Introduction to IBM Z Cryptography
And the Ecosystem around IBM Z Cryptography

z14 / CEX6S
Table of Contents

- IBM Z Crypto History
- IBM Z Crypto Hardware
  - CP Assist for Cryptographic Function (CPACF)
  - Crypto Express 6S (CEX6S)
  - IBM Trusted Key Entry (TKE) Workstation
  - User Defined Extensions (UDX)
- IBM Z Crypto Hardware Virtualization
  - z/VM Virtualization of Crypto Hardware
- IBM Z Crypto Software
  - Linux on Z (and LinuxONE)
  - z/OS
  - z/VM
  - z/VSE
- IBM Z Crypto Solutions
  - Enterprise Key Management Foundation (EKMF)
  - Security Key Lifecycle Manager (SKLM)
  - Advanced Crypto Service Provider (ACSP)
  - Encryption Facility (EF)
  - Guardium Data Encryption (GDE) for DB2 / IMS
  - zSecure Suite
- IBM Z Pervasive Encryption
  - z/OS Data Set Encryption
  - Disk Encryption
  - Coupling Facility (CF) Encryption
  - z/OS and Linux on z Network Security
  - Linux on z Volume Encryption
  - Secure Service Container
  - z/VM Encrypted Paging
  - z/VM Network Security
  - z/TPF Transparent Database Encryption
IBM Z Crypto History
- Cryptographic Coprocessor Facility – Supports “Secure key” cryptographic processing
- PCICC Feature – Supports “Secure key” cryptographic processing
- PCICA Feature – Supports “Clear key” SSL acceleration
- PCIXCC Feature – Supports “Secure key” cryptographic processing
- CP Assist for Cryptographic Functions allows limited “Clear key” crypto functions from any CP/IFL
  - NOT equivalent to CCF on older machines in function or Crypto Express2 capability
- Crypto Express2 – Combines function and performance of PCICA and PCICC

Hardware preceding CCF includes:
- IBM 3845 Channel Attached DES (1977)
- IBM 3848 Channel-Attached TDES (1979)
- IBM 4753 Channel-Attached CCA processor (1989)
CP Assist for Cryptographic Function allows limited “Clear key” crypto functions from any CP/IFL
- NOT equivalent to CCF on older machines in function or Crypto Express2 capability
- Crypto Express3 – PCIe Interface, additional processing capacity with improved RAS
- Crypto Express4S - Enterprise PKCS #11
- Crypto Express5S - ECC HW acceleration and more RSA Engines
- Crypto Express6S – Enhanced Miniboot secure boot
Over 40 Years of IBM Z Security & Encryption Solutions…

A History of Enterprise Security

- IBM submits the Lucifer cipher to become the Data Encryption Standard (DES): 1974 - 1976
- RACF: controls access to resources and applications: 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
- Intrusion Detection Services (IDS): 2001
- z/OS PKI Services: create digital certificates & act as Certificate Authority (CA) – 2002
- Multilevel Security (MLS): 2004
- 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 Enterprise PKCS#11: 2012
- Crypto Express5S Crypto Coprocessor: 2015
- z Systems z13 with Visa Format Preserving Encryption: 2015
- Multi-Factor Authentication for z/OS: 2016
- Secure Service Container: 2016
- IBM z14 with Pervasive Encryption: 2017
- Crypto Express6S Crypto Coprocessor: 2017
IBM Z Crypto Hardware
How is crypto supported in IBM Z hardware?

- **PU SCM**: Each PU is capable of CPACF function.
- **CPC Drawer**: With 64 bytes of input using 256-bit AES-CBC encryption, CPACF was measured to perform ~327,891 operations per second.
- **Crypto Express6S**: With 64 bytes of input using 256-bit AES-CBC encryption, a Crypto Express adapter was measured to perform ~10,569 operations per second.
- **TKE**: Required for management of Crypto Express6S in EP11 or PCI-HSM mode.
- **Trusted Key Entry (TKE)**, **Smart Cards**, and **Smart Card Readers**.

© 2017 IBM Corporation
What is CPACF?

*CP Assist for Cryptographic Function (CPACF)*

IBM z hardware cryptographic function is available on every Processor Unit defined as a CP, IFL, zAAP and zIIP.
- Supported by z/OS, z/VM, z/VSE, z/TPF and Linux on z Systems
- Must be explicitly enabled using a no-charge enablement feature #3863

Provides a set of **symmetric cryptographic functions** and **hashing functions** for:
- Data privacy and confidentiality (DES, TDES, AES)
- Data integrity
  - MD5, SHA-1
  - SHA-2 (SHA-224, SHA-256, SHA-384, SHA-512)
  - SHA-3 (SHA3-224, SHA3-256, SHA3-384, SHA3-512)
  - SHAKE (SHAKE-128, SHAKE-256)
- Random Number generation
  - PRNG
  - DRNG
  - TRNG
- Message Authentication (DES, TDES)

Enhances the encryption/decryption performance of clear and protected key operations such as:
- SSL
- VPN
- DB2
- IMS
- Data sets and files
- Coupling Facility
AES-GCM and AES-XTS encryption modes have encryption rates four to six times faster than z13

- **Galois Counter Mode (GCM)** is a stream cipher used primarily in SSL/TLS workloads.
  - Exploited by z/OS Communications Server and System SSL
  - **Note:** It may be necessary to configure cipher suites to prefer GCM mode.

- **XEX-based tweaked-codebook mode with ciphertext stealing (XTS)** is a block cipher mode typically used for disk encryption.
  - Exploited by DFSMS z/OS data set encryption

- **Cipher Block Chaining (CBC)** is a block cipher used in several crypto applications. AES-CBC decryption on z14 offers a 4x increase in throughput for decryption operations.

Exploiters of the CPACF benefit from the throughput improvements of z14’s CPACF such as:

- DFSMS z/OS data set encryption
- Coupling Facility (CF) encryption **[New]**
- 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, Java 8 JCE
What is the benefit of Crypto Express adapters?

- Provides state-of-the art tamper sensing and responding, programmable hardware to protect cryptographic keys, sensitive cryptographic processing and sensitive custom applications
  - Unauthorized removal and/or tampering of the adapter zero-izes its content

- Suited to applications requiring high-speed security-sensitive cryptographic operations for data encryption and digital signing, and secure management and use of cryptographic keys
  - Functions targeted to Banking/Finance and Public sector

- Supports multiple logically-separate cryptographic domains for use by different LPARS.

- Provides both symmetric and asymmetric cryptographic functions.

- Supported by z/OS, z/VM, z/VSE, z/TPF and Linux on z Systems. **Note:** Crypto function exploitation may vary.
Enhanced Miniboot for Crypto Express6S

Miniboot is a bootstrap Operating System for IBM Crypto Express adapters

- Configures critical hardware systems and performs Power-On Self-Tests to verify card operations
- Follows a well-defined security model
- Acts as gatekeeper for updated firmware (segment1, 2, or 3)
- Provides status and authentication information
- Describes state information for certification (FIPS, …)

Enhanced Miniboot is now available for the IBM Crypto Express6S adapter

- Enhances the security algorithms
- Improves position for security certification
- Implements industry standard cryptographic data structures (keys, certificates, signatures)
- Moves some function from firmware to hardware (FPGA)
  - Improves performance
  - Makes the design more robust and easier to adapt (processor architecture changes)
What are cryptographic domains?

IBM Z uses the concept of a cryptographic domains to virtualize the physical coprocessor and enable sharing across multiple System z logical partitions (LPARs)

- A coprocessor can be shared by multiple LPARs and different types of operating systems
- Secure Keys generated and wrapped with MK of one domain are not usable by another domain using different MK
- Cryptographic domain assignment is defined in LPAR image profile via SE/HMC
- IBM Z firmware enforces domain usage
- The Crypto Express coprocessor manages assignment of master keys to cryptographic domains.
What are the different modes of operation for Crypto Express adapters?

Crypto Express adapters can be configured in three different modes:

**Accelerator Mode:**
- Request is processed fully in hardware (versus PowerPC)
- Supports clear key RSA operations (e.g. SSL Acceleration)

**CCA Coprocessor Mode:**
- Supports the IBM Common Cryptographic Architecture (CCA) for financial services standards.
  - With CEX6S, supports domain-segregated PCI-HSM Compliant mode. **Note:** Requires a TKE Workstation.
- Request is sent first to the internal IBM PowerPC for processing (default mode)
- Supports secure key crypto operations (i.e. keys encrypted by Master Keys)

**EP11 Coprocessor Mode:**
- Supports the PKCS #11 programming interface for public sector requirements. Designed for extended evaluations (FIPS and Common Criteria certifications)
- Request is sent first to the internal IBM PowerPC for processing (default mode)
- Requires the use of the TKE Workstation
- Supports secure key crypto operations (i.e. keys encrypted by Master Keys)

**Native PCIe card (FC 0890):**
- Resides in the PCIe I/O drawer
- Requires CPACF Enablement (FC 3863)
- Up to 16 features per server

**Designed to Meet Physical Security Standards:**
- FIPS 140-2 level 4
- ANSI X9.97
- Payment Card Industry (PCI) HSM
- Deutsche Kreditwirtschaft (DK)
What are Master Keys?

- Master Keys are used to protect sensitive cryptographic keys that are active on your system.
- Master Keys are stored in secure hardware on the crypto express card.
- Master Keys are used only to encipher and decipher keys.
- Master Keys should be changed periodically.

<table>
<thead>
<tr>
<th>Master Key</th>
<th>Key Size</th>
<th>Protects</th>
</tr>
</thead>
<tbody>
<tr>
<td>DES-MK</td>
<td>16-byte or 24-byte</td>
<td>DES keys</td>
</tr>
<tr>
<td>AES-MK</td>
<td>32-byte</td>
<td>AES and HMAC keys</td>
</tr>
<tr>
<td>RSA-MK</td>
<td>24-byte</td>
<td>RSA private keys</td>
</tr>
<tr>
<td>ECC-MK</td>
<td>32-byte</td>
<td>ECC, RSA-AESM, RSA-AESC keys</td>
</tr>
<tr>
<td>P11-MK</td>
<td>32-byte</td>
<td>PKCS #11 keys</td>
</tr>
</tbody>
</table>

**Note:** All Master Keys do not need to be loaded. Load only the Master Keys that you need for your environment.
How do you generate, maintain and manage Master Keys?

- Using the Trusted Key Entry (TKE) Workstation
  - Applicable for initialization of ICSF Key Data Sets (i.e. key stores) and Crypto Express adapters
  - Applicable for master key change operations
  - Required for EP11 Master Key management & PCI-HSM Master Key management
  - Separate, priced product

- Using the Pass Phrase Initialization (PPINIT) Panel
  - Applicable for initialization of ICSF Key Data Sets (i.e. key stores) and Crypto Express adapters
  - **NOT** applicable for master key change operations
  - Included with z/OS and ICSF

- Using the ICSF Master Key Entry Panels
  - Applicable for initialization of ICSF Key Data Sets (i.e. key stores) and Crypto Express adapters
  - Applicable for master key change operations
  - Included with z/OS and ICSF
### Special Considerations for Master Keys

- Master Keys are high value keys that must be protected.
  - Loading Master Keys on a panel means that the key is viewable to passersby!
  - The most secure way to load a Master Key is to use the TKE Workstation with smart cards.
    - The P11 Master Key may ONLY be loaded using a TKE Workstation.
- If you plan to use the PPINIT or the Master Key Entry panels to manage Master Keys, consider how you would save the key material for future re-entry (e.g. new Crypto Express adapter, disaster recovery).
- For disaster recovery, the same Master Keys must be loaded onto the backup system.

<table>
<thead>
<tr>
<th>Option</th>
<th>Details</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Print Screen</td>
<td>Use a Print Screen key or tool to capture the screen</td>
<td>Sensitive material can be immediately printed and stored in envelopes in a locked safe. No need to save on a local machine or USB stick.</td>
<td>Cannot use copy / paste to re-enter key material</td>
</tr>
<tr>
<td>Removable Storage Media</td>
<td>Copy and paste key material to a text file that is saved on a secure storage device (e.g. USB stick).</td>
<td>Easy to copy / paste the key material to the panels for re-entry.</td>
<td>The key material is only as secure as the storage media.</td>
</tr>
<tr>
<td>Other Ideas?</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Trusted Key Entry (TKE) Workstation

TKE is an appliance that simplifies the management of IBM Z Host Cryptographic Modules running in Common Cryptographic Architecture (CCA) or IBM Enterprise PKCS#11 (EP11) mode, using compliant level management techniques.

- Features for Managing Module
  - Scoped and Domain Scoped Administrative settings on Host Cryptographic Modules
- Secure, hardware-based
  - Master Key and Operational key management
- Highly secure and efficient movement of administrative settings from one Host Cryptographic Module to another
TKE Workstation Features

Features for Managing Module Scoped and Domain Scoped Administrative settings on Host Cryptographic Modules

**Featuring:** Secure, simplified administrative management of multiple domain host cryptographic modules in complex configurations

Secure, hardware-based Master Key and Operational key management

**Featuring:** Compliant level hardware-based key management with proper encryption strengths, dual controls, and security relevant auditing

Highly secure and efficient movement of administrative settings from one Host Cryptographic Module to another

**Providing:** Secure, fast, and accurate deployment of new crypto modules on production, test, or disaster recovery systems

---

**Popular Features**

- Domain Grouping to broadcast a command to a set of domains
- Secure Loading of CCA Master Keys (MKs)
- Manage domains higher than 16
- Migration Wizards
- Enable/disable Access Control Points (ACPs)
- Loading MKs for inactive LPARs
- Loading PIN decimalization tables
- Loading EP11 Master Key
TKE Workstation 9.0

- **Base Hardware**
  - TKE 9.0 Workstation with a 4768 Cryptographic Adapter (required to manage Crypto Express 6S on z14)
    - TKE 9.0 Tower Workstation (FC 0086)
    - TKE 9.0 Rack-Mounted Workstation (FC 0085)
  - **Additional Hardware:** Smart card readers and smart cards
    - Smart cards and readers are required for some TKE functions
      - Host module migration wizard
      - Management of EP11
      - NEW: Required for managing of PCI-HSM compliance mode
    - IBM Highly recommends using smart cards to hold key material
  - **Migration Considerations**
    - TKE 8.0 & 7.3 (FC 0842 only) workstations can be upgraded to TKE 9.0 with purchase of 4768
    - Omnikey Cardman 3821 smart card readers can be carried forward to any TKE 9.0 workstation
    - Previously initialized and personalized smart cards can be carried forward and used on any TKE 9.0 workstation.

- **Feature Overview**
  - **Usability**
    - Copy smart card content to new zone
    - Save/Restore data using TKE directory structure
    - Ability to generate key parts without opening a host
    - Reduced number of commands for domain group load, set, and clear (CCA 5.3 or later)
    - New IBM-supplied role for TKE workstation profile group members
  - **Workstation Integrity**
    - Can configure to force sign-on after boot
    - Can use Multi Factor Authentication for workstation sign on
  - **Audit**
    - TKE Audit Log browse application
    - TKE heartbeat audit record for periodic status monitoring
  - **PCI HSM**
    - TKE required to manage PCI-HSM compliant domains on Crypto Express6S
User Defined Extensions (UDX)

UDX support available for Crypto Express adapters defined as **CCA coprocessors**

- Supports **customized functions** in addition to the CCA API, which **execute inside the secure crypto adapter**
  - Standard CCA functions plus UDX enhancements available

- Tied to specific versions of the CCA code and the related host code
  - Must be rebuilt each time these IBM code modules change

Note: Installation of a UDX is a disruptive (non-concurrent) operation for z/OS. Not supported for Linux on z.
IBM Z Crypto Hardware Virtualization
z/VM Virtualization of Hardware Cryptography

Crypto Express features associated with your z/VM partition are **virtualized for the benefit of your guests:**

**APDED (“Dedicated”)**
- Connects a particular AP domain (or set of domains) of one or more Crypto Express adapters directly to a virtual machine – no hypervisor interference
- All card functions are available to the guest

**APVIRT (“Shared”)**
- Virtual machine can access a collection of domains controlled by the hypervisor layer
- Meant for clear-key operations only – sharing crypto material might otherwise break security policy.
Sample of Virtualization: LinuxONE Developer Cloud

- **Crypto operations**: SSH (RSA, SHA-2, AES), and *whatever data is handled inside the guests*
- **Environmental Requirements**: Relocatable (it's a cloud)
- **Recommended Hardware**:
  - CPACF
  - Crypto Express Accelerator in shared configuration (“APVIRT”)
    - Assign 1 domain from 2-3 different features (hardware failover, performance)
Sample of Virtualization: Linux on z Blockchain (*not* HSBN)

- **Crypto operations**: A lot. It's a Blockchain
- **Environmental Requirements**: Protection of key material. (It's a Blockchain.)
- **Recommended Hardware**:
  - CPACF (required for secure and protected key ops on the crypto adapters)
  - Crypto Express Coprocessors
    - One domain per guest participating in the Hyperledger fabric
IBM Z Crypto Software - Linux on z & LinuxOne
The libica library provides a C API for:

- Clear key CPACF symmetric encryption & hashing
  - **Hashing:**
    - SHA1, SHA224, SHA256, SHA384, SHA512, SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE-128, SHAKE-256
  - **Encryption:**
    - DES / 3DES: ECB*, CBC*, CBC_CS, CFB, OFB, CTR, CBC_MAC, CMAC
    - AES128/192/256: ECB*, CBC*, CBC_CS, CFB, OFB, CTR, CBC_MAC, CMAC
    - AES 128/256: XTS
  - Authenticated Encryption with Associated Data (AEAD):
    - AES128/192/256: CCM, GCM
  - **Clear key RSA asymmetric encryption & signing**
  - CCA coprocessor & accelerator support
  - Modular exponentiation:
    - ME*: encrypt/verify (decrypt/sign)
    - CRT*: decrypt/sign
  - **Key generation**
  - **Random numbers (TRNG and PRNG)**
    - CCA coprocessor, CPACF, kernel crypto
  - **CCA coprocessor & accelerator support**
  - **Modular exponentiation:**
    - ME*: encrypt/verify (decrypt/sign)
    - CRT*: decrypt/sign
  - **Key generation**
  - **Random numbers (TRNG and PRNG)**
    - CCA coprocessor, CPACF, kernel crypto
  - **CCA coprocessor & accelerator support**
  - **Modular exponentiation:**
    - ME*: encrypt/verify (decrypt/sign)
    - CRT*: decrypt/sign
  - **Key generation**
  - **Random numbers (TRNG and PRNG)**
    - CCA coprocessor, CPACF, kernel crypto

* with software fallback

---

© 2017 IBM Corporation
openSSL / libcrypto

openSSL implements SSL and TLS protocols
- libcrypto is the crypto library of openSSL
  - used by many open source projects (e.g. openssh, apache mod_ssl, nodes.js, PHP, postgres, MongoDB EE, Ruby)
  - libcrypto version >= 1.0.x has built-in CPACF support
    - CPACF: SHA1, SHA2
    - CPACF: AES: ECB, CBC, CTR, XTS
    - CPACF: GHASH
    - IBM Z assembler: long number arithmetic
- the ibmca dynamic engine supports
  - CPACF: SHA1, SHA256
  - CPACF: DES/3DES/AES: ECB, CBC, CFB, OFB
  - CEX*A/CEX*C: RSA, DH, DSA
  - CPACF, CEX*C: pseudo random number generation
- transparent usage of ibmca engine must be configured in openssl.cnf
openCryptoki (1 of 2)

- **soft token**
  - provides clear key cryptographic functions
  - pure software implementation, relies on libcrypto (openSSL)
  - platform independent

- **ica token**
  - provides clear key cryptographic functions
  - uses libica
  - exploits CPACF, Crypto Express accelerators and CCA co-processors
  - IBM Z specific

- **ep11 token (since openCryptoki 3.1)**
  - provides secure key cryptographic functions
  - exploits Crypto Express EP11 co-processors
  - IBM Z specific
openCryptoki (2 of 2)

- **cca token**
  - provides secure key cryptographic functions
  - uses CCA library (libcsulcca)
  - exploits Crypto Express CCA co-processors
  - IBM Z specific

- **icsf token (since openCryptoki 3.0)**
  - remote access to cryptographic functions on a z/OS based ICSF crypto server
  - uses LDAP protocol
  - platform independent
GSKit is not available as stand-alone library but…

- GSKit is part of many IBM Products (e.g. DB2, WebSphere MQ, TSM)

- GSKit is part of the ICC component
  - uses CPACF functions for
    - AES: ECB, CBC, CFB, OFB, XTS, GCM, CCM
    - SHA1
    - SHA2
    - GHASH

- GSKit can be configured to use crypto functions from a PKCS #11 API (i.e., it can link to openCryptoki)
The IBM Java 8 Java Cryptography Extension (JCE)

- uses CPACF instructions to accelerate
  - DES, 3DES, AES with ECB, CBC, OFB, CFB, CFB x, and GCM modes of operation
  - SHA1, SHA2, SHA3 and SHAKE
  - PRNG, TRNG
- uses IBM Z specific code to accelerate
  - ECDHE – NIST P256 and ECDHE-ECDSA

The IBMPKCS11Impl provider

- provides an PKCS #11 plug-in interface for JCA
- tested on
  - recent Crypto adapters
  - recent SLES and RHEL distributions
  - ICA, CCA and ICSF tokens

For details see: http://www-01.ibm.com/support/docview.wss?uid=swg21967855

The IBMJCECCA provider

- provides plug-in CCA interface for JCA
- common for z/OS and Linux on z

Crypto Statistics (libica)

ICASTATS for libica (as of version 2.4)
- part of libica package
- user specific data collection (root can see all data)
- stores statistics data after application is finished until explicit reset
- options for non-privileged users:
  \texttt{icastats \{--delete|--reset|--help|--version\}}
- additional options for root:
  \texttt{icastats \{--all|--delete-all|--reset-all|--summary
  |--user <username>\}}
  - all: get statistics for all users
  - summary: cumulative statistics of all users

Note: icastats only measures crypto done by libica
- no crypto done w/o libica like
  - kernel, libcrypto, GSKit, JCE
Crypto Statistics (CPACF and Crypto Express)

CPACF statistics with cpacfstats

- available in s390tools 1.29 (developerworks)
- counts AES, DES, SHA and PRNG operations
- can count CPACF usage of SW not using libica (e.g. kernel, GSKit)
- works for Linux running in an LPARs
- LPAR must be authorized to use CPU counters
  - “Crypto activity counter set authorization control" checkbox
- Example
  - start cpacfstatd daemon:
    
    ```
    # cpacfstatd
    ```
  - enable AES counter
    
    ```
    # cpafafstat -e aes
    ```
  - print AES counter
    
    ```
    # cpacfstat -p aes
    aes counter: 144
    ```

Statistics of Crypto Express requests

- use lszcrypt –VV or lszcrypt –VVV
  
  ```
  [root@s3560007 ~]# lszcrypt –VV
  card00: CEX5A ... request_count=2
  card01: CEX5C ... request_count=51
  card05: CEX5P ... request_count=1092
  ```

cpacfstats availability through s390tools

<table>
<thead>
<tr>
<th></th>
<th>upstream (developerworks):</th>
<th>RHEL 6.8</th>
<th>RHEL 7.3</th>
<th>SLES 11 SP4:</th>
<th>SLES 12 SP2:</th>
<th>Ubuntu 16.4:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
End-to-End Data at Rest Encryption with dm-crypt

- E2E data encryption
  - The complete I/O path outside the kernel is encrypted:
    - hypervisor, adapters, links, switches, disks
- dm-crypt
  - a mechanism for end-to-end data encryption
  - data only appears in the clear in application
- Linux kernel component that transparently
  - for all applications
  - for a whole block device (partition or logical volume)
  - encrypts all data written to disk
  - decrypts all data read from disk
- How it works:
  - uses in kernel-crypto
    - can use IBM Z CPACF Crypto:
      - AES-CBC
      - XTS-AES (recommended)
  - encrypted volumes must be opened before usage
    - opening provides encryption key to kernel
    - establishes virtual volume in /dev/mapper to be used by applications
IBM Z Crypto Software - z/OS
ICSF provides the application programming interfaces by which applications request cryptographic services such as:

- Encryption and Decryption
- Digital Signature Generation and Verification
- MAC Generation and Verification
- HMAC Generation and Verification
- Key and Key Pair Generation
- Key Derivation
- Key Agreement
- Data Hashing
- Random Number Generation
- Financial PIN Generate / Verify / Translate / Encrypt

ICSF callable services and programs can be used to generate, maintain, and manage keys that are used in the cryptographic operations.

ICSF uses cryptographic keys to:
- Protect data
- Protect and distribute additional keys
- Verify message integrity
- Generate, protect and verify PINs
- Generate and verify signatures

ICSF provides panels to load CCA master key values onto secure cryptographic features, allowing the hardware features to be used by applications.
Common Cryptographic Architecture (CCA)

IBM Common Cryptographic Architecture
IBM proprietary cryptographic application programming interface (API) providing a broad range of cryptographic services including
- standard cryptographic algorithms
- financial services standards

z/OS ICSF Naming Conventions for CCA
CSNB* = CCA 31-bit Symmetric Key API
CSNE* = CCA 64-bit Symmetric Key API
CSND* = CCA 31-bit Asymmetric Key API
CSNF* = CCA 64-bit Asymmetric Key API

CCA Functions & Algorithms
- Encrypt / Decrypt (AES, DES, DES3, RSA)
- Sign / Verify (RSA, ECC)
- MAC Generate / Verify (AES, DES, DES3)
- HMAC Generate / Verify (HMAC)
- Key Generate (AES, DES, DES3, HMAC)
- Key Pair Generate (RSA, ECC)
- Key Agreement (ECC, DH)
- One Way Hash
- Random Number Generate
- Key Import / Export
- TR-31 Block Import / Export
- Financial Crypto
- PIN Generate / Verify / Translate
- PIN Encrypt
- Diversified Key Generate
- Derive Unique Key Per Transaction (DUKPT)
- … And Many More!
PKCS #11 Cryptographic Token Interface Standard

PKCS #11 Cryptographic Architecture

- Originally published by RSA Laboratories, now maintained by OASIS
  - Standard API for devices that hold cryptographic information and perform cryptographic functions

PKCS #11 Functions & Algorithms

- Encrypt / Decrypt (AES, DES, TDES, RSA)
- Sign / Verify (RSA, DSA, ECDSA)
- HMAC Generate / Verify
- Key Generate (DES, TDES, AES, Blowfish, RC4)
- Key Pair Generate (RSA, DSA, EC)
- Key Derivation
- Domain Parameter Generation (DH)
- One Way Hash
- Random Number Generate
- Wrap / Unwrap Key

Designed for portability and FIPS/Common Criteria certification

z/OS ICSF Naming Conventions for PKCS #11

CSFP* = PKCS #11 APIs
ICSF provides callable services and utilities to generate and store operational keys into ICSF Key Data Sets (KDS) and/or return the keys to the caller.

Each KDS is a VSAM data set for persistent objects (e.g. keys, certificates) with programming interfaces for object management.

Each record in the KDS contains the object and other information about that object.

ICSF uses operational keys in cryptographic functions to:
- Protect data
- Protect other keys
- Verify that messages were not altered
- Generate, protect and verify PINs
- Distribute keys
- Generate and verify signatures

<table>
<thead>
<tr>
<th>ICSF Key Data Sets</th>
<th>CKDS</th>
<th>PKDS</th>
<th>TKDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cryptographic Key Data Set</td>
<td>CCA Symmetric Keys</td>
<td>CCA Asymmetric Keys</td>
<td>PKCS#11 Keys, Certificates</td>
</tr>
<tr>
<td></td>
<td>AES, DES and HMAC</td>
<td>RSA, ECC and Trusted Blocks</td>
<td>All algorithms</td>
</tr>
</tbody>
</table>

© 2017 IBM Corporation
IBM Z Crypto Stack – z/OS

z/OS Software

- ICSF
- RACF
- RMF
- DFSMS
- Websphere
- System SSL
- z/OS PKI Services
- Java PKCS#11 Provider
- z/OS Communications Server
Which z/OS components exploit ICSF?

**z/OS Software Components**
- System SSL
- Java Cryptography Extension
- RACF Security / RACDCERT
- DB2 Database
- PKI Services
- IBM Tivoli Directory Server
- Kerberos Network Authentication Service
- Websphere MQ
- Websphere Application Server
- z/OS Communications Server
- z/OS DFSMS
- Coupling Facility (CF)

**IBM & ISV Solutions**
- Guardium Data Encryption
- Sterling Connect:Direct

...
How do you view the contents of a Key Data Set?

With HCR77C1:

ICSF supports a CKDS Browser (ICSF Panel Option 5.5) that allows users to list records in the active CKDS.

When the format of the CKDS is the common record format (referred to as KDSR), the list of label will show:
- state of the record (i.e. active, pre-active, deactivated, archived)
- options to display the key attributes and record metadata, delete the record, archive the record or recall the record.

When the format of the CKDS is non-KDSR, the options will be to display key attributes and delete the record.

Note: Alternative methods to view the contents of a Key Data Set include IDCAMS REPRO, PKCS #11 Token (TKDS) Browser and the Key Dataset List (CSFKDSL) callable service.
What are clear, secure and protected keys?

Secure keys have key values that are encrypted by a Master Key on a tamper-responding CryptoExpress adapter.

Only protected keys created from secure keys should be used for Pervasive Encryption.

Secure Key
- Key values are encrypted under a Master Key. Crypto operations are performed only on a Crypto Express adapter

Protected Key
- Key values are encrypted under a CPACF wrapping key. Crypto operations are performed only using CPACF

Clear Key
- Key values are not encrypted. Crypto operations may be performed in CPACF or on a Crypto Express adapter

Note: With z/OS data set encryption, protected keys are implicitly created from secure keys.
How do you audit key usage and the key life cycle?

**Key usage auditing** must be explicitly enabled in the ICSF Installation Options Data Set (IODS) or using the SETICSF OPT operator commands.

<table>
<thead>
<tr>
<th>ICSF IODS Option</th>
<th>SMF Record Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDITKEYUSGCKDS(TOKEN(YES),LABEL(YES),INTERVAL((n)))</td>
<td>Type 82 Subtype 44</td>
</tr>
<tr>
<td>AUDITKEYUSGPKDS(TOKEN(YES),LABEL(YES),INTERVAL((n)))</td>
<td>Type 82 Subtype 45</td>
</tr>
<tr>
<td>AUDITPKCS11USG(TOKENOBJ(YES),SESSIONOBJ(YES),NOKEY(YES),INTERVAL((n)))</td>
<td>Type 82 Subtype 46 &amp; Type 82 Subtype 47</td>
</tr>
</tbody>
</table>

**Key life cycle auditing** must be explicitly enabled in the ICSF Installation Options Data Set (IODS) or the SETICSF OPT operator commands.

<table>
<thead>
<tr>
<th>ICSF IODS Option</th>
<th>SMF Record Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDITKEYLIFECKDS(TOKEN(YES),LABEL(YES))</td>
<td>Type 82 Subtype 40</td>
</tr>
<tr>
<td>AUDITKEYLIFEPKDS(TOKEN(YES),LABEL(YES))</td>
<td>Type 82 Subtype 41</td>
</tr>
<tr>
<td>AUDITKEYLIFETKDS(TOKENOBJ(YES),SESSIONOBJ(YES))</td>
<td>Type 82 Subtype 42</td>
</tr>
</tbody>
</table>
How do you track crypto usage?

With HCR77C1, ICSF provides crypto usage tracking of applications and components that invoke ICSF services. Crypto usage tracking can be enabled/disabled at ICSF initialization using the Installation Options Data Set (IODS) or dynamically using SETICSF OPT operator commands.

<table>
<thead>
<tr>
<th>ICSF IODS Option</th>
<th>SMF Record Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATS(ENG,SERV,ALG)</td>
<td>Type 82 Subtype 31</td>
</tr>
</tbody>
</table>

**ENG:** Tracks crypto engine usage. When enabled, ICSF tracks the usage of Crypto Express Adapters, Regional Cryptographic Servers, CPACF and Software.

**SRV:** Tracks crypto service usage. When enabled, ICSF tracks the usage of ICSF callable services and User Defined Extensions (UDX).

**ALG:** Tracks crypto algorithm usage. When enabled, ICSF tracks the usage of crypto algorithms that are referenced in cryptographic operations.

Crypto usage data collection is synchronized to the SMF recording interval. Your SMFPRMxx member must contain:

- The collection interval (INTVAL)
- The synchronization value (SYNCVAL)
- The Crypto Usage Statistics Subtype 31 for ICSF Type 82 records (TYPE)
How do you protect ICSF resources?

**ICSF Keys, APIs and Utilities**
- The **CSFSERV** class controls access to ICSF callable services and *ICSF TSO panel utilities*.
- The **CSFKEYS** class controls access to cryptographic keys in the ICSF Key Data Sets (CKDS and PKDS) and **enables/disables the use of protected keys**.
- The **CRYPTOZ** class controls access to, and defines a policy for PKCS#11 token in the Token Data Set (TKDS).
- The **XCSFKEY** class controls the ability to export a symmetric key with the Symmetric Key Export callable services.

**ICSF Key Data Sets**
- The **DATASET** class can be configured to protect the ICSF Key Data Sets.

**ICSF MVS Console Commands**
- The **OPERCMDS** class controls the ability to issue MVS console commands for “DISPLAY ICSF” and “SETICSF”.

**Key Store Policy**
- Define additional security policies pertaining to the use of key tokens.

**Note: CCA Coprocessor Access Controls** on the cryptographic coprocessor can be used to further control cryptographic operations.
Additional z/OS Key Stores

- **RACF** provides the RACDCERT GENCERT command to generate and store keys into the RACF database and ICSF Key Data Sets (PKDS and TKDS). RACF also provides the RACDCERT CONNECT command to add certificates to RACF Keyrings.

- **SystemSSL** provides the gskkyman utility to generate and store certificates into key database files. SystemSSL can also read from RACF Keyrings and generate and store certificates into PKCS#11 Tokens (TKDS).

- **JCE** provides APIs and utilities to generate and store keys and certificates into ICSF Key Data Sets, RACF Keyrings, and Java Key Stores.
What cryptographic network protocols are supported on z/OS?

z/OS provides 4* main mechanisms to protect TCP/IP traffic:

1. **TLS/SSL direct usage**
   - Application is explicitly coded to use these
   - Per-session protection
   - TCP only

2. **Application Transparent TLS (AT-TLS)**
   - TLS/SSL applied in TCP layer as defined by policy
   - Configured in AT-TLS policy via Configuration Assistant
   - Typically transparent to application
   - TCP/IP stack is user of System SSL services

3. **Virtual Private Networks using IPSec and IKE**
   - "Node to node" encryption
   - IPSec implemented in IP layer as defined by policy
   - Completely transparent to application
   - Wide variety (any to all) of traffic is protected
   - Various topologies supported (host to host, host to gateway, etc.)
   - IKE negotiates IPSec tunnels dynamically

4. **Secure Shell using z/OS OpenSSH**
   - Mainly used for sftp on z/OS, but also offers secure terminal access and TCP port forwarding
   - Configured in ssh configuration file and on command line
   - TCP only

* - z/OS also provides Kerberos support, but that is use mainly for peer authentication only
IBM Z Crypto Software - z/VM
IBM Z Crypto Stack – z/VM

- Software crypto support for z/VM service virtual machines
  - z/VM System SSL (port of z/OS function) and ICSFLIB
  - Pipes crypto stages for CMS application programming
  - CPACF and Crypto Express offload whenever available
- Some capabilities provided as hardware-only
  - KDFAES password encryption for RACF/VM
  - z/VM Encrypted Paging for z14
If Crypto Express domains are defined for sharing (APVIRT), then the z/VM TLS/SSL Server will use them

- **Clear-key RSA operations** are the primary beneficiary
  - Handshaking, rather than data transfer – *benefit will come from a lot of connections*
  - Will still use CPACF when pertinent
- Meant as a performance enabler, not to replace key storage

- Accelerate cryptographic operations for data in flight
  - Connections to hypervisor
  - Connections inside of the hypervisor
z/VSE provides hardware-accelerated encryption support by exploiting cryptographic features on z Systems processors.

- Crypto Express adapters
  - RSA support
  - ECC support
- CP Assist for Cryptographic Function (CPACF)
  - Symmetric algorithms such as Triple-DES, AES, or SHA.

Cryptographic hardware is transparently used by TCP/IP for z/VSE, IPv6/VSE and applications like Encryption Facility for z/VSE.
IBM Z Crypto Solutions
IBM Enterprise Key Management Foundation (EKMФ)

Secure workstation
- is used for generating all new keys by users authenticated with smart cards or automatically based on requests. Workstation utilizes IBM 4765/7

Central repository
- contains keys and metadata for all cryptographic keys produced by the EKMФ workstation. This enables easy backup and recovery of key material.

EKMФ Browser
- features monitoring capabilities and enables planning of future key handling session to be executed on the workstation.

Note that while this is a mainframe centric view, EKMФ supports distributed platforms as well.
Key Management Features for EKMF

- Basic key management functions include:
  - key generation
  - key import
  - key export
  - key print
  - key administration

- Key management functions are controlled by key templates and key policies. Key templates:
  - control functions for a key
  - predefine key attributes

- When generating or entering a key, the key is automatically distributed to the servers specified in the key template.
  - ICSF Key Data Sets
  - RACF Key Rings (i.e. SKLM, z/OS PKI)
  - … and more
IBM Security Key Lifecycle Manager (SKLM)

- IBM Security Key Lifecycle Manager provides centralized key management for self-encrypting devices.

Self-encrypting devices protect data if you lose control of the device:

- Data on the truck traveling between datacenters
- Data at rest within the datacenter
- Decommissioned storage devices
Key Management Features for SKLM

SKLM for Distributed Systems
SKLM v2.7 supports the IBM Proprietary Protocol (IPP) and industry-standard Key Management Interoperability Protocol (KMIP) for key distribution with storage devices. Features include:

- Key generation, import and export
- Secure storage of key material
- Automatic assignment and rotation of keys
- Key serving at the time of use

SKLM for z/OS
SKLM for z/OS supports the IBM Proprietary Protocol (IPP) for key distribution with storage devices. SKLM for z/OS can use ICSF through JCE hwkeytool or RACF GENCERT commands to push RSA key pairs to the ICSF PKDS and AES keys to the ICSF CKDS. Features include:

- Key generation, import and export
- Secure storage of key material
- Key serving at the time of use

Note: SKLM can **not** be used to manage z/OS data set encryption keys.
Advanced Crypto Service Provider (ACSP)

The Advanced Crypto Service Provider is a remote crypto services solution that enables distributed clients to access cryptographic hardware on IBM Systems over a network.

- Utilizes existing z Systems infrastructure (cost efficient)
- Uses multiple crypto processors with ICSF doing the load balancing (scalable)
- Provides efficient ACSP implementation (high performance)
- Enables centralized management (efficient operation, policy compliance)

ACSP supports Java, C and REST APIs to invoke z/OS crypto functions from remote applications and platforms.
Encryption Facility (EF) for z/OS and z/VSE

The Encryption Facility is a host-based encryption and key-management solution specifically designed to protect sensitive data that's being exchanged with trusted business partners or archived for backup and recovery purposes.

- Provides a business-to-business encryption capability to help companies that rely on exchange of tapes with their partners to complete these business transactions.

- Leverages IBM Z software and hardware capabilities to encrypt and compress data as it's sent to tape.

- Written in Java, so the client can be downloaded from the Internet and used on multiple platforms.

Encryption Facility provides services for:
- Public-key based encryption
- Passphrase-based encryption
- Modification detection of encrypted data
- Compression of packaged data before encryption
- Importing and exporting of OpenPGP certificates – Binary or ASCII armor format
- Digital signatures of data

Data encrypted with EF can be exchanged between different operating systems. EF for z/VSE can read encrypted files that were created by the Encryption Facility for z/OS or the z/OS Java client. The z/OS facilities can read encrypted files that were created by EF for z/VSE. Other platforms than z/VSE or z/OS may encrypt / decrypt such data as well.
Guardium Data Encryption (GDE) for DB2 and IMS

Guardium Data Encryption for DB2 and IMS (i.e. GDE4Z) provides row and field based encryption of DB2 and IMS data.
**zSecure Suite**

### Security audit and compliance

- **Enhanced data collection** of SMF audit information from:
  - RACF, DB2, CICS, IMS, MQ, SKLM, WAS, UNIX, Linux on System z, OMEGAMON XE on z/OS, FTP, Communication Server, TCP/IP, PDSE and more

- **Automated remediation** to detect and prioritize potential threats with security event analysis

- **Real-time alerts** of potential threats and vulnerabilities

- **Compliance monitoring and reporting**
  - PCI-DSS, STIGs, GSD331, and site-defined requirements

- **Comprehensive customized audit reporting**

- **Detect harmful system security settings** with automated configuration change checking

### Administration management

- **Reduce administrative overhead** with security management tasks

- **Prevent abuse of special roles and authorization**
  - Privileged user monitoring
  - Entitlement checking for Identity governance

- **Enforce security policies** by blocking dangerous commands and potential errors

- **RACF data set cleanup** of unused security profiles and inactive / terminated users
IBM Z Pervasive Encryption
# IBM Z Pervasive Encryption

Enabled through full-stack platform integration

<table>
<thead>
<tr>
<th>Feature</th>
<th>Details</th>
</tr>
</thead>
</table>
| Integrated Crypto Hardware | Hardware accelerated encryption on every core – CPACF performance improvements of up to 7x  
                             | Next Gen Crypto Express6S – up to 2x faster than prior generation                                                                        |
| Data at Rest             | Broadly protect Linux volumes and z/OS data sets\(^1\) using policy controlled encryption that is transparent to applications and databases |
| Clustering               | Protect z/OS Coupling Facility\(^2\) data end-to-end, using encryption that’s transparent to applications                                 |
| Network                  | Protect network traffic using standards based encryption from end to end, including encryption readiness technology\(^2\) to ensure that z/OS systems meet approved encryption criteria |
| Secure Service Container | Secure deployment of software appliances including tamper protection during installation and runtime, restricted administrator access, and encryption of data and code in-flight and at-rest |
| Key Management           | The IBM Enterprise Key Management Foundation (EKMF) provides real-time, centralized secure management of keys and certificates with a variety of cryptographic devices and key stores. |

\(^{1}\) Statement of Direction* in the z/OS Announcement Letter (10/4/2016) - [http://ibm.co/2ldwKoC](http://ibm.co/2ldwKoC)

\(^{2}\) IBM z/OS Version 2 Release 3 Preview Announcement Letter (2/21/2017) - [http://ibm.co/2I43c0N](http://ibm.co/2I43c0N)

And we’re just getting started...
Data Protection // z/OS Dataset Encryption

Protection of data at-rest

Client Value Proposition:
Reduced cost of encryption along with simple policy controls allows clients to enable extensive encryption to protect data in mission critical databases including DB2, IMS and VSAM

z/OS Dataset Encryption:
• Application transparent & enabled by policy
• Encryption tied to fine grained access control
• Host encryption via CPACF as data written-to or read-from disk.
• Supports ext. format sequential & VSAM
• Includes HSM & DSS migration/backup of encrypted data sets
• Replicated data remains encrypted
• Supports: CICS, DB2, IMS, Logger, & zFS

In-memory system or application data buffers will not be encrypted
Data Protection // Existing Disk Encryption

Protection of data at-rest

DS8000 Disk Encryption
Encrypting disk drives protect data at rest when disk drives are retired, sent for repair or repurposed

Once the key has been served to storage system any system connecting to storage system can retrieve unencrypted data

Legend:
- encrypted data
- unencrypted data
Data Protection // Coupling Facility Encryption

Protection of data in-flight and in-use (CF)

Client Value Proposition:
Simplify and reduce cost of compliance by removing CF and CF data from compliance scope (i.e. ability to encrypt all CF data)

End-to-End encryption of CF Data:
- Host Protected key CPACF Encryption (High Performance / Low Latency)
- Data encrypted in the host and remains encrypted until decrypted by host
- No application enablement required
- List & Cache Structures only – No Lock!

Legend:
- *** - encrypted data
- abc - unencrypted data
Data Protection // z/OS Network Security
Protection of data in-flight

Client Value Proposition:
Not all organizations use host-based network encryption today. Reduced cost of encryption enables broad use of network encryption and enhanced audit simplifies compliance.

Communication Server
z Encryption Readiness Technology (zERT):
A z/OS administrator can determine when network traffic meets specified policy with new discovery and reporting capabilities.

Currently no single method to easily determine which application traffic patterns are protected.
Data Protection // Linux on z Volume Encryption

Protection of data at-rest

Client Value Proposition:
Integration of hardware accelerated Crypto into standard components for wide reach into solutions

Linux on z and LinuxONE
Focus on Transparent Enablement:
• Transparent data encryption optimized with z14 CPACF hardware performance gains
• Leverage industry-unique CPACF encryption which prevents raw key material from being visible to OS and applications.

Status: kernel support for protected keys accepted upstream. Code to manage LUKS formatted dm-crypt disks using protected keys submitted upstream.
Data Protection // Linux on z Network Security

Protection of data in-flight

Client Value Proposition: Not all organizations use host-based network encryption today... reduced cost of encryption enables broad use of network encryption

Focus on Transparent Enablement:
- Transparently accelerate TLS & IPSec using CPACF & SIMD to leverage hardware performance gains

Status: openSSL patches to support z14 HW crypto submitted upstream. openSSL ibmca engine with z14 GCM support accepted upstream.

Networking

SAN

Storage System
**Data Protection // Secure Service Container**

*Extending the value of z hardware crypto*

**Client Value Proposition:**

- Simplified, fast deployment and management of packaged solutions
- Tamper protection during Appliance installation and runtime
- Confidentiality of data and code running within the Appliance both at flight and at rest
- Restricts administrator access to workload and data

Secure Service Container architecture builds on the value of z systems hardware crypto using a runtime environment designed to help clients reduce risk.
Data Protection // z/VM Encrypted Paging

**Client Value Proposition:**
Protect guest paging data from administrators and/or users with access to volumes

**Encrypted Paging**
- **Threat:** access to sensitive data when stored on CP owned disk
- **Solution:** encrypt guest data on page-out.

**Notes:**
- Paging is not SSI-relevant
- Paging data does not need to survive an IPL
- Ephemeral CPACF protected-key stored in CP (not on disk somewhere)
- AES encryption
- Very low overhead via CPACF
Data Protection // z/VM Network Security
Protection of data in-flight

Threat: disclosure of sensitive data in flight to the hypervisor layer
Solution: encrypt traffic in flight.

Notes:
- Automatic use of CPACF for symmetric algorithms
- One-line change to enable automatic use of Crypto Express features for acceleration of asymmetric algorithms
- Built on System SSL and ICSFLIB for z/VM

Client Value Proposition:
Not all organizations use host-based network encryption today… reduced cost of encryption enables broad use of network encryption
Data Protection // z/TPF Transparent Database Encryption

Technical Foundation

z/TPF at-rest Data Encryption

✦ Automatic encryption of at-rest data
✦ No application changes required
✦ Database level encryption using highly efficient CPACF HW crypto
✦ Includes data on disk and cached in memory
✦ Optionally can include data integrity checking to detect accidental or malicious data corruption

Additional Information

✦ Data encrypted using AES CBC (128 or 256)
✦ Optional integrity checking uses SHA-256
✦ Includes tools to migrate an existing DB from unencrypted to encrypted state or change the encryption key/algorithm for a given DB while transactions are flowing (no DB downtime)

Client Value Proposition:
Transparent encryption of TPF database data plus reduced cost of encryption allows clients to enable extensive encryption of TPF data.

Support shipped August 2016
(APAR PI56476)
Additional Resources

IBM Crypto Education Community
https://www.ibm.com/developerworks/community/groups/community/crypto
Contact Information

Eysha S. Powers
Enterprise Cryptography, IBM
eysha@us.ibm.com