RAYMANAGESOFT i

A New Generation of Software Deployment

Preferences for Managed Devices
RayManageSofti 11.4

RAYVENTORY
is part of RayManageSofti

RayManageSofti is part of RaySuite.

rayNET

Preferences for Managed Devices for release 11.4

Raynet GmbH Germany
See our website for locations.

www.raynet.de

Contents

Introduction

This manual describes the preferences that enable you to configure RayManageSofti for your environment.
This manual will help you to:

- Discover how to set and use RayManageSofti preferences.
- Using managed device settings packages.
- In the Windows registry.
- In command line tools.
- In packages, as registry entries or project variables.
- In configuration files.
- Identify preferences associated with particular RayManageSofti behavior.
- Find details of individual preferences.

RayManageSofti Preferences

RayManageSofti provides a large selection of preferences that you can set to control RayManageSofti behavior. Most of these are managed device preferences, which can be set on individual managed devices, or can be embedded in a package to control the behavior of the package on all managed devices.

There are also preferences you can set on administration servers and/or distribution servers.

This volume describes the preferences you can set on the managed device.

Each environment is unique, and the examples in this manual may not suit your implementation. If you are in doubt, Raynet strongly recommends that you speak with your Raynet support representative before you start changing your settings.

Installation Defaults

When RayManageSofti is installed, it sets up default values for many preferences. This volume includes details of the installation default for each preference on the managed device.

Before You Proceed

This document assumes the following knowledge:

- You are familiar with the operating systems on which RayManageSofti is installed.
- You have read the *RMS Release Notes* of RayManageSofti.
- You have a working installation of Microsoft® Active Directory.

Be aware:
If you intend to use Deployment Manager for ongoing software management, it is strongly recommended that you use Active Directory. Deployment Manager interacts with Active Directory to extend Windows Group Policy, providing sophisticated and efficient software management.

As an alternative for smaller enterprises, Deployment Manager does provide a simple native policy mechanism. This alternative is less flexible than using Active Directory..

- You have a computer with the Microsoft® Windows Server® operating system (2003, 2008, or 2012) installed that has access to your Active Directory, and that you can use as your RayManageSofti administration server.

- You are familiar with Microsoft Group Policy concepts.
- You are familiar with the concepts relating to, and use of, Microsoft Management Console (MMC).

Learning More About Preferences

The remainder of this volume contains:

- An overview of how preferences are set, evaluated, and locked.
- Details of how RayManageSofti uses the Windows registry for setting and reading preferences.
- An explanation of using packaging project variables to set preferences.
- Details about preferences with command-line tools.
- An introduction to the global configuration file.
- Lists of preferences according to the behavior that they modify.
- An alphabetical list of preferences, which outlines:
  - The purpose of the preference.
  - Defaults, values or ranges, and example values.
  - The methods by which the preference can be set.
  - Details applicable to each definition method relevant to the preference.

Additional Resources

The flexibility of the Deployment Manager system means that there are many different ways that you can configure it. As well, its interaction with many other parts of your infrastructure means that a successful implementation requires careful planning and a broad understanding of all the elements.

For these reasons, we strongly recommend that you seek assistance from trained Raynet consultants in preparing your implementation.

In addition to the expertise of Raynet consultants and the extensive product documentation, there are several other resources available to you.

- The *RMS Getting Started* provides an overview of documentation contents. Go there first to explore the available guides and references delivered along with RayManageSofti.

- The *RMS Release Notes*, included on the installation media, cover compatibility issues and late-breaking news about this release.

- *Application Help* is installed on the administration server and by default on managed devices:

  - Help on the administration server is intended for qualified personnel operating various aspects of the RayManageSofti system: administrators, package creators, release managers, and the like. Much of the content of the application help is replicated in this manual, but each contains some additional material.

  - Help on managed devices is intended for end-users, and provides a quick introduction to the self-service capabilities that RayManageSofti provides for optional installations.

- The support section of the product website *www.RayManageSoft.com* includes lists of what's new, known problems, and problems repaired for each release of RayManageSofti. There is also an extensive, searchable knowledge base of specialized articles and notes about many aspects of the product. Your enterprise should already have an account name and password for the support area. If not, contact your Raynet representative.

- Raynet and its partners offer a range of training courses that can also be customized to meet your requirements. For more information on these courses, speak with your Raynet consultant.

Feature Availability

With RayManageSofti 11.4 infinity, Raynet introduces a re-invented license control system. By implementing three individually settable options per RayManageSofti feature, customers can easily configure the most suitable RMS toolbox for their corporate needs. Each feature can be enabled, disabled, or set into demo mode. The demo mode allows for preview feature-testing, whilst preventing full feature usage. Features in demo-mode are marked by a flag to indicate the limited state. Some features, such as the AD policy editor, base on the availability of others (e. g. the policy snap-in).
This document describes RayManageSofti in its full stage of expansion. Therefore, if an individual license does not cover all features, readers might miss snap-ins, workflows, wizards etc. as within their actual user interface.

RMS consultants and sales representatives are the right contact persons for a list of available combinations and for advice on the preparation of custom-made RayManageSofti 11.4 infinity licenses.

Documentation Requests

We welcome your suggestions and input on the various documentation resources available with RayManageSofti and its components. Your comments and requests can be forwarded through your Raynet support representative.

Manual Conventions

The following typesetting conventions are used in this manual:

- Cross references to headings or chapters in this manual, or to other manuals, are shown in italics:
  "See *RMS Reference: System Reference* for..."
- Quotations from your computer screen (titles, prompts, and so on) are shown in bold:
  "The **Receive Packages Wizard** appears."
- Code syntax, file samples, directory paths, entries that you may type on screen, and the like are shown in a monospaced font:
  "The default directory is `C:\ManageSoft...`"
- Italics may also be used for emphasis: "This manual is *not* intended..."
- Bold may also be used for inline headings: "**Target**: Indicates a target frame..."

Two note formats are used in RayManageSofti documentation

This is the basic format for giving additional information to the current topic.

It can come with four different headings:

Be aware:
This note format contains important information related to your current activity. You should not skip over this text.

Note:
This format is used for items of interest that relate to the current discussion.

Best practice:
If there is a best practice approach to the current topic you can decide if you want to follow it, or stick to your own plan.

Tip:
Tips are designed to help you find the easiest and quickest way to work with RayManageSofti.

The second format is for very serious alerts.

WARNING
The information here may save you from data loss. Pay particular attention.

Registry

In this manual, registry keys are preceded by the text [Registry]. This text represents the location of all RayManageSofti registry entries in the registry:

- On Windows administration servers, distribution servers and managed devices, RayManageSofti registry entries are usually stored under the key:

  ```
  HKEY_LOCAL_MACHINE\SOFTWARE\ManageSoft Corp\
  ```

- On non-Windows managed devices, registry entries are stored in the `/var/opt/managesoft/etc/config.ini` file. Within this file, registry keys are shown in square brackets.

  For example: `[ManageSoft\Usage Agent\CurrentVersion]`.
  The lines below each key show the registry entries set under that key.
  For more information about RayManageSofti preferences and the registry, see *RMS Reference: Preferences* for Managed Devices.

Example usage

When the manual refers to the registry entry `[Registry]\Launcher\CurrentVersion\ForceReboot`, this represents:

- In the Windows registry, `HKEY_LOCAL_MACHINE\SOFTWARE\ManageSoftCorp\ManageSoft\Launcher\CurrentVersion\ForceReboot`

- On non-Windows devices, the `ForceReboot=True|False` line in the `config.ini` file, located under the section heading [`ManageSoft\Launcher\CurrentVersion`].

Setting, Fixing, and Evaluating

In this chapter you will:

- Discover the range of methods available for setting preferences.
- Find out how to lock a preference to stop it being overridden.
- Learn about the evaluation order of the various method of setting preferences.
- Read about the relationship between preferences and environment variables.

Ways of Setting Preferences

Preferences can be set in a number of ways:

- Defined in managed device settings packages that are assigned to policy and distributed and installed using the same methods as standard software packages.
- Set in the Windows registry, under the Computer or User hives.
- Set as arguments in RayManageSofti command line tools.
- Defined as project variables in packages.
- Set directly in the global configuration file allowing you to specify the settings in a central location.

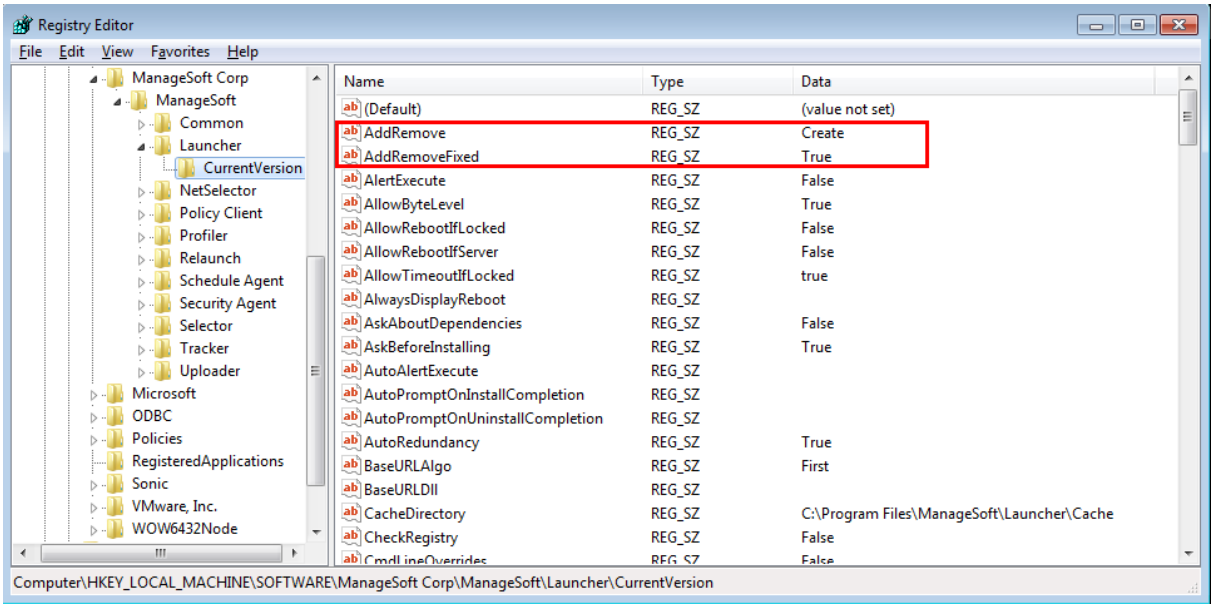
All methods of setting preferences are referenced in this manual.

Fixing Managed Device Preferences

It is possible to define fixed preferences that cannot be modified by users who do not have administrator permissions.
This can be done for any RayManageSofti registry preference on a managed device by creating another entry in the registry with the same name as the setting, suffixed with `"Fixed"` or `"Fxd"` and assigning the value `True` to the new entry.

Example

You decide that all packages installed by RayManageSofti *must* create an entry in the Add/Remove Programs control panel applet. First, you set the `AddRemove` registry entry to `Create`, then you create an entry in the same location, called `AddRemoveFixed` (or `AddRemoveFxd`) and set it to `True`.



Be aware:
Fixed preferences can be over-ridden by command line options if the `CmdLineOverrides` installation preference is set to `True` in the registry. See *CmdLineOverrides* within the *Alphabetical listing of preferences*.

Evaluating RayManageSofti Preferences

RayManageSofti refers to a number of locations when evaluating preference settings on a managed device. The order of precedence is as follows (highest to lowest):

1. Command line arguments.
2. Any preferences read from *UserAlternateRegistryHive*.
3. User preferences, taken from:
   `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\...`
4. Any preferences read from *MachineAlternateRegistryHive*.
5. Computer preferences, taken from one of the locations described in *[Registry]*.
6. Network preferences, taken from the file on the network specified by URL or UNC in the registry setting *GlobalConfigSource*.
7. RayManageSofti factory defaults.

Be aware:
When preferences are defined as "fixed entries" (see above for details) higher precedence settings are ignored. For example, if a setting is fixed through Network preferences, then User settings, Computer settings, and command line arguments are ignored.

Also be aware:
The fact that the system environment does not appear in this list is significant. Preference values are *not* retrieved from the system environment for the purposes of controlling product or package behavior. In some (rare) cases, the system environment is queried when no other value can be determined for a project variable. You should not rely on this behavior, but instead ensure that values are set for all project variables you use.

Alternative Registry Hives

In order to enable handy registry setting evaluation, RayManageSofti uses the registry entries *UserAlternateRegistryHive* and *MachineAlternateRegistryHive*. For registry testing, copy the registry content from the default *[Registry]* hive to a separate test hive, and update the value of the respective alternate registry hive entry. The changes that are applied to the alternate path will have no effect on your productive set of registry settings, and you can switch back to them by simply setting the default values for UserAlternateRegistryHive *and* MachineAlternateRegistryHive. As mentioned above, the alternate hive is always evaluated with a higher precedence than the standard hive.

Preferences in Managed Device Settings Packages

When a managed device updates policy, any new or updated managed device settings packages that apply to the managed device are installed.

These packages modify the computer preferences, set in the registry.

The precedence order listed earlier then applies.

If multiple managed device settings packages are included in the merged policy for a managed device, these packages are installed in the sequence in which they appear in the merged policy (`.npl`) file. Each new managed device settings packages overrides the settings applied from the previous managed device settings package.See the *Managed device settings* chapter of the *RMS Software Deployment* for detailed information on managed device settings packages.

Preferences in Software Packages

Some preference settings can also be defined as project variables in RayManageSofti software packages. These project variables can be assigned special behaviors that control whether they override other preferences settings (for example, *Use value if not set on managed device*).

Because of these controls, packages are not included in the above list of precedency, as RayManageSofti must evaluate the precedence of preferences in each package individually.

Managed Device Settings Packages

The most common method for managing managed device preferences is through the use of managed device settings packages. These settings packages identify specific preference values to be applied to managed devices. They are applicable across all supported platforms.

About Settings Packages

Settings packages are used to manage and distribute preference settings to managed devices across an enterprise.

Creating Settings Packages

Using the RayManageSofti console on the administration server, you can create a series of settings packages. Each settings package contains a list of preferences and values to be set on managed devices.

The settings packages are stored on the administration server and distributed to your distribution locations for retrieval by managed devices. You can update and redistribute settings packages as often as required.

Targeting Users and Computers

To target settings packages to users and computers, you need to assign the settings packages to Group Policy. This is done from:

- The **Devices** node in the RayManageSofti console tree.
- The RayManageSofti Software Management snap-in to the **Group Policy Object** editor, in the same way that software packages and schedules are assigned to policy.

When policy is merged, and the RayManageSofti agent calculates the set of packages that should apply to each computer or user, it includes the settings packages in the calculation.

When each managed device performs its scheduled policy update, the managed device retrieves the appropriate settings package from the closest distribution location and installs it. The installation process sets the `HKEY_LOCAL_MACHINE` registry keys for the preferences included in the settings package.

To Find Out More...

Managed device settings packages are described in the *Managed device settings* chapter of the *RMS Software Deployment*.

Preferences in the Registry

Preferences for RayManageSofti on Windows managed devices are stored in the Windows registry. While some settings are configured during the installation process, and should not be altered, many others can be changed to suit the needs of your organization.

Where are Preferences Stored in the Registry

By default, registry entries are stored as described in *Registry*.
Settings under this location affect the operation of RayManageSofti for all users with accounts on the managed device. You can, however, customize registry entries on a user by user basis by creating equivalent entries in the following location on each Windows managed device:
`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\`

See also the preferences *MachineAlternateRegistryHive* and *UserAlternateRegistryHive* in the *Alphabetical listing of preferences* chapter for a fuller description.

Values for most preferences can be configured in the registry under a key that ends with something like
`…ManageSoft Corp\ManageSoft\<component>\CurrentVersion`.

Each of these settings is specific to the particular component. Such preferences can generally also have default values for all or multiple components configured under
`…ManageSoft Corp\ManageSoft\Common`.

For example, the MachineId preference can be set for specific components in the following entries:
`…\ManageSoft Corp\ManageSoft\Launcher\CurrentVersion\MachineId`
`…\ManageSoft Corp\ManageSoft\Selector\CurrentVersion\MachineId`
`…\ManageSoft Corp\ManageSoft\Usage Agent\CurrentVersion\MachineId`
`…\ManageSoft Corp\ManageSoft\Policy Client\CurrentVersion\MachineId`
`…\ManageSoft Corp\ManageSoft\Schedule Agent\CurrentVersion\MachineId`.

MachineId can also be set in the following location to provide a default value for any component that does not explicitly have a value configured:
`…\ManageSoft Corp\ManageSoft\Common\MachineId`.

How are Managed Device Registry Entries Set?

Most registry entries are automatically created during installation of RayManageSofti on the managed device (either prior to or when the RayManageSofti installation agent runs for the first time on the managed device).

However, some entries can be manually configured:

- By a RayManageSofti package
- By the `mgssetup.ini` configuration file used in automatic adoption of managed devices
- By any other mechanism that affects the registry.

The reference section at the end of this chapter lists each preference and also indicates whether the entry is automatically created during RayManageSofti installation or only set manually.

How Does RayManageSoft Use the Registry?

When RayManageSofti performs an action on a managed device that has preferences associated with it, it checks the registry when evaluating the value of a preference.

You can also alter the registry by setting values in a package. In packages, you may define registry keys and values. If you are defining a registry value, you must first define the key under which it is located. You may define your own registry keys and values, or re-create (override) RayManageSofti registry entries.

You may also retrieve a value from the registry, and place it into a project variable. You can then build in your own logic to determine actions based on the value in the registry.

Taking an Inventory of Registry Keys

RayManageSofti provides the ability to track and report on registry entry values from managed devices.
To track registry entries on managed devices, you must configure Windows registry settings both on the managed device (to identify which areas of the registry are to be tracked) and on the administration server on which inventory resolution occurs (to identify what data about the tracked registry areas should be included in reports).

On the Managed Device

The inventory agent's `IncludeRegistryKey` preference is used on the managed device to control which areas of the registry are to be collected by RayManageSofti as part of inventory. This preference is typically configured on the inventory agent command line or by configuring the registry entry `[Registry]\ManageSoft\Tracker\CurrentVersion\IncludeRegistryKey` on each managed device.

The default value of `IncludeRegistryKey` is configured to collect only information from the uninstall section of the registry (Add/Remove entries). This behavior is equivalent to configuring `IncludeRegistryKey` with the value:
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall.`

Enter the keys you want to track in `IncludeRegistryKey`. For example, to track settings related to the installation of Adobe Photoshop 7.0 (while maintaining collection of the uninstall section of the registry), set `IncludeRegistryKey` to:
`HKEY_LOCAL_MACHINE\SOFTWARE\Adobe\Photoshop\7.0\ApplicationPath,`
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall.`

Be aware:
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall` must be included explicitly in any `IncludeRegistryKey` setting if you wish to track this section of the registry as is normally the case.

On the Administration Server Where the Inventory is Being

Tracked registry settings are stored in the RayManageSofti database according to registry settings specified on the administration server. Once in the database, they are available for reporting under the hardware report pages.

The relevant settings on the administration server appear under the following keys:
`[Registry]\ManageSoft\Reporter\CurrentVersion,`
`[Registry]\ManageSoft\Reporter\CurrentVersion\Registry.`

Be aware:
Collecting large amounts of registry information may significantly degrade the performance of managed device inventory resolution on the administration server. For this reason, the amount of registry information you collect should be minimized and restricted to values that you really do require.

Configuring Reporting for Specific Entries

To configure reporting of specific registry keys as hardware classes, create sub-keys under `[Registry]\ManageSoft\Reporter\Current Version\Registry` using any name that is meaningful to you, and place the following settings under each sub-key:

- **InterestedPath**
  Specifies the path of the key that will be reported.

- **InterestedName**
  Names the entry within the specified key that will be reported.

- **ClassType**
  Specifies the name that will be used as the hardware class for this registry entry. The hardware class is displayed in hardware reports.

- **Description**
  Specifies the name (stored in the `HardwareObject.HardwareName` field in the RayManageSofti database) to be used for instances of `ClassType` that will be reported. `Description` may be set to any one of:

  - **$Path**: Uses the full path of the registry entry as the instance name.
  - **$Value**: Uses the value of the registry entry as the instance name.

    If you set `Description` to `$Value` and you are collecting differential inventories from your managed devices, deleted or changed registry entries will not be reflected in the RayManageSofti database until the next full inventory is received.
  - **$ShortName**: Uses the registry entry's name as the instance name.
  - Anything else: Uses the exact value specified here as the instance name.

Example

To report on the value of the `[Registry]\ManageSoft\Launcher\CurrentVersion\LogFile` entry, configure the following registry entries under `[Registry]\ManageSoft\Reporter\CurrentVersion\Registry\MDLauncherLogFile` on your administration server:

Entry name	Value
ClassType	`Win32_MDLauncher`
Description	`MDLauncher`
InterestedName	`LogFile`
InterestedPath	`HKLM\SOFTWARE\ManageSoft Corp\ManageSoft\Launcher\CurrentVersion` Note that you may use the full (`HKEY_LOCAL_MACHINE`) or short (`HKLM`) form of the key root.

Receipt and processing of any new inventory reports containing the relevant registry key will now show the data in Deployment Manager hardware reports.

Configuring Reporting for Unprocessed Entries

Apart from the registry entries which are used by software tracking (the uninstall keys) or by explicitly configured registry reporting (described above), you might wish to report on all other entries that have been returned in inventories. These will be stored as a hardware class, and are available for reporting under the hardware report pages.

If you wish to track all entries that are otherwise unused, you can set the `SaveRegistry` and `SaveRegistryClass` entries under `HKEY_LOCAL_MACHINE\SOFTWARE\ManageSoft Corp\ManageSoft\Reporter\CurrentVersion`. To do so, complete these steps on the administration server:

1. Create `SaveRegistry` and set it to `True`. This tells RayManageSofti to retain all unused registry entries.
2. Set `SaveRegistryClass` to a suitable value for the `Hardware Class` column in the reports. If you do not specify a value, the default value `Win32_Registry` is used.
3. If you want to exclude certain registry keys from being imported into the hardware records of the RayManageSofti database, you can list these in the following registry key:`HKEY_LOCAL_MACHINE\SOFTWARE\ManageSoft Corp\ManageSoft\Reporter\CurrentVersion\SaveRegistryExclude`.

   `SaveRegistryExclude` is set to `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall` by default.

   This value excludes **Add/Remove Programs** entries from the registry keys imported in the hardware class you specify in `SaveRegistryClass`. These **Add/Remove Programs** entries are defined by default in the `IncludeRegistryKey` settings on managed devices for import as software inventory records (see *On the managed device*).

Be aware:
1. Values for any registry keys may optionally use the `HKLM` short form for the key root value.
2. If you add to the `SaveRegistryExclude` list, be sure to keep the default value present as well so that you do not include software inventory in hardware reporting.
3. To list multiple registry keys for exclusion, separate each key by a comma (,).

Configuring Settings to Differentiate 32-bit and 64-bit Registry Keys

To enable you to differentiate 32-bit and 64-bit registry keys inventoried on a single computer, a " **(64)**" suffix is added to the **Description** field of the 64-bit registry keys when they are imported into the RayManageSofti database.

To change this suffix, or to add a suffix to the 32-bit registry keys:

1. Access the Deployment Manager Configuration Tool snap-in using one of the methods outlined in the *Configuring your administration server* chapter of *RMS Configuration*.
2. Expand the Deployment Manager **Configuration** node.
   Information is displayed in the details pane.
3. Select the **Reporting** node in the console tree.
   The report configuration settings are displayed in the details pane.
4. To configure the suffix for 32-bit registry keys:

- Select the **Suffix for 32-bit Description registry keys** setting.
  Details of this setting are displayed below the list of settings.
- In the **Suffix for 32-bit Description registry keys** field, enter the characters that will be appended to the end of the description for each 32-bit registry key.

5. To configure the suffix for 64-bit registry keys:

- Select the Suffix for **64-bit Description registry keys** setting.
  Details of this setting are displayed below the list of settings.
- In the Suffix for **64-bit Description registry keys** field, enter the characters that will be appended to the end of the description for each 64-bit registry key.

6. Click **OK** to confirm your changes.

Registry Keys and Values in a Package

This section describes how to add registry keys in a RayManageSofti package. Before you start, you must:

- Know the names of the registry keys in the Windows registry tree structure.
- Open the project on the administration server.

To Check the Names of Registry Keys in the Windows Registry

To view the Windows registry:

1. From the **Start** menu select **Run**.
2. In the **Open** field, enter **regdt32** or **regedit**.
3. The **Registry Editor** is displayed.
4. Click on the folder you wish to open.

The registry editor's layout mirrors that of the Windows Explorer; it uses a tree structure to display the relationship between registry keys and their values.

To Add a Registry Key or Value in a RayManageSofti Package

1. Open the project in the RayManageSofti console.
2. From the console tree, expand the **Software** node.
3. Select the package to which you want to add your information.
4. Do one of the following:

- Right-click and select **Edit**.
- Left-click and click the **Edit** button from the sidebar on the right.

The **Edit package properties** dialog is displayed.

5. Open the **resources** tab and select the **registry** sub-section.
6. Click on the **new key** button to add a key to the root, or select an item from the existing folder structure and click the **new key** or **new value** button to add it to the selected item.
   The **registry dialog** is displayed.
7. Set key or value properties within the general and target tab according to your needs.
8. Click **OK**.
   For further details about adding a registry key or value, or for details about changing and deleting registry keys and values in a package, see the *RMS Packaging*.

Preferences Defined as Project Variables

In RayManageSofti, project variables can be used to define strings of text that are used frequently when packaging an application for distribution.
Project variables also allow you to set package processing behavior or use information (such as installation paths) gathered dynamically from the user at the time the application is distributed.

There are three types of project variables that you can use in software packaging:

- *Pre-defined* project variables
- *Property-setting* project variables
- *Custom* project variables.

All three types of project variables operate in the same way. You can use them at appropriate places in your package, and they will be appropriately interpreted by RayManageSofti on managed devices. For details on how to use project variables, refer to the *Using project variables* section of the *RMS Packaging*. The *alphabetical listing of preferences* contains a marker for each preference that can be defined as project variable.

Predefined Project Variables

These project variables are called pre-defined because their values are automatically set. They may be set by RayManageSofti, your operating system, or your network configuration. They are useful in package details and callouts to expand values of the operating system's file locations.
An example of this class is **$(ProgramFiles)** which is usually set to:
`C:\Program Files`

For example, within a RayManageSofti package, you can set an application's install folder to **$(ProgramFiles)**. On one managed device in Australia, RayManageSofti expands the project variable to `C:\Program Files`. On another managed device in Germany, the same project variable may expand to `C:\Programme`. The variable also copes with computers where the operating system has been installed on a different disk drive (such as `D:` or `E:`).

Many RayManageSofti preferences are available as pre-defined project variables, for example:

- **$(WinDirectory)** as `C:\Windows`
- **$(TempDirectory)** as `C:\Users\<USERNAME>\AppData\Local\Temp`
- **$(AppDir)** as default installation folder, which is the same as **$(ProgamFiles)** if not changed by user
- **$(SysDirectory)** as `C:\Windows\System32`

where `<USERNAME>` is the placeholder for the active username of the managed device.

Raynet highly recommends using these project variables in place of hard-coded folder names in package details, to ensure that settings in your application catalog point to the right location, regardless of the configuration of each managed device. During installation of the Windows environment, values are assigned to Windows pre-defined project variables. They may be changed by:

- Using command line parameters in RayManageSofti schedules to alter the project variable values
- Creating a project variable of the same name in your RayManageSofti package and assigning it a value (defined as a custom project variable, as described in the *Custom project variables* section)
- Modifying specific settings in the Windows registry

Each time RayManageSofti installs a package on a managed device, it determines the values of its pre-defined project variables by checking:

- Against the Windows operating system
- For specific project variable values set within the package

The default values for pre-defined project variables may vary according to the version of the Windows operating system operating on your managed devices. Unless otherwise defined, the defaults listed in this document refer to the Windows 7 operating system.

Property-setting Project Variables

Property-setting project variables control the behavior of one or more packages.

Many RayManageSofti preferences can be used as property-setting project variables. You can reference them in package details or in package callouts and RayManageSofti will evaluate the preference at the time of installing the package.

You can also set the preference to a particular value by defining it in the list of project variables in the package details.

Examples of this class of variable include **$(UserInteractionLevel)** and **$(InstallerARPModify)**.

To set one of these project variables within a package, you must define it in the same way as a custom project variable. See the *Custom project variables* section.

Custom Project Variables

Custom project variables are ones that you define in an individual package. If they use the same name as a RayManageSofti preference, the project variable will affect the value of the preference.

When Would I Use a Project Variable?

Project variables are used for three main reasons:

- To avoid using a single string in many places
  Defining a project variable once and using it multiple times not only saves you time, but also reduces the possibility of introducing errors to your project file.

  For example, typing `$(MyApp)` rather than repeatedly typing the string `"C:\Program Files\ManageSoft\Launcher\Cache\MyApplication\ MyApplication.exe"` will not only reduce application packaging time and errors, but also make your project file easier to maintain. You will only need to make changes that affect the application's location in one place.

  Pre-defined and custom project variables are most useful for this purpose.

- To allow flexibility based on the configuration of individual managed devices

  For example, within a RayManageSofti package, you can set an application's install folder to `$(ProgramFiles)`. On one managed device in Australia, RayManageSofti expands the project variable to `C:\Program Files`. On another managed device in Germany, the same project variable may expand to `C:\Programme`.

  Pre-defined project variables are most useful for this purpose.

- To control behavior for an individual package

To set a RayManageSofti preference for an individual package, it must be defined as a project variable in the package. (Registry entries that you define in a package are set after installation of the package, and will therefore be ignored.)

How RayManageSoft Expands Project Variables

When RayManageSofti installs a package on a managed device, it expands project variables you created in the RayManageSofti package. Custom project variables are expanded to the values you assigned to them in the package.

For other project variables, RayManageSofti checks the following settings, in the listed sequence. If the project variable is defined in more than one of these settings, the last value found is used.

- RayManageSofti built-in defaults.
- Network preferences from the *GlobalConfigSource* registry settings.
- Computer registry settings, taken from `[Registry]\ManageSoft\`.
- Computer registry settings, taken from *MachineAlternateRegistryHive*.
- User registry settings, taken from `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\`.
- User registry settings, taken from *UserAlternateRegistryHive*.
- Persistent settings from previous RayManageSofti installations on the managed device. When RayManageSofti installs a package on the managed device, it retains the values assigned to project variables. The next time an action occurs for the same package on the managed device, RayManageSofti uses the same project variable values, unless you specify different values for the project variables on the command line. (See the *Command line tools* chapter of the *RMS System Reference* for details.)
- OSD files (which contain settings explicitly defined in RayManageSofti when the package was created).

To Set a Preference Using Project Variables

To define variables for use in your project file:

1. Open the RayManageSofti console.
2. Expand the **Software** node and browse to the relevant package.
3. Double-click the package to **edit** it.
4. Call the **resources** tab.
5. Click on **variables**.
6. Click on the **Add new variable** button.
   The **New RayManageSofti Variable Properties** dialog is displayed.
7. Enter the details of the variable:
   - In the **Variable name** field, type the preference name to override the value of that preference on the managed device.
   - In the **Value** field, type the text string that the variable represents.

Example

Suppose you are packing an application and you want it to install using a quiet user interaction level. To set the *UserInteractionLevel* preference as a project variable, do this:

a. In the **Variable name** field, type `UserInteractionLevel`.
b. In the **Value** field, type `quiet`.
c. Select **Use value if not set on managed device** if you want to set the variable on the managed device only if it is not already set.
d. Select **Value overrides device settings** if you want to replace any existing settings on the managed device.

Not all project variables can override managed device settings or be set before processing in a package. Some project variables are required before RayManageSofti installs the package (for example,

UserInteractionLevel).

e. Check the **Set variable before processing catalog** box if you want the project variable to be set on the managed device before the package is installed. If you want to use the project variable in a callout, you must check this box to ensure that the project variable is set before the callout begins processing.

8. Click **OK**.

Note:
Typically, variables are set and used only during the installation of the package and will not affect permanent settings on the managed device.

To Use Project Variables in a Package

When you reference a project variable within a RayManageSofti package, you must prefix the project variable name with the **$** symbol and enclose the project variable name in brackets.
The **$** prefix indicates to RayManageSofti that the variable must be expanded when the package is installed.´

Note:
For readability, project variable names are shown with a mix of capital and lower case letters. However, RayManageSofti ignores case when referencing project variables.

You can use project variables instead of text strings in most fields within the RayManageSofti packaging details. The main exceptions to this are:

- Application Details Properties
- URLs.

Syntax for Project Variables

The syntax for referring to project variables is `$(MyApp)`, where `MyApp` is the name of the variable and `$(MyApp)` refers to its value. (The list of pre-defined project variables earlier in this document are shown in the syntax for referencing them, with the **$** prefix and brackets **()**.)
The **$** symbol ensures that the variable will be expanded when the package is installed.
If you need to refer to a file name that begins with or contains the **$** symbol, quote the **$** symbol with a second **$** symbol. For example, a file called `my$tuff.exe` is referred to as `my$$tuff.exe`. For file names that contain a **%** symbol, quote a **%** symbol in front of it. For example, a run command such as `dir/fi$le%.exe` should appear as `dir/fi$$le%%`.exe.

Special Syntax for MSI Preference Project Variables

The syntax for referring to project variables used in relation to Windows Installer is **$(!MyApp)**, where **MyApp** is the name of the variable and **$(!MyApp)** refers to its value.
The **!** symbol (also called the *unquoted value operator*) makes sure that empty strings are not displayed in quotes when a project variable with an empty value is expanded in the `msiexec.exe` command line.

Short File Name Operator

There are occasions when you cannot use long file names. You may be packing applications for use on operating systems that do not support long filenames; or you may be solving problems with applications that do not correctly handle spaces in filenames. For these occasions, use the short filename operator **~** with a project variable to access the short filename version of the path contained in the variable. The short filename operator is placed after the opening parenthesis of the variable reference.
For example, if `$(MyAppPath)` is defined as `C:\Program Files\ManageSoft`, the reference `$(~MyAppPath)` will expand to `C:\Progra~1\Manage~1`.

Using Project Variables in Package Callouts

Callouts are customized pieces of code that are called at a certain point in installation agent actions. For an overview of callouts, see the *RMS Packaging*. This section assumes that you have a basic understanding of callouts.

On managed devices, each time the installation agent is activated, RayManageSofti processes installation functions (including update and self-heal) and execution functions separately. Different sets of preferences are available to each type of function. This means that a preference added during `Install` functions will not be visible during `Launch` or `Uninstall` functions.

You can use the `nDGLauncherSymbols.Item` object and method to access installation agent preferences.

If you want a symbol to remain accessible by all functions (during one instance of the installation agent), you can add a *global* symbol in your script using the `nDGLauncherSymbols.Add` object and method. The name of the symbol must begin with "`Global`.". Any symbol with this prefix is independent of any similar named symbol. For example, `Global.MyVar` and `MyVar` are independent symbols. The `Global` prefix can only be used for symbols defined within the callout script(s).

If your callout will use a project variable defined in a package, you should define the project variable so that it is set on the managed device before the package is processed. This ensures that the project variable can be used in callouts at any time during package processing. To do this when creating a project variable on the administration server, check the Set variable before processing package box in the Project Variables properties dialog. (For more information, see the *RMS Packaging*.)

Preferences in Command Line Tools

RayManageSofti command line features allow you to automate RayManageSofti-related activities through scripts or batch files.

What Do Command Line Tools Do?

Command line tools allow you to interact directly with a RayManageSofti smart agent. Some of the ways you would use command line tools include:

- Creating a batch file to run behind-the-scenes.
- Including the command in a user logon script so that it runs when a user logs in.
- Typing a command on the MS-DOS command line to run it immediately.

For details see the *Command line tools* chapter of the *RMS Reference: System Reference*.

Persistent Managed Device Preferences

When you set a preference on the installation agent command line, the setting is saved in a RayManageSofti symbol file (`.sym`) in the same folder as any details for the application being installed.
These preferences are then used as the default next time an action occurs for the same package, including packages installed using policy. This is referred to as being a *persistent* preference.

This behavior is controlled by the value in the *SaveAllUserSymbols* preference.

- If `True`, then the preference is saved
- If `False`, the setting is used for the particular action in which it is set, but it is not saved

Therefore, when setting a preference on the command line, you need to consider whether you want the preference to be persistent for future actions, or used only for the current action. If you only want the preference to be used for the current action, make sure that `SaveAllUserSymbols` is set to `False` before setting the preference.
See *SaveAllUserSymbols* in the *Alphabetical listing of preferences* chapter for more details about setting of this preference.

Be aware:
When setting preferences on the command line, `SaveAllUserSymbols` must precede other preferences.

To Set Preferences in Command Line Tools

When you use a command line tool, you can set preferences in the command line using command line arguments. For some smart agents, all of the preferences associated with the smart agent can be set in the command line. For other agents, only selected preferences are available on the command line.

Listing of Command Line Tools

The following command line tools make use of RayManageSofti preferences as command line arguments:

- **Adoption agent** (on computers being brought under management)
- **Distribution agent** (on administration servers and distribution servers, and not included in this manual)
- **Installation agent** (on managed devices)
- **Inventory agent** (on managed devices)
- **Peer download agent** (on managed devices)
- **Policy agent** (on managed devices)
- **Reboot agent** (on managed devices)
- **Scheduling agent** (on managed devices)
- **Security agent** (on managed devices)
- **Selection agent** (on managed devices)
- **Upload agent** (on managed devices)

For detailed information about using these agents, see the *RMS System Reference*. This document concentrates only on the preferences you can set for each tool on managed devices. The command line argument syntax is described for each individual preference in the reference section of this manual.
Internal-only, deprecated, and obsolete preferences are not listed in this section.

Adoption Agent Preferences Set on the Command Line

You may set the following preferences using the adoption agent command line:

- ***AllowRebootIfServer*** - determines whether or not a computer being brought under management should be rebooted if it is a server
- ***ForceReboot*** - determines whether RayManageSofti forces a reboot if the package being installed requires it. A forced reboot suppresses any user interaction required to close other applications that may be running
- ***ForceRebootIfLocked*** - determines whether RayManageSofti performs a forced reboot if the user's desktop is locked. A forced reboot suppresses any user interaction required to close other applications that may be running
- ***RebootIfRequired*** - controls whether RayManageSofti reboots the managed device if the package being installed requires it and the user's desktop is not locked
- ***UserInteractionLevel*** (installation agent), ***UserInteractionLevel*** (adoption agent), ***UserInteractionLevel*** (inventory agent) - determines the level of user interaction

Installation Agent Preferences Set on the Command Line

You may set the following preferences using the installation agent command line:

- ***AddRemove*** - determines whether installed packages create an entry in Add/ Remove programs
- ***AllowByteLevel*** - determines whether byte-level differencing is operational on the managed device

- ***AllowedPkgSubtypes*** - specifies the package types allowed for policy installs
- ***AllowPeerToPeer*** - specifies whether or not managed devices can obtain downloaded files from other managed devices on the same LAN
- ***AllowRebootIfLocked*** - controls whether RayManageSofti reboots the managed device if the package being installed requires it, even if the machine is locked
- ***AllowRebootIfServer*** - controls whether RayManageSofti reboots the managed device if it is a server (used by the adoption agent, but not the installation agent)
- ***AllowTimeoutIfLocked*** - controls whether the time interval for prompting the end-user commences immediately if the desktop is locked, or commences when the desktop is unlocked
- ***AlwaysDisplayReboot*** - controls whether RayManageSofti displays a warning to the user before performing any reboot required by a package installation (overrides UserInteractionLevel)
- ***ApplyPolicy*** - used on managed devices configured for peer-to-peer file sharing, in conjunction with **DownloadPolicy**, to distinguish between the *Apply a Deployment Manager policy* and *Update policy and package definitions in peer cache* events
- ***AskAboutDependencies*** - determines whether RayManageSofti prompts the user before prerequisite packages are installed
- ***AskBeforeInstalling*** - determines whether RayManageSofti prompts the user before installing a package
- ***AutoDetectDC*** - determines how RayManageSofti selects a domain controller for client-side policy merging
- ***AutoPromptOnInstallCompletion*** - when the UserInteractionLevel is set to Auto, determines whether RayManageSofti informs the user when package installation is complete
- ***AutoPromptOnUnInstallCompletion*** - when the UserInteractionLevel is set to Auto, determines whether RayManageSofti informs the user when package un-installation is complete
- ***AutoRedundancy*** - determines handling of redundant files during upgrades or downgrades
- ***BrandARP*** - provides the ability to exclude the name “RayManageSofti” from Add/Remove entries for installed applications
- ***CacheDir*** - the location of the peer cache on managed devices. When peer-to-peer file sharing is enabled, files are downloaded to this location, and shared with peer managed devices from this location
- ***CheckCatalogDigest*** - determines whether RayManageSofti performs a check on file-level MD5 digests during self-healing operations
- ***CheckCertificateRevocation*** - determines whether RayManageSofti checks certificate revocation list when accepting web server signatures from an HTTPS server
- ***CheckFileDigest*** - determines whether RayManageSofti performs a check on file-level MD5 digests during self-healing operations
- ***CheckRegistry*** (or reg on the command line) - determines if RayManageSofti performs self-healing on registry keys and preference files
- ***CmdLineOverrides*** - determines whether options set on the command line override fixed preferences in the registry or network preference file
- ***ConfirmSharedFileRemoval*** - determines whether RayManageSofti displays a dialog when removing a file
- ***ConnectionAttempts*** - the number of times that a "no connection is available" error can be reported while

trying to connect to a particular distribution location as a file share

- ***DisplayAllAuthcode*** - determines the subsequent behavior after RayManageSofti encounters an invalid signature when performing an Authenticode check
- ***EnablePolicyFailOver*** - specifies whether a server-side policy file should be applied if a client-side policy file is unavailable
- ***ForceReboot*** - determines whether RayManageSofti forces a reboot if the package being installed requires it. A forced reboot suppresses any user interaction required to close other applications that may be running
- ***ForceRebootIfLocked*** - determines whether RayManageSofti performs a forced reboot if the machine is locked. A forced reboot suppresses any user interaction required to close other applications that may be running
- ***ForceSharedFileRemove*** - allows deletion of redundant files in the Windows system folder
- ***ForceValidSignature*** - determines whether RayManageSofti prompts the user before installing a package when Authenticode signatures are valid
- ***GlobalConfigSource*** - identifies a URL that contains installation preferences
- ***http_proxy*** - proxy settings for the RayManageSofti installation agent
- ***IgnoreConnectionWindows*** - specifies whether to use or ignore the download time periods specified by ParentConnectionWindows and PeerConnectionWindows
- ***InstallationStatusRefreshPeriod*** - specifies how frequently (in seconds) RayManageSofti should recreate installation events for packages that are installed, or flagged as not required
- ***InstallerARPModify*** - determines whether external installer package details can be modified in Add/Remove programs
- ***InstallerARPRemove*** - determines whether external installer packages can be uninstalled using Add/Remove programs
- ***LogFileOld*** (installation agent) - name of file to store additional logging information
- ***LogFileSize*** (installation agent) - maximum log file size
- ***LogInstallCheck*** - specifies whether RayManageSofti should recreate installation events while checking packages for installation or upgrade
- ***LogInstallFail*** - specifies whether RayManageSofti should log failed installation attempts
- ***LogInstallPass*** - specifies whether RayManageSofti should log successful installation events
- ***LogLevel*** (installation agent) - level of logging returned by the smart-agent
- ***LogNotRequiredCheck*** - specifies whether RayManageSofti should log applications that are not required on the managed device while checking packages for installation or upgrade
- ***LogUninstallFail*** - specifies whether RayManageSofti should log failed uninstallation attempts
- ***LogUninstallPass*** - specifies whether RayManageSofti should log successful uninstallation events
- ***LowProfile*** (installation agent, inventory agent) - processing priority used for RayManageSofti processes
- ***MinimumDCSpeed*** - determines the minimum speed between the managed device and domain controller that is required to apply client-side policy

- ***MsiBaseURL*** - the web location from which applications can be retrieved
- ***MsiReinstallFeatures*** - MSI components to be installed (equivalent to MSI property `REINSTALL`)
- ***MsiReinstallModeLevel*** - identifies what will be reinstalled (equivalent to MSI property `REINSTALLMODE`)
- ***MsiRepair*** - determines if MSI repair operations are performed at the same time as RayManageSofti self-healing operations
- ***MsiRepairLevel*** - identifies what will be repaired (equivalent to MSI property `REINSTALLMODE`)
- ***MsiSourceLocation*** - determines whether Windows Installer packages are installed from the managed device's local Windows Installer cache, or a distribution location
- ***MsiUILevel*** - determines user interaction level for MSI (equivalent to option /q in `msiexec.exe` command line)
- ***MsiUninstallArgs*** - arguments to include in the MSI command line for uninstall operations
- ***NetworkHighSpeed*** (installation agent) - lowest network speed to consider to be high-speed network connection
- ***NetworkHighSpeed*** (installation agent) - maximum bandwidth for high-speed connections
- ***NetworkHighUsageLowerLimit*** - minimum NetworkHighSpeed (installation agent) value that can be set for a managed device
- ***NetworkHighUsageUpperLimit*** - maximum NetworkHighSpeed (installation agent) value that can be set for a managed device
- ***NetworkLowUsage*** - maximum bandwidth for low-speed connections
- **NetworkLowUsageLowerLimit** - minimum NetworkLowUsage value that can be set for a managed device vice
- ***NetworkLowUsageUpperLimit*** - maximum NetworkLowUsage value that can be set for a managed device
- ***NetworkMaxByteLevelSpeed*** - speed at which byte-level differencing is disabled (no significant advantages in performing byte-level differencing for very high speed connections)
- ***NetworkMaxRate*** (installation agent) - rate at which the managed device accesses data over the network
- ***NetworkMinSpeed*** (installation agent) - minimum network speed at which RayManageSofti will install or update a package
- ***NetworkRetries*** - number of times failed network operations are retried before an alternative distribution location is attempted
- ***NetworkSense*** (installation agent) - determines whether network checks are bypassed
- ***NetworkTimeout*** (installation agent) - number of seconds of inactivity before a network operation will time out
- ***NoStage*** - determines whether files are downloaded directly to their destination folder or a staging area
- ***PlatformSpecificPackages*** - determines whether information should be included in software inventories about non-Windows, platform-specific packages
- ***PolicyPackageRefreshPeriod*** - the number of seconds after successfully downloading package (`.osd`) files during which the files should not be downloaded again

- ***PolicyRefreshPeriod*** - the number of seconds after successfully downloading policy (`.npl`) files during which the files should not be downloaded again
- ***PolicyServerPriority*** - specifies the priority to apply to the distribution location identified by the PolicyServerURL (internal-only) preference
- ***PostponeByDefault*** - used to postpone the installation of mandatory packages by default (if possible)
- ***PostponementQueryBefore*** - determines whether any alert about postponing an installation is shown before download, before installation, or both
- ***PostponeUserInteractionLevel*** - controls whether end-users on managed devices are interactively asked if they want to postpone installations of mandatory packages that are appropriately configured in policy
- ***PromptOnCOMRegFailures*** - determines whether RayManageSofti prompts the user if it fails to register a COM server
- ***PromptOnInstallCompletion*** - when the UserInteractionLevel (installation agent) is set to `Full`, determines whether RayManageSofti informs the user when package installation is complete
- ***PromptOnUnInstallCompletion*** - when the UserInteractionLevel (installation agent) is set to `Full`, determines whether RayManageSofti informs the user when package un-installation is complete
- ***PropagatePkgChanged*** - reinstalls the base package if the prerequisite package has changed for Third party installer packages
- ***PublicAppAccess*** - determines RayManageSofti access to Common folders on Windows XP
- ***QuietUntilUpdate*** - controls whether the RayManageSofti user interface is hidden if no user interaction is necessary
- ***RebootCmdLine*** - used on the managed device to reboot from the command line
- ***RebootIfRequired*** - sets the default response to dialogs that prompt the user to allow a reboot
- ***RebootContinueAfterCmdFailure*** - controls whether or not to proceed with the reboot if the RebootPreCommand returned a non-zero exit code
- ***RebootPostCommand*** - a command to be executed after the managed device is rebooted
- ***RebootPreCommand*** - a command to be executed before the managed device is rebooted
- ***RebootPromptCycles*** - the number of times the end-user can postpone the managed device reboot. Prompts occur at intervals specified by RebootPromptWait
- ***RebootPromptUnlimited*** - keep prompting the end-user, at intervals specified by RebootPromptWait, until the managed device reboots (equivalent to RebootPromptCycles=-1)
- ***RebootPromptWait*** - the number of seconds to wait after the end-user dismisses the reboot dialog before displaying it again
- ***ReInstallRequiresVersionChange*** - determines when packages will be upgraded, downgraded, or reinstalled, based on the type of changes made to the package
- ***RenotifyTimeout*** - the number of seconds that the installation agent waits before once again showing a user any hidden dialogs that have not yet timed out
- ***SaveAllUserSymbols*** - determines whether RayManageSofti retains installation preferences set by a top-level or pre-requisite package. Also see *Persistent managed device preferences*

- ***SecurityAnalysisFile*** - naming convention for the saved security analysis file
- ***SelfHeal*** - determines whether self-healing should take place for an individual package
- ***ShowIcon*** (installation agent) - controls whether RayManageSofti displays an icon in the system tray. This preference can be set separately for the installation agent and inventory agent
- ***StageInactivePackages*** - determines whether the managed device can download files for packages within policies that have future activation times
- ***StrictInstall*** - if `True`, the policy agent returns a non-zero exit code if any package fails to install
- ***SupplyWorstCaseReturnValue*** - determines whether RayManageSofti returns an error only when an installation agent operation fails, or also when upgrades or self-heal operations fail
- ***UITimeoutWait*** - the number of seconds that installation agent dialogs display before timing out
- ***UserInteractionLevel*** (installation agent) - determines the level of user interaction (previously called UILevel). This preference can be set separately for the installation agent and inventory agent. In relation to reboot options, only the installation agent setting is applicable
- ***UseTrustDatabase*** - determines whether the installation agent takes account of the distribution location from which files are collected when determining whether a package may be installed
- ***VerifyCatalogSigned*** - determines whether Authenticode digital signatures are checked in the RayManageSofti catalog (`.ndc`) file before packages are installed
- ***VerifyFilesSigned*** - determines whether executable files downloaded by RayManageSofti are checked for a valid Authenticode digital signature before being installed
- ***VirusScan*** and ***VirusScanCommand*** - determines whether and how files downloaded by RayManageSofti are scanned for viruses before installation

Inventory Agent Preferences Set on the Command Line

You may set the following preferences using the inventory agent command line:

- ***CheckCertificateRevocation*** - determines whether RayManageSofti checks certificate revocation list when accepting web server signatures from an HTTPS server
- ***Compress*** (inventory agent) - determines whether inventory files are compressed for upload
- ***Difference*** - determines whether differential inventories are performed on the managed device
- ***ExcludeDirectory*** - folders to exclude from inventory
- ***ExcludeExtension*** - file extensions to exclude from inventory
- ***ExcludeFile*** - files to exclude from inventory
- ***ExcludeMD5*** - files matching this MD5 checksum are excluded from inventory
- ***GenerateMD5*** - generate an MD5 for every file being tracked
- ***GenerationMax*** - number of differential inventories performed between full inventories

- ***Hardware*** - determines whether to track hardware (in the machine context)
- ***IncludeDirectory*** - folders to include in inventory
- ***IncludeExtension*** - file extensions to include in inventory
- ***IncludeFile*** - files to include in inventory
- ***IncludeMachineInventory*** - specifies whether or not to conduct a computer inventory of hardware and all user packages
- ***IncludeMD5*** - files matching this MD5 checksum are included in inventory
- ***IncludeRegistryKey*** - registry keys or values to include in inventory
- ***IncludePermissionsMask*** - octal mask for file permissions on non-Windows clients that limits which files are scanned for inventory
- ***IncludeUserInventory*** - specifies whether or not to conduct a user inventory
- ***IncrementalDiff*** - when differential inventories are performed, determines what differences RayManageSofti will collect
- ***InventoryFile*** - file name of a local copy of the inventory file
- ***InventoryScriptsDir*** - if RunInventoryScripts is `True`, this specifies the location of scripts to be run after inventory scanning completes
- ***MachineInventoryDirectory*** - location for computer inventories
- ***ManageSoftPackages*** - determines the installed software packages
- ***MinInventoryInterval*** - specifies the minimum interval (in hours) between collections of inventory
- ***MSI*** - determines whether Microsoft Installer (MSI) packages are included in inventory
- ***NetworkSense*** (inventory agent) - determines whether network checks are bypassed
- ***RunInventoryScripts*** - specifies whether or not to run inventory scripts after gathering inventory data
- ***Security*** - determines whether or not to perform security compliance checking
- ***ShowIcon*** (inventory agent) - controls whether RayManageSofti displays an icon in the system tray. This preference can be set separately for the installation agent and inventory agent
- ***SMBIOSCmdLine*** - specifies a command line for non-WMI hardware inventory collection
- ***UserHardware*** - determines whether to track hardware (in the user context)
- ***UserInteractionLevel*** (inventory agent) - determines the level of user interaction (previously called **UILevel**)
- ***UserInventoryDirectory*** - location for user inventories on the managed device
- ***VersionInfo*** - determines whether file version header information is included in inventory

Peer Download Agent Preferences Set on the Command Line

You may set the following preferences using the peer download agent command line *only when using it in debugging mode*(`-debug`):

- ***CacheDir*** - the location of the peer cache
- ***CatalogName*** - the name of the peer downloads file (located in the parent of the CacheDir directory) listing files required by the peer cache
- ***CheckpointSeconds*** - the frequency (in seconds) with which the peer downloads file is written to disk
- ***DiskAveragingTime*** - the time period to use to smooth the disk I/O traffic estimate. Used in conjunction with DiskMaxRate
- ***DiskMaxRate*** - the maximum allowable averaged rate (in bytes per second) of all reads from and writes to disks caused by peer-to-peer file sharing (used in conjunction with DiskAveragingTime)
- ***GCDiskSlice*** - the maximum percentage of DiskMaxRate that can be used for peer cache cleanup operations
- ***GCMaxInterval*** - the maximum number of minutes the peer download agent should pause between examining files in the peer cache as part of cleanup operations
- ***GCMinInterval*** - the minimum number of minutes the peer download agent should pause between examining files in the peer cache as part of cleanup operations
- ***GCPeriod*** - the time period (in hours) over which peer cache cleanup operations are conducted
- ***MinFreeDisk*** - the amount of disk space (in MB) that must be free for the peer download agent to download files to the peer cache
- ***PeerAveragingTime*** - the time period to use to smooth the peer file sharing traffic estimate. Used in conjunction with PeerMaxRate
- ***PeerListenQueue*** - the number of connection requests to queue before refusing additional connections
- ***PeerMaxRate*** - the maximum allowable averaged rate (in bytes per second) of file transfers between peer managed devices (used in conjunction with PeerAveragingTime)
- ***PeerPullPort*** - the TCP port on which file transfers from managed devices can be received
- ***PeerPush*** - specifies whether peer managed devices can immediately transfer (push) requested files (`True`), or whether they must wait for a request
- ***PeerSearchDuration*** - the time period during which to search for a file from peer managed devices
- ***PeerSearchPort*** - the UDP port on which file transfer requests are broadcast
- ***PeerTransferLimit*** - the number of simultaneous peer-to-peer search and file transfer operations allowed
- ***PipeName*** (peer download agent) - the name of the pipe used to communicate with the peer download agent
- ***SearchFrequency*** - the number of seconds between issuing peer-to-peer file sharing requests
- ***SearchMaxOffer*** - the maximum number of offers of a file from peer managed devices to accept before

terminating the search

- ***SearchMinimum*** - the minimum number of search requests to issue for a file being sought from peer managed devices
- ***SearchRetry*** - the number of seconds to wait after a failed file search attempt before reissuing the request
- ***UnusedFileUptime*** - files in the peer cache that have not been accessed within this number of seconds are removed during cleanup operations
- ***UnusedFilePersistence*** - specifies how long (in minutes) to wait after receiving the first request from the installation agent before starting to look for and delete unused files
- ***WANAveragingTime*** - the time period to use to smooth the estimate of traffic transferred between the managed device and a distribution server across a wide area network. Used in conjunction with WANMaxRate
- ***WANMaxRate*** - the maximum allowable averaged rate (in bytes per second) of file transfers between managed devices and a distribution server across a wide area network (used in conjunction with WANAveragingTime)
- ***WANProgressInterval*** - the frequency (in seconds) with which to update peer managed devices with progress about file downloads from a distribution server
- ***WANRetries*** - the number of times a failed WAN download is retried immediately from each distribution server at each WAN retry interval
- ***WANRetryDuration*** - the length of time (in minutes) to continue to allow a file to be retried since it was last requested by the installation agent
- ***WANRetryInterval*** - the length of time (in seconds) after a failed WAN download to retry the download
- ***WANSearchCurrency*** - how recently (in seconds) a peer search for a file must have occurred, before the file should be downloaded from the closest distribution server
- ***WANTimeout*** - the time (in seconds) after which to abort stalled transfers of files from distribution servers
- ***WANTransferLimit*** - the maximum number of managed devices that may simultaneously download from a distribution location

Policy Agent Preferences Set on the Command Line

You may set the following preferences on the policy agent command line. You may also set any **installation agent preferences** (see *Installation agent preferences set on the command line*) on the policy agent command line:

- ***BootstrappedPolicy*** - the location of the policy to be applied to managed devices that do not use policy attached to Active Directory domains
- ***CheckCertificateRevocation*** - determines whether RayManageSofti checks certificate revocation list when accepting web server signatures from an HTTPS server

Reboot Agent Preferences Set on the Command Line

While the reboot agent is typically called by the installation agent, if you choose to run it independently you may set the following preferences on the scheduling agent command line:

- ***RebootContinueAfterCmdFailure*** - specifies whether or not to continue with rebooting a managed device if execution of a pre-reboot command returns a non-zero exit code
- ***RebootPostCommand*** - a command to run immediately after a managed device is rebooted
- ***RebootPreCommand*** - a command to run immediately before a managed device is rebooted
- ***RebootPromptCycles*** - the number of times an end-user can postpone a managed device reboot
- ***RebootPromptWait*** - the time interval, in seconds, to wait before re-displaying the dialog that prompts the end-user to reboot

Scheduling Agent Preferences Set on the Command Line

You may set the following preferences on the scheduling agent command line:

- ***ApplyPolicyIfLoggedOn*** - specifies whether or not computer policy is applied at the scheduled time if a user is logged on
- ***CheckCertificateRevocation*** - determines whether RayManageSofti checks certificate revocation list when accepting web server signatures from an HTTPS server
- ***DisablePeriod*** - determines the number of seconds for which RayManageSofti user schedules remain disabled when the end-user disables them in the schedule agent on the managed device
- ***EventNetType*** - determines the type of network connections that are required to start events with an **OnConnect** trigger
- ***NativeScheduler*** - determines whether the Microsoft Task Scheduler or RayManageSofti Task Scheduler is in use
- ***ndsensNetType*** - determines what type of network connections are monitored
- ***RetryPolicy*** - determines whether RayManageSofti will attempt to retrieve RayManageSofti policy when the managed device boots, if no computer schedule exists on the managed device
- ***RetryPolicyCommand*** - the command used to retrieve policy if RetryPolicy is set to `True`

Security Agent Preferences Set on the Command Line

You may set the following preferences on the security agent command line:

- ***BlockedApplicationIncludeLocalAdmin*** - if `True`, users with administrator privileges are also prevented from accessing blocked applications on managed devices; if `False`, local administrator users can access blocked applications

- ***BlockedApplicationMessage*** - the message to display to end-users when an application is prevented from running
- ***BlockedApplicationNotify*** - whether or not end-users are to be notified when an application is prevented from running
- ***BlockedApplications*** - applications that are to be prevented from running
- ***LocalDeviceAutoRebootMessage*** - message displayed to the end-user before the managed device is automatically rebooted
- ***LocalDeviceAutorebootNotLoggedIn*** - whether to reboot automatically if LocalDeviceAutoRebootMessage is set and no end-user is logged on
- ***LocalDeviceAutoRebootPeriod*** - number of minutes before a reboot takes place if ForceReboot is not set and the user does not respond
- ***LocalDeviceAutorebootWithNoWait*** - whether to reboot automatically if ForceReboot is not set and the user is not logged on
- ***LocalDeviceForceReboot*** - whether to force the reboot or prompt the user to reboot
- ***NetworkSense*** (security agent) - determines whether network checks are bypassed
- ***PipeName*** (security agent) - the name of the pipe used to communicate with the security agent
- ***RunVulnerabilityScripts*** - specifies whether or not to run vulnerability scripts after vulnerability scanning
- ***SecurityAnalysisFile*** - naming convention for the saved security analysis file
- ***SecurityLogInterval*** - the time interval at which security logs are generated
- ***SessionListInterval*** - the time interval at which user session lists are generated
- ***TechniciansGroup*** - the groups whose members can override security policies on managed devices
- ***VulnerabilityScripts*** - if RunVulnerabilityScripts is `True`, this specifies the scripts to be run after vulnerability scanning completes
- ***VulnerabilityScriptsDir*** - if RunVulnerabilityScripts is `True`, this specifies the location of scripts to be run after vulnerability scanning completes

Selection Agent Preferences Set on the Command Line

You may set the following preferences on the selection agent command line:

- ***ApplicationInstallCommand*** - a template command line to be used to install an application package through the RayManageSofti package selection agent
- ***ApplicationUninstallCommand*** - a template command line to be used to uninstall an application package through the RayManageSofti package selection agent
- ***ApplicationVerifyCommand*** - a template command line to be used to verify/ repair an application package through the RayManageSofti package selection agent
- ***CheckCertificateRevocation*** - determines whether RayManageSofti checks certificate revocation list when

accepting web server signatures from an HTTPS server

- ***ConfigFile*** - name of the system copy of the configuration file used by the selection agent
- ***ConfigFileDefault*** - name of the default configuration file to use when other settings fail
- ***Locale*** - locale to use for selection agent localization
- ***LocaleDefault*** - locale to use in the absence of other settings
- ***ReInstallRequiresVersionChange*** - number of minutes between automatic refresh of data displayed by the selection agent

Upload Agent Preferences Set on the Command Line

You may set the following preferences on the upload agent command line:

- ***CheckCertificateRevocation*** - determines whether RayManageSofti checks certificate revocation list when accepting web server signatures from an HTTPS server
- ***NetworkSense*** (upload agent) - determines whether network checks are bypassed
- ***SourceFile*** - file or files to be uploaded through the upload agent
- ***SourceRemove*** - determines whether the upload agent removes uploaded files from the source location after a successful upload
- ***UploadType*** - determines whether the upload agent uploads computer generated file or user generated files

Preferences in the Global Configuration File

You can use the RayManageSofti global configuration file to set preferences in a central location.

What is the Global Configuration File?

The global configuration file can be any `.ini` file of your choosing and can be set up to apply to RayManageSofti on all managed devices.

Its location is specified by the preference *GlobalConfigSource* read from:

`[Registry]\ManageSoft\Launcher\CurrentVersion.`

Where Can this File Be Kept?

You can store the configuration file anywhere. It can be located on your local network or on your corporate intranet.

The value can be a URL (`http://...`) or a UNC path such as `\\server\share\`.

Be aware:

You must ensure that:

- All URL paths use forward slashes //
  Example: http://myserver/somepath/mgsconfig.ini
- All UNC paths use backward slashes \\
  Example: \\my server\somepath\mgsconfig.ini

Preferences set in the `config` file cannot be:

- Set as project variables because they are processed when the installation agent is first initialized, before any package files are read.
- Set on the command line as they are intended to be under administrative, not user, control.

Why Would You Set Preferences in this File?

Setting preferences directly in the configuration file allows you to specify the settings under which the RayManageSofti installation agent runs, in a central location.

Typically, RayManageSofti administrators make use of the more intuitive Active Directory and Group Policy to achieve the same result (see *MachineAlternateRegistryHive* of the *Alphabetical listing of preferences* chapter).

Raynet recommends using this GlobalConfigSource functionality only on UNIX and Macintosh computers that do not interact with Active Directory environments.

Depending on the number of managed devices referencing this file and how often each one does so, the load on the server may be an issue.

Can You Set Any Preference in the Configuration File?

You can set any preference that are normally read from:

`[Registry]\ManageSoft\Launcher\CurrentVersion.`

How Do You Set a Preference in this File?

The file is in the standard Windows `.ini` file format and can be edited with any text editor. For example, WordPad.

Any preferences set in this file are set in the format: name=value

Each name/value is set inside a section named (in square brackets) for the agent being configured, such as [`Launcher`].

Example

```
[Launcher]
ForceValidSignature=True
ForceValidSignatureFixed=True
VerifyCatalogSigned=True
VerifyCatalogSignedFixed=True
```

RayManageSofti refers to a number of locations when evaluating the precedence of preference settings on a managed device. The order of precedence is as follows (highest to lowest):

1. Command line arguments.
2. Any preferences read from *UserAlternateRegistryHive*.
3. User preferences, taken from:
   `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\...`
4. Any preferences read from *MachineAlternateRegistryHive*.
5. Computer preferences, taken from:
   `[Registry]\ManageSoft\...`
6. Network preferences, taken from the file on the network specified by URL or UNC in the registry setting *GlobalConfigSource*.
7. RayManageSofti factory defaults.

Preferences You Can Set in the Configuration File

You may set the following preferences using the global configuration file:

- ***AddRemove*** - determines whether installed packages create an entry in Add/Remove programs.
- ***AllowByteLevel*** - determines whether byte-level differencing is operational on the managed device.
- ***AllowPeerToPeer*** – specifies whether or not managed devices can obtain downloaded files from other managed devices on the same LAN.
- ***AllowRebootIfLocked*** - controls whether RayManageSofti reboots the managed device if the package being installed requires it, even if the machine is locked.
- ***AllowRebootIfServer*** - controls whether RayManageSofti reboots the managed device if it is a server (used by the adoption agent, but not the installation agent).
- ***AllowTimeoutIfLocked*** - controls whether the time interval for prompting the end-user commences immediately if the desktop is locked, or commences when the desktop is unlocked
- ***AlwaysDisplayReboot*** - controls whether RayManageSofti displays a warning to the user before performing any reboot required by a package installation (overrides UserInteractionLevel).
- ***AskAboutDependencies*** - determines whether RayManageSofti prompts the user before prerequisite packages are installed.
- ***AskBeforeInstalling*** - determines whether RayManageSofti prompts the user before installing a package.
- ***AutoDetectDC*** - determines how RayManageSofti selects a domain controller for client-side policy merging.

- ***AutoPromptOnInstallCompletion*** - when the UserInteractionLevel is set to Auto, determines whether RayManageSofti informs the user when package installation is complete.
- ***AutoPromptOnUnInstallCompletion*** - when the UserInteractionLevel is set to Auto, determines whether RayManageSofti informs the user when package un-installation is complete.
- ***AutoRedundancy*** - determines handling of redundant files during upgrades or downgrades.
- ***BrandARP*** - provides the ability to exclude the name “RayManageSofti” from Add/Remove entries for installed applications.
- ***CacheDir*** - the location of the peer cache on managed devices. When peer-to-peer file sharing is enabled, files are downloaded to this location, and shared with peer managed devices from this location.
- ***CheckCatalogDigest*** - determines whether RayManageSofti performs a check on file-level MD5 digests during self-healing operations.
- ***CheckCertificateRevocation*** - determines whether RayManageSofti checks certificate revocation list when accepting web server signatures from an HTTPS server.
- ***CheckFileDigest*** - determines whether RayManageSofti performs a check on file-level MD5 digests during self-healing operations.
- ***CheckRegistry*** (or reg on the command line) - determines if RayManageSofti performs self-healing on registry keys and preference files.
- ***CmdLineOverrides*** - determines whether options set on the command line override fixed preferences in the registry or network preference file.
- ***ConfirmSharedFileRemoval*** - determines whether RayManageSofti displays a dialog when removing a file.
- ***ConnectionAttempts*** - the number of times that a "no connection is available" error can be reported while trying to connect to a particular distribution location as a file share.
- ***DisplayAllAuthcode*** - determines the subsequent behavior after RayManageSofti encounters an invalid signature when performing an Authenticode check.
- ***ForceReboot*** - determines whether RayManageSofti forces a reboot if the package being installed requires it. A forced reboot suppresses any user interaction required to close other applications that may be running.
- ***ForceRebootIfLocked*** - determines whether RayManageSofti performs a forced reboot if the machine is locked. A forced reboot suppresses any user interaction required to close other applications that may be running.
- ***ForceSharedFileRemove*** - allows deletion of redundant files in the Windows system folder.
- ***ForceValidSignature*** - determines whether RayManageSofti prompts the user before installing a package when Authenticode signatures are valid.
- ***GlobalConfigSource*** - identifies a URL that contains installation preferences.
- ***http_proxy*** - proxy settings for the RayManageSofti installation agent.
- ***InstallationStatusRefreshPeriod*** - specifies how frequently (in seconds) RayManageSofti should recreate installation events for packages that are installed, or flagged as not required.
- ***InstallerARPModify*** - determines whether external installer package details can be modified in Add/Remove programs.
- ***InstallerARPRemove*** - determines whether external installer packages can be uninstalled using Add/Remove programs.
- ***LogFileOld*** (installation agent) - name of file to store additional logging information.
- ***LogFileSize*** (installation agent) - maximum log file size.
- ***LogInstallCheck*** - specifies whether RayManageSofti should recreate installation events while checking packages for installation or upgrade.
- ***LogInstallFail*** - specifies whether RayManageSofti should log failed installation attempts.

- ***LogInstallPass*** - specifies whether RayManageSofti should log successful installation events.
- ***LogLevel*** (installation agent) - level of logging returned by the smart-agent.
- ***LogNotRequiredCheck*** - specifies whether RayManageSofti should log applications that are not required on the managed device while checking packages for installation or upgrade.
- ***LogUninstallFail*** - specifies whether RayManageSofti should log failed uninstallation attempts.
- ***LogUninstallPass*** - specifies whether RayManageSofti should log successful uninstallation events.
- ***LowProfile*** (installation agent, inventory agent) - processing priority used for RayManageSofti processes.
- ***MinimumDCSpeed*** - determines the minimum speed between the managed device and domain controller that is required to apply client-side policy.
- ***MsiBaseURL*** - the web location from which applications can be retrieved.
- ***MsiReinstallFeatures*** - MSI components to be installed (equivalent to MSI property `REINSTALL`).
- ***MsiReinstallModeLevel*** - identifies what will be reinstalled (equivalent to MSI property `REINSTALLMODE`).
- ***MsiRepair*** - determines if MSI repair operations are performed at the same time as RayManageSofti self-healing operations.
- ***MsiRepairLevel*** - identifies what will be repaired (equivalent to MSI property `REINSTALLMODE`).
- ***MsiSourceLocation*** - determines whether Windows Installer packages are installed from the managed device's local Windows Installer cache, or a distribution location.
- ***MsiUILevel*** - determines user interaction level for MSI (equivalent to option /q in msiexec.exe command line).
- ***MsiUninstallArgs*** - arguments to include in the MSI command line for uninstall operations.
- ***NetworkHighSpeed*** (installation agent) - lowest network speed to consider to be high-speed network connection.
- ***NetworkHighSpeed*** (installation agent) - maximum bandwidth for high-speed connections.
- ***NetworkHighUsageLowerLimit*** - minimum NetworkHighSpeed (installation agent) value that can be set for a managed device.
- ***NetworkHighUsageUpperLimit*** - maximum NetworkHighSpeed (installation agent) value that can be set for a managed device.
- ***NetworkLowUsage*** - maximum bandwidth for low-speed connections.
- ***NetworkLowUsageLowerLimit*** - minimum NetworkLowUsage value that can be set for a managed device vice.
- ***NetworkLowUsageUpperLimit*** - maximum NetworkLowUsage value that can be set for a managed device.
- ***NetworkMaxByteLevelSpeed*** - speed at which byte-level differencing is disabled (no significant advantages in performing byte-level differencing for very high speed connections).
- ***NetworkMaxRate*** (installation agent) - rate at which the managed device accesses data over the network.
- ***NetworkMinSpeed*** (installation agent) - minimum network speed at which RayManageSofti will install or update a package.
- ***NetworkRetries*** - number of times failed network operations are retried before an alternative distribution location is attempted.
- ***NetworkSense*** (installation agent) - determines whether network checks are bypassed.
- ***NetworkSense*** (inventory agent) - determines whether network checks are bypassed.
- ***NetworkSense*** (security agent) - determines whether network checks are bypassed.
- ***NetworkSense*** (upload agent) - determines whether network checks are bypassed.
- ***NetworkTimeout*** (installation agent) - number of seconds of inactivity before a network operation will time out.

- ***NoStage*** - determines whether files are downloaded directly to their destination folder or a staging area.
- ***PolicyPackageRefreshPeriod*** - the number of seconds after successfully downloading package (`.osd`) files during which the files should not be downloaded again
- ***PolicyRefreshPeriod*** - the number of seconds after successfully downloading policy (`.npl`) files during which the files should not be downloaded again
- ***PolicyServerPriority*** - specifies the priority to apply to the distribution location identified by the PolicyServerURL (internal-only) preference.
- ***PostponeByDefault*** - used to postpone the installation of mandatory packages by default (if possible).
- ***PostponeUserInteractionLevel*** - controls whether end-users on managed devices are interactively asked if they want to postpone installations of mandatory packages that are appropriately configured in policy.
- ***PromptOnCOMRegFailures*** - determines whether RayManageSofti prompts the user if it fails to register a COM server.
- ***PromptOnInstallCompletion*** - when the UserInteractionLevel (installation agent) is set to `Full`, determines whether RayManageSofti informs the user when package installation is complete.
- ***PromptOnUnInstallCompletion*** - when the UserInteractionLevel (installation agent) is set to `Full`, determines whether RayManageSofti informs the user when package un-installation is complete.
- ***PropagatePkgChanged*** - reinstalls the base package if the prerequisite package has changed for Third party installer packages.
- ***PublicAppAccess*** - determines RayManageSofti access to Common folders on Windows NT 4.0/2000/XP.
- ***QuietUntilUpdate*** - controls whether the RayManageSofti user interface is hidden if no user interaction is necessary.
- ***RebootCmdLine*** - used on the managed device to reboot from the command line.
- ***RebootContinueAfterCmdFailure*** - controls whether or not to proceed with the reboot if the RebootPreCommand returned a non-zero exit code
- ***RebootIfRequired*** - sets the default response to dialogs that prompt the user to allow a reboot.
- ***RebootPostCommand*** - a command to be executed after the managed device is rebooted.
- ***RebootPreCommand*** - a command to be executed before the managed device is rebooted.
- ***RebootPromptCycles*** - the number of times the end-user can postpone the managed device reboot. Prompts occur at intervals specified by RebootPromptWait.
- ***RebootPromptUnlimited*** - keep prompting the end-user, at intervals specified by RebootPromptWait, until the managed device reboots (equivalent to RebootPromptCycles=-1)
- ***RebootPromptWait*** - the number of seconds to wait after the end-user dismisses the reboot dialog before displaying it again.
- ***SaveAllUserSymbols*** - determines whether RayManageSofti retains installation preferences set by a top-level or pre-requisite package. Also see *Persistent managed device preferences*.
- ***SecurityAnalysisFile*** - naming convention for the saved security analysis file.
- ***SelfHeal*** - determines whether self-healing should take place for an individual package.
- ***ShowIcon*** (installation agent) - controls whether RayManageSofti displays an icon in the system tray. This preference can be set separately for the installation agent and inventory agent.
- ***StageInactivePackages*** - determines whether the managed device can download files for packages within policies that have future activation times.
- ***StrictInstall*** - if True, the policy agent returns a non-zero exit code if any package fails to install.
- ***SupplyWorstCaseReturnValue*** - determines whether RayManageSofti returns an error only when an installation agent operation fails, or also when upgrades or self-heal operations fail.
- ***UITimeoutWait*** - the number of seconds that installation agent dialogs display before timing out.

- ***UserInteractionLevel*** (installation agent) - determines the level of user interaction (previously called UILevel). This preference can be set separately for the installation agent and inventory agent. In relation to reboot options, only the installation agent setting is applicable.
- ***UseTrustDatabase*** - determines whether the installation agent takes account of the distribution location from which files are collected when determining whether a package may be installed.
- ***VerifyCatalogSigned*** - determines whether Authenticode digital signatures are checked in the RayManageSofti catalog (`.ndc`) file before packages are installed.
- ***VerifyFilesSigned*** - determines whether executable files downloaded by RayManageSofti are checked for a valid Authenticode digital signature before being installed.
- ***VirusScan*** and ***VirusScanCommand*** - determines whether and how files downloaded by RayManageSofti are scanned for viruses before installation.

Preference Listing By Behavior

Types of Behavior

The following sections describe the types of behavior you can control on managed devices using preferences. Preferences are listed in the following groups:

- *Add/Remove program options*
- *Application usage options*
- *Bandwidth optimization options*
- *Byte-level differencing options*
- *Computer and user information*
- *CPU options*
- *Download options*
- *File handling options*
- *Inventory options*
- *Logging options*
- *RayManageSofti folder locations*
- *MSI package options*
- *Network speed and connection options*
- *Package-level filtering options*
- *Package Selector options*
- *Peer-to-peer file sharing options*
- *Policy merge options*
- *Preference management options*
- *Prerequisite package options*
- *Reboot options*
- *Remote execution options*
- *Scheduling options*
- *Security options*
- *Security management options*
- *Self-heal options*
- *Trusted location options*
- *Uninstall options*
- *Upgrade/downgrade options*
- *Upload options*
- *User interaction options*
- *Virus scanning options*

- *Windows folder information*

Add / Remove Program Options

The following preferences control how RayManageSofti interacts with the Add/Remove programs control panel applet during package processing:

- ***AddRemove*** - determines whether installed packages create an entry in Add/Remove programs
- ***BrandARP*** - provides the ability to exclude the name "RayManageSofti" from Add/Remove entries for installed applications
- ***InstallerARPModify*** - determines whether external installer package details can be modified in Add/Remove programs
- ***InstallerARPRemove*** - determines whether external installer packages can be uninstalled from Add/Remove programs

Application Usage Options

The following preferences can be used to control behavior of the RayManageSofti application usage agent on managed devices:

- ***Compress*** (application usage agent) - specifies whether application usage data files are compressed before being uploaded to the administration server
- ***Disabled*** (application usage agent) - specifies whether the application usage agent is inactive on this managed device
- ***EnableSessionLogging*** - specifies whether session logging takes place on this managed device
- ***ExcludedMGSs*** - specifies RayManageSofti-managed applications that are not monitored when recording application usage data
- ***ExcludedMSIs*** - specifies applications installed using the native package format (MSI) that are not monitored when recording application usage data (Windows devices only)
- ***ManualMapper*** - specifies manual mappings between executable names and application names and versions
- ***ManualMapperDefaultPriority*** - specifies the default priority for manual mappings between executable names and application names and versions
- ***MinRunTime*** - specifies the minimum time that an application must run for before application usage data will be recorded
- ***PreferenceUpdatePeriod*** - specifies how often the application usage agent refreshes its preferences from the registry
- ***ProcessUpdatePeriod*** - specifies how often the application usage agent checks for new applications running
- ***ProductUpdatePeriod*** - specifies how often the application usage agent refreshes its list of applications to be monitored
- ***SessionBackupPeriod*** - specifies how often the application usage agent caches recorded data locally
- ***StartupDelay*** - the wait time at managed device startup before the application usage agent starts
- ***UploadPeriod*** - specifies how often the usage agent should upload recorded data to the administration server
- ***UsageDirectory*** - specifies the directory into which the application usage agent should store its recorded data
- ***UseAddRemove*** - specifies whether the application usage agent should monitor applications found in Add/Remove programs (Windows devices only)
- ***UseManualMapper*** - specifies whether the application usage agent should monitor applications found in the Manual Mapper registry keys
- ***UseMGS*** - specifies whether the application usage agent should monitor applications found in the

RayManageSofti application cache (Windows devices only)

- ***UseMSI*** - specifies whether the application usage agent should monitor applications found in the native package format (MSI, RPM, or PKG)
- ***UserProcessesOnly*** - specifies whether the application usage agent should only monitor applications that are executed by a logged-in user

Apply Policy Condition

Since the RayManageSofti agent works on a "desired state" principle it is rather simple to affect multiple applications with little effort. However this also brings risks around unwanted actions. An additional layer of intelligence will be implemented allowing the agent to independently determine if an abnormal situation based on rules exists, thus leaving the current state unaltered.

'Apply Policy Condition' feature is implemented that is going to be executed

- after the client side merge is done
- and before the policy is processed.

When the 'Apply Policy Condition' function completes and all conditions have been met the process will continue as normal. Otherwise the "Update Machine Policy" gets terminated and a failure message is sent to the administration server of this current fail and the actual condition that failed, further details in the reporting section.

- ***AppyPolicyConditionPackagePath*** – determines whether the 'Apply Policy Condition' feature is activated. A PackageFullName of a dummy package needs to be provided. Installation logs for this package will be created for reporting
- ***RequiredPoliciesList*** – determines whether the given policies are within the managed device npl file. The policies are specified by their name and need to be separated by a semicolon. In case at least on is missing the npl will not be processed and a failure message send via installation log
- ***RequiredPackagesList*** – determines whether the given packages are within the managed device npl file. The packages are specified by their PackageFullName and need to be separated by a semicolon. In case at least on is missing the npl will not be processed and a failure message send via installation log
- ***MaxAllowedPackageUninstalls*** – determines whether the given amount of to be uninstalled packages is lower than the maximum allowed number. In case the number is equal or higher the npl will not be processed and a failure message send via installation log.

Bandwidth Optimization Options

The following preferences allow you to optimize bandwidth on managed devices:

- ***MinimumDCSpeed*** - minimum network speed to the Domain Controller for RayManageSofti to perform client-side policy merge
- ***NetworkHighSpeed*** (installation agent) and ***NetworkHighSpeed*** (upload agent) - lowest network speed to consider to be high-speed network connection
- ***NetworkHighSpeed*** (installation agent) - maximum percentage of bandwidth used on high-speed connections

- ***NetworkHighUsageLowerLimit*** - minimum NetworkHighUsage value that can be set for a managed device
- ***NetworkHighUsageUpperLimit*** - maximum NetworkHighUsage value that can be set for a managed device
- ***NetworkLowUsage*** - maximum percentage of bandwidth used on low-speed connections
- ***NetworkLowUsageLowerLimit*** - minimum NetworkLowUsage value that can be set for a managed device
- ***NetworkLowUsageUpperLimit*** - maximum NetworkLowUsage value that can be set for a managed device
- ***NetworkMaxRate*** (installation agent) and ***NetworkMaxRate*** (upload agent) - maximum absolute bandwidth used (if other settings do not override)
- ***NetworkMinSpeed*** (installation agent) and ***NetworkMinSpeed*** (upload agent) - minimum speed required before RayManageSofti will access the network
- ***NetworkSense*** (installation agent), ***NetworkSense*** (inventory agent), ***NetworkSense*** (security agent) and ***NetworkSense*** (upload agent) - determines whether network checks are bypassed

Byte-level Differencing Options

The following preferences determine how byte-level differencing is performed on the managed device for packages where byte-level differencing has been set as available:

- ***AllowByteLevel*** - determines whether byte-level differencing is operational on the managed device
- ***NetworkMaxByteLevelSpeed*** - speed at which byte-level differencing is disabled (as there are no significant advantages in performing byte-level differencing for very high speed connections)

Computer and User Information

The following preferences contain information about computer and user information on the managed device. You can reference these preferences in package details and programmed callouts:

- ***ComputerDomain*** - name assigned to the computer domain of the managed device
- ***ComputerDNSName*** - DNS name assigned to the managed device
- ***LogonServer*** - name of the logon server computer to which the managed device normally connects
- ***UserLogonDomain*** - name assigned to the domain of the user

CPU Options

The following preferences influence RayManageSofti CPU usage:

- ***LowProfile*** (installation agent, inventory agent) - processing priority used for RayManageSofti processes

Download Options

The following preferences influence RayManageSofti download behavior.

General Download Options

- ***AllowByteLevel*** - determines whether byte-level differencing is operational on the managed device
- ***BootstrappedPolicy*** - the location of the policy to be applied to managed devices that do not use policy attached to Active Directory domains
- ***CheckCertificateRevocation*** - Determines whether RayManageSofti checks certificate revocation list when accepting web server signatures from an HTTPS server

- ***HighestPriority*** - the highest upload/download priority that can be assigned to a distribution server
- ***LowestPriority*** - the lowest upload/download priority that can be assigned to a distribution server
- ***PolicyServerPriority*** - specifies the priority to apply to the distribution location identified by the **PolicyServerURL** (internal-only) preference
- ***PostponeByDefault*** - used to postpone the installation of mandatory packages by default (if possible)
- ***PostponementQueryBefore*** - determines whether any alert about postponing an installation is shown before download, before installation, or both
- ***SelectorAlgorithm*** - the algorithm(s) used to determine relative priorities in selecting the distribution server to use for uploads/downloads
- ***StageInactivePackages*** - determines whether the managed device can download files for packages within policies that have future activation times

Peer-to-peer File Sharing Options

The following preferences can be used to control behavior of the RayManageSofti peer download agent on the managed device:

- ***AllowPeerToPeer*** - enable peer-to-peer file sharing.
- ***ApplyPolicy*** - used on managed devices configured for peer-to-peer file sharing, in conjunction with DownloadPolicy, to distinguish between the *Apply a Deployment Manager policy* and *Update policy and package definitions in peer cache* events.
- ***CacheDir*** - the location of the peer cache.
- ***CatalogName*** - the name of the peer downloads file (located in the parent of the CacheDir directory) listing files required by the peer cache.
- ***CheckpointSeconds*** - the frequency (in seconds) with which the peer downloads file is written to disk.
- ***DiskAveragingTime*** - the time period to use to smooth the disk I/O traffic estimate. Used in conjunction with DiskMaxRate.
- ***DiskMaxRate*** - the maximum allowable averaged rate (in bytes per second) of all reads from and writes to disks caused by peer-to-peer file sharing (used in conjunction with DiskAveragingTime).
- ***GCDiskSlice*** - the maximum percentage of DiskMaxRate that can be used for peer cache cleanup operations.
- ***GCMaxInterval*** - the maximum number of minutes the peer download agent should pause between examining files in the peer cache as part of cleanup operations.
- ***GCMinInterval*** - the minimum number of minutes the peer download agent should pause between examining files in the peer cache as part of cleanup operations.
- ***GCPeriod*** - the time period (in hours) over which the peer download agent aims to view every file in the peer cache as part of cleanup operations.
- ***IgnoreConnectionWindows*** - specifies whether to use or ignore the download time periods specified by ParentConnectionWindows and PeerConnectionWindows.
- ***MinFreeDisk*** - the amount of disk space (in MB) that must be free for the peer download agent to download files to the peer cache.
- ***ParentConnectionWindows*** - the time periods during which downloads from distribution servers are permitted.
- ***ParentActivityTimeout*** - the period of time with no download activity, after which a "When connected to network" event is triggered by a file download from a distribution server.
- ***PeerAveragingTime*** - the time period to use to smooth the peer file sharing traffic estimate. Used in conjunction with PeerMaxRate.

- ***PeerConnectionWindows*** - the time periods during which downloads from peer managed devices are permitted.
- ***PeerListenQueue*** - the number of connection requests to queue before refusing additional connections.
- ***PeerMaxRate*** - the maximum allowable averaged rate (in bytes per second) of file transfers between peer managed devices (used in conjunction with PeerAveragingTime).
- ***PeerPullPort*** - the TCP port on which file transfers from managed devices can be received.
- ***PeerPush*** - specifies whether peer managed devices can immediately transfer (push) requested files, or whether they must wait for a request.
- ***PeerSearchDuration*** - the time period during which to search for a file from peer managed devices.
- ***PeerSearchPort*** - the UDP port on which file transfer requests are broadcast.
- ***PeerTransferLimit*** - the number of simultaneous peer-to-peer search and file transfer operations allowed.
- ***PipeName*** (peer download agent) - the name of the pipe used to communicate with the peer download agent.
- ***PolicyPackageRefreshPeriod*** - the number of seconds after successfully downloading package (`.osd`) files during which download of those files should not be attempted again.
- ***PolicyRefreshPeriod*** - the number of seconds after successfully downloading policy (`.npl`) files during which the files should not be downloaded again.
- ***SearchFrequency*** - the number of seconds between issuing peer-to-peer file sharing requests.
- ***SearchMaxOffer*** - the maximum number of offers of a file from peer managed devices to accept before terminating the search.
- ***SearchMinimum*** - the minimum number of search requests to issue for a file being sought from peer managed devices.
- ***SearchRetry*** - the number of seconds to wait after a failed file search attempt before reissuing the request.
- ***UnusedFilePersistence*** - files in the peer cache that have not been accessed within this number of seconds are removed during cleanup operations.
- ***UnusedFileUptime*** - files in the peer cache that have not been accessed within this number of seconds are removed during cleanup operations.
- ***WANAveragingTime*** - the time period to use to smooth the estimate of traffic transferred between the managed device and a distribution server across a wide area network. Used in conjunction with WANMaxRate.
- ***WANMaxRate*** - the maximum allowable averaged rate (in bytes per second) of file transfers between managed devices and a distribution server across a wide area network (used in conjunction with WANAveragingTime).
- ***WANProgressInterval*** - the frequency (in seconds) with which to update peer managed devices with progress about file downloads from a distribution server.
- ***WANRetries*** - the number of times a failed WAN download is retried immediately from each distribution server at each WAN retry interval.
- ***WANRetryDuration*** - the length of time (in minutes) to continue to allow a file to be retried since it was last requested by the installation agent.
- ***WANRetryInterval*** - the length of time (in seconds) after a failed WAN download to retry the download.
- ***WANSearchCurrency*** - how recently (in seconds) a peer search for a file must have occurred, before the file should be downloaded from the closest distribution server.
- ***WANTimeout*** - the time (in seconds) after which to abort stalled transfers of files from distribution server.
- ***WANTransferLimit*** - the maximum number of managed devices that may simultaneously download from a distribution location.

Options that Affect the Actions that Occur after Download

- ***ApplyLocalPolicy*** - policy is applied from the RayManageSofti client's local copy if a new version cannot be generated (if client-side policy merging is in operation) or downloaded (if server-side policy merging is in operation).
- ***NoStage*** - determines whether files are downloaded directly to their destination folder or a staging area.
- ***PolicyPackageRefreshPeriod*** - specifies the number of hours after successfully downloading package (`.osd`) files during which download of those files should not be attempted again.
- ***PolicyRefreshPeriod*** - specifies the number of hours after successfully downloading policy (`.npl`) files during which policy files should not be downloaded again.
- ***VirusScan*** and ***VirusScanCommand*** - determines whether and how files downloaded by RayManageSofti are scanned for viruses before installation.

File Handling Options

The following preferences determine file handling behavior during package processing:

- ***AutoRedundancy*** - determines handling of redundant files during upgrades or downgrades
- ***CacheDir*** - the location to which packages are downloaded prior to installation when peer-to-peer file sharing is enabled
- ***ConfirmSharedFileRemoval*** - determines whether RayManageSofti displays a dialog when removing a file
- ***ForceSharedFileRemove*** - allows deletion of redundant files in the Windows system folder
- ***NoStage*** - determines whether files are downloaded directly to their destination folder or a staging area
- ***PublicAppAccess*** - determines RayManageSofti access to Common folders on Windows XP
- ***StrictInstall*** - if `True`, the policy agent returns a non-zero exit code if any package in policy fails to install

Inventory Options

The following preferences determine how RayManageSofti performs inventory collection on managed devices:

General

- ***Compress*** (inventory agent) - determines whether inventory files are compressed for upload.
- ***GenerateMD5*** - generate an MD5 for every file being tracked.
- ***Hardware*** - determines whether to track hardware (in the machine context).
- ***Inventory*** - location where RayManageSofti uploads inventory files.
- ***InventoryDirectory*** - Custom directory for the storage of inventory data by the inventory agent.
- ***InventoryFile*** - file name of a local copy of the inventory file.
- ***InventoryScriptsDir*** - if RunInventoryScripts is `True`, this specifies the location of scripts to be run after inventory scanning completes.
- ***MachineInventoryDirectory*** - location for machine inventories.
- ***MachineZeroTouchDirectory*** - In case of a remote call the location is used for the machine inventories.
- ***ManageSoftPackages*** - determines the installed software packages.
- ***MinInventoryInterval*** - specifies the minimum interval (in hours) between collections of inventory.
- ***ProgressDepth*** - the number of directory levels to search at initialization to approximate the number of directories searched during tracking.

- ***RunInventoryScripts*** - specifies whether or not to run vulnerability scripts after vulnerability scanning.
- ***ShowIcon*** (inventory agent) - controls whether RayManageSofti displays an icon in the system tray. This preference can be set separately for the installation agent and inventory agent.
- ***SMBIOSCmdLine*** - specifies a command line for non-WMI hardware inventory collection.
- ***TrackFilesInUserInventory*** - determines whether RayManageSofti collects file evidence inventory data when collecting a user inventory.
- ***UploadType*** - determines whether the upload agent uploads machine generated file or user generated files.
- ***UserHardware*** - determines whether to track hardware (in the user context).
- ***UserInteractionLevel*** (inventory agent) - determines the level of user interaction (previously called UILevel).
- ***UserInventoryDirectory*** - location for user inventories on the managed device.
- ***UserZeroTouchDirectory*** - In case of a remote call the location is used for the user inventories.

Options to Control Custom Scripts on Inventory Data

- ***InventoryScriptsDir*** - the location of scripts to be run on inventory data.
- ***RunInventoryScripts*** - when `True`, specifies that scripts should be run on inventory data, prior to uploading inventory data through the distribution hierarchy.

Options to Control Differential Inventory

- ***Difference*** - determines whether differential inventories are performed on the managed device.
- ***GenerationMax*** - number of differential inventories performed between full inventories.
- ***IncrementalDiff*** - when differential inventories are performed, determines what differences RayManageSofti will collect.

Options to Determine Inventory Inclusions and Exclusions

- ***ExcludeDirectory*** - folders to exclude from inventory.
- ***ExcludeExtension*** - file extensions to exclude from inventory.
- ***ExcludeFile*** - files to exclude from inventory.
- ***ExcludeMD5*** - files matching this MD5 checksum are excluded from inventory.
- ***Hardware*** - tracks hardware (in the machine context).
- ***IncludeDirectory*** - folders to include in inventory.
- ***IncludeExtension*** - file extensions to include in inventory.
- ***IncludeFile*** - files to include in inventory.
- ***IncludeMachineInventory*** - specifies whether or not to conduct a computer inventory of hardware and all user packages (only valid if Security Manager installed).
- ***IncludeMD5*** - files matching this MD5 checksum are included in inventory.
- ***IncludePermissionsMask*** - octal mask for file permissions on non-Windows clients that limits which files are scanned for inventory.
- ***IncludeRegistryKey*** - registry keys or values to include in inventory.
- ***IncludeUserInventory*** - specifies whether or not to conduct a user inventory (only valid if Security Manager

installed).

- ***MSI*** - determines whether Microsoft Installer (MSI) packages are included in inventory.
- ***PlatformSpecificPackages*** - determines whether information is included in software inventories about non-Windows, platform-specific packages.
- ***UserHardware*** - tracks hardware (in the user context).
- ***VersionInfo*** - determines whether file version header information is included in inventory.

How RayManageSofti Uses Inventory Inclusion and Exclusion Preferences

For file tracking, many of the inventory preferences work together to determine whether files are included in an inventory file. RayManageSofti determines this in the following way:

1. RayManageSofti first identifies which folders to track during the inventory process. Only folders identified by the **IncludeDirectory** preference are included, unless they are also identified by the **ExcludeDirectory** preference. (**ExcludeDirectory** preferences override **IncludeDirectory** preferences.)

2. For each file within a folder explicitly included in inventory, RayManageSofti performs the following steps to determine whether to include the file in its inventory file. All Exclude preferences override Include preferences.

3. RayManageSofti determines whether the file is explicitly included or excluded, based on whether the values of **ExcludeMD5** and **IncludeMD5** match the file's MD5 checksum value.

4. If the file is not explicitly included or excluded, RayManageSofti determines whether the file name is explicitly included or excluded, based on the values of **ExcludeFile** and **IncludeFile**.

5. If the file name is not explicitly included or excluded, RayManageSofti determines whether the file extension is explicitly included or excluded based on the values of **ExcludeExtension** and **IncludeExtension**.

6. If the file extension is not explicitly included or excluded, it is deemed to be excluded from the inventory file.

Be aware:
By default, `.exe` and `.dll` file extensions are included. This can be overridden by setting **IncludeExtension** to `NULL` or any other value.

Example

In this example, the following preferences are set:

- `Recurse = True`
- `IncludeDirectory = C:\Program Files\`
- `IncludeFile = template.dot`
- `ExcludeExtension = dot`

RayManageSofti evaluates the files `C:\Program Files\Common Files\template.dot` and `C:\Program Files\Common Files\master.dot` in the following way:

- Because both files are within a subfolder of `C:\Program Files\`, RayManageSofti will evaluate them for inclusion/exclusion.

- There are no **IncludeMD5** or **ExcludeMD5** preferences to evaluate, so RayManageSofti cannot explicitly include or exclude the file, and moves on to evaluate file names.

- The **IncludeMD5** preference specifically shows that `template.dot` should be included. It does not explicitly include or exclude `master.dot`, so RayManageSofti moves on to evaluate file extensions.

- The dot extension is explicitly excluded so `master.dot` is excluded. Because `template.dot` has already been explicitly included, its file extension is not evaluated.

Logging Options

Logging options are available for each of the RayManageSofti smart-agents running on managed devices, although the number of preferences applicable to each agent does vary. They work in the same way for each agent, but work on different log files.

General Logging Preferences

- ***Log*** - location where RayManageSofti uploads logging files from the managed device
- ***PolicyComplianceLog*** - location where RayManageSofti uploads policy compliance log files from the managed device
- ***UploadType*** - determines whether the upload agent uploads machine generated file or user generated files

Installation Agent

- ***LogFileOld*** (installation agent) - name of file to store additional logging information
- ***LogFileSize*** (installation agent) - maximum log file size
- ***LogLevel*** (installation agent) - level of logging returned by the smart-agent

Package Selection Agent

- ***ConfigFile*** - name of the system copy of the configuration file used by the selection agent
- ***ConfigFileDefault*** - name of the default configuration file to use when other settings fail
- ***Locale*** - locale to use for selection agent localization
- ***LocaleDefault*** - locale to use in the absence of other settings
- ***ReInstallRequiresVersionChange*** - number of minutes between automatic refresh of data displayed by the selection agent

Policy Agent

- ***BootstrappedPolicy*** - the location of the policy to be applied to managed devices that do not use policy attached to Active Directory domains
- ***InstallationStatusRefreshPeriod*** - specifies how frequently (in seconds) RayManageSofti should recreate installation events for packages that are installed, or flagged as not required.
- ***LogInstallCheck*** - specifies whether RayManageSofti should recreate installation events while checking packages for installation or upgrade
- ***LogInstallFail*** - specifies whether RayManageSofti should log failed installation attempts
- ***LogInstallPass*** - specifies whether RayManageSofti should log successful installation events
- ***LogUninstallFail*** - specifies whether RayManageSofti should log failed uninstallation attempts
- ***LogUninstallPass*** - specifies whether RayManageSofti should log successful uninstallation events

RayManageSoft Folder Locations

The following preferences contain information about folders that RayManageSofti uses. You can reference these preferences in package details and programmed callouts.

Installation and Package Folders

- ***BaseURL*** - Web location from which the current application can be retrieved
- ***CacheDir*** - the location of the peer cache
- ***GlobalConfigSource*** - identifies a URL that contains installation preferences

Inventory Folders

- ***Inventory*** - location where RayManageSofti uploads inventory files
- ***MachineInventoryDirectory*** - location for machine inventories
- ***UserInventoryDirectory*** - location for user inventories on the managed device

Policy Folders

- ***BootstrappedPolicy*** - the location of the policy to be applied to managed devices that do not use policy attached to Active Directory domains
- ***MachinePolicyDirectory*** - location in which to save the current machine policy
- ***MachinePolicyPackageDirectory*** - location where package information associated with machine policy is cached
- ***PolicySource*** - the location where policy is generated
- ***UserPolicyDirectory*** - location in which to save active user policies
- ***UserPolicyPackageDirectory*** - location where package information associated with user policy is cached

Other

- ***Log*** - location where RayManageSofti uploads logging files from the managed device

MSI Package Options

RayManageSofti has a number of preferences that are used to support Microsoft Windows Installer (also referred to as MSI), and are used for building the command line parameters to be passed to msiexec.exe.

When you reference one of these preferences as a project variable within a RayManageSofti package, you must prefix the project variable name with the **$** symbol and enclose the project variable name in brackets. The **$** prefix indicates to RayManageSofti that the variable must be expanded when the package is installed.

For project variables that are going to be passed to the `msiexec.exe` command line, you must also prefix the project variable name (inside the brackets) with the **!** character (also called the *unquoted value operator*). This makes sure that empty strings are not displayed in quotes when a project variable with an empty value is expanded in the command line of msiexec.exe. An example of the correct syntax is: **$(!VariableName)**.

The following MSI preferences are available. For more information on MSI command lines, refer to the *Windows Platform Windows Installer SDK*.

- ***MSI*** - determines whether Microsoft Installer (MSI) packages are included in inventory
- ***MsiBaseURL*** - web location from which applications can be retrieved
- ***MsiReinstallFeatures*** - MSI components to be installed (equivalent to MSI property `REINSTALL`)
- ***MsiReinstallModeLevel*** - identifies what will be reinstalled (equivalent to MSI property `REINSTALLMODE`)
- ***MsiRepair*** - determines if MSI repair operations are performed at the same time as RayManageSofti self-healing operations
- ***MsiRepairLevel*** - identifies what will be repaired (equivalent to MSI property `REINSTALLMODE`)
- ***MsiSourceLocation*** - determines whether Windows Installer packages are installed from the managed device's local Windows Installer cache, or a distribution location
- ***MsiUILevel*** - determines user interaction level for MSI (equivalent to option /q in msiexec.exe command line)
- ***MsiUninstallArgs*** - arguments to include in the MSI command line for uninstall operations

Network Speed and Connection Options

The following preferences influence how RayManageSofti uses networks for operations.

Speed

- ***MinimumDCSpeed*** - determines the minimum speed between the managed device and domain controller that is required to apply client-side policy
- ***NetworkHighSpeed*** (installation agent) and ***NetworkHighSpeed*** (upload agent) - lowest network speed to consider to be high-speed network connection
- ***NetworkHighSpeed*** (installation agent) - maximum bandwidth for high-speed connections
- ***NetworkLowUsage*** - maximum bandwidth for low-speed connections
- ***NetworkMaxByteLevelSpeed*** - speed at which byte-level differencing is disabled (no significant advantages in performing byte-level differencing for very high speed connections)
- ***NetworkMaxRate*** (installation agent) and ***NetworkMaxRate*** (upload agent) - rate at which the managed device accesses data over the network
- ***NetworkMinSpeed*** (installation agent) and ***NetworkMinSpeed*** (upload agent) - minimum network speed at which RayManageSofti will install or update a package

Protocols, Addresses, and Authentication

- ***LogonServer*** - name of the logon server computer to which the managed device normally connects

Other

- ***ConnectionAttempts*** - the number of times that a "no connection is available" error can be reported while trying to connect to a particular distribution location as a file share
- ***EventNetType*** - determines the type of network connections that are required to start events with an OnConnect trigger
- ***HighestPriority*** - the highest upload/download priority that can be assigned to a distribution server
- ***LowestPriority*** - the lowest upload/download priority that can be assigned to a distribution server
- ***ndsensNetType*** - determines what type of network connections are monitored
- ***NetworkRetries*** - number of times failed network operations are retried before an alternative distribution location is attempted
- ***NetworkSense*** (installation agent), ***NetworkSense*** (inventory agent), ***NetworkSense*** (security agent), and

NetworkSense (upload agent) - determines whether network checks are bypassed

- ***NetworkTimeout*** (installation agent) and ***NetworkTimeout*** (upload agent) - number of seconds of inactivity before a network operation will time out
- ***SelectorAlgorithm*** - the algorithm(s) used to determine relative priorities in selecting the distribution server to use for uploads/downloads

Package-level Filtering Options

The following options contain information on package-level filtering during a RayManageSofti client-side policy merge.

- ***DisablePackageFiltering*** - allows you to skip package-level filtering during a policy merge if filtering is not required
- ***TrackFilesInUserInventory*** - used to detect changes to group membership in subsequent merges by storing group membership from the last policy merge

Package Selector Options

The following preferences can be used to control behavior of the RayManageSofti package selection agent on the managed device.

- ***ApplicationInstallCommand*** - a template command line to be used to install an application package through the RayManageSofti package selection agent
- ***ApplicationUninstallCommand*** - a template command line to be used to uninstall an application package through the RayManageSofti package selection agent
- ***ApplicationVerifyCommand*** - a template command line to be used to verify/ repair an application package through the RayManageSofti package selection agent
- ***ConfigFile*** - name of the configuration file used by the RayManageSofti user interface on managed devices
- ***DiskReservedKB*** - the amount of disk space to reserve on each drive
- ***ReInstallRequiresVersionChange*** - number of minutes between automatic refresh of data displayed by the RayManageSofti user interface on a managed device

Policy Merge Options

These preferences influence how RayManageSofti merges Active Directory group policy. For managed devices that do not use Active Directory group policy, no policy merging is performed. Instead, the location of policy is specified by BootstrappedPolicy.

General

- ***AutoDetectDC*** - determines how RayManageSofti selects a domain controller for client-side policy merging
- ***EnablePolicyFailOver*** - specifies whether a server-side policy file should be applied if a client-side policy file is unavailable
- ***HideMachineUI*** - determines whether RayManageSofti displays a user interface when applying machine policy
- ***LauncherCommandLine*** - installation agent command line options to pass to RayManageSofti when applying policy information
- ***MinimumDCSpeed*** - determines the minimum speed between the managed device and domain controller that is required to apply client-side policy

- ***RetryPolicy*** - determines whether RayManageSofti will attempt to retrieve RayManageSofti policy when the managed device boots, if no machine schedule exists on the managed device
- ***RetryPolicyCommand*** - the command used to retrieve policy if RetryPolicy is set to `True`

Locations

- ***MachinePolicyDirectory*** - location in which to save the current machine policy
- ***MachinePolicyPackageDirectory*** - location where package information associated with machine policy is cached
- ***PolicyComplianceLog*** - location where RayManageSofti uploads policy compliance log files from the managed device
- ***PolicySource*** - the location where policy is generated
- ***UserPolicyDirectory*** - location in which to save active user policies
- ***UserPolicyPackageDirectory*** - location where package information associated with user policy is cached

Preference Management Options

The following preferences can determine the way RayManageSofti evaluates preferences on managed devices.

- ***CmdLineOverrides*** - determines whether options set on the command line override fixed preferences in the registry or network preference file.
- ***GlobalConfigSource*** - identifies a URL that contains installation preferences
- ***IncludeRegistryKey*** - registry keys or values to include in inventory
- ***MachineAlternateRegistryHive*** - alternate registry hive
- ***SaveAllUserSymbols*** - determines whether RayManageSofti retains installation preferences set by a top-level or prerequisite package. Also see *Persistent managed device preferences*

Prerequisite Package Options

The following preferences influence the installation of prerequisite packages.

- ***AskAboutDependencies*** - determines whether RayManageSofti prompts the user before prerequisite packages are installed
- ***PropagatePkgChanged*** - reinstalls the base package if the prerequisite package has changed for Third party installer packages
- ***SaveAllUserSymbols*** - determines whether RayManageSofti retains installation preferences set by a top-level or prerequisite package

Reboot Options

Reboot behavior is controlled by the installation agent in the final stages of its processing. This means that, in the normal course of events where package updates are occurring as part of a policy check, the reboot behavior effectively applies at a policy level (if one or more packages require a reboot to occur, a single reboot process is initiated at the end of the policy check). In other cases, where a package update occurs singly (perhaps because of a special schedule event, or through end-user action in the selection agent), the reboot can be seen to apply at the individual package level.

Installation Agent

The following preferences can be used to control reboot behavior during package processing. (For information on reboot options that relate specifically to Security Manager, see *Reboot options* of *Security management options*.)

- ***AllowRebootIfLocked*** - controls whether RayManageSofti reboots the managed device if the package being installed requires it and the desktop is locked
- ***AllowRebootIfServer*** - controls whether RayManageSofti reboots the managed device if it is a server (used by the adoption agent, but not the installation agent)
- ***AllowTimeoutIfLocked*** - controls whether the time interval for prompting the end-user commences immediately if the desktop is locked, or commences when the desktop is unlocked
- ***AlwaysDisplayReboot*** - controls whether RayManageSofti displays a warning to the end-user before rebooting (overrides UserInteractionLevel (installation agent))
- ***ForceReboot*** - determines whether RayManageSofti performs a forced reboot if the desktop is not locked. A forced reboot suppresses any user interaction required to close other applications that may be running
- ***ForceRebootIfLocked*** - determines whether RayManageSofti performs a forced reboot if the end-user's desktop is locked. A forced reboot suppresses any user interaction required to close other applications that may be running
- ***RebootCmdLine*** - used on the managed device to reboot from the command line
- ***RebootContinueAfterCmdFailure*** - controls whether or not to proceed with the reboot if the RebootPreCommand returned a non-zero exit code
- ***RebootIfRequired*** - controls whether RayManageSofti reboots the managed device if the package being installed requires it and the end-user's desktop is not locked
- ***RebootPostCommand*** - a command to be executed after the managed device is rebooted
- ***RebootPreCommand*** - a command to be executed before the managed device is rebooted
- ***RebootPromptCycles*** - the number of times the end-user can postpone the managed device reboot. Prompts occur at intervals specified by RebootPromptWait
- ***RebootPromptUnlimited*** - keep prompting the end-user, at intervals specified by RebootPromptWait, until the managed device reboots (equivalent to RebootPromptCycles=-1)
- ***RebootPromptWait*** - the number of seconds to wait after the end-user dismisses the reboot dialog before displaying it again
- ***UITimeoutWait*** - the number of seconds that installation agent dialogs display before timing out
- ***RenotifyTimeout*** - the number of seconds that the installation agent waits before once again showing a user any hidden dialogs that have not yet timed out
- ***UserInteractionLevel*** (installation agent) - determines the level of user interaction (previously called UILevel). This preference can be set separately for the adoption agent, installation agent, and inventory agent. In relation to reboot options, only the installation agent setting is applicable

When RayManageSofti identifies that a reboot is required, the combination of these preferences determines the action that is taken.

Reboot Agent Preferences

The installation agent calls the reboot agent (`reboot.exe`) as necessary. If you choose to run the reboot agent independently, the following preferences apply:

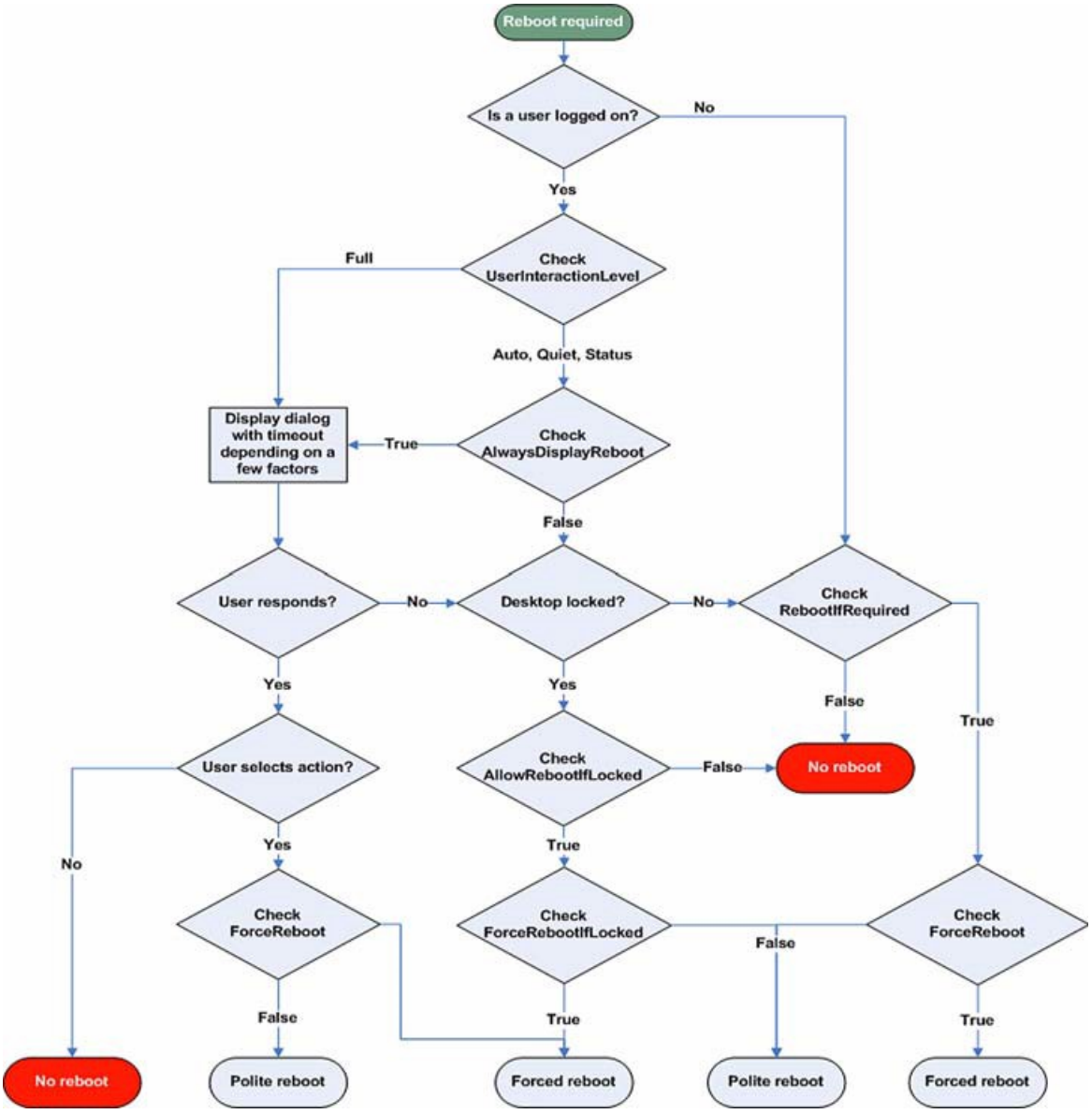
- ***RebootContinueAfterCmdFailure*** - specifies whether or not to continue with rebooting a managed device if execution of a pre-reboot command returns a non-zero exit code
- ***RebootPostCommand*** - a command to run immediately after a managed device is rebooted

- ***RebootPreCommand*** - a command to run immediately before a managed device is rebooted
- ***RebootPromptCycles*** - the number of times an end-user can postpone a managed device reboot
- ***RebootPromptWait*** - the time interval, in seconds, to wait before re-displaying the dialog that prompts the end-user to reboot

The following decision tree illustrates how the installation agent reboot preferences work together. In this diagram:

Polite reboot means a reboot action where RayManageSofti alerts other open applications and requests that they shut down in an orderly manner. Notice that such other applications may present dialogs to the end-user through which the end-user might cancel the reboot process. In that case, some newly-installed or updated applications may not function until the next reboot. (A polite reboot does not involve direct interaction between RayManageSofti and the end-user.)

Forced reboot means that RayManageSofti ignores the state of all other applications and initiates the reboot. Only the operating system can override this. A forced reboot may cause the loss of data from open applications, and should be used with care.



Decision tree to determine correct reboot behavior on managed device, based on preference settings

Remote Execution Options

The following preference can be used to specify behavior when tasks are being remotely executed on managed devices.

- ***ListeningPort*** - the port number that the TCP-based listening agent monitors for incoming requests. If not specified, port `7020` is used

Scheduling Options

The following preferences can be used to control RayManageSofti task scheduling on the managed device.

General

- ***ApplyPolicyIfLoggedOn*** - specifies whether or not computer policy is applied at the scheduled time if a user is logged on (Windows devices only)
- ***DisablePeriod*** - the number of seconds for which RayManageSofti user schedules remain disabled when the end-user disables them in the schedule agent on the managed device (Windows devices only)
- ***NativeScheduler*** - whether the Microsoft Task Scheduler or RayManageSofti Task Scheduler is in use (the Microsoft Task Scheduler is only available on Windows devices)
- ***PolicyPackageRefreshPeriod*** - specifies the number of hours after successfully downloading package (`.osd`) files during which download of those files should not be attempted again
- ***PolicyRefreshPeriod*** - specifies the number of hours after successfully downloading policy (`.npl`) files during which policy files should not be downloaded again
- ***RetryPolicy*** - whether RayManageSofti will attempt to retrieve RayManageSofti policy when the managed device boots, if no machine schedule exists on the managed device
- ***RetryPolicyCommand*** - the command used to retrieve policy if RetryPolicy is set to `True`

Triggers and Events

- ***ndsensNetType*** or ***EventNetType*** - determines the types of events that are set to execute only when the network is available and to identify the type of network to look for (Windows devices only)

Security Options

The following preferences can be used to determine security behavior during package processing, uploads and downloads. They relate to Authenticode checks and passwords for FTP authentication.

- ***CheckCertificateRevocation*** - determines whether RayManageSofti checks certificate revocation list when accepting web server signatures from an HTTPS server
- ***DisplayAllAuthcode*** - determines the subsequent behavior after RayManageSofti encounters an invalid signature when performing an Authenticode check
- ***ForceValidSignature*** - determines whether RayManageSofti prompts the user before installing a package when Authenticode signatures are valid
- ***PublicAppAccess*** - determines RayManageSofti access to **Common** folders on Windows XP
- ***VerifyCatalogSigned*** - determines whether Authenticode digital signatures are checked in the RayManageSofti catalog (`.ndc`) file before packages are installed
- ***VerifyFilesSigned*** - determines whether executable files downloaded by RayManageSofti are checked for a valid Authenticode digital signature before being installed
- ***AllowedGroups*** – determines whether RayManageSofti checks that the current user is allowed to interact directly with the Deployment Manager Agent.

Also see *Trusted location options*.

Security Management Options

The following preferences can be used to control the behavior of Security Manager.

Be aware:
This list is followed by separate lists of preferences that relate specifically to:

- Local device management
- Reboot options

- ***AnalyseMSSECURE*** - specifies whether to perform security analysis using `mssecure.xml` as well as MBSA
- ***BlockedApplicationIncludeLocalAdmin*** - if `True`, users with administrator privileges are also prevented from accessing blocked applications on managed devices; if `False`, local administrator users can access blocked applications
- ***BlockedApplicationMessage*** - the message to display to end-users when an application is prevented from running
- ***BlockedApplicationNotify*** - determines if end-users are to be notified when an application is prevented from running
- ***BlockedApplications*** - applications that are to be prevented from running
- ***MachineSecurityAnalysisDirectory*** - the path to the folder in which the security analysis files are stored locally
- ***MBSA2CmdLine*** - command line to check security compliance using MBSA 2.0
- ***MBSA2PATH*** - the installation folder of MBSA 2.0
- ***MBSACmdLine*** - command line to check security compliance using MBSA 1.2
- ***MBSAPATH*** - the installation folder of MBSA 1.2
- ***MBSAVulnerabilityCmdLine*** - command line to run to scan a managed device for vulnerabilities
- ***MSSECUREPath*** - location of downloaded `mssecure.xml` file used to check security compliance
- ***OfficeConvertPath*** - location of `convert.exe` file, used when determining Microsoft Office patches applicable for a managed device
- ***OfficeInventoryPath*** - location of `inventory.exe` file, used when determining Microsoft Office patches applicable for a managed device
- ***RunVulnerabilityScripts*** - specifies whether or not to run vulnerability scripts after vulnerability scanning
- ***Security*** - determines whether to perform security compliance checking
- ***SecurityAnalysis*** - defines the naming conventions for uploaded files containing security analysis details
- ***SecurityLogInterval*** - the time interval at which security logs are generated
- ***SessionListInterval*** - the time interval at which user session lists are generated
- ***TechniciansGroup*** - the groups whose members can override security policies on managed devices
- ***VulnerabilityScripts*** - if RunVulnerabilityScripts is `True`, this specifies the scripts to be run after vulnerability scanning completes
- ***VulnerabilityScriptsDir*** - if RunVulnerabilityScripts is `True`, this specifies the location of scripts to be run after vulnerability scanning completes

Local Device Management

The following preferences relate specifically to local device settings:

- ***LocalDeviceBluetooth*** - whether Bluetooth devices are enabled (`True`) or disabled (`False`)
- ***LocalDeviceCDDrive*** - whether CD/DVD drives are enabled (`True`) or disabled (`False`)
- ***LocalDeviceFirewire*** - whether all firewire ports are enabled (`True`) or disabled (`False`)
- ***LocalDeviceFloppyDisk*** - whether floppy disk drives are enabled (`True`) or disabled (`False`)
- ***LocalDeviceInfrared*** - whether infrared devices are enabled (`True`) or disabled (`False`)
- ***LocalDeviceModem*** - whether modems are enabled (`True`) or disabled (`False`)

- ***LocalDeviceParallelPorts*** - whether parallel ports are enabled (`True`) or disabled (`False`)
- ***LocalDevicePCMCIA*** - whether PCMCIA cards are enabled (`True`) or disabled (`False`)
- ***LocalDeviceSerialPorts*** - whether serial ports are enabled (`True`) or disabled (`False`)
- ***LocalDeviceTimeout*** - number of seconds the security agent will wait for a managed device to respond to a local device management request before timing out
- ***LocalDeviceUSB*** - whether removal storage devices, such as USB and PCMCIA cards, are enabled (`True`) or disabled (`False`)
- ***LocalDeviceWiFi*** - whether WiFi 802.11x devices are enabled (`True`) or disabled (`False`)

Reboot Options

The following preferences can be used to control reboot behavior for security management.

- ***LocalDeviceAutoRebootMessage*** - message displayed to the end-user before the managed device is automatically rebooted
- ***LocalDeviceAutorebootNotLoggedIn*** - whether to reboot automatically if LocalDeviceAutoRebootMessage is set and no end-user is logged on
- ***LocalDeviceAutoRebootPeriod*** - if LocalDeviceAutorebootWithNoWait is not set, this specifies the number of minutes before a reboot takes place
- ***LocalDeviceAutorebootWithNoWait*** - if LocalDeviceForceReboot is not set and no user is logged on, this determines whether reboot takes place automatically (`True`) or after a specified period (`False`)
- ***LocalDeviceForceReboot*** - if any security management settings require a reboot to become effective, this specifies whether to force the reboot (`True`) or prompt the end-user to reboot (`False`)
- ***SecurityPatchRebootIfRequired*** - sets the default response to dialogs displayed during security patch installation that prompt the end-user to allow a reboot

Security Service Options

The following preference can be used to determine security behavior associated with the security service that runs on managed devices.

- ***Compress*** (security service) - determines whether security service event logs are compressed for upload

Self-heal Options

The following preferences determine how RayManageSofti self-heal operations work on the managed device.

- ***ApplicationVerifyCommand*** - a template command line to be used to verify/ repair an application package through the RayManageSofti package selection agent
- ***CheckCatalogDigest*** - determines whether RayManageSofti performs a check on package-level MD5 digests during self-healing operations
- ***CheckFileDigest*** - determines whether RayManageSofti performs a check on file-level MD5 digests during self-healing operations
- ***CheckRegistry*** (or reg on the command line) - determines if RayManageSofti performs self-healing on registry keys and preference files
- ***MsiRepair*** - determines if MSI repair operations are performed at the same time as RayManageSofti self-healing operations

- ***MsiSourceLocation*** - determines whether Windows Installer packages are installed from the managed device's local Windows Installer cache, or a distribution location
- ***SelfHeal*** - determines whether self-healing should take place for an individual package
- ***SupplyWorstCaseReturnValue*** - determines whether RayManageSofti returns an error only when an installation agent operation fails, or also when upgrades or self-heal operations fail

Trusted Location Options

The following preferences determine whether RayManageSofti trusted locations are used.

- ***UseTrustDatabase*** - determines whether RayManageSofti only downloads from trusted locations
- ***TrustDatabaseFxd*** - determines whether users need administrator permissions to change trusted and excluded locations

To Add a Trusted Location

To identify a trusted location, create a key for the location under the following registry key:
`[Registry]\ManageSoft\Launcher\CurrentVersion\TrustedLocations`

Then create the following subkeys and assign values to identify the location.

- **Directory** - the location of the distribution location within the specified host. To include subfolders, append `\*` to the end of the value
- **Host** - the computer on which the distribution location is hosted
- **Port** - the port number for data transfer to the managed device
- **Protocol** - the protocol for transferring files to the managed device

To Add an Excluded Location

To identify an excluded location, create a key for the location under the following registry key:
`[Registry]\ManageSoft\Launcher\CurrentVersion\ExcludedLocations`

Then create the following subkeys and assign values to identify the location.

- **Directory** - the location of the distribution location within the specified host. To include subfolders, append `\*` to the end of the value.
- **Host** - the computer on which the distribution location is hosted.
- **Port** - the port number for data transfer to the managed device.
- **Protocol** - the protocol for transferring files to the managed device.

How RayManageSofti Identifies Trusted Locations

- If the ***UseTrustDatabase*** preference is set to True, RayManageSofti determines trusted locations based on the following rules.
- Only servers identified under the **TrustedLocations** key are included, unless they are also listed under the **ExcludedLocations** key.
- If there are no servers identified under the **TrustedLocations** key, all servers are trusted unless listed under the **ExcludedLocations** key.

Uninstall Options

The following preferences influence uninstall-behavior on the managed device:

- ***ApplicationUninstallCommand*** - template command line to uninstall an application package through the RayManageSofti package selection agent
- ***ForceSharedFileRemove*** - determines whether shared files in the Windows system folder are deleted during uninstalls
- ***InstallerARPRemove*** - determines whether external installer packages can be uninstalled via the Windows **Add/Remove programs** applet
- ***UninstallIShieldSilently*** - controls whether the user confirmation dialog will be displayed during removal of InstallShield packages
- ***UninstallString*** - the string to uninstall an application

Upgrade / Downgrade Options

The following preferences determine how upgrades and downgrades are performed on the managed device.

- ***AutoRedundancy*** and ***ForceSharedFileRemove*** - determines handling of redundant files during upgrades or downgrades
- ***ReInstallRequiresVersionChange*** - determines when packages will be upgraded, downgraded, or reinstalled, based on the type of changes made to the package
- ***SupplyWorstCaseReturnValue*** - determines whether RayManageSofti returns an error only when an installation agent operation fails, or also when upgrades or self-heal operations fail

Upload Options

The following preferences influence how information is uploaded from the managed device to reporting locations.

General

- ***CheckCertificateRevocation*** - Determines whether RayManageSofti checks certificate revocation list when accepting web server signatures from an HTTPS server.
- ***Compress*** (inventory agent) - determines whether inventory files are compressed for upload
- ***HighestPriority*** - the highest upload/download priority that can be assigned to a distribution server
- ***LowestPriority*** - the lowest upload/download priority that can be assigned to a distribution server
- ***SelectorAlgorithm*** - the algorithm(s) used to determine relative priorities in selecting the distribution server to use for uploads/downloads
- ***SourceRemove*** - determines whether the upload agent removes uploaded files from the source location after a successful upload
- ***UploadType*** - determines whether the upload agent uploads machine generated file or user generated files

Locations from Which Data are Uploaded

- ***Inventory*** - location where RayManageSofti uploads inventory files
- ***Log*** - location where RayManageSofti uploads logging files from the managed device

- ***PolicyComplianceLog*** - location where RayManageSofti uploads policy compliance log files from the managed device
- ***SourceFile*** - file or files to be uploaded via the upload agent
- ***SourceRemove*** - determines whether the upload agent removes uploaded files from the source location after a successful upload

User Interaction Options

The following preferences control RayManageSofti user interaction on Windows managed devices. These preferences are ignored on non-Windows managed devices.

General

- ***HideMachineUI*** - determines whether RayManageSofti displays a user interface when applying machine policy
- ***QuietUntilUpdate*** - controls whether the RayManageSofti user interface is hidden if no user interaction is necessary
- ***ShowIcon*** (installation agent), ***ShowIcon*** (inventory agent) - controls whether RayManageSofti displays an icon in the system tray
- ***UITimeoutWait*** - the number of seconds that installation agent dialogs display before timing out
- ***UserInteractionLevel*** (installation agent), ***UserInteractionLevel*** (adoption agent), ***UserInteractionLevel*** (inventory agent) - determines the level of user interaction (previously called UILevel)

Prompts and Confirmations

- ***AlwaysDisplayReboot*** - controls whether RayManageSofti displays a warning to the user before performing any reboot required by a package installation (overrides `UserInteractionLevel`)
- ***AskAboutDependencies*** - determines whether RayManageSofti prompts the user before prerequisite packages are installed
- ***AskBeforeInstalling*** - determines whether RayManageSofti prompts the user before installing a package
- ***AutoPromptOnInstallCompletion*** - when the UserInteractionLevel is set to Auto, determines whether RayManageSofti informs the user when package installation is complete
- ***AutoPromptOnUnInstallCompletion*** - when the UserInteractionLevel is set to Auto, determines whether RayManageSofti informs the user when package un-installation is complete
- ***ConfirmSharedFileRemoval*** - determines whether RayManageSofti displays a dialog when removing a file
- ***PostponementQueryBefore*** - determines whether any alert about postponing an installation is shown before download, before installation, or both
- ***PostponeUserInteractionLevel*** - controls whether end-users on managed devices are interactively asked if they want to postpone installations of mandatory packages that are appropriately configured in policy
- ***PromptOnCOMRegFailures*** - determines whether RayManageSofti prompts the user if it fails to register a COM server
- ***PromptOnInstallCompletion*** - when the UserInteractionLevel is set to Full, determines whether RayManageSofti informs the user when package installation is complete
- ***PromptOnUnInstallCompletion*** - when the UserInteractionLevel is set to Full, determines whether RayManageSofti informs the user when package un-installation is complete
- ***SupplyWorstCaseReturnValue*** - determines whether RayManageSofti returns an error only when an

installation agent operation fails, or also when upgrades or self-heal operations fail

Virus Scanning Options

The following preferences can be used to run virus scans on the managed device.

- ***VirusScan*** - determines whether files downloaded by RayManageSofti are scanned for viruses before installation
- ***VirusScanCommand*** - the command used to perform virus scanning

Windows Folder Information

The following preferences contain information about Windows folders on the managed device. You can reference these preferences in package details and programmed callouts.

- ***AppDataFolder*** - path to the folder in which user-specific application details are located
- ***CommonProgramMenuFolder*** - path to Start menu program folders and shortcuts for `[ALL USERS]`

Alphabetical Listing of Preferences

This chapter describes each preference, in alphabetical sequence.

Preference Details

For each preference, details in this chapter include:

- A description of the preference
- Possible values
- Registry details
- Project variable usage
- Command line usage

AddRemove

Command line | Registry | Project variable

When set to `Create` or `Default`, all packages installed by RayManageSofti create an entry in the **Add/Remove Programs** control panel applet.
When set to `NoCreate`, RayManageSofti does not create an entry in the **Add/Remove Programs** applet.

Be aware:
This preference only applies to native RayManageSofti packages. It does not, therefore, apply to MSI-based or general third-party packages.

Also be aware:
The "default" setting means two things:

- For new installations, an `AddRemove` entry is created
- For updated installations, creation of an `AddRemove` entry is dictated by the previous version of the application. For example, if the application had an `AddRemove` entry, then one will be created. If it did not, then RayManageSofti will not create an `AddRemove` entry.

Values / range:	Default, Create, NoCreate
Default value:	`Default`

Command Line

Tool:	Installation agent
Example:	`ndlaunch -o AddRemove=NoCreate -r http://myserver/mypkg.osd`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`AddRemove`
Reference as:	`$(AddRemove)`

AllowByteLevel

Command line | Registry | Project variable

When set to True, RayManageSofti uses dynamic byte-level differencing when downloading files in the package for which byte-level differencing has been applied.
When set to False, dynamic byte-level differencing is not used.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o AllowBytelevel=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`AllowByteLevel`
Reference as:	`$(AllowByteLevel)`

AllowedGroups

When set at least to one security group, RayManageSofti checks the current user to be a member of at least one given group when directly interacting with the Deployment Manager Agent.

When no value is set, RayManageSofti does not check any membership. Every user is allowed to directly interact with the Deployment Manager Agent.

Values / range:	String
Default value:	`Empty`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

AllowedPkgSubtypes

Security Manager only
Command line | Registry

A semi-colon (;) separated list of installable package types. Policy installs are restricted to the listed types. An empty value allows installation of all package types.
The package types available are:

- `Standard` (normal RayManageSofti packages)
- `SecurityPatch` (all Security Manager packages)
- `SecurityPatch:MicrosoftWindows` (only Security Manager packages associated with Microsoft Windows packages)
- `SecurityPatch:MicrosoftOffice` (only Security Manager packages associated with Microsoft Office patches)
- `SecuritySettings` (RayManageSofti security patch settings packages, which contain XML files of security bulletin data).

Note:
If you want to define your own package types, set the custom property `PackageSubType` as a project variable in a package (for example, `PackageSubType=MyCustomType`).

Values / range:	String
Default value:	`{empty}`
Example value:	`Standard;SecurityPatch`

Command Line

Tool:	Installation agent, policy agent
Example:	`-o AllowedPkgSubtypes="SecurityPatch"`

Registry

Installed by:	Security Manager internals
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common`
Computer preference:	`[Registry]\ManageSoft\Common`

AllowPeerToPeer

Registry

Specifies whether managed devices can obtain downloaded files from other managed devices on the same LAN (peers). Peer-to-peer file sharing minimizes download volumes from distribution servers, and makes downloaded files available to managed devices at LAN speed.

Values / range:	Boolean (true or false)
Default value:	`False`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common`
Computer preference:	`[Registry]\ManageSoft\Common`

AllowRebootIfLocked

Command line | Registry

Determines whether to reboot if RayManageSofti has determined that a reboot is necessary and the end-user's desktop is locked.

If **AllowRebootIfLocked** and **ForceRebootIfLocked** are set to `True`, the computer reboots immediately without prompting the end-user.

This preference sets the default response to the installation agent dialog that prompts end-users to confirm a reboot:

- When set to False, the default response is to not reboot, and the dialog times out according to **UITimeoutWait**
- When set to True, the default response is to confirm the reboot, and the dialog does not time out, because a polite reboot is not possible while the desktop is locked.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o AllowRebootIfLocked=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

AllowRebootIfServer

Command line | Registry

This registry entry is not used by RayManageSofti on managed devices. It may be used when computers are being automatically adopted under management as described in the *Discovery and Adoption* chapter of the *RMS Configuration*.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Adoption agent
Example:	`-o AllowRebootIfServer=True`

Registry

Installed by:	Adoption of computers under RayManageSofti management, installation of RayManageSofti on managed devices
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common\Rules`
Computer preference:	`[Registry]\ManageSoft\Common\Rules`

AllowTimeoutIfLocked

Registry

Specify whether the process of prompting the end-user to postpone a reboot starts immediately on locked desktops, or only when the desktop is unlocked.

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Installation or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

AlwaysDisplayReboot

Command line | Registry

This preference applies when a reboot is required as part of a package installation.
When set to True, RayManageSofti displays a warning to the end-user, regardless of the setting of **UserInteractionLevel (installation agent)**.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o AlwaysDisplayReboot=True`

Registry

Installed by:	Adoption of computers under RayManageSofti management, installation of RayManageSofti on managed devices
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common\Rules`
Computer preference:	`[Registry]\ManageSoft\Common\Rules`

AnalyseMSSECURE

Command line | Registry

Specifies whether (`True`) or not (`False`) the RayManageSofti installation agent and Security Manager security agent perform security analysis based on data in `mssecure.xml`.
The scan is functionally equivalent to that performed by MBSA 1.2.1, so if MBSA 1.2.1 is installed, Raynet recommends that you set this preference to `False`.

MBSA 2.0 does not yet handle all patches. If MBSA 2.0 is installed, the recommendation for this setting varies according to whether MBSA 1.2.1 is also installed:

- If both MBSA 2.0 and 1.2.1 are installed, Raynet recommends that you set this preference to `False`.
- If both MBSA 2.0 alone is installed, Raynet recommends that you set this preference to `True`.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent, security agent
Example:	`-o AnalyseMSSECURE=True`

Registry

Installed by:	Deployment Manager internals or manual configuration
User preference:	For the RayManageSofti installation agent, in order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Common` For the Security Manager security agent, in order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Common\` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	For the Security Manager security agent, in order of precedence: • `[Registry]\ManageSoft\Common\` • `[Registry]\ManageSoft\Security Agent\CurrentVersion` • `[Registry]\ManageSoft\Launcher\CurrentVersion` For the RayManageSofti installation agent, in order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

AppDataFolder

Project variable

The path to the folder in which user-specific application details are located.

Values / range:	A local directory name. Read-only preference.
Default value:	The default installation of Windows uses the `[USER PROFILE]\Application Data` folder.
Example value:	`C:\Users\Jane\Application Data` `C:\Documents and Settings\Anita\Application Data`

Project Variable

Define as:	Predefined within Windows.
Reference as:	`$(AppDataFolder)`

ApplicationInstallCommand

Command line | Registry

Specifies a template command line to be used to install an application package through the RayManageSofti package selection agent. The selection agent uses the value from this preference to build command lines that are executed in order to install packages selected by a user.

This preference's value should always include the following special substrings:

- `{1}` - this substring will be replaced with the URL of the package to be installed. This value is typically passed as the value of the installation agent's `-r` command line option. The URL may contain space characters, and so should be quoted appropriately in the command line.
- `{2}` - this substring will be replaced with any installation agent command line options that the selection agent determines may be needed to install the package. This value should not be quoted in the command line.

Values / range:	A valid command line string containing the literal substrings `{1}` and `{2}`.
Default value:	`ndlaunch -r "{1}" -o SaveAllUserSymbols=False {2}`
Example value:	To install packages with full user interaction: `ndlaunch -r "{1}" -o SaveAllUserSymbols=False {2} -o UserInteractionLevel=Full`

Note:
`{2}` will normally expand to specify a default `UserInteractionLevel` determined by the selection agent, therefore in this example the option to override this setting on the command line appears *after* the `{2}` substring.

Command Line

Tool:	Package selection agent
Example:	`ndselect -o ApplicationInstallCommand="ndlaunch -r ""{1}"" -o SaveAllUserSymbols=False {2} -o UserInteractionLevel=Full"`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Selector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Selector\CurrentVersion`

ApplicationUninstallCommand

Command line | Registry

Specifies a template command line to be used to uninstall an application package through the RayManageSofti package selection agent. The selection agent uses the value from this preference to build command lines that are executed in order to uninstall packages selected by a user.

This preference's value should always include the following special substrings:

- `{1}` - this substring will be replaced with the package name of the package to be uninstalled. This value is typically passed as the value of the installation agent's -d command line option. This value may contain space characters, and so should be quoted appropriately in the command line.
- `{2}` - this substring will be replaced with any installation agent command line options that the selection agent determines may be needed to uninstall the package. This value should not be quoted in the command line.

Values / range:	A valid command line string containing the literal substrings `{1}` and `{2}`.
Default value:	`ndlaunch -d "{1}" -o SaveAllUserSymbols=False {2}`
Example value:	To install packages with full user interaction: `ndlaunch -d "{1}" -o SaveAllUserSymbols=False {2} -o UserInteractionLevel=Full`

Note:
`{2}` will normally expand to specify a default `UserInteractionLevel` determined by the selection agent, therefore in this example the option to override this setting on the command line appears *after* the `{2}` substring.

Command Line

Tool:	Package selection agent
Example:	`ndselect -o ApplicationUninstallCommand="ndlaunch -d ""{1}"" -o SaveAllUserSymbols=False {2} -o UserInteractionLevel=Full"`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Selector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Selector\CurrentVersion`

ApplicationVerifyCommand

Command line | Registry

Specifies a template command line to be used to verify/repair an application package through the RayManageSofti package selection agent. The selection agent uses the value from this preference to build command lines that are executed in order to verify/repair packages selected by a user.

This preference's value should always include the following special substrings:

- `{1}` - this substring will be replaced with the package name of the package to be verified. This value is typically passed as the value of the installation agent's -a command line option. This value may contain space characters, and so should be quoted appropriately in the command line.
- `{2}` - this substring will be replaced with any installation agent command line options that the selection agent determines may be needed to uninstall the package. This value should not be quoted in the command line.

Values / range:	A valid command line string containing the literal substrings `{1}` and `{2}`.
Default value:	`ndlaunch -a "{1}" -o SaveAllUserSymbols=False -o MsiRepair=True -o CachedVersion=True -o SelfHeal=True -o CheckRegistry=True -o NoExec=True {2}`
Example value:	To require applications to be verified against their source from an appropriate distribution location and to not self-heal registry settings: `ndlaunch -a "{1}" -o SaveAllUserSymbols=False -o MsiRepair=True-o CheckRegistry=False -o NoExec=True {2}`

Note:
`{2}` will normally expand to specify a default `UserInteractionLevel` determined by the selection agent, therefore in this example the option to override this setting on the command line appears *after* the `{2}` substring.

Command Line

Tool:	Package selection agent
Example:	`ndselect -o ApplicationVerifyCommand="ndlaunch -a ""{1}"" -o SaveAllUserSymbols=False -o MsiRepair=True -o CheckRegistry=False -o NoExec=True {2}"`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Selector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Selector\CurrentVersion`

ApplyLocalPolicy

Registry

Specifies whether to use a locally cached copy of policy if a new policy cannot be generated (if client-side policy merging is in operation) or downloaded (if server-side policy merging is in operation).

If set to True, a locally cached copy of policy will be used if a version cannot be generated or downloaded. If set to False, no locally cached copy will be used (which means attempted application of policy will fail if policy cannot be generated or downloaded).

Values / range:	Boolean
Default value:	`False`

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Policy Client\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Policy Client\CurrentVersion`

ApplyPolicy

Command line

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It is used in conjunction with **DownloadPolicy** to distinguish between two types of event: *Apply a Deployment Manager policy* and *Update policy and package definitions in peer cache*. (For details about these event types, refer to the *Scheduling* chapter of the *Software Deployment Guide*.)

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o DownloadPolicy=False`

ApplyPolicyIfLoggedOn

Command line | Registry

This preference applies for Windows devices only.

Specifies whether RayManageSofti should apply computer policy when a user is logged on. If set to `True` (default), computer policy will be applied when scheduled, whether or not any user is logged on.

If set to `False`, computer policy will not be applied if a user is logged on at the scheduled time. RayManageSofti will attempt to apply computer policy next time it is scheduled.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Scheduling agent
Example:	`-o ApplyPolicyIfLoggedOn=False`

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Schedule Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Schedule Agent\CurrentVersion`

AskAboutDependencies

Command line | Registry | Project variable

When set to `True`, RayManageSofti asks the end-user before installing a prerequisite package.
When set to `False`, RayManageSofti installs the prerequisite package without prompting the end-user.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o AskAboutDependencies=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`AskAboutDependencies`
Reference as:	`$(AskAboutDependencies)`

AskBeforeInstalling

Command line | Registry | Project variable

Only applicable when **UserInteractionLevel (installation agent)** is set to `Full`. When set to `True`, RayManageSofti asks the end-user before installing a package. When set to `False`, RayManageSofti installs packages without prompting the end-user.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o AskBeforeInstalling=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`AskBeforeInstalling`
Reference as:	`$(AskBeforeInstalling`

AutoDetectDC

Command line | Registry | Project variable

Applies only for client-side policy merging.

When set to `True`, RayManageSofti allows Windows to automatically determine which domain controller RayManageSofti will connect to in order to apply policy.
When set to `False`, RayManageSofti will apply policy using a domain controller in the same site as the managed device. If no domain controller is available, the policy merge will fail.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o AutoDetectDC=False`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common`
Computer preference:	`[Registry]\ManageSoft\Common`

Project Variable

Define as:	`AutoDetectDC`
Reference as:	`$(AutoDetectDC)`

AutoPromptOnInstallCompletion

Command line | Registry | Project variable

Only applicable when **UserInteractionLevel (installation agent)** is set to `Auto`.
When set to `True`, RayManageSofti informs the end-user that the package installation has been completed.
When set to `False`, RayManageSofti does not inform the end-user.

Values / range:	Boolean
Default value:	No default in registry; default behavior is `False`

Command Line

Tool:	Installation agent
Example:	`-o AutoPromptOnInstallCompletion=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`AutoPromptOnInstallCompletion`
Reference as:	`$(AutoPromptOnInstallCompletion)`

AutoPromptOnUnInstallCompletion

Command line | Registry | Project variable

Only applicable when **UserInteractionLevel (installation agent)** is set to `Auto`.
When set to `True`, RayManageSofti informs the end-user when the un-installation of a package has completed.
When set to `False`, RayManageSofti does not inform the end-user.

Values / range:	Boolean
Default value:	No default in registry; default behavior is `False`

Command Line

Tool:	Installation agent
Example:	`-o AutoPromptOnUnInstallCompletion=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`AutoPromptOnUnInstallCompletion`
Reference as:	`$(AutoPromptOnUnInstallCompletion)`

AutoRedundancy

Command line | Registry | Project variable

When set to `True`, RayManageSofti removes redundant package components (files, registry settings and file preferences) when processing an upgrade or downgrade. The installation agent automatically determines which files are redundant (no longer referenced).

When set to `False`, RayManageSofti will:

- Uninstall the old package before re-installing the new package for downgrades
- Revert/remove redundant registry and preference file edits on upgrades

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o AutoRedundancy=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`AutoRedundancy`
Reference as:	`$(AutoRedundancy)`

BaseURL

Project variable

The URL of the distribution location from which the current application can be retrieved.

Values / range:	A valid URL
Default value:	Usually the folder in which the OSD/NDC files are located.
Example value:	`file://serverName/share/distributionLocation/Packages/Company/Product/`

Project Variable

Define as:	Predefined by RayManageSofti.
Reference as:	`$(BaseURL)`

BlockedApplications

Security Manager only
Command line | Registry

The list of applications that are to be prevented from running. Filenames must be separated by semi-colons (;).

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o AllowBytelevel=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`AllowByteLevel`
Reference as:	`$(AllowByteLevel)`

BlockedApplicationIncludeLocalAdmin

Security Manager only
Command line | Registry

If `True`, users with administrator privileges are also prevented from accessing blocked applications on managed devices.
Else if `False`, local administrator users can access blocked applications.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Security agent
Example:	`-o BlockedApplicationIncludeLocalAdmin=True`

Registry

Installed by:	Creation of security settings package (managed device settings package)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

BlockedApplicationMessage

Security Manager only
Command line | Registry

The message to display to end-users when an application is prevented from running.

Values / range:	String
Default value:	None
Example value:	Running this application contravenes corporate policy.

Command Line

Tool:	Security agent
Example:	`-o BlockedApplicationMessage="Running this application contravenes corporate policy."`

Registry

Installed by:	Creation of security settings package (managed device settings package)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

BlockedApplicationNotify

Security Manager only
Command line | Registry

Specifies if end-users are to be notified when an application is prevented from running.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Security agent
Example:	`-o BlockedApplicationNotify=True`

Registry

Installed by:	Creation of security settings package (managed device settings package)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

BootstrappedPolicy

Command line | Registry

When policies are attached to Active Directory domains, the policy to be applied on a managed device is the result of the policy merge. For policies attached to the RayManageSofti domain, a specific policy file, whose name and location are specified in this preference, is assigned to each managed device.
For more details about the differences between policies attached to Active Directory domains or the RayManageSofti domain, refer to the *Deployment policies* chapter of *RMS Software Deployment*.

Values / range:	String
Default value:	None

Command Line

Tool:	Policy agent
Example:	`-o BootstrappedPolicy="ManageSoftDL\Policy\Marketing.npl"`

Registry

Installed by:	RayManageSofti policy configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft\Policy Client\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Policy Client\CurrentVersion`

BrandARP

Command line

Provides the ability to exclude the name “RayManageSofti” from **Add/Remove Program** entries for installed applications.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o BrandARP=True`

CacheDir

Command line | Registry

This preference specifies the location of the peer cache on the managed device. Files are downloaded to this location, and shared from this location with peer managed devices.

This cache is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is True). For more information see *AllowPeerToPeer*.

If peer-to-peer file sharing is enabled, the installation agent downloads package files to the **CacheDirectory** from the **CacheDir** location instead of the nearest distribution location, if the files are available in the **CacheDir**.

Values / range:	String, any location on a local hard drive
Default value:	`$(CommonAppDataFolder)\ManageSoftCorp\ManageSoft\PeerCache`
Example value:	`C:\ManageSoft\SharedFiles`

Command Line

Tool:	Peer download agent
Example:	`-debug -o CacheDir="C:\ManageSoft\MyPeerCache"`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Downloader`
Computer preference:	`[Registry]\ManageSoft\Downloader`

CatalogName

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the name of the peer downloads file, which records files required by the managed device to complete package installation. The peer downloads file is stored in the parent directory of the peer cache (specified by **CacheDir**).
For more information see *AllowPeerToPeer*.

Values / range:	String
Default value:	`catalog.ctx`
Example value:	`peercachefiles.ctx`

Command Line

Tool:	Peer download agent
Example:	`-debug -o CatalogName="mycatalog.ctx"`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Downloader`
Computer preference:	`[Registry]\ManageSoft\Downloader`

CheckCatalogDigest

Command line | Registry | Project variable

Determines whether RayManageSofti performs a check on the package-level MD5 digest during the self-healing process and security checking.

When set to `True`, RayManageSofti verifies the correctness of the implementation archive (`.ndc` file) by checking the MD5 digest calculated for the downloaded file against the MD5 digest stored in the package catalog.

When set to `False`, RayManageSofti does not check the MD5 digest of the implementation archive during self-healing or security checking.

A related setting for user preferences in the registry will override the machine settings unless the machine settings are locked. See *Fixing managed device preferences*.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o CheckCatalogDigest=True`

Registry

Installed by:	First run of installation agent
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`CheckCatalogDigest`
Reference as:	`$(CheckCatalogDigest)`

CheckCertificateRevocation

Command line | Registry

When downloading or uploading data from an HTTPS web server, a web server certificate is applied to the data being transferred.
When receiving web server certificates from servers, RayManageSofti checks the CA (certification authority) server to ensure that the certificates are not on the CRL (certificate revocation list). If RayManageSofti cannot check the CRL (for example, the CA server is firewalled and cannot be contacted), the system can stall. To avoid this, you can use the `CheckCertificateRevocation` preference to prevent RayManageSofti from performing the CRL check.

You can set this as a common registry entry, so that the same behavior occurs across all RayManageSofti components respectively agents, and you can override the common behavior by setting an overriding registry entry for any component.
By default, this preference is set to check the CRL for all components

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	All agents
Example:	`-o CheckCertificateRevocation="False"`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common` or `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\<Agent>\CurrentVersion` where `<Agent>` is the registry key for an individual RayManageSofti agent
Computer preference:	`[Registry]\ManageSoft\Common` or `[Registry]\ManageSoft\<Agent>\CurrentVersion` where <Agent> is the registry key for an individual RayManageSofti agent

CheckFileDigest

Command line | Registry

Determines whether RayManageSofti performs a check on file-level MD5 digests during the self-healing process.
If `True`, RayManageSofti verifies the correctness of the implementation archive (`.ndc` file) by checking the MD5 digest of the installed file against the relevant MD5 digest stored in the RayManageSofti cache on the managed device. This test is in addition to the basic checking of file presence and file size.

WARNING:
Checking MD5 digests can moderately increase the time required to perform an application repair (*self-heal*) activity.

If `False`, RayManageSofti does not check MD5 digests during self-healing.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o CheckFileDigest=True`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

CheckpointSeconds

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the frequency (in seconds) with which the peer downloads file is written to disk.
For information see *AllowPeerToPeer*.

Values / range:	Integer between 10 - 3600
Default value:	`300`

Command Line

Tool:	Peer download agent
Example:	`-o CheckpointSeconds="600"`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

CheckRegistry (or Reg on the Command Line)

Command line | Registry | Project variable

Determines if RayManageSofti performs self-healing on registry keys and preference files.
When set to `True`, the registry keys installed by the application are self-healed.
When set to `False`, registry keys and preference files are not self-healed.

WARNING:
This can significantly increase the time required to perform an application repair (*self-heal*) activity.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o reg=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`CheckRegistry`
Reference as:	`$(CheckRegistry)`

CmdLineOverrides

Registry

When set to `True`, options set on the command line override fixed preferences (in the registry or in a network preference file).
When set to `False`, options set on the command line do not override fixed preferences.

Values / range:	Boolean
Default value:	`False`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

CommonProgramMenuFolder

Project variable

The path to **Start** menu program folders and shortcuts for `[ALL USERS]`.

Values / range:	Local directory path. Read-only preference.
Default value:	The default installation of Windows uses the `[ALL USERS PROFILE]\Start Menu\Programs` folder. For Windows NT, ALL USERS PROFILE defaults to `C:\Winnt\Profiles\All Users`. For Windows 2000/XP, ALL USERS PROFILE defaults to `C:\Documents and Settings\All Users`. For Windows Vista and later, ALL USERS PROFILE defaults to `C:\Documents and Settings\Public`.
Example value:	`C:\Documents and Settings\All Users\Start Menu\Programs`

Project Variable

Define as:	Predefined within Windows.
Reference as:	`$(CommonProgramMenuFolder)`

Compress (Application Usage Agent)

Command line | Registry

Specifies whether or not application usage data files are compressed before being uploaded to the administration server for inclusion in the database.

When set to `True`, RayManageSofti compresses the application usage data file for upload.

When set to `False`, RayManageSofti leaves the application usage data file uncompressed.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Application usage agent
Example:	`-o Compress=False`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device (computer preference)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

Compress (Inventory Agent)

Command line | Registry

Specifies whether or not application usage data files are compressed before being uploaded to the administration server for inclusion in the database.

When set to `True`, RayManageSofti compresses the inventory file for upload.

When set to `False`, RayManageSofti leaves the inventory file uncompressed.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Inventory agent
Example:	`-o Compress=False`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

Compress (Security Service)

Security Manager only
Registry

Specifies whether or not application usage data files are compressed before being uploaded to the administration server for inclusion in the database.
When set to `True`, RayManageSofti compresses the security service event log file for upload.
When set to `False`, RayManageSofti leaves the security service event log file uncompressed.

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Service\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Service\CurrentVersion`

ComputerDomain

Registry | Project variable

Domain name of the managed device.

Values / range:	The canonical domain name of the managed device. Read-only preference.
Default value:	The default is the value retrieved from Windows.
Example value:	`mycompany.com`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common`
Computer preference:	`[Registry]\ManageSoft\Common`

Project Variable

Define as:	Pre-defined on operating systems supporting Active Directory, such as Windows 2000 and Windows XP.
Reference as:	`$(ComputerDomain)`

ComputerDNSName

Project variable

The DNS name assigned to the managed device.

Values / range:	The fully-qualified domain name of the managed device.
Default value:	`$(ComputerName).$(ComputerDomain)`
Example value:	`mycomputer.mycompany.com`

Project Variable

Define as:	Pre-defined on operating systems supporting Active Directory, such as Windows 2000 and Windows XP.
Reference as:	`$(ComputerDNSName)`

ConfigFile

Command line | Registry

Identifies the system copy of the configuration file used by the package selection agent on managed devices. This is not a pointer to the end-user's copy of the configuration file: the latter is (re)created automatically from this system copy, but only if the system copy has a different version number.
The configuration file determines both the skin (user interface design) and localization of the selection agent.

Be aware:
Setting the value on the command line will cause it to be written to the `HKEY_CURRENT_USER` hive of the registry in order to start the same `skin/locale` next time the selection agent is used.

Setting the value on the command line to a non-empty but invalid value will clear the registry setting again.

Values / range:	Any path to a valid RayManageSofti configuration file
Default value:	`$(ConfigFileDefault)`
Example value:	`C:\Program Files\ManageSoft\Selector\Skins\MySkin\DE\config.xml`

Command Line

Tool:	Package selection agent
Example:	`-o ConfigFile="C:\Program Files\ManageSoft\Selector\Skins\Default\DE\default.xml"`

Registry

Installed by:	Manual configuration, or by the selection agent storing a value set on the command line.
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Selector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Selector\CurrentVersion`

ConfigFileDefault

Command line | Registry

Identifies the system copy of the default configuration file for the selection agent on managed devices. This configuration file will be used whenever the selection agent cannot find a configuration file in paths defined by the end-user's locale setting or by the **ConfigFile** preference.

Be aware:
Be aware: Setting the value on the command line does *not* cause it to be written to the registry. Be cautious about replacing the default value with a literal, as this will negate the effects of other preference settings shown in the default value.

Values / range:	A.ny path and filename of a valid RayManageSofti configuration file
Default value:	`$(SkinsDirectory)\Default\$(Locale)\$(ConfigName)`
Example value:	`C:\Program Files\ManageSoft\Selector\Skins\ MySkin\DE\config.xml`

Command Line

Tool:	Package selection agent
Example:	`-o ConfigFile="C:\Program Files\ManageSoft\ Selector\Skins\Default\DE\default.xml"`

Registry

Installed by:	RayManageSofti internals, or manual configuration.
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Selector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Selector\CurrentVersion`

ConfirmSharedFileRemoval

Command line | Registry | Project variable

Only applicable when **UserInteractionLevel** is set to `Full`.
When set to `True`, RayManageSofti displays a dialog when removing a file that has a reference-count value greater than zero.
When set to `False`, RayManageSofti does not display a dialog.
Domain name of the managed device.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	-o ConfirmSharedFileRemoval=True

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer reference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`ConfirmSharedFileRemoval`
Reference as:	`$(ConfirmSharedFileRemoval)`

ConnectionAttempts

Command line | Registry | Project variable

When RayManageSofti is trying to connect to a particular distribution location, this preference specifies how many times it will accept a "no connection is available" error before discarding that distribution location from the list of available locations.

The "no connection is available" condition is only detected when using file-share as the connection protocol, and occurs when the number of active connections to a file share reaches the maximum allowed. This preference is not relevant for distribution locations that are accessed via HTTP, HTTPS, or FTP.

Also see *NetworkRetries*.

Values / range:	Numeric
Default value:	2

Command Line

Tool:	Installation agent
Example:	`ndlaunch -o ConnectionAttempts=100`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`ConnectionAttempts`
Reference as:	`$(ConnectionAttempts)`

Difference

Command line | Registry

When set to True, the inventory agent will perform differential inventories, rather than full inventories until the maximum number of differential inventories has been performed (defined in *GenerationMax*). RayManageSofti will then perform a full inventory, and re-start differential inventories. See the *IncrementalDiff* preference for details about the type of differential inventory that will be created.

When set to False, the inventory agent will always perform full inventories.

Be aware:
By default, the inventory agent gathers full inventories, and it is recommended that you do *not* alter this. Full inventories process is considerably faster than differential inventories.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Inventory agent
Example:	`-o Difference=False`

Registry

Installed by:	ManageSoft internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

Disabled (Application Usage Agent)

Command line | Registry

Specifies whether the application usage agent is inactive on this managed device (same name is used by the schedule agent).

When set to `True`, RayManageSofti does *not* record application usage data.
When set to `False`, RayManageSofti records application usage data.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Application usage agent
Example:	`-o Disabled=False`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device (computer preference)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

DisablePackageFiltering

Registry

When set to True, **DisablePackageFiltering** improves the performance of client-side policy merging in cases where package-level filtering is not used.

When performing policy merge, RayManageSofti loads each policy (not necessarily each package) and determines its package-level filtering requirements. This setting instructs RayManageSofti to skip that process by assuming there is no package-level filtering applied to the packages.

Be aware:
Only set this option if your organization does not and will not use package-level filtering.

Values / range:	Boolean
Default value:	No default in registry; default behavior is `False`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference; must be manually set in `mgssetup.ini`)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Policy Client\CurrentVersion`

DisablePeriod

Command line | Registry

Specify the number of seconds for which Deployment Manager user schedules remain disabled. (The scheduling agent user interface is only available on Windows devices.)

The default value is 3600 seconds (one hour). When set to 3600, RayManageSofti schedules are automatically re-enabled after one hour.

Values / range:	Integer between 0 -2147493647
Default value:	`3600`

Command Line

Tool:	Scheduling agent
Example:	-o DisablePeriod=600

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Schedule Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Schedule Agent\CurrentVersion`

DiskAveragingTime

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
This preference is used in conjunction with **DiskMaxRate** to limit disk I/O operations activities associated with updating the peer cache and sharing files with peer managed devices. These operations include:

- Downloads from distribution servers
- Transfers to and from peers
- Internal copying of files to the peer cache when duplicate versions of a file are requested

DiskAveragingTime specifies the averaging period (in minutes) used to smooth the estimate of transfers to and from disk. See *DiskMaxRate* for details about how these preferences are used together.
Increasing the value of this preference means that the estimate takes longer to change as the actual transfer rate changes. In normal use, you will not need to change the value of this preference.
RayManageSofti retrieves the value for this preference from the registry every five seconds. You do not need to restart RayManageSofti on managed devices after changing the value of this preference.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 1 - 60
Default value:	`2`

Command Line

Tool:	Peer download agent
Example:	`-debug -o DiskAveragingTime=10`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

DiskMaxRate

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the maximum allowable rate (in bytes per second) for all disk I/O activities associated with updating the peer cache and sharing files with peer managed devices. **DiskMaxRate** is used in conjunction with **DiskAveragingTime** to limit disk I/O operations.
RayManageSofti calculates the sum of disk reads and writes that have occurred to update the peer cache and share files with peer managed devices:

- Downloads from distribution servers
- Transfers to and from peers
- Internal copying of files to the peer cache when duplicate versions of a file are requested

It does *not* include the very small amounts of disk I/O associated with saving the peer downloads file.
Since disk transfers occur in blocks, not as a continuous stream, RayManageSofti smooths out the variation in transfer rates using the **DiskAveragingTime** and a simple exponential decay algorithm. The result is an estimate of the disk transfer rate. Transfer rates will be decreased if the estimated rate exceeds the specified DiskMaxRate, and increased if they are below the specified **DiskMaxRate**. (Transfers can creep up to the **DiskMaxRate**, but will drop back rapidly when the estimated rate is greater than **DiskMaxRate**.)
RayManageSofti retrieves the value for this preference from the registry every five seconds. You do not need to restart RayManageSofti on managed devices after changing the value of this preference.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 1024 - 134217728
Default value:	`134217728`

Command Line

Tool:	Peer download agent
Example:	`-debug -o DiskMaxRate=2048`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

DiskReservedKB

Registry | Project variable

The amount of disk space to reserve on each drive. When checking whether there is enough disk space to install an application, RayManageSofti attempts to leave this amount free on each drive to which it is installing files.

If there will be less than this amount of free space after application installation:

- If **UserInteractionLevel** is set to `Full`, RayManageSofti prompts the end-user to determine whether or not to continue with the installation. If the end-user elects not to proceed with the installation (the default choice), installation fails.
- If **UserInteractionLevel** is set to any *other value*, installation fails.

Values / range:	Integer between 1 - 1000000000
Default value:	`1024`

Command Line

Tool:	Peer download agent
Example:	`-debug -o DiskMaxRate=2048`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

DisplayAllAuthcode

Command line | Registry | Project variable

Only applicable when **UserInteractionLevel** is set to `Full`.
Determines the behavior when RayManageSofti performs an Authenticode check and encounters an invalid signature.

When set to `True`, the installation agent operation fails without alerting the user.
When set to `False`, RayManageSofti displays a dialog to allow the user to choose whether to continue with the installation agent operation.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o DisplayAllAuthCode=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`DisplayAllAuthCode`
Reference as:	`$(DisplayAllAuthCode)`

EnablePolicyFailOver

Command line | Registry

This preference only applies if client-side policy is enabled (`PolicySource=Client`).
When set to True, if a client-side policy file cannot be accessed, RayManageSofti instead uses a server-side policy file if one is available. The server-side policy is applied as normal with one exception: no applications will be uninstalled, even if they are no longer in policy and marked to be removed when no longer in policy.

Using server-side policy is a one-off event. The **PolicySource** setting remains unchanged.
When set to False, no policy will be applied if a client-side policy file is not available.

Values / range:	Boolean
Default value:	`False`
Supported from	`Release 8.0`

Command Line

Tool:	Installation agent
Example:	`-o EnablePolicyFailOver=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

EnableSessionLogging

Registry

Determines whether the following events are recorded in the application usage log:

- A session starts or ends for any application identified by the native package format (MSI, RPM or PKG), the Manual Mapper registry, Add/Remove programs (Windows devices only), or RayManageSofti packages (Windows devices only)
- A session starts or ends for applications not identified by the above mechanisms
- A session ends for an identified application if the session is shorter than the Minimum run time in seconds (See *MinRunTime*).

When set to `True`, these events are logged. When set to `False`, these events are not logged.

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Usage Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

EventNetType

Command line | Registry

This preference applies only on Windows devices.
Events that are set to execute only when the network is available. Use this entry to identify the type of network to look for from one of the following settings:

- Local area network (LAN)
- Wide area network (WAN)
- Either a LAN or a WAN

For example: If an event has been configured to **Only run this event if a network connection is available**, and `EventNetType = 1`, at the time the event is scheduled to run, the scheduling agent will check for a LAN connection before running the event, and skip the event if the connection is not present at that time.

Values / range:	1, 2, 3
Default value:	3

Command Line

Tool:	Scheduling agent
Example:	`-o EventNetType=1`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Schedule Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Schedule Agent\CurrentVersion`

ExcludeDirectory

Command line | Registry

Excludes a specified folder from inventory. If **Recurse** is `True`, then all subfolders are also excluded. This preference can accept multiple values.
If a folder is identified in both the **ExcludeDirectory** and **IncludeDirectory** preferences, it is excluded. Exclusions always override inclusions.

For more information, see *How RayManageSofti uses inventory inclusion and exclusion preferences.*

Values / range:	Valid folders
Default value:	`{empty}`
Example value	`$(WinDirectory)`

Command Line

Tool:	Inventory agent
Example:	`-o ExcludeDirectory=C:\Temp`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

ExcludedMGSs

Command line | Registry

This preference can accept multiple values.
Allows you to exclude specific RayManageSofti-installed executables from your application usage process list. Once excluded, no application usage data will be recorded from these applications.
For example, you may not wish to record the use of Adobe Acrobat and Microsoft Office.

Values / range:	Semi-colon separated list
Default value:	`{empty}`
Example value	acrobat;office

Command Line

Tool:	Application usage agent
Example:	`-o ExcludedMGSs="acrobat;office"`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

ExcludedMSIs

Command line | Registry

This preference is supported only on Windows devices. This preference can accept multiple values.
Excludes specified native package format (MSI) applications from having application usage data recorded. Once excluded, no application usage data will be recorded from these applications.

Values / range:	Valid application names
Default value:	RayManageSofti for managed devices product code GUID
Example value	`{00000409-78E1-11D2-B60F-006097C998E7}`

Command Line

Tool:	Application usage agent
Example:	`-o ExcludedMSIs="{00000409-78E1-11D2-B60F-006097C998E7}"`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

ExcludeExtension

Command line | Registry

For files within a folder included in inventory, RayManageSofti excludes files with the specified extension from inventory, or excludes all files if set to the value * **(asterisk)**. This preference can accept multiple values.

For more information, see *How RayManageSofti uses inventory inclusion and exclusion preferences*.

Values / range:	File extensions (no period required)
Default value:	`{empty}`
Example value	`DLL`

Command Line

Tool:	Inventory agent
Example:	`-o ExcludeExtension=dll`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

ExcludeFile

Command line | Registry

For files within a folder included in inventory, RayManageSofti excludes a specific file from inventory. This preference can accept multiple values.
For more information, see *How RayManageSofti uses inventory inclusion and exclusion preferences.*

Values / range:	Valid file names
Default value:	`{empty}`
Example value	`myfile.txt`

Command Line

Tool:	Inventory agent
Example:	`-o ExcludeFile=myfile.txt`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

ExcludeMD5

Command line | Registry

For files within a folder included in inventory, RayManageSofti performs an MD5 checksum, and excludes any files from the inventory that have an MD5 value equal to the value stored in this preference. This preference can accept multiple values.

For more information, see *How RayManageSofti uses inventory inclusion and exclusion preferences*.

Values / range:	Valid MD5 values
Default value:	`{empty}`
Example value	`7d9d2440656fdb3645f6734465678c6`

Command Line

Tool:	Inventory agent
Example:	`-o ExcludeMD5=7d9d2440656fdb3645f6734465678c60`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

ForceReboot

Command line | Registry

RebootIfRequired is responsible for end-user prompts about a required reboot. If **ForceReboot** is `True` and the default response to the prompt dialog is to reboot, the prompts are modified to prevent end-users from vetoing a reboot.

This preference determines the type of reboot (forced, or polite) initiated by RayManageSofti after that prompting:

- When set to `True`, RayManageSofti performs a *forced reboot*. This suppresses any user interaction required to close down other applications that may be running, which means that unsaved work may be lost.
- When set to `False`, RayManageSofti performs a *polite reboot*. This offers end-users the opportunity to save work and close down other applications before continuing with the reboot.

If the desktop is locked, the installation and adoption agents use **ForceRebootIfLocked** instead of **ForceReboot**.
For details about how this preference works in combination with other installation preferences to determine the appropriate reboot action, see *Reboot options*.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent, adoption agent
Example:	`-o ForceReboot=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

ForceRebootIfLocked

Command line | Registry

AllowRebootIfLocked is responsible for end-user prompts about a required reboot if the desktop is locked. If **ForceRebootIfLocked** is `True` and the default response to the prompt dialog is to reboot, the prompts are modified to prevent end-users from vetoing a reboot.

ForceRebootIfLocked determines the type of reboot (forced or polite) initiated by RayManageSofti after that prompting:

- When set to `True`, RayManageSofti performs a *forced reboot*. This suppresses any user interaction required to close down other applications that may be running, which means that unsaved work may be lost.
- When set to `False`, RayManageSofti performs a *polite reboot*. This offers end-users the opportunity to save work and close down other applications before continuing with the reboot.

For details about how this preference works in combination with other installation preferences to determine the appropriate reboot action, see *Reboot options*.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent, adoption agent
Example:	`-o ForceRebootIfLocked=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

ForceSharedFileRemove

Command line | Registry | Project variable

Allows forced deletion of redundant files in the Windows system folder.
By default, when files in a package are marked as redundant, they are deleted on the managed device after installation. However, because files within the Windows system folder may be shared with other packages, they are not deleted. **ForceSharedFileRemove** allows deletion of these files.

If `True`, redundant files within the Windows system folder are deleted when other redundant files in a package are deleted. If `False`, files in the Windows system folder are not deleted.
This setting also determines the default response to keeping shared files during uninstallation of packages. If `True`, the default response is to not keep shared files. If `False`, the default response is to keep shared files.

Be aware:
To control behavior for a single package, this must be set as a project variable in the package. (Registry entries stored in a package are only set after installation of that package.)

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o ForceSharedFileRemove=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`ForceSharedFileRemove`
Reference as:	`$(ForceSharedFileRemove)`

ForceValidSignature

Command line | Registry | Project variable

When testing whether a package should be installed, the installation agent uses Authenticode technology to validate a digital signature. If the signature is invalid, the installation agent examines this setting.

When set to `True`, the installation stops and the end-user is informed of the failure. When set to `False`, RayManageSofti displays a Microsoft dialog that advises the failure but asks the end-user if they want to proceed anyway. This is not recommended practice.

A related setting for user preferences in the registry will override the machine settings unless the machine settings are locked. See *Fixing managed device preferences*.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o ForceValidSignature=True`

Registry

Installed by:	First run of installation agent
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`ForceValidSignature`
Reference as:	`$(ForceValidSignature`

GCDiskSlice

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`)
It specifies the percentage of **DiskMaxRate** that can be used for checking the consistency of the peer cache, and cleaning up files that are no longer required in it.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 1 and 50
Default value:	`2`

Command Line

Tool:	Peer download agent
Example:	`-debug -o GCDiskSlice="24"`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

GCMaxInterval

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the maximum number of minutes the peer download agent should pause between examining files in the peer cache for cleanup.

This preference is used in conjunction with **GCMinInterval** and **GCPeriod**. **GCMaxInterval** takes precedence over **GCPeriod**. The peer download agent will not increase the time interval between examining files in the peer cache, even if that means it will examine all files in the peer cache more than once within the period specified by **GCPeriod**.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 10 and 1440
Default value:	`60`

Command Line

Tool:	Peer download agent
Example:	`-debug -o GCMaxInterval="720"`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

GCMinInterval

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the minimum number of minutes the peer download agent should pause between examining files in the peer cache for cleanup.

This preference is used in conjunction with **GCMaxInterval** and **GCPeriod**. **GCMinInterval** takes precedence over **GCPeriod**. The peer download agent will not examine files more frequently than specified by this preference, even if that means it will not examine all files in the peer cache within the period specified by **GCPeriod**.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 10 and 600
Default value:	`60`

Command Line

Tool:	Peer download agent
Example:	`-debug -o GCMaxInterval="20"`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

GCPeriod

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the number of hours across which the RayManageSofti peer download agent attempts to visit all files in the peer cache as part of cleaning up operations. The default is to attempt to examine and clean up unused files from the peer cache across a period of 12 hours. This minimizes impact on the end-user of the managed device.

Once started, cleanup operations continue while the managed device is on. **GCPeriod** is used to help determine the time interval at which files in the peer cache are examined by cleanup operations. It is used in conjunction with **GCMinInterval** (which specifies the minimum number of minutes the peer download agent should pause between examining files in the peer cache) and **GCMaxInterval** (which specifies the maximum number of minutes the peer download agent should pause between examining files in the peer cache).

Both **GCMaxInterval** and **GCMinInterval** take precedence over **GCPeriod**. Files will not be examined more frequently than at the intervals allowed by **GCMinInterval** and **GCMaxInterval**, even if it means that not all files in the peer cache will be examined in the period specified by **GCPeriod**.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 1 and 744
Default value:	`12`

Command Line

Tool:	Peer download agent
Example:	`-o GCMaxInterval="24"`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

GenerateMD5

Command line | Registry

Specifies whether or not to calculate the MD5 digest of any file being tracked by the inventory agent and include it with stored inventory data. MD5 digests, if included in inventory, are used as a key to uniquely identify files. If an MD5 is not present, files are identified by date and file size. MD5 digests are more reliable for this purpose, but be aware that calculating MD5 digests will degrade performance where many files are being tracked.

This use of MD5 digests is unrelated to the comparison completed by the installation agent prior to downloading and installing files.

Values / range:	Boolean
Default value:	`False`
Supported from	Release 8.0

Command Line

Tool:	Inventory agent
Example:	`-o GenerateMD5=True`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

GenerationMax

Command line | Registry

Defines the number of differential inventories that may take place before a full inventory is performed. This only applies if differential inventories are being used (Difference is True).

Values / range:	Integer between 1 - 1000000000
Default value:	`9` (every `10th` inventory is a full inventory)

Command Line

Tool:	Inventory agent
Example:	`-o GenerationMax=5`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

GlobalConfigSource

Registry

Points the RayManageSofti installation agent to a URL or UNC path on the network that contains installation preferences. These preferences are stored in `.ini` file format.

Values / range:	Valid URL or UNC path
Default value:	No default
Example value	**UNC:** `\\server\share\path\network.ini` **URL:** `http://server/share/path/network.ini`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Hardware

Command line | Registry

Allows you to track hardware either using Windows Management Instrumentation (WMI) or native APIs. If WMI is available, it is used for tracking.

This preference is only effective when running in the machine context. To track hardware in the user context, use **UserHardware**.

When set to `True`, allows the tracking of hardware inventory. When set to `False`, does not track hardware inventory.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Inventory agent
Example:	`-o Hardware=False`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	N/A - use **UserHardware**
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

HideMachineUI

Command line | Registry

When set to `True`, RayManageSofti does not display a user interface when applying machine policy.

WARNING:
Obsolete: do not edit.Setting this entry to `False` could result in hidden dialog boxes *while* requiring user input.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Scheduling agent
Example:	`-o HideMachineUI=False`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Schedule Agent\CurrentVersion`

HighestPriority

Registry

Specifies the highest upload/download priority that can be assigned to a distribution server. The lower the number, the higher the priority.

When assigning priorities, RayManageSofti normalizes the calculated priority to fit within the range identified by **HighestPriority** and **LowestPriority**. The highest priority is commonly set to 1

Values / range:	Recommended 1-100 (but can extend from -231 to 23)
Default value:	No default in registry; default behavior uses 10.
Example value	`10`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \NetSelector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\NetSelector\CurrentVersion`

.

Http_proxy

Command line | Registry | Project variable

Proxy settings for the RayManageSofti installation agent.

Values / range:	Any valid URL
Default value:	Not to use a proxy.
Example value:	`tmnis.com;tmnis.com.de`

Command Line

Tool:	Installation agent
Example:	`-o http_proxy=tmnis.com;tmnis.com.de`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`http_proxy`
Reference as:	`$(http_proxy)`

IgnoreConnectionWindows

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
If set to True, settings of **ParentConnectionWindows** and **PeerConnectionWindows** are ignored. This means that managed devices can connect to distribution servers and peers at any time.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o IgnoreConnectionWindows=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Common\`

IncludeDirectory

Command line | Registry

Includes a specified folder from inventory. If **Recurse** is `True`, then all subfolders are also included. When the value of this entry is set to "\", it means "include all folders". This preference can accept multiple values.

If a folder is identified in both the **ExcludeDirectory** and **IncludeDirectory** preferences, it is excluded. Exclusions always override inclusions.

For more information, see *Peer download agent preferences set on the command line*.

Values / range:	Valid folder
Default value:	`{empty}`
Example value:	`C:\Program Files`

Command Line

Tool:	Inventory agent
Example:	`-o IncludeDirectory=C:\Temp`

Registry

Installed by:	Installation of ManageSoft on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

IncludeExtension

Command line | Registry

For files within a folder included in inventory, RayManageSofti includes files with the specified extension, or includes all files if this preference is set to the value *. By default, this includes `.exe` and `.dll`. This preference can accept multiple values.

For more information, see *Peer download agent preferences set on the command line.*

Values / range:	Any file extension
Default value:	`exe; dll`
Example value:	`bat`

Command Line

Tool:	Inventory agent
Example:	`-o IncludeExtension=exe`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

IncludeFile

Command line | Registry

For files within a folder included in inventory, RayManageSofti includes a specific file. This preference can accept multiple values.

For more information, see *Peer download agent preferences set on the command line.*

Values / range:	Valid file
Default value:	`{empty}`
Example value:	`xcopy.exe`

Command Line

Tool:	Inventory agent
Example:	`-o IncludeFile=myfile.txt`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

IncludeMachineInventory

Command line | Registry

If `True`, perform a computer inventory including hardware and all user packages.

Values / range:	Boolean
Default value:	`True` if running as **LocalSystem** or running a machine inventory (`-t Machine` on the command line)

Command Line

Tool:	Inventory agent
Example:	`-o IncludeMachineInventory="FALSE"`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

IncludeMD5

Command line | Registry

For files within a folder included in inventory, RayManageSofti includes a specific MD5 digest. This preference can accept multiple values.

For more information, see *Peer download agent preferences set on the command line.*

Values / range:	Valid MD5 values
Default value:	`{empty}`
Example value:	`7d9d2440656fdb3645f6734465678c60`

Command Line

Tool:	Inventory agent
Example:	`-o IncludeMD5=7d9d2440656fdb3645f6734465678c60`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

IncludePermissionsMask

Command line | Registry

This setting applies only to non-Windows platforms, and is ignored on Windows.
This preference specifies which files should be reported in inventory by the inventory agent. The value is an octal mask for file permissions, in the format used for the `chmod` command.

Only files which have permissions settings that give a true result in a logical `AND` with the mask will be reported.

Values / range:	Octal value in format used for `chmod`.
Default value:	`0111`
Example value:	`0777` This value will cause reporting of every file (not recommended for performance reasons).
Supported from Release	Supported from Release 8.1.

Command Line

Tool:	Inventory agent
Example:	`-o IncludePermissionsMask=0113`

Registry

Installed by:	RayManageSofti internals, managed device settings package or manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

IncludeRegistryKey

Command line | Registry

Set this preference to instruct the inventory agent to track the specified registry keys or values.
In order to collect all values under a specified key, the key path specified must end with a trailing backslash. If the path specified corresponds to a key (rather than a registry value) but does not end with a trailing backslash, only the (Default) value (if it is set) for the specified key will be collected.

For example:

- `HKLM\SOFTWARE\ManageSoft Corp\ManageSoft\` will track *all* values under the specified key
- `HKLM\SOFTWARE\ManageSoft Corp\ManageSoft` will only track the *default* values under the specified key. (Note that the *default* values are typically not set.)

When setting this preference, you can use:

- The * wildcard to replace a key or value
- The abbreviations HKLM, HKCU, HKCR, HKU, HKCC. These will be automatically expanded to appropriate values.

Values / range:	Valid registry key or value
Default value:	If no value is specified, RayManageSofti uses `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\`

Example value:	`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\`
	- tracks all registry entries under this key
	`HKEY_LOCAL_MACHINE\SOFTWARE\*\`
	- tracks all registry keys and values under
	`HKLM\SOFTWARE`. `HKLM\SOFTWARE\Microsoft\*`
	- tracks all values under
	`HKLM\SOFTWARE\Microsoft`.`HKEY_LOCAL_MACHINE\SOFTWARE\*\CurrentVersion\*\*`
	- illustrates the use of multiple wildcards.

Command Line

Tool:	Inventory agent
Example:	`-o IncludeRegistryKey="HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\"`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

IncludeUserInventory

Command line | Registry

If `TRUE`, perform a user inventory.

Values / range:	Boolean
Default value:	`TRUE` if running as user or running a user inventory (-t User on the command line)

Command Line

Tool:	Inventory agent
Example:	`-o IncludeUserInventory="FALSE"`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

IncrementalDiff

Command line | Registry

If differential inventory is in use (`Difference=True`) then this entry determines what differences the differential inventory will collect.

- If `True`, the differential inventory file will list differences from the last inventory file (which may be either differential or full).
- If `False`, the differential inventory file will list differences from the last full inventory file.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Inventory agent
Example:	`-o IncrementalDiff=True`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

InstallationStatusRefreshPeriod

Command line | Registry

Specifies how frequently (in seconds) RayManageSofti should recreate installation events for packages that are installed and packages flagged as not required.
This setting is useful if installation records have been removed on the administration server, as the re-created records on managed devices will repopulate the administration server's records.

Setting `InstallationStatusRefreshPeriod=0` has the same effect as setting `LogNotRequiredCheck=True`. **LogNotRequiredCheck** is deprecated. If `LogNotRequiredCheck=True`, **InstallationStatusRefreshPeriod** is ignored.

Values / range:	Integer between 0 and 31556926
Default value:	`604800` (1 week)

Command Line

Tool:	Policy agent, installation agent
Example:	`-o InstallationStatusRefreshPeriod=31556926`

Registry

Installed by:	Installation of ManageSoft on managed devices
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

InstallerARPModify

Command line | Registry | Project variable

When set to `True`, details of software installed using RayManageSofti external installer packages can be modified within the Microsoft **Add/Remove Programs** control panel applet.

When set to `False`, these details cannot be modified within **Add/Remove Programs**.
It is likely that this preference will need to be different for individual packages.

Be aware:
To control behavior for a single package, this must be set as a project variable in the package. (Registry entries stored in a package are only set after installation of that package.)

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o InstallerARPModify=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`InstallerARPModify`
Reference as:	`$(InstallerARPModify)`

InstallerARPRemove

Command line | Registry | Project variable

When set to `True`, software installed using RayManageSofti external installer packages can be uninstalled within the Microsoft **Add/Remove Programs** control panel applet.
When set to `False`, the software cannot be uninstalled within **Add/Remove Programs**.
It is likely that this preference will need to be different for individual packages.

Be aware:
To control behavior for a single package, this must be set as a project variable in the package. (Registry entries stored in a package are only set after installation of that package.)

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o InstallerARPRemove=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`InstallerARPRemove`
Reference as:	`$(InstallerARPRemove)`

Inventory

Registry

Instructs RayManageSofti to upload inventory files from the managed device to the specified server location.

Values / range:	Valid location
Default value:	`$(ServerLocation)/Inventories/$(MachineId).ndi`
Example value:	`$(ServerLocation)/Inventories/$(UserId) on $(MachineId) at $(DateTime) $(Generation)).ndi`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common\Rules`
Computer preference:	`[Registry]\ManageSoft\Common\Rules`

This preference is set during installation and should not be altered by end-users.

InventoryDirectory

Command Line

It is the general parameter that can be set as the custom directory for the storage of inventory data by the inventory agent.

Values / range:	Valid location
Default Value:	{Empty}

Command Line

Tool:	Inventory Agent
Example:	`-o InventoryDirectory=C: \Inventory`

In case the parameter is not set the agent takes the default value as the storage location for the inventory data. For further information refer to the `MachineInventoryDirectory`, `UserInventoryDirectory`, `MachineZeroTouchDirectory` and `UserZeroTouchDirectory` preferences listed in this chapter.

InventoryFile

Command line | Registry

Identifies the name of a local copy of the inventory file.
The name may consist of Windows properties that can be expanded to identify a value. For example, the default value **$(UserName)** on **$(MachineId).ndi** expands so that the name contains the account and machine ID related to the inventory run.

Values / range:	*.ndi
Default value:	`$(UserName) on $(MachineId).ndi`
Example value:	`myComputer.ndi`

Command Line

Tool:	Installation agent
Example:	`-o InventoryFile=myfile.ndi`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

InventoryScriptsDir

Command line | Registry

The location of scripts to be run immediately before inventory data is uploaded through the distribution hierarchy. All scripts that exist in this location are run.

Values / range:	String
Default value:	`$(CommonAppDataFolder)\ManageSoft Corp\ManageSoft`
Example value:	`C:\IT-Admin\`

Command Line

Tool:	Installation agent
Example:	`-o InventoryScriptDir=C:\data`

Registry

Installed by:	Installation of RayManageSofti for managed devices
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

LauncherCommandLine

Registry

Specifies installation agent parameters to pass to RayManageSofti when applying policy information.

Values / range:	Valid installation agent command line parameters. See the *RMS System Reference* for details.
Default value:	`{empty}`
Example value:	`-o UserInteractionLevel=Quiet`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Policy Client\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\PolicyClient\CurrentVersion`

ListeningPort

Registry

Specifies the port number that the TCP-based listening agent monitors for incoming requests.

Be aware:
If you change the port on which managed devices are listening for jobs, you must make a corresponding change to the **ListeningPort** preference on distribution servers. Refer to the *Configuring the port for the TCP-based listening agent* topic in the *Discovery and adoption* chapter of *RMS Configuration* for more details.

Values / range:	Any valid port number
Default value:	`7020`
Example value:	`9080`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \RemoteExecution\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\RemoteExecution\CurrentVersion`

LocalDeviceAutoRebootMessage

Security Manager only
Command line | Registry

If managed devices require a reboot after applying security setting packages, this preference is used to record the message that is displayed to the end-user before the managed device is automatically rebooted.

Also see *LocalDeviceAutoRebootPeriod* and *LocalDeviceAutoRebootWithNoWait*.

Values / range:	String
Default value:	Security settings have been applied to restrict the devices that can be used on your computer.

Command Line

Tool:	Installation agent
Example:	`-o LocalDeviceAutoRebootMessage= ”We’ve updated your computer, and it now needs to reboot. Sorry for the inconvenience.”`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceAutoRebootNotLoggedIn

Security Manager only
Command line | Registry

If the **LocalDeviceForceReboot** preference is set to `False` and no user is logged on, this preference specifies whether reboot takes place automatically (`True`) or not (`False`).

If set to `False`, Security Manager waits for the specified period before rebooting the managed device. The wait period is determined by the **LocalDeviceAutoRebootPeriod** preference.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Security agent
Example:	`-o LocalDeviceAutorebootNotLoggedIn=True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceAutoRebootPeriod

Security Manager only
Registry

If the **LocalDeviceForceReboot** preference is set to `False` and the end-user does not respond, this preference specifies the number of minutes before a reboot takes place.

The minimum number of minutes is 5, and the maximum is 60000.

Values / range:	Integer between 5 and 60000.
Default value:	`5`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceAutoRebootWithNoWait

Security Manager only
Registry

If the **LocalDeviceForceReboot** preference is set to `False` and this preference is set to `True`, Security Manager waits for the specified period before rebooting the managed device.

The period to wait before a reboot is determined by the **LocalDeviceAutoRebootPeriod** preference.

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceBluetooth

Security Manager only
Registry

Security Manager provides the facility to specify local hardware devices that are permitted to run on managed devices (the settings are on the **Device Settings** tab of the **Device Security Settings** node).

This setting specifies whether the default setting for Bluetooth devices is enabled (`True`) or disabled (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceCDDrive

Security Manager only
Registry

Security Manager provides the facility to specify local hardware devices that are permitted to run on managed devices (the settings are on the **Device Settings** tab of the **Device Security Settings** node).

This setting specifies whether the default setting for CD/DVD drives is enabled (`True`) or disabled (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceFirewire

Security Manager only
Registry

Security Manager provides the facility to specify local hardware devices that are permitted to run on managed devices. (The settings are on the **Device Settings** tab of the **Device Security Settings** node.)

This setting specifies whether the default setting for all firewire ports is enabled (`True`) or disabled (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceFloppyDisk

Security Manager only
Registry

Security Manager provides the facility to specify local hardware devices that are permitted to run on managed devices. (The settings are on the **Device Settings** tab of the **Device Security Settings** node.)

This setting specifies whether the default setting for floppy disk drives is enabled (`True`) or disabled (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceForceReboot

Security Manager only
Registry

If managed devices require a reboot after applying security setting packages, this preference specifies whether to force the reboot (`True`) or prompt the end-user to reboot (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceInfrared

Security Manager only
Registry

Security Manager provides the facility to specify local hardware devices that are permitted to run on managed devices. (The settings are on the **Device Settings** tab of the **Device Security Settings** node.)

This setting specifies whether the default setting for infrared devices is enabled (`True`) or disabled (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceModem

Security Manager only
Registry

Security Manager provides the facility to specify local hardware devices that are permitted to run on managed devices. (The settings are on the **Device Settings** tab of the **Device Security Settings** node.)

This setting specifies whether the default setting for modems is enabled (`True`) or disabled (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceParallelPorts

Security Manager only
Registry

Security Manager provides the facility to specify local hardware devices that are permitted to run on managed devices. (The settings are on the **Device Settings** tab of the **Device Security Settings** node.)

This setting specifies whether the default setting for parallel ports is enabled (`True`) or disabled (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDevicePCMCIA

Security Manager only
Registry

Security Manager provides the facility to specify local hardware devices that are permitted to run on managed devices. (The settings are on the **Device Settings** tab of the **Device Security Settings** node.)

This setting specifies whether the default setting for PCMCIA cards is enabled (`True`) or disabled (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceSerialPorts

Security Manager only
Registry

Security Manager provides the facility to specify local hardware devices that are permitted to run on managed devices. (The settings are on the **Device Settings** tab of the **Device Security Settings** node.)

This setting specifies whether the default setting for serial ports is enabled (`True`) or disabled (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceTimeout

Security Manager only
Command line | Registry

Specifies the maximum number of seconds that the security agent will wait for a managed device to respond to a local device management request before timing out and moving onto other local devices.

Values / range:	Any positive integer (number of seconds)
Default value:	30

Command Line

Tool:	Security agent
Example:	`-o LocalDeviceTimeout=30`

Registry

Installed by:	Installation of Security Manager on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceUSB

Security Manager only
Registry

Security Manager provides the facility to specify local hardware devices that are permitted to run on managed devices. (The settings are on the **Device Settings** tab of the **Device Security Settings** node.)

This setting specifies whether the default setting for removable storage devices, such as USB devices, is enabled (`True`) or disabled (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

LocalDeviceWiFi

Security Manager only
Registry

Security Manager provides the facility to specify local hardware devices that are permitted to run on managed devices. (The settings are on the **Device Settings** tab of the **Device Security Settings** node.)

This setting specifies whether the default setting for WiFi 802.11x devices is enabled (`True`) or disabled (`False`).

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

Locale

Command line | Registry

local setting used by the selection agent. This setting normally reflects the value of the **User Locale** preference, but can be manually overridden for testing purposes.

Where there are alternate localizations of the user interface available, the selection agent will try to match the system setting for locale with the locale variable in the application file directory path, thereby switching to the appropriate localized user interface.

Values / range:	Any valid two-character abbreviation for locale. For currently valid values, check *ISO 3166-1-alpha-2 code*.
Default value:	`$(UserLocale)`
Example value:	`DE`

Command Line

Tool:	Package selection agent
Example:	`-o Locale="DE"`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Selector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Selector\CurrentVersion`

LocaleDefault

Command line | Registry

The default locale setting used by the selection agent. The **LocaleDefault** preference only has effect when the selection agent cannot find the locale specified in **Locale**.

Values / range:	Any valid two- or three-character abbreviation for locale. Any third character (representing dialect) is ignored. For currently valid values,check *ISO 3166-1-alpha-2 code*.
Default value:	`EN`
Example value:	`DE`

Command Line

Tool:	Package selection agent
Example:	`-o LocaleDefault="DE"`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Selector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Selector\CurrentVersion`

Log

Registry

Instructs RayManageSofti to upload logging files from the managed device to the specified server location. This preference is set during installation and should not be altered by end-users.

Values / range:	Valid location
Default value:	`$(ServerLocation)/Logs/$(MachineId) at $(DateTime).log`
Example value:	`$(ServerLocation)/Logs/$(MachineId) at $(DateTime)_$(GUID).log`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common \Rules`
Computer preference:	`[Registry]\ManageSoft\Common\Rules`

LogFileOld (Installation Agent)

Command line | Registry | Project variable

When the installation agent log file reaches its maximum size (defined in **LogFileSize (installation agent)**), the file is renamed to the value in **LogFileOld**. This overwrites the previous **LogFileOld** file.

Values / range:	Local and UNC network files
Default value:	`$(TempDirectory)\ManageSoft\launcher.old.log`
Example value:	`C:\temp\Installation old.log`

Command Line

Tool:	Installation agent
Example:	`-o LogFileOld=c:\temp\installation old.log`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`LogFileOld`
Reference as:	`$(LogFileOld)`

LogFileSize (Installation Agent)

Command line | Registry | Project variable

When the main installation agent log file (defined in **LogFile (installation agent)**) reaches the size defined in **LogFileSize**, the file is renamed (to the value in **LogFileOld (installation agent)**).

A new log file is created. This allows you to retain additional log information.
The size must be expressed as the number of bytes of the maximum allowed log size. If this entry is `empty` or set to `zero`, there is no log size limit and the size of the log file continues to grow.

Values / range:	Numeric (number of bytes)
Default value:	`524288`
Example value:	`3126000 (3Mb)`

Command Line

Tool:	Installation agent
Example:	`-o LogFileSize=1024000`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`LogFileSize`
Reference as:	`$(LogFileSize)`

LogInstallCheck

Command line | Registry

Specifies whether RayManageSofti should recreate installation events while checking packages for installation or upgrade. If installation event records are recreated, they use today's date as the installation date.

This setting is useful if installation records have been removed on the administration server. See *How installation event preferences interact* for a description of how this preference interacts with others that create or update installation event records.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Policy agent, installation agent
Example:	`-o LogInstallCheck=True`

Registry

Installed by:	Installation of RayManageSofti on managed devices
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

How Installation Event Preferences Interact

When policy is applied, the following occurs:

- If the same version of the package is already installed:
  - If **LogInstallCheck** is `True`, an installation event record is created
  - If **LogInstallCheck** is `False`, no installation event record is created
- If the same version of the package is not installed, and the package is required on this computer, installation is attempted and:
  - If installation succeeds and **LogInstallPass** is `True`, the successful installation is logged
  - If installation fails and **LogInstallFail** is `True`, the failed installation is logged
- If the same version of the package is not installed, and the package is not required for this computer:
  - If any existing version of this package is currently installed, the existing installation event record is left unchanged
  - If no existing version of this package is installed, and the package is new in policy, a "not required" installation event record is created
  - If no existing version of this package is installed, and the package is not new in policy:

- If **InstallationStatusRefreshPeriod** is `0` or **LogNotRequiredCheck** is `True`, a "not required" installation event record is created
- If **InstallationStatusRefreshPeriod** has a `non-zero` value or **LogNotRequiredCheck** is `False`, no installation event record is created.

LogInstallFail

Command line | Registry

Specifies whether (`True`) or not (`False`) RayManageSofti should log failed installation attempts.

See *How installation event preferences interact* for a description of how this preference interacts with other installation event preferences that create or update installation event records.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Policy agent, installation agent
Example:	`-o LogInstallFail=False`

Registry

Installed by:	Installation of RayManageSofti on managed devices
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

LogInstallPass

Command line | Registry

Specifies whether (`True`) or not (`False`) RayManageSofti should log successful installation events.

See *How installation event preferences interact* for a description of how this preference interacts with other installation event preferences that create or update installation event records.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Policy agent, installation agent
Example:	`-o LogInstallPass=False`

Registry

Installed by:	Installation of RayManageSofti on managed devices
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

LogLevel (Installation Agent)

Command line | Registry | Project variable

Determines the level of logging returned by the RayManageSofti installation agent. This logging is output to the file whose name is stored in the **LogFile (installation agent)** preference.

Values / range:	One or more logging levels, separated by commas. See the *RMS System Reference* for details.
Default value:	`A-z` (all logging)
Example value:	`G0,4`

Command Line

Tool:	Installation agent
Example:	`-o LogLevel=G0,4`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`LogLevel`
Reference as:	`$(LogLevel)`

LogNotRequiredCheck

Command line | Registry

WARNING:
Deprecated: Use InstallationStatusRefreshPeriod=0 instead.

Specifies whether (`True`) or not (`False`) RayManageSofti should log applications that are not required on the managed device while checking packages for installation or upgrade. This setting is useful if installation records have been removed on the administration server. If event records are recreated, they use today's date.

See *How installation event preferences interact* for a description of how this preference interacts with other installation event preferences that create or update installation event records.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Policy agent, installation agent
Example:	`-o LogNotRequiredCheck=True`

Registry

Installed by:	Installation of RayManageSofti on managed devices
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

LogonServer

Project variable

The name of the logon server computer to which the managed device normally connects.

Values / range:	UNC name of the domain controller that validates user logons
Default value:	Name of the logon server
Example value:	`\\myserver`

Project Variable

Define as:	Predefined within Windows network configuration.
Reference as:	`$(LogonServer)`

LogUninstallFail

Command line | Registry

Specifies whether (`True`) or not (`False`) RayManageSofti should log failed uninstall attempts.

See *How installation event preferences interact* for a description of how this preference interacts with other installation event preferences that create or update installation event records.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Policy agent, installation agent
Example:	`-o LogUninstallFail=False`

Registry

Installed by:	Installation of RayManageSofti on managed devices
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

LogUninstallPass

Command line | Registry

Specifies whether (`True`) or not (`False`) RayManageSofti should log successful uninstalls.

See *How installation event preferences interact* for a description of how this preference interacts with other installation event preferences that create or update installation event records.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Policy agent, installation agent
Example:	`-o LogUninstallPass=False`

Registry

Installed by:	Installation of RayManageSofti on managed devices
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

LowestPriority

Registry

Specifies the lowest upload/download priority that can be assigned to a distribution server. The higher the number, the lower the priority.

When assigning priorities, RayManageSofti normalizes the calculated priority to fit within the range identified by **HighestPriority** and **LowestPriority**. The lowest priority is commonly set to 100.

Values / range:	Recommended 1-100 (but can extend from -231 to 231)
Default value:	`99`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \NetSelector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\NetSelector\CurrentVersion`

LowProfile (Installation Agent, Inventory Agent)

Command line | Registry | Project variable

Determines the CPU priority of RayManageSofti on the managed device.

- When set to `True`, RayManageSofti processes run with low priority.
- When set to `False`, RayManageSofti processes run with normal priority.

Values / range:	Boolean
Default value:	No default in registry; default behavior is `False`

Command Line

Tool:	Installation agent, inventory agent
Example:	`-o LowProfile=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	• `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	• `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Tracker\CurrentVersion`

Project Variable

Define as:	`LowProfile`
Reference as:	`$(LowProfile)`

MachineAlternateRegistryHive

Registry | Project variable

Specifies an alternative registry hive to be used by the managed device during self-update.

Be aware:
The default value allows RayManageSofti managed device settings to be controlled by native Active Directory Group Policy.

These default locations are not visible to end users. Alternative registry hives have no effect on the **DownloadSettings**, **UploadSettings**, and Rules keys under `Software\ManageSoft Corp\ManageSoft \Common`.

Be aware:
Be aware:
The loading sequence for registry preferences is:

- Default machine hive
  ("`HKEY_LOCAL_MACHINE\SOFTWARE\ManageSoft Corp\ManageSoft`")
- Alternative machine hive
  (pointed by **MachineAlternateRegistryHive**, under `HKEY_LOCAL_MACHINE`)
- Default user hive
  ("`HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft`")
- Alternative user hive
  (pointed by **UserAlternateRegistryHive**, under `HKEY_CURRENT_USER`)

Values / range:	Valid registry hive
Default value:	`SOFTWARE\Policies\ManageSoft Corp\ManageSoft`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Common`

Project Variable

Define as:	`MachineAlternateRegistryHive`
Reference as:	`$(MachineAlternateRegistryHive)`

MachineInventoryDirectory

Command line | Registry

The location in which to store machine inventories.

Values / range:	Valid location
Default value:	`$(CommonAppDataFolder)\ManageSoft Corp\ManageSoft\Tracker\Inventories`

Command Line

Tool:	Inventory agent
Example:	`-o MachineInventoryDirectory=C:\ManageSoft Corp\ManageSoft\Tracker\Inventories`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

MachineZeroTouchDirectory

Command Line

In case of a remote call the location is used for the machine inventories.

Values / range:	Valid location
Default Value:	`$(CommonAppDataFolder)\ManageSoft Corp\ManageSoft\Tracker\ZeroTouch`

Command Line

Tool:	Inventory Agent
Example:	`-o MachineZeroTouchDirectory=C:\ManageSoft Corp\ManageSoft\Tracker \ZeroTouch`

The default entry can be changed when calling the inventory agent.

MachinePolicyDirectory

Registry

The location to store the current machine policy.

Values / range:	Valid folder and path
Default value:	`$(CommonAppDataFolder)\ManageSoft Corp\ManageSoft\Policy Client\Policies\Merged\ Machine`
Example value:	`C:\Temp\MachinePolicies`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \NetSelector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\NetSelector\CurrentVersion`

MachinePolicyPackageDirectory

Registry

The location where package information associated with machine policy is cached.

Values / range:	Valid folder and path
Default value:	`$(CommonAppDataFolder)\ManageSoft Corp\ManageSoft \Policy Client\Packages`
Example value:	`C:\Temp\MyMachinePolicy\PackageInfo`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\ Policy Client\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Policy Client\CurrentVersion`

MachineSecurityAnalysisDirectory

Registry | Project variable

The path to the folder in which the security analysis (`.msa`) files are stored locally.

Values / range:	Local directory
Default value:	`$(CommonAppDataFolder)\ManageSoft Corp\ManageSoft\Security Agent\Security Analysis`
Example value:	`C:\Program Files\Security Analysis`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

Project Variable

Define as:	Predefined by RayManageSofti
Reference as:	`$(MachineSecurityAnalysisDirectory)`

ManageSoftPackages

Command line | Registry

Information on all software installed onto the computer will be gathered when this value is set to `True`.
No information will be gathered if this option is set to `False`.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Inventory agent
Example:	`-o ManageSoftPackages=False`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\ManageSoftPackages`
Computer preference:	`[Registry]\ManageSoft\Tracker\ManageSoftPackages`

Manual Mapper

Registry

Allows you to map any executable name to a specified product, to include the product in application usage metering. Use this setting to specify an executable that will not be detected as installed by either RayManageSofti or the native package format, or is not detected from **Add/Remove Programs** (Windows devices only).

Create a key under:
`Usage Agent\CurrentVersion\Manual Mapper\`

This key is the application identifier for your mapper. Create multiple values for this key:
`Application="application name (friendly string)" ExecutablePath="path to executable, may contain regular expressions"`

Be aware:
You can specify either a file name or a folder name. When specifying Windows executable paths, you can specify file names under the Windows installation directories, but not simply Windows system Windows system folder names. Windows system folder names such as `C:\Windows and C:\Windows\System32` will be ignored.

```
Version="version"
Regex = "true" (this is only required if ExecutablePath contains regular expressions)
Priority = "priority for this key, which takes precedence over the default priority specified by ManualMapperDefaultPriority"
```

Be aware:
If you do not specify a priority, the value of **ManualMapperDefaultPriority** is used.

Typical regular expression syntax is supported for configuring usage metering. See *Mozilla Developer Network - Regular Expressions* for a reference. Note that expressions that affect vertical spacing, such as newline and carriage returns, have no meaning in this context and are not supported. [] is used to escape characters with special meaning. A summary of commonly-used regular expressions is:

Regular expression	Matches
.	Any single character.
*	Matches the preceding character one or more times.
[xyz]	A character set. Matches any one of the enclosed characters. A range of characters can be specified using a hyphen. For example, [a-d] is the same as [abcd]
x|y	Matches x or y. For example **Office 10|Office 11** matches **Office 10** or **Office 11** but not **Office 12**.

Example: You may not want applications in the Windows Directory to be monitored (recommended). Then at a later date you may want to monitor one program: `Solitaire (sol.exe)`.

Example key:

```
Application="Solitaire"
ExecutablePath="C:\WINNT\System32\sol.exe"
Version="1.0"
```

To track use of `sol.exe` even if users install it in a different location, use wildcards:

```
Application="Solitaire"
ExecutablePath=".*\sol[.]exe"
Version="1.0"
Regex = "true"
```

Values / range:	String
Default value:	None

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Usage Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

ManualMapperDefaultPriority

Registry

For application usage tracking, only one application can "own" a file or directory. Where more than one application being tracked specifies the same file or directory, the value of this preference is used to determine which application the file or directory is allocated to for tracking. This process occurs at the time the application is being specified for tracking. The application with the highest value for this preference owns the file or directory for usage tracking. Where more than one application specifies the same file or directory for usage tracking, and both have identical priorities, the application whose usage tracking is most recently defined takes precedence.

The default value for this preference, 20, is automatically higher than the value assigned to other data sources such as Windows Installer, **Add/Remove Programs**, and so on. These alternate data source are assigned a priority of 10. (The order in which usage data is constructed is: the Manual Mapper preference values, native package format, the RayManageSofti cache, **Add/Remove Programs**.)

Values / range:	Integer between 1 and 10000
Default value:	`20`
Supported from:	Release 8.0

Registry

Installed by:	RayManageSofti internals, or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Usage Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

MBSACmdLine

Security Manager only
Registry

Command line to check security compliance using Microsoft Baseline Security Analyzer version 1.2.1 (MBSA 1.2.1).
Also see *MBSAPATH* and *MSSECUREPath*.

Values / range:	String
Default value:	`"$(MBSAPath)\mbsacli.exe" /hf -o tab -history 3 -x "$(MSSECUREPath)" -f "{}" /nvc`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

MBSA2CmdLine

Security Manager only
Registry

Command line to check security compliance using Microsoft Baseline Security Analyzer version 2.0 (MBSA 2.0). Also see *MBSA2PATH*.

Values / range:	String
Default value:	`"$(MBSA2Path)\mbsacli.exe" /catalog "$(WSUSSCANPath)" /n OS+Password+IIS+SQL/o mbsaresults.xml /nvc /nai /wi`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

MBSAPATH

Security Manager only
Registry

Location of MBSA 1.2 executable used to check security compliance.
Also see *MBSACmdLine*.

Values / range:	String
Default value:	`$(ProgramFiles)\Microsoft Baseline Security Analyzer`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

MBSA2PATH

Security Manager only
Registry

Location of MBSA 2.0 executable used to check security compliance.

Also see *MBSA2CmdLine* and *Windows Platform Windows Installer SDK*.

Values / range:	String
Default value:	`$(ProgramFiles)\Microsoft Baseline Security Analyzer 2`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

MBSAVulnerabilityCmdLine

Security Manager only
Registry

The command line to run to scan a computer for vulnerabilities.

Values / range:	String
Default value:	`"$(MBSAPath)\mbsacli.exe"; /n Updates \ /o mbsaresults.xml /mvc`
Example value:	`"(MBSAPath)\mbsacli.exe"; /n Updates /o mbsaresults.xml /nvc`

Registry

Installed by:	Creation of security settings package (managed device settings package)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

MinFreeDisk

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).

It specifies the amount of free disk space (in MB) that must exist on the managed device in order for the peer download agent to download files. The peer download agent checks free disk space before starting to download a file.

Values / range:	Integer between 10 and 2000000
Default value:	`100`

Command Line

Tool:	Peer download agent
Example:	`-debug -o MinFreeDisk="500"`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

MinimumDCSpeed

Command line | Registry | Project variable

Applies only for client-side policy merging, when **AutoDetectDC** is `False`.

This preference specifies the minimum network speed (in bits per second) between the managed device and the domain controller that is required to apply policy.
If the detected speed is below this value, client-side policy will not be applied.

Values / range:	Numeric (bits per second)
Default value:	No default, so RayManageSofti will attempt to connect regardless of speed.
Example value:	`1000`

Command Line

Tool:	Installation agent
Example:	`-o MinimumDCSpeed=1000`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Common`
Computer preference:	`[Registry]\ManageSoft\Common`

Project Variable

Define as:	`MinimumDCSpeed`
Reference as:	`$(MinimumDCSpeed)`

MinInventoryInterval

Command line | Registry

Specifies the minimum interval (in hours) between collections of inventory. The inventory agent will not generate or upload inventory if it is invoked less than this period of time after the most recent inventory generation. This preference controls the collection of inventory under RayManageSofti as well as zero-touch inventory.

The time of the last inventory generation is determined by looking at the last modified time of the last cached inventory file, typically stored under
`Application Data\ManageSoft Corp\ManageSoft\Tracker\Inventories\`.

Values / range:	Any non-negative integer
Default value:	`0`

Command Line

Tool:	Inventory agent
Example:	`ndtrack -o MinInventoryInterval=24` (Generates inventory at most once per day)

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

MinRunTime

Command line | Registry

Specifies the minimum time in seconds an application must run for before application usage data will be recorded for it. The value must be greater than 0 otherwise the default will be used.

Values / range:	Integer greater than 0
Default value:	60

Command Line

Tool:	Application usage agent
Example:	`-o MinRunTime="90"`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

MSI

Command line | Registry

When set to `True`, Microsoft Installer (MSI) package information is added to the inventories.
When set to `False`, RayManageSofti does not include MSI package information in inventories.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Inventory agent
Example:	`-o MSI=False`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

MsiBaseURL

Registry | Project variable

Identifies the Web location from which the application can be retrieved. The URL value will be returned in the syntax expected by MSI.

This is the same as the RayManageSofti pre-defined project variable **$(BaseURL)**, except that the URL value will be returned in a special syntax expected by MSI.

Values / range:	Valid URL
Default value:	folder in which the package is located
Example value:	`C:\ManageSoft\Packages\App\MsiSource\Appmsi`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	Pre-defined by RayManageSofti
Reference as:	`$(!MsiBaseURL)`

MsiReinstallFeatures

Registry | Project variable

Identifies which MSI components will be installed. This is equivalent to the MSI property **REINSTALL**.

For more information on MSI command lines, refer to the *Windows Platform Windows Installer SDK*.

Values / range:	See your supporting documentation for Windows Installer
Default value:	`ALL` (install all components)

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`MsiReinstallFeatures`
Reference as:	`$(!MsiReinstallFeatures)`

MsiReinstallModeLevel

Registry | Project variable

Identifies what will be reinstalled. This may be changed files, newer files, registry files, or all files. It is equivalent to the MSI property **REINSTALLMODE** and option **/f** in the **msiexec.exe** command line.

For more information on MSI command lines, refer to the *Windows Platform Windows Installer SDK*.

Values / range:	Any combination of the following letters: a, c, d, e, m, p, o, s, u, v. See your supporting MSI documentation for details about what each letter represents.
Default value:	`vomus` (complete reinstall)
Example value:	`osmu`

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`MsiReinstallModeLevel`
Reference as:	`$(!MsiReinstallModeLevel)`

MsiRepair

Registry | Project variable

This works in conjunction with **MsiReinstallFeatures**.
MsiReinstallFeatures controls which MSI repair operations are used to reinstall packages. **MsiRepair** determines whether these repairs are performed at the same time as RayManageSofti self-healing operations.

When set to `True`, MSI repairs are initiated at the same time as self-healing operations.
When set to `False`, MSI repairs are not initiated when RayManageSofti performs self-healing operations.

Values / range:	Boolean
Default value:	`False`

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`MsiRepair`
Reference as:	`$(!MsiRepair)`

MsiRepairLevel

Registry | Project variable

Identifies what will be repaired. This may be changed files, newer files, registry files, or all files. It is equivalent to the MSI property **REINSTALLMODE** and option
/f in the **msiexec.exe** command line.

For more information on MSI command lines, refer to the *Windows Platform Windows Installer SDK*.

Values / range:	Any combination of the following letters: a, c, d, e, m, p, o, s, u, v. See your supporting MSI documentation for details about what each letter represents.
Default value:	No default in registry; default behavior os (or osmu if the installation agent has been configured to repair registry information)
Example value:	`vomus`

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`MsiRepairLevel`
Reference as:	`$(!MsiRepairLevel)`

MsiSourceLocation

Command line | Registry

Specifies whether managed devices should install a Windows Installer package from the local Windows Installer cache, or from a distribution location.

For more information on MSI command lines, refer to the *Windows Platform Windows Installer SDK*.

Values / range:	Cache (to install from local cache), Server (to install from distribution location)
Default value:	`Cache`

Command Line

Tool:	Installation agent, policy agent
Example:	`-o MsiSourceLocation=Server`

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Be aware:
To use this setting on RayManageSofti for managed devices for releases earlier than 7.0, set `HKEY_LOCAL_MACHINE\SOFTWARE\ManageSoft Corp\ManageSoft\Launcher\Symbols \MsiSource`.

MsiUILevel

Registry | Project variable

Indicates the user interaction level for MSI. This may be full, basic, reduced, or no UI. It is equivalent to the **/q** argument on the **msiexec.exe** command line.

For more information on MSI command lines, refer to the *Windows Platform Windows Installer SDK*.

Values / range:	/q, /qn, /qb, /qr, /qf, /qn+, /qb+, /qb+!, /qb-, /qb-! See your supporting MSI documentation for details about what each letter represents.
Default value:	Depends on the **UserInteractionLevel** setting: • If set to `Quiet`, `Auto`, or `Status`, **MsiUILevel** defaults to `/qn` • If set to `Full`, **MsiUILevel** defaults to `/qb`
Example value:	`/qb`

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`MsiUILevel`
Reference as:	`$(!MsiUILevel)`

MsiUninstallArgs

Registry | Project variable

Records any arguments to include in the MSI command line for uninstall operations.

Values / range:	See your supporting MSI documentation
Default value:	`{empty}`
Example value:	`/l*v c:\temp\msi.log` (A command line argument to turn on logging)

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`MsiUninstallArgs`
Reference as:	`$(!MsiUninstallArgs)`

MSSECUREPath

Security Manager only
Registry

Location of downloaded **mssecure.xml** file used to check security compliance.

Values / range:	String
Default value:	`$(ETCPInstallDir)\SecurityPatch\mssecure.xml`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

NativeScheduler

Command line | Registry

Indicates whether or not RayManageSofti task scheduling is used. The options are:

- `taskschd` - Use the Microsoft Task Scheduler to run events (*only* available on Windows devices). If this is not available, use the RayManageSofti Task Scheduler.
- `Ndtask` - Use the RayManageSofti Task Scheduler to run events. If this is *not* available, use the Microsoft Task Scheduler.

Values / range:	`taskschd, ndtask`
Default value:	`ndtask`

Command Line

Tool:	Scheduling agent
Example:	`-o NativeScheduler=ndtask`

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Schedule Agent\CurrentVersion`

ndsensNetType

Command line | Registry

This preference applies only on Windows devices.
This value determines when a **When connected to network** trigger is deemed to have occurred, causing the command given by **ndsensNetUp** to be executed. It will only trigger if the network is of a certain type. There are three possible values:

1. Local area network (LAN)
2. Wide area network (WAN)
3. Either LAN or WAN.

RayManageSofti monitors these network types. For example, if **ndsensNetType**=2, RayManageSofti only monitors for connections to WANs.

Values / range:	1, 2, 3
Default value:	3

Command Line

Tool:	Scheduling agent
Example:	`-o ndsensNetType=2`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Schedule Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Schedule Agent\CurrentVersion`

NetworkHighSpeed (Installation Agent)

Command line | Registry | Project variable

This preference specifies the lowest network speed (in bits per second) that RayManageSofti will consider to be a high-speed network connection to a server.

RayManageSofti needs to identify whether a high-speed network connection or low-speed network connection is operating in order to determine the bandwidth to be used for uploads and downloads. The bandwidth percentage is stored in **NetworkHighSpeed (installation agent)** and **NetworkLowUsage**.

If **NetworkHighSpeed** is set to 0 (*default*), RayManageSofti does not limit its bandwidth usage according to measured network speed. With this configuration, content is downloaded at the maximum rate specified by the **NetworkMaxRate** (installation agent) preference, and the **NetworkHighSpeed** (installation agent) and **NetworkLowUsage** values are ignored.

Values / range:	Numeric (number of bits per second)
Default value:	`0 (zero)`
Example value:	`32`

Command Line

Tool:	Installation agent
Example:	`-o NetworkHighSpeed=32`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkHighSpeed`
Reference as:	`$(NetworkHighSpeed)`

NetworkHighSpeed (Upload Agent)

Command line | Registry | Project variable

Specifies the lowest network speed (in bits per second) that RayManageSofti will consider to be a high-speed network connection to a server.

RayManageSofti needs to identify whether a high-speed network connection or low-speed network connection is operating in order to determine the bandwidth to be used for uploads and downloads. The bandwidth percentage is stored in **NetworkHighSpeed (installation agent)** and **NetworkLowUsage**.

If **NetworkHighSpeed** is set to 0 (the default), RayManageSofti does not limit its bandwidth usage according to measured network speed. With this configuration, content is downloaded at the maximum rate specified by the **NetworkMaxRate** (upload agent) preference, and the **NetworkHighSpeed (installation agent)** and **NetworkLowUsage** values are ignored.

If peer-to-peer file sharing is enabled (**AllowPeerToPeer** is `True`), this preference does not apply. Instead, see *ParentConnectionWindows* and *PeerConnectionWindows*.

Values / range:	Numeric (number of bits per second)
Default value:	`0 (zero)`
Example value:	`32`

Command Line

Tool:	Upload agent
Example:	-o NetworkHighSpeed=32

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Uploader\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Uploader\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkHighSpeed`
Reference as:	`$(NetworkHighSpeed)`

NetworkHighUsage

Command line | Registry | Project variable

This preference specifies the maximum percentage of bandwidth that RayManageSofti uses for uploads and downloads on a high-speed connection.

If **NetworkHighUsage** is configured to be outside the range specified by **NetworkHighUsageLowerLimit** and **NetworkHighUsageUpperLimit**, and the lower limit is strictly less than the upper limit, **NetworkHighUsage** is automatically set to the closest range endpoint. For example, consider a case where **NetworkHighUsageLowerLimit** is 10 and **NetworkHighUsageUpperLimit** is 40. If **NetworkHighUsage** is set to 5, RayManageSofti resets it to 10. If **NetworkHighUsage** is set to 60, RayManageSofti resets it to 40.

If **NetworkHighUsage** is set to 0, the installation agent downloads files using 0.1% of the measured bandwidth.

If peer-to-peer file sharing is enabled (**AllowPeerToPeer** is `True`), this preference does not apply. Instead, see *ParentConnectionWindows* and *PeerConnectionWindows*.

Values / range:	Numeric (percentage 0-100)
Default value:	For downloads, `100`. For uploads, no default.
Example value:	`55`

Command Line

Tool:	Installation agent
Example:	`-o NetworkHighUsage=75`

Registry

Installed by:	For downloads, installation of RayManageSofti on a managed device (computer preference). For uploads, manual configuration.
User preference:	For installation agent downloads, in order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common` For data reporting agent uploads, in order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Uploader\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	For installation agent downloads, in order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common` For data reporting agent uploads, in order of precedence: • `[Registry]\ManageSoft\Uploader\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkHighUsage`
Reference as:	`$(NetworkHighUsage)`

NetworkHighUsageLowerLimit

Command line | Registry | Project variable

Specifies the minimum **NetworkHighSpeed (installation agent)** value that can be set for a managed device by an end-user moving the bandwidth usage slider control in the installation agent.

The bandwidth usage slider control is available only if:

- The connection speed to the distribution location can be determined
- The installation agent is downloading from a distribution location to which the detected connection speed is at least the speed specified by **NetworkHighSpeed (installation agent)**
- The **NetworkHighUsageLowerLimit** is strictly less than the **NetworkHighUsageUpperLimit**.

The **NetworkHighSpeed (installation agent)** value is recorded under the user's `HKEY_CURRENT_USER` registry area.

Values / range:	Numeric (percentage 0-100)
Default value:	`100`
Example value:	`10`

Command Line

Tool:	Installation agent
Example:	`ndlaunch -o NetworkHighUsageLowerLimit=10`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\SOFTWARE\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\SOFTWARE\ManageSoft Corp\ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkHighUsageLowerLimit`
Reference as:	`$(NetworkHighUsageLowerLimit)`

NetworkHighUsageUpperLimit

Command line | Registry | Project variable

Specifies the maximum **NetworkHighSpeed (installation agent)** value that can be set for a managed device by an end-user moving the bandwidth usage slider control in the installation agent.

The bandwidth usage slider control is available only if:

- The connection speed to the distribution location can be determined
- The installation agent is downloading from a distribution location to which the detected connection speed is at least the speed specified by **NetworkHighSpeed (installation agent)**
- The **NetworkHighUsageLowerLimit** is strictly less than the **NetworkHighUsageUpperLimit**.

The **NetworkHighSpeed (installation agent)** value is recorded under the user's HKEY_CURRENT_USER registry area.

Values / range:	Numeric (percentage 0-100)
Default value:	`100`
Example value:	`90`

Command Line

Tool:	Installation agent
Example:	`ndlaunch -o NetworkHighUsageUpperLimit=90`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\SOFTWARE\ManageSoft Corp\ManageSoft\Launcher\CurrentVersion` • `HKEY_CURRENT_USER\SOFTWARE\ManageSoft Corp\ManageSoft\Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkHighUsageUpperLimit`
Reference as:	`$(NetworkHighUsageUpperLimit)`

NetworkLowUsage

Command line | Registry | Project variable

This preference specifies the maximum percentage of bandwidth that RayManageSofti uses for uploads and downloads on a low-speed connection.

If **NetworkLowUsage** is configured to be outside the range specified by **NetworkLowUsageLowerLimit** and **NetworkLowUsageUpperLimit**, and the lower limit is strictly less than the upper limit, **NetworkLowUsage** is automatically set to the closest range endpoint.

For example, consider a range where **NetworkLowUsageLowerLimit** is 10 and **NetworkLowUsageUpperLimit** is 40. If **NetworkLowUsage** is set to 5, RayManageSofti resets it to 10. If **NetworkLowUsage** is set to 60, RayManageSofti resets it to 40. If NetworkLowUsage is set to 0, the installation agent downloads files using 0.1% of the measured bandwidth.

If peer-to-peer file sharing is enabled (**AllowPeerToPeer** is `True`), this preference does not apply. Instead, see *ParentConnectionWindows* and *PeerConnectionWindows*.

Values / range:	Numeric (percentage 0-100)
Default value:	For downloads, `100`. For uploads, no default.
Example value:	`30`

Command Line

Tool:	Installation agent
Example:	`-o NetworkLowUsage=50`

Registry

Installed by:	For downloads, installation of RayManageSofti on a managed device (computer preference). For uploads, manual configuration.
User preference:	For downloads, in order of preference: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common` For uploads, in order of preference: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Uploader\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	For downloads, in order of preference: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common For uploads in order of preference:` • `[Registry]\ManageSoft\Uploader\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkLowUsage`
Reference as:	`$(NetworkLowUsage)`

NetworkLowUsageLowerLimit

Command line | Registry | Project variable

Specifies the minimum **NetworkLowUsage** value that can be set for a managed device by an end-user moving the bandwidth usage slider control in the installation agent.

The bandwidth usage slider control is available only if:

- The connection speed to the distribution location can be determined
- The installation agent is downloading from a distribution location to which the detected connection speed is less than the speed specified by **NetworkHighSpeed (installation agent)**
- **NetworkLowUsageLowerLimit** is strictly less than **NetworkLowUsageUpperLimit**.

The **NetworkLowUsage** value is recorded under the user's `HKEY_CURRENT_USER` registry area.

Values / range:	Numeric (percentage 0-100)
Default value:	`100`
Example value:	`10`

Command Line

Tool:	Installation agent
Example:	`ndlaunch -o NetworkLowUsageLowerLimit=10`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\SOFTWARE\ManageSoft Corp\ManageSoft\Launcher\CurrentVersion` • `HKEY_CURRENT_USER\SOFTWARE\ManageSoft Corp\ManageSoft\Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkLowUsageLowerLimit`
Reference as:	`$(NetworkLowUsageLowerLimit)`

NetworkLowUsageUpperLimit

Command line | Registry | Project variable

Specifies the maximum **NetworkLowUsage** value that can be set for a managed device by an end-user moving the bandwidth usage slider control in the installation agent.

The bandwidth usage slider control can only be changed for **NetworkLowUsage** if:

- The connection speed to the distribution location can be determined
- The installation agent is downloading from a distribution location to which the detected connection speed is less than the speed specified by **NetworkHighSpeed (installation agent)**
- The **NetworkLowUsageLowerLimit** is strictly less than the **NetworkLowUsageUpperLimit**.

The **NetworkLowUsage** value is recorded under the user's `HKEY_CURRENT_USER` registry area.

Values / range:	Numeric (percentage 0-100)
Default value:	`100`
Example value:	`90`

Command Line

Tool:	Installation agent
Example:	`ndlaunch -o NetworkLowUsageUpperLimit=90`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\SOFTWARE\ManageSoft Corp\ManageSoft\Launcher\CurrentVersion` • `HKEY_CURRENT_USER\SOFTWARE\ManageSoft Corp\ManageSoft\Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkLowUsageUpperLimit`
Reference as:	`$(NetworkLowUsageUpperLimit)`

NetworkMaxByteLevelSpeed

Command line | Registry | Project variable

This determines the maximum network connection speed (in Bytes per second) at which byte-level differencing for file downloads is disabled. (This takes into account bandwidth optimization, but does not check actual achieved download speed.)

If the network speed exceeds this maximum, then there is no significant advantage in performing byte-level differencing, and the CPU operations associated with download can be reduced by disabling byte-level differencing.

Values / range:	Numeric
Default value:	`262144` (approximates the speed of a 2Mbps WAN)
Example value:	`56000`

Command Line

Tool:	Installation agent
Example:	`-o NetworkMaxByteLevelSpeed=200000`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`NetworkMaxByteLevelSpeed`
Reference as:	`$(NetworkMaxByteLevelSpeed)`

NetworkMaxRate (Installation Agent)

Command line | Registry | Project variable

Represents the bytes per second at which the managed device accesses data over the network. This preference is not used if the **NetworkSpeed** preference can be determined and the **NetworkHighSpeed (installation agent)** preference is set to a non-zero value.

If peer-to-peer file sharing is enabled (**AllowPeerToPeer** is `True`), this preference does not apply. Instead, see *ParentConnectionWindows* and *PeerConnectionWindows*.

Values / range:	Numeric (bytes per second)
Default value:	`0` (unlimited)
Example value:	`64`

Command Line

Tool:	Installation agent
Example:	`-o NetworkMaxRate=64`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkMaxRate`
Reference as:	`$(NetworkMaxRate)`

NetworkMaxRate (Upload Agent)

Command line | Registry | Project variable

Represents the bytes per second at which the managed device accesses data over the network. This preference is not used if the **NetworkSpeed** preference can be determined and the **NetworkHighSpeed (upload agent)** preference is set to a non-zero value.

If peer-to-peer file sharing is enabled (**AllowPeerToPeer** is `True`), this preference does not apply. Instead, see *ParentConnectionWindows* and *PeerConnectionWindows*.

Values / range:	Numeric (bytes per second)
Default value:	`0` (unlimited)
Example value:	`64`

Command Line

Tool:	Upload agent
Example:	`-o NetworkMaxRate=64`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Uploader\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft` \Common
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Uploader\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkMaxRate`
Reference as:	`$(NetworkMaxRate)`

NetworkMinSpeed (Installation Agent)

Command line | Registry | Project variable

The minimum network speed (bits per second) for RayManageSofti to initiate a check for updates.

Values / range:	Numeric (bytes per second)
Default value:	No default in registry; default behavior `1`
Example value:	`2000`

Command Line

Tool:	Installation agent
Example:	`-o NetworkMinSpeed=2000`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkMinSpeed`
Reference as:	`$(NetworkMinSpeed)`

NetworkMinSpeed (Upload Agent)

Command line | Registry | Project variable

The minimum network speed (bits per second) for RayManageSofti to initiate a check for updates.

Values / range:	Numeric (bytes per second)
Default value:	No default in registry; default behavior `1`
Example value:	`2000`

Command Line

Tool:	Upload agent
Example:	`-o NetworkMinSpeed=2000`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Uploader\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Uploader\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkMinSpeed`
Reference as:	`$(NetworkMinSpeed)`

NetworkRetries

Command line | Registry | Project variable

The number of times a failed network operation is retried before an alternative download location is attempted. Note that the maximum number of attempts to connect to a file share is controlled by the **ConnectionAttempts** preference, and *not* by the **NetworkRetries** preference.

Values / range:	Numeric
Default value:	`1`

Command Line

Tool:	Installation agent
Example:	`-o NetworkRetries=2`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`NetworkRetries`
Reference as:	`$(NetworkRetries)`

NetworkSense (Installation Agent)

Command line | Registry | Project variable

This preference determines whether network checks are bypassed.
When set to `True`, RayManageSofti performs its normal network checks.
When set to `False`, the RayManageSofti installation agent bypasses any network checks it usually performs (such as the bandwidth speed).

If peer-to-peer file sharing is enabled (**AllowPeerToPeer** is `True`), this preference does *not* apply. Instead, see *ParentConnectionWindows* and *PeerConnectionWindows*.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o NetworkSense=False`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkSense`
Reference as:	`$(NetworkSense)`

NetworkSense (Inventory Agent)

Command line | Registry

This preference determines whether network checks are bypassed for uploads performed by the inventory agent.
When set to `True`, RayManageSofti performs its normal network checks.

When set to `False`, the RayManageSofti inventory agent bypasses any network checks it usually performs (such as the bandwidth speed).

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Inventory agent
Example:	`-o NetworkSense=False`

Registry

Installed by:	Update settings package on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Tracker\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Tracker\CurrentVersion` • `[Registry]\ManageSoft\Common`

NetworkSense (Security Agent)

Command line | Registry

This preference determines whether network checks are bypassed for uploads performed by the security agent. When set to `True`, RayManageSofti performs its normal network checks.

When set to `False`, the RayManageSofti security agent bypasses any network checks it usually performs (such as the bandwidth speed).

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Security agent
Example:	`-o NetworkSense=False`

Registry

Installed by:	Update settings package on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Security Agent\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Security Agent\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`

NetworkSense (Upload Agent)

Command line | Registry | Project variable

This preference determines whether network checks are bypassed.
When set to `True`, RayManageSofti performs its normal network checks.

When set to `False`, the RayManageSofti upload agent bypasses any network checks it usually performs (such as the bandwidth speed).

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Upload agent
Example:	`-o NetworkSense=False`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Uploader\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft` \Common
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Uploader\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkSense`
Reference as:	`$(NetworkSense)`

NetworkTimeout (Installation Agent)

Command line | Registry | Project variable

Determines the length of time in seconds of inactivity after which a network operation will time out.

Values / range:	Numeric (seconds)
Default value:	`30`

Command Line

Tool:	Installation agent
Example:	`-o NetworkTimeout=10`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkTimeout`
Reference as:	`$(NetworkTimeout)`

NetworkTimeout (Upload Agent)

Command line | Registry | Project variable

Determines the length of time in seconds of inactivity after which a network operation will time out.

Values / range:	Numeric (seconds)
Default value:	`600` (ten minutes)

Command Line

Tool:	Upload agent
Example:	`-o NetworkTimeout=1000`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Uploader\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Uploader\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`NetworkTimeout`
Reference as:	`$(NetworkTimeout)`

NoStage

Command line | Registry | Project variable

When set to `True`, files are downloaded directly to their install location, without placing them in a staging area. As a result, no checks are performed on the files before they overwrite any existing files.

When set to `False`, RayManageSofti uses a staging area before transferring files to their install location.

Be aware:
This bypasses the staging area and is not recommended.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o NoStage=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`NoStage`
Reference as:	`$(NoStage)`

OfficeConvertPath

Security Manager only
Registry | Project variable

Identifies the path to locate the Microsoft **convert.exe** file. This file is used to convert Microsoft Office patch data from **inventory.exe** into an XML format file.

Values / range:	Valid executable file and path
Default value:	`$(ProgramFiles)\Microsoft Baseline Security Analyzer \OfficeUpd\convert.exe`
Example value:	`C:\MyFiles\convert.exe`

Registry

Installed by:	Installation of Security Manager on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common`
Computer preference:	`[Registry]\ManageSoft\Common`

Project Variable

Define as:	`OfficeConvertPath`
Reference as:	`$(OfficeConvertPath)`

OfficeInventoryPath

Security Manager only
Registry | Project variable

Identifies the path to locate the Microsoft **inventory.exe** file. This file is used in determining the Microsoft Office patches required for a computer.

Values / range:	Valid executable file and path
Default value:	`$(ProgramFiles)\Microsoft Baseline Security Analyzer\OfficeUpd\inventory.exe`
Example value:	`C:\PROGRA~1\MyFiles\inventory.exe`

Registry

Installed by:	Installation of Security Manager on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common`
Computer preference:	`[Registry]\ManageSoft\Common`

Project Variable

Define as:	`OfficeInventoryPath`
Reference as:	`$(OfficeInventoryPath)`

ParentActivityTimeout

Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
The RayManageSofti peer download agent can trigger a "When connected to network" event that you can use to trigger a scheduled task. For example, you may wish to run a task when a dialup line is up and in use.

The peer download agent triggers the “When connected to network” event when either it, or a peer it can see on the network, initiates a file download from a distribution server, when no download has been active for at least **ParentActivityTimeout** seconds.

Values / range:	Integer less than 7200 (120 minutes, or 2 hours)
Default value:	`300` (5 minutes)
Example value:	`150`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Common\`

ParentConnectionWindows

Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`) and **IgnoreConnectionWindows** is `False`.

It specifies the time periods during which the RayManageSofti peer download agent can download packages from its closest distribution server, and upload status information to reporting locations. (To specify time periods during which downloads from peer managed devices are allowed, use **PeerConnectionWindows**.)

Downloads in progress at the end of a time period will be stopped immediately, and subsequent downloads of the same file will continue from that point.

Be aware:
When remote execution operations require data to be uploaded or downloaded, these operations override the peer-to-peer settings.

Also see *AllowPeerToPeer*, *PeerConnectionWindows* and *IgnoreConnectionWindows*.

Values / range:	String in HHMM-HHMM:PP,HHMM-HHMM:PP format, where: • HHMM is a local time, in 24 hour format (0000-2359) If not specified, the time period is the whole day. If the end time is earlier than the start time, it is assumed to be for the day following the start time. For example, 0400-0100 specifies the period between 4am one day, and 1am the following day. • PP is a percentage of the end-to-end bandwidth that can be used by this managed device (optional). The way the maximum available bandwidth is calculated is described in *WANMaxRate*. The amount of bandwidth available for this managed device is the maximum rate, divided by the number of peers on the network conducting transfers (regardless of whether they are transferring to/from the same distribution server.) Multiple, non-overlapping time periods can be specified, separated by commas.
Default value:	`None`
Example value:	`2300-0100:85` (downloads can occur between 11pm and 1am, using a maximum of 85% of available bandwidth)

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Common\`

PeerAveragingTime

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It is used in conjunction with **PeerMaxRate** to limit bandwidth used for peer-to-peer file sharing operations. It specifies the averaging period (in minutes) used to smooth the estimate of transfers to and from peer managed devices. See *PeerMaxRate* for details about how these preferences are used together.

Increasing the value of this preference means that the estimate takes longer to change as the actual transfer rate changes. In normal use, you will not need to change the value of this preference.
RayManageSofti retrieves the value for this preference from the registry every five seconds. You do not need to restart RayManageSofti on managed devices after changing the value of this preference.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line.*

Values / range:	Integer between 1 - 60
Default value:	`5`

Command Line

Tool:	Peer download agent
Example:	`-debug -o PeerAveragingTime=10`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

PeerConnectionWindows

Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`) and **IgnoreConnectionWindows** is `False`.

It specifies the time periods during which the RayManageSofti peer download agent can download packages from peer managed devices. (To specify time periods during which the peer download agent can download packages from the nearest distribution server, use **ParentConnectionWindows**.)
Downloads in progress at the end of a time period will be stopped immediately, and subsequent downloads of the same file will continue from that point.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	String in HHMM-HHMM:PP,HHMM-HHMM:PP format, where: • HHMM is a local time, in 24 hour format (0000-2359) If not specified, the time period is the whole day. If the end time is earlier than the start time, it is assumed to be for the day following the start time. For example, 0400-0100 specifies the period between 4am one day, and 1am the following day. • PP is a percentage of the end-to-end bandwidth that can be used by this managed device (optional). The way the maximum available bandwidth is calculated is described in *PeerMaxRate*. Multiple, non-overlapping time periods can be specified, separated by commas.
Default value:	`None`
Example value:	`2300-0100:85` (downloads can occur between 11pm and 1am, using a maximum of 85% of available bandwidth)

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Common\`

PeerListenQueue

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the maximum number of connection requests to queue before refusing additional connections. This prevents requests from being rejected under some TCP configurations when peer-to-peer is configured to pull files (**PeerPush** is `False`) and a number of files are requested in less than 100 milliseconds.

Consider setting this preference if the following errors are logged by **mgsdl.exe**, or if advised by your Raynet support representative:

- Error code 10054 - An existing connection was forcibly closed by the remote host. This often appears in configurations where
  `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Para meters \SynAttackProtect`
  has a value of 1 or 2. This configuration results in "half open" connections as the Windows TCP-layer SYN attack prevention blocks the connections opened by Deployment Manager for peer-to-peer file transfer.
- Error code 10061 - No connection could be made because the target machine actively refused it.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 1 - 100
Default value:	`5`

Command Line

Tool:	Peer download agent
Example:	`-debug -o PeerListenQueue=15`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

PeerMaxRate

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the maximum allowable rate (in bytes per second) for transfers to and from peer managed devices.

You might choose a value for this preference based on the speed of your LAN connection. For example, if you have a 100MB LAN connection, you could set the value of this preference to 1,048,576 (1 MB/second) to ensure that peer-to-peer file sharing operations do not use more than 10% of the available bandwidth.

PeerMaxRate is used in conjunction with **PeerAveragingTime** to limit bandwidth used by peer-to-peer file sharing operations.
RayManageSofti calculates the sum of file transfers that have occurred between this managed device and peer managed devices. Since transfers occur in blocks, not as a continuous stream, RayManageSofti smooths out the variation in transfer rates using the **PeerAveragingTime** and a simple exponential decay algorithm. The result is an estimate of the transfer rate. Transfer rates will be decreased if the estimated rate exceeds the specified **PeerMaxRate**, and increased if they are below the specified PeerMaxRate. (Transfers can creep up to the **PeerMaxRate**, but will drop back rapidly when the estimated rate is greater than **PeerMaxRate**.)

RayManageSofti retrieves the value for this preference from the registry every five seconds. You do not need to restart RayManageSofti on managed devices after changing the value of this preference.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 1024 - 134217728
Default value:	`16777216`

Command Line

Tool:	Peer download agent
Example:	`-debug -o PeerMaxRate=2048`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

PeerPullPort

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the TCP port used for peer-to-peer file fetch operations. This port must not be firewalled. Refer to the *RayManageSofti and personal firewalls* chapter of *RMS Configuration* for details.

Also see *PeerPush* and *AllowPeerToPeer*.

Values / range:	Integer between 1000 - 65535
Default value:	`6087`

Command Line

Tool:	Peer download agent
Example:	`-debug -o PeerPullPort=7400`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

PeerPush

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
If the value of this preference is `True`, peer managed devices can immediately push a file in response to a request for it. Allowing immediate push of files reduces UDP traffic from searching, but since the port used can't be configured, this is not suitable for managed devices running firewall software. In networks where most peers are firewalled, this preference should be set to `False`.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Peer download agent
Example:	`-debug -o PeerPush=False`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

PeerSearchDuration

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the number of seconds the peer download agent will spend searching for files in peer managed device caches before choosing to download the file from the closest distribution server.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 3 and 600
Default value:	`10`

Command Line

Tool:	Peer download agent
Example:	`-debug -o PeerSearchDuration=30`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

PeerSearchPort

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the UDP port used for peer-to-peer search operations. This port must not be firewalled. Refer to the *RayManageSoft and personal firewalls* chapter of *RMS Configuration* for details.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 1000 - 65535
Default value:	`6087`

Command Line

Tool:	Peer download agent
Example:	`-debug -o PeerSearchPort=7400`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

PeerTransferLimit

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).

It specifies the number of simultaneous peer-to-peer search and file transfer operations allowed across all peers on the subnet. Before commencing a file search or transfer operation, the managed device checks to see how many peers are currently performing searches or transfers. This managed device will not start a search or transfer if the number of managed devices currently downloading is equal to or greater than the **PeerTransferLimit** setting for this device.

Values / range:	Integer between 1 - 64
Default value:	10

Command Line

Tool:	Peer download agent
Example:	`-debug -o PeerTransferLimit=30`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

PipeName (Peer Download Agent)

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).

The name of the operating system pipe used to communicate with the **ManageSoft Peer-to-Peer Download Service** (pipes are used to supply the output of one program as input to another). In normal operation, you will not need to change this value.

Values / range:	String
Default value:	`RayManageSofti Download Service`
Example value:	`My Testing Service`

Command Line

Tool:	Peer download agent
Example:	`-debug -o PipeName="My Testing Service"`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Downloader`
Computer preference:	`[Registry]\ManageSoft\Downloader`

PipeName (Security Agent)

Security Manager only
Command line | Registry

The name of the operating system pipe used to communicate with the security agent. (Pipes are used to supply the output of one program as input to another.) In normal operation, you will not need to change this value.

Values / range:	String
Default value:	`RayManageSofti Security Service`
Example value:	`My Security Service`

Command Line

Tool:	Security agent
Example:	`-debug -o PipeName="My Security Service"`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

PlatformSpecificPackages

Command line | Registry

Specifies whether information is included in software inventory about non-Windows, platform-specific packages (for example `lpp`, `pkg`, `rpm` and `sd-ux`). This setting is ignored on Windows computers.

Values / range:	Boolean
Default value:	`True` (when registry value has not been set)
Example value:	`False`
Supported from:	Release 8.2

Command Line

Tool:	Inventory agent
Example:	`-o PlatformSpecificPackages="True"`

Registry

Installed by:	Update settings package on a managed device (computer preference).
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

PolicyComplianceLog

Registry

Instructs RayManageSofti to upload policy compliance log files from the managed device to the specified server location.
This preference is set during installation and should not be altered by end-users.

Values / range:	Valid location
Default value:	`$(ServerLocation)/PolicyComplianceLogs/$(UserId) on $(MachineId) at $(DateTime).plc`
Example value:	`$(ServerLocation)/PolicyComplianceLogs/$(UserId) on $(MachineId) at $(DateTime).plc`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common` \Rules
Computer preference:	`[Registry]\ManageSoft\Common\Rules`

PolicyPackageRefreshPeriod

Registry

This preference specifies the number of seconds after successfully downloading package (`.osd`) files during which download of those files should not be attempted again.
If a value is set for this preference, each time policy is applied the policy agent checks to see if package files required by policy have been downloaded within this time period. If so, the currently downloaded package files are used for installation.

If package files have not been downloaded within the time interval, they are downloaded only if they have changed since they were last downloaded to the managed device. (The check for this depends on the protocol in use. For HTTP downloads, an If-Modified-Since HTTP request is used. Equivalent requests are made for other protocols.)

For example, imagine that a package file `MyApplication.osd` was downloaded at 4pm, and used to install the application `MyApplication`. The value of **PolicyPackageRefreshPeriod** is 43200 (12 hours). A scheduled task applies policy and attempts to update `MyApplication` at 8pm. Since 8pm is less than 12 hours after `MyApplication.osd` was last downloaded, no attempt is made to download the file.

If no value is set for this preference, package files are always downloaded, regardless of whether or not they have changed since they were last downloaded.
If you always want *newer* package files (those that have changed since the last download to the managed device) to be downloaded when policy is applied, set the value of this preference to 0 (zero).

Values / range:	Integer in the range 0 - 1000000000
Default value:	`86400 seconds` (24 hours)
Example value:	`28800 seconds` (8 hours)

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Common\`
Computer preference:	`[Registry]\ManageSoft\Common\`

PolicyRefreshPeriod

Registry

This preference specifies the number of seconds after successfully downloading policy (`.npl`) files during which the files should not be downloaded again.
If a value is set for this preference, each time policy is applied the policy agent checks to see if the policy files have been downloaded within this time period. If so, the currently downloaded policy files are used to apply policy.

If policy files have not been downloaded within the time interval, they are downloaded only if they have changed since they were last downloaded to the managed device. (The check for this depends on the protocol in use. For HTTP downloads, an If-Modified-Since HTTP request is used. Equivalent requests are made for other protocols.)
For example, imagine that policy files were downloaded and policy applied at 4pm, and that the value of **PolicyRefreshPeriod** is 43200 (12 hours). A scheduled task starts applying policy at 8pm. Since 8pm is less than 12 hours after the policy files were last downloaded, no attempt is made to download newer policy files.

If no value is set for this preference, policy files are always downloaded, regardless of whether or not they have changed since they were last downloaded.
If you always want *newer* policy files (those that have changed since the last download to the managed device) to be downloaded when policy is applied, set the value of this preference to 0 (zero).

Values / range:	Integer in the range 0 - 1000000000
Default value:	`43200 seconds` (12 hours)
Example value:	`28800 seconds` (8 hours)

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common\`
Computer preference:	`[Registry]\ManageSoft\Common\`

PolicyServerPriority

Command line | Registry

Specifies the priority to apply to the distribution location identified by the **PolicyServerURL** preference. Setting this preference to a low value (high priority) such as 0 results in the server identified by **PolicyServerURL** being used as a source for package downloads in preference to other servers. Setting it to a high value (low priority) such as 100 results in the server being prioritized after other servers.

PolicyServerPriority can also be set to the case-insensitive literal string Invalid. With this value, the server identified by **PolicyServerURL** will not be considered at all for package downloads.

For more information about distribution server priorities, see *RMS Configuration*.

Values / range:	Recommended range of 0 - 100, or Invalid
Default value:	`50`

Command Line

Tool:	Installation agent
Example:	`-o PolicyServerPriority=1`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

PolicySource

Registry

The location where the policy is generated. There are two options:

- **Server:** NPL policy files are generated on the administration server and distributed for use
- **Client:** policy is retrieved directly by the managed device from Active Directory.

Client-side merging is available only for managed devices connected to Active Directory. (You can use the **EnablePolicyFailover** preference to switch to using server-side policy in the event that Active Directory is not reachable when client-side policy is due to be applied.)

Values / range:	Server, Client
Default value:	`Server`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Policy Client\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Policy Client\CurrentVersion`

PostponeByDefault

Command line | Registry

Used to determine the default outcome when an end-user does not (or cannot) decide whether to postpone the installation of mandatory packages.

Depending on the value of the **PostponeUserInteractionLevel** and **UserInteractionLevel (installation agent)** preferences, end-users on managed devices may be interactively asked if they want to postpone installation of mandatory software.

Where settings prevent the offer being made, or where the end-user does not give a timely response to the offer, this preference determines the outcome used by the installation agent. By default, the installation agent does not postpone installations. However, if this preference is assessed and the value is `True`, the installation agent defers the installation, and it is re-assessed when the policy is next checked.

Also see *PostponeUserInteractionLevel*.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`ndlaunch -o PostponeByDefault=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

PostponeCmdLine

Registry

The command line to run to offer end-users the choice to postpone software installation.

Values / range:	Any valid command line to execute a program to offer end-users the opportunity to defer installation of software
Default value:	`$(PostponePath)`
Example value:	`$(Program Files)\MyCustomProgram.exe`
Supported from:	Release 8.0

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

See also:

- *PostponeUserInteractionLevel*
- *PostponeByDefault*
- *PostponePath*

PostponementQueryBefore

Command line | Registry

Used to determine when an end-user may be offered an option to postpone the installation of mandatory packages.

Depending on the value of the **PostponeUserInteractionLevel** and **UserInteractionLevel (installation agent)** preferences, end-users on managed devices may be interactively asked if they want to postpone installation of mandatory software.

This setting determines whether the offer to postpone installation may be made before the software package is downloaded to the managed device, or after download but immediately before installation commences. The offer may also be made at both these times. The default value preserves behavior from earlier releases of Deployment Manager.

Values / range:	Download, Install, DownloadAndInstall
Default value:	`Download`
Example value:	`Install`
Supported from:	Release 8.2

Command Line

Tool:	Installation agent
Example:	`ndlaunch -o PostponementQueryBefore=Download`

Registry

Installed by:	Applying a managed device settings package
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

See also:

- *PostponeUserInteractionLevel*
- *PostponeByDefault*
- *PostponePath*
- *PostponeCmdLine*

PostponePath

Registry

The name and location of the executable used to allow end-users to defer software installation. By default, this is **mgspostpone.exe**, described in the *Command line tools* chapter of the *RMS System Reference*.

Values / range:	Any valid local directory path and executable program name
Default value:	`$(ProgramPath)\mgspostpone.exe`
Example value:	`$(Program Files)\MyCustomProgram.exe`
Supported from:	Release 8.0

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

See also:

- *PostponeUserInteractionLevel*
- *PostponeByDefault*
- *PostponeCmdLine*

PostponeUserInteractionLevel

Command line | Registry

Controls whether end-users on managed devices are interactively asked if they want to postpone installations of mandatory packages that are appropriately configured in policy.

Possible values for this preference are:

- **Full:** End-users are asked if they want to postpone the installation of appropriately configured mandatory packages.
- **Default:** End-users are only prompted about postponement if the installation agent is running with a **UserInteractionLevel (installation agent)** of `Full`. End-users are not prompted if the installation agent is running with any other **UserInteractionLevel (installation agent)** setting.
- **Quiet**: End-users are not prompted about postponement.

To ensure the installation agent does not halt for too long without any user response, the postponement dialog is automatically dismissed after the time period specified by the **UITimeoutWait** preference.

The **PostponeByDefault** preference setting determines the default response controlling whether installation of mandatory packages is postponed when a user is not prompted, or the prompt dialog times out according to the **UITimeoutWait** preference.

Also see *UserInteractionLevel (adoption agent)*, *PostponeByDefault* and *UITimeoutWait*.

Values / range:	Full, Default, or Quiet
Default value:	`Full`
Example value:	`Default`

Command Line

Tool:	Installation agent
Example:	`ndlaunch -o PostponeUserInteractionLevel=Default`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

PreferenceUpdatePeriod

Command line | Registry

Specifies how often in seconds the application usage agent will refresh its settings from the registry. The value must be greater than 0 otherwise the default value will be used.

Values / range:	Integer greater than 0
Default value:	`86400`

Command Line

Tool:	Application usage agent
Example:	`-o PreferenceUpdatePeriod="90"`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

ProcessUpdatePeriod

Command line | Registry

Specifies how often in seconds the application usage agent will check for newly started or exited applications. The value must be greater than 0 otherwise the default value will be used.

Values / range:	Integer greater than 0
Default value:	60

Command Line

Tool:	Application usage agent
Example:	-o ProcessUpdatePeriod="90"

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device
User preference:	Not available
Computer preference:	[Registry]\ManageSoft\Usage Agent\CurrentVersion

ProductUpdatePeriod

Command line | Registry

Specifies how often in seconds the application usage agent will check for newly installed applications. The value must be greater than 0 otherwise the default value will be used.

Values / range:	Integer greater than 0
Default value:	`86400`

Command Line

Tool:	Application usage agent
Example:	`-o ProductUpdatePeriod="90"`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

ProgressDepth

Registry

The number of directory levels to search at initialization to approximate the number of directories searched during tracking.

Values / range:	Integer between 1-10
Default value:	No default in registry; default behavior 3

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

PromptOnCOMRegFailures

Command line | Registry | Project variable

Only applicable when **UserInteractionLevel (installation agent)** is set to `Full`.
When set to `True`, RayManageSofti prompts the user when it fails to register a COM server.
When set to `False`, RayManageSofti does not prompt the user and continues with the package installation.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o PromptOnCOMRegFailures=False`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`PromptOnCOMRegFailures`
Reference as:	`$(PromptOnCOMRegFailures)`

PromptOnInstallCompletion

Command line | Registry | Project variable

Only applicable when **UserInteractionLevel (installation agent)** is set to `Full`.
When set to `True`, RayManageSofti informs the user that the installation has been completed.
When set to `False`, RayManageSofti does not inform the user when installations are complete.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o PromptOnInstallCompletion=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`PromptOnInstallCompletion`
Reference as:	`$(PromptOnInstallCompletion)`

PromptOnUnInstallCompletion

Command line | Registry | Project variable

Only applicable when **UserInteractionLevel (installation agent)** is set to `Full`.

When set to `True`, RayManageSofti informs the user that the package uninstall has been completed.
When set to `False`, RayManageSofti does not inform the user when uninstalls are complete.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o PromptOnUninstallCompletion=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`PromptOnUninstallCompletion`
Reference as:	`$(PromptOnUninstallCompletion)`

PropagatePkgChanged

Command line | Registry | Project variable

When set to `True`, RayManageSofti reinstalls the base package if prerequisite package changed.
This is only applicable to Third-party installer packages.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o PropagatePkgChanged=False`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`PropagatePkgChanged`
Reference as:	`$(PropagatePkgChanged)`

PublicAppAccess

Command line | Registry | Project variable

Determines RayManageSofti access to Windows **Common** folders and files on Windows.
Options are:

- **FullAccess** - RayManageSofti can access areas of the file system available to all users
- **NoAccess** - RayManageSofti cannot access areas of the file system available to all users.

Note:
This preference does not override file system access. The **FullAccess** option does not provide access through RayManageSofti if the user does not already have access to the Common areas of their file system.

Values / range:	FullAccess, NoAccess
Default value:	`FullAccess`

Command Line

Tool:	Installation agent
Example:	`-o PublicAppAccess=NoAccess`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`PublicAppAccess`
Reference as:	`$(PublicAppAccess)`

QuietUntilUpdate

Command line | Registry | Project variable

When set to `True`, this option hides the RayManageSofti user interface on the managed device until either user interaction is necessary, or a package requires installation, upgrading, or uninstalling.

The user interface is hidden while RayManageSofti checks to see if an update is required. When the user interface is displayed, it is displayed according to the **UserInteractionLevel** preference.

When set to `False`, the RayManageSofti user interface displays according to the **UserInteractionLevel** preference, whether user interaction is required or not.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	-o QuietUntilUpdate=True

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`QuietUntilUpdate`
Reference as:	`$(QuietUntilUpdate)`

RebootCmdLine

Command line | Registry

Used on the managed device to reboot from the command line.

Values / range:	Name of executable in command path
Default value:	`"$(RebootPath)"`

Command Line

Tool:	Installation agent
Example:	`"$(RebootPath)" -t 60`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

RebootContinueAfterCmdFailure

Command line | Registry

Specifies whether to continue with the reboot of the managed device if the pre-reboot command returned a non-zero exit code (typically indicating that an error has occurred, or a warning generated).

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Reboot agent
Example:	`reboot.exe -c false`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

RebootIfRequired

Command line | Registry

Determines whether to reboot if RayManageSofti has determined that a reboot is necessary.

This preference sets the default response to the dialog that prompts end-users to confirm a reboot:

- When set to False, the default response is not to reboot
- When set to True, the default response on the dialog that prompts the end-user is **Reboot**. If **ForceReboot** is also `True`, the end-user is not given the option to not reboot.

If the dialog times out with no user response, or is not displayed because of the **UserInteractionLevel** and **AlwaysDisplayReboot** settings, or if no user is logged on, RayManageSofti will reboot automatically if **RebootIfRequired** is `True`.

If the desktop is locked, the installation and adoption agents uses **AllowRebootIfLocked** instead of **RebootIfRequired**.

For details about how this preference works in combination with other installation preferences to determine the appropriate reboot action, see *Reboot options*.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Adoption agent, Installation agent
Example:	`-o RebootIfRequired=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

RebootPostCommand

Command line | Registry

Command to execute after rebooting a managed device using reboot.exe. The command specified here is copied to `HKLM\SOFTWARE\ManageSoft Corp\ManageSoft\Common\RunOnce`, from which location it will be executed by the scheduling agent after the managed device reboots.

Values / range:	String
Default value:	`none`
Example value:	`chkdsk /f`

Command Line

Tool:	Reboot agent
Example:	`reboot.exe -a "regsrv32 /s /u C:\filename.dll"` `will register a DLL`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

RebootPreCommand

Command line | Registry

Command to execute before rebooting a managed device using reboot.exe.

Values / range:	String
Default value:	`none`
Example value:	`cleanmgr`

Command Line

Tool:	Reboot agent
Example:	`reboot.exe -b "regsrv32 /s /u C:\filename.dll"` will register a DLL

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

RebootPromptCycles

Command line | Registry

The number of times the end-user can postpone the managed device reboot initiated by Deployment Manager. Each postpone cycle terminates either when the end-user chooses to postpone or reboot, or at the conclusion of the period of time specified by **RebootPromptWait**.
When all postpone cycles are complete, the final reboot dialog is displayed. The appearance and behavior of the final reboot dialog are based on the settings of **RebootIfRequired**, **ForceReboot**, and **UITimeoutWait**.

If **RebootPromptCycles** is set to `0` (default), the final reboot dialog is presented to the end-user immediately, with no option to postpone the reboot.
For example, if **RebootPromptCycles** is set to `2`, the postponement dialog is presented to the end-user a maximum of twice. Assuming that the end-user chooses to postpone the reboot each time, after the second postponement, the final reboot dialog is displayed after the time interval specified by **RebootPromptWait**. The appearance and behavior of this dialog are based on the settings of **RebootIfRequired**, **ForceReboot**, and **UITimeoutWait**.

If both this preference and **RebootPromptUnlimited** are set, **RebootPromptUnlimited** takes precedence. Setting this preference to `-1` is equivalent to setting **RebootPromptUnlimited**=`True`.

Values / range:	Integer
Default value:	`0`

Command Line

Tool:	Reboot agent
Example:	`reboot.exe -p 10 reboot.exe -o RebootPromptCycles=10`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

See also:

- *UITimeoutWait*
- *RebootPromptWait*
- *RebootPromptUnlimited*
- *RebootIfRequired*
- *ForceReboot*
- *AllowTimeoutIfLocked*

RebootPromptUnlimited

Command line | Registry

Specify that prompting to reboot will continue until the managed device is rebooted. This is equivalent to **RebootPromptCycles**=`-1`.
If both **RebootPromptCycles** and this preference are set, this preference takes precedence.

Values / range:	Boolean
Default value:	`False`
Supported from:	`Release 7.9.5`

Command Line

Tool:	Reboot agent
Example:	`reboot.exe -u reboot.exe -p -1`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

See also:

- *UITimeoutWait*
- *RebootPromptWait*
- *RebootIfRequired*
- *ForceReboot*
- *AllowTimeoutIfLocked*
- *RebootPromptCycles*

RebootPromptWait

Command line | Registry

The time interval, in seconds, to wait before re-displaying the dialog that prompts the end-user to reboot.

Values / range:	Integer greater then zero
Default value:	`600`

Command Line

Tool:	Reboot agent
Example:	`reboot.exe -w 1200`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

See also:

- *AllowTimeoutIfLocked*
- *RebootPromptCycles*

RefreshPeriod

Command line | Registry

The number of minutes between automatic refresh of data held by the package selection agent. This refreshes the underlying data, and, in the factory-supplied user interface, also refreshes the tabular data displayed to the end-user.

Values / range:	Number greater than zero (number of minutes)
Default value:	`5` (minutes)

Command Line

Tool:	Package selection agent
Example:	`-o RefreshPeriod=10`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Selector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Selector\CurrentVersion`

ReInstallRequiresVersionChange

Command line | Registry | Project variable

Determines when Deployment Manager will upgrade, downgrade, or reinstall packages.

If set to `True`, Deployment Manager will upgrade, downgrade, or reinstall packages on a managed device if either of the following has changed

- The package version number
- The MD5 digest calculated for all the package details that apply to this device based on the current platform, language, and architecture (as opposed to the MD5 of the overall package)

This behavior protects against re-installation of applications when the package has changed, but when those changes do not affect the current managed device. This still allows self-healing to occur in the event of file corruption, and for upgrades to occur when changes to a package do affect the current managed device.

If set to `False`, Deployment Manager will upgrade, downgrade, or reinstall packages on a managed device if either of the following has changed:

- The MD5 digest for the entire package
- The package version number

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o ReInstallRequiresVersionChange=False`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`ReInstallRequiresVersionChange`
Reference as:	`$(ReInstallRequiresVersionChange)`

RenotifyTimeout

Command line | Registry | Project variable

Determines the length of time in seconds that installation agent dialogs can remain hidden while waiting to time out before they are displayed again to the user.

Values / range:	Numeric (seconds)
Default value:	`240` (4 minutes)
Supported from:	`Release 7.9.5`

Command Line

Tool:	Installation agent
Example:	`-o RenotifyTimout=10`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`RenotifyTimeout`
Reference as:	`$(RenotifyTimeout)`

RetryPolicy

Command line | Registry

When set to `True`, RayManageSofti will attempt to retrieve RayManageSofti policy when the managed device boots if no machine schedule exists on the managed device. RayManageSofti uses the command within the **RetryPolicyCommand** preference to retrieve policy.

This is useful when performing automatic adoption of managed devices to ensure that temporary network outages do not halt the RayManageSofti adoption process.

When set to `False`, RayManageSofti will not retrieve RayManageSofti policy in the event that the managed device boots and no machine schedule exists on the managed device.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Scheduling agent
Example:	`-o RetryPolicy=False`

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Schedule Agent\CurrentVersion`

RetryPolicyCommand

Command line | Registry

If **RetryPolicy** is set to `True`, RayManageSofti uses the command stored in **RetryPolicyCommand** to attempt to retrieve group policy.
The policy is retrieved from the last known policy location, or can be included in the **RetryPolicyCommand** value.

Values / range:	Any valid policy agent command line
Default value:	`mgspolicy -t Machine -o UserInteractionLevel=Quiet`
Example value:	`mgspolicy -t Machine`

Command Line

Tool:	Scheduling agent

Example:	`-o RetryPolicyCommand=mgspolicy`
	`-t Machine`
	`-o UserInteractionLevel=Quiet`

Registry

Installed by:	Installation of RayManageSofti on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Schedule Agent\CurrentVersion`

RunInventoryScripts

Command line | Registry

When `True`, this preference specifies that inventory scripts should be run after managed devices have been inventoried. All scripts located in the location specified by **InventoryScriptsDir** are executed immediately after inventory data collection is complete.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Inventory agent
Example:	`-o RunInventoryScripts=True`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

RunVulnerabilityScripts

Security Manager only
Command line | Registry

When `True`, this preference specifies that scripts should be run after managed devices have been scanned for vulnerabilities. **VulnerabilityScripts** lists the scripts to be executed, and **VulnerabilityScriptsDir** specifies their location.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Security agent
Example:	`-o RunVulnerabilityScripts=True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

SaveAllUserSymbols

Command line | Registry | Project variable

Determines whether RayManageSofti retains installation preferences set by a top-level or prerequisite catalog.

- When set to `True`, RayManageSofti retains the existing values for preferences.
- When set to `False`, it saves only the preferences used by the current package.

Also see *Persistent managed device preferences.*

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o SaveAllUserSymbols=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`SaveAllUserSymbols`
Reference as:	`$(SaveAllUserSymbols)`

SearchFrequency

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).

The requesting managed device sends multiple UDP broadcast requests to cater for the fact that some packets may be dropped during transmission, and so that it can assemble a number of possible sources from which to retrieve required files. This preference specifies the time (in tenths of a second) between requests.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 1 - 10
Default value:	`10`

Command Line

Tool:	Peer download agent
Example:	`-debug -o SearchFrequency=5`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

SearchMaxOffer

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
Offers to send a requested file might be received from more than one peer. This preference specifies the number of offers of a file to receive from peers before terminating the search. Increasing this number can help distribute load and reduce file transfer failures, but waiting for more offers can make searching take longer.
If this number of offers is received before the number of requests specified by **SearchMinimum** is reached, additional requests will be sent, until **SearchMinimum** is reached.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 1 - 10
Default value:	`5`

Command Line

Tool:	Peer download agent
Example:	`-debug -o SearchMaxOffer=3`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

SearchMinimum

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).

The requesting managed device sends multiple UDP broadcast requests to cater for the fact that some packets may be dropped during transmission, and so that it can assemble a number of possible sources from which to retrieve required files. This preference specifies the minimum number of requests to send. It will transmit this number of requests, even if it receives offers of the file from peer managed devices before it has sent them all.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 1 - 20
Default value:	`2`

Command Line

Tool:	Peer download agent
Example:	`-debug -o SearchMinimum=5`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

SearchRetry

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
SearchMinimum specifies the number of requests to send for each required file. This preference, **SearchRetry**, specifies the time interval (in seconds) between requests.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 60 - 3600
Default value:	`600`

Command Line

Tool:	Peer download agent
Example:	`-debug -o SearchRetry=1200`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

Security

Security Manager only
Registry

If `True`, perform security patch compliance check.

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

SecurityAnalysis

Security Manager only
Registry

Defines the naming conventions for uploaded files containing security analysis details.
This preference is set during installation and should not be altered by end-users.

Values / range:	String
Default value:	`$(ServerLocation)/SecurityAnalysis/$(MachineId) at $(DateTime).msa`

Registry

Installed by:	Installation of Security Manager on managed devices
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft` \Common\Rules
Computer preference:	`[Registry]\ManageSoft\Common\Rules`

SecurityLogInterval

Security Manager only
Command line | Registry

Specifies how often, in minutes, the security agent will generate security log files for upload. The default value of zero disables security log generation.
When enabled, this preference is typically set to values between 1440 (24 hours) and 10080 (7 days).

Values / range:	Integer between 0 – 10080
Default value:	`0`

Command Line

Tool:	Security agent
Example:	`-o SecurityLogInterval=10080`

Registry

Installed by:	Installation of RayManageSofti for managed devices
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

SecurityPatchRebootIfRequired

Security Manager only
Registry | Project variable

This preference is only used when security patches are being installed. If the installation of a security patch requires a reboot and the value of this preference is `True`, **RebootIfRequired** is set to `True`.

If you use this preference as a variable in a security patch package, you must set the **Set variable before processing package** check box when creating the package, so that the variable is set before the package installation command is run. Refer to the *RMS Packaging* for details.

Values / range:	Boolean
Default value:	`True`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`SecurityPatchRebootIfRequired`
Reference as:	`$(SecurityPatchRebootIfRequired)`

SelectorAlgorithm

Registry

Specifies the algorithm(s) used to assign values to the **Priority** registry keys for download and upload locations.

After application of the nominated algorithm(s), the managed device will attempt to collect packages from the highest priority server. In the event of connection failure, the managed device uses the other prioritized servers remaining in the list as failover servers.

RayManageSofti includes the following algorithms:

- **MgsADSiteMatch**
  Moves all servers in the current managed device's site to the front of the priority list
- **MgsBandwidth**
  Priorities are based on end-to-end bandwidth availability to the server
- **MgsDHCP**
  Priorities are based on lists of servers specified in DHCP
- **MgsDomainMatch**
  Priorities are determined by closest match in domain name
- **MgsIPMatch**
  Priorities are determined by closest IP address match
- **MgsNameMatch**
  Matches prefixes in computer names
- **MgsPing**
  Priorities are determined by fastest ping response time
- **MgsRandom**
  Random priorities are assigned
- **MgsServersFromAD**
  Priorities are determined according to lists of servers specified in Active Directory
- **MgsSubnetMatch**
  Moves all servers in the current subnet to the front of the priority list, but retaining the relative order of existing priorities.

Be aware:
Each algorithm may be given an integer parameter that determines the number of servers to which priorities will be assigned. Some algorithms may also be given an additional boolean attribute that can cause unmatched servers to be discarded from the list (priority set to the string literal invalid). Some algorithms also accept other parameters. For more information, see *RMS Configuration*.

Values / range:	`MgsADSiteMatch, MgsBandwidth, MgsDHCP, MgsDomainMatch, MgsIPMatch, MgsNameMatch, MgsPing, MgsRandom, MgsServersFromAD, MgsSubnetMatch (nDGRandom, nDGDomainMatch, nDGIPMatch`; also available for backward compatibility.)
Default value:	`MgsPing;MgsSubnetMatch`
Example value:	• `MgsRandom(3)` This means that RayManageSofti should randomly assign the top three servers (based on the priorities currently assigned). • `MgsADSiteMatch(, True);MgsSubnetMatch` This means that RayManageSofti lists servers outside the current managed device's site as "invalid". (**MgsSubnetMatch** will only prioritize valid servers set by **MgsADSiteMatch**.)

Registry

Installed by:	Installation of RayManageSofti on managed device
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \NetSelector\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\NetSelector\CurrentVersion`

SelfHeal

Command line | Registry | Project variable

Specify whether self-healing should occur for an individual package when RayManageSofti updates machine or user policies.

- `True` means that all packages on this managed device should self-heal.
- `False` means that no packages on this managed device should self-heal.
- *Any other value* means that self-healing should be attempted only on packages with a **SelfHeal** property whose value matches this string. For example, if a package has a **SelfHeal** value of `AlwaysHealMe`, and `SelfHeal` on a device is also set to `AlwaysHealMe`, self-healing of that package will occur on that device.

Values / range:	String
Default value:	`True`

Command Line

Tool:	Installation agent
Example:	`-o SelfHeal="False"`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`SelfHeal`
Reference as:	`$(SelfHeal)`

Be aware:
When using **SelfHeal** as a package variable, set the **Set variable before processing package** check box when creating the package. Refer to the *RMS Packaging* for details.

ServiceConnectTimeout

Controls the amount of time that `ndserv.exe` has to establish a named pipe connection with `ndlaunch.exe`. The default value for a timeout is 20 seconds.

Default value:	20 secs
Example value:	`20`

Registry

User preference:	`HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\ManageSoft Corp\Manag`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

ServiceCreateTimeout

Controls the amount of time that `ndlaunch.exe` has to establish a named pipe connection with `ndserv.exe`. The default value for a timeout is 30 seconds.

Default value:	20 secs
Example value:	`20`

Registry

User preference:	`HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\ManageSoft Corp\Manag`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

SessionBackupPeriod

Command line | Registry

Specifies how often in seconds the application usage agent will cache already-recorded application usage data. The value must be greater than 0 otherwise the default value will be used.

Values / range:	Integer greater than 0
Default value:	`3600`

Command Line

Tool:	Application usage agent
Example:	`-o SessionBackupPeriod=90`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

SessionListInterval

Security Manager only
Command line | Registry

Specifies how often, in minutes, the security agent will cache already-recorded user session data. The default value of zero disables this caching.
When enabled, this preference is typically set to values between 60 (1 hour) and 240 (4 hours).

Values / range:	Integer between 0–1440
Default value:	0

Command Line

Tool:	Security agent
Example:	`-o SessionListInterval=90`

Registry

Installed by:	Installation of RayManageSofti for managed devices
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

ShowIcon (Installation Agent)

Command line | Registry | Project variable

When set to `True`, RayManageSofti displays an icon in the system tray when it is installing or uninstalling an application. This icon displays, regardless of the value of the **UserInteractionLevel (installation agent)** preference.

If this icon is double-clicked and **UserInteractionLevel (installation agent)** is set to `Status` or `Auto`, the progress display toggles from being hidden to being visible.
When set to `False`, no icon will display.

Values / range:	Boolean
Default value:	No registry default; default behavior `False`

Command Line

Tool:	Installation agent
Example:	`-o ShowIcon=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`ShowIcon`
Reference as:	`$(ShowIcon)`

ShowIcon (Inventory Agent)

Command line | Registry | Project variable

When set to `True`, RayManageSofti displays an icon in the system tray when it is installing or uninstalling an application. This icon displays, regardless of the value of the **UserInteractionLevel (inventory agent)** preference.

If this icon is double-clicked and **UserInteractionLevel (inventory agent)** is set to `Status` or `Auto`, the progress display toggles from being hidden to being visible.
When set to `False`, no icon will display.

Values / range:	Boolean
Default value:	No registry default; default behavior `False`

Command Line

Tool:	Inventory agent
Example:	`-o ShowIcon=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Tracker\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Tracker\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`ShowIcon`
Reference as:	`$(ShowIcon)`

SMBIOSCmdLine

Command line | Registry

Specifies the command line used to invoke `smbios2.exe` to gather BIOS information during zero-touch hardware inventory collection. This preference is not used during normal inventory collection by RayManageSofti.

`smbios2.exe` is a utility developed by IBM that RayManageSofti uses to collect a range of information about a computer if WMI is not available on the computer. You may execute `smbios2.exe /h` by hand to obtain information about possible command arguments that can be used with `smbios2.exe`.

The default value of the **SMBIOSCmdLine** preference includes the `/G` option, which attempts to collect as much information about the computer as possible.

The `/G` option collects BIOS information using one particular approach, which very old BIOSes may not support. If you have old BIOSes in use within your organization, you may want to consider customizing this preference value to use the `/a` option instead of `/G`. `/a` uses an alternative approach for collecting information supported by older BIOSes. However, use of the `/a` option is known to result in problems such as hangs and crashes on some newer hardware, including many Dell and IBM computers.

Relative paths to executables in this preference setting are treated as relative to the directory containing ndtrack.exe.

This preference can be set to an empty value in order to prevent the inventory agent from using the `smbios2.exe` tool. This results in less BIOS information being collected during zero-touch hardware inventory collection.

The command line used to execute `smbios2.exe` normally includes `conspawn`, as shown in the default value below. `conspawn` is an executable used by RayManageSofti to reliably execute 16-bit DOS applications on various versions of Windows.

In order to successfully use `smbios2.exe` to gather inventory data, `smbios2.exe`, `conspawn.exe`, and `ide21201.vxd` should all exist in the same directory as `ntrack.exe`.

Values / range:	Any valid command line to execute `smbios2.exe` that results in output being written to standard output. This command line should include the `/1` argument.
Default value:	`conspawn smbios2.exe /1 /G`
Example value:	`conspawn smbios2.exe /1 /a`

Command Line

Tool:	Inventory agent
Example:	`ndtrack -t Machine -o SMBIOSCmdLine="conspawn smbios2.exe /1 /a"`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

SourceFile

Command line | Registry | Project variable

Identifies the file or files to be uploaded by the upload agent.

Values / range:	Either a UNC `(\\MYCOMPUTER\...)` or a drive `(C:\)` path to the required file or files. Wildcard characters can be used in the filename component.
Default value:	`None`
Example value:	`C:\Temp\*.log`

Command Line

Tool:	Upload agent
Example:	`-o SourceFile=c:\temp\*.log` `-o SourceFile=c:\temp\mylogfile.log`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Uploader\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Uploader\CurrentVersion`

Project Variable

Define as:	`SourceFile`
Reference as:	`$(SourceFile)`

SourceRemove

Command line | Registry | Project variable

Determines whether the upload agent removes the uploaded file(s) from the source location after a successful upload. If `True`, the files are removed from the source location.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Upload agent
Example:	`-o SourceRemove -o SourceRemove=False`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Uploader\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Uploader\CurrentVersion`

Project Variable

Define as:	`SourceRemove`
Reference as:	`$(SourceRemove)`

StageInactivePackages

Command line | Registry

Used to download (stage) all application files referenced in a policy that is scheduled to be activated some time in the future. This allows packages to be installed immediately once the policy is activated, without having to wait for lengthy downloads, as the files have already been unobtrusively downloaded beforehand. When this preference is False, RayManageSofti does not start downloading application files until the policy is activated.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o StageInactivePackages=True`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

StartupDelay

Registry

Specifies the wait time (in seconds) between a managed device booting up and the application usage agent starting. This can be used to delay the application usage agent startup to allow managed devices to boot up faster and provide the end-user with an interactive desktop more quickly. When the application usage agent startup is delayed, any applications executed prior to its initialization will not be tracked as used.

Values / range:	Integer greater than 0 (number of seconds)
Default value:	0

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Usage Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

StrictInstall

Command line | Registry

If set to `True`, the policy agent returns a non-zero exit code if any package in policy fails to install.
If set to `False`, the policy agent may return a zero exit code even if packages failed to install. Do not use the policy agent's return code to test for success unless this preference is set to `True`.

Values / range:	Boolean
Default value:	No registry default; default behavior `False`

Command Line

Tool:	Policy agent, installation agent
Example:	`-o StrictInstall=True`

Registry

Installed by:	Manual configuration
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

SupplyWorstCaseReturnValue

Command line | Registry | Project variable

When set to `False`, RayManageSofti only returns an error when an installation agent operation fails, regardless of whether the installation is successful or not.

When set to `True`, an error is reported if an installation fails during an application self-heal, revision or upgrade.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o SupplyWorstCaseReturnValue=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`SupplyWorstCaseReturnValue`
Reference as:	`$(SupplyWorstCaseReturnValue)`

TechniciansGroup

Security Manager only
Command line | Registry

The list of groups whose users can override security policies on managed devices.

Values / range:	String
Default value:	`MGS Field Technicians ; MGS Administrators`
Example value:	`Help Desk Staff ; MGS Administrators`

Command Line

Tool:	Security agent
Example:	`-o TechniciansGroup="MGS Field Technicians"`

Registry

Installed by:	Creation of security settings package (managed device settings package)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Security Agent\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

TrackFilesInUserInventory

Registry

This setting controls whether file evidence data is collected for user inventories. By default, file evidence is not collected, as file evidence cannot be directly linked to particular users.

Set this preference to `True` if you want file evidence to be collected for user inventories. In Releases 7.9 and earlier, file evidence was collected for user inventories. To continue this behavior for release 7.9.5 or later, set this preference to `True`

Values / range:	Boolean
Default value:	`False`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

TrustDatabaseFxd

Registry

When set to `True`, trusted and excluded locations can only be changed by users with administrator privileges.

When set to `False`, or if this preference has not been configured on the managed device, RayManageSofti allows any user to change the trusted and excluded locations.

Values / range:	Boolean
Default value:	`False`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

UITimeoutWait

Command line | Registry

Determines the number of seconds that a RayManageSofti installation agent dialog displays before timing out and automatically selecting the default response. The first dialog to time out will do so after the specified period (or never, if the time out period is set to 0 seconds). Subsequent dialogs will time out after a maximum of 60 seconds.

For example, if the time out period is set at 300 seconds, and the first dialog does time out after 300 seconds, subsequent dialogs will time out after 60 seconds.

For details about how this preference works in combination with other installation preferences to determine appropriate reboot actions, see *Reboot options*.

Values / range:	Integer greater than 0 (number of seconds)
Default value:	`300`

Command Line

Tool:	Installation agent
Example:	`-o UITimeoutWait=30`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

UninstallIShieldSilently

Command line | Registry

By default, uninstall operations for InstallShield packages display a dialog prompting the user to confirm the deletion of files. This preference allows you to control whether this dialog displays during an uninstall.

The following options are available:

- **Always**
  The uninstall is always silent (RayManageSofti appends `-a` to the uninstall command line).
- **Never**
  The uninstall is never silent; the dialog displays.
- **Auto**
  If the **UserInteractionLevel** is set to `Full`, the dialog displays. Otherwise, it does not display.

Values / range:	Always, Auto, Never
Default value:	If not set, default behavior is `Never`

Command Line

Tool:	Installation agent
Example:	`-o UninstallIShieldSilently="Always"`

Registry

Installed by:	Manually
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

UninstallString

Registry

The string to uninstall an application.

Values / range:	String
Default value:	None

Registry

Installed by:	Installation of RayManageSofti on managed devices
User preference:	HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion
Computer preference:	[Registry]\ManageSoft\Launcher\CurrentVersion

UnusedFilePersistence

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
Files will be deleted from the peer cache during cleaning up operations if they have not been accessed for more than this specified number of hours.

This preference also influences when cleanup operations are started. Cleanup operations do not start immediately when a managed device starts up, as this may result in the peer cache being cleaned out on managed devices that have been turned off for long periods. Instead, cleanup operations are started as soon as one of the following is true:

- One hour has elapsed after RayManageSofti requests file downloads
- One quarter of the time specified by this preference has passed since the managed device started up. For example, if this preference is set to 120, cleanup operations will start after the managed device has been active for 30 hours, if they have not been started earlier.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 24 and 8760
Default value:	`120`

Command Line

Tool:	Peer download agent
Example:	`-debug -o UnusedFilePersistence="240"`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

UnusedFileUptime

Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies how long (in minutes) to wait after receiving the first request from the installation agent before starting to look for and delete unused files.

Values / range:	Integer between 0 and 1440
Default value:	`60`
Supported from:	Release 7.8.1

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

UploadPeriod

Command line | Registry

Specifies how often in seconds the application usage agent will upload recorded application usage data to the specified server. The value must be greater than 0 otherwise the default value will be used.

Values / range:	Integer greater than 0
Default value:	`86400`

Command Line

Tool:	Application usage agent
Example:	`-o UploadPeriod="3600"`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device (computer preference)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

UploadType

Command line | Registry | Project variable

Determines whether the upload agent uploads machine generated files or user generated files. For example, machine inventory or user inventory, all user installation logs or current user installation logs.

Values / range:	Machine, User
Default value:	• `Machine` if running as the SYSTEM user • `User` if running as any other user.

Command Line

Tool:	Upload agent
Example:	`-o UploadType=Machine`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Uploader\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Uploader\CurrentVersion`

Project Variable

Define as:	`UploadType`
Reference as:	`$(UploadType)`

UsageDirectory

Registry

Specifies the directory under which a cache is created for application usage data before it is uploaded to the administration server.

Be aware:
This preference should not be changed by the user as it is set during the installation.

Values / range:	Valid location

Default value:	**Windows devices:**
	`$(CommonAppDataFolder)\ManageSoft Corp\ManageSoft\Usage Agent\UsageData`
	Non-Windows devices:
	`$(CommonAppDataFolder)/usageagent/usagedata`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device (computer preference)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

UseAddRemove

Command line | Registry

This preference applies only for Windows managed devices.
When set to `True`, RayManageSofti records application usage data for applications that are detected from **Add/Remove Programs**.

When set to `False`, RayManageSofti does not use **Add/Remove Programs** to detect applications.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Application usage agent
Example:	`-o UseAddRemove=True`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device (computer preference)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

UseManualMapper

Command line | Registry

When set to `True`, RayManageSofti records application usage data for applications that are detected from the Manual Mapper registry keys.

When set to `False`, RayManageSofti does not use the Manual Mapper registry keys to detect applications.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Application usage agent
Example:	`-o UseManualMapper=True`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device (computer preference)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

UseMGS

Command line | Registry

This preference applies only for Windows devices.
When set to `True`, RayManageSofti records application usage data for applications that are detected from the RayManageSofti application cache.

When set to `False`, RayManageSofti does not use the RayManageSofti application cache to detect applications.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Application usage agent
Example:	`-o UseMGS=False`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device (computer preference)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

UseMSI

Command line | Registry

When set to `True`, RayManageSofti records application usage data for applications that are detected in the native. package format (MSI, RPM, or PKG).

When set to `False`, RayManageSofti does not use the native package format when detecting applications.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Application usage agent
Example:	`-o UseMSI=False`

Registry

Installed by:	Installation of RayManageSofti usage agent on a managed device (computer preference)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

UserHardware

Command line | Registry

Allows you to track hardware either using Windows Management Instrumentation (WMI) or native APIs. If WMI is available, it is used for tracking.
This preference is only effective when running in the user context. To track hardware in the machine context, use **Hardware**.

When set to `True`, allows the tracking of hardware inventory. When set to `False`, does not track hardware inventory.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Inventory agent
Example:	`-o UserHardware=True`

Registry

Installed by:	RayManageSofti internals or manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	N/A - use **Hardware**

UserInteractionLevel (Installation Agent)

Command line | Registry | Project variable

Previously defined as **UILevel**. Depending on the value, of this entry, some dialogs of the RayManageSofti installation agent are suppressed.

The following options can be set:

- **Full:** RayManageSofti installation activities operate in full interactive mode. The user has full control over an application's installation options, and will see all dialogs during the download, installation and uninstall phases.
- **Auto:** RayManageSofti installation activities are fully displayed, but no user interaction is required unless an error occurs. Installation proceeds automatically, using the default install values.
- **Quiet:** RayManageSofti is not displayed during operation, and no user feedback or interaction is available. Do not use this mode without the approval of your RayManageSofti administrator.
- **Status**: Only status dialogs are displayed (for example, progress dialogs).

Values / range:	Full, Auto, Quiet, Status
Default value:	`Full`

Command Line

Tool:	Installation agent
Example:	`-o UserInteractionLevel=Quiet`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Launcher\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Launcher\CurrentVersion` • `[Registry]\ManageSoft\Common`

Project Variable

Define as:	`UserInteractionLevel`
Reference as:	`$(UserInteractionLevel)`

UserInteractionLevel (Adoption Agent)

Command line | Project variable

Depending on the value, of this entry, some dialogs of the RayManageSofti adoption agent (used to install RayManageSofti for managed devices in the processes described in the *Discovery and Adoption* chapter of *RMS Configuration*) are suppressed.

The following options can be set:

- **Full:** RayManageSofti installation activities operate in full interactive mode. The user has full control over an application's installation options, and will see all dialogs during the download, installation and uninstall phases.
- **Auto:** RayManageSofti installation activities are fully displayed, but no user interaction is required unless an error occurs. Installation proceeds automatically, using the default install values.
- **Quiet:** RayManageSofti is not displayed during operation, and no user feedback or interaction is available. Do not use this mode without the approval of your RayManageSofti administrator.
- **Status**: Only status dialogs are displayed (for example, progress dialogs).

Values / range:	Full, Auto, Quiet, Status
Default value:	`Full`

Command Line

Tool:	Adoption agent
Example:	`-o UserInteractionLevel=Quiet`

Project Variable

Define as:	`UserInteractionLevel`
Reference as:	`$(UserInteractionLevel)`

UserInteractionLevel (Inventory Agent)

Command line | Registry

The user interaction method of the RayManageSofti inventory agent. Possible values are:

- **Full:** RayManageSofti inventory activities operate in full interactive mode
- **Auto:** When **ShowIcon (inventory agent)** is `True`, the RayManageSofti icon displays during inventory activities. The user is able to double-click the icon to access the RayManageSofti user interface. When **ShowIcon (inventory agent)** is `False`, a progress bar displays during inventory activities
- **Quiet**: RayManageSofti is not displayed during operations, and no user feedback or interaction is available
- **Status:** Only status dialogs are displayed (for example, progress dialogs).

Values / range:	Full, Auto, Quiet, Status
Default value:	`Status`

Command Line

Tool:	Inventory agent
Example:	`-o UserInteractionLevel=Quiet`

Registry

Installed by:	Manual configuration
User preference:	In order of precedence: • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Tracker\CurrentVersion` • `HKEY_CURRENT_USER\Software\ManageSoft Corp\ ManageSoft \Common`
Computer preference:	In order of precedence: • `[Registry]\ManageSoft\Tracker\CurrentVersion` • `[Registry]\ManageSoft\Common`

UserInventoryDirectory

Command line | Registry

The location for the user inventories on the managed device.

Values / range:	Valid location
Default value:	`$(AppDataFolder)\ManageSoft Corp\ManageSoft\Tracker\Inventories`

Command Line

Tool:	Inventory agent
Example:	`-o UserInventoryDirectory=C:\ManageSoft Corp\ManageSoft\Tracker\Inventories`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

UserZeroTouchDirectory

Command Line

In case of a remote call the location is used for the user inventories.

Values / range:	Valid location
Default Value:	`$(AppDataFolder)\ManageSoft Corp\ManageSoft\Tracker\ZeroTouch`

Command Line

Tool:	Inventory Agent
Example:	`-o UserZeroTouchDirectory=C:\ManageSoft Corp\ManageSoft\Tracker \ZeroTouch`

The default entry can be changed when calling the inventory agent.

UserLogonDomain

Registry | Project variable

Domain name of the user.
For managed devices running Windows 2000 or later, this is automatically configured during adoption of the managed device

Values / range:	The canonical domain name of the user. Read-only preference.
Default value:	The default is the value retrieved from Windows.
Example value:	`mycompany.com`

Registry

Installed by:	Windows 2000 or later
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Common`
Computer preference:	`[Registry]\ManageSoft\Common`

Project Variable

Define as:	Pre-defined on operating systems supporting Active Directory, such as Windows 2000.
Reference as:	`$(UserLogonDomain)`

UserPolicyDirectory

Registry

The location in which to store active user policies.

Values / range:	Valid folder and path
Default value:	`$(AppDataFolder)\ManageSoft Corp\ManageSoft\ Policy Client\Policies\Merged\User`
Example value:	`C:\MyPolicies\User`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Policy Client\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Policy Client\CurrentVersion`

UserPolicyPackageDirectory

Registry

The location where package information associated with user policy is cached.

Values / range:	Valid folder and path
Default value:	`$(AppDataFolder)\ManageSoft Corp\ManageSoft\ Policy Client\Packages`
Example value:	`C:\MyPolicies\Packages`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft\Policy Client\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Policy Client\CurrentVersion`

UserProcessesOnly

Command line | Registry

When set to `True`, RayManageSofti only records application usage data for applications run by users other than `SYSTEM` (or `root` in non-Windows environments).

When set to `False`, RayManageSofti records application usage data for all applications.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Application usage agent
Example:	`-o UserProcessesOnly=False`

Registry

Installed by:	Installation of application usage agent on a managed device (computer preference)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Usage Agent\CurrentVersion`

UseTrustDatabase

Command line | Registry | Project variable

When the installation agent is deciding whether a package may be installed, this setting specifies whether it takes account of the distribution location from which files are collected.

If `True`, the installation agent will check whether the distribution location from which a RayManageSofti catalog (`.osd` file) is collected is a trusted location. Note that if **VerifyTrustOrSign** is `True`, this setting will be ignored. A related setting for user preferences in the registry will override the machine settings unless the machine settings are locked. See *Fixing managed device preferences*.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o UseTrustDatabase=True`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`UseTrustDatabase`
Reference as:	`$(UseTrustDatabase)`

VerifyCatalogSigned

Command line | Registry | Project variable

When set to `True`, the installation agent uses Authenticode to check the digital signature referenced in the implementation archive before installing a package. RayManageSofti implementation archives have the extension `.ndc`.
When set to `False`, RayManageSofti does not check the digital signature before installing a package.

Note:
If `$(VerifyTrustOrSign)` is `True`, this setting is ignored.

A related setting for user preferences in the registry will override the machine settings unless the machine settings are locked. See *Fixing managed device preferences*.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o VerifyCatalogSigned=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`VerifyCatalogSigned`
Reference as:	`$(VerifyCatalogSigned)`

VerifyFilesSigned

Command line | Registry | Project variable

When set to `True`, RayManageSofti checks for a valid Authenticode digital signature in executable files that it downloads before it installs them.

When set to `False`, RayManageSofti does not check executable files for a valid digital signature.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o VerifyFilesSigned=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`VerifyFilesSigned`
Reference as:	`$(VerifyFilesSigned)`

VersionInfo

Command line | Registry

When set to `True`, RayManageSofti includes file version header information in the inventory.

When set to `False`, RayManageSofti does not include file version header information in the inventory.

Values / range:	Boolean
Default value:	`True`

Command Line

Tool:	Inventory agent
Example:	`-o VersionInfo=False`

Registry

Installed by:	Manual configuration
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Tracker\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Tracker\CurrentVersion`

VirusScan

Command line | Registry | Project variable

When set to `True`, RayManageSofti scans downloaded files for viruses before installation. The **VirusScanCommand** preference defines the virus checking mechanism used.

When set to `False`, RayManageSofti does not scan files for viruses.

Values / range:	Boolean
Default value:	`False`

Command Line

Tool:	Installation agent
Example:	`-o VirusScan=True`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`VirusScan`
Reference as:	`$(VirusScan)`

VirusScanCommand

Command line | Registry | Project variable

Determines the virus scan application used and the location of its binaries. RayManageSofti uses this value to run the virus scanning application. The value should either be enclosed in quotes or use short file names for folder names that are long or contain spaces.

This option is only available when **VirusScan** is set to `True`.

Values / range:	Valid executable file and path
Default value:	`No default`
Example value:	`C:\PROGRA~1\Vet\vet.exe`

Command Line

Tool:	Installation agent
Example:	`-o VirusScanCommand=C:\PROGRA~1\Vet\vet.exe`

Registry

Installed by:	Installation of RayManageSofti on a managed device (computer preference)
User preference:	`HKEY_CURRENT_USER\Software\ManageSoft Corp\ManageSoft \Launcher\CurrentVersion`
Computer preference:	`[Registry]\ManageSoft\Launcher\CurrentVersion`

Project Variable

Define as:	`VirusScanCommand`
Reference as:	`$(VirusScanCommand)`

VulnerabilityScripts

Security Manager only
Command line | Registry

This preference specifies the names of security scripts to be run after managed devices have been scanned for vulnerabilities. These scripts must reside in the location specified by **VulnerabilityScriptsDir**. Scripts will only be run if **RunVulnerabilityScripts** is `True`.

Values / range:	A comma-separated list of scripts
Default value:	`No default`
Example value:	`ReadScan.exe, IMScanner.exe`

Command Line

Tool:	Security agent
Example:	`-o VulnerabilityScripts=ScanIt.exe`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

VulnerabilityScriptsDir

Security Manager only
Command line | Registry

This preference specifies the location of security scripts to be run after managed devices have been scanned for vulnerabilities. The list of scripts to run is specified by **VulnerabilityScripts**. Scripts will only be run if **RunVulnerabilityScripts** is `True`.

Values / range:	String (valid folder location)
Default value:	`$(ScriptDir)\SecurityPatchManagement VulnerabilitiesVulnerabilityScripts`
Example value:	`C:\Program Files\SecurityScans`

Command Line

Tool:	Security agent
Example:	`-o C:\data\scanners`

Registry

Installed by:	Security Manager internals or manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Security Agent\CurrentVersion`

WANAveragingTime

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It is used in conjunction with **WANMaxRate** to limit bandwidth used for downloading files from the distribution server. It specifies the averaging period (in minutes) used to smooth the estimate of transfers to and from the distribution server. See *WANMaxRate* for details about how these preferences are used together.
Increasing the value of this preference means that the estimate takes longer to change as the actual transfer rate changes. In normal use, you will not need to change the value of this preference.

RayManageSofti retrieves the value for this preference from the registry every five seconds. You do not need to restart RayManageSofti on managed devices after changing the value of this preference.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line.*

Values / range:	Integer between 1 - 60
Default value:	`10`

Command Line

Tool:	Peer download agent
Example:	`-debug -o WANAveragingTime=10`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

WANMaxRate

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the maximum allowable rate (in bytes per second) for transfers from distribution servers across all peers on this subnet. **WANMaxRate** is used in conjunction with **WANAveragingTime** to limit bandwidth used between the distribution server and the group of managed devices downloading files.

RayManageSofti calculates the sum of file transfers that have occurred between the distribution server and the peer group of managed devices to which this device belongs. Since transfers occur in blocks, not as a continuous stream, RayManageSofti smooths out the variation in transfer rates using the **WANAveragingTime** and a simple exponential decay algorithm. The result is an estimate of the transfer rate. Transfer rates will be decreased if the estimated rate exceeds the specified **WANMaxRate**, and increased if they are below the specified **WANMaxRate**. (Transfers can creep up to the **WANMaxRate**, but will drop back rapidly when the estimated rate is greater than **WANMaxRate**.)
RayManageSofti retrieves the value for this preference from the registry every five seconds. You do not need to restart RayManageSofti on managed devices after changing the value of this preference.

For a list of all peer-to-peer preferences, see *Peer download agent preferences set on the command line*.

Values / range:	Integer between 1024 - 134217728
Default value:	`16777216`

Command Line

Tool:	Peer download agent
Example:	`-debug -o WANMaxRate=2048`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

WANProgressInterval

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the frequency (in seconds) with which to send progress announcements to peer managed devices about file downloads from the distribution server. This preference only controls the frequency of progress messages while the download operation is in progress. The message sent at the completion of a download is sent immediately.

Values / range:	Integer between 1 - 90
Default value:	`10`

Command Line

Tool:	Peer download agent
Example:	`-debug -o WANProgressInterval=30`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

WANRetries

Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies how many times a failed WAN download is retried immediately, from each distribution server, at each WAN retry interval.

Values / range:	Integer between 0 - 10
Default value:	`1`
Supported from:	Release 7.8.1

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

WANRetryDuration

Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies how long (in minutes) to continue to allow a file to be retried for download since it was last requested by the installation agent.

Values / range:	Integer between 0 - 43200
Default value:	`1440`
Supported from:	Release 7.8.1

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

WANRetryInterval

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).

It specifies how long (in seconds) after a failed WAN download to retry the download.

Values / range:	Integer between 10 - 86400
Default value:	`300`

Command Line

Tool:	Peer download agent
Example:	`-debug -o WANRetryInterval=30`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

WANSearchCurrency

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).

If peer-to-peer file sharing is enabled, files that are available from a peer managed device will always be downloaded from the peer rather than from the distribution server. See the *Package installation system flow* chapter of *RMS System Reference* for details about how RayManageSofti chooses the location from which to download files.

This preference specifies how frequently a device will ask its peers for a file. For example, if managed device BOSTON asks its peers for file `MyDownload.txt` and **WANSearchCurrency** is set to 30, the managed device will not reissue a request for `MyDownload.txt` within 30 seconds of its original request. Instead of requesting the file again from its peers within the interval specified by **WANSearchCurrency**, BOSTON would download `MyDownload.txt` from the closest distribution server.

Avoiding Lock Conditions

Consider the interactions between **ParentConnectionWindows**, **PeerConnectionWindows** and **WANSearchCurrency**, and make sure you do not configure these settings in such a way that you cause a lock condition.

A lock condition can occur, for example, if:

- **ParentConnectionWindows** allows file downloads between 3-6am
- **PeerConnectionWindows** allows file downloads between 10am-12noon
- **WANSearchCurrency** is set to 4 hours.

Under this configuration, the peer download agent might:

- Request a file from peers at 12 noon, but fail to obtain it
- Do nothing until the start of the permitted parent connection time window at 3am.

At 3am, it checks **WANSearchCurrency**, and finds that it must perform a peer search before downloading the file from a parent managed device, since the time interval since it last conducted a peer search for the file is greater than 4.

The peer download agent must then wait until the next permitted peer connection time window to request the file.

Values / range:	Integer between 1 - 600
Default value:	`30`

Command Line

Tool:	Peer download agent
Example:	`-debug -o WANSearchCurrency=10`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

WANTimeout

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the time (in seconds) after which to abort stalled transfers of files from distribution servers.

Values / range:	Integer between 1 - 600
Default value:	`30`

Command Line

Tool:	Peer download agent
Example:	`-debug -o WANTimeout=10`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

WANTransferLimit

Command line | Registry

This preference is only used if managed devices are configured for peer-to-peer file sharing (**AllowPeerToPeer** is `True`).
It specifies the maximum number of managed devices that can simultaneously download files from a distribution location.

Before downloading any files from a distribution location, the managed device checks to see how many peers on this subnet are currently downloading across the network. This managed device will not start downloading across the network if the number of managed devices currently downloading is equal to or greater than the **WANTransferLimit** setting for this device.

Values / range:	Integer between 1 - 1000
Default value:	`3`

Command Line

Tool:	Peer download agent
Example:	`-o WANTransferLimit=10`

Registry

Installed by:	Manual configuration
User preference:	Not available
Computer preference:	`[Registry]\ManageSoft\Downloader`

Preferences on UNIX and Macintosh Managed Devices

General behavior of RayManageSofti on Macintosh and UNIX managed devices is configured by preference settings in a number of locations:

- Initial settings for these non-Windows platforms are set in the `mgssetup.sh` shell script, run automatically as the device is adopted. This includes the defaults for user and group IDs, the domain in which the device is to be reported, the relevant policy path, and upload and download locations. These values are written during adoption into `/var/tmp/mgsft_rollout_response` on the managed device, and they will subsequently be written into the file next described, unless overwritten by other changes.
- Many preferences are stored in the `config.ini` file stored in `$CommonAppDataFolder/etc` (by default `/var/opt/managesoft/etc`). This file acts as a "virtual registry": that is, a repository on non-Windows platforms for settings that, on Windows platforms, are stored in the registry. Initially, this file will contain a few factory default settings, plus the values imported from the `mgsft_rollout_response` file described above. Thereafter, this is the file that is automatically updated (and potentially, extended with additional preferences) when you distribute a managed device settings package that is applicable (by policy) to the non-Windows device. Some client-side RayManageSofti agents (such as the schedule agent and the usage agent) read their preferences directly from this file.
- Several general preferences for Linux, Solaris, and OS X agents are configured in `managesoft.xconf`. In some cases, these overlap with preferences you may set at your administration server and distribute as settings packages (which, as described above, are locally stored in the `config.ini` file). Some of these overlapping cases are automatically synchronized from the updated `config.ini` file into the `managesoft.xconf` file after each update. These synchronized settings include the following:

config.ini	managesoft.xconf
`ComputerDomain`	domain
`MachineID`	machineid
`netselector`	preferences for finding the nearest failover server

Be aware:
Automatic synchronization occurs only unidirectionally from `config.ini` into `managesoft.xconf`, and only on device adoption and after preference updates delivered as settings packages. Be very careful about manually editing `xconf` files, as your changes may be overwritten by future settings updates.

- Bandwidth management settings are stored in `throttling.xconf`. This file is not synchronized with `config.ini`. If you wish to configure these settings across a group of Linux, Solaris, and OS X devices, you can distribute the `xconf` file in a normal deployment package.
- Upload from and download to the managed device also use four more `xconf` files: `download.xconf`, `lastdownload.xconf`, `upload.xconf`, and `lastupload.xconf`. For information on how Linux, Solaris, and OS X managed devices determine which distribution server to use for a download or upload operation, and how you can configure these selections, refer to the *Selecting distribution servers* chapter of *RMS Configuration*.

All these post-adoption preferences files are installed in `$CommonAppDataFolder/etc`, or by default in `/var/opt/managesoft/etc`. They are in XML format, and you can edit them in a text editor of your choice.

Preference settings stored in `$CommonAppDataFolder/etc/config.ini` are managed by preparing managed device settings packages on your administration server. That facility allows for customizing values for different operating systems (as well as for 32/64 bit platforms and different human language interfaces).

How to Configure Preferences

As a general rule, use managed device settings packages to apply preference changes to non-Windows managed devices.

The following sections cover the exceptions to this rule, and describe how to configure bandwidth preferences and some sections and tags in `managesoft.xconf` for Linux, Solaris, and OS X clients. You may notice sections and tags in `managesoft.xconf` that are not described, but you are unlikely to need to edit them. Your Raynet consultant will advise you on editing them if necessary.

See:

- *Configuring the policy file that is applied*
- *Configuring the downloader's "last modified" check*
- *Configuring the locations for downloaded files*
- *Configuring HTTP (or HTTPS) proxy settings*
- *Configuring SSL provider information*
- *Configuring RayManageSofti to use encrypted FTP passwords*
- *Configuring the domain*
- *Configuring bandwidth management preferences*

Configuring the Policy File that is Applied

There are three ways that policy files can be applied to UNIX and OS X devices:

- If you have registered your non-Windows devices in Active Directory, policy files for UNIX and OS X managed devices are created on the administration server, and distributed through the distribution hierarchy that is used for deploying applications. This process is exactly the same as for Windows devices, so that you can have a single process for all managed devices. (See the *Deployment policies* chapter of the *RMS Software Deployment* for details about this process.) The policy filename contains the managed device name.
- Where your non-Windows devices are not registered in Active Directory, you may use RayManageSofti policy attached to a RayManageSofti domain. Once again, these policy files are calculated on the administration server, and distributed to the non-Windows managed devices. Each of these policy filenames also contains the name of the managed device.
- You may also want to manually set a particular policy file name, most often for testing purposes. You may apply this manual change from the administration server to a set of managed devices, or you can locally edit settings on a particular device under test. These processes are described here.

If you want to apply policy from a file other than the default, it is possible to specify the alternative policy by changing the machine ID in a preference setting. For Linux, Solaris, and OS X clients, this change will be synchronized from the updated `config.ini` file to the `<managesoft-configuration/>` section of `managesoft.xconf`. Note that when specifying the settings package on your Administration Server, you should be sure to set the MachineId preference in the Common settings, as this is the only setting of this preference that is synchronized onto Linux, Solaris, and OS X platforms (individual settings of the preference for other Windows clients are not transferred to Linux, Solaris, and OS X clients). If you do not want this setting to be passed on to all managed devices (for example, if you want a special policy file only for Linux, Solaris, and OS X

computers), be sure to make this preference setting platform-specific.

Be aware:
Changing the common machine ID will also change the name of this managed device in RayManageSofti reporting, inventory, and the like. It does not modify the actual machine name used outside RayManageSofti in network settings, and so on.

For temporary use (such as during debugging or testing) on Linux, Solaris, and OS X, you can make a local change to the applied policy file by editing the `managesoft.xconf` file as follows (always remembering that your change may later be over-written by a centrally-deployed settings package):

1. In a text editor of your choice, open `managesoft.xconf` for editing.
2. Locate the `<!-- machineid="" -->` line.
3. Uncomment it, and add the base name of the file from which policy should be applied. For example, if you want this managed device to install policy from a file `fluffy.npl`, specify `machineid="fluffy"`.
4. Move the trailing `>` from the end of the previous line to the end of the machineid line, so that it ends the `<managesoft-configuration ...` tag (with the machineid line now included inside it).
5. Save and close the file.

Configuring the Downloader's "last modified" Check

By default, before downloading any file, the downloader checks the last modified date of the file at the download location against the last modified date of any local copy. If the local copy is up-to-date, the file is not downloaded again.

On Linux, Solaris, and OS X, you can turn off this checking, so that the downloader always downloads files. To do so, in the `<downloader ...>` section of `managesoft.xconf`, set `<lastmodified enable="false"/>`.

Be aware:
This setting is not synchronized from `config.ini` to `managesoft.xconf`. If you wish to set this preference centrally for Linux, Solaris, and OS X managed devices, you need to deploy a version of `managesoft.xconf` with the appropriate settings to the desired computers.

```
<!--+
| Downloader
|
| The Downloader is responsible for end to end file retrieval.
| Its task is to retrieve a file from a remote location, and
| ensure that the file is downloaded completely, handling
| non-fatal error conditions as they arise.
+-->
<downloader id="default" logger="downloader" activation="inline">
<lastmodified enabled="false"/>
</downloader>
```

Configuring the Locations for Downloaded Files

On Linux, Solaris, and OS X, you can configure the locations to which **.rpm, .osd, .ndc, .nds** and **Solaris** packages are downloaded and hosted until they are installed. To do so, edit the `<live-area ...>` tag in the `<managesoft- configuration ...>` section of `managesoft.xconf`.

Be aware:
This setting is not synchronized from `config.ini` to `managesoft.xconf`. If you wish to set this preference centrally for Linux, Solaris, and OS X managed devices, you need to deploy a version of `managesoft.xconf` with the appropriate settings to the desired computers.

```
<managesoft-configuration
        domain="boston.tmnis.com"
        live-area="/var/opt/managesoft/live"
        staging-area="/var/opt/managesoft/cache/package"
        file-staging-area="/var/opt/managesoft/cache"
        atomic-client="true"
        proxy-type="none"
        system="sunossystem"
        username="mgsft"
        password="random"
        keep-transformed="false"
        cache-files="false"
        byte-level-update-enabled="true"
```

Configuring Byte-level Differencing

To enable byte-level differencing on each Linux, Solaris, and OS X managed device (as described in the *Configuring byte-level differentiation* chapter of *RMS Configuration*), edit the `<byte-level-update-enabled ...>` attribute of the `<managesoft-configuration ...>` element in `managesoft.xconf`.

Be aware:
This setting is not synchronized from `config.ini` to `managesoft.xconf`. If you wish to set this preference centrally for Linux, Solaris, and OS X managed devices, you need to deploy a version of `managesoft.xconf` with the appropriate settings to the desired computers.

```
<managesoft-configuration
        domain="boston.tmnis.com"
        live-area="/var/opt/managesoft/live"
        staging-area="/var/opt/managesoft/cache/package"
        file-staging-area="/var/opt/managesoft/cache/file"
        atomic-client="true"
        proxy-type="none"
        system="sunossystem"
        username="mgsft"
        password="random"
        keep-transformed="false"
        cache-files="true"
        byte-level-update-enabled="true"
```

To enable byte-level differencing, byte-level-update-enabled must be set to `True`. The cache-files parameter must also be set to `True`.

Configuring HTTP (or HTTPS) Proxy Settings

If you use an HTTP (or HTTPS) proxy for RayManageSofti file uploads or downloads for Linux, Solaris, and OS X, its details are recorded in the `<httpclient-source ...>` section of `managesoft.xconf`:

```
<!--+
| HTTP Source protocol
```

```
|
| HTTP source protocol supports http:/// URIs for manipulating
| HTTP resources on remote servers. This protocol may be used
| for reading and writing.
|
| The HTTP Source protocol also supports HTTP proxies. To set
| a proxy add the following parameters to the configuration
| below:
|
| <parameter name="proxy.host" value="proxy"/>
| <parameter name="proxy.port" value="1234"/>
|
| It is possible to configure a read timeout in milliseconds.
| The default below is 30 seconds.
+-->
<httpclient-source id="http" logger="resolver.http">
    <!--
        <parameter name="proxy.host" value="webproxy.local"/>
        <parameter name="proxy.port" value="3128"/>
    -->
        <parameter name="timeout.read" value="30000"/>
</httpclient-source>
```

You can edit these details if necessary.

Be aware:
One proxy can be specified, and is used globally. You cannot specify different proxies for different upload/download locations.

The `timeout.read` parameter determines how long a download connection will be maintained if data is not passing between a managed device and a server, in milliseconds.

Configuring SSL Provider Information

If you use HTTPS for RayManageSofti file uploads or downloads, the HTTPS source protocol requires an SSL provider implementation. For Linux, Solaris, and OS X, its details are recorded in the <httpsclient-source ...> section of managesoft.xconf:

```
<!--+
| HTTPS Source protocol
|
| HTTPS source protocol supports https:/// URIs for manipulating
| HTTPS resources on remote servers. This protocol may be used
| for reading and writing.
|
| The HTTPS source protocol requires a valid SSL provider
| implementation. JRE's 1.4 and above include one by default,
| for JRE's 1.3 a separate package is available for free from
| http://java.sun.com/products/jsse/.
|
| If you are using JRE 1.3.x please ensure you have installed
| and configured JSSE in your environment before attempting to
| use this protocol handler.
|
| It's also possible to optionally specify a custom JSSE
| provider if you wish, simply uncomment and fill in the example
| shown below.
|
```

```
| You may also optionally specify a custom secure protocol
| socket factory that will be used to create SSL sockets.
| This may be useful if wish to employ the use of an existing
| trust manager, etc. Your socket factory must implement the
| interface:
| org.apache.commons.httpclient.protocol.SecureProtocolSocketFactory
| (see some examples below).
|
| The HTTPS Source protocol also supports HTTPS proxies. To set
| a proxy add the following parameters to the configuration
| below:
|
| <parameter name="proxy.host" value="proxy"/>
| <parameter name="proxy.port" value="1234"/>
|
+-->
<httpsclient-source id="https" logger="resolver.https">
<!-- Optional JSSE provider specification
<parameter name="provider" value="some.jsse.Provider"/ -->
<!-- Note, if you wish to use self-signed/untrusted
certificates apply the appropriate custom socket factory.
To enforce CA verified certificates, uncomment all -->
<!-- Example for IBM JRE 1.4.1
<parameter name="socket-factory"
value="com.managesoft.networking.ssl.ibmjsse
.EasySSLProtocolSocketFac tory"/ -->
<!-- Example for all JRE 1.3.x vendors
<parameter name="socket-factory"
value="com.managesoft.networking.ssl.sunjsse
.EasySSLProtocolSocketFac tory"/ -->
<!-- The comment token below is used by managesoft-configure to
configure the socket-factory setting. -->
<!-- jsse -->
</httpsclient-source>
```

You can edit the details about your SSL provider implementation if necessary.

Configuring RayManageSofti to Use Encrypted FTP Passwords

If you want to use encrypted FTP passwords on Linux, Solaris, and OS X, you must have a supported Java environment (see the section on installing RayManageSofti on a UNIX managed device in the *RMS Implementation*).

To configure RayManageSofti to use encrypted FTP passwords on Linux, Solaris, and OS X:

1. In the text editor of your choice, open `managesoft.xconf` for editing.
2. Change the `<ftp-source ..../>` tag to `<des-ftp-source ..../>`.
3. Save and close the file.

FTP URLs that contain passwords will be treated as encrypted. They will be decrypted before use.
If you attempt to use encrypted FTP passwords without performing this configuration, a password exception will occur at runtime.

Configuring the Domain

At installation time, you specify whether your RayManageSofti implementation is a single or multi-domain environment. You can also specify the domain in which this UNIX computer resides. (See the section on installing RayManageSofti on a UNIX managed device in the *RMS Implementation* for details about the installation process.)

Domain details are stored in two files on non-Windows managed devices, both of which are located in `$CommonAppDataFolder/etc` (by default `/var/opt/managesoft/etc`):

- `managesoft.xconf` (in the `<managesoft-configuration ... >` section) for Linux, Solaris, and OS X
- `config.ini` (in the `[ManageSoft\Common]` section).

In production, there are three ways that you can update the reported domain for a non-Windows managed device:

- You can centrally prepare a device settings package, if necessary with specialized settings for different operating systems, in which you configure the preference `ComputerDomain`. Once the settings package is distributed to Linux, Solaris, and OS X managed devices, this value is written first into `config.ini` and then synchronized into `managesoft.xconf`.
- Locally on the managed device, you can run the script managesoft-configure that resides in `$InstallDir/bin` (by default, `/opt/managesoft/bin/managesoft-configure`). For Linux, Solaris, and OS X, this updates both `config.ini` and `managesoft.xconf`.
- You could manually edit the configuration files `config.ini` and (on Linux, Solaris, and OS X) `managesoft.xconf`.

Be aware:
Automatic synchronization occurs only unidirectionally from `config.ini` into `managesoft.xconf`, and only on device adoption and after preference updates delivered as settings packages. Be careful about manually editing `xconf` files, as your changes may be overwritten by future settings updates.

Configuring Inventory Agent Preferences

For all clients, inventory agent preferences are initially set in `config.ini` and can be updated by distributing a managed device settings package.

You can read general details about how the inventory system works in the *Inventory system flow* chapter of the *RMS System Reference*.

Where Inventory Data is Gathered on Linux Systems

On Linux systems, RayManageSofti can collect software inventory data from:

- RPM
- Its own cache

RayManageSofti can collect hardware inventory data from:

- SBLIM
- Its internal CIM-compliant hardware tracking

Where Inventory Data is Gathered on Solaris Systems

On Solaris systems, RayManageSofti can collect software inventory data from:

- Solaris
- Its own cache

RayManageSofti can collect hardware inventory data from its internal CIM-compliant hardware tracking.

Where Inventory Data is Gathered on Macintosh Systems

On Macintosh systems, RayManageSofti can collect software inventory data from:

- The Apple System Profiler
- Its own cache

RayManageSofti can collect hardware inventory data from:

- The Apple System Profiler
- Its internal CIM-compliant hardware tracking

Configuring Bandwidth Management Preferences

If bandwidth is limited, for example on a dialup connection, it can be useful to restrict the amount of bandwidth that RayManageSofti can use, so that users can continue with other tasks such as reading email or accessing websites while RayManageSofti is downloading software files. You may also wish to nominate the amount of bandwidth that RayManageSofti can use on a high speed connection. For Linux, Solaris, and OS X, you can specify bandwidth management settings in the `throttling.xconf` file (installed in `$CommonAppDataFolder/etc/`, or by default in `/var/opt/managesoft/etc`). You can edit this XML format file in a text editor of your choice.

To configure bandwidth settings:

1. In the text editor of your choice, open `throttling.xconf` for editing.
2. Locate the `<ThrottlingManager>` section.
3. Uncomment one `<ThrottlingAlgorithm class= ...>` specification by moving it outside the <!-- and --> comment markers. Only one algorithm specification should be uncommented.
   The **ThrottlingAlgorithm** setting, in conjunction with the **BandwidthLookup** setting (discussed next) determines how much bandwidth can be used over a time period. The available algorithms are:
   - **HardLimitAlgorithm**, which specifies that bandwidth be restricted to a fixed number of bytes per second.
   - **AverageAlgorithm**, which retains the average number of bytes used by RayManageSofti over the past 20 seconds, and uses that number to specify how many bytes RayManageSofti can use. Using this algorithm, if RayManageSofti does not read its full quota of bytes (as specified by the **BandwidthLookup**) over one time unit, it can read more in successive time units to "catch up".
4. Uncommend the **BandwidthLookup** option you want to use by moving it outside the `<!-- and -->` comment markers.
   Only one bandwidth lookup specification should be uncommented.
   The **BandwidthLookup** setting specifies how much bandwidth can be used.
   The available options are:
   - **FixBandwidthLookup** and its `<input-bps value="10000000"/>` and `<output- bps value="10000000"/>`, which specifies the number of bytes per second that can be used for downloads and uploads respectively.
   - **SimpleBandwidthDiscovery** and its `<percentage value="90"/>`, which specifies the maximum percentage of the available bandwidth that can be used for both upload and download.
5. Set the *-bps or percentage values appropriately for your use.
6. Save and close the file.

Deploy the edited file to managed devices whose bandwidth use you want to manage.

RayManageSofti is part of the RaySuite

More information online
www.raynet.de

Raynet GmbH
Technologiepark 20
33100 Paderborn, Germany
T +49 5251 54009-0
F +49 5251 54009-29
info@raynet.de

www.raynet.de