CA Asset Converter

Product Guide

Release 12.9.00

CA Technologies Product References

This document set references the following CA Technologies brands and products:

- CA Asset Converter
- CA Asset Portfolio Management (CA APM)
- CA Business Intelligence
- CA Client Automation

  (formerly CA IT Client Manager)
- CA Configuration Management Database (CA CMDB)
- CA Embedded Entitlements Manager (CA EEM)
- CA Management Database (CA MDB)
- CA Process Automation™
- CA Service Catalog
- CA Service Desk Manager
- CA Software Asset Manager (CA SAM)
- CA SiteMinder®

This document set also references the following component, which formerly used a different name:

- Common Asset Viewer

  (formerly Asset Management System or AMS)

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

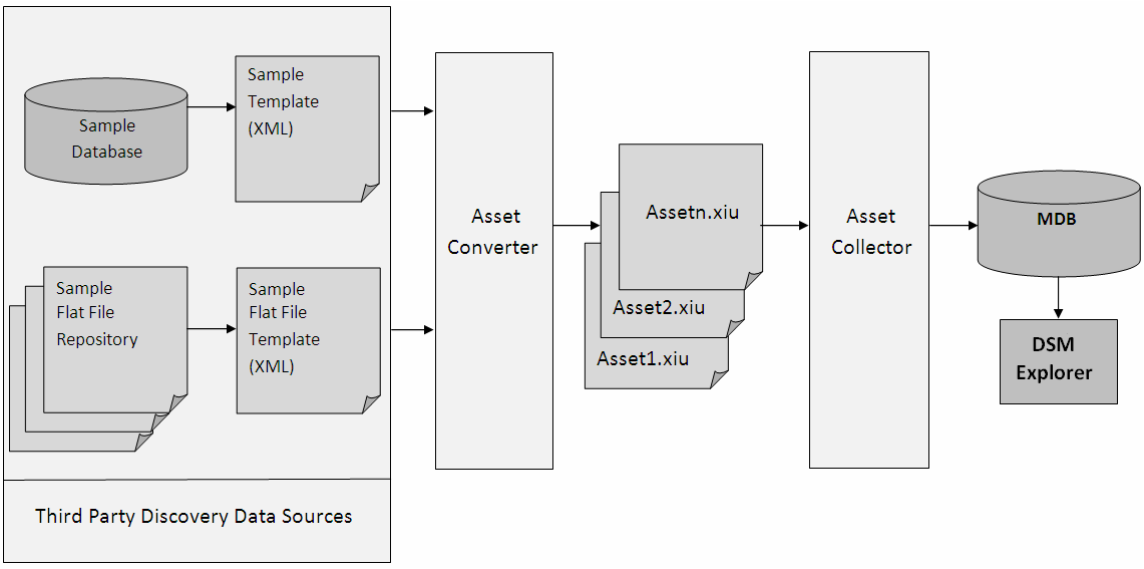
Chapter 1: Introduction

This section contains the following topics:

Overview

The CA Asset Converter extracts asset data from various asset discovery products and maps the extracted data into a predefined target format. The Asset Collector defines the target format, where each asset (computer) is represented as a separate XML file. The target asset XML file contains asset-related data in hierarchical order, and XML tags with parent-child relationships represent the hierarchy. The discovered asset data is passed to the Asset Collector and can be viewed using the DSM Explorer.

The CA Asset Converter provides sample templates for asset conversion. The following illustration provides the process flow of this product component.



Audience

This guide is intended for the asset administrator who is responsible for extracting asset data from various discovery products into a predefined target format. Some tasks you perform using this product component include the following:

- Install the CA Asset Converter (see page 15).
- (Optional) Enable multi-tenancy (see page 16).
- Create a mapping file for a database (see page 17) or flatfile (see page 21).
- Configure the mapping file (see page 25).
- Extract asset data (see page 35).
- View log files (see page 35).

Note: All XML files you create, and the sample XML code used in this guide must be well-formed and comply with XML and XSD standards.

Chapter 2: System Information

This section contains the following topics:

Operating Systems and Databases

CA Asset Converter

The CA Asset Converter supports the following operating systems and database management systems. CA Technologies supports each component for the duration of its lifecycle (as determined by the manufacturer) or until CA Technologies announces that we no longer support it.

Note: Use an NT File System (NTFS) instead of a FAT32-based file system to avoid maximum file size limit errors when the product component operates.

Operating System (OS)	Database
■ Microsoft Windows Server 2003 SP1 (Enterprise Edition, Standard Edition) ■ Microsoft Windows 2000 SP4 (Advanced Server, Server, Professional) ■ Microsoft Windows XP Professional SP2	■ Microsoft SQL Server 2000 or higher. ■ Oracle 8.0 or higher. ■ IBM DB2 8.0 or higher. ■ MySQL 5.1 or higher.

The product also supports Open Database Connectivity (ODBC) to connect to databases other than the databases previously listed. To use ODBC to connect to the required databases, verify that the most current database drivers are installed.

Asset Collector

The Asset Collector supports the following operating systems and database management systems. CA Technologies supports each component for the duration of its lifecycle (as determined by the manufacturer) or until CA Technologies announces that we no longer support it.

Operating System (OS)	Database
The Asset Collector is supported only on scalability servers on Windows operating environments.	■ Microsoft SQL Server 2008 SP1. ■ Microsoft SQL Server 2005 SP3. ■ Microsoft SQL Server 2005 SP2. ■ Oracle 10g Release 2 (10.2.0.4).

SQL Bridge and Oracle Bridge

If you have installed CA Service Desk Manager or CA APM on an MDB based on the SQL Bridge or Oracle Bridge, you can synchronize the tenant information (see page 40) populated by those products into the CA Client Automation database. The SQL Bridge and Oracle Bridge synchronization support the following databases:

- **SQL Bridge**. SQL Server 2005 and 2008 on Windows as both the source and target MDB.
- **Oracle Bridge**. SQL Server 2005 and 2008 on Windows as the source MDB for the synchronization with Oracle 10g Release 2 (10.2.0.4) as the target MDB.

System Requirements

The following requirements must be met or exceeded for the CA Asset Converter to install and operate properly:

Component	Requirement
Processor	Minimum: Single processor, 2.0 GHz Recommended: Dual processor, 2.0 GHz
Memory	Minimum: 256 MB
Hard Drive	Minimum: 8 MB

International Support

The CA Asset Converter only supports English installations.

Chapter 3: Documentation

This section contains the following topics:

View the CA Technologies Bookshelf

The CA Technologies Bookshelf provides your product documentation set in Section 508-compliant HTML format, and a print version of each guide. The CA Technologies Bookshelf is installed automatically with the product and you can access it by clicking the Bookshelf link in the product.

You can download and extract the CA Technologies Bookshelf for your product (a ZIP file) from CA Support Online.

To extract the ZIP file and view the CA Technologies Bookshelf

1. Use an archive product such as WinZip.
2. Extract the content to a local folder.
3. Double-click the Bookshelf.html file in the Bookshelf folder.

   The CA Technologies Bookshelf opens, and you can use it to view and search the product documentation.

Where to Find Documentation

You can access the CA APM documentation in the following locations:

- Click the Bookshelf link in the product.
- The Doc directory on the installation media. Double-click Bookshelf.html.
- Technical Support at http://ca.com/support.

How to View and Search PDFs

To view PDF files, download and install the Adobe Reader from the Adobe website if it is not already installed on your computer.

If you open a PDF file in Adobe Reader in the CA Technologies Bookshelf and search, the individual PDF file is searched and you see the individual instances of the search term.

Chapter 4: Installing

This section contains the following topics:

Installation Planning

To plan for a successful CA Asset Converter installation, complete the following steps:

1. Verify that the Java Runtime Environment (JRE) 1.6 or greater is installed.
2. Verify that the path is set in the Windows environment variables.
3. Determine the home directory in which you want to install the CA Asset Converter. The default home directory is C:\Program Files\CA\CA Asset Converter.

Install the CA Asset Converter

You install the CA Asset Converter to extract asset data from various asset discovery products and map the extracted data into a predefined target format.

To install the CA Asset Converter

1. Extract the contents of the CA Asset Converter.zip file (available on the installation media) into a folder on your computer.
2. Process the CA Asset Converter Setup.msi file.

   The CA Asset Converter Wizard opens.
3. Follow the on-screen instructions to complete the installation.

   The CA Asset Converter is installed in the home directory. You can change this location during the installation.
4. Verify that you can start the product component by selecting Start, Programs, CA, Asset Converter.

Installation Considerations (Asset Collector on Oracle)

Consider the following information when installing the Asset Collector on Oracle:

- Oracle 10g Release 2 (10.2.0.4) database is supported as an MDB for the Asset Collector, but Oracle must be installed as a remote MDB on a dedicated Sun Solaris operating environment.
- On Solaris operating environments, installing the MDB on Oracle requires Oracle 10g Release 2 (10.2.0.4) with the latest Oracle patches p7008262_10204_Solaris-64, p5718815_10204_Solaris-64, and p7706710_10204_Solaris-64.
- Apply Oracle 10g Release 2 (10.2.0.4) on all Oracle client installations.
- The Asset Collector supports only the EZCONNECT method of connection to Oracle. For more information about setting the connection method to EZCONNECT, see your Oracle documentation.

Enable Multi-Tenancy

With multi-tenancy, multiple independent tenants can share a single implementation of the CA Asset Converter. You can enable multi-tenancy to allow multiple tenants to use the CA Asset Converter and track the output asset XML files that are generated from each tenant.

To enable multi-tenancy for the CA Asset Converter

1. In the C:\Program Files\CA\CA Asset Converter folder, locate the asset_converter_config.xml file.
2. Use a text editor such as Notepad to open the asset_converter_config.xml file.
3. Locate the following line of code:
   `<multi-tenant-mode value=""/>`
4. Enter *true* in between the value quotes. For example, value="true".
5. Save the asset_converter_config.xml file.

   Multi-tenancy is enabled and you can use the CA Asset Converter in this mode.

Chapter 5: Extracting Asset Data

This section contains the following topics:

How to Extract Asset Data

You can use the CA Asset Converter to extract asset data from a selected database or flatfile. To extract the data, complete the following steps:

1. Create a mapping file for a database (see page 17) or flatfile (see page 21).
2. (Optional) Configure the mapping file (see page 25).
3. Extract asset data (see page 35).
4. (Optional) View log files (see page 35).

Create a Mapping File (Database)

You can create a mapping file for a database and define the following mapping file parameters:

Note: All XML files you create, and the sample XML code used in this guide must be well-formed and comply with XML and XSD standards.

Datasource type

Enter datasource type "DataBase or database".

Note: The Datasource tag has subtypes and connection properties tags.

Subtype

Each database is assigned a predefined numeral as a unique code. The CA Asset Converter identifies a particular database on the associated codes and loads the property names required for the connection to the database. You can use any of the following codes:

- 13 (DB2)
- 12 (MYSQL)
- 11 (ORACLE)
- 10 (SQLSERVER)
- 14 (ODBC)

Example: Sample XML for Typical Connection Properties

The following sample XML code illustrates the typical connection properties of a mapping file for a database.

```
<?xml version="1.0" encoding="UTF-8"?>
<asset-converter>
    <datasource type="DataBase">
    <subtype>10</subtype>
    <connection-properties>
         <property name="Server">MyHost</property>
         <property name="Port">1433</property>
         <property name="Database">AssetDB</property>
         <property name="Username">sa</property>
         <property name="Password">password</property>
    </connection-properties>
    </datasource>
</asset-converter>
```

Define Connection Properties (Database)

You can define the following connection properties for a mapping file to connect to a database:

Note: For ODBC-specific configurations, only the Database, Username, and Password are required. The other parameters are ignored.

Server

Name of the database server.

Port

Connection port number.

Database

Name of the database. For ODBC-specific configurations, use the Data Source Name in the local host as the database name.

Username

Username to connect to the database.

Password

Password to connect to the database.

IntegratedSecurity

Set to true to enable Windows integrated authentication.

Map and Extract Data with Query and Value Attributes (Database)

The data-related tags that specify how to process and retrieve the mapping file and data typically consist of query and value attributes.

- A *query attribute* in any tag consists of a valid SQL query. The result of the query is used by tags having value attributes referencing the tags having a query attribute referencing the tag.

  The asset tag is the root tag for an asset and should have a query listing unique asset IDs from the target database. One XML file is created for each unique asset ID contained in the result set of the query attribute for the asset tag. All subsequent tags under the asset tag either have a query or value attribute referencing another tag. At runtime, the attributes receive runtime values from the tags they reference.

  For example, if a general tag query uses asset ID extracted in an asset tag query, the query attribute of the general tag will contain an annotation referencing the asset tag. This means that the general tag for each asset ID query changes based on the asset ID.

- The *value attribute* of the tags referencing some other tag is similar to the previous example.

Extract Data Using Annotations (Database)

You can use annotations in the mapping file to extract data. The annotations are typically used in *query* or *value* tags. You can derive any field in the input XML using direct values or annotated values.

- Annotated values. If the query or value contains text within curly brackets ({ }) it is an annotated value.
- Constant values. For example, *<host_name value="myHostName"/>*. Each host_name tag in the final output XML files will be set to "myHostName".

You can group annotations into a *relative annotation path* and an *absolute annotation path*.

Example: Use a Relative Annotation Path

In this example, when the annotation is processed, the XML assigns the value of the ProcCount column to No. of Processors. The column value is derived using the query in the group name System tag. The generated XML will have the No. of Processors value derived by querying the database using the query attribute of the Group Name System tag. The value is directly relative to the parent tag query.

```
<asset query="select AssetID from Computer" translator="ACBsFmt" version="1_0">
    <general>
    ...
    </general>
    <hardware>
    <group name="GeneralInventory">
        <group name="System" query="select p.ProcCount from Computer c where
        c.AssetID = {asset.AssetID}">
        <attribute name="No. of Processors" type="string"
        value="{group(name='System').ProcCount}"/>
        </group>
    </group>
    </hardware>
     ...
</asset>
```

Example: Use an Absolute Annotation Path

```
<asset query="select AssetID from Computer" translator="ACBsFmt" version="1_0">
    <general>
    ...
    </general>
    <hardware>
        <group name="GeneralInventory">
        <group name="System" query="select p.ProcCount from Computer c where
        c.AssetID = {asset.AssetID}">
        <attribute name="No. of Processors" type="string"
        value="{asset:hardware:group(name='GeneralInventory'):group(name='System'
        ).ProcCount}"/>
        </group>
    </group>
    </hardware>
    ...
</asset>
```

Map for ODBC

ODBC supports only forward movement in the query result set and this behavior affects how you create a mapping file. Verify that the value mapping matches the order of the query selections in the mapping file.

Example: Value mapping and query selections for an ODBC mapping file

In this example, the values in the mapping file have been mapped in the same order as the query selection.

```
<general query="select c.devicename, c.hostname, c.address from Computer c">
    <host_name value="{general.devicename}"/>
    <default_hostname value="{general.hostname}"/>
    <default_address value="{general.address}"/>

</general>
```

In the query attribute, devicename, hostname, and address are queried in that order, from left to right. The values that follow are also mapped in the same order. A mismatch in the order results in null values in the output XML files.

Encrypt the Password (Database)

You can use the password encryption utility to provide additional security.

To encrypt the password

1. Open a Command Prompt window and enter the following command:

   ```
   ac -p 'password'
   ```

   In this command, you must enter the password.

   The encrypted password is displayed.

2. Copy the password to the password property of the connection property tag.

   Important! You must add the encrypted="true" attribute to the password field.

   ```
   <property name="password" encrypted="true">hj!89==<property>
   ```

Create a Mapping File (Flatfile)

The mapping file is a template file provided as input to the CA Asset Converter. The CA Asset Converter reads the mapping file, retrieves the connection properties, and reads the specified file path information for each file used for the mapping. The CA Asset Converter uses the mapping file to read data from specified flatfiles, process the data, and generate the expected asset files.

Note: The XML files must be well-formed and comply with XML and XSD standards.

Example: FlatFile Mapping

The following sample FlatFile mapping contains connection properties.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<asset-converter xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<datasource type="flatfile">
    <connection-properties>
        <property filename="main" filepath="MainFile.txt" key="1"
        column_name="true"/>
        <property filename="system" filepath="SystemFile.txt" key="1"
        column_name="true"/>
    </connection-properties>
    <delimiter>
        <text-delimiter value=","/>
    </delimiter>
</datasource>
</asset-converter>
```

Define Connection Properties (Flatfile)

You can define connection properties for a mapping file for a flatfile.

Important! One property tag is mandatory within the connection properties.

- Main or Master file represents unique or non-replicated asset data.
- Dependent files represent replicated asset data.
- Main file contains primary keys, which are reference fields for the dependent files.

The file used in the asset tag is always treated as the Master file. All group tags must reference a file (master or dependant). All the attributes of the group must be mapped to the columns of the file.

To define connection properties for a flatfile

1. Specify the datasource type as either FlatFile or flatfile.
2. Specify the following property tag attributes:

   filename

   (Required) Name of the input flatfile.

   filepath

   (Required) Path of the input flatfile.

key value

(Required) Primary key of the defined file.

column name

(Optional) Defines the column heading. Valid values include the following:

- true. Read the data from the second row.
- false. Read the data from the first row (column heading).

Note: If this attribute is missing, the CA Asset Converter reads data from row 1.

3. (Optional) Specify the delimiter parameter using the text-delimiter tag under Delimiter.

text-delimiter

(Optional) The default text-delimiter tag is a comma (,). You can also specify the following delimiters:

- Pipe (|)
- Pound (#)
- Tilde (~)

Map and Extract Data (Flatfile)

You can define the tags and parameters to map and extract data from flatfiles.

AssetID, MAC_Address, IPAddress, ComputerName
1,01:23:45:67:89:AB,172.16.32.4,system01-xp
2,01:23:45:67:89:AB,172.16.32.3,system02-xp
3,01:23:45:67:89:AB,172.16.32.12,system03-xp
4,01:23:45:67:89:AB,172.16.32.21,system04-xp
5,01:23:45:67:89:AB,172.16.32.22,system05-xp
MainFile.txt

AssetID, Model, Type, Vendor
1, Optiplex, Desktop, Dell Inc
2, Y140, Laptop, IBM
3, Optiplex, Desktop, Dell Inc
4, Optiplex, Desktop, Dell Inc
5, Y140, Laptop, IBM
SystemFile.txt

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<asset-converter xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<datasource type="flatfile">
    <connection-properties>
    <property filename="main" filepath="MainFile.txt" key="1" column_name="true"/>
    <property filename="system" filepath="SystemFile.txt" key="1"
    column_name="true"/>
    </connection-properties>
        <delimiter>
        <text-delimiter value=","/>
        </delimiter>
</datasource>
<processing-info>
    <max-assets-to-process value="3"/>
</processing-info>
<output-settings>
    <dest-dir value="assets-ff"/>
    <output-filename value="assets" host-name="true"/>
</output-settings>
CA MDB
</mdb>
<asset file="main" translator="ACBsFmt" version="1_0">
    <general file="main">
        <host_name value="[4]"/>
        <default_hostname value="[4]"/>
        <default_address value="[3]"/>
        <default_mac value="[2]"/>
    </general>
...
</asset>
</asset-converter>
```

Extract Data Using Annotations (Flatfile)

You can use annotations in a flatfile to extract data. The file name is required for the general tag. Hard-coded values are not permitted for the general tag attributes. For example, host_name="assetname" is not permitted. Enter the value parameter within brackets ([ ]). The value parameter represents the flatfile column index.

The following sample XML code illustrates how you can use annotations in a flatfile to extract data.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<asset-converter xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<datasource type="flatfile">
    <connection-properties>
        <property filename="main" filepath="MainFile.txt" key="1"
        column_name="true"/>
        <property filename="system" filepath="SystemFile.txt" key="1"
        column_name="true"/>
    </connection-properties>
        ...
<asset file="main" translator="ACBsFmt" version="1_0">
    <general file="main">
        <host_name value="[4]"/>

        Note: 4 specifies it is mapped to the fourth column in the main file.
        <default_hostname value="[4]"/>
        <default_address value="[3]"/>
        <default_mac value="[2]"/>
    </general>
    ...
</asset>
</asset-converter>
```

Configure the Mapping File

You can configure the datasource, connection properties, and processing information for a mapping file by entering the following information. This mapping file structure is common across the datasource types used by the CA Asset Converter. Only the mapping expressions and connection properties change from one datasource type to another.

Note: All XML files you create, and the sample XML code used in this guide must be well-formed and comply with XML and XSD standards.

<datasource>

Specify the datasource for the mapping file. For example, DataBase or database.

<processing-info>

(Optional) Specify the following tags for processing purpose only. The tags will not be part of the asset XML file.

max-assets-to-process

Limit to retrieve assets less than or equal to the value specified by this tag. Use this tag and enter a value when creating the mapping file to verify whether a mapping file generates the correct asset XML file. For example, max-assets-to-process to 1, 2, 3, and so on. If you do not use this tag, the CA Asset Converter generates all assets in the datasource.

encoding-type

(Flatfiles only) Type of encoding of the input files. FlatFile supports only UTF-8 , ANSI, and UNICODE formats. If you do not specify an encoding type in the input configuration file, the CA Asset Converter uses the UTF-8 format.

<output-settings>

If you do not specify the output settings, the default values are used. You can use the following tags:

dest-dir

Path of the directory where the output XML files are stored. If the directory does not exist, it is automatically created. The default destination directory is AssetConverterAssets in the installation directory.

output-filename

Name of the final asset XML files. For example, *<output-filename value="asset"/>*. In this example, the generated files are named asset0.xml, asset1.xml, and so on. You can also use the hostname as the generated asset file name. For example, *<output-filename host-name="true"/>*. If the host-name attribute in the output-filename tag is true, the hostname is the file name, because the hostname is unique. If it is not unique, the previous file with the same hostname is overwritten.

- If the host-name attribute is false, the attribute value is the filename and is appended by a unique number.
- If the host-name attribute is false and the value attribute is either empty, the default filename is asset and is appended by a unique number.

<ftp-details>

(Flatfiles only) If the source file is located on a File Transfer Protocol (FTP) site, specify the following FTP details:

is-ftp-site

Indicates if the source file is located on an FTP site. Set to true.

ftp-url

The location of the FTP site.

ftp-port

The port number used for FTP access.

ftp-username

The user name to log in to the FTP site.

ftp-password

The password to log in to the FTP site.

CA MDB

Define the connection properties (see page 18) for a mapping file to connect to the MDB.

Example: Configuration Section of a Typical Mapping File

The following sample XML code illustrates the configuration section of a typical mapping file.

```
<?xml version="1.0" encoding="UTF-8"?>
<asset-converter>
    <datasource type=""> (a valid datasource type: DataBase OR FlatFile)
    </datasource>
    <processing-info>
        <max-assets-to-process value="10"/>
    </processing-info>
    <output-settings>
        <dest-dir value="Relative/Absolute paths"/>
        <output-filename value="asset" host-name="false"/>
    </output-settings>
    CA MDB
        <mdb-type>10<mdb-type>
        <mdb-connection-properties>
        <mdb-property name="server">servername</mdb-property>
        <mdb-property name="database">databasename</mdb-property>
        <mdb-property name="username">username</mdb-property>
        <mdb-property name="port">portnumber</mdb-property>
        <mdb-property name="password">password</mdb-property>
        </mdb-connection-properties>
    </mdb>
<asset-converter>
```

Asset Mapping Structure

The following sample XML code is an example of a typical representation of an asset mapping structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<asset-converter>
    <datasource type="">
        <connection-properties>
        </connection-properties>
    </datasource>
<!-- Specifies the parameters for processing -->
    <processing-info>
        <max-assets-to-process value=""/>
    </processing-info>
<!-- Specifies the output settings -->
    <output-settings>
        <dest-dir value=""/>
        <output-filename value=""/>
    </output-settings>
    CA MDB
        <mdb-type>10</mdb-type>
        <mdb-connection-properties>
        </mdb-connection-properties>
    </mdb>
<!-- data mapping -->
    <asset translator="ACBsFmt" version="1_0">
        <general >
                <host_name  value=""/>
                <default_hostname  value=""/>
                <default_address value=""/>
                <default_mac value=""/>
        </general>
        <hardware>
                <group name="GeneralInventory">
                        <attribute name="" value=""/>
                <group name="">
                </group>
                .
                .
                .
                </group>
                <group name="AdditionalInventory">
                     <attribute name="" value=""/>
                .
                .
                .
                <group name="">
                </group>
```

```
                .
                .
                .
                </group>
        </hardware>
        <software>
                <package name="" >
                        <attribute name="" value=""/>
                .
                .
                .
        </software>
    </asset>
</asset-converter>
```

<asset> tag

The asset tag contains the following tags:

- general
- hardware
- software

<general> tag

The general tag contains the following tags to provide general information about the assets:

Important! The first four tags (host_name, default_mac, default_hostname, and default_address) are required for the CA Asset Converter to load the XML files correctly. The required attributes cannot contain hard-coded values.

- host_name
- default_mac
- default_hostname
- default_address
- vendor
- serial_number
- asset_tag
- host_key
- class_id
- default_subnet_mask

- collect_time
- trustlevel
- origin

Note: All XML files you create, and the sample XML code used in this guide must be well-formed and comply with XML and XSD standards.

Example: General Tag (Database)

The following sample XML code illustrates how to use the general tag for a database.

```
<asset query="select AssetID from Computer" translator="ACBsFmt" version="1_0">
    <general query="select c.hostname,c.IPAddress, c.MAC_Address from Computer c
    where c.AssetID = {asset.AssetID}">
        <host_name  value="{asset:general.hostname}"/>
        <default_hostname value="{asset:general.hostname}"/>
        <default_address value="{asset:general.IPAddress}"/>
        <default_mac value="{asset:general.MAC_Address}"/>
        </general>
       ...
</asset>
```

Example: General Tag (Flatfile)

The following sample XML code illustrates how to use the general tag for a flatfile.

```
<asset file="main" translator="ACBsFmt" version="1_0">
    <general file="main">
        <host_name value="[4]"/> (Note: The 4 specifies it is mapped to the 4th column
        in the main file)
        <default_hostname value="[4]"/>
        <default_address value="[3]"/>
        <default_mac value="[2]"/>
    </general>
          ...
</asset>
```

<hardware> tag

The hardware tag contains the following groups:

- GeneralInventory. (Required). This group can contain multiple <attribute> tags followed by [set the product group or family] tags. The [set the product group or family] tags contain information about individual hardware components of the asset.
- AdditionalInventory. (Optional). You can specify additional attributes or groups to the asset.

Example: Hardware Tag (Database)

The following sample XML code illustrates how to use the hardware tag for a database.

```
<asset query="select AssetID from Computer" translator="ACBsFmt" version="1_0">
    <general>
    ...
    </general>
    <hardware>
        <group name="GeneralInventory">
        <group name="System" query="select p.ProcCount from Computer c where
        c.AssetID = {asset.AssetID}">
        <attribute name="No. of Processors" type="string"
        value="{group(name='System').ProcCount}"/>
        </group>
        </group>
    </hardware>
    ...
</asset>
```

Example: Hardware Tag (Flatfile)

The following sample XML code illustrates how to use the hardware tag for a flatfile.

```
<hardware>
    <group name="GeneralInventory" file="">
    <group name="System" file="system">
        <attribute name="Model" type="string" value="[2]"/>
        <attribute name="Type" type="string" value="[3]"/>
        <attribute name="Vendor" type="string" value="[4]"/>
    </group>
    <group name="Network" file="main">
        <attribute name="Computer Name" type="string" value="[4]"/>
        <attribute name="IP Address" type="string" value="[3]">
    </group>
    ...
</hardware>
```

Example: Additional Inventory Tag

The following sample XML code illustrates how to use the additional inventory tag.

```
<asset>
...
<hardware>
    <group name="GeneralInventory">
    ...
    </group>
    <group name="AdditionalInventory">
        <attribute name="SourceVendor" value="ThirdPartyTool"/>
    </group>
</hardware>
</asset>
```

<software> tag

The software tag contains information about the software packages installed in the asset.

Example: Software Tag (Database)

The following sample XML code illustrates how to use the software tag for a database.

```
<asset query="select AssetID from Computer" translator="ACBsFmt" version="1_0">
...
<hardware>
    <group name="GeneralInventory">
    ...
    </group>
        <group name="AdditionalInventory">
        ...
        </group>
</hardware>
<software>
    <package name="{asset:software:package.title}" query="select
    title,version,vendor from softwareTable where AssetID={asset.AssetID}">
        <attribute name="Ver" type="string"
        value="{asset:software:package.version}"/>
        <attribute name="Pub" type="string"
        value="{asset:software:package.vendor}"/>
    </package>
    </software>
</asset>
```

Example: Software Tag (Flatfile)

The following sample XML code illustrates how to use the software tag for a flatfile.

```
<software>
    <package name="Advanced Network Diagramming" file="system">
        <attribute name="Ver" type="string" value=[1]/>
        <attribute name="Pub" type="string" value=[2]/>
        ...
    </package>
</software>
```

Types and Subtypes

The following table includes the valid types and subtypes that you can specify for the attributes.

Type	Subtype	Description
Boolean		Boolean values can be displayed based on the following subtypes:
	TrueFalse	True or False
	YesNo	Yes or No
	OnOff	On or Off
	SupportedUnsupported	Supported or Not Supported
	ActiveNotactive	Active or Not Active
	OkError	Ok or Error
	PresentNotpresent	Present or Not Present
int32 & int64		Numerical values can be displayed based on the following subtypes:
	Separation	Thousands separated, that is, 1,000,000
	Normal	No separation
	K	Number is divided by 1024 before display
	M	Number is divided by 1024^2 before display
	G	Number is divided by 1024^3 before display

Type	Subtype	Description
	T	Number is divided by 1024^4 before display
	kilo	Number is divided by 1000 before display
	mega	Number is divided by 1e6 before display
	giga	Number is divided by 1e9 before display
	milli	Number is multiplied by 1e3 before display
	micro	Number is multiplied by 1e6 before display
	nano	Number is multiplied by 1e9 before display
	hex	Numbers shown as hex
	time	Displayed as a date-time
	time interval	Displayed as a duration
	bytes	User interface decides to display as KB, MB, GB, or TB
Float		Float values can be displayed based on the following subtypes:
	Auto	Auto Format
	placesXX	Show to XX decimal places
String		No subtypes for string

Extract Asset Data

You can use the CA Asset Converter to extract asset data from a selected database or flatfile.

Important! The CA Asset Converter must have exclusive access to open the flatfile. If the CA Asset Converter cannot open the file, for example, if the file is being accessed by another instance of the product, you receive an error message.

To extract asset data

1. Click Start, Programs, CA, Asset Converter, Asset Converter Command Prompt.

   A Command Prompt window opens.

2. Enter the following command:

   ```
   ac -f mapping file name -t tenant number
   ```

   -t

   (Optional) Use the CA Asset Converter in multi-tenant mode.

   tenant number

   (Optional) A unique number to identify a tenant.

   The asset extraction and transformation process starts.

View the Log Files

View the CA Asset Converter log files to see details, status, and error messages for the asset conversion process. A new log file is created the first-time the CA Asset Converter processes information, and the log file is overwritten on subsequent processes. A log folder containing the log file is available in the same location where the CA Asset Converter is installed.

Chapter 6: Processing Inventory Information

This section contains the following topics:

Introduction

The Asset Collector processes inventory information from various third-party collection tools and provides the following functionality:

- Tenancy collection (see page 38)
- Rules for processing inventory files (see page 43)
- Origin to trust level mapping (see page 45)
- Reject inventory files having a future collect time (see page 46)
- Configurable post-processing actions (see page 47)
- Auditing of Asset Collector actions (see page 48)

The Asset Collector captures details of assets, users, and their associated inventory. You can track the origin and trustworthiness of this inventory, giving you greater control and management over your assets. The Asset Collector collects the hardware and software inventory information from well-formed inventory files. You can create inventory information for any device or user. You can use this inventory information to perform asset management functions in CA APM.

Tenancy Collection

Use *tenancy* to manage the asset information collected from various sources within the same MDB. The collected asset information is imported into the MDB so the tenancy membership of the asset is maintained and managed within the same MDB.

CA Client Automation is not multi-tenant capable, but the product can collect external inventory files and store any tenant information for use by other multi-tenant capable CA products such as CA APM.

The Asset Collector uses the collection folders to receive inventory files. You can configure the collection folders to associate tenants with individual collection folders.

The tenancies are defined in the *ca_tenant* MDB table. Defining a tenancy on a collection folder lets the engine populate a new column named *tenant_id* on the *ca_asset* table in the MDB. The column *tenant_number* from the ca_tenant table is used to configure the Asset Collector.

Note: CA Client Automation cannot populate the ca_tenant table, although other CA products such as CA Service Desk Manager can populate it. So, when you define tenants with CA Service Desk Manager or another CA product, specify a tenancy number for each tenant. The Asset Collector uses this tenancy number to differentiate between tenants.

Enable the Database for Tenancy Collection

By default, the CA Client Automation database is not configured to perform tenancy collection. Enable a number of database triggers in CA Client Automation for the MDB to maintain the tenancy columns in the database.

Execute the following statement to enable the triggers:

Oracle

Execute this statement as the mdbadmin on Oracle so the tables and procedures for the mdbadmin are available. You cannot execute this statement as the ca_itrm user because that user does not have access to the required tables and procedures.

```
execute sp_enableTenantTriggers(1);

commit;
```

Microsoft SQL Server

Execute this statement in the MDB namespace. Issue the command *use mdb;* if your session is not already in the MDB namespace.

```
exec sp_enableTenantTriggers 1
```

Configure Tenancy Collection

You can manage the asset information collected from various sources within the same MDB by using CA Client Automation to configure the following in the Asset Collector configuration folders:

- Tenancy for each collection folder
- Collection folders without specifying tenants
- Multiple collection folders for a single tenant

Note: You cannot configure one collection folder for multiple tenants.

To configure collection folders for tenancy collection

1. Open the DSM Explorer and navigate to the Control Panel, Configuration, Configuration Policy node.
2. Right-click a sealed policy and select Un-Seal.

   The policy is unsealed.
3. Expand the unsealed policy, and navigate to DSM, Scalability Server, Asset Collector.

   The Asset Collector policies appear on the Asset Collector pane.
4. Double-click Collection Folders.

   The Modify Setting dialog appears.
5. For each row, define a collection folder and click OK.

   Note: The tenancy number column is optional. A value specified in this column must match with an entry in the ca_tenancy column of the ca_tenancy table in the MDB.
6. Right-click the policy node and select Seal from the menu.

   The policy is sealed.
7. Drag-and-drop the policy onto the scalability server in the All Computers folder.

   The policy is applied to the scalability server.

Data Synchronization from an MDB to a Separate Target MDB (SQL Bridge and Oracle Bridge)

In some implementations, you want CA Technologies products such as CA Service Desk Manager and CA APM to use management databases (MDB) separate or different from the database used by the DSM Manager. However, in many aspects of their asset management tasks, these CA Technologies products rely on CA Client Automation data.

Therefore, CA Client Automation provides manager features that support and synchronize data discovered by CA Client Automation to a separate MDB which may be based on SQL Server (SQL Bridge) or Oracle (Oracle Bridge). The synchronization features synchronize CA Client Automation assets and inventory data that are collected in a SQL Server MDB on the DSM domain or enterprise manager on Windows with the appropriate data in the target SQL Server or Oracle MDB.

How Tenancy Collection Works (SQL Bridge and Oracle Bridge)

The SQL Bridge and Oracle Bridge synchronization replicates asset and inventory data from the CA Client Automation database to the database used by another CA Technologies product, such as CA APM or CA Service Desk Manager. This asset and inventory data can include tenancy data. CA Client Automation can collect tenancy data and make the data available for other products. However, CA Client Automation cannot populate the ca_tenant database table, which other products use to define tenancies.

Therefore, if you are using SQL Bridge and Oracle Bridge synchronization, first synchronize the ca_tenant table in your product with the CA Client Automation database before tenancy data can be available for use in asset collection. Tenancy collection with SQL Bridge and Oracle Bridge synchronization uses the following general process:

- The administrator configures the tenants in the product (for example, CA APM or CA Service Desk Manager).
- The administrator synchronizes tenant information between the ca_tenant table and the CA Client Automation database.
- The SQL Bridge and Oracle Bridge synchronization supplies asset and inventory data (including tenant data) for use in asset collection.

Synchronize Tenant Information (SQL Bridge and Oracle Bridge)

If you have installed CA Service Desk Manager or CA APM on an MDB based on the SQL Bridge or Oracle Bridge, you can synchronize tenant information in the ca_tenant table. CA Client Automation cannot populate the ca_tenant table, but other CA Technologies products such as CA Service Desk Manager and CA APM can populate the table.

Note: When you synchronize the ca_tenant table, the synchronization happens from the MDB used by CA Service Desk Manager or CA APM to the MDB used by CA Client Automation.

To synchronize the tenant information from the SQL or Oracle Bridge

1. Configure the ca_tenant table on the MDB used by CA Service Desk Manager or CA APM.
2. Open the DSM Explorer and navigate to the Control Panel, Engines, All Engines node.
3. Right-click the engine to perform the tenant synchronization in the database, and select Add New Task.

   The New Task Wizard appears.
4. Select the Task type as *Database Synchronization* and follow the on-screen instructions.

   The database synchronization job is created and the ca_tenant table is synchronized as scheduled.

Scalability Server Configuration for Asset Tenant Numbers

The Asset Collector registers collected assets through a scalability server. If the Asset Collector is configured to register with a specific tenant, the collected assets are assigned the corresponding tenant number. However, if the Asset Collector is not registering with a tenant, but instead with the scalability server through which it is reporting, the Asset Collector registration uses the tenant configured at the scalability server.

Configure the tenants at the scalability server by associating a tenant with a CA Client Automation agent and defining a tenant number for each scalability server within your enterprise. Specify a tenant number if you have other CA Technologies products such as CA Service Desk Manager which require tenant classification for a CA Client Automation agent. You can associate a tenant with a CA Client Automation agent by specifying a tenant number in the configuration policy of the scalability server.

A scalability server can support only a single tenant. If you want to collect agents for different tenant, use a different scalability server for each tenant. The tenant numbers are defined in the ca_tenant table by CA Service Desk Manager or CA Client Automation. The tenant number you configure a scalability server to use must exist in the ca_tenant table.

Configure a Tenant Number on the Scalability Server

You configure the tenant number to be used for assets that are registered through a scalability server and is applied only when the tenant number is not provided by the Asset Collector.

To configure a tenant number on the scalability server

1. Open the DSM Explorer and navigate to the Control Panel, Configuration, Configuration Policy node.
2. Right-click a sealed policy and select Un-Seal.

   The policy is unsealed.
3. Expand the unsealed policy, and navigate to DSM, Scalability Server, Common Server.

   The Common Server policies appear on the Common Server pane.
4. Double-click Tenant Number.

   The Setting Properties dialog appears.

5. Enter a tenant number and click OK.

   Note: The tenant number that is applied must match a tenant number in the ca_tenant table.

6. Right-click the policy node and select Seal from the menu.

   The policy is sealed.

7. Drag-and-drop the policy onto the scalability server in the All Computers folder.

   The policy is applied to the scalability server.

Rules for Processing Inventory Files

Use CA Client Automation to specify rules for processing the inventory files collected from multiple tenants. Base the rules on one of the two attributes (trust level and collection time) of the inventory files.

The following modes of operation are supported:

Trust Enabled Mode (TRUE)

Process the inventory file based on the trust level.

An inventory file with a trust level equal to or greater than the trust level of the previous inventory file propagates to the scalability server.

Trust Disabled Mode (FALSE)

Process the inventory file based on the collect time.

An inventory file with a collect time higher than the collect time of the previous inventory file propagates to the scalability server.

To prevent resubmission of inventory records taken on the same day, define the s*ame day window* configuration in seconds. When you define the configuration in this way, any inventory file having a collection time within the same day window is not processed.

If you set the same day window to zero, the check is not performed, and all inventories with a subsequent collection time are processed.

Note: If you do not want to define processing rules for every tenant, define default-processing rules for the following configurations:

- Every tenant that does not have any rule configured
- Inventory files without a tenancy number in the collection folder

Configure Rules for Processing Inventory Files

Use CA Client Automation to specify rules for processing the inventory files collected from multiple tenants. To accept or reject an inventory file, configure rules based on trust level or collection time.

To configure rules for processing inventory files

1. Open the DSM Explorer and navigate to the Control Panel, Configuration, Configuration Policy node.
2. Right-click a sealed policy and select Un-Seal.

   The policy is unsealed.
3. Expand the unsealed policy, and navigate to DSM, Scalability Server, Asset Collector.

   The Asset Collector policies appear on the Asset Collector pane.
4. Double-click Processing Rules.

   The Modify Setting dialog appears.
5. Complete the fields in the dialog and click OK.

   Note: Define only one set of processing rules for each tenant.

   The following fields need further explanation:

   Tenant Number

   Specify the tenant number. This number must match a tenant number defined in the collection folders table, and therefore the tenant_number column of the ca_tenant table.

   Trust Mode

   Specify if trust mode or collection time is used to process the inventory files. Set this value to TRUE to use trust level or FALSE to use collection time to process the inventory files.

   Same Day Window

   Specify the collection time window in seconds. Any inventory file that has a collection time within the same day window is not processed. To disable the same day window, set the value to zero.
6. Right-click the policy node and select Seal from the menu.

   The policy is sealed.

7. Drag-and-drop the policy onto the scalability server in the All Computers folder.

   The policy is applied to the scalability server.

Note: If you do not want to define processing rules for each tenant, or if you have configuration folders without a tenant number, define the following processing rules in the Asset Collector configuration section:

- Processing Rules: Default Trust Mode
- Processing Rules: Default Same Day Window

Defining the default rules results in the same behavior as the tenant processing rules, but are applied when the specified tenant does not have a rule defined, or the asset submitted does not have an associated tenant.

Map Origin to Trust Level

You can use the Origin to Trust level mapping in CA Client Automation to define the source and trustworthiness of an asset. Defining a trust level is useful when you are collecting inventory information from multiple origins. The mapping is used when the collected asset file does not have a trust level defined.

To configure origin to trust level mapping

1. Open the DSM Explorer and navigate to the Control Panel, Configuration, Configuration Policy node.
2. Right-click a sealed policy and select Un-Seal.

   The policy is unsealed.
3. Expand the unsealed policy, and navigate to DSM, Scalability Server, Asset Collector.

   The Asset Collector policies appear on the Asset Collector pane.
4. Double-click Origin to Trust Mapping.

   The Modify Setting dialog appears.
5. For each origin, define a trust level and click OK.

   Note: Do not define multiple trust levels for the same origin. However, you can use the same trust level for multiple origins.
6. Right-click the policy node and select Seal from the menu.

   The policy is sealed.
7. Drag-and-drop the policy onto the scalability server in the All Computers folder.

   The policy is applied to the scalability server.

Reject Inventory Files Having a Future Collect Time

You can use CA Client Automation to configure the Asset Collector to accept only the inventory files that have a valid time stamp in the *xml inventory unsigned files (.xiu)*, and reject inventory files that have a collect time in the future. Configure same day tolerance to define *future date* to assist processing of inventory files from different time zones.

To configure the rejection of inventory files having a future collect time

1. Open the DSM Explorer and navigate to the Control Panel, Configuration, Configuration Policy node.
2. Right-click a sealed policy and select Un-Seal.

   The policy is unsealed.
3. Expand the unsealed policy, and navigate to DSM, Scalability Server, Asset Collector.

   The Asset Collector policies appear on the Asset Collector pane.
4. Modify the following configuration parameters:

   Collect Time: Allow inventory without a collect time

   Specify whether inventory files without a collect time are allowed. Set this value to TRUE to allow inventory files without a collect time in the xml.

   Collect Time: Future Date Tolerance

   Define the tolerance, in seconds, that is applied to the current time to define a future date. Any inventory file having a collect time in the future will be verified against the future data.

   Collect Time: Reject Future Files

   Specify whether to reject inventory files with a collection time exceeding future date. Set this value to TRUE to reject files that have collection time beyond the future date tolerance.
5. Right-click the policy node and select Seal from the menu.

   The policy is sealed.
6. Drag-and-drop the policy onto the scalability server in the All Computers folder.

   The policy is applied to the scalability server.

Configure Post-Processing Actions

When the Asset Collector processes inventory files, the following results are possible:

- Inventory file is accepted
- Inventory file is rejected
- Inventory file contains an error

You can use CA Client Automation to define post-processing actions for the previous events.

- If the inventory file is rejected or it contains an error, you can configure the Asset Collector to delete the file, copy the file to the output folder, or rename the file with a .error extension.
- If the inventory file is accepted, you can configure the Asset Collector to delete the file or copy the file to the output folder.

To configure post processing actions

1. Open the DSM Explorer and navigate to the Control Panel, Configuration, Configuration Policy node.
2. Right-click a sealed policy and select Un-Seal.

   The policy is unsealed.
3. Expand the unsealed policy, and navigate to DSM, Scalability Server, Asset Collector.

   The Asset Collector policies appear on the Asset Collector pane.
4. Modify the following configuration parameters:

   Inventory File Rejected

   Specify the action on the rejected inventory file. Available values for actions include the following:

   - **0**. Delete the file.
   - **1**. Move the file to the other output folder.
   - **2**. Rename the file with a .error extension.

Inventory File Processed

Specify the action on the processed inventory file. Available values for actions include the following:

- **0**. Delete the file.
- **1**. Move the file to the output folder.

Inventory File Error

Specify the action on the inventory file that contains an error. Available values for actions include the following:

- **0**. Delete the file.
- **1**. Move the file to the output folder.
- **2**. Rename the file with a .error extension.

5. Right-click the policy node and select Seal from the menu.

   The policy is sealed.

6. Drag-and-drop the policy onto the scalability server in the All Computers folder.

   The policy is applied to the scalability server.

Configure Asset Collector MDB Audits

Generate audit information in the MDB for enhanced traceability and reporting. Use CA Client Automation to configure the Asset Collector to generate audit records. These records are written into the *CA_AC_AUDIT_LOG* table in the MDB.

Configure Asset Collector Auditing

The Asset Collector maintains an internal cache of audit events and sends them to the scalability server when certain thresholds for size or age are reached. You can use CA Client Automation to customize the threshold values to match your environment.

To configure Asset Collector auditing

1. Open the DSM Explorer and navigate to the Control Panel, Configuration, Configuration Policy node.
2. Right-click a sealed policy and select Un-Seal.

   The policy is unsealed.

3. Expand the unsealed policy.

   The policy expands.

4. Navigate to DSM, Scalability Server, Asset Collector.

   The Asset Collector policies appear on the Asset Collector pane.

5. Modify the following configuration parameters to match your environment:

   Audit Log: Max Age

   Define the maximum age, in seconds, that the audit log queue must reach before it sends the audit log to the scalability server for inclusion in the MDB.

   Audit Log: Wait Period

   Define the polling period, in seconds. The polling period is used to verify the audit log queue and age by the audit component.

   Audit Log: Max Queue Size

   Define the maximum number of items allowed in the audit log queue before it is sent to the scalability server for inclusion in the MDB.

6. Right-click the policy node and select Seal from the menu.

   The policy is sealed.

7. Drag-and-drop the policy onto the scalability server in the All Computers folder.

   The policy is applied to the scalability server.

Configure Asset Collector Auditing Events

You can use CA Client Automation to configure the events that generate an audit record from the configuration section of the Asset Collector.

To configure Asset Collector auditing events

1. Open the DSM Explorer and navigate to the Control Panel, Configuration, Configuration Policy node.
2. Right-click a sealed policy and select Un-Seal.

   The policy is unsealed.

3. Expand the unsealed policy.

   The policy expands.

4. Navigate to DSM, Scalability Server, Asset Collector, Events.

   The configurable audit events appear.

5. Configure the audit events to match your requirement as follows:

   Audit Accepted Assets

   Specify if an audit record is created for each successfully processed inventory file.

   Audit Reject Collect Time Future Time

   Specify if an audit record is created when an inventory file is rejected, because the collect time specified in the file appears to be a future time.

   Audit Reject Collect Time Older

   Specify if an audit record is created when an inventory file is rejected, because the collect time in the inventory file is older than a file already submitted for the same asset.

   Audit Reject Collect Time Same Day

   Specify if an audit record is created when an inventory file is rejected, because the collection time falls within the same day window of an asset already processed.

   Audit Reject Missing Values

   Specify if an audit record is created when an inventory file is rejected, because a key value in the file is missing.

6. Right-click the policy node and select Seal from the menu.

   The policy is sealed.

7. Drag-and-drop the policy onto the scalability server in the All Computers folder.

   The policy is applied to the scalability server.

Asset Collector Collection Audit Table

The Asset Collector collection audit items are written to the CA_AC_AUDIT_LOG table. This table has the following columns:

Name	Description
Asset Name	Defines the host name of the asset.
MAC Address	Defines the MAC address, if available.
Scalability Server	Defines the scalability server to which the Asset Collector reports.
Origin	Defines the origin of the asset.

Name	Description
Tenant Number	Specifies the tenant identifier of the asset.
State	Specifies if the asset is accepted (0) or rejected (1).
Event Code	Specifies the Event Code for asset rejection.
Details	Indicates the reason for asset rejection.

The Event Code column of the CA_AC_AUDIT_LOG table includes the following possible event codes:

Event Code	Reason	Description
0	Not applicable	Specifies that the asset is accepted. The value is set to zero for asset accepted events.
1	Older collection time	Specifies that the asset is rejected because of a lower collection time than the last accepted inventory file for the same asset.
2	Lower trust level	Specifies that the asset is rejected because of a lower trust level than the last accepted inventory file for the same asset having trust mode enabled.
3	Same day as last submission	Specifies that the asset is rejected because its collection time falls within the same day tolerance of the last accepted inventory file for the same asset.
4	Future collect time	Specifies that the asset is rejected because its collection time represents a time in the future.
5	Missing values	Specifies that the asset is rejected because some key data fields are missing.

Audit Table Management

You can use CA Client Automation to manage the size of the CA_AC_AUDIT_LOG table by purging unnecessary records. You can purge unnecessary records by configuring values in the configuration section of the Asset Collector.

To configure purging of unnecessary records

1. Open the DSM Explorer and navigate to the Control Panel, Configuration, Configuration Policy node.
2. Right-click a sealed policy and select Un-Seal.

   The policy is unsealed.
3. Expand the unsealed policy, and navigate to DSM, Scalability Server, Asset Collector.

   The Asset Collector policies appear on the Asset Collector pane.
4. Modify the following parameters:

   Audit Purge Interval

   Specify the time, in days, before the audit records are purged. To prevent purging, set the value to zero.

   Audit Purge Max Age

   Specify the age, in days, after which the audit records are purged.
5. Right-click the policy node and select Seal from the menu.

   The policy is sealed.
6. Drag-and-drop the policy onto the domain manager in the All Computers folder.

   The policy is applied to the domain manager.

Chapter 7: Known Issues

This section contains the following topics:

Error Processing User-Defined Functions

Valid on all supported operating environments.

Symptom:

When processing a user-defined function or query, you receive the following message:

Error executing query.

Solution:

The error occurs because a column name that is referenced in the function does not exist in the database. Verify that the column name exists in the database when using user-defined functions. If the column name does not exist, verify that a valid alias is assigned to the column when writing the query or user-defined function.

Files Overwritten for Duplicate Hostnames

Valid on all supported operating environments.

Symptom:

If the hostname attribute of the output-filename tag in the configuration file is set to true, and the database or flatfile contains more than one record with the same hostname, the output XML files are overwritten and only one file is generated with the last record.

Solution:

Verify that the filenames for the generated asset XML files are unique.

Output Files Saved in the Default Directory

Valid on all supported operating environments.

Symptom:

The output asset XML files are not saved in the directory specified in the mapping file.

Solution:

This occurs if the name of the specified output directory is entered incorrectly in the mapping file. The output files are saved in the C:\Program Files\CA\CA Asset Converter\Assets directory. Verify that the directory name is entered correctly in the mapping file if you specify a different output directory.

The CA Asset Converter Cannot Retrieve an FTP File

Valid on all supported operating environments.

Symptom:

If the connection to an FTP server is lost when the CA Asset Converter is downloading files from an FTP location, the FTP files cannot be retrieved once the connection is reestablished.

Solution:

This occurs because the CA Asset Converter renames the files during the download and restores the file names only after the download is complete. Manually rename the files on the FTP server before continuing the download.