

Module 2b: Upgrading Kubernetes Cluster

This lab guides you through the process of safely upgrading a Kubernetes cluster from version 1.33 to 1.34.1. The upgrade follows a methodical approach, starting with the control plane (master node) before proceeding to worker nodes. This ensures minimal disruption while maintaining cluster availability throughout the process.



Understanding the Upgrade Process

Why Upgrade Sequentially?

Kubernetes upgrades must follow a specific order to maintain cluster stability. The control plane components are upgraded first, followed by worker nodes one at a time. This approach ensures the cluster remains operational and can handle workload rescheduling during the upgrade process.

Key Components to Upgrade

- kubeadm (upgrade tool)
- kubelet (node agent)
- kubectl (CLI tool)

Each component must be upgraded in the correct sequence to avoid version skew issues.

Phase 1: Update Package Repository

01

Modify Repository Reference

Update the Kubernetes repository from version 33 to 34 using sed command

02

Refresh Package Lists

Run apt update to fetch the latest package information from updated repositories

03

Verify Available Versions

Check available kubeadm versions to confirm access to target version 1.34.1

```
sed -i 's/33/34/g' /etc/apt/sources.list.d/kubernetes.list  
apt update  
apt-cache madison kubeadm
```

The sed command performs an in-place substitution, changing all instances of "33" to "34" in the Kubernetes repository configuration file. This points your package manager to the correct version stream.

Phase 2: Upgrade kubeadm on Master



Unhold Package

Remove the version lock from kubeadm to allow upgrades



Install New Version

Install kubeadm version 1.34.1 using apt package manager



Hold Package

Lock the new version to prevent automatic updates



Verify Installation

Confirm the installed kubeadm version matches target

```
apt-mark unhold kubeadm  
apt install -y kubeadm=1.34.1-*  
apt-mark hold kubeadm  
kubeadm version
```

Phase 3: Prepare and Upgrade Control Plane



Evict Pods from Node

Drain the master node to safely move workloads. DaemonSets like cilium remain as they must run on every node.



Review Upgrade Plan

Run `kubeadm upgrade plan` to see detailed information about the upgrade process and verify prerequisites.



Execute Upgrade

Apply the upgrade to control plane components. This updates etcd, API server, scheduler, and controller manager.

```
kubectrl drain master --ignore-daemonsets  
kubeadm upgrade plan  
kubeadm upgrade apply v1.34.1
```



Important: Read the upgrade plan output carefully. It shows which components will be upgraded and identifies any potential issues before proceeding.

Phase 4: Upgrade kubelet and kubectl

After upgrading the control plane components, the node-level components (kubelet and kubectl) must be upgraded separately. At this stage, **kubectl get nodes** still shows the old version because the kubelet hasn't been updated yet. The master node also shows **SchedulingDisabled** status due to the earlier drain operation.

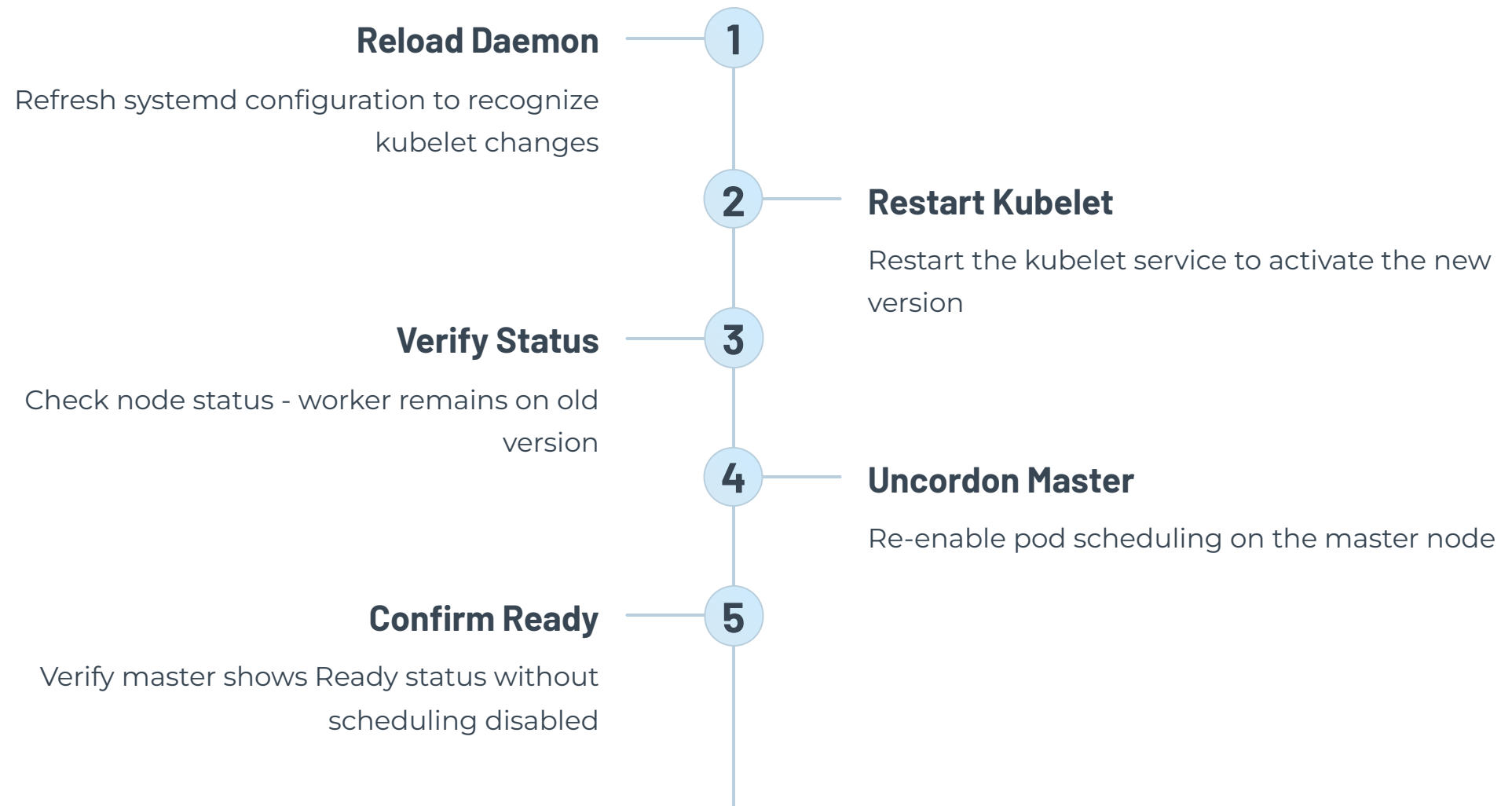
The kubelet is the primary node agent that runs on each node, managing pod lifecycle. The kubectl CLI tool enables cluster interaction from the command line.

Commands

```
apt-mark unhold  
kubelet kubectl  
apt install -y  
kubelet=1.34.1-*  
kubectl=1.34.1-*  
apt-mark hold kubelet  
kubectl
```



Phase 5: Restart Services and Re-enable Scheduling



```
systemctl daemon-reload
systemctl restart kubelet
kubectl get nodes
kubectl uncordon master
kubectl get nodes
```

The uncordon command removes the scheduling restriction, allowing pods to be scheduled on the master node again. The master node upgrade is now complete.

Phase 6: Upgrade Worker Nodes

Worker nodes follow a similar but slightly modified upgrade process. The key difference is using **kubeadm upgrade node** instead of **kubeadm upgrade apply**. This command upgrades the kubelet configuration and certificates on worker nodes without attempting to upgrade control plane components.

1

Update Repository

Modify repository reference and refresh package lists on the worker node

2

Upgrade kubeadm

Install the new kubeadm version following the same unhold-install-hold pattern

3

Drain Worker

Evict pods from worker node to prepare for upgrade

4

Upgrade Node Config

Run **kubeadm upgrade node** to update kubelet configuration

Complete Worker Upgrade Process




Finalize Worker Upgrade

After running `kubeadm upgrade node`, complete the process by upgrading `kubelet` and `kubectl`, restarting services, and re-enabling scheduling. These steps mirror the master node process.

For clusters with multiple worker nodes, repeat this process for each worker node sequentially. Never upgrade all workers simultaneously to maintain cluster availability.

```
apt-mark unhold kubelet kubectl
apt install -y kubelet=1.34.1-* kubectl=1.34.1-*
apt-mark hold kubelet kubectl
systemctl daemon-reload
systemctl restart kubelet
kubectl get nodes
kubectl uncordon worker
kubectl get nodes
```

 **Best Practice:** Verify each worker node reaches Ready status before proceeding to the next node. This ensures workload stability throughout the upgrade.

Upgrade Complete: Verification Checklist

3

Components Upgraded

kubeadm, kubelet, and kubectl all running version 1.34.1

100%

Node Availability

All nodes showing Ready status with scheduling enabled

0

Disrupted Workloads

Pods successfully rescheduled with zero permanent failures

Key Takeaways

- **Sequential Upgrades:** Always upgrade control plane before worker nodes
- **Version Locking:** Use apt-mark hold to prevent accidental automatic updates
- **Graceful Draining:** The drain command ensures pods are rescheduled before node maintenance
- **Service Restarts:** Daemon reload and kubelet restart activate the new version
- **Verification:** Check node status after each phase to confirm successful upgrade

This methodical approach minimizes downtime while ensuring cluster stability throughout the upgrade process. For multi-control-plane clusters, repeat the control plane upgrade steps on additional master nodes before proceeding to workers.