**EKS UPGRADE**

| **Step/Task** | | **Command** | **Description** | **References** |
| --- | --- | --- | --- | --- |
| **Step 1** | **Upgrade and Backup Kubernetes Resources** | | | |
|  | Install Velero | - | Follow Velero installation documentation. | Velero Documentation https://velero.io/docs/v1.7/ |
| Create Velero Credentials | - | Create velero-credentials file with AWS credentials. | - |
| Create Velero Backup Storage Location | kubectl apply -f velero-config.yaml | Creates Velero backup storage location. | - |
| Create Velero Backup | velero backup create eks-backup --include-namespaces=default,kube-system | Creates Velero backup for specified namespaces. | - |
| Check Backup Status | velero backup describe eks-backup | Checks status of the backup. | - |
| Velero Restore | velero restore create --from-backup eks-backup | Creates Velero restore using the backup. | - |
| Check Restore Status | velero restore describe <restore-name> | Checks status of the restore. | - |
| Verify Restored Resources | kubectl get all --all-namespaces | Checks restored resources. | - |
| Monitor Restore Progress | velero restore logs <restore-name> | Monitors progress of the restore. | - |
| Cleanup (Optional) | velero restore delete <restore-name> | Deletes the restore if needed. | - |
| **Step 2** | **Upgrade Worker Nodes** | | | |
|  | Update Kubeconfig | aws eks update-kubeconfig --name <cluster-name> | Fetches cluster info and updates kubeconfig. | [AWS EKS - Update Worker Nodes](https://chat.openai.com/c/94bb8eea-21e7-4771-a411-4df97e9fe96f#) |
| Verify Node Draining | kubectl drain <node-name> --ignore-daemonsets | Drains nodes to prevent disruption to workloads. | [kubectl drain - Kubernetes Documentation](https://chat.openai.com/c/94bb8eea-21e7-4771-a411-4df97e9fe96f#) |
| Upgrade Worker Nodes | aws eks --region <region> update-nodegroup-version --cluster-name <cluster-name> --nodegroup-name <nodegroup-name> --kubernetes-version <new-version> | Upgrades worker nodes by scaling the node group. | - |
| Monitor Node Group Update | aws eks --region <region> describe-update --name <cluster-name> --nodegroup-name <nodegroup-name> | Checks status of node group update. | - |
| Verify Node Update | kubectl get nodes | Ensures all nodes are in Ready state. | - |
| Uncordon Nodes | kubectl uncordon <node-name> | Makes node schedulable again. | - |
| Verify Application Functionality | - | Confirm applications run on updated nodes. | - |
| **Step 3** | **Upgrade Control Plane** | | | |
|  | Upgrade Control Plane | eksctl upgrade cluster --name <cluster-name> --region <region> | Upgrades EKS control plane. | Follow AWS documentation |
| **Step 4** | **Upgrade CNI** | | | |
|  | Identify Current Calico Version | kubectl get daemonset calico-node -n kube-system -o=jsonpath='{.spec.template.spec.containers[0].image}' | Retrieves current Calico version. | - |
| Download Latest Calico Manifest | wget https://docs.projectcalico.org/manifests/calico.yaml | Downloads Calico manifest for the latest release. | - |
| Verify Existing Calico Resources | kubectl get pods -n kube-system -l k8s-app=calico-node<br>kubectl get configmap -n kube-system calico-config -o yaml | Checks current Calico resources. | - |
| Apply New Calico Manifest | kubectl apply -f calico.yaml | Applies new Calico manifest to upgrade CNI plugin. | - |
| Monitor Calico Pods | kubectl get pods -n kube-system -l k8s-app=calico-node | Checks Calico pods status. | - |
| Verify CNI Upgrade | kubectl get pods -n kube-system | Ensures Calico-related pods are running without issues. | - |
| Check Calico Node Status | kubectl get nodes -o wide | Verifies Calico nodes are in Ready state. | - |
| **Step 5** | **Upgrade CRDs** | | | |
|  | Backup CRDs | kubectl get crds -o yaml > crds-backup.yaml | Retrieves all CRDs and writes YAML output to file. | - |
|  | Identify Current CRD Version | kubectl get crds <your-crd-name> | Retrieves the current version of a specific CRD. | - |
| Download Latest CRD Manifest | Visit the official source for your CRD and download the latest manifest. | Download the CRD manifest for the corresponding release. | - |
| Verify Existing CRDs | kubectl get crds <your-crd-name> | Checks the current state of the CRD. | - |
| Apply New CRD Manifest | kubectl apply -f <new-crd-manifest.yaml> | Applies the new CRD manifest to upgrade the CRD. | - |
| Monitor CRD Update | kubectl get crds <your-crd-name> | Checks the status of the CRD update. | - |
| Verify CRD Upgrade | kubectl describe crds <your-crd-name> | Verifies that the CRD has been updated successfully. | - |
| **Step 6** | **Verify Application Functionality** | | | |
|  | Check Pods Status | kubectl get pods -n <your-namespace> | Ensures application pods are in Running state. | - |
| Check Services | kubectl get services -n <your-namespace> | Verifies services have expected ClusterIPs. | - |
| Application Testing | - | Perform application-specific tests. | - |
| Monitor Logs | kubectl logs <pod-name> -n <your-namespace> | Checks logs for errors or issues. | - |
|  | **– THANK YOU –** | | | |

##### 