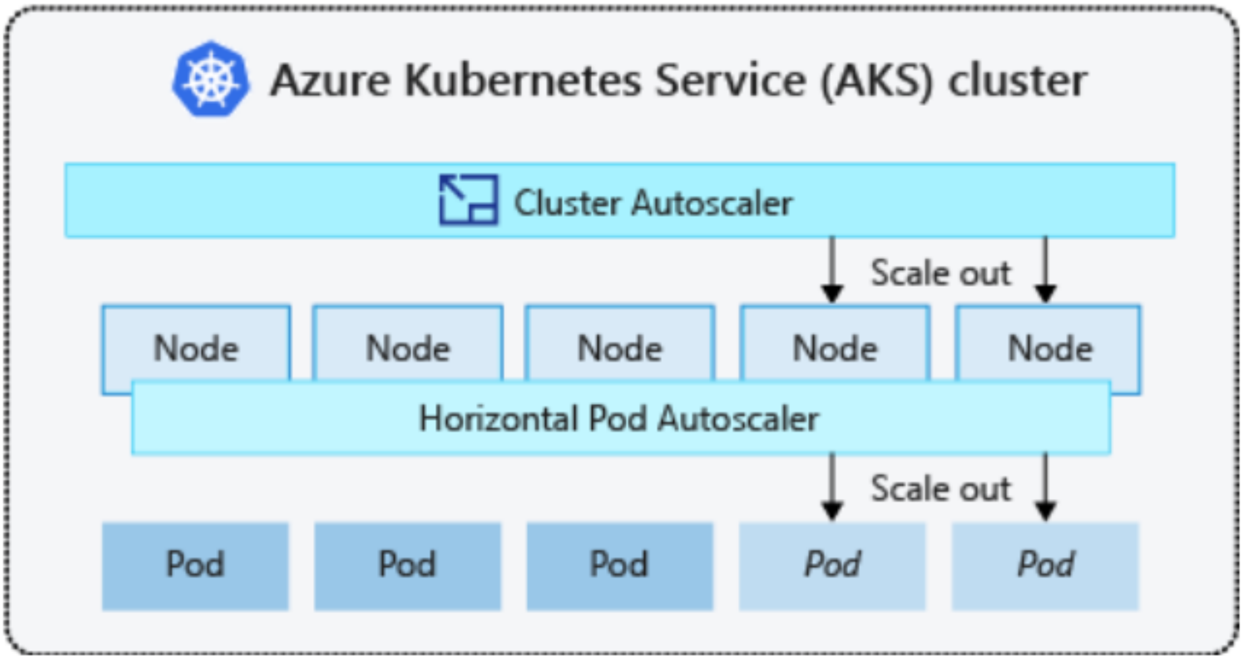

Figure 4 Azure AKS Automatic Scaling

Google Kubernetes Engine supports autoscaling via the Cluster Autoscaler, which operates on a per-node basis. It periodically checks whether any pod has been scheduled to any available nodes, and if there are pods that not yet scheduled due to resource limitation, it will automatically increase the number of nodes. It also checks if any of the nodes is underutilized and will re-allocate all the pods to other nodes so that the underutilized node can be deleted.

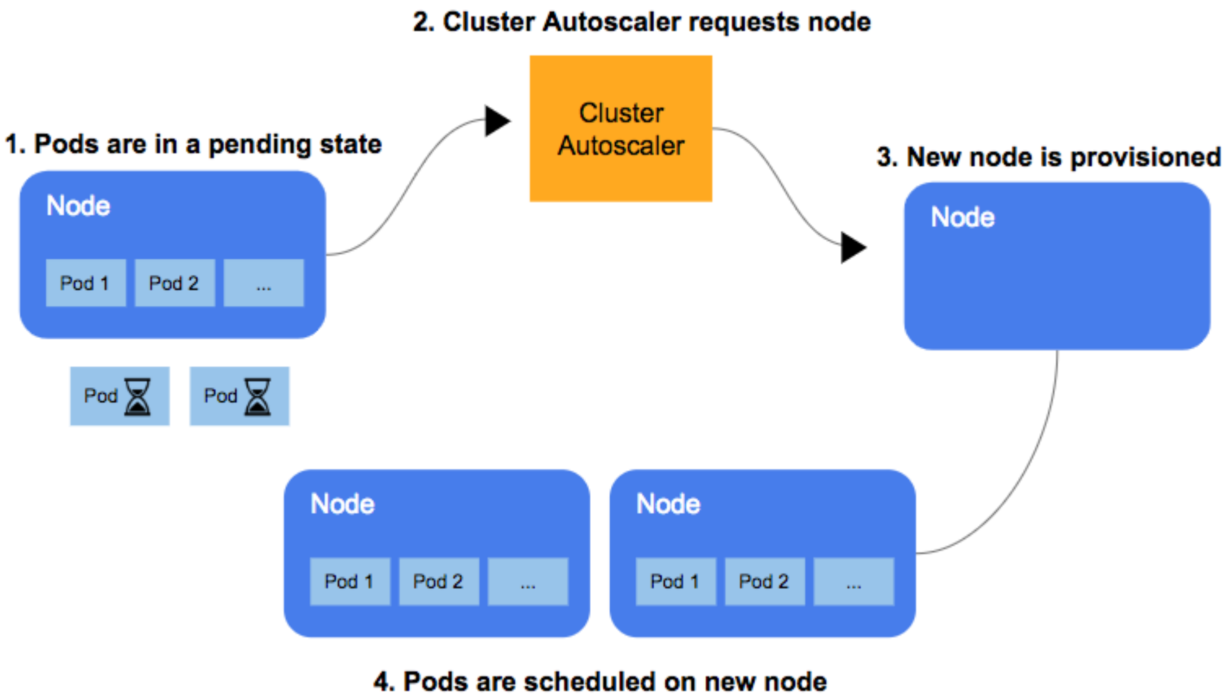

Figure 5 Google Kubernetes Engine Autoscaler

6.4. Security

AWS EKS supports IAM to manage the Kubernetes cluster. However, it still relies on native Kubernetes Role Based Access Control (RBAC) for the authorization. To authenticate to the Kubernetes, the user must use AWS CLI command or install the AWS IAM Authenticator for Kubernetes. For worker node security, it is handled similarly to a normal EC2 instance. The user can access it via SSH or modify the virtual machines.

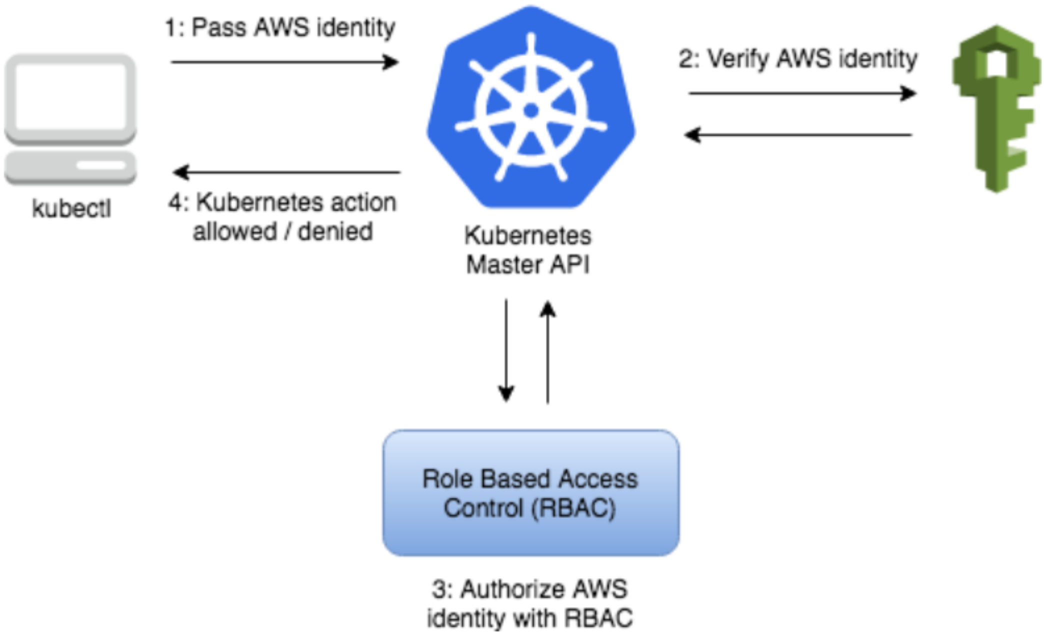

Figure 6 AWS EKS Security

Azure AKS supports Active Directory integration to access AKS or directly use native Kubernetes Role Based Access Control (RBAC). Active Directory integration can only be applied to a new Kubernetes cluster, as there is no mechanism to add the Active Directory integration to an existing cluster. Worker nodes are treated as normal virtual machines that the user can connect to using SSH or Remote desktop (for Windows-based images).

Google Kubernetes Engine supports two types of authentication: User Account and Service Account. A User Account is recognized by Kubernetes but managed using Google's cloud platform. Service Account is the ones created by Kubernetes and are only applicable to Kubernetes entities.

6.5. Availability

From v1.2, Kubernetes added support for running a cluster in multiple availability zones. Currently only AWS EKS and Google Kubernetes Engine support running a cluster in different availability zones. Azure EKS support for multi-availability zones is still in the development phase and will be released later.

References

1. http://i.dell.com/sites/doccontent/business/solutions/whitepapers/en/Documents/Containers_Real_Adoption_2017_Dell_EMC_Forrester_Paper.pdf
2. https://azure.microsoft.com/en-us/updates/availability-zones-az-support-for-aks/
3. https://docs.aws.amazon.com/eks/latest/userguide/what-is-eks.html
4. https://docs.aws.amazon.com/eks/latest/userguide/managing-auth.html
5. https://cloud.google.com/kubernetes-engine/
6. https://docs.microsoft.com/en-us/azure/aks/

Copyright

Disclaimer

Any and all information in this document has been compiled and provided for information purposes only. The information provided herein may include information compiled from a variety of third parties. Mitrais will not be liable for any loss, damage, cost or expense incurred in relation to or arising by reason of any person relying on the information in this document or any link to any website provided herein, whether or not caused by negligence on Mitrais' part. While Mitrais endeavours to provide the information up to date and correct, Mitrais make no representations or warranties of any kind, express or implied, about the accuracy, reliability, completeness or currency of the information or its usefulness in achieving any purpose. Readers of this document are responsible for assessing its relevance and verifying the accuracy of the content. In any event, the information provided herein should not be construed as providing advice whether legal or otherwise.