IBM z Systems

Pervasive Encryption on z/OS®

RACF Users Group of New England

June, 2017

Mark Nelson, CISSP®, CSSLP®
z/OS® Security Server Design and Development
IBM®
markan@us.ibm.com

IBM has been providing Security & Encryption Solutions for over 40 years

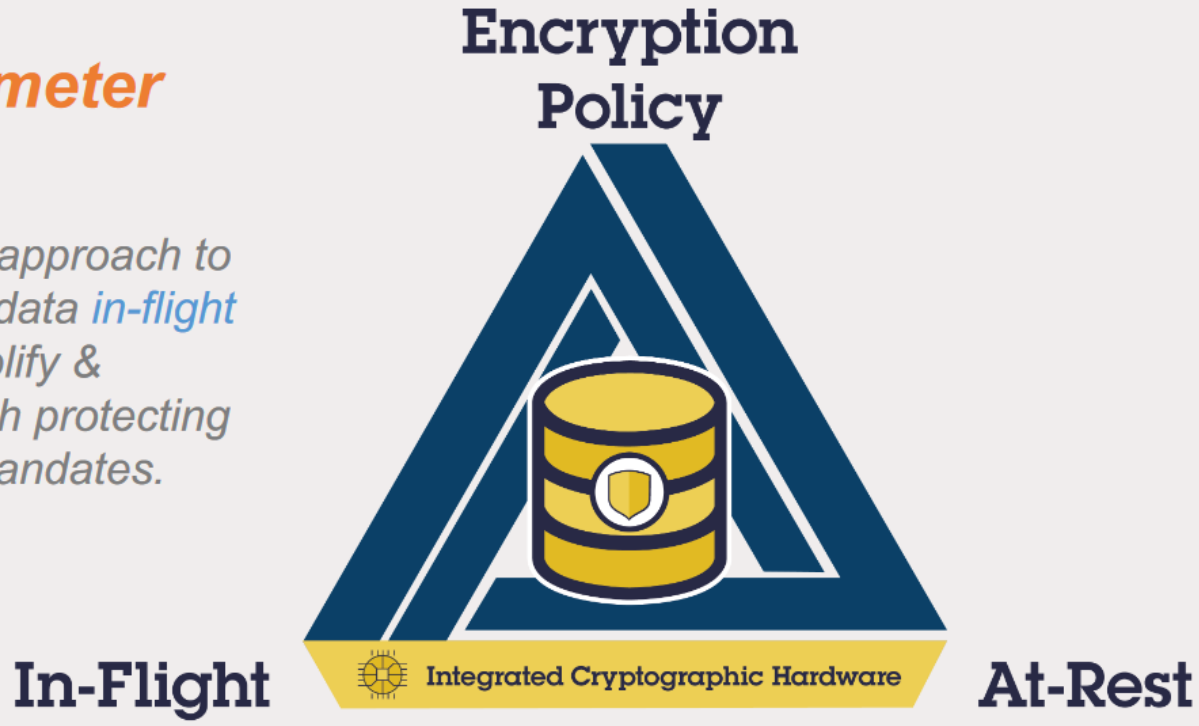
- IBM submits the Lucifer cipher to become the Data Encryption Standard (DES): 1974 - 1976
- Hardware Cryptography using IBM 3845 Channel Attached DES/TDES: 1977 - 1979
- IBM 4753 Channel Attached CCA Unit with smart cards and signature dynamics pen: 1989
- Key management built into operating system (ICSF): 1991
- Distributed Key Management System (DKMS) (1990's)
- Trusted Key Entry (TKE) Workstation: ~1997
- Encryption Facility for z/OS: 2005
- TS1120 Encrypting Tape Drive: 2006
- LTO4 Encrypting Tape Drive: 2007
- Tivoli Encryption Key Lifecycle Manager: 2009
- Self-Encrypting Disk Drives, DS8000: 2009
- System z10 CPACF Protected Key Support: 2009
- Crypto Express3 Crypto Coprocessor: 2009
- z Systems z196 with additional CPACF encryption modes: 2010
- Crypto Express4S Crypto Coprocessor: 2012
- z Systems zEC12 with Public Key Cryptography Standards – PKCS#11: 2012
- Crypto Express5S Crypto Coprocessor: 2015
- z Systems z13 with Visa Format Preserving Encryption: 2015

IBM z Systems Pervasive Encryption

A Data Centric Approach to Information Security

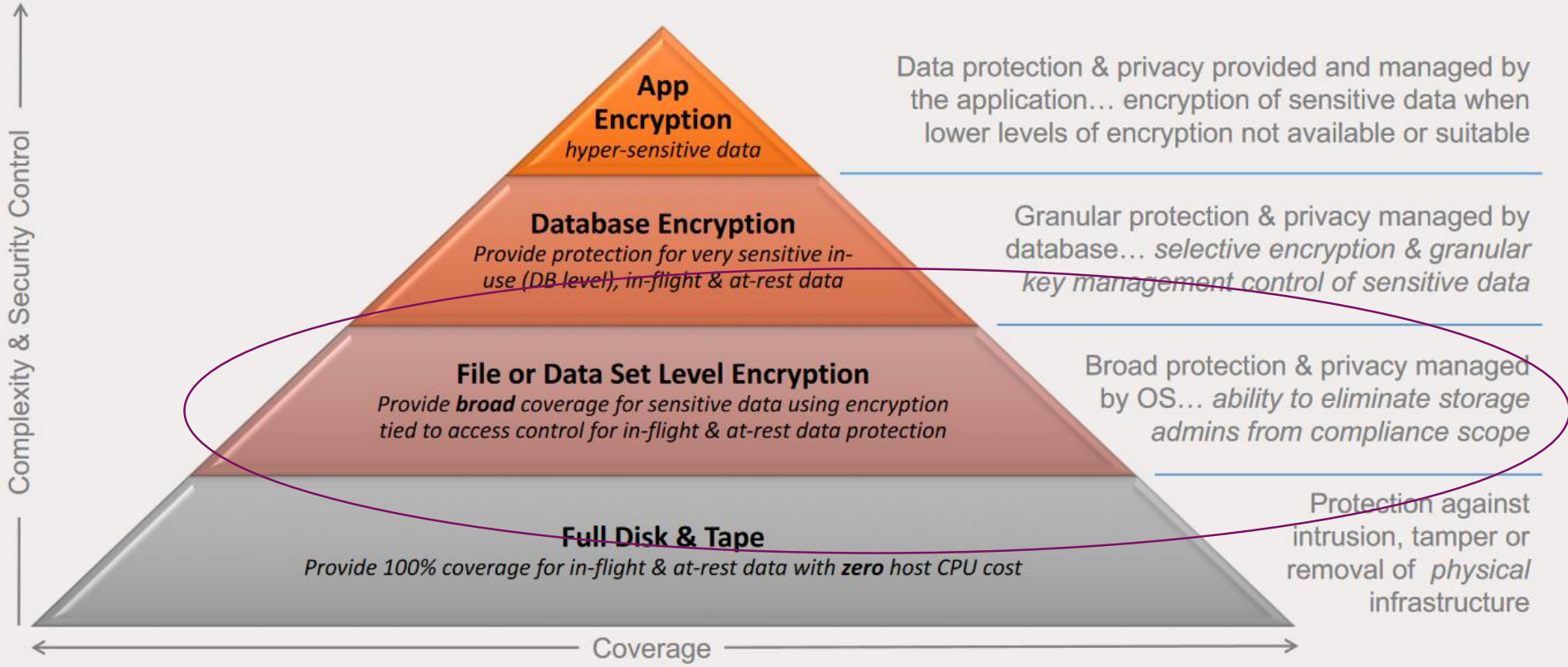
Data is the new perimeter

A transparent and consumable approach to enable extensive encryption of data in-flight and at-rest to substantially simplify & reduce the costs associated with protecting data & achieving compliance mandates.



Multiple layers of encryption for data at rest

Robust data protection



IBM US Software Announcement 216-392, 4 October 2016

- **IBM plans to deliver application transparent, policy-controlled dataset encryption in IBM z/OS. IBM DB2 for z/OS and IBM Information Management System (IMS) intend to exploit z/OS dataset encryption**

IBM Statement of Direction:

IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract.

The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

IBM US Software Announcement 217-085, 21 February, 2017

- **z/OS 2.3 will be designed to provide policy-enabled data protection for z/OS data sets, zFS file systems, and coupling facility structures, giving users the ability to encrypt data to strengthen compliance and audit responsiveness and provide simplified security processes and protection for mission-critical data. These planned enhancements include:**

  – DFSMS provides new enhancements that are designed to give users the ability to encrypt their data sets, using either SAF or SMS policies, without changing their application programs. DFSMS intends to make use of the Central Processor Assist for Cryptographic Functions (**CPACF**) to encrypt and decrypt extended format (version 2 only) sequential BSAM and QSAM data sets and all types of extended format VSAM data sets as written to and read from disk. In addition, data set level encryption is planned to allow the data to remain encrypted during administrative functions such as backup/restore, migration/recall, and replication.

  – zFS plans to make use of the DFSMS data set encryption to support the encryption of individual files (file content), access control lists, security information, and symbolic link contents. The use of zFS encryption can be paired with compression to offset the overhead of encryption.

Implementation Details: What's Supported and What's Not Supported

- **Encrypted data is transparent to the application when the application uses standard access method APIs*. No application changes are needed.***
- **Supported access methods**
  - BSAM/QSAM sequential data sets, extended format only (Version 2)
  - VSAM and VSAM/RLS (KSDS, ESDS, RRDS, VRRDS, LDS), extended format only
- **New option to allow access to data in the encrypted form**
- **Restrictions**
  - System data sets (such as catalogs, HSM data sets, RACF data sets) must not be encrypted unless otherwise specified
  - Encrypted data sets are supported only on 3390 device types
  - Sequential (non-compressed) data sets with a BLKSIZE of less than 16 bytes cannot be encrypted as they cannot be extended format
  - Data sets used during IPL must not be encrypted
  - Data sets must be SMS-managed
  - Data sets used before ICSF is started must not be encrypted

Implementation Details: Hardware and Software Levels

- **All systems sharing the data must be at a minimum level of z/OS V2.1 or higher**
  - Coexistence only on V2.1 (with APARs)
  - Supported on V2.2 (with APARs)

- **Requires:**
  - Feature 3863, CP Assist for Cryptographic Functions (CPACF)
  - The minimum processor hardware is z196 or higher with CEX3 or later

Implementation Details: Secure/Protected/Clear Keys

- **One of the basic strengths of the z System architecture is the support of different levels of protections for keys**
  - **Secure key:** The key material is exposed only when in use in the hardware security module (HSM), which for z Systems is the Crypto Express card
    - Tamper evident/resistant
    - Internal high performance for cryptographic operations
    - **Advantage:** Highly secure as the key is *never* in the clear outside of the HSM
    - **Disadvantage:** Data must be marshalled to and from the card
  - **Clear key:** The key, while it can be protected by normal z Architecture means (storage key, address space isolation, etc.), is located in main memory for at least the time of its use, such as by the CPACF.

Implementation Details: Secure/Protected/Clear Keys…

- **DFSMS is designed to use protected keys which allow the use of a key that never appears in the clear in storage and is not available to the operating system (like z/OS) or any application, but which can still be used by CPACF**
  - Protected keys usually (and should!) start as secret keys, but are marked as eligible for use as protected keys with **SYMCPACFWRAP(YES)** in the ICSF segment for the CSFKEYS profile which defines the key to RACF
  - **SYMCPACFRET(YES)** allows the protected key to be returned to the caller
  - **Both SYMCPACFWRAP(YES) and SYMCPACFRET(YES) are needed for encrypting data sets**
  - **Advantage:** Substantially higher performance than secure key operations as the data does not have to be marshalled/demarshalled to/from the crypto card

Implementation Details: Separation of Duties

- **Data owners who must access content will need authority to the data set as well as authority to the encryption key label**
- **System/Storage administrators who only manage the data sets need authority to the data set but not access to the encryption key label, thus protecting access to the content**
  - Prevent administrators from accessing the content
- **Different keys can be used to protect different data sets – ideal for multiple tenants or data set specific policies.**
- **Many utilities can process data preserving encrypted form**
  - COPY, DUMP and RESTORE
  - Migrate/Recall, Backup/Recover, Dump/Data Set Restore
  - PPRC, XRC, FlashCopy®, Concurrent Copy, etc. or later

Implementation Details: RACF Setup – Access Checks

- **With this support, DFSMSdfp calls ICSF to get key information. Users creating, updating, or accessing encrypted data will need access to certain ICSF services**
  - For example, CSNBKRR2 (CKDS Key Record Read2 service) in the CSFSERV class
- **The key to be used for the encryption is specified by its ICSF key-label. The user must be authorized to this key-label in the CSFKEYS class**
  - You can permit users and groups to the CSFKEYS resources in the “traditional” manner:

```
PERMIT key-label CLASS(CSFKEYS) ID(…) ACCESS(READ)
```

  - Alternatively, you can allow everyone to use the key, but only when using it for data set encryption/decryption using the CRITERIA specification on PERMIT:

```
PERMIT key-label CLASS(CSFKEYS) ID(*) ACCESS(READ)
WHEN(CRITERIA(SMS(DSENCRYPTION)))
```

- **Users who want to create encrypted data sets with the key-label coming from anywhere other than the DFP segment must have READ authority to the resource STGADMIN.SMS.ALLOW.DATASET.ENCRYPT in the FACILITY class.**

Implementation Details: RACF Setup – Defining the Key Label to DFP

- **A sequential or VSAM data set would be defined as 'encrypted' when a key label is supplied on allocation of a new sequential or VSAM data set. The key label can be specified (in order of precedence):**
  - RACF data set profile

```
ALTDSD 'PROJECT.DATA.*' UACC(NONE) DFP(DATAKEY(key-label))
```

  - JCL, dynamic allocation, TSO allocate, IDCAMS

```
//DD1    DD DSN=A.B.C,DISP=(NEW,CATLG),DATACLA=DSN1DATA,
//       MGMTCLASS=DSN1MGMT,STORCLAS=DSN1STOR,
//       DSKEYLBL='LABEL.FOR.DSN1'
```

  - SMS Data Class
- **The derived key label is stored in the catalog when the data set is allocated**

z/OS Data Set encryption – High Level Steps

1

Generate an encryption key and key label, store it in the CKDS .

2

Setup RACF for use of key label

Allow secure key to be used as protected key via ICSF segment
- SYMCPACFWRAP
- SYMCPACFRET

– AND –

Grant access to key label

3

Associate the key label with the desired data set(s).

In RACF, alter DFP segment in data set profile - DATAKEY()

– OR –

In DFSMS, assign to data class

In RACF, permit access to STGADMIN.SMS.ALLOW.DATASET.ENCRYPT resource in FACILITY class

4

Migrate to encrypted data

DB2: Online Reorg ★ Non-disruptive

IMS HA Database: Online Reorg ★ Non-disruptive

zFS Container: zfsadmin encrypt ★ Non-disruptive

VSAM or Seq data set:
1. Stop application
2. Copy data
3. Restart application

Implementation Details: Important Planning Considerations

- **Because data set encryption has both hardware and software requirements, consider sharing systems, backup systems, and disaster recovery systems when you are planning to use it**
  - Be sure that all the systems that must read encrypted data are capable of decrypting it
- **Be sure that your ICSF data sets are a part of your backup and disaster recovery plans**

Implementation Details: Other Things to Note

- When data set level compression is requested, the access methods handle the compression before the encryption
- Only AES-256 keys are supported
- LISTVTOC, LISTCAT, CSI (catalog search interface) all return encryption status of a data set
- SMF records 14/15 (sequential data sets) and SMF record 62 (VSAM data sets) has the encryption status of the data set as does DCOLLECT

Questions?