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

PS:\>Get-HELP Set-AzFirewall -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-AzFirewall

SYNOPSIS

Saves a modified Firewall.

SYNTAX

Set-AzFirewall [-AsJob] -AzureFirewall <Microsoft.Azure.Commands.Network.Models.PSAzureFirewall> [-DefaultProfile <Microsoft.Azure.Commands.Common.Authentication.Abstractions.Core.IAzureContextContainer>] [-Confirm] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The Set-AzFirewall cmdlet updates an Azure Firewall.

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

-AzureFirewall <Microsoft.Azure.Commands.Network.Models.PSAzureFirewall>

The AzureFirewall

Required? true

Position? named

Default value None

Accept pipeline input? True (ByValue)

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

-Confirm <System.Management.Automation.SwitchParameter>

Prompts you for confirmation before running the cmdlet.

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-WhatIf <System.Management.Automation.SwitchParameter>

Shows what would happen if the cmdlet runs. The cmdlet is not run.

Required? false

Position? named

Default value False

Accept pipeline input? False

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.PSAzureFirewall

OUTPUTS

Microsoft.Azure.Commands.Network.Models.PSAzureFirewall

NOTES

1: Update priority of a Firewall application rule collection

```
$azFw = Get-AzFirewall -Name "AzureFirewall" -ResourceGroupName "rg"
$ruleCollection = $azFw.GetApplicationRuleCollectionByName("ruleCollectionName")
$ruleCollection.Priority = 101
Set-AzFirewall -AzureFirewall $azFw
```

This example updates the priority of an existing rule collection of an Azure Firewall. Assuming Azure Firewall "AzureFirewall" in resource group "rg" contains an application rule collection named "ruleCollectionName", the commands above will change the priority of that rule collection and update the Azure Firewall afterwards. Without the Set-AzFirewall command, all operations performed on the local $azFw object are not reflected on the server.

2: Create a Azure Firewall and set an application rule collection later

```
$azFw = New-AzFirewall -Name "AzureFirewall" -ResourceGroupName "rg" -VirtualNetworkName "vnet-name" -PublicIpName "pip-name"

$rule = New-AzFirewallApplicationRule -Name R1 -Protocol "http:80","https:443" -TargetFqdn "*google.com", "*microsoft.com" -SourceAddress "10.0.0.0"
$RuleCollection = New-AzFirewallApplicationRuleCollection -Name RC1 -Priority 100 -Rule $rule -ActionType "Allow"
$azFw.ApplicationRuleCollections = $RuleCollection

$azFw | Set-AzFirewall
```

In this example, a Firewall is created first without any application rule collections. Afterwards a Application Rule and Application Rule Collection are created, then the Firewall object is modified in memory, without affecting the real configuration in cloud. For changes to be reflected in cloud, Set-AzFirewall must be

called.

--- 3: Update Threat Intel operation mode of Azure Firewall ---

```
$azFw = Get-AzFirewall -Name "AzureFirewall" -ResourceGroupName "rg"
$azFw.ThreatIntelMode = "Deny"
Set-AzFirewall -AzureFirewall $azFw
```

This example updates the Threat Intel operation mode of Azure Firewall "AzureFirewall" in resource group "rg". Without the Set-AzFirewall command, all operations
performed on the local $azFw object are not reflected on the server.

---------- 4: Deallocate and allocate the Firewall ----------

```
$firewall=Get-AzFirewall -ResourceGroupName rgName -Name azFw
$firewall.Deallocate()
$firewall | Set-AzFirewall

$vnet = Get-AzVirtualNetwork -ResourceGroupName rgName -Name anotherVNetName
$pip = Get-AzPublicIpAddress -ResourceGroupName rgName -Name publicIpName
$firewall.Allocate($vnet, $pip)
$firewall | Set-AzFirewall
```

This example retrieves a Firewall, deallocates the firewall, and saves it. The Deallocate command removes the running service but preserves the firewall's
configuration. For changes to be reflected in cloud, Set-AzFirewall must be called. If user wants to start the service again, the Allocate method should be called on
the firewall. The new VNet and Public IP must be in the same resource group as the Firewall. Again, for changes to be reflected in cloud, Set-AzFirewall must be
called.

5: Allocate with a management public IP address for forced tunneling scenarios

```
$vnet = Get-AzVirtualNetwork -ResourceGroupName rgName -Name anotherVNetName
$pip = Get-AzPublicIpAddress -ResourceGroupName rgName -Name publicIpName
$mgmtPip = Get-AzPublicIpAddress -ResourceGroupName rgName -Name MgmtPublicIpName
$firewall.Allocate($vnet, $pip, $mgmtPip)
$firewall | Set-AzFirewall
```

This example allocates the firewall with a management public IP address and subnet for forced tunneling scenarios. The VNet must contain a subnet called
"AzureFirewallManagementSubnet".

------- 6: Add a Public IP address to an Azure Firewall -------

```
$pip = New-AzPublicIpAddress -Name "azFwPublicIp1" -ResourceGroupName "rg" -Sku "Standard" -Location "centralus" -AllocationMethod Static
$azFw = Get-AzFirewall -Name "AzureFirewall" -ResourceGroupName "rg"
$azFw.AddPublicIpAddress($pip)

$azFw | Set-AzFirewall
```

In this example, the Public IP Address "azFwPublicIp1" as attached to the Firewall.

----- 7: Remove a Public IP address from an Azure Firewall -----

```
$pip = Get-AzPublicIpAddress -Name "azFwPublicIp1" -ResourceGroupName "rg"
$azFw = Get-AzFirewall -Name "AzureFirewall" -ResourceGroupName "rg"
$azFw.RemovePublicIpAddress($pip)

$azFw | Set-AzFirewall
```

In this example, the Public IP Address "azFwPublicIp1" as detached from the Firewall.

8: Change the management public IP address on an Azure Firewall

```
$newMgmtPip = New-AzPublicIpAddress -Name "azFwMgmtPublicIp2" -ResourceGroupName "rg" -Sku "Standard" -Location "centralus" -AllocationMethod Static
$azFw = Get-AzFirewall -Name "AzureFirewall" -ResourceGroupName "rg"
$azFw.ManagementIpConfiguration.PublicIpAddress = $newMgmtPip

$azFw | Set-AzFirewall
```

In this example, the management public IP address of the firewall will be changed to "AzFwMgmtPublicIp2"

-------- 9: Add DNS configuration to an Azure Firewall --------

```
$dnsServers = @("10.10.10.1", "20.20.20.2")
$azFw = Get-AzFirewall -Name "AzureFirewall" -ResourceGroupName "rg"
$azFw.DNSEnableProxy = $true
$azFw.DNSServer = $dnsServers

$azFw | Set-AzFirewall
```

In this example, DNS Proxy and DNS Server configuration is attached to the Firewall.

10: Update destination of an existing rule within a Firewall application rule collection

```
$azFw = Get-AzFirewall -Name "AzureFirewall" -ResourceGroupName "rg"
$ruleCollection = $azFw.GetNetworkRuleCollectionByName("ruleCollectionName")
$rule=$ruleCollection.GetRuleByName("ruleName")
```

```
$rule.DestinationAddresses = "10.10.10.10"
Set-AzFirewall -AzureFirewall $azFw
```

This example updates the destination of an existing rule within a rule collection of an Azure Firewall. This allows you to automatically update your rules when IP addresses change dynamically.

---------- 11: Allow Active FTP on Azure Firewall ----------

```
$azFw = Get-AzFirewall -Name "AzureFirewall" -ResourceGroupName "rg"
$azFw.AllowActiveFTP = $true

$azFw | Set-AzFirewall
```

In this example, Active FTP is allowed on the Firewall.

- 12: Deallocate and allocate the Firewall from a Virtual Hub -

```
$firewall=Get-AzFirewall -ResourceGroupName rgName -Name azFw
$firewall.Deallocate()
$firewall | Set-AzFirewall

$Hub = Get-AzVirtualHub -ResourceGroupName "testRG" -Name "westushub"
$firewall.Allocate($Hub.Id)
$firewall | Set-AzFirewall
```

This example retrieves a Hub Firewall, deallocates the hub firewall, and saves it. The Deallocate command removes the reference to the virtual hub but preserves the firewall's configuration. For changes to be reflected in cloud, Set-AzFirewall must be called. The Allocate method assigns the virtual hub reference to the firewall.

Again, for changes to be reflected in cloud, Set-AzFirewall must be called.

-------- 13: Enable Fat Flow Logging on Azure Firewall --------

```
$azFw = Get-AzFirewall -Name "ps184" -ResourceGroupName "ps774"
$azFw.EnableFatFlowLogging = $true

$azFw | Set-AzFirewall

AllowActiveFTP                  : null
  ApplicationRuleCollections     : Count = 0
  ApplicationRuleCollectionsText : "[]"
  DNSEnableProxy                 : null
  DNSServer                      : null
  DNSServersText                 : "null"
  Etag                           : "W/\"7533fa1b-8588-400d-857c-6bc372e14f1b\""
  FirewallPolicy                 : null
  HubIPAddresses                 : null
  Id                             :
"/subscriptions/aeb5b02a-0f18-45a4-86d6-81808115cacf/resourceGroups/ps774/providers/Microsoft.Network/azureFirewalls/ps184"
  EnableFatFlowLogging           : "true"
  IpConfigurations               : Count = 0
  IpConfigurationsText           : "[]"
  Location                       : "eastus"
  ManagementIpConfiguration      : null
  ManagementIpConfigurationText : "null"
  Name                           : "ps184"
  NatRuleCollections             : Count = 0
  NatRuleCollectionsText         : "[]"
  NetworkRuleCollections         : Count = 0
  NetworkRuleCollectionsText     : "[]"
  PrivateRange                   : null
```

```
PrivateRangeText            : "null"
ProvisioningState           : "Succeeded"
ResourceGroupName           : "ps774"
ResourceGuid                : null
Sku                         : {Microsoft.Azure.Commands.Network.Models.PSAzureFirewallSku}
Tag                         : null
TagsTable                   : null
ThreatIntelMode             : "Alert"
ThreatIntelWhitelist        : {Microsoft.Azure.Commands.Network.Models.PSAzureFirewallThreatIntelWhitelist}
ThreatIntelWhitelistText    : "{\r\n  \"FQDNs\": null,\r\n  \"IpAddresses\": null\r\n}"
Type                        : "Microsoft.Network/azureFirewalls"
VirtualHub                  : null
Zones                       : Count = 0
privateRange                : null
```

In this example, Enable Fat Flow Logging is enabled on the Firewall.

-------- 14: Upgrade Azure Firewall Standard to Premium --------

```
$azfw = Get-AzFirewall -Name "AzureFirewall" -ResourceGroupName "rg"
$azfw.Sku.Tier="Premium"
Set-AzFirewall -AzureFirewall $azfw
```

This example upgrades your existing Azure Firewall Standard to Premium Firewall. Upgrade process may take several minutes and does not require service down time.

After upgrade is completed successfully you may replace your exiting standard policy with premium.

15: Deallocate and allocate the Firewall with Availability Zones

```
$firewall=Get-AzFirewall -ResourceGroupName rgName -Name azFw
$firewall.Deallocate()
```

```
$firewall | Set-AzFirewall

$vnet = Get-AzVirtualNetwork -ResourceGroupName rgName -Name anotherVNetName
$pip = Get-AzPublicIpAddress -ResourceGroupName rgName -Name publicIpName
$firewall.Zones = "1","2","3"
$firewall.Allocate($vnet, $pip)
$firewall | Set-AzFirewall
```

This example retrieves a Firewall, deallocates the firewall, and saves it. The Deallocate command removes the running service but preserves the firewall's configuration. For changes to be reflected in cloud, Set-AzFirewall must be called. If user wants to start the service again but with Availability Zones, the Zones method needs to be called defining the desired Availability Zones in quotes and separated by comma. In case Availability Zones needs to be removed, the $null parameter needs to be introduced instead. Finally, the Allocate method should be called on the firewall. The new VNet and Public IP must be in the same resource group as the Firewall. Again, for changes to be reflected in cloud, Set-AzFirewall must be called.

RELATED LINKS

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

Get-AzFirewall

New-AzFirewall

Remove-AzFirewall