Db2 Blog Post Template

Use this template to draft your Db2 blog post for the editing process. You can use or exclude sections as necessary for your post.

**Title:**

Encrypting a Db2 Database with Minimal Downtime using HADR

**Authors**: Thomas Drescher

# Introduction

Database encryption to ensure your data is protected at rest is nowadays considered mandatory for all enterprises. One of the challenges for Db2 users is that the easiest way to adopt [Db2 native encryption](https://www.ibm.com/docs/en/db2/12.1?topic=encryption-overview) for an existing database requires unloading and reloading the data, most commonly applied using [Db2 native backup and restore](https://www.ibm.com/docs/en/db2/12.1?topic=encryption-encrypted-backups), and this could incur significant downtime, dependent on the size of your database. In this blog, we will discuss the steps to use an alternative approach for any databases where the downtime incurred by the full restore is unacceptable due to the availability requirements of the applications using the database. This alternative approach is based on the use of Db2 high availability disaster recovery feature (HADR), a well well-known feature in Db2 that is commonplace in all environments that have to provide high availability or disaster recovery capabilities for the databases. The use of HADR ensures that the application can continue to access the database on the Primary HADR hosts while the encryption is taking place on the Standby HADR hosts, ensuring a seamless encryption operation.

In the process that we will describe we are also including the setup of an HADR environment just for the purpose of encrypting the database. This process only incurs two very short downtime phases when the HADR primary and standby are switched, and the connectivity configuration of your application is changed. Since the HADR setup described here is only meant to be used temporarily, we will not use a cluster manager for automatic failover handling, but of course, if this is a permanent HADR setup and the highest availability is expected, it can be automated using [Db2’s integrated HADR automation](https://www.ibm.com/docs/en/db2/12.1?topic=server-db2-high-availability-feature) to make this seamless to the applications.

As a starting point, there is an unencrypted database running on Host A. To encrypt the existing database, take a backup of the database on Host A and restore it as an encrypted database onto an HADR standby on Host B. Continue to complete the HADR setup by starting HADR operations on Host A.

In the first downtime phase, you instruct the HADR standby database to take over as the new HADR primary running on Host B. Since we have not setup automatic failover with a cluster manager, you have to manually adapt the database connectivity information of your application and restart it.

With your application up and running, you can build a new encrypted HADR standby database on Host A while your primary database runs on Host B.

In the second downtime phase, you instruct the encrypted HADR standby database on Host A to take over as the new HADR primary database and switch back the connectivity information of your application to Host A.

Finally, you clean up the temporary HADR pair and then you are done.

The full procedure is described in more detail in the following sections.

# Preparation

Make sure your database on Host A is configured with archive logging as this is a prerequisite for HADR.

As a starting point, we have installed the same Db2 version on a second host (Host B), which we will use during the process to encrypt our existing database on Host A. We have also created a Db2 instance on Host B and adapted the database manager configuration to match the Db2 instance on Host A.

For the purposes of this example, the database is PRD and the instance owner is user db2prd, with /db2/db2prd as its home directory. Host A is db6x13l020, and Host B is db6x13l021.

1. Now create an online backup of your unencrypted database on Host A which will be used to create the HADR standby database on Host B.

# as the instance owner on Host A  
db2 backup database PRD online to /db2/db2prd/backup.Host\_A/online.backup compress include logs

# Create the HADR standby database with Db2 native encryption enabled

## Configure Db2 native encryption on Host B

We have to create and configure a keystore and place the master key in the keystore. Then we have to configure the Db2 instance to use the keystore. To do so, follow the next steps as the instance owner.

1. Create a directory for the keystore. Then create the keystore which will contain the keys.

# on Host B, as the instance owner   
mkdir ~/keystore  
gsk8capicmd\_64 -keydb -create -db ~/keystore/db\_encr.p12 -pw "secret\_password" -type pkcs12 -stash

1. Next, create a secret key and verify that the keystore is populated. This key will be used to restore the database into an encrypted database.

# on Host B, as the instance owner  
gsk8capicmd\_64 -secretkey -create -db ~/keystore/db\_encr.p12 -stashed -label my\_enc\_key -size 16

gsk8capicmd\_64 -cert -list all -db ~/keystore/db\_encr.p12 -stashed  
  
Certificates found  
\* default, - personal, ! trusted, # secret key  
# my\_enc\_key

1. Having prepared the keystore and the keys, update the database manager configuration to use the encryption assets.

# on Host B, as the instance owner   
db2 update dbm cfg using KEYSTORE\_LOCATION /db2/db2prd/keystore/db\_encr.p12  
db2 update dbm cfg using KEYSTORE\_TYPE pkcs12

Make the backup taken on Host A available on Host B. Restore the unencrypted backup into an encrypted database on Host B. Specify your previously created master key for the restore.

1. Restore on Host B as an encrypted database and verify that the database is encrypted.

# on Host B, as the instance owner  
db2 restore database PRD from /db2/db2prd/backup.Host\_A/online.backup ENCRYPT MASTER KEY LABEL 'my\_enc\_key'  
  
DB20000I The RESTORE DATABASE command completed successfully.

db2 get db cfg for PRD | grep -i encr

Encryption Library for Backup (ENCRLIB) = libdb2encr.so  
Encryption Options for Backup (ENCROPTS) = CIPHER=AES:MODE=CBC:KEY LENGTH=256  
Encrypted database = YES

## Initializing HADR

1. Set the HADR specific configuration parameters in the database configuration on Host B.

# on Host B, as the instance owner  
db2 "update db cfg for PRD using \  
 HADR\_LOCAL\_HOST db6x13l021 \  
 HADR\_REMOTE\_HOST db6x13l020 \  
 HADR\_LOCAL\_SVC db2prd\_hadr \  
 HADR\_REMOTE\_SVC db2prd\_hadr \  
 HADR\_REMOTE\_INST db2prd"

1. Next, start HADR standby operations on the database on Host B and verify that HADR is running.

# on Host B, as the instance owner   
db2 START HADR ON DB PRD AS STANDBY

# run command db2pd and check the output. Specifically, the values of HADR\_ROLE and HADR\_STATE should be similar to the following:   
  
db2pd -hadr -db PRD | grep 'HADR\_ROLE\|HADR\_STATE'

HADR\_ROLE = STANDBY  
 HADR\_STATE = REMOTE\_CATCHUP\_PENDING

1. Next, set the HADR specific configuration parameters in the database configuration on Host A.

# on Host A, as the instance owner   
db2 "update db cfg for PRD using \  
 HADR\_LOCAL\_HOST db6x13l020 \  
 HADR\_REMOTE\_HOST db6x13l021 \  
 HADR\_LOCAL\_SVC db2prd\_hadr \  
 HADR\_REMOTE\_SVC db2prd\_hadr \  
 HADR\_REMOTE\_INST db2prd"

1. Now start HADR primary operations on the database on Host A and verify that HADR is running.

# on Host A, as the instance owner   
db2 START HADR ON DB PRD AS PRIMARY  
  
DB20000I The START HADR ON DATABASE command completed successfully.

db2pd -hadr -db PRD | grep 'HADR\_ROLE\|HADR\_STATE'

HADR\_ROLE = PRIMARY  
 HADR\_STATE = PEER

At this point, we have an HADR system up and running with the unencrypted primary database on Host A and the encrypted standby database on Host B.

# Switch the database host - This is a DOWNTIME phase

In this down time phase, we will configure our application to connect to the HADR database on Host B. For this, you will have to adapt the connectivity information and switch the primary and standby database in your HADR environment.

1. Stop your application.
2. Change the connectivity information of your application.  
   Example db2cli.ini:

[PRD]  
Database=PRD  
Protocol=tcpip  
**Hostname=<Host\_A> <= replace here with <Host B**>  
Servicename=5912

Now we can switch the HADR roles of our databases.

1. Initiate the takeover.

# as the instance owner on Host B  
db2 takeover hadr on db PRD  
  
DB20000I The TAKEOVER HADR ON DATABASE command completed successfully.

1. Start your application, which will now connect to the encrypted database on Host B

This is the end of the first downtime phase.

# Replace the database on Host A with an encrypted database

Now we want to encrypt our database on Host A. To do this, we will stop HADR operations, drop the unencrypted database, and create a new encrypted standby database.

1. Take a new backup on Host B.

# on Host B, as the instance owner  
mkdir -p backup.Host\_B/online.backup  
db2 backup database PRD online to /db2/db2prd/backup.Host\_B/online.backup encrypt include logs

1. Stop the HADR operations on your current primary database on Host B.

# on Host B, as the instance owner  
db2 stop hadr on db PRD  
  
DB20000I The STOP HADR ON DATABASE command completed successfully.

1. Stop the HADR operations on your current standby database on Host A and drop the unencrypted database.

# on Host A, as the instance owner  
db2 deactivate db PRD  
DB20000I The DEACTIVATE DATABASE command completed successfully.  
  
db2 stop hadr on db prd  
DB20000I The STOP HADR ON DATABASE command completed successfully.

db2 drop db PRD  
DB20000I The DROP DATABASE command completed successfully.

1. Copy the keystore file used for encryption from Host B to Host A.

# as the instance owner on host B  
scp keystore/db\_encr.\* db2prd@db6x13l020:/db2/db2prd/keystore

(db2prd@db6x13l020) Password:  
db\_encr.p12 100% 3424 5.1MB/s 00:00  
db\_encr.sth 100% 193 631.7KB/s 00:00

1. Update the database manager configuration to use the encryption assets on Host A

# as the instance owner on host A  
db2 update dbm cfg using \  
 KEYSTORE\_LOCATION /db2/db2prd/keystore/db\_encr.p12 \  
 KEYSTORE\_TYPE pkcs12

Make your backup taken on Host B available on your Host A and restore the backup into an encrypted database on Host A. Specify your previously created master key for the restore.

1. Restore the database on Host A and adapt the HADR settings.

# as the instance owner   
db2 restore database PRD from /db2/db2prd/backup.Host\_B/online.backup encrypt MASTER KEY LABEL 'my\_enc\_key'  
  
DB20000I The RESTORE DATABASE command completed successfully.

1. Set the HADR specific configuration parameters in the database configuration on Host A.

# as the instance owner   
db2 "update db cfg for PRD using \  
 HADR\_LOCAL\_HOST db6x13l020 \  
 HADR\_REMOTE\_HOST db6x13l021 \  
 HADR\_LOCAL\_SVC db2prd\_hadr \  
 HADR\_REMOTE\_SVC db2prd\_hadr \  
 HADR\_REMOTE\_INST db2prd"

1. Start HADR standby operations on the database on Host A.

# as the instance owner   
db2 START HADR ON DB PRD AS STANDBY  
DB20000I The START HADR ON DATABASE command completed successfully.  
  
db2pd -hadr -db PRD | grep 'HADR\_ROLE\|HADR\_STATE'

HADR\_ROLE = STANDBY  
 HADR\_STATE = REMOTE\_CATCHUP\_PENDING

1. Start HADR primary operations on the database on Host B.

# as the instance owner  
db2 START HADR ON DB PRD AS PRIMARY  
  
db2pd -hadr -db PRD | grep 'HADR\_ROLE\|HADR\_STATE'

HADR\_ROLE = PRIMARY  
 HADR\_STATE = PEER

# Switch the database host back - This is a DOWNTIME phase

1. Stop your application and switch the connectivity information back to Host A.

Example db2cli.ini:

[PRD]  
Database=PRD  
Protocol=tcpip  
**Hostname=<Host\_B> <= replace here with <Host A>**  
Servicename=5912

1. Switch the HADR roles. The primary database will now be on Host A.

# as the instance owner on Host A  
db2 takeover hadr on db PRD  
DB20000I The TAKEOVER HADR ON DATABASE command completed successfully.  
  
db2pd -hadr -db PRD | grep 'HADR\_ROLE\|HADR\_STATE'  
 HADR\_ROLE = PRIMARY  
 HADR\_STATE = PEER

1. Start your application and check whether it can connect to the database.

This is the end of the second downtime phase.  
Your database on Host A is now encrypted, and your application running again.

In this example, we ignored the log files that are created while Host B has the primary role in the HADR setup. If you want to have a continuous log stream, you must make sure to archive the logs created on Host B.  
The HADR setup will also work without shared log archives, but manual intervention may be required in certain operations. If the archived logs are not accessible to both the HADR primary and standby hosts, some processes may take longer, e.g., remote catchup or the manual copying of log files. For more information see [Log archiving configuration for Db2 high availability disaster recovery (HADR) - IBM Documentation](https://www.ibm.com/docs/en/db2/12.1?topic=hadr-log-archiving-configuration) and [Db2 HADR Wiki](https://ibm.github.io/db2-hadr-wiki/hadrLogShipping.html)

# Clean up the HADR setup

Since we used HADR in this scenario only to reduce the downtime for encrypting your database, we can stop HADR operations und remove the standby database.

1. Stop HADR operations on Host A.

# as the instance owner on Host A  
db2 stop hadr on db PRD  
DB20000I The STOP HADR ON DATABASE command completed successfully.

1. Remove the HADR parameters from the database configuration.

# as the instance owner on Host A  
db2 update db cfg for PRD using \  
 HADR\_LOCAL\_HOST NULL \  
 HADR\_LOCAL\_SVC NULL \  
 HADR\_REMOTE\_HOST NULL \  
 HADR\_REMOTE\_SVC NULL \  
 HADR\_REMOTE\_INST NULL

1. Stop HADR operations and drop the database on Host B

# as the instance owner on Host B  
db2 deactivate db PRD  
  
db2 stop hadr on database PRD  
DB20000I The STOP HADR ON DATABASE command completed successfully.  
  
db2 drop db prd  
DB20000I The DROP DATABASE command completed successfully.

1. You can drop the Db2 instance and uninstall the Db2 software copy on Host B to complete the cleanup

**Conclusion**

In the procedure described above, we used a temporary HADR setup to convert an unencrypted database into an encrypted database with only two short downtimes. This reduces the overall downtime when compared to the procedure of using one host, where the time needed for restoring the database is also downtime. The longer the restore procedure takes, the greater the advantage you have with this approach.

You can also apply the ideas in this blog to an existing HADR setup. If you do so, then you might consider using multiple standbys to ensure the high availability of your system.

**About the Authors**

Thomas Drescher joined IBM in 1997. He has been working on the SAP on Db2 LUW development team since 1998. After working on some initial projects, he joined the SAP Db2 Lifecycle Management Team and was involved in the development of SAP's Software Provisioning Manager and lead the integration of Db2 features like pureScale, TLS, and Db2 native encryption into the SAP Lifecycle Tools. Currently he is the team lead of the SAP Db2 Lifecycle Management Team.

**Further reading**

<https://ibm.github.io/db2-hadr-wiki/>

<https://www.ibm.com/docs/en/db2/12.1?topic=availability-initializing-hadr>

<https://www.ibm.com/docs/en/db2/12.1?topic=rest-db2-native-encryption>  
  
<https://www.ibm.com/docs/en/db2/12.1?topic=hadr-log-archiving-configuration>

<https://ibm.github.io/db2-hadr-wiki/hadrLogShipping.html>