What is the AKS engine on Azure Stack Hub?

1/21/2022 • 2 minutes to read • Edit Online

You can use the AKS engine command-line tool to deploy and manage a Kubernetes cluster on Azure and Azure Stack Hub. Use the AKS engine to create, upgrade, and scale Azure Resource Manager native clusters. You can use the engine to deploy a cluster in both connected and disconnected environments. This article provides an overview of the AKS engine, supported scenarios for using the engine with Azure Stack Hub, and an introduction to operations such as deploy, upgrade, and scale.

Overview of the AKS engine

The AKS engine provides a command-line tool to bootstrap Kubernetes clusters on Azure and Azure Stack Hub. By using the Azure Resource Manager, the AKS engine helps you create and maintain clusters running on VMs, virtual networks, and other infrastructure-as-a-service (IaaS) resources in Azure Stack Hub.

AKS engine on Azure Stack Hub considerations

Before you use the AKS engine on Azure Stack Hub, it's important to understand the differences between Azure Stack Hub and Azure. This section identifies different features and key considerations when using Azure Stack Hub with the AKS engine to manage your Kubernetes cluster.

For more information on the specifics of the AKS engine on Azure Stack Hub and its differences with respect to Azure see AKS engine on Azure Stack Hub.

For more information on the AKS engine and Azure Stack Hub, see Support policies for AKS engine on Azure Stack Hub.

Install the AKS engine and deploy a Kubernetes cluster

To deploy a Kubernetes cluster with the AKS engine on Azure Stack Hub:

1. Set up the prerequisites for the AKS engine
2. Install the AKS engine to a machine with access to your Azure Stack Hub environment.
   - Install the AKS engine on Windows in Azure Stack Hub
   - Install the AKS engine on Linux in Azure Stack Hub
3. Deploy a Kubernetes cluster with the AKS engine on Azure Stack Hub

Next steps

Set up the prerequisites for the AKS engine

Support policies for AKS engine on Azure Stack Hub

1/21/2022 • 4 minutes to read • Edit Online

This article provides details about technical support policies and limitations for AKS engine on Azure Stack Hub. The article also details Kubernetes Marketplace item, third-party open-source components, and security or patch management.

Self-managed Kubernetes clusters on Azure Stack Hub with AKS engine

Infrastructure as a service (IaaS) cloud components, such as compute or networking components, give users access to low-level controls and customization options. AKS engine allows the user to laydown Kubernetes clusters utilizing these IaaS components transparently, so users can access and affect all aspects of their deployments.

When a cluster is created, the customer defines the Kubernetes masters and worker nodes that AKS engine creates. Customer workloads are executed on these nodes. Customers own and can view or modify the master and worker nodes. Carelessly modified nodes can cause losses of data and workloads and can render the cluster non-functional. Also, AKS engine operations such as Upgrade or Scale will overwrite any out-of-bound changes. For example, if the cluster has static pods, these will not be preserved after an AKS engine upgrade operation.

Because customer cluster nodes execute private code and store sensitive data, Microsoft Support can access them in only a limited way. Microsoft Support can't sign in to, execute commands in, or view logs for these nodes without express customer permission or assistance.

Version support

The AKS engine version support follows the same pattern established by the rest of the Azure Stack Hub support policy, that is support of a version of AKS engine on Azure Stack Hub is based on the n-2 formula. For example, if the latest version of AKS engine is v0.55.0, the set of supported versions are: 0.48.0, 0.51.0, 0.55.0. Also important it to follow the Azure Stack Hub update version and corresponding mapping to AKS engine supported version, this is maintained in the AKS engine release notes.

AKS engine supported areas

Microsoft provides technical support for the following:

- Issues with AKS engine commands: deploy, generate, upgrade, and scale. The tool should be consistent with its behavior on Azure.
- Issues with a Kubernetes cluster deployed following the Overview of the AKS engine.
- Issues with connectivity to other Azure Stack Hub services.
- Issues with Kubernetes API connectivity.
- Issues with Azure Stack Hub Kubernetes provider functionality and connectivity with Azure Resource Manager.
- Issues with the AKS engine-generated configuration of Azure Stack Hub native artifacts such as Load Balancers, Network Security Groups, VNETs, Subnets, Network Interfaces, Route table, Availability sets, Public IP addresses, Storage account, and VM Machines.

- Issues with network performance and latency. ASK engine on Azure Stack Hub can use the kubenet networking plugin and the Azure CNI networking plugin.
- Issues with the AKS base image used by the AKS engine in disconnected deployments.

AKS engine areas not supported

Microsoft does not provide technical support for the following:

- Using the AKS engine on Azure.

- Azure Stack Hub Kubernetes Marketplace item.

- Using the following AKS engine cluster definition options and addons.

  - Not supported addons:
    - Azure AD Pod Identity
    - ACI Connector
    - Blobfuse Flex Volume
    - Cluster Autoscaler
    - Container Monitoring
    - KeyVault Flex Volume
    - NVIDIA Device Plugin
    - Rescheduler
    - SMB Flex Volume

  - Not supported cluster definition options:
    - Under KubernetesConfig:
    - cloudControllerManagerConfig
    - enableDataEncryptionAtRest
    - enableEncryptionWithExternalKms
    - enablePodSecurityPolicy
    - etcdEncryptionKey
    - useInstanceMetadata
    - useManagedIdentity
    - azureCNIURLLinux
    - azureCNIURLWindows
    - Under masterProfile:
    - availabilityZones
    - Under agentPoolProfiles:
    - availabilityZones
    - singlePlacementGroup
    - scaleSetPriority
    - scaleSetEvictionPolicy
    - acceleratedNetworkingEnabled
    - acceleratedNetworkingEnabledWindows

- Kubernetes configuration changes persisted outside the Kubernetes configuration store etcd. For example, static pods running in nodes of the cluster.

- Questions about how to use Kubernetes. For example, Microsoft Support doesn't provide advice on how to create custom ingress controllers, use application workloads, or apply third-party or open-source software packages or tools.

- Third-party open-source projects that aren't provided as part of the Kubernetes cluster deployed by AKS engine. These projects might include Kubeadm, Kubespray, Native, Istio, Helm, Envoy, or others.

- Third-party software. This software can include security scanning tools and networking devices or software.
- Issues about multicloud or multivendor build-outs. For example, Microsoft doesn't support issues related to running a federated multipublic cloud vendor solution.
- Network customizations other than those listed in the AKS engine supported areas section.
- Production environments should only use highly available Kubernetes clusters, that is, clusters deployed with a minimum of three masters and three agent nodes. Anything less cannot be supported in production deployments.

Security issues and patching

If a security flaw is found in one or more components of AKS engine or Kubernetes provider for Azure Stack Hub, Microsoft will make available a patch for customers to patch affected clusters to mitigate the issue. Alternatively, the team will give users upgrade guidance. Notice that patches may require downtime of the cluster. When reboots are required, Microsoft will notify the customers of this requirement. If users don't apply the patches according to Microsoft guidance, their cluster will continue to be vulnerable to the security issue.

Kubernetes marketplace item

Users can download a Kubernetes Marketplace item, which allows users to deploy Kubernetes clusters using the AKS engine indirectly through a template in the Azure Stack Hub user portal. This makes it simpler than using the AKS engine directly. Kubernetes Marketplace item is a useful tool to quickly set up clusters for demonstrations, testing, and development. It is not intended for production, so it is not included in the set of items supported by Microsoft.

Preview features

For features and functionality that requires extended testing and user feedback, Microsoft releases new preview features or features behind a feature flag. Consider these features as prerelease or beta features. Preview features or feature-flag features aren't meant for production. Ongoing functionality changes and behavior, bug fixes, and other changes can result in unstable clusters and downtime. These features are not supported by Microsoft.

Next steps

- Read about the The AKS engine on Azure Stack Hub

Release notes for the AKS engine on Azure Stack Hub

1/21/2022 • 31 minutes to read • Edit Online

Applies to version v0.67.3 of the AKS engine.

This article describes the contents of the Azure Kubernetes Service (AKS) engine on Azure Stack Hub update. The update includes improvements and fixes for the latest release of AKS engine targeted to the Azure Stack Hub platform. Notice that this isn't intended to document the release information for the AKS engine for global Azure.

Update planning

The AKS engine upgrade command fully automates the upgrade process of your cluster, it takes care of virtual machines (VMs), networking, storage, Kubernetes, and orchestration tasks. Before applying the update, make sure to review the release note information.

Upgrade considerations

- Are you using the correct marketplace items, AKS Base Ubuntu 16.04-LTS or 18.04 Image Distro or AKS Base Windows Server for your version of the AKS engine? You can find the versions in the section "Download new images and AKS engine".
- Are you using the correct cluster specification (`apimodel.json`) and resource group for the target cluster? When you originally deployed the cluster, this file was generated in your output directory. See the deploy command parameters Deploy a Kubernetes cluster.
- Are you using a reliable machine to run the AKS engine and from which you are performing upgrade operations?
- If you are updating an operational cluster with active workloads, you can apply the upgrade without affecting them, assuming the cluster is under normal load. However, you should have a backup cluster in case there is a need to redirect users to it. A backup cluster is highly recommended.
- If possible, run the command from a VM within the Azure Stack Hub environment to decrease the network hops and potential connectivity failures.
- Make sure that your subscription has enough quota for the entire process. The process allocates new VMs during the process. The resulting number of VMs would be the same as the original, but plan for a couple more VMs to be created during the process.
- No system updates or scheduled tasks are planned.
- Set up a staged upgrade on a cluster that's configured with the same values as the production cluster and test the upgrade there before doing so in your production cluster.

Use the upgrade command

You will be required to use the AKS engine upgrade command as described in the following article Upgrade a Kubernetes cluster on Azure Stack Hub.

Upgrade interruptions

Sometimes unexpected factors interrupt the upgrade of the cluster. An interruption can occur when the AKS engine reports an error or something happens to the AKS engine execution process. Examine the cause of the interruption, address it, and submit again the same upgrade command to continue the upgrade process. The **upgrade** command is idempotent and should resume the upgrade of the cluster once resubmitted the command. Normally, interruptions increase the time to complete the update, but should not affect the completion of it.

Estimated upgrade time

The estimated time is between 12 to 15 minutes per VM in the cluster. For example, a 20-node cluster may take approximately to five (5) hours to upgrade.

Download new image and AKS engine

Download the new versions of the AKS base Ubuntu Image and AKS engine.

As explained in the AKS engine for Azure Stack Hub documentation, deploying a Kubernetes cluster requires:

- The aks-engine binary (required)
- AKS Base Ubuntu 16.04-LTS Image Distro (deprecated - no longer use, change in API Model to use 18.04 instead)
- AKS Base Ubuntu 18.04-LTS Image Distro (required for Linux agents)
- AKS Base Windows Server Image Distro (required for Windows agents)

New versions of these are available with this update:

- Check the table AKS engine and Azure Stack version mapping for the needed AKS base images.

  Follow the instructions in the following article Add the Azure Kubernetes Services (AKS) engine prerequisites to the Azure Stack Hub Marketplace

- The Kubernetes cluster administrator (normally a tenant user of Azure Stack Hub) will need to download the new aks-engine version 0.67.3. See instructions in the following article, Install the AKS engine on Linux in Azure Stack Hub (or equivalent Windows article). You can follow the same process you used to install the cluster for the first time. The update will overwrite the previous binary. For example, if you used the get-akse.sh script, follow the same steps outlined in this section Install in a connected environment. The same process applies if you're installing in on a Windows system, article Install the AKS engine on Windows in Azure Stack Hub.

Upgrading Kubernetes clusters created with the Ubuntu 16.04 distro

Upgrading Kubernetes clusters created with the Ubuntu 16.04 distro Starting with AKS Engine v0.67.0, the Ubuntu 16.04 distro is not longer a supported option as the OS reached its end-of-life. In order to upgrade a cluster, make sure to set the OS distro to `aks-ubuntu-18.04` in your input API model, the one generated by `aks-engine deploy` and passed as input to `aks-engine upgrade`.

```
    "masterProfile": {
        "distro": "aks-ubuntu-18.04"
    },

    "agentPoolProfiles": [{
        "distro": "aks-ubuntu-18.04"
    }]
```

AKS engine and Azure Stack version mapping

AZURE STACK HUB VERSION	AKS ENGINE VERSION
1910	0.43.0, 0.43.1
2002	0.48.0, 0.51.0

AZURE STACK HUB VERSION	AKS ENGINE VERSION
2005	0.48.0, 0.51.0, 0.55.0, 0.55.4
2008	0.55.4, 0.60.1
2102	0.60.1, 0.63.0, 0.67.0
2108	0.63.0, 0.67.0, 0.67.3

Kubernetes version upgrade path in AKS engine v0.67.3

You can find the current version and upgrade version in the following table for Azure Stack Hub. Don't follow the aks-engine get-versions command since the command one also includes the versions supported in global Azure. The following version and upgrade table applies to the AKS engine cluster in Azure Stack Hub.

CURRENT VERSION	UPGRADE AVAILABLE
1.15.12	1.16.14, 1.16.15
1.16.14	1.16.15, 1.17.17
1.17.11, 1.17.17	1.18.18
1.18.15, 1.18.18	1.19.10
1.19.10	1.19.15, 1.20.11
1.20.6	1.20.11

In the API Model json file, please specify the release and version values under the orchestratorProfile section, for example, if you are planning to deploy Kubernetes 1.17.17, the following two values must be set, (see example kubernetes-azurestack.json):

```
    -   "orchestratorRelease": "1.17",
    -   "orchestratorVersion": "1.17.17"
```

AKS engine and corresponding image mapping

AKS ENGINE	AKS BASE IMAGE	KUBERNETES VERSIONS	API MODEL SAMPLES
v0.43.1	AKS Base Ubuntu 16.04-LTS Image Distro, October 2019 (2019.10.24)	1.15.5, 1.15.4, 1.14.8, 1.14.7	
v0.48.0	AKS Base Ubuntu 16.04-LTS Image Distro, March 2020 (2020.03.19)	1.15.10, 1.14.7	

AKS ENGINE	AKS BASE IMAGE	KUBERNETES VERSIONS	API MODEL SAMPLES
v0.51.0	AKS Base Ubuntu 16.04-LTS Image Distro, May 2020 (2020.05.13), AKS Base Windows Image (17763.1217.200513)	1.15.12, 1.16.8, 1.16.9	Linux, Windows
v0.55.0	AKS Base Ubuntu 16.04-LTS Image Distro, August 2020 (2020.08.24), AKS Base Windows Image (17763.1397.200820)	1.15.12, 1.16.14, 1.17.11	Linux, Windows
v0.55.4	AKS Base Ubuntu 16.04-LTS Image Distro, September 2020 (2020.09.14), AKS Base Windows Image (17763.1397.200820)	1.15.12, 1.16.14, 1.17.11	Linux, Windows
V0.60.1	AKS Base Ubuntu 16.04-LTS Image Distro, January 2021 (2021.01.28), AKS Base Ubuntu 18.04-LTS Image Distro, 2021 Q1 (2021.01.28), AKS Base Windows Image (17763.1697.210129)	1.16.14, 1.16.15, 1.17.17, 1.18.15	Linux, Windows
v0.63.0	AKS Base Ubuntu 18.04-LTS Image Distro, 2021 Q2 (2021.05.24), AKS Base Windows Image (17763.1935.210520)	1.18.18, 1.19.10, 1.20.6	API Model Samples (Linux, Windows)
v0.67.0	AKS Base Ubuntu 18.04-LTS Image Distro, 2021 Q3 (2021.09.27), AKS Base Windows Image (17763.2213.210927)	1.19.15, 1.20.11	API Model Samples (Linux, Windows)
v0.67.3	AKS Base Ubuntu 18.04-LTS Image Distro, 2021 Q3 (2021.09.27), AKS Base Windows Image (17763.2213.210927)	1.19.15, 1.20.11	API Model Samples (Linux, Windows)

What's new

If you are interested in participating in a private preview, you can request preview access.

New features include:

- Support for Kubernetes 1.19.15 and 1.20.11
- See list of bug fixes.

Known issues

- Deploying multiple Kubernetes services in parallel inside a single cluster may lead to an error in the basic load balancer configuration. We recommend deploying one service at the time.

- Since the aks-engine tool is a share source code repository across Azure and Azure Stack Hub. Examining the many release notes and Pull Requests will lead you to believe that the tool supports other versions of Kubernetes and OS platform beyond the listed above, ignore them and use the version table above as the official guide for this update.
- AKS Engine v0.67.0 uses the wrong Windows Image when deploying Windows clusters, users should use v0.67.3 to solve this issue.

Reference

This is the complete set of release notes for Azure and Azure Stack Hub combined:

- https://github.com/Azure/aks-engine/releases/tag/v0.64.0
- https://github.com/Azure/aks-engine/releases/tag/v0.65.0
- https://github.com/Azure/aks-engine/releases/tag/v0.65.1
- https://github.com/Azure/aks-engine/releases/tag/v0.66.0
- https://github.com/Azure/aks-engine/releases/tag/v0.66.1
- https://github.com/Azure/aks-engine/releases/tag/v0.67.0

Applies to version v0.67.0 of the AKS engine.

This article describes the contents of the Azure Kubernetes Service (AKS) engine on Azure Stack Hub update. The update includes improvements and fixes for the latest release of AKS engine targeted to the Azure Stack Hub platform. Notice that this isn't intended to document the release information for the AKS engine for global Azure.

Update planning

The AKS engine upgrade command fully automates the upgrade process of your cluster, it takes care of virtual machines (VMs), networking, storage, Kubernetes, and orchestration tasks. Before applying the update, make sure to review the release note information.

Upgrade considerations

- Are you using the correct marketplace items, AKS Base Ubuntu 16.04-LTS or 18.04 Image Distro or AKS Base Windows Server for your version of the AKS engine? You can find the versions in the section "Download new images and AKS engine".
- Are you using the correct cluster specification (`apimodel.json`) and resource group for the target cluster? When you originally deployed the cluster, this file was generated in your output directory. See the deploy command parameters Deploy a Kubernetes cluster.
- Are you using a reliable machine to run the AKS engine and from which you are performing upgrade operations?
- If you are updating an operational cluster with active workloads, you can apply the upgrade without affecting them, assuming the cluster is under normal load. However, you should have a backup cluster in case there is a need to redirect users to it. A backup cluster is highly recommended.
- If possible, run the command from a VM within the Azure Stack Hub environment to decrease the network hops and potential connectivity failures.
- Make sure that your subscription has enough quota for the entire process. The process allocates new VMs during the process. The resulting number of VMs would be the same as the original, but plan for a couple more VMs to be created during the process.
- No system updates or scheduled tasks are planned.
- Set up a staged upgrade on a cluster that's configured with the same values as the production cluster and test the upgrade there before doing so in your production cluster.

Use the upgrade command

You will be required to use the AKS engine upgrade command as described in the following article Upgrade a Kubernetes cluster on Azure Stack Hub.

Upgrade interruptions

Sometimes unexpected factors interrupt the upgrade of the cluster. An interruption can occur when the AKS engine reports an error or something happens to the AKS engine execution process. Examine the cause of the interruption, address it, and submit again the same upgrade command to continue the upgrade process. The `upgrade` command is idempotent and should resume the upgrade of the cluster once resubmitted the command. Normally, interruptions increase the time to complete the update, but should not affect the completion of it.

Estimated upgrade time

The estimated time is between 12 to 15 minutes per VM in the cluster. For example, a 20-node cluster may take approximately to five (5) hours to upgrade.

Download new image and AKS engine

Download the new versions of the AKS base Ubuntu Image and AKS engine.

As explained in the AKS engine for Azure Stack Hub documentation, deploying a Kubernetes cluster requires:

- The aks-engine binary (required)
- AKS Base Ubuntu 16.04-LTS Image Distro (deprecated - no longer use, change in API Model to use 18.04 instead)
- AKS Base Ubuntu 18.04-LTS Image Distro (required for Linux agents)
- AKS Base Windows Server Image Distro (required for Windows agents)

New versions of these are available with this update:

- Check the table AKS engine and Azure Stack version mapping for the needed AKS base images.

  Follow the instructions in the following article Add the Azure Kubernetes Services (AKS) engine prerequisites to the Azure Stack Hub Marketplace

- The Kubernetes cluster administrator (normally a tenant user of Azure Stack Hub) will need to download the new aks-engine version 0.67.0. See instructions in the following article, Install the AKS engine on Linux in Azure Stack Hub (or equivalent Windows article). You can follow the same process you used to install the cluster for the first time. The update will overwrite the previous binary. For example, if you used the get-akse.sh script, follow the same steps outlined in this section Install in a connected environment. The same process applies if you're installing in on a Windows system, article Install the AKS engine on Windows in Azure Stack Hub.

Upgrading Kubernetes clusters created with the Ubuntu 16.04 distro

Upgrading Kubernetes clusters created with the Ubuntu 16.04 distro Starting with AKS Engine v0.67.0, the Ubuntu 16.04 distro is not longer a supported option as the OS reached its end-of-life. In order to upgrade a cluster, make sure to set the OS distro to `aks-ubuntu-18.04` in your input API model, the one generated by `aks-engine deploy` and passed as input to `aks-engine upgrade`.

```
    "masterProfile": {
        "distro": "aks-ubuntu-18.04"
    },

    "agentPoolProfiles": [{
        "distro": "aks-ubuntu-18.04"
    }]
```

AKS engine and Azure Stack version mapping

AZURE STACK HUB VERSION	AKS ENGINE VERSION
1910	0.43.0, 0.43.1
2002	0.48.0, 0.51.0
2005	0.48.0, 0.51.0, 0.55.0, 0.55.4
2008	0.55.4, 0.60.1
2102	0.60.1, 0.63.0, 0.67.0
2108	0.63.0, 0.67.0

Kubernetes version upgrade path in AKS engine v0.67.0

You can find the current version and upgrade version in the following table for Azure Stack Hub. Don't follow the aks-engine get-versions command since the command one also includes the versions supported in global Azure. The following version and upgrade table applies to the AKS engine cluster in Azure Stack Hub.

CURRENT VERSION	UPGRADE AVAILABLE
1.15.12	1.16.14, 1.16.15
1.16.14	1.16.15, 1.17.17
1.17.11, 1.17.17	1.18.18
1.18.15, 1.18.18	1.19.10
1.19.10	1.19.15, 1.20.11
1.20.6	1.20.11

In the API Model json file, please specify the release and version values under the orchestratorProfile section, for example, if you are planning to deploy Kubernetes 1.17.17, the following two values must be set, (see example kubernetes-azurestack.json):

```
    -   "orchestratorRelease": "1.17",
    -   "orchestratorVersion": "1.17.17"
```

AKS engine and corresponding image mapping

AKS ENGINE	AKS BASE IMAGE	KUBERNETES VERSIONS	API MODEL SAMPLES
v0.43.1	AKS Base Ubuntu 16.04-LTS Image Distro, October 2019 (2019.10.24)	1.15.5, 1.15.4, 1.14.8, 1.14.7	
v0.48.0	AKS Base Ubuntu 16.04-LTS Image Distro, March 2020 (2020.03.19)	1.15.10, 1.14.7	
v0.51.0	AKS Base Ubuntu 16.04-LTS Image Distro, May 2020 (2020.05.13), AKS Base Windows Image (17763.1217.200513)	1.15.12, 1.16.8, 1.16.9	Linux, Windows
v0.55.0	AKS Base Ubuntu 16.04-LTS Image Distro, August 2020 (2020.08.24), AKS Base Windows Image (17763.1397.200820)	1.15.12, 1.16.14, 1.17.11	Linux, Windows
v0.55.4	AKS Base Ubuntu 16.04-LTS Image Distro, September 2020 (2020.09.14), AKS Base Windows Image (17763.1397.200820)	1.15.12, 1.16.14, 1.17.11	Linux, Windows
V0.60.1	AKS Base Ubuntu 16.04-LTS Image Distro, January 2021 (2021.01.28), AKS Base Ubuntu 18.04-LTS Image Distro, 2021 Q1 (2021.01.28), AKS Base Windows Image (17763.1697.210129)	1.16.14, 1.16.15, 1.17.17, 1.18.15	Linux, Windows
v0.63.0	AKS Base Ubuntu 18.04-LTS Image Distro, 2021 Q2 (2021.05.24), AKS Base Windows Image (17763.1935.210520)	1.18.18, 1.19.10, 1.20.6	API Model Samples (Linux, Windows)
v0.67.0	AKS Base Ubuntu 18.04-LTS Image Distro, 2021 Q3 (2021.09.27), AKS Base Windows Image (17763.2213.210927)	1.19.15, 1.20.11	API Model Samples (Linux, Windows)

What's new

If you are interested in participating in a private preview, you can request preview access.

New features include:

- Support for Kubernetes 1.19.15 and 1.20.11

Known issues

- Deploying multiple Kubernetes services in parallel inside a single cluster may lead to an error in the basic load balancer configuration. We recommend deploying one service at the time.
- Since the aks-engine tool is a share source code repository across Azure and Azure Stack Hub. Examining the many release notes and Pull Requests will lead you to believe that the tool supports other versions of Kubernetes and OS platform beyond the listed above, ignore them and use the version table above as the official guide for this update.

Reference

This is the complete set of release notes for Azure and Azure Stack Hub combined:

- https://github.com/Azure/aks-engine/releases/tag/v0.64.0
- https://github.com/Azure/aks-engine/releases/tag/v0.65.0
- https://github.com/Azure/aks-engine/releases/tag/v0.65.1
- https://github.com/Azure/aks-engine/releases/tag/v0.66.0
- https://github.com/Azure/aks-engine/releases/tag/v0.66.1
- https://github.com/Azure/aks-engine/releases/tag/v0.67.0

Applies to version v0.60.1 of the AKS engine.

This article describes the contents of the Azure Kubernetes Service (AKS) engine on Azure Stack Hub update. The update includes improvements and fixes for the latest release of AKS engine targeted to the Azure Stack Hub platform. Notice that this isn't intended to document the release information for the AKS engine for global Azure.

Update planning

The AKS engine upgrade command fully automates the upgrade process of your cluster, it takes care of virtual machines (VMs), networking, storage, Kubernetes, and orchestration tasks. Before applying the update, make sure to review the release note information.

Upgrade considerations

- Are you using the correct marketplace items, AKS Base Ubuntu 16.04-LTS or 18.04 Image Distro or AKS Base Windows Server for your version of the AKS engine? You can find the versions in the section "Download new images and AKS engine".
- Are you using the correct cluster specification (`apimodel.json`) and resource group for the target cluster? When you originally deployed the cluster, this file was generated in your output directory. See the deploy command parameters Deploy a Kubernetes cluster.
- Are you using a reliable machine to run the AKS engine and from which you are performing upgrade operations?
- If you are updating an operational cluster with active workloads, you can apply the upgrade without affecting them, assuming the cluster is under normal load. However, you should have a backup cluster in case there is a need to redirect users to it. A backup cluster is highly recommended.
- If possible, run the command from a VM within the Azure Stack Hub environment to decrease the network hops and potential connectivity failures.
- Make sure that your subscription has enough quota for the entire process. The process allocates new VMs during the process. The resulting number of VMs would be the same as the original, but plan for a couple more VMs to be created during the process.
- No system updates or scheduled tasks are planned.
- Set up a staged upgrade on a cluster that's configured with the same values as the production cluster and

test the upgrade there before doing so in your production cluster.

Use the upgrade command

You will be required to use the AKS engine upgrade command as described in the following article Upgrade a Kubernetes cluster on Azure Stack Hub.

Upgrade interruptions

Sometimes unexpected factors interrupt the upgrade of the cluster. An interruption can occur when the AKS engine reports an error or something happens to the AKS engine execution process. Examine the cause of the interruption, address it, and submit again the same upgrade command to continue the upgrade process. The `upgrade` command is idempotent and should resume the upgrade of the cluster once resubmitted the command. Normally, interruptions increase the time to complete the update, but should not affect the completion of it.

Estimated upgrade time

The estimated time is between 12 to 15 minutes per VM in the cluster. For example, a 20-node cluster may take approximately to five (5) hours to upgrade.

Download new image and AKS engine

Download the new versions of the AKS base Ubuntu Image and AKS engine.

As explained in the AKS engine for Azure Stack Hub documentation, deploying a Kubernetes cluster requires:

- The aks-engine binary (required)
- AKS Base Ubuntu 16.04-LTS Image Distro (required)
- AKS Base Ubuntu 18.04-LTS Image Distro (optional)
- AKS Base Windows Server Image Distro (optional)

New versions of these are available with this update:

- The Azure Stack Hub operator will need to download the new AKS Base Images into the stamp marketplace:

  - AKS Base Ubuntu 16.04-LTS Image Distro, January 2021 (2021.01.28)

  - AKS Base Ubuntu 18.04-LTS Image Distro, 2021 Q1 (2021.01.28),

  - AKS Base Windows Image (17763.1697.210129)

    Follow the instructions in the following article Add the Azure Kubernetes Services (AKS) engine prerequisites to the Azure Stack Hub Marketplace

- The Kubernetes cluster administrator (normally a tenant user of Azure Stack Hub) will need to download the new aks-engine version 0.60.1. See instructions in the following article, Install the AKS engine on Linux in Azure Stack Hub (or equivalent Windows article). You can follow the same process you used to install the cluster for the first time. The update will overwrite the previous binary. For example, if you used the get-akse.sh script, follow the same steps outlined in this section Install in a connected environment. The same process applies if you're installing in on a Windows system, article Install the AKS engine on Windows in Azure Stack Hub.

AKS engine and Azure Stack version mapping

AZURE STACK HUB VERSION	AKS ENGINE VERSION
1910	0.43.0, 0.43.1

AZURE STACK HUB VERSION	AKS ENGINE VERSION
2002	0.48.0, 0.51.0
2005	0.48.0, 0.51.0, 0.55.0, 0.55.4
2008	0.55.4, 0.60.1

Kubernetes version upgrade path in AKS engine v0.60.1

You can find the current version and upgrade version in the following table for Azure Stack Hub. Don't follow the aks-engine get-versions command since the command one also includes the versions supported in global Azure. The following version and upgrade table applies to the AKS engine cluster in Azure Stack Hub.

CURRENT VERSION	UPGRADE AVAILABLE
1.15.12	1.16.14, 1.16.15
1.16.14	1.16.15, 1.17.17
1.17.11	1.17.17, 1.18.15
1.17.17	1.18.15

In the API Model json file, please specify the release and version values under the orchestratorProfile section, for example, if you are planning to deploy Kubernetes 1.17.17, the following two values must be set, (see example kubernetes-azurestack.json):

```
    -   "orchestratorRelease": "1.17",
    -   "orchestratorVersion": "1.17.17"
```

AKS engine and corresponding image mapping

AKS ENGINE	AKS BASE IMAGE	KUBERNETES VERSIONS	API MODEL SAMPLES
v0.43.1	AKS Base Ubuntu 16.04-LTS Image Distro, October 2019 (2019.10.24)	1.15.5, 1.15.4, 1.14.8, 1.14.7	
v0.48.0	AKS Base Ubuntu 16.04-LTS Image Distro, March 2020 (2020.03.19)	1.15.10, 1.14.7	
v0.51.0	AKS Base Ubuntu 16.04-LTS Image Distro, May 2020 (2020.05.13), AKS Base Windows Image (17763.1217.200513)	1.15.12, 1.16.8, 1.16.9	Linux, Windows

AKS ENGINE	AKS BASE IMAGE	KUBERNETES VERSIONS	API MODEL SAMPLES
v0.55.0	AKS Base Ubuntu 16.04-LTS Image Distro, August 2020 (2020.08.24), AKS Base Windows Image (17763.1397.200820)	1.15.12, 1.16.14, 1.17.11	Linux, Windows
v0.55.4	AKS Base Ubuntu 16.04-LTS Image Distro, September 2020 (2020.09.14), AKS Base Windows Image (17763.1397.200820)	1.15.12, 1.16.14, 1.17.11	Linux, Windows
V0.60.1	AKS Base Ubuntu 16.04-LTS Image Distro, January 2021 (2021.01.28), AKS Base Ubuntu 18.04-LTS Image Distro, 2021 Q1 (2021.01.28), AKS Base Windows Image (17763.1697.210129)	1.16.14, 1.16.15, 1.17.17, 1.18.15	Linux, Windows

What's new

If you are interested in participating in a private preview, you can request preview access.

New features include:

- General Availability of Ubuntu 18.04
- Certificate Rotation Public Preview #4214
- T4 Nvidia GPU Private Preview #4259
- Azure Active Directory integration private preview
- CSI Driver for Azure Blobs Private Preview #712
- CSI Driver Azure Disks Public Preview #712
- CSI Driver NFS Public Preview #712
- Support for Kubernetes 1. 17.17 #4188 and 1.18.15 #4187

Known issues

- Deploying multiple Kubernetes services in parallel inside a single cluster may lead to an error in the basic load balancer configuration. We recommend deploying one service at the time.
- Since the aks-engine tool is a share source code repository across Azure and Azure Stack Hub. Examining the many release notes and Pull Requests will lead you to believe that the tool supports other versions of Kubernetes and OS platform beyond the listed above, ignore them and use the version table above as the official guide for this update.

Reference

This is the complete set of release notes for Azure and Azure Stack Hub combined:

- https://github.com/Azure/aks-engine/releases/tag/v0.56.0
- https://github.com/Azure/aks-engine/releases/tag/v0.56.1
- https://github.com/Azure/aks-engine/releases/tag/v0.60.0
- https://github.com/Azure/aks-engine/releases/tag/v0.60.1

Applies to version v0.55.4 of the AKS engine.

This article describes the contents of the Azure Kubernetes Service (AKS) engine on Azure Stack Hub update. The update includes improvements and fixes for the latest release of AKS engine targeted to the Azure Stack Hub platform. Notice that this isn't intended to document the release information for the AKS engine for global Azure.

Update planning

The AKS engine upgrade command fully automates the upgrade process of your cluster, it takes care of virtual machines (VMs), networking, storage, Kubernetes, and orchestration tasks. Before applying the update, make sure to review the release note information.

Upgrade considerations

- Are you using the correct marketplace item, AKS Base Ubuntu 16.04-LTS Image Distro for your version of the AKS engine? You can find the versions in the section "Download new image and AKS engine".

- Are you using the correct cluster specification (`apimodel.json`) and resource group for the target cluster? When you originally deployed the cluster, this file was generated in your output directory. See the `deploy` command parameters Deploy a Kubernetes cluster.

- Are you using a reliable machine to run the AKS engine and from which you are performing upgrade operations?

- If you are updating an operational cluster with active workloads you can apply the upgrade without affecting them, assuming the cluster is under normal load. However, you should have a backup cluster in case there is a need to redirect users to it. A backup cluster is highly recommended.

- If possible, run the command from a VM within the Azure Stack Hub environment to decrease the network hops and potential connectivity failures.

- Make sure that your subscription has enough quota for the entire process. The process allocates new VMs during the process. The resulting number of VMs would be the same as the original, but plan for a couple more VMs to be created during the process.

- No system updates or scheduled tasks are planned.

- Set up a staged upgrade on a cluster that's configured with the same values as the production cluster and test the upgrade there before doing so in your production cluster.

Use the upgrade command

You will be required to use the AKS engine `upgrade` command as described in the following article Upgrade a Kubernetes cluster on Azure Stack Hub.

Upgrade interruptions

Sometimes unexpected factors interrupt the upgrade of the cluster. An interruption can occur when the AKS engine reports an error or something happens to the AKS engine execution process. Examine the cause of the interruption, address it, and submit again the same upgrade command to continue the upgrade process. The **upgrade** command is idempotent and should resume the upgrade of the cluster once resubmitted the command. Normally, interruptions increase the time to complete the update, but should not affect the completion of it.

Estimated upgrade time

The estimated time is between 12 to 15 minutes per VM in the cluster. For example, a 20-node cluster may take approximately to five (5) hours to upgrade.

Download new image and AKS engine

Download the new versions of the AKS base Ubuntu Image and AKS engine.

As explained in the AKS engine for Azure Stack Hub documentation, deploying a Kubernetes cluster requires two main components:

- The aks-engine binary
- AKS Base Ubuntu 16.04-LTS Image Distro

New versions of these are available with this update:

- The Azure Stack Hub operator will need to download a new AKS base Ubuntu Image into the stamp marketplace:
  - Name: AKS Base Ubuntu 16.04-LTS Image Distro, September 2020 (2020.09.14)
  - Version: 2020.09.14
  - Follow the instructions in the following article Add the Azure Kubernetes Services (AKS) engine prerequisites to the Azure Stack Hub Marketplace
- The Kubernetes cluster administrator will need to download the new aks-engine version 0.51.0. See instructions in the following article, Install the AKS engine on Linux in Azure Stack Hub. You can follow the same process you use to install the cluster for the first time. The update will overwrite the previous binary. For example, if you used the get-akse.sh script, follow the same steps outlined in this section Install in a connected environment. The same process applies if you're installing in on a Windows system, article Install the AKS engine on Windows in Azure Stack Hub.

AKS engine and Azure Stack version mapping

AZURE STACK HUB VERSION	AKS ENGINE VERSION
1910	0.43.0, 0.43.1
2002	0.48.0, 0.51.0
2005	0.48.0, 0.51.0, 0.55.0, 0.55.4

Kubernetes version upgrade path in AKS engine v0.55.4

You can find the current version and upgrade version in the following table for Azure Stack Hub. Don't follow the aks-engine get-versions command since the command one also includes the versions supported in global Azure. The following version and upgrade table applies to the AKS engine cluster in Azure Stack Hub.

CURRENT VERSION	UPGRADE AVAILABLE
1.15.10	1.15.12
1.15.12, 1.16.8, 1.16.9	1.16.14
1.16.8, 1.16.9, 1.16.14	1.17.11

In the API Model json file, please specify the release and version values under the `orchestratorProfile` section, for example, if you are planning to deploy Kubernetes 1.16.14, the following two values must be set, (see example kubernetes-azurestack.json):

```
    -   "orchestratorRelease": "1.16",
    -   "orchestratorVersion": "1.16.14"
```

AKS engine and corresponding image mapping

KUBERNETES VERSIONS	NOTES
1.15.5, 1.15.4, 1.14.8, 1.14.7	
1.15.10, 1.14.7	
1.15.12, 1.16.8, 1.16.9	API Model Samples (Linux, Windows)
1.15.12, 1.16.14, 1.17.11	API Model Samples (Linux, Windows)
1.15.12, 1.16.14, 1.17.11	API Model Samples (Linux, Windows)

What's new

- Update Azure Stack's Linux VHD to 2020.09.14 #3828
- Adds support for K8s v1.17.11 on Azure Stack #3702
- Adds support for K8s v1.16.14 on Azure Stack #3704
- Linux VHD update to 2020.09.14 #3750
- Windows VHD update to August #3730
- Upgrades Kubernetes metrics-server to v0.3.7 #3669
- Upgrades docker version to fix log rotation issue #3693
- Upgrades CoreDNS to v1.7.0 #3608
- Use moby 19.03.x packages #3549
- Fixes to azure-cni update strategy #3571

Known issues

- Deploying multiple Kubernetes services in parallel inside a single cluster may lead to an error in the basic load balancer configuration. Deploying one service at the time if possible.
- Running aks-engine get-versions will produce information applicable to Azure and Azure Stack Hub, however, there is not explicit way to discern what corresponds to Azure Stack Hub. Do not use this command to figure out what versions are available to upgrade. Use the upgrade reference table described above.
- Since aks-engine tool is a share source code repository across Azure and Azure Stack Hub. Examining the many release notes and Pull Requests will lead you to believe that the tool supports other versions of Kubernetes and OS platform beyond the listed above, ignore them and use the version table above as the official guide for this update.

Reference

This is the complete set of release notes for Azure and Azure Stack Hub combined:

- https://github.com/Azure/aks-engine/releases/tag/v0.51.1
- https://github.com/Azure/aks-engine/releases/tag/v0.52.1

- https://github.com/Azure/aks-engine/releases/tag/v0.53.1
- https://github.com/Azure/aks-engine/releases/tag/v0.54.1
- https://github.com/Azure/aks-engine/releases/tag/v0.55.0
- https://github.com/Azure/aks-engine/releases/tag/v0.55.4

Applies to version 0.48.0 or earlier of the AKS engine.

This article describes the contents of the Azure Kubernetes Service (AKS) engine on Azure Stack Hub update. The update includes improvements and fixes for the latest release of AKS engine targeted to the Azure Stack Hub platform. Notice that this isn't intended to document the release information for the AKS engine for global Azure.

Update planning

The AKS engine upgrade command fully automates the upgrade process of your cluster, it takes care of virtual machines (VMs), networking, storage, Kubernetes, and orchestration tasks. Before applying the update, make sure to review the information in this article.

Upgrade considerations

- Are you using the correct marketplace item, AKS Base Ubuntu 16.04-LTS Image Distro for your version of the AKS engine? You can find the versions in the section Download the new AKS base Ubuntu Image and AKS engine versions.
- Are you using the correct cluster specification (apimodel.json) and resource group for the target cluster? When you originally deployed the cluster, this file was generated in your output directory. See the "deploy" command parameters Deploy a Kubernetes cluster.
- Are you using a reliable machine to run the AKS engine and from which you are performing upgrade operations?
- If you are updating an operational cluster with active workloads you can apply the upgrade without affecting them, assuming the cluster is under normal load. However, you should have a backup cluster in case there is a need to redirect users to it.
- If possible, run the command from a VM within the Azure Stack Hub environment to decrease the network hops and potential connectivity failures.
- Make sure that your subscription has enough quota for the entire process. The process allocates new VMs during the process. The resulting number of VMs would be the same as the original, but plan for a couple more VMs to be created during the process.
- No system updates or scheduled tasks are planned.
- Set up a staged upgrade on a cluster that's configured with the same values as the production cluster and test the upgrade there before doing so in your production cluster.

Use the upgrade command

You will be required to use the AKS engine "upgrade" command as described in the following article Upgrade a Kubernetes cluster on Azure Stack Hub.

Upgrade interruptions

Sometimes unexpected factors interrupt the upgrade of the cluster. An interruption can occur when the AKS engine reports an error or something happens to the AKS engine execution process. Examine the cause of the interruption, address it, and submit again the same upgrade command to continue the upgrade process. The `upgrade` command is idempotent and should resume the upgrade of the cluster once resubmitted the command. Normally, interruptions increase the time to complete the update, but should not affect the completion of it.

Estimated upgrade time

The estimated time is between 12 to 15 minutes per VM in the cluster. For example, a 20-node cluster may take approximately to five (5) hours to upgrade.

Download new image and AKS engine

Download the new versions of the AKS base Ubuntu Image and AKS engine.

As explained in the AKS engine for Azure Stack Hub documentation, deploying a Kubernetes cluster requires two main components:

- The aks-engine binary
- AKS Base Ubuntu 16.04-LTS Image Distro

New versions of these are available with this update:

- The Azure Stack Hub operator will need to download a new AKS base Ubuntu Image:
    - Name: `AKS Base Ubuntu 16.04-LTS Image Distro, March 2020`
    - Version: `2020.03.19`
    - Follow the instructions in the following article Add the Azure Kubernetes Services (AKS) engine prerequisites to the Azure Stack Hub Marketplace
- The Kubernetes cluster administrator will need to download the new aks-engine version 0.48.0. See instructions in this the following article, Install the AKS engine on Linux in Azure Stack Hub. You can follow the same process you use to install the cluster for the first time. The update will overwrite the previous binary. For example, if you used the `get-akse.sh` script, follow the same steps outlined in the article Install in a connected environment. The same process applies if you're installing in on a Windows system, article Install the AKS engine on Windows in Azure Stack Hub.

Kubernetes version upgrade path

You can find the current version and upgrade version in the following table for Azure Stack Hub. Don't follow the aks-engine `get-versions` command since the command one also includes the versions supported in global Azure. The following version and upgrade table applies to the AKS engine cluster in Azure Stack Hub.

CURRENT VERSION	UPGRADE AVAILABLE
1.14.7	1.15.10
1.14.8	1.15.10
1.15.4	1.15.10
1.15.5	1.15.10

What's new

- Support for Kubernetes version 1.15.10 (#2834). When deploying a new cluster make sure to specify in your api model json file (a.k.s. cluster definition file) the release version number as well as the minor version number. You can find an example: kubernetes-azurestack.json:
    - `"orchestratorRelease": "1.15`,

    - `"orchestratorVersion": "1.15.10"`

> **NOTE**
>
> If the Kubernetes version is not explicitly provided in the API model json file, version `1.15` will be used (#2932) and the orchestratorVersion will default to `1.15.11`, which will result in an error during deployment of the cluster.

- With aks-engine v0.43.1, the default frequency settings for the cloud provider to perform its control loop and other tasks do not work well with Azure Stack Hub Resource Manager threshold limits for incoming requests. This update changes defaults for Azure Stack Hub to reduce the retry load to Azure Stack Hub Resource Manager (#2861).
- New verification step in aks-engine will result in either execution stopping or displaying warnings if api model json file contains properties not supported by Azure Stack Hub (#2717).
- With a new verification check-in, the aks-engine will validate availability of the version of the AKS base image needed for the version of aks-engine executing (#2342). This will occur after parsing the api model file and before calling the Azure Stack Hub Resource Manager.
- New aks-engine option "--control-plane-only" in the "upgrade" command allows the user to upgrade operations to target only the master Virtual Machines (#2635).
- Updates to Linux Kernel version 4.15.0-1071-azure for Ubuntu 16.04-LTS. See "Package: linux-image-4.15.0-1071-azure (4.15.0-1071.76) [security]" for details.
- New hyperkube updates to support Kubernetes versions 1.14.8 and 1.15.10.
- Update kubectl to match the version of Kubernetes for the cluster.. This component is available in the Kubernetes cluster master nodes, you can run it by SSH into a master.
- Updates for the Azure Container Monitor add-on with latest February 2020 release (#2850).
- Upgrade of `coredns` to version v1.6.6 (#2555).
- Upgrade `etcd` to version 3.3.18 (#2462).
- Upgrade `moby` to version 3.0.11 (#2887).
- With this release AKS Engine cuts dependency from `k8s.gcr.io` to now use the official `Kubernetes MCR registry @ mcr.microsoft.com` when building its images (#2722).

Known issues

- Deploying multiple Kubernetes services in parallel inside a single cluster may lead to an error in the basic load balancer configuration. Deploying one service at the time if possible.
- Running aks-engine get-versions will produce information applicable to Azure and Azure Stack Hub, however, there is not explicit way to discern what corresponds to Azure Stack Hub. Do not use this command to figure out what versions area available to upgrade. Use the upgrade reference table described above.
- Since aks-engine tool is a share source code repository across Azure and Azure Stack Hub. Examining the many release notes and Pull Requests will lead you to believe that the tool supports other versions of Kubernetes and OS platform beyond the listed above, please ignore them and use the version table above as the official guide for this update.

Reference

Following is a list of some of the bugs fixed as well as the complete set of release notes from 0.44.0 to 0.48.0, notice that the latter list will include Azure and Azure Stack Hub items

Bug fixes

- `userAssignedIdentityId` in windows `azure.json` missing quotes (#2327)
- Addons `update config` is upgrade-only (#2282)
- Bumping timeout for getting management IP on windows nodes (#2284)
- Add 1.0.28 Azure CNI zip file into windows VHD (#2268)
- Correct defaults order for setting IPAddressCount (#2358)
- Update to use single omsagent yaml for all k8s versions to avoid any manual errors and easy maintainability (#2692)

Release notes

This is the complete set of release notes for Azure and Azure Stack Hub combined

- https://github.com/Azure/aks-engine/releases/tag/v0.44.0
- https://github.com/Azure/aks-engine/releases/tag/v0.44.1
- https://github.com/Azure/aks-engine/releases/tag/v0.44.2
- https://github.com/Azure/aks-engine/releases/tag/v0.45.0
- https://github.com/Azure/aks-engine/releases/tag/v0.46.0
- https://github.com/Azure/aks-engine/releases/tag/v0.46.1
- https://github.com/Azure/aks-engine/releases/tag/v0.46.2
- https://github.com/Azure/aks-engine/releases/tag/v0.46.3
- https://github.com/Azure/aks-engine/releases/tag/v0.47.0
- https://github.com/Azure/aks-engine/releases/tag/v0.48.0

Next steps

- Read about the The AKS engine on Azure Stack Hub

Set up the prerequisites for the AKS engine on Azure Stack Hub

1/21/2022 • 4 minutes to read • Edit Online

You can install the AKS engine on a virtual machines (VMs) in your environment, or any client machine with access to your Azure Stack Hub Resource Manager endpoint. You will need the following things in place before you run the engine: an AKS Base Ubuntu server and Linux custom script extension available in your subscription, a service principal identity that has been assigned to a contributor role, and a private/public key pair for SSH access to your Ubuntu server. In addition, if you are using the Azure Stack Development Kit, you will need to have your machine trust the appropriate certificates.

If you have your prerequisites, you can begin to define your cluster.

If you are the cloud operator for Azure Stack Hub and would like to offer the AKS engine, follow the instructions at Add the AKS engine to the Azure Stack Hub Marketplace.

Prerequisites for the AKS engine

To use the AKS engine, you need to have the following resources available. Keep in mind that the AKS engine is meant to be used by tenants of Azure Stack Hub to deploy Kubernetes clusters into their tenant subscription. The only part where involvement of the Azure Stack Hub operator may be required is for downloading Marketplace items and the creation of a service principal identity. You can find details in the following table.

Your cloud operator will need to have the following items in place.

PREREQUISITE	DESCRIPTION	REQUIRED	INSTRUCTIONS
Azure Stack Hub 1910 or greater	The AKS engine requires Azure Stack Hub 1910 or greater.	Required	If you are unsure of your version of Azure Stack Hub, contact your cloud operator.
Linux custom script extension	Linux Custom Script extension 2.0 Offer: Custom Script for Linux 2.0 Version: 2.0.6 (or latest version) Publisher: Microsoft Corp	Required	If you do not have this item in your subscription, contact your cloud operator.
AKS Base images	AKS Base Ubuntu and Windows Image See more information on the version dependency see Matching engine to base image version	Required	If you don't have this item in your subscription, contact your cloud operator. If you are the cloud operator for Azure Stack Hub and would like to offer the AKS engine, follow the instructions at Add the AKS engine to the Azure Stack Hub Marketplace.

PREREQUISITE	DESCRIPTION	REQUIRED	INSTRUCTIONS
Service principal identity (SPN)	An application that needs to deploy or configure resources through Azure Resource Manager, must be represented by a service principal.	Required	You may need to contact your Azure Stack Hub operator for this item. If an Azure Active Directory (Azure AD) Service Principal identity is used, Internet access is required from the VMs in the Kubernetes cluster so that the Service Principal can be authenticated with Azure AD. If there is no internet access, the Kubernetes cluster will not be functional. For instructions see Use an app identity to access resources
(SPN) assigned **Contributor** role	To allow an application to access resources in your subscription using its service principal, you must assign the service principal to a role for a specific resource.	Required	For instructions, see Assign a role

You can set the following items.

PREREQUISITE	DESCRIPTION	REQUIRED	INSTRUCTIONS
Azure Stack Hub subscription	You access offers in your Azure Stack Hub through subscriptions. The offer contains the services that are available to you.	Required	To be able to deploy any tenant workloads in Azure Stack Hub, you need to first get an Azure Stack Hub Subscription.
Resource group	A resource group is a container that holds related resources for an Azure solution. If you don't specify an existing resource group the tool will create one for you.	Optional	Manage Azure Resource Manager resource groups by using the Azure portal
Private Public key	To use an open SSH connection from your development machine to the server VM in your Azure Stack Hub instance that hosts your web app, you need to create a Secure Shell (SSH) public and private key pair.	Required	For instructions on generating a key, see SSH Key Generation.

> **NOTE**
>
> You can also create the prerequisites for the AKS engine with Azure CLI for Azure Stack Hub or Azure Stack Hub PowerShell.

Matching engine to base image version

The AKS engine deploys a customized Ubuntu Server OS to each cluster node image, the **AKS Base Ubuntu and Windows Image Distro**. Any AKS engine version is dependent on a specific image version made available in your Azure Stack Hub by your Azure Stack Hub operator. You can find a table listing the AKS engine versions and corresponding supported Kubernetes versions at Supported Kubernetes Versions. For example, AKS engine version `v0.55.0` depends on version `2020.08.24` of the AKS Base Ubuntu and Windows Image Distro. Ask your Azure Stack Hub operator to download the specific image version from the Azure Marketplace to the Azure Stack Hub Marketplace.

You will trigger and error if the image is not available in your Azure Stack Hub Marketplace. For example, if you're currently using AKS engine version v0.55.0 and AKS Base Ubuntu and Windows Image Distro version `2020.08.24` isn't available, you will see the following error when running the AKS engine:

```
The platform image 'microsoft-aks:aks:aks-ubuntu-1604-202003:2020.08.24' is not available.
Verify that all fields in the storage profile are correct.
```

You can check the current version of your AKS engine by running the following command:

```
$ aks-engine version
Version: v0.55.0
GitCommit: 44a35c00c
GitTreeState: clean
```

Next steps

Deploy the AKS engine on Windows in Azure Stack Hub
Deploy the AKS engine on Linux in Azure Stack Hub

Install the AKS engine on Linux in Azure Stack Hub

1/21/2022 • 3 minutes to read • Edit Online

You can use a Linux machine in your Azure Stack Hub to host the AKS engine in order to deploy and manage a Kubernetes cluster. In this article, we look at preparing the client VM to manage your cluster for both connected and disconnected Azure Stack Hub instances, check the install, and setting up the client VM on the ASDK.

Prepare the client VM

The AKS engine is a command-line tool used to deploy and manage your Kubernetes cluster. You can run the engine on a machine in your Azure Stack Hub. From this machine, you will execute the AKS engine to deploy the IaaS resources and software needed to run your cluster. You can then use the machine running the engine to perform management tasks on your cluster.

When choosing your client machine, consider:

1. If the client machine should be recoverable in case of a disaster.
2. How you will connect to the client machine and how the machine will interact with your cluster.

Install in a connected environment

You can install the client VM to manage your Kubernetes cluster on an Azure Stack Hub connected to the Internet.

1. Create a Linux VM in your Azure Stack Hub. For instructions, see Quickstart: Create a Linux server VM by using the Azure Stack Hub portal.

2. Connect to your VM.

3. Find the version of AKS engine in the AKS engine and Azure Stack version mapping table table. The AKS Base Image must be available in your Azure Stack Hub Marketplace. When running the command, you must specify the version `--version v0.xx.x`. If you don't specify the version, the command will install the latest version, which may need a VHD image that is not available in your marketplace.

   > **NOTE**
   >
   > You can find the mapping of Azure Stack Hub to AKS engine version number in the AKS engine release notes.

4. Run the following command:

   ```
   curl -o get-akse.sh https://raw.githubusercontent.com/Azure/aks-engine/master/scripts/get-akse.sh
   chmod 700 get-akse.sh
   ./get-akse.sh --version v0.xx.x
   ```

   > **NOTE**
   >
   > You can find the mapping of Azure Stack Hub to AKS engine version number in the AKS engine release notes.

> **NOTE**
>
> If you method for installation fails, you can try the steps in the disconnected environment.

Install in a disconnected environment

You can install the client VM to manage your Kubernetes cluster on an Azure Stack Hub disconnected from the Internet.

1. From a machine with access to the Internet, go to GitHub Azure/aks-engine. Download an archive (*.tar.gz) for a Linux machine, for example, `aks-engine-v0.xx.x-linux-amd64.tar.gz`. Find the version of AKS engine in the Supported Kubernetes Versions table.

2. Create a storage account in your Azure Stack Hub instance to upload the archive file (*.tar.gz) with the AKS engine binary. For instructions on using the Azure Storage Explorer, see Azure Storage Explorer with Azure Stack Hub.

3. Create a Linux VM in your Azure Stack Hub. For instructions, see Quickstart: Create a Linux server VM by using the Azure Stack Hub portal.

4. From the Azure Stack Hub storage account blob URL where you uploaded the archive file (*.tar.gz), download the file to your management VM. Extract the archive to the directory `/usr/local/bin`.

5. Connect to your VM.

6. Run the following command:

```
curl -o aks-engine-v0.xx.x-linux-amd64.tar.gz <httpurl/aks-engine-v0.xx.x-linux-amd64.tar.gz>
tar xvzf aks-engine-v0.xx.x-linux-amd64.tar.gz -C /usr/local/bin
```

Verify the installation

Once your client VM is set up, check that you have installed the AKS engine.

1. Connect to your client VM.

2. Run the following command:

```
aks-engine version
```

3. If Azure Resource Manager endpoint is using a self-signed certificate, you need to explicitly add the root certificate to trusted certificate store of the machine. You can find the root certificate in the VM in this directory: /var/lib/waagent/Certificates.pem. Copy the certificate file with the following command:

```
sudo cp /var/lib/waagent/Certificates.pem /usr/local/share/ca-certificates/azurestackca.crt
sudo update-ca-certificates
```

If you are unable to verify that you have installed the AKS engine on your client VM, see Troubleshoot AKS engine install

ASDK installation

You will need to add a certificate when running the client VM for the AKS engine on the ASDK.

When you are using an ASDK your Azure Resource Manager endpoint is using a self-signed certificate, you need explicitly to add this certificate to the machine's trusted certificate store. You can find the ASDK root certificate in any VM you deploy in the ASDK. For example, in an Ubuntu VM you will find it in this directory `/var/lib/waagent/Certificates.pem`.

Copy the certificate file with the following command:

```
sudo cp /var/lib/waagent/Certificates.pem /usr/local/share/ca-certificates/azurestackca.crt

sudo update-ca-certificates
```

Next steps

Deploy a Kubernetes cluster with the AKS engine on Azure Stack Hub

Install the AKS engine on Windows in Azure Stack Hub

1/21/2022 • 4 minutes to read • Edit Online

You can use a Windows machine in your Azure Stack Hub to host the AKS engine in order to deploy and manage a Kubernetes cluster. In this article, we look at preparing the client VM to manage your cluster for both connected and disconnected Azure Stack Hub instances, check the install, and setting up the client VM on the ASDK.

Prepare the client VM

The AKS engine is a command-line tool used to deploy and manage your Kubernetes cluster. You can run the engine on a machine in your Azure Stack Hub. From this machine, you will execute the AKS engine to deploy the IaaS resources and software needed to run your cluster. You can then use the machine running the engine to perform management tasks on your cluster.

When choosing your client machine, consider:

1. If the client machine should be recoverable in case of a disaster.
2. How you will connect to the client machine and how the machine will interact with your cluster.

Install in a connected environment

You can install the client VM to manage your Kubernetes cluster on an Azure Stack Hub connected to the Internet.

1. Create a Windows VM in your Azure Stack Hub. For instructions, see Quickstart: Create a Windows server VM by using the Azure Stack Hub portal.

2. Connect to your VM.

3. Install Chocolatey using the PowerShell instructions..

   According to the Chocolatey website: Chocolatey is a package manager for Windows, like apt-get or yum but for Windows. It was designed to be a decentralized framework for quickly installing applications and tools that you need. It is built on the NuGet infrastructure currently using PowerShell as its focus for delivering packages from the distros to your door, err, computer.

4. Find the version of AKS engine in the AKS engine and Azure Stack version mapping table table. The AKS Base Engine must be available in your Azure Stack Hub Marketplace. When running the command, you must specify the version `--version v0.xx.x`. If you don't specify the version, the command will install the latest version, which may need an VHD image that is not available in your marketplace.

   > **NOTE**
   >
   > You can find the mapping of Azure Stack Hub to AKS engine version number in the AKS engine release notes.

5. Run the following command from an elevated prompt and include the version number:

   ```
   choco install aks-engine --version 0.xx.x -y
   ```

> **NOTE**
>
> If this method for installation fails, you can try the steps in the disconnected environment.

Install in a disconnected environment

You can install the client VM to manage your Kubernetes cluster on an Azure Stack Hub disconnected from the Internet.

1. From a machine with access to the Internet, go to GitHub Azure/aks-engine. Download an archive (*.tar.gz) for a Windows machine, for example, `aks-engine-v0.xx.x-windows-amd64.tar.gz`. Find the version of AKS engine in the Supported Kubernetes Versions table.

2. Create a storage account in your Azure Stack Hub instance to upload the archive file (*.tar.gz) with the AKS engine binary. For instructions on using the Azure Storage Explorer, see Azure Storage Explorer with Azure Stack Hub.

3. Create a Windows VM in your Azure Stack Hub. For instructions, see Quickstart: Create a Windows server VM by using the Azure Stack Hub portal

4. From the Azure Stack Hub storage account blob URL where you uploaded the archive file (*.tar.gz), download the file to your management VM. Extract the archive to a directory that you have access to from your command prompt.

5. Connect to your VM.

6. Install Chocolatey using the PowerShell instructions..

7. Run the following command from an elevated prompt. Include the right version number:

   ```
       choco install aks-engine --version 0.xx.x -y
   ```

Verify the installation

Once your client VM is set up, check that you have installed the AKS engine.

1. Connect to your client VM.

2. Run the following command:

   ```
   aks-engine version
   ```

If you are unable to verify that you have installed the AKS engine on your client VM, see Troubleshoot AKS engine install.

ASDK installation

You will need to add a certificate when running the client VM for the AKS engine on the ASDK on a machine outside of the ASDK. If you're using a Windows VM within the ASDK environment itself, the machine already trusts the ASDK certificate. If your client machine is outside of the ASDK, you need to extract the certificate from the ASDK and add it to your Windows machine.

When you are using an ASDK your Azure Resource Manager endpoint is using a self-signed certificate, you need to explicitly add this certificate to the machine's trusted certificate store. You can find the ASDK root certificate in any VM you deploy in the ASDK.

1. Export the CA root certificate. For instructions, see Export the Azure Stack Hub CA root certificate.
2. Trust the Azure Stack Hub CA root certificate. For instructions, see Trust the Azure Stack Hub CA root certificate.

Next steps

Deploy a Kubernetes cluster with the AKS engine on Azure Stack Hub

Deploy a Kubernetes cluster with the AKS engine on Azure Stack Hub

1/21/2022 • 8 minutes to read • Edit Online

You can deploy a Kubernetes cluster on Azure Stack Hub from a client VM running the AKS engine. In this article, we look at writing a cluster specification, deploying a cluster with the `apimodel.json` file, and checking your cluster by deploying MySQL with Helm.

Define a cluster specification

You can specify a cluster specification in a document file using the JSON format called the API model. The AKS engine uses a cluster specification in the API model to create your cluster.

You can find examples of the API model for your OS and AKS engine version number for recent releases at AKS engine and corresponding image mapping.

1. Find your AKS engine version number, for example, `v.0.63.0`, in the table.
2. In the API Model samples table, select and open the link for your OS.
3. Select **Raw**. You can use the URL in the following instructions.

A URL to the API model may look like:

```
https://raw.githubusercontent.com/Azure/aks-engine/master/examples/azure-stack/kubernetes-azurestack.json
```

For each of the following samples replace `<URL for the API Model>` with the URL.

Update the API model

This section looks at creating an API model for your cluster.

1. Start by using an Azure Stack Hub API Model file for Linux or Windows. From the machine, you installed AKS engine, run:

   ```
   curl -o kubernetes-azurestack.json <URL for the API Model>
   ```

   > **NOTE**
   >
   > If you are disconnected, you can download the file and manually copy it to the disconnected machine where you plan to edit it. You can copy the file to your Linux machine using tools such as PuTTY or WinSCP.

2. To open the API model in an editor, you can use nano:

   ```
   nano ./kubernetes-azurestack.json
   ```

   > **NOTE**
   >
   > If you don't have nano installed, you can install nano on Ubuntu: `sudo apt-get install nano`.

3. In the kubernetes-azurestack.json file, find orchestratorRelease and orchestratorVersion. Select one of the supported Kubernetes versions; you can find the version table in the release notes. Specify the `orchestratorRelease` as x.xx and orchestratorVersion as x.xx.x. For a list of current versions, see Supported AKS engine Versions

4. Find `customCloudProfile` and provide the URL to the tenant portal. For example, `https://portal.local.azurestack.external`.

5. Add `"identitySystem":"adfs"` if you're using AD FS. For example,

```
    "customCloudProfile": {
        "portalURL": "https://portal.local.azurestack.external",
        "identitySystem": "adfs"
    },
```

NOTE

If you're using Azure AD for your identity system, you don't need to add the **identitySystem** field.

6. In `masterProfile`, set the following fields:

FIELD	DESCRIPTION
dnsPrefix	Enter a unique string that will serve to identify the hostname of VMs. For example, a name based on the resource group name.
count	Enter the number of masters you want for your deployment. The minimum for an HA deployment is 3, but 1 is allowed for non-HA deployments.
vmSize	Enter a size supported by Azure Stack Hub, example `Standard_D2_v2`.
distro	Enter `aks-ubuntu-16.04` or `aks-ubuntu-18.04`.

7. In `agentPoolProfiles` update:

FIELD	DESCRIPTION
count	Enter the number of agents you want for your deployment. The maximum count of nodes to use per subscription is 50. If you are deploying more than one cluster per subscription ensure that the total agent count doesn't go beyond 50. Make sure to use the configuration items specified in the sample API model JSON file.
vmSize	Enter a size supported by Azure Stack Hub, example `Standard_D2_v2`.
distro	Enter `aks-ubuntu-16.04`, `aks-ubuntu-18.04` or `Windows`. Use `Windows` for agents that will run on Windows. For example, see kubernetes-windows.json

8. In `linuxProfile` update:

FIELD	DESCRIPTION
adminUsername	Enter the VM admin user name.
ssh	Enter the public key that will be used for SSH authentication with VMs. Use `ssh-rsa` and then the key. For instructions on creating a public key, see Create an SSH key for Linux.

If you're deploying to a custom virtual network, you can find instructions on finding and adding the required key and values to the appropriate arrays in the API Model in Deploy a Kubernetes cluster to a custom virtual network.

> **NOTE**
>
> The AKS engine for Azure Stack Hub doesn't allow you to provide your own certificates for the creation of the cluster.

9. If you're using Windows, in `windowsProfile` update the values of `adminUsername:` and `adminPassword`:

```
"windowsProfile": {
"adminUsername": "azureuser",
"adminPassword": "",
"sshEnabled": true
}
```

More information about the API model

- For a complete reference of all the available options in the API model, refer to the Cluster definitions.
- For highlights on specific options for Azure Stack Hub, refer to the Azure Stack Hub cluster definition specifics.

Add certificate when using ASDK

If you are deploying a cluster on the Azure Stack Development Kit (ASDK) and using Linux, you will need to add the root certificate to the trusted certificate store of the client VM running the AKS engine.

1. Find the root certificate in the VM at this directory: `/var/lib/waagent/Certificates.pem.`
2. Copy the certificate file:

```
sudo cp /var/lib/waagent/Certificates.pem /usr/local/share/ca-certificates/azurestacka.crt
sudo update-ca-certificates
```

Deploy a Kubernetes cluster

After you have collected all the required values in your API model, you can create your cluster. At this point you should:

Ask your Azure Stack Hub operator to:

- Verify the health of the system, suggest running `Test-AzureStack` and your OEM vendor's hardware monitoring tool.
- Verify the system capacity including resources such as memory, storage, and public IPs.

- Provide details of the quota associated with your subscription so that you can verify that there is still enough space for the number of VMs you plan to use.

Proceed to deploy a cluster:

1. Review the available parameters for AKS engine on Azure Stack Hub CLI flags.

PARAMETER	EXAMPLE	DESCRIPTION
azure-env	AzureStackCloud	To indicate to AKS engine that your target platform is Azure Stack Hub use `AzureStackCloud`.
identity-system	adfs	Optional. Specify your identity management solution if you are using Active Directory Federated Services (AD FS).
location	local	The region name for your Azure Stack Hub. For the ASDK, the region is set to `local`.
resource-group	kube-rg	Enter the name of a new resource group or select an existing resource group. The resource name needs to be alphanumeric and lowercase.
api-model	./kubernetes-azurestack.json	Path to the cluster configuration file, or API model.
output-directory	kube-rg	Enter the name of the directory to contain the output file `apimodel.json` and other generated files.
client-id	xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx	Enter the service principal GUID. The Client ID identified as the Application ID when your Azure Stack Hub administrator created the service principal.
client-secret	xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx	Enter the service principal secret. You set up the client secret when creating your service.
subscription-id	xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx	Enter your Subscription ID. You must provide a subscription for the tenent. Deployment to the administrative subscription is not supported. For more information, see Subscribe to an offer

Here is an example:

```
aks-engine deploy \
--azure-env AzureStackCloud \
--location <for asdk is local> \
--resource-group kube-rg \
--api-model ./kubernetes-azurestack.json \
--output-directory kube-rg \
--client-id xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
--client-secret xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
--subscription-id xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
--identity-system adfs # required if using AD FS
```

2. If for some reason the execution fails after the output directory has been created, you can correct the issue and rerun the command. If you are rerunning the deployment and had used the same output directory before, the AKS engine will return an error saying that the directory already exists. You can overwrite the existing directory by using the flag: `--force-overwrite`.

3. Save the AKS engine cluster configuration in a secure, encrypted location.

   Locate the file `apimodel.json`. Save it to a secure location. This file will be used as input in all of your other AKS engine operations.

   The generated `apimodel.json` contains the service principal, secret, and SSH public key you use in the input API model. It also has all the other metadata needed by the AKS engine to perform all other operations. If you lose it, the AKS engine won't be able configure the cluster.

   The secrets are **unencrypted**. Keep the file in an encrypted, secure place.

Verify your cluster

Check your cluster by connect to **kubectl**, getting the info, and then the states of your nodes.

1. Get the public IP address of one of your master nodes using the Azure Stack Hub portal.

2. From a machine with access to your Azure Stack Hub instance, connect via SSH into the new master node using a client such as PuTTY or MobaXterm.

3. For the SSH username, you use "azureuser" and the private key file of the key pair you provided for the deployment of the cluster.

4. Check that the cluster endpoints are running:

   ```
   kubectl cluster-info
   ```

   The output should look similar to the following:

   ```
   Kubernetes master is running at https://democluster01.location.domain.com
   CoreDNS is running at https://democluster01.location.domain.com/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
   Metrics-server is running at https://democluster01.location.domain.com/api/v1/namespaces/kube-system/services/https:metrics-server:/proxy
   ```

5. Then, review node states:

   ```
   kubectl get nodes
   ```

   The output should be similar to the following:

```
k8s-linuxpool-29969128-0   Ready      agent    9d    v1.15.5
k8s-linuxpool-29969128-1   Ready      agent    9d    v1.15.5
k8s-linuxpool-29969128-2   Ready      agent    9d    v1.15.5
k8s-master-29969128-0      Ready      master   9d    v1.15.5
k8s-master-29969128-1      Ready      master   9d    v1.15.5
k8s-master-29969128-2      Ready      master   9d    v1.15.5
```

Troubleshoot cluster deployment

When encountering errors while deploying a Kubernetes cluster using the AKS engine, you can check:

1. Are you using the correct Service Principal credentials (SPN)?
2. Does the SPN have a "Contributors" role to the Azure Stack Hub subscription?
3. Do you have a large enough quota in your Azure Stack Hub plan?
4. Is the Azure Stack Hub instance having a patch or upgrade being applied?

For more information, see the Troubleshooting article in the **Azure/aks-engine** GitHub repo.

Rotate your service principle secret

After the deployment of the Kubernetes cluster with AKS engine, the service principal (SPN) is used for managing interactions with the Azure Resource Manager on your Azure Stack Hub instance. At some point, the secret for this the service principal may expire. If your secret expires, you can refresh the credentials by:

- Updating each node with the new service principal secret.
- Or updating the API model credentials and running the upgrade.

Update each node manually

1. Get a new secret for your service principal from your cloud operator. For instructions for Azure Stack Hub, see Use an app identity to access Azure Stack Hub resources.
2. Use the new credentials provided by your cloud operator to update `/etc/kubernetes/azure.json` on each node. After making the update, restart both **kubelet** and **kube-controller-manager**.

Update the cluster with aks-engine update

Alternatively, you can replace the credentials in the `apimodel.json` and run upgrade using the updated json to the same or newer Kubernetes version. For instructions on upgrading the model see Upgrade a Kubernetes cluster on Azure Stack Hub

Next steps

Troubleshoot the AKS engine on Azure Stack Hub

Deploy a Kubernetes cluster to a custom virtual network on Azure Stack Hub

1/21/2022 • 10 minutes to read • Edit Online

You can deploy a Kubernetes cluster using the Azure Kubernetes Service (AKS) engine on a custom virtual network. This article looks at finding the information you need in your virtual network. You can find steps for calculating the IP addresses used by your cluster, setting the vales in the API Model, and setting the route table and network security group.

The Kubernetes cluster in Azure Stack Hub using the AKS engine uses the kubenet network plugin. The AKS engine on Azure Stack Hub also supports the Azure CNI network plugin.

- For a discussion of the kubenet networking plugin in Azure, see Use kubenet networking with your own IP address ranges in Azure Kubernetes Service (AKS).
- For a discussion of the Azure CNI networking plugin in Azure, see Configure Azure CNI networking in Azure Kubernetes Service (AKS).

Constraints when creating a custom virtual network

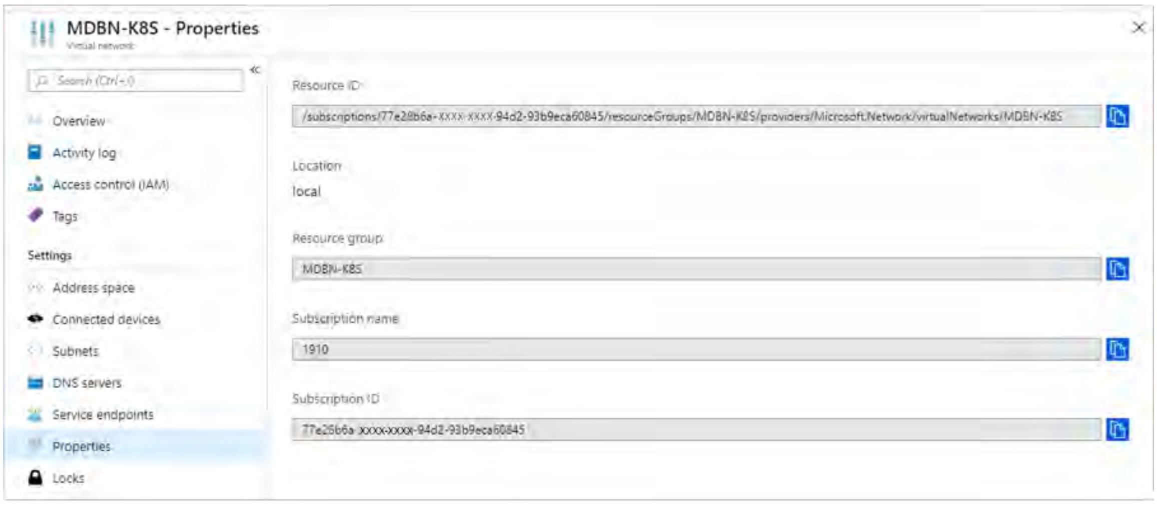
- The custom VNET must be in the same subscription as all of the other components of the Kubernetes cluster.
- The master nodes pool and the agent nodes pool must be in the same virtual network. You can deploy your nodes into different subnets within the same virtual network.
- The Kubernetes cluster subnet must use an IP range within the space of the custom virtual network IP range, see Get the IP address block.
- Consider that the recommended size of the node subnet(s) depends on the type of network plugin being used. As a general guideline, Azure CNI requires a larger number of IP addresses for the subnet supporting the agent node pools than kubenet. See the kubenet and Azure CNI examples below.
- The `169.254.0.0/16` address space may not be used for custom VNETs for Kubernetes clusters.

Create custom virtual network

You must have a custom virtual network in your Azure Stack Hub instance. For more information, see Quickstart: Create a virtual network using the Azure portal.

Create a new subnet in your virtual network. You will need to the get the subnet Resource ID and IP address range. You will use the Resource ID and range in your API model when you deploy your cluster.

1. Open the Azure Stack Hub user portal in your Azure Stack Hub instance.
2. Select **All resources**.
3. Enter the name of your virtual network in the search box.
4. Select **Subnets** > **+ Subnets** to add a subnet.
5. Add a **Name** and an **Address range** using CIDR notation. Select **OK**.
6. Select **Properties** in the **Virtual networks** blade. Copy the **Resource ID**, and then add `/subnets/<nameofyoursubnect>`. You will use this value as your value for the `vnetSubnetId` key in the API model for your cluster. The Resource ID for the subnet uses the following format: `/subscriptions/SUB_ID/resourceGroups/RG_NAME/providers/Microsoft.Network/virtualNetworks/VNET_NAME/subnets/SUBNET_NAME`



MDBN-K8S - Properties

Virtual network

Resource ID

/subscriptions/77e28b6a-XXXX-XXXX-94d2-93b9eca60845/resourceGroups/MDBN-K8S/providers/Microsoft.Network/virtualNetworks/MDBN-K8S

Location

local

Resource group

MDBN-K8S

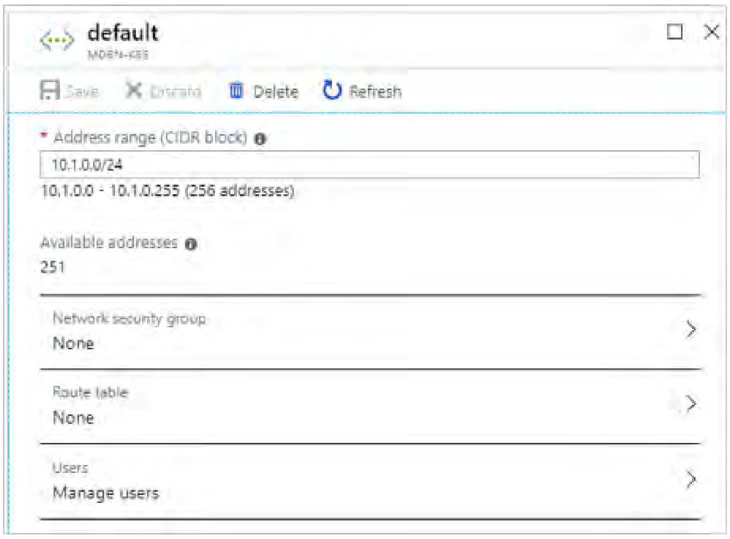
Subscription name

1910

Subscription ID

77e28b6a-xxxx-xxxx-94d2-93b9eca60845

7. Select **Subnets** in the **Virtual networks** blade. Select the subnet name, for example `control-plane-sn`.



default

MDBN-K8S

* Address range (CIDR block)

10.1.0.0/24

10.1.0.0 - 10.1.0.255 (256 addresses)

Available addresses

251

Network security group

None

Route table

None

Users

Manage users

8. In the subnet blade, make a note of the address range (CIDR Block) of each subnet.

Considerations for selecting an address space

When you create a custom virtual network, you specify the IP address space of your network and an IP address range for every subnet. Consider the following factors when you choose the address spaces and ranges to use in your Kubernetes cluster:

- Overlapping address spaces might result in IP address clashes or communication errors. To reduce the risk of overlapping IP addresses, choose a unique address space for your new virtual network.
- Address spaces in the `10/8`, `172.16/12`, and `192.168/16` ranges often are used for private networks, and they might be used by your existing datacenter infrastructure. If your Kubernetes applications use resources in your datacenter, reduce the risk of clashes by choosing an address space for your custom virtual network that's different from your datacenter's address space.
- We recommend that you use a dedicated subnet for your Kubernetes cluster.
- If using multiple existing virtual networks consider using different address spaces on each network if you intend to use virtual network peering. Overlapping address spaces may impair your ability to enable peering.

Get the IP address blocks

The AKS engine supports deploying into an existing virtual network. When deploying into an existing virtual network, your cluster will use blocks of consecutive addresses for agent nodes, master nodes, cluster services, and containers (pods). Each address block can be translated into a subnet within the virtual network. All address blocks in the cluster deployment must be a part of the overall virtual network address space, choosing address blocks outside of the virtual network address space may result in connectivity problems.

There is a minimum of three address blocks required when setting up a Kubernetes cluster:

- Nodes address block: This is the address block used for assigning addresses to the cluster nodes. This can be a single address block for all cluster nodes or can be separate blocks (subnets) for master and agent pools. Take into consideration the node count in your cluster when selecting the address range for this block. For Azure CNI nodes and containers get their addresses from the same address block thus take into account the number of containers you want to deploy to your cluster when choosing the address range when using Azure CNI.
- Services address block: This is the address block from which services deployed to the Kubernetes cluster will get their cluster address from. Take into consideration the maximum number of services you intend to run in your cluster when selecting the address range for this block.
- Cluster address block: This is the address block from which pods will get their cluster address from. Take into consideration the maximum number of pods you intend to run in your cluster when selecting the address range for this block. As mention earlier, for Azure CNI the cluster and nodes address blocks are the same.

In addition to the address blocks, for master nodes you will need to set two more values. You will need to know the number of IP addresses you will need to reserve for your cluster, and the first consecutive static IP within the subnet IP space. The AKS engine requires a range of up to 16 unused IP addresses when you use multiple master nodes. The cluster will use one IP address for each master up to five masters. The AKS engine will also require the next 10 IP address after the last master for headroom IP address reservation. Finally, another IP address will be used by the load balancer after the masters and headroom reservation for a total of 16. When placing your block of IP addresses, the subnet requires the following allocations of the existing IP addresses:

- The first four IP addresses and the last IP address are reserved and can't be used in any Azure subnet
- A buffer of 16 IP addresses should be left open.
- The value of your cluster's first IP should be toward the end of the address space to avoid IP conflicts. If possible, assign to the `firstConsecutiveStaticIP` property to an IP address near the *end* of the available IP address space in the subnet.

For example, for a cluster with three master nodes. If you are using a subnet with 256 addresses, for example 10.100.0.0/24, you will need to set your first consecutive static IP address before 239. The following table shows the addresses and considerations:

RANGE FOR /24 SUBNET	NUMBER	NOTE
10.100.0.0 - 10.100.0.3	4	Reserved by Azure subnet.
10.100.0.239-10.100.0.255	14	Reserved by AKS Engine. 5 IP addresses for 5 masters 10 IP addresses for headroom 1 IP address for the load balancer

In this example, then `firstConsecutiveStaticIP` property would be `10.100.0.239`. For larger subnets, for example /16 with more than 60 thousand addresses, you may not find it to be practical to set your static IP assignments to the end of the network space. Set your cluster static IP address range away from the first 24 addresses in your IP space so that the cluster can be resilient when claiming addresses.

Kubenet address blocks example

In the following example, you can see how these various considerations fill out the address space in the virtual network for a cluster using the kubenet network plugin with dedicated subnets for the master and agent node pools with three nodes per pool.

VNET address space: 10.100.0.0/16.

ADDRESS BLOCK (SUBNET)	CIDR	IP RANGE	IP COUNT (AVAILABLE)
Master nodes block	10.100.0.0/24	10.100.0.0 - 10.100.0.255	255 - 4 reserved = 251
Agent nodes block	10.100.1.0/24	10.100.1.0 - 10.100.1.255	255 - 4 reserved = 251
Services block	10.100.16.0/20	10.100.16.0 - 10.100.31.255	4,096 - 5 reserved = 4,091
Cluster block	10.100.128.0/17	10.100.128.0 - 10.100.255.255	32,768 - 5 reserved = 32,763

In this example, the `firstConsecutiveStaticIP` property would be `10.100.0.239`.

Azure CNI address blocks example

In the following example, you can see how these various considerations fill out the address space in the virtual network for a cluster using the Azure CNI network plugin with dedicated subnets for the master and agent node pools with three nodes per pool.

VNET address space: 172.24.0.0/16.

ADDRESS BLOCK (SUBNET)	CIDR	IP RANGE	IP COUNT (AVAILABLE)
Master nodes block	172.24.0.0/24	172.24.0.0 - 172.24.0.255	255 - 4 reserved = 251
Agent nodes & cluster block	172.24.128.0/17	172.24.128.0 - 172.24.255.255	32,768 - 5 reserved = 32,763
Services block	172.24.16.0/20	172.24.16.0 - 172.24.31.255	4,096 - 5 reserved = 4,091

In this example, the `firstConsecutiveStaticIP` property would be `172.24.0.239`.

Update the API model

Update the API model used to deploy the cluster from the AKS engine to your custom virtual network.

In **masterProfile** set the following values:

FIELD	EXAMPLE	DESCRIPTION
vnetSubnetId	`/subscriptions/77e28b6a-582f-42b0-94d2-93b9eca60845/resourceGroups/MDBN-K8S/providers/Microsoft.Network/virtualNetworks/MDBN-K8S/subnets/control-plane-sn`	Specify the Azure Resource Manager path ID the subnet. This value maps to the master nodes address block above.
firstConsecutiveStaticIP	10.100.0.239	Assign to the `firstConsecutiveStaticIP` configuration property an IP address that is near the *end* of the available IP address space in the desired subnet. `firstConsecutiveStaticIP` only applies to the master pool.

In **agentPoolProfiles** set the following values:

FIELD	EXAMPLE	DESCRIPTION
vnetSubnetId	`/subscriptions/77e28b6a-582f-42b0-94d2-93b9eca60845/resourceGroups/MDBN-K8S/providers/Microsoft.Network/virtualNetworks/MDBN-K8S/subnets/agents-sn`	Specify the Azure Resource Manager path ID the subnet. This value maps to the agent nodes address block above.

In **orchestratorProfile**, find **kubernetesConfig** and set the following value:

FIELD	EXAMPLE	DESCRIPTION
clusterSubnet	`10.100.128.0/17`	The IP subnet used for allocating IP addresses for pod network interfaces. This value maps to the cluster address block above. The subnet must be in the VNET address space. With Azure CNI enabled, the default value is 10.240.0.0/12. Without Azure CNI, the default value is 10.244.0.0/16. Use /16 instead /24 subnet. If you use /24, this subnet will be assigned to one node only. Other node will not get POD network assigned, as you will have run out of the IP space, so they will be not ready in the cluster.
serviceCidr	`10.100.16.0/20`	The IP subnet used for allocating IP addresses for services deployed in the cluster. This value maps to the cluster services block above.
dnsServiceIP	`10.100.16.10`	The IP address to be assigned to the cluster DNS service. The address must come from the serviceCidr subnet. This value must be set when specifying the serviceCidr. The default value is the .10 address of the serviceCidr subnet.

For example, if using kubenet:

With a network address space of `10.100.0.0/16` where the subnet for `control-plane-sn` is `10.100.0.0/24` and `agents-sn` is `10.100.1.0/24`

```
"masterProfile": {
  ...
  "vnetSubnetId": "/subscriptions/77e28b6a-582f-42b0-94d2-93b9eca60845/resourceGroups/MDBN-
K8S/providers/Microsoft.Network/virtualNetworks/MDBN-K8S/subnets/control-plane-sn",
  "firstConsecutiveStaticIP": "10.100.0.239",
  ...
},
...
"agentPoolProfiles": [
  {
    ...
    "vnetSubnetId": "/subscriptions/77e28b6a-582f-42b0-94d2-93b9eca60845/resourceGroups/MDBN-
K8S/providers/Microsoft.Network/virtualNetworks/MDBN-K8S/subnets/agents-sn",
    ...
  },
    ...
"kubernetesConfig": [
  {
    ...
    "clusterSubnet": "10.100.128.0/17",
    "serviceCidr": "10.100.16.0/20",
    "dnsServiceIP" : "10.100.16.10",

    ...
  },
```

For example, if using Azure CNI:
With a network address space of `172.24.0.0/16` where the subnet for `control-plane-sn` is `172.24.0.0/24` and `k8s-sn` is `172.24.128.0/17`

```
"masterProfile": {
  ...
  "vnetSubnetId": "/subscriptions/77e28b6a-582f-42b0-94d2-93b9eca60845/resourceGroups/MDBN-
K8S/providers/Microsoft.Network/virtualNetworks/MDBN-K8S/subnets/control-plane-sn",
  "firstConsecutiveStaticIP": "172.24.0.239",
  ...
},
...
"agentPoolProfiles": [
  {
    ...
    "vnetSubnetId": "/subscriptions/77e28b6a-582f-42b0-94d2-93b9eca60845/resourceGroups/MDBN-
K8S/providers/Microsoft.Network/virtualNetworks/MDBN-K8S/subnets/k8s-sn",
    ...
  },
    ...
"kubernetesConfig": [
  {
    ...
    "clusterSubnet": "172.24.128.0/17",
    "serviceCidr": "172.24.16.0/20",
    "dnsServiceIP" : "172.24.16.10",
    ...
  },
```

Deploy your cluster

After adding the values to your API model, you can deploy your cluster from your client machine using the `deploy` command using the AKS engine. For instructions, see Deploy a Kubernetes cluster.

Set the route table

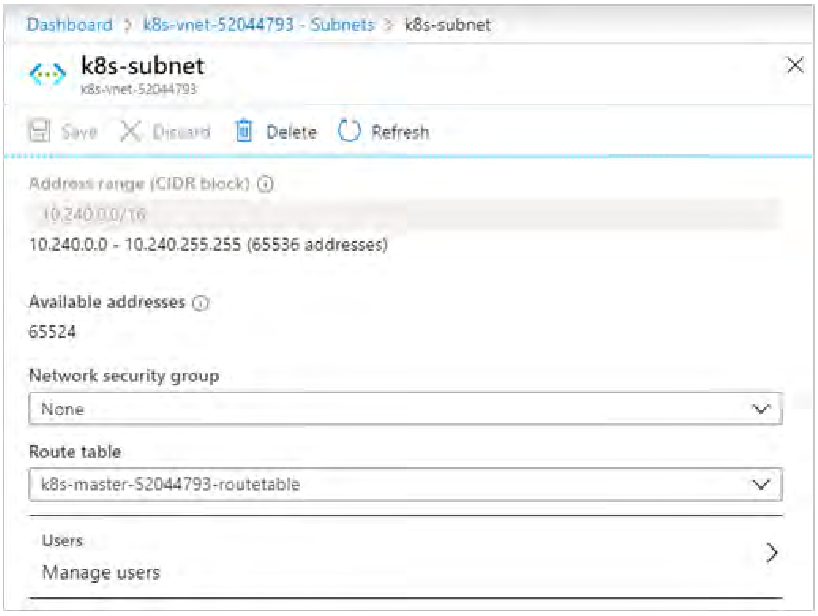
If you are using kubenet, for example, `networkPlugin` : `kubenet` in the `kubernetesConfig` API model configuration object. After you deploy your cluster, return to your virtual network in the Azure Stack user portal. Set both the route table and the network security group (NSG) in the subnet blade. After you have successfully deployed a cluster to your custom virtual network, get the ID of the Route Table resource from **Network** blade in your cluster's resource group.

1. Open the Azure Stack Hub user portal in your Azure Stack Hub instance.

2. Select **All resources**.

3. Enter the name of your virtual network in the search box.

4. Select **Subnets** and then select the name of the subnet that contains your cluster.



Address range (CIDR block)
10.240.0.0/16
10.240.0.0 - 10.240.255.255 (65536 addresses)

Available addresses
65524

Network security group
None

Route table
k8s-master-52044793-routetable

Users
Manage users

5. Select **Route table** and then select the route table for your cluster.

> **NOTE**
>
> Custom virtual network for Kubernetes Windows cluster has a known issue.

Next steps

- Read about the The AKS engine on Azure Stack Hub
- Read about Azure Monitor for containers overview

Scale a Kubernetes cluster on Azure Stack Hub

1/21/2022 • 2 minutes to read • Edit Online

You can scale your cluster with the AKS engine using the **scale** command. The **scale** command reuses your cluster configuration file (`apimodel.json`) inside the output directory as input for a new Azure Resource Manager deployment. The engine executes the scaling operation against the specified agent pool. When the scaling operation is done, the engine updates the cluster definition in that same `apimodel.json` file to reflect the new node count in order to reflect the updated, current cluster configuration.

Scale a cluster

The `aks-engine scale` command can increase or decrease the number of nodes in an existing agent pool in an `aks-engine` Kubernetes cluster. Nodes will always be added or removed from the end of the agent pool. Nodes will be cordoned and drained before deletion.

Values for the scale command

The following parameters are used by the scale command to find your cluster definition file and update your cluster.

PARAMETER	EXAMPLE	DESCRIPTION
azure-env	AzureStackCloud	When using Azure Stack Hub, the environment names needs to be set to `AzureStackCloud`.
location	local	This is the region for your Azure Stack Hub instance. For an ASDK, the region is set to `local`.
resource-group	kube-rg	The name of the resource group that contains your cluster.
subscription-id		The GUID of the subscription that contains the resources used by your cluster. Make sure you have enough quota on your subscription to scale.
client-id		The client ID of the service principal used in creating your cluster from the AKS engine.
client-secret		The service principal secret used when creating your cluster.
api-model	kube-rg/apimodel.json	The path to your cluster definition file (apimodel.json). This may be at: _output/<dnsPrefix>/apimodel.json
new-node-count	9	Desired node count.

PARAMETER	EXAMPLE	DESCRIPTION
apiserver		Master FQDN. Needed when scaling down.
identity-system	adfs	Optional. Specify your identity management solution if you are using Active Directory Federated Services (AD FS).

You must specify the `--azure-env` parameter when scaling a cluster in Azure Stack Hub. For more information about parameters and their values used in the `scale` command for the AKS engine, see Scale - parameters.

Command to scale your cluster

To scale the cluster you run the following command:

```
aks-engine scale \
    --azure-env AzureStackCloud   \
    --location <for an ASDK is local> \
    --resource-group <cluster resource group>
    --subscription-id xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
    --client-id xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
    --client-secret xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
    --api-model <path to your apomodel.json file>
    --new-node-count <desired node count> \
    --apiserver <master FQDN> \
    --identity-system adfs # required if using AD FS
```

Next steps

- Read about the The AKS engine on Azure Stack Hub
- Upgrade a Kubernetes cluster on Azure Stack Hub

Upgrade a Kubernetes cluster on Azure Stack Hub

1/21/2022 • 5 minutes to read • Edit Online

The AKS engine allows you to upgrade the cluster that was originally deployed using the tool. You can maintain the clusters using the AKS engine. Your maintenance tasks are similar to any IaaS system. You should be aware of the availability of new updates and use the AKS engine to apply them.

Upgrade a cluster

The upgrade command updates the Kubernetes version and the base OS image. Every time that you run the upgrade command, for every node of the cluster, the AKS engine creates a new VM using the AKS Base Image associated to the version of **aks-engine** used. You can use the `aks-engine upgrade` command to maintain the currency of every master and agent node in your cluster.

Microsoft doesn't manage your cluster. But Microsoft provides the tool and VM image you can use to manage your cluster.

For a deployed cluster upgrades cover:

- Kubernetes
- Azure Stack Hub Kubernetes provider
- Base OS

When upgrading a production cluster, consider:

- Are you using the correct cluster specification (`apimodel.json`) and resource group for the target cluster?
- Are you using a reliable machine for the client machine to run the AKS engine and from which you are performing upgrade operations?
- Make sure that you have a backup cluster and that it is operational.
- If possible, run the command from a VM within the Azure Stack Hub environment to decrease the network hops and potential connectivity failures.
- Make sure that your subscription has enough space for the entire process. The process allocates new VMs during the process.
- No system updates or scheduled tasks are planned.
- Set up a staged upgrade on a cluster that is configured exactly as the production cluster and test the upgrade there before doing so in your production cluster

Steps to upgrade to a newer Kubernetes version

NOTE

The AKS base image will also be upgrade if you are using a newer version of the aks-engine and the image is available in the marketplace.

The following instructions use the minimum steps to perform the upgrade. If would like more detail, see the article Upgrading Kubernetes Clusters.

1. You need to first determine the versions you can target for the upgrade. This version depends on the version you currently have and then use that version value to perform the upgrade. The Kubernetes versions supported by your AKS Engine can be listed by running the following command:

```
aks-engine get-versions --azure-env AzureStackCloud
```

For a complete mapping of AKS engine, AKS Base Image and Kubernetes versions see Supported AKS Engine Versions.

2. Collect the information you will need to run the `upgrade` command. The upgrade uses the following parameters:

PARAMETER	EXAMPLE	DESCRIPTION
azure-env	AzureStackCloud	To indicate to AKS engine that your target platform is Azure Stack Hub use `AzureStackCloud`.
location	local	The region name for your Azure Stack Hub. For the ASDK, the region is set to `local`.
resource-group	kube-rg	Enter the name of a new resource group or select an existing resource group. The resource name needs to be alphanumeric and lowercase.
subscription-id	xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx	Enter your Subscription ID. For more information, see Subscribe to an offer
api-model	./kubernetes-azurestack.json	Path to the cluster configuration file, or API model.
client-id	xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx	Enter the service principal GUID. The Client ID identified as the Application ID when your Azure Stack Hub administrator created the service principal.
client-secret	xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx	Enter the service principal secret. This is the client secret you set up when creating your service.
identity-system	adfs	Optional. Specify your identity management solution if you are using Active Directory Federated Services (AD FS).

3. With your values in place, run the following command:

```
aks-engine upgrade \
--azure-env AzureStackCloud \
--location <for an ASDK is local> \
--resource-group kube-rg \
--subscription-id xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
--api-model kube-rg/apimodel.json \
--upgrade-version 1.18.15 \
--client-id xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
--client-secret xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
--identity-system adfs # required if using AD FS
```

4. If for any reason the upgrade operation encounters a failure, you can rerun the upgrade command after addressing the issue. The AKS engine will resume the operation where it failed the previous time.

Steps to only upgrade the OS image

1. Review the supported-kubernetes-versions table and determine if you have the version of aks-engine and AKS base Image that you plan for your upgrade. To view the version of aks-engine run: `aks-engine version`.
2. Upgrade your AKS engine accordingly, in the machine where you have installed aks-engine run: `./get-akse.sh --version vx.xx.x` replacing **x.xx.x** with your targeted version.
3. Ask your Azure Stack Hub operator to add the version of the AKS Base Image you need in the Azure Stack Hub Marketplace that you plan to use.
4. Run the `aks-engine upgrade` command using the same version of Kubernetes that you are already using, but add the `--force`. You can see an example in Forcing an upgrade.

Steps to update cluster to OS version Ubuntu 18.04

With AKS engine version 0.60.1 and above you can upgrade your cluster VMs from Ubuntu 16.04 to 18.04. Follow these steps:

1. Locate and edit the `api-model.json` file that was generated during deployment. This should be the same file used for any upgrade or scale operation with `aks-engine`.
2. Locate the sections for `masterProfile` and `agentPoolProfiles`, within those sections change the value of `distro` to `aks-ubuntu-18.04`.
3. Save the `api-model.json` file and use the `api-model.json` file in your `aks-engin upgrade` command as you would in the Steps to upgrade to a newer Kubernetes version

Forcing an upgrade

There may be conditions where you may want to force an upgrade of your cluster. For example, on day one you deploy a cluster in a disconnected environment using the latest Kubernetes version. The following day Ubuntu releases a patch to a vulnerability for which Microsoft generates a new **AKS Base Image**. You can apply the new image by forcing an upgrade using the same Kubernetes version you already deployed.

```
aks-engine upgrade \
--azure-env AzureStackCloud
--location <for an ASDK is local> \
--resource-group kube-rg \
--subscription-id xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
--api-model kube-rg/apimodel.json \
--upgrade-version 1.18.15 \
--client-id xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
--client-secret xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx \
--force
```

For instructions, see Force upgrade.

Next steps

- Read about the The AKS engine on Azure Stack Hub
- Scale a Kubernetes cluster on Azure Stack Hub

Enabling service account token volume projection for the AKS engine on Azure Stack Hub

1/21/2022 • 2 minutes to read • Edit Online

Istio is a configurable, open source service-mesh layer that connects, monitors, and secures the containers in a Kubernetes cluster. Istio 1.3 and higher uses a feature in Kubernetes called *service account token volume projection*. This feature is not enabled by default in Kubernetes clusters deployed by AKS engine. In this article, you can find the API model json properties in the `apiServerConfig` element that shows the Kubernetes API server flags required to enable service account token volume projection for your cluster.

For more information about service account token volume projection, see Service Account Token Volume Projection.

Enable service account token volume projection

To enable service account token volume projection, add the following settings into your API model json file.

```
{
    "kubernetesConfig": {
        "apiServerConfig": {
            "--service-account-api-audiences": "api,istio-ca",
            "--service-account-issuer": "kubernetes.default.svc",
            "--service-account-signing-key-file": "/etc/kubernetes/certs/apiserver.key"
        }
    }
}
```

> **NOTE**
>
> You may have to adjust `--service-account-api-audiences` and `--service-account-issuer` to your specific use case.

For a full example API model, refer to istio.json.

Next steps

- Read about the The AKS engine on Azure Stack Hub
- Upgrade a Kubernetes cluster on Azure Stack Hub

Add container storage to Kubernetes in Azure Stack Hub

1/21/2022 • 2 minutes to read • Edit Online

IMPORTANT

This feature is currently in public preview. This preview version is provided without a service level agreement, and it's not recommended for production workloads. Certain features might not be supported or might have constrained capabilities. For more information, see Supplemental Terms of Use for Microsoft Azure Previews.

As part of the Kubernetes community effort (Kubernetes in-tree to CSI volume migration) to move in-tree volume providers to Container Storage Interface CSI, you can find the following two CSI drivers in Azure Stack: Azure Disk and NFS.

DETAILS	AZURE DISK CSI DRIVER	NFS CSI DRIVER
Project Repository	azuredisk-csi-driver	csi-driver-nfs
CSI Driver Version	v1.0.0+	v3.0.0+
Access Mode	ReadWriteOnce	ReadWriteOnce ReadOnlyMany ReadWriteMany
Windows Agent Node	Support	Not support
Dynamic Provisioning	Support	Support
Considerations	Azure Disk CSI Driver Limitations	Users will be responsible for setting up and maintaining the NFS server.
Slack Support Channel	#provider-azure	#sig-storage

Currently, disconnected environments do not support CSI Drivers.

Requirements

- Azure Stack build 2011 and later.
- AKS engine version v0.60.1 and later.
- Kubernetes version 1.18 and later.
- Since the Controller server of CSI Drivers requires two replicas, a single node master pool is not recommended.
- Helm 3

Install and uninstall CSI drivers

In this section, follow the example commands to deploy a statefulset application consuming CSI Driver.

Azure disk CSI driver

Install CSI driver

```
helm repo add azuredisk-csi-driver https://raw.githubusercontent.com/kubernetes-sigs/azuredisk-csi-
driver/master/charts
helm install azuredisk-csi-driver azuredisk-csi-driver/azuredisk-csi-driver --namespace kube-system --set
cloud=AzureStackCloud --set controller.runOnMaster=true --version v1.0.0
```

Deploy storage class

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/azuredisk-csi-
driver/master/deploy/example/storageclass-azuredisk-csi-azurestack.yaml
```

Deploy example statefulset application

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/azuredisk-csi-
driver/master/deploy/example/statefulset.yaml
```

Validate volumes and applications

You should see a sequence of timestamps are persisted in the volume.

```
kubectl exec statefulset-azuredisk-0 -- tail /mnt/azuredisk/outfile
```

Delete example statefulset application

```
kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/azuredisk-csi-
driver/master/deploy/example/statefulset.yaml
```

Delete storage class

Before you delete the Storage Class, make sure Pods that consume the Storage Class have been terminated.

```
kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/azuredisk-csi-
driver/master/deploy/example/storageclass-azuredisk-csi-azurestack.yaml
```

Uninstall CSI driver

```
helm uninstall azuredisk-csi-driver --namespace kube-system
helm repo remove azuredisk-csi-driver
```

NFS CSI driver

Install CSI driver

```
helm repo add csi-driver-nfs https://raw.githubusercontent.com/kubernetes-csi/csi-driver-nfs/master/charts
helm install csi-driver-nfs csi-driver-nfs/csi-driver-nfs --namespace kube-system --set
controller.runOnMaster=true --version v3.0.0
```

Deploy NFS server

> **NOTE**
>
> The NFS Server is just for validation, set up and maintain your NFS Server properly for production.

Set up and maintain your NFS Server properly for production.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/csi-driver-nfs/master/deploy/example/nfs-provisioner/nfs-server.yaml
```

Deploy Storage Class

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/csi-driver-nfs/master/deploy/example/storageclass-nfs.yaml
```

Deploy example statefulset application

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/csi-driver-nfs/master/deploy/example/statefulset.yaml
```

Validate volumes and applications

You should see a sequence of timestamps are persisted in the volume.

```
kubectl exec statefulset-nfs-0 -- tail /mnt/nfs/outfile
```

Delete example statefulset application

```
kubectl delete -f https://raw.githubusercontent.com/kubernetes-csi/csi-driver-nfs/master/deploy/example/statefulset.yaml
```

Delete storage class

Before you delete the Storage Class, make sure Pods that consume the Storage Class have been terminated.

```
kubectl delete -f https://raw.githubusercontent.com/kubernetes-csi/csi-driver-nfs/master/deploy/example/storageclass-nfs.yaml
```

Delete example NFS server

```
kubectl delete -f https://raw.githubusercontent.com/kubernetes-csi/csi-driver-nfs/master/deploy/example/nfs-provisioner/nfs-server.yaml
```

Uninstall CSI driver

```
helm uninstall csi-driver-nfs --namespace kube-system
helm repo remove csi-driver-nfs
```

Azure Disk CSI Driver limitations on Azure Stack Hub

- Azure Disk IOPS is capped at 2300, read VM sizes supported in Azure Stack Hub for details.
- Azure Stack Hub doesn't support shared disk, so parameter `maxShares` larger than 1 is not valid in a

StorageClass.

- Azure Stack Hub only supports standard locally redundant (Standard_LRS) and Premium Locally redundant (Premium_LRS) Storage Account types, so only Standard_LRS and Premium_LRS are valid for parameter `skuName` in a `StorageClass`.
- Azure Stack Hub doesn't support incremental disk snapshot, so only false is valid for parameter `incremental` in a `VolumeSnapshotClass`.
- For Windows agent nodes, you will need to install Windows CSI Proxy, refer to Windows CSI Proxy. To enable the proxy via AKS engine API model, refer to CSI Proxy for Windows.

Next steps

- Read about the The AKS engine on Azure Stack Hub

Rotate Kubernetes certificates on Azure Stack Hub

1/21/2022 • 4 minutes to read • Edit Online

This document provides guidance on how to rotate certificates on an existing AKS Engine cluster and recommendations for using adopting `aks-engine rotate-certs` as a tool.

IMPORTANT

This feature is currently in public preview. This preview version is provided without a service level agreement, and it's not recommended for production workloads. Certain features might not be supported or might have constrained capabilities. For more information, see Supplemental Terms of Use for Microsoft Azure Previews.

Prerequisites

This guide assumes that you already have deployed a cluster using AKS engine and the cluster is in a healthy state.

Planning for certificate rotation

When considering using this functionality, be aware that the Kubernetes control plane will be unavailable during the update, validation, and restart steps. Plan this maintenance operation accordingly. Also, plan to execute this operation in a staging environment with equal configuration to the production environment before trying in production.

Review the following considerations before attempting this operation:

- You will need access to the API model (`apimodel.json`) that was generated by the commands `aks-engine deploy` or `aks-engine generate` . By default this file is placed into a relative directory such as `_output/<clustername>/` .
- An `aks-engine rotate-certs` operation causes API server downtime.
- `aks-engine rotate-certs` expects an API model that conforms to the current state of the cluster. `aks-engine rotate-certs` executes remote commands on the cluster nodes and uses the API model information to establish a secure SSH connection. `aks-engine rotate-certs` also relies on some resources to be named in accordance with the original `aks-engine` deployment, for example, VMs must follow the naming provided by `aks-engine` .
- `aks-engine rotate-certs` relies upon a working connection to the cluster control plane during certificate rotation:
  - To validate each step of the process.
  - To restart/recreate cluster resources such as **kube-system pods** and service account tokens.

  If you are rotating the certificates of a cluster in a VNet closed to outside access, you must run `aks-engine rotate-certs` from a host VM that has network access to the control plane, for example, a jumpbox VM that resides in the same VNet as the master VMs.
- If you are using `aks-engine rotate-certs` in production, it is recommended to stage a certificate rotation test on a cluster that was built to the same specifications. That is, the cluster is built with the same cluster configuration, the same version of the AKS engine command-line tool, and the same set of enabled addons as your production cluster before performing the certificate rotation. The AKS engine supports

different cluster configurations and the extent of end-to-end testing that the AKS engine team runs cannot practically cover every possible configuration. Therefore, it is recommended that you ensure in a staging environment that your specific cluster configuration works with `aks-engine rotate-certs` before attempting the operation on your production cluster.

- `aks-engine rotate-certs` does **not** guarantee backwards compatibility. If you deployed with aks-engine version 0.60.x, you should prefer executing the certificate rotation process with version 0.60.x.
- Fetching a new set of certificates from Key Vault is not supported at this point.
- Use a reliable network connection. `aks-engine rotate-certs` requires the execution of multiple remote commands, which are subject to potential failures, mostly if the connection to the cluster nodes is not reliable. Running `aks-engine rotate-certs` from a VM running on the target Azure Stack stamp can reduce the occurrence of transient issues.

Parameters

PARAMETER	REQUIRED	DESCRIPTION
--api-model	yes	Relative path to the API model (cluster definition) that declares the expected cluster configuration.
--ssh-host	yes	Fully qualified domain name (FQDN), or IP address, of an SSH listener that can reach all nodes in the cluster.
--linux-ssh-private-key	yes	Path to a valid private SSH key to access the cluster's Linux nodes.
--location	yes	Azure location where the cluster is deployed.
--subscription-id	yes	Azure subscription where the cluster infra is deployed.
--resource-group	yes	Azure resource group where the cluster infra is deployed.
--client-id	depends	The service principal client ID. Required if the auth-method is set to client_secret or client_certificate.
--client-secret	depends	The service principal client secret. Required if the auth-method is set to client_secret.
--azure-env	depends	The target cloud name. Optional if target cloud is AzureCloud.
--certificate-profile	no	Relative path to a JSON file containing the new set of certificates.
--force	no	Force execution even if the API Server is not responsive.

Simple steps to rotate certificates

Once you have read all the requirements, run `aks-engine rotate-certs` with the appropriate arguments:

```
./bin/aks-engine rotate-certs \
  --location <resource-group-location> \
  --api-model <generated-apimodel.json> \
  --linux-ssh-private-key <private-SSH-key> \
  --ssh-host <apiserver-URI> \
  --resource-group <resource-group-name> \
  --client-id <service-principal-id> \
  --client-secret <service-principal-secret> \
  --subscription-id <subscription-id> \
  --azure-env <cloud-name>
```

For example:

```
./bin/aks-engine rotate-certs \
  --location "westus2" \
  --api-model "_output/my-cluster/apimodel.json" \
  --linux-ssh-private-key "~/.ssh/id_rsa" \
  --ssh-host "my-cluster.westus2.cloudapp.azure.com"\
  --resource-group "my-cluster" \
  --client-id "12345678-XXXX-YYYY-ZZZZ-1234567890ab" \
  --client-secret "12345678-XXXX-YYYY-ZZZZ-1234567890ab" \
  --subscription-id "12345678-XXXX-YYYY-ZZZZ-1234567890ab" \
  --azure-env "AzureStackCloud" # optional if targeting AzureCloud
```

Troubleshooting

If the certificate rotation process halts before completion due to a failure or transient issue, for example, network connectivity, it is safe to rerun `aks-engine rotate-certs` using the `--force` flag.

Also notice that `aks-engine rotate-certs` logs the output of every step in file `/var/log/azure/rotate-certs.log` (Linux) and `c:\\k\\rotate-certs.log` (Windows).

For more information on what happens under the hood when running this operation or for further customization, see Under The Hood.

Next steps

- Read about the The AKS engine on Azure Stack Hub

Use Azure Key Vault to store secrets with Azure Kubernetes Service on Azure Stack Hub

1/21/2022 • 2 minutes to read • Edit Online

Azure Key Vault provider for Secrets Store CSI driver allows you to get secret contents stored in an Azure Key Vault instance and use the Secrets Store CSI driver interface to mount them into Kubernetes pods.

Secrets store CSI driver

You can use the Secrets Store CSI driver to mount your secrets, keys, and certificates on pod start using a CSI volume. You can use the drives can be used to:

- Mount multiple secrets store objects as a single volume.
- Pod identity to restrict access with specific identities.
- Pod portability with the SecretProviderClass CRD.
- Windows containers (Kubernetes version v1.18+).
- Sync with Kubernetes Secrets (Secrets Store CSI Driver v0.0.10+).
- Multiple secrets stores providers in the same cluster.

Get started secrets store CSI driver

1. Set up the correct role assignments and access policies.
2. Install Azure Key Vault Provider for Secrets Store CSI Driver through Helm or YAML deployment files.
3. Learn how to use the Azure Key Vault Provider and supported configurations.
4. Get up to speed with the application workflow with the walkthrough.

Support for the driver

Azure Key Vault Provider for Secrets Store CSI Driver is an open source project that is **not** covered by the Microsoft Azure support policy. Please search open issues here, and if your issue isn't already represented open a new one. The project maintainers will respond to the best of their abilities.

Next steps

Overview of the AKS engine

Troubleshoot the AKS engine on Azure Stack Hub

1/21/2022 • 4 minutes to read • Edit Online

You may find an issue when deploying or working with the AKS engine on Azure Stack Hub. This article looks at the steps to troubleshoot your deployment of the AKS engine. Collect information about your AKS engine, collect Kubernetes logs, and review custom script extension error codes. You can also open a GitHub issue for the AKS engine.

Troubleshoot the AKS engine install

If your previous installation steps failed, you can install the AKS engine using the GoFish package manager. GoFish describes itself as a cross-platform Homebrew.

You can find instructions for using GoFish to install the AKS engine at Install the aks-engine command line tool

Collect node and cluster logs

You can find the instructions on collecting node and cluster logs at Retrieving Node and Cluster Logs.

Prerequisites

This guide assumes you've already downloaded the Azure CLI and the AKS engine.

This guide also assumes that you've deployed a cluster using the AKS engine. For more information, see Deploy a Kubernetes cluster with the AKS engine on Azure Stack Hub .

Retrieving logs

The `aks-engine get-logs` command can be useful to troubleshoot issues with your cluster. The command produces, collects, and downloads a set of files to your workstation. The files include node configuration, cluster state and configuration, and set up log files.

At a high level: the command works by establishing an SSH session into each node, executing a log collection script that collects and zips relevant files, and downloading the .ZIP file to your local computer.

SSH authentication

You will need a valid SSH private key to establish an SSH session to the cluster Linux nodes. Windows credentials are stored in the API model and will be loaded from there. Set `windowsprofile.sshEnabled` to true to enable SSH in your Windows nodes.

Upload logs to a storage account container

Once the cluster logs were successfully retrieved, AKS Engine can save them on an Azure Storage Account container if optional parameter `--upload-sas-url` is set. AKS Engine expects the container name to be part of the provided SAS URL. The expected format is `https://{blob-service-uri}/{container-name}?{sas-token}`.

NOTE

Storage accounts on custom clouds using the AD FS identity provider are not yet supported.

Nodes unable to join the cluster

By default, `aks-engine get-logs` collects logs from nodes that successfully joined the cluster. To collect logs from VMs that were not able to join the cluster, set flag `--vm-names`:

```
--vm-name k8s-pool-01,k8s-pool-02
```

Usage for aks-engine get-logs

Assuming that you have a cluster deployed and the API model originally used to deploy that cluster is stored at `_output/<dnsPrefix>/apimodel.json`, then you can collect logs running a command like:

```
aks-engine get-logs \
    --location <location> \
    --api-model _output/<dnsPrefix>/apimodel.json \
    --ssh-host <dnsPrefix>.<location>.cloudapp.azure.com \
    --linux-ssh-private-key ~/.ssh/id_rsa
```

Parameters

PARAMETER	REQUIRED	DESCRIPTION
--location	Yes	Azure location of the cluster's resource group.
--api-model	Yes	Path to the generated API model for the cluster.
--ssh-host	Yes	FQDN, or IP address, of an SSH listener that can reach all nodes in the cluster.
--linux-ssh-private-key	Yes	Path to a SSH private key that can be use to create a remote session on the cluster Linux nodes.
--output-directory	No	Output directory, derived from `--api-model` if missing.
--control-plane-only	No	Only collect logs from control plane nodes.
--vm-names	No	Only collect logs from the specified VMs (comma-separated names).
--upload-sas-url	No	Azure Storage Account SAS URL to upload the collected logs.

Review custom script extension error codes

The AKS engine produces a script for each Ubuntu Server as a resource for the custom script extension (CSE) to perform deployment tasks. If the script throws an error it will log an error in `/var/log/azure/cluster-provision.log`. The errors are displayed in the portal. The error code may be helpful in figuring out the case of the problem. For more information about the CSE exit codes, see `cse_helpers.sh`.

Providing Kubernetes logs to a Microsoft support engineer

If after collecting and examining logs you still cannot resolve your issue, you may want to start the process of creating a support ticket and provide the logs that you collected.

Your operator may combine the logs you produced along with other system logs that may be needed by

Microsoft support. The operator may make them available to the Microsoft.

You can provide Kubernetes logs in several ways:

- You can contact your Azure Stack Hub operator. Your operator uses the information from the logs stored in the .ZIP file to create the support case.
- If you have the SAS URL for a storage account where you can upload your Kubernetes logs, you can include the following command and flag with the SAS URL to save the logs to the storage account:

  ```
  aks-engine get-logs -upload-sas-url <SAS-URL>
  ```

  For instructions, see Upload logs to a storage account container.
- If you're a cloud operator, you can:
  - Use the **Help + support** blade in the Azure Stack Hub Administration portal to upload logs. For instructions, see Send logs now with the administrator portal.
  - Use the **Get-AzureStackLog** PowerShell cmdlet using the Privileged End Point (PEP) For instruction, see Send logs now with PowerShell.

Open GitHub issues

If you are unable to resolve your deployment error, you can open a GitHub Issue.

1. Open a GitHub Issue in the AKS engine repository.

2. Add a title using the following format: CSE error: `exit code <INSERT_YOUR_EXIT_CODE>`.

3. Include the following information in the issue:

   - The cluster configuration file, `apimodel.json`, used to deploy the cluster. Remove all secrets and keys before posting it on GitHub.

   - The output of the following **kubectl** command `get nodes`.

   - The content of `/var/log/azure/cluster-provision.log` from an unhealthy node.

Next steps

- Read about the AKS engine on Azure Stack Hub.

Known issues with the AKS engine on Azure Stack Hub

1/21/2022 • 2 minutes to read • Edit Online

This topic covers known issues for the AKS engine on Azure Stack Hub.

Limit of 50 nodes per subscription

- **Applicable to**: Azure Stack Hub, AKS engine (all)
- **Description**: When creating clusters, you need to ensure that there are not more than 50 Kubernetes nodes (control plane and agent nodes) deployed per subscriptions. The total Kubernetes nodes deployed across all clusters within a single subscription shouldn't exceed 50 nodes.
- **Remediation**: Use less then 51 nodes in your subscription.
- **Occurrence**: When attempting to add more than 50 nodes per subscription.

Unable to resize cluster VMs with the Compute service

- **Applicable to**: Azure Stack Hub, AKS engine (all)
- **Description**: Resizing cluster VMs through the Compute service doesn't work with AKS engine. The AKS engine maintains the state of the cluster in the API model json file. To ensure that the desired VM size is reflected in any create, upgrade, or scale operation done with AKS engine you must update the API model before executing any of those operations. For example, if you change a VM size on an already deployed cluster to a different size using the Compute service, the state will be lost when `aks-engine upgrade` is executed.
- **Remediation**: To make this work locate the API model for the cluster, change the size there, and then run `aks-engine upgrade`.
- **Occurrence**: When attempting to resize using the Compute service.

Disk detach operation fails in AKS engine 0.55.0

- **Applicable to**: Azure Stack Hub (update 2005), AKS engine 0.55.0
- **Description**: When you try to delete a deployment that contains persistence volumes, the delete operation triggers a series of attach/detach errors. This is due to a bug in the AKS engine v0.55.0 Cloud Provider. The Cloud Provider calls the Azure Resource Manager using a newer version of the API than the Azure Resource Manager currently supports in Azure Stack Hub (update 2005).
- **Remediation**: You can find the details and mitigation steps in the AKS engine GitHub repository (Issue 3817). Upgrade as soon as a new build of AKS engine and corresponding image are available.
- **Occurrence**: When deleting a deployment that contains persistence volumes.

Upgrade issues in AKS engine 0.51.0

- During upgrade (aks-engine upgrade) of a Kubernetes cluster from version 1.15.x to 1.16.x, upgrade of the following kubernetes components requires extra manual steps: **kube-proxy**, **azure-cni-networkmonitor**, **csi-secrets-store**, **kubernetes-dashboard**. The following describes what you may observe and how to work around the issues.
  - In connected environments, it is not obvious to notice this issue since there are no signs in the cluster that the affected components were not upgraded. Everything appears to work as expected.

```
kubectl get pods -n kube-system
```

  - As a workaround to solve this issue for each of these components, run the command in the Workaround column in the following table.

COMPONENT NAME	WORKAROUND	AFFECTED SCENARIOS
kube-proxy	`kubectl delete ds kube-proxy -n kube-system`	Connected, Disconnected
azure-cni-networkmonitor	`kubectl delete ds azure-cni-networkmonitor -n kube-system`	Connected, Disconnected

csi-secrets-store	`sudo sed -i s/Always/IfNotPresent/g /etc/kubernetes/addons/secrets-store-csi-driver.yaml`
	`kubectl delete ds csi-secrets-store -n kube-system`
		Disconnected

kubernetes-dashboard	Run the following command on each master node: `sudo sed -i s/Always/IfNotPresent/g /etc/kubernetes/addons/kubernetes-dashboard.yaml`	Disconnected

- Kubernetes 1.17 is not supported in this release. Although there are GitHub pull requests (PR)s referencing 1.17, it is not supported.

Next steps

AKS engine on Azure Stack Hub overview