Intel® Setup and Configuration Software (Intel® SCS)

Solutions Framework

Version 10.0

Document Release Date: February 08, 2016

License

Intel® Setup and Configuration Software (Intel® SCS) is furnished under license and may only be used or copied in accordance with the terms of that license. For more information, refer to the "Exhibit A" section of the "Intel(R) SCS License Agreement.rtf", located in the Licenses folder.

Legal Information

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, specific software, or services activation. Performance varies depending on system configuration. Check with your system manufacturer or retailer.

Intel® AMT should be used by a knowledgeable IT administrator and requires enabled systems, software, activation, and connection to a corporate network. Intel AMT functionality on mobile systems may be limited in some situations. Your results will depend on your specific implementation.
Learn more by visiting Intel® Active Management Technology.

No computer system can be absolutely secure. Intel® IPT requires an enabled system, including a processor, chipset, firmware, software, and (in some cases) integrated graphics, and participating website or service. Intel does not assume any liability for lost or stolen data or systems or any other damages. Check with your manufacturer or retailer for more information. Learn more at http://ipt.intel.com/.

No computer system can provide absolute security under all conditions. Built-in security features available on select Intel® Solid State Drives may require additional software, hardware, services and/or an Internet connection. Results may vary depending upon configuration. Consult your system manufacturer for more details.

Location Based Service (LBS) requires a mobile or Ultrabook™ system with select 3rd generation or later Intel® Core™ vPro™ processor. Local wireless network set up and additional third party software is required to enable this capability. Availability of features and results will depend upon the setup and configuration of your hardware, software and IT environment. Accuracy of location information is dependent on several factors, including placement of wireless access points and the physical environment where the solution is deployed.

Intel and the Intel logo, are trademarks of Intel Corporation in the U.S. and/or other countries.

Microsoft, Windows, and the Windows logo are trademarks, or registered trademarks of Microsoft Corporation in the U.S. and/or other countries.

* Other names and brands may be claimed as the property of others.

© 2016 Intel Corporation

Table of Contents

1 Introduction **1**
1.1 Why was the Solutions Framework Created? 1
1.2 How Do I Use It? 1
1.2.1 Discovering and Publishing Data 2
1.2.2 Configuring Solutions 3
1.3 Which Intel Products are Supported? 4

2 Prerequisites **6**
2.1 Supported Operating Systems 6
2.2 Required Software 7
2.3 Required Permissions 8
2.3.1 Permissions on the Host Platforms 8
2.3.2 Permissions for Integration with RCS 9
2.4 Trusted Root Certificates 10

3 Preparing the Host Platforms **11**
3.1 Installing the Host Solution Manager 11
3.2 Installing the Host Plugin 11

4 Defining Configuration Profiles **12**
4.1 Which Component Should I Use to Create Profiles? 12
4.2 Using the Console to Create Profiles 13
4.3 Using the Profile Editor to Create Profiles 15

5 Using the Scripts **16**
5.1 Execution Policy Setting 17

6 Solutions Framework API Reference **18**
6.1 WMI Client 18
6.2 Solution GUIDs 19
6.3 Solutions Framework Classes 20
6.3.1 SCS_ManagedElement 20
6.3.2 SCS_HostSolutionManager 21
6.3.3 SCS_SolutionEvent 21
6.4 RCS Classes 22
6.4.1 RCS_PlatformDiscovery 22
6.4.2 RCS_PlatformDiscoverySummary 23
6.4.3 RCS_SolutionKPI 23
6.4.4 RCS_SolutionDiscovery 24
6.4.5 RCS_SolutionData 24
6.5 Methods 25

6.5.1 PlatformDiscovery 25
6.5.2 Recalculate 25
6.5.3 RefreshSolutions 26
6.5.4 SolutionDiscovery 26
6.5.5 GetConfiguration 26
6.5.6 Configure 27
6.5.7 UnConfigure 27
6.5.8 GetKPIs 27
6.5.9 GetAndResetKPIs 28
6.5.10 ResetKPIs 28
6.5.11 GetAttribute 29
6.5.12 SetAttribute 29
6.5.13 ApplyCommandSync 30
6.5.14 BackupSolutionSettings 30
6.5.15 RestoreSolutionSettings 31
6.5.16 CountEvents 31
6.5.17 ClearAllEvents 31
6.5.18 EnumeratePermissions 32
6.5.19 AddPermissions 32
6.5.20 RemovePermissions 33

7 XML Reference 34
7.1 XML Data Protocol 34
7.1.1 Platform Discovery DTD 34
7.1.2 Solution Discovery and KPI DTD 35
7.2 Platform Discovery Example 36
7.3 Solution Discovery Example 39
7.4 KPI Example 41

1 Introduction

This document describes how to install and use the Solutions Framework component of Intel SCS.

1.1 Why was the Solutions Framework Created?

The Solutions Framework was created to extend the discovery and configuration capabilities of Intel SCS to other Intel products (also referred to as "solutions"). The purpose of the Solutions Framework is to enable you to discover and configure different types of Intel products, while using one common set of tools.

1.2 How Do I Use It?

This diagram shows the main components of the Solutions Framework and how they integrate with other Intel SCS components.

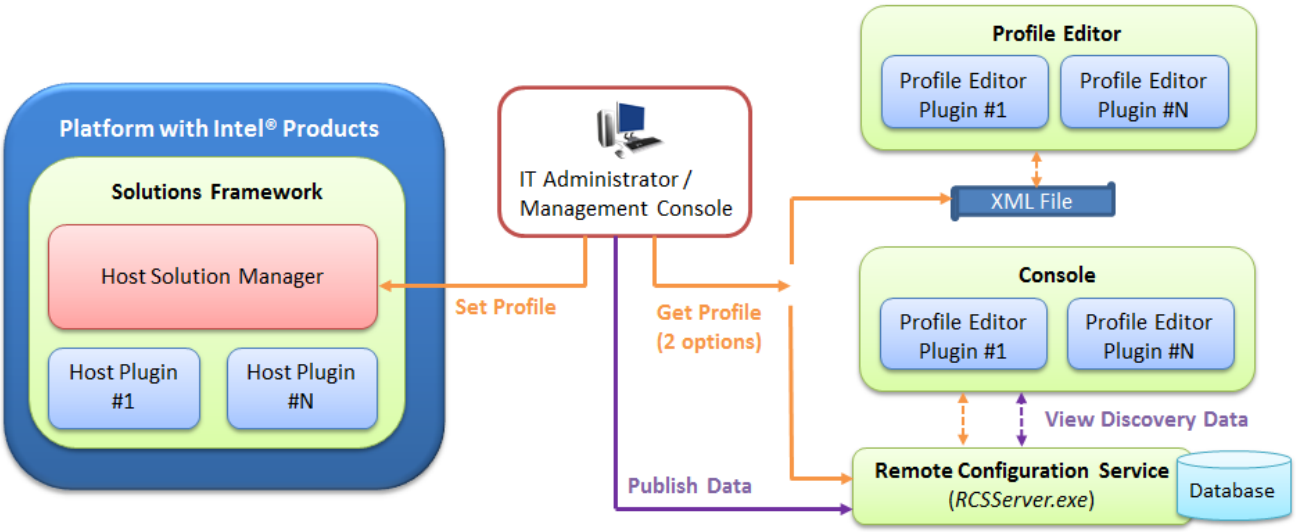

Figure 1: Overview of the Solutions Framework

The Solutions Framework includes these main components:

- **Host Solution Manager** - Exposes an API that uses Windows Management Instrumentation (WMI) and is the single access point for all plugins that want to use the Solutions Framework
- **Host Plugins** - A separate plugin for each solution handles configuration requests and requests for discovery data and Key Performance Indicators (KPI), as supported by the solution
- **Profile Editor Plugins** - A separate plugin for each solution handles requests to create/update configuration profiles with the settings supported by the solution

1.2.1 Discovering and Publishing Data

You can use the Solutions Framework to collect data from the individual host platforms and then add it to your management console database. (To do this, refer to the documentation of your hardware/software inventory application.) You can also use the Solutions Framework to collect data from the individual host platforms and then publish it to the RCS. The RCS stores the data in the Intel SCS database from where it can be retrieved later and used to generate reports.

The Solutions Framework supports three different types of discovery operation:

- **Platform Discovery** - This type of discovery gets "top-level" data about the hardware and software of each Intel product that exists on your platforms. You can use this data to determine which Intel products you can enable on your platforms, and which software or hardware updates are required. This type of discovery does not require any host plugins to be installed on the host platforms. You can run this operation using:
  - An executable - The Platform Discovery Utility (`PlatformDiscovery.exe`) is located in the `PlatformDiscovery` folder. To use this utility you must deploy the `PlatformDiscovery.exe` file to the host platforms.
  - A script - The `SCS-PlatformDiscovery.ps1` script is located in the `Scripts` folder. This script is similar to the Platform Discovery Utility but has the added advantage that you can run the script remotely. This means you can run platform discovery operations without deploying the `PlatformDiscovery.exe` (or even the script) to the platforms. All you need to do is to use your software deployment mechanism to install the Host Solution Manager on the host platforms (see Installing the Host Solution Manager on page 11). After that you can run this script remotely like this:

    ```
    .\SCS-PlatformDiscovery -ComputerName MyHost.Domain.com
    ```

  For an example of the data format, see the Platform Discovery Example on page 36.
- **Solution Discovery** - This type of discovery gets data from the host plugins about each solution. You can use the supplied script (`SCS-Discovery.ps1`) or the API (see SolutionDiscovery on page 26). Data will be returned only for solutions that have a plugin on the host platform. The data that is returned is defined by the host plugin of each solution.

  For an example of the data format, see the Solution Discovery Example on page 39.
- **Key Performance Indicators (KPI)** - Some solutions can report additional data related to monitoring and performance statistics, known as KPI data. If the host plugin supports this type of data, you can use the supplied script (`SCS-ReadKPI.ps1`) or the API (see GetKPIs on page 27).

  For an example of the data format, see the KPI Example on page 41.

Note:

After publishing the data to the RCS, you can view it in the Console. Currently, the Console only supports viewing the Solution Discovery data.

1.2.2 Configuring Solutions

Many Intel products contain settings that you can configure. If supported, you can use the Solutions Framework to configure these settings using a configuration profile. The settings that you can configure, and the format of the profile, are defined by the plugins of each solution. You can configure solutions using the supplied script (`SCS-Configure.ps1`) or API method (see Configure on page 27).

Before you can configure a solution, you must create the configuration profile. You can create the profile using the Console or the Profile Editor (see Which Component Should I Use to Create Profiles? on page 12).

Using the Console

1. Create the profile in the Console (see Using the Console to Create Profiles on page 13).
2. Run the `SCS-Configure.ps1` script using the `-RCS` and `-ProfileName` parameters.

   For example:

```
.\SCS-Configure.ps1 -ComputerName MyHost.Domain.com
  -Component 0b93a9cb-64b1-43ed-bc33-e5b0b8692694
  -RCS rcs.domain.com -ProfileName "Profile1"
```

Using the Profile Editor

1. Create the profile as an XML file using the Profile Editor (see Using the Profile Editor to Create Profiles on page 15).
2. Run the `SCS-Configure.ps1` script using the `-ProfileFile` parameter.

   For example:

```
.\SCS-Configure.ps1 -ComputerName MyHost.Domain.com
  -Component 0b93a9cb-64b1-43ed-bc33-e5b0b8692694
  -ProfileFile "MySolutionProfile.xml"
```

   It is also possible to supply the content of the XML file using the `-ProfileXML` parameter.

   For example:

```
.\SCS-Configure.ps1 -ComputerName MyHost.Domain.com
  -Component 0b93a9cb-64b1-43ed-bc33-e5b0b8692694
  -ProfileXML "<Management><ManagedElement><Profile><dipm>tuned</dipm>
    <superfetch>tuned</superfetch></Profile></ManagedElement></Management>"
```

1.3 Which Intel Products are Supported?

The Solutions Framework is built as a modular system that enables new Intel products to be easily added to the list of supported products.

Platform Discovery

The Platform Discovery Utility can discover "top-level" data from certain products, even if plugins were not yet created for them. The Platform Discovery Utility included in this release can discover these products:

- Intel® AMT
- Intel® SSD Professional Family:
  - Intel® SSD Pro 1500 Series
  - Intel® SSD Pro 2500 Series
- Intel® Enterprise Digital Fence
- Intel® Identity Protection Technology with One Time Password (Intel® IPT with OTP)
- Intel® Identity Protection Technology with Public Key Infrastructure (Intel® IPT with PKI)
- Intel® WiGig
- Intel® Smart Connect Technology
- Intel® Anti-Theft Technology
- Location Based Service
- Intel® WiDi
- Intel® Authenticate

As new releases of the Platform Discovery Utility are released, support for additional products will be added.

Products with Plugins

Most options and features that are made available by the Solutions Framework require the use of plugins.

These are the Intel products for which plugins are available:

- Intel SSD Professional Family — Intel SSD Professional Family products include settings that you can configure to optimize performance, and pro-actively monitor to prevent failures. With the Solutions Framework and the plugin, you can now remotely discover and collect data from all the Intel SSD Professional Family products in your organization. In addition, you can also run diagnostic scans, tune the host system, and run the Intel SSD Optimizer. For more information about the Intel SSD Pro Series, go to this web page: http://www.intel.com/content/www/us/en/solid-state-drives/solid-state-drives-ssd.html.
- Intel Enterprise Digital Fence — Intel Enterprise Digital Fence is a service that periodically checks if sleeping platforms are located in a "Safe Zone" (at work or at home). If not in a Safe Zone, Intel Enterprise Digital Fence forces the platform to hibernate and encrypts the Intel SSD Pro Series drive. With the Solutions Framework and the plugin, you can now remotely configure the Safe Zones and other Intel Enterprise Digital Fence settings.
- Intel Authenticate —Intel Authenticate improves security for Enterprise applications and services by strengthening authentication factors and flows. The authentication factors are "hardened" and protected in the hardware of the Intel platform. This makes it difficult for malware and attackers to access or tamper with sensitive authentication data.

As support for more Intel products becomes available, you can easily add them using the same instructions and conventions described in this document. (Once the Solutions Framework is set up, all you have to do is add some DLL files for the relevant Intel product.)

2 Prerequisites

This section describes the prerequisites and tasks that you must perform before using the Solutions Framework.

2.1 Supported Operating Systems

This table describes on which operating systems the Solutions Framework components of this release can run.

Version	Platform Discovery	Host Solution Manager	Profile Editor
Windows* 10 Pro	Yes	Yes	Yes
Windows 10 Enterprise	Yes	Yes	Yes
Windows 8.1 Pro	Yes	Yes	Yes
Windows 8.1 Enterprise	Yes	Yes	Yes
Windows 7 Professional	Yes	Yes	Yes
Windows 7 Enterprise (SP1)	Yes	Yes	Yes
Windows Server* 2012 R2	No	No	Yes
Windows Server 2012	No	No	Yes
Windows Server 2008 R2 (SP2)	No	No	Yes
Windows Server 2008 (SP2)	No	No	Yes

Additional Requirements

- The Profile Editor requires version 3.5 of Microsoft .NET Framework (SP1) to be installed on the computer.
- Intel SCS components can run on operating systems installed with these languages: Czech, Danish, Dutch, English, Finnish, French, German, Greek, Hungarian, Italian, Japanese, Korean, Norwegian, Polish, Portuguese, Portuguese-Brazilian, Russian, Simplified Chinese, Spanish, Swedish, Traditional Chinese, Turkish.
- Intel SCS does not support Non-Latin or Extended Latin characters in filenames or values in the XML files.
- A minimum screen resolution of 1024 x 768 is necessary to use the Profile Editor. The 800 x 600 screen resolution is not supported.

2.2 Required Software

This table describes the required software.

Software	Description
Host Solution Manager	See Installing the Host Solution Manager on page 11
Solution Plugins	Each solution that is fully supported by the Solutions Framework (and not just the Platform Discovery Utility) can include two plugins: • **Host Plugin** – This plugin is mandatory and is supplied by all solutions that are fully supported by the Solutions Framework. You must install this plugin on each of the host platforms where you want to use the plugin (see Installing the Host Plugin on page 11). • **Profile Editor Plugin** – This plugin is used to create configuration profiles that can be used to configure the solution. This type of plugin will only exist for solutions that support configuration using the Solutions Framework. You can download solution plugins from the Intel SCS website: http://www.intel.com/go/scs.
Console or Profile Editor	The profile editor plugins can be loaded by these Intel SCS components: • Console (requires the RCS installed in database mode) • Profile Editor To decide which component to use, see Which Component Should I Use to Create Profiles? on page 12.
Remote Configuration Service (RCS)	The Solutions Framework includes an option to publish data to the RCS. If you want to use this option, you must install the RCS in database mode. The installer for the RCS is located in the `RCS` folder of the Intel SCS download package. For information how to install the RCS, refer to the `Intel(R)_SCS_User_Guide.pdf`.
Windows* PowerShell	The scripts provided with the Solutions Framework were written in PowerShell 3.0. To use these scripts, version 2.0 or higher of PowerShell must be installed on each computer where you want to use them. You can check the PowerShell version using this PowerShell command: `get-host`. **Note:** The use of PowerShell and the supplied scripts is optional. You can also use Windows Management Instrumentation (WMI) to call the methods directly from the Solutions Framework API (see Solutions Framework API Reference on page 18).

2.3 Required Permissions

This section describes the required permissions when using the Solutions Framework.

2.3.1 Permissions on the Host Platforms

This table describes the permissions on the host platform and when they are required.

Type of Permission	Description
Administrator Permissions	The Platform Discovery Utility requires administrator permissions on the host platform. The local user account running the Platform Discovery Utility must have administrator permissions in the operating system. On operating systems with User Account Control (UAC), the Platform Discovery Utility must be "Run as administrator".
WMI Permissions	The `Intel_SCS_Framework` namespace is added to each of the host platforms when you install the Host Solution Manager. Permissions on this namespace are needed when running any of the API methods or supplied scripts. These are the necessary WMI Permissions required by all users of the Solutions Framework: Execute Methods, Partial Write, Provider Write, Enable Account, Remote Enable, Read Security, Edit Security.
DCOM Permissions	For increased security, each solution plugin is loaded as a separate process and is protected in its own memory space. This is achieved by using what is known as a "COM wrapper application". During installation of the Host Solution Manager, this application is registered as a component service: • Application Name: `SFPluginWrapper.exe` • Application ID: BB1E5C03-C671-4679-AC09-05F030A37FED (The application is shown in the Component Services > DCOM Config window.) These are the necessary DCOM permissions required on `SFPluginWrapper.exe` by all users of the Solutions Framework: • Launch and Activate Permissions: Local Launch, Remote launch, Local Activation, and Remote Activation • Access Permissions: Local Access and Remote Access • Configuration Permissions: Full Control
Folder Permissions	These are the necessary permissions: Modify, Read & Execute, List folder and contents, Read, and Write. • For all users of the Solutions Framework – The folder where the Host Solution Manager is installed. • For users of a specific solution – The folder where the host plugin of that solution is installed. **Note:** The default installation folder for the Host Solution Manager is `C:\Program Files\Intel\SCS\Solution Framework`.

The Solutions Framework API includes methods that you can use to remotely manage the WMI, DCOM, and folder permissions on your host platforms. For more information, see:

- AddPermissions on page 32
- RemovePermissions on page 33
- EnumeratePermissions on page 32

The `Scripts` folder also includes a PowerShell script named `SCS-Permissions.ps1` that you can use instead.

For example, to give a user group named "MyUserGroup" permissions on the Intel SSD Pro Series solution:

```
.\SCS-Permissions.ps1 -ComputerName MyHost.Domain.com
  -Component "0b93a9cb-64b1-43ed-bc33-e5b0b86926941"
  -Operation Add -NameOrSID "MyUserGroup"
```

Note:

The API methods and the `SCS-Permissions.ps1` script only set the "Allow" permissions. No changes are made to any existing "Deny" permissions.

2.3.2 Permissions for Integration with RCS

The Solutions Framework includes options for integration with the Remote Configuration Service (RCS) component of Intel SCS. If you want to use these options, you will need permissions on certain namespaces located on the computer running the RCS.

This table describes the relevant namespaces and when permissions are required.

Namespace	Description
`Intel_RCS`	Permissions on this namespace are needed only if you want to publish data to the RCS.
`Intel_RCS_Editor`	Permissions on this namespace are needed only if you want to configure the solution using a profile created in the Console. For example, when using the `-RCS` option in the `Configure.ps1` script.
`Intel_RCS_Systems`	Permissions on this namespace are needed only if you want to view data that was published to the RCS.

If a user has administrator permissions on the computer running the RCS they automatically have the necessary permissions on these namespaces. If you do not want to give a user administrator permissions, you must give them DCOM and WMI permissions. Intel SCS includes a utility that you can use to give the necessary DCOM and WMI permissions on the RCS computer. The utility (`RCSUtils.exe`) is located in the `Utils` folder.

For example, to give a user group named "MyUserGroup" permissions on all three namespaces:

```
RCSUtils.exe /Permissions Add MyUserGroup /RCSNamespace RCS Editor Systems
```

For more information, refer to the `Intel(R)_SCS_RCSUtility.pdf`.

2.4 Trusted Root Certificates

The executable and DLL files of the Solutions Framework are digitally signed by Intel and include a timestamp. Using digital signatures increases security because it gives an indication that the file is genuine and has not been changed. Digital certificates contain data about the organization from which they were issued. This data forms a "certificate chain" that ends in a trusted root certificate of a known CA.

These are the trusted root certificates currently used by Intel to validate digital signatures:

- AddTrust External CA Root
- Equifax Secure Certificate Authority (being phased out of use)

These are the trusted root certificates currently used by Intel to validate the time-stamp:

- Commodo Time Stamping Signer
- Symantec Time Stamping Services (being phased out of use)

Note:

These trusted root certificates are already pre-installed in most Windows operating systems. You can view the certificate chain by right-clicking the file and selecting **Digital Signatures** > **Details** > **View Certificate** > **Certification Path**. If the root certificate is not installed, an error is shown.

If the trusted root certificate cannot be located and validated, the Solutions Framework will not operate correctly.

For example:

- If the DLL of the host plugin cannot be validated (on the host platform), the plugin will not be loaded by the Host Solution Manager. All attempts to use the functionality of the plugin will result in an error.
- If the DLL of the profile editor plugin cannot be validated (on the computer running the Console), the Console will not load the plugin. The option to select the profile type for that plugin will not be shown in the Profile Wizard window.
- If the DLL of the profile editor plugin cannot be validated (on the computer running the Profile Editor), the Profile Editor will not load the plugin.

Some versions of Windows do not include all the trusted root certificates necessary to validate time-stamped digital signatures. If the trusted root certificate is not installed in the operating system, Windows uses an automatic update mechanism to download the necessary root certificate from Microsoft. If the system does not have Internet access, the automatic update mechanism will fail.

If necessary, you can manually download and install a root certificate update package from the Microsoft update catalog. Select the relevant package for the operating system from this website:

http://catalog.update.microsoft.com/v7/site/Search.aspx?q=root%20certificate%20update

Note:

It is recommended to open this link using Microsoft Internet Explorer 6.0 or later because it takes you directly to the list of all the downloads. (If you use a different web browser you will need to search for the location of the required download.)

3 Preparing the Host Platforms

This section describes the prerequisites and tasks required on each host platform before you can use the Solutions Framework.

3.1 Installing the Host Solution Manager

The Host Solution Manager must be installed on each host platform where you want to use the Solutions Framework. The installer file (`HostSolutionManagerInstaller.msi`) is located in this folder in the Intel SCS download package:

`Solutions_Framework\Framework`

The installer registers the MOF files used by the Solutions Framework with the WMI provider. The installer is based on the Windows Installer CLI and uses the commands available in the standard installation mode. This includes support for "silent installation".

For example, to silently install from a command prompt:

```
HostSolutionManagerInstaller.msi /qn
```

The Solutions Framework folder also contains a batch file named `HostSolutionManager-silent-install.bat`. You can use this file to silently install when you deploy the installer using your software deployment mechanism.

3.2 Installing the Host Plugin

The host plugin DLL of the solution must be installed on each host platform where you want to use the plugin. You can install the plugin in any location that you want on the host platform. The Host Solution Manager searches for solution plugins by looking in the registry. Usually, an installer file will be supplied that you can use to install the DLL (and any additional files it requires) and automatically create a registry key for the DLL.

If you are not using an installation file, you must manually add a registry key for the plugin at this location on the host platform:

- On 32-bit operating systems: `HKLM\SOFTWARE\Intel\Intel(R)Setup and Configuration Software\Solutions\`
- On 64-bit operating systems: `HKLM\SOFTWARE\WOW6432Node\Intel\Intel(R)Setup and Configuration Software\Solutions\`

Add the key underneath the Solutions key. The key must contain a String named `Path`. The value of `Path` must be the full path to the plugin DLL, including the filename.

4 Defining Configuration Profiles

This section describes how to define configuration profiles for solutions.

Note:
The tasks described in this section are only relevant for solutions that include a profile editor plugin.

Configuration profiles contain the configuration settings that will be put in the solution during configuration. Profiles for solutions are created using the profile editor plugin of the solution. The plugins can be loaded by the Console or Profile Editor components of Intel SCS.

The profile editor plugin of each solution will show the settings that it supports. For descriptions of these settings, click the help icon to open the help provided by the solution, or refer to the documentation provided with the plugin.

4.1 Which Component Should I Use to Create Profiles?

Intel SCS includes two different components that you can use to create configuration profiles for solutions. The component that you select to use depends mainly on which components you use to configure Intel AMT. (The objective being to let you use similar components and methods to configure Intel AMT and solutions.)

This table describes the main considerations to help you decide which component to use to create profiles.

This component...	Is recommended if...
Console	• You are using, or intend to use, the RCS in database mode to configure Intel AMT • You want to store all configuration profiles in one central location (in the Intel SCS database) For more information, see Using the Console to Create Profiles on the next page.
Profile Editor	• You are using the Intel AMT Configuration Utility to configure Intel AMT • You are using, or intend to use, the RCS in non-database mode to configure Intel AMT • You don't have or want to install an SQL Server database For more information, see Using the Profile Editor to Create Profiles on page 15.

4.2 Using the Console to Create Profiles

The Console is the user interface to the RCS. When the RCS is installed in non-database mode you can only create profiles for Intel AMT. When the RCS is installed in database mode, you can also use the Console to create configuration profiles for solutions. When the Console is started it looks in the `Plugins` folder for profile editor plugins. For each plugin that is found (and validated) a new option is shown in the Profile Type drop-down list of the Profile Wizard window (see below). Profiles created using the Console are stored in the Intel SCS database.

The installer for the RCS and the Console is located in the `RCS` folder of the Intel SCS download package. For information how to install these components, refer to the `Intel(R)_SCS_User_Guide.pdf`.

Note:

- When new profile editor plugins become available for other solutions, simply place them in the `Plugins` folder. When a new version of an installed plugin becomes available, simply replace the existing plugin. After adding or updating plugins, restart the Console to reload the new versions of the plugins.
- The `Plugins` folder is located in the `Console` folder. For example, the default installation folder for Intel SCS 11.0 is: `C:\Program Files\Intel\SCS11\Console\Plugins`.

To define a configuration profile using the Console:

1. In the Console, click **Profiles** and then click . The Profile Wizard window opens.

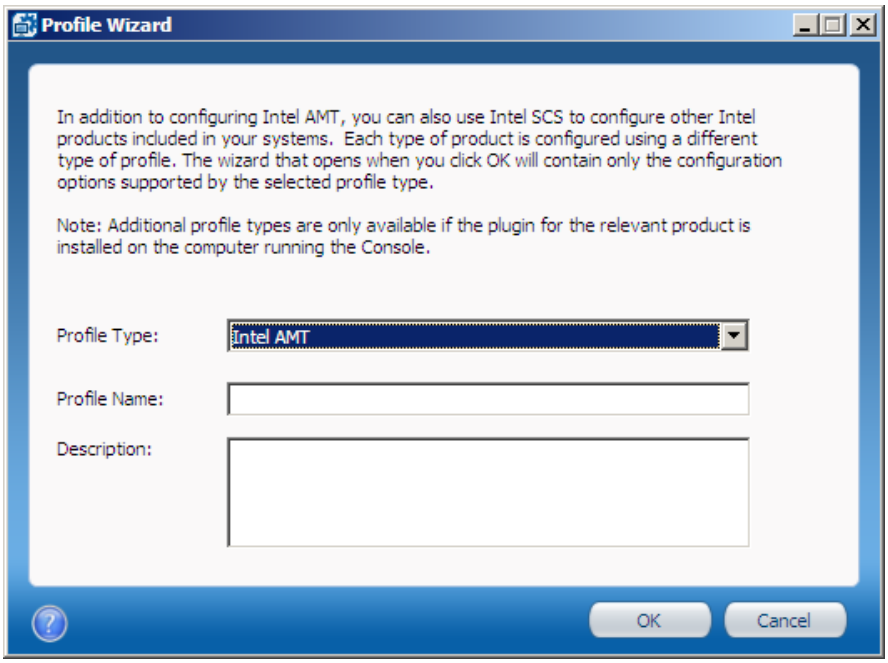

Figure 1: Profile Wizard

2. From the Profile Type drop-down list, select the type of profile for the Intel product that you want to configure. (The profile type is the name of the Intel product.)

> **Note:**
>
> The profile editor plugin DLLs are signed by Intel and will only load if the digital signature is valid and no changes were made to the file. (See Trusted Root Certificates on page 10.)

3. In the Profile Name field, enter a name for this profile. The profile name:
   - Can be a maximum of 32 characters
   - Cannot be empty or include only "whitespace" characters
   - Must include only alpha-numeric characters (7-bit ASCII characters in the range of 32-126), not including these characters:
     (/), (\ ), (:), (*), (?), (<), (>), (.), (,), (&), ("), (`), (|)
4. (Optional) In the Description field, enter a description for the profile. This field is for informational purposes only.
5. Click **OK**. The relevant wizard or profile settings page for the selected profile type opens.
6. Select the settings that you want to configure when using this profile.
7. Close the wizard/settings page. The profile is added to the list of profiles, as shown in this example.

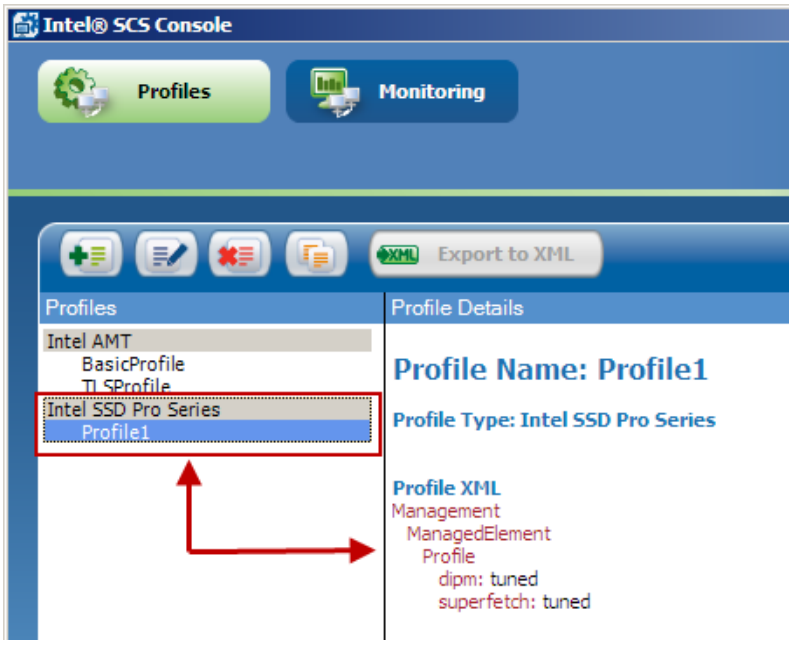

Figure 2: List of Profiles

In the list of profiles (in the left pane of the Profiles tab), the profiles are grouped together by solution. The contents of solution profiles are displayed in the Profile Details pane (on the right) in an XML format. You can also copy, edit, and delete these profiles just like you can with Intel AMT profiles.

4.3 Using the Profile Editor to Create Profiles

The Profile Editor is a standalone executable (`ProfileEditor.exe`), located in the `Solutions_Framework\ProfileEditor` folder. When the Profile Editor is started it looks in the `Plugins` subfolder for profile editor plugins. For each plugin that is found (and validated) a plugin editor is shown. Profiles created using the Profile Editor are saved as individual XML files.

To create a configuration profile:

1. In the `ProfileEditor` folder, double-click **ProfileEditor.exe**. The Profile Editor opens showing the available profile editor plugins in the left pane.
2. In the left pane, select the plugin of the solution for which you want to create a profile.
3. In the right pane, click **Create a New Profile**. A new profile opens containing the default settings for the selected solution.
4. Define the settings that you want to include in this configuration profile. The fields and options that are shown will be different for each solution. For information about the available options for this solution, click the help icon in the plugin toolbar.
5. From the plugin toolbar, click **Save** or **Save As**. The Save As window opens.
6. Specify the name and the location where you want to save the configuration profile and click **Save**. The Save As window closes and the profile is saved. The profile remains open in the profile editor plugin and the name of the profile is shown in the plugin toolbar.

5 Using the Scripts

The `Scripts` folder in the `Solutions_Framework` folder contains example Windows PowerShell scripts. The scripts are ready to use as they are, or you can edit and tailor them to your specific requirements.

For information about running PowerShell scripts, refer to this page in the Microsoft Technet site: http://technet.microsoft.com/en-us/library/ee176949.aspx.

The scripts include command line help. To open the help, from the PowerShell command line, type this:

```
get-help .\script-name.ps1 -full
```

This table describes the scripts.

Use this script...	To do this...
SCS-Permissions.ps1	Manage the WMI, DCOM, and folder permissions for the Solutions Framework on your host platforms (see Permissions on the Host Platforms on page 8).
SCS-PlatformDiscovery.ps1*	Discover the available software and hardware capabilities on the platform. This script is similar to the Platform Discovery Utility but has the added advantage that you can run the script remotely. This means you can run platform discovery operations without deploying the `PlatformDiscovery.exe` (or even the script) to the platforms. All you need to do is install the Host Solution Manager on the host platforms using your software deployment mechanism (see Installing the Host Solution Manager on page 11). After that you can run this script remotely like this: `.\SCS-PlatformDiscovery -ComputerName MyHost.Domain.com`
SCS-SolutionDiscovery.ps1*	Discover data about the solutions. Data will be returned for each solution that has a plugin on the host platform. The data that is returned is defined by the host plugin of each solution.
SCS-GetConfiguration.ps1	Get the current settings that are configured in a solution
SCS-Configure.ps1	Configure the solution with settings in a specified configuration profile. You can supply the profile as an XML file, or get the profile from the RCS. To get the profile from the RCS, you must first create the profile in the Console using the profile editor plugin of the solution.
SCS-Unconfigure.ps1	Reset the settings of a solution component to its default configuration settings
SCS-ReadKPI.ps1*	Get data about Key Performance Indicators (KPI) of the solutions. Data will be returned for each solution that has a plugin on the host platform. The data that is returned is defined by the host plugin of each solution.
SCS-ResetKPI.ps1	Reset the KPI data of a specific solution component, or all solution components

Use this script…	To do this…
SCS-GetAttribute.ps1	Get data about a specific attribute (or list of attributes) of a solution component
SCS-SetAttribute.ps1	Set the data of a specific attribute of a solution component
SCS-ApplyCommand.ps1	Send a command (and the required command input data) to a solution component. The supported commands are defined by the host plugin of each solution.
SCS-GetEvents.ps1	Get the events that currently exist for a specific solution component, or all solution components
SCS-ClearEvents.ps1	Clear all events of a specified solution component
* These scripts include a parameter (`-Publish`) that you can use to publish the data that they collect to the RCS. Data can only be published to an RCS that is installed in database mode. Using this parameter when the RCS is installed in non-database mode will return an error.	

5.1 Execution Policy Setting

By default, PowerShell is set to restrict the execution of any PowerShell scripts. If you do not change this setting before running a script, the script will fail. This means that you will need to change this setting on all computers where you want to run these PowerShell scripts.

To change the execution policy setting:

1. On the host platform, open a command prompt, type `PowerShell` and press <Enter>. The PowerShell command prompt shows.
2. Type `get-executionpolicy` and press <Enter>.
3. Make sure that the PowerShell execution policy is not set to "Restricted" (the default). If the value shown is Restricted, use the `set-executionpolicy` command to change it to one of these:
   - **Unrestricted** – No restrictions; all Windows PowerShell scripts can be run.
   - **AllSigned** – Only scripts signed by a trusted publisher can be run.
   - **RemoteSigned** – Downloaded scripts must be signed by a trusted publisher before they can be run.

Note:

The scripts provided in the Scripts folder are not currently signed. To run them remotely you will need to either sign them or set the execution policy to "Unrestricted".

6 Solutions Framework API Reference

The Solutions Framework API is a Windows Management Instrumentation (WMI) provider that exposes classes and methods used by the Host Solution Manager. You can use these classes and methods in scripts to send queries and commands to the Host Solution Manager. The PowerShell scripts supplied with the Solutions Framework (in the `Scripts` folder) also use the Solutions Framework API classes and methods. This means, for example, that using this API reference you can edit the PowerShell scripts to meet your own requirements. (It is recommended to edit copies of the original scripts.)

Note:

The following sections include all classes and methods available in the Solutions Framework API. Not all solutions will implement all these classes and methods. In addition, the values and descriptions of some of the method parameters and class properties are generic since each solution will have different values. For specific information about how a solution has implemented the API, refer to the documentation supplied with the solution.

6.1 WMI Client

The client that you use for WMI connection to the Solutions Framework (and the RCS) must use the Authentication Level of "Packet Privacy". If not, the Solutions Framework/RCS will return "Access Denied" errors. For example, in PowerShell 2.0, you must use the "-Authentication 6" option. For more information, see:

- http://technet.microsoft.com/en-us/library/ff730973.aspx
- http://msdn.microsoft.com/en-us/library/windows/desktop/aa394603(v=vs.85).aspx

6.2 Solution GUIDs

Each solution that is supported by the Solutions Framework is assigned a Globally Unique Identifier (GUID). To run some of the scripts, or use some of the API commands, you will need to supply this GUID.

Solution/Product	GUID
Intel AMT	a5e80b51-5429-4904-b412-589470884e97
Intel SSD Professional Series	0b93a9cb-64b1-43ed-bc33-e5b0b8692694
Intel Enterprise Digital Fence	e8c6bb4d-eda0-4859-bc7a-197cf252cf88
Intel Identity Protection Technology with OTP	7144b5b8-6883-44a9-80a2-c112ce6d6d7f
Intel Identity Protection Technology with PKI	34e60d6a-457a-483b-82ee-cb4f5ba6f542
Intel WiGig	958f5db1-b722-4fd4-8d5d-68c572751ec3
Intel Smart Connect Technology	a543b148-407c-419d-ab91-6a26af664828
Intel Anti-Theft Technology	8f14b6c3-9981-4347-8f05-08582591f486
Location Based Service	8d101420-2d30-4ff4-b18a-1da73e67d0ef
Intel WiDi	045964d1-6e6a-421a-b161-fd539f14f21f
Intel Authenticate	b4efb937-7caf-4b51-bc30-2224c5790f11

This table will be updated as support for new solutions and plugins is added. But if a new plugin is made available after release, please contact the plugin provider to get the GUID. (Alternatively, after installing the host plugin, you can run the `SCS-SolutionDiscovery.ps1` script. The script will return the GUID of the plugin in the output.)

6.3 Solutions Framework Classes

This section describes the Solutions Framework API classes.

6.3.1 SCS_ManagedElement

This class is the main interface to the solutions that are available on the platform. Enumeration of this class will return a list of the solution elements that are currently known to exist by the Host Solution Manager. The default enumeration (without a WHERE selector) will always list only one element which represents the platform.

All methods in this class require permissions on the `Intel_SCS_Framework` namespace.

Fields

Type	Name	Value or Description
string	ComponentID	A unique identifier of the solution instance in the format: `SolutionID.InstanceID.SubInstanceID`
string	Title	A short text description of the solution

Methods

- SolutionDiscovery on page 26
- GetConfiguration on page 26
- Configure on page 27
- UnConfigure on page 27
- GetKPIs on page 27
- GetAndResetKPIs on page 28
- ResetKPIs on page 28
- GetAttribute on page 29
- SetAttribute on page 29
- ApplyCommandSync on page 30
- BackupSolutionSettings on page 30
- RestoreSolutionSettings on page 31
- EnumeratePermissions on page 32
- AddPermissions on page 32
- RemovePermissions on page 33
- Enumerate and Get

6.3.2 SCS_HostSolutionManager

This is a static class containing methods that you can use to do operations at the platform level.

All methods in this class require permissions on the `Intel_SCS_Framework` namespace.

Fields

None

Methods

- PlatformDiscovery on page 25
- RefreshSolutions on page 26

6.3.3 SCS_SolutionEvent

This class is a data holder for an event recorded by a solution.

Enumeration of this class will return a list of the events for the specified solution (or events of all solutions if a ComponentID is not specified).

All methods in this class require permissions on the `Intel_SCS_Framework` namespace.

Fields

Type	Name	Value or Description
uint64	EventID	The event sequence number
datetime	Timestamp	The time of the event
string	ComponentID	The unique identifier of the solution component instance that generated the event, in this format: `SolutionID.InstanceID.SubInstanceID...`
uint32	Severity	Severity of the event, defined by the solution plugin
uint32	ErrorCode	Predefined solution specific error code number
uint32	Operation	Predefined solution specific operation code number
string	Details	The details of the event

Methods

- CountEvents on page 31
- ClearAllEvents on page 31
- Enumerate and Get

6.4 RCS Classes

This section describes additional classes that are relevant when the Solutions Framework is integrated with the Remote Configuration Service. These classes are available on the computer running the RCS, and require permissions on the `Intel_RCS_Systems` namespace. You can use these classes to get these types of data (if they were published to the RCS):

- Platform Discovery Data
- Solution Discovery Data
- KPI Data

6.4.1 RCS_PlatformDiscovery

This class is a data holder for the platform discovery data of a specific solution instance.

All methods in this class require permissions on the `Intel_RCS_Systems` namespace.

Fields

Type	Name	Value or Description
string	InstanceID	The ID of the platform discovery record
string	SystemUUID	The Universally Unique Identifier (UUID) of the system
string	SolutionName	The name of the solution
string	SolutionUUID	The GUID of the solution (see Solution GUIDs on page 19)
boolean	Exists	Indicates if the platform includes all the necessary hardware and firmware required by the solution (true or false)
boolean	Managed	Indicates if the Solutions Framework is capable of managing the solution (true or false). For solutions, this element indicates if a host plugin is installed and working correctly. For Intel AMT, the value of this element is true if Intel AMT is in a "configured" state. (Intel AMT is not currently a "solution").
string	State	The state of the solution. The values of this field will vary according to the solution.
string	Data	The discovery data of the specified solution (in an XML string) that was published to the RCS
datetime	LastUpdateTime	The last time that this data was published to the RCS

Methods

- Enumerate and Get

6.4.2 RCS_PlatformDiscoverySummary

This class contains platform discovery summary data of a specific solution instance.

All methods in this class require permissions on the `Intel_RCS_Systems` namespace.

Fields

Type	Name	Value or Description
string	InstanceID	The ID of the platform discovery summary record
string	SolutionName	The name of the solution
string	SolutionUUID	The GUID of the solution (see Solution GUIDs on page 19)
uint32	NotExistsPermille	The ratio of platforms where it is known that the current solution does not have the hardware requirements, specified in permille points
uint32	ExistsPermille	The ratio of platforms where the current solution has the hardware requirements, specified in permille points
uint32	ManagedPermille	The ratio of platforms where the current solution has been activated (considered to be "managed"), specified in permille points
datetime	LastCalculated	The last date and time when the report was last calculated

Methods

- Enumerate and Get
- Recalculate on page 25

6.4.3 RCS_SolutionKPI

This class contains KPI data of a specific solution instance.

All methods in this class require permissions on the `Intel_RCS_Systems` namespace.

Fields

Inherited from RCS_SolutionData

Methods

- Enumerate and Get

6.4.4 RCS_SolutionDiscovery

This class contains solution discovery data of a specific solution instance.

All methods in this class require permissions on the `Intel_RCS_Systems` namespace.

Fields

Inherited from RCS_SolutionData

Methods

- Enumerate and Get

6.4.5 RCS_SolutionData

This class is a data holder for data of a specific solution instance. The data can be either solution discovery data or KPI data.

Fields

Type	Name	Value or Description
string	InstanceID	The ID of the discovery record
string	SystemUUID	The Universally Unique Identifier (UUID) of the system
string	SolutionName	The name of the solution
string	SolutionUUID	The GUID of the solution (see Solution GUIDs on page 19)
string	ComponentID	The ID of the solution component
datetime	LastUpdateTime	The last time that this data was published to the RCS
string	Data	The data (in an XML string) that was published to the RCS

Methods

None

6.5 Methods

This section describes the Solutions Framework methods.

6.5.1 PlatformDiscovery

This method runs a discovery operation and reports the available software and hardware capabilities of the platform. This method is the API equivalent of the standalone Platform Discovery Utility included in the `PlatformDiscovery` folder.

Syntax

```
uint32 PlatformDiscovery(string DiscoveryData string RCSAddress)
```

Parameters

Name	Type	Value or Description
DiscoveryData	Output	An XML document describing the platform capabilities as reported by the Platform Discovery Utility. For more information, see Platform Discovery Example on page 36.
RCSAddress*	Input	Address of the computer running the RCS to which you want to publish the platform discovery data. Data can only be published to an RCS that is installed in database mode. Using this parameter when the RCS is installed in non-database mode will return an error.
* Optional		

6.5.2 Recalculate

This method sends a request to the RCS to recalculate the platform discovery summary report from existing data.

Syntax

```
uint32 Recalculate()
```

Parameters

None

6.5.3 RefreshSolutions

This method reloads all the solution plugins.

Syntax

```
uint32 RefreshSolutions()
```

Parameters

None

6.5.4 SolutionDiscovery

This method gets discovery data from the solution.

Syntax

```
uint32 SolutionDiscovery(string DiscoveryData string RCSAddress)
```

Parameters

Name	Type	Value or Description
DiscoveryData	Output	An XML document containing data about the solution as reported by the solution host plugin. For more information, see Solution Discovery Example on page 39.
RCSAddress*	Input	Address of the computer running the RCS to which you want to publish the discovery data. Data can only be published to an RCS that is installed in database mode. Using this parameter when the RCS is installed in non-database mode will return an error.
* Optional		

6.5.5 GetConfiguration

This method gets the current configuration settings that are defined for the solution.

Syntax

```
uint32 GetConfiguration(string Configuration)
```

Parameters

Name	Type	Value or Description
Configuration	Output	An XML document describing the current configuration settings. For information about the format and content of this XML document, refer to the documentation supplied with the solution.

6.5.6 Configure

This method configures the solution with the settings in the specified configuration profile.

Syntax

```
uint32 Configure(string Configuration)
```

Parameters

Name	Type	Value or Description
Configuration	Input	An XML document containing the configuration settings to configure in the solution. For information about the format and content of this XML document, refer to the documentation supplied with the solution.

6.5.7 UnConfigure

This method resets the settings of the solution to its default configuration settings.

Syntax

```
uint32 Unconfigure()
```

Parameters

None

6.5.8 GetKPIs

This method gets the Key Performance Indicators of the solution.

Syntax

```
uint32 GetKPIs(string KPIData string RCSAddress)
```

Parameters

Name	Type	Value or Description
KPIData	Output	An XML document containing the solution KPI data as reported by the solution host plugin. For more information, see KPI Example on page 41.
RCSAddress*	Input	Address of the computer running the RCS to which you want to publish the KPI data. Data can only be published to an RCS that is installed in database mode. Using this parameter when the RCS is installed in non-database mode will return an error.
* Optional		

6.5.9 GetAndResetKPIs

This method gets the Key Performance Indicators of the solution and then resets all the counters.

Syntax

```
uint32 GetKPIs(string KPIData string RCSAddress)
```

Parameters

Name	Type	Value or Description
KPIData	Output	An XML document containing the solution KPI data as reported by the solution host plugin. For more information, see KPI Example on page 41.
RCSAddress*	Input	Address of the computer running the RCS to which you want to publish the KPI data. Data can only be published to an RCS that is installed in database mode. Using this parameter when the RCS is installed in non-database mode will return an error.
* Optional		

6.5.10 ResetKPIs

This method resets the counters of the Key Performance Indicators of the solution.

Syntax

```
uint32 ResetKPIs()
```

Parameters

None

6.5.11 GetAttribute

This method gets data about a specified attribute or feature of the solution.

Syntax

```
uint32 GetAttribute(string TagName, uint8[] Data, uint32 DataType)
```

Parameters

Name	Type	Value or Description
TagName	Input	The name of the attribute/feature
Data	Output	The data in an array of unsigned chars (bytes)
DataType	Output	The type of the data

6.5.12 SetAttribute

This method sets the data of the specified attribute of the solution.

Syntax

```
uint32 SetAttribute(string TagName, uint8[] Data, uint32 DataType)
```

Parameters

Name	Type	Value or Description
TagName	Input	The name of the attribute/feature
Data	Input	The data in an array of unsigned chars (bytes)
DataType	Input	The type of the data

6.5.13 ApplyCommandSync

This method applies the specified command to the solution.

Syntax

```
uint32 ApplyCommandSync(uint32 CommandID, uint8[] InData, uint32 TimeoutInSeconds,
                        uint8[] OutData, uint8[] OutDetailData)
```

Parameters

Name	Type	Value or Description
CommandID	Input	The ID of the command that will be applied to the solution
InData	Input	The input data in an array of unsigned chars (bytes)
TimeoutInSeconds*	Input	The number of seconds to wait before the operations times out. The default value is 60 seconds. Use 0 to specify an infinite waiting period.
OutData	Output	The output data in an array of unsigned chars (bytes)
OuDetailData	Output	The output details data in an array of unsigned chars(bytes)
* Optional		

6.5.14 BackupSolutionSettings

This method makes a backup of the solution settings.

Syntax

```
uint32 BackupSolutionSettings(uint8[] Data)
```

Parameters

Name	Type	Value or Description
Data	Output	The data in an array of unsigned chars (bytes)

6.5.15 RestoreSolutionSettings

This method restores the solution settings from a backup set of data that was previously created using the `BackupSolutionSettings` method.

Syntax

```
uint32 RestoreSolutionSettings(uint8[] Data)
```

Parameters

Name	Type	Value or Description
Data	Input	The data in an array of unsigned chars (bytes)

6.5.16 CountEvents

This method counts the number of events that currently exist for the specified solution component, or all solution components.

Syntax

```
uint32 CountEvents(string ComponentID, string  Count)
```

Parameters

Name	Type	Value or Description
ComponentID	Input	The unique identifier of the solution instance
Count	Output	The number of events

6.5.17 ClearAllEvents

This method clears all events of the specified solution component.

Syntax

```
uint32 ClearAllEvents(string  ComponentID)
```

Parameters

Name	Type	Value or Description
ComponentID	Input	The unique identifier of the solution instance

6.5.18 EnumeratePermissions

This method returns all the users and group accounts that have permissions on the solution.

Note:

The list returned by the `EnumeratePermissions` method will include all users and groups that have the necessary "Allow" permissions. But if a user or group also have existing "Deny" permissions (in addition to the "Allow" permissions) they will also appear in the list.

Syntax

```
uint32 EnumerateWMIPermissions(string[] NameOrSID, string[] Reserved, string[] Description)
```

Parameters

Name	Type	Value or Description
NameOrSID	Output	A list containing the SIDs and names of all the user and group accounts that have permissions on this solution
Reserved	Output	Reserved for future use (currently will be ignored)
Description	Output	A string containing detailed information about the success or failure of the operation

6.5.19 AddPermissions

This method gives a specified user or group account all the necessary WMI, DCOM, and folder permissions required to manage a solution.

Note:

This method only sets the "Allow" permissions. No changes are made to any existing "Deny" permissions.

Syntax

```
uint32 AddPermissions(string NameOrSID, string Reserved, string Description)
```

Parameters

Name	Type	Value or Description
NameOrSID	Input	The name or SID of the user or group account for which to add the permissions
Reserved	Input	Reserved for future use (currently will be ignored)
Description	Output	A string containing detailed information about the success or failure of the operation

6.5.20 RemovePermissions

This method removes the WMI, DCOM, and folder permissions required to manage a solution from a specified user or group account.

Note:

This method only removes the "Allow" permissions that were added by the `AddPermissions` method. No changes are made to any existing "Deny" permissions.

Syntax

```
uint32 RemovePermissions(string NameOrSID, boolean JustPluginPermissions, string Reserved,
string Description)
```

Parameters

Name	Type	Value or Description
NameOrSID	Input	The name or SID of the user or group account from which to remove the permissions. **Note:** You cannot remove permissions from these built-in accounts: SYSTEM, NETWORK SERVICE, and Administrators.
JustPluginPermissions	Boolean	Defines if you want to remove permissions only for this solution, or for the entire Solutions Framework as well. Valid values: • True – Remove permissions only from the solution on which the method is executed. Permissions for other solutions will not be removed. • False – Remove all WMI, DCOM, and folder permissions required to use the Solutions Framework. Setting this value means that the user or group account will not be able to manage any solutions.
Reserved	Input	Reserved for future use (currently will be ignored)
Description	Output	A string containing detailed information about the success or failure of the operation

7 XML Reference

This section describes the data protocols and format of the XML documents used by the Framework for platform discovery, solution discovery, and KPI. For information about XML documents of configuration profiles, refer to the documentation provided with the solution.

7.1 XML Data Protocol

This section describes the Document Type Definitions (DTDs) used by the Solutions Framework API to validate XML documents.

7.1.1 Platform Discovery DTD

Platform Discovery XML documents are validated using this DTD.

```
<!ELEMENT PlatformDiscovery ( Framework, Solution+ ) >
<!ATTLIST  PlatformDiscovery version NMTOKEN #REQUIRED >

<!ELEMENT Framework EMPTY >
<!ATTLIST  Framework available NMTOKEN #REQUIRED >
<!ATTLIST  Framework version NMTOKEN #REQUIRED >

<!ELEMENT Solution ( Hardware+, Software+, FrameworkPlugin )  >
<!ATTLIST  Solution uuid NMTOKEN #REQUIRED >
<!ATTLIST  Solution name NMTOKEN #REQUIRED >
<!ATTLIST  Solution exist NMTOKEN #REQUIRED >
<!ATTLIST  Solution managed NMTOKEN #REQUIRED >
<!ATTLIST  Solution state NMTOKEN #REQUIRED >

<!ELEMENT Hardware EMPTY >
<!ATTLIST  Hardware version NMTOKEN #REQUIRED >

<!ELEMENT Software EMPTY >
<!ATTLIST  Software name NMTOKEN #REQUIRED >
<!ATTLIST  Software required NMTOKEN #REQUIRED >
<!ATTLIST  Software available NMTOKEN #REQUIRED >
<!ATTLIST  Software version NMTOKEN #REQUIRED >

<!ELEMENT FrameworkPlugin EMPTY >
<!ATTLIST  FrameworkPlugin required NMTOKEN #REQUIRED >
<!ATTLIST  FrameworkPlugin available NMTOKEN #REQUIRED >
<!ATTLIST  FrameworkPlugin version NMTOKEN #REQUIRED >
```

For more information about this XML document, see Platform Discovery Example on page 36.

7.1.2 Solution Discovery and KPI DTD

Solution Discovery XML documents and KPI XML documents are validated using this DTD.

```
<!ELEMENT Management( ManagedElement+ ) >
<!ATTLIST Management version CDATA #REQUIRED >

<!ELEMENT ManagedElement (Commands?, Discovery?, KPI?, ManagedElement* ) >
<!ATTLIST ManagedElement component-id NMTOKEN #REQUIRED >
<!ATTLIST ManagedElement name CDATA #IMPLIED >
<!ATTLIST ManagedElement description CDATA #IMPLIED >
<!ATTLIST ManagedElement type NMTOKEN #REQUIRED >
<!ATTLIST ManagedElement capabilities NMTOKENS #IMPLIED >

<!ELEMENT Commands (Command+) >
<!ELEMENT Command ( #PCDATA ) >
<!ELEMENT Command command-id CDATA #REQUIRED >

<!ELEMENT Discovery ( ANY ) >

<!ELEMENT KPI ( ANY ) >
```

For more information about these XML documents, see:

- Solution Discovery Example on page 39
- KPI Example on page 41

7.2 Platform Discovery Example

This is an example of a Platform Discovery XML document.


```xml
<PlatformDiscovery version="10.0.11.0">
    <Framework available="true" version="10.0.11"/>
    <Solution uuid="a543b148-407c-419d-ab91-6a26af664828"
            name="Intel(R) Smart Connect Technology" exist="false" state="not supported"/>
    <Solution uuid="a5e80b51-5429-4904-b412-589470884e97"
            name="Intel(R) AMT" exist="true" managed="true" state="configured">
        <Hardware version="Intel(R) ME- 7.1.20.1119"/>
        <Software name="Intel(R) Management Engine Interface"
            required="true" available="true" version="7.1.70.1198"/>
        <FrameworkPlugin available="false"/>
    </Solution>
    <Solution uuid="0b93a9cb-64b1-43ed-bc33-e5b0b8692694"
            name="Intel(R) SSD Pro Series" exist="true" managed="true" state="ready">
        <Hardware version="INTEL SSDSC2BF180A4L- LSTi"/>
        <FrameworkPlugin available="true" version="1.0.3.400"/>
    </Solution>
    <Solution uuid="8f14b6c3-9981-4347-8f05-08582591f486"
            name="Intel(R) Anti-Theft Technology" exist="true" managed="false" state="not-enrolled">
        <Hardware version="TDT Client version- 3"/>
        <FrameworkPlugin available="false"/>
    </Solution>
    <Solution uuid="8d101420-2d30-4ff4-b18a-1da73e67d0ef"
            name="Location Based Service" exist="false" state="not supported"/>
    <Solution uuid="e8c6bb4d-eda0-4859-bc7a-197cf252cf88"
            name="Intel(R) Enterprise Digital Fence" exist="true" managed="false" state="supported">
        <Hardware version="INTEL SSDSC2BF180A4L- LSTi"/>
        <Software name="Intel(R) Enterprise Digital Fence Service" required="true" available="false"/>
        <FrameworkPlugin available="false" />
    </Solution>
    <Solution uuid="7144b5b8-6883-44a9-80a2-c112ce6d6d7f"
            name="Intel(R) IPT with OTP" exist="true" managed="false" state="supported">
        <Hardware version="Intel(R) ME- 7.1.20.1119"/>
        <Hardware version="Intel(R) Core(TM) i5-2540M CPU @ 2.60GHz"/>
        <Software name="Intel(R) Management Engine Interface" required="true"
            available="true" version="7.1.70.1198"/>
        <FrameworkPlugin available="false"/>
    </Solution>
    <Solution uuid="34e60d6a-457a-483b-82ee-cb4f5ba6f542"
        name="Intel(R) IPT with PKI" exist="false" state="not supported"/>
    <Solution uuid="958f5db1-b722-4fd4-8d5d-68c572751ec3"
        name="Intel(R) WiGig" exist="false" state="not supported"/>
</PlatformDiscovery>
```

Element	Description
<PlatformDiscovery>	A normal (root) element containing all the child elements of the XML document. The version attribute of the start tag of this element defines the version of the Platform Discovery Utility (`PlatformDiscovery.exe`) that created the XML document.
<Framework>	A void element containing attributes describing the Host Solution Manager on the host platform. Possible attributes: • available – Whether the Host Solution Manager is installed (true or false) • version – The version of the Host Solution Manager

<Solution>	A normal element containing attributes and child elements describing a specific solution. Possible attributes of the start tag of this element:
	• uuid – The Globally Unique Identifier (GUID) of the solution
	• name – The descriptive name of the solution
	• exist – Indicates if the platform includes all the necessary hardware and firmware required by the solution (true or false)
	• managed – Indicates if the Solutions Framework is capable of managing the solution (true or false). For solutions, this element indicates if a host plugin is installed and working correctly. For Intel AMT, the value of this element is true if Intel AMT is in a "configured" state. (Intel AMT is not currently a "solution").
	• state – The state of the solution. The values of this attribute will vary according to the solution. Here are some examples:
	  • Intel AMT: {not supported | unconfigured | configured }
	  • Intel SSD Pro Series: {not supported | not-ready | ready}
	  • Intel Enterprise Digital Fence {not supported | supported | enabled | disabled}
	  • Intel Smart Connect Technology: {not supported | enabled | disabled}
	  • Intel Anti-Theft Technology {not supported | not-enrolled | enrolled}
	Note: A <Solution> element is created for all solutions that the version of the Platform Discovery Utility supports, even if they do not exist on the platform. This means you can easily identify exactly which solutions a specific version of the Platform Discovery Utility tried to detect on the platform.

<Hardware>	A void element containing attributes describing the hardware of the solution. Possible attributes: • version – The version of the detected hardware **Note:** Some solutions can include more than one hardware instance (for example, the platform might have two Intel SSD Pro Series disks). A <Hardware> element is created for each instance of hardware that is detected.

Element	Description
<Software>	A void element containing attributes describing the software of the solution. Possible attributes: • name – The name of the software • required – Whether the solution requires software to operate (true or false) • available – Whether the software is installed on the platform (true or false) • version – The version of the software (if the software is installed)
<FrameworkPlugin>	A void element containing attributes describing the host plugin of the solution. Possible attributes: • available – Whether a host plugin for this solution is installed on the platform (true or false) • version – The version of the host plugin (if a host plugin is installed)

7.3 Solution Discovery Example

This is a partial example of a Solution Discovery XML document (using Intel SSD Pro Series as an example).

```
<Management version="1">
    <ManagedElement component-id="0b93a9cb-64b1-43ed-bc33-e5b0b8692694"
                    name="ssdproseries" type="solution"
                    capabilities="profile attribute discover kpi">
        <ManagedElement component-id="XXXXXXXXXXXXXXXXXX"
                        name="Drive0" type="drive"
                        capabilities="kpi commands">
            <Commands>
                <Command command-id="3">Identify Device</Command>
                <Command command-id="1">Read Scan</Command>
            </Commands>
            <Discovery>
                <model>INTEL SSDSC2BW240A4</model>
                <serial>XXXXXXXXXXXXXXXXXX</serial>
                <firmware>LA7i</firmware>
                ...................................
            </Discovery>
        </ManagedElement>
        <ManagedElement component-id="C" name="Partition1" type="partition"
                        capabilities="commands">
            <Commands>
                <Command command-id="2">Data Integrity</Command>
                <Command command-id="0">Intel SSD Optimizer</Command>
            </Commands>
            <Discovery>
                <path>C</path>
                <logical_size>223.08 GB</logical_size>
                <free_space>128.58 GB</free_space>
                ...............................
            </Discovery>
        </ManagedElement>
    </ManagedElement>
</Management>
```

Element	Description
<Management>	A normal (root) element containing all the child elements of the XML document. The version attribute of the start tag of this element defines the version of the Solutions Framework API to which this document applies.

Element	Description

<ManagedElement>	A normal element containing attributes and child elements describing a specific solution component. Possible attributes of the start tag of this element:
	• component-id - The ID of the solution component
	• name - The descriptive name of the solution component
	• description - Additional text describing the solution component
	• type - The type of solution component
	• capabilities - A list of the Solutions Framework capabilities supported by this solution component:
	  • discover - Supports solution discovery
	  • kpi - Supports discovery of KPI data
	  • profile - Supports configuration using a profile
	  • attribute - Supports getting/setting attributes
	  • commands - Supports applying commands

<Commands>	A normal element containing child <Command> elements describing commands that are supported by the solution component. Each solution can define their own list of supported commands. For more information, refer to the documentation provided with the solution. **Note:** A <Command> element is created for each command that the solution component supports.
<Discovery>	An XML data island containing solution discovery data of the solution component. The data that these XML data islands contain is defined by the solution plugin. The only requirement is that the content of these XML data islands must be well formed XML. For more information, refer to the documentation provided with the solution.

7.4 KPI Example

This is a partial example of a KPI XML document (using Intel SSD Pro Series as an example).

```
<Management version="1">
    <ManagedElement component-id="0b93a9cb-64b1-43ed-bc33-e5b0b8692694"
                    name="ssdproseries" type="solution">
        <ManagedElement component-id="XXXXXXXXXXXXXXXXXX"
                        name="Drive0" type="drive">
            <KPI>
                <KPI>
                    <id>05</id>
                    <description>Re-allocated Sector Count</description>
                    <raw>0x0</raw>
                    <value>100</value>
                    <threshold>0</threshold>
                    <worst>100</worst>
                    <status>0</status>
                    <pass>Pass</pass>
                </KPI>
                ...................................
            </KPI>
        </ManagedElement>
    </ManagedElement>
</Management>
```

Element	Description
<Management>	A normal (root) element containing all the child elements of the XML document. The version attribute of the start tag of this element defines the version of the Solutions Framework API to which this document applies.
<ManagedElement>	A normal element containing attributes and child elements describing a specific solution component. Possible attributes of the start tag of this element: • component-id - The ID of the solution component • name - The descriptive name of the solution component • description - Additional text describing the solution component • type - The type of solution component
<KPI>	An XML data island containing KPI data of the solution component. The data that these XML data islands contain is defined by the solution plugin. The only requirement is that the content of these XML data islands must be well formed XML. For more information, refer to the documentation provided with the solution.