EKS | High Availability | DR | AZ

|  |  |
| --- | --- |
| Author | Pradeep/Byron |
| Issue Date | 14th May 2024 |
| Version | 1 |
| Distribution list | Infomagnus-AWS Team |

Contents

[Introduction 3](#_Toc166597598)

[Executive Summary 3](#_Toc166597599)

[EKS Cluster Architecture Overview 3](#_Toc166597600)

[EKS | High Availability (HA)Design 3](#_Toc166597601)

[EKS | Disaster Recovery Strategy (DR) 5](#_Toc166597602)

[EKS | Availability Zones configuration (AZ) 6](#_Toc166597603)

[EKS | Security considerations 7](#_Toc166597604)

[EKS | Incorporating additional services (RDS, MSK, S3) 7](#_Toc166597605)

[EKS | Monitoring and Logging 8](#_Toc166597606)

[EKS | Cost considerations 8](#_Toc166597607)

[EKS | Benefits of HA, DR and AZ 9](#_Toc166597608)

[EKS | Benefits for Dental services client 9](#_Toc166597609)

[EKS | Conclusion 10](#_Toc166597610)

[Appendices 10](#_Toc166597611)

# Introduction

This document outlines the design and implementation of a highly available, disaster-resilient Amazon Elastic Kubernetes Service (EKS) cluster using multiple Availability Zones (AZ). It aims to provide insights into the architecture, implementation steps, best practices, and considerations necessary to ensure the reliability and continuity of services.

# Executive Summary

Amazon EKS provides a managed Kubernetes service, making it easier to run Kubernetes on AWS without needing to install and operate your own Kubernetes control plane. This document details the high availability, disaster recovery, and multi-AZ configuration to ensure minimal downtime and data loss in case of failures. The proposed architecture will leverage AWS's capabilities to deliver a robust, cost-effective solution.

.

# EKS Cluster Architecture Overview

Amazon EKS is a fully managed service that simplifies running Kubernetes on AWS. It handles the Kubernetes control plane, including the API server and the etcd database, allowing users to focus on deploying and managing applications.

Key Components

EKS Control Plane: Managed by AWS, including the API server, etcd, and other control plane components.

Worker Nodes: EC2 instances running Kubernetes pods, managed by the user.

Networking: VPC, subnets, security groups, and network policies.

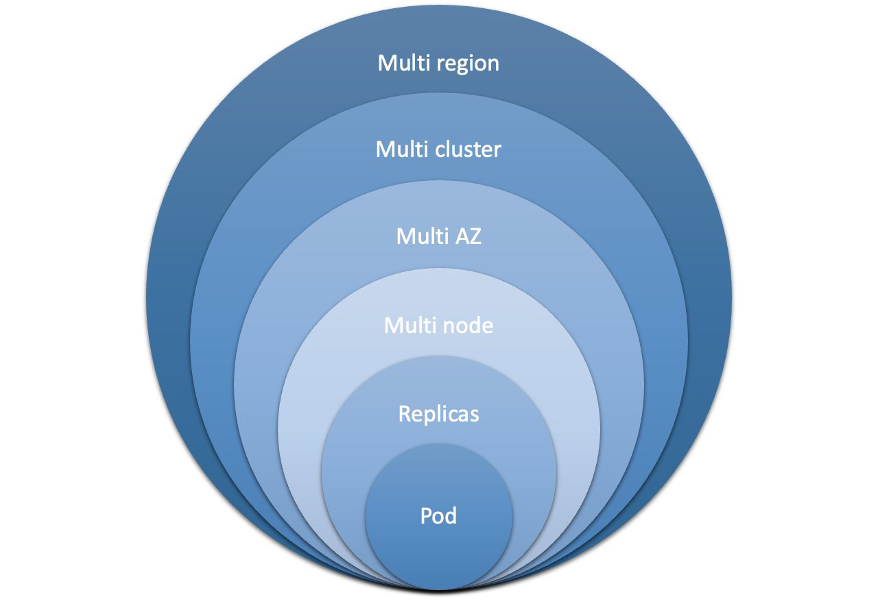
Storage: Persistent storage using EBS, EFS, and S3.

# EKS | High Availability (HA)Design

The primary goal of HA design is to ensure that the EKS cluster can withstand failures and continue to operate without significant downtime, meeting the desired uptime and service level agreements (SLAs).

**Architecture**

* **Multi-AZ Deployment**: Distributing resources across multiple AZs to avoid single points of failure.
* **Redundancy**: Ensuring that there are redundant instances of critical components.
* **Load Balancing**: Using Elastic Load Balancers (ELB) to distribute traffic evenly across nodes.



Kubernetes brings powerful orchestration capabilities to enhance workload resilience:

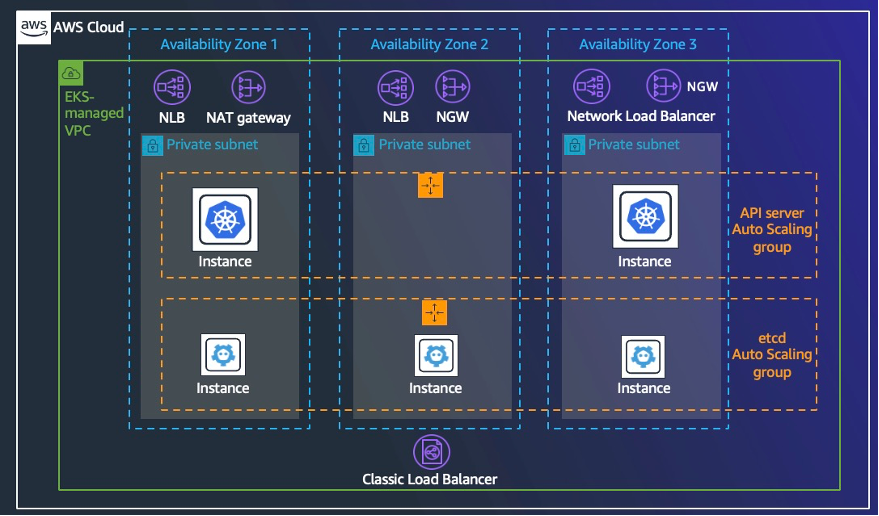
1. Replication and scaling: Kubernetes allows you to define the desired number of replicas for your workloads, ensuring that multiple instances of the application are running concurrently. If a replica fails or becomes unresponsive, Kubernetes automatically replaces it with a healthy one.
2. Health checks and self-healing: Kubernetes continuously monitors the health of individual replicas by performing health checks. If a replica fails these checks, then Kubernetes automatically terminates it and starts a new one in its place. This self-healing feature ensures that the workload remains available and resilient even in the presence of failures.
3. Fault isolation: Kubernetes allows you to define and enforce resource limits and constraints for workloads. By setting resource quotas and limits, you can prevent a single workload from consuming excessive resources, which reduces the risk of resource exhaustion and isolating failures to specific workloads rather than affecting the entire cluster.
4. Rolling updates and rollbacks: Kubernetes facilitates seamless rolling updates, which allows you to update your application without incurring downtime. By rolling out updates in a controlled manner, Kubernetes ensures that a minimum number of replicas are available and operational at all times. Additionally, Kubernetes supports rollbacks that revert to a previous version of an application if issues arise during an update.
5. Multi-domain deployments: Kubernetes supports deploying workloads across multiple failure domains. By deploying replicas across multiple nodes, AZs, and clusters, you increase the resilience of your workload against infrastructure failure at various levels.

**Implementation Steps**

1. **Create a VPC with Multiple Subnets**: Ensure subnets are distributed across at least three different AZs.
2. **Configure Security Groups**: Define security groups to control traffic between nodes and external services.
3. **Deploy EKS Cluster**: Use AWS Management Console, CLI, or eksctl to create an EKS cluster across multiple AZs.

**eksctl create cluster --name my-cluster --region us-west-2 --zones us-west-2a,us-west-2b,us-west-2c**

1. **Set Up Worker Nodes**: Create Auto Scaling Groups (ASGs) for worker nodes across multiple AZs.
2. **Configure Load Balancing**: Deploy an ELB/NLB to distribute traffic to worker nodes.
3. **Implement Health Checks**: Regularly check the health of the control plane and worker nodes.



**Best Practices**

* **Managed Node Groups**: Utilize EKS managed node groups for easier lifecycle management.
* **Health Checks**: Implement regular health checks and failover testing.
* **Auto Scaling**: Use cluster autoscaler to adjust the number of nodes based on demand.

# EKS | Disaster Recovery Strategy (DR)

**Objectives**

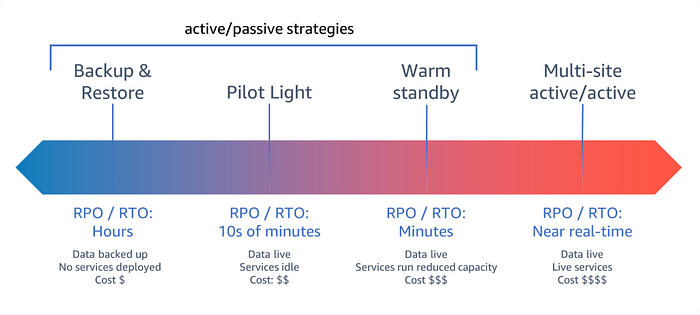
The DR strategy aims to minimize downtime and data loss during catastrophic events, ensuring a quick and efficient recovery.

**Backup and Restore Procedures**

* **Data Backups**: Regularly back up persistent data using EBS snapshots and EFS backups.
* **Control Plane Backups**: While AWS manages the control plane, consider additional backups for critical configuration data.

**DR Architecture**

* **Cross-Region Replication**: Replicate critical data and services to a different AWS region.
* **Backup Storage**: Store backups in durable storage solutions like Amazon S3.
* **The EKS Disaster Recovery project ensures the continuity of Amazon EKS Kubernetes clusters by implementing a multi-region deployment strategy, backup and restore mechanisms, and automated failover procedures.**



**DR Implementation Steps**

1. **Configure Automated Backups**: Set up regular snapshots for EBS volumes and backups for EFS.
2. **Set Up Cross-Region Replication**: Use AWS services like S3 Cross-Region Replication and AWS Backup.
3. **Implement DR Runbooks**: Document the steps to restore services and data in a different region.

**Testing and Validation**

* **Simulate Failures**: Regularly test failure scenarios to ensure the DR plan is effective.
* **Restore from Backups**: Practice restoring from backups to verify data integrity and recovery time.
* **Cross-Region Failover**: Test the failover process to a different region to ensure seamless recovery.

# EKS | Availability Zones configuration (AZ)

**Objectives**

Leverage multiple AZs to enhance fault tolerance and resilience of the EKS cluster.

**AZ Design Considerations**

* **Resource Distribution**: Evenly distribute resources across AZs to prevent overloading a single AZ.
* **Network Latency**: Consider latency and throughput between AZs when designing the network.

**Network Configuration**

* **VPC and Subnets**: Design a VPC with subnets in multiple AZs.
* **Security Groups and NACLs**: Define security groups and network ACLs to control traffic within the VPC.

**Implementation Steps**

1. **VPC Creation**: Create a VPC with subnets in at least three AZs.
2. **EKS Cluster Setup**: Deploy the EKS cluster across these AZs.
3. **Worker Node Configuration**: Configure worker nodes in ASGs spanning multiple AZs.
4. **Network Configuration**: Set up appropriate security groups, NACLs, and routing tables.

# EKS | Security considerations

**Network Security**

* **Network Policies**: Implement Kubernetes network policies to control traffic between pods.
* **PrivateLink and VPC Endpoints**: Use AWS PrivateLink and VPC endpoints to securely connect services.

**Identity and Access Management**

* **IAM Roles and Policies**: Define IAM roles and policies for the EKS cluster and worker nodes.
* **RBAC**: Use Kubernetes Role-Based Access Control (RBAC) to manage permissions within the cluster.

**Data Encryption**

* **Encryption at Rest**: Use AWS KMS to encrypt data stored in EBS, EFS, and S3.
* **Encryption in Transit**: Use TLS to encrypt data in transit between services.

# EKS | Incorporating additional services (RDS, MSK, S3)

**Amazon RDS for MSSQL**

**Benefits:**

* **Multi-AZ Deployment: Automatic failover to a standby instance in another AZ in case of an outage.**
* **Automated Backups: Daily backups and point-in-time recovery.**
* **High Availability and Durability: Built-in replication and automated failover.**

**Integration Steps:**

1. **Create RDS Instance: Set up an RDS instance with Multi-AZ configuration.**
2. **Configure Security: Set up security groups and IAM roles.**
3. **Database Connections: Configure applications to connect to the RDS instance.**
4. **Automate Backups and Maintenance: Enable automated backups and maintenance tasks.**

**The following are some limitations when working with Multi-AZ deployments on RDS for SQL Server DB instances:**

**1.Cross-Region Multi-AZ isn't supported.**

**2.Stopping an RDS for SQL Server DB instance in a multi-AZ deployment isn't supported.**

**3.You can't configure the secondary DB instance to accept database read activity.**

**4.Multi-AZ with Always On Availability Groups (AGs) supports in-memory optimization.**

**5.Multi-AZ with Always On Availability Groups (AGs) doesn't support Kerberos authentication for the availability group listener. This is because the listener has no Service Principal Name (SPN).**

**6.You can't rename a database on a SQL Server DB instance that is in a SQL Server Multi-AZ deployment. If you need to rename a database on such an instance, first turn off Multi-AZ for the DB instance, then rename the database. Finally, turn Multi-AZ back on for the DB instance.**

**7.You can only restore Multi-AZ DB instances that are backed up using the full recovery model.**

**Multi-AZ deployments have a limit of 10,000 SQL Server Agent jobs.**

**If you need a higher limit, request an increase by contacting AWS Support.**

**Managed Streaming for Kafka (MSK)**

**Benefits:**

* **Scalable Messaging Platform**: Facilitates real-time data streaming and analytics.
* **High Availability**: MSK automatically replicates data across multiple AZs.

**Integration Steps:**

1. **MSK Cluster Creation**: Create an MSK cluster across multiple AZs.
2. **Configure Producers and Consumers**: Set up applications to produce and consume messages from Kafka topics.
3. **Monitoring and Maintenance**: Use AWS tools to monitor Kafka performance and health.

**Amazon S3**

**Benefits:**

* **Durable Storage**: Provides scalable storage for backups, logs, and static assets.
* **Cross-Region Replication**: Ensures data durability and availability by replicating data across regions.

**Integration Steps:**

1. **Bucket Creation**: Create S3 buckets for different use cases (e.g., backups, logs).
2. **Configure Permissions**: Set up appropriate IAM policies and bucket policies.
3. **Enable Cross-Region Replication**: Configure replication rules for critical data.

# EKS | Monitoring and Logging

**Monitoring Tools and Metrics**

* **CloudWatch**: Use Amazon CloudWatch for monitoring EKS clusters.
* **Prometheus and Grafana**: Set up Prometheus for metrics collection and Grafana for visualization.

**Logging and Alerting**

* **CloudTrail**: Enable AWS CloudTrail for auditing API calls.
* **ELK Stack**: Use Elasticsearch, Logstash, and Kibana for centralized logging and analysis.

# EKS | Cost considerations

**Cost Estimation**

* **Multi-AZ Deployment**: Estimate costs for running resources in multiple AZs.
* **Backup and DR**: Calculate costs for storage and data transfer related to backups and DR.

**Cost Optimization Strategies**

* **Right-Sizing**: Continuously monitor and adjust the size of instances and resources.
* **Spot Instances**: Use EC2 Spot Instances for cost savings on non-critical workloads.

# EKS | Benefits of HA, DR and AZ

**High Availability (HA)**

**Benefits:**

* **Increased Uptime**: Ensures that applications remain available even if one or more components fail.
* **Improved Performance**: Load balancing distributes traffic across multiple nodes, improving performance and response times.
* **Enhanced User Experience**: Consistent availability leads to a better user experience and higher satisfaction.

**Examples:**

* **E-commerce Platform**: An online store that continues to process transactions and handle customer queries even during a node failure, preventing loss of sales.
* **Healthcare System**: Ensuring that patient data and critical health applications remain accessible, which is vital for providing timely medical care.

**Disaster Recovery (DR)**

**Benefits:**

* **Data Protection**: Regular backups and replication protect against data loss.
* **Quick Recovery**: Enables rapid recovery of services and data after a disaster, minimizing downtime.
* **Compliance**: Meets regulatory requirements for data protection and disaster recovery.

**Examples:**

* **Financial Services**: Banks and financial institutions that can quickly restore services after a cyberattack or natural disaster, ensuring continuous access to financial data.
* **Educational Institutions**: Universities that can recover student records and online learning platforms after a major outage.

**Availability Zones (AZ)**

**Benefits:**

* **Fault Isolation**: Isolates faults to a single AZ, preventing widespread outages.
* **Resilience**: Distributes resources across multiple AZs, increasing fault tolerance.
* **Cost Efficiency**: Optimizes costs by using resources more efficiently across different AZs.

**Examples:**

* **SaaS Applications**: Software-as-a-Service providers can ensure their applications remain available to users globally, even if one AZ goes down.

# EKS | Benefits for Dental services client

**High Availability (HA)**

**Benefits:**

* **Increased Uptime and Reliability**: Ensures that critical dental applications such as appointment scheduling, patient records management, and telehealth services remain available even during infrastructure failures. This minimizes disruptions and maintains patient trust.
* **Improved Performance**: Load balancing across multiple nodes improves application responsiveness, providing a seamless experience for both patients and staff.
* **Enhanced Patient Experience**: Consistent availability of online services such as appointment bookings and patient portals enhances patient satisfaction and loyalty.

**Examples:**

* **Appointment Scheduling**: Even if one server node fails, patients can still book, reschedule, and cancel appointments without any interruption.
* **Patient Records Access**: Dentists and staff can access patient records and treatment history without delays, ensuring continuous care and efficient operations.

**Disaster Recovery (DR)**

**Benefits:**

* **Data Protection**: Regular backups of patient records, billing information, and treatment plans ensure data integrity and protection against data loss.
* **Quick Recovery**: Enables rapid restoration of services and data after a disaster, minimizing downtime and ensuring business continuity.
* **Regulatory Compliance**: Meets stringent healthcare regulations and data protection laws, ensuring that patient data is securely stored and recoverable.

**Examples:**

* **Data Backups**: Regular snapshots and backups of patient records to Amazon S3 ensure that in the event of data corruption or accidental deletion, data can be quickly restored.
* **Cross-Region Replication**: Critical patient data and application configurations are replicated to a different AWS region, ensuring availability even in case of a regional disaster.

**Availability Zones (AZ)**

**Benefits:**

* **Fault Isolation**: By distributing resources across multiple AZs, a failure in one AZ does not impact the overall system, ensuring continuous operation.
* **Resilience and Fault Tolerance**: Resources distributed across multiple AZs can handle high traffic loads and continue to operate seamlessly even if one AZ goes down.
* **Cost Efficiency**: Optimizes costs by efficiently utilizing resources across different AZs, reducing the need for over-provisioning.

**Examples:**

* **Telehealth Services**: Dental telehealth services remain operational even if one AZ experiences an outage, ensuring that virtual consultations can continue without interruption.
* **Patient Portal**: The patient portal remains accessible from different geographic locations, providing reliable access to patients for viewing their treatment history, scheduling appointments, and communicating with their dentist.

# EKS | Conclusion

The proposed HA, DR, and AZ configuration for EKS clusters ensures high availability, resilience, and minimal downtime. By following the best practices and implementation steps outlined in this document, clients can achieve a robust and cost-effective Kubernetes environment on AWS.

# Appendices

The proposed HA, DR, and AZ configuration for EKS clusters ensures high availability, resilience, and minimal downtime. By following the best practices and implementation steps outlined in this document, clients can achieve a robust and cost-effective Kubernetes environment on AWS.

**Glossary**

* **EKS**: Amazon Elastic Kubernetes Service
* **HA**: High Availability
* **DR**: Disaster Recovery
* **AZ**: Availability Zone
* **VPC**: Virtual Private Cloud
* **ASG**: Auto Scaling Group
* **RBAC**: Role-Based Access Control
* **IAM**: Identity and Access Management
* **KMS**: Key Management Service
* **ELB**: Elastic Load Balancer
* **NLB**: Network Load Balancer

**References**

* [Amazon EKS Documentation](https://docs.aws.amazon.com/eks/)
* [AWS Well-Architected Framework](https://aws.amazon.com/architecture/well-architected/)
* <https://docs.aws.amazon.com/eks/latest/userguide/disaster-recovery-resiliency.html>
* [Operating resilient workloads on Amazon EKS | Containers](https://aws.amazon.com/blogs/containers/operating-resilient-workloads-on-amazon-eks/)

**Acronyms**

* **EBS**: Elastic Block Store
* **EFS**: Elastic File System
* **S3**: Simple Storage Service
* **TLS**: Transport Layer Security
* **API**: Application Programming Interface