Upgrading Kubernetes Clusters with kubeadm

40m 46s left

100% **Open Cloud Environment**
Setup completed
Average setup time: 2m 31s

100% **Open Code Environment**
Setup completed

Credentials

Account ID
45141212[illegible] Copy

Username
student Copy

Password
Ca1_HcGb[illegible] Copy

Region
US West [illegible] Copy

PEM Download PPK Download

Bastion Public Ip
35.92.49. Copy

Cluster SSH
ssh ubunt[illegible]

Introduction

`kubeadm` supports upgrading Kubernetes clusters. In this lab step, you will be upgrading Kubernetes from version 1.28.1 to version 1.28.2. Although upgrading is supported, you should always take care to understand any changes between releases by reading the release notes and how they could impact your workloads. You should always backup important data before upgrading, and test upgrades before deploying them to production.

The upgrade process follows the general procedure of:

1. Upgrading the Kubernetes control plane with `kubeadm` (Kubernetes components and add-ons excluding the CNI)
2. Manually upgrading the CNI network plugin, if applicable
3. Upgrading the Kubernetes packages (`kubelet`, `kubeadm`, `kubectl`) on the control-plane and worker nodes
4. Upgrading the `kubelet` config on worker nodes with `kubeadm`

Instructions

1. To begin the upgrade, `kubeadm` needs to be updated to `1.28.2`:

Copy code

```
sudo apt-get update
sudo apt-get install -y --allow-change-held-packages kubeadm=1.2
```

2. Generate an upgrade plan for upgrading Kubernetes to version 1.28.2:

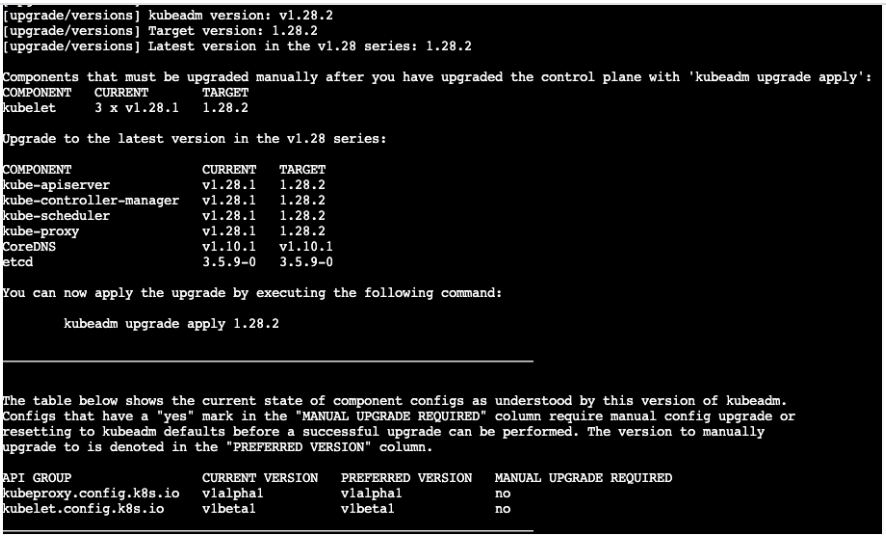
Copy code

```
sudo kubeadm upgrade plan 1.28.2
```

Support

Lab Steps

Need help? Contact our support team



As the output explains, several checks are performed, and the requirements for upgrading the cluster are first verified. A reminder that you need to manually upgrade the kubelet on each node in the cluster is then displayed. Future versions may remove this manual step. Finally, a summary of the planned version changes for all the cluster components (**COMPONENT**) is presented.

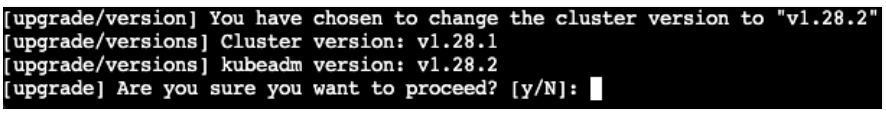
3. Apply the upgrade plan by issuing the following command and entering *y* when prompted:

Copy code

```
sudo kubeadm upgrade apply 1.28.2
```

`kubeadm` begins upgrading the cluster components on the control-plane node. Read through the output to understand what steps are being performed. It takes approximately four minutes to complete.

Enter *y* and press *Enter* to proceed:



You will see the following success message to know everything went as expected:

Note: If the upgrade procedure times out, you can safely try again until it run the command

4. Prepare to upgrade the control-plane node's kubelet by draining the node:

Copy code

```
# Get control-plane node name
control_plane=$(kubectl get nodes | grep control-plane | cut -d'
# Drain the control-plane node
kubectl drain $control_plane --ignore-daemonsets --delete-emptyc
```

```
node/ip-10-0-0-100.us-west-2.compute.internal drained
```

The --delete-emptydir-data is used to drain nodes that contain local storage. In this lab, we know there isn't any important data on them so we can safely perform this. However, it is important to evaluate and restore data if needed prior to using this in a production environment.

The --disable-eviction is used to bypass checking PodDisruptionBudgets. This should be used with caution and will cause downtime.

5. Upgrade the `kubelet`, and `kubectl` apt packages:

Copy code

```
sudo apt-get update
sudo apt-get upgrade -y --allow-change-held-packages \
     kubelet=1.28.2-00 kubectl=1.28.2-00
```

The upgrade may take a few minutes to complete.

Click *Enter* for any **Package configuration** pop-ups:

```
Package configuration
```

6. Uncordon the control-plane to allow pods to be scheduled on it now that is has been upgraded:

Copy code

```
kubectl uncordon $control_plane
```

7. Get the node information to confirm that the version of the control-plane is 1.28.2:

Copy code

```
ip-10-0-0-10.us-west-2.compute.internal   Ready    <none>          35m   v1.28.1
ip-10-0-0-100.us-west-2.compute.internal  Ready    control-plane   26d   v1.28.2
ip-10-0-0-11.us-west-2.compute.internal   Ready    <none>          34m   v1.28.1
```

8. Drain the worker node 1 to prepare it for upgrading:

Copy code

```
# Get the worker's name
worker1=$(kubectl get nodes | grep 10.u | cut -d' ' -f1)
# Drain the worker node
kubectl drain $worker1 --ignore-daemonsets --delete-emptydir-dat
```

```
node/ip-10-0-0-10.us-west-2.compute.internal cordoned
Warning: ignoring DaemonSet-managed Pods: kube-system/calico-node-jxc2g, kube-system/ebs-csi-node-kcg7h, kube-system/kube-proxy-xvcz5
evicting pod kubernetes-dashboard/kubernetes-dashboard-7c745c7dc-w5qzt
evicting pod kube-system/ebs-csi-controller-75b7b645cc-vnlls
evicting pod kube-system/calico-kube-controllers-f88756749-tw9kt
evicting pod kube-system/coredns-565d847f94-c4xdb
evicting pod kube-system/metrics-server-77b976bb4b-2kmxh
evicting pod kube-system/ebs-csi-controller-75b7b645cc-ztjt6
evicting pod kubernetes-dashboard/dashboard-metrics-scraper-77d78b7997-jjfb4
error when evicting pods/"ebs-csi-controller-75b7b645cc-vnlls" -n "kube-system" (will retry after 5s): Cannot evict pod as it would violate the pod's disruption budget.
pod/metrics-server-77b976bb4b-2kmxh evicted
pod/kubernetes-dashboard-7c745c7dc-w5qzt evicted
pod/dashboard-metrics-scraper-77d78b7997-jjfb4 evicted
pod/calico-kube-controllers-f88756749-tw9kt evicted
evicting pod kube-system/ebs-csi-controller-75b7b645cc-vnlls
error when evicting pods/"ebs-csi-controller-75b7b645cc-vnlls" -n "kube-system" (will retry after 5s): Cannot evict pod as it would violate the pod's disruption budget.
pod/ebs-csi-controller-75b7b645cc-ztjt6 evicted
pod/coredns-565d847f94-c4xdb evicted
evicting pod kube-system/ebs-csi-controller-75b7b645cc-vnlls
error when evicting pods/"ebs-csi-controller-75b7b645cc-vnlls" -n "kube-system" (will retry after 5s): Cannot evict pod as it would violate the pod's disruption budget.
evicting pod kube-system/ebs-csi-controller-75b7b645cc-vnlls
pod/ebs-csi-controller-75b7b645cc-vnlls evicted
node/ip-10-0-0-10.us-west-2.compute.internal drained
```

9. SSH into worker node 1:

Copy code

```
ssh worker1 -oStrictHostKeyChecking=no
```

10. Drain the node and upgrade the Kubernetes packages:

Copy code

```
sudo apt-get update
sudo apt-get upgrade -y --allow-change-held-packages \
     kubelet=1.28.2-00 kubeadm=1.28.2-00 kubectl=1.28.2-00
```

Click *Enter* for any **Package configuration** pop-ups:

```
Package configuration
```

11. Restart the worker node's `kubelet` and return to the control-plane :

Copy code

Copy code

```
kubectl uncordon $worker1
```

```
node/ip-10-0-0-10.us-west-2.compute.internal uncordoned
```

13. Confirm the worker node is ready and running version 1.28.2:

Copy code

```
kubectl get nodes
```

```
NAME                                         STATUS   ROLES           AGE   VERSION
ip-10-0-0-10.us-west-2.compute.internal      Ready    <none>          37m   v1.28.2
ip-10-0-0-100.us-west-2.compute.internal     Ready    control-plane   26d   v1.28.2
ip-10-0-0-11.us-west-2.compute.internal      Ready    <none>          36m   v1.28.1
```

The upgrade process is now complete. You can create some pods if you want to further test whether the upgrade succeeded.

14. Drain the worker node 2 to prepare it for upgrading:

Copy code

```
# Get the worker's name
worker2=$(kubectl get nodes | grep 11.u | cut -d' ' -f1)
# Drain the worker node
kubectl drain $worker2 --ignore-daemonsets --delete-emptydir-dat
```

```
node/ip-10-0-0-10.us-west-2.compute.internal drained
```

15. SSH into worker node 2:

Copy code

```
ssh worker2 -oStrictHostKeyChecking=no
```

16. Drain the node and upgrade the Kubernetes packages:

Copy code

```
sudo apt-get update
sudo apt-get upgrade -y --allow-change-held-packages \
     kubelet=1.28.2-00 kubeadm=1.28.2-00 kubectl=1.28.2-00
```

17. Restart the worker node's `kubelet` and return to the control-plane node :

Copy code

```
sudo systemctl restart kubelet
# Return to control-plane node
exit
```

18. Uncordon the worker node 2:

Copy code

```
kubectl uncordon $worker2
```

```
node/ip-10-0-0-55 uncordoned
```

19. Confirm all nodes are running version 1.28.2:

Copy code

```
kubectl get nodes
```

```
NAME                                        STATUS                     ROLES           AGE   VERSION
ip-10-0-0-10.us-west-2.compute.internal     Ready                      <none>          51m   v1.28.2
ip-10-0-0-100.us-west-2.compute.internal    Ready                      control-plane   26d   v1.28.2
ip-10-0-0-11.us-west-2.compute.internal     Ready,SchedulingDisabled   <none>          50m   v1.28.2
```

The upgrade process is now complete. You can create some pods if you want to further test the success of the upgrade.

Summary

In this lab step, you went through the process of upgrading the cluster using `kubeadm`.

VALIDATION CHECKS

1 Checks Start check