Crypto 101: Meet Alice and Bob

Eysha S. Powers

IBM Corporation

Monday, August 10, 2015: 4:30pm – 5:30pm

Session Number 17621

#SHAREorg

Table of Contents

Introduction: **Why Encrypt?**

Chapter 1: Symmetric Encryption
- Encrypting and decrypting data

Chapter 2: Asymmetric Encryption
- Sharing symmetric keys with other parties

Chapter 3: Digital Signatures
- Proving data ownership with digital signatures

Chapter 4: IBM Crypto Hardware
- CPACF and CEX5S

Chapter 5: IBM Crypto Software
- z/OS ICSF and the Crypto Stack

Extra Credit
- Trusted Key Entry Workstation Video

Cryptography is used in a variety of places…

- **Data in flight**
  - Virtual Private Networks (VPNs)
  - SSL/TLS connections (using public/private keys and certificates and symmetric encryption)
  - Messaging infrastructures (using SSL/TLS or shared secrets)
  - WS-Security and SOA
- **Data at rest**
  - File and folder encryption – including the use of intermediate devices
  - Removable media (tape) encryption
- **Transactional environments**
  - Industry specific – finance
  - Mandates highly trust-worthy cryptography
  - Smart ID cards, ePassports…
- **For sharing user credentials between organizations – the establishment of trust**
  - Via certificate exchanges
  - Federated Identity Management
  - Credential formats such as SAML, OpenID Connect…

Encryption Protects IT Assets

The need for encryption is dictated by the need to protect IT assets such as sensitive data or to secure transactions. For certain applications such as payments and health care, the requirements stem directly from the standard bodies and the legislation.

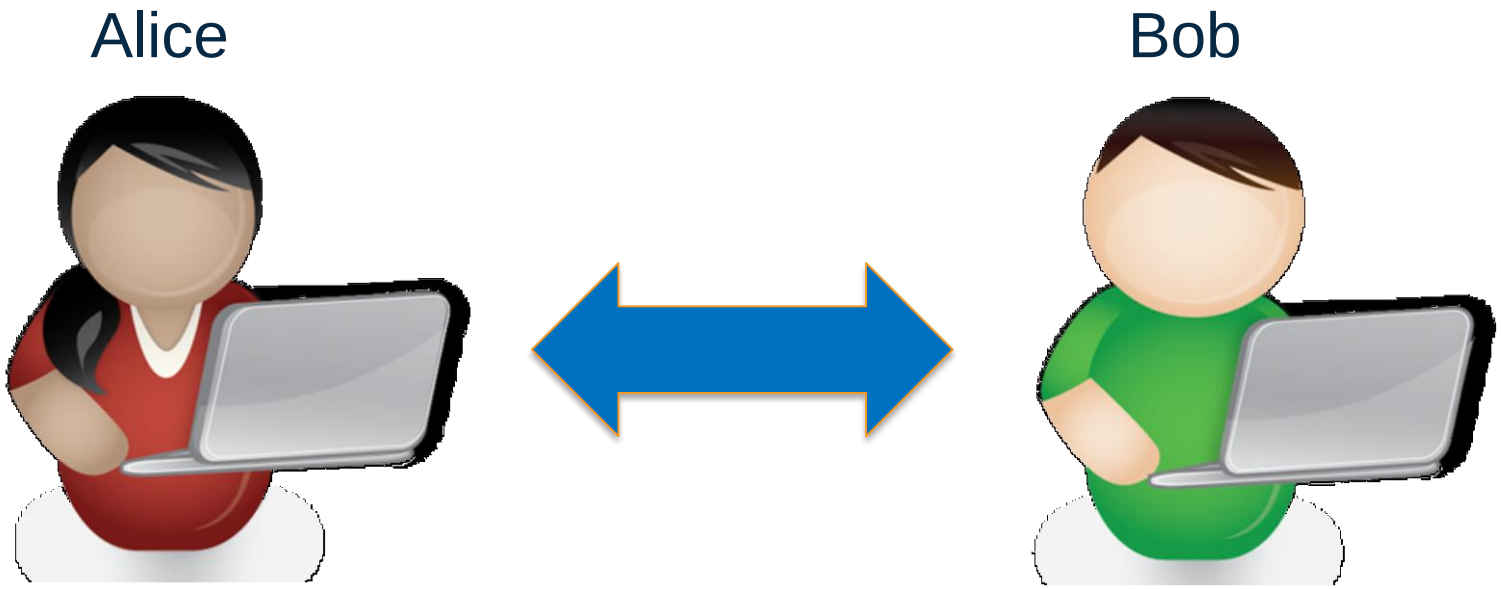
Encryption is Mandated by Industry Standards

The security of sensitive data, such as personal information or payments data, is mandated

- by the payment cards industry via the PCI-DSS and PCI-PIN standards,
- via the European Union in the Data Protection Directive 95/46/EC,
- in the HiPAA, and in local legislation.

Encryption is a necessary mechanism to help you stay compliant with these regulations.

Let's Get Started



Alice needs to send a message to Bob that only Bob should see. What can she do?

“Secret Writing”

Cryptography is the practice and study of techniques for secure communication in the presence of third parties (i.e. adversaries).

- Confidentiality – Preventing the disclosure of information to unauthorized individuals
  - **Encrypt:** Convert clear text to cipher text.
  - **Decrypt:** Convert cipher text to clear text.

- Integrity – Maintaining and assuring the accuracy and consistency of data
  - **Hash**: Translate clear text to a fixed length hash value

    Example (20-byte hash): DFCD 3454 BBEA 788A 751A 696C 24D9 7009 CA99 2D17
  - **Sign:** Hash the clear text and encrypt the hash with a private key
  - **Verify:** Hash the clear text then decrypt the sender's hash using the sender's public key and compare the hash values.

What is the key to “secret writing”?

A cryptographic key. A sequence of bytes of a specific length (i.e. key size) to be used in a cryptographic operation.

- DES = 8 bytes
- TDES = 8, 16 or 24 bytes
- AES = 16, 24 or 32 bytes

Where do the bytes come from?

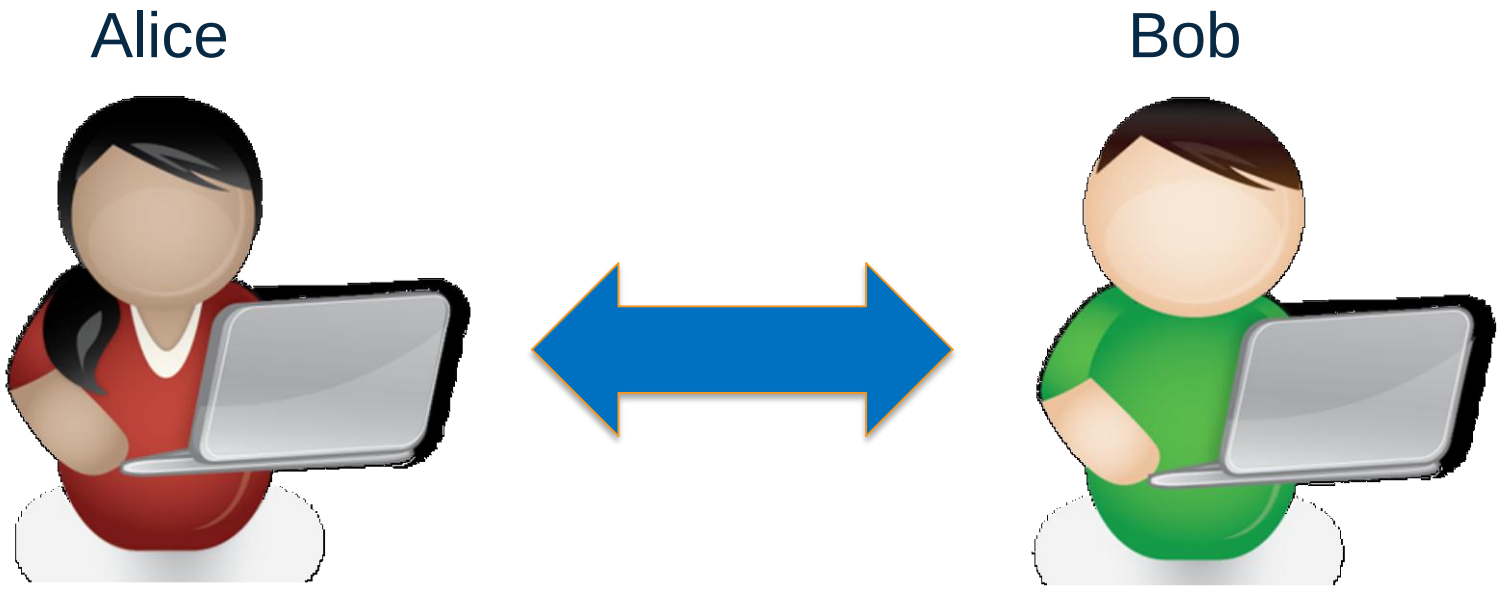
- Typically, a pseudo random number generator
- Pseudo Random number generation requires:
  - An entropy source of randomness **PLUS**
  - A deterministic mathematical algorithm **to**
  - Produce pseudo random bytes

Why does the key length matter?

- Short key lengths can be brute force attacked, especially with today’s computing speeds
  - The NIST standards body recommends symmetric keys of 16 bytes are larger.
- Long key lengths take much more time to generate
  - RSA key pairs can be 1024 – 4096 bits (128 - 512 bytes)
  - ECC key pairs offer stronger encryption but smaller key sizes

Chapter 1: Symmetric Encryption

Back to Alice and Bob…



Alice needs to send a message to Bob that only Bob should be able to see. What can she do?

Encrypt secret data with a symmetric key

Alice

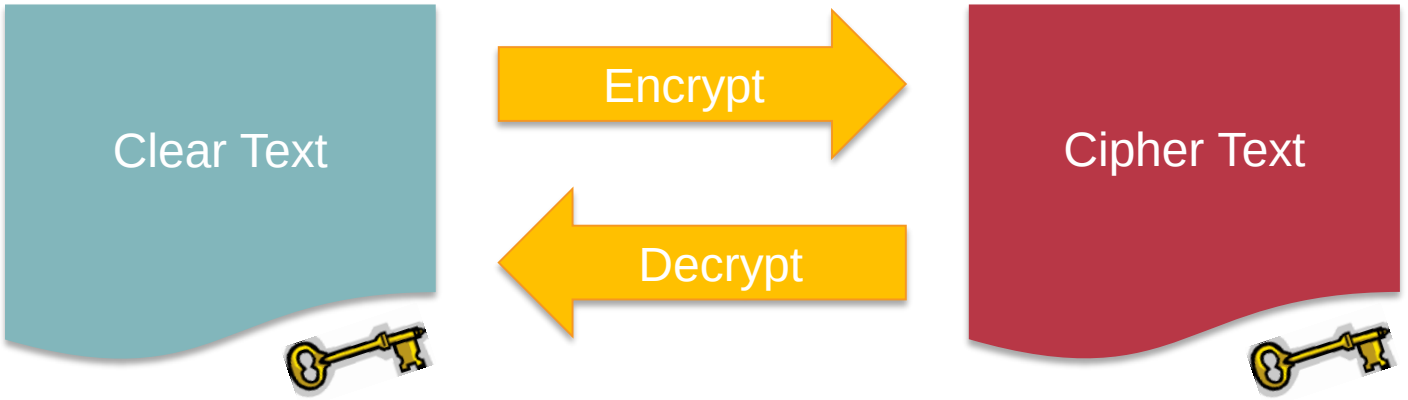
Bob

1. Generate a symmetric key
2. Share the symmetric key with Bob
3. Encrypt the message with the symmetric key
4. Send the encrypted data to Bob

1. Receive the symmetric key
2. Receive the encrypted data
3. Decrypt the message with the symmetric key

How does encryption / decryption work?

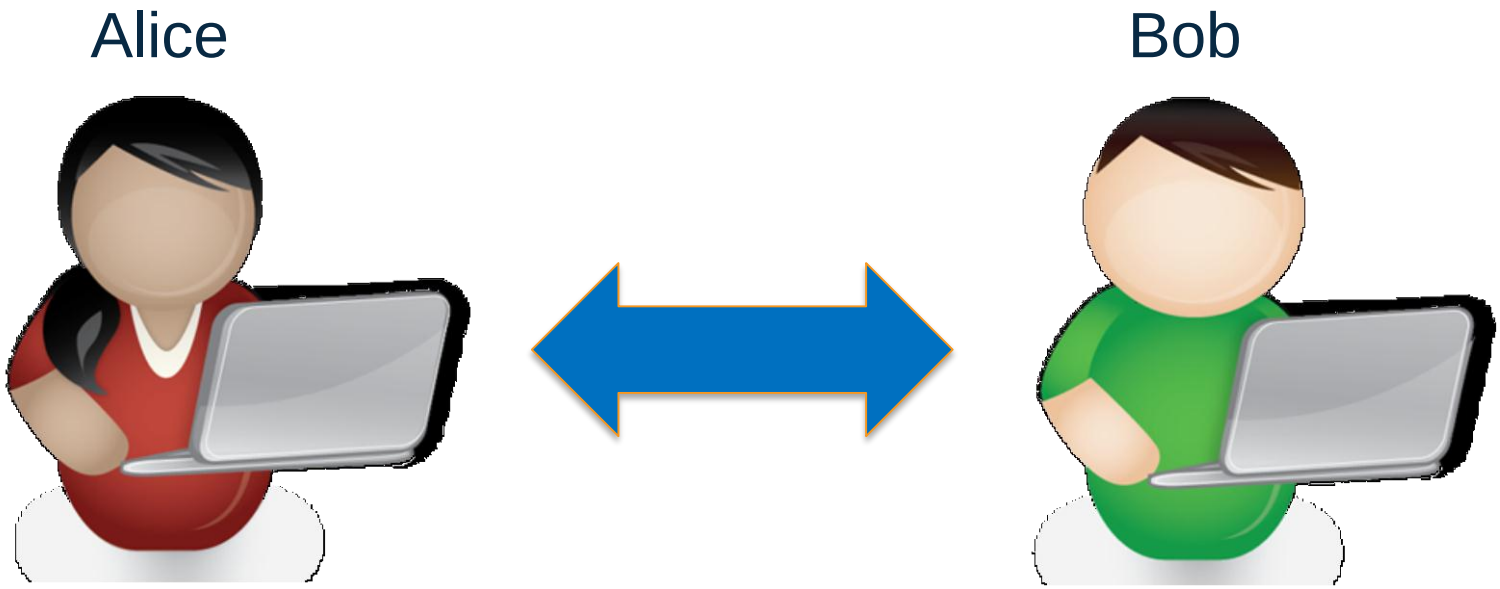
- Provide a cryptographic key and clear text to a mathematical algorithm to produce cipher text (i.e. encryption)
- Provide a cryptographic key and cipher text to a mathematical algorithm to produce clear text (i.e. decryption)



For symmetric encryption, the encryption key and decryption key are the same!

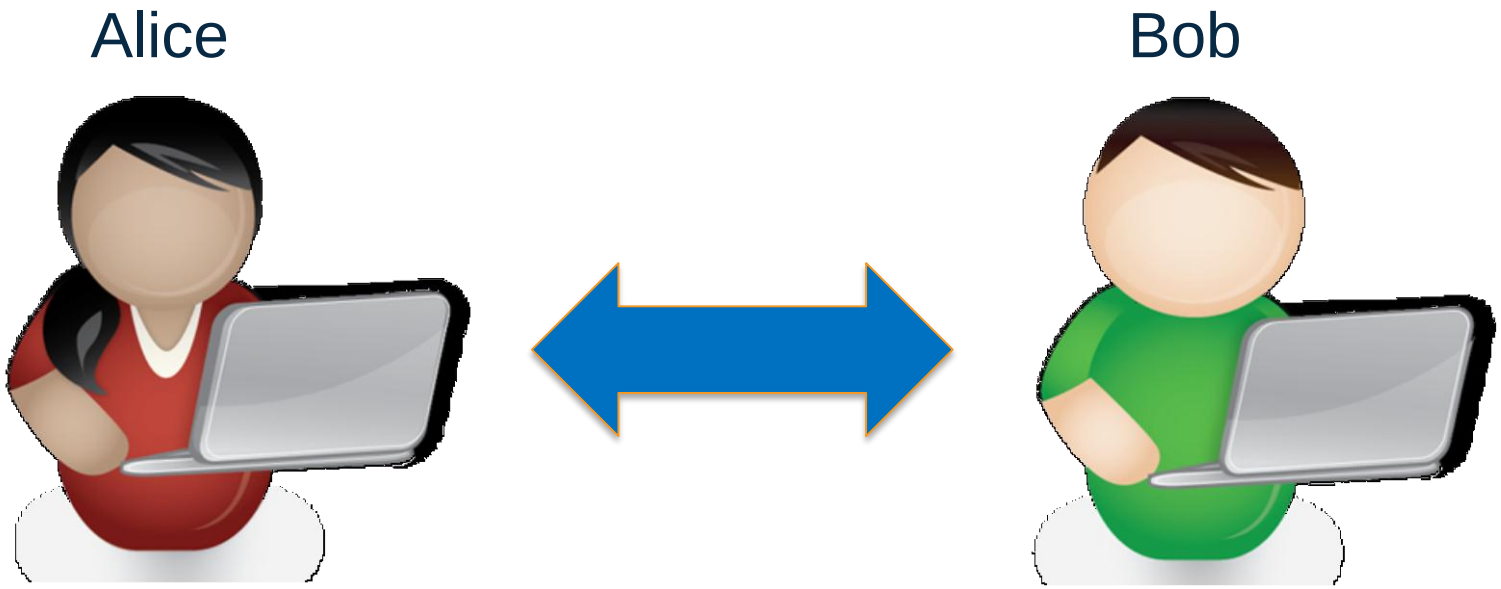
Chapter 2: Asymmetric Encryption

Back to Alice and Bob…



How does Alice send Bob the symmetric key without anyone seeing?

Encrypt secret data with asymmetric key pairs

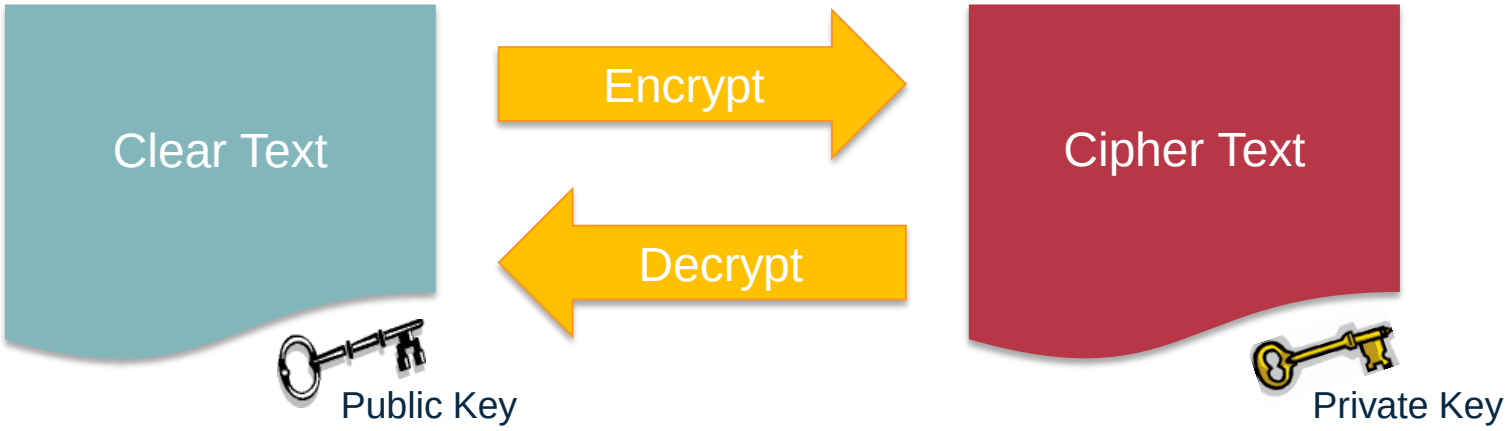


1. Generate an asymmetric key pair
   a. Public key, pubA
   b. Private key, privA
2. Publish public key, pubA
3. Encrypt the symmetric key with Bob's public key, pubB
4. Send the encrypted data to Bob

1. Generate an asymmetric key pair
   a. Public key, pubB
   b. Private key, privB
2. Publish public key, pub
3. Receive the encrypted data from Alice
4. Decrypt the data with Bob's private key, privB

How does asymmetric encryption work?

- Provide a cryptographic public key and clear text to a mathematical algorithm to produce cipher text (i.e. encryption)
- Provide a cryptographic private key and cipher text to a mathematical algorithm to produce clear text (i.e. decryption)



For asymmetric encryption, the encryption key and decryption key are computationally related but different!

Symmetric vs Asymmetric Encryption

Symmetric Encryption

- Fast to generate
- Large data sizes

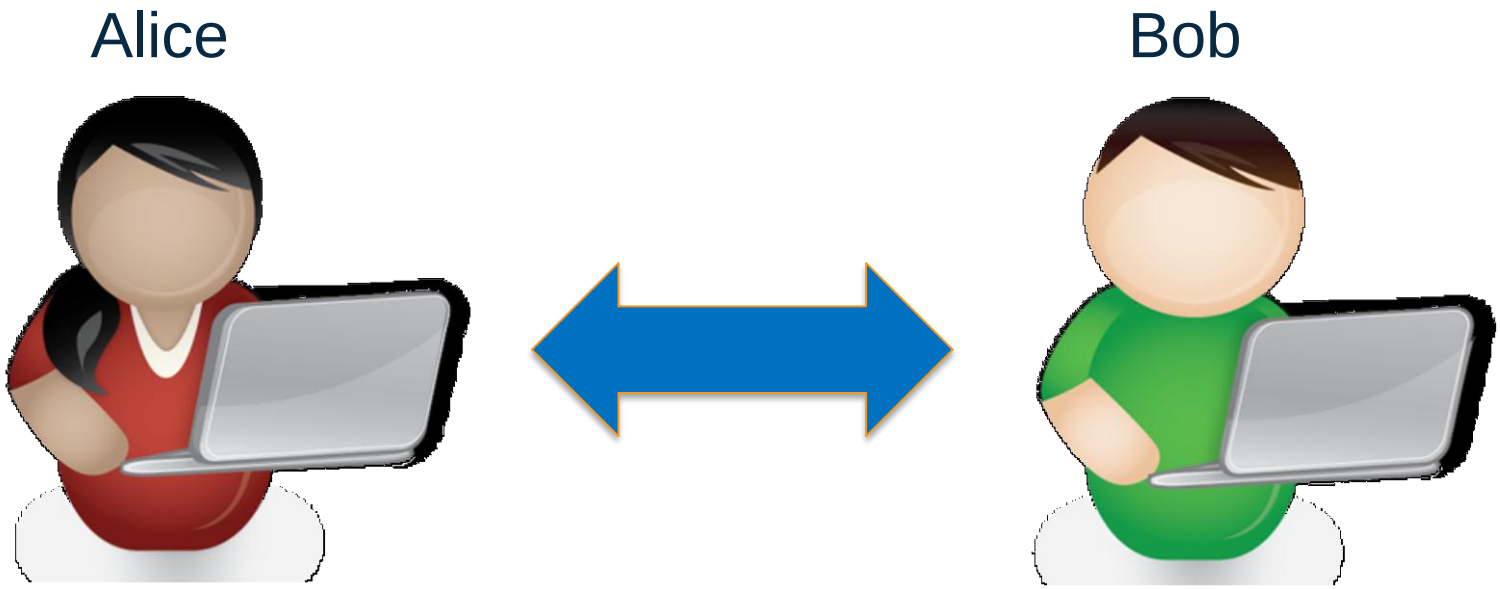
Asymmetric Encryption

- Slower to generate
- Small data sizes
- Easy distribution

Algorithms	Algorithms
DES –	– RSA
TDES –	– ECC
AES –	– DH

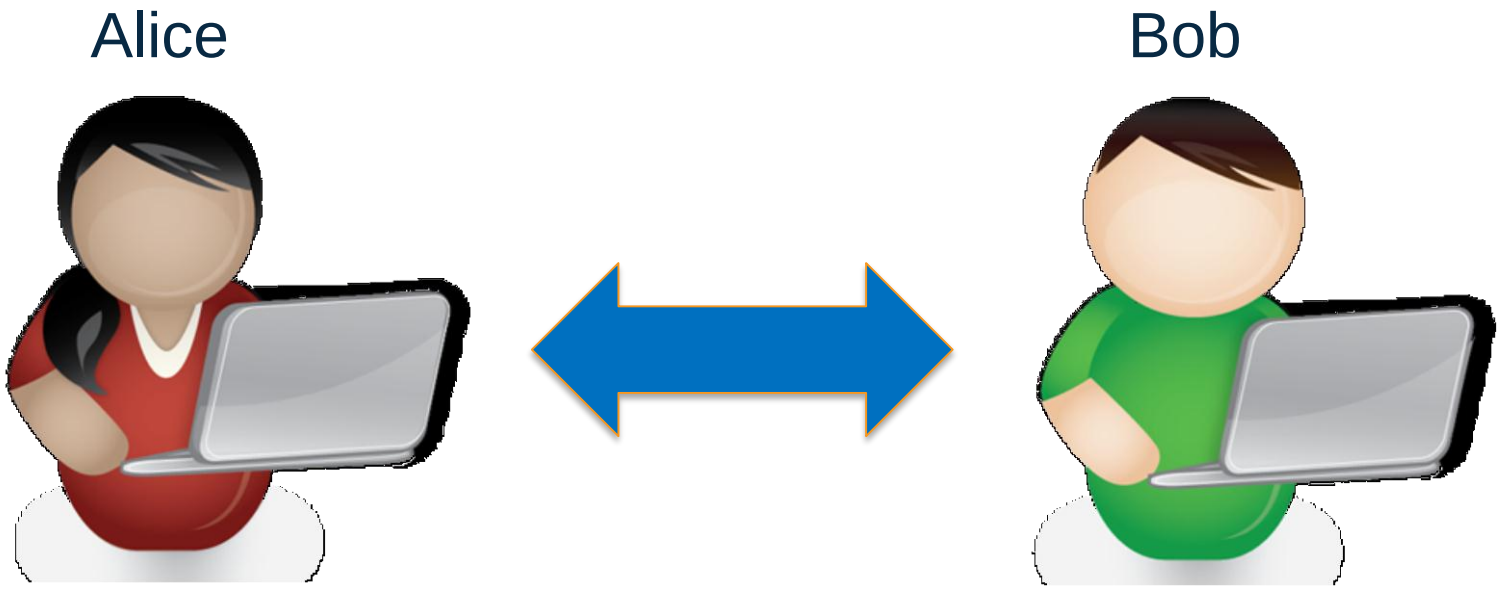
Chapter 3: Digital Signatures

Back to Alice and Bob…



Alice wants to share data with Bob that is public for anyone to see. Bob needs to be sure that the data originated from Alice and not anyone else. What can Alice do?

Sign data with asymmetric key pairs



1. Use Alice's asymmetric key pair
   a. Public key, pubA
   b. Private key, privA
2. Sign hashed data with Alice's private key, privA
3. Send the data and the signature to Bob

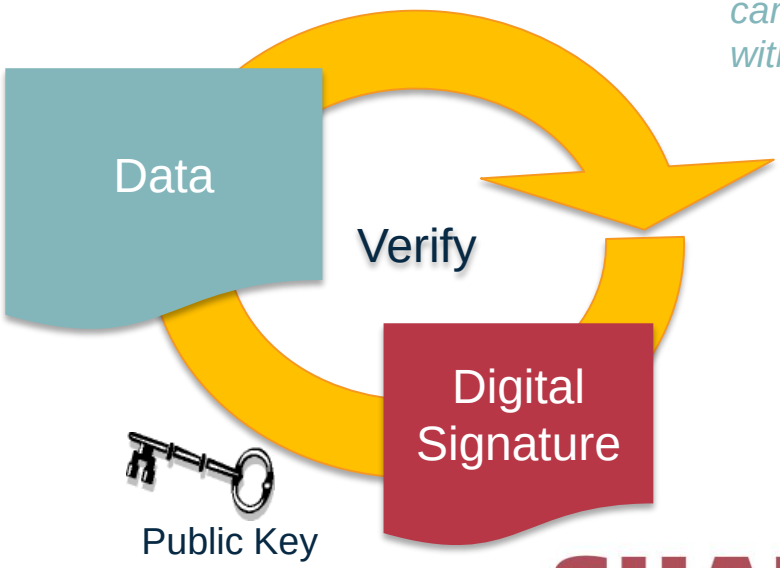
1. Receive the data and signature from Alice
2. Verify the digital signature sent by Alice using Alice's public key

How do digital signatures work?

- Provide a cryptographic private key and data to a mathematical algorithm to produce a digital signature.

Notice how the data is unchanged and how anyone can verify the data with the public key!

- Provide a cryptographic public key, data and the digital signature to a mathematical algorithm to verify the digital signature.



Chapter 4: IBM z Systems Hardware Crypto

IBM has been providing Security & Encryption Solutions for over 30 years…

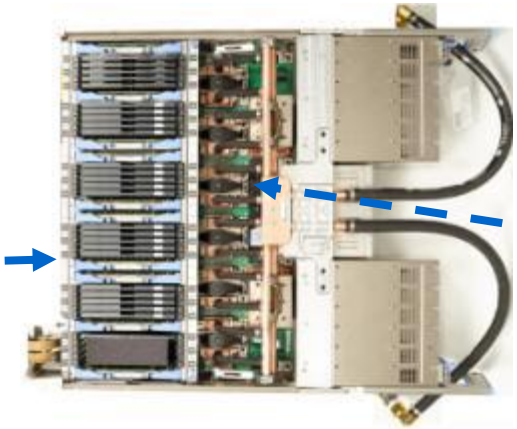
A History of Enterprise Security

- RACF: controls access to resources and applications: 1976
- Hardware Cryptography: 1977
- Key management built into operating system (ICSF): 1991
- Distributed Key Management System (DKMS) (1990's)
- 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
- License ECC Technology from Certicom: 2008
- 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
- z Systems zEC12 with Public Key Cryptography Standards – Enterprise PKCS#11

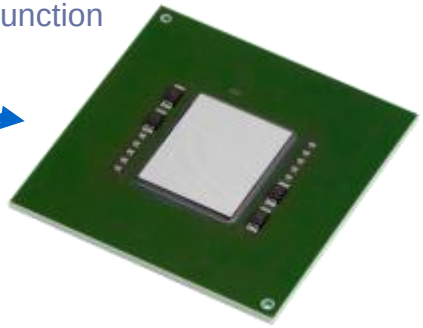
Hardware Crypto in z13

PCIe I/O drawers

CPC Drawer



PU SCM
Each PU is capable of having the CPACF function



Crypto Express5S

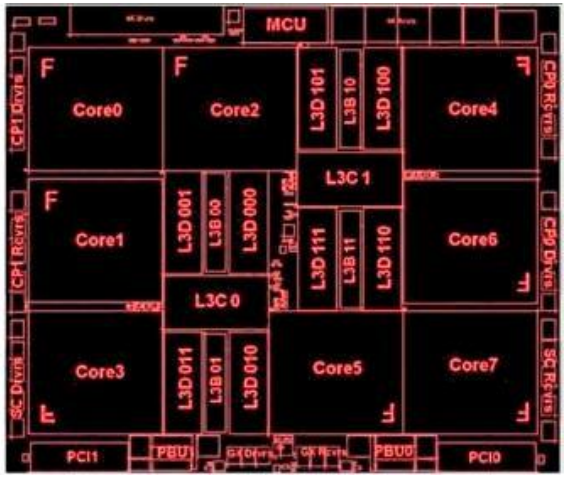
TKE required for management of Crypto Express5S and EP11

Trusted Key Entry (TKE)

Smart Cards

Smart Card Readers

CPACF - CP Assist For Cryptographic Functions



Supported Algorithms	Clear Key	Protect Key
DES, T-DES	Y	Y
AES128	Y	Y
AES192	Y	Y
AES256	Y	Y
SHA-1	Y	N/A
SHA-256	Y	N/A
SHA-384	Y	N/A
SHA-512	Y	N/A
PRNG	Y	N/A
DRNG	Y	N/A

Provides a set of symmetric cryptographic functions and hashing functions for:

- Data privacy and confidentiality
- Data integrity
- Random Number generation
- Message Authentication

Enhances the encryption/decryption performance of clear-key operations for

- SSL
- VPN
- Data storing applications

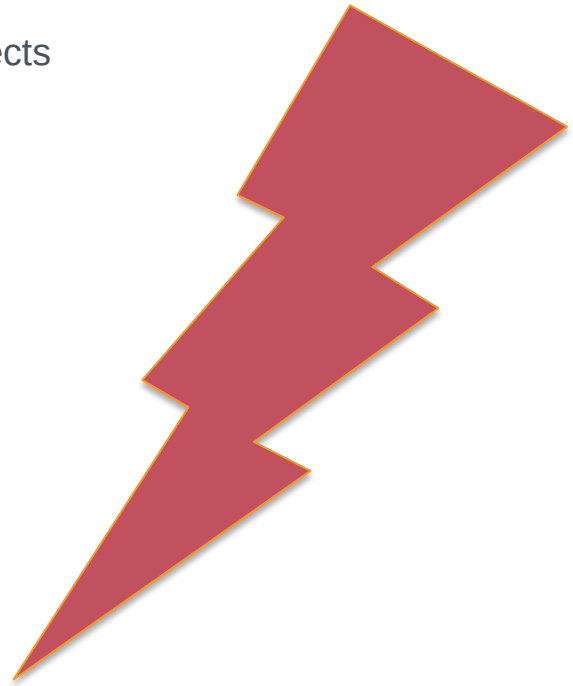
- Available on every Processor Unit defined as a CP, IFL, zAAP and zIIP
- Supported by z/OS, z/VM, z/VSE, z/TPF and Linux on z Systems
- Must be explicitly enabled, using a no-charge enablement feature (#3863),
  - SHA algorithms enabled with each server
- Protected key support for additional security of cryptographic keys
  - Crypto Express5S required in CCA mode

z13 CPACF Performance Enhancements

CP Assist for Cryptographic Function Co-processor redesigned from "ground up“

Enhanced performance over zEC12

- Does not include overhead for COP start/end and cache effects
- Enhanced performance 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 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)
- Designed for 2X performance increase over Crypto Express4S

Three configuration options for the PCIe adapter

- Only one configuration option can be chosen at any given time
- All card secrets are erased when switching to or from EP11 Coprocessor mode

Accelerator	
TKE	N/A
CPACF	NO
UDX	N/A
CDU	YES(SEG3)

Clear Key RSA operations and SSL acceleration

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(VFPE), and simplification of cryptographic key management.
- Designed to be FIPS 140-2 Level certification to meet regulations and compliance for PCI standards

CEX5S Cryptographic Units

- DES/TDES w DES/TDES MAC/CMAC
- AES, AESKW, AES GMAC, AES GCM, AES XTS mode, CMAC
- MD5, SHA-1, SHA-2 (224,256,384,512), HMAC
- VISA Format Preserving Encryption (VFPE)
- RSA (512, 1024, 2048, 4096) -> Performance improvement
- ECDSA (192, 224, 256, 384, 521 Prime/NIST)
- ECDSA (160, 192, 224, 256, 320, 384, 512 BrainPool)
- ECDH (192, 224, 256, 384, 521 Prime/NIST)
- ECDH (160, 192, 224, 256, 320, 384, 512 BrainPool)
- Montgomery Modular Math Engine
- RNG (Random Number Generator)
- PNG (Prime Number Generator) -> NEW
- Clear Key Fast Path (Symmetric and Asymmetric)

Chapter 5: IBM z Systems Software Crypto

Integrated Cryptographic Services Facility (ICSF)

ICSF works with the hardware cryptographic features and the Security Server (RACF element) to provide secure, high-speed cryptographic services in the z/OS environment.

- ICSF provides the application programming interfaces by which applications request cryptographic services.
- ICSF is the default means by which the secure cryptographic features are loaded with master key values, allowing the hardware features to be used by applications.
- ICSF callable services and programs can be used to generate, maintain, and manage keys that are used in the cryptographic functions.

ICSF uses 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

IBM Common Cryptographic Architecture (CCA) for z/OS ICSF

IBM Common Cryptographic Architecture (CCA)

- IBM proprietary cryptographic application programmers 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)
- CVV Generate / Verify
- Secure Messaging for Keys / Pins
- … And Many More!

PKCS#11 Cryptographic Token Interface Standard for z/OS ICSF

PKCS #11 Cryptographic Architecture

- Originally published by RSA Laboratories, now maintained by OASIS
  - Defines a standard API for devices that hold cryptographic information and perform cryptographic functions
  - Enterprise PKCS#11 – EP11

z/OS ICSF Naming Convention for PKCS#11

- CSFP* = PKCS#11 APIs

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

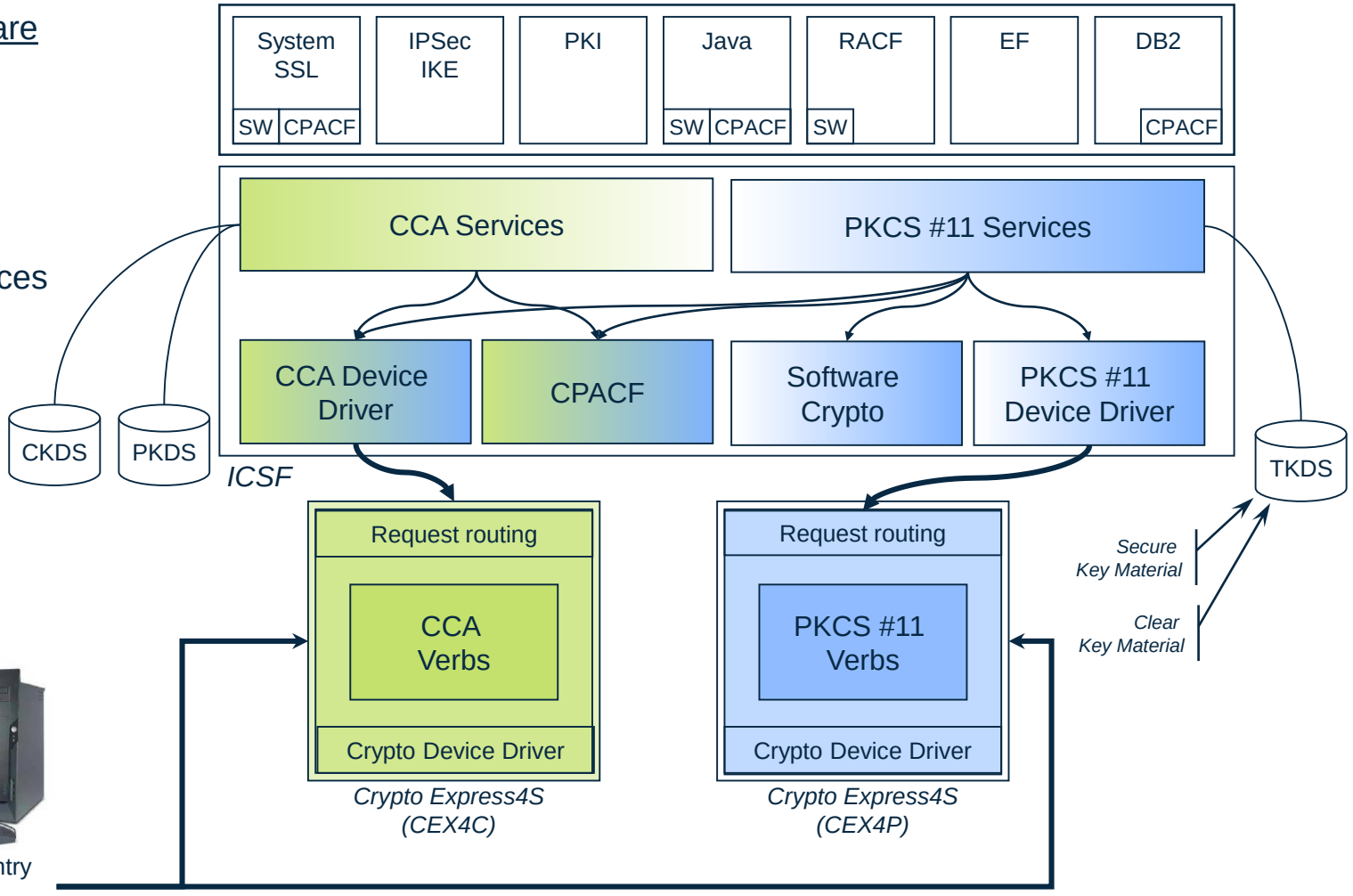
Designed for portability and
FIPS/Common Criteria certification

z/OS Crypto Stack

<u>z Systems Software</u>

z/OS

- ICSF
- RACF
- RMF
- z/OS PKI Services
- System SSL
- Java PKCS#11 Provider



<u>TKE</u>

Trusted Key Entry

CCA | PKCS11

z Systems Software Exploiters of ICSF

z/OS Software Components

- System SSL
- Java Cryptography Extension
- RACF Security
- DB2 Database
- PKI Services
- IBM Tivoli Directory Server
- Kerberos Network Authentication Service
- Websphere MQ
- Websphere Application Server
- z/OS Communications Server
- …

IBM Solutions

- IBM Infosphere Guardium
- Sterling Connect:Direct
- …

Cryptographic Keys & Certificates

IBM z Systems has several means to generate, maintain and manage keys and certificates that are used in cryptographic functions.

- **ICSF** provides callable services and utilities to generate and store keys into ICSF Key Data Sets (CKDS/PKDS/TKDS).
- **RACF** provides the RACDCERT GENCERT command to generate and store keys into the RACF database and ICSF Key Data Sets (PKDS/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 provides APIs and utilities to generate and store keys and certificates into ICSF Key Data Sets, RACF Keyrings, and Java Key Stores.

ICSF Key Data Sets

CKDS

Cryptographic Key Data Set
- CCA Symmetric Keys
- AES and DES

PKDS

PKA Key Data Set
- CCA Asymmetric Keys
- RSA and ECC

TKDS

Token Key Data Set
- PKCS#11 Keys, Certificates
- All algorithms

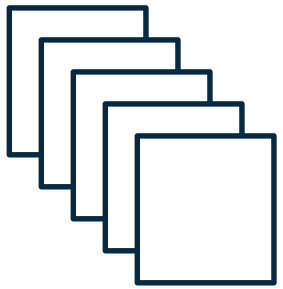
RACF Keyrings

RACF Keyrings
- Certificates
- RSA, ECC

Flat Files

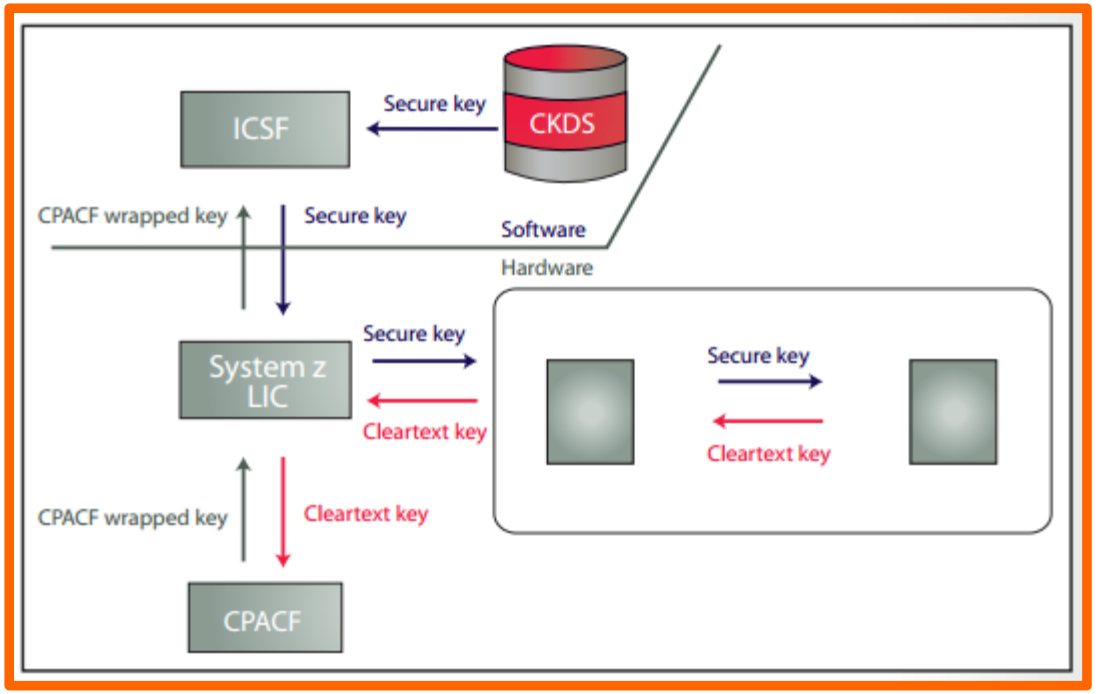
Java Key Store (ks)
•Certificates, Keys

Key Database Files (kdb)
•Certificates, Keys



Secure Keys vs Clear Keys vs Protected Keys

- **Secure Key** - provides high security because the key material is protected by the master key. Master keys are loaded within the cryptographic coprocessor and are used to wrap and unwrap secure key material within the secure boundaries of the HSM. This prevents secure key material from ever appearing in the clear.
- **Clear Key** – when performing symmetric encryption, TDES and AES, with clear keys, ICSF uses the CPACF to provide high performance. Clear Key refers to key material that is in the clear, meaning the clear key value appears within application storage and within the keystore
- Protected Key - provides a high performance and high security solution by taking advantage of the high speed CPACF while utilizing symmetric keys protected by the cryptographic coprocessor Master Key. To use a CKDS encrypted key, the ICSF segment of the CSFKEYS class general resource profile associated with the specified key label must contain SYMCPACFWRAP(YES).



What You've Learned…

Cryptography enables you to protect your sensitive data:

- Encryption and Decryption to hide data
- Random Number Generation to produce keys that are difficult to guess
- Key Distribution to send encrypted data to other parties
- Digital Signatures to prove the originator of the data
- … and more

IBM z Systems provide both hardware and software cryptography to help you protect your IT assets.

Extra Credit: TKE Workstation

Trusted Key Entry (TKE) Workstation

https://www.youtube.com/watch?v=WEG29Qq82Tc

Additional Resources

IBM Crypto Education Community

https://www-304.ibm.com/connections/communities/community/crypto

IBM System z Development Blog

https://ibm.biz/zsystems_development

Thank You!

Feel free to connect…

- Email: eysha@us.ibm.com
- Twitter: http://www.twitter.com/EyshaShirrine
- LinkedIn: http://www.linkedin.com/in/eysha