

Version 1 Release 2

*IBM InfoSphere Guardium Data
Encryption for DB2 and IMS Databases
User's Guide*



Version 1 Release 2

*IBM InfoSphere Guardium Data
Encryption for DB2 and IMS Databases
User's Guide*



Note:

Before using this information and the product it supports, read the "Notices" topic at the end of this information.

Sixth Edition (2017)

This edition applies to Version 1 Release 2 of (product number 5655-P03) and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 2017; Copyright Rocket Software Inc., 2017.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this information v

Chapter 1. InfoSphere Guardium Data

Encryption overview 1

What does InfoSphere Guardium Data Encryption do? 1

Data governance solutions 1

Encryption and decryption overview 2

Encryption methods 4

DB2 encryption and decryption 9

IMS encryption and decryption 15

Encryption with compression 18

InfoSphere Guardium Data Encryption

documentation and updates 19

Accessibility features 19

Summary of changes 20

Chapter 2. InfoSphere Guardium Data

Encryption installation information . . . 23

Prerequisites 23

Hardware requirements 23

Software requirements 24

Performance considerations 25

| Setting up the Guardium Data Encryption
| subsystem 25

Chapter 3. Encrypting data in DB2 . . . 27

Comparison of DB2 encryption methods 27

Creating a DB2 encryption method 29

Creating a DB2 User Defined Function for
encryption 30

Creating a DB2 field procedure for encryption . . 35

Creating a DB2 edit procedure for encryption . . 37

Implementing a DB2 encryption method 40

Implementing a DB2 UDF for encryption 40

Implementing a DB2 field procedure for
encryption 44

Implementing a DB2 edit procedure for
encryption 45

Chapter 4. Encrypting data in IMS . . . 51

Creating an IMS encryption exit routine 51

Creating an IMS encryption exit routine by
editing a sample job 51

Creating an IMS encryption exit routine by using
the ISPF interface 52

Encrypting IMS data without compression 55

Encrypting IMS data with compression 56

Creating an IMS exit routine driver 57

Implementing IMS data encryption with
compression 59

Chapter 5. Upgrading the cryptographic key encryption 61

Chapter 6. Troubleshooting 63

IMS abend codes 63

IMS reason codes 63

Guardium Data Encryption subsystem messages . . 67

DB2 reason codes 68

DECZLDX0 70

IMS and DB2 batch TSO job step messages 70

Common ICSF reason codes 71

DB2 SQL codes 72

DECENF00 72

DECENU00, DECENUBL, DECENUI0,

DECENUP0, and DECENURN 73

IMS exit messages 73

Gathering diagnostic information 74

Notices 77

Trademarks 78

Terms and conditions for product documentation . . 78

Privacy policy considerations 79

Index 81

About this information

IBM® InfoSphere® Guardium Data Encryption for DB2® and IMS™ Databases (also referred to as InfoSphere Guardium Data Encryption) enables you to meet the demand for data privacy and security.

Always check the IBM Knowledge Center for the most current version of this information:

https://www.ibm.com/support/knowledgecenter/SSNPLQ_1.2.0/decuhome.html

Chapter 1. InfoSphere Guardium Data Encryption overview

InfoSphere Guardium Data Encryption protects sensitive and private data, and minimizes the liability risks that are associated with data governance.

What does InfoSphere Guardium Data Encryption do?

InfoSphere Guardium Data Encryption addresses the increased demand for data privacy and security.

InfoSphere Guardium Data Encryption performs encryption and decryption through the use of exit routines. The exit routine code uses the System z[®], zSeries, and S/390[®] Crypto Hardware to encrypt data for storage and to decrypt data for application use. InfoSphere Guardium Data Encryption protects sensitive data that can reside on various storage media.

The exit routines can be DB2 edit procedures, field procedures, user-defined functions, or IMS Segment Edit/Compression exit routines.

Product features

- A single tool for your DB2 and IMS databases
- High performance and low overhead by using the cryptographic hardware available on the platform
- Compliance with privacy and security regulations
- Customization at the DB2 table level and at the IMS segment level
- Fast implementation, after a cryptographic key label has been defined by the security analyst, with the use of standard DB2 and IMS exit routines

Product benefits

- Ensures data privacy by encrypting and decrypting data
- Uses the following encryption algorithms:
 - Triple Data Encryption Algorithm (TDEA), also known as the Triple Data Encryption Standard (Triple DES)
 - ANSI Data Encryption Algorithm (DEA), also known as the Data Encryption Standard (DES)
 - Advanced Encryption Standard (AES)
- Requires no changes to your applications
- Conforms to the existing z/OS[®] and OS/390[®] security model
- Provides an ISPF front end to create and customize encryption exit routines
- Enables you to leverage the power of Storage Area Networks (SANs) safely while complying with privacy and security regulations

Data governance solutions

IBM DB2 and IMS Tools offers data governance solutions that respond to ongoing requirements that are related to the auditing, retention, and privacy of your data.

With data compliance and privacy regulations on the rise, many IT organizations are experiencing new levels of complexity around data governance. The DB2 and

IMS Tools provide IT organizations with the capabilities that they need to more confidently comply with regulations while saving time and expense in the data center.

IBM solutions help IT organizations maximize their investment in DB2 and IMS databases while staying on top of some of today's toughest IT challenges such as performance management, data warehousing, and data governance.

If you are responsible for the data governance of DB2 and IMS databases, consider the following questions, which might expose some challenges for you and your organization:

- Is your DB2 audit reporting strategy lacking institutional controls?
- Are you using *live* production data for unit testing, with no masking of sensitive data values?
- Are you wasting resources storing large amounts of unreferenced and inactive data on your operational databases?
- Is sensitive data potentially being exposed to theft while at rest or in transit between you and your business partners?

InfoSphere Guardium Data Encryption is only one part of IBM's suite of regulatory compliance tools that help minimize the liability risks that are associated with data governance.

Other regulatory compliance tools that can assist with data governance include:

DB2 Tools

- Security Guardium® S-TAP® for DB2
- InfoSphere Guardium S-TAP for DB2
- DB2 Audit Management Expert
- DB2 Data Archive Expert
- DB2 Test Database Generator (data masking)
- IBM Database Encryption Expert
- Optim™ Data Growth
- Optim Data Privacy

IMS Tools

- Security Guardium S-TAP for IMS
- InfoSphere Guardium S-TAP for IMS
- IMS Audit Management Expert

To learn more about data governance solutions, see the following website:

<http://www.ibm.com/software/data/db2imstools/solutions/data-governance.html>

Encryption and decryption overview

InfoSphere Guardium Data Encryption performs two main operations: encryption and decryption.

As shown in the following figure, during encryption DB2 or IMS application data ("PAUL") is converted to database data that is unintelligible ("x@vg"). Only the

person with the cryptographic key label can decrypt the data. The cryptographic key label is assigned by your security administrator.

Decryption is the opposite process. Data is taken from the database ("x@vg") and converted back to its original form ("PAUL").

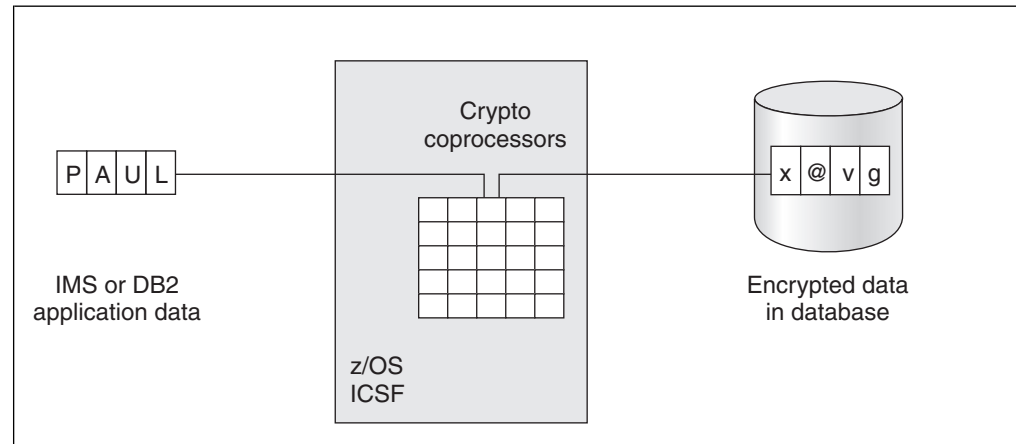


Figure 1. Concept of encryption and decryption

You can implement data encryption by using an encryption method, which can be a DB2 edit procedure, field procedure, or User Defined Function, or an IMS Segment Edit/Compression exit routine. This encryption method can be called by a DB2 table or an IMS segment. The data is encrypted or decrypted each time an application processes the table or segment. This provides a higher level of data protection than performing encryption at the database level because the data remains encrypted even when it is in the buffer pool not actively being accessed.

You can use different encryption methods for different tables or segments. For example, in IMS, a financial application segment can use one encryption method and a personnel segment can use a different encryption method.

Data remains encrypted during channel I/O. However, when the DBMS gains control, it starts the encryption method to decrypt the data (read processing) and encrypt the data (write processing).

For DB2, log records, image copies, and data buffers are encrypted. For IMS, image copies, data buffers, and log records that log changes to database records are encrypted. IMS data that is decrypted is also logged.

To implement InfoSphere Guardium Data Encryption, the following tasks must be done. Steps 3 and 6 are documented in this information. The remaining steps use other IBM products and processes and are outside the scope of this information.

1. Set up and validate Integrated Cryptographic Service Facility (ICSF). The setup process consists of installing the hardware configuration data and setting the system master key.
2. Generate a cryptographic key label for use with the table or segment. Store the cryptographic key label in the cryptographic key data set (CKDS).
3. Build the encryption method by using InfoSphere Guardium Data Encryption. This method must specify the generated encryption key label.
4. Back up your data.
5. Unload your data.

6. Create and install the encryption method.
7. Reload the data, during which process the data is encrypted.
8. Validate your output.

Related concepts:

“Encryption methods”

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related information:

 [z/OS Internet Library](#)

For more information about setting up ICSF, see *Cryptographic Services ICSF System Programmer's Guide*. For information about generating cryptographic keys, see *Cryptographic Services ICSF Administrator's Guide*.

Encryption methods

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

The encryption method that you create is linked with one of the different InfoSphere Guardium Data Encryption encryption methods that are provided with the product. The different InfoSphere Guardium Data Encryption encryption methods use Integrated Cryptographic Service Facility (ICSF), callable services, or z/Architecture Cipher instructions to encrypt the data. Data is encrypted by using a key that is managed by ICSF.

Cryptographic key encryption methods

Each InfoSphere Guardium Data Encryption encryption method uses ICSF callable services to support one or more of the following encryption methods for the cryptographic key label:

CPACF protected key

A high performance key encryption method type that is available on IBM System z10[®] Enterprise Class GA3 and later mainframes. A CPACF protected key is not visible to applications or to the operating system during encryption.

Clear key with CPACF protected key wrapping

A high performance key encryption method type that is available on IBM System z10 Enterprise Class GA3 and later mainframes. A clear key with CPACF protected key wrapping uses an ICSF defined clear key, which is encrypted by using a CPACF instruction and an LPAR wrapping key. The resulting encrypted token is available in the user address space.

Clear key

A key type that is available on IBM zSeries z990, IBM zSeries z890, and later mainframes. A clear key is not encrypted under another key.

Secure key

A key that is encrypted under a master key. The secure key never exists unencrypted outside of the cryptographic coprocessor.

Secure key with CPACF protected key wrapping

A high performance key encryption method type that is available on IBM System z10 Enterprise Class GA3 and later mainframes. This option is

made available through the application of the PTF for APAR OA50450 applied to FMID HCR77B1, HCR77B0, HCR77A1, and HCR77A0.

A secure key with CPACF protected key wrapping uses an ICSF defined secure key, which is encrypted by using a CPACF instruction and an LPAR wrapping key. The resulting encrypted token is available in the user address space.

RACF enables you to restrict access to ICSF managed keys and authorize an ICSF-defined secure key to be used as an ICSF protected key. InfoSphere Guardium Data Encryption processing has no control over the security environment that is used when ICSF performs an authorization check. In some cases, the security environment that is used for the authorization check will be different from the security environment that is associated with the user who makes the request. For more information about how to use RACF to authorize users of specific key labels, see *Using RACF to Protect Keys and Services* on the IBM Knowledge Center: https://www.ibm.com/support/knowledgecenter/SSLTBW_2.1.0/com.ibm.zos.v2r1.csfb300/ctl.htm#ctl.

For more information about protected keys, see the *z/OS Cryptographic Services ICSF Administrator's Guide, Enabling use of encrypted keys in Symmetric Key Encipher and Symmetric Key Decipher callable services*: https://www.ibm.com/support/knowledgecenter/SSLTBW_2.1.0/com.ibm.zos.v2r1.csfb300/enuenc.htm.

For more information about creating keys by using KGUP, see the *z/OS Cryptographic Services ICSF Administrator's Guide, Using KGUP Panels*: https://www.ibm.com/support/knowledgecenter/SSLTBW_2.2.0/com.ibm.zos.v2r2.csfb300/csfb300_Using_KGUP_Pa

The following tables show the relationships between the InfoSphere Guardium Data Encryption encryption methods and the types of cryptographic key label encryption. Performance results might vary in your environment.

Table 1. Key encryption methods provided by DB2 edit procedures

InfoSphere Guardium Data Encryption exit routine for DB2	Key encryption	Sample member	Performance
DECENA00	Clear key	DECDB2CK	Lowest overhead, best performance
DECENAA0	CPACF-wrapped secure or clear key	DECDB2XK	Lowest overhead, best performance
DECENB00	CPACF protected key	DECDB2CL	Low overhead, good performance
DECENBI0	CPACF protected key plus unique Initial Chaining Vector (ICV) generation	DECDB2CL	Low overhead, good performance
DECENC00	Secure key	DECDB2JB and DECDB2SK	Most overhead, most latency
DECENCA0	Secure key plus AES	DECDB2JB	Most overhead, most latency

Table 2. Key encryption methods provided by DB2 field procedures

InfoSphere Guardium Data Encryption exit routine for DB2	Key encryption	Sample member	Performance
DECENF00	CPACF protected key	DECDBFCL	Low overhead, good performance

Table 3. Key encryption methods provided by DB2 User Defined Function

InfoSphere Guardium Data Encryption exit routine for DB2	Key encryption	Sample member	Performance
DECENU00	CPACF protected key	DECDB2UD	Low overhead, good performance
DECENU10	CPACF protected key	DECDB2UD and DECUXUDF (sample SQL statements)	Low overhead, good performance
DECENUP0	CPACF protected key	DECDB2UD	Low overhead, good performance
DECENUBL	CPACF protected key	DECDB2UD	Low overhead, good performance

Table 4. Key encryption methods provided by IMS exit routines

InfoSphere Guardium Data Encryption exit routine for IMS	Key encryption	Sample member	Performance
DECENA01	Clear key	DECIMSCK	Lowest overhead, best performance
DECENAA1	CPACF-wrapped secure key or clear key with CPACF protected key wrapping Batch ICSF CHECKAUTH recurring bypass	DECIMSCB	Low overhead, good performance
DECENB01	CPACF protected key	DECIMSCB	Low overhead, good performance
DECENBB1	CPACF-wrapped secure key or CPACF protected key with batch ICSF CHECKAUTH recurring bypass	DECIMSCB	Low overhead, good performance
DECENC01	Secure key	DECIMSJJB	Most overhead, most latency

If you are operating on an earlier mainframe than the System z10 Enterprise Class GA3, the DECENB01 exit routine can be a clear key exit routine that supports the Advanced Encryption Standard (AES) up to 128-bit.

Important: DECENAA1 and DECENBB1 require the Guardium Data Encryption subsystem to run. For more information, see Setting up the Guardium Data Encryption subsystem.

Note: The following DECENAA0, DECENAA1, and DECENBB1 restrictions apply:

If ICSF APAR OA50450 is installed:

DECENAA0, DECENAA1, and DECENBB1 will work with all clear key labels

DECENAA0 and DECENAA1 will only work with secure key labels that are defined with SYMCPACFWRAP(YES) and SYMCPACFRET(YES). DECENBB1 will work with secure key labels, but performance will be improved when using key labels that are defined with SYMCPACFWRAP(YES) and SYMCPACFRET(YES).

If ICSF APAR OA50450 is not installed:

DECENAA0, DECENAA1, and DECENBB1 will work with all clear key labels

With secure key labels, DECENAA0 and DECENAA1 will not work, and DECENBB1 performance will be degraded

The InfoSphere Guardium Data Encryption encryption methods encrypt and decrypt data differently. Some of the encryption methods employ Integrated Cryptographic Service Facility (ICSF) callable services to perform the processing. Others use the zSeries Cipher Message with Chaining (KMC) or Cipher Message with Feedback (KMF) hardware instruction. When KMC or KMF are used, the encryption key data is obtained by using an ICSF callable service.

Encryption method creation

Each encryption method that you create is linked with one of the InfoSphere Guardium Data Encryption methods and the corresponding ICSF callable service.

The following figure illustrates the process of creating an encryption method.

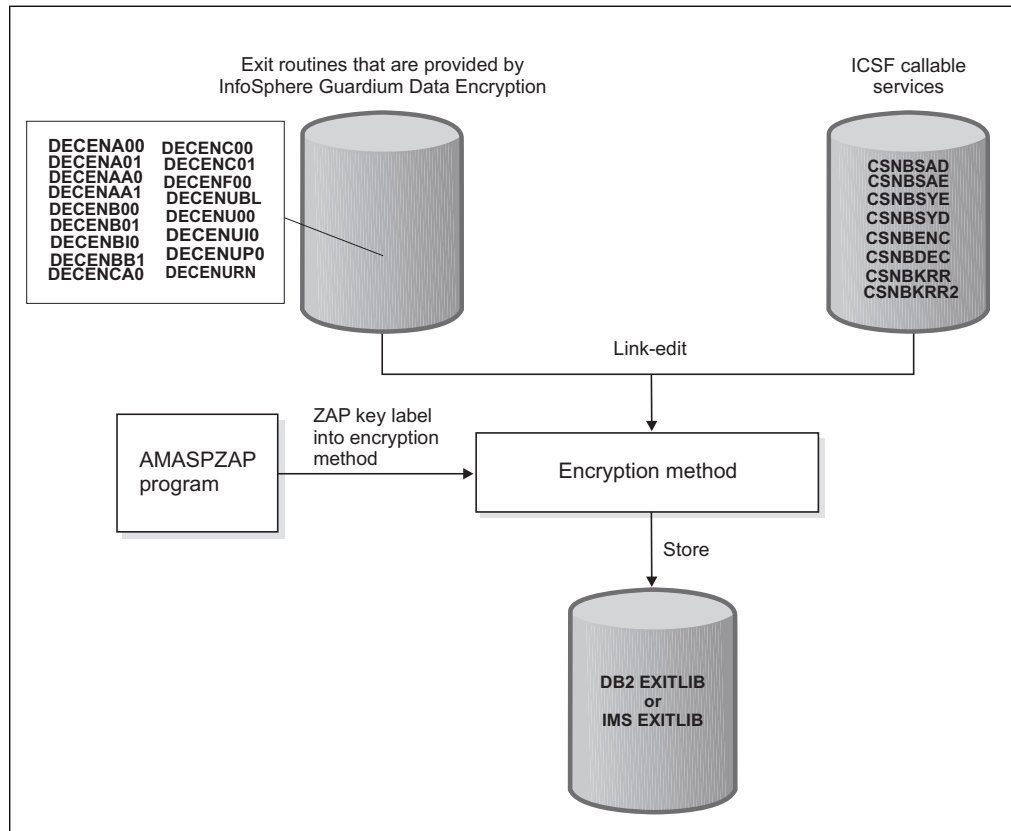


Figure 2. Process of creating an encryption method by using InfoSphere Guardium Data Encryption

As the previous figure illustrates, an encryption method is created with this process:

1. An InfoSphere Guardium Data Encryption encryption method and the corresponding ICSF callable service are link-edited into the encryption method.
2. The AMASPZAP program puts the cryptographic key label into your encryption method.
3. Your encryption method is placed in the IMS or DB2 exit library.

Encryption standards

The following tables show the encryption standards that are supported by each of the InfoSphere Guardium Data Encryption encryption methods.

Table 5. Encryption standards supported by DB2 edit procedures

InfoSphere Guardium Data Encryption exit routine for DB2	Encryption algorithm
DECENAA00	AES, Triple DES, or DES
DECENAA0	AES, Triple DES, or DES
DECENB00	AES
DECENB10	AES
DECENC00	Triple DES, or DES
DECENCA0	AES

Table 6. Encryption standards supported by DB2 field procedures

InfoSphere Guardium Data Encryption exit routine for DB2	Encryption algorithm
DECENF00	AES

Table 7. Encryption standards supported by DB2 User Defined Function

InfoSphere Guardium Data Encryption exit routine for DB2	Encryption algorithm
DECENU00	AES
DECENUBL	AES
DECENUI0	AES
DECENUP0	AES

Table 8. Encryption standards supported by IMS exit routines

InfoSphere Guardium Data Encryption exit routine for IMS	Encryption algorithm
DECENA01	Triple DES or DES
DECENAA1	AES, DES, and Triple DES
DECENB01	AES
DECENBB1	AES
DECENC01	Triple DES or DES
<p>Tip: To use a clear key, use either DECENA01 or DECENAA1. To use a CPACF-wrapped secure key, use either DECENAA1 or DECENBB1.</p> <p>Important: DECENAA1 and DECENBB1 require the Guardium Data Encryption subsystem to run. For more information, see Setting up the Guardium Data Encryption subsystem.</p> <p>Note:</p> <ol style="list-style-type: none"> 1. AES is supported, beginning with ICSF Release HCR7770, if an ICSF AES master key is defined. 	

DB2 encryption and decryption

Encryption and decryption follow a unique processing flow in the DB2 environment, and the DB2 environment poses several requirements and considerations.

DB2 encryption

The following figure shows the components and processing flow for data encryption in DB2.

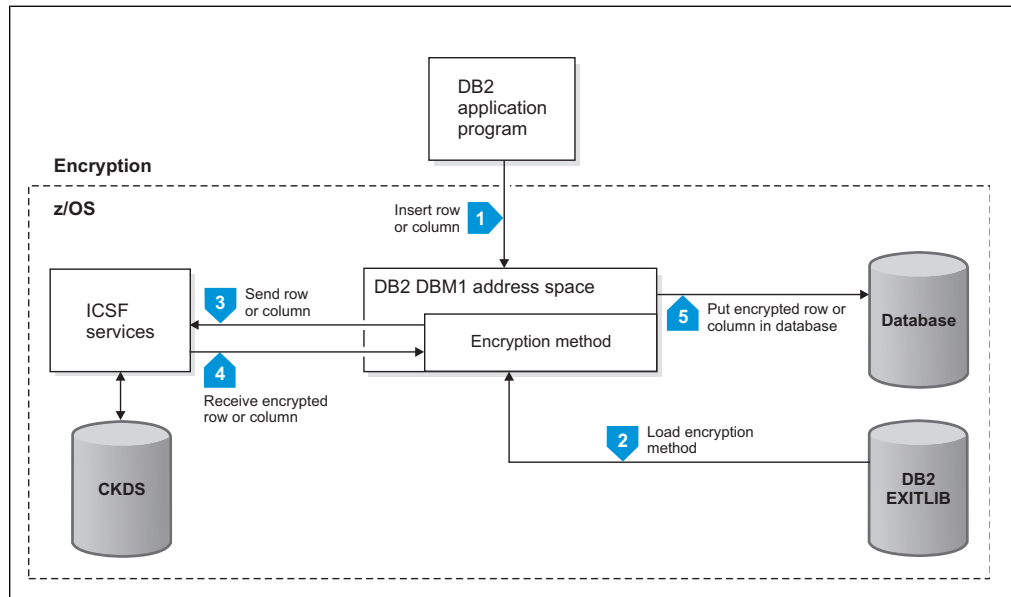


Figure 3. DB2 data encryption

As shown in the previous figure, the DB2 data encryption process consists of these steps:

1. The DB2 application program passes a row or column to DB2.
2. DB2 loads the encryption method. This encryption method is an edit procedure, a field procedure, or a User Defined Function (UDF). The edit procedure and field procedure are specified by the respective SQL CREATE TABLE statement. A UDF is specified by the respective SQL CREATE FUNCTION statement. DB2 passes the row or column to the encryption method.
3. The encryption method uses Integrated Cryptographic Service Facility (ICSF) services to pass the cryptographic key label and the row or column.
4. When the row or column is successfully encrypted, ICSF passes the row or column back to the encryption method that passes it back to DB2.
5. DB2 puts the encrypted row or column into the table.

DB2 decryption

The following figure shows the components and processing flow for data decryption in DB2.

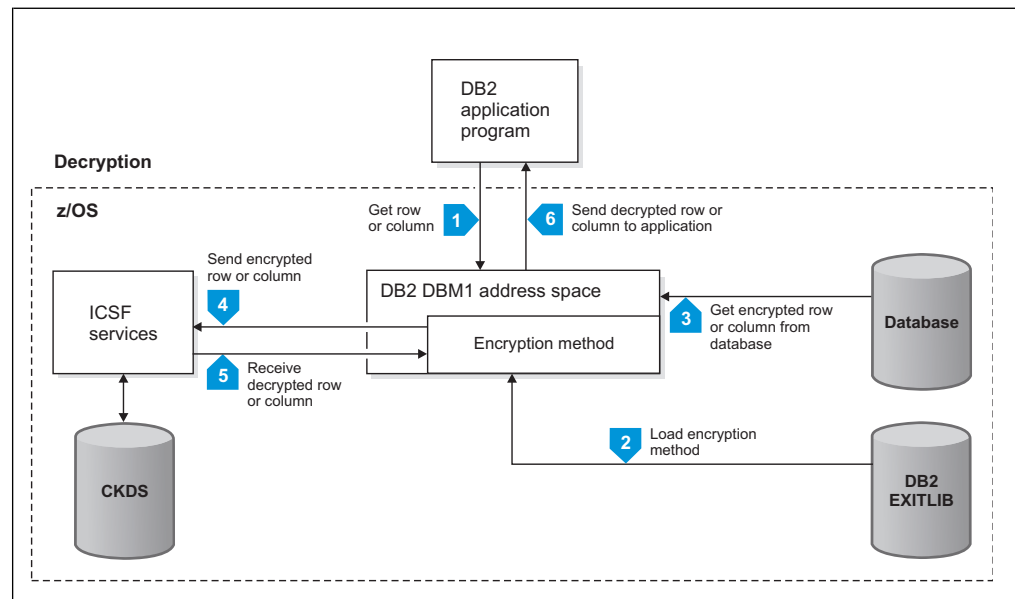


Figure 4. DB2 data decryption

As shown in the previous figure, the DB2 data decryption process consists of these steps:

1. The DB2 application program requests data from DB2.
2. DB2 loads the encryption method. This encryption method is an edit procedure, a field procedure, or a User Defined Function (UDF). The edit procedure and field procedure are specified by the respective SQL CREATE TABLE statement. A UDF is specified by the respective SQL CREATE FUNCTION statement.
3. DB2 retrieves the encrypted row or column from the table, calls the encryption method, and passes it the row or column.
4. The encryption method calls ICSF services to pass the cryptographic key label and the encrypted row or column.
5. When the row or column is successfully decrypted, ICSF passes it back to the encryption method, which passes it back to DB2.
6. DB2 passes the decrypted row or column back to the application.

DB2 clear key encryption

If the encryption method calls the clear key edit procedure, DECENA00, the DB2 data encryption process differs from the process for CPACF protected key and secure key.

The following figure shows the components and processing flow for clear key data encryption in DB2.

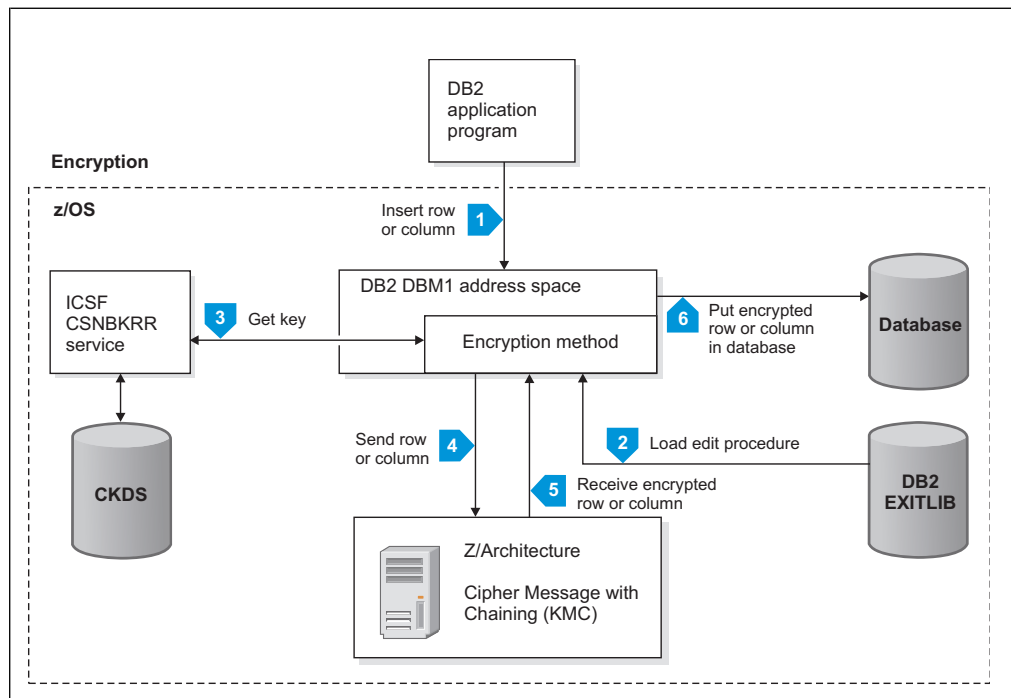


Figure 5. DB2 clear key data encryption

As shown in the previous figure, the DB2 clear key data encryption process consists of these steps:

1. The DB2 application program inserts data into a DB2 table.
2. DB2 loads the encryption method that, in this case, is an edit procedure that is specified in the edit procedure clause of the SQL CREATE TABLE statement. DB2 passes the row to this edit procedure.
3. The encryption method calls ICSF callable service CSNBKRR to obtain the encryption key from the cryptographic key data set (CKDS).
4. The encryption method sends z/OS the cryptographic key label and the row. The encryption method uses the z/OS® KMC instruction to encrypt the row.
5. When the row is successfully encrypted, z/OS passes the row back to the clear key encryption method, which passes it back to DB2.
6. DB2 puts the encrypted row into the table.

DB2 clear key decryption

If the encryption method calls the clear key edit procedure, DECENA00, the DB2 data decryption process differs from the process for CPACF protected key and secure key.

The following figure shows the components and processing flow for clear key data decryption in DB2.

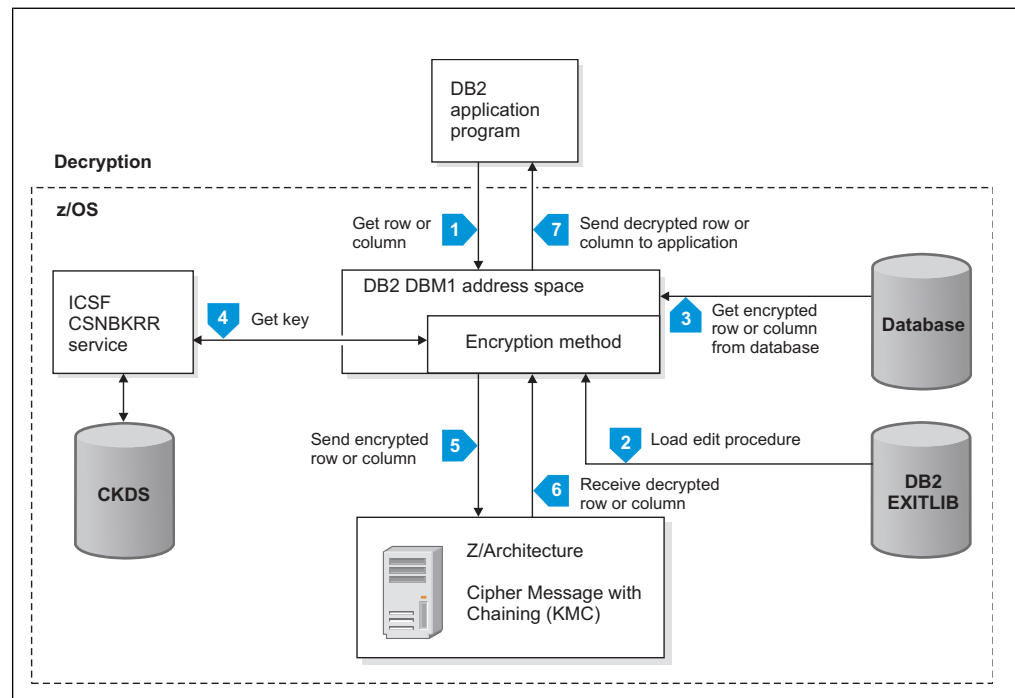


Figure 6. DB2 clear key data decryption

As shown in the previous figure, the DB2 clear key data decryption process consists of these steps:

1. The DB2 application program requests data from DB2.
2. DB2 loads the encryption method that, in this case, is an edit procedure that is specified in the edit procedure clause of the SQL CREATE TABLE statement.
3. DB2 retrieves the encrypted row from the table, calls the encryption method, and passes it the row.
4. The encryption method calls ICSF service CSNBKRR to obtain the encryption key from the CKDS.
5. The encryption method passes z/OS the cryptographic key label and the encrypted row. The encryption method uses the z/OS® KMC instruction to decrypt the row.
6. When the row is successfully decrypted, z/OS passes the row back to the encryption method, which passes it back to DB2.
7. DB2 passes the decrypted row back to the application.

DB2 restrictions

The following restrictions apply to using InfoSphere Guardium Data Encryption in a DB2 environment:

- Adding an edit procedure to encrypt rows within a compressed DB2 table nullifies DB2 internal compression processing. To ensure that DB2 internal compression processing works properly, use the externalized compression edit procedure and edit procedure driver that are supplied with the product.
- Indexes cannot be encrypted with the edit procedures because they do not support encryption of indexes. Field procedures and the UDF support index encryption.
- InfoSphere Guardium Data Encryption generates encryption methods that can support Advanced Encryption Standard 128-byte, 192-byte, and 256-byte key

lengths. However, the type of IBM mainframe server determines the type of support that is available for each key length.

- Encryption field procedures cannot be used with indexes partitioned by ranges. An encryption edit procedure must be used instead.

DB2 considerations

The following considerations apply to using InfoSphere Guardium Data Encryption in a DB2 environment:

- When you install and initialize ICSF, consider setting the CHECKAUTH installation option to NO. Setting CHECKAUTH to YES adds considerable CPU path length. Setting the KEYAUTH installation option to YES also adds CPU path length. If the CSFKEYS class is RACLISTed, the additional path length will be shortened.
- The security environment that is used when ICSF performs an authorization check might be different from the security environment that is associated with the user who is making the request.
- In DB2, you can define only one cryptographic key label per table. Depending on your security requirements, you can define different cryptographic key labels for as many tables as necessary. Cryptographic key labels are set up by your security analyst. A separate encryption exit routine must be built for each cryptographic key label that you define. You must balance your security requirements against the increased maintenance of multiple encryption exit routines.
- In DB2, you can both encrypt and compress data by using the hardware compression that is available with DB2. However, compression happens after encryption, which greatly compromises the effectiveness of the compression. Because of this limitation, you might want to disable the hardware compression that is available with DB2 on objects that are encrypted.
- The first time that you use edit procedures as part of your installation, provide APF authorization for the edit procedure SDSNEXIT library. If you are already using edit procedures, ensure that the edit procedures are stored in an APF-authorized SDSNEXIT library. In addition, you must allocate enough space in an SDSNEXIT library to hold your new edit procedures. If the SDSNEXIT library happens to take an extent while DB2 is running, the DBM1 address space might fail to load an edit procedure. The following errors occur:
 - IEW4008I FETCH FAILED FOR MODULE *mmmmmm* FROM DDNAME STEPLIB BECAUSE OF AN ERROR IN CONVERTING A TTR
 - Other messages
 - S106-0E abend in the DBM1 address space

To correct the errors, recycle DB2.

Attention: DB2 edit procedures do not encrypt indexes. Therefore, if an index is based on more than one column of the data that you are encrypting, the security of the data might be compromised. For example, a combination of columns might display personally identifiable information.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

Chapter 3, “Encrypting data in DB2,” on page 27

You can use InfoSphere Guardium Data Encryption to encrypt your DB2 data.

Related reference:

“Hardware requirements” on page 23

Before you install and configure InfoSphere Guardium Data Encryption, make sure that your environment meets the following minimum hardware requirements.

Related information:

➡ IBM Information Management Software for z/OS Solutions Information Center

For more information about DB2 edit procedures, see DB2 administration.

➡ z/OS Internet Library

For more information about the CHECKAUTH and KEYAUTH installation options, see z/OS Cryptographic Services ICSF System Programmer's Guide.

IMS encryption and decryption

Encryption and decryption follow a unique processing flow in the IMS environment, and the IMS environment poses several requirements and considerations.

IMS encryption

The following figure shows the components and processing flow for data encryption in an IMS environment.

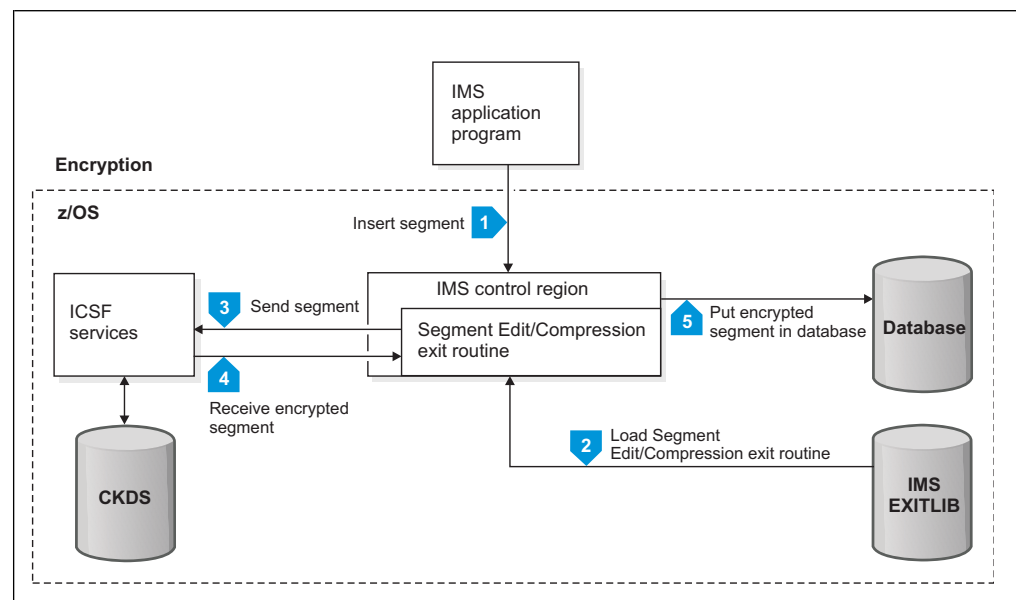


Figure 7. IMS data encryption – components and processing

As shown in the previous figure, the IMS data encryption process consists of these steps:

1. The IMS application program passes a segment REPL, ISRT, or LOAD request to the IMS control region.
2. IMS loads the encryption exit routine, which is a Segment Edit/Compression exit routine that is specified in the COMPTN parameter of the SEGM statement of the DBD. IMS passes the segment to this exit routine.

3. The encryption exit routine uses Integrated Cryptographic Service Facility (ICSF) services to pass the cryptographic key label and the segment.
4. ICSF encrypts the segment and passes it back to the encryption exit routine, which passes it back to IMS.
5. IMS puts the encrypted segment in the database.

IMS decryption

The following figure shows the components and processing flow for data decryption in the IMS environment.

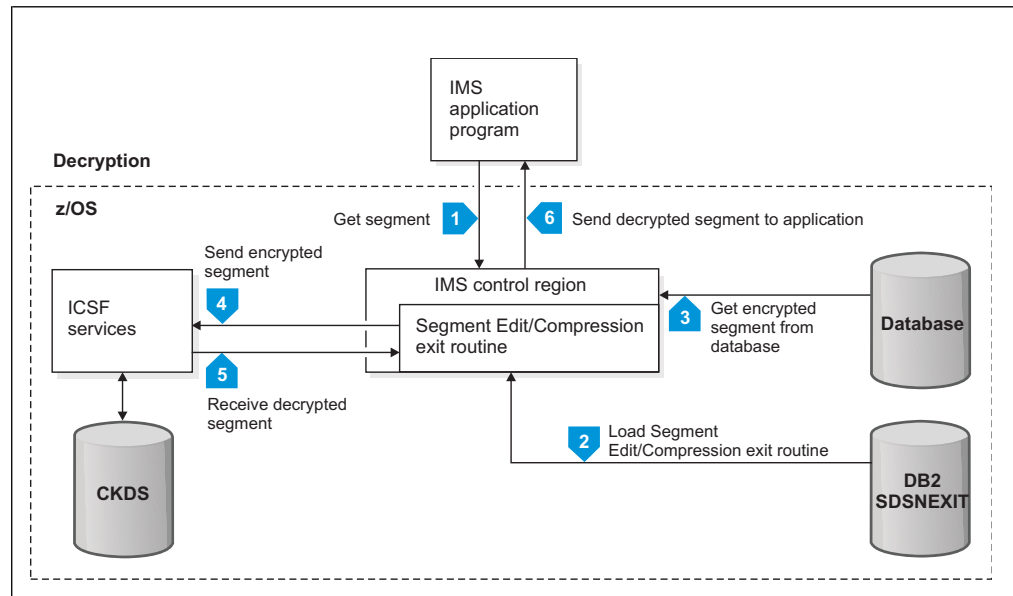


Figure 8. IMS data decryption – components and processing

As shown in the previous figure, the IMS decryption process consists of these steps:

1. The IMS application program passes the segment GET request to the IMS control region.
2. IMS loads the encryption exit routine, which is a Segment Edit/Compression exit routine that is specified in the COMPTN parameter of the SEGM statement of the DBD.
3. IMS retrieves the encrypted segment from the database and passes it to the encryption exit routine.
4. The exit routine uses ICSF services to pass the cryptographic key label and the segment.
5. When the segment has been successfully decrypted, ICSF passes the segment back to the encryption exit routine, which passes it back to IMS.
6. IMS passes the decrypted segment back to the application.

IMS restrictions

The following restrictions apply to using InfoSphere Guardium Data Encryption in an IMS environment:

- An IMS segment can be associated with only one Segment Edit/Compression exit routine. If your IMS segment is already associated with a Segment

Edit/Compression exit routine and you want to implement InfoSphere Guardium Data Encryption, you must use the exit driver that is supplied with this product.

- HIDAM index databases cannot be encrypted. The IMS DBD COMPRTN parameter does not allow index databases to be specified on the Segment Edit/Compression exit routine.
- A cryptographic key data set (CKDS) that was initialized on a z990 or later processor that has the CPACF cannot be used on a z900 or earlier processor that has the CCF.
- InfoSphere Guardium Data Encryption generates Segment Edit/Compression exit routines that can support AES 128, 192, and 256 bits key lengths. However, the type of IBM mainframe server determines the type of support that is available for each key length.

IMS considerations

The following considerations apply to using InfoSphere Guardium Data Encryption in an IMS environment:

- When you install and initialize ICSF, consider setting the CHECKAUTH installation option to NO. Setting CHECKAUTH to YES adds considerable CPU path length. Setting the KEYAUTH installation option to YES also adds CPU path length. If the CSFKEYS class is RACLISTed, the additional path length will be shortened.
- Depending on your security requirements, you can define different cryptographic key labels for as many segments as you need. Cryptographic key labels are set up by your security analyst.

A separate encryption exit routine must be built for each cryptographic key label that you define. You must balance your security requirements against the increased maintenance of multiple encryption exit routines.

- The first time that you use Segment Edit/Compression exit routines at your installation, your system programmer must provide APF authorization for the exit library.

If you are already using Segment Edit/Compression exit routines, you must ensure that the Segment Edit/Compression exit routines are stored in an APF-authorized exit library.

- IMS Control Region loads IMS Segment Edit/Compression exit routines below the 16 MB line. Having too many different exit routines can cause storage problems for the IMS Control Region.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

Chapter 4, “Encrypting data in IMS,” on page 51


You can use InfoSphere Guardium Data Encryption to encrypt your IMS data.

Related reference:

“Hardware requirements” on page 23

Before you install and configure InfoSphere Guardium Data Encryption, make sure that your environment meets the following minimum hardware requirements.

Related information:

 [IBM Information Management Software for z/OS Solutions Information Center](#)

For more information about IMS Segment Edit/Compression exit routines, see IMS administration.

 [z/OS Internet Library](#)

For more information about the CHECKAUTH and KEYAUTH installation options, see Cryptographic Services ICSF System Programmer's Guide.

Encryption with compression

You can use InfoSphere Guardium Data Encryption to encrypt and compress your DB2 or IMS data.

By compressing your data, you can reduce the amount of disk space that it requires for storage. This size reduction can improve database encryption performance and reduce I/O. Data compression must be performed before the data is encrypted because encrypted data does not contain repeating patterns and does not compress well.

You can encrypt and compress data by customizing driver routines that are provided with InfoSphere Guardium Data Encryption. These driver routines link the DB2 or IMS compression routine with the encryption exit routine that you create by using this product. When an application processes a table or segment that is linked to the driver routine, the data is first compressed and then it is encrypted.

DB2 compression

DB2 has its own compression function, but you cannot attain the benefits of compression by using it on tables that are encrypted by InfoSphere Guardium Data Encryption. Therefore, you must use the edit procedure driver that is provided with this product.

This edit procedure driver ensures that compression is performed before encryption in DB2. When DB2 performs compression on a table, it runs any edit procedures that are linked with that table first. Therefore, if you link an encryption exit routine with a table and then use the DB2 compression function on it, encryption occurs before compression.

Compression provides storage benefits by replacing repeating bit strings with shorter strings. However, the encryption process replaces all repeating bit strings with non-repeating ciphertext. Therefore, when encryption occurs before compression, compression does not reduce the size of the data.

IMS compression

Prior to data encryption, existing compression routines can be combined with the supplied compression routine driver to perform compression.

Related tasks:

“Implementing a DB2 encryption edit procedure with compression” on page 45
You can implement a DB2 encryption edit procedure with compression so that your encrypted data requires less disk space for storage.

“Encrypting IMS data with compression” on page 56

You can encrypt your IMS data with compression so that your encrypted data

requires less disk space for storage.

InfoSphere Guardium Data Encryption documentation and updates

This topic summarizes the technical changes for this edition, and explains where to find InfoSphere Guardium Data Encryption information on the web and how to receive information updates automatically.

Information on the web

The IBM Tools Library web page provides current product documentation that you can view, print, and download. To locate publications with the most up-to-date information, refer to the following web page:

<http://www.ibm.com/software/data/db2imstools/library.html>

IBM Redbooks® publications that cover DB2 and IMS Tools are available from the following web page:

<http://www.ibm.com/software/data/db2imstools/support.html>

Receiving documentation updates automatically

To automatically receive a weekly email that notifies you when new DCF documents are released, when existing product documentation is updated, and when new product documentation is available, you can register with the IBM My Support service. You can customize the service so that you receive information about only those IBM products that you specify.

To register with the My Support service:

1. Go to <http://www.ibm.com/support/mysupport>.
2. Enter your IBM ID and password, or create one by clicking **register now**.
3. When the My Support page is displayed, click **add products** to select those products that you want to receive information updates about. The DB2 and IMS Tools category is located under **Software > Data and Information Management > Database Tools & Utilities**.
4. Click **Subscribe to email** to specify the types of updates that you would like to receive.
5. Click **Update** to save your profile.

Accessibility features

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use a software product successfully.

The major accessibility features in InfoSphere Guardium Data Encryption enable users to:

- Use assistive technologies such as screen readers and screen magnifier software. Consult the assistive technology documentation for specific information when using it to access z/OS interfaces.
- Customize display attributes such as color, contrast, and font size.
- Operate specific or equivalent features by using only the keyboard. Refer to the following publications for information about accessing ISPF interfaces:
 - *z/OS ISPF User's Guide, Volume 1*

- *z/OS TSO/E Primer*
- *z/OS TSO/E User's Guide*

These guides describe how to use the ISPF interface, including the use of keyboard shortcuts or function keys (PF keys), include the default settings for the PF keys, and explain how to modify their functions.

Summary of changes

This topic summarizes the technical changes for this edition.

New and changed information is indicated by a vertical bar (|) to the left of a change. Editorial changes that have no technical significance are not noted.

SC19-3219-06

Changes and enhancements following APAR PI83626, including:

Information about setting up the Guardium Data Encryption subsystem subsystem.

Information about subsystem modules DECSSI20 and DECSSI21, which have replaced deprecated subsystem module DECSSI10.

Information about sample job DECIMSCB and ISPF skeleton file DECF0005 has been updated to remove references to DECSSI10 and simplify the binder control statements.

DECENAA1 now supports secure keys when ICSF-protected key read support is available for use.

DECENBI0 has been changed to use the correct reason code for an output length error.

Updates to Troubleshooting documentation including:

- DB2 and IMS reason codes merged into a single list
- New IMS exit reason codes
- New DB2 UDF message, DEC08
- Additional common ICSF reason codes

SC19-3219-05

New DB2 exit routine, DECENAA0, added following APAR PI58257.

SC19-3219-04

Encryption algorithms have been updated in Table 9 following APAR PI53758.

New ICSF messages have been documented following APAR PI53758.

Messages for new exit routine, DECENAA1, and new program, DECSSI10, have been added following PI57908.

Messages for new exit routine, DECENBB1, has been added following PI55772.

Tables 4, 8, and 12 updated following PI57908 and PI55772.

Figure 2 updated to include new exit routines:

- DECENAA1 added, following PI57908.
- DECENBB1 added, following PI55772.

New IMS exit message have been documented following APAR PI57908.

SC19-3219-03

Information about COMPRTN parameters and unique driver names was added to “Building a compression and encryption IMS exit routine” on page 54

DB2 encryption methods DECENURN has been documented following APAR PI44506.

DB2 encryption methods DECENUI0 and DECENUP0 have been documented following APAR PI32270.

Information about new Random Number Generator UDF has been added following PI44506.

Information about new self-generating ICV EDITPROC, DECENBI0, has been added following APAR PI33789.

Information about new BLOB UDF, DECENUBL, has been added following PI50138.

Information about new AES secure key, DECENCA0, has been added following PI49626.

New messages are provided in “DB2 SQL codes” on page 72: DEF1, DEF2, DEC08B, DEC08C, DECO8D, DECO8E, DECO8F

New message is provided in “IMS reason codes” on page 63: C4C5C31E

SC19-3219-02

Table 8 on page 9 was updated to indicate that IMS exit routine DECENB01 supports the AES encryption algorithm only.

SC19-3219-01

The following changes were made following APAR PM55879.

- Information about encryption standards supported by IMS exit routines was added to Encryption methods.
- Information about DB2 encryption was added to DB2 encryption and decryption.
- Information about DB2 data decryption was added to DB2 encryption and decryption.
- Changes were made to information about DB2 compression and encryption restrictions in DB2 encryption and decryption.
- Changes were made to information in Software requirements.
- Information about user access control was added to Creating a DB2 User Defined Function for encryption.
- Information about UDF parameters was added to Implementing a DB2 UDF for encryption.
- Information about ICSF return codes was removed as it is no longer relevant.
- Changes were made to information about Creating a DB2 edit procedure driver.
- Information about the appropriate times to upgrade the encryption method for the cryptographic key was added to Upgrading the cryptographic key encryption.

SC18-9549-07

- Information about activating ICSF cryptographic services for cryptographic coprocessors has been removed. For information about activating ICSF, see *Cryptographic Services ICSF System Programmer's Guide* on the z/OS Internet Library at <http://www.ibm.com/systems/z/os/zos/bkserv/>.
- Overview information about encryption with compression and DB2 restrictions is provided in "Encryption with compression" on page 18.

SC18-9549-06

- Information about the DB2 error message with error code -652 is provided in "DB2 reason codes" on page 68.
- New messages are provided in "IMS reason codes" on page 63: C4C5C3A2, C4C5C3B1, C4C5C3B2, C4C5C3C1, C4C5C3CF.
- New messages are provided in "DB2 reason codes" on page 68: C4C5C307, C4C5C308, C4C5C309, C4C5C30A, C4C5C30B, C4C5C30C, C4C5C31D.

SC18-9549-05

- InfoSphere Guardium Data Encryption supports encryption and compression. Incorrect information about that feature has been removed.

SC18-9549-04

- Information about support for Crypto Express3 was added.

SC18-9549-03

- New information about edit procedure was added.
- Minor changes to Key label information were made.
- A new section describes encrypting DB2 data with compression by using the edit procedure driver.

SC18-9549-02

- ICSF updates were added.
- A new chapter addresses IMS exit driver customization and implementation.
- A new panel updates the IMS Advanced Encryption Standard (AES).

Chapter 2. InfoSphere Guardium Data Encryption installation information

Your system must meet specific prerequisites as well as hardware and software requirements before you install InfoSphere Guardium Data Encryption.

Prerequisites

InfoSphere Guardium Data Encryption requires that its cryptographic keys be stored in the Integrated Cryptographic Service Facility (ICSF) cryptographic key data set (CKDS).

This data set is the key repository where all cryptographic keys are stored. To initialize and use the CKDS, ICSF requires that a secure key device, such as a Crypto Express3 coprocessor, is available. Even if only clear keys are used to protect the databases, a secure key device is required to initialize and use the CKDS.

The secure key devices also require that a master key is loaded before the secure key functions are available.

Related information:

 [z/OS Internet Library](#)

For more information about the procedures for loading master keys see Cryptographic Services ICSF Administrator's Guide.

Hardware requirements

Before you install and configure InfoSphere Guardium Data Encryption, make sure that your environment meets the following minimum hardware requirements.

- To support the z10™ processor encryption technology, Crypto Express3 (feature code 0864) with CP Assist for Cryptographic Function (CPACF) hardware is required and must be installed. Installation of Crypto Express3 feature requires that the CP Assist for Cryptographic Functions (CPACF) DES/TDES Enablement feature (feature code 3863) is installed. For more information, refer to the z/OS V.2.2 server hardware information in the IBM Knowledge Center: https://www.ibm.com/support/knowledgecenter/SSLTBW_2.2.0/com.ibm.zos.v2r2.csfb200/server_HW.htm.
- On the z9® EC and the z10, the Crypto Express2 feature (feature code 0863) is required. On the z9 BC, the Crypto Express2 feature or the Crypto Express2-1P (feature code 0870) is required. At least one of the cryptographic engines must be configured as a coprocessor to provide secure key capability. Installation of either Crypto Express2 feature requires that the CP Assist for Cryptographic Functions (CPACF) DES/TDES Enablement feature (feature code 3863) is installed.
- On z890 and z990 systems, either a PCIXCC (feature code 0868) or a Crypto Express2 (feature code 0863) provides secure key support. Installation of either of these features requires that the CP Assist for Cryptographic Functions (CPACF) DES/TDES Enablement feature (feature code 3863) is installed.
- The Cryptographic Coprocessor Feature (CCF) provides secure key support on z800, z900, and earlier machines (G3, G4, G5, G6, Multiprise 2000, Multiprise

3000). The CCF hardware modules must be enabled with configuration data, a feature that is ordered separately, and requires a processor power-on-reset (POR) to complete data loading into the cryptographic modules. Because this hardware does not support the clear key APIs, the use of clear keys by InfoSphere Guardium Data Encryption is not supported on the CCF-based machines. The PCICC feature (feature code 0861) is an optional secure key device on the z800 and z900 systems.

Related information:

 [Support & downloads](#)

For additional configuration information for your cryptographic hardware, see the following publications: *zEnterprise System Processor Resource/Systems Manager Planning Guide*, *System z10 Processor Resource/Systems Manager Planning Guide*, and *System z10 Support Element Operations Guide*.

Software requirements

Before you install and configure InfoSphere Guardium Data Encryption, make sure that your environment meets the following minimum software requirements.

InfoSphere Guardium Data Encryption works with all supported releases of DB2 and IMS.

InfoSphere Guardium Data Encryption requires that Integrated Cryptographic Service Facility (ICSF) is active. The ICSF version that is required depends on the platform and the cryptographic hardware that is installed.

In addition to being active, the ICSF version must support the secure key device on the specific platform.

On z10 processors that support the Crypto Express3 coprocessor with CP Assist for Cryptographic Function (CPACF protected key), z/OS 1.9 or later is required with ICSF release HCR7770 or later.

For z10 systems without a CEX2C card, encryption can be implemented with the ICSF release HCR7751.

To use clear keys on the IBM System z z890 or IBM System z z990 processor with OS/390 Version 2 Release 10 or later, the following software is required:

- IMS and DB2 Data Encryption Tool PTF UK00049 (APAR-PQ94822).
- To ensure that ICSF also provides clear key support, the following FMIDs must be updated:
 - HCR770A with PTF UA15677
 - HCR770B with PTF UA15678
 - HCR7720 with PTF UA15679
 - HCR7730 (which has the clear key PTF incorporated) or later

Related information:

 [z/OS downloads](#)

For the latest supported versions of ICSF and additional configuration information, see the Cryptographic Support web deliverable.

Performance considerations

The amount of CPU processing that occurs depends on how your applications access data and Integrated Cryptographic Service Facility (ICSF) installation options that you use.

In most cases, the performance of InfoSphere Guardium Data Encryption is superior to the column level encryption that is available with DB2. An exception is the case where you want to encrypt only one or two small columns out of a very large row. The performance cost can be characterized in terms of the overhead and hardware per-byte costs.

ICSF supports Triple DES, DES, and AES symmetric algorithms. DES encryption uses single length (8-byte) keys. Triple DES can use double-length (16-bytes) or triple-length (24-bytes) keys. AES encryption can use 128-bit, 192-bit, or 256-bit keys. However, AES is supported only in hardware on the z9 with 128-bit keys and on the z10 with 128-bit, 192-bit, or 256-bit key lengths. Triple DES and DES can be done on either the clear key hardware (CPACF) or the secure key hardware (PCIXCC or CEX2C). Performance metrics vary depending on the algorithm and the type of hardware that is used. In addition, although clear key encryption is generally faster than both CPACF protected key and secure key, clear key means that the actual key value exists in the DB2 address space. In clear key encryption, operational procedures protect the key value. Operational procedures might include limiting who can view address space and when address space can be viewed either in real time or in memory dumps.

Setting up the Guardium Data Encryption subsystem

The Guardium Data Encryption subsystem provides support for securely obtaining CPACF-wrapped keys from ICSF. The subsystem must be defined and initialized prior to the use of exits DECENAA1 or DECENBB1.

The Guardium Data Encryption subsystem consists of two load modules, DECSSI20 and DECSSI21. Before initializing the subsystem, these modules must be made accessible in either the system link list or the LPA. Adding these modules to the system link list prevents the operational code from being visible to unauthorized applications in common storage and is the recommended way to set up the Guardium Data Encryption subsystem.

The Guardium Data Encryption subsystem can be defined to z/OS and initialized when z/OS is started, or dynamically initialized at a later time. Only one Guardium Data Encryption subsystem must be defined and initialized per instance of z/OS. If an additional Guardium Data Encryption subsystem is defined and initialized, the operational code will be refreshed. If the code is updated, defining and initializing an additional Guardium Data Encryption subsystem can be used as a method to refresh the operational code without an IPL. Do not refresh the operational code while exits are actively running because unpredictable results might occur.

To define the Guardium Data Encryption subsystem when z/OS is initialized, you must update an IEFSSNxx member of SYS1.PARMLIB. For more information, see *z/OS MVS Initialization and Tuning Reference*, https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.1.0/com.ibm.zos.v2r1.ieae200/toc.htm. You can code the IEFSSNxx definition by using a keyword parameter form or a positional parameter form. The keyword parameter form has the following syntax:

```
SUBSYS SUBNAME(ssname) INITRTN(DECSSI20)
```

The positional parameter form has the following syntax:

xxxx,DECSSI20

To define the Guardium Data Encryption subsystem dynamically, issue the following z/OS MVS system command:

SETSSI ADD,SUB=xxxx,INITRTN=DECSSI20

where: *xxxx* is a unique subsystem name of your choice.

Note:

To provide increased protection of ICSF clear key data, and the ability to use CPACF-wrapped secure keys when ICSF APAR OA50450 is installed, APAR PI83626 incompatibly changed the Guardium Data Encryption subsystem interface. The prior subsystem initialization routine, DECSSI10, has been deprecated and was left unchanged to provide a migration path for existing IMS segment compression exit routines that were built by using DECENAA1 or DECENBB1. This enables the exit routines to operate unchanged until they are rebuilt with the PI83626 code level.

Until all existing IMS segment compression exit routines that were built by using DECENAA1 and DECENBB1 are rebuilt, you must initialize two unique systems, one with initialization routine DECSSI20 and the other with DECSSI10. After all exit routines that require the pre-PI83626 version of the Guardium Data Encryption subsystem have been rebuilt with the latest level of code, it is not necessary to initialize a Guardium Data Encryption subsystem by using the initialization routine DECSSI10. The subsystem that was initialized with DECSSI10 can be removed.

Chapter 3. Encrypting data in DB2

You can use InfoSphere Guardium Data Encryption to encrypt your DB2 data.

Procedure

To encrypt your DB2 data, create an encryption exit routine, which is a DB2 edit procedure. You then specify that encryption exit routine as part of a table unload and reload operation.

InfoSphere Guardium Data Encryption can implement encryption in one of two ways:

- Without compression if you create only an encryption exit routine
- With compression if you link the encryption exit routine that you create to an existing compression edit procedure

Comparison of DB2 encryption methods

Knowing the advantages and disadvantages of different IBM DB2 encryption methods can help you determine which method to use.

- UDFs DECENU00, DECENUI0, DECENUP0, and DECENUBL

Key type: ICSF CPACF Protected Key

You can use the UDF to manage access control of encrypted data.

You can use DECENURN in conjunction with DECENUI0 and DECENUP0 to create unique ICVs.

Table 9. Advantages and disadvantages of the DECENU00, DECENUI0, DECENUP0, and DECENURN encryption methods.

Advantages	Disadvantages
<ul style="list-style-type: none">• Is data-type independent.• Uses ICSF release HCR7770 technology on the latest z/OS hardware.• Can use DB2 MASK, VIEW, and TRIGGER features, RACF® SYSADM or SECADM to provide comprehensive granular access control.• Can encrypt indexes.• Can avoid performance issues for non-authorized users (by using DB2 MASK, VIEW, and TRIGGER features).• Can be implemented by using UPDATE, thus maintaining database availability.• Does not decrypt unloaded columns.• DECENURN can be used in conjunction with DECENUI0 and DECENUP0 to create unique ICVs.	<ul style="list-style-type: none">• Can slow down performance during mass updates and predicate processing. Ensure that the DB2 WLM started task is set to the maximum velocity to help performance.• Does not decrypt unload columns.

- FIELDPROC DECENF00

Key type: ICSF CPACF Protected Key

You can use a field procedure to transform values in a single, short string column.

Table 10. Advantages and disadvantages of DECENF00 encryption method.

Advantages	Disadvantages
<ul style="list-style-type: none"> • Performs well. • Uses ICSF release HCR7770 technology on the latest z/OS hardware. • Uses RACF and SYSADM or SECADM access control. • Can encrypt indexes. • Can be implemented by using ALTER ADD NEW COLUMN, thus maintaining database availability. 	<ul style="list-style-type: none"> • Can degrade performance if more than 2 columns are encrypted. • Has column data type restrictions. • Cannot use column lengths that exceed 254 bytes in length. • Cannot use column names that exceed 18 characters in length. • Decrypts columns for users and applications even when it is not necessary for the application or user.

- Edit procedure DECENB00
Key type: ICSF CPACF Protected Key
You can use an edit procedure to transform values on a row.

Table 11. Advantages and disadvantages of DECENB00 encryption method.

Advantages	Disadvantages
<ul style="list-style-type: none"> • Is column data-type independent. • Uses ICSF release HCR7770 technology on the latest z/OS hardware. • Uses acceptable RACF and SYSADM or SECADM access control. 	<ul style="list-style-type: none"> • Cannot encrypt indexes. • Must drop and reload databases to implement encryption that is affecting database availability. • Decrypts rows for users and applications for each access.

- Edit procedure DECENB10
Key type: ICSF CPACF Protected Key
You can use an edit procedure to transform values on a row.

Table 12. Advantages and disadvantages of DECENB10 encryption method.

Advantages	Disadvantages
<ul style="list-style-type: none"> • Generates unique ICVs. • Is column data-type independent. • Uses ICSF release HCR7770 technology on the latest z/OS hardware. • Uses acceptable RACF and SYSADM or SECADM access control. 	<ul style="list-style-type: none"> • Cannot encrypt indexes. • Must drop and reload databases to implement encryption that is affecting database availability. • Decrypts rows for users and applications for each access.

- Edit procedure DECENB00
Key type: ICSF Key Retrieve
You can use an edit procedure to transform values on a row.

Table 13. Advantages and disadvantages of DECENAA0 encryption method.

Advantages	Disadvantages
<ul style="list-style-type: none"> • Is column data-type independent. • Operates quickly. • Uses RACF and SYSADM or SECADM access control. 	<ul style="list-style-type: none"> • Cannot encrypt indexes. • Uses an ICSF feature that moves the encryption key into program storage. This disadvantage is deemed an acceptable risk by some users. • Must drop and reload databases to implement encryption that is affecting database availability. • Decrypts rows for users and applications for every access.

- Edit procedure DECENAA0

Key type: ICSF Key Retrieve

You can use an edit procedure to transform values on a row. DECENAA0 retrieves a CPACF-wrapped version of a secure or clear key from ICSF. For secure keys, the CPACF-wrapped key provides better performance than ICSF secure key APIs. For clear keys, the CPACF-wrapped key provides improved security over clear key exits with a minimal increase in CPU usage.

- Edit procedures DECENCA0 and DECENC00

Key type: Secure Key

You can use an edit procedure to transform values on a row.

Note: Edit procedure DECENCA0 is available for use in instances where AES with Secure Key is required despite the resulting performance impact. In other instances, the use of DECENCA0 is strongly discouraged due to performance concerns.

Table 14. Advantages and disadvantages of DECENCA0 and DECENC00 encryption methods.

Advantages	Disadvantages
<ul style="list-style-type: none"> • Column data type independent. • The most secure technology available. • Uses RACF and SYSADM or SECADM access control. • DECENCA0 provides support for AES with Secure Key encryption 	<ul style="list-style-type: none"> • Recommended only for small databases. • Cannot encrypt indexes. • Must drop and reload databases to implement encryption that is affecting database availability. • Decrypts rows for users and applications for every access. • Performs poorly due to the secure, but restrictive, processor ICSF architecture. DECENCA0 performance is especially poor because it requires multiple ICSF service calls per encrypt/decrypt operation.

Creating a DB2 encryption method

To use InfoSphere Guardium Data Encryption to encrypt DB2 data, you must create an encryption method. An encryption method is a DB2 edit procedure, field procedure, or User Defined Function. Each encryption method includes a user-specified cryptographic key label.

Procedure

To create the encryption method, use the sample jobs or the ISPF interface.

Requirement: After you use SMP/E to apply maintenance, you must re-create the encryption method. Use the same cryptographic key label and encryption method name.

Creating a DB2 User Defined Function for encryption

You can use a User Defined Function (UDF) to manage access control of encrypted data. Separating the roles of people who use and manage encrypted data prevents situations where one person has total control over the security and encryption environments.

Procedure

1. Use the ISPF panel or the sample job to create an encryption UDF.
2. Define the UDF to DB2 by using the CREATE FUNCTION SQL JCL.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Creating a DB2 encryption UDF by editing a sample job

InfoSphere Guardium Data Encryption provides sample jobs that you can edit to create an encryption User Defined Function (UDF).

Before you begin

Obtain the cryptographic key label from the security analyst who installs or administers Integrated Cryptographic Service Facility (ICSF).

About this task

You can create an encryption UDF by editing the sample job that is available in PDS *smphlq.SDECSAMP*, where *smphlq* is the SMP/E high-level qualifier for the product:

DECDB2UD

This job link-edits the DB2 CPACF protected key UDFs, DECENU00, DECENUI0 and DECENUP0, and the Random Number UDF DECENURN ICV, with their corresponding ICSF callable services.

DECUXUDF

This member contains SQL statements and descriptions that demonstrate and describe the use of the DECENURN and DECENUI0 UDFs.

DECUXUDF is provided to demonstrate the use of DB2 UDFs provided by Encryption Tool. The SQL contained is intended to provide a functioning example for:

- Creating the DB2 functions for invoking the UDF
- Inserting rows containing encrypted column values
- Selecting rows, decrypting column values
- Updating existing rows, encrypting a previously non-encrypted column (for migration)

- Using DECENURN in conjunction with DECENUI0 to generate a unique ICV per row

DECENURN

This UDF program retrieves a random value from the CSNBRNGL ICSF callable service. DECENURN can be used in conjunction with DECENUI0, DECENUBL, and DECENUP0.

Note: DECUXUDF is intended to serve as a basis for your own customized use. The sample SQL can be run without modification, but instances of **INFOSPHERE GUARDIUM DATA ENCRYPTION ENCRYPTED KEY** should be replaced with the cryptographic key label that was built by your security analyst.

Procedure

1. Edit the sample job that is associated with the UDF.
2. Replace all lowercase JCL variables and data set names with values for your installation. The UDF name that you specify must be a unique name; it cannot be a DBD name.
3. At the bottom of the jobs, replace the variable yyyyyyyyyy with the cryptographic key label that was built by your security analyst.

The encryption key label that you specify can be up to 64 characters long. If you do not use all 64 characters, include the correct number of trailing blanks before the right parenthesis that ends the parameter list.

Your encryption key label is 50 characters long, include 14 trailing blanks, as shown in this example:

```
(yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy          )
```

What to do next

If you modify the encryption UDF, a DB2 restart is required to refresh the encryption UDF with the new version.

You might have to customize the job to run in your ISPF/PDF environment. For example, you might have to add your ISPF basic target libraries to the appropriate //ISP α LIB ddname concatenations and add //ISPTABL to point to the same libraries as //ISPTLIB.

Related tasks:

“Creating a DB2 User Defined Function for encryption” on page 30

You can use a User Defined Function (UDF) to manage access control of encrypted data. Separating the roles of people who use and manage encrypted data prevents situations where one person has total control over the security and encryption environments.

Creating a DB2 encryption UDF by using the ISPF interface

InfoSphere Guardium Data Encryption provides an ISPF interface that you can use to create an encryption User Defined Function (UDF) and to specify the cryptographic key label.

Before you begin

Obtain the cryptographic key label from the security analyst who installs or administers Integrated Cryptographic Service Facility (ICSF).

About this task

To create the encryption UDF by using the ISPF interface:

Procedure

1. Issue the following command to start the product and to display the ISPF interface:
ex 'smpe_high_level_qualifier..SDECCEXE(DECENC04)' 'smpe_high_level_qualifier'
where:

smpe_high_level_qualifier.SDECCEXE
SMP/E library where the REXX EXECs for the product are installed.

smpe_high_level_qualifier
SMP/E high-level qualifier of the data set name for the SMP/E libraries where InfoSphere Guardium Data Encryption is installed.
2. Press Enter to display the InfoSphere Guardium Data Encryption panel.
3. Select Build a DB2 encryption exit. to display the Data Encryption for DB2 Databases panel.
4. Select Build a DB2 encryption User Defined Function.
5. Specify the following parameters:

Use SECURED

Optionally generates the SECURED keyword in the SQL CREATE FUNCTION statement.

Note: The SECURED keyword is only valid when using DB2 V10 NFM or later. Refer to the *IBM DB2 Universal Database SQL Reference* manual for more information about the SECURED keyword.

Build link step

Optionally generates a LINK step in the JCL, which runs the **PGM=IEWL** command with the input parameters that are necessary for the selected UDF.

Build DROP stmt

Optionally generates a DROP FUNCTION SQL statement prior to the CREATE FUNCTION statement. Can be used to replace a UDF with the same specific name as one that has been previously defined.

DB2 subsystem

Name of the DB2 subsystem where the UDF is to be created.

Run program

Name of the DB2 run program.

Plan

The DB2 plan name for the specified Run program.

DB2 WLM task

Workload manager started task defined by DB2.

Specific name

Name that you want to assign the UDF.

Set current SQLID

This field is optional. If non-blank, the provided SQLID is used in the generated SET CURRENT SQL statement, which appears before any other SQL statements are run. When left blank, no SET CURRENT SQLID statement is generated.

Column data type

Current data type of the data to be encrypted.

Column length

Length of the column to be encrypted.

- Specify the following encryption JCL parameters:

CSF lib

Name of the library that contains the ICSF callable services that you are using. Refer to “Encryption methods” on page 4 to see the ICSF callable service that is used by InfoSphere Guardium Data Encryption UDF.

SMP lib

Name of the SMP/E library in which the product is installed.

DB2 loadlib

Name of the library where DB2 is installed.

DB2 runlib

Name of the library where DB2 plans a bounds into.

CEE bind lib

Name of the library where the Language Environment® is installed.

CEE run lib(s)

Names of one or two Language Environment runtime library data sets.

DB2 exit lib

Name of the exit library to store the UDF.

- Press Enter to save your specifications and to display the job that you created.

What to do next

You might need to customize the job to run in your ISPF/PDF environment. For example, you might need to add your ISPF basic target libraries to the appropriate //ISPxLIB ddname concatenations and add //ISPTABL to point to the same libraries as //ISPTLIB.

Related tasks:

“Creating a DB2 User Defined Function for encryption” on page 30

You can use a User Defined Function (UDF) to manage access control of encrypted data. Separating the roles of people who use and manage encrypted data prevents situations where one person has total control over the security and encryption environments.

Creating CREATE FUNCTION SQL JCL

You can use an ISPF panel to generate the CREATE FUNCTION SQL JCL for a User Defined Function (UDF). The CREATE FUNCTION SQL JCL implements a UDF by defining it to DB2.

About this task

This function is available only for the UDF and is run after the UDF job is built.

Procedure

- Specify the following ICSF encryption key information:

Column data type

Current data type of the data to be encrypted.

Run program

Name of the DB2 run program.

Specific name

Name that you want to assign the UDF.

Column length

Length of the column to be encrypted.

Plan

DB2 plan.

DB2 WLM task

Workload manager started task defined by DB2.

2. Specify the following encryption JCL parameters:

CSF lib

Name of the library that contains the ICSF callable services that you are using. Refer to “Encryption methods” on page 4 to see the ICSF callable services that are used by InfoSphere Guardium Data Encryption field procedures.

SMP lib

Name of the SMP/E library in which the product is installed.

DB2 loadlib

Name of the library where DB2 is installed.

DB2 runlib

Name of the library where DB2 plans a bound into.

CEE bind lib

Name of the library where the Language Environment is installed.

CEE run lib(s)

Names of one or two Language Environment runtime library data sets.

DB2 exit lib

Name of the DB2 exit library to store the JCL.

Results

After you enter all of the parameters, InfoSphere Guardium Data Encryption completes two job steps:

1. *LINK*

This job step binds the user's local copies of Language Environment (LE) and ICSF components into a form of the UDF DECENU00 that you can execute.

2. *BATCHTSO*

This job step executes the CREATE UDF DDL and binds the DB2 run time components into the executable (SPECIFIC NAME) form of the UDF.

After these two job steps, the CREATE FUNCTION SQL JCL is generated.

Related tasks:

“Creating a DB2 User Defined Function for encryption” on page 30

You can use a User Defined Function (UDF) to manage access control of encrypted data. Separating the roles of people who use and manage encrypted data prevents situations where one person has total control over the security and encryption environments.

Creating a DB2 field procedure for encryption

You can create a field procedure by using the ISPF interface or by editing the sample job. A field procedure is a user-written encryption method that is used to transform values in a single, short string column.

About this task

You can assign field procedures to a table by specifying the `FIELDPROC` clause of the `CREATE TABLE` or `ALTER TABLE` statement.

Procedure

To create the field procedure, use either the sample job or the ISPF interface.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Creating a DB2 encryption field procedure by editing a sample job

InfoSphere Guardium Data Encryption provides sample jobs that you can edit to create a field procedure.

Before you begin

Obtain the cryptographic key label from the security analyst who installs or administers Integrated Cryptographic Service Facility (ICSF).

About this task

Create a field procedure by editing the following sample job, which is available in PDS *smpe_high_level_qualifier.SDECSAMP*, where *smpe_high_level_qualifier* is the SMP/E high-level qualifier for the product:

DECDFCL

This job link-edits the DB2 CPACF protected key field procedure, DECENBF00, with its corresponding ICSF callable services.

Procedure

1. Edit the sample job, DECDFCL.
2. Replace all lowercase JCL variables and data set names with values that are appropriate for your installation. The field procedure name that you specify must be a unique name; it cannot be a DBD name.
3. At the bottom of the job, replace the variable `yyyyyyyyyy` with the cryptographic key label that was built by your security analyst.

The encryption key label that you specify can be up to 64 characters long. If you do not use all 64 characters, include the correct number of trailing blanks before the right parenthesis that ends the parameter list. For example, if your encryption key label is 50 characters long, include 14 trailing blanks, as shown in this example:

```
(yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy )
```

What to do next

If you modify the field procedure, a DB2 restart is required to refresh the field procedure with the new version.

You might need to customize the job to run in your ISPF/PDF environment. For example, you might need to add your ISPF basic target libraries to the appropriate //ISPxLIB ddname concatenations and add //ISPTABL to point to the same libraries as //ISPTLIB.

Related tasks:

“Creating a DB2 field procedure for encryption” on page 35

You can create a field procedure by using the ISPF interface or by editing the sample job. A field procedure is a user-written encryption method that is used to transform values in a single, short string column.

Creating a DB2 field procedure by using the ISPF interface

InfoSphere Guardium Data Encryption provides an ISPF interface that you can use to build the JCL jobstream for a field procedure and to specify the cryptographic key label.

Before you begin

Obtain the cryptographic key label from the security analyst who installs or administers Integrated Cryptographic Service Facility (ICSF).

About this task

If you modify the field procedure, a DB2 restart is required to refresh the field procedure with the new version.

Procedure

1. Issue the following command to start the product and to display the ISPF interface:

```
ex 'smpe_high_level_qualifier.SDECCEXE(DECENC04)' 'smpe_high_level_qualifier'
```

where:

smpe_high_level_qualifier.SDECCEXE

SMP/E library where the REXX EXECs for the product are installed.

smpe_high_level_qualifier

SMP/E high-level qualifier of the data set name for the SMPE libraries where InfoSphere Guardium Data Encryption is installed. The high-level qualifier is provided by the person who installs the product.

2. Press Enter to display the InfoSphere Guardium Data Encryption panel.
3. Select Build a DB2 encryption exit to display the Data Encryption for DB2 Databases panel.
4. Select Build a DB2 encryption FIELDPROC.
5. Specify parameters for the selected UDF.

Key label

Cryptographic key label. The encryption process uses this label to build encrypted data.

6. Specify the following encryption JCL parameters:

CSF lib

Name of the library that contains the ICSF callable services that you are

using. Refer to “Encryption methods” on page 4 to see the ICSF callable services that are used by InfoSphere Guardium Data Encryption field procedures.

ZAP lib

Name of the library in which the load module zap program, AMASPZAP, is stored. This program puts the encryption key label into the field procedure.

SMP lib

Name of the SMP/E library in which the product is installed.

DB2 exit lib

Name of the exit library to store the field procedure.

Exit name

Unique name for the field procedure (for example, ICSFSAMP).

7. Press Enter to save your specifications and to display the job that you created.

What to do next

If you modify the field procedure, a DB2 restart is required to refresh the field procedure with the new version.

You might need to customize the job to run in your ISPF/PDF environment. For example, you might need to add your ISPF basic target libraries to the appropriate //ISPxLIB ddname concatenations and add //ISPTABL to point to the same libraries as //ISPTLIB.

Related tasks:

“Creating a DB2 field procedure for encryption” on page 35

You can create a field procedure by using the ISPF interface or by editing the sample job. A field procedure is a user-written encryption method that is used to transform values in a single, short string column.

Creating a DB2 edit procedure for encryption

Create a DB2 edit procedure for encryption using the ISPF interface or by editing the sample job.

About this task

An edit procedure is assigned to a table by the edit procedure clause of the CREATE TABLE statement. An edit procedure receives the entire row of a base table in internal DB2 format. It can transform the row when it is stored by an INSERT or UPDATE SQL statement or by the LOAD utility.

Procedure

Use the ISPF panel or the sample job to create a field procedure.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Creating a DB2 encryption edit procedure by editing a sample job

InfoSphere Guardium Data Encryption provides sample jobs that you can edit to create the edit procedure.

Before you begin

Obtain the cryptographic key label from the security analyst who installs or administers Integrated Cryptographic Service Facility (ICSF).

About this task

You can create encryption edit procedure by editing any of these sample jobs, which are available in PDS *smphlq.SDECSAMP*, where *smphlq* is the SMP/E high-level qualifier for the product:

DECDB2CL

This job link-edits the DB2 CPACF protected key edit procedures, DECENB00 and DECENBI0, with the corresponding ICSF callable services.

DECDB2JB

This job link-edits the DB2 secure key edit procedures, DECENC00 and DECENCA0, with the corresponding ICSF callable services.

DECDB2CK

This job link-edits the DB2 clear key edit procedure, DECENA00, with its corresponding ICSF callable services.

DECDB2XK

This job link-edits the DB2 CPACF exit-protected key edit procedure, DECENAA0, with its corresponding ICSF callable services.

Procedure

1. Edit the sample job that is associated with the edit procedure that you want to use.
2. Replace all lowercase JCL variables and data set names with values that are appropriate for your installation. The edit procedure name that you specify must be a unique name; it cannot be a DBD name.
3. At the bottom of the jobs, replace the variable `yyyyyyyyyy` with the

cryptographic key label that was built by your security analyst.
The encryption key label that you specify can be up to 64 characters long. If you do not use all 64 characters, include the correct number of trailing blanks before the right parenthesis that ends the parameter list. For example, if your encryption key label is 50 characters long, include 14 trailing blanks, as in this example:

```
(yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy )
```

What to do next

If you modify the encryption edit procedure, a DB2 restart is required to refresh the encryption edit procedure with the new version.

You might have to customize the job to run in your ISPF/PDF environment. For example, you might have to add your ISPF basic target libraries to the appropriate `//ISPxLIB` ddname concatenations and add `//ISPTABL` to point to the same libraries as `//ISPTLIB`.

Creating a DB2 edit procedure by using the ISPF interface

InfoSphere Guardium Data Encryption provides an ISPF interface that you can use to build the JCL job stream for an edit procedure and to specify the cryptographic key label.

Before you begin

Obtain the cryptographic key label from the security analyst who installs or administers Integrated Cryptographic Service Facility (ICSF).

Procedure

1. Issue the following command to start the product and to display the ISPF interface:

```
ex 'smpe_high_level_qualifier.SDECCEXE(DECENC04)' 'smpe_high_level_qualifier'
```

where:

smpe_high_level_qualifier.SDECCEXE

SMP/E library where the REXX EXECs for the product are installed.

smpe_high_level_qualifier

SMP/E high-level qualifier of the data set name for the SMPE libraries where InfoSphere Guardium Data Encryption was installed. The high-level qualifier is provided by the person who installs the product.

2. Press Enter to display the InfoSphere Guardium Data Encryption panel.
3. Select Build a DB2 encryption exit. to display the Data Encryption for DB2 Databases panel.
4. Select Build a standalone encryption DB2 EDITPROC.
5. Specify the following information:

Key label

Cryptographic key label. The encryption process uses this label to build encrypted data.

The ICSF key type

Cryptographic key label encryption method that you want to use. The key type options are associated with the exit routines that are provided by InfoSphere Guardium Data Encryption as follows:

- | | |
|---|---|
| 1 | DECENC00 -- DB2 DES SECURE KEY |
| 2 | DECENA00 -- DB2 CLEAR KEY NON-CPK |
| 3 | DECENB00 -- DB2 CPACF PROTECTED KEY |
| 4 | DECENCA0 -- DB2 AES SECURE KEY |
| 5 | DECENBI0 -- DB2 CPACF PROTECTED KEY WITH UNIQUE ICV |
| 6 | DECENAA0 -- DB2 CPACF EXIT-PROTECTED KEY |

CSF lib

Name of the library that contains the ICSF callable services that you are using. Refer to Encryption methods to see the ICSF callable services that are used by InfoSphere Guardium Data Encryption edit procedures.

ZAP lib

Name of the library in which the load module zap program, AMASPZAP, is stored. This program puts the encryption key label into the edit procedure.

SMP lib

Name of the SMP/E library in which the product is installed.

DB2 exit lib

Name of the DB2 exit library to store the edit procedure.

Exit name

Unique name for the edit procedure (for example, ICSFSAMP).

6. Press Enter to save your specifications and to display the job that you created.

What to do next

If you modify the field procedure, a DB2 restart is required to refresh the edit procedure with the new version.

You might have to customize the job to run in your ISPF/PDF environment. For example, you might have to add your ISPF basic target libraries to the appropriate //ISPxLIB ddname concatenations and add //ISPTABL to point to the same libraries as //ISPTLIB.

Implementing a DB2 encryption method

You can implement data encryption to protect data from being accessed by unauthorized users by using either a User Defined Function (UDF) or a field procedure.

Before you begin

You must have in your exit library an encryption method that was created by using InfoSphere Guardium Data Encryption. See “Creating a DB2 encryption method” on page 29.

Procedure

Implement your encryption method by completing its encryption procedures:

Implementing a DB2 UDF for encryption

UDF provides a number of features for encrypting data.

Table 15. UDF names and attributes

UDF name	Attributes
DECENU00	Uses a default Initial Chaining Vector (ICV).
DECENUBL	Uses a user-specified, unique ICV. The ICV is passed to the ICSF CSNBSYE and CSNBSYD service calls. Note: This UDF is exclusively for use with Binary Large Objects (BLOBs) and other DB2 large objects that have a 4-byte length field.
DECENU10	Uses a user-specified, unique ICV. The ICV is passed to the ICSF CSNBSYE and CSNBSYD service calls.

Table 15. UDF names and attributes (continued)

UDF name	Attributes
DECENUP0	Uses a user-specified, unique ICV. ICSF PAD technology provides improved performance with ICSF CSNBSYE and CSNBSYD service calls. Note: DECENUP0 requires you to increase the length of the column to be encrypted to a multiple of 16 with data type VARCHAR during encryption and decryption.
DECENURN	Uses the ICSF random number generation service to provide a cryptographic-quality random value of the specified length. DECENURN can be used in conjunction with DECENUI0 and DECENUP0 to create unique ICVs.

The following table provides examples of UDF usage lengths and the length to which the column must be increased if you are using DECENUP0:

Table 16. Column lengths and redefined VARCHARs

Column length	Redefined VARCHAR required by UDF
1 -- 15	16
16 -- 31	32
32 -- 47	48
48 -- 63	64
64 -- 79	80

The following topics provide examples of and describe the UPDATE, column mask, SELECT, VIEW, and TRIGGER features. These examples demonstrate how these features can be used in masking data.

Parameters

The following parameters are passed to UDF.

- The column type, for example CHAR or VARCHAR.
- The integer value depends on the column type and is defined as follows:
 - If the column is a fixed-length type, the integer value represents the column length.
 - If the column is a variable-length type, the value is zero. The UDF determines the actual variable column length from the length field that precedes the column data.
- The action character is either "E" (encrypt data) or "D" (decrypt data).
- The ICSF Key Label is defined in ICSF and is used by this UDF. The key label can be up to 64 characters.

The examples use the following values for parameters:

- Column type: CHAR
- Integer value: 16
- Action character: "E" (encrypt data) or "D" (decrypt data)
- ICSF Key Label: USERDEFINEDAESKEY

Encrypting data in DB2 V9

You can use the UPDATE feature to encrypt all numbers that are in a table, without affecting database availability. Use the SELECT feature to select all numbers from a table. This feature is only available for DB2 version 9.

Important: In DB2 Version 9, CHAR columns must be cast as VARCHAR before the columns can be processed by the User Defined Function (UDF). A credit card number must be cast as VARCHAR (18) to allow a 2-byte length field at the beginning of the data that was cast.

The following UDF encrypts all credit card numbers that are in a table:

```
UPDATE DB29.TABLE09
SET CREDIT_CARD_NUMBER =
    user_udfname(CAST(CREDIT_CARD_NUMBER AS VARCHAR(18)),
        0,
        CAST('E' AS CHAR(1)),
        CAST('USERDEFINEDAESKEY' AS CHAR(64)));
```

The following UDF selects all the credit card numbers from the table:

```
SELECT user_udfname (CAST(CREDIT_CARD_NUMBER AS CHAR(16)),
    0,
    CAST('D' AS CHAR(1)),
    CAST('USERDEFINEDAESKEY' AS CHAR(64)))
FROM DB29.TABLE09;
```

Encrypting data in DB2 V10

You can use the UPDATE feature to encrypt all numbers that are in a table, without affecting database availability. Use the SELECT feature to select all the numbers from a table. This feature is only available for DB2 version 10.

The following UDF encrypts all credit card numbers that are in a table:

```
UPDATE DB210.TABLE10
SET CREDIT_CARD_NUMBER =
    user_udfname(CREDIT_CARD_NUMBER,
        16,
        CAST('E' AS CHAR(1)),
        CAST('USERDEFINEDAESKEY' AS CHAR(64)));
```

The following UDF selects all the credit card numbers from a table:

```
SELECT user_udfname(CREDIT_CARD_NUMBER,
    16,
    CAST('D' AS CHAR(1)),
    CAST('USERDEFINEDAESKEY' AS CHAR(64)))
FROM DB210.TABLE10;
```

Controlling read access to the encryption UDF

You can control the amount of data that is visible to select users by using the column mask, SELECT, VIEW, and TRIGGER features.

DB2 V10 column mask:

You can use the column mask feature to control the amount of visibility that is granted to users. This feature is only available for DB2 version 10 or later.

The following example grants three possible levels of data visibility. The level is dependent on the user's authorization.

- All 16 digits of CREDIT_CARD_NUMBER are shown. This level visibility is granted to authorized users. Users become authenticated when a mask (CREDIT_CARD_NUMBER_MASK) is referenced by applications that access encrypted data.
- The last four digits of CREDIT_CARD_NUMBER are shown, with the first 12 replaced by "X." This level of visibility is for users that are authorized to see the last four digits.
- Only masked data is shown. This level of visibility is for unauthorized users. UDF is not started because the complete mask replaces the encrypted data. No additional overhead processing is needed to decrypt the data.

```
CREATE MASK CREDIT_CARD_NUMBER_MASK on DB210.TABLE10
FOR COLUMN CREDIT_CARD_NUMBER RETURN
CASE WHEN (VERIFY_GROUP_FOR_USER(SESSION_USER,userid)=1)
THEN user_udfname(CREDIT_CARD_NUMBER,16,
CAST('D' AS CHAR(1)),
CAST('USERDEFINEDAESKEY' AS CHAR(64)))
ELSE /* MASKEDUSERS */
CAST('XXXXXXXXXXXX' ||
CAST(user_udfname(SUBSTR(CREDIT_CARD_NUMBER,13,4),
4,
CAST('D' AS CHAR(1)),
CAST('USERDEFINEDAESKEY' AS CHAR(64))) AS CHAR(4))
AS CHAR(16))
END
ENABLE;
```

```
ALTER TABLE DB210.TABLE10 ACTIVATE COLUMN ACCESS CONTROL;
```

DB2 V9 and DB2 V10 SELECT:

You can use a qualified SELECT command to mask sensitive data.

The following example grants three possible levels of data visibility. The level of visibility a user is granted depends on their authorization.

- All 16 digits of CREDIT_CARD_NUMBER are shown. This level visibility is granted to authorized users. Users become authenticated when a mask (CREDIT_CARD_NUMBER_MASK) is referenced by applications that access encrypted data.
- The last four digits of CREDIT_CARD_NUMBER are shown, with the first 12 replaced by "X." This level of visibility is for users that are authorized to see the last four digits.
- Only masked data is shown. This level of visibility is for unauthorized users. The User Defined Function is not started because the complete mask replaces the encrypted data. No additional overhead processing is needed to decrypt the data.

```
SELECT
CASE WHEN (VERIFY_GROUP_FOR_USER(SESSION_USER,'userid')=1)
THEN user_udfname(CREDIT_CARD_NUMBER,16,CAST('D' AS CHAR(1),
CAST('USERDEFINEDAESKEY' AS CHAR(64)))
ELSE 'XXXXXXXXXXXX' ||
CAST(user_udfname(SUBSTR(CREDIT_CARD_NUMBER,13,4),
4,
CAST('D' AS CHAR(1)),
CAST('USERDEFINEDAESKEY' AS CHAR(64)))
AS CHAR(4))
```

DB2 V9 VIEW:

You can define a VIEW to control read access to the encryption User Defined Function (UDF). This feature is only available for DB2 version 9.

Important: Only users and applications that are authorized to read encrypted data are granted the privilege to use VIEW function. All other users and applications that select the column will see only the encrypted data. For the latter case, UDF is not started and no processing is used to decrypt data.

The following example demonstrates defining a VIEW called CREDIT_CARD_NUMBER_VIEW:

```
CREATE VIEW CREDIT_CARD_NUMBER_VIEW (CREDIT_CARD_NUMBER) AS
SELECT

    User_udfname ( CAST(CREDIT_CARD_NUMBER AS VARCHAR(18)),0,

        CAST('D' AS CHAR(1)),

        CAST('USERDEFINEDAESKEY' AS CHAR(64)))

FROM DB29.TABLE09;

COMMIT;

SELECT CREDIT_CARD_NUMBER FROM CREDIT_CARD_NUMBER_VIEW;
```

DB2 V10 TRIGGER:

You can define a TRIGGER to control access to the encryption User Defined Function (UDF). The UDF is started only when authorized by the TRIGGER logic. This feature is only available for DB2 version 10 or later.

Important: Only users and applications that are authorized to update or insert column data to be encrypted can use the TRIGGER feature. Non-authorized users can still insert or update other columns in the row.

The following example demonstrates defining a TRIGGER called CREDIT_CARD_NUMBER_TRIGGER:

```
DROP TRIGGER CREDIT_CARD_NUMBER_TRIGGER;
COMMIT;

CREATE TRIGGER CREDIT_CARD_NUMBER_TRIGGER
NO CASCADE BEFORE INSERT ON DB210.TABLE10 REFERENCING NEW
AS NEWROW FOR EACH ROW MODE DB2SQL
SET CREDIT_CARD_NUMBER =
    user_udfname (NEWROW.CREDIT_CARD_NUMBER,0,
        CAST('E' AS CHAR(1))
        ,CAST('USERDEFINEDAESKEY' AS CHAR(64)));

COMMIT;

GRANT TRIGGER ON DB210.TABLE10
TO SFCFAI;
```

Implementing a DB2 field procedure for encryption

You can implement data encryption by using the encryption field procedure as part of a table unload and reload operation. Data is encrypted but not compressed.

Before you begin

You must have in your exit library an encryption field procedure that was created by using InfoSphere Guardium Data Encryption.

About this task

You can encrypt sensitive columns that are defined to a table by completing one of the methods in the following procedure:

Procedure

1. Unload, drop, and then reload the database.
 - a. Unload the DB2 database. If the database already has a field procedure, you must unload it by using that field procedure. Make sure that the database is offline and was successfully unloaded (awaiting reload) before you proceed.
 - b. Install the encryption field procedure.
 - c. Redefine the table or tables, which involves dropping and re-creating the table or tables and all dependent objects. Specify the FIELDPROC option and the name of the encrypting field procedure.

Tip: The IBM DB2 Administration Tool can help you complete these tasks.

- d. Reload the table or tables to encrypt the data when it is loaded.
2. Use the ALTER ADD NEW COLUMN statement to point to the FIELDPROC. An application can then run and encrypt the columns without affecting DB2 availability.

Implementing a DB2 edit procedure for encryption

You can implement a DB2 encryption edit procedure either with or without compression. Encrypted data requires less disk space for storage if compressed.

Procedure

InfoSphere Guardium Data Encryption can implement encryption in one of two ways:

- With compression, if you link the encryption edit procedure that you create to an existing compression edit procedure
- Without compression, if you create only an encryption edit procedure.

Implementing a DB2 encryption edit procedure with compression

You can implement a DB2 encryption edit procedure with compression so that your encrypted data requires less disk space for storage.

Procedure

To encrypt your DB2 data with compression, customize the edit procedure driver and then install it.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

“Encryption with compression” on page 18

You can use InfoSphere Guardium Data Encryption to encrypt and compress your

DB2 or IMS data.

Related tasks:

“Creating a DB2 edit procedure for encryption” on page 37

Create a DB2 edit procedure for encryption using the ISPF interface or by editing the sample job.

Creating a DB2 edit procedure driver:

You can encrypt and compress DB2 data by creating an edit procedure driver that links a compression edit procedure to an encryption edit procedure.

Before you begin

Before you create the edit procedure driver, complete the following steps:

1. Back up your compressed data.
2. Use any pre-existing edit procedure to unload your data into its original format.

Procedure

To create the edit procedure driver, you can either use the sample job or the ISPF interface.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

“Creating a DB2 edit procedure for encryption” on page 37

Create a DB2 edit procedure for encryption using the ISPF interface or by editing the sample job.

Creating a DB2 edit procedure driver by editing the sample job:

InfoSphere Guardium Data Encryption provides a sample job that you can edit to create an edit procedure driver, which links an encryption exit routine with a compression edit procedure.

Before you begin

You must have in your exit library an encryption exit routing that was created by using InfoSphere Guardium Data Encryption.

Attention: If you do not code the link edit control statements in the correct sequence, the edit procedure driver will be unusable.

Procedure

To create the edit procedure driver by using the sample job, edit the sample job that is in 'smphlq.SDECLMD0(DECENADV)', where *smphlq* is the SMP/E high-level qualifier. Follow the instructions in the sample job.

What to do next

Implement data encryption as part of a table unload and reload operation.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

“Creating a DB2 edit procedure for encryption” on page 37

Create a DB2 edit procedure for encryption using the ISPF interface or by editing the sample job.

Creating a DB2 edit procedure driver by using the ISPF interface:

You can create the DB2 edit procedure driver by using the ISPF interface that InfoSphere Guardium Data Encryption provides.

Before you begin

You must have in your exit library an encryption exit routine that was created by using InfoSphere Guardium Data Encryption.

Procedure

1. Issue the following command to start the product and to display the ISPF interface:

```
ex 'smphlq.SDECCEXE(DECENC04)' 'smphlq'
```

where:

smphlq.SDECCEXE

SMP/E library where the REXX execs for the product are installed.

smphlq

SMP/E high-level qualifier for the product. The high-level qualifier is provided by the person who installed the product.

2. Press Enter to display the EDITPROC/exit selection panel.
3. In the **OPTION** field, specify 2 to build a DB2 compression and encryption edit procedure driver.
4. In the DB2 EDITPROC Driver Data Entry panel, specify the following information:

JCL parameters

Jobcard

Up to five lines of job card data.

Product SMP lib

SMP/E library where the edit procedure driver (DECENADV) and the compression edit procedure (DECZLDX0) are installed.

Dictionary build input

DSN1COMP library

DB2 load library that contains the DSN1COMP utility.

Source data set name

VSAM table data set that contains DB2 row data.

Object name

Name for the output dictionary object.

Driver EDITPROC input**User member name**

User-specified name of the edit procedure driver.

Library

Library to hold the user edit procedure driver. This library is usually the DB2 SDSNEXIT library.

Compression EDITPROC input**User member name**

Name of the compression edit procedure.

Library

Library to hold the compression edit procedure. This library is usually the DB2 SDSNEXIT library.

Encryption EDITPROC input**User member name**

Name of the encryption edit procedure.

Library

Enter the library that contains the encryption edit procedure. This library is usually the DB2 SDSNEXIT library.

5. Press Enter to save your specifications and to display the job that you just created.

What to do next

Implement data encryption as part of a table unload and reload operation.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

“Creating a DB2 edit procedure for encryption” on page 37

Create a DB2 edit procedure for encryption using the ISPF interface or by editing the sample job.

Implementing DB2 data encryption with compression:

You can use an edit procedure driver to encrypt and compress your DB2 data.

Before you begin

You must have in your exit library an edit procedure driver that was created by using InfoSphere Guardium Data Encryption.

About this task

You implement data encryption with compression as part of a table unload and reload operation. After you unload a table and before reloading it, specify the EDITPROC option and the name of your customized edit procedure driver. When

you reload the database, each row compressed by the compression edit procedure that is called by the edit procedure driver and encrypted by the encryption exit routine that is called by the edit procedure driver.

To implement data encryption with compression:

Procedure

1. Unload the DB2 table or tables that you want to encrypt. If the database already has an edit procedure, you must use that edit procedure to unload it. Make sure that the database is offline and has been successfully unloaded (awaiting reload) before you proceed.
2. Install the customized edit procedure driver. You must install the edit procedure driver after the database is unloaded and before it is reloaded.
3. Redefine the table or tables, which involves dropping and re-creating the table or tables and all dependent objects. Specify the EDITPROC option and the name of the edit procedure driver.
4. Reload the table or tables to encrypt the data when it is loaded.
5. Validate your output.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

“Creating a DB2 edit procedure for encryption” on page 37

Create a DB2 edit procedure for encryption using the ISPF interface or by editing the sample job.

“Creating a DB2 edit procedure driver” on page 46

You can encrypt and compress DB2 data by creating an edit procedure driver that links a compression edit procedure to an encryption edit procedure.

Implementing a DB2 encryption edit procedure without compression

You implement data encryption by using the encryption edit procedure as part of a table unload and reload operation. Data is encrypted but not compressed.

Before you begin

You must have in your exit library an encryption edit procedure that was created by using InfoSphere Guardium Data Encryption.

Procedure

1. Unload the DB2 database. If the database already has an edit procedure, you must unload it by using that edit procedure. Make sure that the database is offline and was successfully unloaded (awaiting reload) before you proceed.
2. Install the encryption edit procedure.
3. Redefine the table or tables, which involves dropping and re-creating the table or tables and all dependent objects. Specify the EDITPROC option and the name of the encrypting edit procedure.

Tip: The IBM DB2 Administration Tool can help you complete these tasks.

4. Reload the table or tables to encrypt the data when it is loaded.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

“Creating a DB2 edit procedure for encryption” on page 37

Create a DB2 edit procedure for encryption using the ISPF interface or by editing the sample job.

Chapter 4. Encrypting data in IMS

You can use InfoSphere Guardium Data Encryption to encrypt your IMS data.

Procedure

Encrypt data in IMS by creating an encryption exit routine, which is an IMS Segment Edit/Compression exit routine. You can implement encryption in one of two ways:

- Without compression if you create only an encryption exit routine
- With compression if you link the encryption exit routine that you create to an existing compression exit routine

Creating an IMS encryption exit routine

To encrypt IMS data, create an encryption exit routine, which is an IMS Segment Edit/Compression exit routine, that includes a user-specified cryptographic key label.

Procedure

To create the encryption exit routine, you can either use the sample jobs or the ISPF interface.

Requirement: After you use SMP/E to apply maintenance, you must re-create the encryption exit routine. Use the same cryptographic key label and exit routine name.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Creating an IMS encryption exit routine by editing a sample job

InfoSphere Guardium Data Encryption provides sample jobs that you can edit to create an encryption exit routine, which is an IMS Segment Edit/Compression exit routine.

Before you begin

Obtain the cryptographic key label from the security analyst who installs or administers Integrated Cryptographic Service Facility (ICSF).

About this task

You can create encryption exit routines by editing any of the following sample jobs, which are available in PDS *smphlq.SDECSAMP*, where *smphlq* is the SMP/E high-level qualifier for the product:

DECIMSCB

This job link-edits the IMS CPACF protected key exit routines, DECENAA1, DECENBB1, and DECENB01 with the corresponding Integrated Cryptographic Service Facility (ICSF) callable services.

DECIMSJJB or DECIMSSK

These jobs link-edit the IMS secure key exit routine, DECENC01, with its corresponding ICSF callable services.

DECIMSCK

This job link-edits the IMS clear key exit routine, DECENA01, with its corresponding ICSF callable services.

Procedure

1. Edit the sample job that is associated with the InfoSphere Guardium Data Encryption exit routine that you want to use.
2. Replace all lowercase JCL variables and data set names with values that are appropriate for your installation. The exit routine name that you specify must be a unique name; it cannot be a DBD name.
3. At the bottom of the jobs, replace the variable `yyyyyyyyyy` with the cryptographic key label that was built by your security analyst.

The cryptographic key label that you specify can be up to 64 characters long. If you do not use all 64 characters, include the correct number of trailing blanks before the right parenthesis that ends the parameter list. For example, if your encryption key label is 50 characters long, include 14 trailing blanks, as shown here:

```
(yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy      )
```

What to do next

If you modify the encryption exit routine, you must recycle the exit library before your changes take effect. This action is necessary because IMS stores the exit routine in memory after IMS loads it for the first time.

You might need to customize the job to run in your ISPF/PDF environment. For example, you might need to add your ISPF basic target libraries to the appropriate `//ISPxLIB` ddname concatenations and add `//ISPTABL` to point to the same libraries as `//ISPTLIB`.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Creating an IMS encryption exit routine by using the ISPF interface

InfoSphere Guardium Data Encryption provides an ISPF interface that you can use to create an encryption exit routine.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Building a stand-alone encryption IMS exit routine

InfoSphere Guardium Data Encryption provides an ISPF interface that you can use to create a stand-alone exit routine for encrypting data.

Before you begin

Obtain the cryptographic key label from the security analyst who installs or administers Integrated Cryptographic Service Facility (ICSF).

About this task

To build the encryption exit routine by using ISPF panels, follow these steps:

Procedure

1. Issue the following command to start the product and to display the ISPF interface:

```
ex 'smpe_high_level_qualifier.SDECCEXE(DECENC04)' 'smpe_high_level_qualifier'
```

where:

smpe_high_level_qualifier.SDECCEXE

SMP/E library where the REXX execs for the product are installed.

smpe_high_level_qualifier

SMP/E high-level qualifier of the data set name for the SMPE libraries where InfoSphere Guardium Data Encryption was installed. The high-level qualifier is provided by the person who installs the product.

2. Press Enter to display the InfoSphere Guardium Data Encryption panel.
3. Select Build an IMS encryption exit.
4. Press Enter to display the Data Encryption for IMS Databases panel.
5. Select Build a standalone encryption IMS exit.
6. Specify the ICSF encryption key to be implemented by entering the key label and key type.
7. Specify the encryption JCL parameters:

CSF 1ib

Name of the library that contains the ICSF callable services that you are using. Refer to “Encryption methods” on page 4 to see the ICSF callable services that are used by InfoSphere Guardium Data Encryption exit routines.

ZAP 1ib

Name of the library in which the load module zap program, AMASPZAP, is stored. This program puts the encryption key label into the encryption exit routine.

SMP 1ib

Name of the SMP/E library in which the product is installed.

IMS exit 1ib

Name of the exit library to store the encryption exit routine.

Exit name

Unique name for the encryption exit routine (for example, ENCREXIT).

8. Press Enter to save your specifications and to display the next IMS Encryption Exit driver data entry panel.

What to do next

If you modify the encryption exit routine, you must recycle the exit library before your changes take effect. This action is necessary because IMS stores the exit routine in memory after IMS loads it for the first time.

You might be required to customize the job to run in your ISPF/PDF environment. For example, you might be required to add your ISPF basic target libraries to the appropriate //ISPxLIB ddname concatenations and add //ISPTABL to point to the same libraries as //ISPTLIB.

Building a compression and encryption IMS exit routine

InfoSphere Guardium Data Encryption provides an ISPF interface that you can use to create an exit routine for compressing and encrypting data.

Before you begin

Important: The following IMS exits are involved in this process:

- The driver routine
- The encryption routine
- The compression routine (usually preexisting)

The driver routine's **COMPRTN** parameters, which are specified in the DBDGEN, should match those of the superseded compression exit's **COMPRTN** parameters, except for the name. The name must be unique to the driver.

Obtain the cryptographic key label from the security analyst who installs or administers Integrated Cryptographic Service Facility (ICSF).

About this task

To build the encryption exit routine by using ISPF panels, follow these steps:

Procedure

1. Issue the following command to start the product and to display the ISPF interface:

```
ex 'smpe_high_level_qualifier.SDECCEXE(DECENC04)' 'smpe_high_level_qualifier'
```

where:

smpe_high_level_qualifier.SDECCEXE

SMP/E library where the REXX execs for the product are installed.

smpe_high_level_qualifier

SMP/E high-level qualifier of the data set name for the SMPE libraries where InfoSphere Guardium Data Encryption was installed. The high-level qualifier is provided by the person who installs the product.

2. Press Enter to display the InfoSphere Guardium Data Encryption panel.
3. Select Build an IMS encryption exit.
4. Press Enter to display the Data Encryption for IMS Databases panel.
5. Select Build an IMS compression/encryption exit.
6. Specify the encryption JCL parameters:

SMP 1ib

Name of the SMP/E library in which the product is installed.

Driver exit input

New user member name for the driver exit routine.

Name of the library where the driver exit routine is stored.

Compression exit input

Name of the compression exit routine.

Name of the library where the compression exit routine is stored.

Encryption exit input

Name of the encryption exit routine.

Name of the library where the encryption exit routine is stored.

7. Press Enter to save your specifications and to display the next IMS Encryption Exit/Compression Exit driver data entry panel.

What to do next

If you modify the encryption exit routine, you must recycle the exit library before your changes take effect. This action is necessary because IMS stores the exit routine in memory after IMS loads it for the first time.

You might be required to customize the job to run in your ISPF/PDF environment. For example, you might be required to add your ISPF basic target libraries to the appropriate //ISPxLIB ddname concatenations and add //ISPTABL to point to the same libraries as //ISPTLIB.

Encrypting IMS data without compression

You implement data encryption by using the encryption exit routine as part of a database unload and reload operation. Data is encrypted but not compressed.

Before you begin

You must have in your exit library an encryption exit routine, which you can build by using InfoSphere Guardium Data Encryption.

About this task

After you unload a database and before reloading it, include the name of your encryption exit routine in a DBD SEGM statement to assign the segment for encryption. After you reload the database with the changed statement, your segments will be encrypted when they are inserted.

Note: You can choose not to encrypt segment keys by specifying `COMPRTN=(encrexit,DATA,INIT)`. If you choose not to encrypt segment keys, IMS can search the database faster because it does not have to decrypt a segment in order to inspect its cryptographic key label. However, the key and any fields of the segment that precede the key are not encrypted.

Procedure

1. Unload the IMS database. If the database already has a Segment Edit/Compression exit routine, you must use that exit routine to unload it. Make sure that the database is offline and has been successfully unloaded (awaiting reload) before you proceed.
2. After the encryption exit routine has been built, change the DBD SEGM statement by adding a `COMPRTN` parameter, as shown in the following figure.

In the `encrexit` keyword of the `COMPRTN` parameter, specify the name of the encryption exit routine that you built.

```
DBD    NAME=DH81SK02,ACCESS=(HIDAM,VSAM)
        DATASET DD1=DH81SK02,DEVICE=3330
        SEGM    NAME=SEGMENTA,PARENT=0,BYTES=(70,55),PTR=TB,
                COMPRTN=(encrexit,KEY,INIT)
        FIELD    NAME=(FIELDA,SEQ,U),BYTES=11,START=1
        DBDGEN
        FINISH
        END
```

Figure 9. Adding the `COMPRTN` parameter to the DBD

3. Install the encryption exit routine.
4. Use the changed DBD to reload the database. Segments that have been specified for encryption are encrypted as they are inserted.
5. Validate your output.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

“Creating an IMS encryption exit routine” on page 51

To encrypt IMS data, create an encryption exit routine, which is an IMS Segment Edit/Compression exit routine, that includes a user-specified cryptographic key label.

Encrypting IMS data with compression

You can encrypt your IMS data with compression so that your encrypted data requires less disk space for storage.

Procedure

To encrypt your IMS data with compression, create the exit routine driver and then install it.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

“Encryption with compression” on page 18

You can use InfoSphere Guardium Data Encryption to encrypt and compress your DB2 or IMS data.

Related tasks:

“Creating an IMS encryption exit routine” on page 51

To encrypt IMS data, create an encryption exit routine, which is an IMS Segment Edit/Compression exit routine, that includes a user-specified cryptographic key label.

Creating an IMS exit routine driver

You can encrypt and compress IMS data by creating an exit routine driver that links an existing compression exit routine to an encryption exit routine.

Before you begin

Before you begin creating the exit driver, complete the following steps:

1. Back up your compressed data.
2. Use any preexisting exit routine to unload your data into its original format.

About this task

To create the exit driver, you can either use the sample jobs or the ISPF interface.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

“Creating an IMS encryption exit routine” on page 51

To encrypt IMS data, create an encryption exit routine, which is an IMS Segment Edit/Compression exit routine, that includes a user-specified cryptographic key label.

Creating an IMS exit routine driver by editing the sample job

InfoSphere Guardium Data Encryption provides a sample job that you can edit to create an exit routine driver. This driver links an encryption exit routine with a compression exit routine.

Before you begin

You must have in your exit library an encryption exit routine that was created by using InfoSphere Guardium Data Encryption.

Attention: If you do not code the link edit control statements in the correct sequence, the exit routine will be unusable.

Procedure

Edit member DECIMSDV in PDS *smphlq*.SDECSAMP, where *smphlq* is the SMP/E high-level qualifier for the product. Follow the instructions that are documented at the beginning of the sample job. When you modify the sample job, you need values for the IBMCOMPX statement and the IBMENCRX statement.

- To locate the IBMCOMPX value, browse the COMPXNAM member, which is located in the COMPRESSIONEXITLIB library, and search for the string HCOZLDX.
- To locate the IBMENCRX value, browse the ENCRXNAM member, which is located in the ENCRYPTIONEXITLIB library, and search for the string DECEN.

What to do next

Implement data encryption with compression as part of a database unload and reload operation.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

“Creating an IMS encryption exit routine” on page 51

To encrypt IMS data, create an encryption exit routine, which is an IMS Segment Edit/Compression exit routine, that includes a user-specified cryptographic key label.

Creating an IMS exit routine driver by using the ISPF interface

You can create the IMS exit routine driver by using the ISPF interface that InfoSphere Guardium Data Encryption provides.

Before you begin

You must have in your exit library an encryption exit routine that was created by using InfoSphere Guardium Data Encryption.

Procedure

1. Issue the following command to start the product and to display the ISPF interface:

```
ex 'smphlq.SDECCEXE(DECENC04)' 'smphlq'
```

where:

smphlq.SDECCEXE

Is the SMP/E library where the REXX execs for the product are installed.

smphlq

Is the SMP/E high-level qualifier for the product. The high-level qualifier is provided by the person who installs the product.

2. Press Enter to display the EDITPROC/exit selection panel.
3. In the **Option** field, specify 3 to build an IMS compression/encryption exit driver. The Exit Driver for IMS Databases panel is displayed.
4. In the Exit Driver for IMS Databases panel, specify the following information:

JCL parameters**Jobcard**

Enter up to five lines of job card data.

Product SMP lib

Enter the name of the SMP/E library in which the exit routine driver DECENCDV is installed.

Driver exit input**New user member name**

Enter a name for the exit routine driver.

Library

Enter the library to store the exit routine driver.

Compression exit input**User member name**

Enter the name of the compression exit routine that you want to use.

Library

Enter the library that contains the compression exit routine.

Encryption exit input**User member name**

Enter the name of the Segment Edit/Compression exit routine for encryption.

Library

Enter the library that contains the Segment Edit/Compression exit routine for encryption.

5. Press Enter to save your specifications and to display the job that you just created.

What to do next

Implement data encryption with compression as part of a database unload and reload operation.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

“Creating an IMS encryption exit routine” on page 51

To encrypt IMS data, create an encryption exit routine, which is an IMS Segment Edit/Compression exit routine, that includes a user-specified cryptographic key label.

Implementing IMS data encryption with compression

You can use an exit routine driver to encrypt and compress your IMS data.

Before you begin

You must have in your exit library an exit routine driver that was created by using InfoSphere Guardium Data Encryption.

About this task

You implement data encryption as part of a database unload and reload operation. After you unload a database and before reloading it, include the name of an exit routine driver in a DBD SEGM statement. When you reload the database, each segment is first compressed with the compression exit routine that is called by the exit routine driver, and then each segment is encrypted by the encryption exit routine that is called by the exit routine driver.

Procedure

1. Unload the IMS database by using a compression exit routine. Ensure that the database is offline and has been successfully unloaded (awaiting reload) before you proceed.
2. After the IMS exit routine driver has been built, change the DBD SEGM statement by adding a COMPTN parameter, as shown in the following figure. In the drvexit keyword of the COMPTN parameter, specify the name of the

IMS exit routine driver.

```
DBD NAME=DH81SK02,ACCESS=(HIDAM,VSAM)
      DATASET DD1=DH81SK02,DEVICE=3330
      SEGM NAME=SEGMENTA,PARENT=0,BYTES=(70,55),PTR=TB,
            COMPRTN=(drvexit,KEY,INIT)
      FIELD NAME=(FIELDA,SEQ,U),BYTES=11,START=1 LCHILD
      DBDGEN
      FINISH
      END
```

Figure 10. Adding the COMPRTN parameter to the DBD

3. Install the exit routine driver. You must install the exit routine driver after the database is unloaded and before it is reloaded.
4. Use the changed DBD to reload the database. IMS segments are compressed and then encrypted as they are inserted.
5. Validate your output.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

“Creating an IMS encryption exit routine” on page 51

To encrypt IMS data, create an encryption exit routine, which is an IMS Segment Edit/Compression exit routine, that includes a user-specified cryptographic key label.

“Creating an IMS exit routine driver” on page 57

You can encrypt and compress IMS data by creating an exit routine driver that links an existing compression exit routine to an encryption exit routine.

Chapter 5. Upgrading the cryptographic key encryption

You can enhance the security and performance of an application by upgrading the encryption method for the cryptographic key label that your data is encrypted with.

About this task

You can upgrade the encryption method for the cryptographic key label in the following situations:

- After a hardware change makes it possible to use an advanced method.
- If it is necessary to change your existing key label for security purposes.

Attention: When you upgrade the encryption method for the cryptographic key label, do not delete the previously used cryptographic key. Retain it in case you must restore the archived data later. Your security administrator should define the new key with a different cryptographic key label.

To upgrade the encryption method for the cryptographic key label:

Procedure

1. Back up your data. For DB2, back up your tables and edit procedures. For IMS, back up your databases and current encryption exit routines.
2. Unload and decrypt data with current encryption exit routines on the current system.
3. Install new hardware and configure Integrated Cryptographic Service Facility (ICSF).
4. Create a new encryption exit routine.
5. Load and encrypt data on the new system.

Related concepts:

“Encryption methods” on page 4

You can use InfoSphere Guardium Data Encryption to create an encryption method to encrypt your data. Several options for encryption methods are available to meet the specific needs of your environment.

Related tasks:

“Creating an IMS encryption exit routine” on page 51

To encrypt IMS data, create an encryption exit routine, which is an IMS Segment Edit/Compression exit routine, that includes a user-specified cryptographic key label.

“Creating a DB2 edit procedure for encryption” on page 37

Create a DB2 edit procedure for encryption using the ISPF interface or by editing the sample job.

Chapter 6. Troubleshooting

InfoSphere Guardium Data Encryption issues messages that can help you diagnose and solve processing errors with Segment Edit/Compression routines, the batch TSO job step, and Integrated Cryptographic Service Facility (ICSF).

IMS abend codes

Use the abend codes that are issued by InfoSphere Guardium Data Encryption to help you diagnose, troubleshoot, and solve InfoSphere Guardium Data Encryption problems.

If pseudo-abend processing has not been requested by the caller, the pseudo-abend will be changed into an actual abend. When InfoSphere Guardium Data Encryption cannot continue normal operations, it issues the following abend code:

2990

Explanation: A user-written Segment Edit/Compression exit routine detected a processing error while attempting compression or expansion services.

System action: The application ends abnormally.

User response: Ensure that the Segment Edit/Compression exit routine was built and linked properly. Use reason codes to diagnose problems.

IMS reason codes

IMS reason codes can accompany an abend that is issued in the IMS environment.

The reason code identifier is X'C4C5C3nn', where *nn* is the unique reason code.

C4C5C310 PC routine not available

Explanation: A required PC routine was not available.

System action: Pseudo-abend 2990 is issued.

User response: The Guardium Data Encryption subsystem has not been successfully initialized. Check that message DEC7000I has been issued to verify that the subsystem has been properly defined and successfully initialized. If the subsystem was successfully initialized and the problem continues to occur, contact IBM Software Support.

Communication Vector Table (CCVT) was invalid.

System action: Pseudo-abend 2990 is issued.

User response: Contact your ICSF Administrator or IBM Software Support.

**C4C5C311 SECONDARY CVT ADDRESS
CVTABEND WAS ZERO**

Explanation: The address that points to the Secondary Communication Vector Table (SCVT) was invalid.

System action: Pseudo-abend 2990 is issued.

User response: Contact your ICSF Administrator or IBM Software Support.

C4C5C314 PC routine not found

Explanation: A required PC routine was not found.

System action: Pseudo-abend 2990 is issued.

User response: The Guardium Data Encryption subsystem has not been successfully initialized. Check that message DEC7000I has been issued to verify that the subsystem has been properly defined and successfully initialized. If the subsystem was successfully initialized and the problem continues to occur, contact IBM Software Support.

**C4C5C312 SCVTCCVT (ICSFCVT) ADDRESS
WAS ZERO**

Explanation: The address that points to the ICSF

C4C5C315 ICSF/MVS NOT ACTIVE

Explanation: ICSF is not started.

System action: Pseudo-abend 2990 is issued.

User response: Start the ICSF address space.

C4C5C316 NOT SETUP FOR CLEAR KEY ENCRYPTION

Explanation: The ICSF key in the CKDS file is invalid.

System action: Pseudo-abend 2990 is issued.

Programmer response: Validate that you created a Clear Key, using CLRDES in place of DATA, in your CKDS file. If you want to run Secure Key Data Encryption, you must use a different IMS exit routine. Contact IBM Software Support for additional help with setting up your encryption keys.

C4C5C381 Unable to retrieve the encryption tool name token.

Explanation: The encryption tool name token could not be retrieved when performing an encryption or decryption request.

System action: Pseudo-abend 2990 is issued.

User response: Contact IBM Software Support.

C4C5C382 Unable to obtain work area storage.

Explanation: An obtain storage request for an internal work area failed.

System action: Pseudo-abend 2990 is issued.

User response: Contact IBM Software Support.

C4C5C383 Unable to create the encryption tool name token.

Explanation: The encryption tool name token could not be created.

System action: Pseudo-abend 2990 is issued.

User response: Contact IBM Software Support.

C4C5C397 Error retrieving encryption key.

Explanation: The hardware encryption key could not be located in the internal cache.

System action: Pseudo-abend 2990 is issued.

User response: Contact IBM Software Support.

C4C5C398 KMC instruction failed

Explanation: A KMC (Cipher Message with Chaining) instruction failed with a non-zero return code.

System action: Pseudo-abend 2990 is issued.

User response: Contact your ICSF Administrator or IBM Software Support.

C4C5C399 KMC instruction failed

Explanation: A KMC (Cipher Message with Feedback) instruction failed with a non-zero return code.

System action: Pseudo-abend 2990 is issued.

User response: Contact your ICSF Administrator or IBM Software Support.

C4C5C3A1 ENCRYPTION INPUT LENGTH < 2 BYTES

Explanation: The input data was invalid.

System action: Pseudo-abend 2990 is issued.

Programmer response: Correct the error in the database segment and run the job again.

C4C5C3A2 INPUT SOURCE LENGTH ERROR

Explanation: The input supplied to the exit has a length error.

System action: Pseudo-abend 2990 is issued.

User response: Check that the same exit used to encrypt the data is being used to decrypt the data before contacting IBM Software Support.

C4C5C3A5 INVALID FUNCTION CODE

Explanation: This is an internal error.

System action: Pseudo-abend 2990 is issued.

User response: Contact IBM Software Support.

C4C5C3A6 SEQUENCE FIELD NOT IN SEGMENT

Explanation: This is an internal error.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3A7 SEGMENT LENGTH NEGATIVE

Explanation: This is an internal error.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3A8 ENCRYPTION SERVICE NOT FOUND

Explanation: The ICSF microcode was not installed on this machine.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact your security administrator.

C4C5C3A9 "INIT" PARM NOT SPECIFIED ON DBD COMPRTN STATEMENT

Explanation: The COMPRTN statement has no valid INIT parameter.

System action: Pseudo-abend 2990 is issued.

Programmer response: Add the INIT parameter to the DBD COMPRTN statement, regenerate the DBD, and then run the job again.

C4C5C3B1 INPUT SOURCE LENGTH ERROR

Explanation: The input supplied to the exit has a length error.

System action: Pseudo-abend 2990 is issued.

Programmer response: Check that the same exit used to encrypt the data is being used to decrypt the data before contacting IBM Software Support.

C4C5C3B2 INPUT SOURCE LENGTH ERROR FOR DECRYPT CALL

Explanation: The input for decryption processing is invalid.

System action: Pseudo-abend 2990 is issued.

User response: Check that the same exit used to encrypt the data is being used to decrypt the data before contacting IBM Software Support.

C4C5C3BB ENCRYPTION RC=12, REAS=0: ICSF NOT STARTED or ICSF STARTED, BUT CRYPTOGRAPHIC UNITS NOT ACCESSIBLE

Explanation: ICSF is installed but is not accessible.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact your ICSF administrator.

C4C5C3BC ENCRYPTION RC=12, REAS=8: ICSF SERVICES NOT AVAILABLE ON CURRENT HARDWARE. YOUR REQUEST CAN NOT BE PROCESSED

Explanation: The ICSF hardware or software is not installed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact your ICSF administrator.

C4C5C3C1 OBTAIN STORAGE FAILURE

Explanation: A storage obtain failed.

System action: Pseudo-abend 2990 is issued.

User response: Contact your IMS System Administrator to ensure the task has access to sufficient storage before contacting IBM Software Support.

C4C5C3C4 CHECK FOR ICSF FUNCTION REQUIRED TO DO ENCRYPTION FAILED

Explanation: The ICSF hardware flags in the CCVT indicate that the ICSF functionality as defined to your environment is not sufficient to perform encryption or decryption.

System action: Pseudo-abend 2990 is issued, or a U840 abend is issued.

Programmer response: Contact your IBM Software Support.

C4C5C3CF OBTAIN STORAGE FAILURE FOR SUB-POOL 1

Explanation: A unique storage obtain requesting sub-pool 1 failed.

System action: Pseudo-abend 2990 is issued.

User response: Contact your IMS System Administrator to ensure the task has access to sufficient storage before contacting IBM Software Support.

C4C5C3D0 OBTAIN STORAGE ERROR FOR COMPRESSED DATA

Explanation: An obtain storage request to hold the intermediate compressed data failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3D2 OBTAIN STORAGE ERROR FOR DECRYPTED DATA

Explanation: An obtain storage request to hold the intermediate decrypted data failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3D4 INVALID FUNCTION CODE

Explanation: This is an internal error.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3DE OBTAIN STORAGE ERROR FOR DECRYPTED DATA

Explanation: An obtain storage request to hold the intermediate decrypted data has failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3DF OBTAIN STORAGE ERROR FOR DECRYPTED DATA

Explanation: An obtain storage request to hold the intermediate decrypted data has failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3DF OBTAIN STORAGE ERROR FOR COMPRESSED DATA

Explanation: An obtain storage request to hold the intermediate compressed data has failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3E0 Batch encryption key cache could not be located.

Explanation: The encryption key cache anchor could not be located.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3E1 Online encryption key cache could not be located.

Explanation: The encryption key cache anchor could not be located.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3E2 The encryption key cache anchor is invalid.

Explanation: The encryption key cache anchor contains an invalid control block identifier.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3E3 Storage obtain for online encryption key cache anchor failed.

Explanation: An IMS obtain storage request to hold the encryption key cache anchor failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3E4 Create for online encryption key cache anchor failed.

Explanation: The encryption key cache anchor directory entry could not be created.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3E5 Storage obtain for batch encryption key cache anchor failed.

Explanation: An obtain storage request to hold the encryption key cache anchor failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3E8 Storage obtain for online encryption key cache has table failed.

Explanation: An IMS obtain storage request to hold the encryption key cache has table failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3E7 Storage obtain for online encryption key cache data failed.

Explanation: An IMS obtain storage request to hold the encryption key cache data failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3E8 Storage obtain for online encryption key cache has table failed.

Explanation: An IMS obtain storage request to hold the encryption key cache has table failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3E9 Storage obtain for batch encryption key cache has data failed.

Explanation: An obtain storage request to hold the encryption key cache data failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3EA Storage obtain for batch encryption key cache data failed.

Explanation: An obtain storage request to hold the encryption key cache data failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3EB Storage obtain for batch encryption key cache hash table failed.

Explanation: An obtain storage request to hold the encryption key cache hash table failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3EC An obtain storage request to hold the encryption key cache hash table failed.

Explanation: An obtain storage request to hold the encryption key cache has table failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3F0 Storage obtain failed.

Explanation: An obtain storage request for work area needed to retrieve an encryption key failed.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3F1 Hardware MSA3 facility is not available.

Explanation: The hardware Message-Security-Assist extension 3 (MSA3) facility is not supported by the current processor.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

C4C5C3F2 Protected key processing error.

Explanation: As part of the key retrieval process, an internal test is performed to identify the encryption algorithm that is associated with the requested key label. The test was unable to successfully use the retrieved CPACF-protected key in either an AES or a DES encryption operation.

System action: Pseudo-abend 2990 is issued.

Programmer response: Contact IBM Software Support.

Guardium Data Encryption subsystem messages

Use the messages that are issued by the Guardium Data Encryption subsystem to help you diagnose, troubleshoot, and solve problems.

DEC7000I SSI INITIALIZATION COMPLETE

Explanation: The Guardium Data Encryption subsystem has been created or enabled.

User response: No action is required.

**DEC7002E SSI ERROR ACTION=xxxxxxx,
RC=yyyyyyyy,RSN=zzzzzzzz**

Explanation: Guardium Data Encryption subsystem initialization processing has encountered an error. The error is described by:

xxxxxxx

z/OS service being performed

yyyyyyyy

Service return code in hexadecimal

zzzzzzzz

Service reason code in hexadecimal

System action: The request that was issued to the subsystem interface was not successful.

User response:

- For a LXRES, a problem occurred allocating a system LX. If the system LX pool has been exhausted, increase the number of available system LXs.
- For a BLDL, STORAGE, or LOAD error, a problem occurred when attempting to locate and load module DECSSI21. Correct the error so that DECSSI21 can be located and loaded into storage.

For any other errors, contact IBM Software Support.

**DEC7003E INVALID INITIALIZATION
PARAMETER SPECIFIED, IGNORED**

Explanation: An invalid initialization parameter was passed to Guardium Data Encryption subsystem initialization processing.

System action: The invalid initialization parameter is ignored.

User response: Correct the initialization parameter value that is specified for the subsystem.

**DEC7004I SSI PREVIOUSLY ESTABLISHED IS
DISABLED**

Explanation: The Guardium Data Encryption

subsystem has been removed or is not enabled.

User response: No action is required.

**DEC7006I SSI UPDATE NOT PERFORMED,
ROUTINE IS UNCHANGED**

Explanation: The Guardium Data Encryption

subsystem initialization processing determined that the subsystem PC routine to be loaded is the same as the previously loaded and active subsystem PC routine. No change was made.

User response: No action is required.

DB2 reason codes

DB2 reason codes can accompany an abend in the DB2 environment.

DB2 issues SQL code -652 as a result of non-zero return code from an edit procedure. When SQL code -652 is issued, DB2 will normally display the DB2 SQLCA control block. The SQLERRD (6) fullword contains the reason code from the edit procedure. Check the hexadecimal display of this value.

**C4C5C301 SCVTCCVT (ICSFCVT) ADDRESS
WAS ZERO**

Explanation: The address that points to the ICSF Communication Vector Table (CCVT) was invalid.

System action: The EDITPROC is abnormally terminated, and DB2 EDITPROC error-handling takes control.

Programmer response: Contact your System Administrator or IBM Software Support.

C4C5C302 ICSF/MVS NOT INITIALIZED

Explanation: ICSF started but did not initialize.

System action: The EDITPROC is abnormally terminated, and DB2 EDITPROC error-handling takes control.

Programmer response: Wait a few minutes and submit the job again.

C4C5C304 ICSF/MVS NOT ACTIVE

Explanation: ICSF is not started.

System action: The EDITPROC is abnormally terminated, and DB2 EDITPROC error-handling takes control.

Programmer response: Start the ICSF address space.

**C4C5C305 SECONDARY CVT ADDRESS
CVTABEND WAS ZERO**

Explanation: The address that points to the Secondary Communication Vector Table (SCVT) was invalid.

System action: The EDITPROC is abnormally terminated, and DB2 EDITPROC error-handling takes control.

Programmer response: Contact your System Administrator or IBM Software Support.

**C4C5C306 NOT SETUP FOR CLEAR KEY
ENCRYPTION**

Explanation: The ICSF key in the CKDS file is invalid.

System action: The EDITPROC is abnormally terminated, and DB2 EDITPROC error-handling takes control.

Programmer response: Validate that you created a Clear Key using CLRDES in place of DATA, in your CKDS file. To run Secure Key Data Encryption you need to use DB2 exit DECENC00. Contact IBM Software Support for additional help with setting up your encryption keys.

**C4C5C307 KMC INSTRUCTION NOT
SUPPORTED**

Explanation: The KMC encryption instruction is not supported.

System action: The EDITPROC is abnormally terminated and DB2 EDITPROC error-handling takes control.

User response: Contact IBM Software Support.

C4C5C30D STORAGE OBTAIN FAILED

Explanation: A get storage for the intermediate work area in driver program DECENA00 failed.

System action: The driver is abnormally terminated and DB2 EDITPROC error-handling takes control.

Programmer response: Contact your System Administrator or the IBM Software Support.

**C4C5C31D WRONG LENGTH EXPLWA CONTROL
BLOCK PASSED FROM DB2**

Explanation: DB2 passed an invalid length EXPLWA control block to the EDITPROC. A length of 512 is expected.

System action: The EDITPROC is abnormally terminated and DB2 EDITPROC error-handling takes control.

User response: DB2 APARs PK71816 and PM08638 must be applied to DB2 to correct this problem. Contact your DB2 System Administrator to ensure that all relevant DB2 maintenance has been applied before contacting IBM Software Support.

C4C5C31A Unidentifiable algorithm for exit key label

Explanation: As part of the key retrieval process, an internal test is performed to identify the encryption algorithm of the key label defined by the exit. The test was unable to use the defined key successfully in either an AES or DES encryption operation.

System action: The EDITPROC is terminated and DB2 EDITPROC error-handling takes control.

User response: Review the cryptographic algorithm of the key label that is specified in the encryption exit routine. AES and DES algorithms are supported.

C4C5C31B Ambiguous algorithm for exit key label

Explanation: As part of the key retrieval process, an internal test is performed to identify the encryption algorithm of the key label defined by the exit. The test was able to use the defined key successfully in both AES and DES encryption operations, meaning the algorithm cannot be clearly identified.

System action: The EDITPROC is terminated and DB2 EDITPROC error-handling takes control.

User response: The key defined in the encryption exit routine cannot be used with DECENAA0. Rebuild the exit specifying a different key label, or delete and recreate the key identified by the key label if it is not used by other exits.

C4C5C31C Unable to perform CPACF-wrapping for secure key

Explanation: Secure keys must be wrapped by the CSNBKRR2 ICSF callable service in conjunction with the PROTKEY option. The key label specified in the encryption exit routine is associated with a secure key and the CSNBKRR2 call for key retrieval failed.

System action: The EDITPROC is terminated and DB2 EDITPROC error-handling takes control.

User response: Ensure that the CSFKEYS profile covering the key label provided in the exit has the ICSF segment fields SYMCPCAFWRAP and SYMCPCAFRET set to YES. Ensure that ICSF maintenance OA50450, which supports the PROTKEY option, is installed.

C4C5C320 An Internal failure occurred with the IEANTRT service

Explanation: An internal error occurred.

System action: The EDITPROC is terminated and DB2 EDITPROC error-handling takes control.

User response: Contact IBM Software Support.

C4C5C321 Invalid encryption input row data.

Explanation: The input row data to be encrypted cannot be expanded to add encryption exit metadata. The row cannot be encrypted without it being added.

System action: The EDITPROC is abnormally terminated and DB2 EDITPROC error-handling takes control.

User response: Contact IBM Software Support.

C4C5C31F INPUT RECORD LENGTH FROM DB2 LESS THAN 16 BYTES

Explanation: DB2 passed an EDITILEN (input row record length) value of less than 16 bytes. The encryption algorithms require a 16-byte minimum input length.

System action: The EDITPROC is abnormally terminated, and DB2 EDITPROC error-handling takes control.

User response: Add a dummy column to increase the row record length to at least 16 bytes, or choose another EDITPROC for the table.

C4C5C31F Invalid decryption input row data

Explanation: The input row data to be decrypted does not contain encryption exit metadata. The row cannot be decrypted without it.

System action: The EDITPROC is abnormally terminated, and DB2 EDITPROC error-handling takes control.

User response: Contact IBM Software Support.

DECZLDX0

The following reason codes are issued for edit procedure DECZLDX0.

C4C5C308 ZERO COMPRESSION DICTIONARY ADDRESS DETECTED

Explanation: A compression dictionary was detected with an address of zero.

System action: The EDITPROC is abnormally terminated and DB2 EDITPROC error-handling takes control.

User response: Check the dictionary build and EDITPROC link output for errors, then retry.

C4C5C309 COMPRESSION DICTIONARY NOT ON A PAGE BOUNDARY

Explanation: A zero length dictionary was linked to the EDITPROC.

System action: The EDITPROC is abnormally terminated and DB2 EDITPROC error-handling takes control.

User response: Check the dictionary build and EDITPROC link output for errors, then retry.

C4C5C30A INVALID DICTIONARY EYECATCHER FOUND

Explanation: An invalid eyecatcher was detected in the compression dictionary.

System action: The EDITPROC is abnormally terminated and DB2 EDITPROC error-handling takes control.

User response: Check the dictionary build and EDITPROC link output for errors, then retry.

C4C5C30B Z/OS HARDWARE DATA COMPRESSION FACILITY UNAVAILABLE

Explanation: The hardware data compression facility was not available.

System action: The EDITPROC is abnormally terminated and DB2 EDITPROC error-handling takes control.

User response: Contact your System Administrator to ensure that the hardware data compression facility is installed before contacting IBM Software Support.

C4C5C30C CMPSC INSTRUCTION EXPANSION CALL FAILED

Explanation: The CMPSC instruction failed during an expansion call.

System action: The EDITPROC is abnormally terminated and DB2 EDITPROC error-handling takes control.

User response: Check that the same compression dictionary used to compress the data is being used to expand the data before contacting IBM Software Support.

IMS and DB2 batch TSO job step messages

The batch TSO job step can issue messages about parameter errors, the AMASPZAP program, and the ICSF key.

DEC100E TOO MANY PARAMETERS SUPPLIED. REMOVE UNNECESSARY PARAMETERS AND RESUBMIT JOB.

Explanation: The batch TSO job step has too many parameters.

System action: None.

Programmer response: Check the sample JCL to determine the parameters that are needed. Remove unnecessary parameters and submit the job again.

Module: REXX exec

DECENC02

Explanation: The batch TSO job step is missing a parameter.

System action: None.

Programmer response: Check to see if the DBMS name (IMS or DB2) or encryption key label parameter is missing. Correct the parameter and submit the job again.

Module: DECENC02

DEC120E "IMS" OR "DB2" NOT SPECIFIED AS FIRST PARAMETER. CORRECT AND RESUBMIT JOB.

Explanation: The first parameter that was passed to the batch TSO job step was invalid.

System action: None.

DEC110E DBMS KEYWORD OR KEY LABEL VALUE MISSING. ADD MISSING PARAMETER AND RESUBMIT JOB.

Programmer response: Ensure that the first parameter is either DB2 or IMS and submit the job again.

Module: DECENC02

**DEC130E ZAP FAILED WITH RETURN CODE
nn. ICSF KEY NOT IMPLEMENTED —
NOTIFY IBM SUPPORT.**

Explanation: The batch TSO job step failed to invoke the AMASPZAP program.

System action: None.

Programmer response: Collect the output from the AMASPZAP program and notify your system programmer and IBM Software Support about the problem.

Module: DECENC02

**DEC140I ICSF KEY SUCCESSFULLY
IMPLEMENTED FOR: DBMS = *dbms*,
KEY LABEL = *key*.**

Explanation: The user exit was successfully built with the selected encryption key label.

System action: None.

Programmer response: Implement encryption for the DBMS.

Module: DECENC02

Common ICSF reason codes

Integrated Cryptographic Service Facility (ICSF) reason codes accompany ICSF errors.

Correcting common ICSF errors

Reason code 271C is one of the most common reason codes ICSF might issue. Reason code 271C can be issued for the following reasons:

- The key type for the clear key exit routine was not defined as CLRDES or CLRAES. To correct this problem, redefine the key with a CLRDES or CLRAES key type.
- The key type for the secure key exit routine was not defined as DATA. To correct this problem, redefine the key with a DATA key type.
- The ICSF CKDS was not REFRESHED since the key was created. To correct this problem, select option 8 from the ICSF menu, then select option 4 and specify the name of the CKDS that is being used.
- The name of the edit procedure or COMPTN is incorrect or the incorrect secure key or clear key job is run. For example, DB2 sample jobs DECDB2SK for secure key and DECDB2CK for clear key.

To correct either of these problems, verify that the key was created in the CKDS with the name that was specified in the job by looking at the output of the ICSF KGUP job in the data set that is identified by the DIAG DD name. If the ICSF KGUP data set is not used, look at the ICSF CKDS directly by using either an OEM product or the IDCAMS PRINT IDS command.

Reason code 271C and other common ICSF errors are documented in this section.

Related information:

 z/OS Internet Library

For more information about API return codes and reason codes, see Cryptographic Services ICSF Application Programmer's Guide.

**0BFB Encryption key cannot be used in
high-performance operations**

Explanation: The encryption key for the specified key label cannot be rewrapped into a CPACF-protected encryption key. This can occur when the CSFKEYS security profile that protects the key label specifies

either SYMCPACFWRAP=NO or SYMCPACFRET=NO.

User response: Update the CSFKEYS security profile protecting the key label to specify both SYMCPACFWRAP=YES and SYMCPACFRET=YES to allow the encryption key to be rewrapped into a CPACF-protected encryption key.

0C04 Encryption key cannot be rewrapped

Explanation: The encryption key for the specified key label cannot be rewrapped into a CPACF-protected encryption key. This can occur when the CSFKEYS security profile protecting the key label specifies either SYMCPACFWRAP=NO or SYMCPACFRET=NO.

User response: Update the CSFKEYS security profile protecting the key label to specify both SYMCPACFWRAP=YES and SYMCPACFRET=YES to allow the encryption key to be rewrapped into a CPACF-protected encryption key.

0C81 A secure key is not supported

Explanation: The specified key label is for a secure encryption key, but only a clear encryption key is supported. This error will occur if the new function that is provided by ICSF APAR OA50450 is not available.

User response: Use a key label for a clear encryption key, or install ICSF APAR OA50450.

271C KEY LABEL NOT FOUND

Explanation: The key label that is specified in the edit procedure that included the DECENA00 program was not found in the ICSF CKDS or PKDS.

User response: Ensure that the key label that is specified in the edit procedure matches the one that is defined in ICSF or that the correct edit procedure is being used, or invoke the ICSF REFRESH function by selecting option 8 and then selecting option 4 on the ICSF ISPF menus. If these actions do not correct the error, contact IBM Software Support.

2738 Invalid encryption key version

Explanation: The specified key label encryption key does not have an acceptable version number. This occurs if a DES key label is specified and only AES encryption keys are supported, or if an AES key label is specified and only DES encryption keys are supported.

User response: Use a Guardium Data Encryption routine that supports the specified key label encryption key type, or use a key label with an encryption key type that is supported by the Guardium Data Encryption routine.

E380 KEY LABEL NOT FOUND

Explanation: The key label that is specified in the edit procedure that included the DECENA00 program was not found in the ICSF CKDS or PKDS.

User response: Ensure that the key label that is specified in the edit procedure matches the one that is defined in ICSF; or that the correct edit procedure is being used; or invoke the ICSF refresh function by selecting option 8, then option 4, in the ICSF ISPF menus. If these actions do not resolve the error, contact IBM Software Support.

3E84 Not authorized to use the encryption key

Explanation: The security product has denied access for the specified key label. This occurs when the CFSKEYS security profile that protects the key label does not allow read access based on the user security environment at the time of the ICSF request.

User response: To use the key label, read access must be granted.

DB2 SQL codes

DB2 SQL codes return information about the statement execution.

SQL codes that are issued by DB2 can be found on the IBM Knowledge Center: https://www-01.ibm.com/support/knowledgecenter/SSEPEK_11.0.0/com.ibm.db2z11.doc.codes/src/tpc/db2z_n.dita. The SQL statement includes the SQL return code (SQLCODE) and the SQLSTATE, which indicate whether statement execution was successful.

The following SQLSTATE RETURN CODES are issued by the product and are separated by the name of the encryption method for which they are issued.

DECENF00

FIELDPROC DECENF00 product-issued SQLSTATE return codes: The following reason codes are issued for field procedure DECENF00.

DEF0 INVALID FUNCTION CODE

Explanation: Internal DB2 error.

System action: FIELDPROC is abnormally terminated.

User response: Contact IBM Software Support.

DEF1 ICSF ENCR FAILED RS=nnnnnnnnnn

Explanation: ICSF encryption failure.

System action: FIELDPROC is abnormally terminated.

User response: Inform system programmer of ICSF RS=nnnnnnnn value.

DECENU00, DECENUBL, DECENUI0, DECENUP0, and DECENURN

UDF product-issued SQLSTATE return codes: The following error messages are issued for User Defined Functions DECENU00, DECENUBL, DECENUI0, DECENUP0, and DECENURN. All UDFs issue the same DEC08 SQLSTATE return code followed by the message text.

DEC08 Scratchpad length less than 500

Explanation: DB2 passed a scratchpad length less than 500.

System action: UDF is abnormally terminated.

User response: Inform the DB2 system programmer.

DEC08 Length integer greater than 32000

Explanation: The user-supplied UDF length parameter exceeds DB2's 32000 limit for a UDF column length.

System action: UDF is abnormally terminated.

User response: Use the tool's encryption EDITPROC solution to encrypt large objects

DEC08 1 or more input parameters missing

Explanation: One or more parameters are missing from the UDF invocation SQL.

System action: UDF is abnormally terminated.

User response: Correct the parameter list and rerun the UDF.

DEC08 Function not E(Encrypt) or D(Decrypt)

Explanation: The user-supplied UDF requested action parameter is invalid

System action: UDF is abnormally terminated.

User response: Correct the parameter and rerun the UDF

DEF2 ICSF DECR FAILED RS=nnnnnnnnnn

Explanation: ICSF decryption failure.

System action: FIELDPROC is abnormally terminated.

User response: Inform the system programmer of ICSF RS=nnnnnnnn value.

DEC08 ICSF retcode=nnnnnnnnn, reas=yyyyyyyy

Explanation: ICSF encryption or decryption error.

System action: UDF is abnormally terminated.

User response: Inform the system programmer of ICSF return code and reason values.

DEC08 ICSF NOT ACTIVE

Explanation: ICSF is not started.

System action: UDF is abnormally terminated.

User response: Inform the ICSF programmer that there is an ICSF configuration issue.

DEC08 Length parm must be zero

Explanation: The user-supplied UDF length parameter is non-0 and is not supported by the UDF.

System action: UDF is abnormally terminated.

User response: Correct the length parameter and rerun the UDF.

DEC08 Length parameter out of range (1 – 255)

Explanation: The user-supplied UDF length parameter is 0, or greater than 255, and is not supported by the UDF.

System action: UDF is abnormally terminated.

User response: Correct the length parameter and rerun the UDF.

IMS exit messages

The following messages are issued by IMS exits.

DEC0001E DECENxx1 encryption exit error.

Explanation: The IMS encryption exit, DECENxx1, encountered an ICSF processing error.

User response: Provide IBM Software Support with the failing DECENxx1 name for their use in determining the problem. Module: Program DECENxx1.

DEC0002E **Function: funcname Function code: 000000nn**

Explanation: The IMS encryption exit, DECENxx2, was processing a funcname call. Function code 000000nn is the IMS internal function code value.

User response: Provide IBM Software Support with the funcname and code for their use in determining the problem.

DEC0003E **ICSF debug data:**

Explanation: This message provides ICSF debug data.

User response: No action is required.

DEC0004E **Retcode: 000000xx Reason code: xxxxxxxx**

Explanation: The IMS encryption exit, DECENxx1, encountered an ICSF processing error. Retcode 000000xx is the ICSF return code and reason code xxxxxxxx is the ICSF reason code in decimal format.

User response: Provide IBM Software Support with the return and reason codes for their use in determining the problem. Module: Program DECENxx1.

DEC0005E **Key len: 000000xx Text length: xxxxxxxx**

Explanation: The IMS encryption exit, DECENxx1, used a key label with a key length of 000000xx. The text length that was processed is specified by the value xxxxxxxx in decimal format.

User response: Provide IBM Software Support with the key and text length for their use in determining the problem.

DEC0006E **The IMS U2990 message also displays a reason code.**

Explanation: This message indicates that an IMS Pseudo Abend U2990 will also display the ICSF reason code in hexadecimal format.

User response: Provide IBM Software Support with

the pseudo abend code and hexadecimal reason code for their use in determining the problem. Module: Program DECENxx1.

DEC0007E **DECSSI21 PC ERROR: ffffffff - RC=xxxxxxx, RSN=yyyyyyyy, DIAG=zz**

Explanation: The Encryption Tool subsystem PC encountered an error condition while obtaining an encryption key for an encryption exit routine.

fffffff The function or service for which the error occurred. The function or service will be:

- CSNBKRR: ICSF service CSNBKRR did not complete normally.
- CSNBKRR2: ICSF service CSNBKRR2 did not complete normally.
- KMC: The KMC hardware instruction did not complete normally.
- FACLSA3: The processor does not support the MSA3 hardware facility that is required by the PC routine.

RC=xxxxxxx

The return code for the error in hexadecimal format.

RSN=yyyyyyyy

The reason code for the error in hexadecimal format.

DIAG=zz

Internal diagnostic information for use by service personnel.

User response: For an ICSF service error, some of the common ICSF error reason codes are documented in this user's guide. For any others, use the return and reason code information that is documented in the Cryptographic Services ICSF Application Programmer's Guide, ICSF and cryptographic coprocessor return and reason codes, to determine the appropriate response. For a KMC or FACLSA3 error, use the reason code that is documented in this user's guide to determine the appropriate response.

Gathering diagnostic information

Before you report a problem with InfoSphere Guardium Data Encryption to IBM Software Support, gather the appropriate diagnostic information.

Procedure

Provide the following information for all InfoSphere Guardium Data Encryption problems:

- A clear description of the problem and the steps that are required to re-create the problem
- All messages that were issued as a result of the problem

- Product release number and the number of the last program temporary fix (PTF) that was installed
- The version of DB2 or IMS that you are using and the type and version of the operating system that you are using

Provide additional information based on the type of problem that you experienced:

For online abends, provide the following information

- A screen shot of the panel that you were using when the abend occurred
- The job log from the TSO session that encountered the abend
- The job log from the server
- A description of the task that you were doing before the abend occurred

For errors in batch processing, provide the following information

- The complete job log
- Print output
- Contents of the any data sets that were used during the processing

For errors with crypto keys or functions, provide the following information

- A screen shot of the ICSF coprocessor panel that shows the status of the coprocessor
- Results from the IDCAMS:

```
print ids('nameofyour.ckds.dataset') fromkey(yyyyyyyyy)
tokey(yyyyyyyyy)
```

- Name of the exit routine that was used
- Status of the ICSF setup, including the FMID and information about allowed key sizes

Notices

This information was developed for products and services offered in the U.S.A.

This material may be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created

programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com)[®] are trademarks or registered marks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at: <http://www.ibm.com/legal/copytrade.shtml>.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions:

Applicability: These terms and conditions are in addition to any terms of use for the IBM website.

Personal use: You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use: You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights: Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Privacy policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at <http://www.ibm.com/privacy> and IBM's Online Privacy Statement at <http://www.ibm.com/privacy/details> the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM

Software Products and Software-as-a-Service Privacy Statement” at <http://www.ibm.com/software/info/product-privacy>.

Index

A

- abend codes 63
- About this information v
- access control 30
- accessibility 19
- Administration Tool 49
- Advanced Encryption Standard
 - See AES
- advantages
 - encryption methods 27
- AES 4, 25
- AMASPZAP program 4

C

- callable services (ICSF) 4
 - CSNBKRR 4
- changes to documentation 20
- CHECKAUTH installation option 9, 15
- Cipher Message with Chaining (KMC) 4
- CKDS
 - requirement 23
 - restriction 9, 15
- clear key encryption methods 4
- column mask 42
 - DECENU00 42
 - mask 40
 - UDF 42
 - user defined function 40, 42
- compare encryption methods 27
- compression 18
 - implementing with encryption
 - DB2 48
 - IMS 59
- COMPRTN parameter 15, 55, 59
- considerations
 - DB2 9
- cookie policy 77
- CP Assist for Cryptographic Functions (CPACF) 23, 25
- CPACF protected key exit routines 4
- create function SQL 33
- create function SQL JCL 33
 - creating 33
 - DECENU00 33
 - user defined function 33
- CREATE TABLE statement 9
- Crypto Express2 Coprocessor (CEX2C) 23, 25
- Crypto Express3 Coprocessor (CEX3C) 23
- cryptographic key data set
 - See CKDS
- cryptographic key labels
 - DB2 9
 - IMS 15
 - upgrading encryption of 61
- CSNBDEC callable service 4
- CSNBENC callable service 4
- CSNBKRR callable service 4, 9

- CSNBSYD callable service 4
- CSNBSYE callable service 4

D

- Data Encryption Algorithm
 - See DES
- Data Encryption Standard
 - See DES
- data governance solutions 1
- data visibility 42, 43
 - user defined function 43
- DB2
 - edit procedure
 - creating 30
 - sample jobs 37, 38
 - edit procedure driver 46
 - ISPF interface 47
 - sample job 46
 - encryption method
 - creating 30
 - ISPF interface 31, 36
 - sample jobs 37
 - errors 9
 - field procedure
 - ISPF interface 36
 - hardware compression 58
 - ISPF interface 39
 - reason codes 68, 72
 - SQL codes 72
 - user defined function
 - ISPF interface 31
- DB2 Administration Tool 49
- DB2 Tools 1, 27
- DBD COMPRTN parameter 15, 59
- DBD SEGM statement
 - COMPRTN parameter 55
- DECDB2CK edit procedure 37, 38
- DECDB2CL edit procedure 37, 38
- DECDB2JB edit procedure 37, 38
- DECDFCL
 - creating 35
 - encryption methods 35
 - field procedures 35
 - samples jobs 35
- DECENA00 edit procedure 27
 - DB2 clear key process flow
 - process flow 9
 - overview 4
 - using 37, 38, 39
- DECENA01 exit routine
 - overview 4
 - using 51, 52, 53, 54
- DECENAA0 edit procedure 27
- DECENAA0 exit routine 4
- DECENAA1 exit routine 4
- DECENB00 edit procedure
 - overview 4
 - using 37, 38, 39
- DECENB01 exit routine
 - overview 4
- DECENB01 exit routine (*continued*)
 - using 51, 52, 53, 54
- DECENBB1 exit routine 4
- DECENC00 edit procedure
 - overview 4
 - using 37, 38, 39
- DECENC01 exit routine
 - overview 4
 - using 51, 52, 53, 54
- DECENCA0 edit procedure
 - overview 4
- DECENF00 field procedure
 - messages 72
 - overview 4
 - using 36
- DECENU00
 - implementing 40
- DECENU00 field procedure 27
- DECENU00 user defined function 27
 - overview 4
 - using 31
- DECENU00, DECENUBL, DECENUI0, DECENUP0, and DECENURN user defined function
 - messages 73
- DECENUBL
 - implementing 40
- DECENUBL user defined function 27
- DECIMSCB exit routine 51
- DECIMSCCK exit routine 51
- DECIMSJJB exit routine 51
- decryption
 - DB2 9
 - IMS 15
 - overview 2
- DECZLDX0 edit procedure
 - messages 70
- DES 4, 25
- diagnostic information
 - gathering 74
- disadvantages
 - encryption methods 27
- documentation
 - accessing 19
- documentation changes 20

E

- edit procedure
 - ISPF interface 39
- edit procedure clause 9
- edit procedure driver 46, 48
 - ISPF interface 47
 - overview 18
 - sample job 46
- edit procedure encryption
 - implementing 45
- edit procedures 2, 9
 - creating 30
 - DECDB2CK 37, 38
 - DECDB2CL 37, 38

- edit procedures *(continued)*
 - DECDB2JB 37, 38
 - DECDB2XK 38
 - DECENAA00
 - See DECENAA00 edit procedure
 - See DECENAA0 edit procedure
 - DECENB00
 - See DECENB00 edit procedure
 - DECENC00
 - See DECENC00 edit procedure
 - linking with compression exit routine 48
- EDITPROC clause 48, 49
- encrypting
 - user defined function 42
- encryption
 - DB2 40
 - clear key process flow 9
 - description 9
 - implementation 27
 - process flow 9
 - with compression 45
 - without compression 45, 49
 - DECENAA00 45
 - DECENB00 45
 - DECENC00 45
 - DECENCA0 45
 - DECENF00 45
 - FIELDPROC 45
 - implementing 45
 - IMS
 - description 15
 - implementation 51
 - process flow 15
 - with compression 56
 - without compression 55
 - overview 2
 - UDF 40
 - user defined function 40
 - with compression 18
- encryption exit routines
 - creating
 - IMS 51
 - DECIMSJB exit routine 51
 - IMS
 - ISPF interface 52, 53, 54
 - sample jobs 51
 - linking with compression exit 59
 - linking with compression exit routine 48
- encryption key labels
 - DB2 35, 37, 38
 - IMS 15, 51
- encryption method
 - DB2 39
 - ISPF interface 39
- encryption methods 2, 4, 27
 - creating 30
 - DB2 30
 - DB2
 - ISPF interface 31, 36
 - sample jobs 37, 38
 - DECDB2CK 37, 38
 - DECDB2CK edit procedure 37, 38
 - DECDB2CL 37, 38
 - DECDB2CL edit procedure 37, 38
 - DECDB2JB 37, 38

- encryption methods *(continued)*
 - DECDB2JB edit procedure 37, 38
 - DECDB2XK 38
 - DECDB2XK edit procedure 38
 - DECENAA00
 - See DECENAA00 edit procedure
 - See DECENAA0 edit procedure
 - DECENB00
 - See DECENB00 edit procedure
 - DECENC00
 - See DECENC00 edit procedure
 - DECIMSCB exit routine 51
 - DECIMSCK exit routine 51
- encryption standards 4
- exit codes
 - IMS 73
- exit driver routines
 - linking with compression exit 59
- exit routine driver 57
 - ISPF interface 58
 - sample job 57
- exit routines
 - creating
 - IMS 51
 - DECENAA01
 - See DECENAA01 exit routine
 - DECENB01
 - See DECENB01 exit routine
 - DECENC01
 - See DECENC01 exit routine
 - DECIMSCB 51
 - DECIMSCK 51
 - DECIMSJB 51
- EXITLIB
 - IMS 15

F

- field procedure
 - encrypting 45
- field procedure encryption 45
- field procedures 2, 35
 - creating 30, 35
 - DB2 30
- creating encryption methods
 - DB2 35
- samples jobs 35
- FIELDPROC clause 9

H

- hardware compression 58
- hardware requirements 23
- HIDAM index databases 15

I

- IBM DB2 Administration Tool 49
- ICSF 2
 - callable services 4
 - CSNBKRR 4, 9
 - CHECKAUTH installation option 9
 - KEYAUTH installation option 9
 - requirement 24
- implementing encryption 40
 - DB2 40

- implementing encryption *(continued)*
 - UDF 42
 - user defined function 42
- implementing encryption method 40
- IMS

- encryption exit routine
 - creating 51
 - ISPF interface 52, 53, 54
 - sample jobs 51
- exit routine driver 57
 - ISPF interface 58
 - sample job 57
- Segment Edit/Compression exit routine
 - creating 51
 - ISPF interface 52, 53, 54
 - sample jobs 51
- IMS Tools 1
- indexes 9
- InfoSphere Guardium Data Encryption
 - overview 1
- installation 23
- Integrated Cryptographic Service Facility
 - See ICSF
- ISPF interface
 - DB2 edit procedure driver 47
 - encryption exit routine
 - IMS 52, 53, 54
 - encryption method
 - DB2 31, 36
 - IMS exit routine driver 58
- ISPF panel
 - creating encryption methods
 - DB2 35

J

- JCL (job control language)
 - DECDB2CK edit procedure 37, 38
 - DECDB2CL edit procedure 37, 38
 - DECDB2JB edit procedure 37, 38
 - DECIMSCB exit routine 51
 - DECIMSCK exit routine 51
 - DECIMSJB exit routine 51

K

- key labels
 - DB2 9
 - IMS 15
 - upgrading encryption of 61
- KEYAUTH installation option 9, 15
- keyboard shortcuts 19
- KMC (Cipher Message with Chaining) 4, 9

L

- legal notices
 - cookie policy 77
 - notices 77
 - programming interface information 77
 - trademarks 77
- links
 - non-IBM Web sites 78

LOBs 9

M

mask

DECENU00 42

mask column 45

masking data 42, 43

members

DECDB2CK edit procedure 37, 38

DECDB2CL edit procedure 37, 38

DECDB2JB edit procedure 37, 38

DECDB2XK edit procedure 38

DECENA00 edit procedure

See DECENA00 edit procedure

See DECENAA0 edit procedure

DECENA01

See DECENA01 exit routine

DECENB00

See DECENB00 edit procedure

DECENB01

See DECENB01 exit routine

DECENC00 edit procedure

See DECENC00 edit procedure

DECENC01

See DECENC01 exit routine

DECENCA0 edit procedure

See DECENCA0 edit procedure

DECIMSCB exit routine 51

DECIMSCK exit routine 51

DECIMSJJB exit routine 51

messages

abend codes 63

diagnosing errors

batch TSO job step 70

DB2 encryption 68, 72

ICSF 71

IMS encryption 63, 73

REXX 70

troubleshooting 63

My Support 19

N

notices 77

O

overview 1

Data Encryption for IMS and DB2

Databases 1

P

parameters

COMPRTN 15, 55, 59

PCI X Cryptographic Coprocessor

(PCIXCC) 23, 25

performance 25

problems

diagnostic information about 74

process flow

DB2 9

IMS 15

processors

requirements 23

programming interface information 77

programs

AMASPZAP 4

PTFs 24

R

read access

DECENU00 42

reason codes

DB2 68

ICSF 71

IMS 63

requirements

hardware 23

software 24

restrictions

DB2 9

IMS 15

ROWIDs 9

S

sample column mask 42

sample job

creating encryption methods

DB2 35

sample jobs

creating encryption method

DB2 35, 37, 38

DB2 edit procedure driver 46

DECDB2CK edit procedure 37, 38

DECDB2CL edit procedure 37, 38

DECDB2JB edit procedure 37, 38

DECDB2XK edit procedure 38

DECIMSCB exit routine 51

DECIMSCK exit routine 51

DECIMSJJB exit routine 51

IMS exit routine driver 57

sample members 4

screen readers and magnifiers 19

SDSNEXIT library 9

secure key encryption methods 4

security 30

Segment Edit/Compression exit

routines 2

creating 51

DECENA01

See DECENA01 exit routine

DECENB01

See DECENB01 exit routine

DECENC01

See DECENC01 exit routine

DECIMSCB 51

DECIMSCK 51

DECIMSJJB 51

linking with compression exit 59

select

DECENU00 42

SELECT 43

DECENU00 43

mask 40

user defined function 40, 43

SMP/E maintenance 30

software requirements 24

SQL codes

DB2 72

SQL CREATE TABLE statement 9

standards for encryption 4

summary of changes 20

support

required information 74

T

TDES 4, 25

See Triple DES encryption standard

technotes 19

Tools 1, 27

trademarks 77

trigger

DECENU00 42

TRIGGER

DECENU00

DB2 44

features

user defined function 44

mask 40

masking 44

DB2 44

UDF 44

user defined function 44

UDF 44

user defined function 40, 44

Triple Data Encryption Algorithm

See Triple DES

Triple Data Encryption Standard

See Triple DES

Triple DES 4

troubleshooting

messages 63

U

UDF

creating 30

implementing 40

UPDATE 42

implementing 40

user defined function 40

UPDATE feature

implementing 42

UDF 42

user defined function 2, 30

creating 30

implementing 40

sample job 30

user defined functions

creating 30

DB2 30

V

V10 column mask 42

V9 VIEW 44

version 10

UPDATE 42

version 10 UPDATE 42

view

DECENU00 42

- VIEW 44
 - DECENU00 44
 - function 40
 - mask 40
 - masking 44
 - UDF 44
 - user defined function 40, 44
- VIEW function 44
- visibility
 - user defined function 44

Z

- z/OS KMC instruction 9



Product Number: 5655-P03

Printed in USA

SC19-3219-06

