Windows PowerShell Get-Help on Cmdlet 'New-CIPolicyRule'

PS:\>Get-HELP New-CIPolicyRule -Full

NAME

New-CIPolicyRule

SYNOPSIS

Generates Code Integrity policy rules for user mode code and drivers.

SYNTAX

New-CIPolicyRule [-AllowFileNameFallbacks] [-AppID <String>] [-Deny] -DriverFilePath <String[]> [-Fallback {None | Hash | FileName | FilePath | SignedVersion | PFN | Publisher | FilePublisher | LeafCertificate | PcaCertificate | RootCertificate | WHQL | WHQLPublisher | WHQLFilePublisher}] -Level {None | Hash | FileName | FilePath | SignedVersion | PFN | Publisher | FilePublisher | LeafCertificate | PcaCertificate | RootCertificate | WHQL | WHQLPublisher | WHQLFilePublisher} [-ScriptFileNames] [-SpecificFileNameLevel {None | OriginalFileName | InternalName | FileDescription | ProductName | PackageFamilyName}] [-UserWriteablePaths] [<CommonParameters>]

New-CIPolicyRule [-AllowFileNameFallbacks] [-Deny] [-DriverFiles <DriverFile[]>] [-Fallback {None | Hash | FileName | FilePath | SignedVersion | PFN | Publisher | FilePublisher | LeafCertificate | PcaCertificate | RootCertificate | WHQL | WHQLPublisher | WHQLFilePublisher}] -Level

{None | Hash | FileName | FilePath |

SignedVersion | PFN | Publisher | FilePublisher | LeafCertificate | PcaCertificate | RootCertificate | WHQL | WHQLPublisher | WHQLFilePublisher} [-ScriptFileNames]

[-SpecificFileNameLevel {None | OriginalFileName | InternalName | FileDescription | ProductName | PackageFamilyName}] [-UserWriteablePaths] [<CommonParameters>]

New-CIPolicyRule [-AllowFileNameFallbacks] [-Deny] [-Fallback {None | Hash | FileName | FilePath | SignedVersion | PFN | Publisher | FilePublisher | LeafCertificate |

PcaCertificate | RootCertificate | WHQL | WHQLPublisher | WHQLFilePublisher}] [-FilePathRule <String>] [-ScriptFileNames] [-SpecificFileNameLevel {None |

OriginalFileName | InternalName | FileDescription | ProductName | PackageFamilyName}] [-UserWriteablePaths] [<CommonParameters>]

New-CIPolicyRule [-AllowFileNameFallbacks] [-Deny] [-Fallback {None | Hash | FileName | FilePath | SignedVersion | PFN | Publisher | FilePublisher | LeafCertificate |

PcaCertificate | RootCertificate | WHQL | WHQLPublisher | WHQLFilePublisher}] [-Package <AppxPackage>] [-ScriptFileNames] [-SpecificFileNameLevel {None |

OriginalFileName | InternalName | FileDescription | ProductName | PackageFamilyName}] [-UserWriteablePaths] [<CommonParameters>]

DESCRIPTION

The New-CIPolicyRule cmdlet generates Code Integrity policy rules for drivers. Specify a rule level and an array of DriverFile objects or the path of a driver.

PARAMETERS

-AllowFileNameFallbacks [<SwitchParameter>]

Indicates that files that do not have an `OriginalFileName` fall back in the following order:

- InternalName

- FileDescription

- ProductName

Required? false
Position? named
Default value False
Accept pipeline input? False
Accept wildcard characters? false

-AppID <String>

Specifies an app. This cmdlet creates per-app rules which control whether specific plug-ins, add-ins, and modules can run from specific apps.

For more information, see Use a Windows Defender Application Control policy to control specific plug-ins, add-ins, and modules (/windows/security/threat-protection

/windows-defender-application-control/use-windows-defender-application-control-policy-to-control-specific-plug-ins-add-ins-and-modules).

Required? false
Position? named
Default value None
Accept pipeline input? False
Accept wildcard characters? false

-Deny [<SwitchParameter>]

Indicates that this cmdlet creates deny rules instead of the default allow rules.

Required? false
Position? named
Default value False
Accept pipeline input? False
Accept wildcard characters? false

-DriverFilePath <String[]>

Specifies the path of a driver on which this cmdlet bases a rule.

Required? true

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-DriverFiles <DriverFile[]>

Specifies an array of DriverFile objects on which this cmdlet bases rules. To obtain a driver file, use the Get-SystemDriver cmdlet.

Required? false

Position? named

Default value None

Accept pipeline input? True (ByValue)

Accept wildcard characters? false

-Fallback <RuleLevel[]>

Specifies an array of levels of detail for generated rules. If this cmdlet cannot generate a rule at the specified level, this cmdlet attempts to generate it at a

fallback level. The acceptable values for this parameter are the same as for the Level parameter. If you specify multiple fallback levels, this cmdlet tries them

in order.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-FilePathRule <String>

Specifies the path of a folder for generating a rule with level set to FilePath. Refer to Filepath Rules Info

(/windows/security/threat-protection/windows-defender-application-control/select-types-of-rules-to-create#more-information-about-filepath-rules)for acceptable

wildcard values and usage. This cmdlet will not check whether the filepath string is a valid filepath.

Required? false

Position? named

Default value None

Accept pipeline input? True

Accept wildcard characters? true

-Level <RuleLevel>

Specifies the primary level of detail for generated rules. Refer to WDAC File Rule Levels (/windows/security/threat-protection/windows-defender-application-control

/select-types-of-rules-to-create#windows-defender-application-control-file-rule-levels)for acceptable parameter values and descriptions.

Required? true

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-Package <AppxPackage>

Specifies the packaged app (MSIX/Appx) to base the rule.

Required? false

Position? named

Default value None

Accept pipeline input? True (ByValue)

Accept wildcard characters? false

-ScriptFileNames [<SwitchParameter>]

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-SpecificFileNameLevel <FileNameLevel>

Specifies the attribute of the file off which to base a file name rule. The -Level must be set to FileName for this option. Refer to File Name Rules Info

(/windows/security/threat-protection/windows-defender-application-control/select-types-of-rules-to-create#windows-defender-application-control-filename-rules)for

a description of the acceptable values.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-UserWriteablePaths [<SwitchParameter>]

Indicates that this cmdlet includes files identified as user writeable in the policy.

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

OUTPUTS

Rule

This cmdlet returns the rules that it creates.

NOTES

---------- Example 1: Create policy rules for drivers ----------

PS C:\> $DriverFiles = Get-SystemDriver -ScanPath '.\temp\' -UserPEs -OmitPaths '.\temp\ConfigCITestBinaries' -NoScript

PS C:\> New-CIPolicyRule -Level FileName -DriverFiles $DriverFiles

Scan completed successfully

Name : \\?\E:\cmdlets\temp\Microsoft.ConfigCI.Commands.dll FileRule

Id : ID_ALLOW_A_1

TypeId : Allow

Root :

FileVersionRef :

Wellknown : False

Ekus :

```
Exceptions     :
FileAttributes :
FileException  : False
UserMode       : False

Name           : \\?\E:\cmdlets\temp\Microsoft.ConfigCI.Commands.Tests.dll FileRule
Id             : ID_ALLOW_A_3
TypeId         : Allow
Root           :
FileVersionRef :
Wellknown      : False
Ekus           :
Exceptions     :
FileAttributes :
FileException  : False
UserMode       : False

Name           : \\?\E:\cmdlets\temp\Microsoft.PackageInspector.Tests.dll FileRule
Id             : ID_ALLOW_A_5
TypeId         : Allow
Root           :
FileVersionRef :
Wellknown      : False
Ekus           :
Exceptions     :
FileAttributes :
FileException  : False
UserMode       : False
```

The first command gets drivers by using the Get-SystemDriver cmdlet, and then stores them in the $DriverFiles variable.

The second command creates policy rules at the file name level for the drivers in $DriverList. For this example, we present only the first few rules.

Example 2: Create policy rules for drivers and include a fallback value

```
PS C:\> New-CIPolicyRule -Level Publisher -Fallback Hash -DriverFiles $DriverFiles
"Scan completed successfully"



Name          : \\?\E:\cmdlets\temp\Microsoft.ConfigCI.Commands.dll Hash Sha1
Id            : ID_ALLOW_A_F
TypeId        : Allow
Root          :
FileVersionRef :
Wellknown     : False
Ekus          :
Exceptions    :
FileAttributes :
FileException  : False
UserMode       : False

Name          : \\?\E:\cmdlets\temp\Microsoft.ConfigCI.Commands.dll Hash Sha256
Id            : ID_ALLOW_A_10
TypeId        : Allow
Root          :
FileVersionRef :
Wellknown     : False
Ekus          :
Exceptions    :
FileAttributes :
FileException  : False
UserMode       : False

Name          : \\?\E:\cmdlets\temp\Microsoft.ConfigCI.Commands.dll Hash Page Sha1
Id            : ID_ALLOW_A_11
TypeId        : Allow
```

Root :

FileVersionRef :

Wellknown : False

Ekus :

Exceptions :

FileAttributes :

FileException : False

UserMode : False

This command generates rule at the Publisher level for the same drivers from the previous example. For files that are unsigned, the cmdlet creates Hash rules, as a

fallback. For this example, we present only the first few rules.

--- Example 3: Specify a policy rule for a kernel component ---

PS C:\> New-CIPolicyRule -DriverFilePath '.\temp\ConfigCITestBinaries\ci.dll' -Level Publisher

Scan completed successfully

Name : MSIT Test CodeSign CA 3

Id : ID_SIGNER_S_B

TypeId : Allow

Root : FA6B9A2230CE08BCA81D096B28CF495672401D3A43A0D285CF352464A6C9C7FD

FileVersionRef :

Wellknown : False

Ekus :

Exceptions :

FileAttributes :

FileException : False

UserMode : False

Name : MSIT Test CodeSign CA 3

Id : ID_SIGNER_S_C

TypeId : Allow

Root : FA6B9A2230CE08BCA81D096B28CF495672401D3A43A0D285CF352464A6C9C7FD

FileVersionRef :

Wellknown : False

Ekus :

Exceptions :

FileAttributes :

FileException : False

UserMode : True

This command generates a publisher rule for the specific file named ci.dll. The file ci.dll is a kernel component. Therefore, the cmdlet generates both a kernel rule

and a user mode rule.

Example 4: Specify a policy rule for a folder path with wildcards

PS C:\> New-CIPolicyRule -FilePathRule '.\temp\ConfigCITestBinaries*'

Name : .\temp\ConfigCITestBinaries* FileRule

Id : ID_ALLOW_A_1

TypeId : Allow

Root :

FileVersionRef :

AppIDRef :

Wellknown : False

Ekus :

Exceptions :

FileAttributes :

FileException : False

UserMode : True

attributes : {[AppIDs,], [MinimumFileVersion, 0.0.0.0], [FilePath, .\temp\ConfigCITestBinaries*]}

This command generates a filepath rule for the specific path verbatim string. This will allow anything in the parent folder.

Example 5: Create a policy rule for a packaged application and its dependencies

```
PS C:\> $packages = Get-AppxPackage -Name *Microsoft*
PS C:\> $packages

Name              : Microsoft.NET.Native.Runtime.1.4
Publisher         : CN=Microsoft Corporation, O=Microsoft Corporation, L=Redmond, S=Washington, C=US
Architecture      : X86
ResourceId        :
Version           : 1.4.24201.0
PackageFullName   : Microsoft.NET.Native.Runtime.1.4_1.4.24201.0_x86__8wekyb3d8bbwe
InstallLocation   : C:\Program Files\WindowsApps\Microsoft.NET.Native.Runtime.1.4_1.4.24201.0_x86__8wekyb3d8bbwe
IsFramework       : True
PackageFamilyName : Microsoft.NET.Native.Runtime.1.4_8wekyb3d8bbwe
PublisherId       : 8wekyb3d8bbwe
IsResourcePackage : False
IsBundle          : False
IsDevelopmentMode : False
NonRemovable      : False
IsPartiallyStaged : False
SignatureKind     : Store
Status            : Ok
...
Name              : Microsoft.NET.Native.Runtime.1.4
Publisher         : CN=Microsoft Corporation, O=Microsoft Corporation, L=Redmond, S=Washington, C=US
Architecture      : X64
ResourceId        :
Version           : 1.4.24201.0
PackageFullName   : Microsoft.NET.Native.Runtime.1.4_1.4.24201.0_x64__8wekyb3d8bbwe
InstallLocation   : C:\Program Files\WindowsApps\Microsoft.NET.Native.Runtime.1.4_1.4.24201.0_x64__8wekyb3d8bbwe
IsFramework       : True
PackageFamilyName : Microsoft.NET.Native.Runtime.1.4_8wekyb3d8bbwe
```

PublisherId : 8wekyb3d8bbwe

IsResourcePackage : False

IsBundle : False

IsDevelopmentMode : False

NonRemovable : False

IsPartiallyStaged : False

SignatureKind : Store

Status : Ok

$package_dependencies = $packages.Dependencies

$package_rule = New-CIPolicyRule -Package $packages[0] #repeat for all desired packages in the array

$package_rule += New-CIPolicyRule -Package $package_dependencies[0] # repeat for all dependencies in the array

$package_rule

Name : Microsoft.NET.Native.Runtime.1.4_8wekyb3d8bbwe FileRule

Id : ID_ALLOW_A_1

TypeId : Allow

Root :

FileVersionRef :

AppIDRef :

Wellknown : False

Ekus :

Exceptions :

FileAttributes :

FileException : False

UserMode : True

attributes : {[AppIDs,], [MinimumFileVersion, 0.0.0.0], [PackageFamilyName,

Microsoft.NET.Native.Runtime.1.4_8wekyb3d8bbwe], [PackageVersion, 1.4.24201.0]}

Name : Microsoft.NET.Native.Framework.2.2_8wekyb3d8bbwe FileRule

Id : ID_ALLOW_A_2

TypeId : Allow

Root :

FileVersionRef :

AppIDRef :

Wellknown : False

Ekus :

Exceptions :

FileAttributes :

FileException : False

UserMode : True

attributes : {[AppIDs,], [MinimumFileVersion, 0.0.0.0], [PackageFamilyName,
Microsoft.NET.Native.Framework.2.2_8wekyb3d8bbwe], [PackageVersion, 2.2.29512.0]}

This set of commands finds a packaged application matching the specified name and generates an allow rule for the packaged application and its dependencies.

RELATED LINKS

Online Version:
https://learn.microsoft.com/powershell/module/configci/new-cipolicyrule?view=windowsserver2022-ps&wt.mc_id=ps-gethelp

Get-SystemDriver