|  |  |
| --- | --- |
| Document type | Template |
| Document title | Management Processes for application specific AKS clusters |
| Valid from | xx.12.2022 |
| Version | Next revision |

**Management Processes for application specific AKS Windows clusters**

**Document History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author(s) | Change Description |
| 1.0 | 11.30.2022s | Shobana |  |

**Table of contents**

Order Namespace

Delete Namspace

Upgrade Node Images

Upgrade Kured

Upgrade Nginx Ingress

Upgrade CSI driver

Upgrade Resource quota

Upgrade Kubernetes version

Scope of the Document is to understand the procedure for namespace creation and patching in windows cluster .

1.Process to order Namespace

**Requestor** : Application Team

**Provider** : AKS Hact Team

**SNOW** : [Service Catalog - Uniper Service Portal (service-now.com)](https://uniperprod.service-now.com/unipersp/?id=sc_cat_item_uni&sys_id=9a5613dadbf82c10d1ebbaf1f39619c6)

Application team to raise requests for namespaces creation .

Details are required from Application team to order a Namespace through below listed form.

* AAD Group name
* Number of vCPU
* Size of Memory in GB
* Number of vCPU (limit)
* Size of Memory in GB (limit)
* Security Environment
* Namespace naming convention followed: applicationname-iterationplanid-env

Application team should ensure the following details for their Namespace.

* **RBAC**   
  Each namespace will be assigned to a namespace owner (an individual or an Active Directory group). The namespace owner can define namespace roles and assign it to additional users or teams within the assigned namespace.
* **Network policies**  
  Namespace are by default isolated from other namespace by a "Block all" network policy. This network policy can be edited by application teams to enable specific communication. It is strongly recommended to allow specific communication and not open communication in general or even delete the network policy.
* **Azure security policies**  
  Azure security policies control the permissions of a pod on the underlying node so that pods can’t execute operations that have an impact on other pods running on the same node. Pod security policies cannot be changed by application teams. Please find below wiki URL for polices

[Azure Policy built-in definitions for Azure Kubernetes Service - Hosting Framework (uniper.energy)](https://wiki.intranet.uniper.energy/sales/display/CF/Azure+Policy+built-in+definitions+for+Azure+Kubernetes+Service)

**How to Verify the newly created Namespace and Resource quota**

**To verify Pods once application is deployed**

$ kubectl get pods -n namespace\_name

**To Fetch details of capacity you have requested to us we set this is ‘Resource Quota’**$ kubectl describe quota requests-resourcequota --namespace=namespace\_name

SNOW for Quota update : [Service Catalog - Uniper Service Portal (service-now.com)](https://uniperprod.service-now.com/unipersp/?id=sc_cat_item_uni&sys_id=2a59125387118910e1b1a8e70cbb355f)

2.Namespace deletion process

**Requestor** : Application Team

**Provider** : AKS Hact Team

**SNOW** : [Service Catalog - Uniper Service Portal (service-now.com)](https://uniperprod.service-now.com/unipersp/?id=sc_cat_item_uni&sys_id=ca5613dadbf82c10d1ebbaf1f3961961)

Namespace administrator has the overall responsibility of managing Namespace but restricted with namespace deletion and HaCT AKS Team is responsible for Namespace deletion.

HaCT AKS team owns the responsibility to delete the namespace from the cluster .

HaCT AKS will update the comments section of ServiceNow ticket after successful deletion of Namespace which triggers an email regarding Namespace deletion status.

**Key Stakeholder**

Requester (Namespace Owner)

HaCT AKS Team

Line Manager Approval

**Input Requirements**

Requested for (Name will be auto filled)

Line manager (Name will be auto filled)

Namespace name

User ID (KID) of Namespace Administrator

Additional Contact Information (If any)

Additional Information (Example: Cluster Name.)

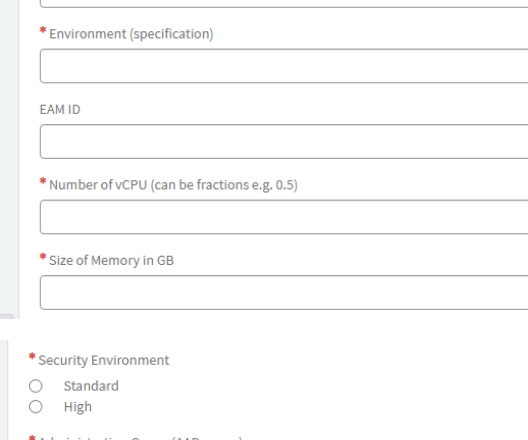
After successful deletion of Namespace, AKS Team will update the requester via email.

**Limitation of Namespace with windows cluster :**

1. By default selecting security environment and environment as Standard or High points to AKS linux clusters namespace creation and not windows so we should add a field to display the clustername as mandatory along with hint mentioning the recently created clusters name and also users should be notified on the selected options

2.Sample Namespace format to be mentioned in catalogue as it eases the reference and also easy for future records tracktion .

3.To add project name as mandatory text field .



**AKS Upgrade Communication Procedure**

**Purpose:**  
Upgrade Azure Kubernetes Service (AKS) & Apps installed on the cluster

**System Overview:**  
AKS cluster and all additional services installed on the cluster requires performing periodic upgrades to the latest versions to apply the latest security releases and the latest features.

HaCT AKS team will send email notifications to the corresponding teams for upgrading the components and its versions table and do changes in the latest versions.

|  |  |
| --- | --- |
| Notification | 6 weeks prior notification for Prod clusters |
|  | 4 weeks prior notification for Pre-Prod clusters |

Application team responsible to build the image and push it to unipercontainerregistry through azure Devops pipeline.​

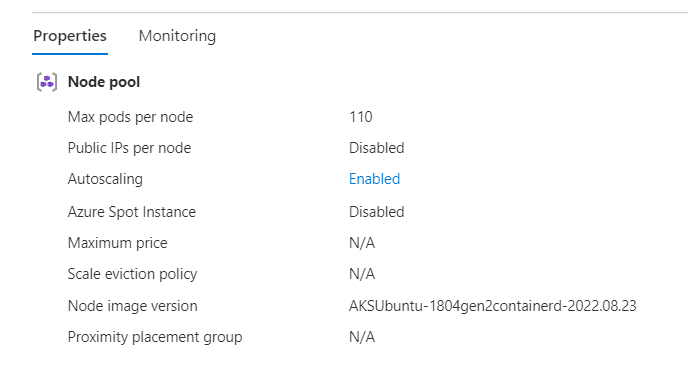
**Patching Difference between Linux and Windows :**

Kured will take care of automatic image upgrade on new patching releases with respect to Linux node pool.

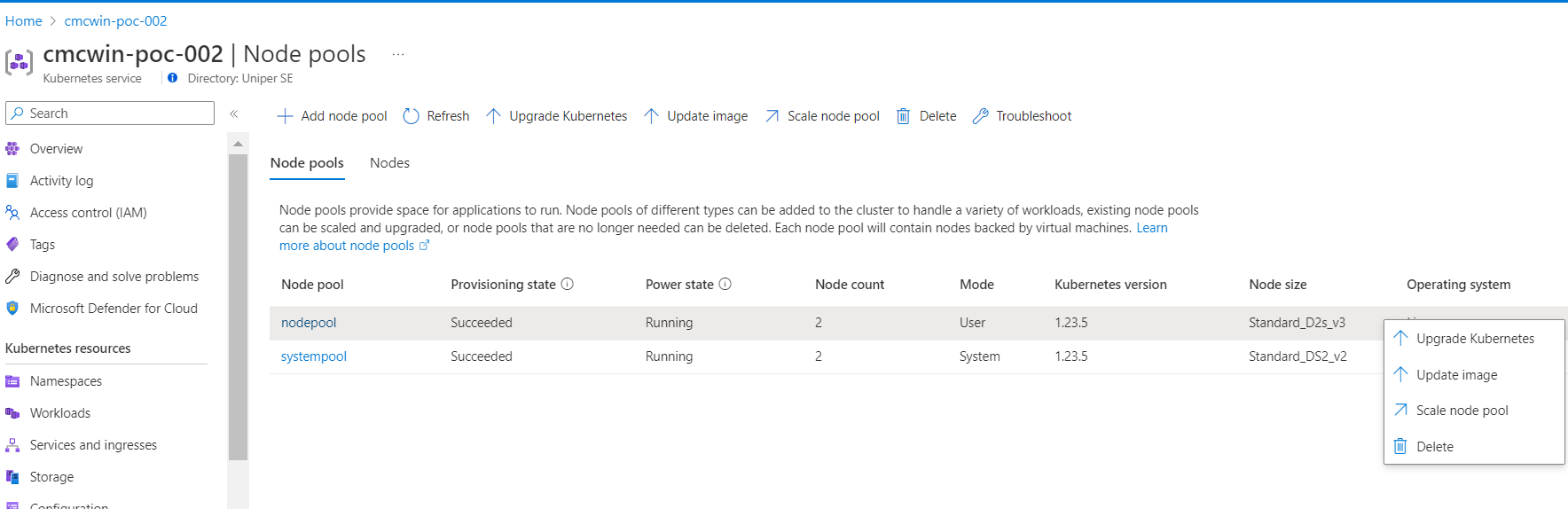
With respect to windows node pool , Hact team is responsible for image upgrade by manually rebooting the clusters .

**Node Image upgrade :**

**Check if your node pool is on the latest node image**



**Upgrade Image :**



Upgrading the node image is done with az aks upgrade in CLI also.To upgrade the node image, use the following command:

az aks upgrade \

--resource-group myResourceGroup \

--name myAKSCluster \

--node-image-only

When the upgrade is complete, use az aks show to get the updated node pool details. The current node image is shown in the nodeImageVersion property.

az aks show \

--resource-group myResourceGroup \

--name myAKSCluster

## Upgrade a specific node pool

Upgrading the image on a node pool is similar to upgrading the image on a cluster.

To update the OS image of the node pool without doing a Kubernetes cluster upgrade, use the --node-image-only option in the following example:

az aks nodepool upgrade \

--resource-group myResourceGroup \

--cluster-name myAKSCluster \

--name mynodepool \

--node-image-only

**AKS componentts upgrade (kured, nginx, CSI driver ( to be removed)​ process**

**Upgrade Kured:**

Kured (Kubernetes Reboot Daemon) is a Kubernetes daemonset that performs safe automatic node reboots when the need to do so is indicated by the package management system of the underlying OS.

**Example Cluster name: cmcwins-poc-001**

Login to cmcwin-poc-001 cluster

$ az account set --subscription d4665e52-c01b-4bdf-aa11-3da2eac23394

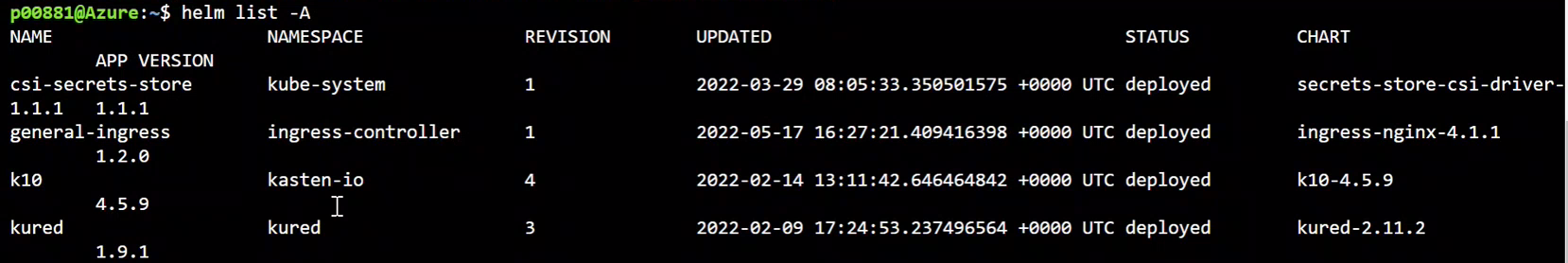
$ az aks get-credentials --resource-group cmcaks-paas-poc-rgp-001 --name cmcaks-poc-001

First, we need to upgrade some of the applications before cluster upgrade

These applications deployed using HELM Package manager

$ helm list -A

$ helm repo list



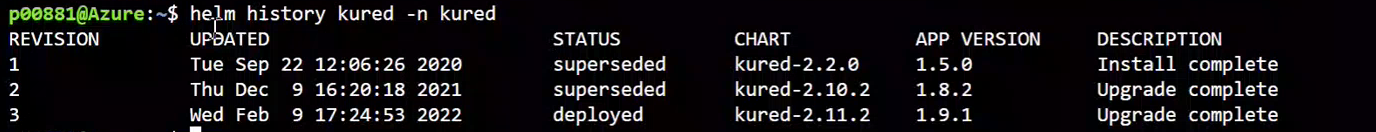
Check the history of kured application

$ helm history kured -n kured

$ k get ds -n kured

$ k get po -n kured

$ kubectl get pods -A --field-selector status.phase!=Running -o=custom-columns=NAME:.metadata.name,STATUS:.status.phase,NAMEDPACE:.metadata.namespace



Delete kured repo if it is existed already

$ helm repo remove kured

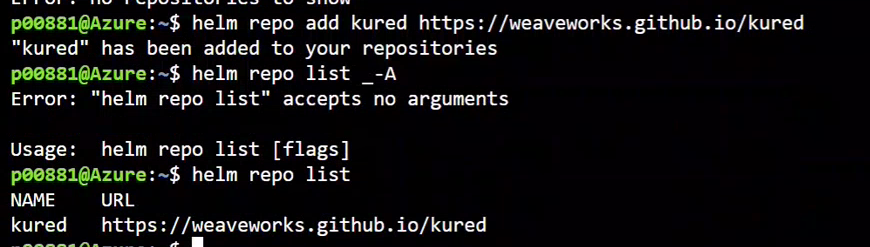
Add repo from Application URL

$ helm repo add kured <https://kubereboot.github.io/charts>

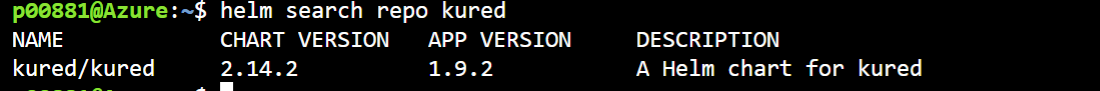
if unable to delete namespace in prod do this one

$ kubectl get apiservice | grep kas

$ k delete apiservice v1alpha1.apps.kio.kasten.io v1alpha1.vault.kio.kasten.io



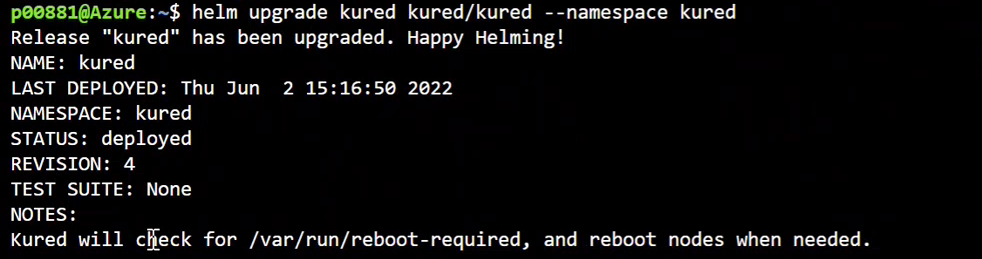
Upgrade kured application to latest one



$ helm upgrade <REPO NAME> <CHART NAME> -N <NAME SPACE NAME>

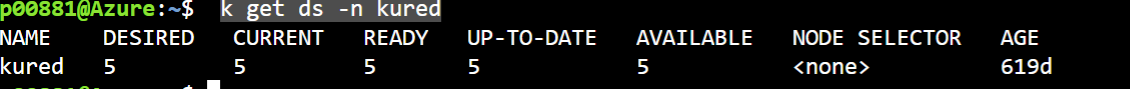
$ helm upgrade kured kured/kured --namespace kured

$ helm install kured kured/kured -n kured



Validate kured daemonsets up and running

$ k get ds -n kured



Validate kured application upgraded successfully or not and check all the pods are newly created or not.

$ Kubectl get pods -n kured

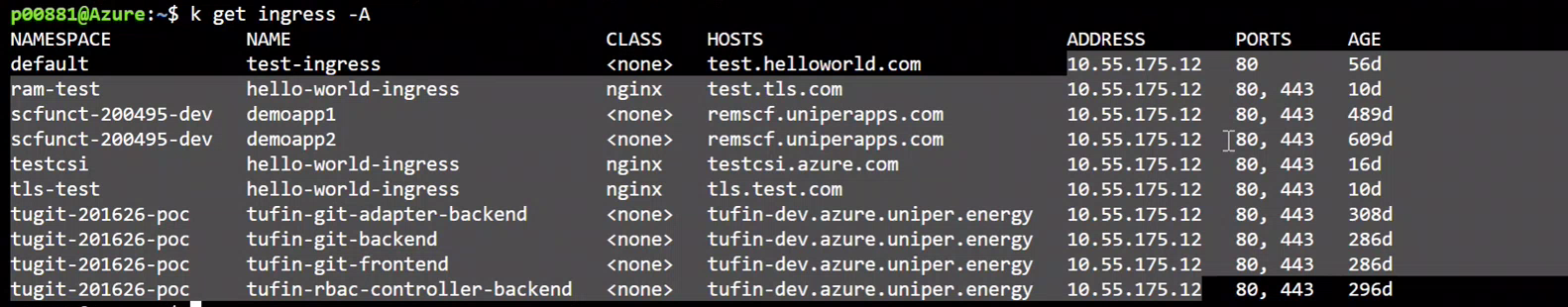


* **Note:-**No Downtime require for this and no need to communicate to Application Team for this.

**Upgrade Nginx:**

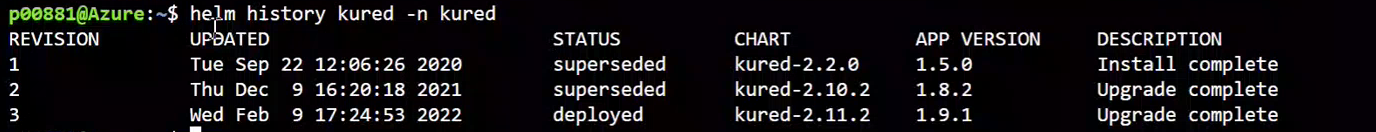
First check all the ingress controller is running or not

$ kubectl get ingress -A



Check the history of general-ingress application

$ helm history general-ingress -n ingress-controller



Delete general-ingress repo if it is existed already

$ helm repo remove general-ingress (DON’T REMOVE)

Add repo from Application URL

$ helm repo add general-ingress https://kubernetes.github.io/ingress-nginx (DON’T REMOVE)

JUST DO

$ helm repo upgrade

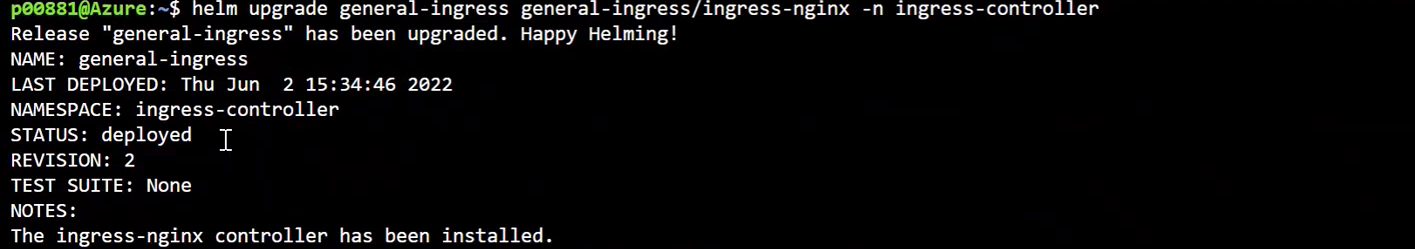


Upgrade general-ingress application to latest one

$ helm upgrade <REPO NAME> <CHART NAME> -N <NAME SPACE NAME>

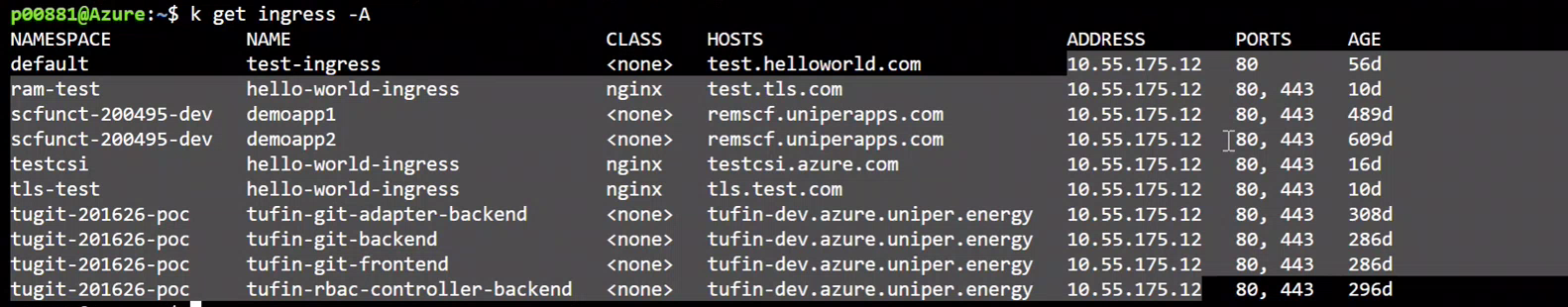
$ helm upgrade general-ingress general-ingress/ingress-nginx -n ingress-controller

$ helm upgrade ingress-nginx general-ingress/ingress-nginx -n ingress-controller[WORKING]

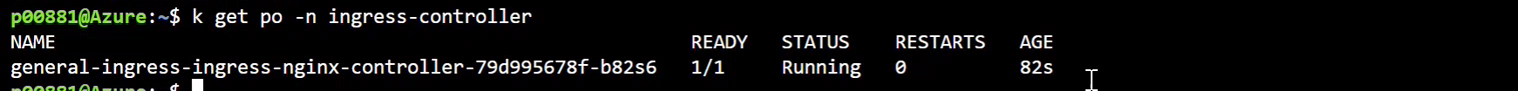


Validate general-ingress up and running

$ kubectl get ingress -A



$ k get po -n ingress-controller



**How does it relate to the cluster patching?**

We need to check the compatibility of the ingress controllers with Kubernetes version. So, we can upgrade this as per required version for Kubernetes Version.

​



**Upgrade Secret Store CSI driver:**

*helm upgrade* to the latest chart release in the repo will update the Secret Store CSI Driver to the compatible versions.

$ helm upgrade csi-secrets-store secrets-store-csi-driver/secrets-store-csi-driver --namespace=NAMESPACE

where the driver was originally installed, (i.e. kube-system)

**How and when are users informed?**

HaCT will initiate email to Application team regarding cluster upgrade 4 weeks prior to upgradation. For Production cluster we inform 6 weeks earlier.

**Who has which responsibility in the process?**

HaCT AKS team have responsibility to upgrade the cluster.

Note:- HaCT AKS team will include any one of the application team during upgradation time and will ask them for validation of their application after Cluster upgradation.

**How does the communication work?**

After successful upgradation AKS team communicate to Application team via email.

**How does it relate to the cluster upgradation?**

HaCT AKS team can check the Compatibility of drivers and Providers with AKS.

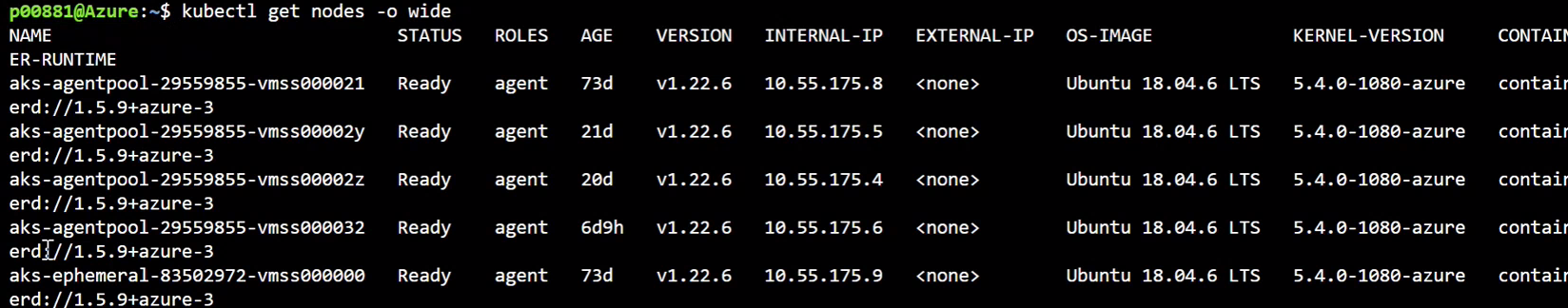
**Request for the resource quota update:**

Based on the Application team request HaCT AKS team can upgrade the resource quota by updating the below catalogue request : <https://uniperprod.service-now.com/unipersp?id=sc_cat_item_uni&sys_id=2a59125387118910e1b1a8e70cbb355f>

AKS cluster upgrade process:

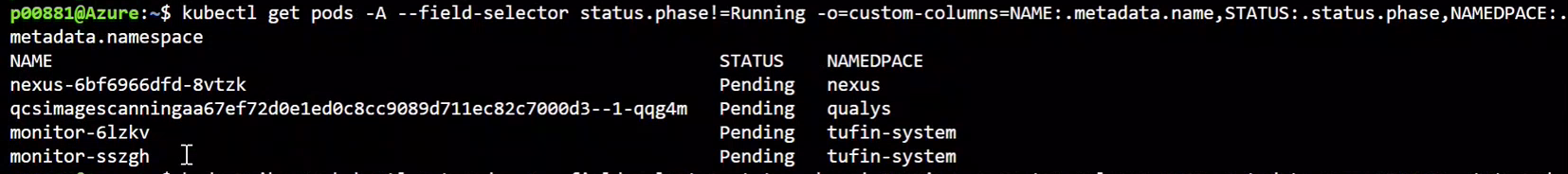
HaCT AKS team will check all the nodes, pods and other resources up and running before starting the upgrade process.

$ kubectl get nodes -o wide

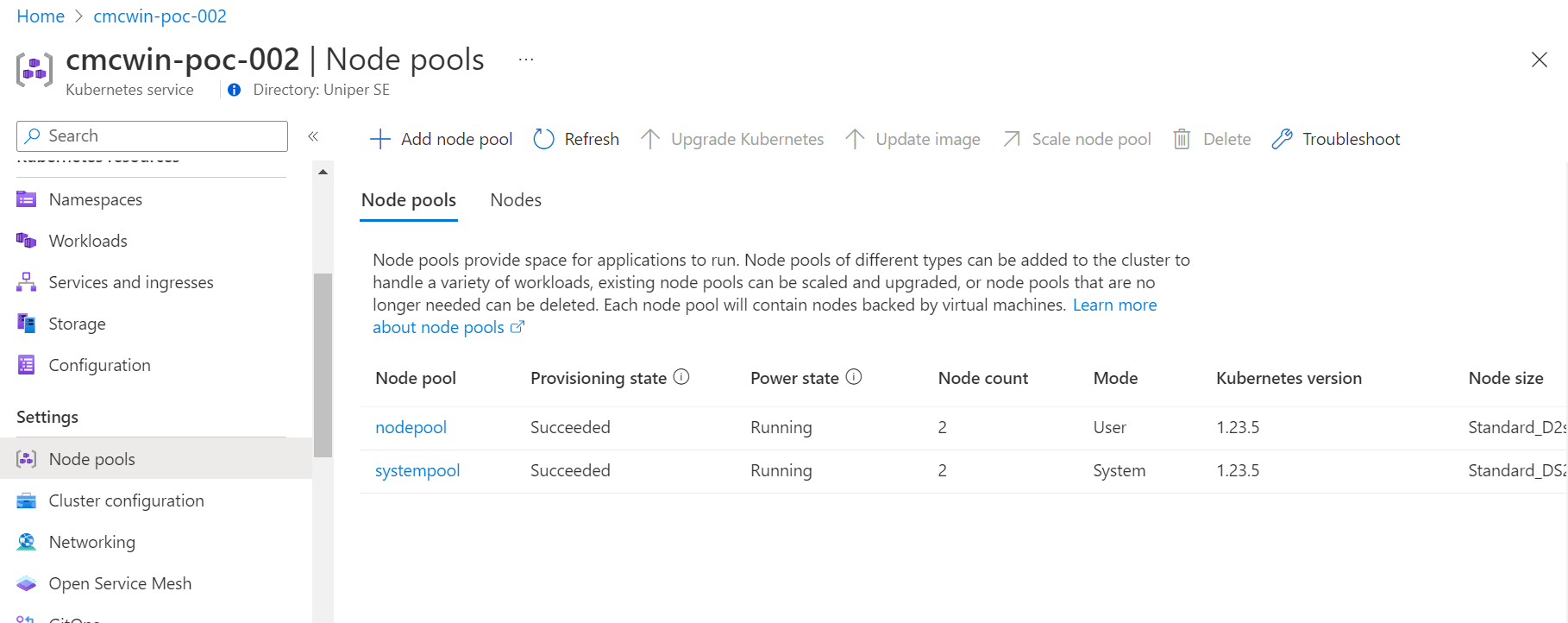


Validate the pods which are not running and get the Namespace of that POD

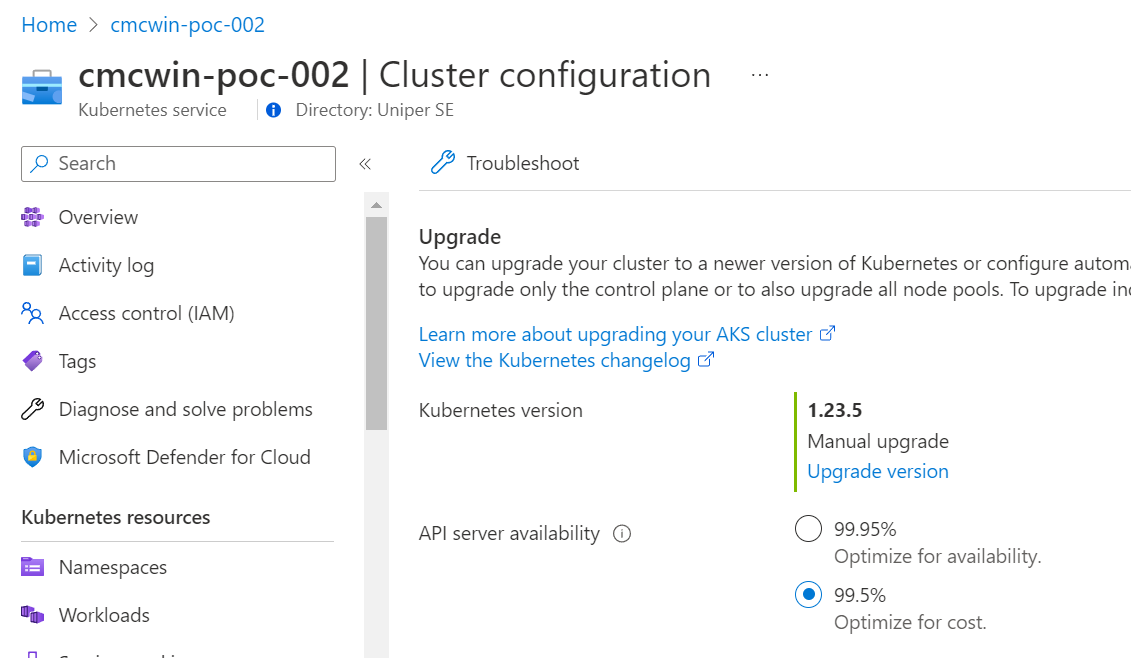
$ kubectl get pods -A --field-selector status.phase!=Running -o=custom-columns=NAME:.metadata.name,STATUS:.status.phase,NAMEDPACE:.metadata.namespace



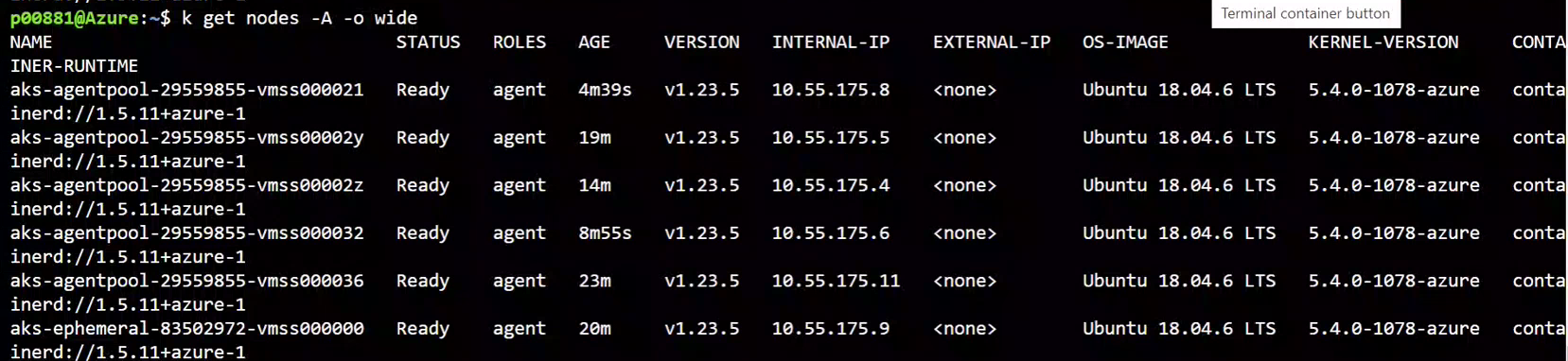
And validate the nodes are running or not in azure portal as well



Upgrade the Kubernetes version as shown below screenshot



After successful of the Kubernetes version upgrade, need to validate the nodes by passing below command.



HaCT team will send email to application teams.

For production environment before start upgradation, HaCT team will RAISE CHANGE REQUEST.

**Responsibility of Application Team after upgrade :**

Once notification mail is sent with release notes (use latestAPI) to application team , team has to verify the application with the updated API .

1.