DB2 native encryption

Last Updated: 2021-03-01

28

DB2® native encryption encrypts your DB2 database, requires no hardware, software, application, or schema changes, and provides transparent and secure key management.

Encryption is the process of transforming data into an unintelligible form in such a way that the original data either cannot be obtained or can be obtained only by using a decryption process. It is an effective way of protecting sensitive information that is stored on media or transmitted through untrusted communication channels. Encryption is mandatory for compliance with many government regulations and industry standards.

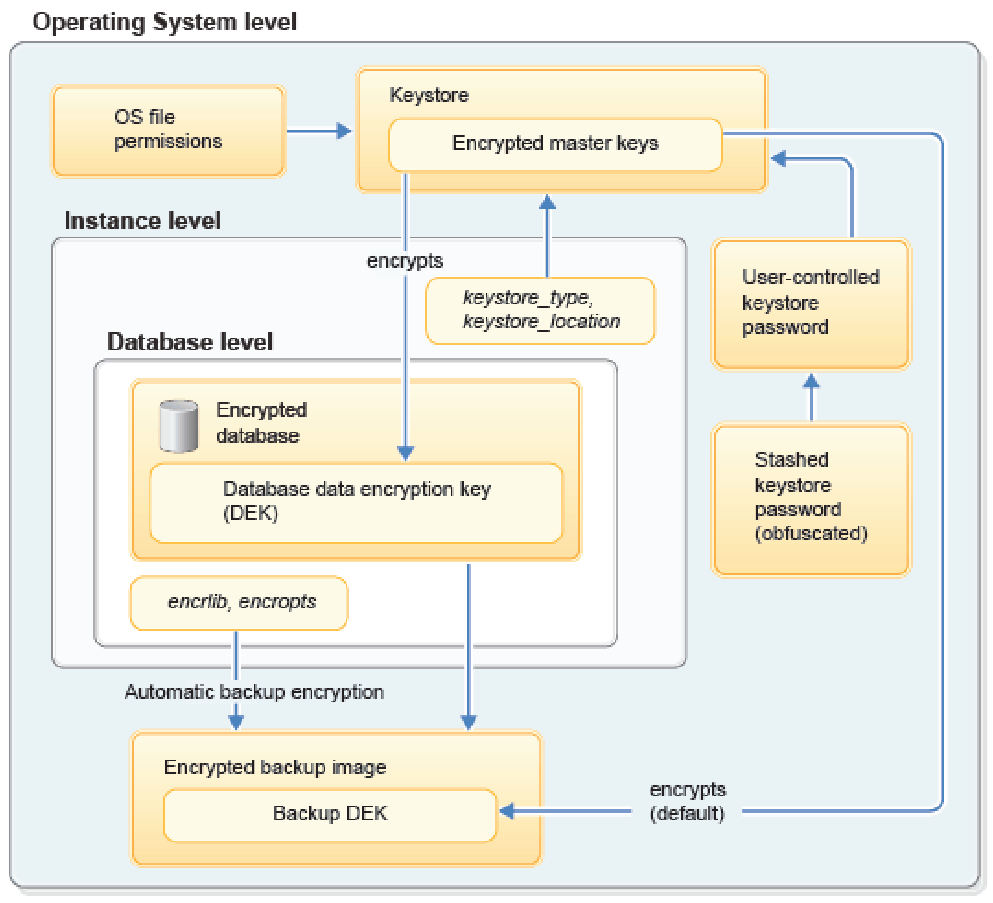
In an encryption scheme, the data requiring protection is transformed into an unreadable form by applying a cryptographic algorithm and an encryption key. A *cryptographic algorithm* is a mathematical function that is used in encryption and decryption processes. An *encryption key* is a sequence that controls the operation of a cryptographic algorithm and enables the reliable encryption and decryption of data.

Some data encryption solutions for protecting data at rest are suitable in cases of physical theft of disk devices, and some can protect against privileged user abuse. With *native database encryption*, the database system itself encrypts the data before it calls the underlying file system to write that data to disk. This means that not only your current data is protected, but also data in new table space containers or table spaces that you might add in the future. Native database encryption is suitable for protecting data in cases of either physical theft of disk devices or privileged user abuse.

A local or external key manager is typically used to manage the keys. A *database data encryption key* (DEK) is the encryption key with which actual user data is encrypted. A *master key* is a "key encrypting key": It is used to protect the DEK. Although the DEK is stored and managed by the database, the master key is stored and managed outside of the database.

These keys are shown in Figure 1, which provides an overview of DB2 native encryption.

Figure 1. An overview of DB2 native encryption



A new master key is automatically added when you create an encrypted database without specifying the `MASTER KEY LABEL` option on the **`CREATE DATABASE`** command. The database manager uses this master key by default, but you can optionally add a different master key.

28

Encrypted master keys are stored in a PKCS#12-compliant *keystore*, which is a storage object for encryption keys that exists at the operating system level. In partitioned database environments or DB2 pureScale® environments, the keystore location must be accessible to all members. There is at most one keystore per DB2 instance. The **`keystore_type`** and **`keystore_location`** database manager configuration parameters are used to specify the type and location of the keystore.

The keystore password (in obfuscated form) can be stashed to a file that automatically provides the password when required. The stash file can be read by only the file owner. Not stashing the password enhances security if the instance owner account becomes compromised. However, this additional security must be weighed against any requirements that the DB2 instance can start without human intervention. If the password is not stashed, you cannot access an encrypted database until you provide the keystore password.

Database backup images are automatically encrypted if the **`encrlib`** and **`encropts`** database configuration parameters are set to a non-null value. The encrypted master key encrypts the backup DEK by default.

DB2 Version 10.5 Fix Pack 5 adds native database encryption to the DB2 database server. This enhancement is easy to implement and provides secure local key management that is based on Public Key Cryptography Standard #12 (PKCS#12). DB2 native encryption enables you to meet compliance requirements in a cost effective manner.

|<

Encrypting and decrypting user-managed data sets using z/OS DFSMS data set encryption

Last Updated: 2025-01-07

To encrypt data sets for user-managed table spaces or index spaces, use IDCAMS to define the shadow data sets with the KEYLABEL option.

Procedure

28

1. Use the DFSMS Access Method Services DEFINE CLUSTER command to define the shadow data sets for the table space and index spaces with the appropriate KEYLABEL option.
2. Run the REORG utility with the SHRLEVEL CHANGE option on the table space.

Parent topic: |<

→ Encrypting your data with z/OS DFSMS data set encryption

>|

Related concepts

|<

→ Types of DB2 data sets that can be encrypted by using DFSMS data set encryption

>|

|<

→ Encrypted FlashCopy image copies, copies made with DFSMSdss concurrent copy, and system-level backups

>|

Related tasks

|<

→ Encrypting log, catalog, and directory data sets with z/OS DFSMS data set encryption

>|

|<

→ Encrypting all data sets using a storage group with z/OS DFSMS data set encryption

>|

|<

Related reference

→ REORG INDEX