Database Encryption on z/OS
Session C04

Thomas J. Hubbard (Tom)
Product Specialist
Rocket Software, Inc.
thubbard@rocketsoftware.com

© 2016 IBM Corporation

Agenda

- Why encrypt your data?
- What is database encryption?
- z/OS Encryption
- Encryption Algorithms
- Keys, keys, which key for what?
- Implementing DB2 Database Encryption
- Summary

Why encrypt your data?

© 2016 IBM Corporation


Really?
You can do this online now.
P.C.VEY

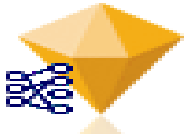
Security & Privacy: Strategic Imperative

Valuable Data

10X

Medical records worth 10X Credit Cards

Heightened Competition & Brand Protection



2-6%

Post-breach customer churn rates

From Compliance to Investment

3-14%

Security spend % of IT varies based on strategic importance

Board-level focus

66%

2/3 of Corporate Directors blame CEO for a breach.

New Delivery Channels

IoT

Mobile

Cloud

Why Should Data be Encrypted?

- **Keeps sensitive information confidential**
  - –Insider threat
  - –Lost/stolen tape or disk
  - –Disk being repaired (Solid-state disks fail in a read-only state)

- **Addresses Standards**
  - – PCI DSS HIPAA NERC NORTH AMERICAN ELECTRIC RELIABILITY CORPORATION GLBA COMPLIANCE others
  - –Privacy breach disclosure laws (e.g. EU Privacy Disclosure Directive)
  - –Protection of financial data

- **Simplifies end-of-life-of-media scenarios**
  - –Destroy the key and the data is unusable
    - Cryptographic Erasure (NIST SP800-88)
  - –Reducing media disposal costs

Threats to Database Data

- **Privileged User access to DB2/IMS Data from outside of DB2/IMS**
  - –Access to Linear VSAM datasets
- **Privileged User access to DB2 Data via SQL**
- **Abuse of privilege without business Need to Know**
- **Theft of privileged user credentials**
- **External Threats**
  - –SQL Injection (Hacking)
  - –Pfishing and spearing attacks directed at credentials
- **Movement of data outside of DB2/IMS**
  - –Backups, Unloads, Replication, Clones, Test data

Data Protection – A Team Effort

- **Initial concerns and questions**
  - –Why encrypt the data at all?
- **What is the right database encryption solution?**
- **Would the application need to be modified?**
- **Would application performance be impacted?**
- **Which group will own key management?**
- **What is the security team's role?**
- **What is the audit team's role?**
- **What is systems programmer role?**
  - –z/OS, DB2, IMS
- **What is the DBA's role?**

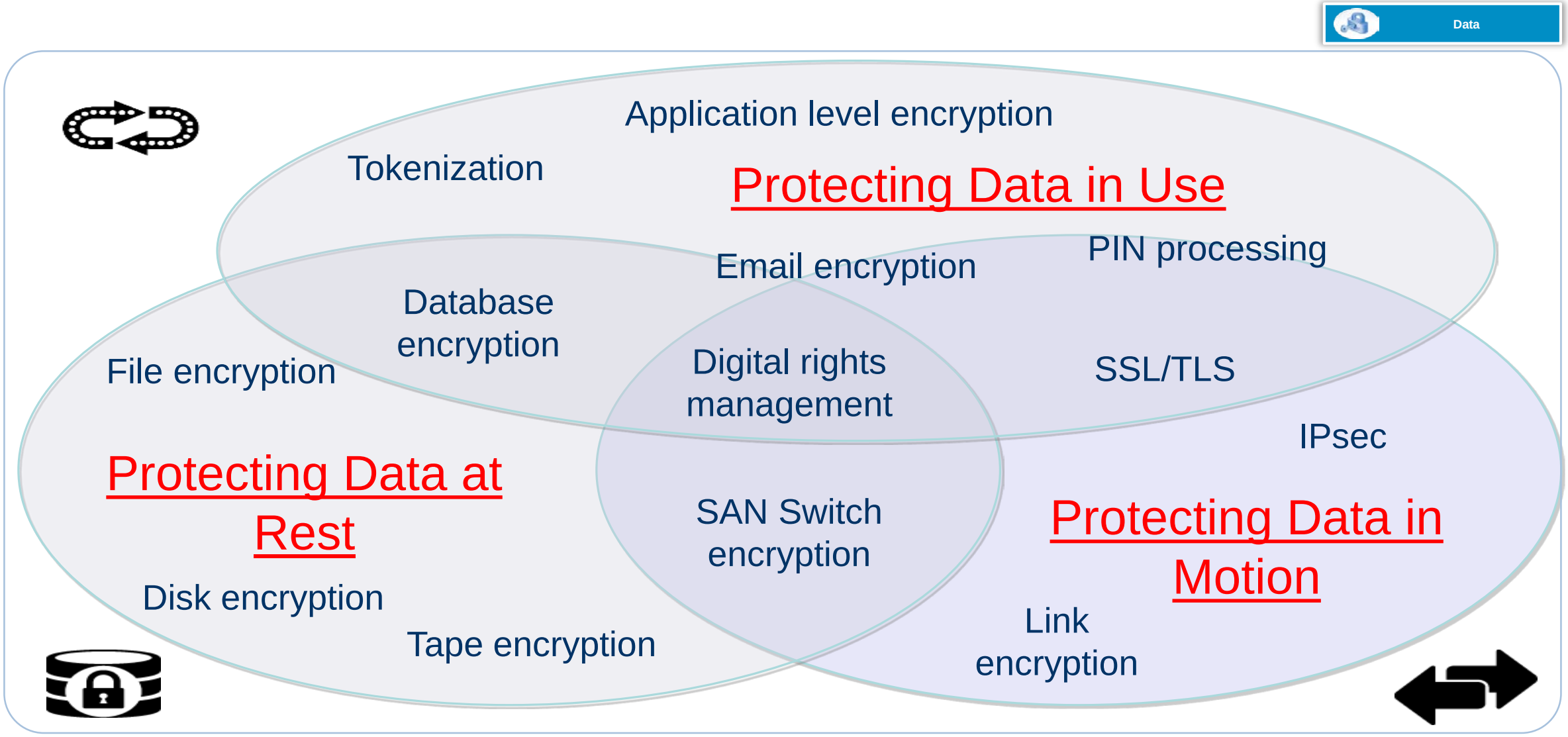
What is database encryption?

© 2016 IBM Corporation

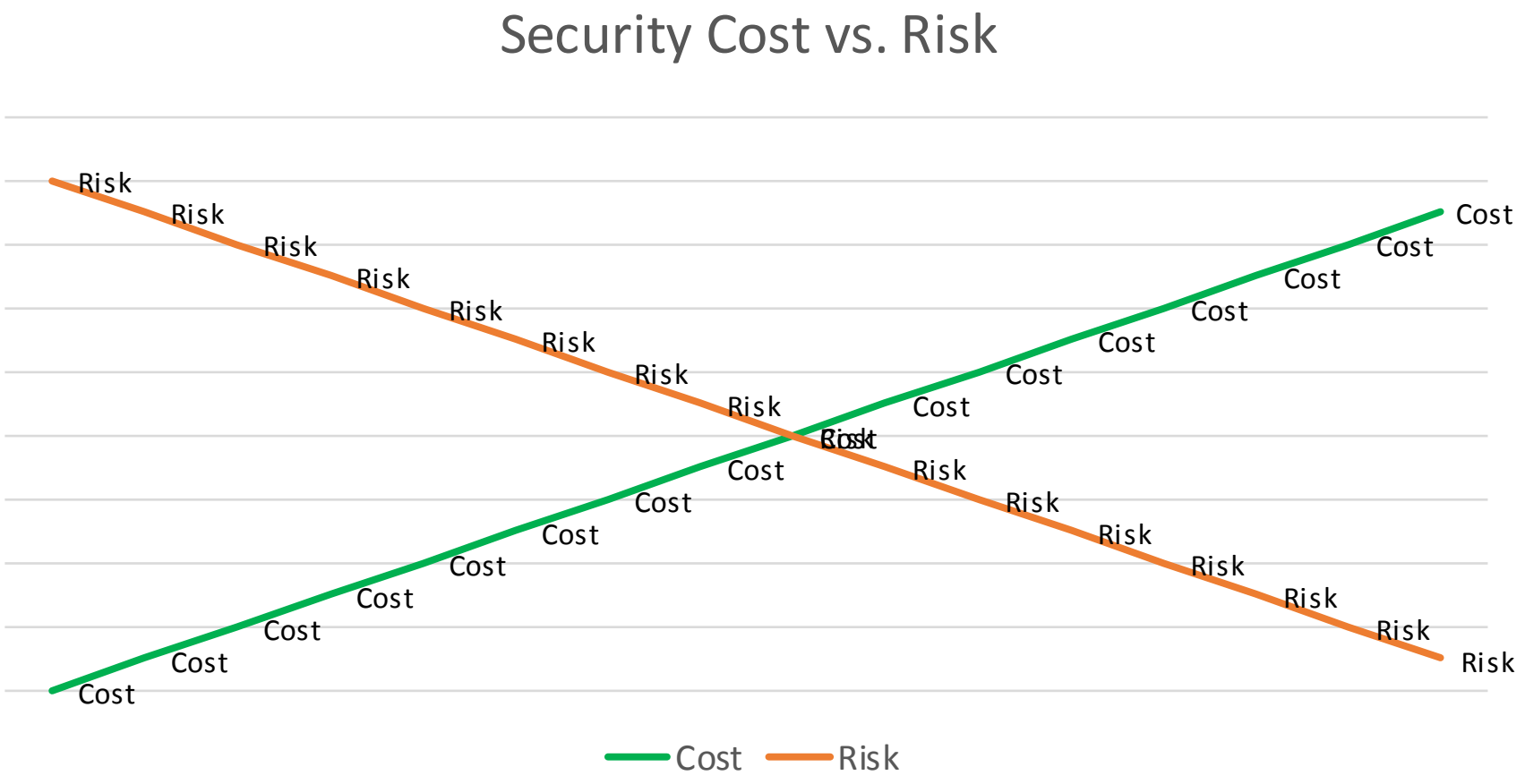
What Is Database Encryption?

- **Encryption of the data stored in a DBMS managed files**
  - –Database datasets
  - –DBMS system logs
  - –Image copy backup datasets
- **The encryption is performed as part of the normal DBMS processing**
- **“Clear text” data cannot be accessed outside of DBMS access methods**

Cryptography has Many Applications



Security Cost/Risk



What value does database encryption add?

- **Protects sensitive data on various storage media**
  - –Encrypted media is now protected by two layers of encryption
  - –Data is encrypted even when stored on non-encrypted media
- **Another layer of security**
- **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**
- **Prevent access to decrypted data outside the control of the DBMS**
  - –DFDSS
  - –FDR
  - –IDCAMS Repro
  - –Volume reassign

Encryption and “Data at Rest” Protection

- **Key requirement most “current” data protection initiatives**
  - Main requirement: protect “data at rest”- ensure access only business need-to-know, and through native security controls mechanisms (such as RACF)
- **Consider the following scenario:**
  - Database datasets controlled via RACF from direct access outside DBMS via dataset access rules
  - DBA or Storage Administrator has RACF authority to read database datasets in order to perform legitimate storage administration activities.
  - Administration privileges can be abused to read database datasets directly and access clear-text data outside DBMS/RACF protections.
  - Now consider above scenario, but with underlying database datasets encrypted

Disk Level Encryption

- DS8000 Disk Encryption
- DASD device encryption is “all or nothing”
- There is only a single key for the entire DS8000
- Disk encryption is only one layer of protection in a comprehensiv
  implementation

Disk vs DB Encryption

Disk Encryption

- **Protects at the DASD subsystem level**
- **All or nothing encryption**
- **Only data at rest is encrypted**
- **Single encryption key for everything**
- **No application overhead**
- **Exposures prevented**
  - –Disk removal
  - –Box removal

Database Encryption

- **Very flexible key granularity**
  - –Down to the field for DB2
  - –Segment level for IMS
- **Excellent separation of duties**
- **Data in-flight is protected**
- **Exposures prevented**
  - –Non-DBMS data access
  - –Unauthorized access to DBMS generated datasets (i.e. logs)

How Does Encryption Happen?

- **DASD device**
- **Application code**
- **Built in DBMS feature**
  - –DB2 built in encryption
- **Using DBMS supplied exit points**
  - –DB2
    - EDITPROC, FIELDPROC, UDF (user defined functions)
  - –IMS
    - Segment edit routine
- **z/OS Encryption facilities**

z/OS Encryption

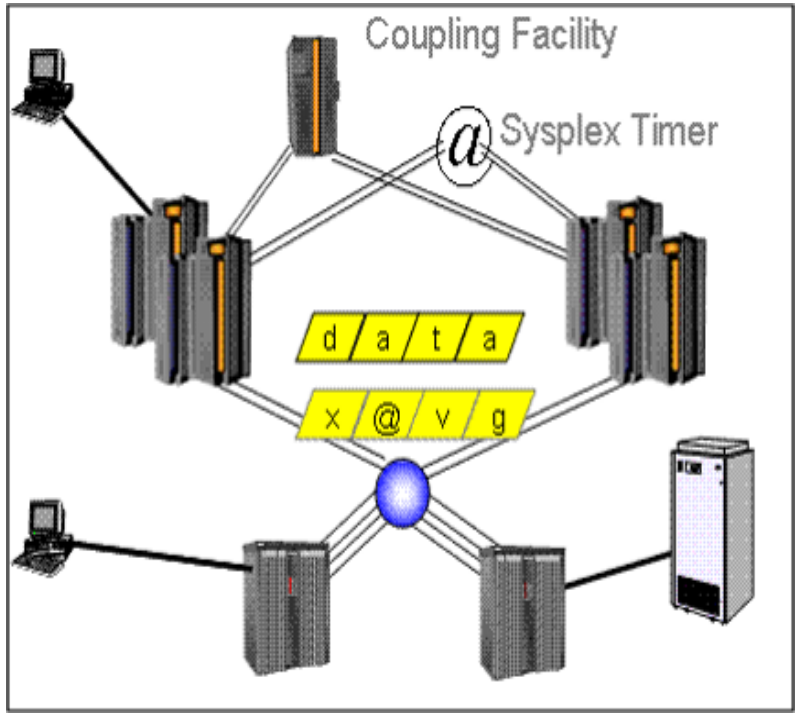
© 2016 IBM Corporation

What is Encryption?

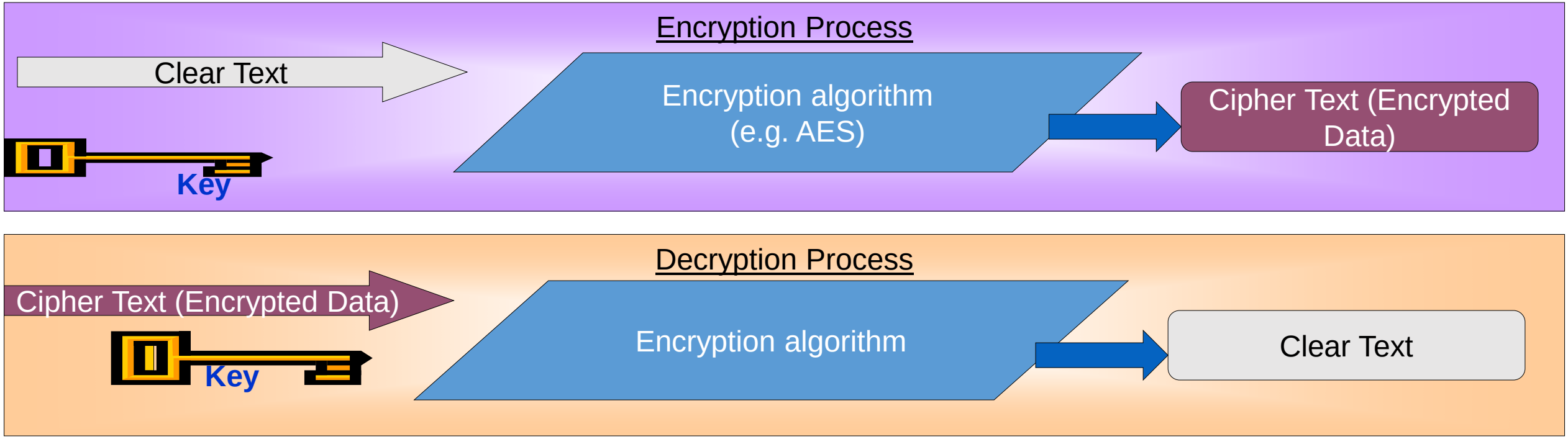
- **Encryption is a process where clear-text is converted using a well known ALGORITHM**
  - DES
  - TDES
  - AES

  along with a key
  - Clear
  - Secure

  to produce “CIPHER TEXT”



Symmetric Encryption Explained



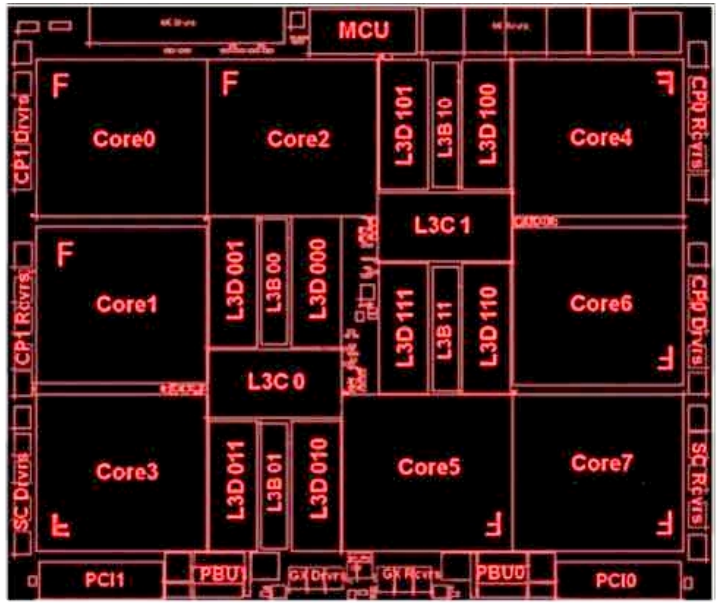
- **Data that is not encrypted is referred to as “clear text”**
- **Clear text is encrypted by processing with a “key” and an encryption algorithm**
  - Several standard algorithms exist, include DES, TDES and AES
- **Keys are bit streams that vary in length**
  - For example AES supports 128, 192 and 256 bit key lengths

Integrated Cryptographic Service Facility (ICSF)

- **Provides: z/OS integrated software support for data encryption**
  - –Operating System S/W API Interface to Cryptographic Hardware
  - –CEX2/3C hardware feature for z114, z10 and z196
  - –CEX4S hardware feature for z12BC and z12EC
  - –CEX5S hardware feature for z13 (2x faster over CEX4S)
- **Enhanced Key Management for key creation and distribution**
  - –Public and private keys, Secure and clear keys, Master keys
  - –Created keys are stored/accessed in the Cryptographic Key Data Set (CKDS) with unique key label
  - –CKDS itself is secured via Security Access Facility

Central Processor Assist for Cryptographic Function (CPACF)

- **CPACF)is available on every processor unit defined as a central processor (CP).**
- **Provides a set of symmetric cryptographic functions that can be used to enhance the encryption and decryption performance of clear-key operations for:**
  - Secure Sockets Layer (SSL) and Virtual Private Networks (VPN)
  - Applications not requiring a high level of security such as Federal Information Processing Standard (FIPS) 140-2 Security Level 4.
- **CPACF Co-processor redesigned from "ground up" and for performance improvements:**
  - Estimates do not include overhead for COP start/end and cache effects
  - Design estimates for large blocks of data
    - AES: 2x throughput vs. zEC12
    - TDES: 2x throughput vs. zEC12
    - SHA: 3.5x throughput vs. zEC12
- **Exploiters of the CPACF benefit from exploited by the throughput improvements of z13's CPACF such as:**
  - DB2/IMS encryption tool
  - DB2® built in encryption
  - z/OS Communication Server: IPsec/IKE/AT-TLS
  - z/OS System SSL
  - z/OS Network Authentication Service (Kerberos)
  - DFDSS Volume encryption
  - z/OS Java SDK
  - z/OS Encryption Facility
  - Linux on z Systems; kernel, openssl, openCryptoki, GSKIT



Crypto Express5S

- One PCIe adapter per feature
  - **Initial order – two features**
- Designed to be FIPS 140-2 Level 4
- Installed in the PCIe I/O drawer
- Up to 16 features per server
- Prerequisite: CPACF (#3863)

NEW

Three configuration options for the PCIe adapter

- Only one configuration option can be chosen at any given time
- Switching between configuration modes will erase all card secrets
  - Exception: Switching from CCA to accelerator or vice versa

Accelerator	
TKE	N/A
CPACF	NO
UDX	N/A
CDU	N/A

Clear Key RSA Operations

CCA Coprocessor	
TKE	OPTIONAL
CPACF	REQUIRED
UDX	YES
CDU	YES(SEG3)

Secure Key crypto Operations

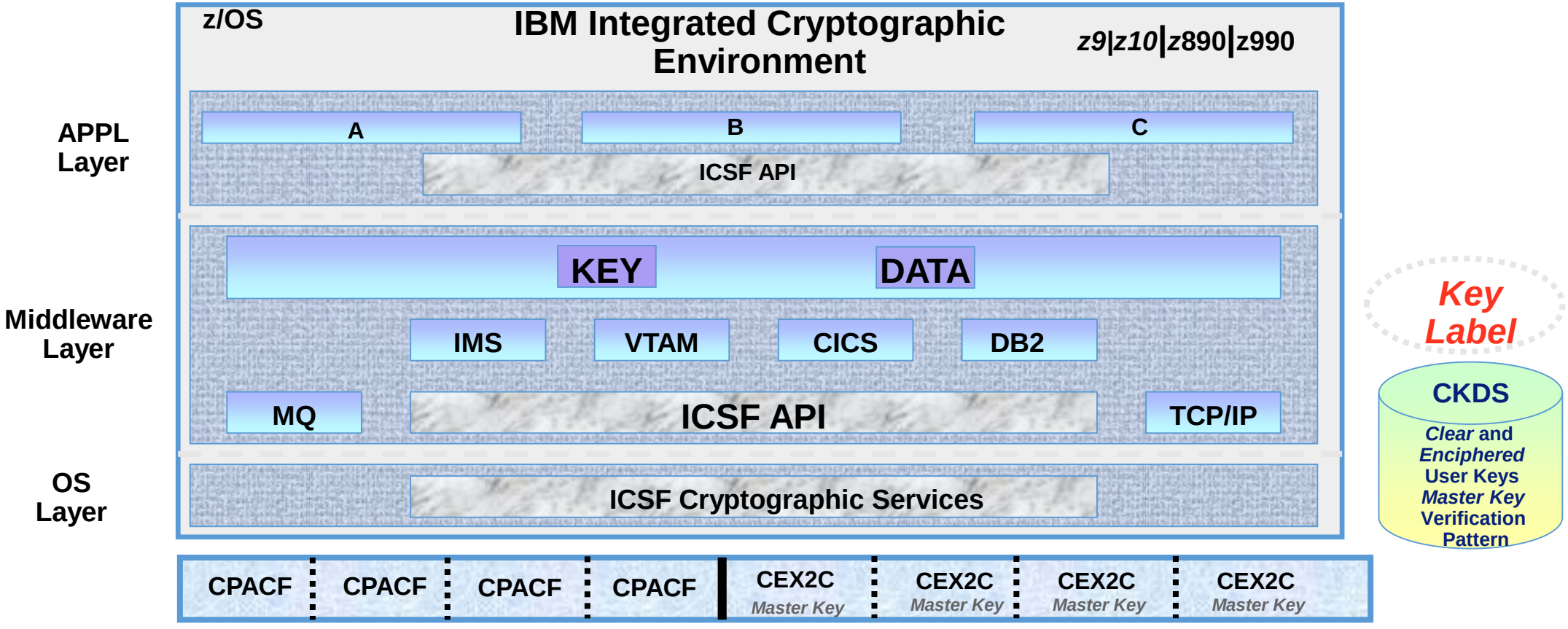
EP11 Coprocessor	
TKE	REQUIRED
CPACF	REQUIRED
UDX	NO
CDU	NO

Secure Key Crypto Operations

Business Value

- High speed advanced cryptography; intelligent encryption of sensitive data that executes off processor saving costs
- PIN transactions, EMV transactions for integrated circuit based credit cards(chip and pin), and general-purpose cryptographic applications using symmetric key, hashing, and public key algorithms, VISA format preserving encryption (FPE), and simplification of cryptographic key management.
- Designed to be FIPS 140-2 Level 4 certification to meet regulations and compliance for PCI standards

IBM Encryption - Flow



CP Assist for Cryptographic Functions

- Problem State Instructions
- Clear Keys Only
- DES/TDES Encryption
- AES (128 Bit)
- SHA-1 (256 on z9)

Crypto Express 2 Coprocessor

- ICSF Access Only (Key 0)
- Master Key Stored Within Crypto Express 2 Feature
- Secure Key DES/TDES Encryption
- SSL Accelerator
- Tamper Resistant

CKDS – Cryptographic Key Dataset

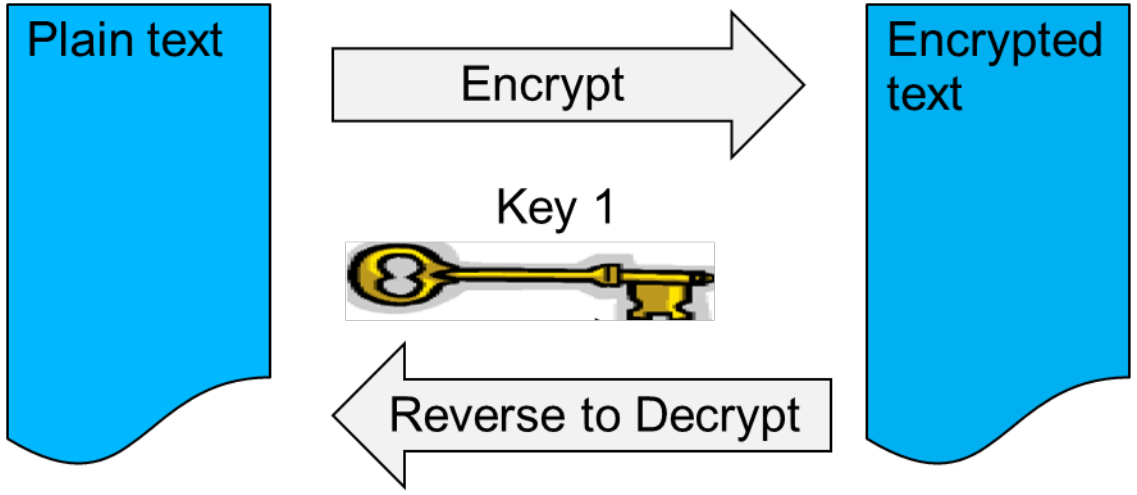
- **Key element of the IBM encryption solution on z/OS**
- **VSAM Key Sequenced Dataset**
- **Contents are ICSF generated data encrypted keys**
- **Accessed by ICSF API and Services**
  - –Key Label (known by application requestor) used to find key record in the CKDS
- **CKDS administration performed using ICSF services and ISPF interfaces.**
- **Use of specific individual keys can be controlled via RACF profiles and permissions**

Encryption Algorithms

© 2016 IBM Corporation

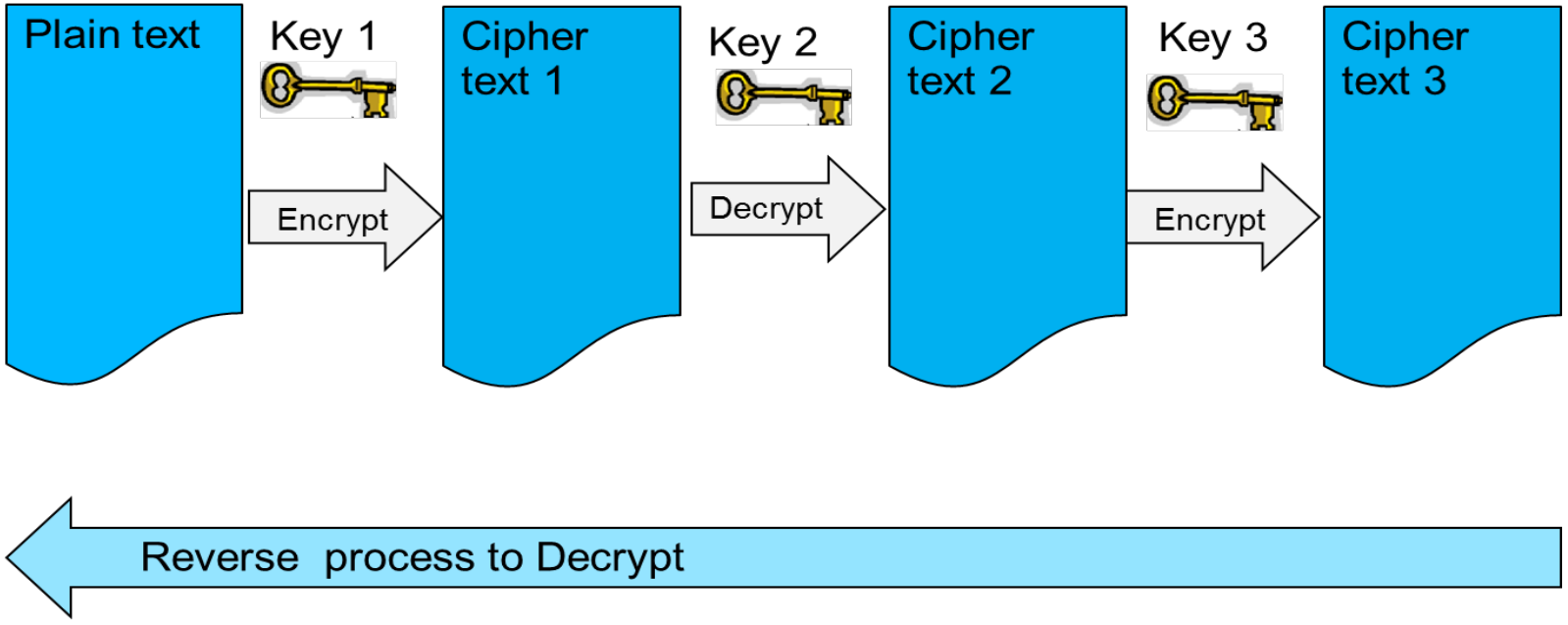
Encryption Algorithms – DES

- DES (Data Encryption Standard)
- 56-bit, viewed as weak and generally unacceptable today



Encryption Algorithms – TDES

- TDES (Triple Data Encryption Standard)
- 128-bit, universally accepted algorithm

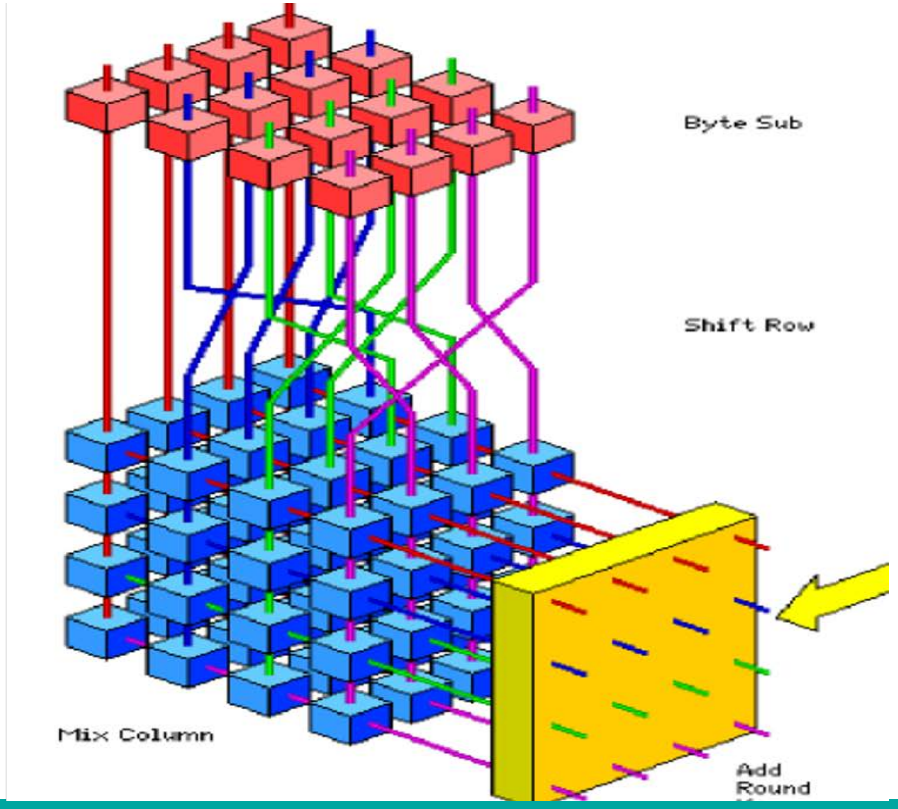


Note: same key can be used for each step for DES compatibility

Encryption Algorithms – AES

- AES (Advanced Encryption Standard)
- 128- , 192- or 256- bit, newest commercially used algorithm

Rijndael Algorithm
- Block Cipher (16-byte blocks)
- 128, 192, 256-bit Key Length
- Multiple Rounds
- Four Steps per Round
  - Byte Substitution
  - Shift Row
  - Mix Column
  - Add Round Key



Encryption Algorithms – Review

- **DES (Data Encryption Standard)**
  - –56-bit, viewed as weak and generally unacceptable today by the NIST
- **TDES (Triple Data Encryption Standard)**
  - –128-bit, universally accepted algorithm
- **AES (Advanced Encryption Standard)**
  - –128- or 256- bit, newest commercially used algorithm
- **What is acceptable?**
  - –DES is viewed as unacceptable
  - –TDES is viewed as acceptable and compliant with NIST (National Institute of Standards and Technology)
  - –AES 128 or 256 is also viewed as acceptable and strategic

Encryption Algorithms - Review

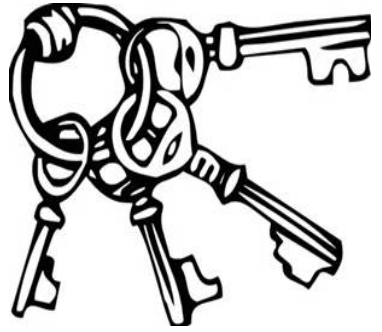
- **For more information:**
  - –TDES NIST Special Publication 800-67 V1 entitled "Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher" and can be found at:
    - http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.pdf
  - –TDES NIST FIPS Publication 197 entitled "Announcing the Advanced Encryption Standard (AES)" and can be found at:
    - http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

Keys, keys, which key for what?

© 2016 IBM Corporation

What are Keys? (From an ICSF Perspective)

- **Master Keys**
  - –Loaded into the “crypto cards” hardware, and stored NO WHERE else
  - –Pass Phrase Utility (Not recommended)
  - –ISPF Panels
  - –TKE workstation (optional hardware feature)
  - –Used to generate, encrypt, and store user keys into the CKDS (Cryptographic Key Data Set)



- **User Keys (Data Encrypting keys)**
  - –Generated via ICSF services
  - –Stored inside the CKDS
  - –Clear or Secure
  - –Used in conjunction with an encryption algorithm to convert user data to “cipher text”

Data Encrypting Keys 1

- **Clear key**
  - –Key material stored in the “clear” in the CKDS
    - Optionally, source for a protected key
  - –Encryption operations performed on General Purpose processors using CPACF
  - –Best overall performance
  - –Some increase for the DB2 SQL component
    - CPACF is supported on IFL and zIIP features.

Data Encrypting Keys 2

- **Secure key**
  - –Key material does not exist outside of the crypto card
    - Key encrypted inside the crypto card using the Master Key
    - Optionally, source for a protected key
  - –Encrypted key material stored in the CKDS
  - –Encryption/decryption operations performed inside the crypto card
  - –Most secure key type
  - –Highest overhead of the available key types

Data Encrypting Keys 3

- **Protected key**
  - –Is usually created as a secure key
    - Security definition allows a secure key to be used as a protected key
  - –Stored in the CKDS like a secure key
  - –When used
    - Decrypted inside the crypto card and wrapped with a wrapping key
    - Passed to ICSF and stored in a HAS only available to ICSF
  - –Performance tracks with clear key
  - –Clear key material can also be used as a source

Key Types Side-by-Side

Clear Key

- **Key material may be exposed in the storage of processor**
- **Key material may exist in the application address space**
- **Can be viewed in storage (via dump or on-line monitor)**
- **If correctly interpreted can expose data**
- **Sometimes acceptable for short-lived keys with other constraints**
- **Used in software-based cryptography**
- **APIs available via Integrated Cryptographic Support Facility (ICSF)**
- **Encrypt/decrypt operations performed by CPACF**

Secure Key

- Key material is never exposed beyond the bounds of a secure tamper resistant card
- Key is encrypted under the Master key and stored in the CKDS
- Crypto Express 2, 3, 4, 5
- APIs available via Integrated Cryptographic Support Facility (ICSF)
- Can be used from Java on z/OS platform
- Encrypt/decrypt operations performed on the Crypto Express card

▪Protected Key

- Key material may be exposed in the storage of processor
  - Only if using clear key material as the source
- Can be viewed in dump of storage
- Crypto Express 2, 3, 4, 5
  - Not required if using clear key material as a source
- APIs available via Integrated Cryptographic Support Facility (ICSF)
- Encrypt/decrypt operations performed by CPACF

Secure or Clear Key Performance

- Clear key elapsed time performance is MUCH superior than secure key
- Secure key (performed inside the “crypto card”) is generally viewed as more secure from a cryptographic perspective
- Clear key uses special instructions that run on the general purpose processors, so performance is measured in milliseconds or microseconds
- Secure key is probably NOT appropriate OLTP workloads
- You must make this encryption decision based on your security requirements and performance expectations

IBM Security Guardium Data Encryption for DB2 and IMS Databases

© 2016 IBM Corporation

IBM Security Guardium Data Encryption for DB2 z/OS & IMS Databases

- InfoSphere Guardium Data Encryption protects Sensitive and Private information minimizing the liability risks associated with Information Governance.

  - Complies with Security and Privacy regulations
  - Requires no changes to your applications
  - Conforms to the existing z/OS security model
  - Choice of encryption algorithms: Triple DES, DES, AES
  - Customize down to IMS segment level or DB2 column level
  - Straightforward implementation using RACF key labels
  - Leverage Storage Area Networks (SANs) safely while complying with privacy and security regulations

IBM Security Guardium Data Encryption for DB2 z/OS & IMS Databases

- **Encrypted Data at Rest**

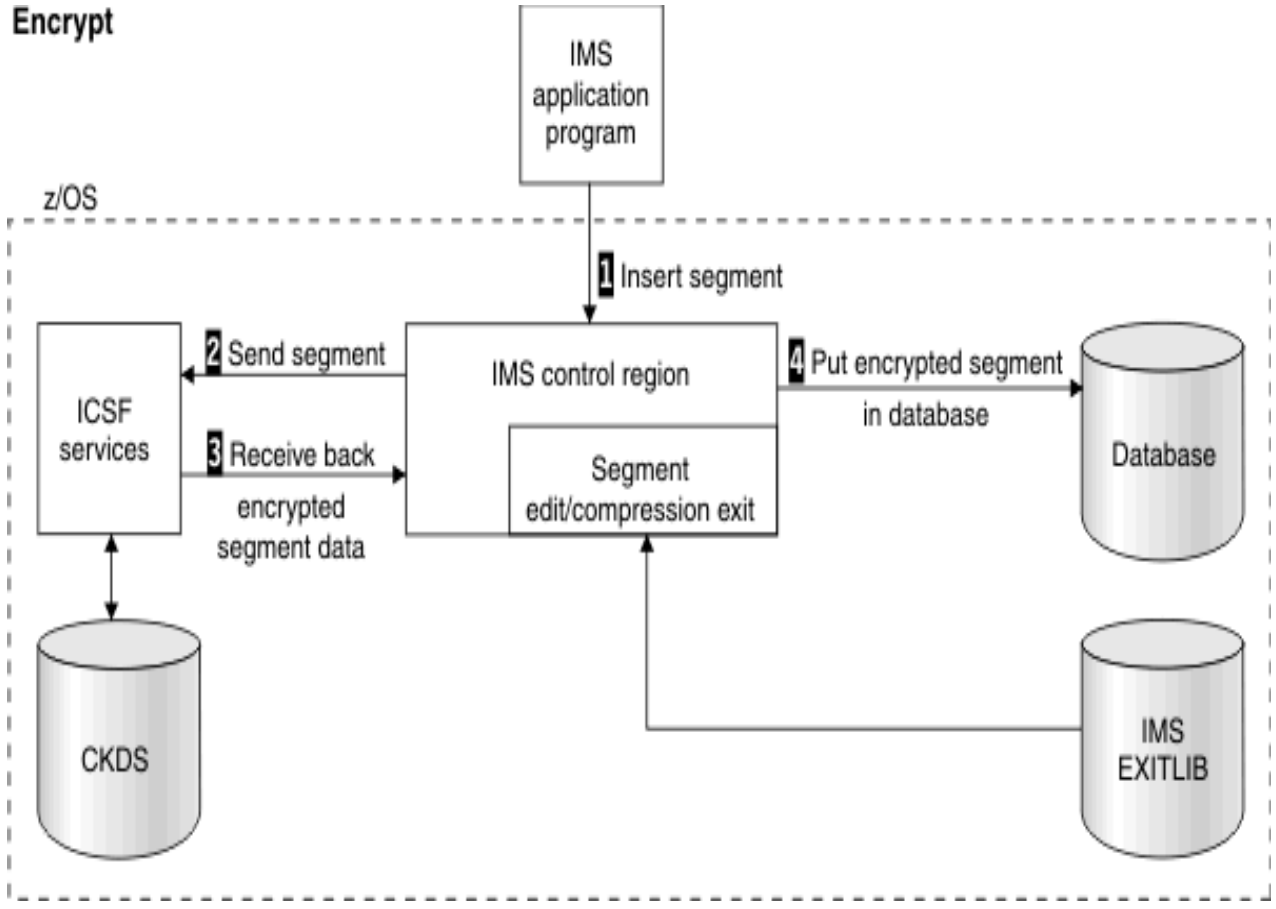
  – DFSUDMP0 (Image Copy)*
    - Does not access database records using IMS DLI
    - Data in image copy remains encrypted
    - Encryption exit is not necessary to be matched with image copy

  – Database Change Log Records (Type x'5050')
    - Contains exactly what is stored in the database on DASD

  – DFSURDB0 (Recovery)*
    - Does not use IMS DLI for recovery
    - Can recover using encrypted database records

- *** Also applies to IBM High Performance Tools**

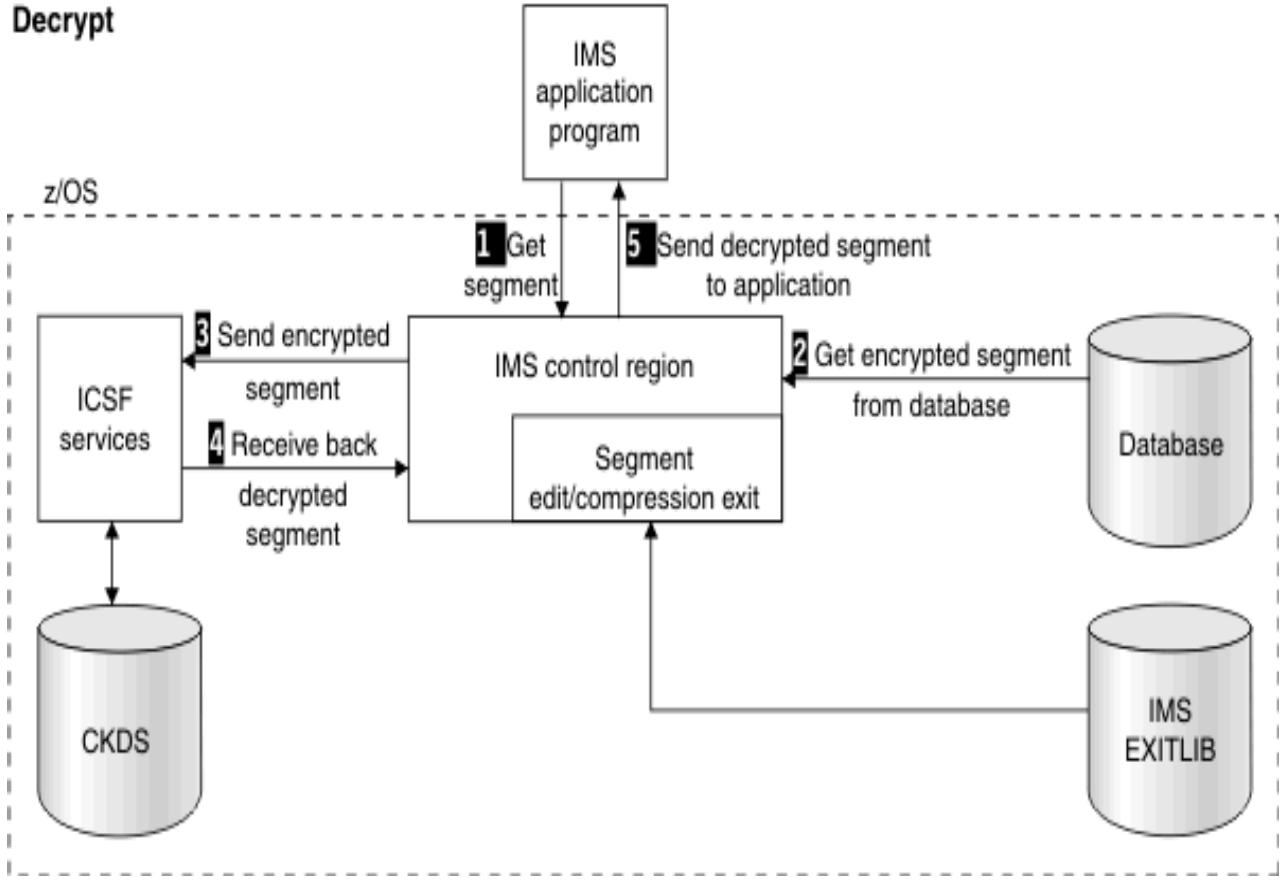
IMS Encryption Flow



Encryption

1. **IMS application program passes a segment REPL, ISRT, or LOAD request to the IMS control region. IMS uses the DBD to determine that a Segment Edit/Compression exit is required, so IMS loads the exit.**
2. **Exit invokes ICSF services, passing user-defined data encryption key label (provided by exit) and unencrypted segment.**
3. **When the segment has been successfully encrypted, the exit passes the segment back to IMS.**
4. **IMS then puts the encrypted segment into the database**

IMS Decryption Flow



Decryption

1. IMS application program passes segment GET request to IMS control region. IMS determines, from DBD, that a Segment Edit/Compression exit is required, so IMS loads the exit.
2. IMS retrieves encrypted segment from the database.
3. IMS then calls the exit and passes it the encrypted segment. The exit invokes ICSF services, which passes the user-defined data encryption key label (provided by exit) and the encrypted segment.
4. When the segment has been successfully decrypted, the exit passes the segment back to IMS.
5. IMS passes the decrypted segment back to the application.

Implementing IMS Database Encryption

© 2016 IBM Corporation

IMS Segment Edit Routines

- **There are three routines supplied with Guardium**
  - DECENA01 – IMS Clear Key Exit routine
  - DECENB01 – IMS CPACF Protected Key exit routine
  - DECENC01 – IMS Secure Key exit routine
- **These routines are found in the installed dataset hlq.SDECLMD0**

Creating the Routines

- To create an exit that encrypts and decrypts IMS data, the Tool can be implemented in one of two ways:

  1. Through JCL. The product provides sample jobs where the JCL can be modified to meet your needs for encrypted IMS databases.

  2. Using the ISPF interface. An ISPF dialog is available for you to create customized jobs for encrypting IMS database segments.

- Both processes allow:

  –A Standalone Encryption/Decryption routine
  –Encryption/Decryption in combination with database Compression.

Comparison of Supported IMS Encryption Methods

Function	DECENA01	DECENAA1	DECENB01	DECENC01
Segment based encryption (Key and or Data)	✓	✓	✓	✓
“Clear Key”	✓ **TDES/DES**	✓ **AES, TDES/DES**		
“Protected Key”			✓ **AES**	
“Secure Key”				✓ **AES, TDES/DES**

Implementing DB2 Database Encryption

© 2016 IBM Corporation

Data Encryption Comparison

Properties	Self Encrypting Devices	DB2 Builtin	EDITPROC	UDF	FIELDPROC
Satisfies PCI	Maybe for data at rest requirement	App Dependent	Yes , For data at rest and in flight to/from application.	Yes , For data at rest and in flight to/from application.	Yes . For data at rest and in flight to/from application.
Complexity	No Application Changes.	Complex Application Changes	No Application Changes	Only impacts Applications that need Encryption	No Application Changes
Implementation	Mount encryptible volume on appropriately configured drive.	ALTER and RELOAD	DROP and RELOAD	SQL UPDATE	DROP and reload unless ALTER ADD column
Clear Passwords	No	Yes	No	No	No
Access Control	No application security provided	App Managed	Table Level	RACF secured UDF	Column Level
Device Hardware	Yes.	No	No	No	No
Key Control	IBM Security Key Lifecycle Manager	Stored in the DB2 Catalog	ICSF or IBM Security Key Lifecycle Manager	ICSF or IBM Security Key Lifecycle Manager	ICSF or IBM Security Key Lifecycle Manager
Application Maintainability	Transparent	Transparent	Transparent	Requires SQL to invoke On-Demand Processing	Transparent
Performance	Best	App Dependent	Acceptable	Overhead for encrypted column	Acceptable
Index Encryption	Yes	Yes	No	Yes	Yes, but SQL Predicate Processing is Negated
Data Protection	Only protect on media	Clear Key	Unload Data in clear	Always Protected	Unloaded Data is clear
Data Type Independence	Yes	Yes	Most data types	Yes	Only CHAR/VARCHAR < 255 bytes Columns

How do the DB2 Built-In Functions work?

- **Under application control – you encrypt the fields that need to be secure**
  - ‘Password for Encryption’ is hashed to generate a unique key
  - Hint can be used as a prompt for remembering the key
  - Encrypted field must be defined as VARCHAR (since it will contain binary data once its encrypted)
  - The encrypted field will be longer (next multiple of 8 bytes + 24 bytes of MetaData + 32 bytes for optional hint field)
  - TDES Only!

Encrypt (StringDataToEncrypt, PasswordOrPhrase, PasswordHint)
Decrypt_Char(EncryptedData, PasswordOrPhrase

DB2 Built-In Functions Example

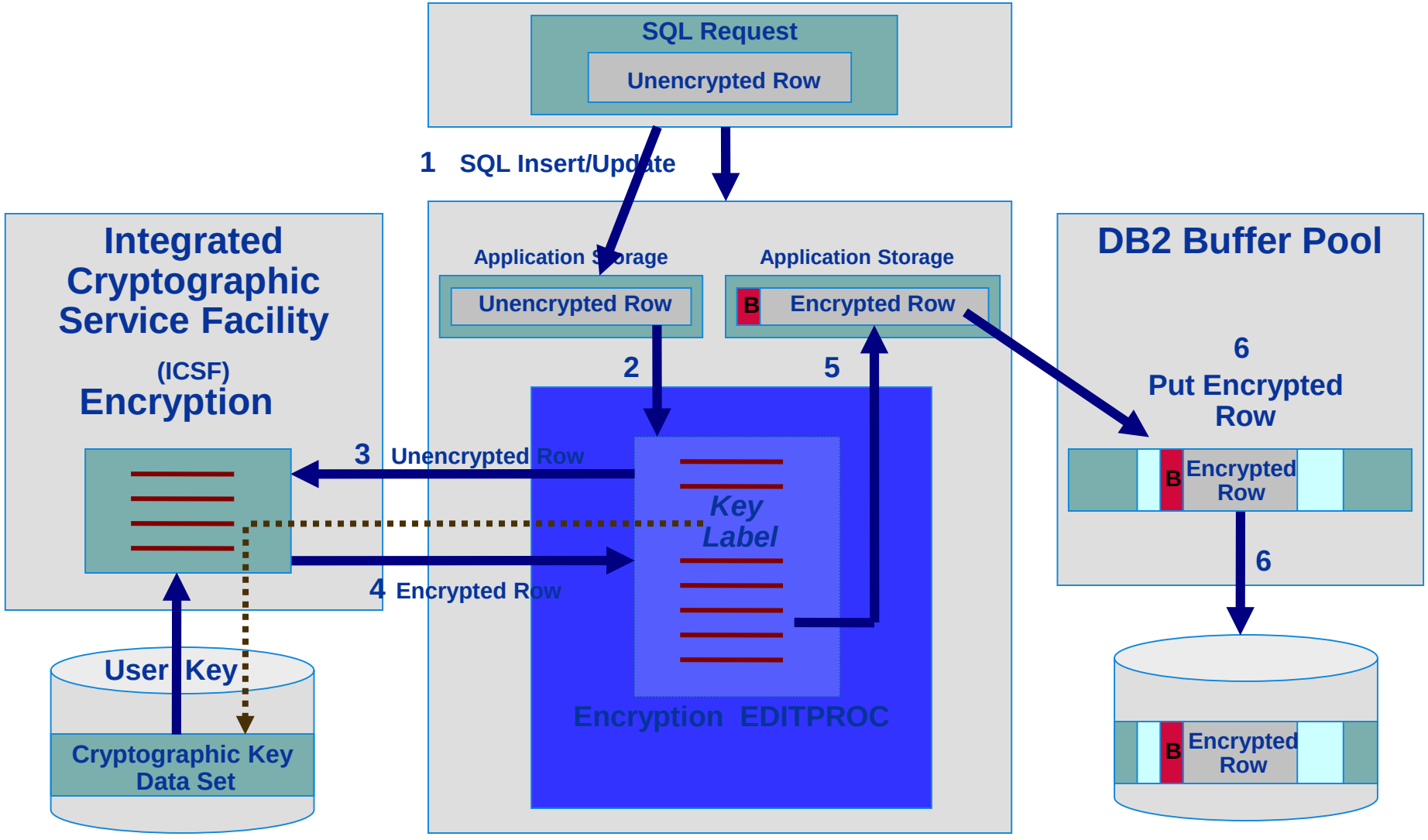
```
CREATE TABLE EMPL
(EMPNO VARCHAR(64) FOR BIT DATA, EMPNAME CHAR(20),
CITY CHAR(20), SALARY DECIMAL(9,2))
IN DSNDB04.RAMATEST ;

COMMIT;

SET ENCRYPTION PASSWORD = 'PEEKAY' WITH HINT 'ROTTIE';

INSERT INTO EMPL(EMPNO, EMPNAME, SALARY)
VALUES (ENCRYPT('123456'),'PAOLO BRUNI',20000.00) ;

INSERT INTO EMPL(EMPNO, EMPNAME, SALARY) VALUES
(ENCRYPT('123457'),'ERNIE MANCILL',20000.00) ;
```

From Redbook SG24-7959, Security Functions of IBM DB2 10 for z/OS

DB2 Row Encryption

- **EDITPROC - for every row**
  - –Encrypted row same length as clear row
  - –No application changes required
  - –One key per table specified in the EDITPROC
  - –Indexes are not encrypted

DB2 Data Encryption Flow – Insert / Update



DB2 Column Encryption

- **FIELDPROC – encrypts at the column level**
  - –No application changes required
  - –Indexes can be encrypted
  - –One key, label specified in the FIELDPROC
  - –Columns must be < 254 bytes; Column names must be < 18 chars in length
- **UDF – User Defined Functions**
  - –No application changes required; Minimally disruptive, columns encrypted in place
  - –Indexes can be encrypted
  - –One key, label specified in the UDF
  - –All data types supported by UDFs can be encrypted
  - –VIEW/TRIGGER – provides access control to the cleartext

Comparison of Supported DB2 Encryption Methods

Function	EDITPROC	FIELDPROC	UDF
Row based encryption	✓		
Column based encryption		✓	✓
Application impact during implementation (Drop/CREATE) [1]	✓	✓ **maybe**	
SQL based implementation			✓
“Clear Key”	✓ **AES, TDES/DES**		
“Protected Key”	✓ **AES**	✓ **AES**	✓ **AES**
“Secure Key”	✓ **AES, TDES/DES**		

1. New encrypted columns can be added using FIELDPROC encryption by ALTER ADD COMUNN SQL. Encrypting an existing column requires DROP/CREATE.

DB2 encryption “to do’s

- Request a new key from the ICSF Administrator
- ICSF Administrator generates Data Encryption Key using ICSF
- Obtain Key Label from ICSF Administrator
- Code the DB2 EDITPROC
- Link-edit the EDITPROC into the appropriate DB2 library(SDSNEXIT orequivalent)
- Back - Up and Unload Databases
- DROP and recreate the DB2 objects
- LOAD the DB2 objects. Objects will be encrypted during the LOAD process
- Backup the affected DB2 objects
- Validate your Output

Note: This list is general only and may note be everything needed in your installation

IBM Security Guardium Data Encryption for DB2 z/OS & IMS Databases

- **A Single tool for both DB2 and IMS**
- **Performs encryption and decryption through the use of exit routines.**
- **Leverages the System z®, zSeries®, and S/390® Crypto Hardware to encrypt data**
- **Protects sensitive data that can reside on various storage media**
  - –DB2 and IMS databases
  - –Image copy datasets
  - –DASD volume backups

Summary

- Database encryption adds another layer of protection
- Limits the availability of data “in the clear”
- Protects data in use
- Leverages z/OS hardware and microcode enhancements

Questions?