Upgrading the Azure Kubernetes Service (AKS) version

is a critical task that must be planned and executed carefully to ensure minimal disruption to your workloads. Here’s a step-by-step guide to upgrade an AKS cluster from version **1.23** to **1.23.15**:

**Step 1: Pre-upgrade Checklist**

1. **Check Supported Versions**: Verify the availability of the target version 1.23.15 for your region and cluster:

#az aks get-upgrades --resource-group <resource-group-name> --name <aks-cluster-name>

1. **Ensure Compatibility**:
   * Ensure all installed add-ons (e.g., Azure CNI, CoreDNS, Ingress Controllers) are compatible with the target version.
   * Verify your application workloads are compatible with Kubernetes 1.23.15.
2. **Backup Configurations**:
   * Backup critical Kubernetes resources (e.g., Deployments, Services, ConfigMaps) by exporting their YAML files:

#kubectl get all --all-namespaces -o yaml > all-resources-backup.yaml

* + Backup cluster state if using Azure-managed etcd or another state management tool.

1. **Update CLI Tools**:
   * Ensure you have the latest version of Azure CLI installed:

#az version

#az upgrade

**Step 2: Upgrade Node Pools**

AKS upgrades follow a two-phase process: the control plane is upgraded first, followed by the node pools.

**Upgrade Control Plane**

1. Run the upgrade command for the control plane:

#az aks upgrade --resource-group <resource-group-name> --name <aks-cluster-name> --kubernetes-version 1.23.15

1. Confirm the upgrade when prompted.

**Upgrade Node Pools**

1. List the node pools in your cluster:

#az aks nodepool list --resource-group <resource-group-name> --cluster-name <aks-cluster-name> --query "[].{Name:name, OrchestratorVersion:orchestratorVersion}" -o table

1. Upgrade each node pool individually:

#az aks nodepool upgrade --resource-group <resource-group-name> --cluster-name <aks-cluster-name> --name <nodepool-name> --kubernetes-version 1.23.15

**Step 3: Post-upgrade Validation**

1. **Verify Cluster Status**: Confirm the Kubernetes version of the cluster:

#kubectl version --short

1. **Check Node Versions**: Ensure all nodes are running the new version:

#kubectl get nodes

1. **Test Workloads**:
   * Verify that workloads are running as expected:

#kubectl get pods --all-namespaces

* + Check application logs for errors:

#kubectl logs <pod-name>

1. **Update Dependencies**: If necessary, update kubectl, Helm charts, or any client-side tools to match the new Kubernetes version.

**Step 4: Rollback Plan**

If the upgrade causes issues, you can recreate the cluster at the previous version using your backup and deployment scripts.