Windows PowerShell Get-Help on Cmdlet 'Set-AzNetworkInterface'

PS:\>Get-HELP Set-AzNetworkInterface -Full

WARNING: The names of some imported commands from the module 'Microsoft.Azure.PowerShell.Cmdlets.Network' include unapproved verbs that might make them less discoverable.

To find the commands with unapproved verbs, run the Import-Module command again with the Verbose parameter. For a list of approved verbs, type Get-Verb.

NAME

Set-AzNetworkInterface

SYNOPSIS

Updates a network interface.

SYNTAX

Set-AzNetworkInterface [-AsJob] [-DefaultProfile <Microsoft.Azure.Commands.Common.Authentication.Abstractions.Core.IAzureContextContainer>] -NetworkInterface <Microsoft.Azure.Commands.Network.Models.PSNetworkInterface> [<CommonParameters>]

DESCRIPTION

The Set-AzNetworkInterface updates a network interface.

PARAMETERS

-AsJob <System.Management.Automation.SwitchParameter>

Run cmdlet in the background

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-DefaultProfile <Microsoft.Azure.Commands.Common.Authentication.Abstractions.Core.IAzureContextContainer>

The credentials, account, tenant, and subscription used for communication with azure.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-NetworkInterface <Microsoft.Azure.Commands.Network.Models.PSNetworkInterface>

Specifies a network interface object representing the state to which the network interface should be set.

Required? true

Position? named

Default value None

Accept pipeline input? True (ByValue)

Accept wildcard characters? false

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug,

ErrorAction, ErrorVariable, WarningAction, WarningVariable,

OutBuffer, PipelineVariable, and OutVariable. For more information, see

about_CommonParameters (https:/go.microsoft.com/fwlink/?LinkID=113216).

INPUTS

Microsoft.Azure.Commands.Network.Models.PSNetworkInterface

OUTPUTS

Microsoft.Azure.Commands.Network.Models.PSNetworkInterface

NOTES

---------- Example 1: Configure a network interface ----------

```
$Nic = Get-AzNetworkInterface -ResourceGroupName "ResourceGroup1" -Name "NetworkInterface1"
$Nic.IpConfigurations[0].PrivateIpAddress = "10.0.1.20"
$Nic.IpConfigurations[0].PrivateIpAllocationMethod = "Static"
$Nic.Tag = @{Name = "Name"; Value = "Value"}
Set-AzNetworkInterface -NetworkInterface $Nic
```

This example configures a network interface. The first command gets a network interface named NetworkInterface1 in resource group ResourceGroup1. The second command

sets the private IP address of the IP configuration. The third command sets the private IP allocation method to Static. The fourth command sets a tag on the network

interface. The fifth command uses the information stored in the $Nic variable to set the network interface.

---- Example 2: Change DNS settings on a network interface ----

```
$nic = Get-AzNetworkInterface -ResourceGroupName "ResourceGroup1" -Name "NetworkInterface1"
$nic.DnsSettings.DnsServers.Add("192.168.1.100")
$nic | Set-AzNetworkInterface
```

The first command gets a network interface named NetworkInterface1 that exists within resource group ResourceGroup1. The second command adds DNS server 192.168.1.100
to this interface. The third command applies these changes to the network interface. To remove a DNS server, follow the commands listed above, but replace ".Add" with
".Remove" in the second command.

---- Example 3: Enable IP forwarding on a network interface ----

```
$nic = Get-AzNetworkInterface -ResourceGroupName "ResourceGroup1" -Name "NetworkInterface1"
$nic.EnableIPForwarding = 1
$nic | Set-AzNetworkInterface
```

The first command gets an existing network interface called NetworkInterface1 and stores it in the $nic variable. The second command changes the IP forwarding value
to true. Finally, the third command applies the changes to the network interface. To disable IP forwarding on a network interface, follow the sample example, but be
sure to change the second command to "$nic.EnableIPForwarding = 0".

----- Example 4: Change the subnet of a network interface -----

```
$nic = Get-AzNetworkInterface -ResourceGroupName "ResourceGroup1" -Name "NetworkInterface1"
$vnet = Get-AzVirtualNetwork -Name VNet1 -ResourceGroupName crosssubcrossversionpeering
$subnet2 = Get-AzVirtualNetworkSubnetConfig -Name Subnet2 -VirtualNetwork $vnet
$nic.IpConfigurations[0].Subnet.Id = $subnet2.Id
$nic | Set-AzNetworkInterface
```

The first command gets the network interface NetworkInterface1 and stores it in the $nic variable. The second command gets the virtual network associated with the

subnet that the network interface is going to be associated with. The second command gets the subnet and stores it in the $subnet2 variable. The third command

associated the primary private IP address of the network interface with the new subnet. Finally the last command applied these changes on the network interface.

>[!NOTE] >The IP configurations must be dynamic before you can change the subnet. If you have static IP configurations, change then to dynamic before proceeding.

>[!NOTE] >If the network interface has multiple IP configurations, the fourth command must be done for all these IP configurations before the final

Set-AzNetworkInterface command is executed. This can be done as in the fourth command but by replacing "0" with the appropriate number. If a network interface has N

IP configurations, then N-1 of these commands must exist.

Example 5: Associate/Dissociate a Network Security Group to a network interface

```
$nic = Get-AzNetworkInterface -ResourceGroupName "ResourceGroup1" -Name "NetworkInterface1"
$nsg = Get-AzNetworkSecurityGroup -ResourceGroupName "ResourceGroup1" -Name "MyNSG"
$nic.NetworkSecurityGroup = $nsg
$nic | Set-AzNetworkInterface
```

The first command gets an existing network interface called NetworkInterface1 and stores it in the $nic variable. The second command gets an existing network security

group called MyNSG and stores it in the $nsg variable. The third command assigns the $nsg to the $nic. Finally, the fourth command applies the changes to the Network

interface. To dissociate network security groups from a network interface, simple replace $nsg in the third command with $null.

RELATED LINKS

Online Version: https://learn.microsoft.com/powershell/module/az.network/set-aznetworkinterface

Get-AzNetworkInterface

Get-AzNetworkInterface

New-AzNetworkInterface

Remove-AzNetworkInterface