Full credit is given to the above companies including the OS that this PDF file was generated!

PowerShell Get-Help on command 'ConvertFrom-SddlString'

PS C:\Users\wahid> Get-Help ConvertFrom-SddlString

NAME

ConvertFrom-SddlString

SYNOPSIS

Converts a SDDL string to a custom object.

SYNTAX

ConvertFrom-SddlString [-Sddl] <System.String> [-Type {FileSystemRights | RegistryRights | ActiveDirectoryRights | MutexRights | SemaphoreRights | CryptoKeyRights | EventWaitHandleRights}] [<CommonParameters>]

DESCRIPTION

The `ConvertFrom-SddlString` cmdlet converts a Security Descriptor Definition Language string to a custom PSCustomObject object with the following properties: Owner, Group, DiscretionaryAcl, SystemAcl and RawDescriptor.

Owner, Group, DiscretionaryAcl and SystemAcl properties contain a readable text representation of the access rights specified in a SDDL string.

This cmdlet was introduced in PowerShell 5.0.

PARAMETERS

-Sddl <System.String>

Specifies the string representing the security descriptor in SDDL syntax.

-Type <System.Object>

Specifies the type of rights that SDDL string represents.

The acceptable values for this parameter are:

- FileSystemRights

- RegistryRights

- ActiveDirectoryRights

- MutexRights

- SemaphoreRights

- CryptoKeyRights

- EventWaitHandleRights

By default cmdlet uses file system rights. CryptoKeyRights and ActiveDirectoryRights are not supported in PowerShell v6 and higher.

<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).

Example 1: Convert file system access rights SDDL to a PSCustomObject

```
$acl = Get-Acl -Path C:\Windows
ConvertFrom-SddlString -Sddl $acl.Sddl
```

The first command uses the `Get-Acl` cmdlet to get the security descriptor for the C:\Windows folder and saves it in the variable.

The second command uses the `ConvertFrom-SddlString` cmdlet to get the text representation of the SDDL string, contained in the Sddl property of the object representing the security descriptor.

Example 2: Convert registry access rights SDDL to a PSCustomObject

```
$acl = Get-Acl HKLM:\SOFTWARE\Microsoft\
ConvertFrom-SddlString -Sddl $acl.Sddl -Type RegistryRights
```

The first command uses the `Get-Acl` cmdlet to get the security descriptor for the HKLM:\SOFTWARE\Microsoft\ key and saves it in the variable.

The second command uses the `ConvertFrom-SddlString` cmdlet to get the text representation of the SDDL string, contained in the Sddl property of the object representing the security descriptor.

It uses the `-Type` parameter to specify that SDDL string represents a registry security descriptor.

Example 3: Convert registry access rights SDDL to a PSCustomObject by using ConvertFrom-SddlString with and without the `-Type` parameter

```
$acl = Get-Acl -Path HKLM:\SOFTWARE\Microsoft\
```

```
ConvertFrom-SddlString -Sddl $acl.Sddl | Foreach-Object
{$_.DiscretionaryAcl[0]}

BUILTIN\Administrators: AccessAllowed (ChangePermissions, CreateDirectories,
Delete, ExecuteKey, FullControl, GenericExecute, GenericWrite, ListDirectory,
ReadExtendedAttributes, ReadPermissions, TakeOwnership, Traverse, WriteData,
WriteExtendedAttributes, WriteKey)

ConvertFrom-SddlString -Sddl $acl.Sddl -Type RegistryRights | Foreach-Object
{$_.DiscretionaryAcl[0]}

BUILTIN\Administrators: AccessAllowed (ChangePermissions, CreateLink,
CreateSubKey, Delete, EnumerateSubKeys, ExecuteKey, FullControl,
GenericExecute, GenericWrite, Notify, QueryValues, ReadPermissions, SetValue,
TakeOwnership, WriteKey)
```

The first command uses the `Get-Acl` cmdlet to get the security descriptor for the HKLM:\SOFTWARE\Microsoft\ key and saves it in the variable.

The second command uses the `ConvertFrom-SddlString` cmdlet to get the text representation of the SDDL string, contained in the Sddl property of the object representing the security descriptor.

It doesn't use the `-Type` parameter, so the access rights shown are for file system.

The third command uses the `ConvertFrom-SddlString` cmdlet with the `-Type` parameter, so the access rights returned are for registry.

Example 4: Convert Active Directory access rights SDDL to a PSCustomObject

```
$user = [ADSI]"LDAP://CN=username,CN=Users,DC=domain,DC=com"
ConvertFrom-SddlString $user.psbase.ObjectSecurity.Sddl -Type
```

ActiveDirectoryRights

The first command uses Active Directory Service Interfaces (ADSI) to get the user object and saves it in the variable.

The second command uses the `ConvertFrom-SddlString` cmdlet to get text representation of the SDDL string, contained in the Sddl property of the object representing the security descriptor.

It uses the `-Type` parameter to specify that SDDL string represents an Active Directory security descriptor.

REMARKS

To see the examples, type: "get-help ConvertFrom-SddlString -examples".

For more information, type: "get-help ConvertFrom-SddlString -detailed".

For technical information, type: "get-help ConvertFrom-SddlString -full".

For online help, type: "get-help ConvertFrom-SddlString -online"