Agent Procedures

Table of Contents:

Introduction

True automated system administration can only be realized if the underlying administration procedures can be encoded as scripts or programs that can run on the managed machines on a regular basis. Such scripts or programs can perform tasks automatically, resulting in more efficient and accurate system administration. For example, a script (also called procedure) can be scheduled to run every day to determine, based on certain criteria, whether or not a disk needs to be defragmented. If a defragmentation is required, the procedure can schedule defragmentation at a convenient time and log the results. A more complex example is to regularly check if a web site is available and is serving up the correct contents. A procedure can be run from a managed computer other than the web server to collect the contents of some preselected pages on the server and verify whether the contents are as expected. Based on the test results, a log entry can be created indicating if everything is ok or not. This log entry is later processed by another procedure that alerts the administrator if necessary.

Kaseya Agent procedures module allows for creation of required administration procedures that are scheduled and executed by deployed agents without end-user intervention. In this Chapter, we will describe how agent procedures are created and deployed. We will also describe Path Deploy Wizard that allows for creation of procedures to deploy non-Microsoft install packages to managed machines and Packager wizard that allows users to create customized installation packages to be deployed on managed machines.

7.1 Manage Procedures

7.1.1 Schedule/Create

The **Schedule / Create** page automates user-defined tasks on managed machines by creating and scheduling agent procedures.

Folder trees
Agent procedures are organized using two folder trees, underneath *Private* and *Shared* cabinets. The following functions can be used to manage objects in these folder trees. As shown in Fig. 7.1, these functions are listed in the header part of this page.

Functions That Are Always Available

Fig. 7.1 shows the Schedule / Create page. The functions always available, no matter where the cursor is, are listed and explained below.

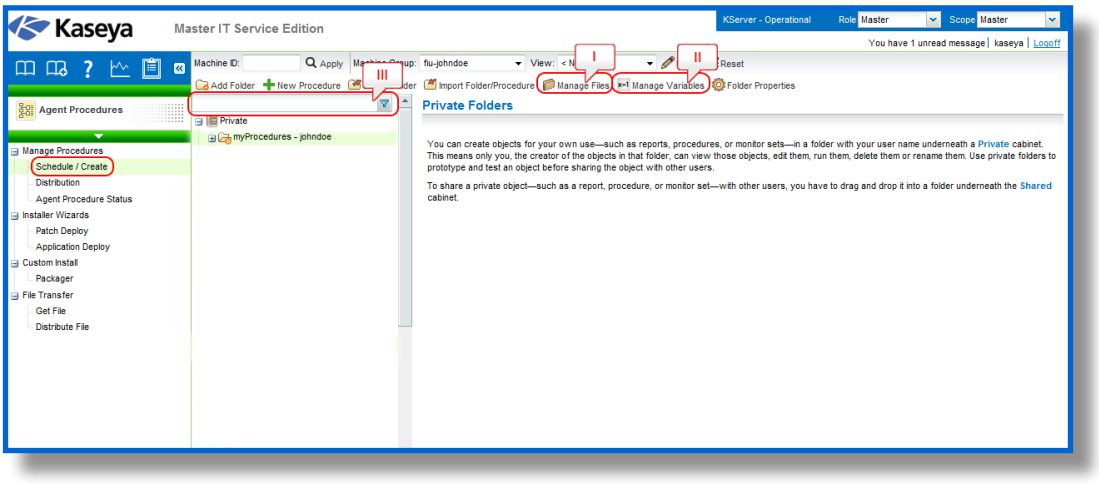


Fig. 7.1: *Schedule / Create Page: Labels point to the functions that are always available.*

I. Manage Files: The *Manage Files Stored on Server* popup window can be used to upload a file and store it on the KServer. You can also list, display and delete files already stored on the KServer. Agent procedures can distribute these files to managed machines using the *Write File* or *Write File in Directory Path* commands (all commands will be explained later in this chapter).

To upload a file:

- Click *Private files* or *Shared files* to select the folder used to store uploaded files. Files stored in the *Private files* folder are not visible to other users.
- As shown in Fig. 7.2, you can click *Browse* to locate files to upload. Then click *Upload* to upload the file to the KServer.

To delete a file stored on the KServer:

- Click *Private files* or *Shared files* to select the folder used to store uploaded files.
- As shown in Fig. 7.2, you can click the delete icon next to a file name to remove the file from the KServer.

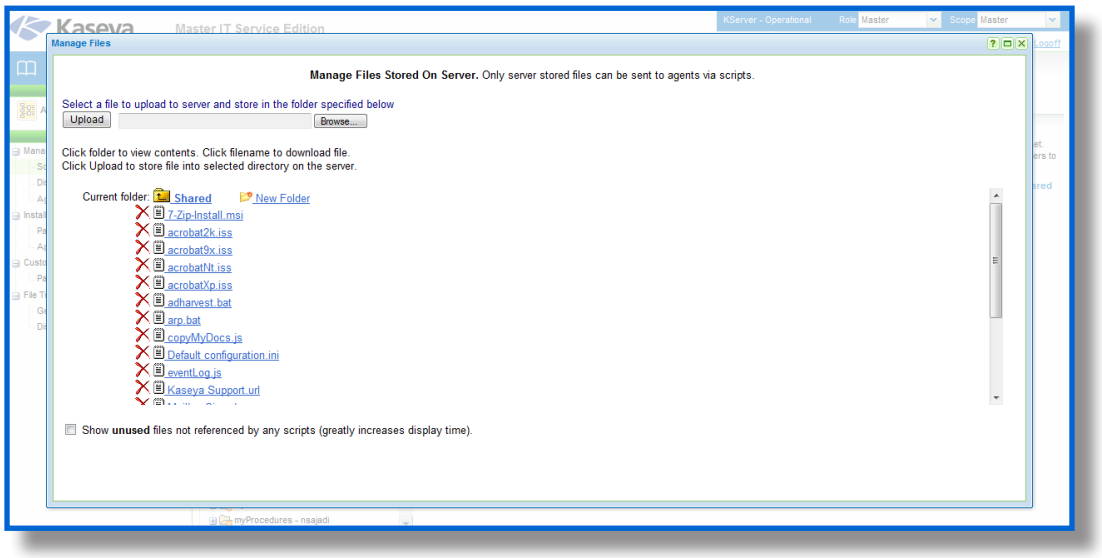
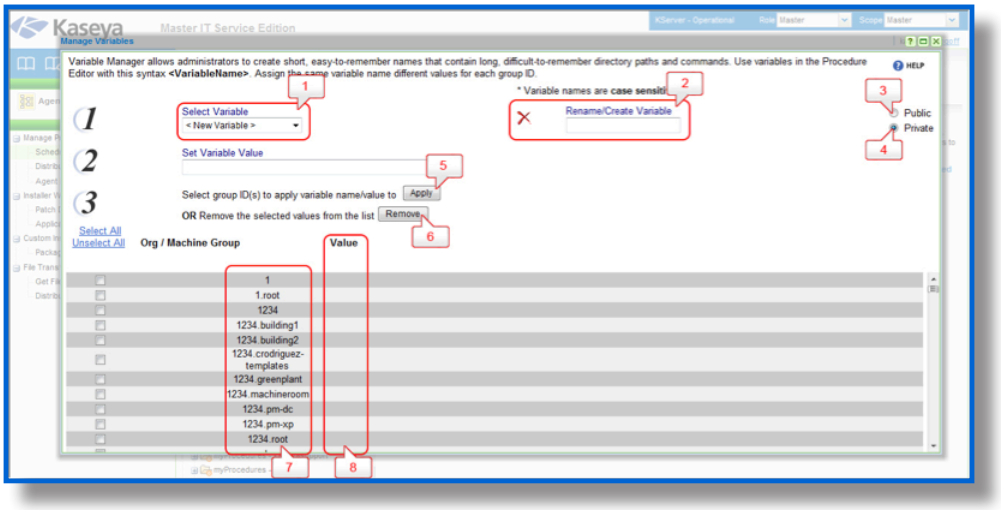


Fig. 7.2: *Manage Files*

II. Manage Variables: The *Variable Manager* enables you to define variables that can be used repeatedly in different agent procedures. You can maintain multiple values for each managed variable, with each value applied to one or more group IDs. Managed variables cannot be re-assigned new values within a procedure. Within a procedure, reference a managed variable by bracketing the variable name with the < and > characters (*e.g.,* <VariableName>).

Fig. 7.3 below shows the Manage variables window. The functions supported on this window are listed and explained below.

Fig. 7.3: *Manage Variables*



1. **Select Variable:** Select a variable name from the drop-down list or select <*New Variable*> to create a new variable. Variable names are case sensitive.

2. **Rename/Create Variable:** Enter a new name for the new variable you are creating or for an existing variable you are renaming. Select the delete icon to delete the entire variable from all groups.

3. **Public:** Selecting the *Public* radio button allows the variable to be used by all users. However, only master role users can create and edit public variables (also called shared variables).

4. **Private:** Selecting the *Private* radio button allows the variable to be used only by the user who created it.

5. **Apply:** Enter the initial value in the *Set Variable Value* box. Then select one or more *Group IDs* and click *Apply*. Empty values are not allowed. Note that you can apply different values to different group IDs, which is a very useful feature.

6. **Remove:** Select one or more group IDs, then click *Delete* to remove the value for this variable from the group IDs it is assigned to.

7. **Group ID:** Displays all group IDs the logged in user is authorized to administer.

8. **Value:** Lists the value of the variable applied to each group ID.

III. Apply Filter: Enter text in the filter edit box, then click the funnel icon to apply filtering to the folder trees. Filtering is case-insensitive. Match occurs if filter text is found anywhere in the folder trees.

Functions That Are Available When a Folder is Selected

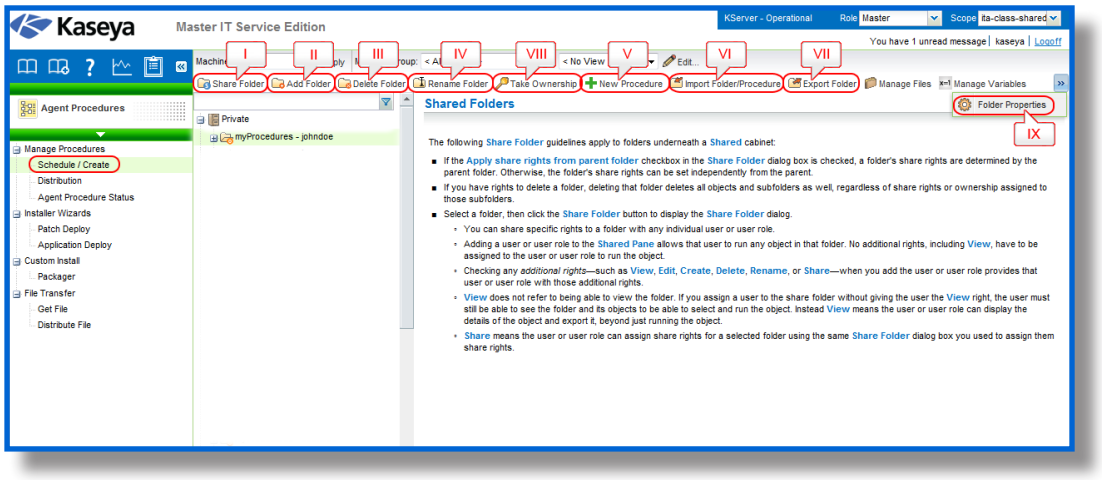


Fig. 7.4: *Schedule / Create Page: Labels point to the functions that are available when a folder is selected.*

Fig. 7.4 shows the Schedule / Create page. The functions are visible to the user when the cursor is on a folder. These functions are listed and explained below.

I. Share Folder: Shares a folder with user roles and individual users. Applies to shared cabinet folders only.

II. Add Folder: Creates a new folder underneath the selected cabinet or folder.

III. Delete Folder: Deletes a selected folder.

IV. Rename Folder: Renames a selected folder.

V. New Procedure - Opens the *Agent Procedure Editor* to create a new procedure in the selected folder of the folder tree. To edit an existing procedure, select the procedure, then click the Edit Procedure button to open the *Agent Procedure Editor*. You can also double-click a procedure to edit it. The outline of the entire agent procedure displays in the left-hand pane of the editor. The parameters for each statement are displayed in the right-hand pane.

Fig. 7.5 below shows the generic view of the Agent Procedure Editor window. The options available on this window are listed and explained below.

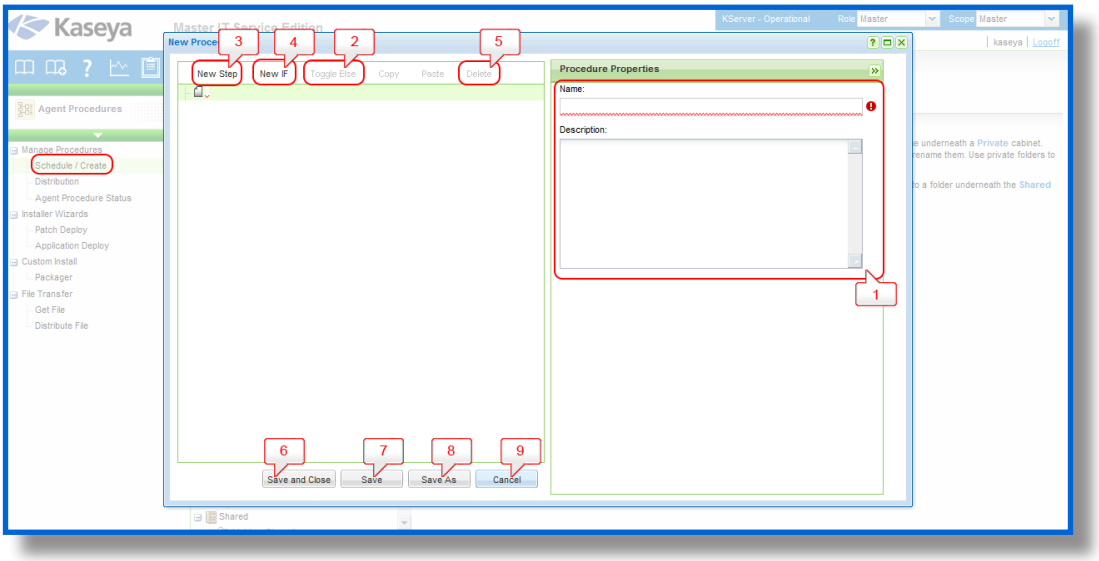


Fig. 7.5: *Agent procedure editor*

1. **Title:** This the first step in the procedure, where you set the name and description of the procedure.

2. **Toggle Else**: Adds or removes the corresponding Else statement for a selected IF statement. Only displays if an IF statement is selected.

3. **New Step**: Creates a step below the currently selected statement.

4. **New IF**: Creates a pair of *IF-Else* statements below the currently selected statement.

5. **Delete**: Deletes the currently selected Step, IF or Else statement.

6. **Save and Close**: Saves and closes the procedure.

7. **Save**: Saves the procedure.

8. **Save As**: Saves the procedure to a different name.

9. **Cancel**: Cancels changes made to the procedure.

Drag and Drop options

- Drag any statement and drop it above another statement.
- Drag any statement and drop it below another statement.
- Drag any statement and drop it between another statement.
- Drag any statement to a procedure Title, IF or Else statement and add it as a child statement.

Guidelines

- Click any Step, IF or Else statement in a procedure to see its properties in the right-hand pane. You can edit these properties immediately.
- You can nest steps within multiple IF or Else statements.
- You can remove an Else statement without removing its corresponding IF statement.
- You can set a Step to allow a procedure to continue running even if that particular Step fails.

IF and STEP Commands
An agent procedure is composed of a number of steps that are either IF (can be nested too) or STEP commands. A complete list of all IF and STEP commands with their definitions follows.

The following is a summary of IF-ELSE-STEP commands used in VSA agent procedures. A more detailed explanation of these commands follows next.

IF Definitions	
Application is Running	Tests to see if the specified application is running.
Check Registry Value	Evaluates the given registry value.
Check 64-bit Registry Value	Evaluates the given 64-bit registry value.
Check Variable	Evaluates the given agent variable.
Evaluate Expression	Compares a variable with a supplied value.
Service is Running	Determines if a service is running on the managed machine.
Test File	Tests for the existence of a file.
Test File in Directory Path	Tests for the existence of a file in the current directory path returned by Get Directory Path From Registry.
Test Registry Key	Tests for the existence of the given registry key.
Test 64-bit Registry Key	Tests for the existence of the given 64-bit registry key.
True	Always returns True, executing IF branch.
User Is Logged In	Tests whether a specific user, or any user, is logged in or not.
User Response is Yes	Presents a Yes/No dialog box to the user.

STEP Definitions	
Close Application	Closes a running application.
Delete File	Deletes a file from the managed machine.
Delete File in Directory Path	Deletes file in directory returned by Get Directory Path From Registry.
Delete Registry Key	Deletes a key from the registry.
Delete 64-bit Registry Key	Deletes a 64-bit key from the registry.
Delete Registry Value	Deletes a value from the registry.
Delete 64-bit Registry Value	Deletes a 64-bit value from the registry.
Execute File	Executes any file as if it was run from th *Run* item in the Windows Start menu.
Execute File in Directory Path	Same as execute file. File location is relative to the directory returned by Get Directory Path From Registry.
Execute Procedure	Starts another VSA agent procedure on the current machine.
Execute Shell Command	Runs any command from a command shell.
Get Directory Path From Registry	Returns the directory path stored in the registry at the specified location. Result used in subsequent steps.

Get File	Gets a file from the managed machine and saves it to the KServer.
Get File in Directory Path	Gets a file from the managed machine located relative to the directory returned by Get Directory Path From Registry and saves it to the KServer.
Get URL	Returns the text and HTML contents of a URL and stores it to a file on the managed machine.
Get Variable	Gets a value from the agent on the managed machine and assigns it to a variable.
Impersonate User	Uses the specified user account to execute a file or shell when Execute as user is specified.
Pause Procedure	The procedure is passed for N seconds.
Reboot	Reboots the managed machine.
Rename Locked File	Renames a file that is currently in use.
Rename Locked File in Directory Path	Renames a file currently in use in directory returned by Get Directory Path From Registry.
Schedule Procedure	Schedules an agent procedure to run on a specified machine.
Send Email	Sends an email to one or more recipients.
Send Message	Displays a message in a dialog box on the managed machine.
Send URL	Opens a browser to the specified URL on the managed machine.
Set Registry Value	Sets the registry value to a specific value.
Set 64-bit Registry Value	Sets the 64-bit registry value to a specific value.
Update System Info	Updates the selected System Info field with the specified value.
Use Credential	Uses the user logon credentials set for the machine ID in Set Credential to execute a file or shell when Execute as user is specified.
Write Directory	Writes a directory from the server to the managed machine.
Write File	Writes a file stored on the KServer to the managed machine.
Write File in Directory Path	Writes a file stored on the KServer to the managed machine using the directory returned by Get Directory Path From Registry.
Write Procedure Log Entry	Writes a string to the Agent Procedure Log.

IF Commands

Application is Running

It checks to see if a specified application is currently running on the managed machine. If the application is running, the IF command is executed; otherwise, the ELSE command is executed. When this option is selected from the drop-down list, the *Enter the application name* field appears. Specify the process name for the application you want to test. For example, to test the *Calculator* application, specify *calc.exe*, which is the

process name that displays in the Processes tab of the Windows Task Manager.

Check Registry Value / Check 64-Bit Registry Value
After entering the registry path, the value contained in the key is returned. A check can be made for existence, absence, equality, or size differences. For example, *HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\AppPaths\AgentMon.exe\path* contains the directory path identifying where the agent is installed on the target machine. The test determines if the value stored for this key exists, thereby verifying the agent is installed.
A backslash character \ at the end of the key returns the default value of that key.*HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\WORDPAD.EXE* returns a default value, such as %ProgramFiles%\Windows NT\Accessories\WORDPAD.EXE

The available tests are:

- *Exists*, true if the registry key exists in the hive.
- *Does Not Exist*, true if the registry key does *not* exist in the hive.
- =, true if value of the registry key equals the test value.
- *Not* =, true if value of the registry key does *not* equal the test value.
- >, true if value of the registry key is greater than the test value (value must be a number).
- >=, true if value of the registry key is greater than or equal to the test value (value must be a number).
- <, true if value of the registry key is less than the test value (value must be a number).
- <=, true if value of the registry key is less than or equal to the test value (value must be a number).
- *Contains*, true if the test value is a sub string of the registry key value (value must be a string).
- *Not Contains*, true if the test value is *not* a sub string of the registry key value (value must be a string).

Check Variable
Enter a variable name, in the form #var_name#, in the space provided. **Check Variable** evaluates the current values assigned #var_name# and compares it with the supplied value. The supplied value may also be another variable name in the form of #var_name2#. If the check is true, **IF** commands are executed. If the check is false, **ELSE** steps are executed.

- *Exists*, true if the variable exists.
- *Does Not Exist*, true if the variable does *not* exist.
- =, true if value of the variable equals the test value.
- *Not* =, true if value of the variable does *not* equal the test value.
- >, true if value of the variable is greater than the test value.
- >=, true if value of the variable is greater than or equal to the test value.
- <, true if value of the variable is less than the test value.
- <=, true if value of the variable is less than or equal to the test value.
- *Contains*, true if the test value is a sub string of the variable value.
- *Not Contains*, true if the test value is *not* a sub string of the variable value.
- *Begins With*, true if the test value begins with the variable value.

- *Ends With*, true if the test value ends with the variable value.

For the tests =, *Not* =, >, >=, <, and <=, the variables compared may be a string, a number, a date in the format of "yyyy/mm/dd" or "yyyy/mm/dd hh:mm" or "yyyy/mm/dd hh:mm:ss", or a version number containing dots or commas such as "1.2.3" or "4,5,6,7". If a date format is specified, it may be offset using "+ dd:hh:mm:ss" or "- dd:hh:mm:ss". Only "dd" days are required; "hh" hours, "mm" minutes, and "ss" seconds may be omitted and are assumed to be zero when absent. CURRENT_TIMESTAMP may be specified to indicate that the current time be substituted in the comparison at the time the procedure is executed. For example, "CURRENT_ TIMESTAMP - 7:12:00:00" will be evaluated as 7 days and 12 hours subtracted from the time that the procedure is executed.

Evaluate Expression
Enter an expression containing one or more variable names, in the form #var_name#, in the space provided. Evaluate Expression uses the current value assigned to each #var_name#, evaluates the mathematical expression, and compares it with the supplied value. The supplied value may also be another expression. The mathematical expression may contain +, -, *, /, (, and). For example, "(3.7 + (200 * #countA#)) / (#countB# - #countC#)". If the check is true, IF steps are executed. If the check is false, ELSE steps are executed. The available tests are:

- =, true if value of the variable equals the test value.
- *Not* =, true if value of the variable does not equal the test value.
- >, true if value of the variable is greater than the test value.
- >=, true if value of the variable is greater than or equal to the test value.
- <, true if value of the variable is less than the test value.
- <=, true if value of the variable is less than or equal to the test value.


Notes

Note: *Cannot be used with* `Exists`, `Does Not Exist`, `Contains`, *or* `Not Contains` *operators.*

Service is Running
This option determines if a service is running on the managed machine. Specify the service name.

- True if the service name is running.
- False if the service name is stopped or does not exist.


Notes

Note: *Use the service name of the service, not the display name of the service. For example, the display name of the service for Microsoft SQL Server is* `SQL Server (MSSQLSERVER)`*, but the service name of the service is* `MSSQLSERVER`*. For Windows machines, right click any service in the* Services *window and click the* Properties *option to see the* service name *of that service.*

Test File
Determines if a file exists on a managed machine. Enter the full path and file name. Test File compares the full path and file name with the supplied value. If the check is true, IF commands are executed. If the check is false, ELSE steps are executed.


Notes

Note: *Environment variables such as %windir%\notepad.exe are acceptable.*

The available tests are:

- *Exists*, true if the full path and file name exists.
- *Does not Exist*, true if the full path and file name does not exist.
- *Contains*, true if the test value is a sub string of the file content.

- *Not Contains*, true if the test value is not a sub string of the file content.
- *Begins With*, true if the test value begins with the variable value.
- *Ends With*, true if the test value ends with the variable value.

Test File in Directory Path
Tests the specified file located at the path returned using the Get Directory Path From Registry step. The available tests are:

- *Exists*, true if the file name exists.
- *Does not Exist*, true if the file name does not exist.
- *Contains*, true if the test value is a sub string of the file content.
- *Not Contains*, true if the test value is not a sub string of the file content.
- *Begins With*, true if the test value begins with the variable value.
- *Ends With*, true if the test value ends with the variable value.

Test Registry Key / Test 64-bit Register Key
Tests the existence of a registry key. **Test Registry Key** differs from **Check Registry Value** since it can check for a directory level registry entry that only contains more registry keys (no values).

True
Selecting `True` directs the **IF** commands to execute. Use *True* to directly execute a series of steps that do not require any decision points, such as determining whether a file exists using **Test File**.

User Is Logged In
Tests to see if a specific user or any user is logged on the managed machine. Enter the machine user's logon name or leave the field blank to check for any user logged on. The **IF** commands are executed if a user is logged on. The **ELSE** steps are executed if the user is not logged on.

User Response is Yes
Displays a dialog box on the managed machine with *Yes* and *No* buttons. Also carries out the ELSE command if a specified amount of time has timed out. If Yes is selected by the machine user, the IF command is executed. If the selection times out or the machine user selects *No*, the ELSE command is executed. This function requests the machine user's permission to proceed with the agent procedure. This query is useful for agent procedures that require a reboot of the managed machine before completion.

Procedure variables, for example #varName#, may be used inside User Response is Yes fields to dynamically generate messages based on procedure data.

STEP Commands

Close Application
If the specified application is running on the managed machine, then that application is closed down. Specify the process name for the application you want to close. For example, to close the *Calculator* application, specify "*calc.exe*", which is the process name that displays in the Processes tab of the Windows Task Manager.

Delete File
Deletes a file on a managed machine. Enter the full path and filename.

Note: *You can delete a file that is currently in use using the* Rename Locked File *command.*

Delete File in Directory Path
Deletes the specified file located at the path returned using the *Get Directory Path from Registry* command.

Delete Registry Key / Delete 64-bit Registry Key
Deletes the specified registry key and all its sub-keys.

Delete Registry Value / Delete 64-bit Registry Value
Deletes the value stored at the specified registry key.

Execute File
Executes the specified file on the managed machine. This function replicates launching an application using the *Run* command located in the Microsoft Windows Start menu. This function takes three parameters:

- Full path filename to the .exe file.
- Argument list to pass to the .exe file
- Option for the procedure to wait until the .exe completes or not.

Execute File in Directory Path
Same as Execute File except the location of the .exe file is located at the path returned from a *Get Directory Path From Registry* command.

Execute Procedure
Causes another named procedure to execute. Use this capability to string multiple **IF-ELSE-STEP** procedures together. If the procedure no longer exists on the KServer, an error message displays next to the procedure drop-down list. You can use this command to run a system procedure. You can nest procedures to 10 levels.

Execute Shell Command
Allows the procedure to pass commands to the command interpreter on the managed machine. When this command is selected, the field *Enter the command to execute in a command shell* is displayed. Enter a command in the field. The command must be syntactically correct and executable with the OS version on the managed machine. Commands and parameters containing spaces should be surrounded by quotes. Since the command is executed relative to the agent directory, absolute paths should be used when entering commands.

Get Directory Path from Registry
Returns a file path stored in the specified registry key. Use this command to fetch the file location. For instance, use this command to find the directory where an application has been installed. The result can be used in subsequent steps by:

- Delete File in Directory Path
- Execute File in Directory Path
- Get File in Directory Path
- Rename Locked File in Directory Path
- Test File in Directory Path (an IF command)
- Write File in Directory Path

Get File
Upload the file at the specified path from the managed machine. Be sure to enter a full path filename that you want to upload. Example: `news\info.txt`. Folders are created when the **Get File** command is run, if they don't already exist. The file is stored on the KServer in a private directory for each managed machine. View or run the uploaded file using **Agent Procedures > Get File.**

- Optionally, existing copies of uploaded files are renamed with a `.bak` extension prior to the next up-

load of the file. This allows you to examine both the latest version of the file and the previous version.

• Optionally create a **Get File** alert if the uploaded file *differs* or is the *same* from the file that was uploaded previously. *You must create a Get File alert for a machine ID* using the **Monitor > Alerts - Get File** page to enable the sending of an alert using the **Get File** command. Once defined for a machine ID, the same **Get File** alert is *active for any agent procedure* that uses a **Get File** command and is run on that machine ID. Turn off alerts for specific files in the agent procedure editor by selecting one of the without alerts options.

Get File in Directory Path
Just like the **Get File** command but it adds the path returned from the *Get Directory Path From Registry command* to the beginning of the remote file path. Access the uploaded file using the **Agent Procedures > Get File** function.

Get URL
Returns the text and HTML contents of a URL and stores it to a file on the managed machine. To demonstrate this to yourself, try specifying "`www.kaseya.com`" as the URL and "`c:\temp\test.htm`" as the file to store the contents of this URL. A copy of the web page is created on the managed machine that contains all of the text and HTML content of this webpage. You can search the contents of the file on the managed machine in a subsequent command. Another use is to download an executable file that is available from a web server, so that you don't need to upload the file to the VSA server nor use the VSA's bandwidth to write the file down to each agent. You can use a subsequent command to run the downloaded executable on the managed machine.

Get Variable
Defines a new agent variable. When the procedure step executes, the system defines a new variable and assigns it a value based on data fetched from the managed machine's agent.

Impersonate User
Enter a username, password, and domain for the agent to logon with. This command is used in a procedure before an *Execute File, Execute File in Directory Path or Execute Shell Command* that specifies the *Execute as the logged on user* option. Leave the domain blank to log into an account on the local machine. Use *Impersonate User* to run an agent procedure using a credential specified by agent procedure. Use *Use Credential* to run an agent procedure using a credential specified by managed machine.

Pause Procedure
Pause the procedure for N seconds. Use this command to give Windows time to complete an asynchronous task, like starting or stopping a service.

Reboot
Unconditionally reboots the managed machine. To warn the user first, use the *User Response is Yes* command before this command. A *User Response is Yes* command prompts the user before rebooting their machine.

Rename Locked File
Renames a file that is currently in use. The file is renamed the next time the system is rebooted. The specified filename is a complete file path name. It can be used to delete a file that is currently in use if the "new file name" is left blank. The file is deleted when the system is rebooted.

Rename Locked File in Directory Path
Renames a file that is currently in use that is located in the path returned from a *Get Directory Path from Registry* command. The file is renamed the next time the system is rebooted. It can be used to delete a file that is currently in use if the "new file name" is left blank. The file is deleted when the system is rebooted.

Schedule Procedure
Schedules a procedure to run on a specified machine. Optionally specifies the time to wait after executing this step before running the procedure and the specified machine ID to run the procedure on. If no machine is specified, then the procedure is run on the same machine running the agent procedure. Enter the complete

name of the machine, for example, `machine.unnamed.org`. *This command allows an agent procedure running on one machine to schedule the running of an agent procedure on a second machine.* You can use this command to run a system procedure. You can nest procedures to 10 levels.

Send Email
Sends an email to one or more recipients. Specifies the subject and body text of the email.

Send Message
Sends the entered message to a managed machine. An additional checkbox, if checked, sends the message immediately. If unchecked, sends the message after the user clicks the flashing agent system tray icon.

Send URL
Displays the entered URL in a web browser window on the managed machine. An additional checkbox, if checked, displays the URL immediately. If unchecked, the URL is displayed after the user clicks the flashing agent system tray icon.

Set Registry Value / Set 64-bit Registry Value
Writes data to the specified registry value. This function takes three parameters:

- **Enter the full path to a registry key containing a value** - Specify the (Default) value for a registry key by adding a trailing backslash \. Otherwise specify a name for an existing value or to create a new value. See Name column in image below.

  Example of setting the (Default) value: HKEY_LOCAL_MACHINE\SOFTWARE\000Sample\

- **Enter the data to write to the registry value**
- **Select the data type**
  - REG_SZ - String value.
  - REG_BINARY - Binary data displayed in hexadecimal format.
  - DWORD - Binary data limited to 32 bits. Can be entered in hexadecimal or decimal format.
  - REG_EXPAND_SZ - An “expandable” string value holding a variable.
    Example: %SystemRoot%.
  - REG_MULTI_SZ - A multiple string array. Used for entering more than one value, each one separated by a \0string. Use \\0 to include \0 within a string array value.

Update System Info

Updates the selected **System Info** field with the specified value for the machine ID this procedure runs on. The **System Info** fields you can update include all columns in vSystemInfo except agentGuid, emailAddr, Machine_GroupID, machName, and groupName. **SystemInfo** column information is used by **Audit > System Info**, **Agent > System Status**, the Filter Aggregate Table in **View Definitions**, and the Aggregate Table report. You can update a **System Info** field using any string value, including the value of any previously defined agent procedure variable.

Use Credential
Uses the credentials set for the machine ID in Set Credential. This command is used in a procedure before an *Execute File,Execute File in Directory Path* or *Execute Shell Command* that specifies the Execute as the logged on user option. Also used to access a network resource requiring a credential from a machine when a user is not logged on.


Notes

Note: *A procedure execution error is logged if a Set Credential procedure command encounters an empty username.*

Write Directory
Writes a selected directory, including subdirectories and files, from Manage Files Stored on Server to the full path directory name specified on the managed machine.

Write File
Writes a file selected from Manage Files Stored on Server to the full path filename specified on the managed machine. Enter a new filename if you want the file to be renamed.
Each time a procedure executes the **Write File** command, the agent checks to see if the file is already there or not by hashing the file to verify integrity. If not, the file is written. If the file is already there, the procedure moves to the next step. You can repeatedly run a procedure with **Write File** that sends a large file to a managed machine and know that the VSA only downloads that file once.

Write File in Directory Path
\Writes the specified filename to the path returned from a *Get Directory Path From Registry* command.

Write Procedure Log Entry
Writes the supplied string to the Agent Procedure Log for the machine ID executing this agent procedure.

VI. Import Folder/Procedure: Imports a folder or procedure as children to the selected folder in the folder tree.

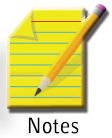
VII. Export Folder: Exports the selected folder and all its procedures as an XML file. The XML file can be re-imported.

VIII. Take Ownership: Takes ownership of a folder you do not own. This option only displays for master role users.

Note: *This option displays only for master role users.*

IX. Folder Properties: Display the name, description, and owner of a folder, and your access rights to the folder.

Objects you create such as reports, procedures, or monitor sets are initially saved in a folder with your user name underneath a **Private** cabinet. This means only you, the creator of the objects in that folder, can view those objects, edit them, run them, delete them or rename them. To share a private object with others you first have to drag and drop it into a folder underneath the *Shared* cabinet.



Note: *A master role user can check the* **Show shared and private folder contents from all users** *checkbox in* **System > Preferences** *to see all shared and private folders.*

Shared Folders
The following Share Folder guidelines apply to folders underneath a *Shared* cabinet:

- If the *Apply share rights from parent folder* checkbox in the *Share Folder* dialog box is checked, a folder's share rights are determined by the parent folder. Otherwise, the folder›s share rights can be set independently from the parent.
- If you have rights to delete a folder, deleting that folder deletes all objects and subfolders as well, regardless of share rights or ownership assigned to those subfolders.
- To set share rights to a folder, select the folder, then click the *Share Folder* button to display the Share Folder dialog.
- You can share specific rights to a folder with any individual user or user role you have visibility of. You have visibility of:
- Any user roles you are a member of, whether you are currently using that user role or not.

- Any individual users that are members of your current scope.

- Adding a user or user role to the Shared Pane allows that user to run any object in that folder. No additional rights, including *View*, have to be assigned to the user or user role to run the object.

- Checking any additional rights such as *View, Edit, Create, Delete, Rename*, or *Share*, when you add the user or user role provides that user or user role with those additional rights. You have to remove the user or user role and re-add them to make changes to their additional rights.

- *View* does not refer to being able to view the folder. If you assign a user to the share folder without giving the user the *View* right, the user must still be able to see the folder and its objects to be able to select and run the object. Instead *View* means the user or user role can display the details of the object and export it, beyond just running the object.

- Share allows the user or user role to assign share rights for a selected folder using the same Share Folder dialog box used to assign the share rights.

Take Ownership
Users are always the one and only owner of their *Private* folders. *Shared* folders are also *owned* and are only owned by one user at a time. Ownership of a shared folder provides "full rights" to a folder's objects, regardless of the share rights assigned to that user. When you first create a shared folder, either as a master role user or a non-master role user, you are the owner of that shared folder. Master role users have an additional right, called *Take Ownership*, that allows them to take ownership of any *Shared* folder that is visible in the folder tree.

As a master role user, if the button displays when you select a *Shared* folder, that means you're not the owner of that folder. If a folder you don't own has been shared with you, then several other buttons may display alongside the button. Until you click the button you're restricted to the actions determined by the share rights you've been assigned.
Clicking the button makes you the one and only owner of that shared folder. Taking ownership displays an orange dot on the folder , indicating ownership. Ownership overrides your assigned shared rights and gives you complete access to:

- Add, edit, change, rename or delete objects in that folder.

- Add, rename or delete subfolders.

- Rename or delete the folder you took ownership of and all its contents.

Typically the reason you take ownership of a shared object is to maintain its contents because the original owner can't do so. For example, the owner of a shared object may have left the company and no longer be available. In most cases, master role users can work within the share rights they've been assigned by other VSA users.

Functions That Are Available When a Procedure is Selected

Fig. 7.6 shows the Schedule / Create page. The functions are visible to the user when the cursor is on a procedure. These functions are listed and explained below.

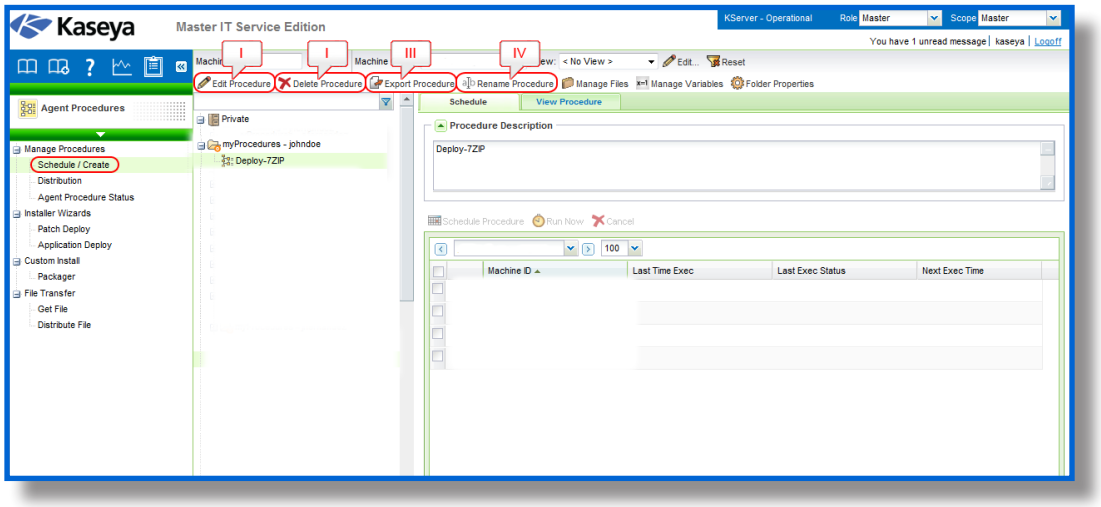


Fig. 7.6: *Schedule / Create Page: Labels point to the functions that are available when a folder is selected.*

I. Edit Procedure - Opens the Agent Procedure Editor to edit the selected procedure.

II. Delete Procedure - Deletes the selected procedure.

III. Export Procedure - Exports the selected procedure.

IV. Rename Procedure - Renames the selected procedure.

Creating / Editing Agent Procedures

To create a new procedure, click on a cabinet or folder in the middle pane, then click the *New Procedure* button on the top to open the Agent Procedure Editor. To edit an existing procedure, select the procedure, and then click the *Edit Procedure* button to open the Agent Procedure Editor. You can also double-click a procedure to edit it.

Note: Access to creating or editing a procedure depends on your Folder Rights.

Running / Scheduling / Viewing Agent Procedures

When a procedure is selected in the middle pane, the following tabs display in the right-hand pane:

Schedule - Select one or more machine IDs in this tab's table, then click one of the following action buttons:

- **Schedule Procedure** - Schedule a task once or periodically.
- **Run Now** - Run this agent procedure on each selected machine ID immediately.
- **Cancel** - Cancel the scheduled agent procedure on each selected machine ID.

View Procedure - Provides a display only view of the procedure. A user can execute an agent procedure and view it without necessarily being able to edit it.

7.1.2 Distribution

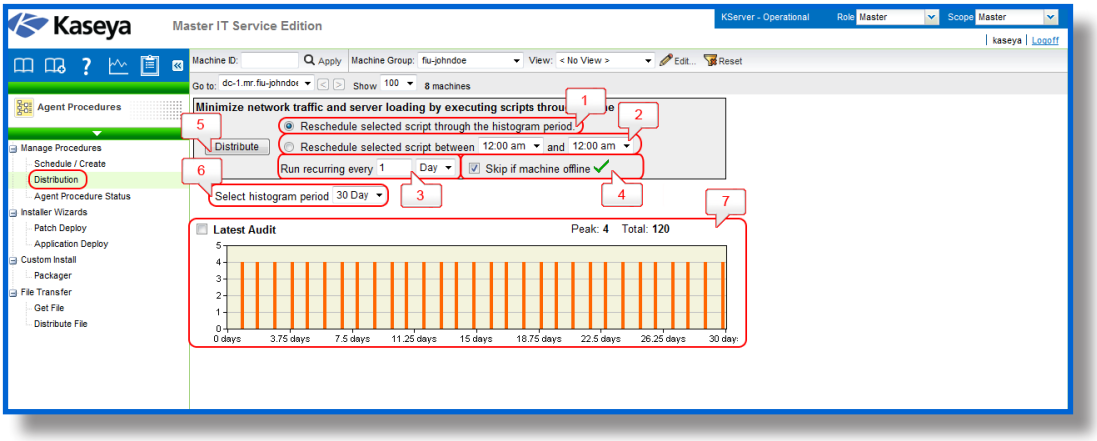
The **Distribution** page spreads network traffic and server loading by executing agent procedures evenly throughout the day or a specific block of time in a day. Applies to agent procedures currently scheduled to run on a **recurring basis** only**.**

Procedures can cause excessive network loading by pushing large files between the KServer and agent. Performing these operations with hundreds of agents simultaneously may cause unacceptable network loading levels.

Procedure Histograms: The system plots a histogram for each procedure currently scheduled to run on a recurring basis. Setting the histogram period to match the recurring interval of the procedure counts how many machines execute the procedure in a specific time interval. Peaks in the histogram visually highlight areas where a lot of machines are trying to execute the procedure at the same time. *Click a peak to display a popup window listing all machine IDs contributing to that peak load.* Use the controls, described below, to reschedule the procedure such that the network loading is spread evenly over time. **Only machine IDs currently matching the Machine ID / Group ID filter are counted in the histogram.**

Fig. 7.7 below shows the generic view of the Distribution page. The functions supported on this page are listed and explained below.

Fig. 7.7: *Distribution page*



1. **Reschedule selected procedure evenly through the histogram period:** Pick this radio control to reschedule selected procedures running on all machines IDs currently matching the Machine ID / Group ID filter. Procedure execution start times are staggered evenly across the entire histogram period.

2. **Reschedule selected procedure evenly between <start time> and <end time>:** Pick this radio control to reschedule selected procedures running on all machines IDs currently matching the Machine ID / Group ID filter. Procedure execution start times are staggered evenly, beginning with the start time and ending with the end time.

3. **Run recurring every <N> <periods>:** This task is always performed as a recurring task. Enter the number of times to run this task each time period.

4. **Skip if Machine Offline:** Check to perform this task only at the scheduled time, within a 15 minute window. If the machine is offline, skip and run the next scheduled period and time. Uncheck to perform this task as soon as the machine connects after the scheduled time.

5. **Distribute:** Click the *Distribute* button to schedule selected procedures, using the schedule parameters you›ve defined.


Notes

Note: The procedure recurring interval is replaced with the histogram period.

6. **Select Histogram Period:** Selects the schedule time period to display histograms.

7. **Histogram Plots:** Each recurring procedure displays a histogram of all the machine IDs that are scheduled to run that procedure within the selected histogram period. Only machine IDs currently matching the Machine ID / Group ID filter are counted in the histogram.

Above the histogram is a:

- **Procedure name** - name of the procedure. Check the box next to the procedure name to select this procedure for distribution.
- **Peak** - the greatest number of machines executing the procedure at the same time.
- **Total** - total number of machines executing the procedure.

7.1.3 Agent Procedure Status

The **Agent Procedure Status** page displays the status of agent procedures for a selected machine ID. The list of machine IDs you can select is based on the Machine ID / Group ID filter. Users can, at a glance, find out what time an agent procedure was executed and whether it was successfully executed.

Fig. 7.8 below shows the generic view of the Agent procedure status page. The functions supported on this page are listed and explained below.

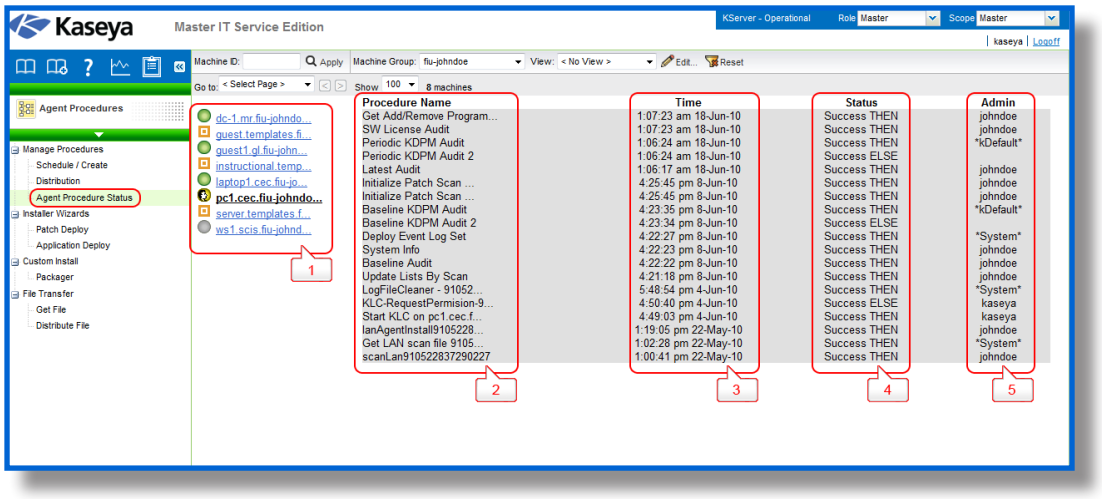


Fig. 7.8: *Agent procedure status*

1. **Machine.Group ID:** The list of Machine.Group IDs displayed is based on the Machine ID / Group ID filter and the machine groups the user is authorized to view.

2. **Procedure Name:** The name of the agent procedure.

3. **Time:** The date and time the agent procedure was last executed.

4. **Status:** Displays the results of the executed agent procedure. Overdue date/time stamps display as red text with yellow highlight. Recurring agent procedures display as red text.

5. **Admin:** Displays the VSA user who scheduled the agent procedure.

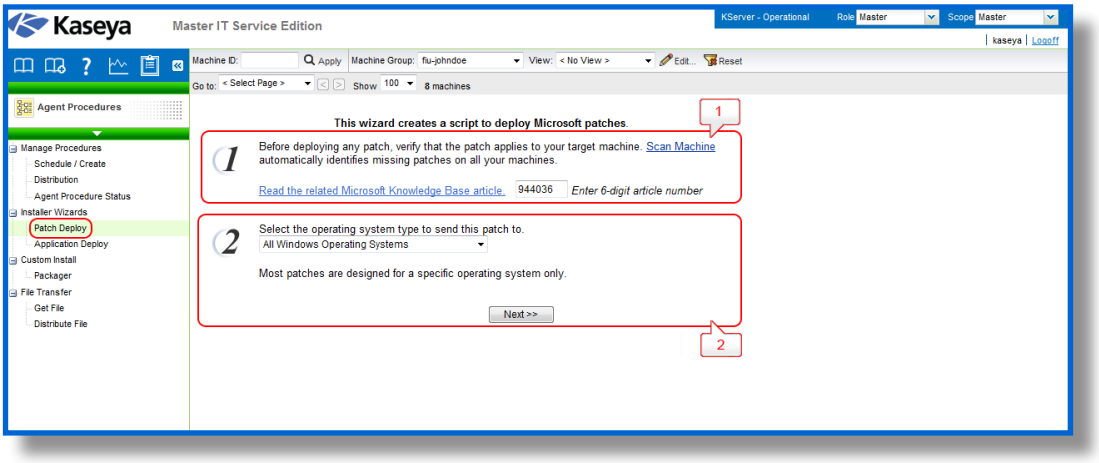
7.2 Installer Wizards

7.2.1 Patch Deploy

The **Patch Deploy** wizard is a tool that creates an agent procedure to distribute and apply Microsoft patches. The wizard walks you through a step by step process resulting in an agent procedure you can schedule, to deploy a patch to any managed machine. Microsoft releases many hot fixes as patches for very specific issues that are not included in the Microsoft Update Catalog or in the Office Detection Tool, the two patch data sources the Patch Management module uses to manage patch updates. Patch Deploy enables customers to create a patch installation procedure for these hot fixes, via this wizard, that can be used to schedule the installation on any desired machine.

The figures below show the generic view of the wizard on the Patch deploy page. The functions, which appear on each figure, are labeled and explained below.

Fig. 7.9: *Patch Deploy wizard 1*



Step 1: Enter 6-digit knowledge base article number: Microsoft publishes a vast assortment of information about its operating system in the Microsoft Knowledge Base. Each article in the Knowledge Base is identified with a 6-digit Q number (*e.g.* 944036) as shown in Fig. 7.5. All Microsoft patches have an associated knowledge base article number.

Step 2: Select the operating system type: Sometimes patches are specific to a certain operating system. If the patch you are trying to deploy applies to a specific OS only, then select the appropriate operating system from the drop-down control. When the wizard creates the patch deploy procedure, it restricts execution of the procedure to only those machines with the selected OS. This prevents inadvertent application of operating system patches to the wrong OS.

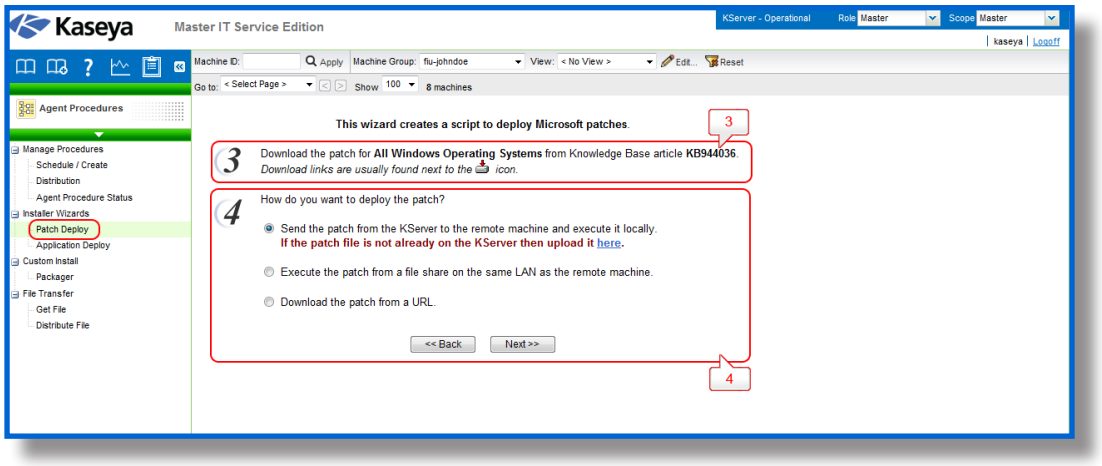


Fig. 7.10: *Patch Deploy wizard 2*

Step 3: Download the patch: This step is just a reminder to fetch the patch from Microsoft. Typically there is a link to the patch on the knowledge base article describing the patch.

Step 4: How do you want to deploy the patch?: The Patch Deploy wizard asks you in step 4 if you want to *Send the patch from the KServer to the remote machine and execute it locally* or *Execute the patch from a file share on the same LAN as the remote machine.* Pushing the patch down to each machine from the VSA may be bandwidth intensive. If you are patching multiple machines on a LAN no Internet bandwidth is used to push out the patch. Each machine on the LAN can execute the patch file directly from a common file share.

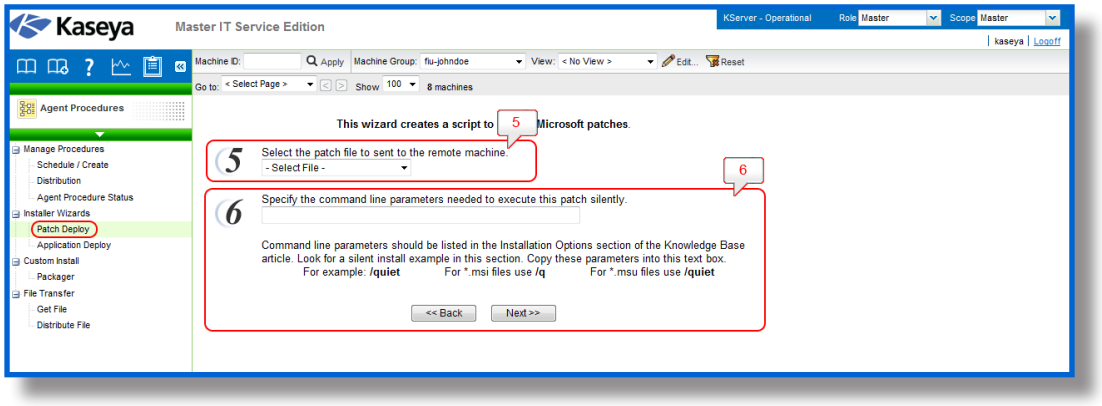


Fig. 7.11: *Patch Deploy wizard 3*

Step 5: Select the patch file or **Specify the UNC path to the patch stored on the same LAN as the remote machine:** If *Send the patch from the KServer to the remote machine and execute it locally* was selected, then the patch must be on the VSA server. Select the file from the drop-down list.

Note: *If the patch file does not appear in the list then it is not on the KServer.*

Notes

If *Execute the patch from a file share on the same LAN as the remote machine* was selected, then the patch must be on the remote file share prior to running the patch deploy procedure. The specified path to the file must be in **UNC format** such as `\\computername\dir\`.

Note: *If the file is not already on the remote file share, you can put it their via FTP. Click the* back *button and then the second* here *link takes you to FTP.*

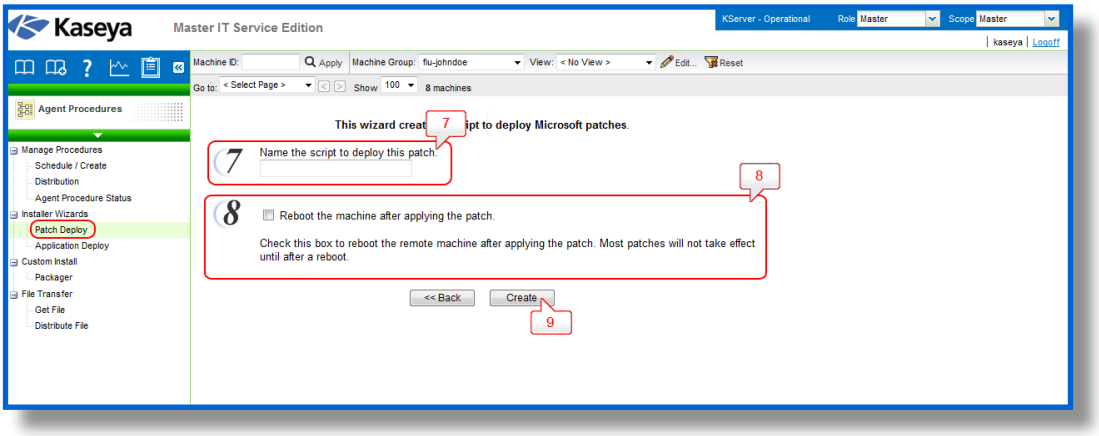
Notes

Step 6: Specify the command line parameters needed to execute this patch silently: To deploy a patch silently you need to add the appropriate command line switches used when executing the patch. Each knowledge base article lists the parameters for silent install. Typical switch settings are /q /m /z.

Note: *Command line parameters are optional. It can be left blank if you are unaware of it.*

Notes

Fig. 7.12: *Patch Deploy wizard 3*



Step 7: Name the procedure: Enter a name for the new agent procedure you can run to deploy the patch.

Step 8: Reboot the machine after applying the patch: Check this box to automatically reboot the managed machine after applying the patch. The default setting is to not reboot.

Step 9: Create: A new agent procedure is created. Use **Agent Procedure > Schedule / Create** to display the new agent procedure in the folder tree, under your private folder user name. You can run this new agent procedure to deploy the patch to any managed machine.

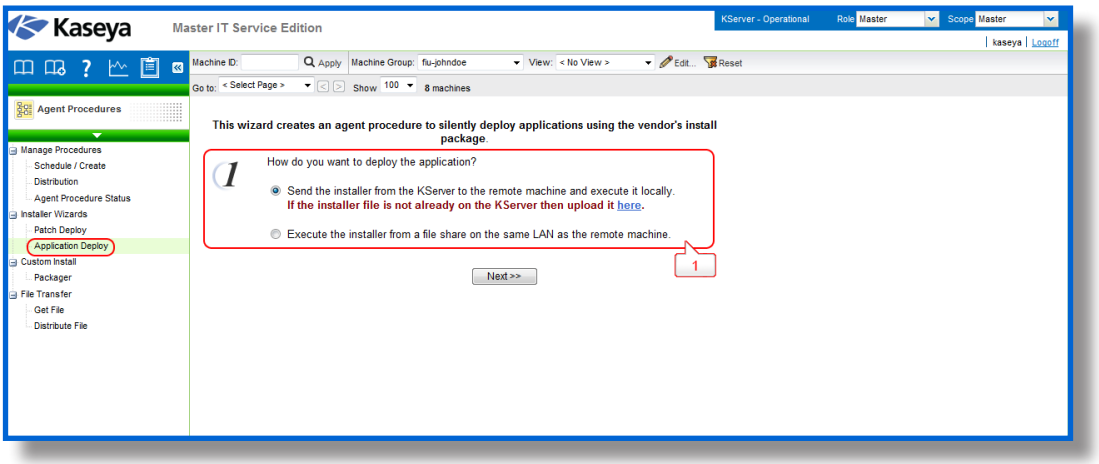
7.2.2 Application Deploy

The **Application Deploy** page is a wizard tool that creates an agent procedure to distribute vendor installation packages, typically `setup.exe`. The wizard walks you through a step by step process resulting in an agent procedure you can schedule, to deploy an application to any managed machine.

Deploying Software Vendor's Install Packages
Most vendors provide either a single file when downloaded from the web or set of files when distributed on a CD. Executing the installer file, typically named `setup.exe` or `abc.msi`, installs the vendor's application on any operating system. The **Application Deploy** wizard takes you through an interview process to determine the type of installer and automatically generates a procedure to deploy install vendor packages. The VSA provides a small utility to automatically identify all supported installer types. Download and run `kInstId.exe` to automatically identify the installer type.

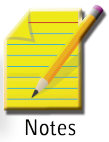
Fig. 7.13, 7.14 and 7.15 below show the generic view of the Application Deploy wizard on the Application deploy page. The functions, which appear on each figure, are labeled and explained below.

Fig. 7.13: *Application Deploy wizard 1*



Step 1: How do you want to deploy the application?: The wizard generated procedure tells the managed machine where to get the application installation file to execute. The *Application Deploy* wizard asks you in step 1 if you want to *Send the installer from the VSA server to the remote machine and execute it locally* or *Execute the installer from a file share on the same LAN as the remote machine*. Pushing the application installation file to each machine from the VSA may be bandwidth intensive. If you are installing to multiple machines on a LAN no internet bandwidth is used to push out the application installation file. Each machine on the LAN can execute the application installation file directly from a common file share.

Note: *If the file is not already on the remote file share, you can put it there via FTP. Click the* **here** *link to start FTP.*



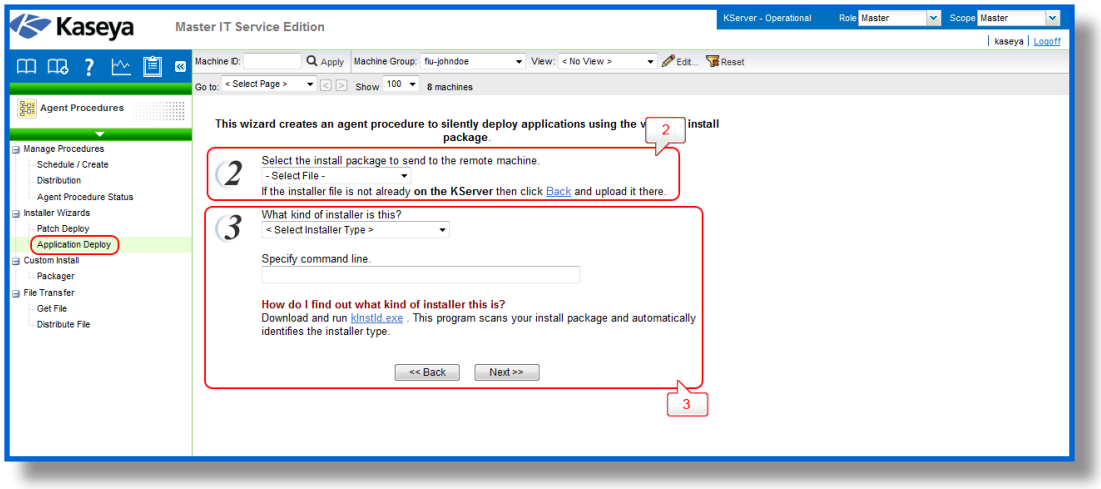


Fig. 7.14: *Application Deploy wizard 2*

Step 2: Select the application install file or Specify the UNC path to the installer stored on the same LAN as the remote machine: If *Send the installer from the VSA server to the remote machine and execute it locally* was selected, then the installer file must be on the VSA server. Select the file from the drop-down list. If *Execute the installer from a file share on the same LAN as the remote machine* was selected, then the installer file must be on the remote file share prior to running the application deploy procedure. The specified path to the file must be in UNC format such as \\computername\dir\.

Step 3: What kind of installer is this?: The wizard need to know what kind of installer was used by your software vendor to create the install package. The VSA provides a small utility to automatically identify all supported installer types. Download and run kInstId.exe to automatically identify the installer type. Supported installer types are:

- Windows Installer (MSI files)
- Wise Installer
- Installshield - Package For The Web
- Installshield - Multiple Files
- Other

t

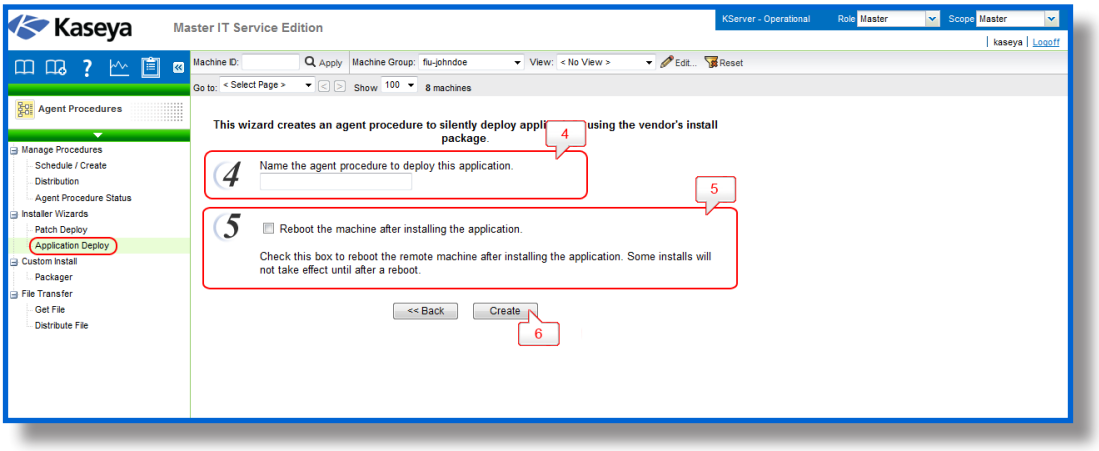


Fig. 7.15: *Application Deploy wizard 3*

Step 4: Name the agent procedure: Enter a name for the new agent procedure you can run to install the application.

Step 5: Reboot the machine after installing the application: Check this box to automatically reboot the managed machine after running the install. The default setting is to not reboot.

Step 6: Create: A new agent procedure is created. Use **Agent Procedure > Schedule / Create** to display the new agent procedure in the folder tree, under your private folder user name. You can run this new agent procedure to install the application to any managed machine.

7.3 Custom Install

7.3.1 Packager

The **Packager** is a wizard tool used to create a package when a pre-defined install solution cannot be used. Packager evaluates the state of a source machine before and after an installation and/or resource change. The Packager compiles the differences into a single executable file that can be distributed via agent procedures to any managed machine. Distribute a package any way you choose. You can email it, or store it on a server where a custom procedure can perform a silent installation on any managed machine.

Step 1: Download the Packager application to the machine you plan to build your install package on: It is recommend creating a package on a representative machine; that is, a machine that closely resembles the managed machines on which the package will be deployed for best results.

Each Package is OS dependent: To deploy to multiple operating systems, you need to build a package for each OS. During installation, Packager checks the target machine's operating system and does not continue if the package is being deployed on an OS different than the source OS.

Step 2: Execute Packager.exe **and follow the on-screen instructions to create a distribution package:** The following tasks are performed:

1. Packager takes a snapshot of the source system.
2. Install any application and/or resource on the source system.
3. Execute Packager again. Packager records the changes in the source system and creates a package.

Packager picks up everything you do to a machine between the time you take the first snapshot and create

the package. Be careful what additional tasks you perform on the source machine as any system changes will be rolled into the package. Close all applications before running Packager. This prevents open applications from modifying the system during package creation.

Step 3: Distribute the package via a procedure: Use **Agent Procedure > Schedule / Create** to create an agent procedure that downloads the package to managed machines and runs it. Packages can only be executed on machines with agents installed. If the package fails to install, Packager has complete rollback capability. The rollback executable and associated restore files are located in the agent directory on the target machine in the directory C:\Program Files\Kaseya\KPackage.

7.4 File Transfer

7.4.1 Get File

The **Get File** page accesses files previously uploaded from a managed machine. Files can be uploaded to a machine-specific directory on the KServer using the **Get File** or **Get File In Directory Path** commands. Clicking the machine ID displays *all* uploaded files for that machine ID. Click the link underneath a file to display the file or run it.

- Each file is displayed as a link. Click any filename to access that file.
- Remove files by clicking the delete icon next to the file.

Example 1: Checking large number of managed machines simultaneously
Get File is designed to support automated checks on a large number of managed machines simultaneously. Use Get File in conjunction with an agent procedure to perform some automated task on a set of managed machines. For example, if you have a utility that reads out some information unique to your client computers you can write a procedure to do the following:

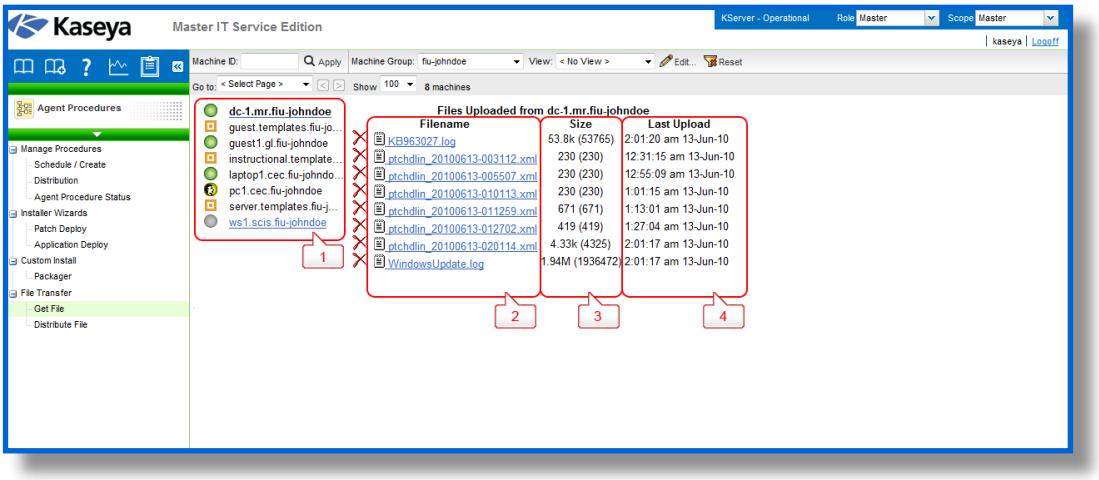
Send the utility to the managed machine using either the *Write File* procedure command or the **Agent Procedure > Distribute File** page. Execute the utility using either the *Execute Shell Command* or *Execute File* agent procedure command and pipe the output to a text file, such as results.txt. Upload the file to the KServer using the *Get File* command.

Example 2: Comparing Versions of a File
As an option in the **Get File** agent procedure command, existing copies of uploaded files can be renamed with a *.bak* extension prior to the next upload of the file. This allows you to examine both the latest version of the file and the previous version. For example, use the IF-ELSE-STEP agent procedure editor to create a simple **Get File** agent procedure. The first time the **Get File** agent procedure command executes on a managed machine the agent sends *c:\temp\info.txt* to the KServer and the KServer stores it as *news\info.txt*. The second time **Get File** agent procedure executes, the KServer renames the original copy of *news\info.txt* to *news\info.txt.bak* then uploads a fresh copy and saves it as *news\info.txt.*

Fig. 7.16 below shows the generic view of the Get File page. The functions supported on this page are listed and explained below.

1. **Machine.Group ID:** The list of Machine.Group IDs displayed is based on the Machine ID / Group ID filter and the machine groups the user is authorized to view.
2. **Filename:** This column displays the name of the file.
3. **Size:** This column displays the size of the file.
4. **Last Upload:** This column displays the time period of the last upload.

Fig. 7.16: *Get File*

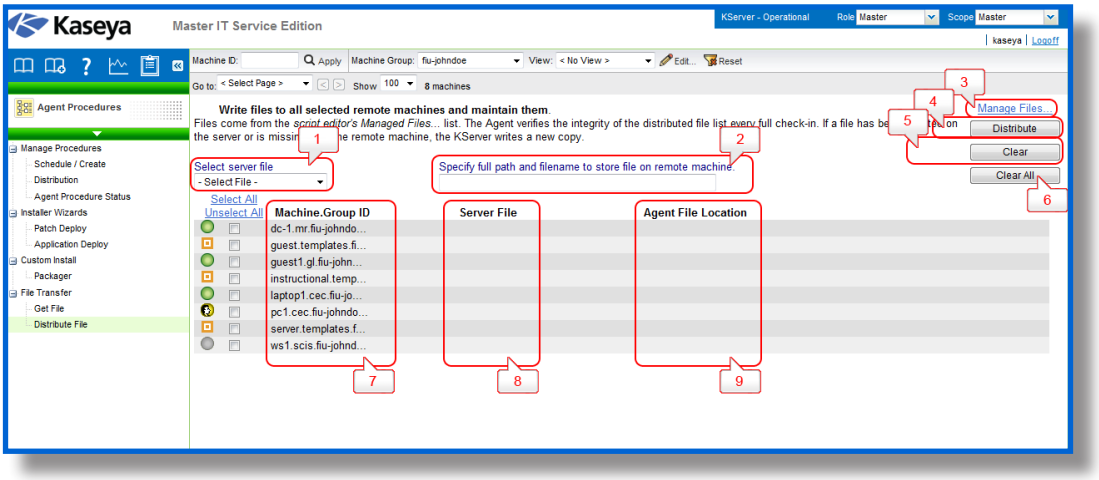


7.4.2 Distribute File

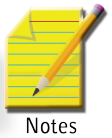
The **Distribute File** function sends files stored on your VSA server to managed machines. It is ideal for mass distribution of configuration files, such as virus foot prints, or maintaining the latest version of executables on all machines. The VSA checks the integrity of the file every full check-in. If the file is ever deleted, corrupted, or an updated version is available on the VSA, the VSA sends down a new copy prior to any procedure execution. Use it in conjunction with recurring procedures to run batch commands on managed machines.

Fig. 7.17 below shows the generic view of the Distribute File page. The functions supported by this page are listed and explained below.

Fig. 7.17: *Distribute File*



1. **Select server file:** Select a file to distribute to managed machines. These are the same set of files managed by clicking the *Manage Files* link on this page.



Note: *The only files listed are the private managed files or shared managed files. If another user chooses to distribute a private file, it is not visible.*

2. **Specify full path and filename to store file on remote machine:** Enter the path and filename to store this file on selected machine IDs.

3. **Manage Files:** Click the *Manage Files* link to display the *Manage Files Stored on Server* popup window. Use this window to add, update, or remove files stored on the KServer. This same window displays when you click the *Managed Files* button using Schedule / Create. Private files are listed with (Priv) in front of the filename.

4. **Distribute:** Click the *Distribute* button to start distribution management of the file selected in *Select server file* and write it to the location specified in *Specify full path and filename to store file on remote machine*. This affects all checked machine IDs.

5. **Clear:** Click the *Clear* button to remove the distribution of the file selected in *Select server file* from all checked machine IDs.

Warning: Clear *and* Clear All *do* not *delete the file from either managed machines or the KServer. These functions simply stop the integrity check and update process from occurring at each full check-in.*

6. **Clear All:** *Clear All* removes all file distributions from all checked managed machines.

7. **Machine.Group ID:** The list of Machine.Group IDs displayed is based on the Machine ID / Group ID filter and the machine groups the user is authorized to view.

8. **Server File:** The name of the file being distributed.

9. **Agent File Location:** The target directory on the managed machine. To the left of each target file location for a specific machine ID are two icons. Click to cancel that file distribution for that machine ID. Click to edit the destination path for that machine ID.