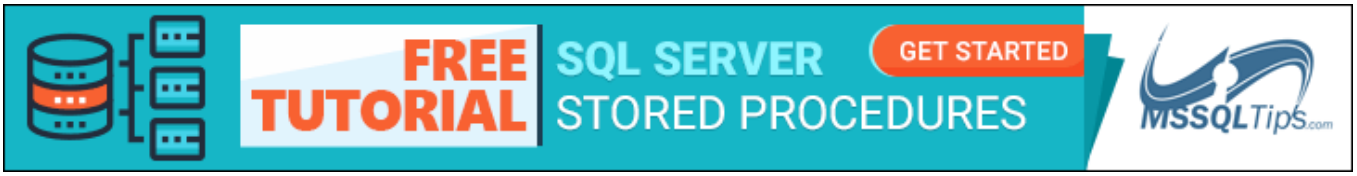
MSSQLTips Free SQL Server Conference Registration + Travel Expenses click to learn more

Configuring Transparent Data Encryption with SQL Server 2012 AlwaysOn Availability Groups



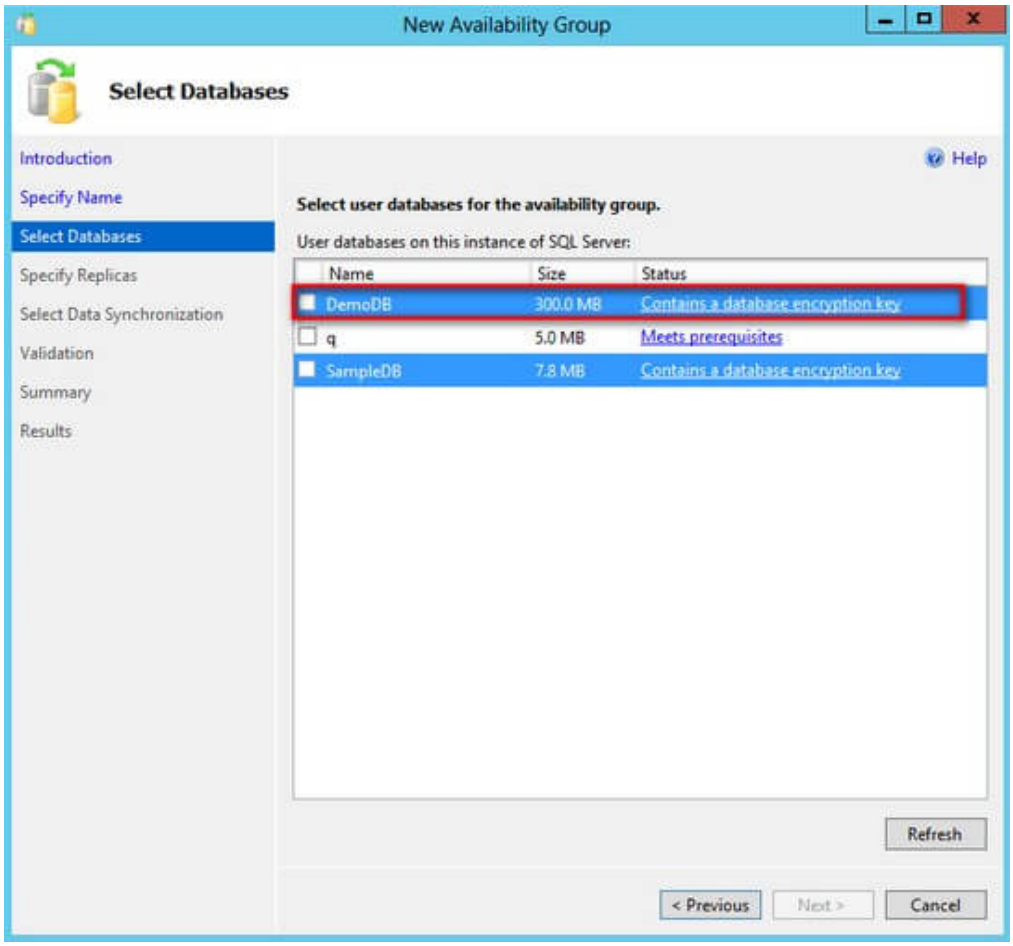
Free Webcast - Simplify SQL Server Administration

Problem

In a previous tip on Implementing Transparent Data Encryption in SQL Server 2008, we've seen how we can configure Transparent Data Encryption (TDE) with SQL Server 2008 databases. I wanted to implement TDE on my SQL Server 2012 databases and join them in an Availability Group. How do I do it?

Solution

Transparent Data Encryption (TDE) performs real-time I/O encryption and decryption of the data and log files, thereby, protecting data at rest. You can use TDE with Availability Groups to provide security and high availability to your databases. However, when working with TDE-enabled databases, you cannot use the **New Availability Group Wizard** or **Add Database to Availability Group Wizard** to add them to an Availability Group as shown in the screenshot below.



This tip will walk you thru the process of enabling TDE on your databases and joining them to an Availability Group. The list below outlines the high-level process to perform this task.

1. Configure SQL Server TDE on the primary and secondary replica
2. Initialize the encrypted SQL Server database on the secondary replica
3. Configure the SQL Server Availability Group

And, since you have very limited options to use the wizards to perform all of these tasks, we will be using T-SQL for all of them.

Configure SQL Server TDE on the primary and secondary replica

We will start by enabling TDE on the primary replica. This process is very similar to the one outlined in this tip. In a new Query Editor window,

1. Run the following command to create the service master key in the **master** database. Remember to document the password used for encrypting the service master key and keep it in a secure place.

```
USE MASTER
GO
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'mYC0mpl3XP@$$w0rd'
GO
```

2. Run the following command to create a certificate to encrypt the database encryption keys on the TDE-enabled databases. This certificate will be protected by the service master key.

```
CREATE CERTIFICATE TDECert
WITH SUBJECT = 'My TDE Certificate for all user database'
```

3. Run the following command to create a database encryption key on the user database that you will join to the Availability Group, protecting it using the certificate. You need to repeat this process for all of the databases that you will join to the Availability Group.

```
USE [SampleDB]
GO
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_128
ENCRYPTION BY SERVER CERTIFICATE TDECert
GO
```

4. Run the following command to turn on TDE on the database.

```
ALTER DATABASE [SampleDB]
SET ENCRYPTION ON
```

5. Run the following command to backup the certificate to a file. We will use this to enable TDE on the secondary replica. Remember to document the password used for encrypting the certificate file and keep it in a secure place.

```
USE MASTER
GO
BACKUP CERTIFICATE TDECert
TO FILE = 'C:\TDECert_File.cer'
WITH PRIVATE KEY (FILE = 'C:\TDECert_Key.pvk' ,
ENCRYPTION BY PASSWORD = 'mYC0mpl3XP@$$w0rd' )
GO
```

6. Copy the certificate file to the secondary replica.

Once TDE has been enabled on the databases in the primary replica, we can now proceed to enable it on the secondary replica. The reason why we are configuring TDE first on the secondary replica prior to joining the databases to an Availability Group is to make sure that we can perform data initialization. This is done by restoring the backups of the TDE-enabled databases to the secondary replica as highlighted in this previous tip.

To enable TDE on the secondary replica,

1. Run the following command to create the service master key in the **master** database. Remember to document the password used for encrypting the service master key and keep it in a secure place.

```
USE MASTER
GO
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'mYC0mpl3XP@$$w0rd'
GO
```

2. Run the following command to create a certificate to encrypt the database encryption keys on the TDE-enabled databases. This certificate will be based on the file we exported from the primary replica and copied to the secondary replica. Thus, we need to provide the password we used to encrypt it initially. This certificate will be protected by the service master key.

```
USE MASTER
GO
CREATE CERTIFICATE TDECert
FROM FILE = 'C:\TDECert_File.cer'
WITH PRIVATE KEY (FILE = 'C:\TDECert_Key.pvk',
DECRYPTION BY PASSWORD =  'mYC0mpl3XP@$$w0rd');
```

Once the service master key and the database master key has been configured on the secondary replica, we can now test the functionality by initializing the data in preparation for joining the databases to the Availability Group.

Initialize the encrypted SQL Server database on the secondary replica

The following steps outline the process of initializing the databases in preparation for the Availability Group configuration. The process involves taking a FULL and LOG backup of the databases from the primary replica and restoring it in NORECOVERY mode on the secondary replica.

1. On the primary replica, run the following command to take FULL and LOG backups.

```
BACKUP DATABASE SampleDB
TO DISK = 'E:\DBBackups\SampleDB.Bak'
   WITH INIT, STATS, ;
GO
BACKUP LOG SampleDB
TO DISK = 'E:\DBBackups\SampleDB.trn'
   WITH INIT, STATS, ;
GO
```

   Do this for all the databases that will be migrated to the new environment.

2. Copy the generated FULL and LOG backups to the secondary replica.

3. On the secondary replica, run the following command to restore the FULL and LOG backup. It is recommended to store the database files in the same location as that of the primary replica.

```sql
RESTORE DATABASE [SampleDB]
FROM DISK = N'E:\DBBackups\SampleDB.Bak'
WITH NORECOVERY, STATS = 5
GO
RESTORE LOG [SampleDB]
FROM DISK = N'E:\DBBackups\SampleDB.trn'
WITH NORECOVERY, STATS = 5
GO
```

4. Do this for all the databases that will be migrated to the new environment.

Note that if this process fails, verify that the certificate exported from the primary replica has been imported on the secondary replica prior to restoring the backups.

Configure the SQL Server Availability Group

Once the databases have been initialized on the secondary replica, we can proceed to configure an Availability Group and join the TDE-enabled database to it. The following steps outline the process for creating and configuring the Availability Group on the primary replica. We will be using the following information to configure the Availability Group.

- **SQL Server service account**: TESTDOMAIN\sqlservice (used by both primary and secondary replicas)
- **Availability Group Name:** AGTest
- **Availability Group Listener Name:** AGTestLN
- **Availability Group Listener IP Address:** 172.16.0.116
- **Availability Group Listener Port Number:** 1433
- **Availability Group Replicas:** WS-ALWAYSON-AG1, WS-ALWAYSON-AG2

On the primary replica, run the following commands to perform the following steps:

a. Create login that will be used to connect to the endpoint. For simplicity's sake, we will use the SQL Server service account for this purpose
b. Create the endpoint that the Availability Group will connect to
c. Grant CONNECT permissions on the login to the endpoint
d. Create the Availability Group, configuring the name, endpoint URL, replicas, databases, replication mode, and failover mode
e. Create the Availability Group Listener Name

```sql
--On the PRIMARY REPLICA
--a) Create login that will be used to connect to the endpoint
USE [master]
GO
CREATE LOGIN [TESTDOMAIN\sqlservice] FROM WINDOWS
GO

--b) Create endpoint
USE [master]
GO
```

```sql
CREATE ENDPOINT HADR_ENDPOINT
    STATE=STARTED
    AS TCP (LISTENER_PORT=5022)
    FOR DATA_MIRRORING (ROLE=ALL, ENCRYPTION = REQUIRED ALGORITHM AES)
GO

--c) Grant CONNECT permissions on the login to the endpoint
USE [master]
GO
GRANT CONNECT ON ENDPOINT::HADR_ENDPOINT TO [TESTDOMAIN\sqlservice];
GO

--d) Create the availability group, [AGTest]
CREATE AVAILABILITY GROUP [AGTest]
   FOR
      DATABASE [SampleDB]
   REPLICA ON
      'WS-ALWAYSON-AG1' WITH
         (
         ENDPOINT_URL = 'TCP://WS-ALWAYSON-AG1.TESTDOMAIN.local:5022',
         AVAILABILITY_MODE = SYNCHRONOUS_COMMIT,FAILOVER_MODE = AUTOMATIC
         ),
      'WS-ALWAYSON-AG2' WITH
         (
         ENDPOINT_URL = 'TCP://WS-ALWAYSON-AG2.TESTDOMAIN.local:5022',
         AVAILABILITY_MODE = SYNCHRONOUS_COMMIT, FAILOVER_MODE = AUTOMATIC
         );
GO

--e) Create the Availability Group Listener Name with virtual IP address
AND port number
USE [master]
GO

ALTER AVAILABILITY GROUP [AGTest]
ADD LISTENER N'AGTestLN' (
WITH IP
((N'172.16.0.116', N'255.255.255.0')
)
, PORT=1433);
GO
```

Once the Availability Group has been created on the primary replica, we can proceed to add the secondary replica. On the secondary replica, run the following commands to perform the following steps:

a. Create login that will be used to connect to the endpoint. For simplicity's sake, we will use the SQL Server service account for this purpose
b. Create the endpoint that the Availability Group will connect to
c. Grant CONNECT permissions on the login to the endpoint
d. Join the secondary replica to the Availability Group
e. e. Join the databases to the Availability Group

```sql
--On the SECONDARY REPLICA
--a) Create login that will be used to connect to the endpoint
```

```sql
USE [master]
GO
CREATE LOGIN [TESTDOMAIN\sqlservice] FROM WINDOWS
GO

--b) Create endpoint
USE [master]
GO
CREATE ENDPOINT HADR_ENDPOINT
    STATE=STARTED
    AS TCP (LISTENER_PORT=5022)
    FOR DATA_MIRRORING (ROLE=ALL, ENCRYPTION = REQUIRED ALGORITHM AES)
GO

--c) Grant CONNECT permissions on the login to the endpoint
USE [master]
GO
GRANT CONNECT ON ENDPOINT::HADR_ENDPOINT TO [TESTDOMAIN\sqlservice];
GO

--d) Join the replica to the existing Availability Group, AGTest
ALTER AVAILABILITY GROUP [AGTest] JOIN;
GO

--e) Join the databases to the existing Availability Group, AGTest
ALTER DATABASE [SampleDB] SET HADR AVAILABILITY GROUP = [AGTest];
GO
```

Testing the SQL Server Availability Group Failover

After the Availability Group has been created and configured, failover needs to be tested. This is the ultimate test to validate if, indeed, both TDE and the Availability Group work well in the given configuration. Failover of the Availability Group can be done via the Fail Over Availability Group Wizard from SQL Server Management Studio, similar to the screenshot below