**Step 1:**Setup credentials to login and connect prod eks

* Get the aws credentials and set in local aws cli profile
* Configure EKS to connect

**aws eks --region us-east-1 update-kubeconfig --name production-eks**

* Optionally verify that the kubeconfig is using the correct profile

**kubectl config view --minify**

* Test connection

**kubectl get pods --all-namespaces**

* Get the output of the get pods command and save it for knowing currently running all nodes

**Step 2:** Update your cluster using eksctl, the AWS Management Console, or the AWS CLI.

* This procedure requires eksctl version 0.74.0 or later. You can check your version with the following command:

**eksctl version**

          Installing or upgrading eksctl - <https://docs.aws.amazon.com/eks/latest/userguide/eksctl.html#installing-eksctl>

* Upgrade one minor version later

**eksctl upgrade cluster --name <my-cluster> --approve**

**Step 3:**Create and update a new image with version that is intended to get updated on

* Go to dpp-infra repository in eks\_ami folder
* Install packer on local machine
* Set AWS credential environment variables
* Get rapid7\_token
* Find the latest eks\_ami id from: <https://docs.aws.amazon.com/eks/latest/userguide/eks-optimized-ami.html>
* Run following command to get packer build

**packer build -var 'rapid7\_token={{ Rapid7 Install Token }}' -var "eks\_ami=<image-id>" eks\_worker.json**

* This will create new ami with intended k8s version - note AMI ID from output
* Update [eks.tf](http://eks.tf/) file of the environment intended to upgrade for "eks\_workers\_ami" variable with new AMI ID got in previous step

**Step 4: Update EKS worker AMI**

* Taint the existing launch templates to pick up the AMI change

**terraform taint module.eks\_cluster.aws\_launch\_template.eks\_worker\_node[0]**

* Run this for every Node group in the cluster

**terraform taint ... .eks\_worker\_node[1], terraform taint ... .eks\_worker\_node[2], etc.**

* Make sure you are using right context

**kubectl config use-context {{ cluster name }}**

* Get all the nodes

**kubectl get nodes**

* Cordon all the nodes.

**kubectl cordon <node-name>**

with xargs:

**kubectl get nodes -o go-template='{{range .items}}{{printf "%s\n" .metadata.name}}{{end}}' | xargs -n1 kubectl cordon**

* Scale down cluster autoscaler to 0.

(You will have to pause Weave Flux or it will keep re-setting the autoscaler replica to 1)

**kubectl scale deployments/flux --replicas=0 -n default**

**kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system**

* On the AWS Console, EC2 Page, Autoscaling tab on the left, double the desired instance amount for each autoscaling group that is being updated. (Make sure you spin up at least 3 new misc nodes. This is to make sure a node comes up in each AZ so persistent volumes can be re-attached correctly for certain pods.)
* Drain all the old cordoned nodes. We use the --ignore-daemonsets flag because the daemonsets run on every node in the cluster so its OK if they stick around as they will be brought up on the new instances.

Without xargs (one node at a time)

**kubectl drain {{ next node }} --ignore-daemonsets --delete-local-data**

With xargs

**kubectl get nodes -o go-template='{{range .items}}{{$node:=.metadata.name}}{{range .spec.taints}}{{if eq .effect "NoSchedule"}}{{printf "%s\n" $node}}{{end}}{{end}}{{end}}' | xargs -n1 kubectl --ignore-daemonsets --delete-local-data drain**

* Scale up cluster autoscaler and flux back to 1

**kubectl scale deployments/flux --replicas=1 -n default**

**kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system**

* At this point, the kubernetes autoscaler will take down the old cordoned nodes, This can take several minutes

**Step 5:** Cluster AutoScaler Upgrade

* Download same version of Autoscaler as control plane needs to be upgraded on

           Version for 1.18 - <https://github.com/kubernetes/autoscaler/releases/tag/cluster-autoscaler-1.18.3>

           Image: [k8s.gcr.io/autoscaling/cluster-autoscaler:v1.18.3](http://k8s.gcr.io/autoscaling/cluster-autoscaler:v1.18.3)

* Set the Cluster Autoscaler image tag to the version that you recorded in the previous step with the following command. If necessary, replace 1.18.3 with your own value.

**kubectl -n kube-system set image deployment.apps/cluster-autoscaler cluster-autoscaler=**[**k8s.gcr.io/autoscaling/cluster-autoscaler:v1.18.3**](http://k8s.gcr.io/autoscaling/cluster-autoscaler:v1.18.3)

**Step 6:**VPC CNI upgrade (<https://docs.aws.amazon.com/eks/latest/userguide/managing-vpc-cni.html#adding-vpc-cni-eks-add-on>)

* Get Current version of VPC CNI add-on

**kubectl describe daemonset aws-node --namespace kube-system | grep Image | cut -d "/" -f 2**

           Output: amazon-k8s-cni:v1.6.3

* Get latest verison - 1.9

           Download the manifest file

**curl -o aws-k8s-cni.yaml**[**https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/release-1.9/config/v1.9/aws-k8s-cni.yaml**](https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/release-1.9/config/v1.9/aws-k8s-cni.yaml)

* Change region name In file

**sed -i.bak -e 's/us-west-2/<region-code>/' aws-k8s-cni.yaml**

* Change Account name in file:

          (Find Account name here: <https://docs.aws.amazon.com/eks/latest/userguide/add-ons-images.html>)

**sed -i.bak -e 's/602401143452/<account>/' aws-k8s-cni.yaml**

* Apply manifest file

**kubectl apply -f aws-k8s-cni.yaml**

**Step 7:**Core-DNS upgrade (<https://docs.aws.amazon.com/eks/latest/userguide/managing-coredns.html#updating-coredns-add-on>)

* Get current Core-DNS version

**kubectl describe deployment coredns --namespace kube-system | grep Image | cut -d "/" -f 3**

          Output: coredns:v1.6.6

* Check to see if your CoreDNS manifest has a line that only has the word upstream

**kubectl get configmap coredns -n kube-system -o jsonpath='{$.data.Corefile}' | grep upstream**

          Output: Upstream

         If found

        Edit the configmap, removing the line near the top of the file that only has the word upstream. Don't change anything else in the file. After the line is removed, save the changes.

**kubectl edit configmap coredns -n kube-system -o yaml**

* Get the current Image of coreDNS getting used

**kubectl get deployment coredns --namespace kube-system -o=jsonpath='{$.spec.template.spec.containers[:1].image}'**

          Output: [602401143452.dkr.ecr.us-east-1.amazonaws.com/eks/coredns:v1.6.6](http://602401143452.dkr.ecr.us-east-1.amazonaws.com/eks/coredns:v1.6.6)

* If updating to CoreDNS 1.8.3,

          Add the endpointslices permission to the system:coredns Kubernetes clusterrole.

**kubectl edit clusterrole system:coredns -n kube-system**

* Add the following line under the existing permissions lines in the file.

- apiGroups:

  - [discovery.k8s.io](http://discovery.k8s.io/)

  resources:

  - endpointslices

  verbs:

  - list

  - watch

* Update the CoreDNS add-on by replacing <602401143452> (including <>) , <cn-north-1>, and <com> with the values from the output returned in the previous step. Replace <1.8.4> with your cluster's recommended CoreDNS version or later - (Find right version here: <https://docs.aws.amazon.com/eks/latest/userguide/managing-coredns.html#coredns-versions>)

**kubectl set image --namespace kube-system deployment.apps/coredns coredns=<602401143452>.dkr.ecr.<us-west-2>.amazonaws.<com>/eks/coredns:v<1.8.4>-eksbuild.1**

**Example:**

**kubectl set image --namespace kube-system deployment.apps/coredns coredns=602401143452.dkr.ecr.us-east-1.amazonaws.com/eks/coredns:v1.7.0-eksbuild.1**

**Step 8:**Upgrading kube-proxy (<https://docs.aws.amazon.com/eks/latest/userguide/managing-kube-proxy.html#adding-kube-proxy-eks-add-on>)

* Check current kube-proxy version

**kubectl get daemonset kube-proxy --namespace kube-system -o=jsonpath='{$.spec.template.spec.containers[:1].image}'**

           Output: [602401143452.dkr.ecr.us-east-1.amazonaws.com/eks/kube-proxy:v1.17.9-eksbuild.1](http://602401143452.dkr.ecr.us-east-1.amazonaws.com/eks/kube-proxy:v1.17.9-eksbuild.1)

* Use proper version of kube-proxy from following link and change account id and region as per previous command's output

<https://docs.aws.amazon.com/eks/latest/userguide/managing-kube-proxy.html#kube-proxy-default-versions-table>

**kubectl set image daemonset.apps/kube-proxy -n kube-system kube-proxy=**[**602401143452.dkr.ecr.us-west-2.amazonaws.com/eks/kube-proxy:v1.18.8-eksbuild.1**](http://602401143452.dkr.ecr.us-west-2.amazonaws.com/eks/kube-proxy:v1.18.8-eksbuild.1)

**Step 9:** Test the upgrade with checking sanity of the applications