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

Implementing Transparent Data Encryption in SQL Server 2008

Easily monitor and improve the performance of SQL Server, Azure and AWS instances with SQL Doctor. Try it for free for 14 days
START MY TRIAL
IDERA

Free Webcast - Simplify SQL Server Administration

Problem

Security is a major concern in any organization. Database backups are prone to theft and can be restored on any SQL Server instance. As we are looking through the new features in SQL Server 2008 we found a potentially interesting one called Transparent Data Encryption which we can use to encrypt our database backups. Can you give us a detailed explanation of how we go about using this new functionality?

Solution

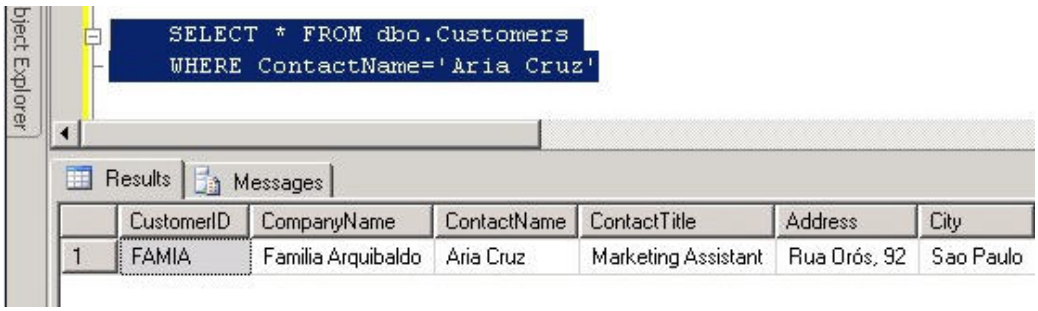
Transparent Data Encryption is a new feature in SQL Server 2008 which implements database-level encryption complementing the row-level encryption which is currently available in SQL Server 2005. This protects database access, either directly or by restoring a database backup on another SQL Server instance.

In this tip we are going to look at how to implement transparent data encryption to protect database backups. Let us first demonstrate how prone database backups are to security threats. By default, SQL Server backups are not encrypted. Let's start by creating a full backup of the **Northwind** database on my **default** instance.

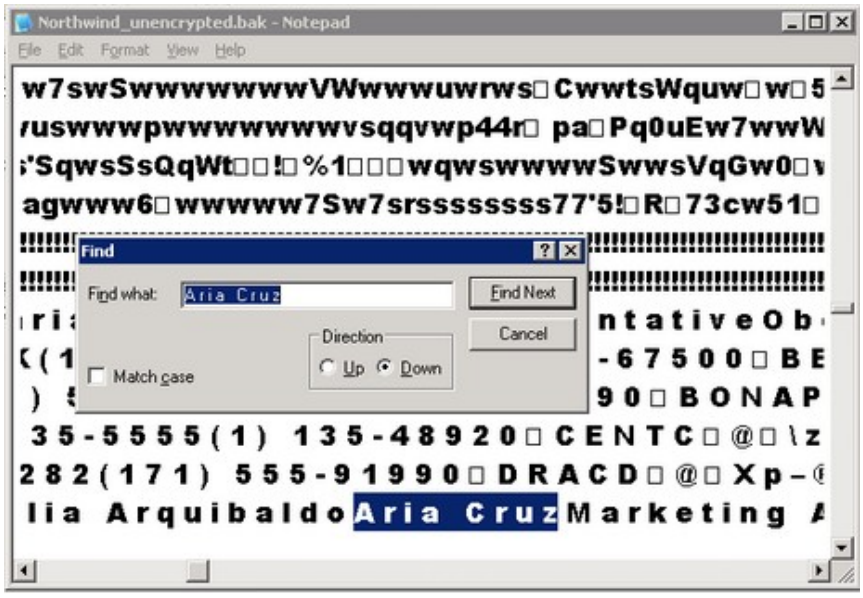
```
BACKUP DATABASE Northwind
TO DISK = 'C:\Northwind_unencrypted.bak'
WITH INIT, STATS = 10
```

Let's query the **Northwind** database for a specific record. We will use this record later on to view the contents of our database backup

```
SELECT * FROM dbo.Customers
WHERE ContactName = 'Aria Cruz'
```



Open the database backup in Notepad and search for '**A r i a C r u z**' (*note the spaces between the letters and three spaces between "Aria" and "Cruz" as this is Unicode text.*) Observe that the text data is readable.



Since your native backups are readable they are prone to security risks since they can be read in clear text. This enables someone to just look at your database backups and retrieve data from the files. What's worse is that anybody can restore your database backups on another SQL Server instance. We will demonstrate this procedure shorlty.

Let's implement transparent data encryption on our **Northwind** database by first setting up encryption on the server. To do that, we create the database master key on the **master** database

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

Then, we create the server-based certificate which will be used to encrypt the database

```
CREATE CERTIFICATE NorthwindCert
WITH SUBJECT = 'My DEK Certificate for Northwind database'
```

We will then set the encryption for the **Northwind** database by creating a database encryption key and password using the certificate we just created.

```
USE Northwind
GO
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_128
ENCRYPTION BY SERVER CERTIFICATE NorthwindCert
GO
```

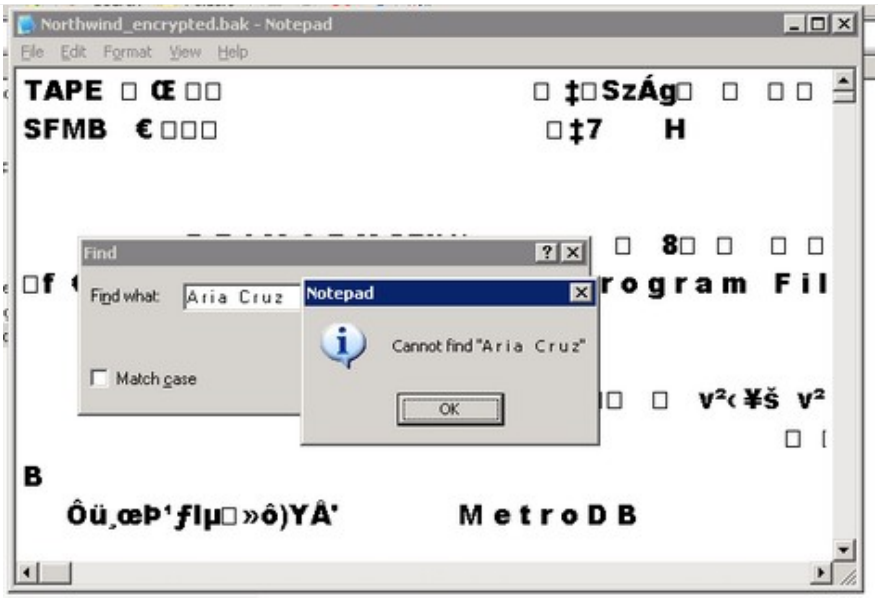
The encryption algorithm choice is yours, since a few different options exist. For guidance on selecting an encryption algorithm on SQL Server, check out this Microsoft TechNet article. After creating the database encryption key, we'll enable encryption on the database level.

```
ALTER DATABASE Northwind
SET ENCRYPTION ON
```

Let's backup the encrypted **Northwind** database this time but using a different filename.

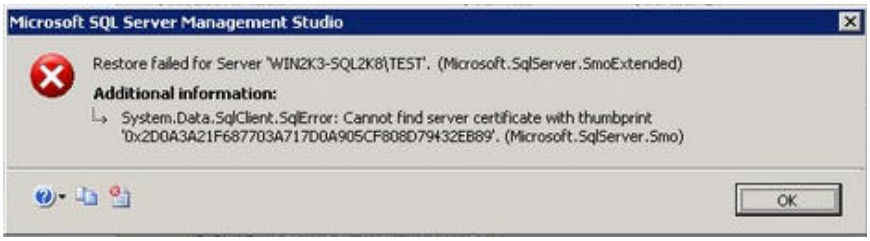
```
BACKUP DATABASE Northwind
TO DISK = 'C:\Northwind_encrypted.bak'
WITH INIT, STATS = 10
```

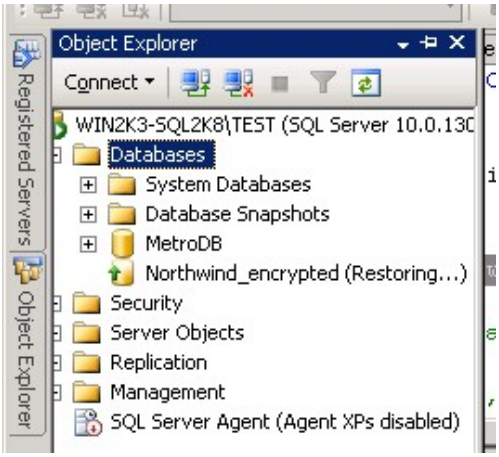
The encrypted backup would take a bit longer than the unencrypted backup. If you open the encrypted database backup and search for the string '**A r i a C r u z**' which we searched for earlier, you will no longer find it. You can scroll thru the text and notice the encrypted data which is unlike the one you'll saw in the unencrypted backup.



Restoring the encrypted database backup on another SQL Server instance

It is easy to restore the unencrypted backup on another SQL Server instance. In fact, this is what makes it prone to security risks. You can try restoring the encrypted database backup as **Northwind_encrypted** and you'll encounter an error. I am doing this on my **TEST** instance. The error message will tell you that the server certificate cannot be found. Even though the restore failed, you will still see the **Northwind_encrypted** database on the Object Explorer on a restoring state.





In order to restore the encrypted database backup on another SQL Server instance we need to first export the certificate we created on the instance on which the encrypted database backup was created. To export

the certificate to a file, I'll connect on my **default** instance and run this query.

```
USE master
GO
BACKUP CERTIFICATE NorthwindCert
TO FILE = 'C:\NorthwindCert_File.cer'
WITH PRIVATE KEY (FILE = 'C:\NorthwindCert_Key.pvk' ,
ENCRYPTION BY PASSWORD = 'mY_P@$$w0rd' )
GO
```

Then, we'll need to copy the certificate and the private key file to the other SQL Server instance. Since I am running my **default** and my **TEST** instances on the same server, I'll just refer to those files when I do the import process. Now, on the **TEST** instance, import the certificate by first creating a master key.

```
USE master
GO

CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'EnKrYpt3d_P@$$w0rd'
GO
```

The password provided here is different from the one you used in the **default** instance as you are creating a new master key for this instance. After a master key has been created, you can create a certificate by importing the certificate we created earlier.

```
CREATE CERTIFICATE NorthwindCert
FROM FILE = 'C:\NorthwindCert_File.cer'
WITH PRIVATE KEY (FILE = 'C:\NorthwindCert_Key.pvk',
DECRYPTION BY PASSWORD = 'mY_P@$$w0rd');
GO
```

Notice that the password specified here in the **DECRYPTION BY PASSWORD** parameter is the same as that which we used to export the certificate to a file. This is because we will use the same certificate on this new instance to access the encrypted database. After the certificate has been created, we can now restore the encrypted database backup on this instance. You might want to drop the initial encrypted database which you attempted to restore before proceeding. You will now be able to restore the encrypted database backup successfully.

Next Steps

- You have seen the following items:
  - Native backups are in clear text
  - How you can secure your SQL Server 2008 database and backups using transparent data encryption
  - How to create a certificate and back it up in order to complete the restore process
- Download a copy of the latest Community Technology Preview of SQL Server 2008 from this site. The above examples were created using the February 2008 CTP.
- You can get the Northwind database used in the sample here.
- To view the SQL Server 2008 Books Online content on Transparent Data Encryption to this site.

Last Update: 2008-05-28

You're invited
to our next free
SQL Server webcast!

click to learn more

Next Tip

Click Here

About the author

Edwin M Sarmiento is a Microsoft SQL Server MVP and Microsoft Certified Master from Ottawa, Canada specializing in high availability, disaster recovery and system infrastructures.

View all my tips

Related Resources

- How to simulate transparent data encryption TDE wi...
- Implementing Transparent Data Encryption in SQL Se...
- SQL Server 2008 Transparent Data Encryption gettin...
- Configuring Transparent Data Encryption with SQL S...
- SQL Server Transparent Data Encryption (TDE) Perfo...
- Recovering a SQL Server TDE Encrypted Database Suc...
- More Database Developer Tips...

Post a comment or let the author know this tip helped.

All comments are reviewed, so stay on subject or we may delete your comment. Note: your email address is not published. Required fields are marked with an asterisk (*).

*Name *Email Notify for updates

*** NOTE *** - If you want to include code from SQL Server Management Studio (SSMS) in your post, please copy the code from SSMS and paste the code into a text editor like NotePad before copying the code below to remove the SSMS formatting.

p

Send me SQL tips:

I'm not a robot

reCAPTCHA
Privacy - Terms

Save Comment

Thursday, January 02, 2014 - 12:30:18 AM - Chhavi_MCITP Back To Top

This is awesome, I respect and appreciate this way of writing.... Very useful, many thanks!!

Tuesday, June 11, 2013 - 2:54:14 AM - Sarang Back To Top

An Excellent Tutorial, precised in really a good manner... Thanks buddy for this awesome article with Demo...

Regards,

Sarang M Tumne,

Information Security Consultant.