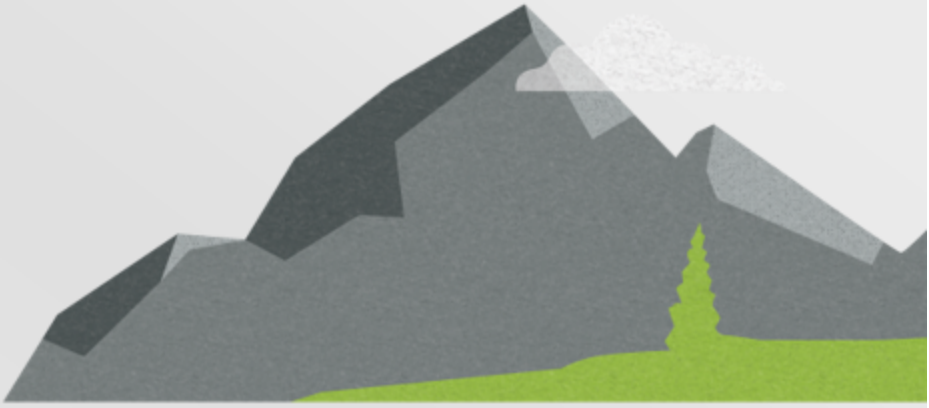
CARBONITE

Carbonite SQLMove

User's Guide



Notices

SQLMove User's Guide, version 1.3.0.2, Thursday, July 26, 2018

Contents

Chapter 1 SQLMove overview

SQLMove is a scripted solution, built on proven Carbonite Migrate technology, that automates identifying SQL Server databases, on a server called the source server, and synchronizes them to a target SQL Server. Using PowerShell, SQLMove coordinates Carbonite Migrate (using the Carbonite Migrate APIs) and SQL Server (using SQL Server Management Object (SMO)) to synchronize SQL-specific data from the source to the target. Once synchronization is complete, replication keeps the target up-to-date with any source changes. When you are ready, you can perform a live or test cutover.

SQLMove provides the following features and benefits.

- **Server modernization and upgrades**
  - **SQL upgrades**—Upgrade from an older SQL Server version to a newer SQL Server version, and the newer SQL Server version will automatically step the database through an upgrade process when attaching
  - **Operating system upgrades**—Upgrade from an older Windows version to a newer Windows version
- **Server migrations**
  - **Platform migrations**—Migrate between the same or different platforms including any combination of on-premise, cloud, physical, and virtual
  - **Hardware migrations**—Migrate from any hardware to any hardware
- **SQL Sever data realignment**—Migrate to new storage on the same SQL Server or change the database file location and name during a migration or upgrade process
- **SQL Server consolidation**—Consolidate multiple SQL Server instances or servers into a single target SQL Server instance
- **SQL Server load balancing**—Distribute SQL Server databases to achieve optimal performance

Chapter 2 Requirements

You must meet the following requirements to use Carbonite Migrate SQLMove.

- **Source and target servers**—Your source and target servers must meet the following requirements.
  - **SQL Server version**—You can migrate from and to the following Microsoft SQL Server versions.
    - **Source SQL Server**
      - SQL Server 2005
      - SQL Server 2008
      - SQL Server 2008 R2
      - SQL Server 2012
      - SQL Server 2014
      - SQL Server 2016
      - SQL Server 2017
    - **Target SQL Server**
      - SQL Server 2008
      - SQL Server 2008 R2
      - SQL Server 2012
      - SQL Server 2014
      - SQL Server 2016
      - SQL Server 2017
  - **SQL Server components**—You can migrate only database engine components. You cannot migrate SQL Server Analysis Services (SSAS), SQL Server Reporting Services (SSRS), or SQL Server Integration Services (SSIS).
  - **SQL Server AlwaysOn Availability Groups (AAG)**—AAG is not supported on the source. It can be supported on the target by migrating into the AAG instance and then promoting the migrated database into an AAG configuration.
  - **Database compatibility level**—You can migrate databases set to compatibility level 80 or later.
  - **SQL Server Management Objects (SMO)**—The SMO version is associated with your SQL version, however the target must have the same or later version of SMO than the source.
  - **Carbonite Migrate**—Your source and target servers must have Carbonite Migrate for Windows version 7.1.2.x or later installed. (Older versions are called Double-Take Move). Because your servers are running Carbonite Migrate, they must meet all Carbonite Migrate requirements. See the Carbonite Migrate for Windows User's Guide for your version from the Carbonite Replication Products Documentation Library.
  - **Licensing**—If your source SQL Server is using SQL Windows Server Failover Clusters (WSFC), you must have Carbonite Migrate Premium licensing. If you are not using WSFC, you can use any Carbonite Migrate licensing.

- **PowerShell**—Your source and target servers must have PowerShell version 2.0 or later. PowerShell remoting must be enabled on both servers. See your PowerShell documentation for details on Enable-PSRemoting.
- **Permissions**—You must meet the following permissions for Carbonite Migrate and SQL Server.
  - **Carbonite Migrate**—The account used to log in to Carbonite Migrate on the source and target must be a member of the local Double-Take Admin security group.
  - **SQL Server**—SQLMove will use Windows Authentication (connecting to SQL Server using the current user's credentials) by default, but you can specify SQL Authentication if the current user lacks appropriate SQL permissions. The SQL account used to access SQL Server must be a member of the sysadmin server role.
- **Connectivity**—The source and target must be able to communicate with each other. You can use SQL Server Management Studio (SSMS) or the Carbonite Migrate Replication Console to test connectivity between these components.
- **Clusters**—Source or target cluster configurations must meet the following requirements.
  - **Windows Management Instrumentation (WMI)**—If you are running on a cluster, all possible owning nodes of the SQL resource group must have Windows Management Framework version 3.0 or later. Version 4.0 or later is recommended.
  - **Dedicated Administrator Connection (DAC)**—Windows 2008 R2 clustered SQL Servers will need to enable a DAC to allow connections. DAC can be enabled by running the following T-SQL query against the clustered SQL virtual instance. See Microsoft documentation for details on DAC.

    ```
    EXEC sp_configure 'remote admin connections', 1;
    GO
    RECONFIGURE
    GO
    ```

  - **Server name**—When using the SQLMove commands, source or target server parameters should specify the virtual (SQL) server name not the cluster name.
- **SQLMove controller**—The machine where you are running SQLMove must meet the following requirements.
  - **Operating system**—The SQLMove controller should be run from Windows 8 or later, or Windows 2012 R2 or later.
  - **SQL Server Management Objects (SMO)**—The SQLMove controller must have SQL Server 2012 SMO or later. The **Management Tools - Basic** and **SQL Client Connectivity SDK** components must be enabled.
  - **Carbonite Migrate**—The SQLMove controller must have Carbonite Migrate for Windows version 7.1.2.x or later installed. If you are running SQLMove from your source or target, install both the server and client components. If you are running SQLMove from a desktop operating system or a server operating system that is not your source or target, install the client components only. See the Carbonite Migrate for Windows User's Guide for your version from the Carbonite Replication Products Documentation Library for details on client only requirements.
  - **PowerShell**—The SQLMove controller must have PowerShell version 3.0 or later. Version 4.0 or later is recommended. The PowerShell ExecutionPolicy must be set to allow scripts to run. See your PowerShell documentation for details on Set-ExecutionPolicy.

- **Connectivity**—The SQLMove controller must have connectivity to both your source and target servers.
- **Cluster and Windows Management Instrumentation (WMI)**—If you are running SQL Server on a cluster, the SQLMove controller must have remote access to any source or target cluster in the configuration.

Chapter 3 Installation

You must install Carbonite Migrate on your source and target servers. SQLMove can run from either of those servers or from a third machine.

1. Make sure you have reviewed the *Requirements* on page 5.
2. Install, license, and active Carbonite Migrate on your source and target servers. See the *Installation, Licensing, and Activation Guide* for your Carbonite Migrate version from the Carbonite Replication Products Documentation Library.
3. If you plan on running SQLMove from a machine that is not your source or target, install the Carbonite Migrate client components on that machine. Reference the same installation guide.
4. Copy the SQLMove zip file that you downloaded to your source, target, or a third machine and extract the files to a single folder, for example C:\SQLMove.

Chapter 4 Migrating SQL Server

Because SQLMove is a PowerShell scripted solution, it provides the flexibility to migrate various SQL Server configurations. You will need to provide parameters to the script for your specific configuration and desired migration, and SQLMove will do the rest of the work. It will determine the SQL Server databases and locations and create a Carbonite Migrate job to synchronize the data from the source to the target. Once the source and target are synchronized, you will be able to perform cutover, again providing parameters for your specific needs.

Review the following scenarios for instructions and examples for migrating that type of SQL Server configuration.

For a complete reference on the script syntax, see *SQLMove script syntax* on page 24.

Migrating the default instance

The following example migrates the default instance. You will need to modify the script syntax to meet your specific environment needs.

If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Source "alpha, 1124" instead of -Source alpha.

1. Make sure you have completed the steps for *Installation* on page 8.
2. Open a command prompt and go to the directory where you unzipped SQLMove.
3. Save Carbonite Migrate credentials for use in the SQLMove script by running the command

   .\SQLMove.ps1 ***-SaveCredentials*** <FileName>

   where FileName is the name of a file where you will save credentials for Carbonite Migrate. For example, you might use the command

   .\SQLMove.ps1 ***-SaveCredentials*** "C:\SQLMove\DTServerCreds.xml"

   Make sure you specify a user name (domain\user) that is a member of the local Double-Take Admin security group.

   See *Credentials* on page 25 for complete details on the ***-SaveCredentials*** syntax.
4. Initiate the migration process, which will query the source for database information and then configure and start Carbonite Migrate mirroring and replication, by running the command

   .\SQLMove.ps1 ***-Migrate -Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-DatabasesOnly***

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), and PathAndName is the name of the Carbonite Migrate credentials file you created in step 3. For example, you might use the command

   .\SQLMove.ps1 ***-Migrate -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DatabasesOnly***

   See *Migration* on page 26 for complete details on the ***-Migrate*** syntax.
5. Once the migration command has completed, the Carbonite Migrate job has been set up and is running. You can monitor the migration progress by running the command

   .\SQLMove.ps1 ***-JobStatus -Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName>

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), and PathAndName is the name of the Carbonite Migrate credentials file you created in step 3. For example, you might use the command

.\SQLMove.ps1 ***-JobStatus*** ***-Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml"

See *Job status* on page 29 for complete details on the ***-JobStatus*** syntax.

6. When mirroring has completed, the job is ready for cutover. The cutover process removes the Carbonite Migrate job and attaches the migrated databases to the target. This is also where you trigger the migration of the SQL configuration information and metadata from the source server. Run the command

   .\SQLMove.ps1 ***-Cutover*** ***-Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-DeleteJobAfterCutover***

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), and PathAndName is the name of the Carbonite Migrate credentials file you created in step 3. For example, you might use the command

   .\SQLMove.ps1 ***-Cutover*** ***-Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DeleteJobAfterCutover***

   See *Cutover* on page 31 for complete details on the ***-Cutover*** syntax.

Migrating a named instance

The following example migrates a named instance. You will need to modify the script syntax to meet your specific environment needs.

If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Source "alpha, 1124" instead of -Source alpha.

1. Make sure you have completed the steps for *Installation* on page 8.
2. Open a command prompt and go to the directory where you unzipped SQLMove.
3. Save Carbonite Migrate credentials for use in the SQLMove script by running the command

   .\SQLMove.ps1 ***-SaveCredentials*** <FileName>

   where FileName is the name of a file where you will save credentials for Carbonite Migrate. For example, you might use the command

   .\SQLMove.ps1 ***-SaveCredentials*** "C:\SQLMove\DTServerCreds.xml"

   Make sure you specify a user name (domain\user) that is a member of the local Double-Take Admin security group.

   See *Credentials* on page 25 for complete details on the ***-SaveCredentials*** syntax.
4. Initiate the migration process, which will query the source for database information and then configure and start Carbonite Migrate mirroring and replication, by running the command

   .\SQLMove.ps1 ***-Migrate -Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-DatabasesOnly -SourceInstance*** <InstanceName> ***-TargetInstance*** <InstanceName>

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), PathAndName is the name of the Carbonite Migrate credentials file you created in step 3, and InstanceName is the name of the instance you want to migrate. For example, you might use the command

   .\SQLMove.ps1 ***-Migrate -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DatabasesOnly -SourceInstance*** PRDSQL\PROD ***-TargetInstance*** PRDSQL\PROD

   See *Migration* on page 26 for complete details on the ***-Migrate*** syntax.
5. Once the migration command has completed, the Carbonite Migrate job has been set up and is running. You can monitor the migration progress by running the command

   .\SQLMove.ps1 ***-JobStatus -Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName>

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), and PathAndName is the name of

the Carbonite Migrate credentials file you created in step 3. For example, you might use the command

.\SQLMove.ps1 ***-JobStatus*** ***-Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml"

See *Job status* on page 29 for complete details on the ***-JobStatus*** syntax.

6. When mirroring has completed, the job is ready for cutover. The cutover process removes the Carbonite Migrate job and attaches the migrated databases to the target. This is also where you trigger the migration of the SQL configuration information and metadata from the source server. Run the command

   .\SQLMove.ps1 ***-Cutover*** ***-Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-SourceInstance*** <InstanceName> ***-TargetInstance*** <InstanceName> ***-DeleteJobAfterCutover***

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), PathAndName is the name of the Carbonite Migrate credentials file you created in step 3, and InstanceName is the name of the instance you migrated. For example, you might use the command

   .\SQLMove.ps1 ***-Cutover*** ***-Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-SourceInstance*** PRDSQL\PROD ***-TargetInstance*** PRDSQL\PROD ***-DeleteJobAfterCutover***

   See *Cutover* on page 31 for complete details on the ***-Cutover*** syntax.

Migrating a specific database

The following example migrates a specific database. You will need to modify the script syntax to meet your specific environment needs.

If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Source "alpha, 1124" instead of -Source alpha.

1. Make sure you have completed the steps for *Installation* on page 8.
2. Open a command prompt and go to the directory where you unzipped SQLMove.
3. Save Carbonite Migrate credentials for use in the SQLMove script by running the command

   .\SQLMove.ps1 ***-SaveCredentials*** <FileName>

   where FileName is the name of a file where you will save credentials for Carbonite Migrate. For example, you might use the command

   .\SQLMove.ps1 ***-SaveCredentials*** "C:\SQLMove\DTServerCreds.xml"

   Make sure you specify a user name (domain\user) that is a member of the local Double-Take Admin security group.

   See *Credentials* on page 25 for complete details on the ***-SaveCredentials*** syntax.
4. Initiate the migration process, which will query the source for database information and then configure and start Carbonite Migrate mirroring and replication, by running the command

   .\SQLMove.ps1 ***-Migrate -Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-DatabasesOnly -IncludeDatabases*** @("Name1", "Name2", "Name3")

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), PathAndName is the name of the Carbonite Migrate credentials file you created in step 3, and IncludeDatabases is an array in the format @("database1", "database2", "database3") that identifies the databases you want to migrate. For example, you might use the command

   .\SQLMove.ps1 ***-Migrate -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DatabasesOnly -IncludeDatabases*** @("DB1","DB2")

   See *Migration* on page 26 for complete details on the ***-Migrate*** syntax.
5. Once the migration command has completed, the Carbonite Migrate job has been set up and is running. You can monitor the migration progress by running the command

   .\SQLMove.ps1 ***-JobStatus -Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName>

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), and PathAndName is the name of

the Carbonite Migrate credentials file you created in step 3. For example, you might use the command

.\SQLMove.ps1 ***-JobStatus*** ***-Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml"

See *Job status* on page 29 for complete details on the ***-JobStatus*** syntax.

6. When mirroring has completed, the job is ready for cutover. The cutover process removes the Carbonite Migrate job and attaches the migrated databases to the target. This is also where you trigger the migration of the SQL configuration information and metadata from the source server. Run the command

   .\SQLMove.ps1 ***-Cutover*** ***-Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-DeleteJobAfterCutover***

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), and PathAndName is the name of the Carbonite Migrate credentials file you created in step 3. For example, you might use the command

   .\SQLMove.ps1 ***-Cutover*** ***-Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DeleteJobAfterCutover***

   See *Cutover* on page 31 for complete details on the ***-Cutover*** syntax.

Migrating using Carbonite Migrate job options

An optional job options file can be used. Job options are automatically generated, so you only need to use a job options file if you want to change any specific settings. The following example specifies a job options file to customize path transformations, physical rule definitions, and/or IP address mappings. A job options file can be used with any type of migration (default instance, named instance, and so on). This example is a specific database migration type. You will need to modify the script syntax to meet your specific environment needs.

If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Source "alpha, 1124" instead of -Source alpha.

1. Make sure you have completed the steps for *Installation* on page 8.
2. Open a command prompt and go to the directory where you unzipped SQLMove.
3. Save Carbonite Migrate credentials for use in the SQLMove script by running the command

   .\SQLMove.ps1 ***-SaveCredentials*** <FileName>

   where FileName is the name of a file where you will save credentials for Carbonite Migrate. For example, you might use the command

   .\SQLMove.ps1 ***-SaveCredentials*** "C:\SQLMove\DTServerCreds.xml”

   Make sure you specify a user name (domain\user) that is a member of the local Double-Take Admin security group.

   See *Credentials* on page 25 for complete details on the ***-SaveCredentials*** syntax.
4. Locate the JobOptions_Sample__*.xml files in the directory where you unzipped SQLMove. Use these sample files as a basis for creating your own job options file that you want to pass to SQLMove. See *Job options* on page 18 for details and examples of using a job options file.
5. Initiate the migration process, which will query the source for database information and then configure and start Carbonite Migrate mirroring and replication, by running the command

   .\SQLMove.ps1 ***-Migrate -Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-DatabasesOnly -IncludeDatabases*** @("Name1", "Name2", "Name3") ***-JobOptions*** <PathAndName2>

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), PathAndName is the name of the Carbonite Migrate credentials file you created in step 3, IncludeDatabases is an array in the format @("database1", "database2", "database3") that identifies the databases you want to migrate, and PathAndName2 is the name of the job options file you created in step 4 that you want SQLMove to use. For example, you might use the command

   .\SQLMove.ps1 ***-Migrate -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml” ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml” ***-DatabasesOnly -IncludeDatabases*** @("DB1","DB2") ***-JobOptionsFile*** "C:\SQLMove\JobOptions.xml"

See *Migration* on page 26 for complete details on the ***-Migrate*** syntax.

6. Once the migration command has completed, the Carbonite Migrate job has been set up and is running. You can monitor the migration progress by running the command

   .\SQLMove.ps1 ***-JobStatus -Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName>

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), and PathAndName is the name of the Carbonite Migrate credentials file you created in step 3. For example, you might use the command

   .\SQLMove.ps1 ***-JobStatus -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml"

   See *Job status* on page 29 for complete details on the ***-JobStatus*** syntax.

7. When mirroring has completed, the job is ready for cutover. The cutover process removes the Carbonite Migrate job and attaches the migrated databases to the target. This is also where you trigger the migration of the SQL configuration information and metadata from the source server. Run the command

   .\SQLMove.ps1 ***-Cutover -Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-JobOptions*** <PathAndName2> ***-DeleteJobAfterCutover***

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name), PathAndName is the name of the Carbonite Migrate credentials file you created in step 3, and PathAndName2 is the name of the job options file you want SQLMove to use. For example, you might use the command

   .\SQLMove.ps1 ***-Cutover -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-JobOptionsFile*** "C:\SQLMove\JobOptions.xml" ***-DeleteJobAfterCutover***

   See *Cutover* on page 31 for complete details on the ***-Cutover*** syntax.

Job options

You can pass in a customized job options file when you migrate or cutover. These job options are advanced configurations with strict rules that must be followed. You may want or need to engage Carbonite Professional Services for implementation assistance.

- **Changing the target path for migration**—In this example, if the source SQL Server instance is located in F:\SQL\Data, you can use a job options file to migrate to H:\SQL\Data on the target. You can also use this customization if you are doing a same server storage migration. You can only have one path transformation per source path. Also, path transformations are an all or none configuration. If you map one source path to a path on the target, you must map all of your source paths to a path on the target. If you have a mix of paths that are and are not changing, your source and target paths for those paths that are not changing must still be in the job options file, but they would be mapped to the same path on the target.

  ```
  <d4p1:PathTransformation>
    <d4p1:SourcePath>F:\SQL\DATA\</d4p1:SourcePath>
    <d4p1:TargetPath>H:\SQL\DATA\</d4p1:TargetPath>
  </d4p1:PathTransformation>
  ```

  The SQLMove zip file contains a sample job options file called JobOptions_Sample___Change_ Target_Path.xml which you can use as a guide.

- **Changing the target file name for migration**—In this example, if the source SQL Server files are called AdventureWorks1_Data.mdf and AdventureWorks1_Log.ldf, you can use a job options file to rename the files during the migration to AdWrk1_Dat.mdf and AdWrk1_Log.ldf.

  ```
  <d4p1:PathTransformation>
    <d4p1:SourcePath>F:\Program Files\Microsoft SQL
  Server\MSSQL.1\MSSQL\DATA\AdventureWorks1_Data.mdf</d4p1:SourcePath>
    <d4p1:TargetPath>F:\Program Files\Microsoft SQL
  Server\MSSQL.1\MSSQL\DATA\AdWrk1_Dat.mdf</d4p1:TargetPath>
  </d4p1:PathTransformation>
  <d4p1:PathTransformation>
    <d4p1:SourcePath>F:\Program Files\Microsoft SQL
  Server\MSSQL.1\MSSQL\DATA\AdventureWorks1_Log.ldf</d4p1:SourcePath>
    <d4p1:TargetPath>F:\Program Files\Microsoft SQL
  Server\MSSQL.1\MSSQL\DATA\AdWrk1_Log.ldf</d4p1:TargetPath>
  </d4p1:PathTransformation>
  ```

- **Adding additional non-SQL data for the migration**—In this example, if your source has additional data, like C:\test, you can migrate that additional non-SQL data by adding physical rules to a job options file. You must add a path transformation for any physical rules you add.

```
<d4p1:PhysicalRule>
  <d4p1:Inclusion>Include</d4p1:Inclusion>
  <d4p1:IsReadOnly>false</d4p1:IsReadOnly>
  <d4p1:Metadata i:nil="true" />
  <d4p1:Path>C:\test</d4p1:Path>
  <d4p1:Recursion>Recursive</d4p1:Recursion>
</d4p1:PhysicalRule>
```

  The SQLMove zip file contains a sample job options file called JobOptions_Sample___Add_Non_SQL_Data.xml which you can use as a guide.

- **Changing IP address mappings for cutover**—In this example, if your source IP address is 172.31.202.149, you can cutover to an IP address of 172.31.202.159 using a job options file. Only use this option if you are using the -DnsFailover parameter with cutover. With this option, any clients that were connecting to the source via the original IP address will redirect to the target IP address once the TTL (time to live) for the original source DNS record expires or the client's DNS cache is flushed.

```
<d6p1:IPAddressMap>
  <d6p1:SourceIP>172.31.202.149</d6p1:SourceIP>
  <d6p1:TargetIP>172.31.202.159</d6p1:TargetIP>
</d6p1:IPAddressMap>
```

  The SQLMove zip file contains a sample job options file called JobOptions_Sample___Specify_DNS_Settings.xml which you can use as a guide.

Chapter 5 Testing cutover

When a job is ready for cutover, you can perform a test cutover or a live cutover. In a test cutover, SQLMove will not take the source or SQL databases offline, but it will stop Carbonite Migrate replication between the source and target. Because the source and target servers are uniquely named, they can coexist during the test, and you can complete testing on the target to determine status and connectivity on the new server.

1. Make sure the job has completed mirroring and is ready for cutover. See *Job status* on page 29 for details.
2. Begin the test cutover by running the command

   .\SQLMove.ps1 ***-Cutover -Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-DatabasesOnly -DeleteJobAfterCutover -KeepSourceOnline***

   where ServerName is the name of your source and target servers (in a cluster configuration, specify the virtual (SQL) server name not the cluster name) and PathAndName is the name of the Carbonite Migrate credentials file you created when you created the migration. For example, you might use the command

   .\SQLMove.ps1 ***-Cutover -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DeleteJobAfterCutover -KeepSourceOnline***

   See *Cutover* on page 31 for complete details on the ***-Cutover*** syntax.

   If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Source "alpha, 1124" instead of -Source alpha.

3. When cutover is complete, perform your testing on the target to determine status and connectivity on the new server.
4. When you are done with your testing, detach the upgraded database from the target.
5. Because the Carbonite Migrate job is deleted during a test migration, repeat the process to initiate the migration again. See *Migrating SQL Server* on page 9 for links to examples of each of the different migration types and see *Migration* on page 26 for complete details on the ***-Migrate*** syntax.

Chapter 6 SQL Server configuration and metadata

When you cutover, the SQL Server configuration and metadata will be migrated to the target, unless you specifically exclude it by using the ***-DatabasesOnly*** parameter. This functionality is scripted using a separate utility called dbatools, which can be found in dbatools\Functions in the SQLMove zip file. You can use this script as is or modify it as you desire. You also have the option of writing your own SQL object and metadata migration script or manually creating or scripting a custom process, which you then use in the Invoke-SecondarySqlMigrationModule function of SQLMove.ps1. Keep in mind, you are responsible for troubleshooting and diagnosing issues that occur from any customizations.

The following list identifies the SQL objects that SQLMove migrates by default, without any customizations.

- **Defaults when the source SQL Server is version 2005, 2008, or 2008 R2**
  - All logins
  - All credentials
  - All objects within the Job Server (SQL Agent)
  - All linked servers
  - All groups and servers within Central Management Server
  - All user objects in system databases
  - All system triggers
  - All system backup devices
- **Defaults when the source SQL Server is version 2012 or higher**
  - All logins
  - All database mail objects
  - All credentials
  - All objects within the Job Server (SQL Agent)
  - All linked servers
  - All groups and servers within Central Management Server
  - All user objects in system databases
  - All system triggers
  - All system backup devices
- **Defaults when the source and target are the same SQL Server version**
  - All logins
  - All database mail objects (if source and target SQL Server 2012 or higher)
  - All credentials
  - All objects within the Job Server (SQL Agent)
  - All linked servers
  - All groups and servers within Central Management Server
  - All SQL Server configuration objects (everything in sp_configure)
  - All user objects in system databases

- All system triggers
- All system backup devices

Chapter 7 Log file

All output from SQLMove is logged to the file SQLMove.log in the location where you are running SQLMove. Below are some common errors you may see in the log and how to resolve them.

- **File C:\SQLMove\SQLMove.ps1 cannot be loaded because the execution policy of scripts is disabled on this system.**—This error is caused by not setting the PowerShell ExecutionPolicy. Use Set-ExecutionPolicy to allow scripts to run. See PowerShell documentation for more information about Set-ExecutionPolicy.
- **Get-ItemProperty : Cannot find path 'HKEY_LOCAL_MACHINE\SOFTWARE\NSI_Software\Double-Take\CurrentVersion' because it does not exist.**— Carbonite Migrate is not installed on the source or target or it was incorrectly installed. Install or repair Carbonite Migrate on the source and target and make sure that each is properly licensed for your Carbonite Migrate version.
- **The script 'SQLMove.ps1' cannot be run because it contained a "#requires" statement at line 1 for Windows PowerShell version 3.0.**—The wrong version of PowerShell is on the source or target. Install PowerShell version 3 or later.
- **SQL Server SMO required. Refer to 'Microsoft SQL Server 2014 Feature Pack' for more info on installing SQL Server SMO on this machine.**—SQL Server SMO is not installed on the machine running SQLMove. Install SQL Server 2012 SMO or later. The Management Tools - Basic and SQL Client Connectivity SDK components must be enabled.
- **Get-SmoServer : Exception calling "Connect" with "0" argument(s): Failed to connect to server.**—This error is caused by network firewall issues. Make sure the source and target can communicate with each other (ping) and make sure the Carbonite Migrate ports 6320, 6325, and 6326 are open.
- **Write-Message : Required dbatools module was not found.**—The secondary script for migrating SQL configuration and metadata is missing. Make sure you have the folder /dbatools and its files in the location where you are running SQLMove. If you do not, download and unzip SQLMove again.
- **Write-Message : Unable to invoke source PowerShell command.**—PowerShell remoting is not enabled on the source or target. Enable PowerShell remoting. See your PowerShell documentation for details on Enable-PSRemoting.
- **SQL Server: alpha: RegQueryValueEx() returned error 2, 'The system cannot find the file specified.'**—SMO is reporting it cannot find an item it is looking for. For example, it may not find a registry key being queried because it does not exist on the server. Generally, this error can be ignored.
- **Unable to open the physical file … Operating system error 5 (Access is denied).**— The databases listed in the error message are not attached during cutover. This can also impact SQL configuration or metadata that points to the databases that were not attached. This issue can be seen when specifying a SQL login for accessing the target server rather than a Windows login that has administrator-level permissions on the target. If the SQL Server service on the target does not have sufficient permissions to access the SQL databases files or the folder they reside in on the target server, SQLMove is unable to attach the databases. Verify that the MSSQLServer service (or the MSSQL$Instance service if a named instance is used) has target-side write access to each of the database files that comprise the databases being migrated and that they are started under a domain account that has write access to the files and folders referenced in order to be able to attach the databases.

Chapter 8 SQLMove script syntax

The name of the SQLMove PowerShell script is SQLMove.ps1, and it is located in the directory where you unzipped SQLMove. See the following sections for the parameters to use with each function of SQLMove.

- ***Credentials* on page 25**—If you do not save Carbonite Migrate credentials, SQLMove will prompt you for them when necessary. You can save Carbonite Migrate and SQL Server credentials to an encrypted file.
- ***Migration* on page 26**—The migration process queries the source for database information and then configures and starts Carbonite Migrate mirroring and replication.
- ***Job status* on page 29**—Once the migration command has completed, the Carbonite Migrate job has been set up and is running, and you will want to monitor the migration process.
- ***Cutover* on page 31**—When mirroring has completed and the job is ready for cutover, you can perform a test or live cutover.
- ***Undo* on page 35**—The undo process deletes a Carbonite Migrate job and reverts any DNS and SPN updates that were made during cutover.

Credentials

You can save Carbonite Migrate and SQL Server credentials to an encrypted, XML, UTF8 encoded file. If you need to persist unique credentials for different users, use unique file names because persisting credentials will overwrite an existing file with the specified name. If you do not save Carbonite Migrate credentials needed to log in to the source and target servers, SQLMove will prompt you for them when necessary.

Syntax

.\SQLMove.ps1 ***-SaveCredentials*** <FileName> ***-Verbose***

Parameters

-SaveCredentials—Prompts for and saves credentials to an encrypted, XML, UTF8 encoded file

-Verbose—Includes additional debug messages in the SQLMove log file

Examples

1. **Carbonite Migrate credentials**—This command prompts you for a username and password. The username you specify (domain\user) must be a member of the local Double-Take Admin security group. The credentials you specify are saved to the file you specify in the command. If you want to use unique credentials for your source and target servers, repeat this command specifying a unique file name.

   PS C:\SQLMove> .\SQLMove.ps1 ***-SaveCredentials***
   "C:\SQLMove\DTServerCreds.xml"

2. **SQL Server credentials**—This command is like example number 1, except you are specifying a file to store credentials for SQL Server Authentication.

   PS C:\SQLMove> .\SQLMove.ps1 ***-SaveCredentials***
   "C:\SQLMove\SQLServerCreds.xml"

See the examples in *Migration* on page 26 and *Cutover* on page 31 for how you can use these saved credentials.

Migration

The migration process queries the source for database information and then configures and starts Carbonite Migrate mirroring and replication.

Syntax

.\SQLMove.ps1 ***-Migrate-Source*** <ServerName> ***-Target*** <ServerName> ***-SourceIsStandAlone -TargetIsStandAlone -SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-DatabasesOnly-SourceInstance*** <InstanceName> ***-TargetInstance*** <InstanceName> ***-IncludeDatabases*** @("Name1", "Name2", "Name3") ***-ExcludeDatabases*** @("Name1", "Name2", "Name3") ***-JobOptions*** <PathAndName> ***-UseSqlLoginSource -SourceSqlCredsFile*** <PathAndName> ***-UseSqlLoginTarget -TargetSqlCredsFile*** <PathAndName> ***-RecreateLogins -Verbose***

Parameters

-Migrate—Queries the source for database information and then configures and starts Carbonite Migrate mirroring and replication

-Source—The source server name or IP address for standalone servers. Use the virtual server name for cluster instances. If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Source "alpha, 1124" instead of -Source alpha.

-Target—The target server name or IP address for standalone servers. Use the virtual server name for cluster instances. If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Target "beta, 1124" instead of -Target beta.

-SourceIsStandAlone—The source server will be treated as a standalone (not clustered) server

-TargetIsStandAlone—The target server will be treated as a standalone (not clustered) server

-SourceDtCredsFile—The path to a file containing credentials for logging in to Carbonite Migrate on the source server

-TargetDtCredsFile—The path to a file containing credentials for logging in to Carbonite Migrate on the target server

-DatabasesOnly—Only the database files will be migrated. No SQL metadata or configuration information will be migrated.

-SourceInstance—The source SQL Server instance name. Do not specify this option if you are using the default instance or a cluster instance.

-TargetInstance—The target SQL Server instance name. Do not specify this option if you are using the default instance or a cluster instance.

-IncludeDatabases—An array in the format @("database1", "database2", "database3") that identifies the databases to include for migration. This parameter takes precedence over the ExcludeDatabases parameter.

-ExcludeDatabases—An array in the format @("database1", "database2", "database3") that identifies the databases to exclude for migration. By default, the system databases master, msdb, tempdb, and model are excluded.

-JobOptions—The path and file name of a job options configuration file that contains path transformation and physical rule definitions. See *Job options* on page 18 for details on using a job options file.

-UseSqlLoginSource—Prompts for credentials for SQL Server Authentication when connecting to the source server. If this parameter is not used, the default login type of Windows Authentication will be used to connect to SQL Server using the current user's credentials.

-SourceSqlCredsFile—The path to a file containing credentials for SQL Server Authentication when connecting to the source server. If this parameter is not used, the default login type of Windows Authentication will be used to connect to SQL Server using the current user's credentials.

-UseSqlLoginTarget—Prompts for credentials for SQL Server Authentication when connecting to the target server. If this parameter is not used, the default login type of Windows Authentication will be used to connect to SQL Server using the current user's credentials.

-TargetSqlCredsFile—The path to a file containing credentials for SQL Server Authentication when connecting to the target server. If this parameter is not used, the default login type of Windows Authentication will be used to connect to SQL Server using the current user's credentials.

-RecreateLogins—Deletes and re-creates SQL logins that already exist on the target. This parameter can be useful when performing a live cutover after a test cutover or if SQL Server credentials exist on the target from a previous cutover activity.

-Verbose—Includes additional debug messages in the SQLMove log file

Examples

1. **Migration of default instance**—This command migrates from a source server called alpha to a target server called beta. Because the ***-DatabasesOnly*** parameter is used and no instance name is specified, this command will migrate only the non-system database files from the default instance. SQLMove will not migrate the SQL Server configuration or metadata. You will migrate those later during cutover. You will be prompted for Carbonite Migrate credentials for your source and target. The username you specify (domain\user) must be a member of the local Double-Take Admin security group.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Migrate*** ***-Source*** alpha ***-Target*** beta ***-DatabasesOnly***

2. **Migration of default instance using saved credentials**—This command is like example number 1, except instead of you being prompted for Carbonite Migrate credentials, the credentials stored in the specified file will be used.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Migrate*** ***-Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DatabasesOnly***

3. **Migration of named instance using saved credentials**— This command is like example 2, except only the non-system database files from the specified named instance called PRDSQL\PROD will be migrated.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Migrate -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DatabasesOnly -SourceInstance*** PRDSQL\PROD ***-TargetInstance*** PRDSQL\PROD

4. **Migration of specific databases using saved credentials**— This command is like example 2, except only the databases called DB1 and DB2 will be migrted.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Migrate -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DatabasesOnly -IncludeDatabases*** @("DB1","DB2")

5. **Migration with job options using saved credentials**— This command is like example 2, except it specifies a job options file to customize path transformations, physical rule definitions, and/or IP address mappings. See *Job options* on page 18 for details on the job options available.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Migrate -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DatabasesOnly -JobOptionsFile*** "C:\SQLMove\JobOptions.xml"

Job status

Once the migration command has completed, the Carbonite Migrate job has been set up and is running, and you will want to monitor the migration process.

Syntax

.\SQLMove.ps1 ***-JobStatus -Source*** <ServerName> ***-Target*** <ServerName>
-SourceDtCredsFile <PathAndName> ***-TargetDtCredsFile*** <PathAndName>
-SourceInstance <InstanceName> ***-TargetInstance*** <InstanceName>
-ExistingJobName <JobName> ***-Verbose***

Parameters

-JobStatus—Displays the current status of the Carbonite Migrate job

-Source—The source server name or IP address for standalone servers. Use the virtual server name for cluster instances. If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Source "alpha, 1124" instead of -Source alpha.

-Target—The target server name or IP address for standalone servers. Use the virtual server name for cluster instances. If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Target "beta, 1124" instead of -Target beta.

-SourceDtCredsFile—The path to a file containing credentials for logging in to Carbonite Migrate on the source server

-TargetDtCredsFile—The path to a file containing credentials for logging in to Carbonite Migrate on the target server

-SourceInstance—The source SQL Server instance name. Do not specify this option if you are using the default instance or a cluster instance.

-TargetInstance—The target SQL Server instance name. Do not specify this option if you are using the default instance or a cluster instance.

-ExistingJobName—Specifies the Carbonite Migrate job name to cutover. If no name is specified, the default job name created by Carbonite Migrate will be used.

-Verbose—Includes additional debug messages in the SQLMove log file

Examples

1. **Prompt for credentials**—This command retrieves the job status for the source server called alpha and the target server called beta. You will be prompted for Carbonite Migrate credentials for the source and target. The username you specify (domain\user) must be a member of the local Double-Take Admin security group.

PS C:\SQLMove> .\SQLMove.ps1 ***-JobStatus -Source*** alpha ***-Target*** beta

2. **Used saved credentials**—This command is like example number 1, except instead of you being prompted for Carbonite Migrate credentials, the credentials stored in the specified file will be used.

   PS C:\SQLMove> .\SQLMove.ps1 ***-JobStatus -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml"

Cutover

When mirroring has completed and the job is ready for cutover, you can perform a test or live cutover. The cutover process removes the Carbonite Migrate job and attaches the migrated databases to the target. Cutover is also where you should trigger the migration of the SQL configuration information and metadata from the source server.

Syntax

.\SQLMove.ps1 ***-Cutover -Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-DatabasesOnly -SourceInstance*** <InstanceName> ***-TargetInstance*** <InstanceName> ***-IncludeDatabases*** @("Name1", "Name2", "Name3") ***-ExcludeDatabases*** @("Name1", "Name2", "Name3") ***-JobOptions*** <PathAndName> ***-DnsFailover -DeleteJobAfterCutover -UseSqlLoginSource -SourceSqlCredsFile*** <PathAndName> ***-UseSqlLoginTarget -TargetSqlCredsFile*** <PathAndName> ***-ExistingJobName*** <JobName> ***-ExistingJobId*** <JobId> ***-OverrideDbOwner -RecreateLogins -KeepSourceOnline -Verbose***

Parameters

-Cutover—Removes the Carbonite Migrate job and attaches the migrated databases to the target

-Source—The source server name or IP address for standalone servers. Use the virtual server name for cluster instances. If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Source "alpha, 1124" instead of -Source alpha.

-Target—The target server name or IP address for standalone servers. Use the virtual server name for cluster instances. If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Target "beta, 1124" instead of -Target beta.

-SourceDtCredsFile—The path to a file containing credentials for logging in to Carbonite Migrate on the source server

-TargetDtCredsFile—The path to a file containing credentials for logging in to Carbonite Migrate on the target server

-DatabasesOnly—Only the database files will be cutover. No SQL metadata or configuration information will be migrated and be included after cutover.

-SourceInstance—The source SQL Server instance name. Do not specify this option if you are using the default instance or a cluster instance.

-TargetInstance—The target SQL Server instance name. Do not specify this option if you are using the default instance or a cluster instance.

-IncludeDatabases—An array in the format @("database1", "database2", "database3") that identifies the databases to include for migration. This parameter takes precedence over the ExcludeDatabases parameter.

-ExcludeDatabases—An array in the format @("database1", "database2", "database3") that identifies the databases to exclude for migration. By default, the system databases master, msdb, tempdb, and model are excluded.

-JobOptions—The path and file name of a job options configuration file that contains IP address mappings for DNS updates. See *Job options* on page 18 for details on using a job options file.

-DnsFailover—Updates the source server DNS records and SQL Server service account SPNs to redirect users to the target server. This feature requires elevated permissions to alter DNS records and to modify Service Principle Name attributes within Active Directory, specifically admin-level access to both DNS and AD. Additionally, the account use to login to the Carbonite Migrate source server has to have read and modify permissions for changing the Service Principle Name attributes for the source SQL Server service startup account. This parameter does not support named instances of SQL Server when the source and target instance names are different, and this parameter does not support clustered configurations.

-DeleteJobAfterCutover—Automatically deletes the Carbonite Migrate job after cutover

-UseSqlLoginSource—Prompts for credentials for SQL Server Authentication when connecting to the source server. If this parameter is not used, the default login type of Windows Authentication will be used to connect to SQL Server using the current user's credentials.

-SourceSqlCredsFile—The path to a file containing credentials for SQL Server Authentication when connecting to the source server. If this parameter is not used, the default login type of Windows Authentication will be used to connect to SQL Server using the current user's credentials.

-UseSqlLoginTarget—Prompts for credentials for SQL Server Authentication when connecting to the target server. If this parameter is not used, the default login type of Windows Authentication will be used to connect to SQL Server using the current user's credentials.

-TargetSqlCredsFile—The path to a file containing credentials for SQL Server Authentication when connecting to the target server. If this parameter is not used, the default login type of Windows Authentication will be used to connect to SQL Server using the current user's credentials.

-ExistingJobName—Specifies the Carbonite Migrate job name to cutover. If no name is specified, the default job name created by Carbonite Migrate will be used.

-ExistingJobId—Specifies the specific Carbonite Migrate job ID to cutover

-OverrideDbOwner—Identifies the login that will be used as the new database owner on the target after cutover. This parameter overrides the original dbowner setting for the source database.

-RecreateLogins—Deletes and re-creates SQL logins that already exist on the target. This parameter can be useful when performing a live cutover after a test cutover or if SQL Server credentials exist on the target from a previous cutover activity.

-KeepSourceOnline—Performs a test cutover by not making any changes to the source SQL server functionality. For example, the source databases will not be taken offline during the test cutover.

-Verbose—Includes additional debug messages in the SQLMove log file

Examples

1. **Cutover of default instance databases only**—This command begins cutover. You will be prompted for Carbonite Migrate credentials for your source and target. The username you specify

(domain\user) must be a member of the local Double-Take Admin security group. SQLMove will take the source databases offline and monitor the replication queues to make sure all data changes have been transmitted before attaching the replica databases on the target. The Carbonite Migrate job will then be deleted. Since you have specified databases only, the source database will remain offline and no configuration information and metadata will be migrated to the target. When cutover is complete, the source server can be taken offline and the target can be used as the production server. Redirection to the target can be accomplished by pointing clients to the new SQL Server (connection string change), renaming the new SQL Server to the same name as the old SQL Server, or with DNS redirection or IP address change.

PS C:\SQLMove> .\SQLMove.ps1 ***-Cutover -Source*** alpha ***-Target*** beta ***-DatabasesOnly -DeleteJobAfterCutover***

2. **Cutover of default instance databases and metadata**—This command is like example number 1, except it migrates SQL configuration and metadata. After the Carbonite Migrate job is deleted, the source database is brought back online and a secondary script is automatically launched to migrate the SQL configuration and metadata to the target. Performing this action at cutover ensures both databases are online and the latest metadata can be properly synchronized.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Cutover -Source*** alpha ***-Target*** beta ***-DeleteJobAfterCutover***

3. **Cutover of default instance databases and metadata using saved credentials**—This command is like example number 2, except it uses saved credentials instead of prompting for Carbonite Migrate credentials.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Cutover -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DeleteJobAfterCutover***

4. **Cutover of named instance and metadata using saved credentials**—This command is like example number 3, except it specifies the instances to cutover.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Cutover -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-SourceInstance*** PRDSQL\PROD ***-TargetInstance*** PRDSQL\PROD ***-DeleteJobAfterCutover***

5. **Cutover of specific databases and metadata using saved credentials**—If you migrated specific databases, you do not need to specify those databases again at cutover time. You can cutover the databases and metadata like example number 3.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Cutover -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DeleteJobAfterCutover***

6. **Cutover with DNS updates using saved credentials**— This command is like example

number 3, except it also updates DNS. SQLMove will change the source server's DNS records to point to the target server, and will update Active Directory Service Principle Names for the source SQL Server service startup account during cutover for client redirection. The DnsFailover parameter requires the Active Directory module for Windows PowerShell to be imported, and the account used to log in to Carbonite Migrate on the target to have admin-level privileges to Active Directory and DNS in order to make necessary updates

PS C:\SQLMove> .\SQLMove.ps1 ***-Cutover -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DeleteJobAfterCutover -DnsFailover***

7. **Cutover with job options using saved credentials**— This command is like example number 6, except it uses a job options file. The job options file allows the specification of IP address mapings. See *Job options* on page 18 for details on the cutover options available.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Cutover -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DeleteJobAfterCutover -DnsFailover -JobOptionsFile*** "C:\SQLMove\JobOptions.xml"

8. **Test cutover**—Any of the examples above can be changed from a live cutover to a test cutover by adding the ***-KeepSourceOnline*** parameter. SQLMove will not take the source or SQL databases offline, but it will stop Carbonite Migrate replication between the source and target. Because the source and target servers are uniquely named, they can coexist during the test. The Carbonite Migrate job will be deleted as part of this process. When cutover is complete, you can complete testing on the target to determine status and connectivity on the new server. When you are done with your testing, detach the upgraded database from the target and run SQLMove again to establish a new job in preparation for a live cutover.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Cutover -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-DeleteJobAfterCutover -KeepSourceOnline***

Undo

The undo process deletes a Carbonite Migrate job and reverts any DNS and SPN updates that were made during cutover by the -DnsFailover option. This command is typically only used after a test cutover. In some cases, you may want to undo (delete the Carbonite Migrate job) before you have cutover and start another migration process, perhaps with different migration settings.

Syntax

.\SQLMove.ps1 ***-Undo-Source*** <ServerName> ***-Target*** <ServerName> ***-SourceDtCredsFile*** <PathAndName> ***-TargetDtCredsFile*** <PathAndName> ***-SourceInstance*** <InstanceName> ***-TargetInstance*** <InstanceName> ***-UseSqlLoginSource -SourceSqlCredsFile*** <PathAndName> ***-ExistingJobName*** <JobName> ***-Verbose***

Parameters

-Undo—Deletes the Carbonite Migrate job and reverts any DNS and SPN updates that were made during cutover

-Source—The source server name or IP address for standalone servers. Use the virtual server name for cluster instances. If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Source "alpha, 1124" instead of -Source alpha.

-Target—The target server name or IP address for standalone servers. Use the virtual server name for cluster instances. If you need to specify a non-default port number, specify it as ServerName, Port. For example, you might use -Target "beta, 1124" instead of -Target beta.

-SourceDtCredsFile—The path to a file containing credentials for logging in to Carbonite Migrate on the source server

-TargetDtCredsFile—The path to a file containing credentials for logging in to Carbonite Migrate on the target server

-SourceInstance—The source SQL Server instance name. Do not specify this option if you are using the default instance or a cluster instance.

-TargetInstance—The target SQL Server instance name. Do not specify this option if you are using the default instance or a cluster instance.

-UseSqlLoginSource—Prompts for credentials for SQL Server Authentication when connecting to the source server. If this parameter is not used, the default login type of Windows Authentication will be used to connect to SQL Server using the current user's credentials.

-SourceSqlCredsFile—The path to a file containing credentials for SQL Server Authentication when connecting to the source server. If this parameter is not used, the default login type of Windows Authentication will be used to connect to SQL Server using the current user's credentials.

-ExistingJobName—Specifies the Carbonite Migrate job name to cutover. If no name is specified, the default job name created by Carbonite Migrate will be used.

-Verbose—Includes additional debug messages in the SQLMove log file

Examples

1. **Basic undo**—This command will delete the Carbonite Migrate job between the source server called alpha and the target server called beta, and it will revert any DNS and SPN updates that were made during cutover. You will be prompted for Carbonite Migrate credentials for the source and target. The username you specify (domain\user) must be a member of the local Double-Take Admin security group.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Undo -Source*** alpha ***-Target*** beta

2. **Undo using saved credentials**—This command is like example 1, except instead of you being prompted for Carbonite Migrate credentials, the credentials stored in the specified file will be used.

   PS C:\SQLMove> .\SQLMove.ps1 ***-Undo -Source*** alpha ***-Target*** beta ***-SourceDtCredsFile*** "C:\SQLMove\DTServerCreds.xml" ***-TargetDtCredsFile*** "C:\SQLMove\DTServerCreds.xml"