

Kubernetes Updating Cluster

Scenario:

Updating the kubernetes cluster of the EKS v1.30 to v1.31 through the awscli without zero downtime.

Steps:

Step-1: Launched a ec2 instance at first and then with awscli completely setup the EKS cluster in the region eu-west-1 without node group.

```
eksctl create cluster --name=EKS-1 \  
    --region=eu-west-1 \  
    --zones=eu-west-1a,eu-west-1b \  
    --version=1.30 \  
    --without-nodegroup
```

The screenshot displays the AWS EKS console interface for a cluster named 'EKS-1'. At the top, there are buttons for 'Delete cluster', 'Upgrade version', and 'View dashboard'. Below these, a blue banner contains a warning: 'End of standard support for Kubernetes version 1.30 is July 28, 2025. On that date, your cluster will enter the extended support period with additional fees. For more information, see the [pricing page](#).' An 'Upgrade now' button is located on the right of this banner. The main section, titled 'Cluster info', is divided into a grid of six cards. The first card shows 'Status' as 'Active'. The second card shows 'Kubernetes version' as '1.30'. The third card shows 'Support period' as 'Standard support until July 28, 2025'. The fourth card shows 'Provider' as 'EKS'. The fifth card shows 'Cluster health issues' with a green checkmark and a '0'. The sixth card shows 'Upgrade insights' with a green checkmark and a '0'. The seventh card shows 'Node health issues' with a green checkmark and a '0'.

Cluster info			
Status	Kubernetes version	Support period	Provider
Active	1.30	Standard support until July 28, 2025	EKS
Cluster health issues	Upgrade insights	Node health issues	
0	0	0	

Step-2: Then by using iam-oidc provider to communicate cluster nodes, pods to remaining aws services.

```
eksctl utils associate-iam-oidc-provider \  
    --region eu-west-1 \  
    --cluster EKS-1 \  
    --approve
```

Step-3: Created a node group with t2.medium and remain requirements as below.

```
eksctl create nodegroup --cluster=EKS-1 \  
    --region=eu-west-1 \  
    --name=node2 \  
    --node-type=t2.medium \  
    --nodes=2 \  
    --nodes-min=1 \  
    --nodes-max=2 \  
    --node-volume-size=20 \  
    --ssh-access \  
    --ssh-public-key=ireland \  
    --managed \  
    --asg-access \  
    --external-dns-access \  
    --full-ecr-access \  
    --appmesh-access \  
    --alb-ingress-access
```

Node groups (1) Info						Edit	Delete	Add node group
Node groups implement basic compute scaling through EC2 Auto Scaling groups.								
	Group name ▲	Desired size ▼	AMI release version ▼	Launch template ▼	Status ▼			
<input type="radio"/>	node2	2	1.30.7-20241225	eksctl-EKS-1-nodegroup-node2 (1)	Creating			

```
ubuntu@ip-172-31-33-119:~$ kubectl get nodes  
NAME                                STATUS    ROLES    AGE   VERSION  
ip-192-168-23-105.eu-west-1.compute.internal Ready    <none>   17m   v1.30.7-eks-59bf375  
ip-192-168-53-213.eu-west-1.compute.internal Ready    <none>   17m   v1.30.7-eks-59bf375  
ubuntu@ip-172-31-33-119:~$
```

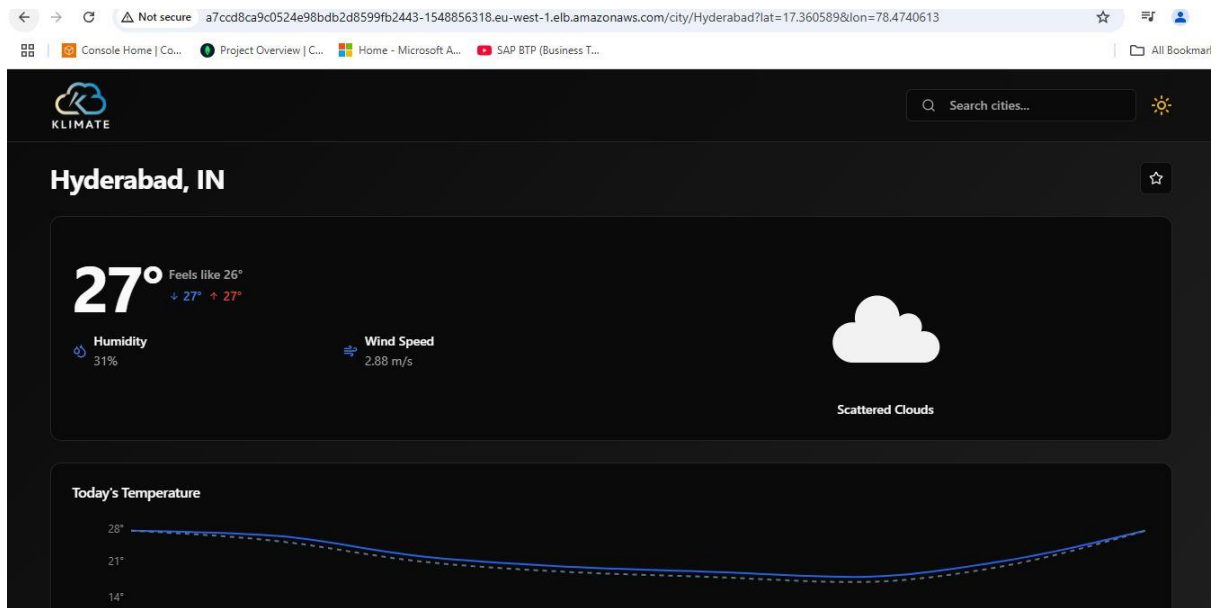
Step-4: After successful setup of the EKS cluster then, I deployed an application using Jenkins CICD pipeline in the ec2 instance.

The screenshot shows the Jenkins web interface for a pipeline named 'weatherapp'. The left sidebar contains navigation links: Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Stages, Rename, and Pipeline Syntax. The main area displays the 'Stage View' for the pipeline. It shows three stages: 'GitCheckout' (3s), 'K8 Deployment' (5s), and 'Verify Deployment' (783ms). Below the stages, there is a 'Permalinks' section with a list of build links: 'Last build (#1), 4 min 38 sec ago', 'Last stable build (#1), 4 min 38 sec ago', 'Last successful build (#1), 4 min 38 sec ago', and 'Last completed build (#1), 4 min 38 sec ago'. A 'Builds' section on the left shows a search filter and a list of builds.

Step-5: Load Balancer has been created and assigned with target instances and output

The screenshot shows the AWS Management Console for a Load Balancer. The 'Target instances' tab is selected, displaying a table of registered instances. The table has columns for Instance ID, Name, Health status, Health status description, and Security group. Two instances are listed, both with a health status of 'In-service'. The 'Target instances' section also includes a search filter, a 'Deregister' button, and a 'Manage instances' button.

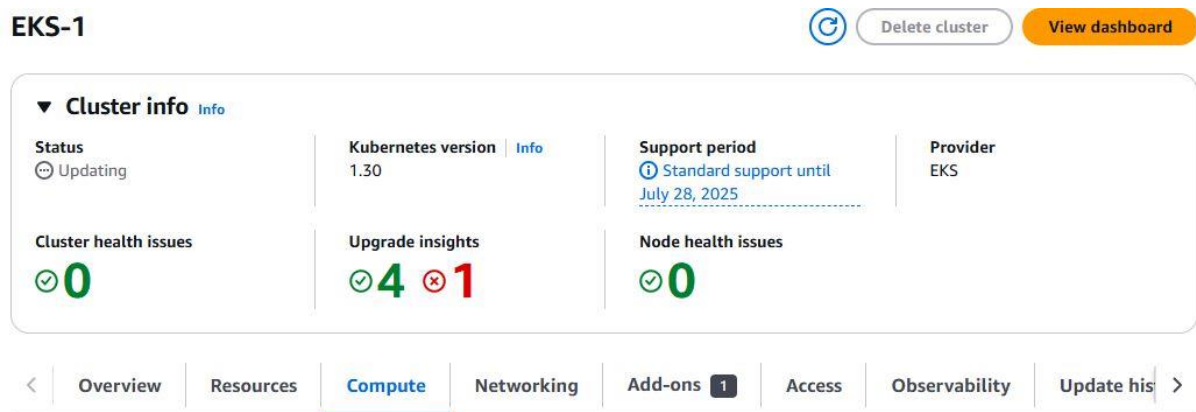
Instance ID	Name	Health status	Health status description	Security group
i-068f30ad622310125	EKS-1-node2-Node	In-service	Not applicable	eks-cluster-s
i-0d963fce78e365ddc	EKS-1-node2-Node	In-service	Not applicable	eks-cluster-s



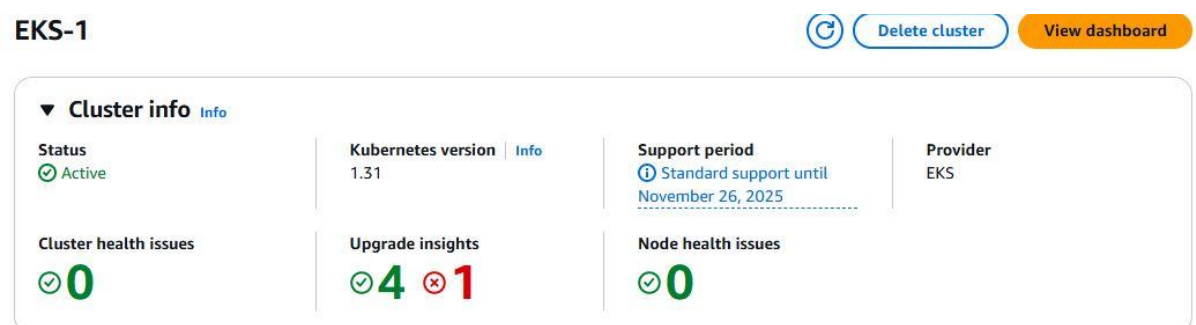
Step-6: Now updating the kubernetes cluster from v1.30 to v1.31 by awscli.

Syntax: `eks upgrade cluster --name <cluster-name> --region <region> --version 1.31 --approve`

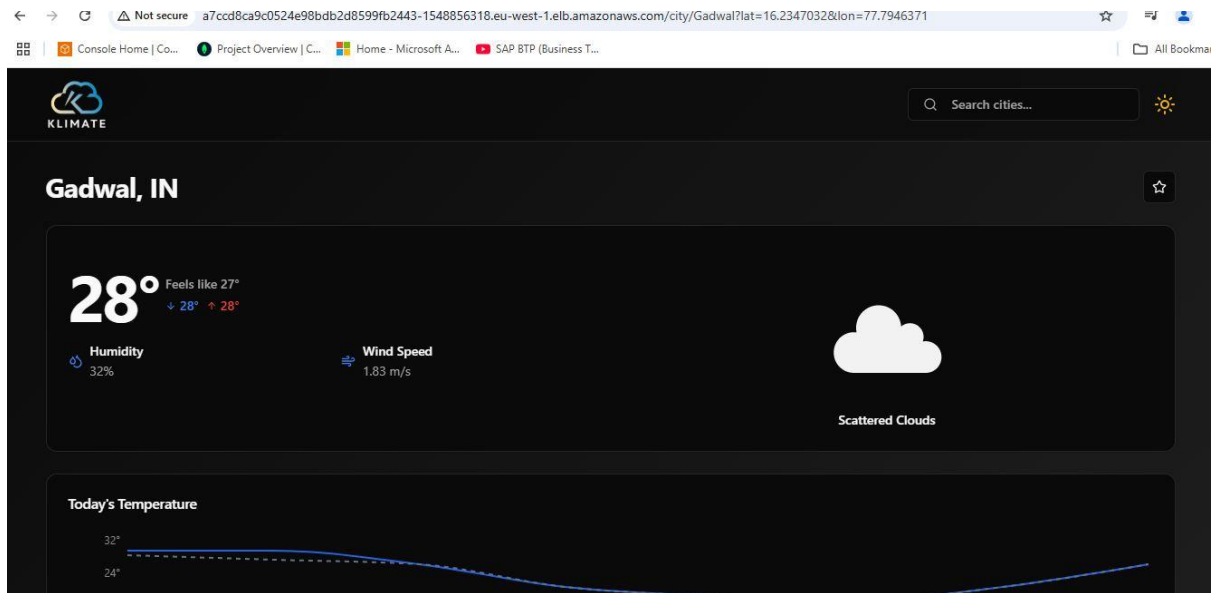
Step-7: The cluster updating in process and it will take some mean time to update it.



Updated to 1.31



Step-8: While updating the cluster there will be **zero Downtime** for the application, so here is the output



Step-9: After successful update of the cluster, then trying to update the node group by following the strategy Rollout strategy.

The screenshot shows the "Update node group version: node2" dialog in the AWS Management Console. The dialog has three main sections: "Update node group version" (selected with a radio button), "Change launch template version" (unselected), and "Update strategy" (selected with a radio button). The "Update strategy" section has a dropdown menu set to "Rolling update". A blue box contains a warning message: "This option respects pod disruption budgets for your cluster. The update fails if Amazon EKS is unable to gracefully drain the pods that are running on this node group due to a pod disruption budget issue." At the bottom, there are links for "Learn more" and "AMI release version notes", and two buttons: "Cancel" and "Update". A blue arrow points to the "Update" button.

Node name ▲	Instance type ▼	Compute ▼	Managed by ▼	Created ▼	Status ▼
<u>ip-192-168-23-105.eu-west-1.compute.internal</u>	t2.medium	Node group	node2	Created 36 minutes ago	Ready
<u>ip-192-168-28-113.eu-west-1.compute.internal</u>	t2.medium	Node group	node2	Created 5 minutes ago	Ready
<u>ip-192-168-34-48.eu-west-1.compute.internal</u>	t2.medium	Node group	node2	Created 5 minutes ago	Ready

Step-10: After update the cluster works fine

