VMware vCenter Configuration Manager Troubleshooting Guide

vCenter Configuration Manager 5.6

This document supports the version of each product listed and supports all subsequent versions until the document is replaced by a new edition. To check for more recent editions of this document, see http://www.vmware.com/support/pubs.

EN-001042-00

You can find the most up-to-date technical documentation on the VMware Web site at:

http://www.vmware.com/support/

The VMware Web site also provides the latest product updates.

If you have comments about this documentation, submit your feedback to:

docfeedback@vmware.com

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Contents

About This Book

The *VMware vCenter Configuration Manager Troubleshooting Guide* explains problems that might occur with VMware vCenter Configuration Manager.

In addition, parts of this document describe how to find diagnostic information to help you or VMware Technical Support analyze problems.

Intended Audience

This information is for experienced Windows, Linux, UNIX, or Mac OS X system administrators who are familiar with managing network users and resources, and with performing system maintenance.

To use this information effectively, you must have a basic understanding of how to configure network resources, install software, and administer operating systems. You also need to fully understand your network topology and resource naming conventions.

Document Feedback

VMware welcomes your suggestions for improving our documentation. If you have comments, send your feedback to docfeedback@vmware.com.

Technical Support and Education Resources

The following technical support resources are available to you. To access the current version of this book and other books, go to http://www.vmware.com/support/pubs.

Online and Telephone Support

To use online support to submit technical support requests, view your product and contract information, and register your products, go to http://www.vmware.com/support.

Customers with appropriate support contracts should use telephone support for priority 1 issues. Go to http://www.vmware.com/support/phone_support.html.

Support Offerings

To find out how VMware support offerings can help meet your business needs, go to http://www.vmware.com/support/services.

VMware Professional Services

VMware Education Services courses offer extensive hands-on labs, case study examples, and course materials designed to be used as on-the-job reference tools. Courses are available onsite, in the classroom, and live online. For onsite pilot programs and implementation best practices, VMware Consulting

Services provides offerings to help you assess, plan, build, and manage your virtual environment. To access information about education classes, certification programs, and consulting services, go to http://www.vmware.com/services.

Troubleshooting Overview

1

Before making changes to your environment to solve a troubleshooting situation, learn as much as possible about the problem.

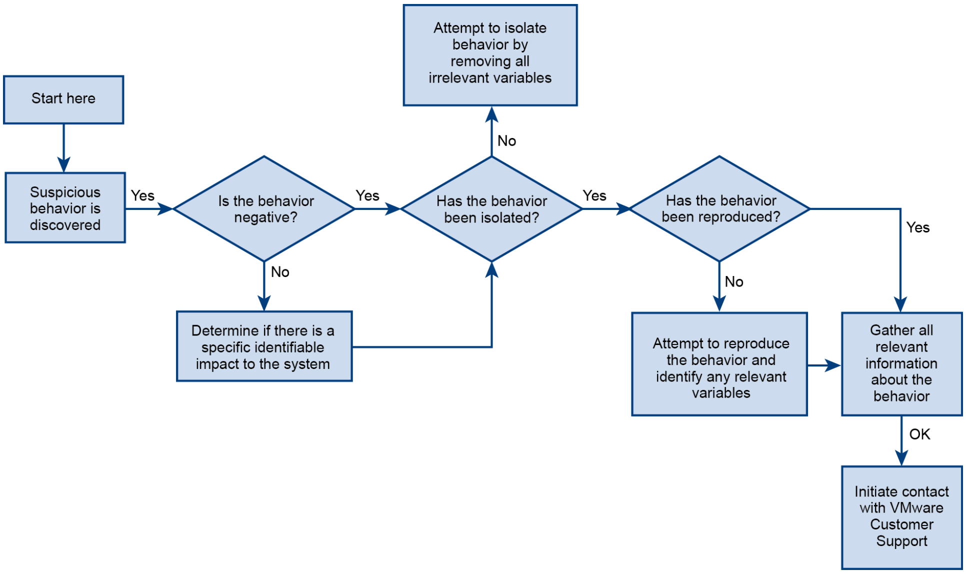
This chapter includes the following topics:

Troubleshooting Workflow

Many troubleshooting situations follow a workflow that might allow you to discover the cause of simple problems and correct them on your own.

In other cases, the workflow involves gathering information in the form of files or other evidence, and contacting VMware Technical Support for further assistance.

Figure 1–1. Troubleshooting Workflow



Start here
Suspicious behavior is discovered
Yes
Is the behavior negative?
No
Determine if there is a specific identifiable impact to the system
Yes
Has the behavior been isolated?
No
Attempt to isolate behavior by removing all irrelevant variables
Yes
Has the behavior been reproduced?
No
Attempt to reproduce the behavior and identify any relevant variables
Yes
Gather all relevant information about the behavior
OK
Initiate contact with VMware Customer Support

Verifying That Behavior Is Negative

Sometimes, an apparent error might not be a real error.

- A message might seem like an error when it is only a warning.
- A designed behavior might seem like an error if the result is not what you expect.

For example, in compliance, VCM highlights noncompliant systems because they *fail* to meet rules, but this convention might be the opposite of your own thinking. In other words, you might define rules that look for incorrect settings to eliminate, instead of correct settings to keep. When that happens, VCM marks machines noncompliant that you thought were configured properly. VCM is performing as designed though: the machines *failed* to have the incorrect settings.

The best first step to troubleshooting is to verify that you have a real problem. Click the context-sensitive Help buttons on the Console or wizards, and determine whether a message is only a warning, or whether the suspect behavior is an operation that is running the way that it should.

Isolating Behavior

After determining that you have a real problem, try to reduce the number of factors that might be causing the problem.

Eliminate, one by one, conditions that might be contributing to the behavior until a minimum of factors are still present when the error occurs.

For example, if you see a problem on a single machine during a collection on 50 machines, with 10 data types per machine, the first step is to eliminate the 49 machines that did not exhibit the problem and rerun the collection only against the problem machine. You can usually determine which machine failed by viewing the details of the job from the Job History screen under Administration. Assuming that the problem still exists, you then eliminate data types, one or two at a time, until you find the data type that is causing the behavior. Again, the job details might indicate which data types fail.

With process of elimination, you can understand and document when the undesired behavior occurs and you can reproduce it consistently.

Identifying External Factors

Sometimes a behavior cannot be predicted or isolated in VCM alone. When that happens, note the environment at the time that the behavior occurs.

Performance

System load can indicate performance problems.. VCM processes a large amount of data, and the servers that run the database and main application are subject to periods of high resource consumption. For Windows machines, the **Performance** tab in Windows Task Manager provides this kind of information.

On the Collector server and the Agent machines, note the system load, memory usage, network traffic, time of day, other running applications, or other details that might decrease performance. With this approach, you might be able to find a specific time or a set of other conditions that must be present for the undesired behavior to occur.

For example, every time a collection stalls at a specific step in the process, you notice that a single CPU is running at 100% capacity. The problem might be caused by a number of factors, but knowing that the CPU always correlates with the behavior is a significant piece of external evidence.

Operating System Logs

The operating system event log can identify external factors that might cause problems..You can detect simple problems by reviewing the security and application logs. Common errors include authentication problems:

```
Report Server (MSSQLSERVER) cannot connect to the report server database.
```

However, any message that occurs during the time frame surrounding the undesired behavior is suspect, especially if the same error or warning always accompanies the behavior.

Hardware

Errors and behaviors caused by hardware problems are among the most erratic. You might see a series of seemingly unrelated errors in sequence, or you might see the same error at random times. You might also see VCM operation degrade over time. This sort of irregular evidence might indicate a need for system hardware diagnostics.

Checking VCM Logs

During normal operation, VCM writes files with the .dbe extension to various locations, which store debug information about VCM.

The Collector writes additional debug information to the VCM database whenever jobs are running in the VCM Console.

The Collector log is vital because it records all primary functions of VCM, and when VMware Technical Support engineers mention the debug log in the singular, they mean this Collector log.

Debug logs are a common troubleshooting tool for VMware Technical Support, so if you have called for assistance before, you might already be familiar with the process of gathering them. You might even recognize certain messages that appear in the logs. Even if that is the case, be aware that the debug logs were not designed for the average user. The messages were written to help engineers understand why VCM is operating the way that it is. You are free to review the logs yourself, but usually you forward the logs in their entirety to VMware Technical Support for analysis.

While it is not necessary to know every message that a debug log might contain, it is helpful to know about the types of messages.

Info Messages

Although VCM is not configured to save them by default, info messages are the most common message category. VCM usually discards info messagesto reduce debug file size, but you temporarily enable the saving of info messages to help you are look for clues related to a problem.

After enabling info messages, nearly every action that VCM performs is recorded in debug logs. Even though the messages are only informational, they provide a context for error behavior and can be important in solving a troubleshooting problem.

Warning Messages

Warnings indicate an unexpected situation but do not necessarily indicate a permanent problem. Many common situations such as network timeouts or authentication failures produce warnings.

Warning messages usually do not prevent VCM from normal operation. They are intended to alert engineers about a possible problem.

Error Messages

Error messages indicate serious problems. VCM processes do not stop running because of an error, but VCM might not continue normally.

For example, if the collection of a specific data type produces a value that is not of the type that the database expects, VCM might have to discard the collected information for that entire data type. Discarding collected information produces an error message that indicates that something did not work properly. VCM can recover and continue, but the data you wanted for that collection is not correct.

Exception Messages

Exception messages indicate a problem that unexpectedly stops VCM because of a complete failure of the current process.

For example, the following exception might occur when there is a communication problem between the Collector and Agent.

```
A connection with the server could not be established HRESULT 0x80072efd;
```

Types of Problems 2

VCM problems usually fall into common categories.

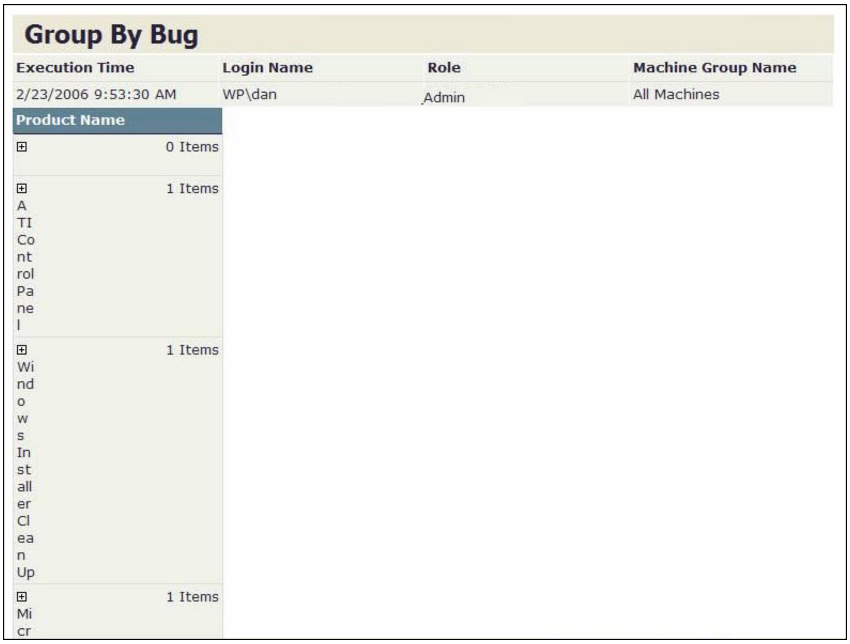
This chapter includes the following topics:

User Interface Problems

Unwanted VCM behaviors often reveal themselves in the user interface, but the user interface is rarely the root cause.

Common problems directly associated with the user interface are often display anomalies. Duplicate columns in the data grid, inappropriately enabled or disabled fields, and poor formatting are all examples of user interface problems. In this example, the root nodes of a report are incorrectly compressed into a space that is too narrow.

Figure 2–1. User Interface Formatting Problem



Group By Bug

Execution Time	Login Name	Role	Machine Group Name
2/23/2006 9:53:30 AM	WP\dan	Admin	All Machines

Product Name

⊞ 0 Items

⊞ 1 Items
A
TI
Co
nt
rol
Pa
ne
l

⊞ 1 Items
Wi
nd
o
w
s
In
st
all
er
Cl
ea
n
Up

⊞ 1 Items
Mi
cr

Security and Authentication Problems

VCM acts as a multiple-system administration tool and must have administrator authority on all the machines that it manages.

The necessary authority might be accidentally removed or restricted. Common authentication problems include changed passwords, dropped administrator access, or security measures such as a proxy server, that are added without updating VCM. In addition to managed machine access, VCM also requires access to its SQL Server database with the authority to insert, modify, and delete data.

Both kinds of access problems might reveal themselves in the user interface, in the VCM debug logs, or in the operating system event log:

```
The VCM Collector service failed to start due to the following error:
The service did not start due to a logon failure.
```

SQL Server Problems

Nearly all data associated with VCM is stored in its SQL Server databases.

Database	Description
VCM	Contains the collected managed machine data gathered from VCM Agents
VCM_Coll	Stores information about the user interface, such as Collector settings and options.
VCM_Raw	Performance-enhancing database that temporarily holds collection data before bulk insertion into the VCM database
VCM_UNIX	Contains collected managed machine data from any UNIX Agents in the environment

SQL Server errors might include resource, disk space, or authentication problems, among others. Poor tuning of your SQL Server might also cause performance problems such as data bottlenecks. Evidence of SQL Server trouble might appear in the user interface or the debug logs, but messages directly related to SQL Server typically appear directly in the SQL Server logs:

```
INSERT statement conflicted with COLUMN FOREIGN KEY constraint 'fk_vcm_
sysdat_role_rules_role_id'. The conflict occurred in database 'VCM', table
'vcm_sysdat_rules', column 'rule_id'.
```

VCM Agent Problems

The VCM Agent is the locally installed mechanism by which VCM collects information from a managed machine. If the Agent is not functioning properly, collections from that machine might fail.

In the VCM Console, the Running Jobs display is the first place to check when looking for problems with an Agent. A typical error message for an Agent that cannot start collecting might be:

```
There was a problem parsing the agent instructions document.
```

The message indicates that a conflict occurred in the instruction set sent to the Agent, the Agent could not process its instructions, and collection failed as a result.

UNIX Agent Problems

UNIX operates differently from Windows. To accommodate the differences, VCM uses a separate Agent for Linux or UNIX based managed machines.

Troubleshooting UNIX Agent problems might be easier than Windows, because certain operations available for Windows Agents are not available on the UNIX side. For example, compliance enforcement is not available for UNIX, so you do not need to investigate compliance messages when you troubleshoot on UNIX.

Report Server Problems

The Report Server is responsible for the graphical display of information in VCM and for VCM scheduled reports. A Report Server error usually appears in the user interface.

```
An error has occurred during report processing.
System.Web.Services.Protocols.SoapException: An error has occurred during
report processing -
Microsoft.Reporting.Services.Diagnostics.Utilities.RSException: An error has
occurred during report processing -
. . .
```

You can determine if a problem exists with the Report Server by navigating to the Report Server home page to see if the native interface for the Report Server is working. The URL is usually the name or IP address of the Report Server machine followed by Reports:

`http://`*`report-server-name-or-IP`*`/Reports`

If the same error seen in the VCM user interface also occurs directly in the Report Server interface, the problem is almost certainly with the Report Server itself, and not with VCM. If the Report Server interface does not show the error, the cause is likely in VCM.

Internet Information Services Problems

The VCM user interface is hosted on Microsoft Internet Information Server (IIS), which might be incorrectly configured.

Sometimes the VCM user interface displays an error similar to what you see when you fail to connect to a Web page. Error messages such as `404 File Not Found` or `403 Forbidden` are typical IIS messages, and indicate improper configuration of IIS in relation to VCM or Report Server pages. You might also see ASP.NET errors:

```
Server object error 'ASP 0177:8007007e'
Server.CreateObject Failed
/vcm5/L1033/SumAdminDeployment.asp, line 68
8007007e
```

Network Connectivity Problems

The VCM Collector must have a network connection to all the Agents that send data. In addition, split installations require network connections between the separate VCM servers: Collector, SQL database, and Web server.

Failed network connections do not usually occur between all systems at the same time, so network issues are often easier to diagnose. A failed connection typically appears in the user interface as a failure of a VCM or VCM Patching job on a single or subset of managed machines. In the Jobs Detail display, a `PingFailed` message indicates connectivity problems.

A way to test network connectivity is to ping the Agent from the Collector. From the Collector, it might be necessary to connect to the HTTP port of an Agent if any of the following are true:

- The environment contains a firewall.
- The Agent is a UNIX Agent.
- The Agent machine is using HTTP instead of the DCOM protocol for its primary communication method.

To connect to the HTTP port, type **`telnet`** ***`agent-machine-name-or-IP`*** **`26542`** at the command prompt. A successful command returns only a blank screen, which indicates that the managed machine has answered the connection request and awaits further instructions. You can break the connection and exit telnet by typing **`Ctrl+]`** and then **`quit`**.

If you send the telnet command and receive any other message (such as `Connection refused` or `Connection timeout`) that is a good indication of a network problem. Failed network connections must be resolved with the help of local support before further VCM troubleshooting. Otherwise, you cannot know whether or not the problems you are seeing are network related.

Hardware and Performance Problems

Hardware and performance problemsare among the most difficult to diagnose because they are often intermittent and random.

Sometimes, the only common denominator is that the problem is random. When that happens, look for possible hardware or performance evidence.

Running out of disk space is the most common hardware problem. On the VCM servers, use the operating system file or disk management tools to check that enough disk space is available for the database and for the VCM application itself.

Next, make sure that enough memory and CPU cycles exist for VCM services and processes to start and continue properly. If VCM is competing with other, non-VCM processes, performance might degrade to the point that errors and exceptions appear in the debug log.

Finally,you might need to run diagnostics on hardware components, such as memory chips, processors, or system boards.

For the recommended hardware sizing and configuration needed to run VCM, see the installation documentation.

Gathering Diagnostic Information 3

To solve a problem, you need to collect information for your own analysis or for forwarding to VMware Technical Support.

This chapter includes the following topics:

What to Send to VMware Technical Support

To help VMware Technical Support analyze a VCM problem, you often need to gather and send files, exports of system logs, message text, or interface images.

Table 3–1. What to Send to VMware Technical Support

Type of Problem	What to Send
User interface	Screenshots
Security and authentication	Screenshots Error messages VCM debug logs

Type of Problem	What to Send
SQL Server	SQL Server logs Windows system and application event logs VCM debug logs
VCM Agent	Collector debug log Agent ARS files Windows system and application event logs
UNIX Agent	Collector debug log UNIX Agent debug log UNIX Agent ZRP files UNIX system log
Report server	Screenshots SQL profiler trace files Collector debug log
IIS	Screenshots Entries from IIS logs
Network connectivity	Network connectivity problems are usually investigated by local site support.
Hardware and performance	Hardware and performance problems are usually investigated by local site support.

Capture a Desktop Image

Desktop screenshots capture the exact behavior that you see, in the broad context of your entire workspace.

Use desktop screenshots alone or in a series to capture error messages, changes in behavior over time, or to verify data that you enter in wizards or other interfaces.

Procedure

1. On the keyboard, press `Print Screen` (`PrtScn`).
2. Open a new message or document.
3. Press `Ctrl+v` to paste the image into the message or document.

Capture a Window Image

Window screenshots capture the exact behavior that you see, with the focus on one interface.

Use window screenshots alone or in a series to capture error messages, changes in behavior over time, or to verify data that you enter in wizards or other interfaces.

Procedure

1. To bring it into focus, click the window that you want.
2. On the keyboard, press `Alt+Print Screen` (`PrtScn`).
3. Open a new message or document.
4. Press `Ctrl+v` to paste the image into the message or document.

Set the Debug Log to Store all Message Types

Because Info messages might contain important troubleshooting clues, turn them on before extracting the log.

By default, the Collector debug log saves on performance and space by not storing Info messages. After extracting the debug log, turn Info messages back off.

CAUTION This procedure involves editing the Windows Registry.

Procedure

1. In the VCM Console, click **Administration**, and select **Settings** > **General Settings** > **Collector**.
2. In the Description column, select **Type of information that should be logged**, and click **Edit Settings**.
3. Select all message types: Exception, Error, Warning, and Info.
4. Follow the prompts to finish turning on the messages, and click **Finish**.
5. Repeat steps 1 through 4 for the following **Administration** settings:
   - **Settings** > **General Settings** > **Database**
   - **Settings** > **Windows** > **Agent - General**
   - **Settings** > **UNIX** > **Agent - General**
6. On Windows Agent machines, edit the Windows Registry to create a DWORD under the following Registry key.

   ```
   HKEY_LOCAL_MACHINE\SOFTWARE\Configuresoft\CSI\5.0\Common\DebugEvent
   ```

   DWORD name = `Filter`
   DWORD value = `0000003C`
7. On the Collector, in the Windows Services Manager, restart the VCM Collector service.

What to do Next

Extract the log.

Extract the Debug Log

To create a DBE file from your Collector debug log messages, extract the log from the database.

Prerequisites

Configure the debug log to capture all messages, including Info messages. See"Set the Debug Log to Store all Message Types" on page 21.

Procedure

1. To put the full set of messages into the log, rerun the job that caused the problem you are troubleshooting.
2. After the job completes, wait five minutes before proceeding.
3. In Windows, navigate to the following VCM tools folder.

   (By default) `C:\Program Files (x86)\VMware\VCM\Tools`
4. To open the debug log viewer, right-click the following executable, and**Run As Administrator**.

```
ECMDebugEventViewer.exe
```

5. Click **Filter Settings**.
6. In the **Message Type** and **Message Source** areas, select all of the check boxes and click **OK**.
7. In the **Data Source** area, type the names of the servers and databases and click **OK**.
8. Click **Date/Time**, and select the **between** option.
9. Specify the start and end times when the job ran, and pad the times with an additional five or more minutes at each end.
10. Click **File**, and select **Fetch**.

    The displayed data refreshes.
11. Click **File**, and select **Fetch Next**.

    Continue the **Fetch Next** process until no additional data is added to the displayed debug log.
12. Click **File**, and select **Save as DBE**.
13. Name the DBE file and note where it is saved.

What to do Next

Using "Set the Debug Log to Store all Message Types" on page 21 as a guideline, restore the original logging levels. Usually, you only log Exception, Error, and Warning messages.

Extract SQL Server Logs

To save messages about database operations, extract SQL Server logs.

Procedure

1. On your SQL Server machine, open **SQL Server Management Studio**.
2. Select the server name and authentication method, and click **Connect**.
3. In the **Object Explorer** pane, expand ***server-name*** > **Management** > **SQL Server Logs**.
4. Right-click a log, and select **View SQL Server Log**.

    The Log File Viewer displays the logs.
5. Click **Export**.

    The **Export Last Fully Retrieved Log** dialog box appears.
6. Save the logs.

    Give each saved file a meaningful name and note where it is saved.

Collect IIS Logs

To save messages about VCM Web server operations, extract Internet Information Services (IIS) Server logs.

Procedure

1. On the Web server, select **Start** > **Administrative Tools** > **Internet Information Services (IIS) Manager**.
2. Expand **Internet Information Services** > ***server-name*** > **Web Sites**.
3. Right-click **Default Web Site**, and select **Properties**.
4. Verify that the **Enable Logging** check box is selected.
5. In the **Active log format** drop-down menu, select **W3C Extended Log File Format**, and click **Properties**.
6. Make a note of where the logs are stored.

   The default location is `C:\WINDOWS\system32\LogFiles\W3SVC1\`

   Each log file has the notation `u_ex`*`yymmdd`*`.log` where *`yymmdd`* represents the date.
7. In Windows, copy the log files for the timeframe that you want.

Collect ARS Files

ARS files are raw data files sent to the Collector by the Agent. The Collector deletes the files after processing the data, but you can configure the Collector to keep the files.

ARS files might hold valuable troubleshooting clues.

CAUTION This procedure involves editing the Windows Registry.

Procedure

1. On the Collector, type **`regedit`** at the command prompt to open the Registry.
2. Navigate to the following key:

   (32-bit) `\HKEY_LOCAL_MACHINE\Software\Configuresoft\ECM\4.0\Agent`
   (64-bit) `\HKEY_LOCAL_MACHINE\Software\Wow6432Node\Configuresoft\ECM\4.0\Agent`

   If the `Agent` key does not exist, navigate to `4.0`, right-click, and select **New** > **Key** to create it.
3. Under `Agent`, verify that the `AreResultsSaved` key value is 1.

   If `AreResultsSaved` does not exist, create it as a new DWORD and set its value to 1.
4. Navigate to the following key:

   ```
   \HKEY_LOCAL_MACHINE\Software\Configuresoft\ECM\4.0\Collector
   ```

   If the `Collector` key does not exist, navigate to `4.0`, right-click, and select **New** > **Key** to create it.
5. Under `Collector`, verify that the `AreResultsSaved` key value is 1.

   If `AreResultsSaved` does not exist, create it as a new DWORD and set its value to 1.
6. Close the Registry.
7. Stop all running jobs, or make sure that no jobs are currently running.
8. Navigate to the following VCM folder:

   The default location is `C:\Program Files (x86)\VMware\VCM\CollectorData`
9. Wait until no jobs are running, and delete any existing folders under the directory.

10. Start a collection, and note its Job ID as seen in the Running Jobs window.
11. After the job is finished, look for a CollectorData subfolder named with the Job ID of the job that just finished.
12. Create a ZIP file of the entire subfolder.

    The ZIP file is what you send to VMware Technical Support.

What to do Next

Reopen the Registry, and set the two `AreResultsSaved` values to 0.

Collect the UNIX Syslog Messages

The UNIX syslog functions like the Event Log in Windows, because it records a large amount of detailed system messages.

NOTE The `messages` file location varies.

Procedure

1. Log in to the UNIX Agent system as the root user.
2. Use `cat`to view the contents of the `/etc/syslog.conf` file.

   In the file, look for an entry that indicates where the `messages` file is located.
3. Find the `messages` file itself, and transfer a copy of it to your local workstation or Collector.
4. Look for log files that have been rotated, and copy those as well.

   Rotated files typically have some value appended to their name, such as `/var/log/messages.1`.

Collect Import/Export Tool Logs

The VCM import/export tool creates a debug file that might provide troubleshooting clues to data import or export problems.

Procedure

1. Log in to the VCM machine where the import/export tool is installed.
2. Navigate to the following VCM folder:

   The default location is `C:\Program Files (x86)\VMware\VCM\Tools\ImportExport`
3. Copy any DBE files in the folder.

Extract Windows Event Logs

The Windows Event Log categories are Application, Security, and System. VCM errors almost always appear in the Application or System categories.

Procedure

1. On the Collector, select **Start** > **Administrative Tools** > **Event Viewer**.
2. On the left, locate and highlight the Application or System log.
3. From the pull-down menus, select **Action** > **Save Log File As**.
4. Type an appropriate name and click **Save**.

Extract Windows System Information

Windows includes an executable application file `msinfo32.exe` that can provide a detailed snapshot of the current state of a system.

Procedure

1. On the Collector, click **Start** > **Run**.
2. Type **`msinfo32.exe`**in the text box.
3. From the pull-down menus, click **File** > **Save**.
4. Type a meaningful file name, and click **Save**.

   The save process might take a few minutes to create the NFO file that holds the detailed snapshot.

Collect UNIX ETL Logs

UNIX extract, transform, load (ETL) logs record the detailed workings of the ETL service.

Procedure

1. On the Collector, navigate to the following folder:

   The default location is `C:\ProgramData\Configuresoft\ECM\ExceptionLog`
2. Copy the log files that begin with `etl`.

Collect VCM Installation Logs

When installing VCM, the installer script writes log files that record the progress and status of the installation.

Procedure

1. On the Collector, log in as the user who installed VCM.
2. Select **Start** > **Run**.
3. In the text box, type **`%TEMP%`** to open the `\Temp` directory.

   If you are not taken to the `\Temp` directory, it might be because of the `FlatTempDir` Registry key. See Microsoft Knowledge Base article 243215 for more information.
4. Make a ZIP file of all the files in the `\_csi_installation` folder.

Enable VCM Patching Logging

The VCM Patching module creates its own logs, which supplement the VCM debug file. VCM Patching logging is disabled by default.

Procedure

1. On your SQL Server machine, open SQL Server Management Studio.
2. Connect to the VCM database using your preferred authentication method.
3. From the toolbar, click **New Query**.

   A blank query pane appears.
4. From the toolbar drop-down menu, select the VCM database.

5. To enable VCM Patching debugging, type and run the following query:

   ```
   update csi_hf_settings set val = '1' where setting = 'debug'
   ```

6. In Windows, in the Services Manager, restart the VCM Patch Management service.

What to do Next

Collect the VCM Patching logs.

Collect VCM Patching Logs

For problemswith the VCM Patching process, collect the VCM Patching log files.

Prerequisites

Turn on VCM Patching logging. See "Enable VCM Patching Logging" on page 25.

Procedure

1. With logging on, rerun the patching that produced the unwanted behavior.
2. On the Collector, navigate to the following VCM folder.

   The default location is `C:\Program Files (x86)\VMware\VCM\SUM\Collector`

3. Copy the following TXT files.

   ```
   CSISUMWorker_SumDBDebug.txt
   CSISUMSvc_SumDBDebug.txt
   CSISUMSvc_debug.txt
   ```

What to do Next

Using "Enable VCM Patching Logging" on page 25 as a guideline, turn VCM Patching logging back off. In the query, set the value to zero (0).

Collect Agent Logging

To collect Agent logs, modify the logging configuration file on the Agent.

Procedure

1. Locate the following file.

   The default location is `*agent-path*\Installers\Providers\Logging.conf`

   If you are on a Collector, the default path for the same file is:
   `C:\Program Files (x86)\VMware\VCM\Installer\Providers\Logging.conf`

2. Make a copy of `Logging.conf` so you can restore it later.
3. Open `Logging.conf` in a text editor.
4. Locate the `[handler_fileHandler]` section.
5. Change the `args` setting in that section to the directory where you want to store logs. For example:

   ```
   args=('C:\tmp\provider.log', 'a')
   ```

6. From the Collector, rerun the action that caused the problem.
7. Review the log stored in the directory that you specified.

   The directory contains all output generated during precollection, the action itself, and post-collection.

What to do Next

Using the backup copy, restore `Logging.conf` to its previous state.

4 Troubleshooting Problems with VCM

This information describes troubleshooting situations that occur in general VCM operations such as installing, upgrading, patching, or reporting.

This chapter includes the following topics:

Patch Content Does Not Download for Red Hat and SUSE Machines

Patch assessment content fails to download on certain Red Hat and SUSE managed machines.

VCM 5.6 and later selectively download bulletins. For example, if the inventory contains no Red Hat machines, VCM does not download Red Hat bulletins.

Problem

The log file reports missing bulletins or filters for patch assessments on Red Hat and SUSE machines that run a VCM Agent older than version 5.4.1.

```
Level: Error
Message: No filters found for filter set 66 [ RequestId = your-request-ID,
Requestor = Collector-name ] HRESULT 0x00040ec6;
```

```
Level: Warning
Message: No available bulletins matched filter for machine id: machine-ID.
Setting filter to: !/*
```

Cause

Administration settings are not configured to enable patch assessment for Agents older than version 5.4.1.

Solution

1. Click **Administration**, and select **Settings** > **General Settings** > **Patching** > **UNIX** > **Additional Settings**.
2. Based on the Agent version, configure the options.

Option	Action
Enable Assessment Content for 5.4.1 and later Linux Agents	Set the value to **Yes** when any of the following are true. ▪ Your managed machines have only the 5.4.1 or later Agent installed. ▪ Your managed machines have a combination of pre-5.4.1 and 5.4.1 or later Agents installed.
Enable Assessment Content for pre-5.4.1 and earlier Linux Agents	Set the value to **Yes** when any of the following are true. ▪ Your managed machines have only pre-5.4.1 Agents installed. ▪ Your managed machines have a combination of pre-5.4.1 and 5.4.1 or later Agents installed. After you enable content for pre-5.4.1, the job does not run immediately. Wait for the next scheduled job, or manually initiate a UNIX bulletin download. ▪ To download UNIX bulletins, click **Patching**, and select **All UNIX/Linux Platforms** > **Bulletins** > **By Bulletin** > **Check For Update**. ▪ To monitor the job, click **Patching**, and select **VCM Patching Administration** > **UNIX** > **Job Manager** > **Running**.

UNIX Patch Deployment Fails

Reading or retrieving a patch bulletin name causes UNIX patch deployment to fail.

Problem

During UNIX patch deployment, VCM reports an error similar to the following message:

```
spiChangeResultsHelper->GetChangeActionInfo HRESULT 0x80070057 = The
parameter is incorrect.
```

In addition, the following exception messages appear in the Collector debug log.

```
Invalid Argument because an STL container is empty: m_deqBulletins HRESULT
0x80070057 = The parameter is incorrect.

spiChangeRequestPopulator->Initialize() HRESULT 0x80070057 = The parameter is
incorrect.
```

Cause

The patch bulletin name contains information that VCM cannot process.

Solution

Because this failure originates in the patching content, contact VMware Technical Support, and provide details about the bulletin being used.

UNIX Patch Assessment Returns No Results

UNIX patch assessments do not finish successfully.

Problem

UNIX patch assessments do not finish successfully and do not display assessment results. Assessments must succeed before VCM Patching for UNIX can install patches.

Cause

The problem can occur because of any of the following situations:

- The assessment template contains patch bulletins that do not match the selected machine type.
- The selected patch is for the wrong machine architecture, 32-bit or 64-bit.
- You defined a custom filter for Patch Assessment that caused bulletins to not match the selected machine type.

  For example, a filter attribute such as Severity might not apply to bulletins for some platforms. If you defined a filter based on Severity, the bulletins do not match platforms that do not use Severity.
- For VCM 5.1 Agents or earlier, you have not run a **Machines - General** collection.
- The bulletins are missing from their required location on the UNIX Agent machine.

Solution

Use these corrective measures, respectively:

- Review the patch bulletins to make sure they match the selected machine type.
- Review the patch bulletins to make sure they match the selected machine architecture, 32-bit or 64-bit.
- Review or change the custom filter to remove attributes that the selected machine type does not support.
- VCM supports UNIX Patch Assessment custom filters, which can be used when assessing older Agents. For 5.1 or earlier Agents, first collect the **Machines - General** data class so that the assessment can succeed.

  For 5.1.x or later Agents, you do not need to first perform a **Machines - General** collection.
- See "UNIX Bulletins Missing from the Required Location" on page 31.

UNIX Bulletins Missing from the Required Location

UNIX patch bulletins are not in the required location on the Agent machine.

Problem

UNIX patch bulletins are not in the required location on the Agent machine, which causes UNIX patch assessments to fail and not display assessment results.

Cause

The problem can occur because of any of the following situations:

- The Agent version does not match the UNIX platform support for Patch Assessment.
- The process of distributing the bulletin information to the UNIX Agent machine failed.
- The bulletin information was removed from the UNIX Agent machine.
- Bulletin information is not loaded on the Collector.
- A Collector upgrade failed to reprocess bulletin information.

Solution

Use these corrective measures, respectively:

- Make sure the Agent version is supported for how you are performing UNIX Patch Assessment:

  Agent earlier then 5.0. No support.

  Version 5.0 Agent. Support for some UNIX platforms, but requires manual distribution of bulletin information to the UNIX Agent machine.

  Version 5.1 Agent and later. Support for additional UNIX platforms, along with automated distribution of bulletin information to the UNIX Agent machine.

- Retry the process of distributing the bulletin information to the UNIX Agent machine.
- Add the bulletin information back to the UNIX Agent machine.
- Check for updates on the Collector. If the check reports that no updates are available, run **Check for Updates** with the **Force** option.
- The Collector should reprocess bulletin information during an upgrade. If it did not, running **Check for Updates** with the **Force** option might correct the problem.

Report and Node Summary Errors

You see error messages when looking at reports and node summaries.

Problem

After installing or upgrading VCM, you see any of the following errors when you look at reports and node summaries.

```
Server Unavailable

The web application you are attempting to access on this web server is
currently unavailable.

Client found response content type of "text/html" but expected "text/xml".

No results returned for specified parameters.
```

The last error might occur even if the report appears to run, and part of the report appears.

Cause

Problems with Visual Studio 2005 and the .NET Framework are responsible for these errors.

Solution

1. Go to the Microsoft Web site.
2. Search for Knowledge Base article KB913384.

   The article describes a hotfix for the following problem:

   ```
   A .NET Framework 2.0 application that runs under a user account context
   when no user profile is associated with the user account context might
   crash, or you might receive an access violation error message.
   ```

3. Download and apply the hotfix for your machine.
4. Restart the machine.

Report Parameter Errors

A report does not contain the correct parameters.

Problem

After upgrading VCM, a report contains parameters that are out of date or wrong.

Cause

Report parameter values changed, but Report Manager did not handle the changes. The report was not uploaded correctly, and the error occurs because the report is overwritten instead of being first removed in Report Manager.

Solution

Remove the existing report, and upload it again.

1. On the Collector, open Report Manager:

   `http://collector-name-or-IP-address/Reports.`

2. In Report Manager, open the folder that holds the affected report.

Folder	Report
ECMAD	Active Directory
ECMu	UNIX
RSCA	RSCA
Service Desk	Service Desk and Change Reconciliation
SMS	Systems Management Server
Standard	Windows reports, and Change Management and Compliance
SUM	VCM Patching
Virtualization	Virtualization

3. On the right, click **Show Details**.
4. Select the check box next to the affected report.

5. Click **Delete** and click **OK**.
6. Click **Upload File**.
7. On the **Upload File** page, next to the **File to Upload** text box, select **Browse**.
8. From the reports directory, select the report.
9. Click **OK**.

The report now includes all of the new parameter modifications.

Protected Storage Errors

Trying to generate key pairs on the Agent Proxy machine results in a protected storage error.

Problem

When you attempt to generate key pairs on the Agent Proxy machine, a protected storage error similar to the following message appears:

```
CsiCommProxyUtil::wmain(): Failed to get protected storage for VCMv. HRESULT
0x8009000b = Key not valid for use in specified state.
```

Cause

Files that are not accessible because they reside in a protected folder are preventing the key generation process from succeeding.

Solution

1. On the Agent Proxy machine, open the command prompt.
2. Change directory to the following VCM folder.

   The default location is `C:\Program Files (x86)\VMware\VCM\AgentData\protected`
3. Delete the following files:

   ```
   ECMv.csi.pds
   ECMv.csi.pds.lck
   ```
4. Run the following command:

   ```
   GenerateAgentProxyKeys.cmd
   ```
5. Verify that the following files were generated:

   `agent-proxy-machine-name``_securecomm_public_key.txt`
   `agent-proxy-machine-name``_ssh_public_key.txt`
6. Run the following command:

   `DatabaseUploadKey.cmd` *`agent-proxy-machine-name`*`_securecomm_public_key.txt`

SSL Becomes Disabled

During a Collector upgrade, the option for secure socket layer (SSL) communication is turned off.

Problem

The check box to require a secure SSL channel becomes unchecked in the VCM virtual directory properties. After upgrading, you are then logging in to VCM over an unsecured connection without HTTPS.

Cause

The upgrade process causes this problem on a VCM Collector that is using SSL, even though the settings for secure SSL were in effect before starting the upgrade.

Solution

Restore the VCM virtual directory setting:

1. From a command prompt on the VCM Web server, type `compmgmt.msc`.
2. Expand the **Services and Applications** node, and expand **Internet Information Services** > **Web Sites** > **Default Web Site**.
3. Right-click the VCM virtual directory, and select **Properties**.
4. Click the **Directory Security** tab, and in the **Secure Communications** panel, click **Edit**.
5. Select the **Require secure channel (SSL)** check box, and click **OK** twice.

Restore the IIS setting in VCM:

1. On the VCM Collector, select **Administration**.
2. Select **Settings** > **General Settings** > **Database**.
3. Select **IIS HTTP string http or https**.
4. Click **Edit Setting** and change the IIS HTTP string setting to **https**.

OS Provisioning Troubleshooting 5

This information describes troubleshooting situations found in VCM operating system provisioning.

This chapter includes the following topics:

Required Services Not Running

One or more of the required services on the OS Provisioning Server is not running.

Problem

When you run the `service FastScale status` command, you see that one or more of the OS Provisioning Server services did not start or was stopped.

Cause

The OS Provisioning Server IP address is set to a value other than its default of `10.11.12.1`, or one or more services failed to start or stopped.

Solution

1. Change DHCP to the IP address assigned to the OS Provisioning Server.
2. From a command prompt on the OS Provisioning Server, type **`service FastScale stop`**.

   The command shuts down all OS Provisioning Server daemons.
3. To restart the OS Provisioning Server, type **`service FastScale start`**.
4. Type **`service FastScale status`**, and verify that all services are running.

Distribution File Cannot Be Opened

A Red Hat OS distribution encounters a network problem during installation, and the installation fails.

Problem

When installing Red Hat OS distributions, the installation fails and displays the following error on the target machine:

```
The file distribution-file cannot be opened
```

Cause

This is a known issue in Red Hat installers. During a network installation, the installer does not retry the installation if it encounters a transient network problem.

Solution

The error on the target machines includes options to Reboot or Retry. Select **Retry** to complete the installation.

Unable to Retrieve Stage2.img Error

A Red Hat OS distribution fails because of a DHCP time out during installation.

Problem

When installing Red Hat OS distributions, the installation fails and displays the following error on the target machine.

```
Unable to Retrieve http: /path.Stage2.img Error
```

If you click OK, the error persists.

Cause

This is a DHCP timeout problem caused by the way network interfaces are configured. During installation of the distribution, DHCP times out, and the installation fails while getting the `Stage2.img` file.

When the kickstart action requests a DHCP IP address, the interface is removed and then redisplayed. The DHCP time out is shorter than the time needed for the switch ports to go into a forwarding state.

Solution

Modify the values in the following kernel command line arguments in the PXE boot script for Red Hat.

Option	Description
`dhcptimeout=`*x*	Stops attempting to get a DHCP lease after *x* seconds.
`nicdelay=`*x*	Sleeps for *x* seconds before trying again to access the network.
`linksleep=`*x*	Checks the network device for a link once every second for *x* seconds.

1. On the OS Provisioning Server, edit the following file:

   ```
   /opt/FastScale/var/fsadmin/include/config.inc
   ```

2. Modify the values, and save the file.

   ```
   $rhelDhcpTimeout = 120;
   $rhelNicDelay = 60;
   $rhelLinkSleep = 60;
   ```

3. Provision the machine with a Red Had OS distribution.

Corrupt Media or Cannot Access Installation Media Errors

A custom Red Hat or SUSE distribution is missing required files.

Problem

When installing the distribution, the installation fails and displays errors on the target machine.

- **Red Hat:**

  ```
  The file file-name.rpm cannot be opened. This is due to a missing file, a
  corrupt package or corrupt media. Please verify your installation source.
  ```

- **SUSE:**

  ```
  Cannot access installation media operating-system-name. Check that the
  server is accessible.
  ```

Cause

The custom ISO that you are using to install Red Hat or SUSE is missing one or more of the packages required by VCM.

Solution 1

Add the missing packages to the ISO image.

1. On the OS Provisioning Server, review the list of packages required by VCM.
   - **Red Hat:** `/FSboot/repository/linux/`*`RHEL-version`*.

     For example, `/FSboot/repository/linux/RHEL6.0server-x86_64/packages`
   - **SLES 10.3:** `/opt/FastScale/var/fsadmin/jobs/SLES10.0_sp3.basic.php`
   - **SLES 11.1:** `/opt/FastScale/var/fsadmin/jobs/SLES11.0_sp1.basic.php`
2. If packages or dependency packages are in the list but not in the ISO, add them to the ISO.
3. Reimport the distribution to the OS Provisioning Server repository.
4. In VCM, run the Provision wizard again to create a newly configured session with the updated distribution.

Solution 2

Modify the package list to account for the missing packages.

NOTE Changes to the list apply to any future provisioning until you add the packages back to the list.

1. On the OS Provisioning Server, edit the list of packages required by VCM.
   - **Red Hat:** `/FSboot/repository/linux/`*`RHEL-version`*.

     For example, `/FSboot/repository/linux/RHEL6.0server-x86_64/packages`
   - **SLES 10.3:** `/opt/FastScale/var/fsadmin/jobs/SLES10.0_sp3.basic.php`
   - **SLES 11.1:** `/opt/FastScale/var/fsadmin/jobs/SLES11.0_sp1.basic.php`
2. In the list, remove the packages or dependency packages that you are not including in your custom ISO.
3. In VCM, run the Provision wizard again to create a newly configured session with the updated distribution.

Windows OS Installation Failure

The Windows operating system fails to install because of hardware or virtual machine incompatibility.

Problem

During installation of a Windows OS distribution, the screen of the target machine turns blue and the `STOP: 0.0000007B` error code appears.

Cause

The target hardware requires drivers that are not included in the operating system distribution. The error often occurs with distributions of older versions of Windows.

Solution

Install the Windows operating system on a machine that is compatible with the hardware and drivers. You can also try installing Windows on an incompatible machine by turning off the advanced feature in the BIOS of the storage hardware, and using Compatibility Mode when installing the OS.

Windows 2008 R2 OS Provisioning Failure

A Windows 2008 R2 OS provisioning action fails.

Problem

When provisioning a target machine with Windows 2008 R2, the provisioning job fails and displays an error about a null XML field.

Cause

The License Key Type was not specified when you configured the provisioning action in the Provision Operating System wizard.

Solution

Rerun the wizard, making sure to complete all required fields, including key type. Key type examples include retail versus volume licensed Windows 2008 R2.

Software Provisioning Troubleshooting 6

This information describes troubleshooting situations found in VCM software provisioning. Software provisioning issues usually fall into one of the following categories:

- The software repository
- Package Studio
- Package Manager or commands to Package Manager, which are issued from VCM

This chapter includes the following topics:

Package Deleted

A deleted package cannot be found when you request it from the repository.

Problem

An error appears, stating that the package CRATE file cannot be found.

Cause

The package CRATE file was deleted from the repository.

Solution

1. Remove the package entries from the following file:

   The default location is `C:\Program Files (x86)\VMware\VCM\Tools\Repository\.hive\repository.index`

2. Reindex all platforms and sections from which you removed the entry.

   To reindex a repository configured at `C:\Repository`, run the following command:

   ```
   C:\Repository> "C:\Program Files (x86)\VMware\VCM\Tools\Package
   Studio\hive.exe" reindex "C:\Repository\dists\Any\Release\binary-platform"
   platform section
   ```

3. Use Package Studio to republish the package to the repository.

Package Copied Instead of Published

A copied package cannot be found when you request it from the repository.

Problem

An error appears, stating that the package CRATE file cannot be found.

Cause

The package CRATE file was copied to the repository rather than published.

Solution

Publish the package to the repository using Package Studio.

Do not directly publish the already-copied version of the package. Publish a different, locally saved version, or copy the already-copied version to another machine and publish it from there.

Repository Reindex Needed

A copied package cannot be found because the repository needs to be reindexed.

Problem

An error appears, stating that the package CRATE file cannot be found.

Cause

The package was copied to the repository, and the `Repository.index` file was updated, but the platform and section were not reindexed, and the package was not added to the `crates.gz` file.

Solution

Reindex all platforms and sections. To reindex, run the following command:

```
C:\Repository> "C:\Program Files (x86)\VMware\VCM\Tools\Package
Studio\hive.exe" reindex "C:\Repository\dists\Any\Release\binary-platform"
platform section
```

Manually Reorganized Repository

A package cannot be found because the repository was manually reorganized.

Problem

An error appears, stating that the package CRATE file cannot be found.

Cause

Someone manually reorganized the locations of platforms and sections in the repository.

Solution

1. Edit the `Repository.index` file to remove the old locations and add the new ones.
2. Reindex all platforms and sections. To reindex, run the following command:

```
C:\Repository> "C:\Program Files (x86)\VMware\VCM\Tools\Package
Studio\hive.exe" reindex "C:\Repository\dists\Any\Release\binary-platform"
platform section
```

Repository Entry Missing

A package cannot be found because its repository entry is missing.

Problem

An error appears, stating that the package CRATE file cannot be found.

Cause

A package was published to the repository using the `hive.exe` command or Package Studio, but its repository entry was not included in the `Repository.xml` file.

Solution

Add the repository entry to `Repository.xml` using Package Manager. Add the entry to the same platform and section to which the package was published.

Invalid Repository Index

Packages cannot be accessed because the repository index was corrupted.

Problem

Packages cannot be accessed for provisioning operations.

Cause

Someone manually edited the `Repository.index` file, and its XML code has become corrupted.

Solution

Review the `Repository.index` file XML syntax, and look for missing tags or other obvious problems. If the problem is not obvious, perform the following procedure:

1. Create a batch file that contains publish commands for all the packages (CRATE files), platforms, and sections in the `Repository.index` XML file.
2. Rename the failed `Repository.index` to another name.
3. Create a new `Repository.index` file, and add an empty `<RepositoryIndex/>` tag in it.
4. Run the batch file.

   The batch file rebuilds the index in the new `Repository.index` file and refreshes all the `crates.gz` files.

Corrupt Packages ZIP File

Packages cannot be accessed because the compressed file that holds the packages is corrupted.

Problem

Packages cannot be accessed for provisioning operations.

Cause

Someone manually edited the `crates.gz` file, and it has become corrupted.

Solution

Reindex all platforms and sections. To reindex, run the following command:

```
C:\Repository> "C:\Program Files (x86)\VMware\VCM\Tools\Package
Studio\hive.exe" reindex "C:\Repository\dists\Any\Release\binary-platform"
platform section
```

Too Many Platforms and Sections

The repository table of contents file contains too many platforms and sections.

Problem

Too many users are adding platforms and sections to the `repository.toc` file.

Cause

Too many users have write permission to the `repository.toc` file.

Solution

Disable write permission to the `repository.toc` file for the users that you do not want adding platforms and sections.

Repository Not Found

The repository cannot be found when reindexing or publishing.

Problem

When you attempt to reindex or publish packages to the repository, an error appears stating that the repository cannot be found.

Cause

The path to the repository is incomplete or incorrect.

Solution

When reindexing or publishing, type the fully qualified path to the root of the repository.

For example, the fully qualified path for publishing a package might be similar to the following path:

```
C:\Repository> "C:\Program Files (x86)\VMware\VCM\Tools\Package
Studio\hive.exe" publish "C:\Repository\internetexplorer_8.0_x86.crate"
crates\i platform section
```

Package Studio Installed Before Software Repository

For proper installation, Package Studio needs the software repository to already be installed.

Problem

An error appears, stating that Package Studio cannot create an instance of the repository editor view model:

```
Cannot create instance of RepositoryEditorViewModel defined in assembly
PackageStudio...
```

Cause

Package Studio was installed before the software repository was installed. The Package Studio installation process cannot populate the repository location because the default repository directory does not exist.

Solution

1. If it is not installed, install the repository.
2. Open the following Package Studio file in a text editor.

   The default location is `C:\Program Files (x86)\VMware\VCM\Tools\Package Studio\PackageStudio.exe.config`
3. Modify the `<RepositorySpecification>` entry to point to the repository path. For example:

   ```
   <Hive>
   <Repositories>
   <!--This is a list of the hive repositories on this server and their
   locations-->
   <RepositorySpecification name="default" localPath="C:\Program Files (x86)
   \VMware\VCM\Tools\Repository\">
   </Repositories>
   </Hive>
   ```

Generate Button Disabled

The option to generate a package is disabled.

Problem

When creating a package, the button to generate the package is dimmed.

Cause

Some data is required, and you did not yet provide valid data.

Solution

Add valid data to the required text boxes, located on the **Manage Packages** > **Properties** tab.

The required data is Name, Version, Architecture, and Description.

Invalid Dependency Package Name

Mismatched support for uppercase and lowercase causes an error.

Problem

A case discrepancy between the `Provides` name in Package A and the `Depends` name in Package B causes an error.

Cause

The `Provides` name includes uppercase letters. A current limitation allows uppercase in `Provides` but only lowercase in `Depends`.

For example, `Provides` in Package A is InternetExplorer, but `Depends` in Package B can only be internetexplorer. This discrepancy results in an error stating that the `Depends` name is invalid.

Solution

Use Package Studio to change the `Provides` name in Package A to lower case, and republish the package.

If you cannot change the `Provides` name without reversioning the package, manually edit the dependencies in Package B instead.

1. Locate the Package B CRATE file.
2. Rename its `.crate` extension to the `.zip` extension.
3. Unzip the files.
4. Open `control.xml` in a text editor.
5. Edit the lowercase `<Depends>` name entry to include uppercase so that it matches the `Provides` name from Package A. For example:

   ```
   <Depends Name="InternetExplorer" ...
   ```

6. Save `control.xml`.
7. Save and close the ZIP file.
8. Rename its `.zip` extension back to the `.crate` extension.

Signed Package Becomes Unsigned

A signed package is generated and becomes unsigned.

Problem

You create, sign, save, and generate a package, but the package is no longer signed.

Cause

You can sign a package only after it is generated.

Solution

Generate the package before signing it.

Space Runs Out During Installation

A software provisioning operation cannot finish because you run out of space on the target system.

Problem

The installed image size is *x*, and you have *x* free on the target system, but the software installation runs out of space.

Cause

The value *x* is the installed size after the operation completes, not the size required to unpack, download, and install the package contents.

Solution

Set the installed size to the amount of space required to unpack, download, and install, even if the final installed size is smaller.

Uninstallation Does Not Work

After successful installation, the application cannot be uninstalled.

Problem

Software provisioning installs the application, but the application cannot be uninstalled.

Cause

The Removal options were not configured in the package.

Solution

On the **Manage Packages** > **Files** tab, select **Removal** in the **Installation/Removal** drop-down menu, and configure any necessary Pre-Command, Commands, Arguments, and Post-Command options.

Incomplete List of Files

Files do not appear in the list for the project.

Problem

You added files to the project data directory, but the files do not appear in the list.

Cause

You must refresh the display.

Solution

Click the **Refresh Files List** button.

Requiring a Restart

A restart after installing software might or might not be needed or wanted.

Problem

Some packages might or might not require a restart after installation, or you might have external reasons for postponing or omitting a restart.

Cause

Some software installations can inspect a system and dynamically determine whether a restart is necessary. In addition, experienced administrators might want to omit a restart after installing software, for example to wait for a time when resource use is low.

Solution

Software provisioning does not support conditional restarts. When configuring a package, you must indicate whether to require a post-installation restart.

Whether to require a restart depends on what applications will be running at the time, the state of the machine, the operating system, and other factors. If you are in doubt but know for certain that you need a functioning application and system after installation or removal completes, require a restart.

Software Provisioning Logs on the Collector

To help VMware Technical Support troubleshoot a software provisioning problem, you might need to gather and send files that provide important troubleshooting clues for software provisioning issues.

The files are stored on the Collector when you enable ARS. See "Collect ARS Files" on page 23 for instructions.

Table 6–1. Files for Troubleshooting Software Provisioning Issues

File	Description
Provider.log	Provides information on what the provider did, how it interacted with Package Manager (wasp.exe), and any problems encountered when formatting the results. The AgentBridge also writes to this log, recording any problems in formulating provider instructions or transforming CDIF into element-normal XML.
CDIF files	Provides the raw results written by the various providers. CDIF files are useful in identifying special characters that are preventing the transformation to element-normal XML.
RequestInterop.xml	Provides a request document generated by the AgentBridge, which is used to startthe providers. An error in the provider log usually tells you which field you should evaluate.
Python output from Stdout	Provides Python output for collections and actions. Stdout includes all provider or AgentBridge collection results and wasp output in XML format.
Python output from Stderr	Provides Python output for collections and actions. Stderr includes any result that is determined to be an error condition.

Software Provisioning Logs on the Agent

To help VMware Technical Support troubleshoot a software provisioning problem, you might need to gather and send logging information collected from the Agent.

Prerequisites

- Enable Agent logging. See "Collect Agent Logging" on page 26.
- Enable Info message logging, which captures how Python is started. See "Set the Debug Log to Store all Message Types" on page 21

Procedure

1. Modify the following sample command to account for the folder names in your environment, and run it from the command prompt on the Agent machine.

   Because the command is long, you might want to put it in a batch file for editing, and run the batch file.

   ```
   C:\WINDOWS\CMAgent\Installer\Python\python.exe -E
   C:\WINDOWS\CMAgent\Installer\Providers\CommonPy\AgentBridge.py --
   root=C:\WINDOWS\CMAgent\Installer\Providers --action-
   template=C:\WINDOWS\CMAgent\Installer\Providers\Providers\Provisioning\
   Wasp\SourcesProvider\WaspSourcesProvider-RemoveRepositoryAction.template --action=template
   --provider=Providers\Provisioning\Wasp\SourcesProvider\WaspSourcesProvider.py --
   parameter=Uri="http://<RepositoryMachine>/SoftwareRepository" --parameter=Platform="Any" --
   parameter=Section="Release" --parameter=request_timeout_secs="28740"
   ```

2. To save all intermediate files and the provider log to a specified directory, add the following parameter.

   Specifying this directory saves all the intermediate files for actions, some of which are not available on the Collector.

   ```
   --temp-dir=c:\myProviderOutput
   ```

   The directory must be one that you create in advance.

3. Because of the temporary nature of the `LoginFile` in the following required parameter, modify the `AgentBridge` Python script to capture the file.

   ```
   --parameter=LoginFile=
   ```

   The `LoginFile` output normally exists only for the duration of the job.

   In *path*`\CMAgent\CommonPy\AgentBridge.py`, modify the argument handling for the `'parameter'` argument:

   ```
   elif o in ('--parameter='):
      param = str(a).strip("\"'")
      paramList = param.split('=', 1)
      key = paramList[0].strip("\"'")
      value = ''
      if len(paramList) > 1:
         value = paramList[1].strip("\"'")
      self.parameters[key] = value
      if key == 'LoginFile':
         from shutil import copy
         copy(value, 'C:\\myTemp\\')
   ```

7 VCM Windows Agent

To troubleshoot problems with your VCM Windows Agent, you must understand the Agent requirements to operate in your network, to communicate with your Collector. You must also understand how the Agent is installed so that you can trace the process and identify possible failure points.

The VCM Windows Agent is installed on managed Windows physical and virtual machines. The Agent is used to collect information from the machines. If the Agent is not functioning properly, VCM cannot collect data from the machine.

This chapter includes the following topics:

Windows Agent Installation Environment

To install the VCM Windows Agent, your target Windows machines must have the required components and services, and your Collector must be configured with the correct network credentials.

- "Windows Agent Installation Networking Requirements" on page 51

  The Windows machine on which you install the VCM Agent must have the required components and services to support the installation process and the Agent.

- "Windows Agent Installation Collector Credentials" on page 52

  The Collector must have domain accounts with the necessary domain authority so that you can install the VCM Windows Agent on your target physical or virtual machines.

Windows Agent Installation Networking Requirements

The Windows machine on which you install the VCM Agent must have the required components and services to support the installation process and the Agent.

- IPC$ share must exist.

  RPC uses IPC$.

- Windows Server Service must be running.

  The Collector uses the server services to resolve the target machine's share to a local path to register the bootstrap service, the Registration Service, with the Remote Service Control Manager (SCM).

- You must be able to attach to the share files using the credentials provided during installation.
  - DNS must correctly resolve the machine and share to the appropriate machine.
  - Collector service authority must correctly attach to the SCM.
- DCOM must be set to receive DCOM creation requests for various components. DCOM requires the following settings.
  - DCOM must be enabled on the target machine. On Windows 2003 Servers, this option is turned off by default.
  - On machines running the Windows NT operating systems, configure the default DCOM security authorities for Access, Launch, and Configuration.
  - Verify that the default authentication level is Connect and that the default Impersonation level is Identify.

Windows Agent Installation Collector Credentials

The Collector must have domain accounts with the necessary domain authority so that you can install the VCM Windows Agent on your target physical or virtual machines.

In VCM, define one or more domain administrator accounts in the Available Accounts data grid. These accounts are associated By Domain or By Machine Group with each target machine. You can configure multiple accounts that are specified in priority order.

When you install the Agent, the Collector cycles through these accounts until it locates the account needed to establish communication with each target machine.

Configure the accounts in VCM in the network authority available account settings.

Windows Agent Installation Process

When you install the VCM Windows Agent from the Collector using install Agent action rather than a manual action, the installation process comprises many steps, the success and failure of which are tracked in the Jobs Manager. If an Agent installation fails, you can evaluate the job as it is running or view the details in the history to determine which part of the process you must evaluate more closely.

Review the job history, identify the step that failed, and determine what might be impeding the installation of the Agent.

Detect Previous Install

The detect previous install action evaluates the target Windows machine to determine whether a previous version of the VCM Agent is installed.

When installing VCM 4.11.x Agent or later, if a previous Agent version is detected, VCM must remove it before installing the new Agent.

The detect previous install action determines if a previous Agent is present by attempting to connect to the Agent installation DCOM components, the Basic and Agent Installers. If a connection is made to either of these components, the detect previous install action sets a state member that the Validate Installation Environment, Interrogate Target Environment, and Resolve Uninstall Dependencies actions use to determine if they should run.

These actions remove the previous Agents and are included in all types of install and uninstall requests.

Validate Installation Environment

The validate installation environment action ensures that the target Windows machine is available for the installation of the VCM Agent.

1. If the validate job can contact the Module Installer, it calls methods on that component to obtain all the installation manifests on the target. It also checks the registry for conditions that might prevent the installation.

   This job fails if the following conditions are satisfied:

   - The Agent is locked. The `HKEY_LOCAL_ MACHINE\SOFTWARE\Configuresoft\ECM\4.0\Agent\IsLocked`, for 32-bit Windows servers, or `HKEY_LOCAL_ MACHINE\SOFTWARE\Wow6432Node\Configuresoft\ECM\4.0\Agent\IsLocked`, for 64-bit Windows servers, the registry entry is evaluated by this job to determine if the Agent is locked. If this key exists and has a value other than 0, then the Agent is locked.

     The installation infrastructure cannot remove or modify this value.

   - The Agent is a Collector. The `HKEY_LOCAL_ MACHINE\SOFTWARE\Configuresoft\ECM\4.0\Installer\CDInstallor` or `HKEY_LOCAL_ MACHINE\SOFTWARE\Wow6432Node\Configuresoft\ECM\4.0\Installer\CDInstall` registry entry is evaluated by this job to determine if the Agent is on a Collector machine. If this key exists and has a value of 2, then the target Windows machine is identified as a Collector machine.

     The installation infrastructure cannot remove or modify this value.

2. If the validate job cannot communicate with the installation infrastructure, it flags the installation as valid and the request continues.

3. If an error is generated by an attempt to communicate with the installation infrastructure, the validation fails, as does the installation process, usually with an error that mandates that an uninstall action must occur before installing. An error is any result other than Install required or Success.

4. If the validation job can communicate with the Module installer, it gathers all the module information available from the various installation manifests and stores this information as a state variable so that other installation jobs can use the information to determine what was installed. The validation job does this in the `RecordModuleInstallerVersion` method and stores the module names and versions in the `install_module` state parameter . The module installer job uses this parameter to determine which parts of the installation infrastructure must to be installed.

Interrogate Target Environment

The interrogate target environment action runs on the target Windows machine after the installation infrastructure is deployed to install the various product modules during the VCM Agent installation process.

The following actions determine which product modules must be deployed to the target machine.

1. A connection is made to the module installer.
2. The runtime Agent lock is updated.
3. The module installer returns all modules currently installed.
4. The product modules are recorded in the database.

A failure in this action causes the installation to fail.

Resolve Uninstall Dependencies

The resolve uninstall dependencies action reconciles the product modules that were found on the target Windows machine by the interrogation action and the product modules that must be deployed for the VCM Agent.

The result of the action is a list of product modules that might need to be uninstalled and a list of product modules that must be installed. Any product modules that require deployment are recorded in the state matrix.

A failure in this action causes the installation to fail.

Uninstall Module

The uninstall module action removes all product modules that must be uninstalled from the target Windows machine before the process installs the VCM Agent.

1. A list of product modules is obtained from the database.
2. A connection is made to the module installer.
3. The runtime Agent lock is updated.
4. Each module is uninstalled in turn by the module installer.

A failure in this action causes the installation to fail.

Uninstall Module Installer

The uninstall module installer action completes the removal of a previous Windows Agent as preparation for installing the new VCM Agent.

1. If you are uninstalling the Agent, this action tries all of the following methods to completely remove the Agent.
   - Asynchronous command using the Registration Service.
   - Command to simple installer
   - Remote share.
2. If the current request is for any other kind of installation action, the action does nothing.

Install Simple Installer

The install simple installer action uses a service to deploy the installation infrastructure modules to the target Windows machines so that the infrastructure can manage the installation of the VCM Agent.

The installation infrastructure modules included in this action are `ECMCommon.exe`, `ECMSimpleInstaller.exe`, and `ECMComSocketListener.exe`.

1. An attempt is made to contact the simple installer on the target Windows machine. If this action is successful, then this step is complete. If the action is not successful, the process continues.
2. An attempt is made to attach to the share on the target that was specified in the user interface and using the specified authority.

   If this action fails, then the installation fails.
3. If the attach to share action succeeds, the process determines if the Agent has a runtime lock. This lock, which is not the same as the IsLocked registry entry, is used by the installation process to prevent two or more Collectors from installing on the same target machine to the same location at the same time.

   The lock manifests as a `ECMMachineActionLock.dat` file on the target machine. This lock file is located in the root directory of the installation file and contains the type of action being performed. The types of actions include installing, uninstalling, or collecting. The lock file includes the time stamp and the request ID of the action doing the work. If the lock file exists on the target machine, and it contains a valid action with a request ID and a time stamp that has not expired, the target machine is considered locked and the installation fails.

   When the target machine is considered locked, the lock is partially unlocked. To partially unlock the lock, the time stamp is zeroed and the request ID is purged. These actions allow a subsequent installation to continue if it was not relocked by another Collector. In all other cases, the lock file is zeroed and the installation process is allowed to continue.
4. A preinstallation check ensures that the subsequent installation will succeed. If components, for example, `ComSocketServiceListener`, the debug event `.dll`, or the subsystem singleton `.dll` are located, they are removed.
5. The directory structure is created and the following components are copied to the target machine.
   - `ECMColInstallAgtRegistrationService.exe`

     This binary file is the bootstrap service that is invoked by the target machine's Remote Service Control Manager (SCM).
   - `Psapi.dll`

     This file is a dependency of the Registration Service.
   - `ECMTargetShareInfo.dat`

     This file is generated and contains the share that is used for this installation as well as the relative path to the Agent.
6. An attempt is made to connect to the SCM using the same authority as the Collector service.

   If this action fails, the installation fails.
7. The `ECMColInstallAgtRegistrationService.exe` is registered with the SCM on the target machine.
8. The following loop runs until all the installation infrastructure modules are deployed.
   - Copy an installation infrastructure module to the target.
   - Start the registration service.

     This service runs the module and waits until it is fully expanded.
   - If the module reports an error, terminate the installation.

     A log file is left on the target machine that specifies the nature of the problem.
   - Stop the service.

9. The registration service is unregistered with VCM.
10. An attempt is made to contact the simple install using DCOM.

    If this action succeeds, this step in the process is finished.

If the process fails on any of the actions, the simple installer job enters the rollback state. All modules are uninstalled, all files are removed, and the registry is purged. In this state, the debug event files are not deleted on the target machine. The file might contain useful information about the error.

Install Module Installer

The module installer action manages the installation of the product modules when you install the VCM Agent on target Windows machines.

The installer is called on to terminate the Windows Agent installation process when an inspection job is canceled. It can update the installation infrastructure. The module installer completely removes the remainder of the VCM files during the uninstallation process.

1. Determine if the module installer component was deployed. If it exists, a check is preformed to see if it needs to be upgraded.
2. If the module installer needs to be installed, the following actions are run.
   - A connection is made to the simple installer.
   - The runtime install lock is updated.
   - The module installer is deployed and run by the simple installer.
3. Determine if any installation infrastructure modules must be upgraded.

   Outdated components are listed and upgraded.
4. If a component of the installation infrastructure, other than the module installer, must be updated, the following actions run.
   - A connection is made to the module installer.
   - The registration service is copied if it is not present on the target Windows machine.
   - All infrastructure modules that must be updated are copied to the target machine.
   - A call is made to the module installer to install the installation infrastructure modules. This call is received and delegated to the registration service to perform the action.
   - A connection is made to the simple installer.
   - The runtime install lock is updated.
   - The module installer is pushed out and run by the simple installer.
5. Any installation infrastructure modules that must be removed are removed.

If an action fails at any point, the changes are rolled back and the installation fails.

Resolve All Versions of Modules Based on Highest Version Number

The resolve all versions of modules action evaluates the modules to be installed against those already on the target Windows machine and determines what must be installed for the VCM Agent.

The action uses the resolve highest version modules resolution algorithm.

This algorithm is provided with a requested install module list (RIML), the already installed module list (AIML), the module dependency graph, and the module updates map. The algorithm uses the map to determine what must be installed and uninstalled on the target machine to achieve the latest versions of modules based on the modules provided in the RIML and the AIML.

Install Module

The install module action installs all product modules related to the VCM Agent.

1. A list of product modules is obtained from the database.
2. A connection is made to the module installer.
3. The runtime Agent lock is updated.
4. Each module is installed by the module installer.

A failure in this action causes the installation to fail.

Fully Release the Synchronization Lock on the Target Machine

The fully release the synchronization lock action clears the runtime Agent lock. Clearing the Agent lock allows the VCM Agent to inspect the target Windows machine.

1. A connection is made to the module installer.
2. Unlock is called.

If this action is not run because of a previous failure, the installation is considered invalid. Attempts to collect from the Agent results in failure because the installation process failed.

Submit Request to Agent

The submit request to Agent action is the first VCM Agent inspection on the Windows machine.

After installing the Agent, a machine environment inspection is performed.

Check If Request Is Complete

The check if request is complete action verifies that the Collector successfully collected from the VCM Agent on the Windows machine.

The Collector checks to determine if the Windows machine environment collection from the Agent is finished.

Transfer Request Results

The transfer request results action sends the results of the inspection on the Windows machine to the Collector as part of the VCM Agent installation process.

After the Agent completes the machine environment inspection on the Windows machine, the results are sent back to the Collector.

Acknowledge Successful Data Transfer

The acknowledge successful data transfer action is a record of the data from the VCM Agent inspection on the Windows machine during the installation of the Agent.

The Collector records the receipt of the data from the Windows machine environment inspection.

Prepare Request Results for Insert

The prepare request results for insert action prepares the data from the Windows machine inspection for insertion in the VCM database during the VCM Agent installation process.

A bulk insert of the machine environment inspection data is prepared for insertion in the VCM database.

Insert Data Into Database

The insert data into database action adds the data collected from the Windows machine by the VCM Agent to the database as a step in the Agent installation process.

The data from the machine environment inspection is inserted in the VCM database.

Transform Inserted Data

The transform inserted data action takes data collected from the Windows machine by the VCM Agent during the Agent installation process and transforms it from the temp tables to the proper database format for the VCM database.

The inserted machine environment data is transformed in the database from temp tables and the meta data is updated

Cleanup Machine Data

The cleanup machine data action cleans up the files on the target Windows machine after installing the VCM Agent.

The data on the target Windows machine is cleaned up.

Partially Release the Synchronization Lock on the Target Machine

The partial release of the synchronization lock on the target Windows machine action is a finalization job that runs even if any of the previous steps fail during the VCM Agent installation process.

This action clears out the Agent runtime lock so that if another installation of the Agent is attempted, it will succeed.

This step does not clear the action type in the lock. Because this condition is a result of an invalid installation, this action is a partial release that renders the Agent incapable of doing useful work.

The Agent is available for collections only if the fully release the synchronization lock on the Agent action runs successfully.

Cleanup Request Data

The cleanup request data action removes unneeded and transform data on the Collector as the final action in the VCM Agent installation process.

The VCM Collector cleans up the inserted and transformed data, the request is removed, and the state of the request is complete.

Windows Agent Uninstallation Process

The Windows Agent uninstallation process removes the VCM Agent from Windows machines. This action removes VCM products and the installation infrastructure. The result of uninstall is a clean Windows machine with no VCM remnants.

To troubleshoot this process, review the job history, identify the step that failed, and determine what might be impeding the removal of the Agent.

Detect Previous Install

The detect previous install action evaluates the Windows machine from which you are removing the VCM Agent to determine whether a version of the Agent is installed that must be uninstalled.

The detect previous install action determines if a previous Agent is present by attempting to connect to the Agent installation DCOM components, the Basic and Agent Installers. If a connection is made to either of these components, the detect previous install action sets a state member that the Validate Installation Environment, Interrogate Target Environment, and Resolve Uninstall Dependencies actions use to determine if they should run.

These actions remove the previous Agents and are included in all types of install and uninstall requests.

Validate Installation Environment

The validate installation environment action ensures that the target Windows machine is available so that the Collector can uninstall the VCM Agent.

1. If the validate job can contact the Module Installer, it calls methods on that component to obtain all the installation manifests on the target. It also checks the registry for conditions that might prevent the installation.

   This job fails if the following conditions are satisfied:

   - The Agent is locked. The `HKEY_LOCAL_MACHINE\SOFTWARE\Configuresoft\ECM\4.0\Agent\IsLocked`, for 32-bit Windows servers, or `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Configuresoft\ECM\4.0\Agent\IsLocked`, for 64-bit Windows servers, the registry entry is evaluated by this job to determine if the Agent is locked. If this key exists and has a value other than 0, then the Agent is locked.

     The installation infrastructure cannot remove or modify this value.

   - The Agent is a Collector. The `HKEY_LOCAL_MACHINE\SOFTWARE\Configuresoft\ECM\4.0\Installer\CDInstallor` or `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Configuresoft\ECM\4.0\Installer\CDInstall` registry entry is evaluated by this job to determine if the Agent is on a Collector machine. If this key exists and has a value of 2, then the target Windows machine is identified as a Collector machine.

     The installation infrastructure cannot remove or modify this value.

2. If the validate job cannot communicate with the installation infrastructure, it flags the installation as valid and the request continues.

3. If an error is generated by an attempt to communicate with the installation infrastructure, the validation fails, as does the installation process, usually with an error that mandates that an uninstall action must occur before installing. An error is any result other than Install required or Success.

4. If the validation job can communicate with the Module installer, it gathers all the module information available from the various installation manifests and stores this information as a state variable so that other installation jobs can use the information to determine what was installed. The validation job does this in the `RecordModuleInstallerVersion` method and stores the module names and versions in the `install_module` state parameter . The module installer job uses this parameter to determine which parts of the installation infrastructure must to be installed.

Interrogate Target Environment

The interrogate target environment action runs on the Windows machine after the installation infrastructure is deployed to uninstall the various product modules during the VCM Agent uninstallation process.

The following actions determine which product modules must be deployed to the target machine.

1. A connection is made to the module installer.
2. The runtime Agent lock is updated.
3. The module installer returns all modules currently installed.
4. The product modules are recorded in the database.

This action in the uninstallation process runs even if one or more of the actions fail.

Resolve Uninstall Dependencies

The resolve uninstall dependencies action uninstalls all the product modules on the target box. Product modules are reconciled and flagged for removal and recorded in the database.

The result of the action is a list of product modules that might need to be uninstalled and a list of product modules that must be installed. Any product modules that require deployment are recorded in the state matrix.

This action in the uninstallation process runs even if one or more of the actions fail.

Uninstall Module

The uninstall module action removes all product modules that must be uninstalled from the Windows machine as part of the VCM uninstallation process.

1. A list of product modules is obtained from the database.
2. A connection is made to the module installer.
3. The runtime Agent lock is updated.
4. Each module is uninstalled in turn by the module installer.

This action in the uninstallation process runs even if one or more of the actions fail.

Uninstall Module Installer

The uninstall module installer action removes the VCM Agent during the uninstallation process.

1. The product and installation infrastructure modules are removed based on the following workflow.
   - The Collector connects to the module installer.
   - A complete uninstallation request is generated and sent to the module installer.
   - The module installer copies the registration service to the user's temp directory.
   - The module installer calls on the registration service to perform the complete uninstall and the module installer terminates itself.
   - The registration service, now running as an EXE iterates over the Agent’s directories and notes all

the installed modules.

- The registration service then uninstalls each module it finds.
- The registration service ensures that all VCM files are removed from the file system and that the registry is purged.
- The installer marks itself for deletion on the next reboot of the Windows machine.

2. The uninstall module installer attempts to attach to share on the Window machine. This attempt is made even if the previous step succeeds. The attachment process during uninstall might fail. If it can attach, it removes all the VCM files that it finds bases on a filter stored in the database.
3. The uninstall module installer attempts to connect to the remote registry. If it can, it removes all VMware registry entries that it finds.

This action in the uninstallation process runs even if one or more of the actions fail.

Fully Release the Synchronization Lock on the Target Machine

The fully release the synchronization lock action, which releases the runtime lock, runs only if the uninstall fails and the module installer is reached.

If uninstall worked correctly, a connection cannot be made with the module installer. No further action is necessary to fully release synchronization lock.

1. A connection is made to the module installer.
2. Unlock is called.

This action in the uninstallation process runs even if one or more of the actions fail.

Partially Release the Synchronization Lock on the Target Machine

The partially release the synchronization lock action, which is run only if the uninstallation of the VCM Agent failed, clears the runtime action lock time stamp and the request ID.

This action in the uninstallation process runs even if one or more of the actions fail.

Cleanup Request Data

The cleanup request data action removed unneeded and transformed data on the Collector and the final action in the VCM Agent uninstallation process.

The VCM Collector cleans up the inserted and transformed data, the request is removed, and the state of the request is complete.

This action in the uninstallation process runs even if one or more of the actions fail.

Windows Agent Upgrade Process

The process to upgrade the VCM Agent on Windows machines upgrades various modules, either installation infrastructure or product modules. The upgrade request is identical to an Agent installation request except that an Agent must be present on the target Windows machine.

For process details, see "Windows Agent Installation Process" on page 52.

Windows Agent Manual Installation Process

The manual Agent installation process deploys the Agent to the target without using the Collector. The main goal of manual installation is to produce an Agent environment that is identical to an Agent environment that one would expect if a Collector deployed the Agent. Manual installation does not create entries in the add and remove programs section, nor is the manually installed Agent designed to be uninstalled manually. When manually installing the Agent, you can use the Agent only option on the VCM Product installation CD or the use of the actual Wise binary.

To troubleshoot the manual installation process, review the job history, identify the step that failed, and determine what might be impeding the installation of the Agent. See "Manual Window Agent Installation " on page 62.

⚠ **CAUTION** Do not use a manual installation to uninstall the Agent. Doing so can cause errors.

No Manual Upgrade

You cannot use the manual installation process to upgrade the Windows Agent.

Manual Window Agent Installation

The manual Agent installation process is a group of modules deployed using the Wise installation system rather than pushing the Agent files from the Collector.

The VCM installation media is an Install Shield program. If you select the Agent Only installation, then Install Shield delegates the work of installing the Agent to the `CMAgentInstall.exe` file. This executable file, built by Wise Installation Systems 9 is the manual installation program. This is the program that installs the Agent. You can run this executable file interactively or silently.

The manual installation runs the same modules that the Collector runs to install an Agent. The following modules are run by `CMAgentInstall.exe`.

- `ECMNotUpdateable.exe`
- `ECMSimpleInstaller.exe`
- `ECMModuleInstaller.exe`
- `ECMCommon.exe`
- Optionally, `CsiWin32SocketListener.exe`

You can use the interactive mode or the silent mode to perform a manual installation.

- Interactive Mode

  Configure the options during the installation process.

Option	Description
Installation directory	Specify the directory to which the Agent is installed.
Lock agent	To lock the Agent, select the check box.
HTTP	To install the Agent so that is uses HTTP communication protocols, select the check box and provide the port on which the Agent receives requests from the Collector. If you specify an HTTP port already in use by the Windows machine, the installation process prompts you with another port selection. This process continues until an open port is found.

- Silent Mode

  The silent mode runs the `CMAgentInstall.exe` file without requiring responses during the installation process. The silent mode allows you to deploy the Agent using scripts or using a command line. For example, a common way to deploy an Agent is to create a script that runs on the target Windows machine and runs the manual Agent install program silently. To install the manual install program using scripts or the command line, the following options are available.

Option	Description
/s	Silent mode switch.
DCOM Protocol	the command line is `CMAgentInstall.exe /S INSTALLPATH=C:\MyVCM`. Where INSTALLPATH is the destination of the Agent files.
HTTP Protocol	the command line is `CMAgentInstall.exe /S INSTALLPATH=C:\MyVCMPORT=26542`. Where PORT is the port that the Agent is to use for communication. If the port is in use when installing silently, the manual install fails. The path for `CMAgentInstall.exe` may be a UNC path.

Windows Agent Communication Protocols

The communication protocols determine how the VCM Agent communicates with the Collector.

The `CSIWin32SocketListener.exe` file is an installation infrastructure module that determines the protocol availability on the Agent. This module contains the HTTP service that listens for Collector requests on the specified port. The service delegates the requests to the components that are part of the standard installation. The Collector determines the protocol that is used to communicate with the Agent.

- HTTP. An Agent configured with the HTTP communication protocol is accessed using a port or using DCOM.

  The HTTP protocol was added to VCM as an alternative to DCOM in all cases except you install the Agent. HTTP requires the addition of the `CsiWin32SocketListener.exe` module on the Agent.

- DCOM. An Agent configured with the DCOM communication protocol responds only to DCOM requests.

  The Agent is installed using DCOM. Uninstalling and upgrading the Agent are not bound by this limitation.

DCOM is also the lowest common protocol used for installing the Agent and for collecting data. Uninstalling and upgrading the Agent are not bound by this limitation.

If the Collector lists an Agent as listening with HTTP when communicating with the Agent, and the HTTP connection cannot be established, DCOM communication is attempted.

Communication Protocol Change Process

A change protocol request is applied when you change the protocol the Collector uses to communicate with an Agent on a Windows machine. The supported communications protocols are HTTP and DCOM.

DCOM is the oldest supported protocol and dates back to the VCM 3.0x.

HTTP protocol was introduced with VCM4.5. A change protocol request makes the following changes.

- When changing the protocol from DCOM to HTTP, the `EcmComSocketListenerService` module is deployed and run.
- When changing from HTTP to DCOM, the `EcmComSocketListenerService` is removed.

To troubleshoot problems related to communication protocols, review the job history, identify the step that failed, and determine what might be impeding communication protocol change process.

Detect Previous Install

The detect previous install action evaluates the target Windows machine to determine whether a previous version of the VCM Agent is installed.

When you install the VCM 4.11.x Agent or later, if a previous Agent version is detected, it must be removed.

The detect previous install action determines if a previous Agent is present by attempting to connect to the Agent installation DCOM components, the Basic and Agent Installers. If a connection is made to either of these components, the detect previous install action sets a state member that the Validate Installation Environment, Interrogate Target Environment, and Resolve Uninstall Dependencies actions use to determine if they should run.

These actions remove the previous Agents and are included in all types of install and uninstall requests.

Uninstall Agent

The uninstall Agent action removes all product modules that must be uninstalled from the target Windows machine before installing the new communication protocol modules.

1. A list of product modules is obtained from the database.
2. A connection is made to the module installer.
3. The runtime Agent lock is updated.
4. Each module is uninstalled in turn by the module installer.

A failure in this action causes the installation to fail.

Uninstall Package Installer

The uninstall package installer action completes the removal of the previous communication protocol modules in preparation for the new modules.

Uninstall Basic Installer

The uninstall basic installer action removes the basic Agent installation component in preparation for the new communication protocol.

Validate Installation Environment

The validate installation environment action ensures that the target Windows machine is available for the installation of the new VCM Agent communication protocol modules.

1. If the validate job can contact the Module Installer, it calls methods on that component to obtain all the installation manifests on the target. It also checks the registry for conditions that might prevent the installation.

   This job fails if the following conditions are satisfied:

   - The Agent is locked. The `HKEY_LOCAL_MACHINE\SOFTWARE\Configuresoft\ECM\4.0\Agent\IsLocked`, for 32-bit Windows servers, or `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Configuresoft\ECM\4.0\Agent\IsLocked`, for 64-bit Windows servers, the registry entry is evaluated by this job to determine if the Agent is locked. If this key exists and has a value other than 0, then the Agent is locked.

     The installation infrastructure cannot remove or modify this value.

   - The Agent is a Collector. The `HKEY_LOCAL_MACHINE\SOFTWARE\Configuresoft\ECM\4.0\Installer\CDInstallor` or `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Configuresoft\ECM\4.0\Installer\CDInstall` registry entry is evaluated by this job to determine if the Agent is on a Collector machine. If this key exists and has a value of 2, then the target Windows machine is identified as a Collector machine.

     The installation infrastructure cannot remove or modify this value.

2. If the validate job cannot communicate with the installation infrastructure, it flags the installation as valid and the request continues.

3. If an error is generated by an attempt to communicate with the installation infrastructure, the validation fails, as does the installation process, usually with an error that mandates that an uninstall action must occur before installing. An error is any result other than Install required or Success.

4. If the validation job can communicate with the Module installer, it gathers all the module information available from the various installation manifests and stores this information as a state variable so that other installation jobs can use the information to determine what was installed. The validation job does this in the `RecordModuleInstallerVersion` method and stores the module names and versions in the `install_module` state parameter . The module installer job uses this parameter to determine which parts of the installation infrastructure must to be installed.

Install Simple Installer

The install simple installer action uses a service to deploy the installation infrastructure modules to the target Windows machines so that the infrastructure can manage the installation of the VCM Agent communication protocol modules.

During a change protocol request, this job receives special parameters that describe how to handle the Com Socket Service Listener module.

If the change protocol request is from DCOM to HTTP, the simple installer component receives a request to install the `ComSocketServiceListener`.

It the change protocol request was from HTTP to DCOM, the simple installer receives a request to remove the `ComSocketServiceListener`.

The simple installer does not modify the contents of the installation infrastructure. The simple installer job is only used to initially deploy the infrastructure using the Registration service as a bootstrap loader. Beacuae the simple installer job cannot take action on the information it receives, it delegates the work to the Install Module Installer step.

Store Installation Data in the Database

The store installation data in the database process updates the list of modules in the database.

Install Module Installer

The install module installer action reconciles the information regarding what infrastructure modules are installed, whether to install or remove the `ComSocketServiceListener`, and determine the actions to run.

If you are switching from DCOM to HTTP and the `ComSocketServiceListener` should be installed, the install module installer creates a request and sends the work of installing this module to the Registration Service.

If you are switching from HTTP to DCOM and the `ComSocketServiceListener` should be removed, the module installer creates a request and sends the work of uninstalling this module to the Registration Service.

Fully Release the Synchronization Lock On the Target Machine

The fully release the synchronization lock action clears the runtime Agent lock. Clearing the Agent lock allows the VCM Agent to inspect the target Windows machine.

1. A connection is made to the module installer.
2. Unlock is called.

Submit Request to Agent

The submit request to Agent action is the first VCM Agent inspection on the Windows machine.

After installing the Agent, a machine environment inspection is performed.

Check If Request Is Complete

The check if request is complete action verifies that the Collector successfully collected from the VCM Agent on the Windows machine.

The Collector checks to determine if the Windows machine environment collection from the Agent is complete.

Transfer Request Results

The transfer request results action sends the results of the inspection on the Windows machine to the Collector as part of the change communication protocol process.

After the Agent completes the machine environment inspection on the Windows machine, the results are sent back to the Collector.

Acknowledge Successful Data Transfer

The acknowledge successful data transfer action is a record of the data from the VCM Agent inspection on the Windows machine during the change communication protocol process.

The Collector records the receipt of the data from the Windows machine environment inspection.

Prepare Request Results For Insert

The prepare request results for insert action prepares the data from the Windows machine inspection for insertion into the VCM database during the change communication protocol process.

A bulk insert of the machine environment inspection data is prepared for insertion in the VCM database.

Insert Data Into Database

The insert data into database action adds the data collected from the Windows machine by the VCM Agent to the database as a step in the change communication protocol process.

The data from the machine environment inspection is inserted in the VCM database.

Transform Inserted Data

The transform inserted data action takes data collected from the Windows machine by the VCM Agent during the change communication protocol process and transforms it from the temp tables to the proper database format for the VCM database.

The inserted machine environment data is transformed in the database from temp tables and the meta data is updated

Cleanup Machine Data

The cleanup machine data action cleans up the files on the target Windows machine after changing the communication protocol.

The data on the target Windows machine is cleaned up.

Partially Release the Synchronization Lock on the Target Machine

The partial release of the synchronization lock on the target Windows machine action is a finalization job that runs even if any of the previous steps fail during the change communication protocol process.

The goal of this action is to clear out the Agent runtime lock so that if another installation of the Agent is attempted, it succeeds.

This step does not clear the action type in the lock. Because this condition is a result of an invalid installation, this action is a partial release that renders the Agent incapable of doing useful work.

The Agent is available for collections only if the Fully release the synchronization lock on the Agent action runs successfully.

Cleanup Request Data

The cleanup request data action removes unneeded and transformed data on the Collector and the final action in the change communication protocol process.

The VCM Collector cleans up the inserted and transformed data, the request is removed, and the state of the request is complete.

Debug Window Agent Installations

Use the detailed information in the debug event view when debugging problems related to module resolution or to gain insight on the impact that installing or uninstalling have on a particular Agent.

The debug event viewer provides module management debugging information as each module resolution algorithm is run by the Collector during each Agent process. Each resolution algorithm dumps the current and requested Agent modules, module dependency graph, the module updates mapping before running the algorithm, and each algorithm's step-by-step output as it is runs.

The debug event viewer is available on the Collector.

Procedure

1. On the Collector, open the tools folder.

   The default location is `C:\Program Files (x86)\VMware\VCM\Tools`.

2. Right-click the `ECMDebugEventViewer.exe` file and select **Run as Administrator**.
3. In the event viewer, select **File > Fetch**.
4. Locate the time during which the installation failed, or rerun the Agent installation so that you can capture the events during the installation process.

VCM UNIX Agent

8

To troubleshoot problems with the VCM UNIX Agent, you must understand how the Agent is installed, how it operates in your network, and how it communicates with the VCM Collector. Knowing these processes helps you trace the flow of settings and data so that you can identify possible failure points.

The UNIX Agent is installed on managed UNIX systems that include variants such as those from Red Hat, Solaris, AIX, and others. The Agent is the means by which information is gathered from the machines, and if the Agent is not functioning properly, VCM cannot collect the data that you need for managing your UNIX environment.

This chapter includes the following topics:

UNIX Agent Directory Structure After Installation

The `/opt` directory is the default installation location for the UNIX Agent.

```
/opt/:
dr-xr-x--x   9 root      cfgsoft      320 Oct 31 16:47 CMAgent
```

if you installed using the defaults, you can type the command **`ls -laR`** to return a full listing of all the directories and files under the UNIX Agent directory structure.

You can use the list to verify that no important files are missing. Understanding the contents of the directories helps clarify the relationship between files and functions when you troubleshoot problems.

/opt/CMAgent

The `CMAgent` directory is the root of the Agent installation. The `CSIRegistry` file is an XML file representing configuration data for the Agent. Conceptually, it performs the same function as the Windows Registry. Other than the `data` directory, all files are owned by `root` and the `cfgsoft` group.

```
/opt/CMAgent:
drwxr-x---    3 root     cfgsoft      4096 Oct 31 15:01 Agent
drwxr-x---    3 root     cfgsoft      4096 Oct 31 15:01 CFC
-rw-rw----    1 root     cfgsoft     51039 Oct 31 15:01 CSIRegistry
-rw-rw----    1 root     cfgsoft     51039 Oct 31 15:01 CSIRegistry.bkup
```

```
drwxrwx---    3 csi_acct cfgsoft      4096 Oct 31 15:01 data
drwxrwx---    3 root     cfgsoft      4096 Oct 31 15:01 ECMu
drwxr-x---    6 root     cfgsoft      4096 Oct 31 15:01 install
drwxr-x---    3 root     cfgsoft      4096 Oct 31 15:01 Installer
lrwxrwxrwx    1 root     cfgsoft        20 Oct 31 15:01 log ->
/var/log/CMAgent/log
dr-xr-x--x    3 root     cfgsoft      4096 Oct 31 15:01 ThirdParty
drwxr-xr-x    2 root     root         4096 Oct 31 15:01 uninstall
```

/opt/CMAgent/Agent

The `Agent` directory contains code libraries that are specific to the Agent. The `Agent` directory contains an `x.x` directory for the code version, which in turn contains a `bin` directory that contains all executable files and a `lib` directory that contains all libraries.

```
/opt/CMAgent/Agent:
drwxr-x---    4 root     cfgsoft      4096 Oct 31 15:01 3.0
-rw-r-----    1 root     cfgsoft      3201 Oct 31 15:01 manifest_Agent.3.0.Linux

/opt/CMAgent/Agent/3.0:
drwxr-x---    2 root     cfgsoft      4096 Oct 31 15:01 bin
drwxr-x---    2 root     cfgsoft      4096 Oct 31 15:01 lib

/opt/CMAgent/Agent/3.0/bin:
[ currently empty ]

/opt/CMAgent/Agent/3.0/lib:
-r-xr-x---    1 root     cfgsoft    301984 Oct 31 15:01
libAgentFileManagerSubsystem.so
-r-xr-x---    1 root     cfgsoft     76696 Oct 31 15:01 libAgentResponse.so
-r-xr-x---    1 root     cfgsoft    105560 Oct 31 15:01 libChangeCommon.so
-r-xr-x---    1 root     cfgsoft    102024 Oct 31 15:01
libChangeFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft     28192 Oct 31 15:01 libCommonQueues.so
-r-xr-x---    1 root     cfgsoft    113796 Oct 31 15:01 libCommonStates.so
-r-xr-x---    1 root     cfgsoft     82056 Oct 31 15:01
libConfigurationFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft     65088 Oct 31 15:01
libCsiAgentCompatibilitySerialization.so
-r-xr-x---    1 root     cfgsoft     41304 Oct 31 15:01
libCsiAgentFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft    110164 Oct 31 15:01
libCsiAgentFileTransferHandlerSubsystem.so
-r-xr-x---    1 root     cfgsoft    972288 Oct 31 15:01 libCsiAgentSubsystem.so
-r-xr-x---    1 root     cfgsoft    274868 Oct 31 15:01 libCsiAgentUNIXIPC.so
-r-xr-x---    1 root     cfgsoft    212960 Oct 31 15:01
libCsiAgentUNIXProxySubsystem.so
-r-xr-x---    1 root     cfgsoft    114808 Oct 31 15:01 libDataStorageCommon.so
-r-xr-x---    1 root     cfgsoft    208428 Oct 31 15:01 libDeltaEngine.so
-r-xr-x---    1 root     cfgsoft     85192 Oct 31 15:01
libDeltaFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft    139712 Oct 31 15:01 libDeltaSerialization.so
-r-xr-x---    1 root     cfgsoft    850712 Oct 31 15:01 libDtmClient.so
-r-xr-x---    1 root     cfgsoft    102992 Oct 31 15:01 libDtmCommon.so
-r-xr-x---    1 root     cfgsoft     77416 Oct 31 15:01 libDtmFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft     85148 Oct 31 15:01 libDtmManager.so
-r-xr-x---    1 root     cfgsoft    479664 Oct 31 15:01 libDtmUpdateManager.so
-r-xr-x---    1 root     cfgsoft     94880 Oct 31 15:01 libEcmCoreGlobals.so
-r-xr-x---    1 root     cfgsoft    114572 Oct 31 15:01 libFileManagerCommon.so
-r-xr-x---    1 root     cfgsoft    139140 Oct 31 15:01
libFileManagerSubsystem.so
```

```
-r-xr-x---    1 root     cfgsoft    770416 Oct 31 15:01 libFilterCommon.so
-r-xr-x---    1 root     cfgsoft     94784 Oct 31 15:01 libFilterDocs.so
-r-xr-x---    1 root     cfgsoft    278088 Oct 31 15:01 libFilterDriver.so
-r-xr-x---    1 root     cfgsoft     36392 Oct 31 15:01
libFilterFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft    221500 Oct 31 15:01 libHdsCommon.so
-r-xr-x---    1 root     cfgsoft     68688 Oct 31 15:01 libHdsParsers.so
-r-xr-x---    1 root     cfgsoft    221624 Oct 31 15:01 libHdsStreams.so
-r-xr-x---    1 root     cfgsoft    185264 Oct 31 15:01
libInstallInteropSubsystem.so
-r-xr-x---    1 root     cfgsoft     34724 Oct 31 15:01 libRdmDataStorageBase.so
-r-xr-x---    1 root     cfgsoft    388728 Oct 31 15:01
libRdmDataStorageCommon.so
-r-xr-x---    1 root     cfgsoft     72616 Oct 31 15:01
libRdmDataStoreManagerSubsystem.so
-r-xr-x---    1 root     cfgsoft    367956 Oct 31 15:01
libRdmDtmProviderSubsystem.so
-r-xr-x---    1 root     cfgsoft    140272 Oct 31 15:01
libRdmLockManagerSubsystem.so
-r-xr-x---    1 root     cfgsoft    342516 Oct 31 15:01
libRdmStatusManagerProviderSubsystem.so
-r-xr-x---    1 root     cfgsoft     94772 Oct 31 15:01 libRegistryConfig.so
-r-xr-x---    1 root     cfgsoft     65064 Oct 31 15:01
libRemoteCommandObjectCommon.so
-r-xr-x---    1 root     cfgsoft    114352 Oct 31 15:01
libReplicationFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft    225308 Oct 31 15:01
libReplicationSerialization.so
-r-xr-x---    1 root     cfgsoft    261464 Oct 31 15:01
libRequestFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft    332628 Oct 31 15:01 libRequestObjectCommon.so
-r-xr-x---    1 root     cfgsoft    114864 Oct 31 15:01 libRequestSubsystem.so
-r-xr-x---    1 root     cfgsoft     69224 Oct 31 15:01
libResponseFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft    347788 Oct 31 15:01
libScriptEngineSubsystem.so
-r-xr-x---    1 root     cfgsoft    126260 Oct 31 15:01 libSimpleDtmManager.so
-r-xr-x---    1 root     cfgsoft    155720 Oct 31 15:01 libSimpleXmlDtmManager.so
-r-xr-x---    1 root     cfgsoft    287320 Oct 31 15:01
libStateMachineDefinitions.so
-r-xr-x---    1 root     cfgsoft    295944 Oct 31 15:01 libStateMachineEngine.so
-r-xr-x---    1 root     cfgsoft    250800 Oct 31 15:01
libStateMachineFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft    965736 Oct 31 15:01 libStatusManager.so
```

/opt/CMAgent/CFC

The `CFC` directory contains code libraries that are common components. It contains an `x.x` directory for the code version, which in turn contains a `bin` directory that contains all executable files and a `lib` directory that contains all libraries.

```
/opt/CMAgent/CFC:
drwxr-x---    4 root     cfgsoft      4096 Oct 31 15:01 3.0
-rw-r-----    1 root     cfgsoft      2770 Oct 31 15:01 manifest_CFC.3.0.Linux

/opt/CMAgent/CFC/3.0:
drwxr-x---    2 root     cfgsoft      4096 Oct 31 15:01 bin
drwxr-x---    2 root     cfgsoft      4096 Oct 31 15:01 lib

/opt/CMAgent/CFC/3.0/bin:
-r-xr-x---    1 root     cfgsoft     87449 Oct 31 15:01 CsiAgentListener
```

```
-r-xr-x---    1 root     cfgsoft    208547 Oct 31 15:01 CsiAgtStartupCli
-r-xr-x---    1 root     cfgsoft    313560 Oct 31 15:01 CsiListenerWorkerDaemon
-r-xr-x---    1 root     cfgsoft    109214 Oct 31 15:01 CSI_
ManageCertificateStore
-r-xr-x---    1 root     cfgsoft     15324 Oct 31 15:01 RegisterSubSystem
-r-xr-x---    1 root     cfgsoft     27080 Oct 31 15:01 RegistryAdd
-r-xr-x---    1 root     cfgsoft     17404 Oct 31 15:01 RegistryRead
-r-xr-x---    1 root     cfgsoft     15176 Oct 31 15:01 UnregisterSubSystem
-r-xr-x---    1 root     cfgsoft    261548 Oct 31 15:01 vcmcrypt

/opt/CMAgent/CFC/3.0/lib:
-r-xr-x---    1 root     cfgsoft   1881248 Oct 31 15:01 libCfcCommonAggregator.so
-r-xr-x---    1 root     cfgsoft    400160 Oct 31 15:01 libCfcDataSerializable.so
-r-xr-x---    1 root     cfgsoft    179776 Oct 31 15:01
libCfcDebugEventSubSystemSingleton.so
-r-xr-x---    1 root     cfgsoft    633924 Oct 31 15:01 libCfcEncoding.so
-r-xr-x---    1 root     cfgsoft    219932 Oct 31 15:01 libCfcFileCompression.so
-r-xr-x---    1 root     cfgsoft     53784 Oct 31 15:01 libCfcGlobals.so
-r-xr-x---    1 root     cfgsoft     35528 Oct 31 15:01 libCfcSecCommon.so
-r-xr-x---    1 root     cfgsoft    479356 Oct 31 15:01
libCfcStreamCompression.so
-r-xr-x---    1 root     cfgsoft   2421024 Oct 31 15:01 libChannelCommon.so
-r-xr-x---    1 root     cfgsoft    443000 Oct 31 15:01 libCommunicationCommon.so
-r-xr-x---    1 root     cfgsoft    102320 Oct 31 15:01
libCommunicationFactorySubSystem.so
-r-xr-x---    1 root     cfgsoft    295880 Oct 31 15:01
libComponentInteropSubsystem.so
-r-xr-x---    1 root     cfgsoft    248724 Oct 31 15:01
libCryptographicUtilities.so
-r-xr-x---    1 root     cfgsoft    286980 Oct 31 15:01
libCsiHttpProtocolHandlerCommon.so
-r-xr-x---    1 root     cfgsoft    123292 Oct 31 15:01
libCsiHttpTlsProtocolHandlerSubsystem.so
-r-xr-x---    1 root     cfgsoft    372088 Oct 31 15:01
libCsiSocketListenerSubsystem.so
-r-xr-x---    1 root     cfgsoft     63536 Oct 31 15:01
libCsiSocketListenerUtils.so
-r-xr-x---    1 root     cfgsoft    283288 Oct 31 15:01 libCssBase.so
-r-xr-x---    1 root     cfgsoft    269880 Oct 31 15:01 libCssCryptoCommon.so
-r-xr-x---    1 root     cfgsoft    156644 Oct 31 15:01
libCssDataProtectionServiceSubsystem.so
-r-xr-x---    1 root     cfgsoft    213096 Oct 31 15:01
libCssOpensslCertificateStoreSubsystem.so
-r-xr-x---    1 root     cfgsoft   1594700 Oct 31 15:01 libCssOpensslCommon.so
-r-xr-x---    1 root     cfgsoft     89424 Oct 31 15:01
libCssOpensslCryptoProviderSubsystem.so
-r-xr-x---    1 root     cfgsoft    250320 Oct 31 15:01
libCssOpensslProtectedStorageSubsystem.so
-r-xr-x---    1 root     cfgsoft     47680 Oct 31 15:01
libCssServicesManagerSubsystemSingleton.so
-r-xr-x---    1 root     cfgsoft    249988 Oct 31 15:01 libCssTlsCommon.so
-r-xr-x---    1 root     cfgsoft    383200 Oct 31 15:01
libCssTlsOpensslSubsystem.so
-r-xr-x---    1 root     cfgsoft    197568 Oct 31 15:01
libDataSerializableFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft    271448 Oct 31 15:01 libFile.so
-r-xr-x---    1 root     cfgsoft    238248 Oct 31 15:01
libFileTransferInteropSubsystem.so
-r-xr-x---    1 root     cfgsoft    441896 Oct 31 15:01 libFormattersSubsystem.so
-r-xr-x---    1 root     cfgsoft     98748 Oct 31 15:01 libMemoryBuffers.so
-r-xr-x---    1 root     cfgsoft    216688 Oct 31 15:01 libSerializationCommon.so
-r-xr-x---    1 root     cfgsoft     90412 Oct 31 15:01
```

```
libSubsystemObjectCacheSubsystem.so
-r-xr-x---    1 root     cfgsoft     43488 Oct 31 15:01
libSubSystemSingletonManagerSubSystem.so
-r-xr-x---    1 root     cfgsoft     71996 Oct 31 15:01 libSynchronization.so
-r-xr-x---    1 root     cfgsoft    249692 Oct 31 15:01 libThreadPool.so
-r-xr-x---    1 root     cfgsoft    433704 Oct 31 15:01 libUNIXIPCCore.so
-r-xr-x---    1 root     cfgsoft    435896 Oct 31 15:01 libXMLParser.so
```

/opt/CMAgent/data

You can configure the location of the `data` directory when you install the Agent. The `data` directory contains all of the inspection results, master files, and so on, and is used for any temporary files that are created during the inspection process.

```
/opt/CMAgent/data:
drwxrwx---   4 csi_acct     cfgsoft  96 Oct 31 14:11 <collector-name>
drwxrwx---   5 root         cfgsoft 152 Oct 31 12:28 db
drwxrwx---   4 csi_acct     cfgsoft 152 Oct 31 11:53 tmp
```

/opt/CMAgent/data/db

The `db` directory contains directories for the data model and status manager Birdstep databases. It also contains a directory for the Agent certificate store.

```
/opt/CMAgent/data/db:
drwxrwx---    3 root     cfgsoft      4096 Oct 31 15:01 DtmDB
drwxrwx---    3 root     cfgsoft      4096 Oct 31 15:01 PDS
drwxrwx---    3 root     cfgsoft      4096 Oct 31 15:01 SM

/opt/CMAgent/data/db/DtmDB:
drwxrwx---    2 root     cfgsoft      4096 Oct 31 15:01 RDM
```

/opt/CMAgent/data/db/DtmDB/RDM

The `DtmDB/RDM` directory is the Birdstep database that contains the default data model. This data model forms the basis for replicated data models from the Agent.

```
/opt/CMAgent/data/db/DtmDB/RDM:
-rw-rwx---    1 root     cfgsoft      1865 Oct 31 15:01 DtmDB.dbd
-rw-rwx---    1 root     cfgsoft      2048 Oct 31 15:01 DtmKeys.dbd
-rw-rwx---    1 root     cfgsoft      9216 Oct 31 15:01 DtmRecord.dbd
```

/opt/CMAgent/data/db/PDS

The `PDS` directory contains the certificate store. For the Agent to validate a Collector, the files in this directory must be readable by the `cfgsoft` group.

```
/opt/CMAgent/data/db/PDS:
-rw-rw----    1 root     cfgsoft      2895 Oct 31 15:01 CertStore
drwxrwx---    2 root     cfgsoft      4096 Oct 31 15:01 sib
```

If the Agent package was copied from a Collector, the certificate for the Collector is preloaded to the certificate store. You can use the `CSI_ManageCertificateStore` executable file to add other certificates.

NOTE On some UNIX variants, replace `LD_LIBRARY_PATH` with the following environment variable name:
AIX: `LIBPATH`
HP-UX: `SHLIB_PATH`
Mac OS: `DYLD_LIBRARY_PATH`

```
export LD_LIBRARY_
PATH=/opt/CMAgent/CFC/3.0/lib:/opt/CMAgent/ThirdParty/1.0/lib

export CSI_REGISTRY_PATH=/opt/CMAgent

/opt/CMAgent/CFC/3.0/bin/CSI_ManagerCertificateStore -iz -fcert-file
```

/opt/CMAgent/data/db/SM/RDM

The SM/RDM directory contains the Birdstep database, which holds information about running requests, the average time it takes a request to be run, and so on.

```
/opt/CMAgent/data/db/SM/RDM:
-rw-rwx---    1 root     cfgsoft      1536 Oct 31 15:01 MachineStatusLog.dbd
-rw-rwx---    1 root     cfgsoft      2048 Oct 31 15:01 MachineStatusLog_K1.dbd
-rw-rwx---    1 root     cfgsoft      3072 Oct 31 15:01 RequestStatusLog.dbd
-rw-rwx---    1 root     cfgsoft      2048 Oct 31 15:01 RequestStatusLog_K1.dbd
-rw-rwx---    1 root     cfgsoft      1536 Oct 31 15:01
StateMachineStateHistoryLog.dbd
-rw-rwx---    1 root     cfgsoft      2048 Oct 31 15:01
StateMachineStateHistoryLog_K1.dbd
-rw-rwx---    1 root     cfgsoft      2560 Oct 31 15:01 StateMachineStatusLog.dbd
-rw-rwx---    1 root     cfgsoft      2048 Oct 31 15:01 StateMachineStatusLog_
K1.dbd
-rw-rwx---    1 root     cfgsoft      2361 Oct 31 15:01 StatusManagerDB.dbd
```

/opt/CMAgent/ECMu

The ECMu directory contains code libraries specific to the UNIX Agent. The ECMu directory contains a directory x.x for the code version, which in turn contains directories for the libraries and registration scripts for ECMu.

```
/opt/CMAgent/ECMu:
drwxrwx---    6 root     cfgsoft      4096 Oct 31 15:01 1.0
-rw-r-----    1 root     cfgsoft      1391 Oct 31 15:01 manifest_ECMu.1.0.Linux
-rw-r--r--    1 root     root            4 Oct 31 15:01 version

/opt/CMAgent/ECMu/1.0:
drwxr-x---    2 root     cfgsoft      4096 Oct 31 15:01 bin
drwxr-x---    2 root     cfgsoft      4096 Oct 31 15:01 lib
drwxr-x---    2 root     cfgsoft      4096 Oct 31 15:01 registration
drwxr-x---    2 root     cfgsoft      4096 Oct 31 15:01 scripts

/opt/CMAgent/ECMu/1.0/lib:
-r-xr-x---    1 root     cfgsoft     13216 Oct 31 15:01 libAgentXPCommon.so
-r-xr-x---    1 root     cfgsoft    315140 Oct 31 15:01
libAwkScriptDriverSubsystem.so
-r-xr-x---    1 root     cfgsoft    249236 Oct 31 15:01
libAwkScriptDriverSyslogEventsSubsystem.so
-r-xr-x---    1 root     cfgsoft    545168 Oct 31 15:01 libCsiPpxLibHelper.so
-r-xr-x---    1 root     cfgsoft    361100 Oct 31 15:01
libEcmAgentInspectorCommon.so
-r-xr-x---    1 root     cfgsoft     81704 Oct 31 15:01
libEcmAgentInspectorScript.so
-r-xr-x---    1 root     cfgsoft    274360 Oct 31 15:01 libEcmFileUploadJob.so
-r-xr-x---    1 root     cfgsoft    230176 Oct 31 15:01 libEcmRemoteCommandJob.so
-r-xr-x---    1 root     cfgsoft    204968 Oct 31 15:01
libEcmScriptInspectionJob.so
-r-xr-x---    1 root     cfgsoft     64392 Oct 31 15:01
libPatchFactorySubsystem.so
-r-xr-x---    1 root     cfgsoft    147424 Oct 31 15:01
```

```
libScriptChangeStateMachineJob.so
-r-xr-x---    1 root     cfgsoft    171456 Oct 31 15:01 libXpChangeDriverState.so

/opt/CMAgent/ECMu/1.0/registration:
-rw-r-----    1 root     cfgsoft       622 Oct 31 15:01 cmagent.deb
-rw-r-----    1 root     cfgsoft      2117 Oct 31 15:01 CMAgent.rpm
-r-xr-x---    1 root     cfgsoft      2299 Oct 31 15:01 RegisterAgent.sh
-r-xr-x---    1 root     cfgsoft      1044 Oct 31 15:01 UnregisterAgent.sh
```

/opt/CMAgent/ECMu/x.x/bin

There are three files in the bin directory that are used when you run inspections and remote commands.

- **RunHigh.** Runs privileged inspections, which is possible because it is owned by root and has the suid permission set as seen with the r-s in the permissions.
- **RunLow.** Runs unprivileged inspections, which is possible because it is owned by the primary group (a nobody group) of the user that the Agent runs as and has the sgid permission set.

  When this program runs, it switches to the nobody group and cannot run commands that require root privilege.
- **RunRemote.** Runs privileged remote commands and operates in the same manner as RunHigh.

```
/opt/CMAgent/ECMu/1.0/bin:
-r-xr-x---    1 root     cfgsoft     68697 Oct 31 15:01 Agent
-r-xr-x---    1 root     cfgsoft     53341 Oct 31 15:01 cabextract
-r-xr-x---    1 root     cfgsoft    111995 Oct 31 15:01 csipccli
-r-sr-x---    1 root     cfgsoft     11192 Oct 31 15:01 RunHigh
-r-xr-s---    1 csi_acct csi_acct    11253 Oct 31 15:01 RunLow
-r-sr-x---    1 root     cfgsoft      9686 Oct 31 15:01 RunRemote
-r-xr-x---    1 root     cfgsoft    124059 Oct 31 15:01 TestMetadata
```

If these executable files fail, they log errors as described in"Run Executable Logging" on page 87.

/opt/CMAgent/ECMu/x.x/scripts

The scripts directory contains scripts that run with the Agent. The csi-agent file is a copy of the details installed to the (x)inetd configuration. The inetd-agent is the script that (x)inetd runs when an attempt is made to contact the Agent.

The stopagent.sh script stops all of the Agent processes in a clean manner.

```
/opt/CMAgent/ECMu/1.0/scripts:
-r-xr-x---    1 root     cfgsoft       351 Oct 31 15:01 boot-init.sh
-r--r-----    1 root     cfgsoft      1279 Oct 31 15:01 boot-init.sh.lsb
-r--r-----    1 root     cfgsoft      1439 Oct 31 15:01 boot-init.sh.RH
-r--r-----    1 root     cfgsoft      3375 Oct 31 15:01 boot-init.sh.SuSE
-r--r-----    1 root     cfgsoft        75 Oct 31 15:01 csi-agent
-r--r-----    1 root     cfgsoft       249 Oct 31 15:01 csi-agent-xinetd
-r-xr-x---    1 root     cfgsoft       265 Oct 31 15:01 inetd-agent
-r-xr-x---    1 root     cfgsoft      4163 Oct 31 15:01 KillAgent.sh
-r-xr-x---    1 root     cfgsoft      1130 Oct 31 15:01 killprocs.sh
-r-xr-x---    1 root     cfgsoft       231 Oct 31 15:01 SrfHapErrorTemplate.xml
-r-xr-x---    1 root     cfgsoft       743 Oct 31 15:01 stopagent.sh
```

/opt/CMAgent/install

The `install` directory contains the infrastructure used to install and uninstall the Agent. The `install` directory also contains log files that might help determine why an installation failed. The `BootStrapInstall.log` file contains a log of all of the actions that the installer took. The `DebugEvent_cis.dbe` is an error log file that you can copy to a Collector to view in the Debug Event Viewer.

```
/opt/CMAgent/install:
-rw-r-----    1 root     cfgsoft     37356 Oct 31 15:01 BootStrapInstall.log
-r-xr-xr-x    1 root     root        39624 Oct 31 15:01 BootStrapInstall.sh
-rw-r-----    1 root     cfgsoft       243 Oct 23 14:17 checksum
drwxr-x---    2 root     cfgsoft      4096 Oct 31 15:01 cis
-r--r-----    1 root     cfgsoft       192 Oct 11 10:32 CMAgentPkgReadme.txt
-r--r--r--    1 root     root        11095 Oct 31 15:01 csi.config
-rw-r-----    1 root     cfgsoft     25926 Oct 31 15:01 DebugEvent_cis.dbe
-rw-r-----    1 root     cfgsoft  11688722 Oct 31 15:01 install.log
-rw-r-----    1 root     cfgsoft       513 Oct 31 15:00 KillAgent.log
-rwxr-x---    1 root     cfgsoft        70 Oct 23 14:17 package.sizes.Linux
drwxr-x---    3 root     cfgsoft      4096 Oct 31 15:01 python
-rw-rw----    1 root     cfgsoft      1830 Feb 27  2012 reinstall.log
dr--------    2 root     root         4096 Oct 31 15:01 saved
-rw-r-----    1 root     cfgsoft     14250 Oct 31 15:01 status
drwxr-x---    2 root     cfgsoft      4096 Oct 31 15:01 uninstall

/opt/CMAgent/install/cis:
-r--r-----    1 root     cfgsoft      6935 Oct 23 14:17 CInstallPackage.py
-rw-r-----    1 root     cfgsoft      3321 Oct 31 15:01 CInstallPackage.pyc
-rw-r-----    1 root     cfgsoft       919 Oct 23 14:17 cis.1.0.Linux
-r--r-----    1 root     cfgsoft      1285 Oct 23 14:17 CisAgent.py
-r--r-----    1 root     cfgsoft      6330 Oct 23 14:17 CisCommon.py
-rw-r-----    1 root     cfgsoft      7136 Oct 31 15:01 CisCommon.pyc
-r--r-----    1 root     cfgsoft      2360 Oct 23 14:17 CisEnvironment.py
-rw-r-----    1 root     cfgsoft      2083 Oct 31 15:01 CisEnvironment.pyc
-r--r-----    1 root     cfgsoft      3351 Oct 23 14:17 CisFilesystemBasic.py
-rw-r-----    1 root     cfgsoft      3053 Oct 31 15:01 CisFilesystemBasic.pyc
-r--r-----    1 root     cfgsoft      7771 Oct 23 14:17 CisFilesystem.py
-rw-r-----    1 root     cfgsoft      8201 Oct 31 15:01 CisFilesystem.pyc
-r--r-----    1 root     cfgsoft      7003 Oct 23 14:17 CisInstall.py
-rw-r-----    1 root     cfgsoft      5641 Oct 31 15:01 CisInstall.pyc
-r--r-----    1 root     cfgsoft       953 Oct 23 14:17 cis.py
-rw-r-----    1 root     cfgsoft       284 Oct 31 15:01 cis.pyc
-r--r-----    1 root     cfgsoft      8772 Oct 23 14:17 CisRegistry.py
-rw-r-----    1 root     cfgsoft      8813 Oct 31 15:01 CisRegistry.pyc
-r--r-----    1 root     cfgsoft     10274 Oct 23 14:17 CisRollback.py
-rw-r-----    1 root     cfgsoft      8088 Oct 31 15:01 CisRollback.pyc
-r-xr-x---    1 root     cfgsoft    882696 Oct 23 14:17 _cis.so
-r--r-----    1 root     cfgsoft      3432 Oct 23 14:17 CisValues.py
-rw-r-----    1 root     cfgsoft      1844 Oct 31 15:01 CisValues.pyc
-r--r-----    1 root     cfgsoft      2547 Oct 23 14:17 ConfigCommon.py
-rw-r-----    1 root     cfgsoft      1789 Oct 31 15:01 ConfigCommon.pyc
-r--r-----    1 root     cfgsoft      2788 Oct 23 14:17 CUninstallProduct.py
-rw-r-----    1 root     cfgsoft      1268 Oct 31 15:01 CUninstallProduct.pyc
-r--r-----    1 root     cfgsoft      3538 Oct 23 14:17 InstallCleanup.py
-r--r-----    1 root     cfgsoft     12944 Oct 23 14:17 InstallMain.py
-r--r-----    1 root     cfgsoft      6997 Oct 23 14:17 InstallPackages.py
-r--r-----    1 root     cfgsoft      2858 Oct 23 14:17 Install.py
-r--r-----    1 root     cfgsoft      4777 Oct 23 14:17 Process.py
-rw-r-----    1 root     cfgsoft      3213 Oct 31 15:01 Process.pyc
-r--r-----    1 root     cfgsoft     34203 Oct 23 14:17 Service.py
-rw-r-----    1 root     cfgsoft     27037 Oct 31 15:01 Service.pyc
-r-xr-x---    1 root     cfgsoft      1433 Oct 23 14:17 TestGroup.sh
```

```
-r-xr-x---    1 root     cfgsoft     1084 Oct 23 14:17 TestUserId.sh
-r-xr-x---    1 root     cfgsoft     1031 Oct 23 14:17 TestUser.sh
-r--r-----    1 root     cfgsoft    10534 Oct 23 14:17 UninstallProducts.py
-r--r-----    1 root     cfgsoft    48925 Oct 23 14:17 UserGroup.py
-rw-r-----    1 root     cfgsoft    40548 Oct 31 15:01 UserGroup.pyc

/opt/CMAgent/install/python:
-r--r-----    1 root     cfgsoft    22409 Oct 23 14:17 codecs.py
-rw-r-----    1 root     cfgsoft    27697 Oct 31 15:01 codecs.pyc
-r--r-----    1 root     cfgsoft     6433 Oct 23 14:17 copy_reg.py
-rw-r-----    1 root     cfgsoft     6161 Oct 31 15:01 copy_reg.pyc
drwxr-x---    2 root     cfgsoft     4096 Oct 31 15:01 encodings
-r--r-----    1 root     cfgsoft     9173 Oct 23 14:17 grp.so
-r-xr-x---    1 root     cfgsoft    41383 Oct 23 14:17 libgcc_s.so.1
-r-xr-x---    1 root     cfgsoft   867468 Oct 23 14:17 libstdc++.so.6
-r--r-----    1 root     cfgsoft     2803 Oct 23 14:17 linecache.py
-rw-r-----    1 root     cfgsoft     3167 Oct 31 15:01 linecache.pyc
-r--r-----    1 root     cfgsoft    21675 Oct 23 14:17 os.py
-rw-r-----    1 root     cfgsoft    26424 Oct 31 15:01 os.pyc
-r--r-----    1 root     cfgsoft    13111 Oct 23 14:17 posixpath.py
-rw-r-----    1 root     cfgsoft    13950 Oct 31 15:01 posixpath.pyc
-r--r-----    1 root     cfgsoft     9584 Oct 23 14:17 pwd.so
-r-xr-----    1 root     cfgsoft   791546 Oct 23 14:17 python
-rw-r-----    1 root     cfgsoft      767 Oct 23 14:17 python.23.Linux
-r--r-----    1 root     cfgsoft     4981 Oct 23 14:17 shutil.py
-rw-r-----    1 root     cfgsoft     7029 Oct 31 15:01 shutil.pyc
-r--r-----    1 root     cfgsoft    12757 Oct 23 14:17 site.py
-rw-r-----    1 root     cfgsoft    11570 Oct 31 15:01 site.pyc
-r--r-----    1 root     cfgsoft     1753 Oct 23 14:17 stat.py
-rw-r-----    1 root     cfgsoft     3355 Oct 31 15:01 stat.pyc
-r--r-----    1 root     cfgsoft    11749 Oct 23 14:17 string.py
-rw-r-----    1 root     cfgsoft    13916 Oct 31 15:01 string.pyc
-r--r-----    1 root     cfgsoft    19002 Oct 23 14:17 time.so
-r--r-----    1 root     cfgsoft    10374 Oct 23 14:17 traceback.py
-rw-r-----    1 root     cfgsoft    13147 Oct 31 15:01 traceback.pyc
-r--r-----    1 root     cfgsoft     2244 Oct 23 14:17 types.py
-rw-r-----    1 root     cfgsoft     3154 Oct 31 15:01 types.pyc
-r--r-----    1 root     cfgsoft     5610 Oct 23 14:17 UserDict.py
-rw-r-----    1 root     cfgsoft    11066 Oct 31 15:01 UserDict.pyc
-r--r-----    1 root     cfgsoft     9092 Oct 23 14:17 warnings.py
-rw-r-----    1 root     cfgsoft    10055 Oct 31 15:01 warnings.pyc

/opt/CMAgent/install/python/encodings:
-r--r-----    1 root     cfgsoft     4768 Oct 23 14:17 __init__.py
-rw-r-----    1 root     cfgsoft     4288 Oct 31 15:01 __init__.pyc
-r--r-----    1 root     cfgsoft      639 Oct 23 14:17 utf_8.py
-rw-r-----    1 root     cfgsoft     1360 Oct 31 15:01 utf_8.pyc

/opt/CMAgent/install/saved:
-rw-r--r--    1 root     root        1009 Apr  5  2011 group
-rw-r--r--    1 root     root        3036 Dec  6  2011 passwd
-rw-r--r--    1 root     root       19936 Oct 31 15:00 services

/opt/CMAgent/install/uninstall:
-rw-r--r--    1 root     root       11588 Oct 31 15:01 Agent.py
-rw-r--r--    1 root     root       11018 Oct 31 15:01 CFC.py
-rw-r--r--    1 root     root       12555 Oct 31 15:01 ECMu.py
-rw-r--r--    1 root     root        6385 Oct 31 15:01 ThirdParty.py
-rw-r--r--    1 root     root          20 Oct 31 15:01 timestamp.py
```

/opt/CMAgent/Installer

The `Installer` directory contains Agent components that are dynamically available based on the VCM actions being performed. Directories and files under the `Content` directory vary.

```
/opt/CMAgent/Installer:
drwxrwx---    3 root     cfgsoft      4096 Oct 31 15:01 Content
```

/opt/CMAgent/ThirdParty

The `ThirdParty` directory contains code libraries that are common components. It contains an `x.x` directory for the code version, which in turn contains directories with all of the binary files and libraries.

The `gawk` executable file is world executable. This configuration allows nonprivileged inspectors to use it. On a Solaris Agent, this directory also contains the `libiconv.so.2.1.0` library, which is also world readable because it is used by the `gawk` executable.

```
/opt/CMAgent/ThirdParty:
dr-xr-x--x    5 root     cfgsoft      4096 Oct 31 15:01 1.0
-rw-r-----    1 root     cfgsoft      1870 Oct 31 15:01 manifest_
ThirdParty.1.0.Linux

/opt/CMAgent/ThirdParty/1.0:
dr-xr-x--x    2 root     cfgsoft      4096 Oct 31 15:01 bin
dr-xr-x--x    2 root     cfgsoft      4096 Oct 31 15:01 lib
drwxr-xr-x    5 root     root         4096 Oct 31 15:01 PatchAssessment

/opt/CMAgent/ThirdParty/1.0/bin:
-r-xr-x--x    1 root     cfgsoft    308956 Oct 31 15:01 gawk
-r-xr-x---    1 root     cfgsoft     30736 Oct 31 15:01 lm
-r-xr-x---    1 root     cfgsoft      9352 Oct 31 15:01 lmmgr
-r-xr-x---    1 root     cfgsoft    104992 Oct 31 15:01 unzip
-r-xr-x---    1 root     cfgsoft      4924 Oct 31 15:01 VMwareFingerPrint
-r-xr-x---    1 root     cfgsoft     61640 Oct 31 15:01 zip

/opt/CMAgent/ThirdParty/1.0/lib:
lrwxrwxrwx    1 root     cfgsoft        47 Oct 31 15:01 libACE.so ->
/opt/CMAgent/ThirdParty/1.0/lib/libACE.so.5.3.0
-r-xr-x---    1 root     cfgsoft   1610400 Oct 31 15:01 libACE.so.5.3.0
-r-xr-x---    1 root     cfgsoft    417448 Oct 31 15:01 libboost_regex.so
lrwxrwxrwx    1 root     cfgsoft        45 Oct 31 15:01 libgcc_s.so ->
/opt/CMAgent/ThirdParty/1.0/lib/libgcc_s.so.1
-r-xr-xr-x    1 root     cfgsoft     33740 Oct 31 15:01 libgcc_s.so.1
-r-xr-x---    1 root     cfgsoft    208152 Oct 31 15:01 librdmm3.so
-r-xr-x---    1 root     cfgsoft     80036 Oct 31 15:01 librdmmpsp3.so
lrwxrwxrwx    1 root     cfgsoft        50 Oct 31 15:01 libstdc++.so ->
/opt/CMAgent/ThirdParty/1.0/lib/libstdc++.so.6.0.0
lrwxrwxrwx    1 root     cfgsoft        50 Oct 31 15:01 libstdc++.so.6 ->
/opt/CMAgent/ThirdParty/1.0/lib/libstdc++.so.6.0.0
-r-xr-x---    1 root     cfgsoft    867468 Oct 31 15:01 libstdc++.so.6.0.0
```

/opt/CMAgent/ThirdParty/x.x/PatchAssessment

The `PatchAssessment` directory contains dynamic patch assessment components and is only present on Linux systems. Directories and files under `patchagent`, `share`, and `templates` vary.

```
/opt/CMAgent/ThirdParty/1.0/PatchAssessment:
drwxr-xr-x    4 root     root         4096 Oct 31 15:01 patchagent
drwxr-xr-x    3 root     root         4096 Oct 31 15:01 share
drwxr-xr-x    2 root     root         4096 Oct 31 15:01 templates
```

/opt/CMAgent/uninstall

The `uninstall` directory contains the script to remove the Agent.

```
/opt/CMAgent/uninstall:
-rwxr-xr--    1 root     root        54135 Oct 31 15:01 UninstallCMAgent
```

Directories Created During an Inspection

When a Collector first contacts the Agent, it copies its data model to the Agent. The Agent stores the data model as a Birdstep database in a Collector-named directory immediately under `DtmDB/RDM`. If you delete the directory, the next collection fails, but the subsequent collection recopies the data model.

```
opt/CMAgent/data/db/DtmDB/RDM/collector-name:
-rw-rw----    1 nobody   cfgsoft     35508 May 20 10:05 327B23C6-0AE4-428E-0001-
7F00F14D0500_data.dat
-rw-rw----    1 nobody   cfgsoft     19220 May 20 10:05 327B23C6-0AE4-428E-0001-
7F00F14D0500_keys.dat
-rw-rw----    1 nobody   cfgsoft     36358 May 20 10:02 643C9869-064D-428E-0001-
7F0063CE0800_data.dat
-rw-rw----    1 nobody   cfgsoft     18765 May 20 10:02 643C9869-064D-428E-0001-
7F0063CE0800_keys.dat
-rw-rw----    1 nobody   cfgsoft     27124 May 20 10:05 6B8B4567-0AE2-428E-0001-
7F00A5190200_data.dat
-rw-rw----    1 nobody   cfgsoft     15808 May 20 10:05 6B8B4567-0AE2-428E-0001-
7F00A5190200_keys.dat
-rw-rw----    1 nobody   cfgsoft      1865 May 20 10:02 DtmDB.dbd
-rw-rw----    1 nobody   cfgsoft       498 May 20 14:11 DtmDB.taf
-rw-rw----    1 nobody   cfgsoft   1325056 May 20 10:05 DtmKeys.dbd
-rw-rw----    1 nobody   cfgsoft   3078144 May 20 10:05 DtmRecord.dbd
```

When an inspection occurs, a Collector-named directory appears under `/opt/CMAgent/data`.

```
/opt/CMAgent/data/collector-name:
drwxrwx---    2 nobody   cfgsoft      4096 May 20 14:11 Master
drwxrwx---    2 nobody   cfgsoft      4096 May 20 14:12 Package
```

The `Master` directory contains the inspected data and is used when performing deltas. One MFL file is collected per data class. If you delete the files, the next collection is a full collection instead of a delta.

```
/opt/CMAgent/data/collector-name/Master:
-rw-rw----    1 nobody   cfgsoft      8516 May 20 14:11 UnixAccountGroup.mfl
-rw-rw----    1 nobody   cfgsoft     13892 May 20 14:11 UnixAccountUser.mfl
-rw-rw----    1 nobody   cfgsoft   2869360 May 20 14:11 UnixFileSystem.mfl
```

The `Package` directory temporarily contains the results that are sent back to the Collector. You can expand the ZRP file with the **`/opt/CMAgent/ThirdParty/1.0/bin/unzip *zrp`** command.

```
/opt/CMAgent/data/collector-name/Package:
-rw-rw----    1 root     root        10210 May 20 14:11 215D4C5A-AF55-40EB-BADD-
B634B18EF734.zrp
```

The Agent deletes the ZRP file after receiving acknowledgement that the Collector received it. You can capture the file before it is deleted though. See "Capture the ZRP on the Agent" on page 85.

The ZRP file might contain a debug event (DBE) file, which is returned to the Collector and inserted in the SQL database so that it can be viewed at the Collector. If the file is missing on the Collector, capture the ZRP file on the Agent, extract the DBE, and manually copy it to the Collector for viewing in the Debug Event Viewer.

Directory of Executed Scripts and Results

If the `SaveTempScriptFiles` entry in the `/opt/CMAgent/CSIRegistry` file is set to `true`, copies of executed `gawk` scripts, remote command scripts, and output are stored in the `ScriptFiles` directory.

```
/opt/CMAgent/data/ScriptFiles:
total 8
drwxrwx---    2 nobody   cfgsoft      4096 May 20 14:08 .
drwxrwx---    6 root     cfgsoft      4096 May 20 14:07
```

The file names are as follows, where *xxxxxx* is a random alphanumeric string:

- script_*xxxxxx*: `gawk` or remote command script
- hds_*xxxxxx*: Output of a `gawk` script
- rcmd_*xxxxxx*: Output of a remote command script

Collector Certificates

For a Collector to communicate with the Agent, you must upload the Collector certificate PEM file to the following UNIX Agent directory, and make it readable by the `cfgsoft` group.

```
/opt/CMAgent/data/db/PDS/CertStore
```

On the Collector, the certificate file is stored as follows, by default.

```
\Program Files (x86)\VMware\VCM\CollectorData\enterprise-certificate-GUID.pem
```

The Agent already has the certificate if the Agent was installed using the package in the Collector `Packages` folder.

```
C:\Program Files (x86)\VMware\VCM\Installer\Packages
```

The certificate is for that Collector only. If the Agent was installed from a different Collector package, you must copy the certificate from the Collector you want. You can use FTP in binary mode to copy the certificate.

After you copy the PEM file to the Agent machine, use the `CSI_ManageCertificateStore` utility to add it to the Agent certificate store. The command is slightly different depending on your UNIX variant.

HPUX

```
CSI_REGISTRY_PATH=/opt/CMAgent SHLIB_
PATH=/opt/CMAgent/CFC/3.0/lib:/opt/CMAgent/ThirdParty/1.0/lib
/opt/CMAgent/CFC/3.0/bin/CSI_ManageCertificateStore -iz -fpath-to-
pem/filename.pem
```

Solaris and Linux

```
CSI_REGISTRY_PATH=/opt/CMAgent LD_LIBRARY_
PATH=/opt/CMAgent/CFC/3.0/lib:/opt/CMAgent/ThirdParty/1.0/lib
/opt/CMAgent/CFC/3.0/bin/CSI_ManageCertificateStore -iz -fpath-to-
pem/filename.pem
```

MAC

```
CSI_REGISTRY_PATH=/opt/CMAgent DYLD_LIBRARY_
PATH=/opt/CMAgent/CFC/3.0/lib:/opt/CMAgent/ThirdParty/1.0/lib
/opt/CMAgent/CFC/3.0/bin/CSI_ManageCertificateStore -iz -fpath-to-
pem/filename.pem
```

AIX

```
CSI_REGISTRY_PATH=/opt/CMAgent
LIBPATH=/opt/CMAgent/CFC/3.0/lib:/opt/CMAgent/ThirdParty/1.0/lib
/opt/CMAgent/CFC/3.0/bin/CSI_ManageCertificateStore -iz -fpath-to-
pem/filename.pem
```

Because certificate information is maintained in memory while the Agent is running, you must restart the Agent after you add a certificate. Agents that run in inetd mode periodically stop themselves, but Agents that run in daemon mode need to be manually restarted.

Patch Assessment

VCM5.0 and later can do patch assessment inspections. Support for these inspections includes new metadata in the Collector IMD tables. The new metadata gets replicated to the Agent the same way as previous metadata—at first communication between Collector and Agent, all metadata is sent in the initial request, and the Agent stores the metadata on a per-Collector basis in `/opt/CMAgent/data/db/DtmDB/RDM/`*`collector-name`*. On subsequent requests from the Collector, only modified metadata is replicated to the Agent.

The files needed for patch assessment are stored in the `/opt/CMAgent/data/db/DtmDB/RDM/`*`collector-name`*`/PatchContent` directory. This directory contains Lumension patch data (PLS files) for each patch to be evaluated.

When patch assessment data exists, the large initial copy operation might be a problem. The size of an initial request might cause the `ListenerWorkerDaemon` and Agent processes to grow to over 250MB each. To protect the Agent machine from excessively large VCM processes, Agents earlier than 5.0 shut down the processes when they exceed approximately 150MB. In 5.0 and later, Agents allow up to approximately 380MB before shutting down the processes.

Exploratory UNIX Agent Troubleshooting

Problems with the UNIX Agent might be varied, occur because of factors that are not readily observable, and be difficult to work around by following tightly constrained instructions. The following sections explain some of the broad circumstances that might arise and the approaches that you might take to identify and solve the problem.

Installation Errors

If the UNIX Agent installation reports an error, explore the following.

- Copy `/opt/CMAgent/install/DebugEvent_cis.dbe` to a Collector, and examine its contents using the Debug Event Viewer.
- Examine the contents of `/opt/CMAgent/install/BootstrapInstall.log` using a text editor.
- Examine `/opt/CMAgent/install/csi.config` using a text editor. The file contains installation configuration information, including which user account can run the Agent.
- Examine the contents of `/opt/CMAgent/install/install.log` using a text editor.

Collector Cannot Contact the Agent

When the Collector cannot contact the Agent machine, look at the following on the Collector server.

- Try an `nslookup` of the Agent.

  If it fails, edit the `etc\hosts` file on the Collector to map the Agent machine name to its IP address.

- From a command prompt on the Collector, `ping` the Agent machine by name or IP address.

- Try to `telnet` or `ssh` to the Agent.

  On Windows, you can install the Putty application to open a UNIX console session to another machine.

Agent is Unresponsive

To make sure that the Agent is enabled and listening, look at the following items on the Agent machine.

- Check `/opt/CMAgent/data/db/PDS/CertStore` to make sure that the Collector certificate is installed.

- Check that the Collector PEM certificate was pushed to `CSI_ManageCertificateStore`.

  Use the same commands from "Collector Certificates" on page 80, but change `-iz -f` to `-l`.

- Check the file in `/var/log/messages` (Linux) or `/var/adm/messages` (Solaris) to see if (x)inetd reported any errors when it was reconfigured to enable the Agent.

- Modify `/opt/CMAgent/ECMu/1.0/scripts/inetd-agent`, and add `-b` immediately before `-u`.

  Try to contact the Agent again, and check `/var/log/messages` (Linux) or `/var/adm/messages` (Solaris) for entries that are reported by `CsiAgentListener`.

- Check `/var/log/messages` (Linux) or `/var/adm/messages` (Solaris) for entries that show the `csi-agent` process starting.

  The messages might only appear if the machine is set up to log the message type.

- Copy `/var/log/CSI/log/DebugEvent_Default.dbe` to the Collector, and examine its contents using the Debug Event Viewer.

- On Linux, type **`netstat -l | grep csi-agent`**

  The command should return `tcp 0 0 :csi-agent *: LISTEN`

- On Solaris, type **`netstat -a | grep csi-agent`**

  The command should return `.csi-agent *. 0 0 0 0 LISTEN`

- You can use a machine that has `nmap` installed, usually Linux, to determineif the port is open to the network.

  The command **`nmap -sT -v -p 26542`** ***`agent-machine`*** returns a result similar to the following.

```
Starting nmap V.  3.00 ( www.insecure.org/nmap/ )
Host {agent-machine} ({ip-address}) appears to be up … good.
Initiating Connect() Scan against {agent-machine} ({ip-address})
Adding open port 26542/tcp
The Connect() Scan took 0 seconds to scan 1 ports.
Interesting ports on {agent-machine} ({ip-address})
Port     State     Service
26542/tcp  open     unknown

Nmap run completed – 1 IP address (1 host up) scanned in 0 seconds
```

- (Optional) Install `top` on the Agent machine, and monitor to determine if the listener starts when you `telnet` to the Agent port: **`telnet`** ***`agent-machine`*** **`26542`**

  See "Monitor Processes with the top Utility" on page 84

  As `telnet` runs, you should see the `CsiAgentListener` process appear and disappear in the `top` display.

Collections Return No Data

Sometimes the UNIX Agent machine is reachable but does not return data when you perform a collection. There are many reasons why this might happen, and you need to consider a variety of factors before choosing a solution. The following sections explain the circumstances that might arise and the areas in which you need to look.

Agent Processes Have Not Started

To verify that Agent processes are running, do the following. You might see other processes such as `gawk` or `zip` depending on when you run the commands.

- To show all Agent processes, type **`ps -ef | grep CMAgent | grep -v grep`**
- (Optional) Install `top` on the Agent machine, and monitor the processes.

  See "Monitor Processes with the top Utility" on page 84.

Agent Processes

- These processes appear when the Agent is running.

  ```
  /opt/CMAgent/ECMu/3.0/bin/Agent
  /opt/CMAgent/CFC/1.0/bin/CsiListenerWorkerDaemon
  ```

  Because Red Hat Enterprise 2.1 reports all threads in a process, and not just the process itself, Red Hat might display many instances of the processes. On other platforms, you see only one `Agent` and `CsiListenerWorkerDaemon`.

- Each time the Collector sends a message to the Agent, the following process is started by (x)inetd.

  ```
  /opt/CMAgent/CFC/3.0/bin/CsiAgentListener
  ```

- On Linux, AIX, and HPUX you see the following external Birdstep lock manager. The external lock manager does not appear on Solaris, which uses an internal lock manager.

  ```
  /opt/CMAgent/ThirdParty/1.0/bin/lm
  ```

  On Linux, an `lm` process is always running for the status manager (SM) database when `Agent` is running. When a collection occurs, a second `lm` appears for the Collector-specific data model (DtmDB).

To list all Collector directories, with the most recently collected directory at the top, type **`find /opt/CMAgent/data -name Master | xargs ls -ldt`**

Monitor Processes with the top Utility

If the `top` utility is installed and available, you can use it to monitor processes.

Procedure

1. Start the `top` utility.
2. Type **`u`**.
3. At the **Which User (Blank for All):** prompt, type the user account that the Agent is installed as.
4. Type **`s`**.
5. At the **Delay between updates:** prompt, type **`1`**.

Agent Was Reinstalled

The first collection after reinstalling the Agent always fails, and returns an error stating that the replication timestamp is out of sync.

Performing a second collection forces the Collector to re-replicate the data model to the Agent.

Monitoring Collections

At the first collection, the following directory appears when the Agent starts to process the replicated data model.

```
/opt/CMAgent/data/db/DtmDB/RDM/collector-name
```

Next, the following directory appears when the Agent begins to process the inspection request.

```
/opt/CMAgent/data/collector-name
```

On subsequent collections, an additional directory appears when the Agent begins to process the inspection request. The request ID format is similar to a Windows GUID.

```
/opt/CMAgent/data/collector-name/request-ID
```

The request ID directory contains files downloaded for remote commands, and a `Results` directory to hold files that will be returned. To monitor the `Results` directory, do the following.

- To see HDS files being created for each individual data class, type **`ls -lt`**

  The most recently inspected data class appears at the top of the list. If the Agent seems to have stopped, the list shows which data class it was processing at the time.

  Typing **`ps -ef | grep gawk | grep -v grep`** shows if the Agent is actively inspecting the data class and might show if the `gawk` script has hung. For example, `gawk` hangs when trying to perform a checksum on a pipe file.

- To report the number of files in the directory, type **`ls | wc -l`**

  A collection of all data classes generally creates about 40 files in the `Results` directory, including HDS files and FileUpload* files. It might also create a debug event (DBE) file.

After inspection finishes, the contents of the `Results` directory are zipped and stored as follows, and the `Results` directory is deleted.

```
/opt/CMAgent/data/collector-name/Package/request-ID.zrp
```

If the Results directory is not deleted, look for a DBE file in the directory and copy it to the Collector for viewing.

To examine the ZRP file on the Agent, capture it when it is created. See "Capture the ZRP on the Agent" on page 85.

Capture the ZRP on the Agent

The Agent sends the Results.zrp file to the Collector, and deletes it after the Collector acknowledges receipt of the file. To examine the ZRP file on the Agent, capture a copy when the file is created, before the Agent deletes it.

Procedure

Enter the following shell command.

```
until ls *zrp 2>/dev/null; do sleep 1; done; cp *zrp save.zrp
```

Logging Errors

The following file captures errors reported by the Agent processes that are not specific to a request, such as when an unauthorized Collector tries to contact the Agent.

(Linux) /var/log/CSI/log/DebugEvent_Default.dbe
(Solaris) /var/adm/CSI/log/DebugEvent_Default.dbe

After the Agent begins processing a request, it switches to capturing errors in a DBE file in the following directory. The DBE file is eventually returned to the Collector.

/opt/CSI/data/*collector-name*/*request-ID*

NOTE Effective DBE interpretation often requires expert knowledge of the Agent software.

The following file captures errors when they occur before the Agent logs to the DebugEvent_Default.dbe file. You might need to check /etc/syslog.conf to verify that entries are being written to the messages file.

(Linux) /var/log/messages
(Solaris) /var/adm/messages

To make the Agent log additional entries to the messages file, edit /opt/CMAgent/ECMu/1.0/scripts/inetd-agent, and add -b immediately before -u. The -b causes the CsiAgentListener to log information about which user and group it is using, each step as it daemonizes itself to detach from (x)inetd, and when it exits.

If you suspect a race condition in startup, shutdown, and communication between CsiAgentListener and Agent or CsiListenerWorkerDaemon, enable informational messages, which reveal each of the processes starting up, shutting down, checking to see that it is safe to shut down, and determining that the process is available to be contacted. Edit /opt/CMAgent/ECMu/1.0/scripts/inetd-agent, and add LOG_INFOS=1 immediately before CSI_REGISTRY_PATH.

Inspections are Failing

The following files capture errors when inspections are failing.

(Linux) /var/log/secure
(Solaris) /var/adm/messages

Look for the following errors in the files.

- CSISecureHigh

  Errors from the RunHigh executable file, indicating violation of rules enforced by the suid program for running root privilege inspections.

- CSISecureLow

  Errors from the RunLow executable file, indicating violation of rules enforced by the sgid program for running non-root privilege inspections.

- CSISecure

  Errors from the RunRemote executable file, indicating violation of rules enforced by the suid program for running root privilege remote commands. The File Upload job also uses this executable file when it needs to copy a file that has restricted permissions.

  CSISecure messages are deliberately unhelpful in debugging because they would otherwise expose the rules being enforced.

NOTE An error is logged if you run any of the executable files manually.

Look for the following problem areas.

- The user who is configured to run the Agent is not a member of the cfgsoft group.
- The user who is configured to run the Agent does not have the cfgsoft group as the default group.
- The user who is configured to run the Agent has a no-login shell.

If inspections are not returning all of the expected information, examine the DBE file from the Collector *request-ID* directory. The file captures errors reported to the stderr of the gawk inspector. Messages will be from the ReadInternal function of the CEcmScriptResultStream class and indicate which data class was being processed at the time. The error might span several sequential messages.

Save the ZRP File at the Collector

If the Agent returns a ZRP file to the Collector, debug events are logged to the SQL database, and you can view them there. To preserve the source of the errors, prevent the ZRP file from being deleted after processing.

Procedure

1. In the VCM Console, select **Administration**.
2. Click **Settings** > **General Settings** > **Database**.
3. Select one of the following options.

   SAS: Should SAS data files be kept on the disk after they have been processed
   ETL: Should ETL data files be kept on the disk after they have been processed
4. Look for saved Results.zrp files in a request ID directory under the DSRoot directory.

Run Executable Permissions

The Collector might report that the job succeeded, but show no data. For data to appear, executable files in /opt/CMAgent/ECMu/x.x/bin need the following permissions.

NOTE The csi_acct name might be different if the Agent is installed using a different account.

- **RunHigh.** owner `root`, group `cfgsoft`, mode `r-sr-x---`
- **RunLow.** owner `csi_acct`, group `csi_acct`, mode `r-xr-s---`
- **RunRemote.** owner `root`, group `cfgsoft`, mode `r-sr-x---`

If permissions are correct, check DBE files for errors stating that `RunHigh`, `RunLow`, or `RunRemote` failed. See "Run Executable Logging" on page 87 for information about the level of error logging.

Run Executable Logging

If `RunHigh`, `RunLow`, or `RunRemote` fails, the executable file logs errors of type `auth.err` to `syslog` as follows.

- (Linux) `/var/log/secure`
- (Solaris) `/var/adm/messages`
- Wherever these message types are configured to be logged as set up in `/etc/syslog.conf`

The error messages only say that the executable program failed. The messages deliberately avoid details about the failure so that a hacker cannot use the information to design an attack that defeats the security of the program.

To get detailed messages, rebuild `RunHigh`, `RunLow`, and `RunRemote` with more logging enabled. Search for a commented-out `syslog` entry in the code, remove the comment markers, and rebuild the programs.

Detailed logging creates messages in `syslog` that have an error code, which VCM engineering uses to trace to a source file and determine the cause of the failure.

Account and Group Configuration

The Collector might report that the job succeeded, but no data appears. For data to appear, the three executable files in `/opt/CMAgent/ECMu/x.x/bin` need accounts and groups to be configured.

NOTE If the installation creates the accounts and groups, the uninstall process removes them. It the accounts and groups were preexisting, the uninstall process does not remove them.

- **The `csi_acct` user account.** Must be properly created and cannot have a shell that permits logins. The shell for `csi_acct` must be listed under the CSIRegistry `NoLoginShells`, and the no login shell must exist on the Agent machine.
- **The `csi_acct` group.** By default, the primary group for the `csi_acct` user is the `csi_acct` group. Like the `csi_acct` user account name, the group name can be changed during Agent installation if you want to use another name or an existing group. Using an existing group might create a security risk depending on the existing group privileges. Use a group that has no elevated permissions, like the standard `nobody` group.
- **The `cfgsoft` group.** Must be created and have this exact name. The `csi_acct` user must be a member of the `cfgsoft` group, but the `cfgsoft` group should not be the primary group for `csi_acct`.

When troubleshooting the `setuid` binary files, check `nsswitch.conf` to confirm that all user lookups are going to the files first. If they are not, the accounts might need to be created in your environment (for example: YP, LDAP, or Active Directory). A common problem is that the user account is partially created in the cloud, so the security checks fail. If none of the user information is in the cloud, the secondary check to files should work properly.

Also, check the mount options for the file system. A common security practice is to mount `/usr`, `/opt`, and `/usr/local` with `notsetuid` and `nosuid` options to prevent `setuid` binaries from running. Doing so prevents `RunHigh`, `RunLow`, and `RunRemote` from running.

Monitoring Network Traffic

In rare cases, you might want to monitor TCP/IP traffic for the Agent machine. One way to monitor traffic is to have X-Windows access to the Agent machine and have the Wireshark/Ethereal package installed there.

Configure Wireshark/Ethereal to Capture Data

Set up Wireshark/Ethereal to capture network traffic data.

Procedure

1. From the Wireshark/Ethereal tool bar, click **Capture** > **Capture Filters**.
2. In the **Filter name** text box, type **`csi`**.
3. In the **Filter string** text box, type **`port 26542`**.
4. Click **New**.
5. Click **Save**.
6. Click **Close**.

Configure Wireshark/Ethereal Coloring Rules

Set up Wireshark/Ethereal to display the start of meaningful messages in colors.

Procedure

1. From the Wireshark/Ethereal tool bar, click **View** > **Coloring Rules**.
2. Click **New**.
3. In the **Name** text box, type **`Agent Responses`**.
4. In the **String** text box, type **`data contains HTTP and data contains 200 and data contains OK`**.
5. Click **Foreground color**, and select a color (for example, blue).
6. Click **OK** and click **New**.
7. In the **Name** text box, type **`VCM Pings`**.
8. In the **String** text box, type **`data contains HTTP and data contains ping`**.
9. Click **Foreground color**, and select a color (for example, green).
10. Click **OK** and click **New**.
11. In the **Name** text box, type **`VCM Commands`**.
12. In the **String** text box type **`data contains HTTP and data contains POST and data contains execute`**.
13. Click **Foreground color**, and select a color (for example, red).
14. Click **OK**.
15. Click **Save** and click **OK**.

Capture Traffic with Wireshark/Ethereal

Start the capture of network traffic in Wireshark/Ethereal.

Procedure

1. From the Wireshark/Ethereal tool bar, click **Capture** > **Start**.
2. In the **Capture Options**, click **Capture Filter**.
3. Select **csi**.
4. Click **OK**.

Wireshark/Ethereal Capture Results

Messages that flow in and out of the Agent port (26542) appear in the Wireshark/Ethereal display. A single inspection might show all of these results.

The message sizes indicated are current as of VCM Linux or UNIX Agent build 1.0.0.1270.

Table 8–1. Wireshark/Ethereal Capture Results

Sequence	Color	Description	Content Length (approximate)
1	Green	Ping	n/a
2	Blue	Ping result	164 bytes
3	Red	Session Negotiation	2,222 bytes
4	Blue	Negotiation Complete	3,538 bytes
5	Red	Inspection Request	Varies based on number of data classes selected and if replication is occurring. For example, a full collection of `Machine.General` with full data model replication has a `Content-Length` of 71,934 bytes.
6	Blue	Request Scheduled	8,386 bytes
7 (one or more of this set)	Red	Session Negotiation	2,222 bytes
	Blue	Negotiation Complete	3,538 bytes
	Red	Check Status	2,618 bytes
	Blue	Current Status	Varies depending on the status of the request. For example, if the request is still in progress, `Content-Length` is usually 9,110 bytes, but when the request is finished, `Content-Length` is 8, 330 bytes.
8	Red	Session Negotiation	2,222 bytes
9	Blue	Negotiation Complete	3,538 bytes

Sequence	Color	Description	Content Length (approximate)
10	Red	Transfer Results	2,254 bytes
11	Blue	Transferred data	Varies based on the number of data classes inspected, delta versus full, and operating system. For example, a full collection of `Machine.General` on a Red Hat 9 platform has a `Content-Length` of 6,618 bytes.
12	Red	Session Negotiation	2,222 bytes
13	Blue	Negotiation Complete	3,538 bytes
14	Red	Acknowledge Transfer	2,630 bytes
15	Blue	Acknowledged	7,554 bytes

Capturing Traffic with tcpdump

You can use the `tcpdump` command to gather the same network traffic data that Wireshark/Ethereal does, but in a less user friendly format.

Start `tcpdump` with the following command.

```
/usr/sbin/tcpdump -s -l 256 -x -X port 26542 | tee tcpdump.log
```

The `tcpdump.log` file contains information similar to what you see in Wireshark/Ethereal, but without color highlighting to show you the boundaries between message types. To determine where each of the different message types begins, search for the string `HTTP`.

Index