## **Solution Approach Document for Setting up Argo CD for canary deployment on EKS**

### **Description:**

A canary deployment is a progressive rollout of an application that splits traffic between an already deployed version and a new version, rolling it out to a subset of users before rolling out fully.

### **Goals:**

* Efficient Canary Deployments:

Develop a streamlined and controlled canary deployment process, enabling incremental releases of application updates with precision and reliability.

* High Availability and Scalability:

Ensure the Argo CD installation on EKS exhibits high availability, scalability, and resilience, capable of seamlessly accommodating a growing number of deployments.

* Monitoring and Observability:

Establish comprehensive monitoring and observability measures to track canary deployment health, performance, and success, facilitating swift issue identification and mitigation.

* Security and Access Control:

Implement robust security and access controls for Argo CD, restricting management and deployment privileges to authorized personnel exclusively.

* Rollback Capability:

Institute an effective rollback strategy for canary deployments, enabling a prompt and secure reversion to the previous application version in the event of deployment issues.

### **How Canary Deployments Work:**

Here is a general process for canary deployments:

* In the beginning, the current version receives 100% of user traffic
* A new deployment, the “canary” is performed with brand new pods and only a small amount of traffic, e.g. 5% while maintaining 95% of users on the older version.
* Different types of tests (e.g. smoke tests) can be performed on the new version with no impact on the bulk of the users.
* A decision regarding the current canary/subset of traffic takes place in an automated manner.
* If the new version works as expected, a larger portion of live traffic is sent to the new version and the process repeats again for different percentages of canary traffic (e.g. 5%, 25%, 50%, 75%, 100%).
* If the new version has issues, the service is switched back to the original version. This has minimal impact on most users. The canary version is destroyed and everything is back to the original state.
* In the end, 100% of traffic goes to the new version and the old version can be discarded.

### **Objectives:**

* Deploy Argo CD on EKS: The objective is to successfully deploy Argo CD on EKS cluster, making the continuous delivery tool available for managing application deployments.
* Canary Deployment Configuration: Define and configure the canary deployment strategy, specifying how new releases will be incrementally rolled out and monitored.
* Automated Synchronization: Enable Argo CD to continuously synchronize the state of the Kubernetes cluster
* Monitoring and Alerts: Set up monitoring tools to track key metrics and implement alerting mechanisms to promptly address issues or deviations from expected behavior during canary deployments.
* Testing and Validation: Implement automated testing and validation procedures to ensure that canary releases meet predefined criteria and are safe for promotion.
* Rollback Procedure: Define and implement a rollback procedure that allows for a quick and reliable rollback in case the canary release experiences problems or fails to meet criteria.
* Security and Access Controls: Establish strong security measures and access controls to protect the Argo CD installation and its configurations.
* Continuous Improvement: Continuously evaluate and refine the canary deployment process based on feedback, performance data, and lessons learned.

### **Deployment of Argo CD:**

* Argo CD Installed: Argo CD is successfully installed and running on the Amazon EKS cluster.
* Access to Web UI: The Argo CD web user interface is accessible and secured using proper authentication, and the admin password has been changed from the initial password.
* Canary Application Configuration: The canary application is defined within Argo CD

### **Benefits**

Here are three key benefits of canary deployments:

* Capacity testing – when deploying a new microservice to replace a legacy system, it is useful to be able to test in a production environment how much capacity you’ll need. By launching a canary version and testing it on a small fraction of your users, you can predict how much capacity you’ll need to scale the system to full size.
* Early feedback – many issues that affect end-users only occur in a production environment. Canary deployments can expose features to users in a realistic environment, to observe errors or bugs and obtain user feedback. This allows quick feedback from users, allowing developers to add new features and deliver what the end-user needs. This helps improve the software and the user experience.
* Easy rollback – in a canary deployment, if any severe issues are detected, rollback is instantaneous. It is just a matter of switching traffic back to the primary version or adjusting a feature flag.

### **Solution Approach:-**

Canary strategy

A canary deployment is a method that exposes a new feature to an early sub-segment of users. The goal is to test new functionality on a subset of customers before releasing it to the entire user base. You can choose randomly or a specific group of users and roll back if anything break

* Install ArgoCD on an EKS cluster by following the installation documentation.
* Access the ArgoCD Web UI to manage applications.
* Configure ArgoCD by providing application details through the "New App" option in the web UI.
* Set up the source repository and specify the path for your application configuration.
* Define the synchronization policy and enable automated synchronization for efficient management.
* Implement a canary deployment strategy within your application configuration. This involves specifying Kubernetes resources for a canary version, configuring Services and Ingress, and defining traffic splitting rules for controlled deployment.

### **Architecture diagram:-**

### 

### **Reference:-**

Argocd:-

* https://blog.devgenius.io/how-to-deploy-argocd-in-eks-cluster-for-continuous-deployment-6ebbb300902
* https://www.infracloud.io/blogs/progressive-delivery-argo-rollouts-canary-deployment/

Canary:-

* https://www.redhat.com/architect/blue-green-canary-argo-rollouts
* https://dev.to/pavanbelagatti/kubernetes-deployments-rolling-vs-canary-vs-blue-green-4k9p

Canary Architecture:-

* https://dev.to/mostlyjason/intro-to-deployment-strategies-blue-green-canary-and-more-3a3