z/OS
Cryptographic Services
Integrated Cryptographic Service Facility

Overview

SA22-7519-15

Note!

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 85.

This edition applies to Version 1 Release 13 of z/OS (5694–A01) and to all subsequent releases and modifications until otherwise indicated in new editions. This edition applies to ICSF FMID HCR7790.

This edition replaces SA22-7519-14

Contents

Figures

Tables

About this information

This information supports z/OS (5694-A01) and z/OS.e (5655-G52). It contains overview and planning information for the z/OS Integrated Cryptographic Service Facility (ICSF). The z/OS Cryptographic Services includes these components:

- z/OS Integrated Cryptographic Service Facility (ICSF)
- z/OS Open Cryptographic Services Facility (OCSF)
- z/OS System Secure Socket Level Programming (SSL)
- z/OS Public Key Infrastructure Services (PKI)

ICSF is a software element of z/OS that works with hardware cryptographic features and the Security Server (RACF) to provide secure, high-speed cryptographic services in the z/OS environment. ICSF provides the application programming interfaces by which applications request the cryptographic services. The cryptographic feature is secure, high-speed hardware that performs the actual cryptographic functions.

The cryptographic hardware features available to your applications depend on the server.

ICSF features

ICSF provides support for:

- The ANSI Data Encryption Algorithm (DES) and Advanced Encryption Standard (AES) encryption and decryption
- DES key management and transport
- AES key management and transport
- Financial services including PINs, payment card industry transactions and ATMs
- Public key operations including key generation, digital signatures and wrapping symmetric keys for transport
- MAC and hash generation
- Acceleration of handshake and frame encryption for SSL
- PKCS #11 API

Who should use this information

This information is for chief information officers, information system executives, and information security professionals and auditors. Installation managers and security administrators who are responsible for planning the data security strategy for their installation will also find this information to be helpful. This publication applies to installations that have z/OS with ICSF and a hardware cryptographic feature installed.

How to use this information

The major topics are:

- Chapter 1, “Introducing cryptography and ICSF” introduces the general subject of cryptography, and describes why ICSF may be right for your installation.
- Chapter 2, “Solving your business needs with ICSF” describes how ICSF can help your business.

- Chapter 3, “Application Programming Interfaces and key management” describes the cryptographic callable services available with ICSF and the basic concepts of managing cryptographic keys.
- Chapter 4, “Using ICSF with other cryptographic products” describes how ICSF relates to other cryptographic products.
- Chapter 5, “Planning for the Integrated Cryptographic Service Facility” identifies the system facilities and system resources that ICSF requires and presents guidelines and suggestions to help you when you plan for installing, operating, and migrating ICSF.
- Appendix A, “Standards,” on page 69 provides a list of International and USA standards for the Cryptographic Coprocessor Feature, PCI Cryptographic Coprocessor, and ICSF.
- Appendix B, “Summary of callable service support by hardware configuration,” on page 71 summarizes ICSF callable services by configuration.
- Appendix C, “Accessibility,” on page 83 contains information on accessibility features in z/OS.
- “Notices” on page 85 contains notices and trademarks.

Where to find more information

This topic describes what contains ICSF information.

The ICSF library

The ICSF library includes these publications:

- *z/OS Cryptographic Services ICSF Overview*, SA22-7519

  This publication provides an introduction to ICSF, an overview of cryptography, and planning information.
- *z/OS Cryptographic Services ICSF Administrator's Guide*, SA22-7521

  See this for information on managing cryptographic keys. It describes the tasks of entering DES and PKA master keys, changing a DES master key, using the key generator utility program, and viewing the status of the cryptographic feature.
- *z/OS Cryptographic Services ICSF System Programmer's Guide*, SA22-7520

  See this for information on initialization, customization, migration, and problem diagnosis.
- *z/OS Cryptographic Services ICSF Application Programmer's Guide*, SA22-7522

  See this for information on writing application programs that use the callable services that are provided by ICSF to access cryptographic functions. These callable services can be used in high-level languages such as C, COBOL, FORTRAN, and PL/I, as well as in Assembler.
- *z/OS Cryptographic Services ICSF Messages*, SA22-7523

  See this for explanations of messages that are produced by ICSF, and for the routing and descriptor codes for those messages.
- *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*, SA23-2231

  See this for information about the PKCS #11 support provided by ICSF.
- *z/OS Cryptographic Services ICSF TKE Workstation User's Guide*, SA23-2211

  This information is available with the optional Trusted Key Entry (TKE) PCIX workstation and explains how to install and run the TKE PCIX workstation for key distribution (Version 5).

These publications contain additional ICSF information:

- *z/OS MVS System Codes*, SA22-7626

  This describes reason codes for ICSF X'18F' abend code.
- *z/OS MVS System Management Facilities (SMF)*, SA22-7630

  This describes SMF record type 82, where ICSF records events.
- *z/OS Migration*
- *z/OS Migration to the IBM System z10*
- *z/OS MVS Initialization and Tuning Guide*, SA22-7591
- *z/OS MVS Initialization and Tuning Reference*, SA22-7592
- *z/OS MVS Programming: Callable Services for High-Level Languages*, SA22-7613
- *z/OS MVS Programming: Authorized Assembler Services Guide*, SA22-7608
- *z/OS MVS Programming: Extended Addressability Guide*, SA22-7614
- *z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN*, SA22-7609
- *z/OS MVS Programming: Authorized Assembler Services Reference EDT-IXG*, SA22-7610
- *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU*, SA22-7611
- *z/OS MVS Programming: Authorized Assembler Services Reference SET-WTO*, SA22-7612

Related publications

- *IBM Encryption Facility for z/OS: User's Guide*
- *z/OS Security Server RACF Auditor's Guide*
- *z/OS Security Server RACF Command Language Reference*
- *z/OS Security Server RACF Security Administrator's Guide*
- *z/OS Security Server RACF Macros and Interfaces*
- *z/OS Security Server RACF System Programmer's Guide*
- *IBM Distributed Key Management System, Installation and Customization Guide*
- *IBM Common Cryptographic Architecture: Cyrptographic Application Programming Interface Reference*
- *IBM ES/3090 Processor Complex PR/SM Planning Guide*
- *IBM Security Architecture: Securing the Open Client/Server Distributed Enterprise*
- *VTAM Programming for LU 6.2*
- *RSA's Frequently Asked Questions About Today's Cryptography*, available on the World Wide Web. See RSA's home page at http://www.rsa.com.
- *BSAFE User's Guide*
- *BSAFE Library Reference Manual*
- *Applied Cryptography*, Second Edition, by Bruce Schneier

Information on other IBM cryptographic products

- *IBM Transaction Security System: General Information Manual and Planning Guide*
- *IBM Transaction Security System: Concepts and Programming Guide: Volume I, Access Controls and DES Cryptography*
- *IBM Transaction Security System: Basic CCA Cryptographic Services*

- *IBM Transaction Security System: Concepts and Programming Guide: Volume II, Public-Key Cryptography*

How to send your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or give us any other feedback that you might have.

Use one of the following methods to send us your comments:

1. Send an email to mhvrcfs@us.ibm.com
2. Visit the Contact z/OS web page at http://www.ibm.com/systems/z/os/zos/webqs.html
3. Mail the comments to the following address:
   IBM Corporation
   Attention: MHVRCFS Reader Comments
   Department H6MA, Building 707
   2455 South Road
   Poughkeepsie, NY 12601-5400
   U.S.A.
4. Fax the comments to us as follows:
   From the United States and Canada: 1+845+432-9405
   From all other countries: Your international access code +1+845+432-9405

Include the following information:

- Your name and address
- Your email address
- Your telephone or fax number
- The publication title and order number:
  z/OS Cryptographic Services Overview
  SA22-7519-15
- The topic and page number related to your comment
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you submit.

If you have a technical problem

Do not use the feedback methods listed above. Instead, do one of the following:

- Contact your IBM service representative
- Call IBM technical support
- Visit the IBM zSeries support web page at http://www.ibm.com/systems/z/support/

Chapter 1. Introducing cryptography and ICSF

The Internet is rapidly becoming the basis for electronic commerce. More businesses are automating their data processing operations. Online databases are becoming increasingly large and complex. Many businesses transmit sensitive data on open communication networks and store confidential data offline. Every day the potential for unauthorized persons to access sensitive data increases.

To achieve security in a distributed computing environment, a combination of elements must work together. A security policy should be based on an appraisal of the value of data and the potential threats to that data. This provides the foundation for a secure environment.

IBM has categorized these security functions according to International Organization for Standardization (ISO) standard 7498-2:

- Identification and authentication — includes the ability to identify users to the system and provide proof that they are who they claim to be.
- Access control — determines which users can access which resources.
- Data confidentiality — protects an organization's sensitive data from being disclosed to unauthorized persons.
- Data integrity — ensures that data is in its original form and that nothing has altered it.
- Security management — administers, controls, and reviews a business security policy.
- Nonrepudiation — assures that the appropriate individual sent the message.

Only cryptographic services can provide the data confidentiality and the identity authentication that is required to protect business commerce on the Internet.

What is cryptography?

Cryptography includes a set of techniques for scrambling or disguising data. The scrambled data is available only to someone who can restore the data to its original form. The purpose is to make data unintelligible to unauthorized persons, but readily decipherable to authorized persons. Cryptography deals with several processes:

- **Enciphering** is converting *plaintext*, which is intelligible, into *ciphertext*, which is not intelligible. Enciphering is also called encrypting.
- **Deciphering** is converting ciphertext back into plaintext. Deciphering is also called decrypting.
- **Hashing** involves using a one-way calculation to condense a long message into a compact bit string, or message digest.
- **Generating and verifying digital signatures** involves encrypting a message digest with a private key to create the electronic equivalent of a handwritten signature. Both a handwritten signature and a digital signature verify the identity of the signer and cannot be forged.

  Digital signatures also serve to ensure that nothing has altered the signed document since it was signed.

The growth of distributed systems and the increasing use of the Internet have resulted in the need for increased data security. Cryptography provides a strong, economical basis for keeping data confidential and for verifying data integrity.

Cryptography is already playing a critical and expanding role in electronic commerce and electronic mail services. Emerging markets that require secure data transmission and the authentication of the sender are already relying on cryptography.

The basic elements of a cryptographic system

Most practical cryptographic systems combine two elements:

- A process or algorithm which is a set of rules that specify the mathematical steps needed to encipher or decipher data.
- A cryptographic key (a string of numbers or characters), or keys. The algorithm uses the key to select one relationship between plaintext and ciphertext out of the many possible relationships the algorithm provides. The selected relationship determines the composition of the algorithm's result.

ICSF supports two main types of cryptographic processes:

- Symmetric, or secret key, algorithms, in which the same key value is used in both the encryption and decryption calculations.
- Asymmetric, or public key, algorithms, in which a different key is used in the decryption calculation than was used in the encryption calculation.

Secret key cryptography

Secret key cryptography uses a conventional algorithm such as the Data Encryption Standard (DES) algorithm or the Advanced Encryption Standard (AES) algorithm that are supported by ICSF. Another term for secret key cryptography is symmetric cryptography. To have intelligent cryptographic communications between two parties who are using a conventional algorithm, this criteria must be satisfied:

- Both parties must use the same cryptographic algorithm.
- The cryptographic key that the sending party uses to encipher the data must be available to the receiving party to decipher the data.

Figure 1 on page 3 is a simplified illustration of the cryptographic components that are needed to encipher and decipher data in a secret key cryptographic system. In this system, Tom and Linda have established a secure communications channel by sharing a secret key. Tom enciphers the plaintext by using the algorithm and the secret key before sending it to Linda. When she receives the ciphertext, Linda deciphers it using the same algorithm and the same secret key. In a secret key system, it is critically important to maintain the secrecy of the shared key.

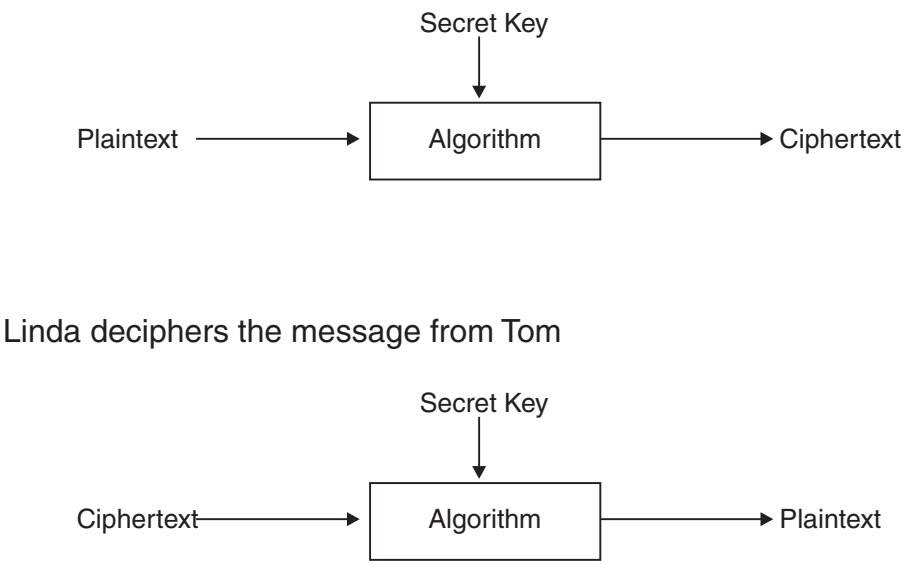

Figure 1. Enciphering and Deciphering Data in a Secret Key System

Public key cryptography

Each party in a public key cryptography system has a pair of keys. One key is public and is published, and the other key is private. Another term for public key cryptography is asymmetric cryptography because the public key and private key are not identical. The sending party looks up the receiving party's public key and uses it to encipher the data. The receiving party then uses its private key to decipher the data. In a public key system, it is critically important to maintain the secrecy of the private key.

Public key cryptography requires complex mathematical calculations. For this reason, these types of systems are not used for enciphering messages or large amounts of data. They are, however, used to encipher and decipher symmetric keys that are transported between two systems.

Public key cryptography systems are often used to generate and verify digital signatures on electronic documents. The sender uses his or her private key to generate the digital signature. The receiver then uses the sender's public key to verify the identity of the sender. On the emerging information highway, the digital signature replaces the handwritten signature as a legal proof of authenticity. Digital signatures are the principal mechanism in any system of nonrepudiation.

Figure 2 on page 4 shows an example of a nonrepudiation system that uses digital signatures. Linda sends her broker Tom an electronic order to buy 100 shares of IBM stock. The electronic transmission application on Linda's system attaches Linda's digital signature to the order before sending the order to Tom. Linda's digital signature provides Tom with proof that Linda sent the order. When Tom receives the purchase order, an acknowledgment of his receipt, including his own digital signature, is returned to Linda. This receipt serves as proof that Tom received the order. Nonrepudiation is critical for the security of electronic data interchange (EDI).

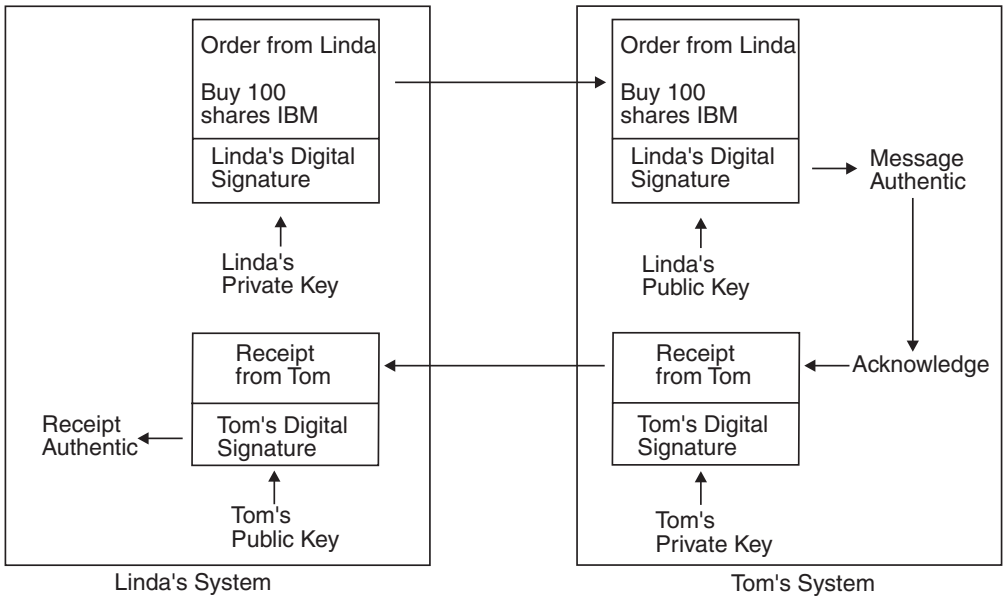

Figure 2. An Example of Nonrepudiation Using Digital Signatures

How does ICSF support cryptography?

ICSF supports IBM's Common Cryptographic Architecture (CCA). The CCA is based on the ANSI Data Encryption Algorithm (DEA) and the Advanced Encryption Standard (AES). DEA is also known as the U.S. National Institute of Science and Technology Data Encryption Standard (DES) algorithm. In these secret key cryptography systems, two parties share secret keys that are used to protect data and keys that are exchanged on the network. Sharing secret keys establishes a secure communications channel. The only way to protect the security of the data in a shared secret key cryptographic system is to protect the secrecy of the secret key.

ICSF supports triple DES encryption for data privacy. Triple DES uses three, single-length keys to encipher and decipher the data. This results in a stronger form of cryptography than that available with single DES encipherment.

ICSF supports the Advanced Encryption Standard (AES). Data can be encrypted and decrypted using 128-bit, 192-bit, and 256-bit secure and clear keys. The availability of this support is the same as triple-DES.

For public key cryptography, ICSF supports the Rivest-Shamir-Adelman (RSA) algorithm [1], the NIST Digital Signature Standard (DSS) algorithm, and the Elliptic Curve Digital Signature Algorithm (ECDSA). RSA, DSS, and ECDSA are the most widely used public key encryption algorithms. With public key encryption, each party establishes a pair of cryptographic keys, which includes a public key and a private key. Both parties publish their public keys in a reliable information source, and maintain their private keys in secure storage.

1. Invented in 1977 by Ron Rivest, Adi Shamir, and Leonard Adelman

How does ICSF extend the uses of cryptography?

In addition to the encryption and decryption of data, ICSF provides application programs with a callable interface to perform these tasks:

- Generate, install, and distribute DES cryptographic keys securely using both public and secret key cryptographic methods
- Generate, install, and distribute AES cryptographic keys securely using public key cryptographic methods
- Generate, verify, and translate personal identification numbers (PINs)
- Ensure the integrity of data by using message authentication codes (MACs), hashing algorithms, digital signatures, or the VISA Card Verification Value/MasterCard Card Verification Code
- Develop Secure Electronic Transaction (SET) applications at the merchant and acquirer payment gateway
- PKA-encrypt and PKA-decrypt symmetric key data that Secure Sockets Layer (SSL) applications can use to generate session keys
- Develop EMV ICC applications using CSNBDKG, CSNBSPN, CSNBSKY, and CSNBPCU callable services
- Provide enhanced key management for Crypto Assist instructions
- Provide remote key loading for automated teller machines (ATMs) from a central administrative site using DES keys
- Support the EMV2000 key generation algorithm
- Enables customers to write applications implementing the Diffie-Hellman key agreement protocol using the PKA encrypt callable service (CSNDPKE)
- Provide an application programming interface (API) for applications to store objects and perform cryptographic functions using PKCS #11

Key generation and distribution

With ICSF callable services, you can generate a variety of cryptographic keys for use on your system or distribution to other systems. You can develop key distribution protocols by using both secret key and public key cryptographic methods. With a secret key distribution system, you must first share a secret key with the system to which you intend to distribute keys. This is a major drawback with secret key distribution systems. With public key cryptography, however, you encrypt the keys you are distributing under the receiver's public key. The receiver decrypts the keys by using the receiving system's private key. Public key encryption provides methods for key distribution and authentication.

Personal Identification Numbers (PINs)

Many people are familiar with PINs, which enable them to use an automated teller machine (ATM). Financial networks use PINs primarily to authenticate users. Typically, the financial institution assigns a PIN. The user enters the PIN at automated teller machines (ATMs) to gain access to his or her accounts. It is extremely important to keep the PIN private, so that no one other than the account owner can use it.

ICSF enables your applications to generate PINs, to verify supplied PINs, to translate PINs from one format to another, and to store and transmit PINs in encrypted PIN blocks.

Message Authentication Codes (MACs)

MACs are used to authenticate and verify data that is transmitted over a network, stored on the system, or stored outside the system (for example, on removable media such as tape). The MAC is generated by using the data itself and a symmetric key. The MAC is sent or stored with the data. The MAC is verified when the data is received or retrieved from storage. The MAC verification process uses the data and the symmetric key.

MACs give you these benefits:

- You can validate the authenticity of data that is transmitted over a network. You can also ensure that nothing has altered the data during transmission. For example, an active eavesdropper might tap into a transmission line, and either interject bogus messages or alter sensitive data that is being transmitted. Since the sender and the receiver share a secret key, the receiver can use a callable service to calculate a MAC on the received message. The application then compares the MAC it calculates to the MAC that was transmitted with the message. The message is accepted as genuine and unaltered only if the two MACs are identical.
- Similarly, you can store a MAC with data on tape or DASD. Then, when the system retrieves the data, an application can generate a MAC and compare it with the original MAC to detect alterations.
- In either data transmission or storage, you can use MACs in an anti-virus campaign. MACs help ensure that no unauthorized executable code has been inserted into your system.

Hashing algorithms

The use of a hashing algorithm is another means of verifying that data has not been altered during transmission or storage. A hash, or message digest, is calculated with a public, one-way function, rather than with a secret key like a MAC. A hash, therefore, cannot be used to verify the authenticity of a message. Hashes are used in situations where it is impractical to share a secret key. For example, you can use a hash as part of a software delivery process to uncover deliberate or inadvertent modifications to software.

The originator of the data calculates the hash using the data itself and the hashing algorithm. The originator then ensures that the hash is transmitted with integrity to the intended receiver of the data. One way to ensure this is to publish the hash in a reliable source of public information. When the receiver gets the data, an application can generate a hash and compare it to the original one. If the two are equal, the data can be accepted as genuine; if they differ, the data is assumed to be bogus.

You can use the ICSF hashing algorithms to generate modification detection codes (MDCs), support the Public Key Cryptographic Standard (PKCS), and create hashes for digital signatures.

Digital signatures

The RSA, DSS, and ECC public key cryptography systems authenticate messages and their senders through the use of *digital signatures.* A digital signature on an electronically distributed document is similar to a handwritten signature on a paper document. It is not easy to forge either type of signature. Both types of signatures authenticate that the signing party either agreed to, or generated and/ agreed to, the signed document.

The originator of the data uses a hash of the data and the originator's private key to create the digital signature. The digital signature is then attached to the message. The receiver uses the originator's public key and the signed message to verify that the message was signed by the originator.

Figure 3 is an example of using digital signatures. The sender uses a hash of the message and the private key to create the digital signature and attach it to the message before sending it to the receiving system. The receiver uses the sender's public key to regenerate the hash value from the digital signature and compares this hash value to a hash calculated on the received message. If the two hash values match, the message is considered to be authentic.

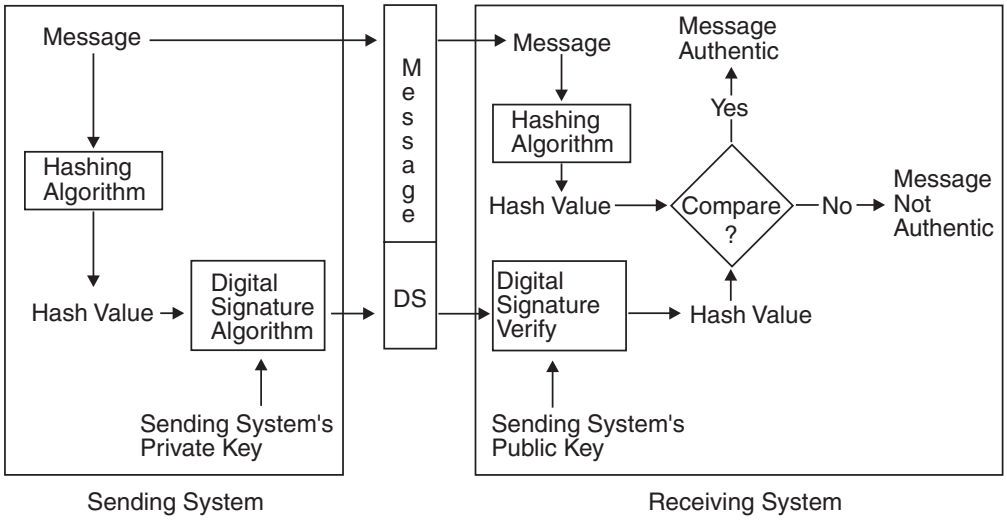

Figure 3. Creating and Verifying Digital Signatures in a Public Key System

Card-verification values

The Visa International Service Association (VISA) and MasterCard International, Incorporated have specified a cryptographic method to calculate a card verification value or code. This value relates to the personal account number (PAN), the card expiration date, and the service code. ICSF provides callable services to generate and verify the VISA card-verification value (CVV) and the MasterCard card-verification code (CVC) by the track-2 method. ICSF supports the generation and validation of American Express card security codes (CSC) and Diner's Club CVV.

Translation of data and PINs in networks

Increasingly data is being transmitted across networks in which, for various reasons, the keys that are used on one network cannot be used on another network. Encrypted data and PINs that are transmitted across these boundaries must be “translated” securely from encryption under one key to encryption under another key. For example, a traveler visiting a foreign city might wish to use an ATM to access an account at home. The PIN that is entered at the ATM might need to be encrypted there and sent over one or more financial networks to the traveler's home bank. The home bank must verify the PIN before the ATM in the foreign city allows access. On intermediate systems (between networks), applications can use the Encrypted PIN translate callable service to reencrypt a PIN block from one key to another. These applications can use ICSF to ensure that PINs never appear in the clear and that the keys for encrypting the PIN are isolated on their own networks.

SET Secure Electronic Transaction

The SET Secure Electronic Transaction standard is a global industry specification that was developed jointly by Visa International, MasterCard, and other companies. The SET protocol uses digital certificates to protect credit card transactions that are conducted over the Internet. The SET standard is a major step toward securing Internet transactions, paving the way for more merchants, financial institutions, and consumers to participate in electronic commerce.

ICSF provides callable services that support the development of SET applications that run at the merchant and acquirer payment gateway.

Secure Sockets Layer (SSL)

Secure Sockets Layer (SSL) is an industry-standard protocol that the Netscape Development Corporation designed to provide a data security layer between application protocols and TCP/IP. The SSL security protocol is widely deployed in applications on both the Internet and private intranets. SSL defines methods for data encryption, server authentication, message integrity, and client authentication for a TCP/IP connection. SSL uses public key and symmetric techniques to protect information.

SSL requires the decryption of a 48-byte SSL seed and the manipulation of this seed in the clear to produce symmetric session keys. The Common Cryptographic Architecture (CCA), however, does not permit even privacy session keys to appear in the clear in host storage. The ICSF SSL support services permit the RSA encryption and decryption of any PKCS 1.2-formatted symmetric key data. The PKA encrypt callable service CSNDPKE encrypts a supplied clear key under an RSA public key. Using the PKA decrypt callable service CSNDPKD makes it possible to unwrap the RSA-encrypted SSL seed and return it to the application in the clear. The application can then use this clear key to generate session encryption keys.

EMV integrated circuit card specifications

EMV (Europay, MasterCard and VISA) have worked together in the creation of one common standard for retail terminals accepting chip cards. Chip cards are also called stored value cards or smart cards. An algorithm or formula is stored in the chip. Chip card transactions are PIN-based for maximum security.

With ICSF, you can develop EMV ICC integrated circuit card applications using diversified key generate (CSNBDKG), secure messaging for PINs (CSNBSPN) and secure messaging for keys (CSNBSKY) services. ICSF supports the PIN change algorithms specified in the VISA Integrated Circuit Card Specification. PIN block/change (CSNBPCU), provides this support. Additionally, the diversified key generate service (CSNBDKG) supports the EMV2000 key generation algorithm.

ATM remote key loading

The process of remote key loading is loading DES keys to automated teller machines (ATMs) from a central administrative site. Because a new ATM has none of the bank's keys installed, getting the first key securely loaded is currently done manually by loading the first key-encrypting key (KEK) in multiple cleartext key parts. ANSI X9.24-2 defines the acceptable methods of doing this using public key cryptographic techniques, which will allow banks to load the initial KEKs without having to send anyone to the ATMs. This method is quicker, more reliable and much less expensive.

Once an ATM is in operation, the bank can install new keys as needed by sending them enciphered under a KEK it installed at an earlier time. Cryptographic architecture in the ATMs is not Common Cryptographic Architecture (CCA) and it is difficult to export CCA keys in a form understood by the ATM. Remote key loading makes it easier to export keys to non-CCA systems without compromising security.

Public Key Cryptography Standard #11 (PKCS #11)

PKCS #11 specifies an application programming interface (API) to devices (virtual or real) which hold cryptographic information and perform cryptographic functions. Applications written in C can code to the PKCS #11 cryptographic API and on the z/OS platform ICSF will be invoked in order to manage PKCS #11 tokens and objects and to perform cryptographic functions.

PKCS #11 supports Java Security's use of the PKCS #11 API and allows Java applications, RACF and SSL to replace their individual key stores with a single repository for keys managed by ICSF.

For more information about the specific PKCS #11 APIs see *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*.

Chapter 2. Solving your business needs with ICSF

As more businesses automate their operations and start conducting electronic commerce over the Internet, the increased use of workstations and automated teller machines generates high transaction loads. Attacks on security are becoming more sophisticated. Criminals can gain tremendous payoffs from wiretapping and theft of data from storage.

Electronic commerce, electronic funds transfer (EFT), and electronic data interchange (EDI) applications can use ICSF callable services to secure Internet transactions. These applications can make use of cryptography to protect funds transfers, purchase orders, letters of intent, contracts, credit card information, and other sensitive data from the risks of theft, fraud, or sabotage. A business can also decrease the amount of sensitive material that is exchanged by couriers. This allows a business to provide better service, become more competitive, and potentially reduce its expenses.

ICSF provides a high level of security and integrity for your data. It can help you meet many of the current needs and the future needs of your business by solving many of the information system security problems you face. This topic explains how you can use ICSF for data security, key exchange, and personal authentication.

Keeping your data private

ICSF cryptographic functions are specifically designed for high security. In addition, ICSF also provides these security precautions:

- ICSF uses the DES and AES algorithms, which are widely regarded as highly secure, to encipher and decipher data.
- The master keys are stored in highly secure hardware.
- DES keys, AES keys and PKA private keys may be encrypted under a master key for protection.
- You can use cryptographic keys only for their intended function. For example, a program that uses a key to verify a MAC cannot use the same key to generate MACs.
- You can use IBM Resource Access Control Facility (RACF) to control access to specific ICSF callable services, to specific keys that are stored in a CKDS, PKDS or TKDS or to both. RACF can also be used to protect the use of tokens passed in when calling a service using the Key Store Policy.
- With the optional Trusted Key Entry (TKE) workstation, you can create a logical secure channel. You can then use this channel to distribute master keys and operational keys to remote systems. The TKE workstation is particularly suited to the distributed computing environment that requires remote key management of one or more systems. For added security, you can require that multiple security officers perform critical operations or you can implement TKE smart card support.

Transporting data securely across a network

You may need to protect data that is sent between two applications when the data must pass through one or more intermediate systems.

In a DES cryptographic system, if the two applications cannot share a key, you must set up an application on one or more of the intermediate systems to translate

the ciphertext from encryption under the sending system's key. Translation re-encrypts the ciphertext under a new key for which the receiving system has a complementary key.

An application can use the ICSF ciphertext translate callable service to do this. ICSF prevents the recovery of plaintext on intermediate systems, because you cannot decrypt the data with the same key that is used to translate the ciphertext on the intermediate system. Figure 4 illustrates the use of the ciphertext translate callable service.

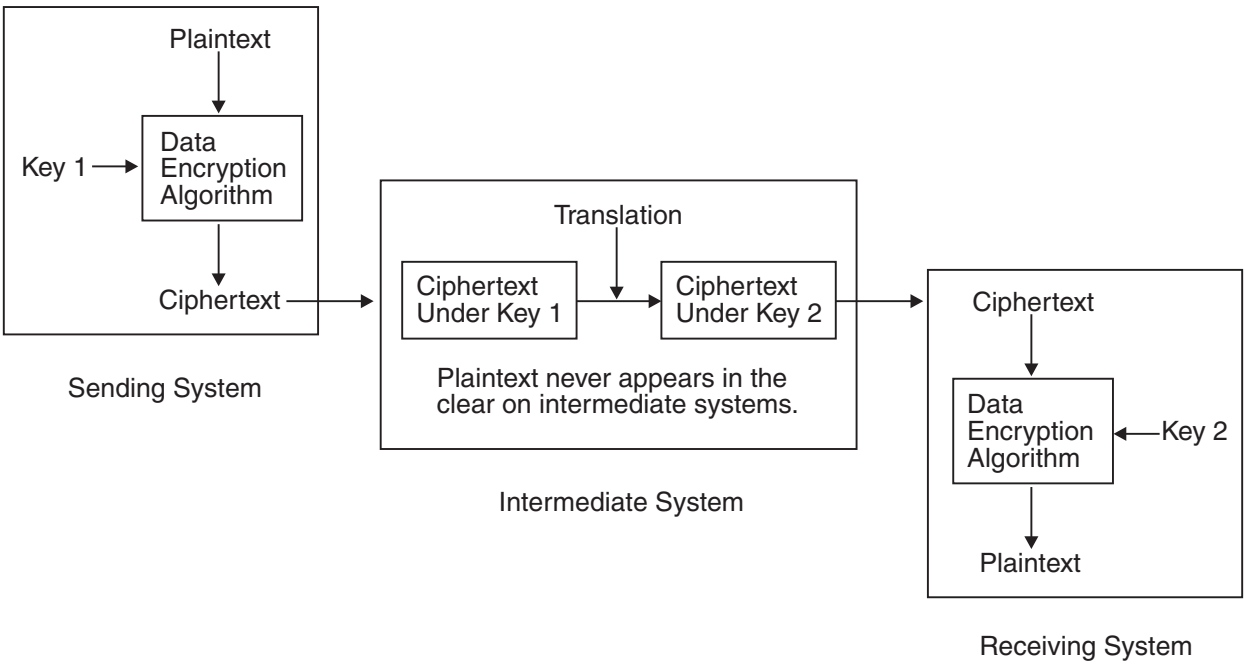

Figure 4. DES Encrypted Data Protected When Sent on Intermediate Systems

In a PKA cryptographic system, you can develop an application that does not require translation of ciphertext by the intermediate systems. The sender enciphers the message by using a DES or AES data-encrypting key. The sender then uses the receiver's PKA public key to encipher the DES or AES data-encrypting key. The intermediate system merely transfers the ciphertext and the enciphered key to the receiving system. The intermediate system does not have the receiver's PKA private key and, therefore, cannot decipher the enciphered data-encrypting key. Without the deciphered data-encrypting key, the intermediate system cannot decipher the message. The receiving system uses its PKA private key to decipher the DES or AES data-encrypting key, which it then uses to decipher the message Figure 5 on page 13.

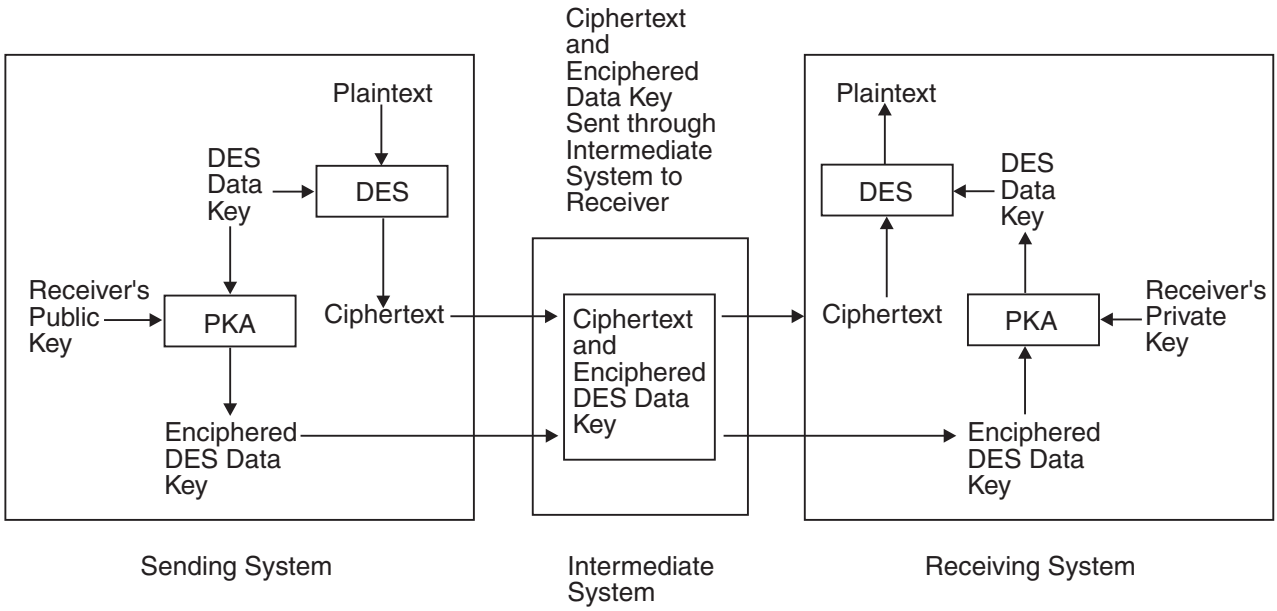

Figure 5. PKA Encrypted Data Protected When Sent on Intermediate Systems

Supporting the Internet Secure Sockets Layer protocol

The Secure Sockets Layer (SSL) provides a data security layer between the network layer and various internet transfer protocol applications. For example, SSL can provide a secure session between the transmission control protocol/internet protocol (TCP/IP) network layer and the hypertext transfer protocol (HTTP) or file transfer protocol (FTP) application. SSL provides data encryption, message integrity, and server authentication for TCP/IP connections between clients and servers. SSL ensures that credit card numbers and other sensitive information can be sent over the Internet without fear of interception.

To begin a secure session, the server and client exchange a handshake. In this digital handshake, the client and server are authenticated and also agree on the SSL version, data compression method, and cryptographic algorithm they will use when exchanging data. They also exchange an RSA-encrypted seed key that SSL manipulates to create symmetric session keys that are used to encrypt the data that the client and server exchange. The ICSF PKA encrypt and PKA decrypt callable services provide a secure method for SSL applications to exchange this seed key.

You can exploit PCI cryptographic accelerators, Crypto Express2 accelerators, and Crypto Express3 accelerators without entering master keys if SSL uses clear keys. This enhances performance.

Transacting commerce on the Internet

The Internet is rapidly becoming a major arena of commerce. For electronic commerce to grow to its full potential, however, we need to resolve several barriers to buying and selling over the Internet. Consumers are reluctant to send their bank card data over the Internet without assurances that this information is secure. Merchants need to be able to determine the clear identities of their online customers. The SET Secure Electronic Transaction protocol can help to break down

these major barriers to electronic commerce. MasterCard and Visa, with the assistance of IBM and a number of technology industry partners, cooperatively developed the SET protocol.

SET is an industry-wide, open standard for online credit card transactions. The SET protocol addresses the transaction payment phase of a transaction from the individual, to the merchant, to the acquirer (the merchant's current credit card processor). The SET protocol ensures the privacy and integrity of real time bank card payments over the Internet. In addition, with SET in place, everyone in the payment process knows the identity of everyone else. The core protocol of SET is the use of digital certificates to fully authenticate the card holder, the merchant, and the acquirer. Each participant in the payment transaction holds a certificate that validates his or her identity. Public key cryptography makes it possible to exchange, check, and validate these digital certificates for every Internet transaction. The mechanics of this operation are transparent to the application.

Under the SET protocol, a digital certificate which identifies the card-holder to the merchant must accompany every online purchase. The buyer's digital certificate serves as an electronic representation of the buyer's bank card but does not actually show the credit card number to the merchant. The merchant's SET application authenticates the buyer's identity. The application then decrypts the order information, processes the order, and forwards the still-encrypted payment information to the acquirer for processing. The acquirer's SET application authenticates the buyer's credit card information, identifies the merchant, and arranges settlement. With SET, the Internet becomes a safer, more secure environment for the use of payment cards.

Exchanging keys safely between networks

The practice of transmitting clear keys between networks can be a security exposure. Persons that obtain the clear keys can use them to decrypt transmitted data. ICSF offers several ways to eliminate this problem and ensure that keys are transmitted safely.

Exchanging keys using DES callable services

ICSF provides these security measures for DES key exchange:

- Encrypting the keys to be sent between systems, so that they are not in the clear.
- Requiring that specialized transport keys protect the data-encrypting keys or key-encrypting keys. Transport keys can be used only to protect other keys; they cannot be used for other cryptographic operations.
- Requiring that the sending (exporting) and receiving (importing) of a key be by two different, complementary forms of the same transport key (for example, export and import). These two forms are complements of each other. You cannot use a key in place of its complement.
- Requiring that a key protected under a transport key be made no longer operational—that is, not usable for other cryptographic functions such as encryption, MAC verification, and PIN verification. Only the receiving system can make a protected key operational.

An “exported” key is a key that leaves your system. The transport key that is used to protect it is called an exporter key-encrypting key. When another system receives the key, the key is still protected under the same key-encrypting key. This key-encrypting key must be installed as an importer key-encrypting key on the

receiving system. Before two systems can exchange keys, they must establish pairs of transport keys. The exporter key-encrypting key and the importer key-encrypting key are a complementary pair. You can set up pairs of transport keys, using the key generator utility program (KGUP) or callable services. To exchange keys in only one direction, you need a single pair of transport keys. To exchange keys in both directions, you need two pairs of transport keys. The illustration in Figure 6 shows an example of using DES transport keys to exchange keys between systems.

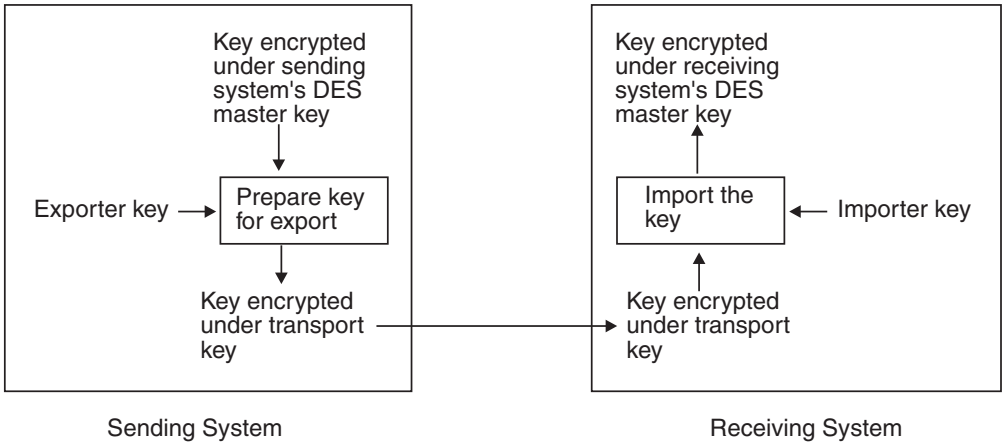

Figure 6. Key Exchange in a DES Cryptographic System

Note: In Program Cryptographic Facility (PCF) applications, transport keys could only protect data-encrypting keys. In ICSF, all DES keys can be protected and securely distributed through the use of transport keys.

Exchanging DES or AES data-encrypting keys using an RSA key scheme

The ability to create secure key-exchange systems is one of the advantages of combining DES or AES and PKA support in the same cryptographic system. Because PKA cryptography uses more intensive computations than DES or AES cryptography, it is not the method of choice for all cryptographic functions. PKA cryptography enhances the security of DES or AES key exchanges. DES or AES data-encrypting keys that are encrypted using an RSA public key can be exchanged safely between two systems. The sending system and the receiving system do not need to share a secret key to be able to exchange RSA-encrypted DES or AES data-encrypting keys. Figure 7 shows an example of this. The sending system enciphers the DES data-encrypting key under the receiver's RSA public key and sends the enciphered data-encrypting key to the receiver. The receiver deciphers the data-encrypting key by using the receiving system's RSA private key.

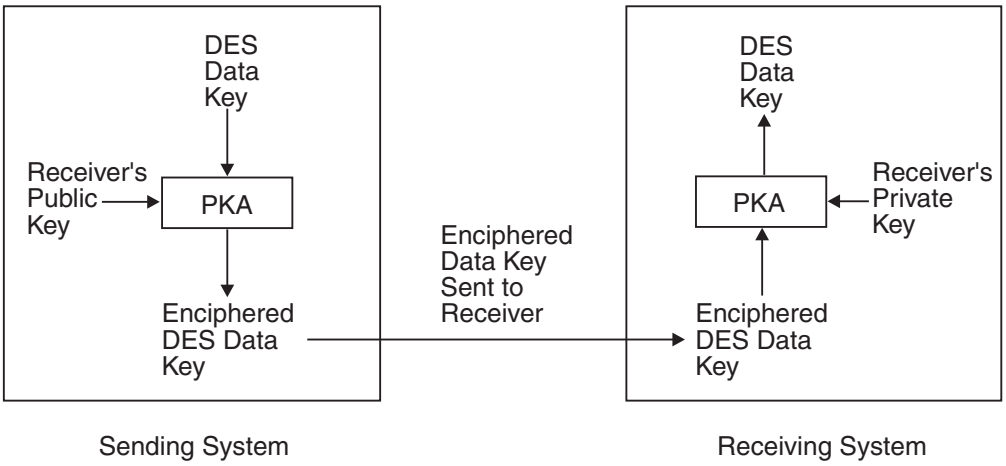

Figure 7. Distributing a DES Data-encrypting Key Using an RSA Cryptographic Scheme

Creating DES or AES Keys using an ECC Diffie-Hellman key scheme

ECC Diffie-Hellman allows two systems to create a symmetric key without exchanging the key.

The sender's private ECC key and the receiver's public ECC keys are combined with the party information to generate a symmetric key that could be used to encipher a message. The ciphertext is transferred to the receiver's system. On the receiver's system, the receiver's private ECC key and the sender's public keys are combined with the party information to generate a symmetric key that could be used to decrypt the ciphertext.

Exchanging keys and their attributes with non-CCA systems

A TR-31 key block is a format defined by the American National Standards Institute (ANSI) to support the interchange of keys in a secure manner with key attributes included in the exchanged data. The TR-31 key block format has a set of defined key attributes that are securely bound to the key so that they can be transported together between any two systems that both understand the TR-31 format. ICSF enables applications to convert a CCA token to a TR-31 key block for export to another party, and to convert an imported TR-31 key block to a CCA token. This enables you to securely exchange keys and their attributes with non-CCA systems.

Only DES/TDES keys can be transported in TR-31 key blocks. There is no support for transporting AES keys.

Refer to *z/OS Cryptographic Services ICSF Application Programmer's Guide* for more information.

Managing master keys using a Trusted Key Entry workstation

ICSF supports the Trusted Key Entry (TKE) workstation. It is available as an optional feature on:

- IBM z196
- IBM System z10 Business Class
- IBM System z10 Enterprise Class

- IBM System z9 EC
- IBM System z9 BC
- IBM @server zSeries 990
- IBM @server zSeries 890

The TKE workstation enables the creation of a logically secure channel for master key entry and key distribution. All versions of the TKE workstation are secure.

Integrity and Privacy

The TKE workstation uses a variety of public key cryptographic techniques to ensure both the integrity and the privacy of the master key transfer channel. In addition, you can use a single TKE workstation to set up master keys in all the cryptographic coprocessors to which it is TCP/IP attached without manual intervention. The TKE workstation also provides support for loading operational transport and PIN keys on CCF systems and all operational keys on PCIXCC, CEX2C, and CEX3C systems.

Using Personal Identification Numbers (PINs) for personal authentication

Personal authentication is the process of validating personal identities. The personal identification number (PIN) is the basis for verifying the identity of a customer across financial industry networks. ICSF provides callable services to generate and verify PINs, and translate PIN blocks. You can use the callable services to prevent unauthorized disclosures when organizations handle PINs. Except for the Clear PIN generate callable service, PINs never appear in the clear.

ICSF provides services for handling a wide variety of PIN block formats, including:

- ISO Format 0 (same as ANSI X9.8, ECI Format 1, and VISA Format 1)
- ISO Format 1 (same as ECI Format 4)
- ISO Format 2
- ISO Format 3
- VISA Format 2
- VISA Format 3
- VISA Format 4
- IBM 4704 Encrypting PINPAD Format
- IBM 3624 Format
- IBM 3621 Format (same as IBM 5906)
- ECI Format 2
- ECI Format 3

ICSF also supports these Clear PIN generate and verification algorithms:

- IBM 3624 Institution-Assigned PIN
- IBM 3624 Customer-Selected PIN (through a PIN offset)
- IBM German Bank Pool PIN (verify through an institution key)
- IBM German Bank Pool PIN (verify through a pool key and a PIN offset)
- VISA PIN (through a VISA PIN validation value)
- Interbank PIN

For more information about PIN block formats and the ICSF callable services that support PINs, refer to Financial Services in *z/OS Cryptographic Services ICSF Application Programmer's Guide*.

Verifying data integrity and authenticity

ICSF provides several processes for verifying the integrity of transmitted messages and stored data:

- Message authentication codes (MAC)
- Modification detection codes (MDC) or hashes
- Digital signatures
- VISA card-verification value, MasterCard Card Verification Code, Diner's Club CVV, American Express card security codes

These processes enable your applications to verify that a message you have received has not been altered. The message itself can be in clear or encrypted form. In addition, digital signatures also authenticate the message sender's identity. VISA card-verification values ensure the safe transmission of credit card information over a computer network.

Your choice of callable service depends on the security requirements of your environment. If the sender and receiver share a secret key, use MAC processing to ensure both the authenticity of the sender and the integrity of the data. If the sender and receiver do not share a secret key, use a digital signature to ensure both the authenticity of the sender and the integrity of the data. If the sender and the receiver do not share a secret cryptographic key and you need to ensure only the integrity of transmitted data, use a hashing process.

Using Message Authentication Codes

To use message authentication when sending a message, an application generates a MAC for it using the MAC generate callable service and one of these methods:

- The ANSI standard X9.9, option 1 with either a single-length MAC key or a single-length DATA key
- The X9.19 optional double key MAC procedure with a double-length MAC key
- The EMV padding rules with either a single-length or double-length MAC key
- The ISO 16609 CBC mode with a double-length MAC or double-length DATA key
- The AES XCBC MAC algorithm from the IETF RFC 3566 which uses AES DATA key to produce a 96-bit result
- The AES XCBC PRF algorithm from the IETF RFC 3566 which uses AES DATA key to produce a 128-bit result
- The FIPS-198 Keyed-Hash Message Authentication Code (HMAC) algorithm with a variable length HMAC key

The originator of the message then sends the MAC with the message text.

When the receiver gets the message, an application program calls the MAC verification callable service. The service again encrypts the message text by using the same method that was used to compute the original MAC. The callable service then notifies the receiver whether the MAC has been verified or not. The callable service does not allow the receiver to have access to the MAC it generates. Because the sender and the receiver share secret cryptographic keys that are used in the MAC calculation, the MAC comparison also ensures the authenticity of the message.

Generating and verifying digital signatures

An application generates a digital signature for a message by first supplying a hash of the message to the digital signature generate callable service. The callable service then uses the signer's private key to create the signature. ICSF supports the use of RSA, DSS, and ECC digital signatures. To verify the digital signature, the receiver's application supplies a hash of the message and the digital signature to the digital signature verify callable service. The callable service then uses the sender's public key to verify the signature. A return code indicates that the verification either succeeded or failed. Figure 3 on page 7 provides an example of using digital signatures.

Using modification detection codes and message hashing

When you are sending a message, use either the MDC generate callable service, or the one-way hash generate callable service to generate a message hash. The choice depends on the cryptographic standard you are using.

The MDC is a 128-bit value that is generated by a one-way cryptographic calculation. The originator of the message transmits the MDC with integrity to the intended receiver of the file. For instance, the originator could publish the MDC in a reliable source of public information. The receiver of the message can use an application program and the same callable service to generate another MDC. If the two MDCs are identical, the receiver assumes that the message is genuine. If they differ, the receiver assumes that someone or some event altered the message.

A hash is a message digest that is generated by a one-way cryptographic calculation. ICSF supports these hash algorithms:

- MD5 produces a 128-bit hash value
- SHA-1 produces a 160-bit hash value
- SHA-224 produces a 224-bit hash value
- SHA-256 produces a 256-bit hash value
- SHA-384 produces a 384-bit hash value
- SHA-512 produces a 512-bit hash value
- RIPEMD-160 produces a 160-bit hash value

Applications can use the hash value and the originator's private key to generate a digital signature and attach it to the message. The receiver of the message uses the originator's public key to authenticate the digital signature.

Both MACs and hashes can be used similarly to ensure the integrity of data that is stored on the system or on removable media such as tape.

Verifying payment card data

The Visa International Service Association (VISA) and MasterCard International, Incorporated have specified a cryptographic method to calculate the VISA card-verification value (CVV) and the MasterCard card-verification code (CVC). This value relates to the personal account number (PAN), the card expiration date, and the service code and is used to detect forged cards. The CVC can be encoded on either track 1 or track 2 of a magnetic-striped card. Because most online transactions use track-2, the ICSF callable services generate and verify the CVV[2] by the track-2 method.

2. The VISA CVV and the MasterCard CVC refer to the same value. This information uses CVV to mean both CVV and CVC.

The VISA CVV service generate callable service calculates a 1- to 5-byte CVV. This value results from using two data-encrypting keys to DES-encrypt the PAN, the card expiration date, and the service code. The VISA CVV service verify callable service calculates the CVV by the same method. The service compares the CVV it calculates to the CVV supplied by the application (which reads the credit card's magnetic stripe). The service then issues a return code that indicates whether the card is authentic.

Maintaining continuous operations

ICSF provides continuous cryptographic operations. Cryptographic keys stored in a cryptographic key data set (CKDS or PKDS) can be reenciphered under a new master key or updated by using either the key generator utility program or the dynamic CKDS or PKDS update callable services. ICSF performs these updates without disrupting applications in process. With PCF, you need to stop cryptographic functions before changing the master key or updating the CKDS or PKDS. You do not need to stop ICSF or interrupt cryptographic applications before changing the DES or AES master key, refreshing the CKDS or PKDS, or dynamically updating either the CKDS or PKDS.

Note: The ability to change the DES or AES master key or update the CKDS or PKDS without interruption requires that ICSF be running in noncompatibility mode. That is, you must convert all existing PCF applications to the new callable services. For a description of noncompatibility mode, see “Running PCF applications under ICSF” on page 45.

These features and actions enhance the security of cryptographic functions:

- Performing cryptographic calculations and storing master keys within tamper-resistant hardware
- Enforcing separation of DES keys
- Controlling access to functions and keys through the use of RACF
- Generating system management facility (SMF) audit records

Reducing costs by improving productivity

ICSF improves productivity by simplifying routine operations and providing interfaces and callable services that help you manage your enterprise's cryptographic environment.

ICSF simplifies the job of the security administrator by providing ISPF dialogs for key management and distribution. ICSF also provides a pass phrase initialization procedure that generates and loads all needed master keys. Use pass phrase initialization to fully enable your cryptographic system in a minimum of steps. In addition, a series of Master Key Entry panels simplifies the master key entry procedure. These panels permit the administrator to change the DES or AES master key without interrupting application programs that use cryptographic functions.

In enterprises that require enhanced key-entry security, a Trusted Key Entry (TKE) workstation is available as an optional feature. The TKE workstation allows the security administrator to securely load DES, AES, and PKA master keys and operational keys (PIN keys and Transport keys on the CCF). The security administrator can use the TKE workstation to load keys into multiple CCFs, PCICCs, PCIXCCs, CEX2Cs, and CEX3Cs from a remote location.

Restriction: Operational key entry is only available on PCIXCCs/CEX2Cs with TKE Version 4.1 or later, or CEX3Cs with TKE 6.0 or later. PCIXCC, CEX2C, and CEX3C support all operational key types as well as a USER DEFINED key type that allows the user to specify a non-default control vector. This control vector must conform to the rules of valid control vectors. AES master key and operational key entry is only available with TKE Version 5.3 or later.

ICSF provides the application programmer with a set of callable services that support cryptographic functions and key management protocols. Applications written in Assembler and several high-level programming languages can use these callable services.

ICSF provides the systems programmer with an easy method of setting and changing the ICSF installation options. The systems programmer needs only to edit an options data set rather than altering an object module. ICSF provides a sample installation options data set in members CSFPRM00 and CSFPRM01 of SYS1.SAMPLIB.

Improving cryptographic performance

ICSF uses state-of-the-art hardware to improve performance of DES, AES and PKA calculations. This can remove limitations on the growth of your installation and enable it to use cryptography in high-transaction-rate applications. ICSF also improves performance by exploiting z/OS and by using an in-storage copy of the CKDS and PKDS. Maintaining DES, AES or PKA cryptographic keys in a protected data space (in addition to a data set) improves performance and availability by reducing requirements for read access to cryptographic keys.

Using RMF and SMF to monitor z/OS ICSF events

You can run ICSF in different configurations and use installation options to affect ICSF performance. While ICSF is running, you can use the Resource Management Facilities (RMF) and System Management Facilities (SMF) to monitor certain events. For example, ICSF records information in the MVS SMF data set when ICSF status changes in a processor or when you enter or change the master key. ICSF also sends information and diagnostic messages to data sets and consoles.

With the availability of cryptographic hardware on an LPAR basis, RMF provides performance monitoring in the Postprocessor Crypto Hardware Activity report. This report is based on SMF record type 70, subtype 2. The Monitor I gathering options on the REPORTS control statement are CRYPTO and NOCRYPTO. Specify CRYPTO to measure cryptographic hardware activity and NOCRYPTO to suppress the gathering. In addition, overview criteria is shown for the Postprocessor in the Postprocessor Workload Activity Report - Goal Mode (WLMGL) report. Refer to *z/OS RMF Programmer's Guide*, SC33-7994, *z/OS RMF User's Guide*, SC33-7990, and *z/OS RMF Report Analysis*, SC33-7991 for additional information.

ICSF also supports enabling RMF to provide performance measurements on ICSF services (Encipher, Decipher, MAC Generate, MAC Verify, One Way Hash, PIN Translate, and PIN Verify) and functions that use Direct Access Crypto (DAC) CCF instructions.

These functions are also performed on the PCIXCC, CEX2C, and CEX3C, except for One-Way Hash which executes on the CPACF (CP Assist for Cryptographic

Functions). For PCIXCC, CEX2C, or CEX3C services, ICSF counts the number of requests to the PCIXCCs, CEX2Cs, or CEX3Cs as the instruction counts in the measurements.

For diagnosis monitoring, use Interactive Problem Control System (IPCS) to access the trace buffer and to format control blocks.

Improving performance in a CICS environment

ICSF supports a CICS-ICSF attachment facility that improves the performance of applications in the CICS regions when an application in the region requests a long-running ICSF service. The attachment facility consists, in part, of a CICS Task Related User Exit (TRUE) that attaches a task control block that does the actual call to the ICSF services. The CICS Resource Manager Interface allows a CICS application program to invoke code that is not written expressly for use under CICS, using the application programming interface that is native to that code. Code that is accessed in this manner is called a resource manager. In the case of the CICS-ICSF attachment facility, ICSF becomes a resource manager for CICS. This means that a CICS application desiring to use long-running ICSF services (such as PKA operations) can be placed in a CICS WAIT rather than an OS WAIT for the duration of the operation. This results in improved performance for other applications that are running in the same CICS region.

The CICS TRUE offloads CICS transaction cryptographic work that might give up control to a z/OS subtask. Synchronous work done on the CCFs would not benefit from the use of the TRUE. This is quite different when a PCICA, PCICC, PCIXCC, CEX2C, CEX2A, CEX3C, or CEX3A feature is used. All work that is performed on these features is asynchronous, and gives up control at least once due to PAUSE processing or LATCH suspension. If the TRUE is not used under CICS, cryptographic work directed to these features will be effectively single threaded. Use of the CICS TRUE is mandatory. The default CICS WAITLIST contains the names of all services that use the PCICA, PCICC, PCIXCC, CEX2C, CEX2A, CEX3C, or CEX3A features, and should be used without modification. On a z900 system, with a PCICC, the default CICS WAITLIST should be examined for the services used by your applications. Any of the services you use, that are shown in the default CICS WAITLIST as being executed on the PCICC, should be added to your WAITLIST.

For additional information about installing the CICS-ICSF attachment facility or creating a modifiable CICS Wait List, refer to the WAITLIST parameter in *z/OS Cryptographic Services ICSF System Programmer's Guide*. The parameter is an option in the Installation Options data set and points to a modifiable data set which contains the names of services that are placed in the CICS Wait List. If this option is not specified, the default ICSF CICS Wait List will be utilized by ICSF when a CICS application invokes an ICSF callable service.

Customizing ICSF to meet your installation's needs

ICSF provides the flexibility your installation needs to customize your cryptographic system.

Using ICSF exits to meet special needs

Exits are programs that your system programmer writes to meet your installation's particular needs. These exits (and installation-defined callable services) perform tasks such as tailoring, monitoring, changing, or diagnosing ICSF. Use of such interfaces can create dependencies on the detailed design or implementation of ICSF. For this reason, use installation exits only for these specialized purposes.

ICSF exits include:

- Exits called when an operator command starts, stops, or changes ICSF
- An exit for each of the callable services
- Exits that are called when you access the disk copy of the CKDS
- An exit that is called when an application accesses the in-storage CKDS

For more information about ICSF exits, refer to Installation Exits in *z/OS Cryptographic Services ICSF System Programmer's Guide*.

Creating installation-defined callable services

Your installation can define a callable service that will run in the ICSF address space and have access to selected ICSF control blocks.

The UDX function is invoked by an "installation-defined" or generic callable service. The callable service is defined in the Installation Options data set (UDX parameter) and the service stub is link-edited with the application. The application program calls the service stub which accesses the UDX installation-defined service.

There is a one-to-one correspondence between a specific generic service in ICSF and a specific UDX command processor in the PCICC, PCIXCC, CEX2C, or CEX3C. The administrator, through ISCF panels, performs UDX authorization processing on each PCICC. (Explicit authorization through the ICSF TSO panels is not required for UDXs in the PCIXCC, CEX2C, and CEX3C.) Authorization is not LPAR specific. See Managing User Defined Extensions in *z/OS Cryptographic Services ICSF Administrator's Guide*, SA22-7521 for additional information. Contact IBM Global Services for any problems with UDX.

Development of a UDX for a PCIXCC, CEX2C, or CEX3C requires a special contract with IBM. See the *UDX Reference and Guide* and the *4758 Custom Software Developer's Toolkit Guide* for additional information. These, and other publications related to the IBM 4758 Coprocessor can be obtained in PDF format from the Library page located at http://www.ibm.com/security/cryptocards.

Using options to tailor ICSF

ICSF lets your installation use different sets of options at different times in the operation of your system. Your installation can specify which options are in each set. These are some of your choices:

- You can choose which of three migration options to use when migrating from or coexisting with PCF: noncompatibility mode, compatibility mode, or coexistence mode.

  For more information on running PCF applications with ICSF, refer to “Running PCF applications under ICSF” on page 45.
- You can allow processing in special secure mode, in which you can work with clear keys and clear PINs. Alternatively, you can disallow processing in that mode.
- For each exit point, you can specify the name of the exit routine and operating information.
- You can alter the REASONCODES options parameter in the Installation Options data set to determine which set of reason codes (ICSF or TSS values) are returned to application service calls. If the REASONCODES option is not specified, the default of REASONCODES(ICSF) is used. The codes will only be converted if there is a 1-to-1 correspondence.

- You can use the WAITLIST (data_set_name) options parameter in the Installation Options Data Set to point to a modifiable data set that contains the names of services that are placed into the CICS Wait List. If the WAITLIST option is not specified, the default ICSF CICS Wait List will be utilized by ICSF when a CICS application invokes an ICSF callable service.
- You can use the UDX(UDX-id,service-number,load-module name,'comment_text',FAIL(fail-option)) parameter to define a User Defined Extension (UDX) service to ICSF.
- You can use the KEYAUTH initialization option to specify if verification of the record authentication pattern of the in-storage CKDS records should be performed. It is intended to detect corruption in these in-storage records.

  On z900 processors, this processing was performed on the CCFs, and the performance impact was minimal. On later processors with use of the PCIXCC, CEX2C, or CEX3C features, the MACing is performed on the features and the performance impact is substantial. There has never been a reported instance of a MAC verification failure. It is recommended that KEYAUTH(NO) be specified in the options data set on these processors.

  Note: The KEYAUTH setting will be ignored if the CKDS was initialized with record level authentication disabled. KEYAUTH(YES) is only supported if record level authentication was enabled when the CKDS was initialized.

Isolating and protecting PR/SM partitions

If you are using the Processor Resource/Systems Manager (PR/SM) feature to run in logically partitioned mode, each PR/SM partition is able to use its own DES and PKA master keys on the Cryptographic Coprocessor Feature and DES, AES and asymmetric-keys master keys on the PCI X Cryptographic Coprocessor, Crypto Express2 Coprocessor, Crypto Express3 Coprocessor, or PCI Cryptographic Coprocessor. This allows your installation to have multiple independent cryptographic systems running on the same processor with the same degree of isolation and protection as if they were running on physically separate processors.

Enabling growth

For applications that need to protect critical data against disclosure or modification, ICSF provides callable services that enable high-level language applications to easily access the system's underlying cryptographic functions.

By providing callable services that comply with IBM's Common Cryptographic Architecture (CCA), ICSF enables application designers and programmers to extend the uses of their current applications. Most of the callable services provided by ICSF are also provided by the IBM 4753 Network Security Processor MVS Support Program (4753-HSP), program number 5706-028. This allows the development of significant applications for both CCA and ANSI X9.17 key management that will run without change on both systems.

ICSF's callable services enable installations to add cryptographic functions (such as MAC generation and verification) to current applications without redesign.

The combination of the hardware cryptographic features and ICSF provides high-performance cryptography, which removes bottlenecks on high-volume transaction applications and gives them needed protection. ICSF can support various combinations of cryptographic hardware.

An installation can use ICSF installation exits to change or extend the callable services.

Protecting your investment

The use of an enterprise's computing resources is improved and protected by built-in product features.

ICSF also ensures that existing Program Cryptographic Facility (PCF) cryptographic applications, skills, and equipment can continue to be used effectively. This facilitates the earlier implementation of desired security applications while minimizing the disruption of existing applications.

- Existing PCF applications can run without change and without reassembly on ICSF in compatibility mode.
- A PCF conversion program has been provided to convert a PCF cryptographic key data set to an ICSF format.
- ICSF applications can run concurrently on the same processor with PCF applications.

Chapter 3. Application Programming Interfaces and key management

This topic describes the ICSF callable services and some of the concepts of cryptographic key management.

Callable services

ICSF provides access to cryptographic functions through callable services. A callable service is a routine that receives control from a CALL statement in an application language. Each callable service performs one or more cryptographic functions or a utility function. Many of these callable services comply with IBM's Common Cryptographic Architecture (CCA), while others are extensions to the CCA.

The callable services available to your applications depend on your processor or server. For a list of the callable services available with each configuration, refer to Appendix B, “Summary of callable service support by hardware configuration,” on page 71.

The ICSF Query Facility (CSFIQF) returns coprocessor information as well as general information about ICSF. The ICSF Query Algorithm (CSFIQA) returns the cryptographic and hash algorithms available.

The application programs can be written in high-level languages such as C, COBOL, FORTRAN, and PL/I, as well as in Assembler. ICSF callable services allow applications to perform these tasks:

- Enciphering and deciphering data by using the cipher block chaining (CBC) form of the DES or TDES algorithms, with or without record chaining using a single (64-bit), double (128-bit), or triple (192) length key. If using the AES algorithm, key lengths are 128, 192 or 256 bits.
- Translating ciphertext from encryption under one key to encryption under another key by use of the Ciphertext translate callable service.

  This service securely deciphers the text that was enciphered under one key, and then enciphers it under another key. The service uses the cipher block chaining (CBC) form of the DES algorithm.
- Generating DES cryptographic keys of all types for use by application programs.
- Generating AES operational cryptographic keys for use by application programs.
- Importing and exporting keys.
- Generating PKA keys.

  Application programs can use the PKA key generate callable service to generate:
  - a DSS internal token for use in digital signature services.
  - RSA public and private keys on a PCICC, PCIXCC, CEX2C, or CEX3C
  - ECC keys on a CEX3C.
- Listing and deleting retained RSA private keys.

  Application programs can list and delete RSA private keys retained within the secure boundaries of a PCICC, PCIXCC, CEX2C, or CEX3C.
- Generating random numbers.

  Application programs can use a callable service to generate a random number for use in cryptography or for other general use. The callable service uses the

cryptographic feature to generate a random number for use in encryption. The foundation for the random number generator is a time-variant input with a very low probability of recycling.

- Encoding and decoding data through the use of clear keys and the electronic code book (ECB) form of the DES algorithm.
- Generating and verifying PINs and translating PIN blocks.

  An application program can use the callable services in generating and verifying PINs. In addition, use the Encrypted PIN translate callable service to reencrypt a PIN block from one PIN-encrypting key to another, or to reformat a PIN block.
- Generating and verifying DES MACs.

  An application can use single-length or double-length MAC or MACVER keys, or single-length DATA keys to generate and verify message authentication codes.
- Generating and verifying AES MACs.

  An application can use an AES DATA key to generate and verify message authentication codes.
- Generating and verifying HMAC MACs.

  An application can use an HMAC key to generate and verify message authentication codes.
- Generating MDCs and other hash patterns.
- Generating and verifying Visa CVVs.
- Developing EMV ICC applications
- Enabling exploitation of clear key DES instructions on CPACF
- Writing Diffie-Hellman applications
- Updating the CKDS and PKDS dynamically.

  ICSF provides callable services that application programs can use to create, read, write, and delete records in the CKDS and PKDS.
- Distributing DATA keys enciphered under an RSA key.
- Generating and verifying digital signatures.
- Composing and decomposing SET blocks.
- PKA-encrypting and PKA-decrypting any PKCS 1.2-formatted symmetric key data.
- Supporting the ANSI X9.17 key management standard.

  ICSF provides callable services for generating, exporting, importing, translating, and notarizing ANSI X9.17 keys.

Protecting and controlling DES keys

DES keys are protected by encryption under a DES master key (DES-MK). The DES master key always remains within the secure boundary of the cryptographic coprocessor (cryptographic coprocessor feature, PCICC, PCIXCC, CEX2C, or CEX3C) on the server. There is only one DES master key and it is used only to encrypt and decrypt other DES keys. All coprocessors must have the same DES master key for the coprocessors to be active.

Note: On a z990 or z890 without a PCI X Cryptographic Coprocessor or Crypto Express2 Coprocessor, there is no encrypted key support. Also, on a z9 EC, z9 BC, z10 EC, z10 BC, or z196 without a Crypto Express2 Coprocessor or Crypto Express3 Coprocessor, there is no encrypted key support.

The cryptographic hardware controls the use of DES keys by separating them into unique types. A unique key type can be used only for a specific purpose. For example, you cannot protect a key with a key that is intended to protect data. This hardware-enforced key separation provides better key protection than software key separation techniques. To enforce key separation, the cryptographic hardware automatically encrypts each type of key under a unique variation of the DES-MK. Each variation encrypts a different type of key. Although you enter only one DES-MK, in effect you have a unique master key to encrypt each DES key type.

Note: In ICSF, key separation applies to keys that are encrypted under the master key, as well as keys that are encrypted under transport key or key-encrypting keys. This enables the creator of a key to transmit the key to another system and to enforce its use at the other system.

DES master key variant

Each key must be enciphered under the DES master key before it can be used in any cryptographic function. Each key type is enciphered with a unique variation of the master key called a *Master key variant*. ICSF creates a master key variant by exclusive ORing a fixed pattern, called a *control vector*, onto the master key. For information about control vectors, refer to “Control vectors” on page 30.

Each master key variant protects a different type of key. The effect is similar to having a unique master key to protect all the keys of a certain type. The master key, in its variants, protects keys that operate on the system. When systems want to share keys, they use transport keys to protect keys sent outside of systems.

DES transport key variant

As with the master key, ICSF also creates variations of a DES transport key to encrypt a key according to its type. This allows for key separation when transporting keys off the system. A *transport key variant*, or *key-encrypting key variant*, is created in the same way as a master key variant. The transport key is exclusive ORed with a control vector that is associated with the key type of the key it protects.

Note: To exchange keys with systems that do not recognize transport key variants, ICSF allows you to encrypt selected keys under a transport key itself, not under the transport key variant.

DES key forms

A key that is protected under the DES master key is in *operational form*, which means that ICSF can use it in cryptographic functions on the system.

When you store a key with a file or send it to another system, the key is enciphered under a transport key rather than the master key. When ICSF enciphers a key under a transport key, the key is not in operational form and cannot be used to perform cryptographic functions.

When a key is enciphered under a transport key, the sending system considers the key to be in the *exportable form*. The receiving system considers the key to be in the *importable form*. When a key is re-enciphered from under a transport key to under a system's master key, it is in operational form again.

Control vectors

For each type of DES key the master key enciphers, there is a unique control vector. The cryptographic feature exclusive ORs the master key with the control vector associated with the type of key the master key will encipher. For example, all the different types of DATA, PIN, MAC, and transport keys are each exclusive ORed with a unique control vector. The control vector ensures that an operational key can be used in only cryptographic functions for which it is intended. For example, the control vector for an input PIN-encrypting key ensures that such a key can be used only in the PIN translation and PIN verification functions. “Types of DES keys” describes the different DES key types.

Types of DES keys

ICSF groups DES cryptographic keys into these categories according to the functions they perform.

- DES Master key

  The DES master key is a double-length (128-bit) key that is used only to encrypt other DES keys. The ICSF administrator installs and changes the DES master key using the ICSF panels. Alternatively, you can use the optional TKE workstation. The master key always remains within the secure boundary of the cryptographic feature. The DES master key is installed in the cryptographic coprocessors.

  The DES master key is used only to encipher and decipher operational keys. Cryptographic keys that are in exportable or importable form are not enciphered under the master key. They are enciphered under the appropriate transport key, which has itself been enciphered under the master key.

- Transport keys (or key-encrypting keys)

  Transport keys are also known as key-encrypting keys. They are double-length (128-bit) keys that are used to protect keys when you distribute them from one system to another. For installations that do not support double-length 128-bit keys, ICSF supports the use of effective single-length keys. In an effective single-length key, the left half equals the right half.

  The DES transport keys are:

  – *EXPORTER or OKEYXLAT key-encrypting keys* protect keys of any type that are sent from your system to another. The exporter key at the originator is the same as the importer key of the receiver. An exporter key is paired with an importer key or a IKEYXLAT key.

  – *IMPORTER or IKEYXLAT key-encrypting keys* protect keys of any type that are sent from another system to yours. It also protects keys that you store externally in a file that you can import to your system at another time. The importer key at the receiver is the same as the exporter key at the originator. An importer key is paired with an exporter key or a OKEYXLAT key.

  – *ANSI X9.17 key-encrypting keys (AKEKs)* are used exclusively with the ANSI X9.17 key management callable services. AKEKs are used to transport DATA keys, AKEKs, and CCA key-encrypting keys.

  Note: Transport keys replace the local, remote, and cross keys that PCF uses.

- Data-encrypting keys

  Data-encrypting (DATA) keys are single-length (64-bit), double-length (128-bit), or triple-length (192-bit) keys. DATA keys are used to encipher and decipher data. ICSF provides support for the use of single-length data-encrypting keys in the callable services that generate and verify MACs.

DATAC, a double length key for enciphering and deciphering data, is supported with a PCIXCC, CEX2C, or CEX3C.

- CIPHER Keys

  These consist of CIPHER, ENCIPHER and DECIPHER keys. They are single and double length keys for enciphering and deciphering data.

- Data-translation keys

  These single-length (64-bit) keys are used for the ciphertext translate callable service as either the input or the output data-translation (DATAXLAT) key.

- MAC keys

  These can be single (64-bit) or double-length (128-bit) MAC and MACVER keys and double-length DATAM or DATAMV keys. These keys can be used to generate and verify MACs

- PIN keys

  The personal identification number (PIN) is a basis for verifying the identity of a customer across financial industry networks. PIN keys are double-length (128-bit) keys. The callable services that generate, verify, and translate PINs use PIN keys.

  For installations that do not support double-length 128-bit keys ICSF provides effective single-length keys. In an effective single-length key, the left key half of the key equals the right key half.

- Cryptographic variable encrypting keys

  These single-length keys are used to encrypt special control values in CCA DES key management. The Control Vector Translate and Cryptographic Variable Encipher callable services use cryptographic variable encrypting keys.

Protecting and controlling AES keys

AES keys may be clear keys or secure keys protected by encryption under a AES master key. The AES master key always remains within the secure boundary of the cryptographic coprocessor on the server. There is only one AES master key and it is used only to encrypt and decrypt other AES keys and HMAC keys. All coprocessors must have the same AES master key for the coprocessors to be active.

AES key forms

A key that is protected under the AES-MK is in *operational form*, which means that ICSF can use it in cryptographic functions on the system.

When you store a key with a file or send it to another system, the key can be protected using an RSA key pair.

Types of AES keys

ICSF groups AES cryptographic keys into these categories according to the functions they perform.

- AES Master key

  A 256-bit AES key that is used only to encrypt and decrypt AES or HMAC operational keys. The ICSF administrator installs and changes the AES master key using the ICSF panels or the optional TKE workstation. The AES master key always remains within the secure boundaries of the cryptographic coprocessors.

- Transport keys (or key-encrypting keys)

Transport keys protect a key that is sent to another system, received from another system, or stored with data in a file. AES transport keys are variable-length keys up to 725 bytes in length.

The AES transport keys are:

- EXPORTER Key-encrypting Key

  An EXPORTER key-encrypting key protects keys that are sent from your system to another system. The exporter key at the originator has the same clear value as the importer key at the receiver. An exporter key is paired with an importer key-encrypting key.

- IMPORTER Key-encrypting Key

  An importer key-encrypting key protects keys that are sent from another system to your system. It also protects keys that you store externally in a file that you can import to your system later. The importer key at the receiver has the same clear value as the exporter key at the originator. An importer key is paired with an exporter key-encrypting key.

- Data-encrypting keys

  Data-encrypting keys, also referred to as DATA keys, are used to encrypt and decrypt data. AES DATA keys can be 128-bits, 192-bits, or 256-bits in length. DATA keys can be either encrypted under the master key or in the clear.

- CIPHER keys

  AES CIPHER keys are used for enciphering and deciphering data. 128-, 192-, or 256-bits in length.

Protecting and controlling HMAC keys

HMAC keys are protected by encryption under the AES master key. The AES master key always remains within the secure boundary of the cryptographic coprocessor on the server. There is only one AES master key and it is used only to encrypt and decrypt other AES and HMAC keys. All coprocessors must have the same AES master key for the coprocessors to be active.

HMAC key forms

A key that is protected under the AES-MK is in operational form, which means that ICSF can use it in cryptographic functions on the system.

When you store a key with a file or send it to another system, the key can be protected using an RSA key pair.

HMAC keys

HMAC keys are variable length keys used to generate and verify MACs using the FIPS-198 Keyed-Hash Message Authentication Code (HMAC) algorithm.

DES key token wrapping

ICSF wraps the key value in a DES key token using one of two possible methods.

- The original method of DES key wrapping has been used by ICSF since its initial release, and is the only key wrapping method that was available prior to FMID HCR7780. Using this original key wrapping method, the key value in DES tokens are encrypted using triple DES encryption, and key parts are encrypted separately.
- The enhanced method of symmetric key wrapping, introduced in FMID HCR7780, is designed to be ANSI X9.24 compliant. Using the enhanced method, the key

value for keys is bundled with other token data and encrypted using triple DES encryption and cipher block chaining mode. The enhanced method is available only on the z196 with a CEX3C, and applies only to DES key tokens.

The ICSF system programmer can specify the default wrapping method that ICSF will use for internal key tokens and external key tokens. The default wrapping method for internal key tokens and the default wrapping method for external key tokens are independent to each other and are specified separately. ICSF will use the specified method unless overridden by rule array keywords or by supplying a skeleton token with a different wrapping.

A CKDS conversion utility, CSFCNV2, enables you to convert all tokens in the CKDS to use either the original or the enhanced wrapping method.

If you are sharing a CKDS with a release of ICSF that doesn't support the enhanced wrapping method, you should use the original wrapping method until your systems all support the enhanced method. Releases of ICSF that don't support the enhanced method can't use the key tokens.

Protecting and controlling PKA keys

In a public key cryptographic system, it is a priority to maintain the security of the private key. It is vital that only the intended user or application have access to the private key.

On supported IBM servers, ICSF and the cryptographic hardware features (CCFs, PCICCs, PCIXCCs, CEX2Cs, or CEX3Cs) ensure this by enciphering PKA private keys under a unique PKA object protection key. The PKA object protection key has itself been enciphered under a PKA master key. Each PKA private key also has a name that is cryptographically bound to the private key and cannot be altered. ICSF uses the private key name or the PKDS key label to control access to the private key. This combination of hardware-enforced coupling of cryptographic protection and access control, through the use of the Security Server (RACF), is unique to ICSF. It provides a significant level of security and integrity for PKA applications.

On servers that support them, the PCIXCC, CEX2C, and CEX3C provide additional security for PKA applications. You can generate RSA public and private key pairs within the secure hardware boundary of the PCIXCC/CEX2C. In addition, you can retain the RSA private key within the cryptographic coprocessor where it is generated, a requirement to be a SET Certificate Authority. The RSA private key is protected by the ASYM-MK on the PCIXCC, CEX2C, or CEX3C.

PKA master keys

The PKA master keys on the Cryptographic Coprocessor Feature are triple-length (192-bit) keys. As with the DES master key, the PKA master keys are used only to encipher and decipher PKA keys. There are two PKA master keys on the Cryptographic Coprocessor Feature. One PKA master key, the signature master key (SMK), protects private keys that are intended for creating digital signatures. The other PKA master key, the key management master key (KMMK), protects private keys that are used in DES key distribution. Private keys that are protected by the KMMK can also be designated to be usable to generate digital signatures. It is highly recommended that the KMMK have the same value as the SMK.

The asymmetric keys master key (ASYM-MK) on the PCICC, PCIXCC, CEX2C, or CEX3C is a triple-length key used to encipher and decipher PKA keys. In order for the PCI Cryptographic Coprocessor to function, the hash pattern of the ASYM-MK

must have the same value as the hash pattern of the SMK on the Cryptographic Coprocessor Feature. There is no such dependency with the PCIXCC, CEX2C, or CEX3C.

On the IBM z196 with a CEX3C, there are two PKA master keys: RSA and ECC. The RSA master key (RSA-MK) is the same as the ASYM-MK. The ECC master key is a 256-bit AES key used to protect ECC private keys.

The ICSF administrator installs the PKA master keys on the Cryptographic Coprocessor Feature and the ASYM-MK on the PCICC, PCIXCC, CEX2C, or CEX3C by using either the ICSF pass phrase initialization panel, the clear master key entry panels, or the optional TKE workstation.

RSA private and public keys

An RSA key pair includes a private and a public key. The RSA private key is used to generate digital signatures, and the RSA public key is used to verify digital signatures. The RSA public key is also used for key encryption of DES or AES DATA keys and the RSA private key for key recovery.

The RSA public key algorithm is based on the difficulty of the factorization problem. The factorization problem is to find all prime numbers of a given number, n. When n is sufficiently large and is the product of a few large prime numbers, this problem is believed to be difficult to solve. For RSA, n is typically at least 512 bits, and n is the product of two large prime numbers. For more information about the RSA public key algorithm, refer to the ISO 9796 standard and *RSA's Frequently Asked Questions About Today's Cryptography*.

Generating RSA keys on a Cryptographic Coprocessor Feature

The Cryptographic Coprocessor Feature does not provide the ability to generate RSA public and private keys within the secure hardware boundary. There are several ways to generate RSA key pairs and load them.

- You can use the optional TKE Workstation with Version 2.0 or higher of the TKE Workstation code to generate the RSA key pair and load them directly into the ICSF public key data set (PKDS) on the server.
- You can generate RSA key pairs in the encrypted form on a workstation with a 4755 cryptographic adapter or a 4764 PCIX Cryptographic Coprocessor installed. A workstation with a 4758 PCI Cryptographic Coprocessor can also be used. Use the PKA key import callable service to import the RSA key pairs in the form of an external PKA private key token.
- You can generate RSA keys in the clear on another platform or by using any suitable software program. Use the PKA Key Token Build callable service to build an external token with clear key values. Then use the PKA key import callable service to import the RSA keys into the PKA key token in the internal format.

Generating RSA keys on a PCICC, PCIXCC, CEX2C, or CEX3C

With the PCICC, PCIXCC, CEX2C, or CEX3C, you can use the PKA key generate callable service to generate RSA public and private key pairs within the secure boundary of the cryptographic coprocessor. The PCICC/PCIXCC can generate RSA keys with a modulus size of 512 to 2048 bits. The CEX2C and CEX3C can generate RSA keys with a modulus size of 512 to 4096 bits. The RSA private key may be retained and used within the secure boundary of the cryptographic coprocessor. This capability is a requirement to be a SET Certificate Authority. The public key and the key name for the private key are stored in the ICSF public key data set (PKDS), but the value of a retained private key never appears in any form outside the cryptographic coprocessor.

ECC private and public keys

An ECC key pair includes a private and public key. The ECC private key is used to generate digital signatures, and the ECC public key is used to verify digital signatures.

ICSF generates ECC key pairs using the Elliptic Curve Digital Signature Algorithm (ECDSA). This algorithm uses elliptic curve cryptography (an encryption system based on the properties of elliptic curves) to provide a variant of the Digital Signature Algorithm.

ECC keys are supported on the IBM z196 with a CEX3C. With a CEX3C that is ECC capable, you can use the PKA key generate callable service to generate ECC keys.

DSA private and public keys

A DSA key pair also includes a private and a public key. The DSA private key is used to generate digital signatures, and the DSA public key is used to verify digital signatures.

The difficulty of the discrete logarithm problem is the basis for the NIST Digital Signature Standard (DSS) public key algorithm. The discrete logarithm problem is to find x given a large prime p, a generator g and a value y=(g**x) mod p, where ** represents exponentiation. This problem is believed to be very hard when p is sufficiently large and x is a sufficiently large random number. For DSA, p is at least 512 bits, and x is 160 bits. The NIST FIPS 186 Digital Signature Standard defines DSA.

ICSF provides a callable service to generate PKA internal key tokens for use with the DSA algorithm in digital signature services.

Restriction: The IBM @server zSeries 990, z890 or later servers do not support DSS.

Exchanging encrypted keys and PINs on a DES system

When a system sends a DATA key to another system, the sending system encrypts the DATA key under an *exporter* key-encrypting key. The receiving system re-encrypts the DATA key from encryption under an *importer* key-encrypting key to encryption under its master key. The importer and exporter key-encrypting keys at these systems complement each other and have the same clear value.

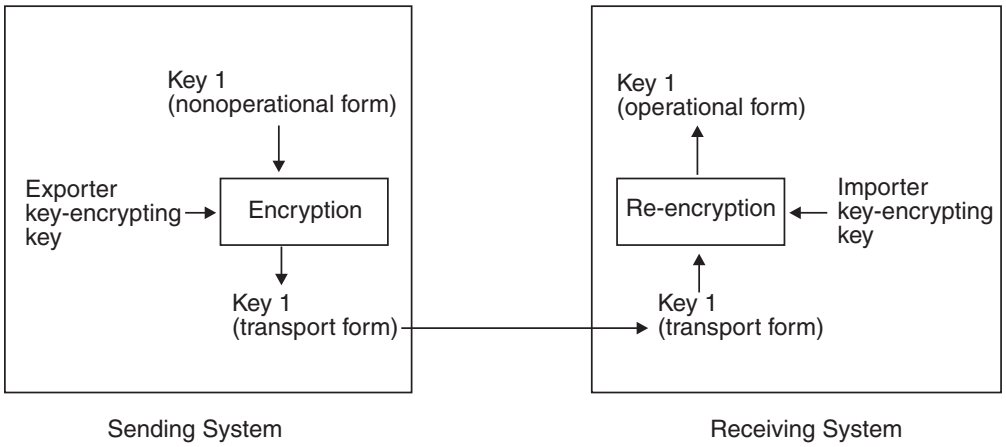

Figure 8. Using Transport Keys to Exchange Keys

In ICSF, you work with these complementary keys:

- Importer key-encrypting key and exporter key-encrypting key
- Importer key-encrypting key and OKEYXLAT key-encrypting key
- Exporter key-encrypting key and IKEYXLAT key-encrypting key
- Input PIN-encrypting key and output PIN-encrypting key
- PIN-generation key and PIN-verification key
- MAC-generation key and MAC-verification key

Your installation can use the key generator utility program (KGUP) or the callable services to generate and maintain complementary pairs of keys.

When KGUP generates a key, it also generates a KGUP control statement to create the complement of that key. You can send the control statement to the system with which you are exchanging keys or PINs.

Exchanging RSA-encrypted data keys

In an RSA cryptographic system, the sending system and the receiving system do not need to share complementary importer and exporter key pairs to exchange DATA keys. The sender enciphers the DATA key by using the receiver's public key. The receiver deciphers the DATA key by using his or her own private key. Refer to “Exchanging DES or AES data-encrypting keys using an RSA key scheme” on page 15 for a more detailed explanation.

Using multiple DES encipherment to protect keys and data

The Crypto Express2 Coprocessor, Crypto Express3 Coprocessor, PCI X Cryptographic Coprocessor, Cryptographic Coprocessor Feature and PCI Cryptographic Coprocessor use multiple DES encipherment whenever they encipher a key under a key-encrypting key like the master key or a transport key. In addition to protecting and retrieving cryptographic keys, the Cryptographic Coprocessor Feature uses multiple DES encipherment and decipherment to protect or retrieve 64-bit PIN blocks in the area of PIN applications. Multiple DES encipherment is superior to single encipherment because it is much harder to break. The actual process to encipher a key depends on the type of key that is being enciphered and the type of key-encrypting key that is being used to encipher it.

Figure 9 shows an example of multiple DES encipherment. In this example, the left half of the enciphering key is used to encrypt the key in the first step. The result is then decrypted under the right half of the enciphering key. Finally, this result is encrypted under the left half of the enciphering key again.

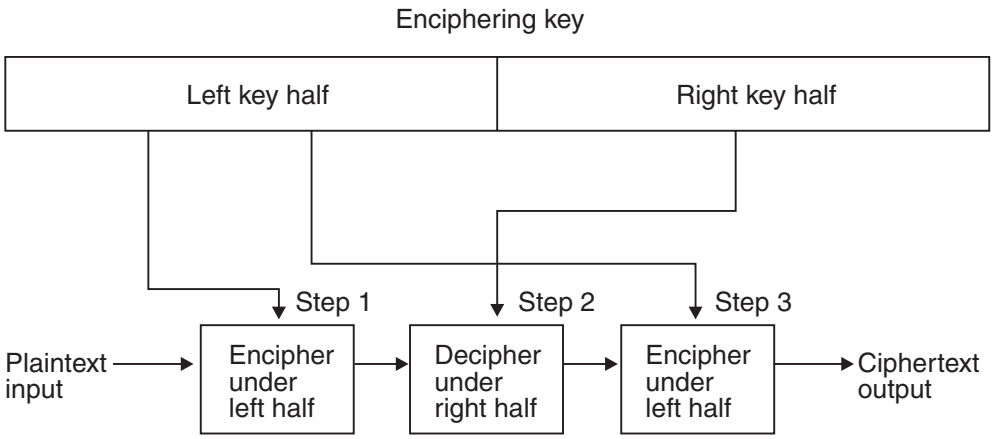

Figure 9. An Example of Multiple Encipherment

Triple DES data encryption uses multiple DES encipherment with either double-length or triple-length DATA keys to protect data. For this procedure the data is first enciphered using the first DATA key. The result is then deciphered using the second DATA key. When using a triple-length key, this second result is then enciphered using the third DATA key. When using a double-length key, the first DATA key is reused to encrypt the second result.

Note: Multiple DES decipherment is the inverse of multiple encipherment (decipher-encipher-decipher).

Running in special secure mode

Special secure mode is a special processing mode for the entry of clear keys. To perform these tasks, you must enable Special Secure Mode:

- Use the secure key import or multiple secure key import callable service, which work with clear keys.
- Use the Clear PIN generate service which works with clear PINs.
- Use the symmetric key generate callable service with the IM keyword on a CCF system. Special Secure Mode is not required if a PCICC is available.
- Use KGUP to enter clear keys into the CKDS.

On the IBM @server zSeries 900, special secure mode can be entered only if the hardware is enabled for special secure mode, and if the installation options data set allows it. With the optional TKE workstation, you can access the environmental control mask to enable or disable special secure mode.

On the z990, z890, z9 EC, z9 BC, z10 EC, z10 BC, and z196, special secure mode can be entered only if the installation options data set allows it. Additional hardware control for these callable services can be enforced with the optional TKE workstation.

Cryptographic Key Data Set (CKDS)

ICSF stores AES, DES, and HMAC keys in a specialized data set called a cryptographic key data set (CKDS). ICSF maintains both a disk copy and an in-storage copy of the CKDS. This makes it possible to refresh the cryptographic keys without interrupting the application programs. ICSF provides a sample CKDS allocation job (member CSFCKDS) in SYS1.SAMPLIB. For more information on running in a sysplex environment, see *z/OS Cryptographic Services ICSF Administrator's Guide*, SA22-7521.

ICSF updates the CKDS at these times:

- When you use KGUP to generate keys, enter keys into the system or load keys from the PCIXCC, CEX2C, or CEX3C key part registers, ICSF updates the disk copy, rather than the in-storage copy. ICSF does not require that you stop cryptographic functions before updating the CKDS, unlike PCF. After the update has been made, you can replace the in-storage copy of the CKDS with the disk copy using the ICSF panels.
- When you change the master key, ICSF enables you to reencipher the disk copy of the CKDS. ICSF then automatically refreshes the in-storage copy of the CKDS with the re-enciphered keys.
- When you convert a PCF CKDS to an ICSF CKDS, the PCF conversion program updates the disk copy of the ICSF CKDS.
- When an application uses the dynamic CKDS update callable services, both the disk copy and in-storage copy of the CKDS are dynamically updated.
- When a key or key part is imported from the TSO panel, the key is loaded using the DES Operational Key Load for PCIXCC, CEX2C, or CEX3C systems. If running on the IBM @server zSeries 900, the key part is loaded using the TKE Operational Key Entry for CCF systems.

ICSF allows these operations without interrupting cryptographic functions that are used by application programs.

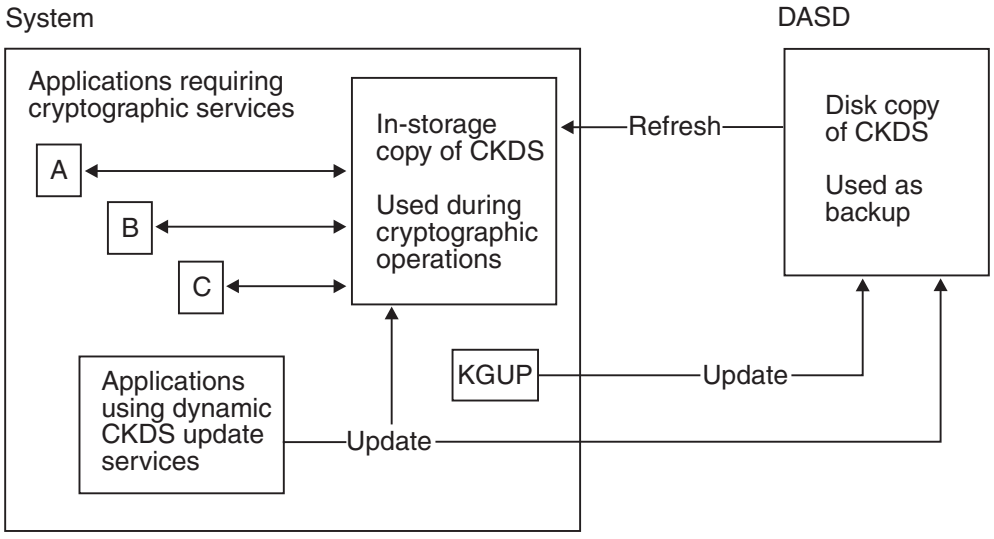

Figure 10. How the Cryptographic Key Data Set Is Maintained and Used

Callable services use the in-storage copy of the CKDS. For example, in Figure 10 on page 38 applications A, B, and C might make many calls for services that require the CKDS. Having the CKDS in storage avoids time-consuming I/O to a data set that is stored on DASD.

KGUP updates the disk copy rather than the in-storage copy. The ICSF administrator can then use the ICSF panel dialog or a batch job to refresh the in-storage CKDS with the updated disk copy of the CKDS on every system sharing the updated CKDS. Cryptographic functions do not have to stop while KGUP updates the CKDS.

The dynamic CKDS update callable services permit an application to perform dynamic update of both the disk copy and the in-storage copy of the CKDS.

Dynamic CKDS update callable services

The dynamic CKDS update callable services allow applications to directly manipulate both the in-storage copy and the DASD copy of the CKDS. These callable services have the identical syntax as the 4753-HSP *verbs* of the same name. Key management applications that use these common callable services, or verbs, can be run on either system without change. Cryptographic functions do not have to stop while the dynamic CKDS update callable services update the CKDS.

Sysplex-wide consistency of CKDS

ICSF implements sysplex-wide consistent updates to the CKDS through the use of Cross-System Coupling Facility (XCF) signalling services and global (that is, sysplex-wide) ENQs. This solves the problem of CKDS copying into an ICSF-managed data space. If a CKDS record is modified by create, update, or delete operation, the DASD version of the CKDS is updated and the ICSF in-storage copy is updated to reflect the new contents of the record. If a CKDS is shared by one or more members of a sysplex, the ICSF in-storage copy is only updated on the sysplex system that initiated the record modification. The only way the ICSF in-storage copy will be updated on other systems of the sysplex is:

1. by manual intervention (the operator must use ICSF's TSO panels to initiate a refresh of the in-storage CKDS data) OR
2. by invocation of CSFEUTIL as a batch job or a called program.

See *z/OS Cryptographic Services ICSF Administrator's Guide*, SA22-7521 for more information.

This will now be avoided by using the ICSF sysplex-wide coherency support.

Restrictions

The restrictions while using the sysplex-wide coherency support are:

- If multiple sysplexes or a sysplex and other non-sysplex system, share a CKDS, there is no provision for automatic update of the in-storage copies of the CKDS on the systems that are not in the same sysplex as the system initiating the CKDS update.
- If KGUP is used to update the CKDS, the update is only made to the DASD copy of the CKDS. A manual or CSFEUTIL refresh of the in-storage copy of the CKDS, via the ICSF TSO panels, is required on all systems sharing the updated CKDS.
- If a master key change operation is to be performed on some or all members of the ICSF sysplex group, the change should be managed carefully. As is the case today, CKDS updates should be disabled until the master key has been changed and the newly reenciphered CKDS is active on all systems.

- The CKDS Entry Retrieval installation exit will not be given control if SYSPLEXCKDS(YES,FAIL(xxx)) is coded in the ICSF Installation Options Data Set.

PKA Cryptographic Key Data Set (PKDS)

You can store RSA, DSS, and ECC public and private keys, and trusted blocks in a specialized external VSAM data set that is called a public key data set (PKDS). ICSF maintains both a disk copy and an in-storage copy of the PKDS. This makes it possible to refresh the cryptographic keys without interrupting the application programs. ICSF provides a sample PKDS allocation job (member CSFPKDS) in SYS1.SAMPLIB. For more information on running in a sysplex environment, see *z/OS Cryptographic Services ICSF Administrator's Guide*, SA22-7521.

PKDS initialization support is available on the Master Key Management panel, CSFPUTIL utility, and PassPhrase Initialization. In order to enable PKA operations, the PKDS must be initialized.

Support to REENCIPHER PKDS and REFRESH PKDS is available in the Master Key Management Panels and CSFPUTIL utility. CSFPUTIL is a utility that performs the same functions as REENCIPHER PKDS and REFRESH PKDS. These functions allow you to reencipher the PKDS from the old asymmetric-keys master key to the current master key and activate the reenciphered PKDS. Other systems with lower levels of ICSF which are sharing the PKDS will still have to ACTIVATE the reenciphered PKDS

Restrictions

The restrictions while using the sysplex-wide coherency support are:

- All members of the sysplex, sharing the PKDS, must be running ICSF HCR7751, or later, in order to participate in sysplex-wide consistency for the PKDS.
- If multiple sysplexes or a sysplex and other non-sysplex system, share a PKDS, there is no provision for automatic update of the in-storage copies of the PKDS on the systems that are not in the same sysplex as the system initiating the PKDS update.
- If a master key change operation is to be performed on some or all members of the ICSF sysplex group, the change should be managed carefully. As is the case today, PKDS updates should be disabled until the master key has been changed and the newly reenciphered PKDS is active on all systems.

Dynamic PKDS update callable services

ICSF provides dynamic PKDS update callable services that permit an application to create, read, write, and delete PKDS records. You do not need to stop cryptographic functions while applications use these services to update the PKDS.

Sysplex-wide consistency of PKDS

ICSF implements sysplex-wide consistent updates to the PKDS through the use of Cross-System Coupling Facility (XCF) signalling services and global (that is, sysplex-wide) ENQs. This solves the problem of PKDS copying into an ICSF-managed data space. If a PKDS record is modified by create, update, or delete operation, the DASD version of the PKDS is updated and the ICSF in-storage copy is updated to reflect the new contents of the record. If a PKDS is shared by one or more members of a sysplex, the ICSF in-storage copy is only

updated on the sysplex system that initiated the record modification. The only way the ICSF in-storage copy will be updated on other systems of the sysplex is:

1. by manual intervention (the operator must use ICSF's TSO panels to initiate a refresh of the in-storage PKDS data) OR
2. by invocation of CSFEUTIL as a batch job or a called program.

See *z/OS Cryptographic Services ICSF Administrator's Guide*, SA22-7521 for more information.

This will now be avoided by using the ICSF sysplex-wide coherency support.

Key Generator Utility Program and key generate callable service

With ICSF, you can use either the key generator utility program (KGUP) or the key generate callable service to generate DES or AES keys.

With KGUP, you can generate key-encrypting keys, PIN keys, data-encrypting keys, data-translation keys, and MAC keys. A master key variant enciphers each type of key that KGUP creates (except for CLRDES and CLRAES keys). After this program generates a key, it stores it in the CKDS where it can be saved and maintained.

The key generate callable service creates all types of DES or AES keys. It generates a single key or a pair of keys. Unlike KGUP, however, the key generate service does not store DES or AES keys in the CKDS but returns them to the application program that called it.

ANSI X9.17 key management callable services

ICSF supports the ANSI X9.17 key management standard, which defines a process for protecting and exchanging DES keys. The ANSI X9.17 standard uses the processes of notarization and offset to create a key identifier for both the sender and the receiver of a key. The key identifier includes a sequence number, and an ASCII-coded origin and destination identifier. The key identifier is cryptographically coupled with the key. In addition to providing callable services that support these functions, ICSF also defines and permits a process of partial notarization. You can use these callable services to develop ANSI X9.17 key management applications that exploit the offset, notarization, and partial notarization processes.

Offsetting combines an ANSI key-encrypting key (AKEK) with a counter by exclusive ORing the two values. The application initializes the counter the first time the key is used and increments the counter with each use. By checking the counter, applications can detect if a message has been transmitted out of sequence and take the appropriate action.

Notarization involves the coupling of an AKEK with ASCII character strings that contain identifiers for both origin and destination, and then offsetting the AKEK with a counter.

Partial notarization involves coupling the AKEK with the origin and destination ASCII character strings without offset. A partially notarized AKEK may be offset at another time to form a fully notarized key.

Restriction: ANSI X9.17 services are only available on the IBM @server zSeries 900.

Composing and decomposing SET blocks

ICSF provides callable services for developing SET applications that make use of the cryptographic hardware at the merchant and acquirer payment gateway. The SET Block Compose callable service performs DES encryption of data, OAEP-formatting through a series of SHA-1 hashing operations, and the RSA-encryption of the Optimal Asymmetric Encryption Padding (OAEP) block. The SET Block Decompose callable service decrypts both the RSA-encrypted and the DES-encrypted data.

Exchanging Secure Sockets Layer session key seed

ICSF provides two callable services that make it possible to exchange the seed key that the SSL application needs to generate session keys. The PKA encrypt callable service encrypts a supplied clear key value under an RSA public key. Currently, this service supports the PKCS 1.2 and ZERO-PAD formats. The PKA decrypt callable service decrypts the supplied key value using the corresponding RSA private key and returns the seed key value to the application in the clear. Currently, this service supports only the PKCS 1.2 format. The SSL application can then use the clear key value to generate symmetric session keys.

Enhanced key management for Crypto Assist instructions

ICSF can generate and build clear DES and AES tokens that can be used in callable services and stored in the cryptographic key data set (CKDS). Clear key tokens on the CKDS can be referenced by labelname by the Symmetric Key Encipher (CSNBSYE and CSNBSYE1) and the Symmetric Key Decipher (CSNBSYD and CSNBSYD1) services. With support for clear DES and AES keys in the CKDS, clear keys do not have to appear in application storage during use, allowing applications to exploit the performance of the CPACF with additional protection for the clear keys.

Restriction: CPACF is not available on the IBM @server zSeries 900.

The DES token support requires a PCIXCC, CEX2C, or CEX3C for key generation (using KGUP) and is not available on the IBM @server zSeries 900. The AES token support requires a CCF, PCIXCC, CEX2C, or CEX3C for key generation (using KGUP) and is available on all systems. To use secure keys in the Symmetric Key Encipher or Symmetric Key Decipher service, a CEX3C is required.

Encrypted key support for Crypto Assist instructions

ICSF will exploit the performance of the CP Assist for Cryptographic Functions using encrypted AES and DES keys stored in the CKDS. Symmetric Key Encipher (CSNBSYE, CSNBSYE1, CSNESYE and CSNESYE1) and Symmetric Key Decipher (CSNBSYD, CSNBSYD1, CSNESYD and CSNESYD1) callable services will accept the label of a encrypted key as the key identifier.

PKCS #11

RSA Laboratories of RSA Security Inc. offers its Public Key Cryptography Standards (PKCS) to developers of computers that use public key and related technology. PKCS #11, also known as Cryptoki, is the cryptographic token interface standard. It specifies an application programming interface (API) to devices, referred to as tokens, that hold cryptographic information and perform cryptographic functions. The PKCS #11 API is an industry-accepted standard commonly used by

cryptographic applications. ICSF supports PKCS #11, providing an alternative to IBM's Common Cryptographic Architecture (CCA) and broadening the scope of cryptographic applications that can make use of zSeries cryptography. PKCS #11 applications developed for other platforms can be recompiled and run on z/OS.

The PKCS #11 standard is defined on the RSA Laboratories Web site at http://www.rsasecurity.com/rsalabs/. This topic describes how ICSF supports that standard. The support includes:

- A token data set (TKDS) that serves as a repository for cryptographic keys and certificates used by PKCS #11 applications
- A C application programming interface (API) that supports a subset of the V2.20 level of the PKCS #11 specification
- Token management callable services. The C API uses these callable services.

Tokens

On most single-user systems a token is a smart card or other plug-installed cryptographic device, accessed through a card reader or slot. The PKCS #11 specification assigns numbers to slots, known as slot IDs. An application identifies the token that it wants to access by specifying the appropriate slot ID. On systems that have multiple slots, it is the application's responsibility to determine which slot to access.

z/OS must support multiple users, each potentially needing a unique keystore. In this multiuser environment, the system does not give users direct access to the cryptographic cards installed as if they were personal smart cards. Instead, z/OS PKCS #11 tokens are virtual, conceptually similar to RACF (SAF) key rings. An application can have one or more z/OS PKCS #11 tokens, depending on its needs.

Typically, PKCS #11 tokens are created in a factory and initialized either before they are installed or upon their first use. In contrast, z/OS PKCS #11 tokens can be created using system software such as RACF, the gskkyman utility, or by applications using the C API. Each token has a unique token name, or label, that is specified by the end user or application at the time that the token is created.

In addition to any tokens your installation may create, ICSF creates a token that is available to all applications. This "omnipresent" token is created by ICSF in order to enable PKCS #11 services when no other token has been created. In this situation, key types and cryptographic mechanisms are available in software. The token label for the omnipresent token is SYSTOK-SESSION-ONLY.

Because PKCS #11 tokens are typically physical hardware devices, the PKCS #11 specification provides no mechanism to delete tokens. However, because z/OS PKCS #11 tokens are virtual, z/OS must provide a way to delete them. To delete a z/OS PKCS #11 token, call C_InitToken with a special label value, $$DELETE-TOKEN$$ (assuming code page IBM1047).

Token Data Set (TKDS)

ICSF stores the PKCS #11 tokens and token objects in a specialized data set called the token data set (TKDS). ICSF maintains both a disk copy and an in-storage copy of the TKDS. This makes it possible to refresh the PKCS #11 tokens and objects without interrupting the application programs. ICSF provides a sample TKDS allocation job (member CSFTKDS) in SYS1.SAMPLIB.

A TKDS is no longer required in order to run PKCS #11 applications. If ICSF is started without a TKDS, however, only the omnipresent token will be available.

Callable services use the in-storage copy of the TKDS. Having the TKDS in storage avoids time-consuming I/O to a data set that is stored on DASD. The dynamic TKDS update callable services permit an application to perform dynamic update of both the disk copy and the in-storage copy of the TKDS.

ICSF supports sysplex-wide consistent updates to the TKDS through the use of Cross-System Coupling Facility (XCF) signalling services and global (that is, sysplex-wide) ENQs. This support maintains the consistency of the in-storage TKDS in a sysplex environment. If a TKDS record is modified by create, update, or delete operations, the DASD version of the TKDS is updated and the ICSF in-storage copy is updated to reflect the new contents of the record for all systems in the ICSF sysplex group.

PKCS #11 and FIPS 140-2

The National Institute of Standards and Technology (NIST), the US federal technology agency that works with industry to develop and apply technology, has published the Federal Information Processing Standard Security Requirements for Cryptographic Modules standard (FIPS 140-2), that can be required by organizations who specify that cryptographic-based security systems are to be used to provide protection for sensitive or valuable data.

The z/OS PKCS #11 services are designed to meet FIPS 140-2 Level 1 criteria, and can be configured to operate in compliance with FIPS 140-2 specifications. Applications that need to comply with the FIPS 140-2 standard can therefore use the z/OS PKCS #11 services in a way that allows only the cryptographic algorithms (including key sizes) approved by the standard and restricts access to the algorithms that are not approved. There are two modes of FIPS-compliant operation:

- The services can be configured so that all z/OS PKCS #11 applications are forced to comply with the FIPS 140-2 standard.
- For installations where only certain z/OS PKCS #11 applications need to comply with the FIPS 140-2 standard, the services can be configured so that only the necessary applications are restricted from using the non-approved algorithms, while other applications are not.

For more information on PKCS #11 and FIPS 140-2 standards, refer to *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*.

Chapter 4. Using ICSF with other cryptographic products

This topic describes how ICSF works with other cryptographic products.

Using IBM's Common Cryptographic Architecture

ICSF provides callable services that comply with IBM's Common Cryptographic Architecture (CCA). This allows application programs written in high-level languages such as C, COBOL, FORTRAN, and PL/I, as well as in Assembler, to be used under more than one cryptographic product.

Another family of products that provide these services is the IBM Transaction Security System. The Transaction Security System includes the IBM 4753 Network Security Processor MVS Support Program (4753-HSP), the IBM 4755 Cryptographic Adapter, and the IBM 4754 Security Interface Unit.

The PCI Cryptographic Coprocessor adds a high security environment to your z/OS and OS/390 server systems for DES and RSA cryptographic functions and sensitive custom applications.

The PCI X Cryptographic Coprocessor adds a high security environment to your z/OS server systems for DES and RSA cryptographic functions and sensitive custom applications.

The Crypto Express2 Coprocessor adds a high security environment to your z/OS server systems for DES, AES and RSA cryptographic functions and sensitive custom applications. The Crypto Express3 Coprocessor adds a high security environment to your z/OS server systems for DES, AES, RSA, and ECC cryptographic functions and sensitive custom applications.

Coexisting with other IBM cryptographic products

ICSF can coexist simultaneously with other IBM cryptographic products within the same operating system image. This protects your installation's investment in programming skills and user applications, and provides a framework for migrating to ICSF.

Running PCF applications under ICSF

If your installation uses PCF, you can run PCF applications on ICSF. Your applications can benefit from the enhanced performance and availability of ICSF. Running PCF applications on ICSF allows you to test ICSF. ICSF also helps you migrate PCF applications. As soon as you can, you should convert these applications to use ICSF callable services rather than the PCF macros. This will permit you to change the master key without interrupting the converted applications.

Restriction: If you are using a PCF application on an IBM @server zSeries 990, z890, z9 EC, or z9 BC, a PCIXCC or CEX2C is required. If you are using a PCF application on an IBM z10 EC or z10 BC, a PCIXCC, CEX2C, or CEX3C is required. If you are using a PCF application on an IBM z196, a CEX3C is required.

ICSF continues to support the PCF macros (GENKEY, RETKEY, EMK, and CIPHER). If an application uses these PCF macros, you can run the application on ICSF. The CIPHER macro will use the DES algorithm on a Cryptographic Coprocessor Feature that is configured for DES. The CIPHER macro will use the

CDMF algorithm on a Cryptographic Coprocessor Feature that is configured for CDMF. If exits exist for either the GENKEY or the RETKEY macro, you should evaluate their applicability to ICSF. If your applications still need these exits, you must rewrite them for ICSF.

You can run PCF applications on systems with ICSF installed. How they run depends on the mode in which ICSF is running. You can run ICSF in any of these modes:

- In **compatibility mode**, you can run PCF applications on ICSF without reassembling them, because ICSF supports the PCF macros.

  You cannot start PCF at the same time as ICSF on the same operating system.
- In **coexistence mode**, you can run a PCF application on PCF, or you can reassemble it to run on ICSF. ICSF provides coexistence macros for this purpose.

  You can start PCF at the same time as ICSF on the same operating system.
- In **noncompatibility mode**, you can run PCF applications only on PCF, and you can run ICSF applications only on ICSF. You cannot run PCF applications on ICSF, because ICSF does not support the PCF macros in this mode.

  You can start PCF at the same time as ICSF on the same operating system.

An application that is running under PCF may use a key in a CKDS managed by PCF. Before you run such an application on ICSF, you should convert the key to an ICSF format. ICSF provides a program to do this conversion.

You should use noncompatibility mode unless you are migrating from PCF to ICSF.

Running 4753-HSP applications under ICSF

You can run both 4753-HSP and ICSF applications on the same z/OS operating system. This allows your installation to take advantage of the flexibility in PIN and key management systems provided by 4753-HSP and the high-volume throughput and bulk data-encryption available with ICSF.

Restriction: If you are using 4753-HSP applications on an IBM @server zSeries 990, z890, z9 EC, or z9 BC, a PCIXCC or CEX2C is required. If you are using 4753-HSP applications on an IBM z10 EC or z10 BC, a PCIXCC, CEX2C, or CEX3C is required. If you are using 4753-HSP applications on an IBM z196, a CEX3C is required.

The key management callable services available in ICSF are identical to the key management *verbs* that are supported by the 4753-HSP. These common extensions beyond the CCA make it possible to develop significant key management applications that run without change on both systems. If your installation currently uses the 4753-HSP, you may be able to run your key storage management applications on ICSF without change if you have used these common verbs.

With the optional PCICC, PCIXCC, CEX2C, or CEX3C, ICSF has the flexibility to route cryptographic functions to the PCICC, PCIXCC, CEX2C, or CEX3C for processing. In OS/390 V2 R10 ICSF, ICSF provides support for callable services which are supported by 4753-HSP (CSNBCPE, CSNBCVG, CSNBCVT, CSNBCVE, CSNBDKM, CSNBEPG, CSNBKTR, CSNBPEX) and enhances existing ICSF callable services to provide suppport for additional key types which are supported by 4753-HSP. This ICSF support will allow many 4753-HSP applications to run without change on ICSF.

There are some restrictions when running both ICSF and the 4753-HSP in the same operating system environment. Although both applications are capable of running in PCF compatibility mode, only one system can provide this service at any time. Because both ICSF and the 4753-HSP support the CCA callable services, applications need to be linked with the appropriate library routines to access the intended service. Note that internal key tokens are not interchangeable between the two products.

There are also some differences between the PKA implementations on z/OS ICSF and the 4753-HSP. The Transaction Security System family of products has two implementations of PKA: PKA92 and PKA96. Applications that are written to one PKA implementation will not run on the other, and techniques that use RSA keys for DEA key distribution are incompatible between the two PKA versions. ICSF supports only the PKA96 and APIs are the same for the services that ICSF and the 4753-HSP have in common, with these exceptions:

- The 4753-HSP does not support the Digital Signature Standard (DSS)

You can exchange RSA digital signatures between the two products if the digital signatures use ISO9796 formatting.

Managing keys with the Distributed Key Management System (DKMS)

The Distributed Key Management System (DKMS) provides online key management to ICSF as well as to IBM cryptographic products on other platforms. DKMS offers centralized key management for symmetric and asymmetric keys and for certificates. DKMS automates the key management process, and exchanges and replaces keys and certificates on demand. Further, to assure continuous operation DKMS maintains backup copies of all critical keys.

With the introduction of Payment Card Industry Data Security Standard (PCI DSS) and other guidelines and requirements key management is much more than a suitable toolbox. DKMS helps enforcing the organization's policies and procedures by offering dual control, split knowledge, audit trail, and key separation between applications and production environments.

The DKMS system is comprised of a workstation that constitutes the user interface and a server component – the DKMS agent - that interacts with ICSF and the backup key repository. The architecture allows for multiple agents, supporting simultaneous management of keys on several servers. The applications get cryptographic support by using ICSF callable services or by utilizing high level DKMS application programming interfaces.

In addition to essential management of symmetric and asymmetric keys, DKMS offers a number of business-focused features to meet specific needs. These features include:

- **Certificate Management**

  DKMS manages certificates stored in RACF Key Rings that are accessible online and certificates used by SSL servers or other connection implementations that must be addressed offline.

  DKMS supports all aspects of RACF Key Ring administration. Key Rings can be created or deleted. Further, new keys can be generated and these keys, together with their certificates, can be attached to one or more key rings. All functions are performed online from the DKMS workstation.

  Many web services and other communication connections rely on a RSA based certificate scheme to assure authenticity and privacy. This scheme requires that

RSA keys and certificates are renewed at regular intervals. The DKMS SSL certificate management feature centralizes and unifies most of the tasks traditionally performed manually for components utilizing SSL or other certificate based schemes. Further, functions are offered that ease administration of certificates for a large population of SSL servers. The DKMS SSL certificate management supports numerous SSL server implementations.

- **EMV support**

  DKMS supports all parties of EMV business: Brand Certificate Authorities (CAs), Integrated Circuit Card (ICC) card issuers, and ICC transaction acquirers.

  – Brand CAs totally support all the security and procedural requirements needed to implement CAs to Visa and MasterCard specifications.

  – ICC card issuers supports generation of the key material to be stored in the EMV ICC card for both the SDA and the DDA/CDA schemes. For the DDA/CDA schemes, DKMS implements a key pre-generation mechanism that utilizes low-load periods of ICSF's cryptographic coprocessors for generation of the huge amount of RSA keys required.

  – ICC transaction acquirers has transaction authorization support for verification of application cryptograms, generation of response cryptograms and secure scripts.

- **Remote Key Loading for ATMs**

  Contemporary ATMs support keys to be exchanged with back-end systems using an RSA key exchange scheme defined in ANS X9.24 part 2. DKMS provides all functions to support this scheme. This includes exchange of keys and certificates with the ATM vendors and APIs that supplies the ATM keys in a format suitable for the ATMs. The DKMS RKL feature supports the major ATM vendors.

For further information please contact the Crypto Competence Center at ccc@dk.ibm.com

Encrypting and decrypting information from other products

ICSF can exchange encrypted information with other cryptographic products. The only limitation is the form of DES encryption used. Some examples:

- MACs: ICSF supports both the ANSI standard X9.9, option 1, and the X9.19 optional double-MAC procedure for generating message authentication codes (MACs). Therefore, if a MAC has been generated with another product that uses either of these standards, ICSF can verify that MAC. ICSF provides support for the use of data-encrypting keys in both the MAC generating and verifying services. This support allows these services to interface more smoothly with non-CCA key distribution systems, including those that follow the ANSI X9.17 protocol.
- PINs: ICSF supports a wide variety of PIN block formats, as is shown in “Using Personal Identification Numbers (PINs) for personal authentication” on page 17.
- Data: ICSF can exchange encrypted data with other products that use the cipher block chaining (CBC) form of the DES and AES algorithms.
- Keys: ICSF can exchange encrypted keys with other products that conform to the IBM's Common Cryptographic Architecture. If you need to exchange keys with systems that do not recognize transport key variants, ICSF enables you to encrypt selected keys under the transport key rather than under the transport key variant. You can use either an application program or KGUP to do this. ICSF can exchange RSA-encrypted data-encrypting keys with systems that format the key according to the PKCS 1.2 Standard. Remote Key Loading can be used to communicate with non-CCA compliant devices.

- Digital Signatures: ICSF can exchange digital signatures with systems that support any of these standards:
  - DSS

    Restriction: Only the IBM @server zSeries 900 support DSS.
  - RSA signatures with MDC, SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 or MD5 hashes and ISO-9796 formatting
  - RSA signatures with MD5 hashes and PKCS 1.0 or PKCS 1.1 formatting

Encryption facility

What is encryption facility?

The need for creating secure archived copies of business data is a critical security concern. Encrypting data that can be recovered at any time offers a high degree of privacy protection from unwanted access. Encryption Facility provides this protection by offering encryption of data for exchange between different systems and platforms and for archiving purposes. It makes use of hardware compression and encryption and relies on a centralized key management based on the z/OS Integrated Cryptographic Service Facility (ICSF) that is highly secure and easy to use.

Encryption Facility makes use of ICSF to perform encryption and decryption and to manage cryptographic keys. To encrypt data files Encryption Facility uses these kinds of cryptographic keys:

- TDES triple-length keys
- 128-bit AES keys

Note: The System z format can use secure symmetric keys.
For information about cryptographic keys, see *z/OS Cryptographic Services ICSF Administrator's Guide* and *z/OS Cryptographic Services ICSF Application Programmer's Guide*.

Features available with encryption facility

Version 1 Release 2.0 of IBM Encryption Facility for z/OS provides these optional features:

Table 1. Features for Encryption Facility

Feature	Description
IBM Encryption Facility for z/OS Encryption Services, called Encryption Services	Support for openPGP and complementary encryption and decryption batch programs that run on z/OS and allow you to encrypt and decrypt data. The algorithms that can be used are: • AES 128, 192 and 256 • 3DES • blowfish
IBM Encryption Facility for z/OS DFSMSdss Encryption, called DFSMSdss Encryption	Services that run on z/OS DFSMSdss and allow you to use DFSMSdss commands to encrypt and decrypt data. With this feature, you can also use DFSMShsm.

For more information about how Encryption Services works see *IBM Encryption Facility for z/OS: User's Guide*.

Virtual Telecommunications Access Method (VTAM) session-level encryption

ICSF supports VTAM session-level encryption, which provides protection for messages within SNA sessions—that is, between pairs of logical units.

When VTAM session- level encryption is in effect, only the originating logical unit can encipher the data, and only the destination logical unit can decipher the data. Thus, the data never appears in the clear while passing through the network.

ICSF places no restrictions on the addressing mode of calling programs. In particular, when VTAM session-level encryption is used with ICSF, VTAM can use storage above 16 megabytes.

For information on setting up VTAM session-level encryption, refer to *VTAM Programming for LU 6.2*.

Access Method Services Cryptographic Option

ICSF supports the Access Method Services Cryptographic Option. The option enables the user of the Access Method Services REPRO command to encipher data by using the Data Encryption Algorithm. The Access Method Services user can use REPRO to encipher data, write it to a data set, and then store the enciphered data set offline. When the user needs the enciphered data set, he or she can retrieve it and use REPRO to decipher it. The user can decipher the data either on the host processor where it was enciphered or on another host processor that contains the Access Method Services Cryptographic Option and the cryptographic key needed.

With the exception of catalogs, all data set organizations that are supported for input by REPRO are eligible as input for enciphering. Similarly, and with the same exception, all data set organizations supported for output by REPRO are eligible as output for deciphering. The resulting enciphered data sets are always sequentially organized (SAM or VSAM entry-sequenced data sets).

Cryptographic keys can either be created by ICSF or be supplied by the Access Method Services user.

Using ICSF with BSAFE

ICSF works in conjunction with RSA Security, Inc.'s BSAFE toolkit (BSAFE 3.1 or later). If you are currently using applications developed with BSAFE, you may want to take advantage of the increased security and performance available with the zSeries and ICSF.

For increased security, you can use ICSF to generate and protect DES cryptographic keys. The keys are encrypted under the DES master key and stored in the Cryptographic Key Data Set. The DES encryption and decryption processes take place within the secure hardware boundary of the Cryptographic Coprocessor Feature.

You can also take advantage of the high transaction rates available with the cryptographic solution to increase the performance of DES encryption and decryption and hashing operations.

Chapter 5. Planning for the Integrated Cryptographic Service Facility

This topic contains guidelines and suggestions to help you plan the installation and operation of ICSF.

System requirements

ICSF is an element of z/OS, but does not always follow the z/OS release schedule. It sometimes provides independent ICSF releases as web deliverables. These are identified by their FMID.

HCR7790 runs on zSeries servers (z800, z900, z890, z990, z9 EC, z9 BC, z10 EC, z10 BC, and z196).

z/OS ICSF FMIDs

These tables explain the relationships of z/OS releases, ICSF FMIDs and servers.

Table 2. z/OS ICSF FMIDs

z/OS and z/OS.e	ICSF FMID[1]	Web deliverable name
V1.11	HCR7751	Cryptographic Support for z/OS V1R8-R10 & z/OS.e V1R8
	HCR7770	Cryptographic Support for z/OS V1R9-R11
	HCR7780	Cryptographic Support for z/OS V1R10-R12
	HCR7790	Cryptographic Support for z/OS V1R11-R13
V1R12	HCR7780	Cryptographic Support for z/OS V1R10-R12
	HCR7790	Cryptographic Support for z/OS V1R11-R13
V1R13	HCR7790	Cryptographic Support for z/OS V1R11-R13

Notes:

1. PTF information can be found in the PSP bucket '2094DEVICE'.

Refer to this chart to determine what release is associated with each ICSF FMID and what server it will run on.

Table 3. FMID and Hardware

ICSF FMID	Applicable z/OS Releases	Servers where FMID will run
HCR7751 (Base of z/OS 1.11)	1.9, 1.10, and 1.11	z800, z900, z890, z990, z9 EC, z9 BC, z10 EC and z10 BC
HCR7770 (Base of z/OS 1.12)	1.9, 1.10, and 1.11	z800, z900, z890, z990, z9 EC, z9 BC, z10 EC and z10 BC
HCR7780 (Base of z/OS 1.13)	1.10, 1.11, and 1.12	z800, z900, z890, z990, z9 EC, z9 BC, z10 EC, z10 BC, and z196.
HCR7790	1.11, 1.12, and 1.13	z800, z900, z890, z990, z9 EC, z9 BC, z10 EC, z10 BC, and z196.

Migration information

The migration topic is covered in detail in *z/OS Cryptographic Services ICSF System Programmer's Guide*.

Cryptographic hardware features

This topic describes the cryptographic hardware features available. Information on adding and removing cryptographic coprocessors can be found in *z/OS Cryptographic Services ICSF Administrator's Guide*.

Crypto Express3 Feature (CEX3C or CEX3A)

The Crypto Express3 Feature is an asynchronous cryptographic coprocessor or accelerator. The feature contains two cryptographic engines that can be independently configured as a coprocessor (CEX3C) or as an accelerator (CEX3A). It is available on the z10 EC, z10 BC, and z196.

Crypto Express2 Feature (CEX2C or CEX2A)

The Crypto Express2 Feature is an asynchronous cryptographic coprocessor or accelerator. The feature contains two cryptographic engines that can be independently configured as a coprocessor (CEX2C) or as an accelerator (CEX2A). It is available on the z9 EC, z9 BC, z10 EC, and z10 BC.

PCI X Cryptographic Coprocessor (PCIXCC)

The PCI X Cryptographic Coprocessor is an asynchronous cryptographic coprocessor. It is a replacement for the Cryptographic Coprocessor Feature and PCI Cryptographic Coprocessor. It is only available on a IBM @server zSeries 990 or IBM @server zSeries 890.

The PCIXCC/CEX2C DES master key is used in place of the CCF DES master key. The asymmetric-keys master key is used in place of the CCF signature and key management master keys. The PCIXCC/CEX2C supports up to 2048-bit RSA keys in all PKA services except SET services (Set Block Compose and Set Block Decompose).

This feature is in the process of being certified for Federal Information Processing Standard (FIPS) 140-2. This includes algorithmic certification under FIPS 46-2 (DES) and FIPS 180-1 (Secure Hash Standard).

CP Assist for Cryptographic Functions (CPACF)

CPACF is a set of cryptographic instructions available on all CPs of z990, z890, z9 EC, z9 BC, z10 EC, z10 BC, and z196. Use of the CPACF instructions provides improved performance. The SHA-1 algorithm is always available. Additionally, SHA-224 and SHA-256 algorithms are available on the z9 EC and z9 BC. Additionally, SHA-384 and SHA-512 algorithms are available on IBM System z10 Enterprise Class and IBM System z10 Business Class.

CP Assist for Cryptographic Functions (CPACF) DES/TDES Enablement, feature 3863, provides for clear key DES and TDES instructions. On the z9 EC and z9 BC, this feature includes clear key AES for 128-bit keys. On z10 EC, z10 BC, and z196, this feature also includes clear key AES for 192-bit and 256-bit keys.

If you want to include a PCIXCC, CEX2C, PCICA (z990, z890) Crypto Express2 feature (z9 EC, z9 BC, z10 EC, z10 BC), or Crypto Express3 Coprocessor (z10 EC, z10 BC, or z196), then feature 3863 is required.

PCI Cryptographic Accelerator (PCICA)

On all systems, the PCI Cryptographic Accelerator provides support for clear keys in the CSNDPKD callable services for better performance than when executed in a cryptographic coprocessor. On z990 or z890, it also supports CSNDDSV and CSNDPKE.

PCICAs enable maximum SSL performance.

Cryptographic Coprocessor Feature (CCF)

The Cryptographic Coprocessor Feature (CCF) can have up to two cryptographic coprocessors as high-speed extensions of the central processor. Each CCF contains both DES and PKA cryptographic processing units. You can configure the processor complex to run in either single-image mode or logical partition mode.

If the Cryptographic Coprocessor Feature is in single-image mode, the same master keys must be installed on both CCFs. If you bring a second coprocessor online, ICSF verifies that the master keys are the same. If the DES master keys are different, ICSF will not use the second coprocessor. The PKA master keys must be the same on both Coprocessors in order to enable the PKA services.

This feature is currently certified for Federal Information Processing Standard (FIPS) 140-1 level 4. This includes algorithmic certification under FIPS 46-2 (DES), FIPS 180-1 (Secure Hash Standard), and FIPS 186 (Digital Signature Standard).

The possible configurations include:

- DES with PKA

  These servers are configured for full 64-bit DES keys (effective length 56 bits), 1024-bit PKA keys for DES key distribution, and 1024-bit PKA signature keys.
- Triple DES with PKA

  These servers are configured for 192-bit DES keys (effective length 169 bits), 1024-bit PKA keys for DES key distribution, and 1024-bit PKA signature keys. This configuration is available on S/390 G5 Enterprise Servers and higher.

PCI Cryptographic Coprocessor (PCICC)

The PCI Cryptographic Coprocessor, which works in conjunction with the Cryptographic Coprocessor Feature, provides the capability of generating and retaining RSA keys in secure hardware. This capability meets a requirement to become a SET Certificate Authority. A PCI Cryptographic Coprocessor is required on a CCF system for:

- UDX capability
- Generating RSA public and private keys
- The retained key list and retain key delete callable service.

The PCICC cards are in addition to the Cryptographic Coprocessor Feature. In order for the PCI Cryptographic Coprocessor to operate, the verification pattern for the SYM-MK master key must match the verification pattern of the DES master key on the server's Cryptographic Coprocessor Feature. Before you can use the PKA services of the PCI Cryptographic Coprocessor, you must install both the KMMK and the SMK on the Cryptographic Coprocessor Feature and the ASYM-MK master key on the PCI Cryptographic Coprocessor. The hash pattern of the ASYM-MK master key must match the hash pattern of the SMK in order to use the PCI Cryptographic Coprocessor.

Note: For new installations, it is recommended that the installation enter the KMMK equal to the SMK master key. Existing customers should reencipher their PKDS and migrate to a system with the KMMK equal to the SMK.

Performance considerations

Customers migrating from a machine that uses the Cryptographic Coprocessor Facility (CCF on z900 and earlier) should be aware of potential performance

differences when doing cryptography on later systems. Some of the functions supported on the CCF (which is part of the MCM) have been moved to the Crypto Express2 or Crypto Express3 which are PCI cards installed in the I/O cage. A limited set of functions from the CCF are now available on the CPACF which will provide better performance than the PCI cards.

For example, the secure key encryption and decryption APIs (CSNBENC/1, CSNBDEC/1) and PIN and MAC functions are now supported on the PCI cards and depending on how the APIs are invoked (blocksizes, application logic, parameters) and what is measured (CPU vs. wall-clock time), response time may be significantly slower than on the CCF.

The Symmetric Key Encipher and Symmetric Key Decipher callable services are supported on the CPACF and performance should be better than the CCF. RSA Private Key operations are executed on the Crypto Express2 or Crypto Express3 card and are also expected to perform better.

Performance for the hashing APIs will be about the same for APIs handled in software (MD5 and RIPEMD-160), and since SHA-1 is now done on the CPACF, it will be faster than on a CCF.

There are two ways to improve performance for APIs that are slower on the z890, z990, z9 EC, z9 BC, z10 EC, z10 BC, or z196. First, using the Clear Key APIs which run on the CPACF will greatly improve performance versus the Secure Key APIs, but is appropriate ONLY if the customer's encryption environment allows the use of Clear Key APIs. Second, the customer may need to modify their application to use multi-threading to take advantage of the multiprocessing capability of the crypto cards on the Crypto Express2 or Crypto Express3. If their application is single threaded using only one CCF then the application should be modified to use multi-threading and take advantage of the multiple crypto engines to drive workload and increase the throughput. Depending on the complexity of the application, this may not be a trivial change. In addition, large blocking tends to be more efficient, and the customer should also consider using large blocksizes when using the secure key encryption and decryption APIs.

Finally, for SSL, if the SSL volume was low, and the customer was using only software in a CCF environment, then they may be able to use only software on the z890/z990 or later servers. In this enviornment, they will see a slight improvement in performance when using the clear key encryption/decryption APIs mentioned previously on the CPACF for the record level component of SSL, that improvement would be overwhelmed by the slower performance of the handshakes in software. If the customer required a PCICC or PCICA to support SSL volumes, then they will likely require a Crypto Express2 or Crypto Express3 on the z890/z990. If their SSL volume was being handled only on the CCF (no PCICC, PCICA and too much volume for software alone), then they should consider using a Crypto Express2 or Crypto Express3 in the z890/z990 environment. If the customer had a PCICA installed to support SSL volumes on their current machine, adding Crypto Express2 or Crypto Express3 is now required.

Servers

This topic describes the servers on which the cryptographic hardware features are available.

IBM zEnterprise 196 (z196)

The z196 provides constraint relief and addresses various customer demands. It has several cryptographic features.

- **CP Assist for Cryptographic Functions** is implemented on every processor. SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512 secure hashing is directly available to application programs.
- Feature code 3863, **CP Assist for Cryptographic Functions (CPACF) DES/TDES Enablement** – enables clear key DES and TDES instructions on all CPs. AES 128-bit, AES 192-bit and AES 256-bit support is also available.
- Feature code 0864, **Crypto Express3 Feature** – optional, and only available if you have feature 3863, CPACF DES/TDES Enablement installed. The z196 can support a maximum of 8 features. Each feature code has two coprocessors/accelerators.

IBM System z10 Enterprise Class and IBM System z10 Business Class (z10 BC)

The IBM System z10 Enterprise Class and IBM System z10 Business Class provide constraint relief and addresses various customer demands. It has several cryptographic features.

- **CP Assist for Cryptographic Functions** is implemented on every processor. SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512 secure hashing is directly available to application programs.
- Feature code 3863, **CP Assist for Cryptographic Functions (CPACF) DES/TDES Enablement** – enables clear key DES and TDES instructions on all CPs. AES 128-bit, AES 192-bit and AES 256-bit support is also available.
- Feature code 0863, **Crypto Express2 Feature** – optional, and only available if you have feature 3863, CPACF DES/TDES Enablement installed. The z10 EC and z10 BC can support a maximum of 8 features. Each feature code has two coprocessors/accelerators.
- Feature code 0864, **Crypto Express3 Feature** – optional, and only available if you have feature 3863, CPACF DES/TDES Enablement installed. The z10 EC and z10 BC can support a maximum of 8 features. Each feature code has two coprocessors/accelerators.

IBM System z9 Business Class (z9 BC)

The IBM System z9 BC provides constraint relief and addresses various customer demands. It has several cryptographic features.

- **CP Assist for Cryptographic Functions** is implemented on every processor. SHA-1, SHA-224 and SHA-256 secure hashing is directly available to application programs.
- Feature code 3863, **CP Assist for Cryptographic Functions (CPACF) DES/TDES Enablement** – enables clear key DES and TDES instructions on all CPs. In addition, ICSF supports hardware implementation of AES 128-bit keys and software implementation of AES 192-bit and AES 256-bit key lengths.
- Feature code 0863, **Crypto Express2 Feature** – optional, and only available if you have feature 3863, CPACF DES/TDES Enablement installed. The IBM System z9 BC can support a maximum of 8 features. Each feature code has two coprocessors/accelerators.

IBM System z9 Enterprise Class (z9 EC)

The IBM System z9 EC provides constraint relief and addresses various customer demands. It has several cryptographic features.

- **CP Assist for Cryptographic Functions** is implemented on every processor. SHA-1, SHA-224 and SHA-256 secure hashing is directly available to application programs.

- Feature code 3863, **CP Assist for Cryptographic Functions (CPACF) DES/TDES Enablement** – enables clear key DES and TDES instructions on all CPs. In addition, ICSF supports hardware implementation of AES 128-bit keys and software implementation of AES 192-bit and AES 256-bit key lengths.
- Feature code 0863, **Crypto Express2 Feature** – optional, and only available if you have feature 3863, CPACF DES/TDES Enablement installed. The IBM System z9 EC can support a maximum of 8 features. Each feature code has two coprocessors/accelerators.

IBM @server zSeries 990 (z990)

The **IBM @server zSeries 990** provides constraint relief and addresses various customer demands. It has several cryptographic features.

- **CP Assist for Cryptographic Functions** is implemented on every processor. SHA-1 secure hashing is directly available to application programs.
- Feature code 3863, **CP Assist for Cryptographic Functions (CPACF) DES/TDES Enablement** – enables clear key DES and TDES instructions on all CPs. In addition, ICSF supports software implementation of AES.
- Feature code 0862, **PCI Cryptographic Accelerator** – optional, and only available if you have feature 3863, CPACF DES/TDES Enablement installed. The IBM @server zSeries 990 can support a maximum of 12 PCI Cryptographic Accelerators. Each feature code has two coprocessors.
- Feature code 0868, **PCI X Cryptographic Coprocessor** – optional, and only available if you have feature 3863, CPACF DES/TDES Enablement installed. The IBM @server zSeries 990 can support a maximum of 4 PCIXCCs. Each feature code has one coprocessor.
- Feature code 0863, **Crypto Express2 Coprocessor** – optional, and only available if you have feature 3863, CPACF DES/TDES Enablement installed. The IBM @server zSeries 990 can support a maximum of 16 CEX2Cs. Each feature code has two coprocessors.

Note: You can have a maximum of 6 PCICA features (12 cards), 4 PCIXCC features (4 cards) and 16 CEX2Cs (8 features), with maximum number of 8 installed features.

IBM @server zSeries 890 (z890)

The **IBM @server zSeries 890** provides constraint relief and addresses various customer demands. It has several cryptographic features.

- **CP Assist for Cryptographic Functions** are implemented on every processor. SHA-1 secure hashing is directly available to application programs.
- Feature code 3863, **CP Assist for Cryptographic Functions (CPACF) DES/TDES Enablement** – enables clear key DES and TDES instructions on all CPs. In addition, ICSF supports software implementation of AES.
- Feature code 0862, **PCI Cryptographic Accelerator** – optional, and only available if you have feature 3863, CPACF DES/TDES Enablement installed. The IBM @server zSeries 890 can support a maximum of 12 PCI Cryptographic Accelerators. Each feature code has two coprocessors.
- Feature code 0868, **PCI X Cryptographic Coprocessor** – optional, and only available if you have feature 3863, CPACF DES/TDES Enablement installed. The IBM @server zSeries 890 can support a maximum of 4 PCIXCCs. Each feature code has one coprocessor.

- Feature code 0863, **Crypto Express2 Coprocessor** – optional, and only available if you have feature 3863, CPACF DES/TDES Enablement installed. The IBM @server zSeries 890 can support a maximum of 16 CEX2Cs. Each feature code has two coprocessors.

Note: You can have a maximum of 6 PCICA features (12 cards), 4 PCIXCC features (4 cards) and 16 CEX2Cs (8 features), with maximum number of 8 installed features.

IBM @server zSeries 900 (z900) — Feature Code 800

You can enable these features on this server:

- **Cryptographic Coprocessor Feature** – one or two cryptographic coprocessors protected by tamper-detection circuitry and a cryptographic battery unit.
- Feature code 0861, **PCI Cryptographic Coprocessor** (PCICC) – based on the 4758 model 2 standard PCI-bus card package. You must have at least one Cryptographic Coprocessor Feature on your system with a PCICC. Note that each feature has two coprocessors.
- Feature code 0862, **PCI Cryptographic Accelerator** (PCICA). You must have at least one Cryptographic Coprocessor Feature on your system with a PCICA. Note that each feature has two coprocessors.

Note: The IBM @server zSeries 900 can support a combination of PCI Cryptographic Coprocessors (maximum of 16) or PCI Cryptographic Accelerators (maximum of 12), but the total must not exceed 16.

IBM @server zSeries 800 (z800) — Feature Code 800

These features are available on this server:

- **Cryptographic Coprocessor Feature** (CCF) – one or two cryptographic coprocessors protected by tamper-detection circuitry and a cryptographic battery unit.
- Feature code 0861, **PCI Cryptographic Coprocessor** (PCICC) – based on the 4758 model 2 standard PCI-bus card package. You must have at least one Cryptographic Coprocessor Feature on your system with a PCICC. Note that each feature has two coprocessors.
- Feature code 0862, **PCI Cryptographic Accelerator** (PCICA). You must have at least one Cryptographic Coprocessor Feature on your system with a PCICA. Note that each feature code has two coprocessors.

Note: The IBM @server zSeries 800 can support a combination of PCI Cryptographic Coprocessors (maximum of 16) or PCI Cryptographic Accelerators (maximum of 12), but the total must not exceed 16.

Configurations by server

This topic describes some of the different configurations available with the various servers.

Configuring the IBM @server zSeries 990, IBM @server zSeries 890, z9 EC, z9 BC, z10 EC, z10 BC, and z196

There is only LPAR mode on a z990, z890, z9 EC, z9 BC, z10 EC, z10 BC, and z196. You can divide your processor complex into PR/SM logical partitions. When you create logical partitions on your processor complex, you use the usage domain index on the Support Element Customize Image Profile page only if you have (or plan to add) PCICAs, PCIXCCs, CEX2Cs, or CEX3Cs.

The DOMAIN parameter in the ICSF installation options data set is optional. The number that is specified for the usage domain index must correspond to the domain number you specified with the DOMAIN(n) keyword in the installation options data set – if you specified one. The DOMAIN keyword is required if more than one domain is specified as the usage domain on the PR/SM panels.

All cryptographic features can be configured and shared across multiple partitions.

There is support for up to 30 LPARs. On previous systems, where the maximum number of LPARs was 16, a domain was unique to an LPAR.

Note: The domain assigned to the TKE Host LPAR must be unique if TKE is to control all the PCIXCC, CEX2C, or CEX3C cards in the environment. No other LPAR can use the domain assigned to the TKE Host.

With more than 16 LPARs to support, the domain may not be unique across LPARs but the same domain may be assigned to different LPARs if they are accessing different PCICAs. This is illustrated by LPAR 1 and LPAR 3 in Figure 11. They are both assigned to usage domain 0 but on two different PCICAs.

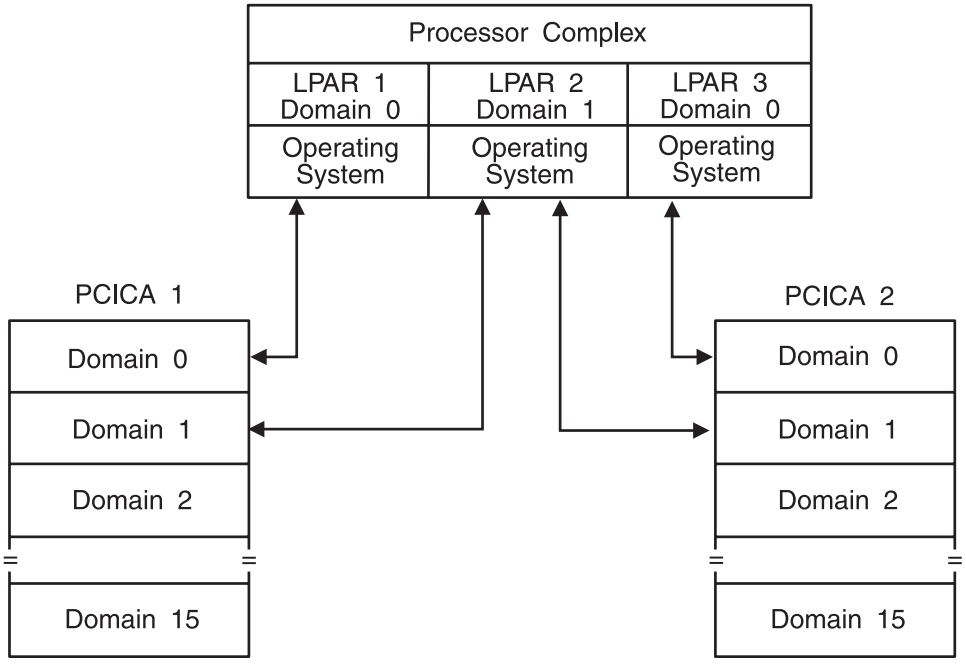

Figure 11. Two Crypto PCICAs on a Processor Complex Running in LPAR Mode

The example in Figure 11 shows that LPAR 2 has assigned access to Domain 1 on both PCICA 1 and PCICA 2. If you were to add another PCICA or PCIXCC to LPAR 2, Domain 1 on the new PCICA/PCIXCC would also be assigned.

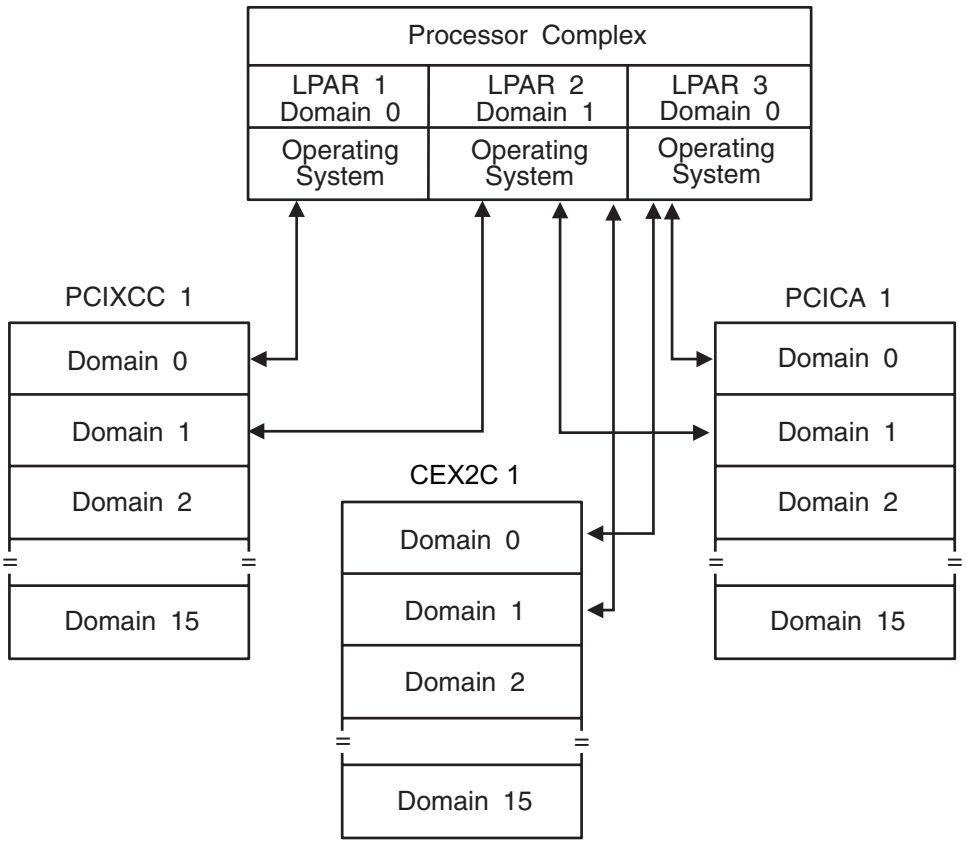

Figure 12. Multiple Crypto Coprocessors on a Complex Running in LPAR Mode

The example in Figure 12 shows that LPAR 2 has assigned access to Domain 1 on PCIXCC 1, PCIXCC 2, and PCICA 1. LPAR 3 has assigned access to Domain 0 on PCIXCC2 and PCICA 1.

Configuring the IBM @server zSeries 900

The Cryptographic Coprocessor Feature can include up to two cryptographic coprocessors, each of which is attached to a central processor within the central processor complex. Each cryptographic coprocessor has sixteen master key register sets, referred to as domains. ICSF uses the domain usage index to access each domain. You can configure the complex to run in one of these modes:

- Single image mode
- Logical partition mode

Single image mode

In single image mode, the processor complex has access to the same domain on both Crypto CP 0 and Crypto CP 4. The domain is specified in the installation options data set. Beginning in z/OS V1 R2, the DOMAIN parameter is optional. It is required if more than one domain is specified as the usage domain on the PR/SM panels or if running in native mode. See *z/OS Cryptographic Services ICSF System Programmer's Guide* for additional information on the DOMAIN parameter. The accessed domain on both coprocessors must have the same master key. Figure 13 on page 62 shows an example single image mode configuration.

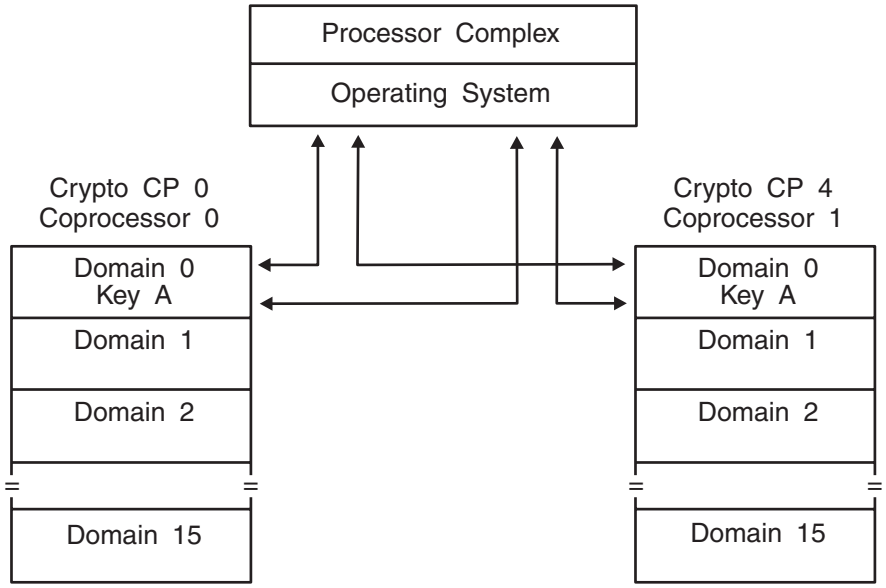

Figure 13. Two Crypto CPs on a Processor Complex Running in Single Image Mode

Logical Partition (LPAR) mode

You can divide your processor complex into PR/SM logical partitions (LPARs). When you create logical partitions on your processor complex, you use the usage domain index on the Support Element Customize Image Profile page to enable access to a Crypto CP domain. The number that is specified for the usage domain index must correspond to the domain number you specify with the DOMAIN(n) keyword in the installation options data set. Beginning in z/OS V1 R2, the DOMAIN parameter is optional. It is required if more than one domain is specified as the usage domain on the PR/SM panels or if running in native mode. See the DOMAIN parameter in *z/OS Cryptographic Services ICSF System Programmer's Guide* for additional information.

You can assign more than one domain to an LPAR, but you must use a unique installation options data set to define each domain. Assigning more than one domain to an LPAR makes it possible to have separate master keys for different purposes. For example, you might have one master key for production operations and a different master key for test operations.

The PCI Cryptographic Coprocessor can be configured like a Cryptographic Coprocessor Feature. It can be dedicated or shared across multiple partitions with each card supporting up to 16 domains.

The PCI Cryptographic Accelerator can be configured like a Cryptographic Coprocessor Feature. It can be dedicated or shared across multiple partitions with each card supporting up to 16 domains.

When using logical partitions, there is no domain sharing unless TKE is being used. The 'HOST' LPAR can control the domains of the other LPARS if the control domain for the first LPAR is setup for it. The example in Figure 14 on page 63 shows that LPAR 1 has access to Domain 0 on Crypto CP 0, Crypto CP 1, and PCICC. LPAR 2 has access to Domain 1 and Domain 2 on both Crypto CPs and on the PCICC. LPAR 1 cannot access Domain 1 or Domain 2 on the PCICC or on either of the

Crypto CPs. Likewise, LPAR 2 cannot access Domain 0 on either Crypto CP or the PCICC. The ICSF system running on the operating system in LPAR 2 has access to only one domain at any particular time, as specified in the installation options data set.

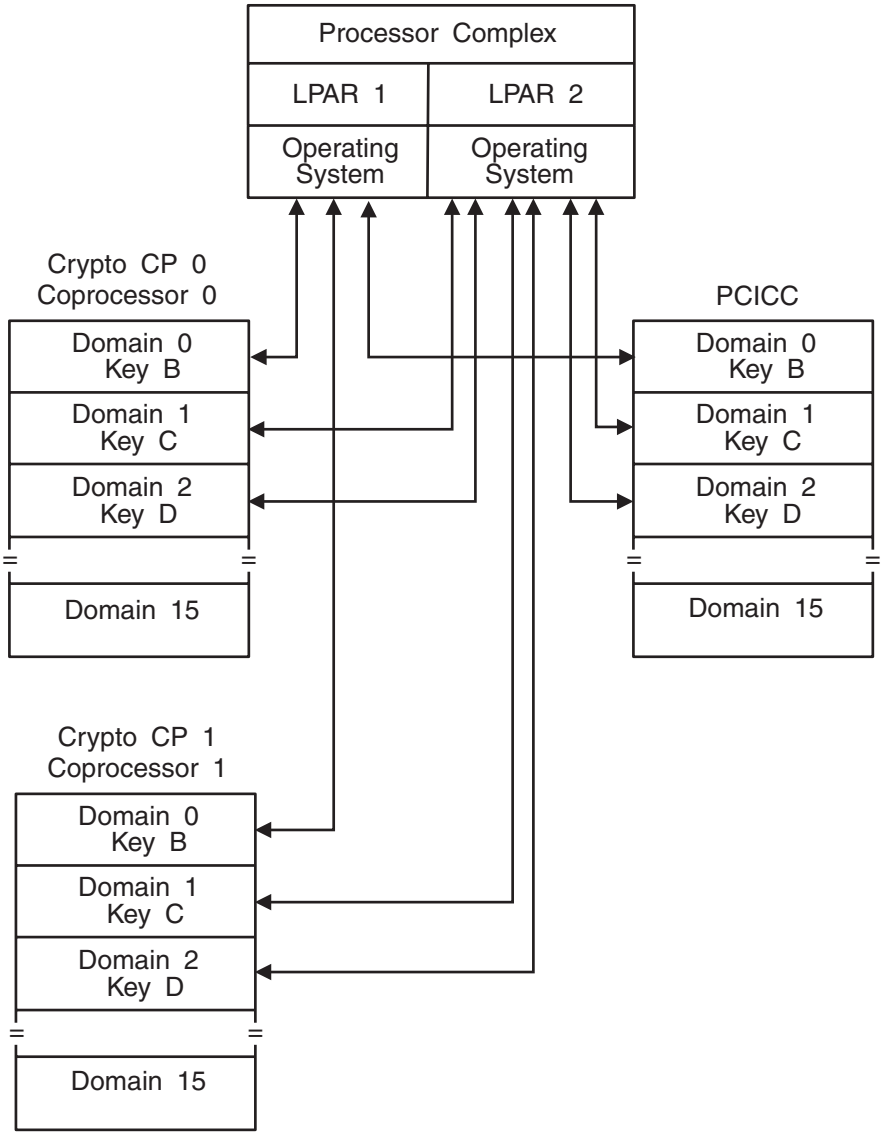

Figure 14. Two Crypto Coprocessors and one PCICC on a Processor Complex Running in LPAR Mode

Hardware features by server

Configuration of the cryptographic features is dependent on U.S. Export controls. Common configurations are:

- A. **CCF only**

  IBM @server zSeries 900 configured for full DES with PKA or configured for DES with exportable PKA.
- B. **CCF with PCICC**

  IBM @server zSeries 900 configured for full DES with PCI Cryptographic Coprocessor.
- C. **PCIXCC/CEX2C**

IBM @server zSeries 990 and later with CP Assist for Cryptographic Functions DES/TDES Enablement (feature 3863) and PCI X Cryptographic Coprocessor or Crypto Express2 Coprocessor (feature 0868).

For a complete list of callable services and the hardware configurations that support them, refer to Appendix B, "Summary of callable service support by hardware configuration," on page 71.

Security

In reviewing your installation security plan before installing ICSF, consider these points:

- **Controlling Access to Disk Copies of the CKDS**

  You should determine which users and applications should have access to each copy of the CKDS on your system.

  Note: The in-storage copy of the CKDS can be accessed only through ICSF functions such as callable services, KGUP, or the ICSF panels. To protect the in-storage copy of the CKDS, control who can use these services.

- **Controlling Access to the PKDS**

  You should determine which users and applications should have access to the PKDS on your system.

  Note: The in-storage copy of the PKDS can be accessed only through ICSF functions such as callable services or the ICSF panels. To protect the in-storage copy of the PKDS, control who can use these services.

- **Controlling Access to the Key Generator Utility Program (KGUP)**

  Anyone who is running the KGUP can read and change an unprotected CKDS. To prevent unauthorized persons from using the KGUP, store the program in an APF-authorized library that is protected by the Security Server (RACF). KGUP is also protected by CSFSERV(CSFKGUP).

- **Controlling Access to Services and Keys**

  Users of the Security Server (RACF) can use the CSFSERV and CSFKEYS classes to perform access checking and auditing of services and keys, respectively. The CSFSERV class also protects some critical administrative TSO panels, such as changing the master key and refreshing the CKDS. The audit records that are produced by these routines are SMF type 80 records.

  You can also define profiles in the XFACILIT class to establish a Key Store Policy. Each profile you define in the XFACILIT class is a separate Key Store Policy control. Together, these profiles define your overall Key Store Policy. By establishing a Key Store Policy, you can control access to secure symmetric keys in the CKDS and asymmetric keys in the PKDS. A Key Store Policy can also specify how keys in a PKDS or CKDS can be used. By enabling Key Store Policy controls, you can:

  – have ICSF verify, when an application passes a callable service a key token instead of a key label, that the user has authority to the secure token. ICSF does this be identifying key labels associated with the key token, so that a SAF authorization check (which depends on key labels) can be carried out against profiles in the CSFKEYS class.
  – prevent applications from storing duplicate tokens in a CKDS or PKDS.
  – raise the level of access authority required to create, write to, or delete a key label.

– raise the level of access authority required to export a symmetric key (transfer it from encryption under a master key to encryption under an application-supplied RSA public key) when an application calls the Symmetric Key Export callable service (CSNDSYX or CSNFSYX). In this case, a SAF authorization check is performed against profiles in the XCSFKEY class rather than the CSFKEYS class.
– Set additional restrictions on how keys can be used. These additional restrictions are specified in the ICSF segment of CSFKEYS (or XCSFKEY) profiles. Using the ICSF segment of profiles in these classes, you can:
  - specify that asymmetric keys covered by the profile can not be used for secure export or import operations.
  - specify that asymmetric keys covered by the profile can not be used in the handshake operations performed by the following callable services:
    • Digital Signature Generate (CSNDDSG and CSNFDSG)
    • Digital Signature Verify (CSNDDSV and CSNFDSV)
    • PKA Encrypt (CSNDPKE and CSNFPKE)
    • PKA Decrypt (CSNDPKD and CSNFPKD)
  - specify whether symmetric keys covered by the profile can be exported using the Symmetric Key Export callable service (CSNDSYX or CSNFSYX). If allowing the symmetric keys covered by the profile to be exported, you can specify which asymmetric keys can be used to perform the export operation. You can specify this by supplying a list of labels of RSA keys in the PKDS, or a list of certificates in either a PKCS #11 token, or a SAF key ring.

You should familiarize yourself with the controls you can enable and decide on the Key Store Policy that is best for your installation.

• **Enforcing PIN block restrictions**

Requirements in the ANSI X9.8 PIN standard intended to guard against PIN block attacks can be implemented by enabling certain access control points on the Crypto Express3 Coprocessor using the optional Trusted Key Entry (TKE) workstation. These access control points are:

– ANSI X9.8 PIN - Enforce PIN block restrictions
– ANSI X9.8 PIN - Allow modification of PAN
– ANSI X9.8 PIN - Allow only ANSI PIN blocks

When enabled, these access control points limit the capabilities of the following callable services:

– Clear PIN Generate Alternate (CSNBCPA and CSNECPA)
– Encrypted PIN Translate (CSNBPTR and CSNEPTR)
– Secure Messaging for PINs (CSNBSPN and CSNESPN)

• **Scheduling Changes for Cryptographic Keys**

To reduce the possibility of exposing a key value, you may want to change the value of cryptographic keys, including master keys, from time to time:

– You can use the ICSF panels to change the DES, AES and PKA master keys.
– If you have an optional Trusted Key Entry (TKE) workstation installed, you can use it to change DES, AES and PKA master keys on all cryptographic coprocessors.
– You can use KGUP or the ICSF panel to change the CKDS.
– You can develop applications that use the dynamic CKDS update callable services to change both the in-storage and DASD copies of the CKDS.

You can perform all of these operations without interrupting cryptographic functions.

- **Allowing or Preventing Clear Cryptographic Keys**

  With ICSF, keys exist in the clear only in these cases:

  – if you specifically allow special secure mode *and* actually set special secure mode during operations, applications can use the secure key import callable service, the clear PIN generate callable service, the clear PIN generate alternate callable service and the multiple secure key import and symmetric key generate with the IM keyword. On a system with a PCIXCC/CEX2C/CEX3C, access control points for these services must be enabled.

  – If ICSF is not in special secure mode, most keys in the system are encrypted except DATA keys that a user may enter through the use of the clear key import callable service. CLRAES and CLRDES keys are also not encrypted.

    Note: The clear key import callable service is equivalent to the PCF EMK macro.

  – The encode callable service can use a clear key to encipher data.

  – If you use the Master Key Entry panels to enter the key parts of a master key, the key parts appear briefly in the clear in host storage.

  – When an application calls the symmetric key generate callable service to generate a DATA key, the DATA key appears briefly in the clear in host storage when executed on a CCF. The DATA key is then quickly encrypted under the DES master key and the RSA public key.

  – When an application calls the symmetric key export callable service to transfer a DATA key from encryption under the host DES master key to encryption under an RSA public key, the DATA key appears briefly in the clear in host storage when executed on a CCF.

  – When an application calls the SET block compose and SET block decompose callable services, the DATA key exists briefly in the clear in host storage when executed on a CCF.

  – On the z890, z990, z9 EC, z9 BC, z10 EC, z10 BC, and z196, clear keys are used to provide improved performance for the DES, TDES and AES algorithms. Symmetric key encrypt and decrypt services (CSNBSYD and CSNBSYE) are available on all CP's.

  On a CCF system, these services will be routed to the PCI Cryptographic Coprocessor if one is available: CSNDSYG, CSNDSYX, CSNDSBC, CSNDSYI, CSNDSBD, CSNBSKY, and CSNBSPN. If no PCI Cryptographic Coprocessor is available, then keys will appear briefly in the clear as stated previously.

- **Sending Cryptographic Keys to Other Installations**

  To eliminate the need to have a courier deliver clear keys between installations, you can use either or both of these options:

  – DES transport keys to encrypt keys for network distribution

  – The receiving installation's RSA public key to encrypt a DES or AES DATA key prior to electronic distribution

  Both of these methods make key distribution more secure.

- **Controlling access to the Disk Copies**

  You should determine which users and applications should have access to each DASD copy of the CKDS, PKDS and TKDS on your system.

- **SMF Records Generated by ICSF**

  ICSF generates type 82 records in the SMF data set when these conditions occur:

- ICSF starts
- ICSF status changes on a processor
- When you enable or disable special secure mode
- When you enter a clear master key part to the Cryptographic Coprocessor Feature through the use of the ICSF panels
- You enter a master key part
- When the in-storage CKDS is refreshed
- When the in-storage PKDS is refreshed
- You use the ICSF panels to process an operational key part or key part register loaded using the TKE workstation
- When an application uses any of the dynamic services that write to the CKDS
- When ICSF handles error conditions or tampering
- When you issue a command from the TKE workstation to the Cryptographic Coprocessor Feature
- When an application uses any of the dynamic services that write to the PKDS
- When you use the Master Key Entry panels to enter a master key in the PCICC, PCIXCC, CEX2C, or CEX3C
- When you create or delete a retained key on a PCICC, PCIXCC, CEX2C, or CEX3C
- When you use the TKE workstation to communicate with the PCICC, PCIXCC, CEX2C, or CEX3C
- To capture measurements of timing and configuration for the PCICC, PCIXCC, CEX2C, CEX3C, CEX2A, CEX3A, or PCICA
- When ICSF issues IXCJOIN or IXCLEAVE to join or leave the ICSF sysplex group.
- When you use the Trusted Block Create callable service to create or activate a trusted block.
- When you use the PKCS #11 token management callable services to create or delete a token or object or to modify an attribute value of an object
- When the security administrator has indicated that duplicate key tokens must be identified
- When a callable service checks the key store policy

The SMF record type 82 subtype indicates the type of event that caused ICSF to write the SMF record. For subtypes that log state changes, the SMF record will contain additional audit information about the server user and, optionally, the end user.

You can also use the Security Server (RACF) or an equivalent product to record attempts to use protected cryptographic keys or functions.

- **Recording and Formatting type 82 SMF Records in a Report**

  Sample jobs are available (in SYS1.SAMPLIB) to assist in the recording and formatting of type 82 SMF data:

  - **CSFSMFJ** - JCL that executes the code to dump and format SMF type 82 records for ICSF. Before executing the JOB step, you need to make modifications to the JCL (see the prologue in the sample for specific instructions). After the JCL has been modified, terminate SMF recording of the currently active dump dataset (by issuing I SMF) to allow for the unloading of SMF records. After SMF recording has been terminated, execute the JCL. The output goes into the held queue.
  - **CSFSMFR** - An EXEC that formats the SMF records into a readable report.

- **Recording and Formatting type 80 SMF Records in a Report**

  RACF provides support to log type 80 SMF records when a user attempts to access an ICSF service, utility, or key label when a profile is defined for the service, utility or label. See the *z/OS Security Server RACF Auditor's Guide* for guidance on how to activate this logging and to format the type 80 SMF records.

Operating considerations

Before operating a computing system that has ICSF installed, you should consider certain items.

ICSF initialization options

Your system operator can use the START and STOP operator commands to start and stop ICSF. Also, your system programmer can set up different sets of options such as a PARMLIB member that the operator can specify on the START command. This enables you to set ICSF up to run differently at different times. For more information, see “Using options to tailor ICSF” on page 23.

Effect of multiple records on performance

If you add more than 10,000 records to a CKDS, and Local Shared Resource (LSR) is installed on your system, you should consider using the batch LSR subsystem with the VSAM deferred-write option. You should also plan to do sequential additions rather than insertions. This can greatly improve the performance of this operation.

LPAR considerations

You can divide your processor complex into PR/SM logical partitions (LPARs) by assigning crypto CP master key registers or domains to each LPAR. When running in LPAR mode, use system symbols in the installation options data set to define different domains. You can assign one or more domains to an LPAR.

The DOMAIN parameter is an optional parameter in the installation options data set. It is required if more than one domain is specified as the usage domain on the PR/SM panels or if running in native mode. If you assign multiple domains to an LPAR, you can have separate master keys for different purposes. For instance, you might have one master key for production operations and another master key for test operations.

With z890, z990, z9 EC, z9 BC, z10 EC, z10 BC, and z196 servers, you are able to use the same domain in more than one LPAR as long as you're not sharing the same PCIXCC, CEX2C, or CEX3C cards.

Link Pack Area (LPA) considerations

ICSF uses dynamic LPA to load the pre-PC routines, CICS-related routines, and other modules into above-the-line ECSA. The dynamic LPA load will occur the first time that ICSF is started within an IPL, and the modules will persist across subsequent restarts of ICSF.

Appendix A. Standards

The Cryptographic Coprocessor Feature, PCI Cryptographic Coprocessor, and ICSF provide support for these International and USA standards (at least in part):

ISO 8730
Banking—Requirements for Standard Message Authentication (Wholesale)

ISO 8731
Banking—Approved Algorithms for Message Authentication—Part 1: DES-1 algorithm

ISO 8732
Information Processing: Modes of Operation for a 64-bit Cipher Algorithm

ISO 9564
Personal Identification Number Management and Security Part 1—PIN Protection Principles and Technique

ISO 9796
Information Technology — Security Techniques — Digital Signature Scheme Giving Message Recovery

FIPS 46-2
Data Encryption Standard

FIPS 180-1
Secure Hash Standard

FIPS 186
Digital Signature Standard

FIPS 197
Advanced Encryption Standard

FIPS 198
Keyed-Hash Message Authentication Code (HMAC)

ANSI X3.92 - 1981
Data Encryption Algorithm

ANSI X3.106 - 1983
Modes of DEA Operation

Two modes specified in this standard are supported:
1. Electronic Code Book (ECB) mode
2. Cipher Block Chaining (CBC) mode

ANSI X9.8 - 1982
Personal Identification Number (PIN) Management and Security

ANSI X9.9 - 1986
Financial Institution Message Authentication (Wholesale)

ANSI X9.17 - 1985 (Reaffirmed 1991)
Financial Institution Key Management (Wholesale)

ANSI X9.19
Optional Double-MAC Procedure

ANSI X9.23 - 1988
Encryption of Wholesale Financial Messages

ANSI X9.24 - 2002 Retail Financial Services Symmetric Key Management - approved November 8, 2002

Derived Unique Key Per Transaction (for double-length PIN keys)

EMV2000 Integrated Circuit Card Specifications for Payment Systems Book 2-Security and Key Management Version 4.0 - December, 2000

ViSA - Welcome to Visa Integrated Circuit Card Specification - effective October 31, 2001

Appendix B. Summary of callable service support by hardware configuration

The callable services available to your applications depend on the configuration of your server and cryptographic features. The configuration of the cryptographic features is dependent on U.S. Export Regulations. For information on the configurations available in your country, contact your IBM Marketing Representative.

Without a PCI X Cryptographic Coprocessor, Crypto Express2 Coprocessor, or Crypto Express3 Coprocessor, only these services are available on the z890, z990, z9 EC, z9 BC, z10 EC, z10 BC, or z196:

- Character/Nibble Conversion (CSNBXBC and CSNBXCB)
- Code Conversion (CSNBXEA and CSNBXAE)
- Control Vector Generate (CSNBCVG)
- ICSF Query Algorithms (CSFIQA)
- ICSF Query Facility (CSFIQF)
- One–Way Hash Generate (CSNBOWH and CSNBOWH1)
- PKA Key Token Build (CSNDPKB)
- PKA Public Key Extract (CSNDPKX)
- PKCS #11 Token Record Create (CSFPTRC)
- PKCS #11 Token Record Delete (CSFPTRD)
- PKCS #11 Token Record List (CSFPTRL)
- PKCS #11 Get Attribute Value (CSFPGAV)
- PKCS #11 Set Attribute Value (CSFPSAV)
- X9.9 Data Editing (CSNB9ED)
- Services requiring CP Assist for Cryptographic Functions
  - Decode (CSNBDCO)
  - Encode (CSNBECO)
  - MDC Generate (CSNBMDG and CSNBMDG1)
  - Symmetric Key Decipher (CSNBSYD and CSNBSYD1) - for the DES, TDES and AES algorithms.
  - Symmetric Key Encipher (CSNBSYE and CSNBSYE1) - for the DES, TDES and AES algorithms.
- Services requiring a PCI Cryptographic Accelerator
  - PKA Decrypt (CSNDPKD)
  - PKA Encrypt (CSNDPKE) ZERO-PAD formatting only
  - Digital Signature Verify (CSNDDSV)

ICSF will route requests with clear RSA keys to accelerators when available. The following services are supported:

- PKA Decrypt (CSNDPKD) with ZERO-PAD formatting
- PKA Encrypt (CSNDPKE) with ZERO-PAD formatting
- PKA Decrypt (CSNDPKD) with ZERO-PAD and MRP formatting (z990/z890 and later)
- Digital signature verify (CSNDDSV) (z990/z890 and later)

In Table 4, letters represent various configurations according to:

- Letter A **(CCF only**) - IBM @server zSeries 900 configured for full DES with PKA or configured for DES with exportable PKA
- Letter B (**CCF with PCICC**) — IBM @server zSeries 900 configured for full DES with PCI Cryptographic Coprocessor
- Letter C (**PCIXCC/CEX2C**) — IBM @server zSeries 990 or IBM @server zSeries 890 with CP Assist for Cryptographic Functions DES/TDES Enablement and PCIXCC/CEX2C
- Letter D (**CEX2C/CEX3C**) — z9 EC, z9 BC, z10 EC and z10 BC with CP Assist for Cryptographic Functions DES/TDES Enablement and CEX2C, or z10 EC and z10 BC with CP Assist for Cryptographic Functions DES/TDES Enablement and CEX3C.
- Letter E – (**CEX3C**) z196 with CP Assist for Cryptographic Functions DES/TDES Enablement and CEX3C.

Note: Feature code 0800 is required for the IBM @server zSeries 900.

Table 4. Summary of ICSF Callable Services Support

Service Name	Function	A	B	C	D	E
ANSI X9.17 EDC Generate	Generates an ANSI X9.17 error detection code on an arbitrary length string using the special MAC key (x'0123456789ABCDEF').	X	X			
ANSI X9.17 Key Export	Uses the ANSI X9.17 protocol to export a DATA key or a pair of DATA keys with or without an AKEK. Supports the export of a CCA IMPORTER or EXPORTER KEK. Converts a single DATA key or combines two DATA keys into a single MAC key.	X	X			
ANSI X9.17 Key Import	Uses the ANSI X9.17 protocol to import a DATA key or a pair of DATA keys with or without an AKEK. Supports the import of a CCA IMPORTER or EXPORTER KEK. Converts a single DATA key or combines two DATA keys into a single MAC key.	X	X			
ANSI X9.17 Key Translate	Uses the ANSI X9.17 protocol to translate, in a single service call, either one or two DATA keys or a single KEK from encryption under one AKEK to encryption under another AKEK. Converts a single DATA key or combines two DATA keys into a single MAC key.	X	X			
ANSI X9.17 Transport Key Partial Notarize	Permits the preprocessing of an AKEK with origin and destination identifiers to create a partially notarized AKEK.	X	X			
Character/Nibble Conversion	Converts a binary string to a character string or vice versa.	X	X	X	X	X
CVV Key Combine	Combines two single-length CCA internal key tokens into 1 double-length CCA key token containing a CVVKEY-A key type.					X
Ciphertext Translate	Translates the user-supplied ciphertext from one key and enciphers the ciphertext to another key.	X	X			
CKDS Key Record Create	Adds a key record containing a key token set to binary zeros to both the in-storage and DASD copies of the CKDS.	X	X	X	X	X

Table 4. Summary of ICSF Callable Services Support (continued)

Service Name	Function	A	B	C	D	E
CKDS Key Record Create2	Adds a key record containing a key token to both the in-storage and DASD copies of the CKDS.					X
CKDS Key Record Delete	Deletes a key record from both the in-storage and DASD copies of the CKDS.	X	X	X	X	X
CKDS Key Record Read	Copies an internal key token from the in-storage copy of the CKDS to application storage.	X	X	X	X	X
CKDS Key Record Read2	Copies an internal key token from the in-storage copy of the CKDS to application storage.					X
CKDS Key Record Write	Writes an internal key token to the CKDS record specified in the key label parameter. Updates both the in-storage and DASD copies of the CKDS currently in use.	X	X	X	X	X
CKDS Key Record Write2	Writes an internal key token to the CKDS record specified in the key label parameter. Updates both the in-storage and DASD copies of the CKDS currently in use.					X
Clear Key Import	Imports a clear DATA key, enciphers it under the master key, and places the result into an internal key token.	X	X	X	X	X
Clear PIN Encrypt	Formats a PIN into a PIN block format (IBM 3621, IBM3624, ISO-0, ISO-1, ISO-2, IBM 4704 encrypting PINPAD, VISA 2, VISA 3, VISA 4, ECI 2, ECI 3) and encrypts the results.		X	X	X	X
Clear PIN Generate	Generates a clear personal identification number (PIN), a PIN verification value (PVV), or an offset using one of these algorithms: Interbank PIN (INBK-PIN) IBM 3624 (IBM-PIN or IBM-PINO) IBM German Bank Pool (GBP-PIN or GBP-PINO) VISA PIN validation value (VISA-PVV)	X	X	X	X	X
Clear PIN Generate Alternate	Generates a clear VISA PIN validation value (PVV) from an input encrypted PIN block.	X	X	X	X	X
Code Conversion	Converts EBCDIC data to ASCII data or vice versa.	X	X	X	X	X
Control Vector Generate	Builds a control vector from keywords specified as input to the service.	X	X	X	X	X
Control Vector Translate	Changes the control vector used to encipher an external key.		X	X	X	X

Table 4. Summary of ICSF Callable Services Support (continued)

Service Name	Function	A	B	C	D	E
Coordinated KDS Administration	Performs a CKDS refresh or CKDS reencipher and change master key operation while allowing applications to update the CKDS. In a sysplex environment, this callable service performs a coordinated sysplex-wide refresh or change master key operation from a single ICSF instance.			X	X	X
Cryptographic Variable Encipher	Encrypts plaintext using the Cipher Block Chaining (CBC) method.		X	X	X	X
Data key Export	Converts a DATA key from operational form into exportable form.	X	X	X	X	X
Data key Import	Imports an encrypted single-length or double-length DES data key and creates or updates a target internal key token with the master key-enciphered source key.		X	X	X	X
Decipher	Deciphers data using the cipher block chaining mode of the DES.	X	X	X	X	X
Decipher	Deciphers data using the CDMF mode of the DES.	X	X			
Decode	Decodes an 8-byte string of data using the electronic code book mode of the DES.	X	X	X	X	X
Digital Signature Generate	Generate a digital signature using a supplied hash and a private key. **Note:** Does not support DSS tokens on a PCIXCC, CEX2C, or CEX3C.	X	X	X	X	X
Digital Signature Verify	Verifies a digital signature using the same supplied hash that was used to generate the signature and the public key that corresponds to the private key used to generate the signature. **Note:** Does not support DSS tokens on a PCIXCC, CEX2C, or CEX3C.	X	X	X	X	X
Diversified Key Generate	Generates a key based on the key-generating key, the processing method, and the parameter supplied. The control vector of the key-generating key also determines the type of target key that can be generated.		X	X	X	X
ECC Diffie-Hellman	Creates symmetric key material from a pair of ECC keys using the Elliptic Curve Diffie-Hellman protocol and the static unified model key agreement scheme or "Z" data (the "secret" material output from D-H process).					X
Encipher	Enciphers data using the cipher block chaining mode of the DES.	X	X	X	X	X
Encipher	Enciphers data using the CDMF mode of the DES.	X	X			
Encode	Encodes an 8-byte string of data using the electronic code book mode of the DES.	X	X	X	X	X

Table 4. Summary of ICSF Callable Services Support (continued)

Service Name	Function	A	B	C	D	E
Encrypted PIN Generate	Generates and formats a PIN and encrypts the PIN block.		X	X	X	X
Encrypted PIN Translate	Reenciphers a PIN block from one PIN-encrypting key to another and, optionally, changes the PIN block format.	X	X	X	X	X
Encrypted PIN Verify	Verifies a supplied PIN using one of these algorithms: Interbank PIN (INBK-PIN) IBM 3624 (IBM-PIN or IBM-PINO) IBM German Bank Pool (GBP-PIN or GBP-PINO) VISA PIN validation value (VISA-PVV)	X	X	X	X	X
HMAC Generate	Generates a keyed-hashed message authentication code (MAC) for a text string that the application program supplies. The MAC is computed using the FIPS-198 algorithm.					X
HMAC Verify	Verifies a keyed-hashed message authentication code (MAC) for a text string that the application program supplies. The MAC is computed using the FIPS-198 algorithm.					X
ICSF query Algorithm	Provides information on cryptographic and hashing algorithms.	X	X	X	X	X
ICSF Query Facility	Provides ICSF status information and retrieves information on the PCIXCC/CEX2C/CEX3C and PCICC. **Note:** For column A, only the ICSFSTAT part of the ICSF service is available.	X	X	X	X	X
Key Export	Converts any key from operational form into exportable form.	X	X	X	X	X
Key Generate	Generates a 64-bit or 128-bit odd parity key, or a pair of keys, and returns them in encrypted forms.	X	X	X	X	X
Key Generate2	Generates a variable length key or a pair of keys, and returns them in encrypted forms.					X
Key Import	Converts any key from importable form into operational form.	X	X	X	X	X
Key Part Import	Combines the clear key parts of an AKEK and returns the combined key value in an internal key token or an update to the CKDS.	X	X	X	X	X
Key Part Import2	Combines the clear key parts of any key type and returns the combined key value in an internal key token or an update to the CKDS.					X
Key Test	Generates or verifies a secure verification pattern for keys. CSNBKYT requires the tested key to be in the clear or encrypted under the master key. CSNBKYTX also allows the tested key to be encrypted under a key-encrypting key.	X	X	X	X	X

Table 4. Summary of ICSF Callable Services Support (continued)

Service Name	Function	A	B	C	D	E
Key Test2	Generates or verifies a secure verification pattern for keys in the clear, encrypted under the master key, or encrypted under a key-encrypting key.					X
Key Token Build	Builds an internal token from the supplied parameters.	X	X	X	X	X
Key Token Build2	Builds an internal or external HMAC or AES token from the supplied parameters.					X
Key Translate	Uses one key-encrypting key to decipher an input key and then enciphers this using another key-encrypting key.		X	X	X	X
Key Translate2	uses one key-encrypting key to decipher an input key and then enciphers this key using another key-encrypting key within the secure environment.					X
MAC Generation	Generates a 4-, 6-, or 8-byte message authentication code (MAC) for a text string that the application program supplies. The MAC can be computed using either the ANSI X9.9-1 algorithm, the ANSI X9.19 optional double-MAC algorithm, or the EMV padding rules.	X	X	X	X	X
MAC Verification	Verifies a 4-, 6-, or 8-byte message authentication code (MAC) for a text string that the application program supplies. The MAC is computed using either the ANSI X9.9-1 algorithm, the ANSI X 9.19 optional double-MAC algorithm, or the EMV padding rules and is compared with a user-supplied MAC.	X	X	X	X	X
MDC Generation	Generates a 128-bit modification detection code (MDC) for a text string that the application program supplies.	X	X	X	X	X
Multiple Clear Key Import	Imports a clear DATA key of one, two, or three parts, enciphers it under the master key, and places the result into an internal key token.	X	X	X	X	X
Multiple Secure Key Import	Enciphers a clear key under the master key or an IMPORTER KEK, and places the result into an internal or external key token as any key type. Permits the import of double-length DATA, MAC and MACVER keys and triple-length DATA keys.	X	X	X	X	X
One-way Hash Generate	Generates a one-way hash on specified text using the SHA-1, SHA-2, RIPEMD-160 or MD5 method.	X	X	X	X	X
PCI Interface	Trusted Key Entry (TKE) workstation interface to the PCICC, PCIXCC, CEX2C, or CEX3C.		X	X	X	X
PIN Change/Unblock	Supports PIN change algorithms specified in the VISA Integrated Circuit Card Specifications.			X	X	X

Table 4. Summary of ICSF Callable Services Support (continued)

Service Name	Function	A	B	C	D	E
PKA Decrypt	Decrypts an RSA-encrypted key value and returns it to the application in the clear.	X	X	X	X	X
PKA Encrypt	Encrypts a PKCS 1.2 or ZERO-PAD formatted clear key value under an RSA public key to support Secure Sockets Layer (SSL) applications.	X	X	X	X	X
PKA Key Generate (DSS)	Generate PKA internal tokens for use with the DSS algorithm in the digital signature services.	X	X			
	Generate RSA keys for use on the CCF, PCICC, PCIXCC, CEX2C, or CEX3C.		X	X	X	X
	Generate ECC keys for use on the Crypto Express3 Coprocessor.					X
PKA Key Import	Import a PKA key token.	X	X	X	X	X
PKA Key Token Build	Create an external PKA key token containing an unenciphered private key.	X	X	X	X	X
PKA Key Token Change	Changes PKA key tokens (RSA, DSS, and ECC) or trusted block key tokens, from encipherment under the cryptographic coprocessor's old RSA master key or ECC master key to encipherment under the current cryptographic coprocessor's RSA master key or ECC master key.		X	X	X	X
PKA Key Translate	Translate a source CCA RSA key token into a target external smart card key token.				X	X
PKA Public Key Extract	Extract a PKA public key from a supplied PKA internal or external private key token.	X	X	X	X	X
PKCS #11 Derive Multiple Keys	Generate multiple secret key objects and protocol dependent keying material from an existing secret key object.	X	X	X	X	X
PKCS #11 Derive key	Generate a new secret key object from an existing key object.	X	X	X	X	X
PKCS #11 Get Attribute Value	Lists the attributes of an object.	X	X	X	X	X
PKCS #11 Generate Key Pair	Generate an RSA, DSA, Elliptic Curve, or Diffie-Hellman key pair.	X	X	X	X	X
PKCS #11 Generate Secret Key	Generate a secret key or set of domain parameters.	X	X	X	X	X
PKCS #11 Generate HMAC	Generate a hashed message authentication code (MAC).	X	X	X	X	X
PKCS #11 Verify HMAC	Verify a hash message authentication code (MAC).	X	X	X	X	X
PKCS #11 One-way hash, sign, or verify	Generate a one-way hash on specified text, sign specified text, or verify a signature on specified text	X	X	X	X	X

Table 4. Summary of ICSF Callable Services Support (continued)

Service Name	Function	A	B	C	D	E
PKCS #11 Private Key Sign	Decrypt or sign data using an RSA private key using zero-pad or PKCS #1 1.5 formatting, sign data using a DSA private key, or sign data using an Elliptic Curve private key in combination with DSA.	X	X	X	X	X
PKCS #11 Public Key Verify	Encrypt or verify data using an RSA public key using zero-pad or PKCS #1 1.5 formatting, verify a signature using a DSA public key, or verify a signature using an Elliptic Curve public key in combination with DSA.	X	X	X	X	X
PKCS #11 Pseudo-random Function	Generate pseudo-random output of arbitrary length.	X	X	X	X	X
PKCS #11 Secret Key Decrypt	Decipher data using a clear symmetric key.	X	X	X	X	X
PKCS #11 Secret Key Encrypt	Encipher data using a clear symmetric key.	X	X	X	X	X
PKCS #11 Set Attribute Value	Updates the attributes of an object.	X	X	X	X	X
PKCS #11 Token Record Create	Initializes or reinitializes a z/OS PKCS #11 token, creates or copies a token object in the token data set or creates or copies a session object for the current PKCS #11 session.	X	X	X	X	X
PKCS #11 Token Record Delete	Deletes a z/OS PKCS #11 token, token object or session object.	X	X	X	X	X
PKCS #11 Token Record List	Obtains a list of z/OS PKCS #11 tokens or a list of token and session objects for a token.	X	X	X	X	X
PKCS #11 Unwrap Key	Unwrap and create a key object using another key.	X	X	X	X	X
PKCS #11 Wrap Key	Wrap a key with another key.	X	X	X	X	X
PKDS Key Record Create	Writes a new record to the PKDS.	X	X	X	X	X
PKDS Key Record Delete	Deletes an existing record from the PKDS.	X	X	X	X	X
PKDS Key Record Read	Reads a record from the PKDS and returns the content of the record.	X	X	X	X	X
PKDS Key Record Write	Writes over an existing record in the PKDS.	X	X	X	X	X
PKSC Interface	Trusted Key Entry (TKE) workstation interface.	X	X			
Prohibit Export	Modifies an operational key so that it cannot be exported.		X	X	X	X
Prohibit Export Extended	Changes the external token of a key in exportable form so that it can be imported at the receiver node but not exported from that node.	X	X	X	X	X
Random Number Generate	Generates an 8-byte random number or a user-specified length random number. The output can be specified in three forms of parity: RANDOM, ODD, and EVEN.	X	X	X	X	X

Table 4. Summary of ICSF Callable Services Support (continued)

Service Name	Function	A	B	C	D	E
Remote Key Export	Generates DES keys for local use and for distribution to an ATM or other remote device.				X	X
Restrict Key Attribute	Modifies an operational key so that it cannot be exported.					X
Retained Key Delete	Deletes a key that has been retained within a PCICC, PCIXCC, CEX2C, or CEX3C.		X	X	X	X
Retained Key List	Lists the key labels of keys that have been retained within the PCICC, PCIXCC, CEX2C, or CEX3C.		X	X	X	X
Secure Key Import	Enciphers a clear key under the master key or an IMPORTER KEK, and places the result into an internal or external key token as any key type.	X	X	X	X	X
Secure Key Import2	Enciphers a variable-length clear HMAC or AES key under the master key and places the result into an internal key token.					X
Secure Messaging for Keys	Encrypts a text block, including a clear key value decrypted from an internal or external DES token.		X	X	X	X
Secure Messaging for PINs	Encrypts a text block, including a clear PIN block recovered from an encrypted PIN block.		X	X	X	X
SET Block Decompose	Compose the RSA-OAEP block and the DES-encrypted data block in support of the SET protocol.	X	X	X	X	X
SET Block Compose	Decompose the RSA-OAEP block and the DES-encrypted data block in support of the SET protocol.	X	X	X	X	X
Symmetric Algorithm Decipher	Deciphers data with the AES algorithm in an address space or a data space using the cipher block chaining or electronic code book modes.				X	X
Symmetric Algorithm Encipher	Enciphers data with the AES algorithm in an address space or a data space using the cipher block chaining or electronic code book modes.				X	X
Symmetric Key Decipher	Deciphers data in an address space or a data space. This service is available on machines with triple-DES feature codes.	X	X	X	X	X
Symmetric Key Encipher	Enciphers data in an address space or a data space. This service is available on machines with triple-DES feature codes.	X	X	X	X	X
Symmetric Key Generate	Generates a symmetric (DATA) key and returns it in two forms: encrypted under the DES master key and encrypted under a PKA public key.	X	X	X	X	X
Symmetric key Import	Imports a symmetric (DATA) key enciphered under an RSA public key and enciphers it under the DES master key.	X	X	X	X	X

Table 4. Summary of ICSF Callable Services Support (continued)

Service Name	Function	A	B	C	D	E
Symmetric Key Import2	Imports an HMAC or AES key enciphered under an RSA public key or AES EXPORTER key. It returns the key in operational form, enciphered under the master key.					X
Symmetric Key Export	Transfers an application-supplied symmetric key from encryption under the host master key to encryption under an application-supplied RSA public key or AES EXPORTER key. The application-supplied key must be an internal key token or the label in the CKDS of a DES DATA, AES DATA, or variable-length symmetric key token.	X	X	X	X	X
Symmetric MAC Generate	Uses the symmetric MAC generate callable service to generate a 96- or 128-bit message authentication code (MAC) for an application-supplied text string using an AES key.	X	X	X	X	X
Symmetric MAC Verify	Uses the symmetric MAC generate callable service to verify a 96- or 128-bit message authentication code (MAC) for an application-supplied text string using an AES key.	X	X	X	X	X
Transaction Validation	Supports the generation and validation of American Express card security codes.			X	X	X
Transform CDMF Key	Changes a CDMF DATA key in an internal or external token to a transformed shortened DES key.	X	X			
Trusted Block Create	Creates a trusted block under dual control that will be in external form, encrypted under an IMP-PKA transport key.				X	X
TR-31 Export	Converts a CCA token to TR-31 format for export to another party.					X
TR-31 Import	Converts a TR-31 key block to a CCA token.					X
TR-31 Optional Data Build	Constructs the optional block data structure for a TR-31 key block.	X	X	X	X	X
TR-31 Optional Data Read	Obtains lists of the optional block identifiers and optional block lengths, and obtains the data for a particular optional block.	X	X	X	X	X
TR-31 Parse	Retrieves standard header information from a TR-31 key block without importing the key.	X	X	X	X	X
User Derived Key	Generates a single- or double-length SESSION MAC key or updates an existing user derived key.	X	X			
VISA CVV Generate	Generates a Card Verification Value (CVV) or Card Verification Code (CVC). **Note:** 19-digit PAN support requires a PCIXCC or CEX2C.	X	X	X	X	X

Table 4. Summary of ICSF Callable Services Support (continued)

Service Name	Function	A	B	C	D	E
VISA CVV Verify	Verifies a Card Verification Value (CVV) or Card Verification Code (CVC). **Note:** 19-digit PAN support requires a PCIXCC, CEX2C, or CEX3C	X	X	X	X	X
X9.9 Data Editing	Edits an ASCII text string according to the editing rules of ANSI X9.9-4.	X	X	X	X	X

Appendix C. Accessibility

Publications for this product are offered in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when using PDF files, you may view the information through the z/OS Internet Library Web site or the z/OS Information Center. If you continue to experience problems, send an e-mail to mhvrcfs@us.ibm.com or write to:

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
U.S.A.

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS® enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Vol I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

z/OS information is accessible using screen readers with the BookServer or Library Server versions of z/OS books in the Internet library at:

`http://www.ibm.com/systems/z/os/zos/bkserv/`

Notices

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

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

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

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

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Site Counsel
IBM Corporation
2455 South Road
Poughkeepsie, NY 12601-5400
USA

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

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

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM®, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Glossary

This glossary defines terms and abbreviations used in Integrated Cryptographic Service Facility (ICSF). If you do not find the term you are looking for, refer to the index of the appropriate Integrated Cryptographic Service Facility document or view *IBM Glossary of Computing Terms* located at:

`http://www.ibm.com/ibm/terminology`

This glossary includes terms and definitions from:

- *IBM Glossary of Computing Terms*. Definitions are identified by the symbol (D) after the definition.
- The *American National Standard Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies can be purchased from the American National Standards Institute, 11 West 42nd Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.
- The *Information Technology Vocabulary*, developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published parts of this vocabulary are identified by the symbol (I) after the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.

Definitions specific to the Integrated Cryptographic Services Facility are labeled “In ICSF.”

A

access method services (AMS). The facility used to define and reproduce VSAM key-sequenced data sets (KSDS). (D)

Advanced Encryption Standard (AES). In computer security, the National Institute of Standards and Technology (NIST) Advanced Encryption Standard (AES) algorithm. The AES algorithm is documented in a draft Federal Information Processing Standard.

AES. Advanced Encryption Standard.

American National Standard Code for Information Interchange (ASCII). The standard code using a coded character set consisting of 7-bit characters (8 bits including parity check) that is used for information exchange among data processing systems, data communication systems, and associated equipment. The ASCII set consists of control characters and graphic characters.

ANSI key-encrypting key (AKEK). A 64- or 128-bit key used exclusively in ANSI X9.17 key management applications to protect data keys exchanged between systems.

ANSI X9.17. An ANSI standard that specifies algorithms and messages for DES key distribution.

ANSI X9.19. An ANSI standard that specifies an optional double-MAC procedure which requires a double-length MAC key.

application program. (1) A program written for or by a user that applies to the user's work, such as a program that does inventory control or payroll. (2) A program used to connect and communicate with stations in a network, enabling users to perform application-oriented activities. (D)

application program interface (API). (1) A functional interface supplied by the operating system or by a separately orderable licensed program that allows an application program written in a high-level language to use specific data or functions of the operating system or the licensed program. (D) (2) In ICSF, a callable service.

asymmetric cryptography. Synonym for public key cryptography. (D)

authentication pattern. An 8-byte pattern that ICSF calculates from the master key when initializing the cryptographic key data set. ICSF places the value of the authentication pattern in the header record of the cryptographic key data set.

authorized program facility (APF). A facility that permits identification of programs authorized to use restricted functions. (D)

C

callable service. A predefined sequence of instructions invoked from an application program, using a CALL instruction. In ICSF, callable services perform cryptographic functions and utilities.

CBC. Cipher block chaining.

CCA. Common Cryptographic Architecture.

CCF. Cryptographic Coprocessor Feature.

CDMF. Commercial Data Masking Facility.

CEDA. A CICS transaction that defines resources online. Using CEDA, you can update both the CICS system definition data set (CSD) and the running CICS system.

CEX2A. Crypto Express2 Accelerator

CEX2C. Crypto Express2 Coprocessor

CEX3A. Crypto Express3 Accelerator

CEX3C. Crypto Express3 Coprocessor

checksum. (1) The sum of a group of data associated with the group and used for checking purposes. (T) (2) In ICSF, the data used is a key part. The resulting checksum is a two-digit value you enter when you use the key-entry unit to enter a master key part or a clear key part into the key-storage unit.

Chinese Remainder Theorem (CRT). A mathematical theorem that defines a format for the RSA private key that improves performance.

CICS. Customer Information Control System.

cipher block chaining (CBC). A mode of encryption that uses the data encryption algorithm and requires an initial chaining vector. For encipher, it exclusively ORs the initial block of data with the initial control vector and then enciphers it. This process results in the encryption both of the input block and of the initial control vector that it uses on the next input block as the process repeats. A comparable chaining process works for decipher.

ciphertext. (1) In computer security, text produced by encryption. (2) Synonym for enciphered data. (D)

CKDS. Cryptographic Key Data Set.

clear key. Any type of encryption key not protected by encryption under another key.

CMOS. Complementary metal oxide semiconductor.

coexistence mode. An ICSF method of operation during which CUSP or PCF can run independently and simultaneously on the same ICSF system. A CUSP or PCF application program can run on ICSF in this mode if the application program has been reassembled.

Commercial Data Masking Facility (CDMF). A data-masking algorithm using a DES-based kernel and a key that is shortened to an effective key length of 40 DES key-bits. Because CDMF is not as strong as DES, it is called a masking algorithm rather than an encryption algorithm. Implementations of CDMF, when used for data confidentiality, are generally exportable from the USA and Canada.

Common Cryptographic Architecture: Cryptographic Application Programming Interface. Defines a set of cryptographic functions, external interfaces, and a set of key management rules that provide a consistent, end-to-end cryptographic architecture across different IBM platforms.

compatibility mode. An ICSF method of operation during which a CUSP or PCF application program can run on ICSF without recompiling it. In this mode, ICSF cannot run simultaneously with CUSP or PCF.

complementary keys. A pair of keys that have the same clear key value, are different but complementary types, and usually exist on different systems.

console. A part of a computer used for communication between the operator or maintenance engineer and the computer. (A)

control-area split. In systems with VSAM, the movement of the contents of some of the control intervals in a control area to a newly created control area in order to facilitate insertion or lengthening of a data record when there are no remaining free control intervals in the original control area. (D)

control block. (1) A storage area used by a computer program to hold control information. (I) Synonymous with control area. (2) The circuitry that performs the control functions such as decoding microinstructions and generating the internal control signals that perform the operations requested. (A)

control interval. A fixed-length area of direct-access storage in which VSAM stores records and creates distributed free space. Also, in a key-sequenced data set or file, the set of records pointed to by an entry in the sequence-set index record. The control interval is the unit of information that VSAM transmits to or from direct access storage. A control interval always comprises an integral number of physical records. (D)

control interval split. In systems with VSAM, the movement of some of the stored records in a control interval to a free control interval to facilitate insertion or lengthening of a record that does not fit in the original control interval. (D)

control statement input data set. A key generator utility program data set containing control statements that a particular key generator utility program job will process.

control statement output data set. A key generator utility program data set containing control statements to create the complements of keys created by the key generator utility program.

control vector. In ICSF, a mask that is exclusive ORed with a master key or a transport key before ICSF uses that key to encrypt another key. Control vectors ensure that keys used on the system and keys

distributed to other systems are used for only the cryptographic functions for which they were intended.

CPACF. CP Assist for Cryptographic Functions

CP Assist for Cryptographic Functions. Implemented on all z890, z990, z9 EC, z9 BC, z10 EC and z10 BC processors to provide SHA-1 secure hashing.

cross memory mode. Synchronous communication between programs in different address spaces that permits a program residing in one address space to access the same or other address spaces. This synchronous transfer of control is accomplished by a calling linkage and a return linkage.

CRT. Chinese Remainder Theorem.

Crypto Express2 Coprocessor. An asynchronous cryptographic coprocessor available on the z890, z990, z9 EC, z9 BC, z10 EC and z10 BC.

Crypto Express3 Coprocessor. An asynchronous cryptographic coprocessor available on z10 EC and z10 BC.

cryptographic adapter (4755 or 4758). An expansion board that provides a comprehensive set of cryptographic functions for the network security processor and the workstation in the TSS family of products.

cryptographic coprocessor. A microprocessor that adds cryptographic processing functions to specific z890, z990, z9 EC, z9 BC, z10 EC and z10 BC processors. The Cryptographic Coprocessor Feature is a tamper-resistant chip built into the processor board.

cryptographic key data set (CKDS). (1) A data set that contains the encrypting keys used by an installation. (D) (2) In ICSF, a VSAM data set that contains all the cryptographic keys. Besides the encrypted key value, an entry in the cryptographic key data set contains information about the key.

cryptography. (1) The transformation of data to conceal its meaning. (2) In computer security, the principles, means, and methods for encrypting plaintext and decrypting ciphertext. (D) (3) In ICSF, the use of cryptography is extended to include the generation and verification of MACs, the generation of MDCs and other one-way hashes, the generation and verification of PINs, and the generation and verification of digital signatures.

CUSP (Cryptographic Unit Support Program). The IBM cryptographic offering, program product 5740-XY6, using the channel-attached 3848. CUSP is no longer in service.

CUSP/PCF conversion program. A program, for use during migration from CUSP or PCF to ICSF, that converts a CUSP or PCF cryptographic key data set into a ICSF cryptographic key data set.

Customer Information Control System (CICS). An IBM licensed program that enables transactions entered at remote terminals to be processed concurrently by user written application programs. It includes facilities for building, using, and maintaining databases.

CVC. Card verification code used by MasterCard.

CVV. Card verification value used by VISA.

D

data encryption algorithm (DEA). In computer security, a 64-bit block cipher that uses a 64-bit key, of which 56 bits are used to control the cryptographic process and 8 bits are used for parity checking to ensure that the key is transmitted properly. (D)

data encryption standard (DES). In computer security, the National Institute of Standards and Technology (NIST) Data Encryption Standard, adopted by the U.S. government as Federal Information Processing Standard (FIPS) Publication 46, which allows only hardware implementations of the data encryption algorithm. (D)

data key or data-encrypting key. (1) A key used to encipher, decipher, or authenticate data. (D) (2) In ICSF, a 64-bit encryption key used to protect data privacy using the DES algorithm or the CDMF algorithm. AES data keys are now supported by ICSF.

data set. The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access. (D)

data-translation key. A 64-bit key that protects data transmitted through intermediate systems when the originator and receiver do not share the same key.

DEA. Data encryption algorithm.

decipher. (1) To convert enciphered data in order to restore the original data. (T) (2) In computer security, to convert ciphertext into plaintext by means of a cipher system. (3) To convert enciphered data into clear data. Contrast with encipher. Synonymous with decrypt. (D)

decode. (1) To convert data by reversing the effect of some previous encoding. (I) (A) (2) In ICSF, to decipher data by use of a clear key.

decrypt. See decipher.

DES. Data Encryption Standard.

diagnostics data set. A key generator utility program data set containing a copy of each input control statement followed by a diagnostic message generated for each control statement.

digital signature. In public key cryptography, information created by using a private key and verified by using a public key. A digital signature provides data integrity and source nonrepudiation.

Digital Signature Algorithm (DSA). A public key algorithm for digital signature generation and verification used with the Digital Signature Standard.

Digital Signature Standard (DSS). A standard describing the use of algorithms for digital signature purposes. One of the algorithms specified is DSA (Digital Signature Algorithm).

domain. (1) That part of a network in which the data processing resources are under common control. (T) (2) In ICSF, an index into a set of master key registers.

double-length key. A key that is 128 bits long. A key can be either double- or single-length. A single-length key is 64 bits long.

DSA. Digital Signature Algorithm.

DSS. Digital Signature Standard.

E

ECB. Electronic codebook.

ECI. Eurochèque International S.C., a financial institution consortium that has defined three PIN block formats.

EID. Environment Identification.

electronic codebook (ECB) operation. (1) A mode of operation used with block cipher cryptographic algorithms in which plaintext or ciphertext is placed in the input to the algorithm and the result is contained in the output of the algorithm. (D) (2) A mode of encryption using the data encryption algorithm, in which each block of data is enciphered or deciphered without an initial chaining vector. It is used for key management functions and the encode and decode callable services.

electronic funds transfer system (EFTS). A computerized payment and withdrawal system used to transfer funds from one account to another and to obtain related financial data. (D)

encipher. (1) To scramble data or to convert data to a secret code that masks the meaning of the data to any unauthorized recipient. Synonymous with encrypt. (2) Contrast with decipher. (D)

enciphered data. Data whose meaning is concealed from unauthorized users or observers. (D)

encode. (1) To convert data by the use of a code in such a manner that reconversion to the original form is possible. (T) (2) In computer security, to convert plaintext into an unintelligible form by means of a code system. (D) (3) In ICSF, to encipher data by use of a clear key.

encrypt. See encipher.

exit. (1) To execute an instruction within a portion of a computer program in order to terminate the execution of that portion. Such portions of computer programs include loops, subroutines, modules, and so on. (T) (2) In ICSF, a user-written routine that receives control from the system during a certain point in processing—for example, after an operator issues the START command.

exportable form. A condition a key is in when enciphered under an exporter key-encrypting key. In this form, a key can be sent outside the system to another system. A key in exportable form cannot be used in a cryptographic function.

exporter key-encrypting key. A 128-bit key used to protect keys sent to another system. A type of transport key.

F

file. A named set of records stored or processed as a unit. (T)

G

GBP. German Bank Pool.

German Bank Pool (GBP). A German financial institution consortium that defines specific methods of PIN calculation.

H

hashing. An operation that uses a one-way (irreversible) function on data, usually to reduce the length of the data and to provide a verifiable authentication value (checksum) for the hashed data.

header record. A record containing common, constant, or identifying information for a group of records that follows. (D)

I

ICSF. Integrated Cryptographic Service Facility.

importable form. A condition a key is in when it is enciphered under an importer key-encrypting key. A key is received from another system in this form. A key in importable form cannot be used in a cryptographic function.

importer key-encrypting key. A 128-bit key used to protect keys received from another system. A type of transport key.

initial chaining vector (ICV). A 64-bit random or pseudo-random value used in the cipher block chaining mode of encryption with the data encryption algorithm.

initial program load (IPL). (1) The initialization procedure that causes an operating system to commence operation. (2) The process by which a configuration image is loaded into storage at the beginning of a work day or after a system malfunction. (3) The process of loading system programs and preparing a system to run jobs. (D)

input PIN-encrypting key. A 128-bit key used to protect a PIN block sent to another system or to translate a PIN block from one format to another.

installation exit. See exit.

Integrated Cryptographic Service Facility (ICSF). A licensed program that runs under MVS/System Product 3.1.3, or higher, or OS/390 Release 1, or higher, or z/OS, and provides access to the hardware cryptographic feature for programming applications. The combination of the hardware cryptographic feature and ICSF provides secure high-speed cryptographic services.

International Organization for Standardization. An organization of national standards bodies from many countries, established to promote the development of standards to facilitate the international exchange of goods and services and to develop cooperation in intellectual, scientific, technological, and economic activity. ISO has defined certain standards relating to cryptography and has defined two PIN block formats.

ISO. International Organization for Standardization.

J

job control language (JCL). A control language used to identify a job to an operating system and to describe the job's requirements. (D)

K

key-encrypting key (KEK). (1) In computer security, a key used for encryption and decryption of other keys. (D) (2) In ICSF, a master key or transport key.

key generator utility program (KGUP). A program that processes control statements for generating and maintaining keys in the cryptographic key data set.

key output data set. A key generator utility program data set containing information about each key that the key generator utility program generates except an importer key for file encryption.

key part. A 32-digit hexadecimal value that you enter for ICSF to combine with other values to create a master key or clear key.

key part register. A register in the key storage unit that stores a key part while you enter the key part.

key store policy. Ensures that only authorized users and jobs can access secure key tokens that are stored in one of the ICSF key stores - the CKDS or the PKDS.

key store policy controls. Resources that are defined in the XFACILIT class. A control can verify the caller has authority to use a secure token and identify the action to take when the secure token is not stored in the CKDS or PKDS.

L

linkage. The coding that passes control and parameters between two routines.

load module. All or part of a computer program in a form suitable for loading into main storage for execution. A load module is usually the output of a linkage editor. (T)

LPAR mode. The central processor mode that enables the operator to allocate the hardware resources among several logical partitions.

M

MAC generation key. A 64-bit or 128-bit key used by a message originator to generate a message authentication code sent with the message to the message receiver.

MAC verification key. A 64-bit or 128-bit key used by a message receiver to verify a message authentication code received with a message.

magnetic tape. A tape with a magnetizable layer on which data can be stored. (T)

master key. (1) In computer security, the top-level key in a hierarchy of key-encrypting keys. (2) ICSF uses master keys to encrypt operational keys. Master keys are known only to the cryptographic coprocessors and are maintained in tamper proof cryptographic coprocessors. Examples of cryptographic coprocessors are CCF, PCICC, PCIXCC, CEX2C, and CEX3C. Some of the master keys that ICSF supports are a 128-bit DES master key, a 192-bit signature master key, and the 192-bit key management master key, a 192-bit symmetric master key (that is, DES), a 192-bit asymmetric master key, and a 256-bit AES master key.

master key concept. The idea of using a single cryptographic key, the master key, to encrypt all other keys on the system.

master key register. A register in the cryptographic coprocessors that stores the master key that is active on the system.

master key variant. A key derived from the master key by use of a control vector. It is used to force separation by type of keys on the system.

MD4. Message Digest 4. A hash algorithm.

MD5. Message Digest 5. A hash algorithm.

message authentication code (MAC). (1) The cryptographic result of block cipher operations on text or data using the cipher block chain (CBC) mode of operation. (D) (2) In ICSF, a MAC is used to authenticate the source of the message, and verify that the message was not altered during transmission or storage.

modification detection code (MDC). (1) A 128-bit value that interrelates all bits of a data stream so that the modification of any bit in the data stream results in a new MDC. (2) In ICSF, an MDC is used to verify that a message or stored data has not been altered.

multiple encipherment. The method of encrypting a key under a double-length key-encrypting key.

N

new master key register. A register in the key storage unit that stores a master key before you make it active on the system.

NIST. U.S. National Institute of Science and Technology.

NOCV processing. Process by which the key generator utility program or an application program encrypts a key under a transport key itself rather than a transport key variant.

noncompatibility mode. An ICSF method of operation during which CUSP or PCF can run independently and simultaneously on the same z/OS, OS/390 or MVS system. You cannot run a CUSP or PCF application program on ICSF in this mode.

nonrepudiation. A method of ensuring that a message was sent by the appropriate individual.

notarization. The ANSI X9.17 process involving the coupling of an ANSI key-encrypting key (AKEK) with ASCII character strings containing origin and destination identifiers and then exclusive ORing (or offsetting) the result with a binary counter.

O

OAEP. Optimal asymmetric encryption padding.

offset. The process of exclusively ORing a counter to a key.

old master key register. A register in the key storage unit that stores a master key that you replaced with a new master key.

operational form. The condition of a key when it is encrypted under the master key so that it is active on the system.

output PIN-encrypting key. A 128-bit key used to protect a PIN block received from another system or to translate a PIN block from one format to another.

P

PAN. Personal Account Number.

parameter. Data passed between programs or procedures. (D)

parmlib. A system parameter library, either SYS1.PARMLIB or an installation-supplied library.

partial notarization. The ANSI X9.17 standard does not use the term partial notarization. IBM has divided the notarization process into two steps and defined the term partial notarization as a process during which only the first step of the two-step ANSI X9.17 notarization process is performed. This step involves the coupling of an ANSI key-encrypting key (AKEK) with ASCII character strings containing origin and destination identifiers.

partitioned data set (PDS). A data set in direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data. (D)

PCI Cryptographic Coprocessor. The 4758 model 2 standard PCI-bus card supported on the field upgraded IBM S/390 Parallel Enterprise Server - Generation 5, the IBM S/390 Parallel Enterprise Server - Generation 6 and the IBM @server zSeries.

PCICA. PCI Cryptographic Accelerator.

PCICC. PCI Cryptographic Coprocessor.

PCI X Cryptographic Coprocessor. An asynchronous cryptographic coprocessor available on the IBM @server zSeries 990 and IBM @server zSeries 800.

PCIXCC. PCI X Cryptographic Coprocessor.

Personal Account Number (PAN). A Personal Account Number identifies an individual and relates that individual to an account at a financial institution. It consists of an issuer identification number, customer account number, and one check digit.

personal identification number (PIN). The 4- to 12-digit number entered at an automatic teller machine to identify and validate the requester of an automatic teller machine service. Personal identification numbers are always enciphered at the device where they are entered, and are manipulated in a secure fashion.

Personal Security card. An ISO-standard "smart card" with a microprocessor that enables it to perform a variety of functions such as identifying and verifying users, and determining which functions each user can perform.

PIN block. A 64-bit block of data in a certain PIN block format. A PIN block contains both a PIN and other data.

PIN generation key. A 128-bit key used to generate PINs or PIN offsets algorithmically.

PIN key. A 128-bit key used in cryptographic functions to generate, transform, and verify the personal identification numbers.

PIN offset. For 3624, the difference between a customer-selected PIN and an institution-assigned PIN. For German Bank Pool, the difference between an institution PIN (generated with an institution PIN key) and a pool PIN (generated with a pool PIN key).

PIN verification key. A 128-bit key used to verify PINs algorithmically.

PKA. Public Key Algorithm.

PKCS. Public Key Cryptographic Standards (RSA Data Security, Inc.)

PKDS. Public key data set (PKA cryptographic key data set).

plaintext. Data in normal, readable form.

primary space allocation. An area of direct access storage space initially allocated to a particular data set or file when the data set or file is defined. See also secondary space allocation. (D)

private key. In computer security, a key that is known only to the owner and used with a public key algorithm to decrypt data or generate digital signatures. The data is encrypted and the digital signature is verified using the related public key.

processor complex. A configuration that consists of all the machines required for operation.

Processor Resource/Systems Manager. Enables logical partitioning of the processor complex, may provide additional byte-multiplexer channel capability, and supports the VM/XA System Product enhancement for Multiple Preferred Guests.

Programmed Cryptographic Facility (PCF). (1) An IBM licensed program that provides facilities for enciphering and deciphering data and for creating, maintaining, and managing cryptographic keys. (D) (2) The IBM cryptographic offering, program product 5740-XY5, using software only for encryption and decryption. This product is no longer in service; ICSF is the replacement product.

PR/SM. Processor Resource/Systems Manager.

public key. In computer security, a key made available to anyone who wants to encrypt information using the public key algorithm or verify a digital signature generated with the related private key. The encrypted data can be decrypted only by use of the related private key.

public key algorithm (PKA). In computer security, an asymmetric cryptographic process in which a public key is used for encryption and digital signature verification and a private key is used for decryption and digital signature generation.

public key cryptography. In computer security, cryptography in which a public key is used for encryption and a private key is used for decryption. Synonymous with asymmetric cryptography.

R

RACE Integrity Primitives Evaluatiuon Message Digest. A hash algorithm.

RDO. Resource definition online.

record chaining. When there are multiple cipher requests and the output chaining vector (OCV) from the previous encipher request is used as the input chaining vector (ICV) for the next encipher request.

Resource Access Control Facility (RACF). An IBM licensed program that provides for access control by identifying and verifying the users to the system, authorizing access to protected resources, logging the detected unauthorized attempts to enter the system, and logging the detected accesses to protected resources. (D)

retained key. A private key that is generated and retained within the secure boundary of the PCI Cryptographic Coprocessor.

return code. (1) A code used to influence the execution of succeeding instructions. (A) (2) A value returned to a program to indicate the results of an operation requested by that program. (D)

Rivest-Shamir-Adleman (RSA) algorithm. A process for public key cryptography that was developed by R. Rivest, A. Shamir, and L. Adleman.

RMF. Resource Manager Interface.

RMI. Resource Measurement Facility.

RSA. Rivest-Shamir-Adleman.

S

SAF. Security Authorization Facility.

save area. Area of main storage in which contents of registers are saved. (A)

secondary space allocation. In systems with VSAM, area of direct access storage space allocated after primary space originally allocated is exhausted. See also primary space allocation. (D)

Secure Electronic Transaction. A standard created by Visa International and MasterCard for safe-guarding payment card purchases made over open networks.

secure key. A key that is encrypted under a master key. When ICSF uses a secure key, it is passed to a cryptographic coprocessor where the coprocessor decrypts the key and performs the function. The secure key never appears in the clear outside of the cryptographic coprocessor.

Secure Sockets Layer. A security protocol that provides communications privacy over the Internet by allowing client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery.

sequential data set. A data set whose records are organized on the basis of their successive physical positions, such as on magnetic tape. (D)

SET. Secure Electronic Transaction.

SHA (Secure Hash Algorithm, FIPS 180) . (Secure Hash Algorithm, FIPS 180) The SHA (Secure Hash Algorithm) family is a set of related cryptographic hash functions designed by the National Security Agency (NSA) and published by the National Institute of Standards and Technology (NIST). The first member of the family, published in 1993, is officially called SHA. However, today, it is often unofficially called SHA-0 to avoid confusion with its successors. Two years later, SHA-1, the first successor to SHA, was published. Four more variants, have since been published with increased output ranges and a slightly different design: SHA-224, SHA-256, SHA-384, and SHA-512 (all are sometimes referred to as SHA-2).

SHA-1 (Secure Hash Algorithm 1, FIPS 180). A hash algorithm required for use with the Digital Signature Standard.

SHA-2 (Secure Hash Algorithm 2, FIPS 180). Four additional variants to the SHA family, with increased output ranges and a slightly different design: SHA-224, SHA-256, SHA-384, and SHA-512 (all are sometimes referred to as SHA-2).

SHA-224. One of the SHA-2 algorithms.

SHA-256 . One of the SHA-2 algorithms.

SHA-384. One of the SHA-2 algorithms.

SHA-512 . One of the SHA-2 algorithms.

single-length key. A key that is 64 bits long. A key can be single- or double-length. A double-length key is 128 bits long.

smart card. A plastic card that has a microchip capable of storing data or process information.

special secure mode. An alternative form of security that allows you to enter clear keys with the key generator utility program or generate clear PINs.

SSL. Secure Sockets Layer.

supervisor state. A state during which a processing unit can execute input/output and other privileged instructions. (D)

System Authorization Facility (SAF). An interface to a system security system like the Resource Access Control Facility (RACF).

system key. A key that ICSF creates and uses for internal processing.

System Management Facility (SMF). A base component of z/OS that provides the means for gathering and recording information that can be used to evaluate system usage. (D)

T

TDEA. Triple Data Encryption Algorithm.

TKE. Trusted key entry.

Transaction Security System. An IBM product offering including both hardware and supporting software that provides access control and basic cryptographic key-management functions in a network environment. In the workstation environment, this includes the 4755 Cryptographic Adapter, the Personal Security Card, the 4754 Security Interface Unit, the Signature Verification feature, the Workstation Security Services Program, and the AIX Security Services Program/6000. In the host environment, this includes the 4753 Network Security Processor and the 4753 Network Security Processor MVS Support Program.

transport key. A 128-bit key used to protect keys distributed from one system to another. A transport key can either be an exporter key-encrypting key, an importer key-encrypting key, or an ANSI key-encrypting key.

transport key variant. A key derived from a transport key by use of a control vector. It is used to force separation by type for keys sent between systems.

TRUE. Task-related User Exit (CICS). The CICS-ICSF Attachment Facility provides a CSFATRUE and CSFATREN routine.

U

UAT. UDX Authority Table.

UDF. User-defined function.

UDK. User-derived key.

UDP. User Developed Program.

UDX. User Defined Extension.

V

verification pattern. An 8-byte pattern that ICSF calculates from the key parts you enter when you enter a master key or clear key. You can use the verification pattern to verify that you have entered the key parts correctly and specified a certain type of key.

Virtual Storage Access Method (VSAM). An access method for indexed or sequential processing of fixed and variable-length records on direct-access devices. The records in a VSAM data set or file can be organized in logical sequence by means of a key field (key sequence), in the physical sequence in which they are written on the data set or file (entry-sequence), or by means of relative-record number.

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in an SNA network. It provides single-domain, multiple-domain, and interconnected network capability. (D)

VISA. A financial institution consortium that has defined four PIN block formats and a method for PIN verification.

VISA PIN Verification Value (VISA PVV). An input to the VISA PIN verification process that, in practice, works similarly to a PIN offset.

Numerics

3621. A model of an IBM Automatic Teller Machine that has a defined PIN block format.

3624. A model of an IBM Automatic Teller Machine that has a defined PIN block format and methods of PIN calculation.

4753. The Network Security processor. The IBM 4753 is a processor that uses the Data Encryption Algorithm and the RSA algorithm to provide cryptograpic support for systems requiring secure transaction processing (and other cryptographic services) at the host computer. The NSP includes a 4755 cryptographic adapter in a workstation which is channel attached to a S/390 host computer.

4758. The IBM PCI Cryptographic processor provides a secure programming and hardware environment where DES and RSA processes are performed.

Index

Numerics

A

B

C

D

E

F

G

H

I

K

R

S

T

U

V

W

Z

Product Number: 5694-A01

Printed in USA

SA22-7519-15