Defendpoint API Reference Guide

Software Version: 5.2.21.0 GA

Document Version: 1.0

Document Date: August 2018

Table of Contents

Chapter 1 - Defendpoint Powershell API Reference Guide

1.1 - Introduction

The Avecto Defendpoint PowerShell API enables administrators to configure Defendpoint using PowerShell scripts. This enables integrations with external systems, and provides an alternative to using the Avecto Management Console.

1.2 - Prerequisites

Before running any other cmdlets, run the following two commands to initialize the Avecto cmdlets:

```
Import-Module 'C:/Program Files/Avecto/Privilege Guard Client/PowerShell/Avecto.Defendpoint.Cmdlets/Avecto.Defendpoint.Cmdlets.dll'

Import-Module 'C:/Program Files/Avecto/Privilege Guard Client/PowerShell/Avecto.Defendpoint.Cmdlets/Avecto.Defendpoint.Settings.dll'
```

Chapter 2 - Get Information about a DefendpointSettings File: `Get-DefendpointFileInformation`

`Get-DefendpointFileInformation` - Get information from one or more files for constructing Defendpoint settings.

2.1 - Overview

2.1.1 - Syntax

```
Get-DefendpointFileInformation [-Path '/path/to/file'] [-
Directory '/path/to/directory' [-Recurse]] [-FileType
'ApplicationType.$FileType'] [-COM]
```

2.1.2 - Description

The `Get-DefendpointFileInformation` cmdlet retrieves information from a file, or list of files. This information can then be used to construct Defendpoint settings. This cmdlet supports Common Parameters.

2.2 - Parameters

Parameter	Type	Description	Required
`Path`	String	Define the path of the file to investigate. This parameter supports regular expressions (regex). This parameter can be piped.	Yes if `Directory` is not set.
`Directory`	String	Define the directory containing multiple files to investigate. To search all subfolders and files in the directories, include the Recurse parameter.	Yes if `Path` is not set.
`Recurse`	Boolean	Search all subfolders and files in the directory/directories specified by the `-Directory` parameter.	
`COM`	Boolean	Returns registered COM objects stored on the local machine.	
`EncodeUnicodeChars`	Boolean	Encode any Unicode characters in strings within the object.	
`FileType`	String	Define which file type to search for. Only one file type can be supplied for each command. See below for a list of possible file types.	

2.2.1 - File Types

Input String	File Type
ActiveXControl	Active X
BatchFile	Batch File
COMClass	COM Class

Input String	File Type
Content	File Resource
ControlPanelApplet	Control Panel
Dll	DLL
Executable	Executable
InstallerPackage	Installer Package
ManagementConsoleSnapin	MMC Snap In
PowerShellScript	PowerShell Script
RegistrySettings	Registry Settings
RemotePowerShellCommand	Remote PowerShell Command
RemotePowerShellScript	Remote PowerShell Script
Service	Service
Unistaller	Uninstaller
Url	URL
WindowsScript	Windows Script
WindowsStoreApplication	AppX Package

2.3 - Return Values

`Get-DefendpointFileInformation` returns a list of application definitions.

2.4 - Examples

2.4.1 - Get all executables in a specified path

```
Get-DefendpointFileInformation –Path “C:\Program Files\Internet Explorer\*.exe”
```

The above example outputs the following:

```
...
RegistryKeyNameUseRegExpression       : False
RegistryKeyNameStringMatchType        : Contains
AvectoZoneIdentifierExists            : False
AvectoZoneIdentifierNegateMatch       : False
ID                                    : 299a0645-4521-4336-8e90-66aca0b2720e
...
```

2.4.2 - Get all application files in a specified directory

```
Get-DefendpointFileInformation -Directory "C:\Program
Files\Internet Explorer\"
```

The above example outputs the following:

```
...
CheckMaxFileVersion                    : False
MinFileVersion                         : 11.00.14393.2007 (rs1_
release.171231-1800)
MaxFileVersion                         : 11.00.14393.2007 (rs1_
release.171231-1800)
FileVersionNegateMatch                 : False
CheckURL                               : False
...
```

2.4.3 - Get all registered COM objects

```
Get-DefendpointFileInformation -COM
```

The above example outputs the following:

```
...
RegistryKeyNameMatchCase               : False
RegistryKeyNamePatternMatching         : False
RegistryKeyNameUseRegExpression        : False
RegistryKeyNameStringMatchType         : Contains
CheckURL                               : False
...
```

2.4.4 - Recursively search for all batch files within a directory

```
Get-DefendpointFileInformation -Directory
"C:/Users/admin/Desktop" -FileType "BatchFile" -Recurse
```

The above example outputs the following:

```
...
ParentApplicationGroupAsText           :
Type                                   : BatchFile
Description                            : test_batch.bat
ChildrenInheritToken                   : False
OpenDlgDropRights                      : False
CheckElevationIsRequired               : False
...
```

Chapter 3 - Retrieve the Defendpoint Settings: `Get-DefendpointSettings`

`Get-DefendpointSettings` - Retrieve the Defendpoint settings from the local network or a domain.

3.1 - Overview

3.1.1 - Syntax

```
Get-DefendpointSettings [-LocalPolicy] [-LocalFile -FileLocation 'path/to/file'] [-UserPolicy] [-Domain -LDAP 'path/to/LDAP'] [-XML]
```

3.1.2 - Description

The `Get-DefendpointSettings` cmdlet gets the Defendpoint settings either from the Local Group Policy, from a specified Group Policy Object (GPO), or from a specified XML file. The output is a DefendpointSettings object or an XML- formatted string, depending on the parameters supplied. This cmdlet supports Common Parameters.

3.2 - Parameters

Parameter	Type	Description	Required
`LocalPolicy`	Boolean	Return the Defendpoint settings object from the local group policy.	Yes
`LocalFile`	Boolean	Return the Defendpoint settings from a Defendpoint settings file. Specify the file using the `-FileLocation` parameter. If `-FileLocation` is not set the cmdlet returns an error.	Yes
`FileLocation`	String	Specify the location of the Defendpoint settings file. This cmdlet defaults to `%PROGRAMDATA%\Avecto\Privilege Guard\PrivilegeGuardConfig.xml` if a file path is not supplied when the `-LocalFile` parameter is used.	Yes if `LocalFile` is used
`UserPolicy`	Boolean	Return the policy of a user. This cmdlet defaults to a machine policy if this parameter is not used.	
`Domain`	Boolean	Return the Defendpoint settings from the Group Policy Object (GPO) specified by the `-LDAP` parameter.	
`LDAP`	String	Specify the LDAP path of the Group Policy Object (GPO). This parameter must be used in conjunction with `-Domain`.	Yes if `Domain` is used
`XML`	Boolean	Return the Defendpoint settings as an XML formatted string.	

3.3 - Return Values

By default, `Get-DefendpointSettings` returns a DefendpointSettings object. Using the `-XML` parameter returns the Defendpoint policy as an XML formatted string. The cmdlet returns errors if there are any.

3.4 - Examples

3.4.1 - Get the local Defendpoint settings as a DefendpontSettings object

```
Get-DefendpointSettings -LocalPolicy
```

The above example outputs:

```
Version            : 5.2.102.0
ID                 : 1e71ef8e-4ffc-4769-9a5b-11ea102b0f8e
ConfigRevision     : 510
ApplicationGroups  : {cmd}
ContentGroups      : {}
URLGroups          : {}
Tokens             : {}
GlobalOptionsSets  : {}
Files              : Avecto.Defendpoint.Settings.FileList
Messages           : {Block Message, Allow Message (Elevate)}
Policies           : {New Workstyle}
Licenses           : {Avecto.Defendpoint.Settings.License}
RegistryValues     : {}
```

3.4.2 - Save the local Defendpoint settings into an XML file

```
Get-DefendpointSettings -LocalPolicy -XML >
C:/Users/admin/Desktop/DefendpointSettings.xml
```

The above example does not output anything to the terminal. A file called `DefendpointSettings.xml` is created at `C:/Users/admin/Desktop`.

3.4.3 - Get the Defendpoint settings of the Group Policy Object specified by an LDAP path

```
Get-DefendpointSettings -Domain -LDAP "LDAP://DC13.Acme.com/CN=
{31B2F340-016D-11D2-945F-
00C04FB984F9},CN=Policies,CN=System,DC=Acme,DC=com"
```

The above example outputs:

```
Version            : 5.2.102.0
ID                 : 1e71ef8e-4ffc-4769-9a5b-11ea102b0f8e
ConfigRevision     : 510
ApplicationGroups  : {cmd}
ContentGroups      : {}
URLGroups          : {}
Tokens             : {}
GlobalOptionsSets  : {}
Files              : Avecto.Defendpoint.Settings.FileList
Messages           : {Block Message, Allow Message (Elevate)}
Policies           : {New Workstyle}
Licenses           : {Avecto.Defendpoint.Settings.License}
RegistryValues     : {}
```

Chapter 4 - Define the Defendpoint Settings: `Set-DefendpointSettings`

`Set-DefendpointSettings` - Save Defendpoint settings to either Local Group Policy, a local file, or a specified Group Policy Object (GPO).

4.1 - Overview

4.1.1 - Syntax

```
Set-DefendpointSettings [-SettingsObject 'path/to/object'] [-
LocalPolicy -XML 'path/to/file'] [-UserPolicy] [-Merge
'path/to/file']
```

4.1.2 - Description

The `Set-DefendpointSettings` cmdlet takes an XML file or Defendpoint settings object as input, and saves it to either a local file, Local Group Policy, or Group Policy Object (GPO). By default this function overwrites the existing Defendpoint settings at the target location, unless the `-Merge` parameter is used.

4.2 - Parameters

Parameter	Type	Description	Require-d
`SettingsObject`	Defendpoint Configuration Object	Supply the DefendpointSettings object that should be used as input. It can be obtained from `Get-DefendpointSettings`.	Yes
`-LocalPolicy`	Boolean	Set a local policy file as the Defendpoint Settings. This only works with XML files, and must be used in conjunction with the `-XML` parameter, along with the full path to a Defendpoint Settings XML file.	
`UserPolicy`	Boolean	Update the user policy. If not set, the machine policy is updated instead.	
`Merge`	Boolean	Merge the input settings with the target file. If this parameter is not set, the target file is overwritten.	
`LocalFile`	String	Save the Defendpoint settings to a local file. This argument defaults to `%PROGRAMDATA%\Avecto\Privilege Guard\PrivilegeGuardConfig.xml` if `-FileLocation` is not used.	
`TapConfigPath`	String	Define the file save destination. If not set, the file is saved to the local Defendpoint settings file destination: `%PROGRAMDATA%\Avecto\Privilege Guard\PrivilegeGuardConfig.xml`.	

Parameter	Type	Description	Require-d
`Domain`	Boolean	Save to a Group Policy Object (GPO). This is used in conjunction with the `-LDAP` parameter.	
`LDAP`	String	The LDAP path of the GPO. For example: `LDAP://DC13.Acme.com/CN={31B2F340-016D-11D2-945D-00D04CB984F9},CN=Policies,CN=System,DC=Acme,DC=com`	Yes when `-Domain` is supplied
`XML`	String	The path of a Defendpoint settings XML file that should be used as input.	

4.3 - Return Values

`Set-DefendpointSettings` returns errors if there are any. If not, the function does not return anything.

4.4 - Examples

4.4.1 - Add Licence to Defendpoint Settings Configuration

Although this example is not part of the API, it is useful to know in this context.

```
$PGLicence = "YOUR_LICENCE_HERE"
$PGConfig = Get-DefendpointSettings -LocalFile
$PGLicence = New-Object Avecto.Defendpoint.Settings.License
$PGLicence.Code = "$PGLicense"
$PGConfig.Licenses.Add($PGLicence)
```

The above example does not output anything to the terminal.

4.4.2 - Get and set the local Defendpoint settings from a Defendpoint object

```
$DefendpointSettingsObject = Get-DefendpointSettings -LocalPolicy
Set-DefendpointSettings -SettingsObject
$DefendpointSettingsObject
```

The above example does not output anything to the terminal.

4.4.3 - Set the local Defendpoint settings from an XML file

```
Set-DefendpointSettings –LocalPolicy -XML
C:/Users/admin/Desktop/PrivilegeGuardConfig.xml
```

The above example does not output anything to the terminal.

4.4.4 - Merge a Defendpoint config with the domain machine policy Defendpoint config

```
# Set the licence and LDAP.
$Ldap = "LDAP://DC13.Acme.com/CN={31B2F340-016D-11D2-945F-
>> 00C04FB984F9},CN=Policies,CN=System,DC=Acme,DC=com"

# Get the local PG Config file.
$PGConfig = Get-DefendpointSettings -Domain -LDAP $Ldap

# Create a new license object.
$PGLicence = New-Object Avecto.Defendpoint.Settings.License
$PGLicence.Code = "$PGLicense"

# Add the license to the defendpoint config.
$PGConfig.Licenses.Add($PGLicence)

# Merge the Defendpoint config with the existing Domain Machine
policy Defendpoint config.
Set-DefendpointSettings -SettingsObject $PGConfig -Domain -Ldap
$Ldap
```

The above example does not output anything to the terminal.