Centrify Server Suite 2017

Database Management Guide

February 2017

Centrify Corporation

Legal notice

This document and the software described in this document are furnished under and are subject to the terms of a license agreement or a non-disclosure agreement. Except as expressly set forth in such license agreement or non-disclosure agreement, Centrify Corporation provides this document and the software described in this document “as is” without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose. Some states do not allow disclaimers of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This document and the software described in this document may not be lent, sold, or given away without the prior written permission of Centrify Corporation, except as otherwise permitted by law. Except as expressly set forth in such license agreement or non-disclosure agreement, no part of this document or the software described in this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, or otherwise, without the prior written consent of Centrify Corporation. Some companies, names, and data in this document are used for illustration purposes and may not represent real companies, individuals, or data.

This document could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein. These changes may be incorporated in new editions of this document. Centrify Corporation may make improvements in or changes to the software described in this document at any time.

U.S. Government Restricted Rights: If the software and documentation are being acquired by or on behalf of the U.S. Government or by a U.S. Government prime contractor or subcontractor (at any tier), in accordance with 48 C.F.R. 227.7202-4 (for Department of Defense (DOD) acquisitions) and 48 C.F.R. 2.101 and 12.212 (for non-DOD acquisitions), the government’s rights in the software and documentation, including its rights to use, modify, reproduce, release, perform, display or disclose the software or documentation, will be subject in all respects to the commercial license rights and restrictions provided in the license agreement.

Centrify, DirectControl, DirectAuthorize, DirectAudit, DirectSecure, DirectControl Express, Centrify User Suite, and Centrify Server Suite are registered trademarks and Centrify for Mobile, Centrify for SaaS, Centrify for Mac, DirectManage, Centrify Express, DirectManage Express, Centrify Identity Platform, Centrify Identity Service, and Centrify Privilege Service are trademarks of Centrify Corporation in the United States and other countries. Microsoft, Active Directory, Windows, and Windows Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and other countries.

Centrify software is protected by U.S. Patents 7,591,005; 8,024,360; 8,321,523; 9,015,103 B2; 9,112,846; 9,197,670; and 9,378,391.

The names of any other companies and products mentioned in this document may be the trademarks or registered trademarks of their respective owners. Unless otherwise noted, all of the names used as examples of companies, organizations, domain names, people and events herein are fictitious. No association with any real company, organization, domain name, person, or event is intended or should be inferred.

Contents

About this guide

The *Database Management Guide* provides complete information for planning, installing, and maintaining Microsoft SQL Server databases required for the auditing features available in Centrify Server Suite, Enterprise Edition. Centrify software helps you comply with regulatory requirements and improve accountability by collecting detailed information about user activity on Linux, UNIX, and Windows computers. Auditing also enables you to monitor user activity for immediate analysis or to investigate specific incidents, such as application failures or security breaches.

Intended audience

This guide is intended for database administrators who are responsible for preparing and maintaining the databases required to store audit-related information. If you are a Centrify administrator, but not a Microsoft SQL Server administrator, you should review the information in this guide with your Microsoft SQL Server system administrator.

If you have an account that has been assigned the Microsoft SQL Server system administrator role, but you are new to Microsoft SQL Server, you should use this guide in conjunction with documentation provided by Microsoft for the version of SQL Server you are using.

Using this guide

This guide does not provided detailed instructions for configuring or managing Microsoft SQL Server. In addition, different versions of Microsoft SQL Server might operate differently than described in this guide. Every attempt has been made to give accurate, representative steps to guide you. However, you might find discrepancies between the steps in this guide and what you see in your own environment.

Conventions used in this guide

The following conventions are used in this guide:

- `Fixed-width` font is used for sample code, program names, program output, file names, and commands that you type at the command line. When *`italicized`*, this font indicates variables. Square brackets (`[ ]`) indicate optional command-line arguments.
- **Bold** text is used to emphasize commands or key command results; buttons or user interface text; and new terms.
- *Italics* are used for book titles and to emphasize specific words or terms. In fixed-width font, italics indicate variable values.

Finding information about Centrify software

Centrify provides extensive documentation targeted for specific audiences, functional roles, or topics of interest. If you want to learn more about Centrify and Centrify products and features, start by visiting the Centrify website. From the Centrify website, you can download data sheets and evaluation software, view video demonstrations and technical presentations about Centrify products, and get the latest news about upcoming events and webinars.

Contacting Centrify

You can contact Centrify by visiting our website, www.centrify.com. On the website, you can find information about Centrify office locations worldwide, email and phone numbers for contacting Centrify sales, and links for following Centrify on social media. If you have questions or comments, we look forward to hearing from you.

Getting customer support

If you have a Centrify account, click Support on the Centrify website to log on and access the Centrify Customer Support Portal. From the support portal, you can to search knowledge base articles, open and view support cases, connect with other Centrify users on customer

forums, and access additional resources—such as online training, how-to videos, and diagnostic tools.

Introduction to the databases used for auditing

Centrify Server Suite, Enterprise Edition provides an auditing infrastructure that enables your organization to capture and store session activity on audited computers. The auditing infrastructure also enables auditors to query and report on specific events, view all or selected session activity, change the status of reviewed sessions, and delete sessions that are no longer needed. The auditing infrastructure relies on two types of databases to store information: the **management database** and the **audit store database**.

If you are not familiar with the components and architecture of the auditing infrastructure, see the *Auditing with Centrify Server Suite Administrator's Guide*. That guide provides detailed information about how the components in the auditing infrastructure communicate with each other and how to configure and manage an audit installation.

Management databases store installation information

In most organizations, there is only one management database for each audit installation and it stores information about the components of the auditing infrastructure for that installation. For example, the management database stores information about which computers are audited, where the collector service is installed, and the scope (site or subnet) of each audit store.

In most cases, you create the management database when you create a new installation and update it whenever you add components to the auditing infrastructure. For example, if you enable auditing on additional computers or deploy the collector service on a new server, the change is recorded in the management database. Because the management database stores information about the auditing infrastructure and not audited sessions, it requires little to no maintenance over time.

Audit store databases store audited sessions

Like the management database, you create the first audit store database during deployment. However, unlike the management database, the audit store database stores the activity collected from audited computers. Over time, the audit store database would grow and become unmanageable. Therefore, most organizations periodically add a new audit store database to capture current activity. When the new audit store database becomes active, the previous audit store database can remain “attached” to provide access to stored information or be “detached” if access to the information stored in that database is no longer required.

The process of adding a new audit store database and changing the status of an existing audit store database from “active” to “attached” is called database rotation. Database rotation is the primary on-going administrative task to manage the auditing of user activity using Centrify software. There are, however, also steps to take during the planning phase and during deployment that apply specifically to preparing Microsoft SQL Server to support the auditing infrastructure.

The audit store database stores all of the activity collected on audited computers. When auditors or administrators want to review captured activity, they must be able to connect to the audit store database to retrieve it. Therefore, the audit store database must be accessible and the auditors and administrators who need to retrieve data from it must have the appropriate permissions to connect to the database instance, and to read and write data where applicable.

Using multiple databases for the audit store

Depending on the number of computers you are auditing, the level of detail you capture, and the length of time captured activity must be available for review, an audit store database can grow too large to manage effectively in a short period of time.

To prevent the audit store database from growing too large, you can split it into multiple databases. Only one database at a time can be the active—that is, the database currently receiving captured activity from an audited computer and its collector service.

However, because large databases are harder to manage and take longer to search than smaller ones and you cannot allow a single active database to grow indefinitely, you can change the active database to be an attached database—that is, available for searching and retrieving stored information but no longer receiving captured activity—and make a new database the currently active database.

Changing which database is active without interrupting the monitoring of audited computers is also referred to as *rolling* or *rotating* the database. By adding new databases and changing the audit store's active database to an attached database before it gets too large, you can optimize database performance and storage requirements.

Detaching and retiring audit store databases

All of the information stored in audit store databases that are attached to an audit installation is available for queries and reports and can be viewed in the session player. When the information in an attached database is no longer needed, you can detach the database from the installation. You cannot detach a database while it is the active database.

After a database that has been the active database is made an attached or detached database, it is considered a retired or decommissioned database. It cannot be used as the active database again.

Automating database rotation

Although you can do database rotation manually using Audit Manager, you might want to automate the process to perform it automatically on a regular schedule. You might also want to automate and schedule the detachment of old databases from the audit store. The API described in the reference enables you to write scripts to perform database rotation and attach or detach databases.

The software development kit (SDK) for auditing includes four sample scripts that you can modify to suit your purposes: two VBScript samples and two Power Shell samples. One pair of sample scripts (`db_rotation`) use default database settings. The second pair

(`db_rotation_sql_script`) let you customize the database scripts to set up the database and the server.

The sample scripts perform the following steps:

1. Create a new audit store database and attach it to an audit store.
2. Grant permission to the management database and collectors to access the newly created audit store database.
3. Make the newly created audit store database the active database.
4. Detach any audit store databases older than two years.
5. Publish the settings to Active Directory so that audited computers and collectors can look up the information.

Note that the sample scripts require the user to respond to informational messages at various points during execution. To make these scripts run without user interaction, remove or comment out all the `wscript.echo` commands in the script, or redirect the echo commands to STDOUT so that the scheduled task will not hang waiting for user input.

The following command adds the script `db_rotation.vbs` as a monthly scheduled task named `rolldb` to be run as user `domain_name\administrator`. By using `cscript.exe` to launch the script, it redirects output to STDOUT.

```
PS C:\Program Files\Centrify\DirectManage
Audit\SDK\Samples> schtasks.exe /Create /TN "rolldb" /TR
"cscript.exe
'C:\Program Files\Centrify\DirectManage Audit\SDK\Sample
s\db_rotation.vbs' DefaultInstallation DefaultAuditStore
sqlserver.domain_name.com subtest3" /RU
domain_name\administrator /SC Monthly /MO 1
```

The components of this command are as follows:

```
Schtasks.exe /Create /TN <Task_name> /TR <Task_Command> /RU <Run_As_User>
/SC <Reoccurance_rate> /MO <Reoccurance_increment>
```

where

- *Task_Name*: `rolldb`

- *Task_Command*: `cscript.exe 'C:\Program Files\Centrify\DirectManage Audit\SDK\Samples\db_rotation.vbs' DefaultInstallation DefaultAuditStore sqlserver.domain_name.com subtest3`
- *Run_as_user*: `domain_name\Administrator`
- *Reoccurance_rate*: Monthly
- *Reoccurance_increment*:1

The task command consists of the following elements:

<parser> '<install_path>\<VBS_script>' <Installation> <auditstore> <DB_Server> <DB_prefix>

where

- *parser*: `cscript.exe`
- *install_path*: `C:\Program Files\Centrify\DirectManage Audit\SDK\Samples`
- *VBS_script*: `db_rotation.vbs`
- *Installation*: `DefaultInstallation`
- *auditstore*: `DefaultAuditStore`
- *DB_Server*: sqlserver.domain_name.com
- *DB_prefix*: `subtest3`

The prefix is attached to a date stamp in the name of the newly created audit store database.

Audit-related object reference

This chapter describes the classes, methods, and properties in the Centrify software development kit for auditing. The following classes are used for managing auditing features and are defined in the `Centrify.DirectAudit.API` namespace:

Class	Description
`Account class`	Manages `Account` objects.
`Accounts class`	Enumerates `Account` objects.
`AuditServer class`	Manages `AuditServer` objects.
`AuditServers class`	Enumerates `AuditServer` objects.
`AuditStore class`	Manages `AuditStore` objects.
`AuditStoreDatabase class`	Manages `AuditStoreDatabase` objects.
`AuditStoreDatabases class`	Enumerates `AuditStoreDatabase` objects.
`Connection class`	Manages an auditing connection.
`Installation class`	Manages `Installation` objects.

Account class

Manages `Account` objects.

Syntax

```
class Account
```

Properties of the Account class

The `Account` class provides the following properties:

Property	Description
`IsSystemAccount property`	Gets a value indicating whether the account is a Windows system account.
`IsWindowsAccount property`	Gets a value indicating whether the account is a Windows domain account.
`UserName property`	Gets the user name of the account.

Description of the Account class

The accounts used for auditing include the management database account, audit store database account, and collector accounts. This class provides properties to retrieve information about an account.

See also

- Accounts class
- OutgoingAccount property
- AuditServerAccounts property
- CollectorAccounts property

IsSystemAccount property

Gets a value indicating whether the account is a Windows system account.

Syntax

```
bool IsSystemAccount{get;}
```

Return value

Returns `true` if the account is a Windows system account; otherwise, `false`.

Discussion of the IsSystemAccount property

When you attach a new database to the audit store, you must set the database to allow access by the management database account. Before you call the `AddAuditServerAccount` method, you should check to see if the management database account is a Windows system account because if it is, the `Account.UserName` property is not a Windows domain account name and therefore cannot be passed directly to the `AddAuditServerAccount` method.

Example

The following code sample first checks to make sure the management database account is not a system account. If it is not a system account, the sample calls the `AddAuditServerAccount` method. If the management database is a system account, the sample returns an error message.

```
...
' Grant permission to management database to access the
audit store database
SET objAuditServers = objInstallation.AuditServers
FOR EACH objAuditServer IN objAuditServers
    SET objAuditServerAccount =
objAuditServer.OutgoingAccount
    IF NOT objAuditServerAccount.IsSystemAccount THEN
        objAuditStoreDatabase.AddAuditServerAccount
objAuditServerAccount.UserName, & _
  objAuditServerAccount.IsWindowsAccount
        wscript.echo "Added management database account
'" & objAuditServerAccount.UserName & "'."
    ELSE
        wscript.echo "Cannot add account for management
database '" & objAuditServer.Name & _
 & "' because the account '" &
objAuditServerAccount.UserName & _
 & "' is a system account."
        wscript.echo "NOTE: Please add allowed incoming
management database for '" & _
 & objAuditServer.Name & _
 & "' to the new audit store database in Audit Manager."
```

```
    END IF
```

See also

- IsWindowsAccount property
- AddAuditServerAccount method

IsWindowsAccount property

Gets a value indicating whether the account is a Windows domain account.

Syntax

```
bool IsWindowsAccount {get;}
```

Return value

Returns `true` if the account is a Windows domain account; `false` if the account is an SQL Server login account.

Discussion

The management database-to-audit store database connection and the collector-to-audit store connection can use either Windows authentication or SQL Server authentication.

Example

The following code sample illustrates using this property as an input parameter to the `AddAuditServerAccount` method:

```
...
'Add management database accounts for those management
databases running in
' system account; e.g. NT Authority/Network Service
'
DIM strAuditServerAccount
DIM isAuditServerWindowsAccount
isAuditServerWindowsAccount = true
strAuditServerAccount = "DOMAIN\MACHINE$"
objAuditStoreDatabase.AddAuditServerAccount strAuditServerAccount,
```

```
& _
 isAuditServerWindowsAccount
wscript.echo "Added management database account '" &
strAuditServerAccount & "'."
```

See also

- AddAuditServerAccount method
- AddCollectorAccount method

UserName property

Gets the user name of the account.

Syntax

```
string UserName {get;}
```

Return value

Returns the user name of the account.

Discussion

If the account is a Windows account, the user name is the Windows domain account name. If the account is an SQL Server login account, the user name is the SQL Server account name.

Example

The following code sample illustrates using this property as an input parameter to the `AddCollectorAccount` method:

```
...
' Copy Collector accounts from current active audit store
database
    SET objCollectorAccounts =
objActiveDatabase.CollectorAccounts
    FOR EACH objCollectorAccount IN objCollectorAccounts
        objAuditStoreDatabase.AddCollectorAccount
objCollectorAccount.UserName
```

```
        wscript.echo "Added Collector account '" &
objCollectorAccount.UserName & "'."
```

Accounts class

Enumerates `Account` objects.

Syntax

```
class Accounts
```

Discussion

The accounts used for auditing include the management database account, the audit store database account, and collector accounts. Use this class to enumerate a set of accounts.

Example

In the following code sample, the `CollectorAccounts property` property returns an `Accounts` object and a `FOR EACH—IN` statement is used to enumerate the collector accounts:

```
...
' Copy Collector accounts from current active audit store
database
    SET objCollectorAccounts =
objActiveDatabase.CollectorAccounts
    FOR EACH objCollectorAccount IN objCollectorAccounts
        objAuditStoreDatabase.AddCollectorAccount
objCollectorAccount.UserName
        wscript.echo "Added Collector account '" &
objCollectorAccount.UserName & "'."
```

See also

- Account class
- AuditServerAccounts property
- CollectorAccounts property

AuditServer class

Manages `AuditServer` objects.

Syntax

```
class AuditServer
```

Properties

The `AuditServer` class provides the following properties:

Property	Description
`DatabaseName property`	Gets the database name of the management database.
`Name property`	Gets the display name of the management database.
`OutgoingAccount property`	Gets the outgoing account of the management database.
`ServerName property`	Gets the Microsoft SQL Server instance name of the management database.

Discussion

An `AuditServer` object holds information about an management database that is part of the audit installation. The management database stores license information, audit roles, and information about the components of the auditing infrastructure, including the scope of each audit store and the active and attached audit store databases.

See also

- AuditServers class

DatabaseName property

Gets the database name of the management database.

Syntax

```
string DatabaseName {get;}
```

Return value

Returns the database name of the management database.

Discussion

The management database stores license information, audit roles, and information about the components of the auditing infrastructure, including the scope of each audit store and the active and attached audit store databases.

See also

- Name property
- ServerName property
- AuditStoreDatabase class

Name property

Gets the display name of the management database.

Syntax

```
string Name {get;}
```

Return value

Returns the display name of the management database.

Discussion

The management database display name is used in the Audit Manager console and must be unique in the installation. Note that this is not the management database instance name, which is the fully-qualified domain name of the management database, and is not necessarily the same as the management database name, which need not be unique in the installation.

See also

- DatabaseName property
- ServerName property

OutgoingAccount property

Gets the outgoing account of the management database.

Syntax

```
Account class OutgoingAccount {get;}
```

Return value

Returns the outgoing account of the management database.

Discussion

The user name of the outgoing account is the name by which the management database identifies itself when connecting to an audit store.

ServerName property

Gets the Microsoft SQL Server instance name of the management database.

Syntax

```
string ServerName {get;}
```

Return value

Returns the Microsoft SQL Server instance name of the management database.

Discussion

The Microsoft SQL Server instance name is the fully qualified domain name of the management database.

See also

- DatabaseName property
- Name property

AuditServers class

Enumerates `AuditServer` objects.

Syntax

```
class AuditServers
```

Discussion

In most cases, an audit installation includes only one management database.

See also

- AuditServer class

AuditStore class

Manages `AuditStore` objects.

Syntax

```
class AuditStore
```

Properties

The `AuditStore` class provides the following properties:

Property	Description
`ActiveDatabase property`	Gets the active audit store database.
`Databases property`	Gets the list of the audit store databases.
`Name property`	Gets the display name of the audit store.

Methods

The `AuditStore` class provides the following methods:

Method	Description
`AddDatabase method`	Creates a new audit store database and attaches the database to the audit store using default settings.
`AddDatabaseByScript method`	Creates a new audit store database and attaches the database to the audit store using custom settings specified in SQL scripts.
`AttachDatabase method`	Attaches an existing audit store database to the audit store.
`ChangeActiveDatabase method`	Changes which database is currently active in the audit store.

Method	Description
`DetachDatabase method`	Detaches a database from the audit store.
`GetDatabase method`	Retrieves the audit store database object given the database display name.

Discussion

An audit store can have multiple databases attached, but only one can be active at a time. This class allows you to manage the audit store, including attaching and detaching databases and specifying which database is active. To get information about the attached databases, use the AuditStoreDatabase class and AuditStoreDatabases class classes.

See also

- AuditStoreDatabase class

ActiveDatabase property

Gets the active audit store database.

Syntax

```
AuditStoreDatabase class ActiveDatabase {get;}
```

Return value

Returns the active audit store database.

Discussion

An audit store can have multiple databases attached, but only one can be active at a time.

See also

- Databases property

Databases property

Gets the list of the audit store databases.

Syntax

```
AuditStoreDatabases class Databases {get;}
```

Return value

Returns the list of the audit store databases.

Discussion

This property returns a list of all the databases attached to the audit store.

See also

- ActiveDatabase property

Name property

Gets the display name of the audit store.

Syntax

```
string Name {get;}
```

Return value

Returns the display name of the audit store.

Discussion

The display name of the audit store is used in the Audit Manager console. It is distinct from the display name of the active database.

See also

- Name property

AddDatabase method

Creates a new audit store database and attaches the database to the audit store using default settings.

Syntax

```
AuditStoreDatabase class AddDatabase(
      string name,
      string serverName,
      string database
)
```

Parameters

Return value

Returns the `AuditStoreDatabase` object of the new audit store database.

Errors

The `AddDatabase` method may throw one of the following exceptions:

- `Centrify.DirectAudit.Common.Logic.AuthenticationException` if you do not have permission to connect to the Microsoft SQL Server instance that hosts the management database or you do not have permission to connect to the Microsoft SQL Server instance of the audit store database to be created.
- `Centrify.DirectAudit.Common.Logic.ConnectDatabaseException` if you cannot connect to the Microsoft SQL Server instance

either because the instance is not running or does not allow remote connections.

- `Centrify.DirectAudit.Common.Logic.UnauthorizedException` if you do not have the Manage Database permission on the audit store or you do not have the SQL Server permission to create SQL Server databases on the Microsoft SQL Server instance.
- `Centrify.DirectAudit.Common.Logic.AlreadyExistsException` if the specified display name is already being used by another audit store database, or the specified database name is already being used by another database in the Microsoft SQL Server instance.

Discussion

Use this method to create a new audit store database and attach it to the audit store. To customize the database or attach an existing database to the audit store, use one of the methods listed in the “See also” section.

Example

The following code sample argument illustrates the use of `AuditStore.AddDatabase`:

```
...
strInstallationName = wscript.arguments.item(0)
strAuditStoreName = wscript.arguments.item(1)
strServerName = wscript.arguments.item(2)
strDatabaseName = wscript.arguments.item(3)

SET objAuditStoreDatabase =
objAuditStore.GetDatabase(strDatabaseName)

IF NOT objAuditStoreDatabase IS NOTHING THEN
    wscript.echo "Audit Store database '" &
strDatabaseName & "' already exists."
    wscript.quit
END IF
```

```
' Create a new audit store database and attach to the
audit store
SET objAuditStoreDatabase =
objAuditStore.AddDatabase(strDatabaseName, & _
strServerName, strDatabaseName)

IF objAuditStoreDatabase IS NOTHING THEN
    wscript.echo "Failed to add audit store database '" &
strDatabaseName & "'."
    wscript.quit
END IF
wscript.echo "Created and attached audit store database
'" & strDatabaseName & "'."
```

See also

- AddDatabaseByScript method
- AttachDatabase method
- ChangeActiveDatabase method

AddDatabaseByScript method

Creates a new audit store database and attaches the database to the audit store using custom settings specified in SQL scripts.

Syntax

```
AuditStoreDatabase class AddDatabaseByScript(
        string name,
        string serverName,
        string database,
        string scriptFile1,
        string scriptFile2
)
```

Parameters

Return value

Returns the `AuditStoreDatabase` object of the new audit store database.

Errors

The `AddDatabaseByScript` method may throw one of the following exceptions:

- `Centrify.DirectAudit.Common.Logic.AuthenticationException` if you do not have permission to connect to the Microsoft SQL Server instance that hosts the management database or you do not have permission to connect to the Microsoft SQL Server instance of the audit store database to be created.
- `Centrify.DirectAudit.Common.Logic.ConnectDatabaseException` if you cannot connect to the Microsoft SQL Server instance either because the instance is not running or does not allow remote connections.
- `Centrify.DirectAudit.Common.Logic.UnauthorizedException` if you do not have the Manage Database permission on the audit store or you do not have the SQL Server permission to create SQL Server databases on the Microsoft SQL Server instance.
- `Centrify.DirectAudit.Common.Logic.AlreadyExistsException` if the specified display name is already being used by another audit store database, or the specified database name is already being use by another database in the Microsoft SQL Server instance.

Discussion

The database name you specify in the `database` parameter is substituted for the keyword `#database` in the SQL script. To create a new database using standard settings or to attach an existing database to the audit store, use on the methods listed in the “See also” section.

Example

The following code sample illustrates using `AuditStore.AddDatabaseByScript` in a script:

```
...
' Create a new audit store database and attach to the
audit store
SET objAuditStoreDatabase =
objAuditStore.AddDatabaseByScript(strDatabaseName, & _
strServerName, strDatabaseName, strServerScriptFile,
strDatabaseScriptFile)

IF objAuditStoreDatabase IS NOTHING THEN
    wscript.echo "Failed to add audit store database '" &
strDatabaseName & "'."
    wscript.quit
END IF
wscript.echo "Created and attached audit store database
'" & strDatabaseName & "'."
```

See also

- AddDatabase method
- AttachDatabase method
- ChangeActiveDatabase method

AttachDatabase method

Attaches an existing audit store database to the audit store.

Syntax

```
AuditStoreDatabase class AttachDatabase(
        string name,
        string server,
        string database
)
```

Parameters

Return value

Returns the `AuditStoreDatabase` object of the attached audit store database.

Errors

The `AttachDatabase` method may throw one of the following exceptions:

- `Centrify.DirectAudit.Common.Logic.AuthenticationException` if you do not have permission to connect to the Microsoft SQL Server instance that hosts the management database or you do not have permission to connect to the Microsoft SQL Server instance of the audit store database to be created.
- `Centrify.DirectAudit.Common.Logic.ConnectDatabaseException` if you cannot connect to the Microsoft SQL Server instance either because the instance is not running or does not allow remote connections.
- `Centrify.DirectAudit.Common.Logic.UnauthorizedException` if you do not have the Manage Database permission on the audit store or you do not have the SQL Server permission to create SQL Server databases on the Microsoft SQL Server instance.
- `Centrify.DirectAudit.Common.Logic.AlreadyExistsException` if the specified display name is already being used by another audit store database, or the specified database name is already being use by another database in the Microsoft SQL Server instance.

Discussion

Use this method if you already have a database that you want to attach to the audit store. To create a new database and attach it to the audit store, use the `AddDatabase` or `AddDatabaseByScript` method instead.

Example

The following code sample illustrates using `AuditStore.AttachDatabase` in a script:

```
...
' Attach an audit store database to the audit store
SET objAuditStoreDatabase =
objAuditStore.AttachDatabase(strDatabaseName, & _
strServerName, strDatabaseName)

IF objAuditStoreDatabase IS NOTHING THEN
    wscript.echo "Failed to attach audit store database
'" & strDatabaseName & "'."
    wscript.quit
END IF
wscript.echo "Attached audit store database '" &
strDatabaseName & "'."
```

See also

- AddDatabase method
- AddDatabaseByScript method

ChangeActiveDatabase method

Changes which database is currently active in the audit store.

Syntax

```
void ChangeActiveDatabase(
 AuditStoreDatabase class database
)
```

Parameters

Errors

The `ChangeActiveDatabase` method may throw one of the following exceptions:

- `Centrify.DirectAudit.Common.Logic.AuthenticationException` if you do not have permission to connect to the Microsoft SQL Server instance that hosts the management database or you do not have permission to connect to the Microsoft SQL Server instance of the audit store database to be created.
- `Centrify.DirectAudit.Common.Logic.ConnectDatabaseException` if you cannot connect to the Microsoft SQL Server instance either because the instance is not running or does not allow remote connections.
- `Centrify.DirectAudit.Common.Logic.UnauthorizedException` if you do not have the Manage Database permission on the audit store or you do not have the SQL Server permission to create SQL Server databases on the Microsoft SQL Server instance.

Discussion

An audit store can have multiple databases attached, but only one can be active at a time. Once you have made a database inactive by calling this method, you cannot make it active again. You cannot detach the active database.

Example

The following code sample illustrates using `AuditStore.ChangeActiveDatabase` in a script:

```
' Change active Audit Store database
objAuditStore.ChangeActiveDatabase(objAuditStoreDatabase
)
wscript.echo "Changed active database to '" &
objAuditStore.ActiveDatabase.Name & "'."
```

See also

- IsActive property

DetachDatabase method

Detaches a database from the audit store.

Syntax

```
void DetachDatabase(
      AuditStoreDatabase class database
)
```

Parameters

Errors

The `DetachDatabase` method may throw one of the following exceptions:

- `Centrify.DirectAudit.Common.Logic.AuthenticationException` if you do not have permission to connect to the Microsoft SQL Server instance that hosts the management database or you do not have permission to connect to the Microsoft SQL Server instance of the audit store database to be created.
- `Centrify.DirectAudit.Common.Logic.ConnectDatabaseException` if you cannot connect to the Microsoft SQL Server instance either because the instance is not running or does not allow remote connections.
- `Centrify.DirectAudit.Common.Logic.UnauthorizedException` if you do not have the Manage Database permission on the audit store or you do not have the SQL Server permission to create SQL Server databases on the Microsoft SQL Server instance.

Discussion

An audit store can have multiple databases attached, but only one can be active at a time. You cannot detach the active database.

Example

The following code sample illustrates using `AuditStore.DetachDatabase` in a script:

```
...
' Detach any Audit Store databases older than 2 years
FOR EACH objDatabase IN objAuditStore.Databases
```

```
    IF DateDiff("d", today, objDatabase.ActiveEndTime) >
728 THEN
        objAuditStore.DetachDatabase(objDatabase)
       wscript.echo "Detached Audit Store database '" &
objDatabase.Name & "'."
    END IF
```

GetDatabase method

Retrieves the audit store database object given the database display name.

Syntax

```
AuditStoreDatabase class GetDatabase(
      string displayname
)
```

Parameters

Return value

Returns the `AuditStoreDatabase` object of the specified database.

Errors

The `GetDatabase` method may throw one of the following exceptions:

- `Centrify.DirectAudit.Common.Logic.AuthenticationException` if you do not have permission to connect to the Microsoft SQL Server instance that hosts the management database or you do not have permission to connect to the Microsoft SQL Server instance of the audit store database to be created.
- `Centrify.DirectAudit.Common.Logic.ConnectDatabaseException` if you cannot connect to the Microsoft SQL Server instance either because the instance is not running or does not allow remote connections.

Discussion

Use this method to obtain the audit store database object of any database attached to the audit store if you already have the audit store database display name.

Example

The following code sample illustrates using `AuditStore.GetDatabase` in a script:

```
...
today = Date
strDatabaseName = strDatabaseName & "-" & Year(today) &
"-" & Month(today) & _
& "-" & Day(today)

SET objAuditStoreDatabase =
objAuditStore.GetDatabase(strDatabaseName)

IF NOT objAuditStoreDatabase IS NOTHING THEN
    wscript.echo "Audit Store database '" &
strDatabaseName & "' already exists."
    wscript.quit
END IF
```

AuditStoreDatabase class

Manages `AuditStoreDatabase` objects.

Syntax

```
class AuditStoreDatabase
```

Properties

The `AuditStoreDatabase` class provides the following properties:

Property	Description
`ActiveEndTime property`	Gets the end time of a formerly active database.
`ActiveStartTime property`	Gets the start time of an active or formerly active database.
`AuditServerAccounts property`	Gets the list of management database accounts that are allowed to access this audit store.
`CollectorAccounts property`	Gets the list of collector accounts that are allowed to access this audit store.
`DatabaseName property`	Gets the audit store database name.
`IsActive property`	Indicates whether this database is the current active database in the audit store.
`IsRetired property`	Indicates whether this database was formerly the active database and is now retired.
`Name property`	Gets the display name of the audit store database.
`ServerName property`	Gets the Microsoft SQL Server instance name of the audit store database.

Methods

The `AuditStoreDatabase` class provides the following methods:

Method	Description
`AddAuditServerAccount method`	Adds a management database account to the list of accounts allowed to access this audit store database.
`AddCollectorAccount method`	Adds a collector account to the list of accounts allowed to access this audit store database.

Discussion

An audit store can have multiple databases attached, but only one can be active at a time. This class provides information about any attached database. You can also add an management database or collectors to the list of accounts allowed access to an audit store database. To get information about the audit store, use the AuditStore class class.

See also

- AuditStoreDatabases class
- AuditStore class

ActiveEndTime property

Gets the end time of a formerly active database.

Syntax

```
DateTime ActiveEndTime {get;}
```

Return value

Returns the end time of the database's active period. If the database was never active or is currently active, the return value is `System.DateTime.MinValue` (12:00:00 AM).

See also

- ActiveStartTime property
- IsRetired property

ActiveStartTime property

Gets the start time of an active or formerly active database.

Syntax

```
DateTime ActiveStartTime {get;}
```

Return value

Returns the start time of the database's active period. If the database was never active, the return value is `System.DateTime.MinValue` (12:00:00 AM).

See also

- ActiveEndTime property
- IsRetired property

AuditServerAccounts property

Gets the list of management database accounts that are allowed to access this audit store.

Syntax

```
Accounts class AuditServerAccounts {get;}
```

Return value

Returns the list of allowed incoming management database accounts.

Discussion

Although most audit installations include only one management database, it's possible to add more.

See also

- CollectorAccounts property
- AddAuditServerAccount method

CollectorAccounts property

Gets the list of collector accounts that are allowed to access this audit store.

Syntax

```
Accounts class CollectorAccounts {get;}
```

Return value

Returns the list of allowed incoming collector accounts.

See also

- AuditServerAccounts property
- AddCollectorAccount method

DatabaseName property

Gets the audit store database name.

Syntax

```
string DatabaseName {get;}
```

Return value

Returns the database name of the audit store database.

Discussion

An audit store can have multiple databases attached, but only one can be active at a time. This property returns the database name of the database.

To get information about the active database attached to the management database, use the AuditServer class class.

See also

- Name property
- ServerName property
- DatabaseName property

IsActive property

Indicates whether this database is the current active database in the audit store.

Syntax

```
Bool IsActive {get;}
```

Return value

Returns `true` if the database is the current active database in the audit store; otherwise, `false`.

Discussion

An audit store can have multiple databases attached, but only one can be active at a time.

See also

- ChangeActiveDatabase method
- IsRetired property

IsRetired property

Indicates whether this database was formerly the active database and is now retired.

Syntax

```
Bool IsRetired {get;}
```

Return value

Returns `true` if the database was formerly the active database for the audit store and is now retired; otherwise, `false`.

Discussion

An audit store can have multiple databases attached, but only one can be active at a time. Once a database has been retired, it cannot be made active again.

See also

- ChangeActiveDatabase method
- IsActive property

Name property

Gets the display name of the audit store database.

Syntax

```
string Name {get;}
```

Return value

The display name of the audit store database.

Discussion

The display name of the audit store database is the name used in the Audit Manager console when displaying information about the database.

Example

```
...
wscript.echo "Changed active database to '" &
objAuditStore.ActiveDatabase.Name & "'."
```

See also

- DatabaseName property
- ServerName property

ServerName property

Gets the Microsoft SQL Server instance name of the audit store database.

Syntax

```
string ServerName {get;}
```

Return value

Returns the Microsoft SQL Server instance name of the audit store database.

Discussion

The SQL Server instance name of the audit store database is the fully qualified domain name of the SQL Server to which the audit store database is attached.

See also

- DatabaseName property
- Name property

AddAuditServerAccount method

Adds a management database account to the list of accounts allowed to access this audit store database.

Syntax

```
void AddAuditServerAccount(
      string userName,
      bool isWindowsAccount
)
```

Parameters

Errors

The `AddAuditServerAccount` method may throw one of the following exceptions:

- `Centrify.DirectAudit.Common.Logic.AuthenticationException` if you do not have permission to connect to the Microsoft SQL Server instance or the management database.
- `Centrify.DirectAudit.Common.Logic.ConnectDatabaseException` if you cannot connect to the Microsoft SQL Server instance either because the Microsoft SQL Server instance is not running and does not allow remote connections.
- `Centrify.DirectAudit.Common.Logic.UnauthorizedException` if you do not have the Manage SQL Login permission on the audit store.

Discussion

When you attach a new database to the audit store, you must set the database to allow access by the management database account. If the management database account is a Windows system account, you must explicitly specify the Windows domain account name in the `username` parameter. For other Windows accounts and for SQL accounts, you can pass the management database's `Account.UserName` property to this method as the user name.

Example

The following code sample first checks each account to see if it's a Windows system account. If the installation does not use a system account, the code passes the `Account.UserName` property to the `AddAuditServerAccount` method as the user name. If the installation uses a system account, it passes the Windows domain account name instead.

```
...
' Grant permission to management database to access the
audit store database
SET objAuditServers = objInstallation.AuditServers
```

```
FOR EACH objAuditServer IN objAuditServers
SET objAuditServerAccount = objAuditServer.OutgoingAccount
IF NOT objAuditServerAccount.IsSystemAccount THEN
 objAuditStoreDatabase.AddAuditServerAccount & _
  objAuditServerAccount.UserName, & _
  objAuditServerAccount.IsWindowsAccount
        wscript.echo "Added management database account
'" & objAuditServerAccount.UserName & "'."
    ELSE
'Add management database accounts for those management
databases running in
' system account; e.g. NT Authority/Network Service
'
DIM strAuditServerAccount
DIM isAuditServerWindowsAccount
isAuditServerWindowsAccount = true
strAuditServerAccount = "DOMAIN\MACHINE$"
objAuditStoreDatabase.AddAuditServerAccount strAuditServerAccount,
& _
 isAuditServerWindowsAccount
wscript.echo "Added management database account '" &
strAuditServerAccount & "'."
END IF
NEXT
```

See also

- AddCollectorAccount method
- AuditServerAccounts property
- IsSystemAccount property
- IsWindowsAccount property
- UserName property

AddCollectorAccount method

Adds a collector account to the list of accounts allowed to access this audit store database.

Syntax

```
void AddCollectorAccount(
```

```
 string userName,
)
```

Parameters

Errors

The `AddCollectorAccount` method may throw one of the following exceptions:

- `Centrify.DirectAudit.Common.Logic.AuthenticationException` if you do not have permission to connect to the Microsoft SQL Server instance or the management database.
- `Centrify.DirectAudit.Common.Logic.ConnectDatabaseException` if you cannot connect to the Microsoft SQL Server instance either because the Microsoft SQL Server instance is not running and does not allow remote connections.
- `Centrify.DirectAudit.Common.Logic.UnauthorizedException` if you do not have the Manage SQL Login permission on the audit store.

Discussion

When you attach a new database to the audit store, you must set the database to allow access by each collector account that passes data to that audit store. You can pass the collector's `Account.UserName` property to this method as the user name.

Example

The following code sample illustrates using `AuditStoreDatabase.AddCollectorAccount` in a script:

```
...
' Copy Collector accounts from current active Audit Store
database
SET objCollectorAccounts =
objActiveDatabase.CollectorAccounts
FOR EACH objCollectorAccount IN objCollectorAccounts
objAuditStoreDatabase.AddCollectorAccount
objCollectorAccount.UserName
```

```
wscript.echo "Added Collector account '" &
objCollectorAccount.UserName & "'."
NEXT
```

See also

- AddAuditServerAccount method
- CollectorAccounts property

AuditStoreDatabases class

Enumerates `AuditStoreDatabase` objects.

Syntax

```
class AuditStoreDatabases
```

Example

In the following code sample, the `AuditStore.Databases` property returns an `AuditStoreDatabases` object and a `FOR EACH—IN` statement is used to enumerate the audit store databases:

```
...
' Detach any Audit Store databases older than 2 years
FOR EACH objDatabase IN objAuditStore.Databases
    IF DateDiff("d", today, objDatabase.ActiveEndTime) >
728 THEN
        objAuditStore.DetachDatabase(objDatabase)
        wscript.echo "Detached Audit Store database '" &
objDatabase.Name & "'."
    END IF
NEXT
```

See also

- AuditStoreDatabase class
- AuditStore class

Connection class

Manages an auditing connection.

Syntax

```
class Connection
```

Constructors

The `Connection` class provides the following overloaded constructor:

Constructor	Description
`Connection constructor`	Creates a `Connection` object.

Methods

The `Connection` class provides the following overloaded method:

Method	Description
`GetInstallation method`	Retrieves an audit installation by name or by management database connection.

Discussion

The Active Directory domain controller stores information about the audit installation, including the installation name and the management database being used by the installation. The `Connection` object provides a way to connect to an Active Directory domain controller and retrieve the installation information stored there.

See also

- Installation class

Connection constructor

Creates a `Connection` object.

Syntax

```
Connection()
Connection(string domainController)
```

Parameters

Specify the following parameter when needed:

Parameter	Description
dominController	The domain controller of the Active Directory domain to which you wish to connect in order to get information about the audit installation.

Discussion

The `Connection` object constructor is overloaded. Use the constructor without parameters to create a connection in the current domain. Use the second version of the constructor if you want to specify the Active Directory domain of the connection in order to administer an audit installation on an Active Directory domain other than the one to which your workstation is joined.

GetInstallation method

Retrieves an audit installation by name or by management database connection.

Syntax

```
Installation class GetInstallation(
      string installationName)

Installation class GetInstallation(
      string server,
      string database)
```

Parameters

Return value

Returns the `Installation` object found.

Errors

The `GetInstallation` method may throw the following exception:

- `Centrify.Cfw.DirectoryServices.ServerNotOperationalException` if the domain controller is not operational. Check to make sure you entered the correct domain name when you called the constructor for the `Connection` object.

Discussion

The `Connection.GetInstallation` method is overloaded to provide two ways to search for an installation: by the name of the installation, or by the management database that is part of the installation.

Example

The following code sample illustrates using `Connection.GetInstallation` in a script to get the `Installation` object for the audit installation in the current Active Directory domain. The `Installation` object is then used to get the name of the object store database:

```
...
SET objInstallation =
objConnection.GetInstallation(strInstallationName)
SET objAuditStore =
objInstallation.GetAuditStore(strAuditStoreName)
SET objAuditStoreDatabase =
objAuditStore.GetDatabase(strDatabaseName)
```

See also

- Installation class

Installation class

Manages `Installation` objects.

Syntax

```
class Installation
```

Properties

The `Installation` class provides the following properties:

Property	Description
`AuditServers property`	Gets the list of management databases in this installation.
`CurrentAuditServer property`	Gets the currently connected management database.
`Name property`	Gets the name of the audit installation.

Methods

The `Installation` class provides the following methods:

Method	Description
`GetAuditStore method`	Retrieves an audit store given its display name.
`Publish method`	Publishes installation information to Active Directory.

Discussion

An Installation object holds information about a specific audit installation. This class lets you retrieve information about an installation and publish changed information to the Active Directory domain controller so that it can be retrieved by the components of the installation.

See also

- GetInstallation method

AuditServers property

Gets the list of management databases in this installation.

Syntax

```
AuditServers class AuditServers {get;}
```

Return value

Returns the list of management databases in the installation.

Discussion

In most cases, an installation includes only one management database.

See also

- AuditServer class

CurrentAuditServer property

Gets the currently connected management database.

Syntax

```
AuditServer class CurrentAuditServer {get;}
```

Return value

Returns the connected management database.

Discussion

You can use the SQL Server instance name property of the management database object returned by this property as a

parameter value when you call the `Connection.GetInstallation (server,database)` method.

See also

- AuditServer class
- GetInstallation method

Name property

Gets the name of the audit installation.

Syntax

```
String Name {get;}
```

Return value

Returns the installation name.

Discussion

The audit installation is named when it is created and this name is not normally changed during the life of the installation.

GetAuditStore method

Retrieves an audit store given its display name.

Syntax

```
AuditStore class GetAuditStore(
 string Name
)
```

Parameters

Errors

The `GetAuditStore` method may throw one of the following exceptions:

- `Centrify.DirectAudit.Common.Logic.AuthenticationException` if you do not have permission to connect to the Microsoft SQL Server instance or the management database.
- `Centrify.DirectAudit.Common.Logic.ConnectDatabaseException` if you cannot connect to the Microsoft SQL Server instance either because the Microsoft SQL Server instance is not running and does not allow remote connections.

Example

The following code sample accepts the audit store display name as an argument when the script is executed, calls the `GetAuditStore` method to get the audit store, then attaches a new audit store database to the audit store:

```
...
strInstallationName = wscript.arguments.item(0)
strAuditStoreName = wscript.arguments.item(1)
strServerName = wscript.arguments.item(2)
strDatabaseName = wscript.arguments.item(3)

SET objConnection =
CreateObject("Centrify.DirectAudit.Connection")
SET objInstallation =
objConnection.GetInstallation(strInstallationName)
SET objAuditStore =
objInstallation.GetAuditStore(strAuditStoreName)
today = Date
strDatabaseName = strDatabaseName & "-" & Year(today) &
"-" & Month(today) &_
& "-" & Day(today)

SET objAuditStoreDatabase =
objAuditStore.GetDatabase(strDatabaseName)
```

```
' Create a new Audit Store database and attach to the
Audit Store
SET objAuditStoreDatabase =
objAuditStore.AddDatabase(strDatabaseName,
strServerName, strDatabaseName)
```

See also

- AuditStore class

Publish method

Publishes installation information to Active Directory.

Syntax

```
void Publish()
```

Errors

The `Publish` method may throw the following exception:

- `Centrify.DirectAudit.Common.Logic.DirectAuditException` if you do not have write permission for the installation's service connection point (SCP) object in Active Directory.

Discussion

Audit Manager publishes installation information to a service connection point (SCP) object in Active Directory so that audited computers and collectors can look up the information. For example, collectors publish which audit store they are part of so that once an agent determines which audit store is to receive its audit data, it can determine the list of collectors that service that audit store by querying Active Directory.

When you use the methods in the API to change settings in the installation, you must call the `Publish` method to write the new settings to the Active Directory domain controller so that other auditing components in the installation can find the new information.

Example

The following code sample illustrates using `Installation.Publish` in a script:

```
...
objInstallation.Publish
wscript.echo "Published settings to Active Directory."
```