z/OS
Integrated Cryptographic Service Facility

Administrator's Guide

SA22-7521-04

z/OS
Integrated Cryptographic Service Facility

Administrator's Guide

SA22-7521-04

Note!
Before using this information and the product it supports, be sure to read the general information under “Notices” on page 225.

Fifth Edition (June 2003)

This is a complete revision of SA22-7521-03.

This edition applies to Version 1 Release 4 of z/OS (5694–A01) and Version 1 Release 4 of z/OS.e (5655–G52) to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

A form for reader’s comments may be provided at the back of this publication, or you may address your comments to:

IBM Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States & Canada): 1+845+432-9405
FAX (Other Countries): Your International Access Code +1+845+432-9405

IBMLink (United States Customers Only): IBMUSM10(MHVRCFS)
Internet e-mail: mhvrcfs@us.ibm.com
World Wide Web: http://www.ibm.com/servers/eserver/zseries/zosqs.html

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure you include the following in your comment or note:
- Title and order number of this book
- Page number or topic related to your comment

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

Contents

Figures

Tables

About This Book

This book describes how to manage cryptographic keys by using the z/OS Integrated Cryptographic Service Facility (ICSF), which is part of z/OS Cryptographic Services. The z/OS Cryptographic Services includes the following components:

- z/OS Integrated Cryptographic Service Facility (ICSF)
- z/OS Open Cryptographic Services Facility (OCSF)

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

Hardware Features

Cryptographic hardware features include CP Assist for Cryptographic Functions, PCI Cryptographic Accelerator, PCI Cryptographic Coprocessor, and Cryptographic Coprocessor Feature.

Cryptographic Assist Instructions

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

- Cryptographic assist instructions are implemented on every processor. SHA-1 secure hashing is directly available to application programs.
- Feature code 3863, **CP Assist for Cryptographic Functions**, enables clear key DES and TDES instructions on all CPs. In addition, ICSF supports software implementation of AES.
- Feature code 0862, **PCI Cryptographic Accelerator**, is available on the IBM @server zSeries 990, but only if you have CP Assist for Cryptographic Functions. The IBM @server zSeries 990 can support a maximum of 12 PCI Cryptographic Accelerators.

Restriction: The IBM @server zSeries 990 does not support the Cryptographic Coprocessor Feature or PCI Cryptographic Coprocessor.

Cryptographic Coprocessor Feature

The **Cryptographic Coprocessor Feature** (CCF) can have up to two cryptographic coprocessors protected by tamper-detection circuitry and a cryptographic battery unit. The Cryptographic Coprocessor Feature is available on the following servers:

- IBM S/390 G6 Enterprise Server with feature code 0800 and one of the following feature codes: 0814, 0815, 0834, 0835
- IBM @server zSeries 800 with feature code 0800 plus feature code 0875.
- IBM @server zSeries 900 with feature code 0800 plus feature code 0875

Restriction: The IBM @server zSeries 990 does not support the Cryptographic Coprocessor Feature.

PCI Cryptographic Coprocessor

The **PCI Cryptographic Coprocessor** (PCICC) is based on the 4758 model 2 standard PCI-bus card package and is available on the following servers. You must have at least one Cryptographic Coprocessor Feature on your system with a PCICC.

- S/390 G6 Enterprise Server with feature codes 0864 and 0865. Feature code 0860 is needed for each PCI Cryptographic Coprocessor. Note that each feature code has one coprocessor.
- IBM @server zSeries 800 with feature codes 0861 and 0865. Note that each feature code has two coprocessors.
- IBM @server zSeries 900 with feature codes 0861 and 0865. Note that each feature code has two coprocessors.

Restriction: The IBM @server zSeries 990 does not support the PCI Cryptographic Coprocessor.

PCI Cryptographic Accelerator

The **PCI Cryptographic Accelerator** (PCICA) is available on the following servers. You must have at least one Cryptographic Coprocessor Feature on your system with a PCICA. Note that each feature code has two coprocessors.

- IBM @server zSeries 800 with feature code 0862.
- IBM @server zSeries 900 with feature code 0862.

Note: The IBM @server zSeries 800 and 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.

ICSF Features

ICSF enhances z/OS security as follows:

- It ensures data privacy by encrypting and decrypting the data.
- It manages personal identification numbers (PINs).
- It ensures the integrity of data through the use of modification detection codes (MDCs), hash functions, or digital signatures.
- It ensures the privacy of cryptographic keys themselves by encrypting them under a master key or another key-encrypting key.
- It enforces DES key separation, which ensures that cryptographic keys are used only for their intended purposes.
- It enhances system availability by providing continuous operation.
- It enables the use of Rivest-Shamir-Adelman (RSA) and Digital Signature Standard (DSS) public and private keys on a multi-user, multi-application platform.
- It provides the ability to generate RSA key pairs within the secure hardware boundary of the PCI Cryptographic Coprocessor.

Resource Access Control Facility (RACF), an element of the z/OS Security Server can be used to control access to cryptographic keys and functions.

This book explains the basic concepts of protecting and managing the keys used in cryptographic functions. It provides step-by-step guidance for the ICSF administration tasks.

Who Should Use This Book

This book is intended for anyone who manages cryptographic keys. Usually, this person is the ICSF administrator.

The ICSF administrator performs the following major tasks:

- Entering and changing master keys
- Generating, entering, and updating cryptographic keys
- Viewing system status, which includes hardware status, installation options, installation exits, and installation services

How to Use This Book

The first three chapters of this book give you background information you need to manage cryptographic keys on ICSF.

- Chapter 1, “Introduction”, on page 1, gives a brief introduction to the role of cryptography in data security. It describes the cryptographic algorithms that ICSF supports and discusses the importance of key secrecy.
- Chapter 2, “Understanding Cryptographic Keys”, on page 7, describes how ICSF protects keys and controls their use. It also describes the types of keys and how ICSF protects data and keys within a system and outside a system.
- Chapter 3, “Managing Cryptographic Keys”, on page 23, describes how to manage keys with ICSF. It introduces how to generate or enter, maintain, and distribute keys using ICSF. It also describes how to use keys to distribute keys and PINs between systems.

The remaining chapters of this book describe how to use the ICSF panels to manage cryptographic keys and also to view system status. Each chapter gives background information about a major task and leads you through the panels, step-by-step, for the task.

- Chapter 4, “Using the Pass Phrase Initialization Utility”, on page 45 discusses pass phrase initialization and gives step-by-step instructions on how to get your cryptographic system up and running quickly. The pass phrase initialization utility allows you to install the necessary master keys on both the Cryptographic Coprocessor Features and the PCI Cryptographic Coprocessors, and initialize the CKDS with a minimal effort.
- Chapter 5, “Managing Master Keys”, on page 51 describes how to enter, activate, and manage master keys with both the Cryptographic Coprocessor Feature and the PCI Cryptographic Coprocessor.
- Chapter 6, “Running in a Sysplex Environment”, on page 91, describes various considerations for the PKDS and CKDS.
- Chapter 7, “Managing Cryptographic Keys by Using the Key Generator Utility Program”, on page 95, describes how to use the *key generator utility program* (KGUP). The program generates keys and stores them in the *cryptographic key data set* (CKDS).
- Chapter 8, “Viewing and Changing System Status”, on page 157, describes how to display information about parts of ICSF that your installation can specify and

change. It describes how to use the panels to display installation options, hardware status, PCI management status, installation exits, and installation-defined services.

- Chapter 9, “Managing User Defined Extensions on PCI Cryptographic Coprocessors”, on page 187, describes how to use panels to manage your own cryptographic callable service.
- Chapter 10, “Using the Utility Panels to Encode and Decode Data”, on page 191, describes how to use utility panels to encipher and decipher data with a key that is not enciphered.
- Chapter 11, “Using the ICSF Utility Program CSFEUTIL”, on page 195, describes how to use the CSFEUTIL utility program to change master keys and refresh or reencipher the CKDS.
- Chapter 12, “Using the ICSF Utility Program CSFPUTIL”, on page 201, describes how to use the CSFPUTIL utility program to reencipher, activate and refresh a PKDS.
- Appendix A, “CCC Bit Assignments”, on page 205, contains selected CCC (crypto configuration control) definitions.
- Appendix B, “Control Vector Table”, on page 207, contains a table of the control vector values that are associated with each key type.
- Appendix C, “Supporting Algorithms and Calculations”, on page 209, shows algorithms that are used to calculate checksums, verification patterns, and other values.
- Appendix D, “PR/SM Considerations during Key Entry”, on page 213, discusses additional considerations when running in PR/SM logical partition mode.
- “Notices” on page 225, discusses notices, programming interface information and trademarks.

Where to Find More Information

The information in this book is supported by other books in the ICSF library and other system libraries. The ICSF library is shown in Figure 1 on page xix.

The following publications contain additional ICSF information:

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

  This book describes the 18F abend code ICSF issues.
- *z/OS MVS System Management Facilities (SMF)*, SA22-7630

  This book describes SMF record type 82, where ICSF records events.
- *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 HLL*, 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 ENF-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

- *MVS Batch Local Shared Resources*, GC28-1469
- *MVS/DFP Managing VSAM Data Sets*, SC26-4568
- *MVS/ESA Data Administration: Macro Instruction Reference*, SC26-4506-02

Related Publications

- *IBM Common Cryptographic Architecture: Cryptographic Application Programming Interface Reference*, SC40-1675
- *Support Element Operations Guide*, SC28-6802
- *PR/SM Planning Guide*, SB10-7032
- *S/390 Hardware Management Console Guide*, SC28-6805-00
- *IBM Security Architecture: Securing the Open Client/Server Distributed Enterprise*, SC28-8135
- *VTAM Programming for LU 6.2*, SC31-6551
- *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.
- *Applied Cryptography, Second Edition*

Documentation for the PCI Cryptographic Coprocessor is found on the web at **http://www.ibm.com/security/cryptocards/html/library.shtml**.

- *IBM 4758 PCI Cryptographic Coprocessor CCA Support Program Installation Manual for IBM 4758 Models 002 and 023 with Release 2.40*
- *IBM 4758 PCI Cryptographic Coprocessor CCA Basic Services Reference and Guide Version 2.40 for the IBM 4758 Models 002 and 023*
- *IBM 4758 PCI Cryptographic Coprocessor General Information*
- *IBM 4758 PCI Cryptographic Coprocessor Installation*

Using LookAt to look up message explanations

LookAt is an online facility that lets you look up explanations for most messages you encounter, as well as for some system abends and codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can access LookAt from the Internet at: http://www.ibm.com/eserver/zseries/zos/bkserv/lookat/ or from anywhere in z/OS or z/OS.e where you can access a TSO/E command line (for example, TSO/E prompt, ISPF, z/OS UNIX System Services running OMVS).

The LookAt Web site also features a mobile edition of LookAt for devices such as Pocket PCs, Palm OS, or Linux-based handhelds. So, if you have a handheld device with wireless access and an Internet browser, you can now access LookAt message information from almost anywhere.

To use LookAt as a TSO/E command, you must have LookAt installed on your host system. You can obtain the LookAt code for TSO/E from a disk on your *z/OS Collection* (SK3T-4269) or from the LookAt Web site's **Download** link.

Accessing z/OS™ licensed documents on the Internet

z/OS licensed documentation is available on the Internet in PDF format at the IBM Resource Link™ Web site at:

`http://www.ibm.com/servers/resourcelink`

Licensed documents are available only to customers with a z/OS license. Access to these documents requires an IBM Resource Link user ID and password, and a key code. With your z/OS order you received a Memo to Licensees, (GI10-0671), that includes this key code. [1]

To obtain your IBM Resource Link user ID and password, log on to:

`http://www.ibm.com/servers/resourcelink`

To register for access to the z/OS licensed documents:

1. Sign in to Resource Link using your Resource Link user ID and password.
2. Select **User Profiles** located on the left-hand navigation bar.

Note: You cannot access the z/OS licensed documents unless you have registered for access to them and received an e-mail confirmation informing you that your request has been processed.

Printed licensed documents are not available from IBM.

To print licensed documents, you can use the PDF format on either **z/OS Licensed Product Library CD-ROM** or IBM Resource Link .

Do You Have Problems, Comments, or Suggestions?

Your suggestions and ideas can contribute to the quality and the usability of this book. If you have problems while using this book, or if you have suggestions for improving it, complete and mail the Reader's Comment Form found at the back of the book.

1. z/OS.e™ customers received a Memo to Licensees, (GI10-0684) that includes this key code.

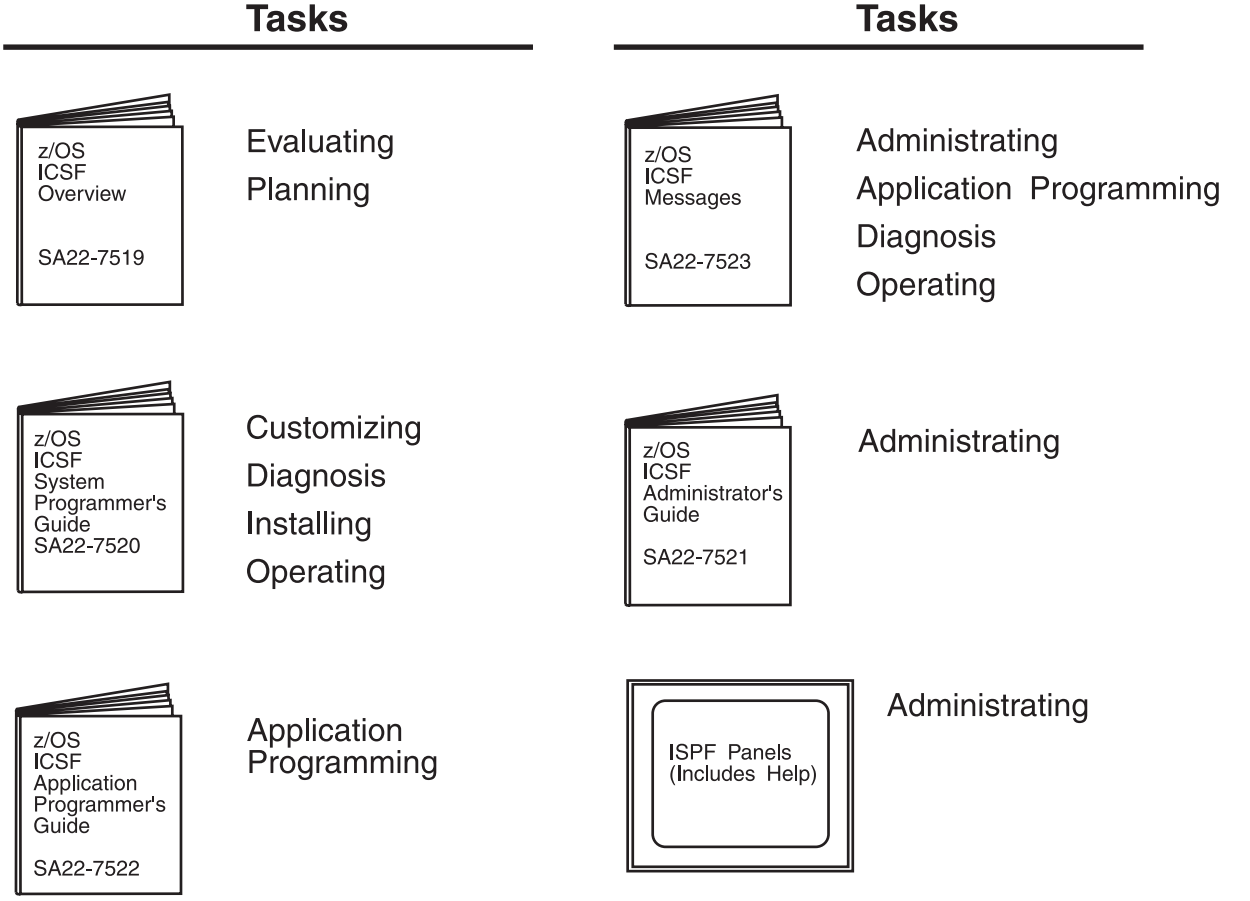


Figure 1. The z/OS ICSF Library

Summary of Changes

Summary of Changes
for SA22-7521-04
z/OS Version 1 Release 4
as updated May 2003

The book contains information previously presented in *z/OS ICSF Administrator's Guide*, SA22-7521-03, which supports z/OS Version 1 Release 3.

New information

- Information is added to indicate this document supports z/OS.e and the IBM @server zSeries 800.
- Support for the IBM @server zSeries 990 server has been added. If you are running ICSF in this environment, refer to Appendix E, "Running HCR7708 on an IBM @server zSeries 990", on page 217.
- Callable services
  - Symmetric Key Decipher (CSNBSYD1) - ALET support
  - Symmetric Key Encipher (CSNBSYE1) - ALET support
- Access Control Points
  - Data Key Export - Unrestricted
  - Data Key Import - Unrestricted
  - Key Export - Unrestricted
  - Key Import - Unrestricted
  - Key Part Import - Unrestricted

Deleted information

References to DATAC have been removed. The services affected are CV Generate, Key Export, Key Import, Key Generate, and Key Token Build. Double-length DATA keys should be used instead of DATAC.

References to Cryptographic Unit Support Product (CUSP) have been removed as the product is no longer supported.

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

You may notice changes in the style and structure of some content in this document—for example, headings that use uppercase for the first letter of initial words only, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our documents.

Summary of Changes
for SA22-7521-03
z/OS Version 1 Release 3
as updated March 2002

The book contains information previously presented in *z/OS ICSF Administrator's Guide*, SA22-7521-02, which supports z/OS Version 1 Release 2.

New Information

There is a new panel for displaying the PCICC default role.

An appendix with z/OS product accessibility information has been added.

Changed Information

There have been extensive changes to the panels to improve usability and panel flows. The following major changes were made:

- CKDS initialization was moved to the master key management panels.
- PCICC and CCF coprocessor selection have been combined into the coprocessor management panel.
- Clear master key entry is under coprocessor management and there is the ability to load any of the coprocessores with clear master key parts from one panel.
- An option was added to the utility panels to generate key values from a pass phrase.
- CSFEUTIL can be used to load DES and PKA master keys using a pass phrase.
- Hardware status is under coprocessor management.
- TSO panels required for TKE are combined into one option.

This book includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Starting with z/OS V1R2, you may notice changes in the style and structure of some content in this book--for example, headings that use uppercase for the first letter of initial words only, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our books.

Summary of Changes
for SA22-7521-02
z/OS Version 1 Release 2
as updated October 2001

The book contains information previously presented in *z/OS ICSF Administrator's Guide*, SA22-7521-01, which supports z/OS Version 1 Release 1.

New Information

There are two new chapters: Chapter 6, "Running in a Sysplex Environment", on page 91 and Chapter 12, "Using the ICSF Utility Program CSFPUTIL", on page 201.

There is a more streamlined procedure for adding PCICCs after first-time Pass Phrase Initialization.

This revision also includes information on reenciphering and activating the PKDS and refreshing the PKDS cache.

This book includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Summary of Changes
for SA22-7521-01
as updated June 2001

New Information: This revision includes a new appendix on CCC bit assignments.

This book includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Chapter 1. Introduction

In today's business environment, data is one of the most valuable resources that is required for maintaining a competitive edge. As a result, businesses must often be able to maintain data secrecy, readily determine the authenticity of data, and closely control access to data.

Data systems commonly consist of many types and sizes of computer systems that are interconnected through many different electronic data networks. It is now common for an organization to interconnect its data systems with systems that belong to customers, vendors, and competitors. Larger organizations might include international operations, or they might provide continual services. As the Internet becomes the basis for electronic commerce and as more businesses automate their data processing operations, the potential for disclosing sensitive data to unauthorized persons increases. As a result, approaches to data security must provide the following:

- Common services for each computing environment
- Support for national and international standards
- Graduated degrees of support
- Flexibility to work with existing and emerging systems
- Management of the increased risks to data assets

A combination of elements must work together to achieve a more secure environment. To provide a foundation for a secure environment, a security policy should be based on the following evaluations:

- An appraisal of the value of data
- An analysis of the potential threats to that data

The Tasks of a Data Security System

To help you select the products and services that you need to put a data security policy into effect, IBM has categorized the following security functions. These functions are based on the International Organization for Standardization (ISO) standard 7498-2:

- **Identification and authentication**—identifies users to the system and provides 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 individuals.
- **Data integrity**—ensures that data is in its original and unaltered form.
- **Security management**—administers, controls, and reviews a business security policy.
- **Nonrepudiation**—assures that a message sender cannot deny later that he or she sent the message.

The z/OS Integrated Cryptographic Service Facility (ICSF) provides a cryptographic application programming interface that you can use along with your system's cryptographic feature to put these functions into effect in your data security policy.

The Role of Cryptography in Data Security

Cryptography includes a set of techniques for scrambling or disguising data so that it is available only to someone who can restore the data to its original form. In current computer systems, cryptography provides a strong, economical basis for keeping data secret and for verifying data integrity.

ICSF supports the following two main types of cryptographic processes:

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

Symmetric Cryptography

ICSF supports two symmetric cryptography algorithms: The Data Encryption Algorithm, and the Commercial Data Masking Facility.

The Data Encryption Algorithm and the Data Encryption Standard

For commercial business applications, the cryptographic process that is known as the Data Encryption Algorithm (DEA)[2] has been widely adopted. The Data Encryption Standard (DES), as well as other documents, defines how to use the DES algorithm to encipher data. The Data Encryption Standard is the basis for many other processes for concealing data, such as protection of passwords and personal identification numbers (PINs). DES uses a key to vary the way that the algorithm processes the data. DES data-encrypting keys can be single-, double-, or triple-length. A single-length DES key is a 56-bit piece of data that is normally retained in 8 bytes of data. Each eighth bit of the key data is designated as a parity bit. A symmetric cryptographic system uses the same key both to transform the original data (plaintext) to its disguised, enciphered form (ciphertext) and to return it to its plaintext form.

The DES algorithm, which has been proven to be efficient and strong, is widely known. For this reason, data security is dependent on maintaining the secrecy of the cryptographic keys. Because the DES algorithm is common knowledge, you must keep the key secret to ensure that the data remains secret. Otherwise, someone who has the key that you used to encipher the data would be able to decipher the data. Key management refers to the procedures that are used to keep keys secret.

When you want someone to be able to confirm the integrity of your data, you can use the DES algorithm to compute a message authentication code (MAC). When used in this way, the DES algorithm is a powerful tool. It is almost impossible to meaningfully change the data and still have it produce the same MAC for a given key. The standardized approaches authenticate data such as financial transactions, passwords, and computer programs.

The originator of the data sends the computed MAC with the data. To authenticate the data, the receiver uses the DES algorithm to recompute the MAC. The receiver's application then compares this result with the MAC that was sent with the

2. The Data Encryption Algorithm is often referred to as the DEA, the DES algorithm or just as DES. This document uses the term DES to refer to this algorithm.

data. Someone could, of course, change both the data and the MAC. Therefore, the key that is used to compute the MAC must be kept a secret between the MAC's originator and the MAC's authenticator.

An alternative approach to data-integrity checking uses a standard key value and multiple iterations of the DES algorithm to generate a modification detection code (MDC). In this approach to data-integrity checking, the MDC must be received from a trusted source. The person who wants to authenticate the data recomputes the MDC and compares the result with the MDC that was sent with the data.

The Commercial Data Masking Facility

The Commercial Data Masking Facility (CDMF) defines a scrambling technique for data confidentiality. CDMF is a substitute for DES for those customers who have been previously prohibited from receiving IBM products that support DES data confidentiality services.

The CDMF data confidentiality algorithm is a cryptographic system that provides data masking and unmasking. The algorithm includes both a key-shortening process and a standard DES encryption and decryption process. The first process shortens the key to an effective length of 40 bits prior to its use in the data masking process. CDMF uses the DES algortihm with the shortened key to ensure confidence in the CDMF algorithm.

Advanced Encryption Standard

ICSF supports the Advanced Encryption Standard algorithm for data privacy. This provides strong encryption. Key lengths of 128-bits, 192-bits and 256-bits are supported. The algorithm has the same availability as triple DES.

Asymmetric Algorithm or Public Key Cryptography

In an asymmetric cryptographic process one key is used to encipher the data, and a different but corresponding key is used to decipher the data. A system that uses this type of process is known as a public key system. The key that is used to encipher the data is widely known, but the corresponding key for deciphering the data is a secret. For example, many people can use your public key to send enciphered data to you with confidence, knowing that only you should possess the secret key for deciphering the data.

Public key cryptographic algorithms are used in processes that simplify the distribution of secret keys, assuring data integrity and provide nonrepudiation through the use of digital signatures.

The widely known and tested public key algorithms use a relatively large key. The resulting computer processing time makes them less than ideal for data encryption that requires a high transaction rate. Public key systems, therefore, are often restricted to situations in which the characteristics of the public key algorithms have special value, such as digital signatures or key distribution. PKA calculation rates are fast enough to enable the common use of digital signatures.

ICSF supports the following public key algorithms:

- Rivest-Shamir-Adelman (RSA)
- Digital Signature Standard (DSS)

The RSA Public Key Algorithm

The Rivest-Shamir-Adelman (RSA)[3] 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. The ISO 9796 standard and *RSA's Frequently Asked Questions About Today's Cryptography* provide more information about the RSA public key algorithm.

The DSS Public Key Algorithm

The U.S. National Institute of Science and Technology (NIST) Digital Signature Standard (DSS) public key algorithm is based on the difficulty of the discrete logarithm problem. The discrete logarithm problem is to find x given a large prime p, a generator g and a value y = (g**x) mod p. In this equation, ** represents exponentiation. This problem is believed to be very hard when p is sufficiently large and x is a sufficiently large random number. For DSS, p is at least 512 bits, and x is 160 bits. DSS is defined in the NIST Federal Information Processing Standard (FIPS) 186 Digital Signature Standard.

A DSS key pair includes a private and a public key. The DSS private key is used to generate a digital signature, and the DSS public key is used to verify a digital signature.

The Cryptographic Facilities Supported by z/OS ICSF

The cryptographic hardware, or *cryptographic feature*, available to your applications depends on your processor or server model. z/OS ICSF supports the following cryptographic features.

- Cryptographic Coprocessor Feature

  The Cryptographic Coprocessor Feature is available as a feature on the zSeries, S/390 Enterprise Servers and S/390 Multiprise. These complementary metal oxide semiconductor (CMOS) servers are referred to by their shortened names throughout this manual. The Cryptographic Coprocessor Feature supports all the cryptographic algorithms and callable services available with ICSF.

  Restriction: This feature is not supported on the IBM @server zSeries 990.

- PCI Cryptographic Coprocessor Feature

  The PCI Cryptographic Coprocessor is based on a 4758 model 2 standard PCI-bus card available on zSeries processors and as a field upgrade on the S/390 G5 Enterprise Server and on the S/390 G6 Enterprise Server. The PCI Cryptographic Coprocessor supports all the cryptographic algorithms and callable services available with ICSF.

  Restriction: This feature is not supported on the IBM @server zSeries 990.

- PCI Cryptographic Accelerator Feature

  The PCI Cryptographic Accelerator is available on zSeries processors. You must have the CP Assist for Cryptographic Functions feature to have a PCICA on an IBM @server zSeries 990.

- CP Assist for Cryptographic Functions Feature

  The CP Assist for Cryptographic Functions Feature is only available on the IBM @server zSeries 990.

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

You can have a combination of PCI Cryptographic Coprocessors and PCI Cryptographic Accelerators, but the total number must not exceed 16.

For a complete listing of the OS/390 ICSF and z/OS ICSF callable services and the cryptographic features that support them, refer to the *z/OS ICSF Application Programmer's Guide*.

Because the DES algorithm has been used for many years, its strength has been well demonstrated. The DES algorithm can be implemented in both software and specialized hardware. A hardware solution, such as the Cryptographic Coprocessor Feature or the PCI Cryptographic Coprocessor Feature, is often desirable because it provides the following advantages:

- More secure protection to maintain the secrecy of keys
- Greater transaction rates

If a data security threat comes from an external source, a software implementation of the cryptographic algorithm might be sufficient. Unfortunately, however, much fraud originates with individuals within the organization (insiders). As a result, specialized cryptographic hardware can be required to protect against both insider and outsider data security threats. Well-designed hardware can do the following:

- Ensure the security of cryptographic keys
- Ensure the integrity of the cryptographic processes
- Limit the key-management activities to a well-defined and carefully controllable set of services

The Role of Key Secrecy in Data Security

In both the symmetric key and asymmetric key algorithms, no practical means exists to identically cipher data without knowing the cryptographic key. Therefore, it is essential to keep a key secret at a cryptographic node. In real systems, however, this often does not provide sufficient protection. If adversaries have access to the cryptographic process and to certain protected keys, they could possibly misuse the keys and eventually compromise your system. A carefully devised set of processes must be in place to protect and distribute cryptographic keys in a secure manner.

ICSF, and other products that comply with the IBM Common Cryptographic Architecture (CCA), provide a means of controlling the use of cryptographic keys. This protects against the misuse of the cryptographic system.

This manual explains the concepts of key management and gives step-by-step instructions for using ICSF to generate, enter, and manage cryptographic keys.

Chapter 2. Understanding Cryptographic Keys

To understand cryptographic keys, you need to know the types of keys that exist and how ICSF protects them and controls their use. The Integrated Cryptographic Service Facility uses a hierarchical key management approach. A master key protects all the keys that are active on your system. Other types of keys protect keys that are transported out of the system. This chapter gives you an understanding of how ICSF organizes and protects keys.

Values of Keys

Keys can either be clear or encrypted. A clear key is the base value of a key. A clear key is not encrypted under another key. To create an encrypted key, either a master key or a transport key is used to encrypt the base value of the key.

Clear keys, if used carelessly, can compromise security. In symmetric cryptographic processes, such as DES, anyone can use the clear key and the publicly known algorithm to decipher data, key values, or PINs. In asymmetric cryptographic processes it is important to protect the clear value of the private key. It would cause a serious security exposure if the wrong person obtained the value of the private key. It could be used to forge electronic signatures on documents, or decipher key values encrypted under the corresponding public key.

CP Assist for Cryptographic Functions uses clear keys to improve performance for the AES, DES, and TDES algorithms. Note that this feature is only available on the IBM @server zSeries 990.

ICSF uses clear key values to *encode* and *decode* data. You can use the encode and decode callable services or the ICSF utility panels to encode and decode data. For a description of the callable services, see *z/OS ICSF Application Programmer's Guide*. For a description of how to use the utility panels, see Chapter 10, "Using the Utility Panels to Encode and Decode Data".

ICSF may have to input and output clear keys. For example, it might receive and send clear keys when it communicates with other cryptographic systems that use clear keys in their functions. When you give ICSF a clear key value, ICSF can encrypt the key before using it on the system. ICSF has specific callable services that perform this function. These callable services are clear key import and secure key import, which are described in *z/OS ICSF Application Programmer's Guide*.

You can use the ICSF panels to enter and output clear keys. See "Entering Keys into the Cryptographic Key Data Set (CKDS)" on page 26 and "Distributing Cryptographic Keys" on page 31 for a description of how to do this. For a description of entering a clear master key, see "Entering Clear Master Key Parts" on page 51.

Types of Keys

ICSF groups the cryptographic keys into the following categories, which correspond to the functions they perform:

- DES master keys
- Symmetric-keys master key on the PCI Cryptographic Coprocessor
- PKA master keys
- Asymmetric-keys master key on the PCI Cryptographic Coprocessor

- Data-encrypting keys
- Data-translation keys
- MAC keys
- PIN keys
- Transport keys
- PKA keys

Master Keys

ICSF uses master keys to protect other keys. Keys are active on a system only when they are encrypted under a master key variant, so the master key protects all keys that are used on the system. A key is in operational form when it has been encrypted under a master key variant.

Restriction: There are no master keys on the IBM @server zSeries 990.

The ICSF administrator initializes and changes master keys using the ICSF panels. Master keys always remain in a secure area in the cryptographic hardware.

ICSF uses three types of master keys to protect keys that are used with the zSeries and S/390 cryptographic feature:

DES Master Key
The DES master key is a double-length (128-bit) key that is used to protect DES and CDMF keys.

PKA Key Management Master Key
The PKA key management master key (KMMK) is a triple-length (192-bit) key. The KMMK protects PKA private keys that are used in both the digital signature services and in the CDMF and DES data key distribution functions. Support for the PKA KMMK is available only on the Cryptographic Coprocessor Feature on the IBM @server zSeries 900 processors, the S/390 G3 Enterprise Server, or higher and the S/390 Multiprise.

PKA Signature Master Key
The PKA signature master key (SMK) is a triple-length (192-bit) key. The SMK protects PKA private keys that are used only in digital signature services. Support for the PKA SMK is available only on the Cryptographic Coprocessor Feature on the IBM @server zSeries 900 processors, on the S/390 G3 Enterprise Server, or higher and the S/390 Multiprise.

ICSF uses two types of master keys to protect keys that are used with the PCI Cryptographic Coprocessor:

Symmetric-keys Master Key
The symmetric-keys (SYM-MK) master key is a double-length (128-bit) key that is used to protect DES keys used on the PCI Cryptographic Coprocessor. SYM-MK is actually a triple length (192-bit) Master Key that ICSF enforces to be equivalent to a double length (128-bit) master Key. This key must have the same value as the DES master key on the zSeries and S/390 cryptographic feature.

Asymmetric-keys Master Key
The asymmetric-keys (ASYM-MK) master key is a triple-length (192-bit) key. The ASYM master key protects PKA private keys that are used on the PCI Cryptographic Coprocessor. This key must have the same value as the SMK on the zSeries and S/390 cryptographic feature.

Data-Encrypting Keys

Data-encrypting keys, also referred to as data keys, can be single-length, double-length, or triple-length.

Single-length (64-bit) keys are used with the DES algorithm and the CDMF in data confidentiality services. When used with the DES algorithm, the effective key length is 56 bits; the other 8 bits contain parity information. The CDMF algorithm prior to the data confidentiality calculation shortens the data-encrypting keys to an effective length of 40 bits.

Double-length and triple-length DATA keys can be used only with the DES algorithm.

In the operational form, a data key can be used to encipher and decipher data. In the clear form, a data key can be used to encode and decode data on a DES system only. Single-length data-encryption keys can also be used in place of the MAC keys to generate or verify a message authentication code.

Data-Translation Keys

Data-translation keys are single-length (64-bit) keys that protect data that is transmitted through intermediate systems when the originator and receiver do not share a common key. Data that is enciphered under one data-translation key is reenciphered under another data-translation key on the intermediate node. During this process, the data never appears in the clear.

A data-translation key cannot be used in the decipher callable service to decipher data directly. It can translate the data from encipherment under one data-translation key to encipherment under another data-translation key. See “Protection of Data” on page 19 for a description of how data-translation keys protect data that is sent through intermediate systems.

MAC Keys

Message authentication is the process of verifying the integrity of transmitted messages. Message authentication code (MAC) processing enables you to verify that a message has not been altered. You can use a MAC to check that a message you receive is the same one the message originator sent. The message itself may be in clear or encrypted form. MAC keys are either single-length (64-bit) or double-length (128-bit) keys.

Note: In order to generate and use double-length MAC keys in importable or exportable form, the CKDS must contain NOCV-enablement keys and ANSI system keys. You will need to refresh any existing CKDS and add these keys during the process. For information on refreshing a CKDS refer to “Refreshing the CKDS at Any Time” on page 71. When creating a new CKDS, add the NOCV-enablement keys and ANSI system keys during the initialization process. For information on initializing a CKDS, refer to “Initializing the CKDS at First-Time Startup” on page 68.

ICSF uses the following MAC keys in message authentication:

MAC Generation Keys
Before sending a message, an application program can generate an authentication code for the message, using the MAC generate callable service. The callable service computes the message authentication code by using a

MAC generation key to process the message text. The originator of the message sends the message authentication code with the message text.

Single-length MAC generation keys (MAC keys) are used in the ANSI X9.9-1 MAC procedure. They support EMV algorithms. Double-length MAC generation keys (DATAM keys) are used in the ANSI X9.19 optional double key MAC procedure. For compatibility with ICSF Version 2 Release 1, ICSF continues to support the MACD key type, which uses the single-length control vector for both the left and right half of the key to create an external token (MAC || MAC).

MAC Verification Key
The message receiver uses a single-length (MACVER) or double-length (DATAMV) MAC verification key to verify the message authentication code that the message originator sends.

Note: Double-length DATAMV keys are supported only on the S/390 G5 Enterprise Server, or above.

When the receiver gets the message, an application program calls the MAC verify callable service. The callable service verifies a message authentication code by using the MAC verification key to process the message text. It compares the MAC it generates internally with the MAC that was sent with the message. If the two MACs are the same, the message that was sent is identical to the message that was received.

The MAC generation key the sender uses and the MAC verification key the receiver uses have the same clear value. However, each is protected under the master key variant for its key type.

PIN Keys

Personal authentication is the process of validating personal identities in a financial transaction system. The personal identification number (PIN) is the basis for verifying the identity of a customer across the financial industry networks. A PIN is a number that the bank customer enters into an automatic teller machine (ATM) to identify and validate a request for an ATM service.

You can use ICSF to generate PINs and PIN offsets. A PIN offset is a value that is the difference between two PINs. For example, a PIN offset may be the difference between a PIN that is chosen by the customer and one that is assigned by an institution. You can use ICSF to verify the PIN that was generated by ICSF. You can also use ICSF to protect PIN blocks that are sent between systems and to translate PIN blocks from one format to another. A PIN block contains a PIN and non-PIN data. You use PIN keys to generate and verify PINs and PIN offsets, and to protect and translate PIN blocks. All PIN keys are double-length (128-bit) keys.

PIN Keys for Generating and Verifying PINs and PIN Offsets

The following PIN keys generate and verify PINs and PIN offsets:

PIN Generation Key
A PIN generation key is used in an algorithm to generate PINs or PIN offsets.

To generate PINs, use an application program to call the PIN generate callable service. The PIN generation algorithm uses the PIN generation key and some relevant data to generate a clear PIN, a PIN verification value, or an offset.

PIN Verification Key
A PIN verification key is used in an algorithm to verify PINs and PIN offsets.

To verify a supplied PIN, use an application program to call the PIN verification callable service. You need to specify the supplied enciphered PIN block and PIN-encrypting key that enciphers it. You must also specify the PIN verification key, the PIN verification algorithm, and other relevant data. The callable service generates a verification PIN. It compares the supplied PIN and the verification PIN, and if they are the same, it verifies the supplied PIN.

For a specific PIN generation key and PIN verification key pair, the PIN generation key and the PIN verification key have the same clear value. However, each key is protected by the master key variant for its key type.

PIN Keys to Protect and Translate PIN Blocks

The following PIN keys protect and translate PIN blocks:

Output PIN-Encrypting Key

Two systems must share a common key for securely transmitting PIN blocks. The output PIN-encrypting key protects PIN blocks that are sent from your system to another system.

PIN-encrypting keys are used in the PIN translate service. Use the PIN translate service to translate PIN blocks from protection under one PIN-encrypting key to protection under another PIN-encrypting key. You can also use the PIN translate service to translate a PIN block from one PIN block format to another PIN block format. For more information about the PIN translate service, see *z/OS ICSF Application Programmer's Guide*.

Input PIN-Encrypting Key

Two systems must share a common key for securely transmitting PIN blocks. The input PIN-encrypting key protects PIN blocks that are sent from another system to your system.

PIN-encrypting keys are used in the PIN translate service. You also use the input PIN-encrypting key in the PIN verify service. For more information about the PIN translate service and PIN verify service, see *z/OS ICSF Application Programmer's Guide*.

For a specific pair of PIN-encrypting keys, the input PIN-encrypting key and the output PIN-encrypting key have the same clear value. However, each key is protected by the master key variant for its key type.

Cryptographic Variable Keys

These single or double-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.

Transport Keys

Transport keys protect a key that is sent to another system, received from another system, or stored with data in a file. Transport keys are double-length (128-bit) keys.

The following transport keys support the Common Cryptographic Architecture:

Exporter or OKEYXLAT Key-encrypting Key

An exporter or OKEYXLAT 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. Exporter key-encrypting keys are double-length keys. An exporter key is paired with an importer or IKEYXLAT key-encrypting key.

Importer or IKEYXLAT Key-encrypting Key
An importer or an IKEYXLAT 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. Importer key-encrypting keys are double-length keys. An importer key is paired with an exporter or OKEYXLAT key-encrypting key.

For a specific pair of transport keys, the importer key-encrypting key and the exporter key-encrypting key have the same clear value. However, each key is protected by the master key variant for its key type.

ICSF provides the following transport key type to support the ANSI X9.17 standard.

ANSI Key-encrypting Key
An importer and exporter key-encrypting key that is used in the ANSI key management callable services. ANSI key-encrypting keys (AKEKs) are bidirectional and are either single- or double-length keys.

Key Generating Keys

Key-generating keys are double-length keys used to derive unique-key-per-transaction keys.

PKA Keys

ICSF supports the use of public key cryptography on the IBM @server zSeries 900 processors, on the S/390 G3 Enterprise Server, or higher and the S/390 Multiprise. This requires the generation of a pair of PKA keys. One key is made public, and the other key is kept private. The private key is protected through encryption under the appropriate PKA master key. The public key is used to encrypt DES data-encrypting keys in a key distribution system. The private key is then used to decrypt the DES data-encrypting key. The private key is also used for generating digital signatures which are verified using the corresponding public key.

ICSF supports the use of the following PKA keys.

RSA
An RSA key pair includes a private key and a public key. RSA keys can be used for key distribution and authentication. When used for key distribution, a DES key is encrypted under an RSA public key by the sender. The key can only be decrypted with the receiver's private key. When used for authentication, the RSA private key is used for digital signature generation and the RSA public key is used for digital signature verification.

The optional PCI Cryptographic Coprocessor provides the ability to generate RSA public and private key pairs within its secure hardware boundary. In OS/390 V2 R9 ICSF, the PKA key generate callable service has been enhanced to provide support for the generation of RSA keys on the PCI card.

The Cryptographic Coprocessor Feature (CCF) does not provide the ability to generate RSA public and private keys within its secure hardware boundary. If you have CCF without a PCI cryptographic coprocessor, you can generate RSA key pairs in the encrypted form on a TKE Workstation with APAR OW32982 or a workstation with a 4755 or 4758 cryptographic adapter installed. If you are below OS/390 Release 9 and your TKE workstation is below V2.1, then you must order ECA 153. RSA keys generated on the TKE workstation can be loaded directly to the PKDS from the TKE workstation. RSA keys generated on

a non-TKE workstation can use the PKA key import callable service to import the RSA key pair to the Cryptographic Coprocessor Feature.

DSS

A DSS key pair also includes a private and a public key. The DSS private key is used for digital signature generation, and the DSS public key is used for digital signature verification.

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

RSA and DSS public and private keys can be stored in the PKA key data set (PKDS), a VSAM data set. Alternatively, an RSA private key may be retained in the PCI Cryptographic Coprocessor where it was generated. For retained private keys, only the public key is stored in the PKDS. For more information about the PKDS, refer to “Setting Up and Maintaining the PKDS” on page 30.

Protection and Control of Cryptographic Keys

Because the cryptographic algorithms are all key-controlled algorithms, the security of protected data depends on the security of the cryptographic key. With the exception of master keys, which are physically secured, all keys are enciphered under another key to provide this necessary security.

A key is protected under either a master key, a transport key, or a PKA key. The master key protects a key you use on the system. When you send a key to another system, you protect it under a transport key rather than under the master key. You can also use RSA public keys to protect DES data-encrypting keys that are transported between systems.

ICSF controls the use of keys by separating them into types that can be used to do only specific functions.

Master Key Concept

ICSF uses the master key concept to protect cryptographic keys. Master keys, which are stored in secure hardware in the cryptographic feature, are used to encrypt all other keys on the system. All other keys that are encrypted under these master keys are stored outside the protected area of the cryptographic feature. This is an effective way to protect a large number of keys while needing to provide physical security for only a few master keys.

The master keys are used only to encipher and decipher keys. Other key-encrypting keys that are called *transport keys* also encipher and decipher keys and are used to protect cryptographic keys you transmit to other systems. These transport keys, while on the system, are also encrypted under a master key.

Key Separation

The cryptographic hardware, or cryptographic feature, controls the use of keys by separating them into unique types. How a key is used distinguishes it from other keys. The cryptographic feature allows you to use only a specific type of key for its intended purpose. For example, a key that is used to protect data cannot be used to protect a key. Depending on the processor model, you may have one or two of the following cryptographic features:

- Cryptographic Coprocessor Feature

This feature is available on the IBM @server zSeries 900 processors, on the S/390 G3 Enterprise Server, or higher, and the S/390 Multiprise.

- PCI Cryptographic Coprocessor

  This option is available on the IBM @server zSeries 900 processors, on the S/390 G5 and G6 Enterprise Server and works along with the Cryptographic Coprocessor Feature.

Depending on the cryptographic feature, an ICSF system may have up to five master keys.

- A DES master key, which protects keys that are used in DES or CDMF operations on the cryptographic feature.

  In this manual, the term “DES master key” refers to the master key that is used to protect keys that you use in DES or CDMF services. All cryptographic features require a DES master key.
- A symmetric-keys (SYM-MK) master key, which protects keys that are used in DES operations on the PCI Cryptographic Coprocessor
- A PKA key management master key (KMMK), which protects keys that are used in PKA key distribution operations on the Cryptographic Coprocessor Feature.
- A PKA signature master key (SMK), which protects keys that are used in digital signature operations on the Cryptographic Coprocessor Feature.
- An asymmetric-keys (ASYM-MK) master key, which protects RSA keys used in key distribution and authentication operations on the PCI Cryptographic Coprocessor.

DES Master Key Variants Protect DES and CDMF Keys

To provide for key separation, the cryptographic feature automatically encrypts each type of key that is used in either DES or CDMF services under a unique variation of the DES master key. Each variation encrypts a different type of key. Although you define only one master key, in effect you have a unique master key to encrypt each type of key that is used in DES or CDMF services.

A key that is protected under the master key is in *operational form*, which means that ICSF can use it in cryptographic functions on the system. As is shown in Figure 2 on page 15, all keys that you want ICSF to use in cryptographic functions are enciphered under the master key.

Whenever the master key is used to encipher a key, the cryptographic feature produces a variation of the master key according to the type of key that is being enciphered. These variations are called *master key variants*. The cryptographic feature creates a master key variant by exclusive ORing a fixed pattern, called a *control vector*, with the master key. Each type of key that is used in DES or CDMF services has a unique control vector associated with it. For example, the cryptographic feature uses one control vector when the master key enciphers a PIN generation key, and a different control vector when the master key enciphers a PIN verification key.

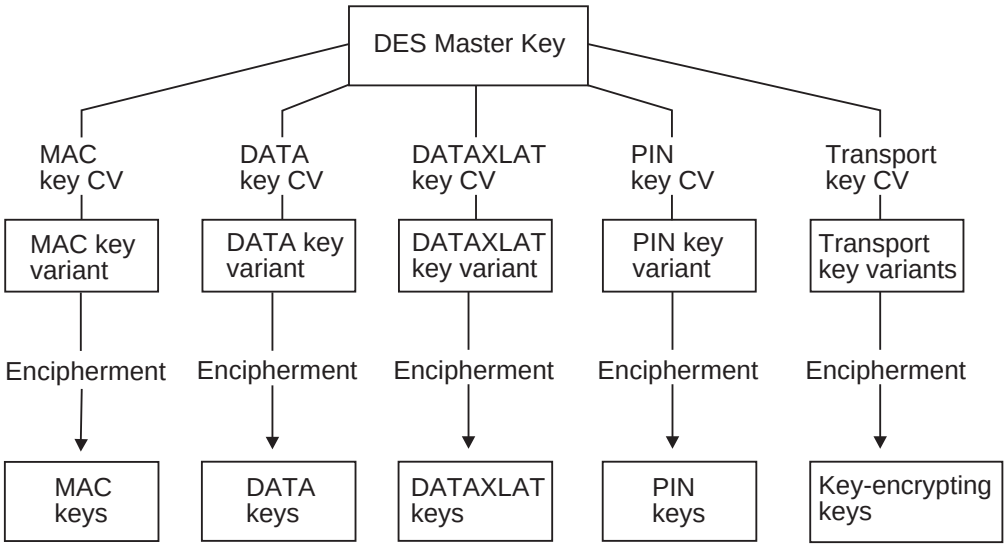


Figure 2. Keys Protected in a System

When systems want to share keys, transport keys can be used to protect keys sent outside of systems. A key that is enciphered under a transport key cannot be used in a cryptographic function. The key must first be brought into a system, deciphered from under the transport key, and enciphered under the system's master key.

ICSF creates variations of a transport key to encrypt a key according to its type. Whenever a transport key is used to encipher a key, the cryptographic feature produces the variation of the transport key according to the type of key that is being enciphered. This allows for key separation when a key is transported off the system.

A transport key variant, also called a *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. See Appendix B, "Control Vector Table" for a listing of the control vector that is used for each key type.

DES cryptographic keys can be single- or double-length keys, depending on their key type. A single-length key is 64 bits, and a double-length key is 128 bits. For double-length keys, one control vector exists for the left half of the key and another control vector for the right half. Therefore, ICSF creates a master key variant or transport key variant for each half of the key the master key or transport key will protect.

Multiple Encipherment

The cryptographic feature uses multiple encipherment when it enciphers a key under a key-encrypting key such as the master key or a transport key. Multiple encipherment is used whenever the key-encrypting key is double-length. The cryptographic feature enciphers each half of the key that it is encrypting.

To multiple-encipher the left half of a key, the cryptographic feature performs the following steps:

1. Exclusive ORs the left half of the key-encrypting key with the control vector for the left half of the key to create the variant. The cryptographic feature then enciphers the left half of the key under this variant.

2. Exclusive ORs the right half of the key-encrypting key with the control vector for the left half of the key to create the variant. The cryptographic feature then deciphers the value that results from step 1 on page 15 under this variant.
3. Exclusive ORs the left half of the key-encrypting key with the control vector for the left half of the key. The cryptographic feature then enciphers the value that results from step 2 under this variant.

To multiple-encipher the right half of the key, the cryptographic feature performs the following steps:

1. Exclusive ORs the left half of the key-encrypting key with the control vector for the right half of the key to create the variant. The cryptographic feature then enciphers the right half of the key under this variant.
2. Exclusive ORs the right half of the key-encrypting key with the control vector for the right half of the key to create the variant. The cryptographic feature then deciphers the value that results from step 1 under this variant.
3. Exclusive ORs the left half of the key-encrypting key with the control vector for the right half of the key. The cryptographic feature then enciphers the value that results from step 2 under this variant.

On ICSF, an effective single-length key can exist as a double-length key; each key half has an identical value. The result of the multiple encipherment process on an effective single-length key is the key value that is encrypted once under the variant.

Migrating from PCF Key Types

Your installation may use Programmed Cryptographic Facility (PCF). ICSF provides key types that are similar to the PCF key types and provides other key types for enhanced key separation and more functions. You cannot use a PCF key on ICSF, but you can convert a PCF key into an ICSF key. Table 1 lists which ICSF key types correspond to the PCF key types.

Table 1. PCF and Corresponding ICSF Key Types

PCF Key Type	**ICSF Key Type**
Local key	Exporter key-encrypting key or Output PIN-encrypting key
Remote key	Importer key-encrypting key or Input PIN-encrypting key
Cross key	Importer key-encrypting key and exporter key-encrypting key or Input PIN-encrypting key and output PIN-encrypting key

ICSF provides compatibility modes and a conversion program to help you run PCF with ICSF and to migrate from PCF to ICSF. The conversion program converts PCF keys to ICSF keys. For information about using the compatibility modes and the conversion program, see *z/OS ICSF System Programmer's Guide*.

Migrating from 4753 Key Storage

TKE Version 3 supplies a 4753 Migration Utility. The utility allows you to migrate internal DES key tokens from the 4753 to ICSF. For details about the TKE 4753 Migration Utility, refer to *z/OS ICSF TKE Workstation User's Guide 2000*.

Protection of Distributed Keys

When you store a key with a file or send it to another system, you can protect the key in either of the following ways:

- Encipher it under a DES transport key.
- Encipher it under the receiver's RSA public key.

When ICSF enciphers a key under a DES transport key, the key is not in operational form and cannot be used to perform cryptographic functions. When you receive a key from a system, the key is enciphered under a transport key. You can reencipher the key from under the transport key to under your master key. You can then use the key on your system. When a key is enciphered under a transport key, the sending system considers it in exportable form, and the receiving system considers it in importable form. When a key is reenciphered from under a transport key to under a system's master key, it is in operational form again.

In an RSA public key cryptographic system, the sending system and receiving system do not need to share complementary importer and exporter key pairs to exchange data-encrypting keys. The sender uses the receiver's public key to encipher the data-encrypting key. The receiver uses his or her own private key to decipher the data-encrypting key. You can use RACF to control which applications can use specific keys and services. For more information, see "Controlling Who Can Use Cryptographic Keys and Services" on page 38.

Protecting Keys Stored with a File

You may want to store encrypted data in a file that is stored on DASD or on magnetic tape. For example, if you use a data-encrypting key to encrypt data in a file, you can store the data-encrypting key with the encrypted data. As is shown in Figure 3, you use an importer key-encrypting key to encrypt the data-encrypting key.

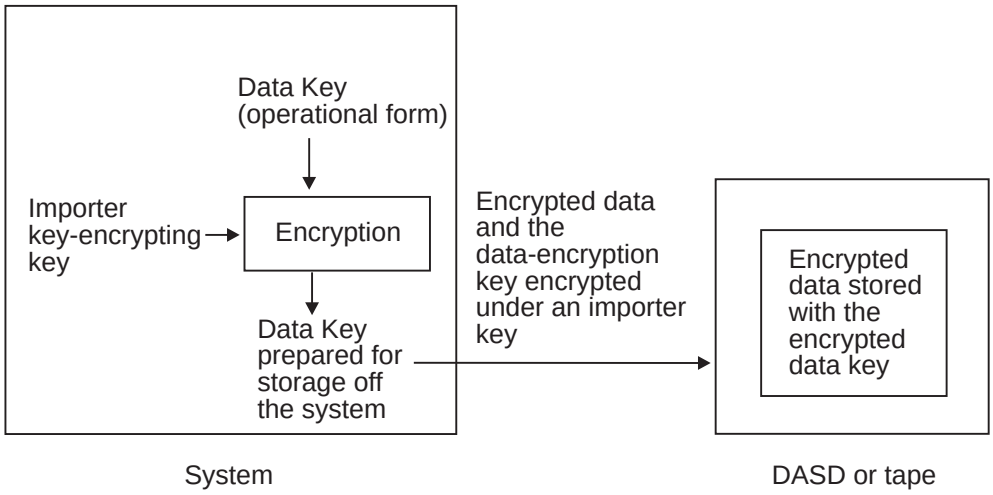


Figure 3. Keys Protected in a File Outside the System

When you encipher a key under an importer key, the key is no longer enciphered under the master key and is no longer operational. You can store the key off the system because the key will not become obsolete if you change the master key. The importer key that protects the data-encrypting key is reenciphered under the correct master key during a master key change. Therefore, when enciphered under the importer key, the data-encrypting key is not directly affected by a master key change.

When you are ready to use the data-encrypting key, use ICSF to reencipher it from under the transport key to under the master key. This makes the data-encrypting key operational. You can then use the data-encrypting key to decrypt the data.

Using DES Transport Keys to Protect Keys Sent between Systems

You can send and receive keys and PINs between your system and another system. For example, if you send encrypted data to another system, you also send the data-encrypting key that enciphered the data. The other system can then use the data-encrypting key to decipher the data. In a financial system, you might need to send a PIN from the system that received the PIN from a customer to a system that uses it to verify a customer's identity. As shown in Figure 4, when you send the PIN between systems, you encipher the PIN under a PIN-encrypting key.

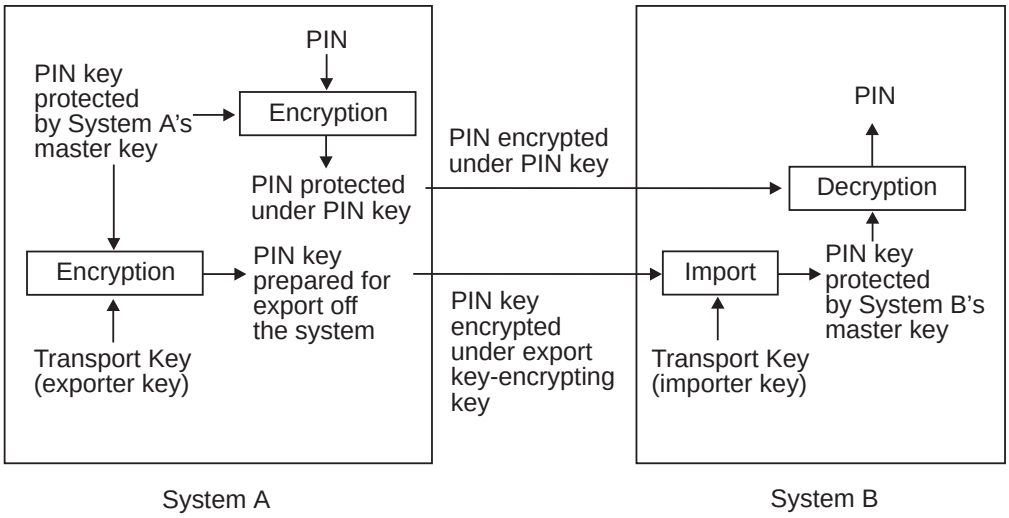


Figure 4. Keys and PINs Protected When Sent between Two Systems

Two systems do not share a master key. When you send a key to another system, you do not encrypt it under a master key. You encrypt it under a transport key.

Two systems that exchange keys share transport keys that have the same clear value. At the sending system, the transport key is an exporter key-encrypting key. At the receiving system, the transport key is an importer key-encrypting key. When the sending system wants to send a key, the sending system encrypts the key under an exporter key-encrypting key. The key is in exportable form on the system that sends the key.

The key is in importable form on the system that receives the key. The receiving system reencrypts the key from under the importer key-encrypting key to under its own master key. The key is then in operational form and can be used on the system.

Using RSA Public Keys to Protect Keys Sent between Systems

The ability to create more-secure key-exchange systems is one of the advantages of combining both DES and PKA support in the same cryptographic system. Because PKA cryptography is more computationally intensive than DES cryptography, it is not the method of choice for all cryptographic functions. It can be used, however, in combination with DES cryptography to enhance the security of key exchange. DES data-encrypting keys can be exchanged safely between two systems when encrypted using an RSA public key. Sending system and receiving system do not need to share a secret key to be able to exchange RSA-encrypted DES data-encrypting keys. An example of this is shown in Figure 5. 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 uses his or her RSA private key to decipher the data-encrypting key.

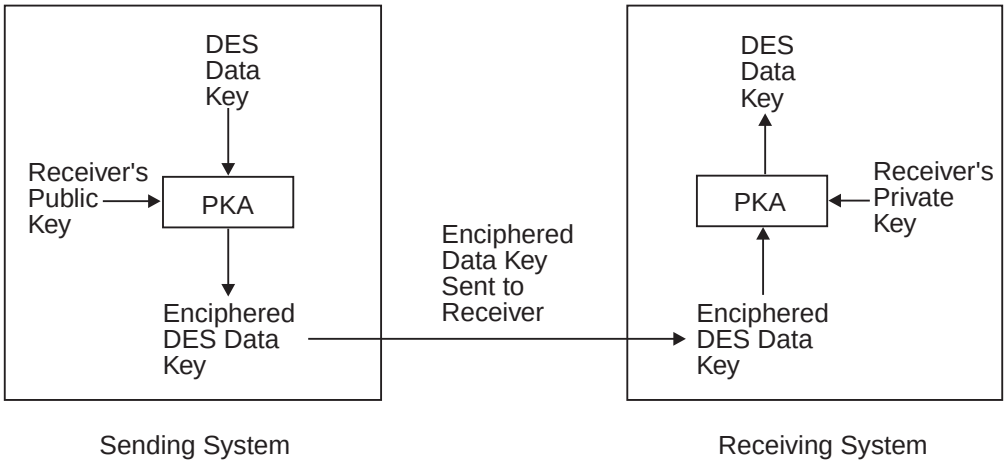


Figure 5. Distributing a DES Data-Encrypting Key Using an RSA Cryptographic Scheme

Note: Only DES and CDMF data-encrypting keys can be encrypted under RSA public keys.

Protection of Data

You use data-encrypting keys to encrypt data. On a system, a data-encrypting key is encrypted under the master key.

A data-encrypting key can encrypt data that is stored in a file outside the system. The data-encrypting key itself is encrypted under a transport key.

You may also need to protect data that you send from one system to another system. The data-encrypting key that protects this data must be sent with the data so that the receiving system can decrypt the data. In this case, the data-encrypting key is encrypted under a transport key.

Sometimes two systems that want to exchange data are not directly connected. There may be intermediate systems between the systems that the data must travel through, as in Figure 6 on page 20.

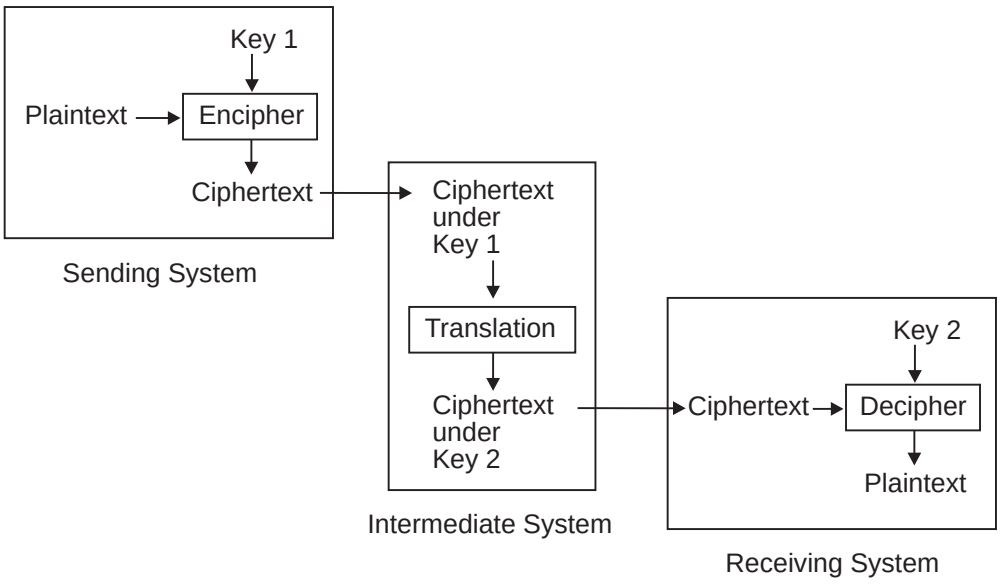


Figure 6. Data Protected When Sent between Intermediate Systems

In this situation, when you pass enciphered data to a system, you do not send a data-encrypting key to decipher the data at the receiving system. Instead, the systems establish pairs of data-encrypting and data-translation keys that exist on the systems. These keys encipher and reencipher the data. The data ends up enciphered under a data-encrypting key that exists on the receiving system. Transport keys may be needed to establish the data-encrypting keys and the data-translation keys on the systems.

Both the sending and receiving systems give data-translation keys to the intermediate system. On the intermediate system, a data-translation key from the sending system matches a data-encrypting key on the sending system. In Figure 6, this key is called *Key 1*. Also on the intermediate system, a data-translation key from the receiving system matches the data-encrypting key on the receiving system. In Figure 6, this key is called *Key 2*. Note that *Key 1* and *Key 2* do not have the same clear key value.

The data-translation keys cannot decipher data. They are used in the ciphertext translate callable service, which reenciphers data from protection under one key to protection under another key.

On the sending system, the plaintext is enciphered under *Key 1*, so it is ciphertext. Then the ciphertext is sent to the intermediate system. At the intermediate system, the data is reenciphered from under *Key 1* to under *Key 2* without appearing as plaintext. When the receiving system receives the ciphertext, the system can decipher the ciphertext from under *Key 2*, so it is plaintext.

Data-translation keys are also used when there is more than one intermediate system between the sending system and receiving system. The sending system and the first intermediate system share a data-encrypting/data-translation key pair. Each pair of neighboring intermediate systems shares a data-translation key pair. The final intermediate system and the receiving system share a data-translation/data-encrypting key pair.

Triple DES for Privacy

ICSF supports triple DES encryption for data privacy. This provides stronger encryption than the current DES algorithm and single-length DES data-encryption keys. Triple DES uses three, single-length keys to encipher and decipher the data which results in a stronger form of cryptography.

Data that has been encrypted under a double-length or triple-length DATA key cannot be reenciphered using data-translation keys as described in “Protection of Data” on page 19.

Advanced Encryption Standard (AES)

ICSF supports the Advanced Encryption Standard (AES) algorithm for data privacy. This provides strong encryption. Data can be encrypted and decrypted using 128-bit, 192-bit, and 256-bit keys. The algorithm has the same availability as triple DES.

Chapter 3. Managing Cryptographic Keys

To perform cryptographic services, you need to know how to create, maintain, and use cryptographic keys. This chapter gives an overview on entering master keys, generating keys, creating and maintaining the cryptographic key data sets (CKDS and PKDS), and entering keys into the CKDS. This chapter also discusses distributing keys and controlling access to keys, and presents a summary of key use.

If you are running on an IBM @server zSeries 990, see Appendix E, “Running HCR7708 on an IBM @server zSeries 990”, on page 217.

Generating Cryptographic Keys

Using ICSF, you can generate keys by using either the key generator utility program (KGUP) or the key generate callable service. Both KGUP and the key generate callable service create all types of keys except PKA keys and ANSI X9.17 keys. KGUP stores the key that it generates in the CKDS. The key generate callable service returns the key to the application program that called it, instead of storing it in the CKDS. The application program can then call the dynamic CKDS update service to store the key in the CKDS.

Generating PKA Keys

If the PCI Cryptographic Coprocessor Feature (PCICC) is installed, ICSF is able to generate RSA keys using the PKA Key Generate service. The RSA key format can be the Modulus Exponent form or the Chinese Remainder form. Retained keys are RSA keys generated within the secure boundary of the PCICC and never leave the secure boundary. Only the domain that created the retained key can access it. Retained key format can be the Modulus Exponent form or the Chinese Remainder form. For more information on how to retain a generated key, see *z/OS ICSF Application Programmer’s Guide*.

Key Generator Utility Program (KGUP)

You can use KGUP to generate keys in either an operational form or an exportable form. When KGUP generates a key in the operational form, it stores it in the cryptographic key data set (CKDS). When KGUP generates a key in exportable form, you can send it to another system.

To specify the function that you want KGUP to perform, you use KGUP control statements. For a detailed description of how to use the program to generate keys, see Chapter 7, “Managing Cryptographic Keys by Using the Key Generator Utility Program”, on page 95.

Key Generate Callable Service

The key generate callable service generates a single key or a pair of keys. Unlike KGUP, the key generate callable service does not store the keys in the CKDS but returns them to the application program that called the service. The application program can then call the dynamic CKDS update service to store the keys in the CKDS.

When you call the key generate callable service, you pass parameters that specify information about the key you want generated. The key generate callable service generates keys in the following possible forms:

- Operational, if the master key protects it
- Importable, if an importer key-encrypting key protects it
- Exportable, if an exporter key-encrypting key protects it

To use ICSF you need to enter master keys and operational keys. For more information about the key generate callable service, see *z/OS ICSF Application Programmer's Guide*.

Entering Keys

This section gives you an overview of key entry and the methods of key entry.

Entering Master Keys

Master keys are used to protect all cryptographic keys that are active on your system. The number and types of master keys you need to enter depends on your hardware configuration.

- A DES master key protects DES keys and PKA master keys protect DSS and RSA keys.
- On the optional PCI Cryptographic Coprocessor, the symmetric-keys master key (SYM-MK) protects symmetric keys such as DES keys and the asymmetric-keys master key (ASYM-MK) protects RSA keys.

The first time you start ICSF on your system, you must enter master keys and initialize the cryptographic key data set (CKDS). You can then generate and enter the keys you use to perform cryptographic functions. The master keys you enter protect the keys stored in the CKDS and the PKDS.

Because master key protection is essential to the security of the other keys, ICSF stores the master keys within the secure hardware of the cryptographic feature. This nonvolatile key storage area is unaffected by system power outages, because it is protected by a battery power unit. The values of the master keys never appear in the clear outside the cryptographic feature.

Managing the master key involves the following tasks:

- Entering the master keys the first time you start ICSF
- Reentering the master keys if they are cleared
- Changing the DES master key periodically
- Changing the PKA master keys periodically

The types of master keys you can enter and the steps you take to enter master keys depend on your system processor and hardware features.

For either the Cryptographic Coprocessor Feature or the PCI Cryptographic Coprocessor, you can use any of the following methods to enter the master keys:

- Pass Phrase Initialization

  The pass phrase initialization utility allows the casual user of ICSF to set both the DES and PKA master keys on theCryptographic Coprocessor Features, set the SYM-MK and ASYM-MK on the PCI Cryptographic Coprocessors, and initialize the CKDS with a minimal effort. For steps in using the pass phrase initialization utility, refer to Chapter 4, "Using the Pass Phrase Initialization Utility", on page 45.
- Clear Master Key Entry panels

  The clear master key entry panels are enhanced ISPF panels through which you enter master key parts in the clear. You can use these panels to enter master key parts into both the Cryptographic Coprocessor Feature and the PCI

Cryptographic Coprocessor. The master key parts appear briefly in the clear in MVS host storage within the address space of the TSO user before being transferred to the secure hardware. Within the boundaries of the secure hardware, the key parts are combined to produce the master key. The clear master key part entry panels provide a level of security for master key entry that is at least equivalent to that provided with CUSP and superior to that provided with PCF. Clear master key part entry is provided for installations where the security requirements do not warrant the additional expense and complexity of the optional TKE workstation. For clear master key entry steps on the coprocessors, seeChapter 5, “Managing Master Keys”, on page 51.

- Trusted Key Entry (TKE) workstation

  The TKE workstation is an optional hardware feature available from IBM Customized Solutions. The TKE workstation uses a variety of public key cryptographic techniques to ensure both the integrity and privacy of the logically secure master key transfer channel. You can use a single TKE workstation to set up master keys in all Cryptographic Coprocessor Features and PCI Cryptographic Coprocessors within a server complex. For information on using the TKE workstation, see *z/OS ICSF TKE Workstation User's Guide 2000*.

After you have entered the master keys, choose option 1 on the ICSF Initialize a CKDS panel to do the following:

- Create the CKDS header record.
- Activate the DES master key and read the CKDS into storage.
- Create keys that ICSF uses for internal processing, and read the CKDS into storage again.

If you wish to add ANSI, NOCV, or Enhanced System Keys to your CKDS, choose the appropriate option. Refresh the CKDS.

Servers or processor models may have up to two cryptographic coprocessor features and multiple PCI crytographic coprocessor features. The master keys must be the same for all coprocessors accessed by the same operating system.

Entering System Keys into the Cryptographic Key Data Set (CKDS)

The ICSF CKDS has several sets of system keys. These are the keys with labelname of X'00' and are installed during CKDS initialization. The system keys are required in the CKDS. Other keys are optional; however, their absence will affect functions in many services.

If the system keys are not in the CKDS, an 18F abnormal end with reason code X'A1' can occur. If the ANSI, NOCV enablement, or the ESYS keys are not in the CKDS, an 18F abnormal end with reason code X'A3' can occur.

The following is a summarization of where the keys are used:

- Required System Keys

  These keys are used to validate CKDS entries and used in many services. These keys are required.

- NOCV-enablement Keys
  - These keys are needed for all services where NOCV key-encrypting keys are required. See *z/OS ICSF Application Programmer's Guide* for more information.
  - These keys are needed in CSNBKGN and KGUP where replicated keys are generated, that is, where key length of SINGLE is specified for double-length keys.

  - These keys are used during VP generation on a CDMF-only system.
  - These keys are used by CSNBSBC on a CDMF-only system.
  - These keys are used during CKDS conversion.
  - These keys are required to export and import double-length DATAM and DATAMV keys.
- ANSI System Keys
  - These keys are used by CSNBSBD on a CDMF-only system.
  - These keys are used when installing the extended system keys (ESYS) on the CKDS initialization panel.
  - These keys are needed for key part import services.
  - These keys are required for key test service CSNBKYT if there are no PCICCs active.
  - These keys are required to generate double-length DATAM and DATAMV keys in the importable form.
- Extended System Keys

  These keys are required for symmetric key export if there are no PCICCs active.

Entering Keys into the Cryptographic Key Data Set (CKDS)

All DES keys except the DES master key can be stored in the CKDS. There are several methods you can use to enter keys into the CKDS.

- Key generator utility program (KGUP)

  Regardless of your processor or server model, you can use KGUP to enter keys into the CKDS.
- Dynamic CKDS update callable services

  Regardless of your processor or server model, you can program applications to use the dynamic CKDS update callable services to enter keys into the CKDS.
- Trusted Key Entry (TKE) workstation

  With the TKE workstation you can load key parts for operational (PIN and transport) keys into a key queue. To load these key parts into the CKDS, you must also use the ICSF Operational Key panel and perform a CKDS refresh. For more information, refer to *z/OS ICSF TKE Workstation User's Guide 2000*.

The table in Table 2 shows which keys can be entered by each of these methods.

Table 2. Methods for Entering Each Key Type into the CKDS

Key Type	KGUP	Dynamic Update	TKE Workstation
PIN	X	X	X
Importer and Exporter key-encrypting keys	X	X	X
Data-encrypting	X	X	
Data-translation	X	X	
MAC and MACVER	X	X	
DATAM and DATAMV	X	X	
ANSI key-encrypting keys		X	
IMP-PKA keys		X	X
Non-standard CV keys		X	

Entering Keys by Using the Key Generator Utility Program

One function that KGUP performs is to enter key values that you supply into the CKDS. You can enter a clear or encrypted key value by using KGUP.

You submit KGUP control statements to specify to KGUP the function that you want KGUP to perform. To enter a key, you specify the key value in a KGUP control statement. You can either specify an encrypted or clear key value.

When you enter an encrypted key value, the key value must be encrypted under an importer key-encrypting key that exists in the CKDS. You use the KGUP control statement to specify which importer key-encrypting key encrypts the key. KGUP reenciphers the key from under the importer key-encrypting key to under the master key and places the key in the CKDS.

When you enter a clear key value, KGUP enciphers the clear key value under the master key and places the key in the CKDS. Because entering clear keys may endanger security, ICSF must be in special secure mode before you can enter a clear key by using KGUP. Special secure mode lowers the security of your system to allow you to use KGUP to enter clear keys, and to produce clear PINs.

To use special secure mode, several conditions must be met.

- The installation options data set must specify YES for the SSM installation option.

  For information about specifying installation options, see *z/OS ICSF System Programmer's Guide*.
- The environmental control mask (ECM) must be configured to permit special secure mode.

  The ECM is a 32-bit mask that is defined for each crypto domain during hardware installation. The second bit in this mask must have been turned on to enable special secure mode.
- If you are running in LPAR mode, special secure mode must be enabled

  You enable special secure mode during activation using the Crypto page of the Customize Activation Profiles task. After activation, you can enable or disable special secure mode on the Change LPAR Crypto task. Both of these tasks can be accessed from the Hardware Master Console.

If these conditions permit the use of special secure mode, it is enabled automatically when you specify that you are entering clear key values in a KGUP statement.

For a detailed description of how to use KGUP to enter keys, see Chapter 7, "Managing Cryptographic Keys by Using the Key Generator Utility Program", on page 95.

Entering Keys by Using the Dynamic CKDS Update Services

ICSF provides a set of callable services that allow applications to dynamically update the CKDS. Applications can use these services to create, write, and delete records from the CKDS. These dynamic updates affect both the DASD copy of the CKDS currently in use and the in-storage copy. Another service allows an application to retrieve the key token from a record in the in-storage CKDS. That token can be used directly in subsequent CALLs to cryptographic services. The key part import callable service combines the clear key parts and returns the key value either in an internal key token or as a dynamic update to the CKDS. For more information on using the dynamic CKDS update services or the key part import service, refer to *z/OS ICSF Application Programmer's Guide*.

Entering Keys into the PKDS

You can store RSA and DSS public and private keys in the PKA key data set (PKDS). ICSF provides a set of callable services that allow applications to update the PKDS. Applications can use some of these services to create, write, and delete records from the PKDS. To improve performance and eliminate I/O, ICSF maintains a cache of frequently used PKDS records.

For more information on using the PKDS update services, refer to the *z/OS ICSF Application Programmer's Guide*.

RSA private keys can also be stored in the PKDS from TKE. For more information, see *z/OS ICSF TKE Workstation User's Guide 2000*.

Maintaining Cryptographic Keys

You can use either KGUP or the dynamic CKDS update services to generate and enter keys into the cryptographic key data set (CKDS), or to maintain keys already existing in the CKDS. The keys are stored in records. A record exists for each key that is stored in the CKDS.

A record in the CKDS is called a *key entry* and has a label associated with it. When you call some ICSF callable services, you specify a key label as a parameter to identify the key for the callable service to use.

Use KGUP to change the key value of an entry, rename entry labels, and delete entries in the CKDS. For more information about how to use KGUP to update key entries in the CKDS, see Chapter 7, "Managing Cryptographic Keys by Using the Key Generator Utility Program", on page 95.

Use the dynamic CKDS update services in applications to create entries, change the key value of an entry, and delete entries in the CKDS.

You can use RACF to control which applications can use specific keys and services. For more information, see "Controlling Who Can Use Cryptographic Keys and Services" on page 38.

Setting Up and Maintaining the Cryptographic Key Data Set (CKDS)

The cryptographic key data set (CKDS) stores operational keys of all types. It contains an entry for each key.

Note: PKA keys are stored in the PKA key data set (PKDS) and not in the CKDS.

Keys that are stored in the CKDS are encrypted under the appropriate variants of the DES master key. Before you generate keys that you store in the CKDS, you must define a DES master key to your system. You define a master key by entering its value and setting it so it is active on the system. After you enter the master key, you must make it active on the system by setting it when you initialize the CKDS. For information about entering and setting the master key and initializing CKDS , see Chapter 5, "Managing Master Keys", on page 51.

Once you define a master key, you generate keys and store them in the CKDS. You use KGUP to generate keys and change key values and other information for a key entry in the CKDS. For more information about running KGUP, see Chapter 7, "Managing Cryptographic Keys by Using the Key Generator Utility Program", on page 95

page 95. You can also program applications to use callable services to generate keys and change key information in the CKDS. For more information about how to use callable services to update key entries in the CKDS, see *z/OS ICSF Application Programmer's Guide*. You can use the optional TKE workstation to load key parts for operational (PIN and transport) keys into a key queue through a secure logical channel. To load these key parts into the CKDS, you must also use the ICSF Operational Key panel and perform a CKDS refresh. For more information on using the TKE workstation, see *z/OS ICSF TKE Workstation User's Guide 2000*.

When you initialize ICSF, the system obtains space in storage for the CKDS. For more information about initializing space for the CKDS, see *z/OS ICSF System Programmer's Guide*.

Besides the in-storage CKDS, there is a copy of the CKDS on disk. Your installation can have many disk copies of CKDSs, backup copies, and different disk copies. For example, an installation may have a separate CKDS with different keys for each shift. When a certain shift is working, you can load the CKDS for that shift into storage. Then only the keys in the CKDS loaded for that shift can be accessed for ICSF functions. However, only one disk copy is read into storage at a time.

You use KGUP to make changes to any disk copy of the CKDS. When you use KGUP to generate and maintain keys, or enter keys directly into the KSU, you change only the disk copy of a CKDS. Therefore, you can change keys in the disk copy of the data set without disturbing ICSF functions that are using the keys in the in-storage copy of the data set. To make the changes to the disk copy of the CKDS active, you need to replace the in-storage CKDS using the refresh utility. When you use the dynamic CKDS update callable services to change entries in the CKDS, you change both the in-storage copy of the CKDS and the disk copy. This allows for the immediate use of the new keys without an intervening refresh of the entire CKDS. Figure 7 shows that ICSF callable services use keys in the in-storage copy of the CKDS.

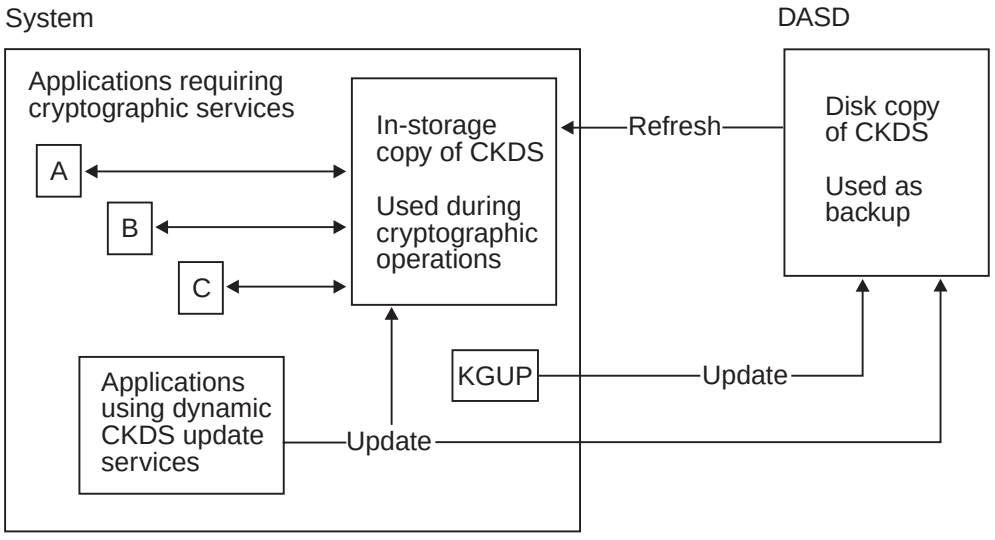


Figure 7. Updating the In-Storage Copy and the Disk Copy of the CKDS

You just specify the name of the disk copy of the CKDS when you run KGUP. You can also read any disk copy of the CKDS into storage, by specifying the name of the disk copy of the CKDS on a Refresh In-Storage CKDS panel. You can also run a utility program to read a disk copy of the CKDS into storage. However, the disk

copy must be enciphered under the correct master key. All the copies of your disk copies of the CKDS should be enciphered under the same master key.

Your installation should periodically change the master key. To change the master key, you enter a new master key value and make that value active. The keys in a CKDS must then be enciphered under the new master key. Therefore, before you make the new master key active, the CKDS must be reenciphered from under the current master key to under the new master key.

First, you reencipher the disk copy of the CKDS under the new master key. Then you activate the new master key using the change master key option. This option automatically replaces the old in-storage CKDS with the disk copy that is reenciphered under the new master key. If you have multiple disk copies of CKDSs, reencipher all of them under the new master key before changing the master key.

You can reencipher a CKDS under a new master key by using the master key panels or a utility program. For more information about reenciphering a CKDS, see “Changing the DES Master Key and Reenciphering the CKDS” on page 76.

Note: When you perform any functions that affect the in-storage copy of the CKDS, you should consider temporarily disallowing the dynamic CKDS update services. Functions that affect the in-storage copy of the CKDS include changing the master key, reenciphering, or refreshing. For more information, refer to “Disallowing Dynamic CKDS Updates During KGUP Updates” on page 96.

If running in a sysplex, see Chapter 6, “Running in a Sysplex Environment”, on page 91.

Setting Up and Maintaining the PKDS

RSA and DSS public and private keys can be stored in the PKA key data set (PKDS), a VSAM data set. The PKDS is maintained as an external data set only. ICSF optionally maintains a cache of frequently used PKDS records. If a PKDS is updated, its cache record is deleted or reread from DASD on its next usage. Applications can use the dynamic PKDS callable services to create, write, read and delete PKDS records.

The PKDS is automatically initialized at ICSF startup. There are internal and external tokens in the PKDS. External tokens may be used irrespective of the PKA master keys. Internal tokens, however, can only be used if they are encrypted under the current PKA signature master key (SMK), the key management master key (KMMK), or the asymmetric-keys master key (ASYM-MK). The PKDS cache is refreshed automatically whenever ICSF is started or when the PKDS cache is reenciphered or activated. For additional information, see “Reenciphering and Activating the PKDS” on page 84.

Changing PKA master keys should be done with care. For information on initializing the PKDS, see *z/OS ICSF System Programmer’s Guide*. Also see “PKA Master Keys and the PKDS” on page 79 for information on the keys.

You can program applications to use the PKDS callable services to create entries, change entries and delete entries in the PKDS. For more information about how to use callable services to update key entries in the PKDS, see *z/OS ICSF Application Programmer’s Guide*.

If running in a sysplex, see Chapter 6, “Running in a Sysplex Environment”, on page 91.

Distributing Cryptographic Keys

With ICSF you can develop key distribution systems as defined in any of the following:

- The IBM Common Cryptographic Architecture
- The ANSI X9.17 Standard
- The Public Key Cryptographic Standard

These key distribution systems are explained in the following sections.

Common Cryptographic Architecture Key Distribution

ICSF provides protection for keys when the keys are sent outside your system. You must generate complementary keys for key distribution. A complementary pair of keys has the following characteristics:

- The keys have the same clear key value.
- The key types are different but complementary.
- Each key usually exists on a different system.

Complementary keys are the following types:

- Importer key-encrypting key and exporter key-encrypting key (transport keys)
- PIN generation key and PIN verification key
- Input PIN-encrypting key and output PIN-encrypting key
- MAC generation key and MAC verification key
- Data-encrypting key and data-translation key
- Input key translate and output key translate keys

When protected data is sent between intermediate systems, the following keys exist as complementary keys. For more information about this situation, see “Protection of Data” on page 19.

- Data-encrypting key and data-translation key
- Data-translation key and data-translation key

The same data-encrypting key can also exist on two different systems so that both systems can encipher and decipher the data.

You can use ICSF to protect keys that are distributed across networks. You distribute keys across a network for some of the following reasons:

- When you send encrypted data to another system, you send the data-encrypting key with the data or before it.
- When you share complementary keys with another system.

Transport keys protect keys being sent to another system. When a key leaves your system, an exporter key-encrypting key encrypts the key. When another system receives the key, the key is still encrypted under the same key-encrypting key, but the key-encrypting key is now considered an importer key-encrypting key. The exporter key-encrypting key at the sending system and the importer key-encrypting key at the receiving system must have the same clear value. Before two systems can exchange keys, they must establish pairs of transport keys.

In Figure 8 on page 32 System A wants to send an output PIN-encrypting key to System B.

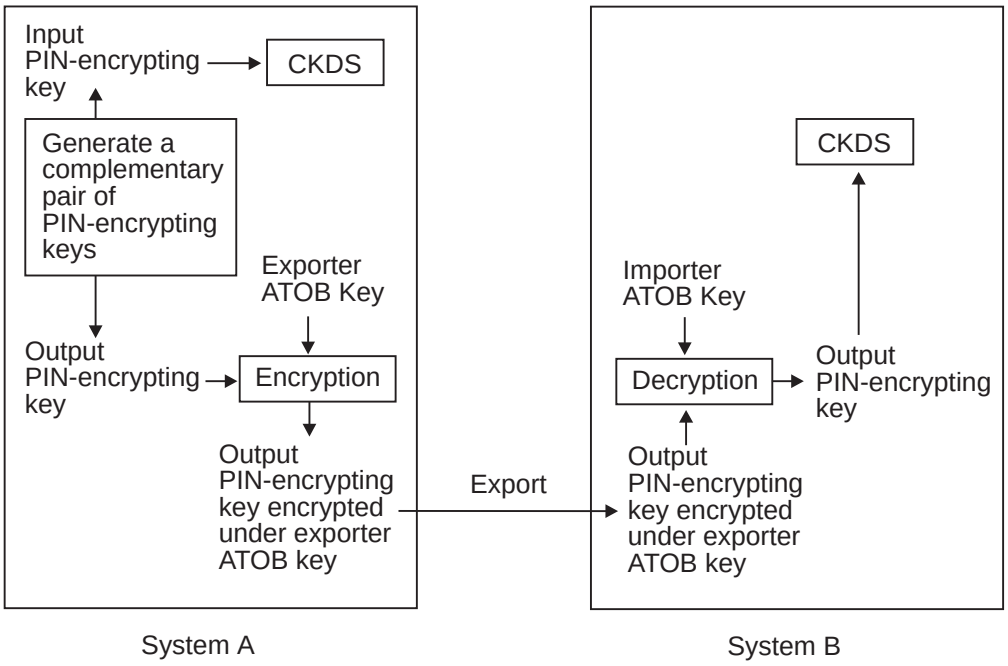


Figure 8. Key Sent from System A to System B

Before sending the key, System A and System B must establish a pair of transport keys between them. System A has an exporter key-encrypting key called Exporter ATOB, which has the same key value as the importer key-encrypting key called Importer ATOB at System B. This pair of transport keys is unidirectional, because they are used only for distributing keys from System A to System B.

When System A generates the input PIN-encrypting key, the system also creates a complementary output PIN-encrypting key. System A enciphers the input PIN-encrypting key under System A's master key and stores the input PIN-encrypting key in the CKDS. It encrypts the complementary output PIN-encrypting key under the Exporter ATOB key so it can send the output PIN-encrypting key to System B. System B decrypts the output PIN-encrypting key using the Importer ATOB key, and encrypts the output PIN-encrypting key under System B's master key.

For the systems to send keys in both directions, they must establish two pairs of transport keys at each site, as in Figure 9 on page 33.

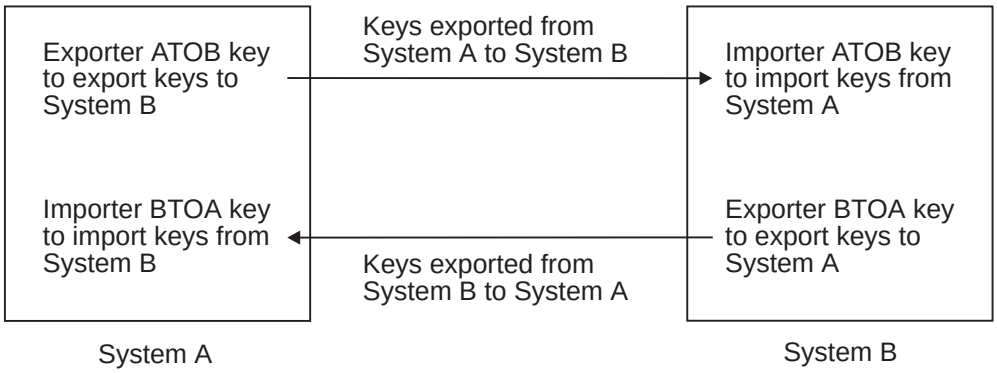


Figure 9. Keys Sent between System A and System B

To send keys from System A to System B, use the key generator utility program (KGUP) to establish an importer and exporter complementary key pair. You establish an exporter key, Exporter ATOB key, on System A and establish the complementary importer key, Importer ATOB key, on System B. Then when System A sends a key to System B, System A sends the key in exportable form encrypted under Exporter ATOB key. When System B receives the key, System B considers the key in importable form encrypted under Importer ATOB key.

To send keys from System B to System A, use KGUP to establish an importer and exporter complementary key pair. You establish an exporter key, Exporter BTOA key, on System B and the complementary importer key, Importer BTOA key, on System A. When System B sends a key to System A, System B sends the key in exportable form encrypted under Exporter BTOA key. When System A receives the key, System A considers the key in importable form encrypted under Importer BTOA key.

KGUP can create a pair of complementary keys, one key in operational form, and its complement in exportable form. You can also use KGUP to receive keys that are in importable form. When you want KGUP to create a key value in exportable form or import a key value in importable form, you specify the transport key that encrypts the key value. For more information about using KGUP for key distribution, see Chapter 7, “Managing Cryptographic Keys by Using the Key Generator Utility Program”, on page 95.

You can also use one of two callable services to reencipher a key from operational form into exportable form. Both the key export callable service and the data key export callable service reencipher a key from encryption under the master key to encryption under an exporter key-encrypting key.

You can call the key import callable service to convert a key from importable form to operational form. The key import callable service reenciphers a key from encryption under an importer key-encrypting key to encryption under the system’s master key.

With interlinked computer networks, sensitive data passes through multiple nodes before reaching its final destination. The originator and the receiver do not share a common key. Data-translation keys are shared between the originator and an intermediate system, between two intermediate systems, and between an intermediate system and the receiver system. As the data is passed along between these systems, they must reencipher it under the different data-translation keys without it ever appearing in the clear. Each system can call the ciphertext translate

callable service to do this function. For a description of sending data between intermediate systems, see “Protection of Data” on page 19.

ANSI X9.17 Key Distribution

ICSF provides callable services that allow you to develop key distribution systems that adhere to the ANSI X9.17 standard.

When protected data is sent between two systems, it is protected by data-encrypting keys. The same data-encrypting key exists on two different systems so that both systems can encipher and decipher the data.

Before two systems can exchange keys, they must establish a shared transport key, the ANSI key-encrypting key (AKEK), which is distributed manually. This transport key is bidirectional, and can be used for distributing keys in both directions between System A and System B, as shown in Figure 10.

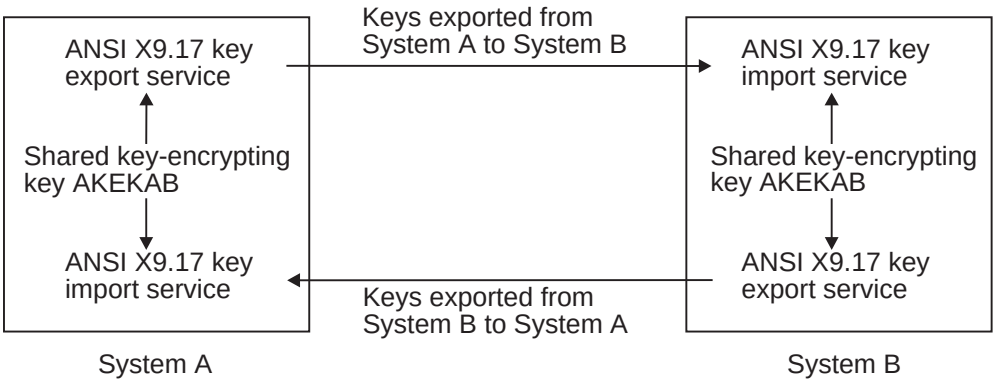


Figure 10. ANSI X9.17 Keys Sent between System A and System B

System A generates the data-encrypting key, enciphers it under System A’s master key, and stores it in the CKDS. System A uses the ANSI X9.17 key export callable service to encrypt the data-encrypting key under the shared transport key, AKEKAB, and export it to System B. System B then uses the ANSI X9.17 key import callable service to decrypt the data-encrypting key using the shared transport key, AKEKAB, and then encrypts it under System B’s master key. The shared transport key is coupled with source and destination identifiers for System A and System B, and a message counter as defined in the ANSI offset and notarization processes.

The shared ANSI key-encrypting key is bidirectional. System B can also send keys to System A. The systems can also exchange data keys along with the AKEK used to encrypt them. The AKEKs are themselves encrypted under the transport AKEK.

ANSI X9.17 key distribution can take place in several types of environments:
- Point-to-point environment
- Key distribution center environment
- Key translation center environment

For more information on ANSI X9.17 key distribution, refer to the ANSI X9.17 Standard.

Public Key Cryptographic Standard Key Distribution

ICSF provides support for the Public Key Cryptographic Standard (PKCS). PKSC is a set of standards for public-key cryptography developed by RSA Data Security, Inc.

An example of using RSA public-key cryptography to distribute DES and CDMF data-encrypting keys is presented in “Using RSA Public Keys to Protect Keys Sent between Systems” on page 19.

Summary of Key Use

Table 3 summarizes the use of keys for different cryptographic functions. For each cryptographic function, the table identifies the type of key or keys you can use and the callable service an application can call to perform the task.

Note: For the following services, you do not need a key:
- Character/nibble conversion (CSNBXBC and CSNBXCB)
- Code conversion (CSNBXAE and CSNBXEA)
- Modification detection code generate (CSNBMDG and CSNBMDG1)
- One-way hash generate (CSNBOWH and CSNBOWH1)
- Random number generate (CSNBRNG)
- X9.9 data editing (CSNB9ED)
- Control Vector Generate (CSNBCVG)

Table 3. Summary of Key Use

Cryptographic Function	Required Key	Callable Service
Change control vector in an external key token	• External key token • Key-encrypting key	Control Vector Translate
Change PKA tokens from old encipherment to new	• Internal RSA or DSS private key • Key-token	PKA Key Token Change
Compose data into a form required by SET protocol	• RSA public key	Set Block Compose
Convert any non-ANSI key from operational form into exportable form	• Internal key token • Exporter key-encrypting key	Key export
Convert a data key from importable form into operational form	• Internal key token • Exporter key-encrypting key	Data key import
Convert a data key from operational form into exportable form	• Internal key token • Exporter key-encrypting key	Data key export
Convert a DES data encrypting key to encryption under an RSA public key	• Internal key token of a data key • RSA public key	Symmetric Key Export
Convert a non-ANSI key from importable form into operational form	• External key token • Importer key-encrypting key	Key import
Convert a clear data key into operational form	• Clear data key	Clear key import Multiple clear key import
Convert any clear non-ANSI key into importable or operational form	• Clear key • Importer key-encrypting key	Secure key import Multiple secure key import

Table 3. Summary of Key Use (continued)

Cryptographic Function	Required Key	Callable Service
Decipher data	• Encrypted data-encrypting key for decipher	Decipher
	• Clear data-encrypting key	Decode Symmetric key decipher
Decompose data from a form required by SET protocol	• RSA private key	Set Block Decompose
Decrypt an RSA key encrypted PKCS 1.2 formatted key value	• RSA private key	PKA Decrypt
Encipher data	• Encrypted data-encrypting key for encipher	Encipher
	• Clear data-encrypting key	Encode Symmetric key encipher
		Cryptographic Variable Encipher
Encrypt a clear key value under an RSA public key	• RSA public key	PKA Encrypt
Encrypt a clear PIN	• Output PIN-encrypting key	Clear PIN Encrypt
Encrypt a text block including clear key value	• Internal/External key token • SECMSG Key-identifier	Secure messaging for keys
Encrypt a text block including clear PIN block	• PIN encrypting key • SECMSG Key-identifier	Secure messaging for PINs
Extract a PKA public key from a PKA private key token	• PKA private key token	PKA public key extract
Generate a clear VISA PIN validation value from an encrypted PIN block	• Input PIN-encrypting key or • Output PIN-encrypting key	Clear PIN generate alternate
Generate a VISA CVV or MasterCard CVC	• Internal key token1 • Internal key token2	VISA CVV Service Generate
Generate a digital signature	• PKA private key	Digital signature generate
Generate a message authentication code	• MAC generation key or • Data-encrypting key	MAC generate
Generate encrypted PINs	• PIN generation key • Output PIN encrypting key	Encrypted PIN Generate
Generate single-length and double-length MAC keys	• Exporter key-encrypting key	User derived key
Generate clear PINs	• PIN generation key	Clear PIN generate Clear PIN generate alternate
Generate a symmetric key in two forms (DES-encrypted and PKA public key-encrypted)	• Key-encrypting key (optional) • RSA public key	Symmetric key generate
Generate a unique key for transaction	• Key generating key	Diversified Key Generate

Table 3. Summary of Key Use (continued)

Cryptographic Function	Required Key	Callable Service
Generate or verify secure verification patterns for keys	• Key-encrypting key	Key test Key test extended
Import, export, or translate keys according to the ANSI X9.17 standard	• ANSI key-encrypting key	ANSI X9.17 key import ANSI X9.17 key export ANSI X9.17 key translate Key part import
Import an encrypted PKA private key	• External PKA key token • Importer key-encrypting key	PKA key import
Import a DES data-encrypting key enciphered under an RSA public key	• DES data-encrypting key protected under an RSA public key • Corresponding RSA private key	Symmetric key import
Modify an operational key so that it cannot be exported	• Internal key token	Prohibit Export
Prohibit the export of an external key token from a receiving node	• Exporter key-encrypting key	Prohibit export extended
Transform a CDMF data-encrypting key to a transformed, shortened DES data-encrypting key	• Data-encrypting key	Transform CDMF key
Translate an external key token from encryption under one key-encrypting key to encryption under another key-encrypting key	• External key token • Key-encrypting key1 • Key-encrypting key2	Key translate
Translate text from one data key to another in a multiple system network	• Data-translation key1 • Data-translation key2	Ciphertext translate
Verify a digital signature	• PKA public key	Digital signature verify
Verify a message authentication code	• MAC verification key or • Data-encrypting key or • MAC generation key	MAC verify
Verify a VISA CVV or MasterCard CVC	• Internal key token1 • Internal key token2	VISA CVV Service Verify
Verify PINs	• PIN verification key • Input PIN-encrypting key	PIN verify
Translate PINs	• Input PIN-encrypting key • Output PIN-encrypting key	PIN translate

Controlling Who Can Use Cryptographic Keys and Services

You can use the z/OS SecureWay Security Server RACF, to control which applications can use specific keys and services. This can help you ensure that keys and services are used only by authorized users and jobs. You can also use RACF to audit the use of keys and services.

To set up these controls, create and maintain RACF general resource profiles in the CSFKEYS class, and in the CSFSERV class. The CSFKEYS class controls access to cryptographic keys, and the CSFSERV class controls access to ICSF services

If you are not the RACF security administrator, you need to ask for assistance from that person. To use the auditing capabilities of RACF, you need to ask for reports from a RACF auditor. Your installation's security plan should show who is responsible for maintaining these RACF profiles and auditing their use.

The following procedure describes one approach to doing this:

1. Decide whether you will protect keys, services, or both. You can select which keys and services to protect.
2. You may want to organize the users who need access to ICSF keys and services into groups. To do this, obtain a list of the user IDs of users who need to use ICSF keys and services. If batch jobs or started tasks need to use ICSF, obtain the user IDs under which they will run.

   Group any of the user IDs together if they require access to the same keys and services. For example, you might want to set up groups as follows:
   - Users who work with MAC-related callable services
   - Users who work with PIN-related callable services
   - Users who work with a particular MAC, or a particular PIN
   - Users who call applications to dynamically update the CKDS
   - Users who perform functions available on the User Control Functions panel

   Usually, all users of ICSF should have access to keys and services by virtue of their membership in one of these RACF groups, rather than specific users. This is because RACF maintains the access lists in in-storage profiles. When the in-storage profiles are created or changed, the in-storage profiles must be refreshed. (Merely changing them in the RACF data base is not sufficient. This is analogous to the in-storage CKDS maintained by ICSF.) To refresh the in-storage RACF profiles, the RACF security administrator must use the SETROPTS command:

   ```
   SETROPTS RACLIST(CSFKEYS) REFRESH

   SETROPTS RACLIST(CSFSERV) REFRESH
   ```

   If you place *RACF groups* in the access lists of the RACF profiles, you can change a user's access to the protected services and keys by adding or removing the user from the groups. Ask your RACF security administrator to create the RACF groups.

   You should also ask your RACF security administrator to connect you to these groups with CONNECT group authority. This permits you to connect and remove users from the groups.

   For example, the security administrator could issue the following commands:

```
ADDGROUP groupid

CONNECT  your-userid  GROUP(groupid)  AUTHORITY(CONNECT)
```

With CONNECT group authority, you are able to connect other users to the groups:

```
CONNECT  other-userid  GROUP(groupid)
```

With CONNECT group authority, you are also able to remove users from the groups:

```
REMOVE  other-userid  GROUP(groupid)
```

3. Ask your RACF security administrator for the authority to create and maintain profiles in the CSFKEYS and CSFSERV general resource classes. Usually, this is done by assigning a user the CLAUTH (class authority) attribute in the specified classes. For example, the security administrator can issue the following command:

   ```
   ALTUSER  your-userid  CLAUTH(CSFKEYS CSFSERV)
   ```

4. If you want to use generic profiles that contain characters such as * and %, ask your RACF security administrator to activate generic profile checking in the CSFKEYS and CSFSERV classes:

   ```
   SETROPTS  GENERIC(CSFKEYS CSFSERV)
   ```

   Note: Using generic profiles has several advantages. Using generic profiles you can reduce the number of profiles that you need to maintain. You can also create a "top" generic profile that can be used to protect all keys and services that are not protected by a more specific profile.

5. Define profiles in the CSFKEYS and CSFSERV classes. For further instructions, see "Setting Up Profiles in the CSFKEYS General Resource Class" and "Setting Up Profiles in the CSFSERV General Resource Class" on page 40.

Setting Up Profiles in the CSFKEYS General Resource Class

To set up profiles in the CSFKEYS general resource class, take the following steps:

1. Define appropriate profiles in the CSFKEYS class:

   ```
   RDEFINE  CSFKEYS label  UACC(NONE)
            other-optional-operands
   ```

   where *label* is the label by which the key is defined in the CKDS or PKDS (this is not the transport key label). Note that if an application uses a token instead of a key label, no authorization checking is done on the use of the key.

   Notes:

   a. If you have ICSF/MVS Version 1 Release 1 profiles that specify *key-type.label*, you need to change them to specify only *label*.
   b. As with any RACF profile, if you want to change the profile later, use the RALTER command. To change the access list, use the PERMIT command as described in the next step.
   c. If you have already started ICSF, you need to refresh the in-storage profiles. See Step 3.
   d. You can specify other operands, such as auditing (AUDIT operand), on the RDEFINE or RALTER commands. The NOTIFY operand is ignored when specified for profiles in the CSFKEYS class.

e. If the RACF security administrator has activated generic profile checking for the CSFKEYS class, you can create generic profiles using the generic characters * and %. This is the same as any RACF general resource class.

2. Give appropriate users (preferably groups) access to the profiles:

```
PERMIT  profile-name  CLASS(CSFKEYS)
        ID(groupid)  ACCESS(READ)
```

3. When the profiles are ready to be used, ask the RACF security administrator to activate the CSFKEYS class and refresh the in-storage RACF profiles:

```
SETROPTS  CLASSACT(CSFKEYS)

SETROPTS RACLIST(CSFKEYS) REFRESH
```

Setting Up Profiles in the CSFSERV General Resource Class

To set up profiles in the CSFSERV general resource class, take the following steps:

1. Define appropriate profiles in the CSFSERV class:

```
RDEFINE  CSFSERV service-name  UACC(NONE)
         other-optional-operands
```

Where *service-name* is one of the following:

CSFAEGN	ANSI X9.17 EDC generate callable service
CSFAKEX	ANSI X9.17 key export callable service
CSFAKIM	ANSI X9.17 key import callable service
CSFAKTR	ANSI X9.17 key translate callable service
CSFATKN	ANSI X9.17 key transport key partial notarize callable service
CSFDKG	Diversified key generate callable service
CSFCKI	Clear key import callable service
CSFCKM	Multiple Clear Key Import
CSFCMK	Change master key panel service
CSFCPA	Clear PIN generate alternate
CSFCPE	Clear PIN Encrypt
CSFCSG	VISA CVV Service Generate
CSFCSV	VISA CVV Service Verify
CSFCTT	Cipher text translate callable service
CSFCTT1	Cipher text translate (with ALET) callable service
CSFCVE	Cryptographic Variable Encipher
CSFCVT	Control Vector Translate
CSFDCO	Decode callable service
CSFDEC	Decipher callable service
CSFDEC1	Decipher (with ALET) callable service
CSFDKCS	Clear master key entry panel service (PCICC)
CSFDKEF	Clear master key entry panel service (CCF)
CSFDKM	Data Key Import
CSFDKX	Data key export callable service

CSFDSG	Digital signature generate callable service
CSFDSV	Digital signature verify callable service
CSFECO	Encode callable service
CSFEDC	Compatibility service for the CUSP or PCF CIPHER macro
CSFEMK	Compatibility service for the CUSP or PCF EMK macro
CSFENC	Encipher callable service
CSFENC1	Encipher (with ALET) callable service
CSFEPG	Encrypted PIN Generate
CSFGKC	Compatibility service for the CUSP or PCF GENKEY macro
CSFKEX	Key export callable service
CSFKGN	Key generate callable service
CSFKIM	Key import callable service
CSFKPI	Key part import callable service
CSFKRC	Key record create callable service
CSFKRD	Key record delete callable service
CSFKRR	Key record read callable service
CSFKRW	Key record write callable service
CSFKTR	Key Translate
CSFKYT	Key test callable service
CSFKYTX	Key test extended callable service
CSFMDG	MDC generate callable service
CSFMDG1	MDC generate (with ALET) callable service
CSFMGN	MAC generate callable service
CSFMGN1	MAC generate (with ALET) callable service
CSFMVR	MAC verify callable service
CSFMVR1	MAC verify (with ALET) callable service
CSFOWH	One-way hash generate callable service
CSFOWH1	One-way hash generate (with ALET) callable service
CSFPCI	PCI interface
CSFPCM	PCICC management panel service
CSFPEX	Prohibit Export
CSFPEXX	Prohibit export extended callable service
CSFPGN	PIN generate callable service
CSFPKD	PKA decrypt callable service
CSFPKDR	PKDS reencipher panel service; PKDS activate
CSFPKE	PKA encrypt callable service
CSFPKG	PKA key generate

CSFPKI	PKA key import callable service
CSFPKRC	PKDS Record Create
CSFPKRD	PKDS Record Delete
CSFPKRR	PKDS Record Read
CSFPKRW	PKDS Record Write
CSFPKSC	PKSC interface
CSFPKTC	PKA key token change
CSFPKX	PKA public key extract
CSFPMCI	Pass phrase master key/CKDS initialization panel service
CSFPTR	PIN translate callable service
CSFPVR	PIN verify callable service
CSFREFR	Refresh CKDS panel service
CSFRENC	Reencipher CKDS panel service
CSFRKD	Retained key delete callable service
CSFRKL	Retained key list callable service
CSFRNG	Random number generate callable service
CSFRSWS	User control functions (TSO panel)
CSFRTC	Compatibility service for the CUSP or PCF RETKEY macro
CSFSBC	SET Block Compose
CSFSBD	SET Block Decompose
CSFSKI	Secure key import callable service
CSFSKM	Multiple Secure Key Import
CSFSKY	Secure messaging for keys
CSFSMK	Set master key panel service
CSFSPN	Secure messaging for PINs
CSFSSWS	User control functions (TSO panel)
CSFSYG	Symmetric key generate callable service
CSFSYI	Symmetric key import callable service
CSFSYX	Symmetric key export callable service
CSFTCK	Transform CDMF key callable service
CSFUDK	User derived key callable service

Notes:

a. As with any RACF general resource profile, if you want to change the profile later, use the RALTER command. To change the access list, use the PERMIT command as described in the next step.

b. If you have already started ICSF, you need to refresh the in-storage profiles. See Step 3 on page 43.

c. You can specify other operands, such as auditing (AUDIT operand), on the RDEFINE or RALTER commands. The NOTIFY operand is ignored when specified for profiles in the CSFSERV class.

d. If the RACF security administrator has activated generic profile checking for the CSFSERV class, you can create generic profiles using the generic characters * and %. This is the same as with any RACF general resource class. You *cannot* use RACF variables (generic profiles that are defined using an &) for the CSFSERV class.

Example

If generic profile checking is in effect, the following profiles enable you to specify which users and jobs can use the ciphertext translate callable services. No other services can be used by any job on the system. The user ID specified on the NOTIFY keyword enables you to determine who is using any other protected ICSF services.

```
RDEFINE  CSFSERV  CSFCTT   UACC(NONE)

RDEFINE  CSFSERV  CSFCTT1  UACC(NONE)

RDEFINE  CSFSERV  *        UACC(NONE)
         NOTIFY(userid)
```

2. Give appropriate users (preferably groups) access to the profiles:

```
PERMIT  profile-name  CLASS(CSFSERV)
        ID(groupid)  ACCESS(READ)
```

3. When the profiles are ready to be used, ask the RACF security administrator to activate the CSFKEYS class and refresh the in-storage RACF profiles:

```
SETROPTS  CLASSACT(CSFSERV)

SETROPTS RACLIST(CSFSERV) REFRESH
```

Controlling PCICC Services

This section only applies if you have a TKE workstation. For non-TKE users, all access control points are enabled.

If you use TKE to administer your systems, new access control points must be enabled before the services are available.

Whether the various services are enabled or disabled on your system is dependent upon TKE workstation installation. Prior to TKE Version 3.1, only ISPF services could be updated. With TKE Version 3.1, access control points for API and UDX services can be updated.

If you have never installed a TKE workstation on your system, all services (ISPF and API) will be enabled when you first logon to the workstation. (Note that for UDXs with access control points, enablement of UDX access control points requires a TKE workstation.)

To list the access control points that are enabled, see “Displaying PCICC Default Roles” on page 174.

If, however, you have previously installed a TKE Version 3 workstation, your ISPF service settings will be the same as those for your existing system. The API settings will also be the same as your existing system, except for the new access control points (which are disabled). The UDX access control points would all be disabled.

As new access control points are added, they are enabled for new (first-time) TKE installations. For existing TKE installations, API services would reflect what had

been enabled/disabled in Version 3.1, and new access control points would be disabled. UDX support is implemented likewise. If your installation wants to use UDX callable services, the corresponding access control point must be enabled.

For more information, see *z/OS ICSF TKE Workstation User's Guide 2000*.

Chapter 4. Using the Pass Phrase Initialization Utility

The pass phrase initialization utility allows the casual user of ICSF to install the necessary master keys on both the Cryptographic Coprocessor Features and the PCI Cryptographic Coprocessors, and initialize the CKDS with a minimal effort. This chapter describes how to use this utility to get up and running quickly.

Performing Pass Phrase Initialization

With ICSF, you can install the DES and PKA master keys and initialize the CKDS on the Cryptographic Coprocessor Features by using the pass phrase initialization utility. You can also use this utility to install both the SYM-MK and ASYM-MK on any PCI Cryptographic Coprocessors on zSeries Servers or S/390 G5 Enterprise Servers, or above. The pass phrase is case sensitive and should be chosen according to the following rules:

- It can contain a minimum of 16 and a maximum of 64 characters.
- It can include any characters in the EBCDIC character set.
- It can contain imbedded blanks, but leading and trailing blanks are truncated.

The pass phrase initialization utility can be used to initialize the system and to initialize PCI Cryptographic Coprocessors that are brought online after system initialization. To use this utility, special secure mode must be enabled and all master key registers must be empty. You cannot use this utility to change master keys. To change master keys you need to use either the clear master key entry panels or the TKE workstation.

Since the same pass phrase will always produce the same master key values, you should secure the pass phrase in a safe place.

Before Running the Pass Phrase Initialization Utility

Before you run the pass phrase initialization utility for the first time, you must initialize ICSF by following these steps:

1. Install the ICSF program product according to the instructions in *z/OS and z/OS.e Planning for Installation* and *z/OS Program Directory*.
2. Create an empty CKDS.
3. Create an empty PKDS.
4. Create an installation options data set.
5. Create an ICSF startup procedure.
6. Start ICSF.
7. Access the ICSF panels.

These steps are described in *z/OS ICSF System Programmer's Guide*

Running the Pass Phrase Initialization Utility

After you start ICSF, you can use the ICSF panels to run the pass phrase initialization utility. When you access the ICSF panels, the primary menu panel appears. Note that the ICSF FMID appears in the upper left hand corner (it will toggle to the panel identification ID). See Figure 11 on page 46.

```
HCR7708 ------------- Integrated Cryptographic Service Facility ---------
OPTION ===> 6

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

      Licensed Materials - Property of IBM

     5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 11. Selecting the Pass Phrase Initialization Option on the ICSF Primary Menu Panel

1. Select option 6, PPINIT, and press ENTER to begin the pass phrase initialization utility.

   The Pass Phrase MK/CKDS Initialization panel appears. See Figure 12.

```
CSFPMC00 ------- ICSF - Pass Phrase MK/CKDS Initialization ---
Command ===>
Enter your pass phrase and the name of the CKDS:

Pass Phrase (16 to 64 characters)
===>

CKDS
===>

Initialize the CKDS? (Y/N) ===>
Signature MK = Key Management MK? (Y/N) ===>
Initialize new PCICCs only? (Y/N) ===>



Press ENTER to process.
Press END   to exit to the previous menu.
```

Figure 12. ICSF Pass Phrase MK/CKDS Initialization Panel

2. Type the pass phrase and the data set name in the spaces that are provided. Refer to the example in Figure 13 on page 47.

   The CKDS name must be a valid MVS data set.

   Note: If you are reentering master keys after they have been cleared, use the same pass phrase as when you originally entered the keys. You should have saved the pass phrase in a secure place after you entered the master keys previously.

3. Answer the ″Initialize the CKDS?″ question by typing your response in the space following the question.
   a. If the CKDS has not been initialized, type Y.

      If you select Y, the CKDS name must refer to a valid, uninitialized CKDS.
   b. If this is an existing CKDS, type N.

      If you select N, the CKDS must have already been initialized with the pass phrase initialization utility and the identical pass phrase.

      ICSF checks and refreshes the existing CKDS.
4. Answer the ″Signature MK = Key Management MK?″ question by typing your response in the space following the question.
   a. If you have a new system with PCI Cryptographic Coprocessors installed, type Y.

      The signature master key and the key management master key will have the same value as the ASYM master key on the PCI Cryptographic Coprocessors. This increases the flexibility in routing services among the cryptographic coprocessors.
   b. If you have previously used pass phrase initialization and you have PKA key tokens that are encrypted under a key management master key that cannot be recreated, type N.
5. Answer the ″Initialize new PCICCs only?″ question by typing your response in the space following the question.
   a. If you have already initialized your system with the Pass Phrase Initialization utility and now want to initialize new PCI cards, type Y.
   b. If this is the first time you are running the Pass Phrase Initialization Utility, type N.

```
CSFPMC00 --------- ICSF - Pass Phrase MK/CKDS Initialization ----------------
Enter your pass phrase and CKDS data set name:

Pass Phrase (16 to 64 characters)
===> winnie the pooh and tigger too

CKDS
===> CRYPTO.CKDS.JAN1996

Initialize the CKDS? (Y/N) ===> Y
Signature MK = Key Management MK? (Y/N) ===> Y
Initialize new PCICCs only? ===> N
```

Figure 13. Entering Options on the Pass Phrase MK/CKDS Initialization Panel

6. Press ENTER to run the utility.

   This utility uses the pass phrase, a series of constants, and the MD5 hash algorithm to:
   - Calculate the DES master key and load the new master key registers on the Cryptographic Coprocessor Features with the value.
   - Use the value of the DES master key as the value of the SYM-MK key and load the new master key registers on the PCI Cryptographic Coprocessors with the value.
   - Calculate the PKA master keys and set the PKA signature master key register and the PKA key management master key register with these values. If you specified ″Y″ for the question about making the signature master key

equal to the key management master key, then the value calculated for the key management master key will be used for both PKA master keys.

- Use the value of the PKA signature master key as the value of the ASYM-MK and set the new asymmetric-keys master key registers on the PCI Cryptographic Coprocessors with the value.
- Set the master key register.
- Initialize the CKDS or refresh an existing CKDS.

For details of these calculations, refer to “Pass Phrase Initialization Master Key Calculations” on page 211.

Messages on the bottom half of the panel display the progress of the utility.

7. When the utility has completed successfully, press END to return to the primary menu.

Adding PCICC after First Time Pass Phrase Initialization

The pass phrase initialization utility can be used to initialize PCI Cryptographic Coprocessors after system initialization. The procedure is to re-run the Pass Phrase Initialization Utility.

Note: Special Secure Mode is not required when adding PCICC after first time pass phrase initialization.

The step-by-step procedure is:

1. Run the Pass Phrase Initialization Utility.

   Access the primary menu panel.

```
CSF@PRIM ------------- Integrated Cryptographic Service Facility ---------
OPTION ===> 6

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

       Licensed Materials - Property of IBM

      5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
      US Government Users Restricted Rights - Use, duplication or
      disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 14. Selecting the Pass Phrase Initialization Option on the ICSF Primary Menu Panel

2. Select option 6, PPINIT, and press ENTER to begin the pass phrase initialization utility.

The Pass Phrase MK/CKDS Initialization panel appears. See Figure 15.

```
CSFPMC00 ------- ICSF - Pass Phrase MK/CKDS Initialization ---
Command ===>
Enter your pass phrase and the name of the CKDS:

Pass Phrase (16 to 64 characters)
===>

CKDS
===>

Initialize the CKDS? (Y/N) ===>
Signature MK = Key Management MK? (Y/N) ===>
Initialize new PCICCs only? (Y/N) ===>




Press ENTER to process.
Press END   to exit to the previous menu.
```

Figure 15. ICSF Pass Phrase MK/CKDS Initialization Panel

3. Type the pass phrase and the data set name in the spaces that are provided. Refer to the example in Figure 16.

   The CKDS name must be the current, active CKDS.

   Note: You are reentering master keys and must use the same pass phrase as when you originally entered the keys. You should have saved the pass phrase in a secure place after you entered the master keys previously.
4. The ″Initialize the CKDS?″ and ″Signature MK = Key Management MK?″ questions are ignored.
5. Answer the ″Initialize new PCICCs only″ question by typing your response in the space following the question. Your response should be Y.

```
CSFPMC00 --------- ICSF - Pass Phrase MK/CKDS Initialization ----------
Enter your pass phrase and CKDS data set name:

Pass Phrase (16 to 64 characters)
===> winnie the pooh and tigger too

CKDS
===> CRYPTO.CKDS.JAN1996

Initialize the CKDS? (Y/N) ===> N
Signature MK = Key Management MK? (Y/N) ===> Y
Initialize new PCICCs only? ===> Y
```

Figure 16. Entering Options on the Pass Phrase MK/CKDS Initialization Panel

6. Press ENTER to run the utility.

   For details of these calculations, refer to “Pass Phrase Initialization Master Key Calculations” on page 211.

   Messages on the bottom half of the panel display the progress of the utility.

7. When the utility has completed successfully, press END to return to the primary menu.

Chapter 5. Managing Master Keys

This chapter describes how to use the clear master key entry panels to enter master keys in both types of cryptographic coprocessors: CCF and PCICC.

You can have up to two Cryptographic Coprocessor Features on each IBM @server zSeries 900 processor, S/390 G6 Enterprise processor, IBM @server zSeries 800, or S/390 Multiprise. Each Cryptographic Coprocessor Feature is capable of performing cryptographic functions and holding the master keys within a secure boundary.

You can have multiple PCI Cryptographic Coprocessors on the IBM @server zSeries 900 processor, on the IBM @server zSeries 800, or on the S/390 G6 Enterprise Server with field upgrade. You can have multiple PCI Cryptographic Coprocessors and PCI Cryptographic Accelerators on the IBM @server zSeries 900 processors and IBM @server zSeries 800. There can be a total of 16.

Each PCI Cryptographic Coprocessor is capable of performing cryptographic functions and holding the master keys within a secure boundary. The PCI Cryptographic Coprocessors work in conjunction with the Cryptographic Coprocessor Features on your server.

Restriction: The CCF and PCICC are not available on the IBM @server zSeries 990 processor.

Requests for cryptographic services are routed to either the PCICC or CCF, depending on key types specified in the request. In order for these two types of cryptographic coprocessors to work together, you need to install the same master key values for each coprocessor.

Note: The PCI Cryptographic Accelerators improve private key decryption performance. They do not require setting of master keys.

Entering Clear Master Key Parts

You can use the Clear Master Key Entry panels to enter clear master key parts. The way you obtain master key parts depends on the security guidelines in your enterprise. You may receive master key parts from a key distribution center or you may generate your own key parts using the ICSF random number utility.

When you enter the PKA master keys and the asymmetric-keys master key (ASYM-MK) the first time, the PKA callable services are initially disabled. Once you have entered the PKA master keys and the ASYM-MK, you must enable the PKA callable services for these services to work. Before you change the PKA master keys and the ASYM-MK, you need to disable the PKA callable services. To enable and disable the PKA callable services refer to “Enabling and Disabling PKA Services” on page 79.

To enter master key parts that you do not generate using the random number utility, continue with “Entering the First Master Key Part” on page 58.

To begin master key entry by generating random numbers for the key parts, continue with “Generating Master Key Data for Clear Master Key Entry” on page 52.

Generating Master Key Data for Clear Master Key Entry

If you intend to use the clear key entry panels to enter master keys, you need to generate and record the following values before you begin:

- Key parts
- Checksums
- Verification patterns (optional)
- Hash patterns (optional)

Note: If you are reentering master keys after they have been cleared, use the same master key part values as when you originally entered the keys. You should have saved the key part values in a secure place after you entered the master keys previously.

A DES master key is 16 bytes long. A symmetric-keys master key (SYM-MK) is 24 bytes long. ICSF enforces the SYM-MK to be 16 bytes long. ICSF defines these master keys by exclusive ORing two or more key parts. Each of the master key parts is also 16 bytes long. To enter either a DES master key or a SYM-MK, you must enter a first key part and a final key part. If you choose to, you can also enter one or more intermediate key parts after entering the first key part and before entering the final key part.

Note: The combined DES master key is forced to have odd parity, but the parity of the individual key parts can be odd, even or mixed. We refer to even or mixed parity keys as non-odd parity keys.

Attention: The PCICC will not allow certain 'weak' keys as master keys. The list of weak keys are documented in *IBM 4758 PCI Cryptographic Coprocessor CCA Basic Services Reference and Guide Version 2.40 for the IBM 4758 Models 002 and 023* under the Master_Key_Process verb. If you have an existing CCF installed with a weak master key, you can not install that master key in the PCICC. You must change the CCF master keys and load those same master keys in the PCICCs.

PKA master keys and the ASYM-MKs are each 24 bytes long. ICSF defines these master keys by exclusive ORing two or more key parts. Each of the PKA master key parts is also 24 bytes long.

If you are using ICSF to generate random numbers, generate a random number for each key part that you need to enter to create the master key.

Note: It is recommended that you enter the same key value for the SMK and KMMK of the Cryptographic Coprocessor Feature and the ASYM-MK of the PCI Cryptographic Coprocessor Feature. This will allow ICSF flexibility in workload balancing.

A 16-byte key part consists of 32 hexadecimal digits. A 24-byte key part consists of 48 hexadecimal digits. To make this process easier, each part is broken into segments of 16 digits each.

When you are manually entering the master key parts, you also enter a checksum that verifies whether you entered the key part correctly. A checksum is a two-digit result of putting a key part value through a series of calculations. The coprocessors calculate the checksum with the key part you enter and compare the one they calculated with the one you entered. The checksum verifies that you did not transpose any digits when entering the key part. If the checksums are equal, you have successfully entered the key.

After you enter a key part and its checksum for a DES master key or SYM-MK, the coprocessor calculates an eight-byte verification pattern and sixteen byte hash pattern. After you enter a key part and its checksum for a PKA master key (SMK, KMMK or ASYM-MK), the coprocessor calculates a sixteen-byte hash pattern.

Before the verification and hash patterns can be calculated, the DES master key must have been set.

The ICSF Clear Master Key Entry panel displays the verification pattern or hash pattern. Check the displayed verification pattern against the optional verification pattern you may have generated at the time you generated the DES or SYM-MK master key parts and the checksum. Check the displayed hash pattern against the optional hash pattern that you may have generated at the same time you generated the PKA master key part and the checksum. The verification pattern or hash pattern checks whether you entered the key part correctly, and whether you entered the correct key type.

ICSF displays a verification and hash pattern for each DES master key part. It also displays a verification and hash pattern for the DES master key after you enter all the key parts. If the verification and hash patterns are the same, you have entered the key part correctly. Likewise, in addition to displaying a hash pattern for each PKA master key part, ICSF also displays a hash pattern for the PKA master key after you enter all the key parts. If the hash patterns are the same, you have entered the key part correctly.

Note: Keys stored in the CKDS are enciphered under the DES master key. The master key verification pattern is stored in the CKDS header record. Checking the verification pattern is optional; it is not required for key entry.

To generate the value for a key part, you can use one of the following methods:

- Choose a random number yourself.
- Access the ICSF utility panels to generate a random number.
- Call the random number generate callable service. For more information, see *z/OS ICSF Application Programmer's Guide*.

  Note: ICSF must be initialized with a DES master key before you can use the random number generate callable service or the Random Number Generator panel.

The following topics describe using the ICSF utilities to generate key parts, checksums, verification patterns, and hash patterns.

Generating Key Parts Using ICSF Utilities

1. Access ICSF utilities by choosing option 5, UTILITY, on the Primary Menu panel, as shown in Figure 17 on page 54.

```
CSF@PRIM --------- Integrated Cryptographic Service Facility ---------
OPTION ===> 5

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions


      Licensed Materials - Property of IBM

     5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 17. Selecting the Utility Option on the ICSF Primary Menu Panel

The Utilities panel appears. See Figure 18. You use the RANDOM and CHECKSUM options to generate random numbers, checksums, and verification patterns for master key management.

Attention: If you are running on an IBM @server zSeries 990, this panel has changed. See “TSO panels” on page 220.

```
CSFUTL00 ---------------- ICSF - Utilities --------------------------
OPTION ===> 3


Enter the number of the desired option.

  1  ENCODE        -  Encode data
  2  DECODE        -  Decode data
  3  RANDOM        -  Generate a random number
  4  CHECKSUM      -  Generate a checksum and verification and
                      hash pattern
  5  PPKEYS        -  Generate master key values from a pass phrase
```

Figure 18. ICSF Utilities Panel

2. Choose option 3, RANDOM, to access the Random Number Generator panel, shown in Figure 19 on page 55.

```
CSFRNG00 ---------------- ICSF - Random Number Generator -------------
COMMAND ===>


Enter data below:

  Parity Option  ===> RANDOM            ODD, EVEN, RANDOM
  Random Number1    : 0000000000000000  Random Number 1
  Random Number2    : 0000000000000000  Random Number 2
  Random Number3    : 0000000000000000  Random Number 3
```

Figure 19. ICSF Random Number Generator Panel

3. To select the parity of the random numbers, enter ODD, EVEN, or RANDOM next to Parity Option and press ENTER.

   The DES master key is forced to have odd parity, regardless of the parity option you select for each key part.

   A random 16-digit number appears in each of the Random Number fields. You can use each of these random numbers for a segment of a key part.

   Note: The third random number is only for PKA master keys. It is not used for DES master keys or operational keys.

```
CSFRNG00 ---------------- ICSF - Random Number Generator -------------
COMMAND ===>


Enter data below:

  Parity Option  ===> RANDOM            ODD, EVEN, RANDOM
  Random Number1    : 51ED9CFA90716CFB  Random Number 1
  Random Number2    : 58403BFA02BD13E8  Random Number 2
  Random Number3    : 9B28AEFA8C47760F  Random Number 3
```

Figure 20. ICSF Random Number Generator Panel with Generated Numbers

4. *Record the random numbers* so you can store them in a safe place. If you ever need to reenter a master key that has been cleared for any reason, you will need the key part values.
5. Press END to return to the Utilities panel.
6. Continue with Generating a Checksum, Verification Pattern, or Hash Pattern for a Key Part.

Generating a Checksum, Verification Pattern, or Hash Pattern for a Key Part

You can use the ICSF utilities panel to generate a checksum and either an optional verification pattern or an optional hash pattern for a key part. You can use this panel to generate a checksum for a key part even if ICSF has not been initialized. The random number generator and the hash and verification pattern, however, do not work until ICSF has been initialized with a valid master key.

Note: The use of these utility panels to generate the key part, the checksum, and the verification pattern exposes the key part in storage for the duration of the dialogs. For this reason, you can choose to calculate both the checksum, the verification pattern or the hash pattern values manually or by using a PC program. See “Checksum Algorithm” on page 209 for a description of the checksum algorithm. See “Algorithm for Calculating a Verification Pattern” on page 210

page 210 for a description of the algorithm for the verification pattern. See "The MDC–4 Algorithm for Generating Hash Patterns" on page 211 for a description of the MDC-4 algorithm that is used to calculate a hash pattern for a key part. The use of the verification pattern or hash pattern is optional.

Follow these steps to generate a checksum and the optional verification pattern or hash pattern for a key part.

1. Select option 4, CHECKSUM, on the ICSF Utilities panel as shown in Figure 21.

```
CSFUTL00 ---------------- ICSF - Utilities ------------------------
OPTION ===> 4


Enter the number of the desired option above.

  1  ENCODE        -  Encode data
  2  DECODE        -  Decode data
  3  RANDOM        -  Generate a random number
  4  CHECKSUM      -  Generate a checksum and verification and
                      hash patterns
  5  PPKEYS        -  Generate master key values from a pass phrase
```

Figure 21. Selecting the Checksum Option on the ICSF Utilities Panel

The Checksum and Verification and Hash Pattern panel appears. See Figure 22.

```
CSFMKV00 ------------ ICSF - Checksum and Verification and Hash Pattern -----
COMMAND ===>


Enter data below:

 Key Type      ===>                    (Selection panel displayed if blank)

 Key Value     ===> 51ED9CFA90716CFB   Input key value 0 - 7
               ===> 58403BFA02BD13E8   Input key value 8 - 15
               ===> 9B28AEFA8C47760F   Input key value 16 - 23 (PKA keys only)

 Checksum         : 00                 Check digit for key value
 Key Part VP      : 0000000000000000   Verification Pattern
 Key Part HP      : 0000000000000000   Hash Pattern
                  : 0000000000000000
```

Figure 22. ICSF Checksum and Verification and Hash Pattern Panel

If you accessed the Random Number Generator panel before this panel, the random numbers that are generated appear automatically in the Key Part fields. You can skip the next step.

2. If you did not use the ICSF panels to generate random numbers, enter the numbers for which you want to create checksum, verification pattern, or hash patterns into these fields.
3. In the Key Type field, specify either:
   - MASTER to generate a checksum and hash and verification pattern for a DES master key part.

- PKAMSTR to generate a checksum and hash pattern for a PKA master key part.

If you leave the Key Type field blank and press ENTER, the Key Type Selection panel appears. See Figure 23.

```
CSFMKV10 ------------- ICSF - Key Type Selection Panel ---- ROW 1 to 9 OF 9
COMMAND ===>                                                SCROLL ===> PAGE

Select one key type only
    KEY TYPE      DESCRIPTION
   EXPORTER   Export key encrypting key
   IMP-PKA    Limited authority importer key
   IMPORTER   Import key encrypting key
   IPINENC    Input PIN encrypting key
s  MASTER     DES master key
   OPINENC    Output PIN encrypting key
   PINGEN     PIN generation key
   PINVER     PIN verification key
   PKAMSTR    PKA master key
***************************** BOTTOM OF DATA ****************************
```

Figure 23. Key Type Selection Panel Displayed During Hardware Key Entry

4. Type 'S' to the left of the MASTER key type, and press ENTER to return to the Checksum and Verification Pattern panel as shown in Figure 24.

   In this example, we have selected the DES master key.

```
CSFMKV00 ------ ICSF - Checksum and Verification and Hash Pattern ---
COMMAND ===>


Enter data below:

 Key Type      ===> MASTER            (Selection panel displayed if blank)

 Key Value     ===> 51ED9CFA90716CFB  Input key value 0 - 7
               ===> 58403BFA02BD13E8  Input key value 8 - 15
               ===> 9B28AEFA8C47760F  Input key value 16 - 23 (PKA keys only)

 Checksum        : 00                Check digit for key part
 Key Part VP     : 0000000000000000  Verification Pattern
 Key Part HP     : 0000000000000000  Hash Pattern
                 : 0000000000000000
```

Figure 24. ICSF Checksum and Verification Pattern Panel

5. On the Checksum and Verification Pattern panel, press ENTER.

   ICSF calculates the checksum, verification pattern, and hash pattern for the key part segments and displays them on the panel as shown in Figure 25 on page 58. Since a DES master key was selected for this example, the key part last segment was not used in the calculations. The key part last field is zeroed out on the panel. For a PKA master key, ICSF uses all three key part segments to calculate the checksum, verification pattern, and hash pattern.

```
CSFMKV00 ------ ICSF - Checksum and Verification and Hash Pattern ---
COMMAND ===>


Enter data below:

  Key Type      ===> MASTER            (Selection panel displayed if blank)

  Key Value     ===> 51ED9CFA90716CFB  Input key value 0 - 7
                ===> 58403BFA02BD13E8  Input key value 8 - 15
                ===> 0000000000000000  Input key value 16 - 23 (PKA keys only)

  Checksum         : 40                Check digit for key part
  Key Part VP      : 0CCE190A635A6C89  Verification Pattern
  Key Part HP      : EA58E51179754FB7  Hash Pattern
                   : C102957465CE479E
```

Figure 25. Checksum, Verification Pattern, and Hash Pattern Calculated for a DES Master Key Part

6. *Record the checksum, verification pattern, and hash pattern.*

   Save these values in a secure place along with the key part values in case of a tamper. If the Cryptographic Coprocessor Feature detects tampering, it clears the master key, and you have to reenter the same master key again.
7. Press END to return to the Utilities panel.
8. Press END again to return to the ICSF Primary menu.

Continue with the appropriate section for steps to enter the master key part you have just generated.

- If you have generated the first master key part, continue with “Entering the First Master Key Part”.
- If you have generated an intermediate master key part, continue with “Entering Intermediate Key Parts” on page 61.
- If you have generated a final master key part, continue with “Entering the Final Key Part” on page 63.

Entering the First Master Key Part

Use the Clear Master Key Entry panels to enter each key part.

If you use the random number generator utility to generate key parts, enter each key part directly after you generate the key part data and before generating another key part.

To enter master key parts:

1. Select option 1, COPROCESSOR MGMT, on the ICSF Primary menu, as shown in Figure 26 on page 59, and press ENTER.

```
CSF@PRIM --------- Integrated Cryptographic Service Facility ---------
OPTION ===> 1

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions


      Licensed Materials - Property of IBM

     5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 26. ICSF Selecting the Master Key Option on the Primary Menu Panel

The ICSF Coprocessor Management panel appears (Figure 27).

2. Select the coprocessor(s) to be processed by entering an 'E' and then pressing ENTER. Select as many coprocessors as required. This loads the same master key for all coprocessors selected.

   Note: During first time initialization, the coprocessor status will be `ONLINE`. After the master keys are set, status will be `ACTIVE`.

```
CSFCMP00 ---------------- ICSF Coprocessor Management -------------
COMMAND ===>

Select the coprocessors to be processed and press ENTER.
Action characters are: A, D, E, R, and S. See the help panel for details.

COPROCESSOR  MODULE ID/SERIAL NUMBER                 STATUS
-----------  -------------------------------         -------

_ A06                                                 ACTIVE
_ A07                                                 ACTIVE
E C0         E589C396944007A6 5D40369997A386F4        ONLINE
E C1         0AA379BFD2387960 0367DC04533125FF        ONLINE
E P00        41-00YE1                                 ONLINE
E P01        41-00K11                                 ONLINE
E P02        41-0A355                                 ONLINE
E P03        41-0BA3F                                 ONLINE
_ P04        41-0RT2T                                 DEACTIVATED
_ P05        41-00342                                 DISABLED
```

Figure 27. Selecting the coprocessor on the Coprocessor Management Panel

3. The ICSF Clear Master Key Entry panel appears. See Figure 28 on page 60.

```
CSFDKE10------------- ICSF - Clear Master Key Entry -----------------
COMMAND ===>

             CCF DES/PCICC SYM-MK new master key register     : EMPTY
             CCF Signature/PCICC ASYM-MK master key register  : EMPTY
             CCF Key management master key register           : EMPTY


  Specify information below
    Key Type  ===> ___                (DES, SMK, KMMK, ALL-PKA)

    Part      ===> ______             (RESET, FIRST, MIDDLE, FINAL)

    Checksum  ===> 00

    Key Value ===> 0000000000000000
              ===> 0000000000000000
              ===> 0000000000000000   (SMK, KMMK and ALL-PKA only)



  Press ENTER to process.
  Press END   to exit to the previous menu.
```

Figure 28. Clear Master Key Entry Panel

4. Fill in the panel
   a. Enter the master key type in the Key Type field.

      In this example we are entering the DES master key.
   b. Enter FIRST in the Part field.
   c. Enter the two-digit checksum and the two 16-digit key values.
   d. When all the fields are complete, press ENTER.

      If the checksum entered in the checksum field matches the checksum that the Cryptographic Coprocessor Feature calculated, the key part is accepted. The message at the top of the panel states `KEY PART LOADED`, as shown in Figure 29 on page 61. The new master key register status changes to `PART FULL`. The verification pattern and hash pattern that are calculated for the key part appear near the bottom of the panel. Compare them with the patterns generated by the random number generator or provided by the person who gave you the key part value to enter.
   e. Record the verification pattern and hash pattern.

```
CSFDKE10 -------------- ICSF - Clear Master Key Entry --- KEY PART LOADED
COMMAND ===>

           CCF DES/PCICC SYM-MK new master key register     : PART FULL
           CCF Signature/PCICC ASYM-MK master key register  : EMPTY
           CCF Key management master key register           : EMPTY


 Specify information below
   Key Type  ===> DES         (DES, SMK, KMMK, ALL-PKA)

   Part      ===> FIRST       (RESET, FIRST, MIDDLE, FINAL)

   Checksum  ===> 40

   Key Value ===> 51ED9CFA90716CFB
             ===> 58403BFA02BD13E8
             ===> 0000000000000000   (SMK, KMMK and ALL-PKA only)


Entered key part VP: 0CCE190A63546489  HP: 9C92A343479D33F2 66229FCD55B49C26

                    (Record and secure these patterns)

 Press ENTER to process.
 Press END   to exit to the previous menu.
```

Figure 29. The Clear Master Key Entry Panel Following Key Part Entry

5. If the checksums do not match, the message `Invalid Checksum` appears. If this occurs, follow this sequence to resolve the problem:
   a. Reenter the checksum.
   b. If you still get a checksum error, recalculate the checksum.
   c. If your calculations result in a different value for the checksum, enter the new value.
   d. If your calculations result in the same value for the checksum, or if a new checksum value does not resolve the error, reenter the key part halves and checksum.

When you have entered the first key part successfully, continue with:

- “Generating Key Parts Using ICSF Utilities” on page 53 if you are using the ICSF utilities to generate random numbers for key values.
- “Entering Intermediate Key Parts” if you are entering key parts manually.

Entering Intermediate Key Parts

If you want to enter more than two key parts, you must enter one or more intermediate key parts. Enter intermediate key parts after you enter the first key part and before you enter the final one.

To enter intermediate master key parts:

1. Select option 1, COPROCESSOR MGMT, on the ICSF Primary menu and press ENTER.

   The Coprocessor Management panel appears.
2. Select the coprocessor(s) to be processed by entering an 'E' on the Coprocessor Management panel. Select the same coprocessors that were selected when entering the first key value.

3. After pressing ENTER, the Clear Master Key Entry panel appears (Figure 30).

```
CSFDKE10 -------------- ICSF - Clear Master Key Entry ---------
COMMAND ===>

             CCF DES/PCICC SYM-MK new master key register     : PART FULL
             CCF Signature/PCICC ASYM-MK master key register  : EMPTY
             CCF Key management master key register           : EMPTY


 Specify information below
   Key Type  ===> ___          (DES, SMK, KMMK, ALL-PKA)

   Part      ===> ______       (RESET, FIRST, MIDDLE, FINAL)

   Checksum  ===> 00

   Key Value ===> 0000000000000000
             ===> 0000000000000000
             ===> 0000000000000000   (SMK, KMMK and ALL-PKA only)
```

Figure 30. The Clear Master Key Entry Panel for Intermediate Key Values

4. Fill in the panel
   a. Enter the master key type in the Key Type field.

      In this example we are continuing to enter the DES master key.
   b. Enter MIDDLE in the Part field.
   c. Enter the two-digit checksum and the two 16-digit key values.
   d. When all the fields are complete, press ENTER.

      If the checksum entered in the checksum field matches the checksum that the Cryptographic Coprocessor Feature calculated, the key part is accepted. The message at the top of the panel states `KEY PART LOADED`, as shown in Figure 31 on page 63. The new master key register status changes to `PART FULL`. The verification pattern and hash pattern that are calculated for the key part appear near the bottom of the panel.

      Compare them with the patterns generated by the random number generator or provided by the person who gave you the key part value to enter.
   e. Record the verification pattern and hash pattern.

```
CSFDKE10 -------------- ICSF - Clear Master Key Entry -----KEY PART LOADED
COMMAND ===>

             CCF DES/PCICC SYM-MK new master key register     : PART FULL
             CCF Signature/PCICC ASYM-MK master key register  : EMPTY
             CCF Key management master key register           : EMPTY


 Specify information below
   Key Type  ===> DES      (DES, SMK, KMMK, ALL-PKA)

   Part      ===> MIDDLE   (RESET, FIRST, MIDDLE, FINAL)

   Checksum  ===> 4F

   Key Value ===> 834B4864BA8E8B68
             ===> FA3C8664FBC93A0D
             ===> 0000000000000000   (SMK, KMMK and ALL-PKA only)

 Entered key part VP:  8D8A000BE067EBF7  HP: 9D92F343479D77F2  229FD4CDB49C2679

                    (Record and secure these patterns)
```

Figure 31. The Clear Master Key Entry Panel with Intermediate Key Values

5. If the checksums do not match, the message `Invalid Checksum` appears. If this occurs, follow this sequence to resolve the problem:
   a. Reenter the checksum.
   b. If you still get a checksum error, recalculate the checksum.
   c. If your calculations result in a different value for the checksum, enter the new value.
   d. If your calculations result in the same value for the checksum, or if a new checksum value does not resolve the error, reenter the key part halves and checksum.

When you have entered the middle key part successfully, continue with:

- “Generating Key Parts Using ICSF Utilities” on page 53 if you are using the ICSF utilities to generate random numbers for key values.
- “Entering the Final Key Part” if you are entering key parts manually.

Entering the Final Key Part

After you enter the first key part, and any intermediate key parts, you then enter the final master key part.

1. Select option 1, COPROCESSOR MGMT, on the ICSF Primary menu and press ENTER.

   The Coprocessor Management panel appears.
2. Select the coprocessor(s) to be processed by entering an 'E' on the Coprocessor Management panel.
3. After pressing ENTER, the Clear Master Key Entry panel appears.

```
CSFDKE10 -------------- ICSF - Clear Master Key Entry ---------
COMMAND ===>

             CCF DES/PCICC SYM-MK new master key register     : PART FULL
             CCF Signature/PCICC ASYM-MK master key register  : EMPTY
             CCF Key management master key register           : EMPTY


 Specify information below
   Key Type  ===>  ___          (DES, SMK, KMMK, ALL-PKA)

   Part      ===>  ________     (RESET, FIRST, MIDDLE, FINAL)


   Checksum  ===> 00

   Key Value ===> 0000000000000000
             ===> 0000000000000000
             ===> 0000000000000000   (SMK, KMMK and ALL-PKA only)



 Press ENTER to process.
 Press END   to exit to the previous menu.
```

Figure 32. The Clear Master Key Entry Panel before entering Final Key Values

4. Fill in the panel
   a. Enter the master key type in the Key Type field.

      In this example we are continuing to enter the DES master key.
   b. Enter FINAL in the Part field.
   c. Enter the two-digit checksum and the two 16-digit key values.
   d. When all the fields are complete, press ENTER.

      If the checksum entered in the checksum field matches the checksum that the Cryptographic Coprocessor Feature calculated, the key part is accepted. The message at the top of the panel states `KEY PART LOADED`, as shown in Figure 33 on page 65. The new master key register status changes to `FULL`. The verification pattern and hash pattern that are calculated for the key part appear near the bottom of the panel. Compare them with the patterns generated by the random number generator or provided by the person who gave you the key part value to enter.
   e. Record the verification pattern and hash pattern.

```
CSFDKE10 -------------- ICSF - Clear Master Key Entry ------  KEY PART LOADED
COMMAND ===>

             CCF DES/PCICC SYM-MK new master key register     : FULL
             CCF Signature/PCICC ASYM-MK master key register  : EMPTY
             CCF Key management master key register           : EMPTY


 Specify information below
   Key Type  ===>  DES          (DES, SMK, KMMK, ALL-PKA)

   Part      ===>  FINAL        (RESET, FIRST, MIDDLE, FINAL)

   Checksum  ===>  4F

   Key Value ===> 834B4864BA8E8B68
             ===> FA3C8664FBC93A0D
             ===> 0000000000000000   (SMK, KMMK and ALL-PKA only)

 Entered key part VP: 8D8A000BE067EBF7 HP: 9D92F343479D77F2 229FD4CDB49C2679
 Master Key       VP: 8F887096A8D4922C HP: 4C887096A8D4922B 33387096A8D4922B
                    (Record and secure these patterns)
```

Figure 33. The Clear Master Key Entry Panel with Final Key Values

5. If the checksums do not match, the message `Invalid Checksum` appears. If this occurs, follow this sequence to resolve the problem:
   a. Reenter the checksum.
   b. If you still get a checksum error, recalculate the checksum.
   c. If your calculations result in a different value for the checksum, enter the new value.
   d. If your calculations result in the same value for the checksum, or if a new checksum value does not resolve the error, reenter the key part halves and checksum.
6. When you have entered the final key part successfully, it is combined with the first key part and any intermediate key parts in the new master key register.

   The new master key register status is now FULL, and the panel displays two verification patterns and two hash patterns. It gives you verification patterns and hash patterns for both the final key part and the new master key, since it is now complete.
7. Check that the key part verification pattern or hash pattern you may have previously calculated matches the verification pattern or hash pattern that is shown on the panel. If they do not, you may want to restart the key entry process. For information on how to restart the key entry process, see “Restarting the Key Entry Process” on page 66.
8. *Record the verification pattern and hash pattern* for the new master key, because you may want to verify it at another time.

   Note: When you initialize or reencipher a CKDS, ICSF places the verification pattern for the DES master key into the CKDS header record.

When you have entered the master keys correctly, they are in the new master key registers and are not active on the system.

Note: Ensure that the new master key is installed on all cryptographic coprocessors.

After you enter the master keys, you should do *one* of the following:

- If you are defining the DES master key and SYM-MK for the first time, initialize the CKDS with the DES master key. For a description of the process of initializing a DES master key on your system, see “Initializing the CKDS at First-Time Startup” on page 68.
- If you are defining a DES master key after it was cleared, set the DES master key to make it active. For a description of the process of recovering from tampering, see “Reentering Master Keys After They have been Cleared” on page 72.
- If you are changing a DES master key, reencipher the CKDS under the new DES master key and make it active. For a description of the process of changing a DES master key, see “Changing Master Keys” on page 74.

Restarting the Key Entry Process

If you realize that you made an error when entering a key part, you can restart the process of entering the new master key. For example, if the verification pattern or the hash pattern that was calculated does not match the one that you calculated, you may want to restart the process. Restarting the key entry process clears the new master key register, which erases all the new master key parts you entered previously.

Note: If you are working on a CCF, when you enter the first key part, your old master key is lost, even if you restart the process.

To restart the key entry process, follow the steps below:

1. On the Clear Master Key Entry panel, enter the master key type in the Key Type field.

   In this example, we are entering a new DES master key.
2. Enter RESET in the Part field.

```
CSFDKE10 -------------- ICSF - Clear Master Key Entry ---------
COMMAND ===>

             CCF DES/PCICC SYM-MK new master key register     : PART FULL
             CCF Signature/PCICC ASYM-MK master key register  : EMPTY
             CCF Key management master key register           : EMPTY


 Specify information below
   Key Type  ===>  DES              (DES, SMK, KMMK, ALL-PKA)

   Part      ===>  RESET_           (RESET, FIRST, MIDDLE, FINAL)

   Checksum  ===>  40

   Key Value ===> 51ED9CFA90716CFB
             ===> 58403BFA02BD13E8
             ===> 0000000000000000   (SMK, KMMK and ALL-PKA only)
```

Figure 34. Selecting Reset on the Clear Master Key Entry Panel

3. Press ENTER.

   The Restart Key Entry Process panel appears. See Figure 35 on page 67. This panel confirms your request to restart the key entry process.

```
CSFDKE40 -------------- ICSF - Restart Key Entry Process -------------
COMMAND ===>

ARE YOU SURE YOU WISH TO RESTART THE KEY ENTRY PROCESS?


  Restarting the process will clear the DES master key register.


  WARNING: Ressetting the KMMK or SMK will invalidate any private
           internal key tokens in the PKDS.


Press ENTER to confirm restart request
Press END   to cancel restart request
```

Figure 35. Confirm Restart Request Panel

Note: If you are restarting the key entry process for one or all of the PKA master keys, the panel message will differ. ICSF substitutes either 'KMMK register', 'SMK register' or 'ALL-PKA register' for ' *the DES master key register*' phrase in the panel message.

4. If you want to restart the key entry process, press ENTER.

   The restart request automatically empties the master key register.

5. If you do not want to restart, press END.

   After you make a choice, you return to the Clear Master Key Entry panel. If you selected to continue with the restart process, the new master key register status field is reset to EMPTY, as shown in Figure 36. This indicates that the register has been cleared.

```
CSFDKE10 -------------- ICSF - Clear Master Key Entry ---------
COMMAND ===>

             CCF DES/PCICC SYM-MK new master key register     : EMPTY
             CCF Signature/PCICC ASYM-MK master key register  : EMPTY
             CCF Key management master key register           : EMPTY


 Specify information below
   Key Type  ===> ___          (DES, SMK, KMMK, ALL-PKA)

   Part      ===> ________     (RESET, FIRST, MIDDLE, FINAL)


   Checksum  ===> 40

   Key Value ===> 51ED9CFA90716CFB
             ===> 58403BFA02BD13E8
             ===> 0000000000000000   (SMK, KMMK and ALL-PKA only)
```

Figure 36. The Clear Master Key Entry Panel Following Reset Request

6. Either begin the key entry process again or press END to return to the ICSF primary menu panel.

Initializing the CKDS at First-Time Startup

The first time you start ICSF, you must:

- Enter a DES new master key into the Cryptographic Coprocessor Feature
- Enter a new SYM-MK into each PCI Cryptographic Coprocessor - if you have PCICCs in your environment
- Create a cryptographic key data set (CKDS)
- Initialize the CKDS

When you initialize the CKDS, ICSF creates a header record for the CKDS, installs the required system keys in the CKDS, and sets the DES master key and SYM-MK. Keys stored in the CKDS are enciphered under the DES master key.

After you define the DES master key and initialize a CKDS, you can generate or enter any additional system keys you need to perform cryptographic functions.

There are four different types of system keys you can install in the CKDS:

- Required SYSTEM keys are automatically generated when you first initialize the CKDS. These include the MAC and MACVER keys that ICSF uses to generate and validate the MAC code in each CKDS record.
- NOCV-enablement keys are required for NOCV IMPORTERs and EXPORTERs. The NOCV-enablement system keys are used to twist on and twist off the CVs on external tokens during key import and key export. This allows ICSF to communicate with systems that do not use control vectors.
- ANSI system keys are required for almost all ANSI services to perform the notarization and offset that are required by ANSI X9.17.
- ESYS, or enhanced system keys, are used only in Symmetric Key Export service.

For information on system keys, see “Entering System Keys into the Cryptographic Key Data Set (CKDS)” on page 25.

You have to initialize a CKDS only the first time you start ICSF on a system. After you initialize a CKDS, you can copy the disk copy of the CKDS to create other CKDSs for use on the system. You can also use a CKDS on another ICSF system if the system has the same master key value. At any time, you can read a different disk copy into storage. For information about how to read a disk copy into storage, see “Refreshing the CKDS at Any Time” on page 71.

For a description of how to use the Clear Master Key Entry panels to enter the master key, see “Entering the First Master Key Part” on page 58. For a description of how to use the TKE workstation to enter the master key, refer to *z/OS ICSF TKE Workstation User’s Guide 2000*.

To initialize the CKDS:

1. Return to the Primary Menu panel by pressing END from the Clear Master Key Entry panel.
2. Select Option 2, MASTER KEY, on the Primary Menu panel as shown in Figure 37 on page 69.

```
CSF@PRIM --------- Integrated Cryptographic Service Facility ---------
OPTION ===> 2

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

      Licensed Materials - Property of IBM

     5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 37. ICSF Selecting the CKDS Initialization Option on the Primary Menu Panel

The Master Key Management panel appears. See Figure 38.

```
CSFMKM00 ---------------- ICSF - Master Key Management  ----------------
OPTION ===>  1


Enter the number of the desired option above.

  1 INIT/REFRESH CKDS - Initialize a Cryptographic Key Data Set or activate
                        an updated Cryptographic Key Data Set
  2 SET MK            - Set a DES master key
  3 REENCIPHER CKDS   - Reencipher the CKDS prior to changing the DES master
                        key
  4 CHANGE MK         - Change the DES master key and activate the
                        reenciphered CKDS
  5 REENCIPHER PKDS   - Reencipher the PKA Cryptographic Key Data Set
  6 ACTIVATE PKDS     - Activate the PKDS after it has been reenciphered
  7 REFRESH CACHE     - Refresh the PKDS cache if enabled
```

Figure 38. ICSF Master Key Management Panel

3. Select option 1, INIT/REFRESH CKDS and the Initialize a CKDS panel appears. See Figure 39 on page 70.

```
CSFCKD00 ---------------- ICSF - Initialize a CKDS  ----------------
COMMAND ===>


Enter the number of the desired option.

  1  Initialize an empty CKDS (creates the header and system keys)

  2  NOCVKEYS  -  Create NOCV-Enablement keys (for keys without CVs)
  3  ANSI      -  Create ANSI system keys (for ANSI X9.17 services)
  4  ESYS      -  Create enhanced system keys (for Symmetric services)

  5  REFRESH   -  Activate an updated CKDS

Enter the name of the CKDS below.

  CKDS ===> 'FIRST.EMPTY.CKDS'
```

Figure 39. ICSF Initialize a CKDS Panel

4. In the CKDS field, enter the name of the empty VSAM data set that was created to use as the disk copy of the CKDS.

   The name you enter should be the same name that is specified in the CKDSN installation option in the installation options data set. For information about creating a CKDS and specifying the CKDS name in the installation options data set, see *z/OS ICSF System Programmer's Guide*.
5. Choose option 1, Initialize an empty CKDS, and press ENTER.

   ICSF creates the header record in the disk copy of the CKDS. Next, ICSF sets the DES master key. ICSF then adds the required system keys to the CKDS and refreshes the CKDS. When ICSF completes all these steps, the message `INITIALIZATION COMPLETE` appears. If you did not enter a master key into the new master key register previously, the message `NMK REGISTER NOT FULL` appears and the initialization process ends. You must enter a master key into the new master key register before you can initialize the CKDS.

   Note: If any part of the option 1 fails, you must delete the CKDS and start over. If the failure occurs after the master key has been set and before the system keys have been created, you will need to reset the master keys.
6. If you want ICSF to create NOCV-enablement keys after the initialization process has been completed, select option 2, NOCVKEYS, and press ENTER.

   The creation of NOCV-enablement keys is optional. It allows you to use either the key generator utility program or the Key Token Build callable service to create NOCV keys. NOCV keys allow you to send and receive keys from systems that do not use control vectors. For a description of NOCV keys, see the description of the NOCV keyword for the key generator utility program on 105.

   Note: If you want to run the ICSF conversion program to convert a CUSP/PCF CKDS into ICSF format, the CKDS you start ICSF with must contain NOCV-enablement keys. For more information about the conversion program, see *z/OS ICSF System Programmer's Guide*.
7. To create ANSI system keys that are used for the ANSI X9.17 services, choose option 3, ANSI.

   The creation of ANSI system keys is optional. ANSI system keys are required if you intend to also create enhanced system keys.

The message ANSI KEYS ADDED appears on the top right of the panel, if the process succeeds.

8. To create enhanced system keys, choose option 4, ESYS.

   The creation of enhanced system keys is optional. To create enhanced system keys, you must have previously installed the ANSI system keys in the CKDS.

   The message ESYS KEYS ADDED appears on the top right of the panel, if the process succeeds.

After you complete the entire process, a master key and CKDS exist on your system. You can now generate keys using the key generate callable service and key generator utility program, or convert CUSP/PCF keys to ICSF keys using the conversion program. ICSF services use the keys to perform the cryptographic functions you request.

Note: You enable special secure mode to initialize ICSF for the first time. After you perform the initialization process, you may choose to disable special secure mode.

Refreshing the CKDS at Any Time

After you initialize a CKDS for the first time, you can copy the disk copy of the CKDS to create other CKDSs for the system. You can use KGUP to add and update any of the disk copies on your system. You can use the dynamic CKDS update callable services to add or update the disk copy of the current in-storage CKDS. For information about using KGUP, see Chapter 7, “Managing Cryptographic Keys by Using the Key Generator Utility Program”, on page 95. For information on using the dynamic CKDS callable services, refer to the *z/OS ICSF Application Programmer’s Guide*.

You can refresh the in-storage CKDS with an updated or different disk copy of the CKDS by following the steps below. You can refresh the CKDS at any time without disrupting cryptographic functions.

Note: Before you refresh a CKDS, consider temporarily disallowing dynamic CKDS update services. For more information, refer to “Disallowing Dynamic CKDS Updates During KGUP Updates” on page 96.

1. Enter option 2, MASTER KEY, on the ICSF Primary Menu panel to access the Master Key Management Panel.
2. Enter option 1, INIT/REFRESH CKDS to access the Initialize a CKDS panel, which is shown in Figure 40 on page 72.

```
CSFCKD00 ---------------- ICSF - Initialize a CKDS  ----------------
COMMAND ===>  5


Enter the number of the desired option.

  1  Initialize an empty CKDS (creates the header and system keys)

  2  NOCVKEYS  -  Create NOCV-Enablement keys (for keys without CVs)
  3  ANSI      -  Create ANSI system keys (for ANSI X9.17 services)
  4  ESYS      -  Create enhanced system keys (for Symmetric services)

  5  REFRESH   -  Activate an updated CKDS

Enter the name of the CKDS below.

  CKDS ===> 'PIN1.CKDS'
```

Figure 40. Selecting the Refresh Option on the ICSF Initialize a CKDS Panel

3. In the CKDS field, specify the name of the disk copy of the CKDS that you want ICSF to read into storage.
4. Choose option 5, REFRESH, and press ENTER.

   ICSF places the disk copy of the specified CKDS into storage. During a REFRESH, ICSF does not load into storage any partial keys that may exist when you enter keys manually. A REFRESH does not disrupt any applications that are running on ICSF. A message that states that the CKDS was refreshed appears on the right of the top line on the panel.

   After ICSF reads the CKDS into storage, it performs a MAC verification on each record in the CKDS. If a record fails the MAC verification, ICSF sends a message that gives the key label and type to the z/OS system security console. You can then use either KGUP or the dynamic CKDS update services to delete the record from the CKDS. Any other attempts to access a record that has failed MAC verification results in a return code and reason code that indicate that the MAC is not valid.
5. Press END to return to the Primary Menu panel.

Note: You can use either a KGUP panel or a utility program, instead of the CKDS panel, to refresh the CKDS. For information about these other methods, see “Refreshing the In-Storage CKDS” on page 126.

Reentering Master Keys After They have been Cleared

In the following situations, the Cryptographic Coprocessor Feature clears the master key registers so that the master key values are not disclosed.

- If the Cryptographic Coprocessor Feature detects tampering
- If you issue a command from the TKE workstation to zeroize a domain
- If you issue a command from the Support Element to zeroize all domains

In the following situations, the PCI Cryptographic Coprocessor Feature (PCICC) clears the master key registers so that the master key values are not disclosed.

- If the PCI Cryptographic Coprocessor Feature detects tampering (the intrusion latch is tripped), ALL installation data is cleared: master keys, retained keys for all domains, as well as roles and profiles.

- If the PCI Cryptographic Coprocessor Feature detects tampering (the secure boundary of the card is compromised), it self-destructs and can no longer be used.
- If you issue a command from the TKE workstation to zeroize a domain

  This command zeroizes the data specific to a domain: master keys and retained keys.
- If you issue a command from the Support Element panels to zeroize all domains.

  This command zeroizes ALL installation data: master keys, retained keys and access control roles and profiles.

Although the values of the master keys are cleared, the keys in the CKDS are still enciphered under the cleared DES master key. The RSA and DSS private keys are also each enciphered under one of the cleared PKA master keys. Therefore, to recover the keys in the CKDS, and the PKA private keys, you must reenter the same master keys and activate the DES master key. For security reasons, you may then want to change all the master keys.

PR/SM Considerations: If you are running in PR/SM logical partition (LPAR) mode, there are several situations (listed previously) that can cause loss of master keys and other data. In these cases, you must first ensure that key entry is enabled for each LP on the Change LPAR Crypto page on the support element Hardware Master Console. You must then reenter the master keys in each LP. If you zeroize a domain using the TKE workstation, however, the master keys are cleared only in that domain. Master keys in other domains are not affected and do not need to be reentered. For more information about reentering master keys in LPAR mode, see Appendix D, “PR/SM Considerations during Key Entry”, on page 213.

After the Cryptographic Coprocessor Feature clears the master keys, reenter the same master keys by following these steps:

1. Check the status of the PKA callable services. If they are enabled, use the Administrative Control Functions to disable them. See “Enabling and Disabling PKA Services” on page 79 for details.
2. Retrieve the key parts, checksums, verification patterns, and hash patterns you used when you entered the master keys originally.

   These values should be stored in a secure place as specified in your enterprises security process.
3. Access the Clear Master Key Entry panels and enter the master keys as described in “Entering the First Master Key Part” on page 58.
4. After you enter the new DES master key, select option 2, MASTER KEY, from the primary menu. The Master Key Management panel appears. See Figure 41 on page 74.

   To activate the DES master key you just entered, you need to set it.
5. To set the DES master key, choose option 2 on the panel and press ENTER.

```
CSFMKM00 ---------------- ICSF - Master Key Management -------------
OPTION ===>  2


Enter the number of the desired option above.

  1 INIT/REFRESH CKDS - Initialize a Cryptographic Key Data Set or activate
                        an updated Cryptographic Key Data Set
  2 SET MK            - Set a DES master key
  3 REENCIPHER CKDS   - Reencipher the CKDS prior to changing the DES master
                        key
  4 CHANGE MK         - Change the DES master key and activate the
                        reenciphered CKDS
  5 REENCIPHER PKDS   - Reencipher the PKA Cryptographic Key Data Set
  6 ACTIVATE PKDS     - Activate the PKDS after it has been reenciphered
  7 REFRESH CACHE     - Refresh the PKDS cache if enabled
```

Figure 41. Selecting the Set Host Master Key Option on the ICSF Master Key Management Panel

After you select option 2, ICSF checks that the states of the registers are correct. ICSF then transfers the DES master key from the new master key register to the master key register. This process sets the DES master key.

When ICSF attempts to set the DES master key, it displays a message on the top right of the Master Key Management panel. The message indicates either that the master key was successfully set, or that an error prevented the completion of the set process.

Notes:

a. If your system is using both crypto modules provided by a Cryptographic Coprocessor Feature, ICSF sets the DES master key for each crypto module whose new DES master key enciphers the in-storage CKDS. You should reenter the DES master key into the new master key register for each of the crypto modules.

b. The operator console receives messages that state that the crypto module is offline and then online for each crypto module. These actions should not affect cryptographic operations. However, if a crypto module does not have either a current DES master key or a new DES master key that enciphers the current in-storage CKDS, the crypto module is left offline.

When you set the reentered DES master key, the DES master key that enciphers the existing CKDS now exists.

6. You can now change the DES master key, if you choose to, for security reasons. Continue with “Changing Master Keys”.

Changing Master Keys

For security reasons your installation should change the master keys periodically. In addition, if the master keys have been cleared, you may also want to change the master keys after you reenter the cleared master keys.

There are three main steps involved in changing the DES master key:

1. Enter the DES and SYM-MK master key parts.
2. Reencipher the CKDS under the new DES master key.
3. Change the new DES master key and activate the reenciphered CKDS.
4. Update the options data set with the new CKDS.

There are six main steps involved in changing the PKA master keys:

1. Disable PKA Services
2. Enter the PKA master keys (SMK and KMMK, if equal to the SMK) and ASYM-MK.
3. Enable PKA Services
4. Reencipher the PKDS under the new PKA master keys.
5. Activate the PKDS.
6. Enable PKDS read and write access.
7. Update the options data set with the new PKDS.

Note: PKA master keys should only be changed if there is a PCICC available on the system.

DES Master Keys and the CKDS

The step-by-step procedure for changing the DES master key, reenciphering the CKDS, and activating the new DES master key are presented in “Changing the DES Master Key and Reenciphering the CKDS” on page 76. This section provides some background on the contents of the master key registers during the key change process, and some compatibility mode considerations.

A DES master key and a CKDS that contains keys that are enciphered under that DES master key already exist. Before you replace this existing DES master key with the new DES master key, you must reencipher the CKDS under the new DES master key.

Note: Before you reencipher a CKDS, consider temporarily disallowing dynamic CKDS update services. For more information, refer to “Disallowing Dynamic CKDS Updates During KGUP Updates” on page 96.

For the CCF, if you changed the DES master key before, the previous DES master key was stored in the auxiliary (or new/old) master key register. The currently active DES master key exists in the master key register. When you enter the key parts of a new DES master key, they displace the previous DES master key in the auxiliary master key register. Therefore, the previous DES master key is lost. This is not true for the PCICC, which has separate registers for the old, new and current master key.

If you are using the Cryptographic Coprocessor Feature (CCF), to make the new DES master key the current active DES master key, you have ICSF swap the contents of the master key register and the auxiliary master key register. If you also have the PCICC, ICSF will change the PCI SYM-MKs. In this way, the new DES master key you have just entered becomes the current DES master key, and the previous DES master key is stored in the auxiliary master key register.

Before the new DES master key is placed into the master key register, you must reencipher all disk copies of the CKDS under the new DES master key. Then you are ready to activate the master key. When you change the master key, you have ICSF replace the in-storage copy of the CKDS with the reenciphered disk copy. This also makes the new master key active on the system.

The procedures you use to activate the new master key depend on your system’s compatibility mode. ICSF runs in noncompatibility, compatibility, or co-existence mode with the IBM cryptographic products, Cryptographic Unit Support Program (CUSP) and Programmed Cryptographic Facility (PCF). You specify which mode

ICSF runs in by using an installation option. For a description of the modes and how to specify an installation option, see *z/OS ICSF System Programmer's Guide*.

In noncompatibility mode, ICSF allows you to change the master key with continuous operations. Therefore applications can continue to run without disruption. However, when ICSF is in compatibility mode or co-existence mode, you should use a different procedure to activate the changed master key. This is to ensure that no application is holding an internal token with the wrong master key.

In all three modes, you enter the new master key and reencipher the disk copy of the CKDS under the new master key using the master key panels. In noncompatibility mode, you then activate the new master key and refresh the in-storage copy of the CKDS with the disk copy using the master key panels or a utility program.

In compatibility mode and coexistence mode, however, activating the new master key and refreshing the in-storage copy of the CKDS does not reencipher internal key tokens under the new master key. ICSF applications that are holding internal key tokens which have been enciphered under the wrong master key will fail with a warning message. Applications that use the CUSP and PCF macros, run with no warning message and produce erroneous results.

If you are using the CCF, the safest method to use after changing the master key in either compatibility or coexistence mode is as follows:
1. Ensure that the name of the new CKDS is in the installation data set.
2. Re-IPL MVS.
3. Start CSF.

If you also have PCICC installed, after you start CSF, you must go to the Master Key Management panel (Figure 41 on page 74) and do a set (option 2). This will change the master keys of all the PCICC that match the CCF.

A re-IPL ensures that a program does not access a cryptographic service that uses a key that is encrypted under a different master key. If a program is using an operational key, the program should either re-create or reimport the key, or generate a new key.

If a re-IPL is not practical in your installation, you can use this alternative method. Stop all cryptographic applications, especially those using CUSP or PCF macros, before activating the new master key and refreshing the in-storage copy of the CKDS. This eliminates all operational keys that are encrypted under the current master key. After you start CSF again, applications using an operational key can either re-create or reimport the key.

Changing the DES Master Key and Reenciphering the CKDS

1. Enter the key parts of the new master key that you want to replace the current master key. For information about how to do this procedure, see "Entering Clear Master Key Parts" on page 51.

   The new master key register must be full before you change the master key.
2. Select option 3, REENCIPHER CKDS, on the Master Key Management panel, as shown in Figure 42 on page 77, and press ENTER.

   Before you change the master key, you must first reencipher the disk copy of the CKDS under the new master key.

Note: If your system is using multiple coprocessors, they must have the same master key. When you change the master key in one coprocessor, you should change the master key in the other coprocessors. Therefore, before you can reencipher a CKDS under a new master key, the new master key registers in all coprocessors must contain the same value.

```
CSFMKM00 ---------------- ICSF - Master Key Management -------------
OPTION ===>  3


Enter the number of the desired option above.

  1 INIT/REFRESH CKDS - Initialize a Cryptographic Key Data Set or activate
                        an updated Cryptographic Key Data Set
  2 SET MK            - Set a DES master key
  3 REENCIPHER CKDS   - Reencipher the CKDS prior to changing the DES master
                        key
  4 CHANGE MK         - Change the DES master key and activate the
                        reenciphered CKDS
  5 REENCIPHER PKDS   - Reencipher the PKA Cryptographic Key Data Set
  6 ACTIVATE PKDS     - Activate the PKDS after it has been reenciphered
  7 REFRESH CACHE     - Refresh the PKDS cache if enabled
```

Figure 42. Selecting the Change Master Key Option on the ICSF Master Key Management Panel

3. The Reencipher CKDS panel appears. See Figure 43.

```
CSFCMK10 ----------------- ICSF - Reencipher CKDS ------------------
COMMAND ===>


To reencipher all CKDS entries from encryption under the current master key
to encryption under the new master key enter the CKDS names below.



   Input CKDS ===> CKDS.CURRENT.MASTER

   Output CKDS ===> CKDS.NEW.MASTER
```

Figure 43. Reencipher CKDS

4. In the Input CKDS field, enter the name of the CKDS that you want to reencipher. In the Output CKDS field, enter the name of the data set in which you want to place the reenciphered keys.

   Note: The output data set should already exist although it must be empty. For more information about defining a CKDS, see *z/OS ICSF System Programmer's Guide*.

   Reenciphering the disk copy of the CKDS does not affect the in-storage copy of the CKDS. On this panel, you are working with only a disk copy of the CKDS.
5. Press ENTER to reencipher the input CKDS entries and place them into the output CKDS.

The message REENCIPHER SUCCESSFUL appears on the top right of the panel if the reencipher succeeds.

6. If you have more than one CKDS on disk, specify the information and press ENTER as many times as you need to reencipher all of them. Reencipher all your disk copies at this time. When you have reenciphered all the disk copies of the CKDS, you are ready to change the master key.
7. Press END to return to the Master Key Management panel.

   Changing the master key involves refreshing the in-storage copy of the CKDS with a disk copy and activating the new master key.
8. If you are running in compatibility or co-existence mode, *do not* select option 4, the Change option. To activate the changed master key when running in compatibility or co-existence mode, you need to re-IPL MVS and start ICSF. When you re-IPL MVS and start ICSF, you activate the changed master key and refresh the in-storage CKDS. To do this, you must exit the panels at this time.
9. If you are running in noncompatibility mode, to change the master key select option 4, CHANGE MK, on the Master Key Management panel.

   When you press the ENTER key, the Change Master Key panel appears. See Figure 44.

```
CSFCMK20 ---------------------- ICSF Change Master Key --------------
COMMAND ===>


Enter the name of the new CKDS below:

  New CKDS ===> CKDS.NEW.MASTER

When the master key is changed, the new CKDS will become active.
```

Figure 44. Change Master Key Panel

10. In the New CKDS field, enter the name of the disk copy of the CKDS that you want ICSF to place in storage.

    You should have already reenciphered the disk copy of the CKDS under the new master key. The last CKDS name that you specified in the Output CKDS field on the Reencipher CKDS panel, which is shown in Figure 43 on page 77, automatically appears in this field.
11. Press ENTER.

    ICSF loads the data set into storage where it becomes operational on the system. ICSF also places the new master key into the master key register so it becomes active.

    After you press ENTER, ICSF attempts to change the master key. It displays a message on the top right of the panel. The message indicates either that the master key was changed successfully or that an error occurred that prevented the successful completion of the change process. For example, if you indicate a data set that is not reenciphered under the new master key, an error message displays, and the master key is not changed.

    Note: Each Cryptographic Coprocessor Feature includes two crypto modules, which ICSF recognizes as C0 and C1. You must enter the new master key into each of the coprocessors, before you perform the change. ICSF activates the new master key of both coprocessors that contain a

new master key value that will encipher the CKDS. If you also have PCICCs on your system, load the new master key into all of the coprocessors.

If only one coprocessor new master key value matches the new CKDS, then that coprocessor will be used. The other coprocessor will remain offline until the new master key is changed to match the other coprocessor.

When the change occurs, the operator console receives messages that state that the Cryptographic Coprocessor Feature is off line and then online for each coprocessor. These actions should not affect cryptographic operations.

You can use a utility program to reencipher the CKDSs and change the master key instead of using the panels. “Reenciphering a Disk Copy of a CKDS and Changing the Master Key” on page 195 describes how to use the utility program for these procedures.

PKA Master Keys and the PKDS

The step-by-step procedure for changing the PKA master keys is documented in this section. The procedure assumes that SMK=KMMK. It is recommended that the KMMK=SMK to maximize the routing capability to the PCICC and to enable PKDS reencipher. Once that is completed, it is necessary to reencipher and activate the PKDS.

If the SMK does not equal KMMK, see “Setting the Same Value for the SMK and KMMK” on page 86.

Attention: If you do not have a PCICC, you should not change the PKA Master Keys. Changing the PKA master keys will make all internal tokens in the current PKDS unusable. You will need to reencipher and activate the PKDS in order to use them with the changed master key. This requires a PCICC on your system. See “Reenciphering and Activating the PKDS” on page 84 for more information.

When the PKDS is shared by multiple images in a sysplex environment, the PKA master key must also be changed on all the sharing systems. See Chapter 6, “Running in a Sysplex Environment”, on page 91.

Enabling and Disabling PKA Services

When you enter or change the PKA master keys or the ASYM-MK, you must first disable the PKA services. To enable or disable PKA services:

1. Access the administrative control functions by choosing option 4, ADMINCNTL, on the Primary Menu panel, as shown in Figure 45 on page 80.

```
CSF@PRIM --------- Integrated Cryptographic Service Facility -------
OPTION ===> 4

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions


      Licensed Materials - Property of IBM

     5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 45. Selecting Administrative Control on the ICSF Primary Menu Panel

The Administrative Control Function panel appears. See Figure 46.

```
CSFACF00 ------------- ICSF Administrative Control Functions
COMMAND ===>
          Active CKDS: CRYPTO25.HCR7704.CKDS
          Active PKDS: CRYPTO25.HCR7704.PKDS

To change the status of a control, enter the appropriate character
(E - ENABLE, D - DISABLE) and press ENTER.

         Function                                  STATUS
         --------                                  ------
.  Dynamic CKDS Access                           ENABLED
.  PKA Callable Services                         ENABLED
.  PKDS Read Access                              ENABLED
.  PKDS Write, Create, and Delete Access         DISABLED
```

Figure 46. Enabling and Disabling the PKA Callable Services

2. Enter the appropriate character and press ENTER.
   - To enable the PKA callable services, enter an 'E' before the PKA Callable Services function.

     Note: If using a PKDS, you must also enable PKDS Read and PKDS Write.
   - To disable the PKA callable services, enter a 'D' before the PKA Callable Services function.

     Note: Disabling PKA callable services also disables PKDS Read and PKDS Write access.

Changing PKA Master Keys

To change the PKA master keys:

1. Disable the PKA callable services as described previously.
2. Return to the primary menu and select option 1, COPROCESSOR MGMT, and press enter.

   The Coprocessor Management panel appears.

```
CSFCMP00 ---------------- ICSF Coprocessor Management -------------
COMMAND ===>

Select the coprocessors to be processed and press ENTER.
Action characters are: A, D, E, R, and S. See the help panel for details.

COPROCESSOR  MODULE ID/SERIAL NUMBER                  STATUS
-----------  -------------------------------          -------

_ A06                                                  ACTIVE
_ A07                                                  ACTIVE
E C0         E589C396944007A6 5D40369997A386F4         ACTIVE
E C1         0AA379BFD2387960 0367DC04533125FF         ACTIVE
E P00        41-00YE1                                  ACTIVE
E P01        41-00K11                                  ACTIVE
E P02        41-0A355                                  ACTIVE
_ P03        41-0BA3F                                  ONLINE
_ P04        41-0RT2T                                  DEACTIVATED
_ P05        41-00342                                  DISABLED
```

Figure 47. Selecting the coprocessor on the Coprocessor Management Panel

3. Select the coprocessor(s) for PKA master key entry by entering 'E' before the coprocessor and pressing enter.

   The Clear Master Key Entry panel appears. See Figure 48. You need to RESET to clear the contents of the registers before you can set a new key value.

   In this example, `ALL-PKA` has been entered, as SMK=KMMK. If this was not the case, SMK would have been used.

```
CSFDKE10 -------------- ICSF - Clear Master Key Entry ---------
COMMAND ===>

             CCF DES/PCICC SYM-MK new master key register     : EMPTY
             CCF Signature/PCICC ASYM-MK master key register  : NOT THE SAME
             CCF Key management master key register           : FULL


 Specify information below
   Key Type  ===> ALL-PKA      (DES, SMK, KMMK, ALL-PKA)

   Part      ===> RESET        (RESET, FIRST, MIDDLE, FINAL)


   Checksum  ===> 00

   Key Value ===> 0000000000000000
             ===> 0000000000000000
             ===> 0000000000000000   (SMK, KMMK and ALL-PKA only)
```

Figure 48. The Clear Master Key Entry Panel to Reset Registers

4. When you select RESET, the Restart Key Entry Process panel is displayed. See Figure 49.

   This panel confirms your request to restart the key entry process. Press ENTER.

```
CSFDKE40 -------------- ICSF - Restart Key Entry Process -------------


ARE YOU SURE YOU WISH TO RESTART THE KEY ENTRY PROCESS?


  Restarting the process will clear the ALL-PKA master key register.


 WARNING:  Resetting the KMMK or SMK will invalidate any private
           internal key tokens in the PKDS


Press ENTER to confirm restart request
Press END   to  cancel restart request
```

Figure 49. Confirm Restart Request Panel

5. The Clear Master Key Entry panel again appears. See Figure 50. Enter the type of PKA master key you are changing and enter the key part.

```
CSFDKE10 -------------- ICSF - Clear Master Key Entry ---------
COMMAND ===>

             CCF DES/PCICC SYM-MK new master key register     : EMPTY
             CCF Signature/PCICC ASYM-MK master key register  : EMPTY
             CCF Key management master key register           : EMPTY


 Specify information below
   Key Type  ===> ALL-PKA      (DES, SMK, KMMK, ALL-PKA)

   Part      ===> FIRST        (RESET, FIRST, MIDDLE, FINAL)


   Checksum  ===> 59

   Key Value ===> 8F887096A8D4922B
             ===> 75D1189666F4DAA7
             ===> 9B28AEFA8C47760F  (SMK, KMMK and ALL-PKA only)
```

Figure 50. The Clear Master Key Entry Panel with First Key Values

6. Fill in the panel
   a. Enter the master key type in the Key Type field.

      In this example we are entering ALL-PKA. A PKA master key requires at least two key parts. You may enter additional key parts if necessary. ALL-PKA includes the SMK, KMMK and ASYM-MK.
   b. Enter FIRST in the Part field.
   c. Enter the two-digit checksum and the three 16-digit key values.
   d. When all the fields are complete, press ENTER.

If the checksum entered in the checksum field matches the checksum that the cryptographic coprocessor calculated, the key part is accepted. The message at the top of the panel states `KEY PART LOADED`, as shown in Figure 29 on page 61.

The Signature/PCICC ASYM-MK register status and KMMK status change to `PART FULL`. The hash pattern that is calculated for the key part appears near the bottom of the panel. Compare it with the pattern generated by the checksum, VP, HP utility or provided by the person who gave you the key part value to enter.

e. Record the hash pattern.

7. If the checksums do not match, the message `Invalid Checksum` appears. If this occurs, follow this sequence to resolve the problem:
   a. Reenter the checksum.
   b. If you still get a checksum error, recalculate the checksum.
   c. If your calculations result in a different value for the checksum, enter the new value.
   d. If your calculations result in the same value for the checksum, or if a new checksum value does not resolve the error, reenter the key part halves and checksum.
8. Now enter the FINAL key part.

```
CSFDKE10 -------------- ICSF - Clear Master Key Entry ---------
COMMAND ===>

             CCF DES/PCICC SYM-MK new master key register     : EMPTY
             CCF Signature/PCICC ASYM-MK master key register  : PART FULL
             CCF Key management master key register           : PART FULL


 Specify information below
   Key Type  ===> ALL-PKA      (DES, SMK, KMMK, ALL-PKA)

   Part      ===> FINAL        (RESET, FIRST, MIDDLE, FINAL)


   Checksum  ===> 53

   Key Value ===> 8FDAD096A8D4922B
             ===> 75D1189ADAF4DAA7
             ===> 9B28333A8C47760F  (SMK, KMMK and ALL-PKA only)
```

Figure 51. The Clear Master Key Entry Panel with Final Key Values

9. Fill in the panel
   a. Enter the master key type in the Key Type field.

      In this example we are entering ALL-PKA. ALL-PKA includes the SMK, KMMK and ASYM-MK.

   b. Enter FINAL in the Part field.
   c. Enter the two-digit checksum and the three 16-digit key values.
   d. When all the fields are complete, press ENTER.

      If the checksum entered in the checksum field matches the checksum that the cryptographic coprocessor calculated, the key part is accepted. The message at the top of the panel states `KEY PART LOADED`, as shown in Figure 29 on page 61.

The Signature/PCICC ASYM-MK master key register status changes to NOT THE SAME. This is because the PCICC current ASYM-MK register is loaded with the value in the new master key register and the new ASYM-MK register is empty. The KMMK status changes to FULL.

The hash pattern that is calculated for the key part appears near the bottom of the panel. Compare it with the pattern generated by the checksum, VP, HP utility or provided by the person who gave you the key part value to enter.

e. Record the hash pattern.

10. If the checksums do not match, the message Invalid Checksum appears. If this occurs, follow this sequence to resolve the problem:
    a. Reenter the checksum.
    b. If you still get a checksum error, recalculate the checksum.
    c. If your calculations result in a different value for the checksum, enter the new value.
    d. If your calculations result in the same value for the checksum, or if a new checksum value does not resolve the error, reenter the key part halves and checksum.
11. When you have entered the PKA master keys correctly, the PKA master key registers are active when the final key part is loaded. You must then reencipher and activate the PKDS (“Reenciphering and Activating the PKDS”) and enable PKA callable services“Enabling and Disabling PKA Services” on page 79. Also enable PKDS Read and PKDS Write, Create and Delete.

Reenciphering and Activating the PKDS

After changing the PKA master keys, you must reencipher the private keys. You must have a PCICC to reencipher. Reenciphering and activating the PKDS automatically refreshes the PKDS cache, as does starting ICSF.

To reencipher the PKDS after the PKA SMK and ASYM-MK have been changed, go to the Master Key Management panel and select option 5.

Note: Only keys enciphered under the SMK and the ASYM-MK are reenciphered. PKDS reencipher will not be able to reencipher private keys encrypted under the CCF key management key (KMMK) if the KMMK does not equal the SMK. If this is the case, see “Setting the Same Value for the SMK and KMMK” on page 86 before you reencipher.

```
CSFMKM00 ---------------- ICSF - Master Key Management -------------
OPTION ===> 5


Enter the number of the desired option.

  1 INIT/REFRESH CKDS - Initialize a Cryptographic Key Data Set or activate
                        an updated Cryptographic Key Data Set
  2 SET MK            - Set a DES master key
  3 REENCIPHER CKDS   - Reencipher the CKDS prior to changing the DES master
                        key
  4 CHANGE MK         - Change the DES master key and activate the
                        reenciphered CKDS
  5 REENCIPHER PKDS   - Reencipher the PKA Cryptographic Key Data Set
  6 ACTIVATE PKDS     - Activate the PKDS after it has been reenciphered
  7 REFRESH CACHE     - Refresh the PKDS cache if enabled
```

Figure 52. Selecting the Reencipher PKDS Option on the Master Key Management Panel

The Reencipher PKDS panel appears. In the Input PKDS field, specify the name of the PKDS that you want ICSF to reencipher under the current SMK and ASYM-MK.

In the Ouput PKDS field, specify the name of an empty VSAM data set. ICSF places the reenciphered keys in this data set.

```
CSFCMK11 ---------------- ICSF - Reencipher PKDS -------------
COMMAND ===>


To reencipher all PKDS entries from encryption under the old signature/
asymmetric-keys master key to encryption under the current master key, enter
the PKDS names below.

   Input  PKDS ===> PKDS.CURRENT.MASTER

   Output PKDS ===> PKDS.NEW.MASTER

Press ENTER to reencipher the PKDS.
Press END   to exit to the previous menu
```

Figure 53. Reencipher PKDS

Press enter to reencipher the PKDS. Reenciphering automatically refreshes the PKDS cache. Once successful, you will then want to activate the PKDS. Return to the Master Key Management panel and select option 6.

```
CSFMKM00 ---------------- ICSF - Master Key Management -------------
OPTION ===> 6


Enter the number of the desired option.

  1 INIT/REFRESH CKDS - Initialize a Cryptographic Key Data Set or activate
                        an updated Cryptographic Key Data Set
  2 SET MK            - Set a DES master key
  3 REENCIPHER CKDS   - Reencipher the CKDS prior to changing the DES master
                        key
  4 CHANGE MK         - Change the DES master key and activate the
                        reenciphered CKDS
  5 REENCIPHER PKDS   - Reencipher the PKA Cryptographic Key Data Set
  6 ACTIVATE PKDS     - Activate the PKDS after it has been reenciphered
  7 REFRESH CACHE     - Refresh the PKDS cache if enabled
```

Figure 54. Selecting the Activate PKDS Option on the Master Key Management Panel

The Activate PKDS panel appears. Enter the name of the PKDS that you want ICSF to use. The PKDS must have already been reenciphered under the current Signature/Asymmetric-keys master key.

```
 CSFCMK21 ---------- ICSF - Activate PKA Cryptographic Key Data Set -------
 COMMAND ===>


 Enter the name of the new PKDS below.

    New  PKDS ===> PKDS.NEW.MASTER

Press ENTER to activate the PKDS.
Press END   to exit to the previous menu
```

Figure 55. Activate PKDS

After you press ENTER, the PKDS becomes active. Activation automatically refreshes the PKDS cache.

Setting the Same Value for the SMK and KMMK

It is highly recommended that the KMMK, SMK and ASYM-MK be equal. This will facilitate migration to new features on crypto hardware.

If you are a new user and using Pass Phrase Initialization, ensure that you answer Y for the 'Signature MK = Key Management MK? on Figure 12 on page 46. If using Clear Key Entry, make sure that you enter the same value for your SMK and KMMK.

If you are an existing user and for some reason your KMMK does not equal the SMK and ASYM-MK, you should follow this procedure. You must have a PCICC on your system.

1. Disable PKA services (see "Enabling and Disabling PKA Services" on page 79).
2. Determine the value of the KMMK
   a. If you used Pass Phrase Initialization, go to the main menu and and choose option 5, UTILITY. Select option 5, PPKEYS.

```
CSFUTL00 ---------------- ICSF - Utilities --------------------------
OPTION ===> 5


Enter the number of the desired option.

  1  ENCODE        -  Encode data
  2  DECODE        -  Decode data
  3  RANDOM        -  Generate a random number
  4  CHECKSUM      -  Generate a checksum and verification and
                      hash pattern
  5  PPKEYS        -  Generate master key values from a pass phrase
```

Figure 56. ICSF Utilities Panel

The Master Key Values from Pass Phrase panel appears (Figure 57).

```
 CSFPPM00 ------------ ICSF - Master Key Values from Pass Phrase -----

Pass Phrase ( 16 to 64 characters)
 ==>_____________________________________________________________

Signature/Asymmetric-keys master key  : 0000000000000000
                                      : 0000000000000000
                                      : 0000000000000000

Key Management master key             : 0000000000000000
                                      : 0000000000000000
                                      : 0000000000000000
```

Figure 57. ICSF Master Key Values from Pass Phrase Panel

Enter the previously used pass phrase and your SMK and KMMK values will be displayed.

b. If you used Clear Master Key entry, you must retrieve the value from your written files.

3. Use this value (of the KMMK) as the new SMK and ASYM-MK values (see “PKA Master Keys and the PKDS” on page 79).
4. Reencipher and Activate the PKDS (see “Reenciphering and Activating the PKDS” on page 84).

Clearing Master Keys

For security reasons, your installation may need to clear the master keys. This may be required, for example, before turning the processor hardware over for maintenance.

If you have a TKE workstation, you can use it to zeroize all domains that have keys loaded. Refer to *z/OS ICSF TKE Workstation User’s Guide 2000* for more information.

If you do not have a TKE workstation, you might want to consider nullifying the master keys. To do this you would need to enter a new DES master key, reencipher a dummy CKDS, and change the master key. You would need to perform this operation twice to ensure that the master key is cleared from the auxiliary (old) master key register. You would also need to reset both of the PKA master keys and process the PCICC master keys.

You can also use the zeroize function on the Support Element panel. Besides clearing the master keys, this also clears all domains and user data.

Adding PCICC After CCF Initialization

You may need to initialize PCI Cryptographic Coprocessors after system initialization.

Note: Use this procedure if you did not run the Pass Phrase Initialization utility. If you used the utility, see Chapter 4, “Using the Pass Phrase Initialization Utility”, on page 45.

Follow the following procedure.

1. Select option 1, COPROCESSOR MGMT, on the Primary Menu panel.
2. The Coprocessor Management panel, as shown in Figure 58, appears.

```
CSFCMP00 ---------------- ICSF Coprocessor Management -------------
COMMAND ===>

Select the coprocessors to be processed and press ENTER.
Action characters are: A, D, E, R, and S. See the help panel for details.

COPROCESSOR  MODULE ID/SERIAL NUMBER                 STATUS
-----------  ------------------------------          -------

_ A06                                                 ACTIVE
_ A07                                                 ACTIVE
_ C0         E589C396944007A6 5D40369997A386F4        ACTIVE
_ C1         0AA379BFD2387960 0367DC04533125FF        ACTIVE
_ P00        41-00YE1                                 ACTIVE
_ P01        41-00K11                                 ACTIVE
_ P02        41-0A355                                 ACTIVE
E P03        41-0BA3F                                 ONLINE
_ P04        41-0RT2T                                 DEACTIVATED
_ P05        41-00342                                 DISABLED
_
```

Figure 58. Selecting a coprocessor on the Coprocessor Management Panel

3. Select the Coprocessor to be processed by entering 'E' next to the Coprocessor.
4. The Clear Master Key Entry panel appears. See Figure 59 on page 89.

```
CSFDKE10------------- ICSF - Clear Master Key Entry -----------------
COMMAND ===>

              CCF DES/PCICC SYM-MK new master key register     : EMPTY
              CCF Signature/PCICC ASYM-MK master key register  : EMPTY
              CCF Key management master key register           : EMPTY


  Specify information below
    Key Type  ===> ___               (DES, SMK, KMMK, ALL-PKA)

    Part      ===> ______            (RESET, FIRST, MIDDLE, FINAL)

    Checksum  ===> 00

    Key Value ===> 0000000000000000
              ===> 0000000000000000
              ===> 0000000000000000   (SMK, KMMK and ALL-PKA only)



  Press ENTER to process.
  Press END   to exit to the previous menu.
```

Figure 59. The Clear Master Key Entry Panel to Reset Registers

Ensure that the CCF Signature/PCICC ASYM-MK master key register field indicates EMPTY. If it does not, you will need to RESET to clear the contents of the registers before you can set a new key value.

5. You must now load the SYM-MK and ASYM-MK keys into your system.

   If you are going to reload your current master keys, you need to know the current master key value and checksum. If you want the PCICC to become ACTIVE after CCF initialization, you MUST enter the same master key values.

   Follow the instructions on “Entering the First Master Key Part” on page 58.

6. After all key parts have been loaded, SET the master key. From the Primary Menu panel choose option 2 - Master Key. From the Master Key Management panel, choose option 2 - SET MK.

Chapter 6. Running in a Sysplex Environment

ICSF is supported in a SYSPLEX environment. Both the CKDS and PKDS can be shared across systems in a sysplex.

There is no impact on the SYSPLEX environment if you have an IBM @server zSeries 990.

Managing the CKDS

The systems sharing a CKDS may be different LPARs on the same system or different systems across multiple zSeries and S/390 Processors. The only requirement for sharing the CKDS is that the same DES Master Key be installed on all systems sharing that CKDS. It is not required to share the CKDS across a sysplex. Each system may have its own DES Master Key and its own CKDS. A sysplex may have a combination of systems that share a CKDS and individual systems with separate CKDSs.

There is no requirement that the DOMAINs must be the same to share a CKDS.

When sharing the CKDS, a few precautions should be observed:

- Dynamic CKDS services update the DASD copy of the CKDS and the in-storage copy on the system where it is run. There is no sysplex broadcast of the update. In order to update the in-storage copy of all images that share the CKDS, you must perform a CKDS REFRESH on each image. This can be done by using either the TSO panels or the CSFEUTIL utility.
- The CKDS may be shared between ICSF V2.1, and OS/390 and z/OS ICSF systems. However, you must take care when adding keys of type IMPORTER, EXPORTER, PINGEN, PINVER, IPINENC, or OPINENC to the CKDS if the key has a control vector supported by the PCI cryptographic coprocessor but not supported by the CCF. A toleration APAR must be installed on ICSF systems below V2R10 to ensure that ICSF services will fail a request to use a key which contains a non-CCF control vector. The toleration APAR is OW43926.

Setting DES Master Keys when Sharing a CKDS

Setting master keys for the first time in a sysplex environment is best accomplished by using the Pass Phrase Initialization utility. You need to allocate an empty CKDS and update the options data set on all the LPARs that will be sharing the CKDS. When ICSF is started for the first time, the CKDS needs to be initialized once. All the other LPARs need only to load the same DES master key, and then set the master key.

Using Pass Phrase Initialization

Use the Pass Phrase Initialization utility to initialize ICSF in a CKDS shared environment. From the first LPAR, follow the instructions in Chapter 4, “Using the Pass Phrase Initialization Utility”, on page 45. Once the LPAR has been successfully initialized, from each LPAR that is sharing the same CKDS, go to the Pass Phrase Initialization panel:

- Enter the same exact pass phrase as entered on the first LPAR
- Enter the same exact CKDS name as entered on the first LPAR
- Respond **N** to ″Initialize the CKDS″
- Respond to the remaining questions as for the first LPAR

These steps will load and set the same master keys as in the first LPAR and activate the same CKDS.

Using Clear Master Key Entry

You can alternatively use clear master key entry to set master keys in a sysplex environment. Load your master keys in the first LPAR as described in “Entering Clear Master Key Parts” on page 51. For all subsequent LPARs, enter the master keys as described in “Reentering Master Keys After They have been Cleared” on page 72.

Changing DES Master Keys when Sharing a CKDS

Changing master keys should be done with care in a sysplex environment. Follow the procedure in “Changing Master Keys” on page 74. Changing the master key and reenciphering the CKDS should be done on an image running the latest level of ICSF. On the other images sharing that CKDS, enter the new master key and then change the master key. Reenciphering the CKDS is not necessary. During the master key change across a sysplex there should not be any applications that pass internal tokens from one image to another.

Managing the PKDS

The systems sharing a PKDS may be different LPARs on the same system or different systems across multiple zSeries and S/390 Processors. The only requirement for sharing the PKDS is that the same PKA Master Keys be installed on all systems sharing that PKDS. It is not required to share the PKDS across a sysplex. Each system may have its own PKA Master Keys and its own PKDS. A sysplex may have a combination of systems that share a PKDS and individual systems with separate PKDSs.

It is highly recommended that the SMK and KMMK be the same on all systems sharing the PKDS in order to reencipher the PKDS after a PKA master key change. PKDS reencipher requires a PCICC on your system. PKDS reencipher is not supported on CCF-only systems. For instuctions on creating this environment, see “Setting the Same Value for the SMK and KMMK” on page 86.

In addition, ICSF optionally maintains a cache of frequently used PKDS records. The size of the PKDS cache is set in the installation options data set. It is an optional feature, with a default of 64 records.

If a cache is being maintained, care must be taken when deleting or changing an existing PKDS record. When such an update is made on one system, that change is not automatically reflected in the cache of other systems. To ensure integrity of the cache after PKDS updates, the PKDS cache on other systems should be refreshed. Note that adding PKDS records does not require refreshing of the PKDS cache.

There are implications to sharing the PKDS across multiple levels of ICSF. Refer to *z/OS ICSF System Programmer’s Guide*.

When sharing the PKDS, a few precautions should be observed:

- Support for reenciphering and activating the PKDS is available in z/OS V1 R2. If you are sharing a PKDS between z/OS V1R2 and a previous level of ICSF, you need to install a toleration PTF on the backlevel systems. The toleration PTF will enable backlevel systems to activate the reenciphered PKDS. Toleration APAR OW49386 is required on the following systems in order to activate the PKDS: -

HCRP210 (standalone), HCRP220(OS/390 V2 R6, OS/390 V2 R7, OS/390 V2 R8), HCRP230 (OS/390 V2 R9), and HCR7703 (OS/390 V2 R10 and z/OS V1 R1).

- Support for new tokens is available in OS/390 V2 R9. If you are sharing a PKDS between OS/390 R9 and a previous level of ICSF, you need to install a toleration PTF on the back level systems. The toleration PTF prevents the backlevel system from updating PKDS records of retained keys. It will also convert new X'06' modulus exponent RSA internal tokens to old X'02' forms (useable on back level systems). However, the back level system can use the converted token ONLY if the KMMK is equal to the SMK.
- With OS/390 V2 R6 and above, if you share the PKDS with lower level releases of ICSF, the following APARS must be installed:
  - HCRP210 (ICSF/MVS V2 R1, OS/390 V2 R4 ICSF, OS/390 V2 R5 ICSF) must have APARS OW33234 and OW37623 installed. If you are running OS/390 V2 R9 ICSF, you must also have APAR OW43275 installed on HCRP210.
  - HCRP220 (OS/390 V2 R6 ICSF) must have APAR OW37623 installed. If you are running OS/390 V2 R9 ICSF, you must also have APAR OW43275 installed on HCRP220.

Changing PKA Master Keys when Sharing a PKDS

If you have multiple systems sharing a PKDS and make changes to PKDS records, you must reencipher and activate the PKDS. A PCICC on your system is required for this process.

Assume you have two systems, A and B sharing a PKDS data set, OLDPKDS. The steps to reencipher and activate are:

1. From SYSTEM A, disable PKA callable services (enter a 'D' before the function (see "Enabling and Disabling PKA Services" on page 79).
2. On SYSTEM B, disable PKDS Write, Create and Delete Access (enter a 'D' before the function as in Figure 60)

```
CSFACF00 ------------- ICSF Administrative Control Functions
COMMAND ===>
          Active CKDS: CRYPTO25.HCR7704.CKDS
          Active PKDS: CRYPTO25.HCR7704.PKDS

To change the status of a control, enter the appropriate character
(E - ENABLE, D - DISABLE) and press ENTER.

         Function                                  STATUS
         --------                                  ------

 .  Dynamic CKDS Access                            ENABLED
 .  PKA Callable Services                          ENABLED
 .  PKDS Read Access                               ENABLED
 D  PKDS Write, Create, and Delete Access          ENABLED
```

Figure 60. Administrative Control Functions

3. On system A, change the Master Key (see "PKA Master Keys and the PKDS" on page 79)
4. On system A, enable PKA callable services (see "Enabling and Disabling PKA Services" on page 79).

5. On system A, reencipher OLDPKDS, creating NEWPKDS (see “Reenciphering and Activating the PKDS” on page 84)
6. On system A, activate NEWPKDS
7. On system A, enable PKA Read, Write, Create and Delete Access (see Figure 60 on page 93)
8. On system B, disable PKA callable services (see “Enabling and Disabling PKA Services” on page 79).
9. On system B, change the Master Key (see “PKA Master Keys and the PKDS” on page 79)
10. On system B, activate NEWPKDS
11. On system B, enable PKA callable services (see “Enabling and Disabling PKA Services” on page 79).
12. On system B, enable PKA Read, Write, Create and Delete Access (see Figure 60 on page 93)

Refreshing the PKDS Cache

When you are sharing the PKDS in a sysplex, there will be occasions when you change or delete PKDS records, causing changes to the PKDS cache. In order to reflect the change on other systems in your sysplex, you must refresh the cache. From the Master Key Management panel, select option 7 and press enter.

The PKDS cache is refreshed automatically whenever ICSF is started or when the PKDS is reenciphered or activated.

Note: PKDSCACHE, an installation option, defines the size of the PKDS Cache in records. PKDSCACHE can be implemented on OS/390 V2 R10 and z/OS V1 R1 by installing APAR OW48568.

```
CSFMKM00 ---------------- ICSF - Master Key Management -------------
OPTION ===> 7


Enter the number of the desired option above.

  1 INIT/REFRESH CKDS - Initialize a Cryptographic Key Data Set or activate
                        an updated Cryptographic Key Data Set
  2 SET MK            - Set a DES master key
  3 REENCIPHER CKDS   - Reencipher the CKDS prior to changing the DES master
                        key
  4 CHANGE MK         - Change the DES master key and activate the
                        reenciphered CKDS
  5 REENCIPHER PKDS   - Reencipher the PKA Cryptographic Key Data Set
  6 ACTIVATE PKDS     - Activate the PKDS after it has been reenciphered
  7 REFRESH CACHE     - Refresh the PKDS cache if enabled
```

Figure 61. Selecting the Refreshing the PKDS Cache Option on the Master Key Management Panel

Chapter 7. Managing Cryptographic Keys by Using the Key Generator Utility Program

The key generator utility program (KGUP) generates and maintains keys in the cryptographic key data set (CKDS). The CKDS stores DATA keys, MAC keys, PIN keys, and transport keys. Although ANSI transport keys are stored in the CKDS, KGUP does not support the generation or import of ANSI transport keys. KGUP does not support non-standard CV keys.

To run KGUP, ICSF must be active, it must contain a master key, and the CKDS must be initialized.

You use KGUP to perform the following tasks:

- Generate or enter keys
- Maintain CKDS entries by deleting or renaming the entries

When KGUP generates or receives a key value, the program either adds a new entry or updates an existing entry in the CKDS. For information about how KGUP generates and receives keys to establish key exchange with other systems, see “Using KGUP for Key Exchange” on page 98.

Each key that KGUP generates exists in the CKDS enciphered under your system’s master key. Before the master key enciphers a key, the cryptographic facility exclusive ORs the master key with a pattern of characters called a control vector. A master key exclusive ORed with a control vector is called a master key variant.

A unique control vector exists for each type of key the master key enciphers. The cryptographic facility exclusive ORs the master key with the control vector associated with the type of key the master key will encipher. The control vector ensures that a key is only used in the cryptographic functions for which the key is intended. For example, the control vector for an input PIN encryption key ensures that such a key can be used only in PIN translate and PIN verification functions.

When you specify to KGUP to generate an input PIN-encrypting key, the cryptographic facility creates a master key variant for the key. The master key variant is a product of exclusive ORing the master key with the control vector associated with an input PIN-encrypting key. This master key variant enciphers the input PIN-encrypting key so the input PIN-encrypting key is in operational form. KGUP places the input PIN-encrypting key in a CKDS entry.

You use control statements to specify the functions for KGUP to perform. The control statement specifies the task you want KGUP to perform and information about the CKDS entry that is affected. For example, to have KGUP generate an importer key-encrypting key, you use a control statement like:

```
ADD  LABEL(KEY1)  TYPE(IMPORTER)
```

When KGUP processes the control statement, the program generates a key value and encrypts the value under a master key variant for an importer key-encrypting key. KGUP places the key in a CKDS entry labelled KEY1. The key type field of the entry specifies IMPORTER. For a description of the fields in a CKDS entry, see “Specifying KGUP Data Sets” on page 119.

You store the control statements in a data set. You must also specify other data sets that KGUP uses when the program processes control statements. You submit a batch job stream to run KGUP. In the job control statements, you specify the names of the data sets that KGUP uses.

KGUP changes a disk copy of the CKDS according to the functions you specify with the control statements. After KGUP changes the disk copy of the CKDS, you may replace the in-storage copy of the CKDS with the disk copy using the ICSF panels.

To use KGUP, you must perform the following tasks:
- Create control statements
- Specify data sets
- Submit a job stream

You may also want to refresh the CKDS with the disk copy of the CKDS that KGUP updated. You can use the KGUP panels to help you perform these tasks. However you can also use KGUP without accessing the panels. This chapter first describes each of the tasks to run KGUP, and then describes how to use the panels to perform the tasks.

Disallowing Dynamic CKDS Updates During KGUP Updates

ICSF prioritizes changes to the CKDS sequentially, regardless of the source. A KGUP job does not have priority over application calls to the dynamic CKDS update services. Exclusive use of the CKDS by any one application call is minimal, however. For this reason, ICSF allows for a maximum concurrent usage of the CKDS by both KGUP and the dynamic update services.

Before you perform any function that affects the current CKDS (such as reenciphering, refreshing, or changing the master key), you should consider temporarily disallowing the dynamic CKDS update services.

If you are planning to use KGUP to make significant changes to the CKDS, you should disallow dynamic CKDS update. If an application tries to use the dynamic CKDS update services when they are disallowed, the return code indicates that the CKDS management service has been disabled by the system administrator.

To disallow dynamic CKDS access, perform the following tasks:

1. Choose option 4, Administrative Control Functions, on the Primary Menu Panel, as shown in Figure 62 on page 97.

```
CSF@PRIM ---- Integrated Cryptographic Service Facility ---------
OPTION ===> 4

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

        Licensed Materials - Property of IBM

     5685-051 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.


Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 62. Selecting the Adminstrative Control Option on the Primary Menu Panel

The Administrative Control Functions panel appears. See Figure 63.

2. Enter a 'D' to disallow dynamic CKDS access.

```
CSFACF00 ------------- ICSF Administrative Control Functions
COMMAND ===>
          Active CKDS: CRYPTO25.HCR7704.CKDS
          Active PKDS: CRYPTO25.HCR7704.PKDS

To change the status of a control, enter the appropriate character
(E - ENABLE, D - DISABLE) and press ENTER.

         Function                                  STATUS
         --------                                  ------

 D  Dynamic CKDS Access                             ENABLED
 .  PKA Callable Services                           ENABLED
 .  PKDS Read Access                                ENABLED
 .  PKDS Write, Create, and Delete Access           DISABLED

 Press ENTER to go to the selected option.
 Press END   to exit to the previous menu.
```

Figure 63. Selecting to Disallow Dynamic CKDS Access on User Control Functions Panel

3. Press ENTER.

   The message CKDS UPDATES DISABLED appears in the upper right-hand corner of the panel.

4. Press END to return to the Primary Menu panel.

Using KGUP for Key Exchange

KGUP generates keys that are complementary keys. Complementary keys have the same clear key value for corresponding key types. KGUP generates and maintains the following types of complementary keys:

- Data-encrypting (DATA) and data-translation (DATAXLAT) keys
- Importer key-encrypting key and exporter key-encrypting key
- Input PIN-encrypting key and output PIN-encrypting key
- MAC generation key and MAC verification key
- PIN generation key and PIN verification key

When you distribute keys or PINs, your system has one key, and the other system has the complementary key. For example, when your system sends a DATA key to another system, the importer and exporter key-encrypting keys at the systems complement each other. The DATA key is encrypted under an exporter key-encrypting key at your system. The DATA key is decrypted by the complementary importer key-encrypting key at the receiving system.

When KGUP generates a key, the other system involved in the key or PIN exchange needs the complement of the key. When KGUP generates a key, the program also generates a control statement to create the complement of the key. You send the control statement to the other system which uses the statement to create the complementary key.

For example, when you use KGUP to create an input PIN-encrypting key, KGUP also creates a control statement for the complementary output PIN-encrypting key. You send the control statement to another system. The other system uses the control statement to create the output PIN-encrypting key. Then your system can send PIN blocks to the other system.

For some key types you can choose the output key type by specifying the OUTTYPE parameter on a KGUP ADD or CREATE statement. For example, you can generate a DATA key for inclusion into the CKDS and export a copy of the key as either a DATA key or a DATAXLAT key. If you export the copy of the DATA key as a DATA key, the receiver of the key can use it to decipher data. If you export the copy of the DATA key as a DATAXLAT key, the receiver can use the key only to translate cipher text from one DATAXLAT key to another. The receiver of the DATAXLAT key cannot use the key to actually decipher the data.

KGUP stores the complementary key control statement in a data set. Because some cryptographic systems may not use KGUP control statements, KGUP also stores complementary key information as a record in a different data set. The information is not in the form of a control statement. You process and send the information to a system which creates the complementary key.

When KGUP generates a key, the program also generates information to create the complementary key. This information includes the complementary key value. The value is either a clear key value or encrypted key value. For an encrypted key value, the program encrypts the value under an exporter key. The importer key that complements this exporter key already exists at the other system. The importer key is one key in a complementary transport key pair that your system already established with the other system. The pair would be an importer key on the other system and an exporter key on your system. The other system reenciphers the value from under the importer key to under its master key to generate the complementary key.

Besides generating keys and complementary key information, KGUP imports key values that are sent from other systems. The program can receive a control statement to create a key that is the complement of a key on another system. The key value your KGUP receives may be encrypted under a transport key. The transport key would be one key of a complementary transport key pair that you already established with the other system. The pair would be an exporter key on the other system and an importer key on your system. KGUP reenciphers the complementary key from under the importer key to under the master key and places the key in the CKDS.

For KGUP to send or receive keys in a key exchange with another system, the systems must previously establish a pair of complementary transport keys. For example, KGUP on one system defines the pair and generates the importer key in the clear. KGUP on the other system uses this value to define a pair of keys that are complements of the keys at the original site. For an example of how two ICSF systems establish pairs of complementary transport keys for key exchange, see “Scenario of Two ICSF Systems Establishing Initial Transport Keys” on page 151.

The cryptographic facility exclusive ORs a transport key with a control vector before using the transport key to encipher a key. A transport key exclusive ORed with a control vector is called a transport key variant. ICSF uses the control vector associated with the key type that the transport key will encipher. The control vector ensures that when another site imports the key, the resulting operational key can only be the type that the control vector indicates. For example, the control vector for a PIN verification key ensures that the system that receives the key can import the key only as a PIN verification key.

When KGUP generates a PIN generation key, the program generates a key value to create a PIN verification key. You can specify that the key value be an encrypted key value. When you do this, ICSF exclusive ORs the transport key with the control vector for a PIN verification key to create the transport key variant. Then the cryptographic facility enciphers the PIN verification key under the transport key variant.

To view the specific control vector value that is associated with each type of key to create master key variants and transport key variants, see Appendix B, “Control Vector Table”.

Transport key variants ensure that the receiving system uses the key as the type of key that the sending system intended. However transport key variants can only be used if both systems recognize transport key variants. You should use transport key variants when exchanging keys with the 4758 PCI Cryptographic Coprocessor. However, systems with some cryptographic products, such as PCF, do not recognize control vectors. When you exchange keys with such a system, a key that you send or receive is enciphered under a transport key rather than a transport key variant. You just specify to KGUP that the transport key should not be exclusive ORed with a control vector.

You can define a pair of complementary transport keys with another system so your system and the other system can exchange keys without control vectors. You use a control statement to indicate to KGUP to produce these keys. Then send the clear value that KGUP produced to the PCF system so the system can generate the corresponding complementary pair of keys. Then you use the transport keys to exchange other keys. Refer to “Scenario of an ICSF System and a PCF System

Establishing Initial Transport Keys" on page 152 for an example of how to establish pairs of complementary transport keys for key exchange between an ICSF system and a PCF system.

You can also use KGUP to create complementary keys that are used by two different systems. Neither key would be operational on your system so KGUP would not update your CKDS. After KGUP generates the complementary key information, you send it to the two systems that need to share complementary keys.

Using KGUP Control Statements

You use control statements to specify the function you want the key generator utility program (KGUP) to perform. You use job control language (JCL) to submit the control statements to KGUP. You can create and submit KGUP control statements either on your own or using the KGUP panels.

You specify information to KGUP using an ADD, UPDATE, DELETE, RENAME or SET control statement. You use keywords on the control statement to specify:

- The function KGUP performs
- Information about the key that KGUP processes

For example, if you specify the KEY keyword on an ADD control statement, you supply a key which KGUP adds to the CKDS in an entry.

This topic describes the syntax of the control statements with their keywords. Use the following rules when interpreting the syntax of the control statements:

- Specify uppercase letters and special characters as shown in the examples.
- Lowercase letters represent keyword values that you must specify.
- A bar (|) indicates a choice (OR).
- Ellipses (...) indicates that multiple entries are possible.
- Braces ({ }) denote choices, one of which you must specify.
- Brackets ([ ]) denote choices, one of which you may specify.

General Rules for CKDS Records

There are some general rules for creating labels for CKDS key records.

- Each label can consist of up to 64 characters. The first character must be alphabetic or a national character (#, $, @). The remaining characters can be alphanumeric, a national character (#, $, @), or a period (.).
- Labels must be unique for DATA, DATAXLAT, MAC, MACVER, DATAM, DATAMV, and NULL keys.
- For compatibility with Version 1 Release 1 function, transport and PIN keys can have duplicate labels for different key types. Keys that use the dynamic CKDS update services to create or update, however, must have unique key labels.
- Labels must be unique for any key record, including transport and PIN keys, created or updated using the dynamic CKDS update services.

KGUP and the dynamic CKDS update services, unless they are modified by user-written exits, check for uniqueness according to these rules before making any change to the CKDS.

KGUP Uniqueness Checking

KGUP first checks to see if the label in the control statement matches a label that already exists in the CKDS.

If KGUP is processing an ADD control statement and there is no matching record, KGUP continues processing. Also, if KGUP is processing a RENAME control statement and there is no match for the *new-label* parameter, KGUP continues processing the control statement. If KGUP finds a matching label, KGUP then checks whether the key requires a unique label. If the key does not require a unique label, KGUP continues processing the ADD or RENAME control statement. If the key does require a unique label, KGUP stops processing the control statement and issues a message.

If KGUP is processing an UPDATE or DELETE control statement and there is no matching record, KGUP ends processing and issues an error message. Also, if KGUP is processing a RENAME control statement and there is no match for the *old-label* parameter, KGUP ends processing and issues an error message. If KGUP finds a matching label, KGUP continues processing the UPDATE, DELETE, or RENAME control statement.

Dynamic CKDS Update Services Uniqueness Checking

The dynamic CKDS update services require unique record labels in the CKDS. Each service checks to see if the label in the application call matches a label that already exists in the CKDS. For the Key Record Create service, if there is no matching record in the CKDS, ICSF continues processing the application call. If there is a match, ICSF stops processing and returns a return code and reason code to the application. For the Key Record Write and Key Record Delete services, if there is only one record in the CKDS that matches the label in the application call, ICSF continues processing the application call. If there is more than one matching record in the CKDS, ICSF stops processing and returns a return code and reason code to the application.

Syntax of the ADD and UPDATE Control Statements

The ADD and UPDATE control statements use the same keywords. The ADD control statement adds new keys to the CKDS. UPDATE changes existing key entries. Use the ADD or UPDATE control statement to specify that KGUP generate a key value or import a key value that you provide.

Refer to Figure 64 for the syntax of the ADD and UPDATE control statements.

```
{ADD | UPDATE}

  {LABEL(label1[,...,label64]) | RANGE(start-label,end-label)}

  TYPE(key-type)

  [OUTTYPE(key-type)]

  [TRANSKEY(key-label1[,key-label2]) | CLEAR]

  [NOCV]

  [LENGTH]

  [CDMF | DES]

  [KEY(key-value[,ikey-value])]
```

Figure 64. ADD and UPDATE Control Statement Syntax

LABEL (label1[,...,label64])

This keyword defines the names of the key entries for KGUP to process within the CKDS. KGUP processes a separate entry for each label. If you specify more than one label on an ADD or UPDATE control statement, the program uses identical key values in each entry.

You must specify at least one key label, and you can specify up to 64 labels with the LABEL keyword. For the general rules about key label conventions and uniqueness, see “General Rules for CKDS Records” on page 100.

On a KGUP control statement, you must specify either the LABEL or RANGE keyword. When you supply a key value on the control statement with the KEY keyword, you must specify the LABEL keyword.

RANGE (start-label, end-label)

This keyword defines the range of the multiple labels that you want KGUP to create or maintain within the CKDS.

The label consists of between 2 and 64 characters that are divided as follows:

- The first 1 to 63 characters are the label base. These characters must be identical on both the start-label and end-label and are repeated for each label in the range. For the general rules about key label conventions and uniqueness, see “General Rules for CKDS Records” on page 100.
- The last 1 to 4 characters form the suffix. The number of digits in the start-label and end-label must be the same, and the characters must all be numeric. These numeric characters establish the range of labels KGUP creates. The start-label numeric value must be less than the end-label numeric value.

KGUP creates a separate CKDS entry for each label including the start and end labels. The program generates a different key value for each entry it creates.

You cannot use the RANGE keyword when you supply a key value to KGUP. Only use RANGE to generate a key value. The RANGE and KEY keywords are mutually exclusive.

On a KGUP control statement, you must specify either the LABEL or RANGE keyword.

TYPE (key-type)

This keyword specifies the type of key you want KGUP to process. You can specify only one key type for each control statement. For DATA, DATAXLAT, MAC, MACVER, DATAM, DATAMV, and NULL key types, KGUP allows only one key per label. For all other key types, you can have keys with the same labels but different key types.

You can specify any of the following key types:

DATA Encryption/decryption key

DATAXLAT
: Cipher text translate key

DATAM
: Double-length MAC generation key

DATAMV
: Double-length MAC verification key

EXPORTER
: Exporter key-encrypting key

IMPORTER
: Importer key-encrypting key

IPINENC
Input PIN encryption key
MAC Single-length MAC generation key
MACVER
Single-length MAC verification key
NULL Used to create a null CKDS entry
OPINENC
Output PIN encryption key
PINGEN
PIN generation key
PINVER
PIN verification key

All these types of keys are stored in the CKDS.

Note: For compatibility with previous releases of OS/390 ICSF, KGUP stores internal versions of DATAM and DATAMV keys in the CKDS under the key types of MACD and MACVER, respectively.

OUTTYPE (key-type)
This keyword specifies the type of complementary key you want KGUP to generate for export. This keyword is valid only when you are requesting KGUP to generate keys and you also specify the CLEAR or TRANSKEY keywords. OUTTYPE is mutually exclusive with the KEY keyword. You cannot specify an OUTTYPE when you have chosen either DATAMV, PINVER, MACVER, or NULL for the key TYPE.

Refer to Table 4 for a list of the default and optional complementary key types for each of the 11 different key types. If OUTTYPE is not specified, KGUP generates the default complementary key that is shown in this table.

Table 4. Default and Optional OUTTYPES Allowed for Each Key TYPE

TYPE	OUTTYPE (Default)	OUTTYPE (Allowed)
DATA	DATA	DATA, DATAXLAT
DATAXLAT	DATAXLAT	DATAXLAT
DATAM	DATAMV	DATAM, DATAMV
DATAMV	Not Allowed	Not Allowed
EXPORTER	IMPORTER	IMPORTER
IMPORTER	EXPORTER	EXPORTER
IPINENC	OPINENC	OPINENC
MAC	MACVER	MAC, MACVER
MACVER	Not Allowed	Not Allowed
NULL	Not Allowed	Not Allowed
OPINENC	IPINENC	IPINENC
PINGEN	PINVER	PINVER
PINVER	Not Allowed	Not Allowed

TRANSKEY (key-label1[,key-label2])
This keyword identifies the label of a transport key that already exists in the CKDS. KGUP uses the transport key either to decrypt an imported key value or to encrypt a key value to send to another system.

When KGUP generates a key, the program enciphers the key under a master key variant. KGUP may also generate a key value that can be used to create the key's complement. You can have KGUP encrypt the key value under a transport key or transport key variant. On the control statement, use the TRANSKEY keyword to specify the transport key that KGUP should use to encipher the complementary key. You can send the encrypted key value to another system to create the complementary key.

When you generate an importer key-encrypting key to encipher a key stored with data in a file, you can request that KGUP not generate the complementary export key-encrypting key. You do this by not specifying the TRANSKEY or CLEAR keyword. This is also true for DATA and MAC keys.

When you input a key value that is in importable form, the key that is specified by the KEY keyword is enciphered under a transport key. KGUP reenciphers the key value from under the transport key to under a master key variant. On the control statement, you use the TRANSKEY keyword to specify the transport key that enciphers the key.

You can import or export a new version of a key that is encrypted under the current version of the same key. You can do this by specifying the same key label in the TRANSKEY keyword as in the LABEL or RANGE keyword on an UPDATE control statement.

Your site can generate keys for key exchange between two other sites. These sites do not need to know the clear value of the keys used for this communication. KGUP generates control statements that you send to the sites. Then the sites' KGUPs establish the keys they need for key exchange.

To do this procedure, submit an ADD or UPDATE control statement with two TRANSKEY key labels. The first TRANSKEY label identifies the transport key that is valid between your site and the first recipient site. The second TRANSKEY label identifies the transport key that is valid between your site and the second recipient site. KGUP generates of a pair of control statements to create the complementary pair of keys that are needed at the two sites.

Note: You cannot specify two transport keys that were installed without control vectors. For more information about control vectors, see the description of the NOCV keyword.

The TRANSKEY keyword and the CLEAR keyword are mutually exclusive.

If you have specified a key type of NULL for the TYPE keyword, you cannot use the TRANSKEY keyword.

CLEAR

This keyword indicates that either:

- You are supplying an unencrypted key value with the KEY keyword.
- KGUP should create a control statement that generates an unencrypted complementary key value.

You can supply either encrypted or unencrypted key values to KGUP with the KEY keyword. On the control statement to supply the unencrypted key, you specify the CLEAR keyword.

When KGUP generates a key, KGUP enciphers the key under a master key variant. KGUP may also generate a key value to be used to create the key's complement. KGUP can create the complementary key value in unencrypted

form. To generate an unencrypted complementary key value, you specify the CLEAR keyword. Your ICSF system must be in special secure mode to use this keyword.

The CLEAR keyword and the TRANSKEY keyword are mutually exclusive. You cannot use the CLEAR keyword on a control statement when you use the TRANSKEY keyword. You cannot use the CLEAR keyword if you specify a NULL key for the TYPE keyword.

NOCV

To exchange keys with systems that do not recognize transport key variants, ICSF provides a way to by-pass transport key variant processing. KGUP or an application program encrypts a key under the transport key itself not under the transport key variant. This is called NOCV processing.

The NOCV keyword indicates that the key that is generated or imported is a transport key to use in NOCV processing. The transport key has the NOCV flag set in the key control information when stored in the CKDS.

Note: To create keys for NOCV processing, NOCV-Enablement keys must exist. For a description of how to create NOCV-Enablement keys, see “Initializing the CKDS at First-Time Startup” on page 68.

The NOCV keyword is only valid for generating transport keys. The keyword is not valid if you specify the TRANSKEY keyword with two transport key labels.

LENGTH

LENGTH indicates the length of a DATA key to generate. LENGTH(8) generates a single-length key. LENGTH(16) generates a double-length key, and LENGTH(24) generates a triple-length key. LENGTH(24) applies only to DATA keys. It is no longer necessary to specify LENGTH when generating MAC or MACVER keys, as these keys are now single-length only. If a LENGTH is specified for MAC or MACVER keys, however, it must be LENGTH(8). Similarly, it is not necessary to specify LENGTH when generating DATAM or DATAMV keys. However, if a LENGTH is specified when generating DATAM or DATAMV keys, it must be LENGTH(16).

For double-length key types, LENGTH(8) or SINGLE in an ADD or UPDATE statement causes KGUP to generate a double-length key with both halves the same. On the KGUP panel, you can achieve this by specifying 8 in the LENGTH field for a double-length key type.

In either case, LENGTH is used only for generating keys. If you are specifying clear or encrypted key parts, do not use the LENGTH keyword (and do not fill in a value for LENGTH on the KGUP panel).

The LENGTH keyword and KEY keyword are mutually exclusive. The LENGTH keyword is valid when you create control statements to generate DATA, MAC, MACVER, DATAM, or DATAMV keys. The LENGTH keyword is not necessary when generating MAC, MACVER, DATAM, or DATAMV keys. KGUP ignores the LENGTH keyword for DATAXLAT keys, which KGUP automatically generates as single-length keys.

CDMF

This keyword indicates that KGUP should generate, or you supply, a key marked for use in conjunction with the CDMF algorithm. A CDMF key, like a DES key, is a single-length (64-bit) key. This marks the key token that contains the CDMF key so that the use of the CDMF algorithm is automatically triggered. KGUP encrypts the key with a master key variant and stores the key in the

CKDS. You can specify CDMF only with TYPE keywords DATA, IMPORTER, or EXPORTER. If you specify DATA, KGUP marks the DATA key for use in the CDMF algorithm, and LENGTH(8) is the only valid specification for LENGTH. If you specify IMPORTER or EXPORTER, KGUP marks the key to indicate that it will transport DATA keys that are intended for use in the CDMF algorithm.

The CDMF and DES keywords are mutually exclusive.

DES

This keyword indicates that KGUP should generate, or you supply, a key marked for use with the DES algorithm. A DES key is a single-length (64-bit) key. This marks the key token that contains the DES key so that the use of the DES algorithm is automatically triggered. KGUP encrypts the key with a master key variant and stores the key in the CKDS. You can specify DES only with TYPE keywords DATA, IMPORTER, or EXPORTER. If you specify DATA, KGUP marks the DATA key for use in the DES algorithm. If you specify IMPORTER or EXPORTER, KGUP marks the key to indicate that it will transport DATA keys that are intended for use in the DES algorithm.

The DES and CDMF keywords are mutually exclusive.

KEY (key-value[,ikey-value])

This keyword allows you to supply KGUP with a key value. KGUP can use this key value to add a key or update a key entry.

This keyword is required when you specify either DATAMV, MACVER, or PINVER for the TYPE keyword. Because KGUP cannot generate PIN verification or MAC verification keys in operational form, you must always supply values for these types of keys.

When you enter a double-length key, you enter the key in two parts. Each key part consists of exactly 16 characters that represent 8 hexadecimal values. These parts are:

- The *key-value*, the first part, or left half of the key
- The *ikey-value*, the second part, or right key half is also known as the intermediate key value

When you are adding a DATA key, you can add the key in one, two, or three parts.

KGUP links the two values to form a full double-length key.

To supply an effectively single-length key to KGUP, only specify one key-value on the KEY keyword. KGUP duplicates this value to create an identical intermediate key value. KGUP concatenates these two identical values, and then stores and uses the key as if the key was double-length. If you do not specify this keyword, KGUP generates the key value for you.

Because DATAXLAT is a single-length key, you cannot supply a second key value for this key type. If you supply an ikey-value for a DATAXLAT key, KGUP discontinues processing the control statement and issues an error message.

For double-length keys, when you use the TRANSKEY keyword with the KEY keyword, the transport key you specify is the importer key that encrypts the key value. If you supply only one key value for a double-length key and also specify TRANSKEY, the TRANSKEY must be an NOCV importer. You cannot use the RANGE keyword or the LENGTH keyword the RANGE or LENGTH keyword with this keyword.

Attention: NOCV processing takes place automatically when KGUP or an application specifies the use of a transport key that was generated by KGUP with a NOCV keyword specified.

The use of NOCV processing eliminates the ability of the system that generates the key to determine the use of the key on a receiving system. Therefore, access to these keys should be strictly controlled. For a description of security considerations, see *z/OS ICSF System Programmer's Guide*.

Using the ADD and UPDATE Control Statements for Key Management and Distribution Functions

You use the ADD and UPDATE control statements to run KGUP for functions that involve key generation, maintenance, and distribution. For ADD and UPDATE control statements, KGUP either imports a key value that you supply or generates a key value. This section describes the combinations of control statement keywords you use to perform these functions. Table 5 shows the keyword combinations permitted on ADD and UPDATE control statements.

Table 5. Keyword Combinations Permitted in ADD and UPDATE Control Statements

Control Statement	LABEL or RANGE	TYPE	OUTTYPE	TRANSKEY or CLEAR	NOCV	CDMF or DES	LENGTH or KEY
ADD	Yes	Yes	Yes[1]	Yes[2]	Yes[3]	Yes[4]	Yes[1]
UPDATE	Yes	Yes	Yes[1]	Yes[2]	Yes[3]	Yes[4]	Yes[1]

Notes:

1. OUTTYPE can be used with either TRANSKEY or CLEAR but is mutually exclusive with KEY.
2. TRANSKEY is not valid when TYPE is NULL.
3. NOCV is not valid when TRANSKEY is specified with two key labels.
4. The DES or CDMF keywords can only be used with TYPE of DATA, EXPORTER, or IMPORTER.

To Import Keys

You use an ADD or UPDATE control statement to supply a value to KGUP. The program receives the value, enciphers the value under a master key variant, and places the value in a CKDS entry. The value that you supply may be in clear form or it may be encrypted under a transport key. The statement that contains the value may be sent from another system. The other system sends the value to create a key on your system. This key is the complement of a key that was generated on the other system.

You can supply a transport key value to KGUP from a system that does not use control vectors. You use the key for key exchange with that system. KGUP places the key into the CKDS with an indication that the key is to be used without control vectors.

Note: If you are sharing a CKDS between OS/390 ICSF and any release of ICSF/MVS Version 1, you should use caution. New key types introduced in ICSF Version 2 Release 1 (the level of ICSF in OS/390 V2R5) and the CDMF/DES key token markings will result in an error if used on the ICSF/MVS Version 1 product.

Import a Clear Key Value: You can supply a clear key value on a control statement for KGUP to import.

The following statements show the syntax when you supply a clear key value to KGUP.

Note: For these control statements, your system should be in special secure mode.

When you supply a single-length, clear key value:

```
ADD or UPDATE  LABEL(label)    TYPE(data,dataxlat,exporter,importer,
mac,macver, or any PIN key)  CLEAR  KEY(key-value)
```

When you supply a double-length, clear key value:

```
ADD or UPDATE  LABEL(label)  TYPE(data,datam,datamv,exporter,importer,
or any PIN key)  CLEAR  KEY(key-value,ikey-value)
```

When you supply a triple-length, clear key value:

```
ADD or UPDATE  LABEL(label)  TYPE(data)
CLEAR  KEY(key-value, key-value, key-value)
```

When you supply a single-length clear key value and you use the key to exchange keys with a cryptographic product that does not use control vectors or double-length keys:

```
ADD or UPDATE  LABEL(label)  TYPE(exporter or importer)
CLEAR  KEY(key-value)  NOCV
```

When you supply a double-length, clear key value, and you use the key to exchange keys with a cryptographic product that does not use control vectors:

```
ADD or UPDATE  LABEL(label)  TYPE(exporter or importer)
CLEAR  KEY(key-value,ikey-value)  NOCV
```

When you supply a single-length, clear key value, and you use the key to exchange data with a CDMF-only system:

```
ADD or UPDATE  LABEL(label)  TYPE(data, exporter or importer)
CLEAR  KEY(key-value)  CDMF
```

Import an Encrypted Key Value: When you supply KGUP with an encrypted key value, the value is encrypted under a transport key. The transport key is one key in a complementary key pair that you share with another system. When the other system's KGUP generated a key, the program also stored a control statement to use to create the complementary key. The other system sends the control statement to your system. You can use the statement to supply an encrypted key value to KGUP to create the key.

The following statements show the syntax when you supply an encrypted key value to KGUP.

When you supply a single-length, encrypted key value:

```
ADD or UPDATE  LABEL(label)  TYPE(data,dataxlat,exporter,importer,
mac,macver, or any PIN key) TRANSKEY(key-label 1)  KEY(key-value)
```

When you supply a double-length, encrypted key value:

```
ADD or UPDATE  LABEL(label)  TYPE(data,datam,datamv,exporter,importer,
or any PIN key) TRANSKEY(key-label 1)  KEY(key-value,ikey-value)
```

When you supply a triple-length, encrypted key value:

```
ADD or UPDATE  LABEL(label)  TYPE(data)
TRANSKEY(key-label 1)  KEY(key-value, key-value, key-value)
```

When you supply a single-length, encrypted key value, and you are using the key to exchange keys with a cryptographic product that does not use control vectors or double-length keys:

```
ADD or UPDATE  LABEL(label)  TYPE(exporter or importer)
TRANSKEY(key-label 1)  KEY(key-value)  NOCV
```

Note: Single-length keys with replicated key parts can be brought in under a TRANSKEY only if the TRANSKEY is an NOCV IMPORTER.

When you supply a double-length encrypted key value and you will use the key to exchange keys with a cryptographic product that does not use control vectors:

```
ADD or UPDATE  LABEL(label)  TYPE(exporter or importer)
TRANSKEY(key-label 1)  KEY(key-value,ikey-value)  NOCV
```

When you supply a single-length, encrypted key value, and you are using the key to exchange data or keys with a CDMF system:

```
ADD or UPDATE  LABEL(label)  TYPE(data, exporter, or importer)
TRANSKEY(key-label 1)  KEY(key-value)  CDMF
```

To Generate Keys

You use an ADD or UPDATE control statement to have KGUP generate a key value to place in the CKDS. The program generates the value, enciphers the value under a master key variant, and places the value in the CKDS. When KGUP generates a key, the program may also store information to create the key's complement in a data set.

You can have KGUP generate a transport key that you use to send or receive keys from a system that does not use control vectors. KGUP places the key into the CKDS with an indication that the key is to be used without control vectors.

Generate an Importer Key For File Encryption: You can have KGUP create an importer key without having KGUP store information about the complement of the key. You do not use the importer key in key exchange with another system. You use the importer key to encrypt a data-encrypting key that you use to encrypt data in a file on your system. You can store the data-encrypting key with the file, because the data-encrypting key is encrypted under the importer key.

The following statements show the syntax when you generate an importer key to use in file encryption on a system:

When you generate a single-length key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(importer)  SINGLE
```

When you generate a double-length key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(importer)
```

When you generate a key to import a data-encrypting intended for use in the CDMF algorithm:

```
ADD or UPDATE  LABEL(label)  TYPE(importer)  CDMF
```

Generate a Complementary, Clear Key Value: You can have KGUP store complementary key information when KGUP generates a key. This information includes the key value. You send the information to another system which uses the information to generate the complementary key. KGUP stores the key value to

create the complementary key in either clear or encrypted form. KGUP stores information both in and not in the form of a control statement.

The following statements show the syntax when you have KGUP store the complementary key value in clear form.

Note: For these control statements, your system should be in special secure mode.

When you generate a single-length, transport or PIN clear key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(exporter,importer,ipinenc,opinenc, or pingen) CLEAR  SINGLE
```

When you generate a single-length, DATA clear key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data) LENGTH(8) CLEAR
```

When you generate a double-length, DATA clear key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data) LENGTH(16) CLEAR
```

When you generate a triple-length, DATA clear key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data) LENGTH(24) CLEAR
```

When you generate a single-length, MAC clear key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(mac) OUTTYPE(mac or macver) CLEAR
```

When you generate a double-length, DATAM clear key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(datam) LENGTH(16) OUTTYPE(datam or datamv) CLEAR
```

When you generate a single-length, DATAXLAT clear key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(dataxlat) CLEAR
```

When you generate a double-length, clear key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(exporter,importer,ipinenc,opinenc, or pingen) CLEAR
```

When you generate a single-length, clear key value, and you are using the key to exchange keys with a cryptographic product that does not use control vectors:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(exporter or importer)  CLEAR  NOCV  SINGLE
```

When you generate a double-length, clear key value, and you are using the key to exchange keys with a cryptographic product that does not use control vectors:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data) LENGTH(16) CLEAR NOCV
```

When you generate a triple-length, clear key value, and you are using the key to exchange keys with a cryptographic product that does not use control vectors:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data) LENGTH(24)  CLEAR  NOCV
```

When you generate a double-length, clear key value, and you are using the key to exchange keys with a cryptographic product that does not use control vectors:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(exporter or importer)  CLEAR  NOCV
```

When you generate a clear key value to transport data-encrypting keys for use in the DES algorithm:

```
ADD or UPDATE LABEL(label) TYPE(exporter or importer) CLEAR DES
```

Generate a Complementary, Encrypted Key Value: KGUP encrypts the complementary key value under the exporter key that you specify.

The following statements show the syntax when you have KGUP generate the complementary key value in encrypted form.

When you generate a single-length, transport or PIN encrypted key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(exporter,importer,ipinenc,opinenc, or pingen)
TRANSKEY(key-label 1)  SINGLE
```

When you generate a single-length, DATA encrypted key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data) OUTTYPE(data or dataxlat) TRANSKEY(key-label 1)
```

When you generate a single-length, MAC encrypted key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(mac) OUTTYPE(mac or macver) TRANSKEY(key-label 1)
```

When you generate a single-length, DATAXLAT encrypted key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(dataxlat) TRANSKEY(key-label 1)
```

When you generate a double-length, encrypted key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(exporter,importer,ipinenc,opinenc, or pingen)  TRANSKEY(key-label 1)
```

When you generate a double-length DATA encrypted key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data or datam)  LENGTH(16) TRANSKEY(key-label 1)
```

When you generate a double-length DATAM encrypted key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(datam)  TRANSKEY(key-label 1)
```

When you generate a triple-length DATA encrypted key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data)  LENGTH(24) TRANSKEY(key-label 1)
```

When you generate a single-length, encrypted key value, and you are using the key to exchange keys with a cryptographic product that does not use control vectors:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(exporter or importer)  TRANSKEY(key-label 1)  SINGLE  NOCV
```

When you generate a double-length, encrypted key value, and you are using the key to exchange keys with a cryptographic product that does not use control vectors.

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(exporter or importer)  TRANSKEY(key-label 1)  NOCV
```

When you generate a double-length, encrypted key value that is marked to indicate that it will transport data-encrypting keys that are intended for use in the CDMF algorithm:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(exporter or importer) TRANSKEY(key-label 1) CDMF
```

Generate a Complementary Key Pair For Other Systems: You can also use KGUP as a key distribution center. KGUP generates a pair of complementary key values that are both used on other systems. KGUP encrypts the values under appropriate variants of two different exporter key-encrypting keys. KGUP does not alter your system's CKDS. The program stores two control statements each containing one of the keys that are encrypted under a transport key. You send the statements to two other sites which can create the keys and use the keys to exchange keys.

The following statements show the syntax when you have KGUP generate a pair of complementary key values to send to other systems.

When you generate single-length transport or PIN key values:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(exporter,importer,ipinenc,opinenc, or pingen)
TRANSKEY(key-label 1,key-label 2)  SINGLE
```

When you generate single-length DATA key values:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data) OUTTYPE(data or dataxlat) TRANSKEY(key-label 1,
key-label 2)
```

When you generate double-length DATA key values:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data) LENGTH(16) TRANSKEY(key-label 1,key-label 2)
```

When you generate triple-length DATA key values:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data) LENGTH(24) TRANSKEY(key-label 1,key-label 2)
```

When you generate single-length MAC key values:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(mac) OUTTYPE(mac or macver) TRANSKEY(key-label 1,
key-label 2)
```

When you generate double-length DATAM key values:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(datam) OUTTYPE(datam or datamv)
TRANSKEY(key-label 1,key-label 2)
```

When you generate single-length DATAXLAT key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(dataxlat) TRANSKEY(key-label 1,key-label 2)
```

When you generate a double-length key value:

```
ADD or UPDATE  LABEL(label) or RANGE(start-label,end-label)
TYPE(exporter,importer,ipinenc,opinenc, or pingen)
TRANSKEY(key-label 1,key-label2)
```

To Create NULL Keys

You can use KGUP to create an initial record in the CKDS. To do this, you create an ADD control statement with a key TYPE of NULL. Once you have created this key record, you can use the Key Record Write callable service to place a key value in the record.

If you are generating a large number of keys, you will get better performance if you create the NULL key records with KGUP. This is preferrable to using the Key_Record_Create callable service.

Create NULL Key Records: You can use KGUP to create a single NULL key record or a range of NULL key records. The following statement shows the syntax you use:

```
ADD  LABEL(label) or RANGE(start-label,end-label)  TYPE(null)
```

Syntax of the RENAME Control Statement

The RENAME control statement changes the label of a key entry in the CKDS. KGUP does not change any other information in the entry.

The RENAME control statement has the following syntax:

```
RENAME

   LABEL(old-label,new-label)

   TYPE(key-type)
```

Figure 65. RENAME Control Statement Syntax

LABEL(old-label,new-label)
This keyword specifies the labels of the CKDS entries that you want KGUP to process. For the general rules about key label conventions and uniqueness, see “General Rules for CKDS Records” on page 100.

First you specify the old label which is the current label in the CKDS that KGUP changes. Then you specify the new label to replace the old label.

TYPE(key-type)
Because you can use the same label in entries with different key types, this keyword specifies the type of key for the old entry and the new entry.

Syntax of the DELETE Control Statement

DELETE control statements instruct KGUP to remove key entries from the CKDS.

The DELETE control statement has the following syntax:

```
DELETE

   {LABEL(label1[,...,label64]) | RANGE(start-label,end-label)}

   TYPE(key-type)
```

Figure 66. DELETE Control Statement Syntax

LABEL (label1[,...,label64])
This keyword defines the names of the key entries for KGUP to delete from the CKDS. KGUP deletes a separate entry for each label.

You must specify at least one key label, and you can specify up to 64 labels with the LABEL keyword. For the general rules about key label conventions and uniqueness, see “General Rules for CKDS Records” on page 100.

On a KGUP control statement, you must specify either the LABEL or RANGE keyword.

RANGE (start-label, end-label)
This keyword defines the range of the multiple labels that you want KGUP to delete from the CKDS.

The label consists of between 2 and 64 characters that are divided as follows:

- The first 1 to 63 characters are the label base. These characters must be identical on both the start-label and end-label and are repeated for each label in the range. For the general rules about key label conventions and uniqueness, see “General Rules for CKDS Records” on page 100.
- The last 1 to 4 characters form the suffix. The number of digits in the start-label and end-label must be the same, and the characters must all be numeric. These numeric characters establish the range of labels KGUP creates. The start-label numeric value must be less than the end-label numeric value.

TYPE(key-type)
Because you can use the same label in entries with different key types, this keyword specifies the type of key that is being deleted.

To Delete Keys
You can use a KGUP control statement to remove a key or a range of keys from the CKDS. The following statement shows the syntax when you delete keys from the CKDS:

```
DELETE  LABEL(label) or RANGE(start-label,end-label)
TYPE(data,dataxlat,exporter,importer,ipinenc,mac,macver,
null,opinenc,pingen, or pinver)
```

Syntax of the SET Control Statement
The SET control statement specifies data you want KGUP to pass to the installation-defined exit routine for processing.

The SET control statement has the following syntax:

```
SET

   INSTDATA(data-value)
```

Figure 67. SET Control Statement Syntax

INSTDATA(data-value)
This keyword specifies the data KGUP sends to the KGUP exit routine while processing control statements.

During a KGUP job, the data you specify with the INSTDATA keyword is held and sent to the exit routine each time the exit is entered for control statement processing. The same information is sent until KGUP encounters another SET

control statement. The data you specified in this SET control statement replaces the data you specified in the previous SET control statement.

A KGUP exit routine performs different operations that depend on the data that is sent and the time of the call. A KGUP exit routine can change the data you send the exit and send the changed data to the user area of a key entry in the CKDS. The user area of a key entry can contain any information that you choose to store in the area.

For more information about the KGUP exit routine, see *z/OS ICSF System Programmer's Guide*.

The maximum length of the character string that you can specify to an exit routine is 52 bytes. If you use blanks or special characters within the string, then you must delimit the entire string with single quotes ('). These quotes are not included as part of the 52-byte string.

Examples of Control Statements

Example 1: ADD Control Statement

This example shows a control statement that specifies that KGUP add an entry to the CKDS.

```
ADD  TYPE(IMPORTER) LABEL(DASDOCT93401E)
```

KGUP checks that an entry labeled `DASDOCT93401E` with a keytype of importer does not already exist in the CKDS. It also checks that there are no DATA, DATAXLAT, DATAM, DATAMV, MAC, MACVER, or NULL key entries with that label. Each of these keys requires a unique label. If the key entry already exists, KGUP stops processing the control statement.

If the entry does not exist, KGUP creates the entry with a label of `DASDOCT93401E` and type of IMPORTER. KGUP generates a double-length key and encrypts the key under the master key variant for an importer key. KGUP places the key in the entry.

Note: Because neither the TRANSKEY nor CLEAR keyword is specified, KGUP does not create a complementary key. You cannot use this key to communicate with another system. You can, however, use the key to encipher a key stored with data in a file. IMPORTER, DATA, DATAM, and MAC are the only key types that do not require either the TRANSKEY or CLEAR keyword specified.

Example 2: ADD Control Statement with CLEAR Keyword

This example shows a control statement that specifies that KGUP add an entry to the CKDS. Because the CLEAR keyword is specified, KGUP processes only this control statement if ICSF is in special secure mode.

```
ADD  TYPE(EXPORTER)  LABEL(ATMBRANCH5MO001)  CLEAR
```

KGUP checks that an entry with the label `ATMBRANCH5MO001` with the type EXPORTER does not already exist in the CKDS. It also checks that there are no DATA, DATAXLAT, DATAM, DATAMV, MAC, MACVER, or NULL key entries with that label. Each of these keys requires a unique label. If the entry already exists, KGUP stops processing the control statement.

If the entry does not exist, KGUP creates the entry for the label specified and the type exporter. KGUP generates a double-length key, encrypts the key under the master key variant for an exporter key, and places the key in the entry.

KGUP stores information to the key output data set. You can send the information to another system that does not use KGUP. The other system uses the information to create the complements of the keys you created. The information contains the clear key value and specifies the key type as importer.

KGUP also stores a control statement to the control statement output data set. You can send this control statement to another system. The other system's KGUP uses the control statement to create a key that complements the key that you just created.

For example, the control statement would be in the following format:

```
ADD  TYPE(IMPORTER)  LABEL(ATMBRANCH5MO001)  CLEAR,
KEY(6709E5593933DA00,9099937DDE93A944)
```

The key value is the clear key value of the key created. The type of key is the complement of the type of key created.

Note: The key in the above example is a mixed parity key. KGUP imports mixed parity keys, but issues a warning message.

Example 3: ADD Control Statement with one TRANSKEY Keyword

This example shows a control statement that specifies that KGUP add an entry to the CKDS. Because the TRANSKEY keyword is specified, KGUP also creates a control statement that another installation uses to create the complement of the key for PIN exchange.

```
ADD  TYPE(IPINENC)  LABEL(LOCTOJWL.JUNE99)  TRANSKEY(SENDJWL.JUNE99)
```

KGUP checks that an entry with the label `LOCTOJWL.JUNE99` for an input PIN-encrypting key does not already exist in the CKDS. It also checks that there are no DATA, DATAXLAT, DATAM, DATAMV, MAC, MACVER, or NULL key entries with that label. Each of these keys requires a unique label. If the entry already exists, KGUP stops processing the control statement.

If the entry does not exist, KGUP creates the entry with a label of `LOCTOJWL.JUNE99` and type of IPINENC. KGUP generates a double-length key. KGUP encrypts the key under the master key variant for an input PIN-encrypting key and places the key in the entry.

KGUP stores information to the key output data set. You can send the information to another system that does not use KGUP. The other system uses the information to create the complement of the key you created. The information contains the key in exportable form. The key is encrypted under the exporter key, labelled `SENDJWL.JUNE99`, that was specified by the TRANSKEY keyword. The information specifies the key type as output PIN-encrypting key (OPINENC).

Note: If `SENDJWL.JUNE99` is an NOCV exporter, the exportable OPINENC key is encrypted without a control vector.

KGUP stores a control statement to the control statement output data set. You can send the control statement to another system. The other system's KGUP uses the statement to create a key that complements the key that you created.

For example, the control statement would be in the following format:

```
ADD  TYPE(OPINENC)  LABEL(LOCTOJWL.JUNE99)  TRANSKEY(SENDJWL.JUNE99),
KEY(6709E5593933DA00,9099937DDE93A944)
```

The key value is the encrypted value of the key that KGUP created. The key is encrypted under the exporter key, labeled SENDJWL.JUNE99, which was the transport key label that was specified on the original control statement. The type of key is the complement of the type of key it created.

Example 4: ADD Control Statement with two TRANSKEY Keywords

This example shows a control statement specifying that KGUP create keys for key exchange between two other sites.

```
ADD  TYPE(EXPORTER)  LABEL(JWL@SSIJUNE99)
TRANSKEY(SENDTOJWLJUNE99,SENDTOSIIJUNE99)
```

KGUP generates a key value and encrypts the value under the variants of the exporter key-encrypting keys that are specified by the TRANSKEY keyword. KGUP does not alter the CKDS in any way.

KGUP stores the following two control statements to the control statement output data set:

```
ADD  TYPE(EXPORTER)  LABEL(JWL@SSIJUNE99)  TRANSKEY(SENDTOJWLJUNE99),
KEY(4542E37B570033AD,3C00F6850A99E11B)

ADD  TYPE(IMPORTER)  LABEL(JWL@SSIJUNE99)  TRANSKEY(SENDTOSIIJUNE99),
KEY(6709E5993933DA00,1449A3D9ED0A1586)
```

The control statements create keys that complement each other. You send the statements to two sites that want to exchange keys. The receiving sites process the statements to create a complementary pair of transport keys.

KGUP also stores information to create the keys in the key output data set.

Example 5: ADD Control Statement with a Range of NULL Keys

This example shows a control statement that creates a range of empty key records in a CKDS. Once the key labels exist, you can enter key types and key values for these records in several ways. One method is to use KGUP to create UPDATE control statements. Another method is to write application programs that use the Key_Record_Write callable service to add key types and key values to the existing empty key records.

```
ADD  TYPE(NULL)  RANGE(BRANCH5M0001,BRANCH5M0025)
```

KGUP checks for any entries with labels between BRANCH5M001 and BRANCH5M0025 in the CKDS. If any entries in this range already exist, KGUP processes the control statement up to the point where a duplicate label is found. It then stops processing the control statement and issues error messages.

If no entries exist, KGUP creates a range of 25 sequentially-numbered key records and adds them to the CKDS.

Example 6: ADD Control Statement with OUTTYPE and TRANSKEY Keywords

This example shows a control statement that specifies that KGUP add an entry with the key type of DATA to the CKDS. The TRANSKEY keyword instructs KGUP to create a control statement for an intermediate node to use to create the complement DATAXLAT key for intermediate node data translation.

```
ADD  LABEL(DATAKEY.TO.TRANSLATION)  TYPE(DATA)  OUTTYPE(DATAXLAT)
TRANSKEY(TKBRANCH2.INTER)
```

KGUP checks that an entry withthe label DATAKEY.TO.TRANSLATION does not already exist in the CKDS, because DATA keys require unique labels. If the entry already exists, KGUP stops processing the control statement.

If the entry does not exist, KGUP creates the entry with a label of DATAKEY.TO.TRANSLATION and a type of DATA. KGUP then generates a single-length key, encrypts the key under the master key variant for a DATA key, and places the key in the CKDS entry.

KGUP stores information to the key output data set. You can send the information to another system that does not use KGUP. The other system uses the information to create the complement of the key you created. The information contains the key value of the key in exportable form. The key is encrypted under the exporter key, labeled TKBRANCH2.INTER, that was specified by the TRANSKEY keyword. The information specifies the key type as data-translation key (DATAXLAT).

KGUP stores a control statement to the control statement output data set. You can send the control statement to another system. The other system's KGUP uses the statement to create a key that complements the key you created.

For example, the control statement would be in the following format:

```
ADD  TYPE(DATAXLAT)  LABEL(DATAKEY.TO.TRANSLATION)
TRANSKEY(TKBRANCH2.INTER), KEY(2509F2869257BD00)
```

The key value is the encrypted value of the key that KGUP created. The key is encrypted under the exporter key, labelled TKBRANCH2.INTER, which was the transport key label that was specified on the original control statement. The type of key is the complement of the type of key it created.

Example 7: ADD Control Statement for a CDMF Key

This example shows a control statement that specifies that KGUP should add an entry to the CKDS and mark it as a CDMF key.

```
ADD  TYPE(DATA) LABEL(COMKEY26596E) CDMF
```

KGUP checks that an entry with the label COMKEY26596E with a key type of data does not already exist in the CKDS. It also checks that there are no DATAXLAT, DATAM, DATAMV, MAC, MACVER, or NULL key entries with that label (because each of these keys requires a unique label). If the key entry already exists, KGUP stops processing the control statement.

If the entry does not exist, KGUP creates the entry with a label of COMKEY26596E and type of DATA. KGUP marks the key token to indicate that the key is to be used in the CDMF algorithm. KGUP generates a single-length key and encrypts the key under the master key variant for a data key. KGUP places the key in the entry.

Note: Because neither the TRANSKEY nor CLEAR keyword is specified, KGUP does not create a complementary key. You cannot use this key to communicate with another system. You can, however, use the key to encipher a key stored with data in a file. IMPORTER, DATA, DATAM, and MAC are the only key types that do not require either the TRANSKEY or CLEAR keyword specified.

Example 8: UPDATE Control Statement with Key Value and Transkey Keywords

This example shows a control statement that specifies that KGUP import a key value. KGUP places the key value into an entry in the CKDS that already exists.

```
UPDATE  LABEL(PINVBRANCH5M0002)  TYPE(PINVER)  TRANSKEY(TKBRANCH5JUNE99)
KEY(7165865940460A48,2237451B4545718B)
```

The key value on the control statement is encrypted under a transport key that is shared with another system. The label for the transport key is TKBRANCH5JUNE99. KGUP uses the importer key labelled TKBRANCH5JUNE99 to decrypt the key value.

KGUP encrypts the key value under the master key variant for a PIN verification key. KGUP then places the key in a key entry labelled PINVBRANCH5M0002 with the type PINVER in the CKDS.

Example 9: DELETE Control Statement

This example shows a control statement that specifies that KGUP delete an entry from the CKDS.

```
DELETE  LABEL(GENBRANCH2M0003)  TYPE(PINGEN)
```

KGUP deletes the entry with a label of GENBRANCH2M0003 and type of PIN generation key from the CKDS. If KGUP cannot find the entry, KGUP gives you an error message.

Example 10: RENAME Control Statement

This example shows a control statement that specifies that KGUP rename an entry in the CKDS.

```
RENAME  LABEL(JWL@SSIDEC97,JWL@SSIJUNE99)  TYPE(EXPORTER)
```

KGUP checks if an entry with a label of JWL@SSIJUNE99 and a key type of EXPORTER already exists in the CKDS. If the entry does exist, KGUP does not process the control statement. KGUP checks if an entry with the label JWL@SSIDEC97 contains a key type of EXPORTER exists. If the entry exists, KGUP renames the entry JWL@SSIJUNE99.

Example 11: SET Control Statement

This example shows a control statement that specifies that KGUP send certain installation data every time an exit is called during KGUP processing. KGUP sends the data every time an exit is called until KGUP encounters another SET statement or the job stream completes.

```
SET  INSTDATA('This key is valid effective 9/9/99')
```

KGUP sends the installation data each time an installation exit is called during KGUP processing.

Specifying KGUP Data Sets

During key generator utility program (KGUP) processing, you store the information you supply and receive in the following data sets:

- The cryptographic key data set (CKDS) contains key entries that you have KGUP add, update, rename, or delete.
- The control statement input data set contains the control statements that specify the functions you want KGUP to perform.
- The diagnostics data set contains information you can use to check that the control statement succeeded.
- The key output data set contains information that another system uses to create keys that are complements of keys on your system.
- The control statement data set contains control statements that another system uses to create keys that are complements of keys on your system.

You specify the names of the data sets in the job control language to submit the job.

The following sections describe the data sets that KGUP accesses or generates in detail.

Cryptographic Key Data Set (CKDS)
This VSAM key sequenced data set contains the cryptographic keys for a particular KGUP job. It has a fixed logical record length (LRECL) of 252 bytes.

Programming Interface information

The records in the CKDS are in the following format:

Key label
(Character length 64 bytes) The key label specified on the control statement.

Key type
(Character length 8 bytes) The key type specified on the control statement.

Creation date
(Character length 8 bytes) The initial date the record was created, in the format `YYYYMMDD`.

Creation time
(Character length 8 bytes) The initial time the record was created, in the format `HHMMSSTH`.

Last update date
(Character length 8 bytes) The most recent date the record was updated, in the format `YYYYMMDD`.

Last update time
(Character length 8 bytes) The most recent time the record was updated, in the format `HHMMSSTH`.

Key token
(Character length 64 bytes) A key token is composed of the key value and control information. The master key encrypts the key value in this field. For a description of format of a key token, see *z/OS ICSF System Programmer's Guide*.

CKDS flag bytes
(Bit length 2 bytes) If bit zero is set to one, the key within the token is a partial key. All the other bits are reserved.

Reserved
(Character length 26 bytes) Reserved. This field contains binary zeros.

Installation Data
(Character length 52 bytes) Using the KGUP exit, conversion program exit, or single-record, single-record, read-write exit, you can place information associated with the key entry into this field.

Authentication code
(Character length 4 bytes) The message authentication code computed on the previous fields of the record using a system key that is a MAC generation key. ICSF uses the code to verify the record when the record is updated.

The first record in the CKDS is a header record. The header record in the CKDS is in the following format:

Key label
(Character length 64 bytes) Binary zeros. This field is not to be used.

Key type
(Character length 8 bytes) Binary zeros. This field is not to be used.

Creation date
(Character length 8 bytes) The initial date the record was created, in the format YYYYMMDD.

Creation time
(Character length 8 bytes) The initial time the record was created, in the format HHMMSSTH.

Last update date
(Character length 8 bytes) The most recent date the record was updated, in the format YYYYMMDD.

Last update time
(Character length 8 bytes) The most recent time the record was updated, in the format HHMMSSTH.

Sequence number
(Character length 2 bytes) Initially binary zero, incremented each time the data set is processed.

CKDS header flag bytes
(Bit length 2 bytes) If bit zero is set to one, the master key verification pattern is valid. If bit one is set to one, the master key authentication pattern is valid. All the other bits are reserved.

Master key verification pattern
(Character length 8 bytes) The system master key verification pattern.

When you initialize the CKDS and master key or change the master key, ICSF calculates a verification pattern and places it into this field. ICSF calculates the verification pattern by using the current master key and the verification algorithm that is described in “Algorithm for Calculating a Verification Pattern” on page 210.

Master key authentication pattern
(Character length 8 bytes) The system master key authentication pattern.

When you initialize the CKDS and master key or change the master key, ICSF calculates an authentication pattern and places it into this field. ICSF calculates the authentication pattern by using the current master key and the authentication pattern algorithm that is described in “Algorithm for Calculating an Authentication Pattern” on page 211.

Whenever you start ICSF, ICSF uses the authentication pattern to verify that the current master key is the master key that enciphers the current CKDS. ICSF fails if the authentication pattern that is stored in the CKDS and the authentication pattern that ICSF calculates at startup do not match.

Installation Data
(Character length 52 bytes) Using the KGUP installation exit, you can place information associated with the key entry into this field.

Authentication code
(Character length 4 bytes) The message authentication code computed on the previous fields of the record using a system key that is a MAC generation key. ICSF creates the code after ICSF creates the system keys at CKDS initialization. ICSF uses the code to verify the CKDS when the CKDS is read.

End of Programming Interface information

In the KGUP job stream, it is defined by the CSFCKDS data definition statement.

Control Statement Input Data Set
This data set contains the control statements that the particular KGUP job processes. For a description of the syntax of these control statements, see "Using KGUP Control Statements" on page 100.

This data set is a physical sequential data set with a fixed logical record length (LRECL) of 80 bytes.

Note: If a control statement adds or updates a key, later control statements in the control statement input data set for that KGUP job use the new or updated key.

In the KGUP job stream, the control statement input data set is defined by the CSFIN data definition statement.

Diagnostics Data Set
This data set contains a copy of each input control statement that is followed by one or more diagnostic messages that were generated for that control statement. It is a physical sequential data set with a fixed logical record length (LRECL) of 133 bytes. It should be fixed with ASA codes. Figure 68 shows an example of a diagnostics data set.

```
KEY GENERATION DIAGNOSTIC REPORT  DATE:1997/9/14 TIME:12:10:15  PAGE 1


 /* THIS IS A KEY USED TO EXPORT KEYS FROM A TO B */
 ADD TYPE(EXPORTER) TRANSKEY(TK1),
  LABEL(ATOB)
 > > > CSFG0321  STATEMENT SUCCESSFULLY PROCESSED.




 /* THIS IS A KEY USED TO IMPORT KEYS FROM B TO A */
 ADD TYPE(IMPORTER) TRANSKEY(TK1),
  LABEL(BTOA)
 > > > CSFG0321  STATEMENT SUCCESSFULLY PROCESSED.
```

Figure 68. Diagnostics Data Set Example

In the KGUP job stream, the data set is defined by the CSFDIAG data definition statement.

Key Output Data Set

This data set contains information about each key KGUP generates, except an importer key used to protect a key that is stored with a file. Each entry contains the key value and the complement key type of the key created. Another system can use this information to create a key that is the complement of the key your system created.

This data set is a physical sequential data set with a fixed logical record length (LRECL) of 208 bytes.

To establish key exchange with a system that does not use KGUP control statements, you can send that system information from this data set. The receiving system can then use this information to create the complement of the key you created. You can print or process this data set after KGUP ends.

KGUP only lists a record for the key if the TRANSKEY or CLEAR keyword was in the control statement. If the TRANSKEY keyword was specified in the output key data set, KGUP lists, for the key type, the complement of the control statement key type. KGUP lists, for the key value, the key encrypted under the transport key as specified by the TRANSKEY keyword.

The encrypted key is in the form of an external key token. An external key token contains the encrypted key value and control information about the key. For example, the token contains the control vector for the key type.

If the CLEAR keyword was specified, in the output key data set KGUP lists, for the key type, the complement of the control statement key type. KGUP lists, for the key value, the clear key value of the key. With this information another system could generate keys that are complements of the keys your system generated. This would permit your system and the other system to exchange keys.

When KGUP generates two complementary keys, each encrypted by a different transport key, KGUP lists a record for each key. The first record contains a key that is encrypted under the first transport key variant and the type that is specified on the control statement. The second record contains a key that is encrypted under the second transport key variant and a type that is the complement of the first key.

The records in the key output data set are in the following format:

Key label
(Character length 64 bytes) The key label specified on the control statement.

Key type
(Character length 8 bytes) The key type specified on the control statement or the complement of that key type if the TRANSKEY keyword was specified.

TRANSKEY label or `CLEAR`
(Character length 64 bytes) Either the key label of a transport key which encrypts the key entry or the character string `CLEAR` (left justified) if the key is unencrypted.

TRANSKEY type
(Character length 8 bytes) The key type of the TRANSKEY, which is always exporter.

Key Token
(Character length 64 bytes) A key token is composed of the key value and control information. The key value in this field is either unencrypted

or encrypted under a transport key. For a description of format of a key token, see *z/OS ICSF System Programmer's Guide*.

In the KGUP job stream, the data set is defined by the CSFKEYS data definition statement.

Control Statement Output Data Set

KGUP produces an output control statement for every key that is generated as a result of an input control statement with the TRANSKEY keyword specified. The output control statement contains the complement key type of the key type that is specified on the input control statement. The value that is output for the KEY keyword is encrypted under the transport key that is specified on the input control statement.

You can edit the output control statements and distribute them to the appropriate sites for input to KGUP at those locations.

The data set is a physical sequential data set with a fixed logical record length (LRECL) of 80 bytes.

One output control statement appears when you have KGUP generate a key value and create an operational and exportable key pair using a transport key.

Two output control statements appear when you have KGUP generate two exportable keys by using two different transport keys. These statements generate complementary keys types. You can send each statement to a different site to establish communication between the two sites.

In the KGUP job stream, the data set is defined by the CSFSTMNT data definition statement.

The specific name of these types of data sets must appear in the job stream that runs KGUP.

Submitting a Job Stream for KGUP

The key generator utility program (KGUP) is an APF-authorized program that runs as a batch job. It requires certain JCL statements to run. Submit the JCL to run KGUP after you create the KGUP control statements and data sets.

The JCL to run KGUP should be in the following format:

```
//KGUPPROC EXEC PGM=CSFKGUP,PARM=('SSM')
//CSFCKDS  DD   DSN=PROD.CKDS,DISP=OLD
//CSFIN    DD   DSN=PROD.KGUPIN.GLOBAL,DISP=OLD
//CSFDIAG  DD   DSN=PROD.DIAG.GLOBAL,DISP=OLD
//CSFKEYS  DD   DSN=PROD.KEYS.GLOBAL,DISP=OLD
//CSFSTMNT DD   DSN=PROD.STMT.GLOBAL,DISP=OLD
//
```

Figure 69. KGUP Job Stream

The EXEC statement specifies the load module name for KGUP. The PARM keyword on the EXEC statement passes information to KGUP. The keyword specifies either:

- NOSSM to indicate that special secure mode must be disabled
- SSM to indicate that special secure mode must be enabled

You must pass the SSM parameter if any KGUP control statements for the KGUP run contain the CLEAR keyword. NOSSM is the default.

If special secure mode is not enabled and you pass the SSM parameter to KGUP, the program ends immediately without processing any KGUP control statements. If you pass the NOSSM parameter and KGUP encounters a control statement with the CLEAR keyword, the job ends immediately.

In the JCL example, the PARM keyword specifies SSM to indicate that special secure mode should be enabled. You specify SSM if any control statement in the control statement input data set, PROD.KGUPIN.GLOBAL, contains the CLEAR keyword.

In the JCL, the data definition (DD) statements name the data sets necessary to input information to KGUP and output information from the program. See "Specifying KGUP Data Sets" on page 119 for a detailed description of these data sets.

Attention: If a KGUP job ends prematurely, results of the job are unpredictable. You should not read that cryptographic key data set into storage for use.

For a description of the KGUP return codes, see the explanation of message CSFG0002, which is in *z/OS ICSF Messages* manual.

Enabling Special Secure Mode

Before you pass the SSM parameter to KGUP in a JCL statement, you need to enable special secure mode processing. You must specify SSM(YES) in the installation options data set

If you use logical partition (LPAR) mode, you also need to enable special secure mode on the Change LPAR Crypto panel from the Hardware Master Console of the server support element. If you have the optional TKE workstation, you can use it to enable and disable special secure mode.

Running KGUP Using the MVS/ESA Batch Local Shared Resource (LSR) Facility

The MVS/ESA batch LSR subsystem improves performance for random access file processing by reducing the number of inputs and outputs to VSAM data sets. Batch LSR allows a program to use local shared resources rather than non-shared resources. For information about the batch LSR subsystem, see *MVS Batch Local Shared Resources* manual.

VSAM provides a deferred write option on VSAM ACB processing when a program uses shared resources. For more information about VSAM processing, see *MVS/DFP Managing VSAM Data Sets* and the *MVS/ESA Data Administration: Macro Instruction Reference* manual.

By using the batch LSR subsystem and the VSAM deferred write option together, you may improve KGUP performance when adding many keys, for example 10,000 keys, to the CKDS. If your installation has batch LSR and VSAM deferred write, you may improve performance when adding a large number of keys by using different JCL in the KGUP job stream.

Instead of using the following CSFCKDS DD statement:

```
//CSFCKDS DD DSN=cryptographic-key-data-set-name,DISP=OLD
```

Use the following DD statements:

```
//CSFALT  DD DSN=cryptographic-key-data-set-name,DISP=OLD
//CSFCKDS DD SUBSYS=(BLSR,'DDNAME=CSFALT',
//           'DEFERW=YES')
```

You should specify a large amount of storage for the REGION parameter (for example, REGION=32M) on the JOB or EXEC JCL statement. The rest of the JCL statements to run the KGUP job should be in the format that is shown in Figure 69 on page 124.

Reducing Control Area Splits and Control Interval Splits from a KGUP Run

KGUP processes keys on a disk copy of a CKDS which is a VSAM data set. KGUP uses key-direct update processing to process the keys. To access keys, VSAM uses the key's label as the VSAM key. This means that keys are added to the data set in collating sequence. That is, if two keys named A and B are in the data set, A appears earlier in the data set than B. As a result, adding keys to the data set can cause multiple VSAM control interval splits and control area splits. For example, a split might occur if the data set contains keys A, B, E and you add C (C must be placed between B and E). These splits can leave considerable free space in the data set.

The amount of control area splits and control interval splits in the CKDS affects performance. You may want to periodically use the TSO LISTCAT command to list information about the number of control area splits and control interval splits in a CKDS.

You can help reduce the frequency of control interval and control area splits by ensuring that key generator utility control statements are always in the correct collating sequence, A-Z, 0-9, if possible. When adding keys to a new CKDS, add the key entries in sequential order. Also, after adding new entries to the CKDS, you can reorganize the data set to reduce control area splits and control interval splits. To do this, copy the disk copy of the CKDS into another disk copy using the AMS REPRO command or AMS EXPORT/IMPORT commands. You may want to reorganize the data set after every KGUP run.

Note: If it is practical, you may want to perform the following procedure to reduce control area splits. If you are inserting a large number of keys in the middle of a CKDS, you may want to remove and save all the keys after the place in the data set where you are inserting the keys. In this way, you are adding the keys to the end rather than the middle of the data set. When you finish adding the keys, place the keys that you removed back in the data set.

For a detailed explanation of keyed-direct update processing and a description of what happens when control area and control interval splits occur, refer to *MVS/DFP Managing VSAM Data Sets*.

Refreshing the In-Storage CKDS

ICSF functions access an in-storage copy of the CKDS when the functions reference keys by label. However when you use KGUP, the program makes changes to a disk copy of the CKDS. This situation allows you to maintain the keys in the data set without disturbing current cryptographic operations.

After you update the disk copy, you can use the Refresh option on the Key Administration panel to replace the in-storage copy with the disk copy. For a description of this panel path, see “Refreshing the Current CKDS Using the ICSF Panels” on page 149. Besides using the panels to refresh the in-storage CKDS, you can invoke a utility program to perform the task. Refer to “Refreshing the In-Storage CKDS Using a Utility Program” on page 196 for details.

Using KGUP Panels

The key generator utility program (KGUP) panels help you run KGUP by providing panels to do the following tasks:

- Create KGUP control statements.
- Specify the data sets for KGUP processing.
- Invoke KGUP by submitting job control language (JCL) statements.
- Replace the in-storage copy of the cryptographic key data set (CKDS) with the disk copy that KGUP processing changed.

Using the panels, you can perform the tasks to use KGUP to generate or receive keys for PIN and key distribution and to maintain the CKDS.

To access the KGUP panels, select option 8 ***rev bars ***, KGUP, on the Primary Menu panel as shown in Figure 70.

```
CSF@PRIM ----- Integrated Cryptographic Service Facility  ---------
OPTION ===> 8

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions
```

Figure 70. Selecting the KGUP Option on the Primary Menu Panel

The Key Administration panel appears. See Figure 71 on page 128.

```
CSFSAM00 ----------------- ICSF - Key Administration ---------------
OPTION ===>


   Enter the number of the desired option.

   1  Create         - Create key generator control statements

   2  Dataset        - Specify datasets for processing

   3  Submit         - Invoke Key Generator Utility Program (KGUP)

   4  Refresh        - Activate an existing cryptographic key dataset


   Press ENTER to go to the selected option
   Press END   to exit to the previous panel
```

Figure 71. Key Administration Panel

This panel allows you to access panels to perform the tasks to run KGUP. The following sections describe the KGUP tasks.

Creating KGUP Control Statements Using the ICSF Panels

You create the control statements to specify the functions you want KGUP to perform. When you create the control statements, ICSF stores the statements in the control statement input data set.

After you create the control statements, do one of the following procedures:

- Process the control statements by running KGUP.
- Do not process the control statements and just save the statements in the data set. Then at another time you can access the data set to add more control statements and submit the data set for KGUP processing.

To create the KGUP control statements:

1. Select option 1, Create, on the Key Administration panel, as shown in Figure 72, and press ENTER.

```
CSFSAM00 ----------------- ICSF - Key Administration ---------------
OPTION ===> 1


   Enter the number of the desired option.

   1  Create         - Create key generator control statements

   2  Dataset        - Specify datasets for processing

   3  Submit         - Invoke Key Generator Utility Program (KGUP)

   4  Refresh        - Activate an existing cryptographic key dataset
```

Figure 72. Selecting the Create Option on the Key Administration Panel

   The KGUP Control Statement Data Set Specification panel appears. See Figure 73 on page 129.

```
CSFSAE10 - ICSF - KGUP Control Statement Data Set Specification ----
COMMAND ===>

Enter control statement input data set (DDNAME = CSFIN)

   Data Set Name ===> _____________________________________________________
   Volume Serial ===> ______  (if uncataloged)


   Press ENTER to open or create and open specified data set
   Press END   to exit to the previous panel
```

Figure 73. KGUP Control Statement Data Set Specification Panel

2. Enter the name of the data set that you want to contain the control statements for KGUP processing.
   a. For partitioned data sets, specify a member name as part of the data set name.
   b. If the data set is not cataloged, you must also specify the volume serial for the data set in the Volume Serial field. This volume serial allows ICSF to access the correct volume when ICSF opens the data set.

   Note: If you specify NOPREFIX in your TSO profile, so data sets are not automatically prefixed with your userid, you must specify the fully qualified data set name within apostrophes. If you specify PREFIX without a valid prefix, your TSO userid becomes the prefix.

   Depending on your requirements, there are several options to choose from when entering the data set name. Refer to Table 6 for a list of these options and the steps to follow for each.

Table 6. Data Set Name Options

Option	Steps
To have KGUP append the control statements to an existing data set when you know the data set name and the member name	a. Specify the data set name and member name of the existing data set and press ENTER. The KGUP Control Statement Menu appears. See Figure 77 on page 132. The new control statements will be appended after any existing control statements in the data set.

Table 6. Data Set Name Options (continued)

Option	Steps
To have KGUP append the control statements to an existing data set when you know the data set name but not the member name	a. Specify the data set name of the existing data set and press ENTER. If the partitioned data set is not empty, the Member Selection List appears. See Figure 75 on page 131. b. On the Member Selection List panel: • To select a member that already exists, place an `s` to the left of the member name in the list and press ENTER. For example, in Figure 75 on page 131 SHIFT2 is selected so the data set LARSON.CSFIN.TESTDS1P(SHIFT2) becomes the input control statement data set. • To locate a member on the selection list, type an `l` (the lowercase letter L) and the member name on the command line and press ENTER. The list moves so the member appears on the top line of the list and the cursor appears to the left of the member. • To create a new member, type `s` and the new member name on the command line and press ENTER. The KGUP Control Statement Menu appears. See Figure 77 on page 132. The new control statements will be appended after any existing control statements in the data set.
To have KGUP create a new data set	a. Specify a name for the new data set and press ENTER. The Allocation panel appears. See Figure 76 on page 131. b. Enter the necessary information to allocate a new data set and press ENTER. The KGUP Control Statement Menu appears. See Figure 77 on page 132. The new control statements will be stored in the new data set.

Figure 74 shows an example of the KGUP Control Statement Data Set Specification panel with the partitioned data set CSFIN.TESTDS1P and a member name of TEST1.

```
CSFSAE10 - ICSF - KGUP Control Statement Data Set Specification ----
COMMAND ===>

Enter control statement input data set (DDNAME = CSFIN)

   Data Set Name ===> CSFIN.TESTDS1P(test1)_____________________________
   Volume Serial ===> ______  (if uncataloged)


   Press ENTER to open or create and open specified data set
   Press END   to exit to the previous panel
```

Figure 74. Entering a Data Set Name on the KGUP Control Statement Data Set Specification Panel

If the member TEST1 did not previously exist, ICSF creates the member. If the member already exists, ICSF appends the control statements to the end of the data set. <Prefix>.CSFIN.TESTDS1P(test1) becomes the control statement input data set.

If you specify CSFIN.TESTDS1P without the member name, the Member Selection List panel appears. See Figure 75.

```
CSFSAE12 --------- ICSF - Member Selection List --------- ROW 1 To 6 OF 6
COMMAND ===>                                                SCROLL ===> PAGE

Data Set:  LARSON.CSFIN.TESTDS1P
Select one member name only
    NAME          CREATED     CHANGED       SIZE   INIT    MOD    USERID
   PINEX1        95/08/04 96/08/05 10:44      26     24      1   LARSON
   PINEX2        95/08/04 96/07/04 11:23      14     14      0   LARSON
   KEYEX1        95/08/04 96/08/05 12:44       6      6      1   LARSON
s  SHIFT2        95/08/04 96/08/12 10:55     195    137      2   LARSON
   SHIFT3        95/08/04 96/08/05 12:44      48      4      1   LARSON
   TEST1         95/08/04 96/08/05 11:44       4      4      1   LARSON
****************************** BOTTOM OF DATA ********************************
```

Figure 75. Member Selection List Panel

If you specify a new data set name, the Allocation panel appears. See Figure 76.

```
CSFSAE11 --------------- ICSF - Allocation ------------------------
COMMAND ===> _

DATA SET NAME: LARSON.CSFIN.TESTDS1P
  Data set cannot be found.  Specify allocation parameters below.

   VOLUME SERIAL      ===> ______       (Blank for authorized default volume) *
   GENERIC UNIT       ===> ________     (Generic group name or unit address) *
   SPACE UNITS        ===> BLOCK_____   (BLKS, TRKS, or CYLS)
   PRIMARY QUANTITY   ===> 10____       (In above units)
   SECONDARY QUANTITY ===> 5_____       (In above units)
   DIRECTORY BLOCKS   ===> 10____       (Zero for sequential data set)
   RECORD FORMAT      ===> FB
   RECORD LENGTH      ===> 80
   BLOCK SIZE         ===> 6400__       (In multiples of record length)
   EXPIRATION DATE    ===> _____        (Format is YYDDD)

   ( * Only one of these fields may be specified)



   Press ENTER to allocate specified data set and continue
   Press END   to exit to the previous panel without allocating
```

Figure 76. Entering Data Set Information on the Allocation Panel

Once the data set has been selected or created, the data set becomes the control statement input data set on the KGUP Control Statement Menu, as shown in Figure 77 on page 132. The name of the control statement input data set you specified appears at the top of the panel.

From this panel, you can press END to go back to the KGUP Control Statement Data Set Specification panel. On the later panel you can either specify another

data set to store control statements, or press END again to return to the Key Administration panel.

```
CSFCSM00 ----------- ICSF - KGUP Control Statement Menu ------------
OPTION ===> _

  Storage data set for control statements   (DDNAME = CSFIN)

  Data Set Name: LARSON.CSFIN.TESTDS1P(TEST2)


   Enter the number of the desired option above.

   1  Maintain       - Create ADD, UPDATE, or DELETE control statements
   2  Rename         - Create statement to RENAME entry label
   3  Set            - Create a statement to SET installation data
   4  Edit           - Edit the statement storage data set


   Press ENTER to go to the selected option
   Press END   to exit to the previous panel
```

Figure 77. KGUP Control Statement Menu Panel

3. Choose the type of control statement you want to create and press ENTER.
   - To create an ADD, UPDATE, or DELETE control statement, select option 1. For information, see “Creating ADD, UPDATE, or DELETE Control Statements”.
   - To create a RENAME control statement, select option 2. For information, see “Creating a RENAME Control Statement” on page 138.
   - To create a SET control statement, select option 3. For information, see “Creating a SET Control Statement” on page 140.
   - To edit the input control statement data set, select option 4. For information, see “Editing Control Statements” on page 142.

   After you choose the Maintain, Rename, or Set option, you access the panels to create the control statement you want. When you create a control statement, the statement is placed in the specified control statement input data set. To edit the control statements that are stored in this data set, choose the Edit option.

Creating ADD, UPDATE, or DELETE Control Statements

When you select Maintain (option 1) on the KGUP Control Statement Menu panel, the Create ADD, UPDATE, or DELETE Key Statement panel appears. See Figure 78 on page 133.

```
CSFCSE10 --- ICSF - Create ADD, UPDATE, or DELETE Key Statement -----------
COMMAND ===>
Specify control statement information below

   Function ===> ______     ADD, UPDATE, or DELETE
   Key Type ===> ________    Outtype ===> ________    (Optional)
   Label ===> ____________________________________________________________
    Group Labels  ===> NO_   NO or YES
 or Range:
   Start ===> ____________________________________________________________
   End   ===> ____________________________________________________________

    Transport Key Label(s)
         ===> ____________________________________________________________
         ===> ____________________________________________________________
 or Clear Key              ===> NO_        NO or YES

   Control Vector ===> YES  NO or YES    CDMF ===> NO   NO or YES
   Length of Key  ===> 16   8, 16 or 24  DES  ===> YES  NO or YES
   Key Values     ===> ________________ ,________________ ,________________
   Comment Line   ===> ____________________________________________________

Press ENTER to create and store control statement
Press END   to exit to the previous panel without saving
```

Figure 78. Create ADD, UPDATE, or DELETE Key Statement Panel

1. On the panel, fill out the fields to create the ADD, UPDATE, or DELETE control statement that you want KGUP to process. Each field on the panel corresponds to a control statement keyword. The panel helps you to create a complete, syntactically correct ADD, UPDATE, or DELETE control statement.

   The panel creates control statements according to the syntax described in “Syntax of the ADD and UPDATE Control Statements” on page 101. See that section for more information about the control statement keywords.
2. In the Function field, select the function you want KGUP to perform.

Function	Result
ADD	Enter new key entries in the CKDS. Generate and receive key values for key distribution.
UPDATE	Change existing entries in the CKDS. Generate and receive key values for key distribution.
DELETE	Remove entries from the CKDS.

   You can just type the first letter of the function in the first position in a field on the panel. For example, in Figure 79 on page 134, `a` was entered in the Function field to specify the ADD function. ICSF recognizes the abbreviation.

   For a description of the keywords you must specify for each function, see “Using the ADD and UPDATE Control Statements for Key Management and Distribution Functions” on page 107.

```
CSFCSE10 --- ICSF - Create ADD, UPDATE, or DELETE Key Statement -----------
COMMAND ===>
Specify control statement information below

   Function ===> a_____     ADD, UPDATE, or DELETE
   Key Type ===> ________    Outtype ===> ________    (Optional)
   Label ===> ____________________________________________________________
    Group Labels  ===> NO_   NO or YES
 or Range:
   Start ===> ____________________________________________________________
   End   ===> ____________________________________________________________

    Transport Key Label(s)
         ===> ____________________________________________________________
         ===> ____________________________________________________________
 or Clear Key              ===> NO_        NO or YES

   Control Vector ===> YES  NO or YES    CDMF ===> NO_  NO or YES
   Length of Key  ===> 16   8, 16 or 24  DES  ===> YES  NO or YES
   Key Values     ===> ________________ ,________________ ,________________
   Comment Line   ===> ____________________________________________________

Press ENTER to create and store control statement
Press END   to exit to the previous panel without saving
```

Figure 79. Selecting the ADD Function on the Create ADD, UPDATE, or DELETE Key Statement Panel

3. In the Key Type field, enter the type of key you want KGUP to process with the control statement. This field represents the TYPE keyword on the control statement.

   If you leave the Key Type Field blank and press ENTER, the Key Type Selection panel appears. See Figure 80.

```
CSFCSE12-------- ICSF - Key Type Selection Panel ----  ROW 1 TO 13 OF 11
COMMAND ===>                                            SCROLL ===> PAGE

Select one key type only
    KEY TYPE      DESCRIPTION
   DATA       Data-encrypting key
   DATAM      Double-length MAC generation key
   DATAMV     Double-length MAC verification key
   DATAXLAT   Data-translation key
s  EXPORTER   Export key-encrypting key
   IMPORTER   Import key-encrypting key
   IPINENC    Input PIN-encrypting key
   MAC        Message authentication key
   MACVER     Message verification key
   NULL       Dummy CKDS records
   OPINENC    Output PIN-encrypting key
   PINGEN     PIN generation key
   PINVER     PIN verification key
*******************************BOTTOM OF DATA********************************
```

Figure 80. Selecting a Key on the Key Type Selection Panel

   a. Type s to the left of the key type you want to specify from the displayed list of key types.

In Figure 80 on page 134, the exporter key is selected.

b. After you have specified a key type, press ENTER to return to the Create ADD, UPDATE, or DELETE Key Statement panel, as shown in Figure 81.

```
CSFCSE10 --- ICSF - Create ADD, UPDATE, or DELETE Key Statement -----------
COMMAND ===>
Specify control statement information below

   Function ===> ADD___      ADD, UPDATE, or DELETE
   Key Type ===> EXPORTER    Outtype ===> ________    (Optional)
   Label ===> ATMBRANCH5M0001_____________________________________________
    Group Labels  ===> NO_   NO or YES
 or Range:
   Start ===> ____________________________________________________________
   End   ===> ____________________________________________________________

    Transport Key Label(s)
         ===> tkatmbranch5m0001_____________________________________________
         ===> ____________________________________________________________
 or Clear Key               ===> NO_        NO or YES

   Control Vector ===> YES  NO or YES    CDMF ===> NO_  NO or YES
   Length of Key  ===> 16   8, 16 or 24  DES  ===> YES  NO or YES
   Key Values     ===> ________________ ,_______________ ,_______________
   Comment Line   ===> export test key_____________________________________

Press ENTER to create and store control statement
Press END   to exit to the previous panel without saving
```

Figure 81. Completing the Create ADD, UPDATE, or DELETE Key Statement Panel

If you abbreviated the control statement function, the function now appears in its full form. The type of key you selected on the Key Type Selection panel appears in the Key Type field.

4. Specify either a label or range to identify the label of the key entry in the CKDS that you want KGUP to process.

   The Label field represents the LABEL keyword on the control statement. The Range field represents the RANGE keyword on the control statement. In the Range fields, specify the first and last label in a range of labels you want KGUP to process.

Table 7. Selecting Range and Label Options

Option	Steps
To have KGUP process only one key label	a. Specify the key label in the Label field. b. Type NO in the Group Labels field.
To have KGUP process more than one key label	a. Specify the first label in the Label field. b. Type YES in the Group Labels field.

5. Specify either a transport key label or YES in the Clear Key field.

   The Transport Key Label field represents the TRANSKEY keyword on the control statement. The Clear Key field represents the CLEAR keyword. These keywords are mutually exclusive.

   When KGUP generates a key, the program places the key value in a data set so you can send the value to another system. The other system uses the

value to create the complement of the key. You send the key value as either a clear key value or a key value encrypted under a transport key.

When KGUP imports a key value, the program may import a clear or encrypted key value. KGUP decrypts the encrypted key value from under the transport key that you specify in the Transport Key Label field.

Table 8. Selecting the Transport Key Label and Clear Key Label Options

Option	Steps
To have KGUP generate a key other than an importer key and encrypt the key value	a. Specify the label of the transport key you want KGUP to use to encrypt the key in the Transport Key Label field. b. Type NO in the Clear Key field.
To have KGUP generate a key other than an importer key and leave the key value in the clear	a. Leave the Transport Key Label field blank b. Type YES in the Clear Key field.
To have KGUP import an encrypted key	a. Specify the label of the transport key you want KGUP to use to dencrypt the key in the Transport Key Label field. b. Type NO in the Clear Key field.
To have KGUP import a clear key	a. Leave the Transport Key Label field blank b. Type YES in the Clear Key field.

6. Specify either YES or NO in the Control Vector field.

   Usually the cryptographic facility exclusive ORs a transport key with a control vector before the transport key encrypts a key. However, if your system is exchanging keys with a system like PCF that does not use control vectors, you need to specify that no control vector be used. If you want KGUP to generate a transport key that uses a control vector, type YES in the Control Vectors field. Otherwise type NO. If you type NO in this field, the control statement contains the NOCV keyword.
7. If you want KGUP to work with a single-length key in its processing, type YES in the Length of Key field. Otherwise, type NO. If you type YES in the field, the control statement contains the LENGTH keyword.
8. If you want KGUP to generate a key that is marked for use with the DES algorithm, specify YES for DES and NO for CDMF. If you want KGUP to generate a key that is marked for use with the CDMF algorithm, specify NO for DES and YES for CDMF.

   The CDMF and DES keywords are valid only with the ADD and UPDATE requests and only when the key type is IMPORTER or EXPORTER.
9. If you are entering a key value, enter the key value in the Key Values field.

   You enter the value as three values if the key is a triple-length key, two values if the key is a double-length key, or as one value if the key is a single-length key. The Key Values field represents the KEY keyword on the control statement.
10. In the Comment Line field, you can enter up to 45 characters of information about the control statement. The information appears as a comment that precedes the control statement in the input control statement data set.
11. After you enter all the information on this panel, press ENTER.

    If you entered YES in the Group Labels field, the Group Label panel appears. See Figure 82 on page 137.

```
CSFCSE11 ----------------- ICSF - Group Label Panel ------------------
COMMAND ===>

First label:

  ATMBRANCH5M0001_____________________________________________________

Enter at least one other label:

  ATMBRANCH5M0020_____________________________________________________
  ATMBRANCH5M0030_____________________________________________________
  ATMBRANCH5M0050_____________________________________________________
  _____________________________________________________________________
  _____________________________________________________________________
  _____________________________________________________________________
  _____________________________________________________________________
  _____________________________________________________________________
  _____________________________________________________________________



Press ENTER to add more labels or create and store control statement
Press END   to exit to the previous panel without saving
```

Figure 82. Specifying Multiple Key Labels on the Group Label Panel

a. Enter any additional key labels you want KGUP to process with the control statement.

   The first label you entered in the Label field of the Create ADD, UPDATE, or DELETE Key Statement panel appears at the top of this panel. If you enter duplicate labels, an error message appears on the right side of the panel and the cursor appears on the duplicate label. If the syntax of the label is incorrect, an error message appears and the cursor appears on the incorrect label.

b. If you have more labels than will fit on this panel, press the ENTER key after you have filled each line on the panel. An additional Group Label Panel appears. Type the remaining labels and press ENTER.

   ICSF writes the control statement to the input control statement data set. You return to the Create ADD, UPDATE, or DELETE Key Statement panel.

If you entered `NO` in the Group Labels field, you do not access the Group Label panel. You remain on the Create ADD, UPDATE, or DELETE Key Statement panel.

12. Press ENTER to have ICSF write the control statement in the input control statement data set.

    If a specification in any field is incorrect, when ICSF processes the control statement it displays an appropriate message on the top line of the panel. The cursor then appears in the field with the error. To display the long version of the error message at the bottom of the panel, press the HELP key (F1). If you correct the error and press ENTER again, ICSF writes the control statement to the control statement input data set.

    If a control statement was created, the message `SUCCESSFUL UPDATE` appears on the right side of the top line of the panel, as shown in Figure 83 on page 138.

```
CSFCSE10 - ICSF - Create ADD, UPDATE, DELETE Statement SUCCESSFUL UPDATE
COMMAND ===>
Specify control statement information below

   Function ===> ADD___       ADD, UPDATE, or DELETE
   Key Type ===> EXPORTER    Outtype ===> ________    (Optional)
   Label ===> ATMBRANCH5M0001_____________________________________________
    Group Labels  ===> NO_   NO or YES
 or Range:
   Start ===> ____________________________________________________________
   End   ===> ____________________________________________________________

    Transport Key Label(s)
         ===> TKATMBRANCH5M0001_____________________________________________
         ===> ____________________________________________________________
 or Clear Key              ===> NO_        NO or YES

   Control Vector ===> YES  NO or YES   CDMF ===> NO_  NO or YES
   Length of Key  ===> 16   8,16 or 24  DES  ===> YES or NO
   Key Values     ===> ________________ ,________________ ,________________
   Comment Line   ===> EXPORT TEST KEY_____________________________

Press ENTER to create and store control statement
Press END   to exit to the previous panel without saving
```

Figure 83. Create ADD, UPDATE, or DELETE Key Statement Panel Showing Successful Update

13. If you want to create another ADD, UPDATE, or DELETE control statement, enter new information in the fields to create the control statement.
14. After you specify the information, press ENTER to place the control statement in the control statement input data set.
15. If you do not want to create another ADD, UPDATE, or DELETE control statement, press END to return to the KGUP Control Statement Menu panel.

Creating a RENAME Control Statement

The Create RENAME Control Statement panel appears. The RENAME control statement changes the label of a key entry in a CKDS. To create a RENAME control statement:

1. Choose option 2 on the KGUP Control Statement Menu, as shown in Figure 84.

```
CSFCSM00 ----------- ICSF - KGUP Control Statement Menu ------------
OPTION ===> 2

  Storage data set for control statements   (DDNAME = CSFIN)

  Data Set Name: LARSON.CSFIN.TESTDS1P(TEST2)


   Enter the number of the desired option above.

   1  Maintain       - Create ADD, UPDATE, or DELETE control statements
   2  Rename         - Create statement to RENAME entry label
   3  Set            - Create a statement to SET installation data
   4  Edit           - Edit the statement storage data set
```

Figure 84. Selecting the Rename Option on the KGUP Control Statement Menu Panel

2. See Figure 85. If you leave this field blank, the On this panel, you enter information in the fields to create a RENAME control statement. This panel creates a RENAME control statement according to the syntax described in “Syntax of the RENAME Control Statement” on page 113. See that section for more information about the RENAME control statement keywords.

```
CSFCSE20 --------- ICSF - Create RENAME Control Statement ----------
COMMAND ===>

Enter the following information:

   Existing Key Label
     _____________________________________________________________

   New Key Label
     _____________________________________________________________

   Key Type           ===> ________         Selection panel displayed if blank

   Comment Line       ===> ________________________________________

Press ENTER to create and store control statement
Press END   to exit to the previous panel
```

Figure 85. Create RENAME Control Statement Panel

3. In the Existing Key Label field, specify the current label on the CKDS that you want KGUP to change.
4. In the New Key Label field, specify the new label that you want to replace the existing label.
5. In the Key Type field, specify the key type of the key entry whose label you want changed. Key Type Selection panel appears. See Figure 86.

```
CSFCSE12--------- ICSF - Key Type Selection Panel ----- ROW 1 To 13 OF 11
COMMAND ===>                                            SCROLL ===> PAGE

Select one key type only
    KEY TYPE      DESCRIPTION
   DATA       Data-encrypting key
   DATAM      Double-length MAC generation key
   DATAMV     Double-length MAC verification key
   DATAXLAT   Data-translation key
s  EXPORTER   Export key-encrypting key
   IMPORTER   Import key-encrypting key
   IPINENC    Input PIN-encrypting key
   MAC        Message authentication key
   MACVER     Message verification key
   NULL       Dummy CKDS records
   OPINENC    Output PIN-encrypting key
   PINGEN     PIN generation key
   PINVER     PIN verification key
*********************************BOTTOM OF DATA*********************************
```

Figure 86. Selecting a Key Type on the Key Type Selection Panel

   a. Type `s` to the left of the key type you want to specify.
      In Figure 86, the exporter key is selected.
   b. Press ENTER to return to the Create RENAME Control Statement panel.

The RENAME control statement The key type you choose on the Key Type Selection panel appears in the key type field.

An example of a Create RENAME Control Statement panel which creates a control statement to change the key label JWL@SSIDEC95 to JWL@SSIJUNE96 for an exporter key is shown in Figure 87.

```
CSFCSE20 --------- ICSF - Create RENAME Control Statement ----------
COMMAND ===>

Enter the following information:

   Existing Key Label
     JWL@SSIDEC95_____________________________________________________

   New Key Label
     JWL@SSIJUNE96____________________________________________________

   Key Type           ===> ex______         Selection panel displayed if blank

   Comment Line       ===> export test key renamed_____________________

Press ENTER to create and store control statement
Press END   to exit to the previous panel
```

Figure 87. Completing the Create RENAME Control Statement Panel

6. In the Comment Line field, you can enter up to 45 characters of information about the control statement.

   The information appears as a comment that precedes the control statement in the input control statement data set.
7. After you enter all the information on the Create RENAME Control Statement panel, press ENTER.

   ICSF writes the control statement in the input control statement data set.

   If a specification in any field is incorrect, when ICSF processes the control statement it displays an appropriate message on the top line of the panel. The cursor then appears in the field with the error. To display the long version of the error message at the bottom of the panel, press the HELP key (F1). You can correct the error and press ENTER again so ICSF can write the control statement to the control statement input data set.

   The Create SET Control Statement panel appears. If a control statement was created, the message `SUCCESSFUL UPDATE` appears on the right side of the top line of the panel.
8. To create another RENAME control statement, enter new information in the fields to create the control statement.
9. After you specify the information, press ENTER to place the control statement in the control statement input data set.
10. When you have finished creating RENAME control statements, press END to return to the KGUP Control Statement Menu panel.

Creating a SET Control Statement

The SET control statement specifies data for KGUP to send to a KGUP exit routine. To create a SET control statement:

1. Choose option 3 on the KGUP Control Statement Menu, as shown in Figure 88 on page 141.

```
CSFCSM00 ----------- ICSF - KGUP Control Statement Menu ------------
OPTION ===> 3

  Storage data set for control statements   (DDNAME = CSFIN)

  Data Set Name: LARSON.CSFIN.TESTDS1P(TEST2)


   Enter the number of the desired option above.

   1  Maintain       - Create ADD, UPDATE, or DELETE control statements
   2  Rename         - Create statement to RENAME entry label
   3  Set            - Create a statement to SET installation data
   4  Edit           - Edit the statement storage data set
```

Figure 88. Selecting the Set Option on the KGUP Control Statement Menu Panel

2. See Figure 89. From this panel you can creates a SET control statement. For information about the SET control statement keywords, refer to “Syntax of the SET Control Statement” on page 114.

```
CSFCSE30 ----------- ICSF - Create SET Control Statement -----------
COMMAND ===>

Specify installation data for exit processing

  Installation Data ===> ________________________________________________

  Comment Line      ===> ____________________________________________

Press ENTER to create and store control statement
Press END   to exit to the previous panel without saving
```

Figure 89. Create SET Control Statement Panel

3. In the Installation Data field, enter the data to pass to a KGUP installation exit.
4. In the Comment Line field, you can enter up to 45 characters of information about the control statement.

   The information appears as a comment that precedes the control statement in the input control statement data set.

   An example of a Create SET Control Statement panel which passes date information to the installation exit is shown in Figure 90.

```
CSFCSE30 ----------- ICSF - Create SET Control Statement -----------
COMMAND ===>

Specify installation data for exit processing

  Installation Data ===> BRANCH0519921101199301310________________________

  Comment Line      ===> Branch 5 POS terminal date information_________

Press ENTER to create and store control statement
Press END   to exit to the previous panel without saving
```

Figure 90. Completing the Create SET Control Statement Panel

5. After you enter all the information on this panel, press ENTER.

ICSF writes the control statement in the input control statement data set.

When the control statement is created, the message SUCCESSFUL UPDATE appears on the right side of the top line of the panel.

6. Press END to return to the KGUP Control Statement Menu panel.

Editing Control Statements

You can edit the control statement input data set that you specified for this KGUP job. The control statement input data set contains the control statements you created after you specified the control statement input data set.

To edit the control statements you created:

1. Choose option 4 on the KGUP Control Statement Menu panel, as shown in Figure 91.

```
CSFCSM00 ----------- ICSF - KGUP Control Statement Menu ------------
OPTION ===> 4

  Storage data set for control statements   (DDNAME = CSFIN)

  Data Set Name: LARSON.CSFIN.TESTDS1P(TEST2)


   Enter the number of the desired option above.

   1  Maintain       - Create ADD, UPDATE, or DELETE control statements
   2  Rename         - Create statement to RENAME entry label
   3  Set            - Create a statement to SET installation data
   4  Edit           - Edit the statement storage data set


   Press ENTER to go to the selected option
   Press END   to exit to the previous panel
```

Figure 91. Selecting the Edit Option on the KGUP Control Statement Menu Panel

The ISPF editor displays the control statement input data set. An example of a data set called LARSON.CSFIN.TESTDS1P(TEST2) with a SET, ADD, and RENAME control statement is shown in Figure 92.

```
ISREDDE - LARSON.CSFIN.TESTDS1P(TEST2) - 00.00 --------------- COLUMNS 001 072
COMMAND ===> _                                                SCROLL ===> CSR
****** ***************************** TOP OF DATA ******************************
000001 /* TEST INSTALLATION DATA                          */
000002 SET INSTDATA('This is test installation data')
000003 /* EXPORT TEST KEY                                 */
000004 ADD TYPE(EXPORTER),
000005        TRANSKEY(SENDTOBRANCH5JUNE99)
000006        LABEL(ATMBRANCH5M0001)
000007 /* EXPORT TEST KEY RENAMED                         */
000008 RENAME LABEL(JWL@SSIDEC97,JWL@SSIJUNE99) TYPE(EXPORTER)
****** **************************** BOTTOM OF DATA ****************************
```

Figure 92. Edit Control Statement Initial Display Panel

2. You can change any information on the control statements in the data set. You can also add lines to the data set that contains comments or control statements.
3. To specify many similar control statements, copy lines in this file and edit them to create additional control statements.

Note: The panel does not check whether the control statements that you change are syntactically correct.

Figure 93 shows the insertion of a comment line in the file.

```
ISREDDE - LARSON.CSFIN.TESTDS1P(TEST2) - 00.00 --------------- COLUMNS 001 072
COMMAND ===>                                                  SCROLL ===> CSR
****** ***************************** TOP OF DATA ******************************
'' /* This comment was inserted using the editor */_
000001 /* TEST INSTALLATION DATA                          */
000002 SET INSTDATA('This is test installation data')
000003 /* EXPORT TEST KEY                                 */
000004 ADD TYPE(EXPORTER),
000005        TRANSKEY(SENDTOBRANCH5JUNE99)
000006        LABEL(ATMBRANCH5M0001)
000007 /* EXPORT TEST KEY RENAMED                         */
000008 RENAME LABEL(JWL@SSIDEC97,JWL@SSIJUNE99) TYPE(EXPORTER)
****** **************************** BOTTOM OF DATA ****************************
```

Figure 93. Edit Control Statement Data Set with Insert

4. After you make any changes, press END to save the changes and return to the KGUP Control Statement Menu panel.

Specifying Data Sets Using the ICSF Panels

Before you run a KGUP job, you must specify the KGUP data sets for the program to use in its processing.

1. To access the panels to specify KGUP data sets, select option 2 on the Key Administration panel, as shown in Figure 94, and press ENTER.

```
CSFSAM00 ----------------- ICSF - Key Administration ---------------
OPTION ===> 2


   Enter the number of the desired option.

   1  Create         - Create key generator control statements

   2  Data Set       - Specify data sets for processing

   3  Submit         - Invoke Key Generator Utility Program (KGUP)

   4  Refresh        - Activate an existing cryptographic key data set


   Press ENTER to go to the selected option
   Press END   to exit to the previous panel
```

Figure 94. Selecting the Specify Data Set Option on the Key Administration Panel

The Specify KGUP Data Sets panel appears. See Figure 95 on page 144.

```
CSFSAE20 ---------------- ICSF - Specify KGUP Data Sets ------------
COMMAND ===> _

Enter data set names for all cryptographic files.
 Cryptographic Key         (DDNAME = CSFCKDS)
   Data Set Name ===> _____________________________________________

 Control Statement Input   (DDNAME = CSFIN)
   Data Set Name ===> _____________________________________________
   Volume Serial ===> ______  (if uncataloged)

 Diagnostics               (DDNAME = CSFDIAG)  (use * for printer)
   Data Set Name ===> _____________________________________________
   Volume Serial ===> ______  (if uncataloged)

 Key Output                (DDNAME = CSFKEYS)
   Data Set Name ===> _____________________________________________
   Volume Serial ===> ______  (if uncataloged)

 Control Statement Output  (DDNAME = CSFSTMNT)
   Data Set Name ===> _____________________________________________
   Volume Serial ===> ______  (if uncataloged)

Press ENTER to set the data set names. Press END to exit to the previous panel.
```

Figure 95. Specify KGUP Data Sets Panel

This panel contains all the data sets that KGUP uses for input or output during processing. In the Data Set Name field under each type of data set, you specify the name of the data set for KGUP to use.

2. In the Cryptographic Key Data Set Name field, specify the name of the CKDS which contains the key entries that KGUP processes.

   You must initialize the CKDS by using the method that is described in “Initializing the CKDS at First-Time Startup” on page 68. The data set can be any disk copy of a CKDS that is enciphered under the current master key.
3. In the Control Statement Input Data Set Name field, specify the name of the data set that contains the control statements you want KGUP to process for this job.
4. In the Volume Serial field, enter the volume serial for the data set if it is not cataloged.

   If you specified a control statement input data set on the KGUP Control Statement Data Set Specification panel, the data set name appears in the Control Statement Input Data Set Name field on this panel. If you change the data set name on this panel, it automatically changes on the KGUP Control Statement Data Set Specification panel. Refer to Figure 73 on page 129 for an example of the KGUP Control Statement Data Set Specification panel.
5. In the Diagnostics Data Set Name field, specify the name of the data set where KGUP places the image of the control statements and any diagnostic KGUP generates.

   You do not have to allocate this data set before you specify the data set in this field. If the data set does not already exist, then a job control language statement that allocates the data set can be used when you submit the job.
6. In the Volume Serial field, enter the volume serial for the data set if the data set already exists but is not cataloged.

   If you enter an * in the Diagnostics Data Set Name field, the information is printed directly to a printer instead of a data set.

7. In the Key Output Data Set Name field, specify the name of the data set that contains key values that are generated to use to create complementary key values.

   You do not have to allocate this data set before you specify the data set in this field. If the data set does not already exist, then a job control language statement that allocates the data set can be used when you submit the job.
8. In the Volume Serial field, enter the volume serial for the data set if the data set already exists but is not cataloged.
9. In the Control Statement Output Data Set Name field, specify the name of the data set that contains control statements generated to use to create complementary key values.

   You do not have to allocate this data set before you specify the data set in this field. If the data set does not already exist, then a job control language statement that allocates the data set can be used when you submit the job.
10. In the Volume Serial field, enter the volume serial for the data set if the data set already exists but is not cataloged.

    For a more complete description of each of the data sets, see “Specifying KGUP Data Sets” on page 119.

    The data sets that you name appear on this panel the next time you access it.

    An example of a Specify KGUP Data Sets panel with the names of data sets specified for KGUP processing is shown in Figure 96.

```
CSFSAE20 ---------------- ICSF - Specify KGUP Data Sets ------------
COMMAND ===> _

Enter data set names for all cryptographic files.
 Cryptographic Key          (DDNAME = CSFCKDS)
   Data Set Name ===> TEST.CSFCKDS_____________________________

 Control Statement Input    (DDNAME = CSFIN)
   Data Set Name ===> CSFIN.TESTDS1P(TEST)_____________________
   Volume Serial ===> ______  (if uncataloged)

 Diagnostics                (DDNAME = CSFDIAG)  (use * for printer)
   Data Set Name ===> *________________________________________
   Volume Serial ===> ______  (if uncataloged)

 Key Output                 (DDNAME = CSFKEYS)
   Data Set Name ===> TEST.CSFKEYS_____________________________
   Volume Serial ===> ______  (if uncataloged)

 Control Statement Output   (DDNAME = CSFSTMNT)
   Data Set Name ===> TEST.CSFSTMNT____________________________
   Volume Serial ===> ______  (if uncataloged)

Press ENTER to set the data set names. Press END to exit to the previous panel.
```

Figure 96. Completing the Specify KGUP Data Sets Panel

11. Press ENTER to set the data set names.
12. Press END to return to the ICSF Key Administration panel.

Creating the Job Stream Using the ICSF Panels

The Set KGUP JCL Job Card panel appears. After you create the control statements and specify the data sets for KGUP processing, you submit the job to run KGUP. You submit a KGUP job stream to process control statements which

modify a CKDS and output information to other data sets. The names of the data sets that KGUP uses are specified in the job stream.

1. To access the panels to create the KGUP job stream, select option 3 on the Key Administration panel, as shown in Figure 97, and press ENTER.

```
CSFSAM00 ----------------- ICSF - Key Administration ---------------
OPTION ===> 3


   Enter the number of the desired option.

   1  Create         - Create key generator control statements

   2  Data Set       - Specify data sets for processing

   3  Submit         - Invoke Key Generator Utility Program (KGUP)

   4  Refresh        - Activate an existing cryptographic key data set


   Press ENTER to go to the selected option
   Press END   to exit to the previous panel
```

Figure 97. Invoking KGUP by Selecting the Submit Option on the Key Administration Panel

See Figure 98. The first time you access this panel, the panel displays a JOB statement similar to the one that is shown in this example. ICSF displays your userid as the job name. From this panel you can create a job to run KGUP.

```
CSFSAE30 ---------------- ICSF - Set KGUP JCL Job Card --------------
COMMAND ===> _

   S  - Submit the KGUP job stream for execution
   E  - Edit the KGUP job stream and issue the TSO SUBMIT command

        Note:  If you choose E, and want to submit the job stream with
        your changes, issue the TSO SUBMIT command before you leave the
        edit session; your updates to the job stream will NOT be saved.

   Enter or verify job statement information:

   ===> //LARSON  JOB (ACCOUNT),'NAME',MSGCLASS=C_____________________________
   ===> //*_____________________________________________________________________
   ===> //*_____________________________________________________________________
   ===> //*_____________________________________________________________________

   Enter dsname of library containing Installation Exit Module:

   ===> ____________________________________________

   Special Secure Mode       ===> NO_  NO or YES

   Press END to exit to previous panel
```

Figure 98. Set KGUP JCL Job Card Panel

2. Change the job statement according to the specifications of your installation.

   The line of the job control language that appears on this panel contains the job card that is needed to submit the job on the Job Entry Subsystem (JES). This panel displays some commonly used parameters that are installation dependent.

A job name and the word JOB are the only required parameters on a job statement. All the other parameters are only required depending on your installation. You can delete or specify these parameters and add more parameters depending on the requirements of your installation. When you change the information that is displayed, ICSF saves these changes so they appear every time you display the panel.

a. In the ACCOUNT parameter, enter accounting information as specified by your installation.
b. In single quotes, enter the name that appears on the output of the job.
c. In the MSGCLASS parameter, set the output class for the job log.

   After you specify the JOB statement information, the panel displays three comment lines where you can include any information about the job.
d. If all the parameters do not fit on the first line, delete the * on the second line and continue the JOB statement parameters.

3. If your installation calls an installation exit during KGUP processing and the library containing the exit load module is not in the link list, specify the library in the "Enter dsname of library containing Installation Exit Module" field.

   Because the library must be an authorized library, the library must be defined in your installation's IEAAPFxx member.
4. If any of the control statements contain the CLEAR keyword, specify YES in the Special Secure Mode field. Otherwise, ICSF does not have to be in special secure mode, and you should specify NO in the Special Secure Mode field.
5. After you specify the necessary information, you can either:
   - Enter S to submit the job.

     KGUP creates the job stream and automatically submits the job to run the program.
   - Enter E to edit the job.

     KGUP creates the job stream and then displays the job stream on a panel in ISPF edit mode. Figure 99 on page 148 shows an example of a panel in ISPF edit mode that contains a job stream to run KGUP. When ICSF creates the job stream, ICSF defines the data sets that KGUP uses in the job. It defines these data sets according to the information you specified on the Specify KGUP Data Sets Panel. Refer to Figure 96 on page 145.

     a. On this panel, you can view the job stream ICSF created and make any necessary changes to the job stream.
     b. To submit your job with the changes, you must use the TSO SUBMIT command from the edit session. Type SUBMIT on the command line and press ENTER to submit the job and run KGUP.
     c. To return to the Set KGUP JCL Job Card panel without submitting the job stream, press END.

        The job stream is not saved after you leave this panel.

```
ISREDDE - SYS88218.T095045.RA000.LARSON.R0000002 ------------- COLUMNS 001 072
COMMAND ===> _                                                 SCROLL ===> CSR
****** ***************************** TOP OF DATA ****************************
000001 //LARSON  JOB (ACCOUNT),'NAME',MSGCLASS=C
000002 //*
000003 //*
000004 //*
000005 //KGUP     EXEC PGM=CSFKGUP,PARM=('NOSSM')
000006 //CSFCKDS  DD   DSN=LARSON.TEST.CSFCKDS,
000007 //              DISP=OLD
000008 //CSFIN    DD   DSN=LARSON.CSFIN.TESTDS1P(TEST),
000009 //              DISP=OLD
000010 //CSFDIAG  DD   SYSOUT=*
000011 //CSFKEYS  DD   DSN=LARSON.TEST.CSFKEYS,
000012 //              DISP=OLD
000013 //CSFSTMNT DD   DSN=LARSON.TEST.CSFSTMNT,
000014 //              DISP=OLD
****** **************************** BOTTOM OF DATA ****************************
```

Figure 99. KGUP JCL Set for Editing and Submitting (Files Exist)

Example of a KGUP Job Stream with Existing Data Sets

The KGUP job stream in Figure 99 is an example of a job stream in which the data sets already exist.

In the EXEC statement of the job stream that ICSF created, the PGM parameter specifies that the job run KGUP. The PARM parameter notifies KGUP whether special secure mode is enabled. The keyword SSM indicates that the mode is enabled, and NOSSM indicates that the mode is not enabled.

The data definition (DD) statements identify the data sets that KGUP uses while processing. ICSF uses the names you provide on the Specify KGUP Data Sets panel. The cryptographic key data set (CSFCKDS) and the control statement input data set (CSFIN) have to exist before ICSF can generate the job stream. The other data sets do not have to already exist. In the example that is shown on this panel, all the data sets existed before ICSF created the job stream.

On the DD statements, the DSN parameter specifies the data set name. ICSF uses the name you provide on the Specify KGUP Data Sets panel for the data set name. The DISP parameter indicates the data set's status. On this panel, all the data sets existed before ICSF created this job stream, therefore the job stream indicates a status of OLD for the data sets.

In Figure 99, the DD statement for the diagnosis data set (CSFDIAG) is different from the other DD statements. The SYSOUT=* parameter specifies that ICSF print the data set on the output listing.

Note: You can change the default values that are used with the job control language such as the record format and record length by changing the outline file, CSFSAJ30. The information appears in the front of CSFSAJ30. CSFSAJ30 resides in the ICSF skeleton library.

Example of a KGUP Job Stream with Non-Existing Data Sets

Figure 100 on page 149 shows an example of a panel in ISPF edit mode that contains a KGUP job stream where certain data sets did not exist previously.

```
ISREDDE - SYS88218.T095045.RA000.LARSON.R0000003 ------------ COLUMNS 001 072
COMMAND ===> _                                                SCROLL ===> CSR
****** ***************************** TOP OF DATA ****************************
000001 //LARSON  JOB (ACCOUNT),'NAME',MSGCLASS=C
000002 //*
000003 //*
000004 //*
000005 //KGUP     EXEC PGM=CSFKGUP,PARM=('NOSSM')
000006 //CSFCKDS  DD   DSN=LARSON.TEST.CSFCKDS,
000007 //              DISP=OLD
000008 //CSFIN    DD   DSN=LARSON.CSFIN.TESTDS2P(TEST2),
000009 //              DISP=OLD
000010 //CSFDIAG  DD   DSN=LARSON.TEST.CSFDIAG,
000011 //              DISP=(,CATLG,CATLG),UNIT=SYSDA,
000012 //              DCB=(RECFM=FBA,LRECL=133,BLKSIZE=13300),
000013 //              SPACE=(TRK,(220,10),RLSE)
000014 //CSFKEYS  DD   DSN=LARSON.TEST.CSFKEYS,
000015 //              DISP=(,CATLG,CATLG),UNIT=SYSDA,
000016 //              DCB=(RECFM=FB,LRECL=208,BLKSIZE=3328),
000017 //              VOL=SER=TSO001,SPACE=(TRK,(60,10),RLSE)
000018 //CSFSTMNT DD   DSN=LARSON.TEST.CSFSTMNT,
000019 //              DISP=(,CATLG,CATLG),UNIT=SYSDA,
000020 //              DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),
000021 //              SPACE=(TRK,(60,10),RLSE)
****** **************************** BOTTOM OF DATA ****************************
```

Figure 100. KGUP JCL Set for Editing and Submitting (Files Do Not Exist)

The job stream contains information to create the diagnosis data set (CSFDIAG), key output data set (CSFKEYS), and the control statement output data set (CSFSTMNT) that did not previously exist. On the DISP parameter, the CATLG keyword specifies that you want the data set cataloged when the job ends normally and when the job ends abnormally. The unit parameter indicates the device you want the data set to reside on. The DCB parameter specifies the necessary data control block information such as the record format (RECFM), record length (LRECL) and block size (BLKSIZE).

When you submit the job, KGUP performs the functions you specified on the control statements. The functions KGUP performs change the CKDS. You can view the diagnostics data set to know whether KGUP successfully processed the control statements.

Refreshing the Current CKDS Using the ICSF Panels

KGUP processing affects keys that are stored on a disk copy of the CKDS. You specify the name of the data set when you submit the KGUP job. For information on specifying the disk copy of the CKDS for KGUP processing, see “Specifying Data Sets Using the ICSF Panels” on page 143.

ICSF functions use an in-storage copy of the CKDS. To make the changes caused by the KGUP processing active, you replace the in-storage copy of the CKDS with the disk copy that the KGUP processing changed. You refresh the current copy of the CKDS with the changed disk copy of the CKDS.

1. To access the panels to refresh the current CKDS, choose option 4 on the Key Administration panel, as shown in Figure 101 on page 150.

```
CSFSAM00 ----------------- ICSF - Key Administration ---------------
OPTION ===> 4


   Enter the number of the desired option.

   1  Create         - Create key generator control statements

   2  Data Set       - Specify data sets for processing

   3  Submit         - Invoke Key Generator Utility Program (KGUP)

   4  Refresh        - Activate an existing cryptographic key data set


   Press ENTER to go to the selected option
   Press END   to exit to the previous panel
```

Figure 101. Selecting the Refresh Option on the Key Administration Panel

The Refresh in-storage CKDS panel appears. See Figure 102.

```
CSFSAE40 ------------------- ICSF - Refresh in-storage CKDS --------
COMMAND ===> _

  Enter the Cryptographic Key Data Set (CKDS) to be loaded.

  Cryptographic Keys ===> TEST.CSFCKDS_____________________________


  Press ENTER to refresh the in-storage copy of CKDS
  Press END   to exit to previous panel
```

Figure 102. Refresh In-Storage CKDS

2. Enter the name of the disk copy of the CKDS to replace the current in-storage copy.

   The name of the CKDS that you chose when you specified data sets for KGUP processing on the Specify KGUP Data Sets panel, automatically appears on this panel. If you change the data set name on this panel, the data set name on the Specify KGUP Data Sets panel also changes. Refer to Figure 96 on page 145 for an example of the Specify KGUP Data Sets panel.
3. Press ENTER to replace the in-storage copy of the CKDS with the disk copy.

   Applications that are running on ICSFare not disrupted. A message that stating that the CKDS was refreshed appears on the right of the top line on the panel.

   ICSF performs a MAC verification on the records before reading the CKDS into storage. If a record fails the MAC verification, the record is not loaded into storage. The operator receives a message indicating the key label and type for that record.
4. Press END to return to the Key Administration Panel.

   Note: If you restart ICSF, the name of the disk copy that you specify in the CKDSN installation option is read into storage.

Scenario of Two ICSF Systems Establishing Initial Transport Keys

This scenario describes how two ICSF systems, System A and System B, establish initial transport keys between themselves. They establish two pairs of complementary importer and exporter keys at each location, as shown in Figure 103.

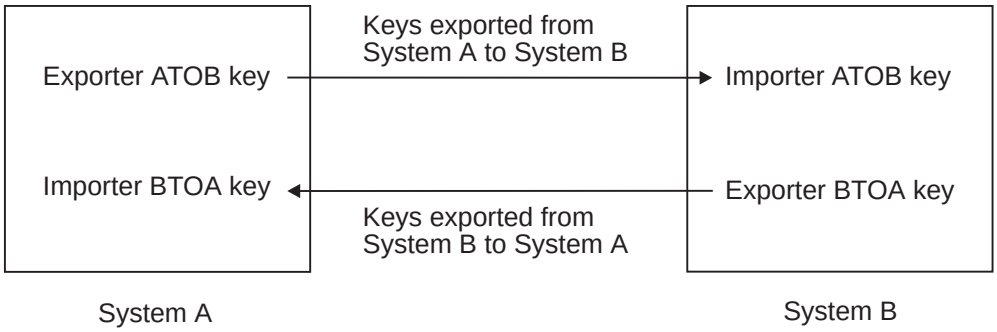


Figure 103. Key Exchange Establishment between Two ICSF Systems

The systems can use these importer and exporter keys during key exchange. First the ICSF administrators at the two locations establish the complementary transport keys to send keys from System A to System B. These keys are the Exporter ATOB key at System A and the Importer ATOB key at System B.

The ICSF administrator at System A submits the following control statement to System A's KGUP to create the Exporter ATOB key.

```
ADD  LABEL(ATOB)  TYPE(EXPORTER)  CLEAR
```

KGUP processes this control statement to generate the Exporter ATOB key and places the key in System A's CKDS. KGUP creates a record containing the clear key created for the system, and that record is written to the CSFKEYS data set. This key value must be used to create a control statement like the following.

```
ADD  LABEL(ATOB)  TYPE(IMPORTER)  CLEAR,
KEY(B2403EF8125A036F,239AC35A72941EF2)
```

System A can send this control statement to System B, and System B can create the Importer ATOB key. The key value in this control statement is the clear value of the Exporter ATOB key. System A does not send this control statement to System B over the network, because the key value is a clear key value. System A has a courier deliver the control statement to System B.

The administrator at System B submits the control statement to its KGUP. KGUP processes the control statement to create the ATOB importer key. The ATOB exporter key at system A and the ATOB importer key at System B are complementary keys.

This procedure creates a pair of complementary transport keys for keys sent from System A to System B. When System A sends a key to System B it enciphers the key using the ATOB exporter key. When System B receives the key, System B deciphers the key using the ATOB importer key.

Then the ICSF administrators at the two locations establish the complementary transport keys to send keys from System B to System A. These keys are the Importer BTOA key at System A and the Exporter BTOA key at System B.

The ICSF administrator at System A submits the following control statement to System A's KGUP to generate the Importer BTOA key.

```
ADD  LABEL(BTOA)  TYPE(IMPORTER)  TRANSKEY(ATOB)
```

KGUP processes this control statement to generate the Importer BTOA key and places the key in System A's CKDS. KGUP also creates the following control statement and places the statement in the control statement output data set.

```
ADD  LABEL(BTOA)  TYPE(EXPORTER)  TRANSKEY(ATOB),
KEY(AF04C35A7F1C9636,03CBB854653A0BCF)
```

System A can send this control statement to System B and System B can use the statement to create the Exporter BTOA key. The key value in this control statement is the value of the Importer BTOA key enciphered under the Exporter ATOB key. System A can send this control statement to System B over the network, because the key value is enciphered.

The ICSF administrator at System B submits the control statement to its KGUP. The program processes the control statement to generate the Exporter BTOA key. The Importer BTOA key at System A and the Exporter BTOA key at System B are complementary keys.

This procedure creates a pair of complementary transport keys for keys sent from System B to System A. When System B sends a key to System A, System B enciphers the key using the Exporter BTOA key. When System A receives the key, System A deciphers the key using the Importer BTOA key.

Using these procedures two pairs of complementary transport keys are established at each facility to allow key exchange between the two facilities.

Notes:

1. During these procedures, the special secure mode at each system must be enabled, while KGUP is generating or receiving clear key values.
2. The ICSF administrator at System A can submit in the same KGUP job both the ADD control statements meant for processing at System A.
3. The ICSF administrator at System B can submit in the same KGUP job both the ADD control statements meant for processing at System B.

Scenario of an ICSF System and a PCF System Establishing Initial Transport Keys

This scenario describes how an ICSF system and a PCF system establish initial transport keys between themselves. They establish two pairs of complementary importer and exporter keys at each location, as shown in Figure 104 on page 153.

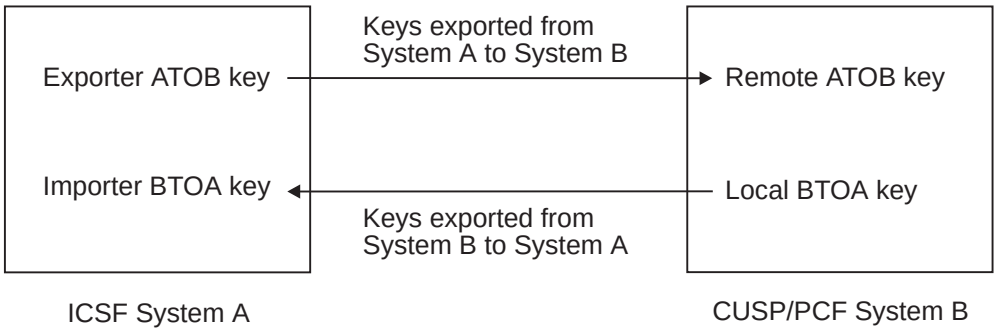


Figure 104. Key Exchange Establishment between an ICSF System and a PCF System

The systems can use these importer and exporter keys during key exchange.

First the ICSF administrators at the two locations establish the complementary transport keys to send keys from ICSF System A to PCF System B. These keys are the Exporter ATOB key at ICSF System A and the Remote ATOB key at PCF System B.

The ICSF administrator at ICSF System A submits the following control statement to ICSF System A's KGUP to create the Exporter ATOB key.

```
ADD  LABEL(ATOB)  TYPE(EXPORTER)  CLEAR  NOCV
```

Note: If System B is a PCF system, the ICSF administrator must also specify the keyword SINGLE on this control statement.

KGUP processes this control statement to generate the Exporter ATOB key and places the key in ICSF System A's CKDS. KGUP also creates the following control statement and places the statement in the control statement output data set.

```
ADD  LABEL(ATOB)  TYPE(IMPORTER)  CLEAR,
KEY(B2403EF8125A036F,239AC35A72941EF2)  NOCV
```

ICSF System A needs to send this control statement to PCF System B so that PCF System B can create the Remote ATOB key. The key value in this control statement is the clear value of the ATOB exporter key. ICSF System A does not send this control statement to PCF System B over the network, because the key value is a clear key value. ICSF System A has a courier deliver the control statement to System B.

The administrator at either system must change the ICSF control statement format into the PCF control statement format. The administrator could also use information from the key output data set to create the PCF control statement.

The control statement submitted at PCF System B would have the following syntax:

```
REMOTE ATOB,KEY=B2403EF8125A036F,IKEY=239AC35A72941EF2,ADD
```

The administrator at PCF System B submits the control statement to the PCF key generation utility program, which processes the control statement to create the ATOB Remote key. The ATOB Exporter key at System A and the ATOB Remote key at PCF System B are complementary keys.

This procedure creates a pair of complementary transport keys for keys sent from ICSF System A to PCF System B. When ICSF System A sends a key to PCF

System B, System A enciphers the key using the ATOB exporter key. When PCF System B receives the key, PCF System B deciphers the key using the Remote ATOB key.

Then the ICSF administrators at the two locations establish the complementary transport keys to send keys from PCF System B to ICSF System A. These keys are the Importer BTOA key at ICSF System A and the Local BTOA key at PCF System B.

The ICSF administrator at ICSF System A submits the following control statement to ICSF System A's KGUP to generate the Importer BTOA key.

```
ADD  LABEL(BTOA)  TYPE(IMPORTER)  CLEAR  NOCV
```

KGUP processes this control statement to generate the Importer BTOA key and places the statement in ICSF System A's CKDS. KGUP also creates the following control statement and places the statement in the control statement output data set.

```
ADD  LABEL(BTOA)  TYPE(EXPORTER)  CLEAR
KEY(6F3463CA3FBC0626,536B1864954A0B1F)  NOCV
```

System A can send this control statement to System B, which can then use it to create the Local BTOA key. The key value in this control statement is the clear value of the BTOA importer key. ICSF System A does not send this control statement to PCF System B over the network, because the key value is a clear key value. ICSF System A has a courier deliver the control statement to PCF System B.

The administrator at either system must change the ICSF control statement format into the PCF control statement format. The administrator can also use information from the key output data set to create the PCF control statement.

The control statement submitted at PCF System B would have the following syntax:

```
LOCAL BTOA,KEY=6F3463CA3FBC0626,IKEY=536B1864954A0B1F,ADD
```

The administrator at PCF System B submits the control statement to the PCF key generation utility program, which processes the control statement to generate the Local BTOA key. The Importer BTOA key at ICSF System A and the Local BTOA key at PCF System B are complementary keys.

Note: A single PCF key generation control statement can be used to generate both Remote and Local BTOA keys, also called a CROSS key pair.

```
CROSS BTOA,KEYLOC=6F3463CA3FBC0626,IKEYLOC=536B1864954A0B1F,
          KEYREM=B2403EF8125A036F,IKEYREM=239AC35A72941EF2,ADD
```

This procedure creates a pair of complementary transport keys for keys sent from PCF System B to ICSF System A. When PCF System B sends a key to ICSF System A, System B enciphers the key, using the Local BTOA key. When ICSF System A receives the key, ICSF System A deciphers the key, using the Importer BTOA key.

By these procedures, two pairs of complementary transport keys are established at each location so that the two systems can exchange keys.

Note: During these procedures, the special secure mode should be enabled while KGUP generates or receives clear key values.

Scenario of an ICSF System and 4758 PCI Cryptographic Coprocessor Establishing Initial Transport Keys

This scenario describes how an ICSF system and a 4758 PCI Cryptographic Coprocessor establish initial transport keys between themselves. They establish two pairs of complementary importer and exporter keys at each location, as shown in Figure 105.

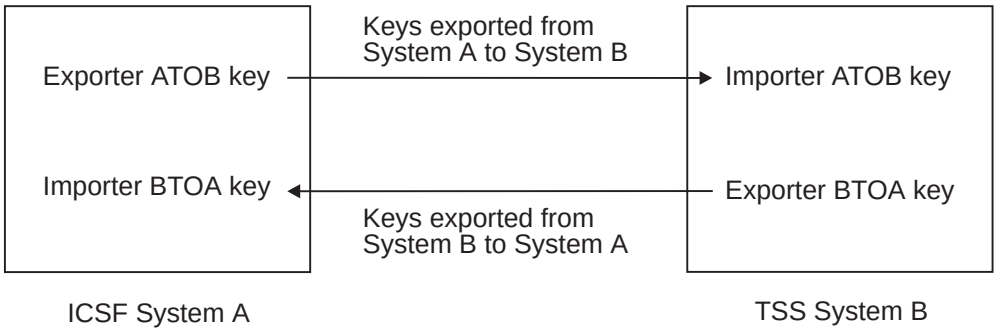


Figure 105. Key Exchange Establishment between a 4758 PCI Cryptographic Coprocessor System and an ICSF System

The systems can use these importer and exporter keys during key exchange. First, the ICSF System A administrator and the TSS System B administrator establish the complementary transport keys to send keys from ICSF System A to TSS System B. These keys are the Exporter ATOB key at System A and the Importer ATOB key at System B.

The ICSF administrator at System A submits the following control statement to System A's KGUP to create the Exporter ATOB key.

```
ADD  LABEL(ATOB)  TYPE(EXPORTER)  CLEAR
```

KGUP processes this control statement to generate the Exporter ATOB key and places the key in System A's CKDS. KGUP creates a record containing the clear key created for the system, and that record is written to the CSFKEYS data set. ICSF System A then sends this clear key to TSS System B. Because the key value is in the clear, System A has a courier deliver the key, rather than sending it over the network.

The TSS administrator at System B uses the Secure_Key_Import verb to import the ATOB importer key, because the key value is in the clear. The administrator can then use the Key_Record_Create and the Key_Record_Write verbs to place the key in TSS key storage. The ATOB exporter key at ICSF system A and the ATOB importer key at TSS System B are complementary keys.

This procedure creates a pair of complementary transport keys for keys sent from ICSF System A to TSS System B. When ICSF System A sends a key to TSS System B, it enciphers the key using the ATOB exporter key. When TSS System B receives the key, it deciphers the key using the ATOB importer key.

Next, the administrators at the two facilities establish the complementary transport keys to send keys from TSS System B to ICSF System A. These keys are the Importer BTOA key at ICSF System A and the Exporter BTOA key at TSS System B. The ICSF administrator at System A submits the following control statement to System A's KGUP to generate the Importer BTOA key.

```
ADD  LABEL(BTOA)  TYPE(IMPORTER)  TRANSKEY(ATOB)
```

KGUP processes this control statement to generate the Importer BTOA key and places the key in System A's CKDS. The ICSF System A administrator can send this key to the TSS System B over the network, because the key value is enciphered.

The TSS administrator at System B uses Key_Import, Key_Record_Create, and the Key_Record_Write verbs to import the key and place it in TSS key storage. The Importer BTOA key at System A and the Exporter BTOA key at System B are complementary keys.

This procedure creates a pair of complementary transport keys for keys sent from TSS System B to ICSF System A. When TSS System B sends a key to ICSF System A, TSS System B enciphers the key using the Exporter BTOA key. When ICSF System A receives the key, it deciphers the key using the Importer BTOA key.

Using these procedures two pairs of complementary transport keys are established at each location to allow key exchange between the two systems.

Notes:

1. During these procedures, the special secure mode must be enabled on ICSF while KGUP is generating or receiving clear key values, and the Secure_Key_Import verb must be enabled on TSS to receive clear keys.
2. The ICSF administrator at System A can submit in the same KGUP job both the ADD control statements meant for processing at System A.

Chapter 8. Viewing and Changing System Status

This chapter describes how to do the following:

You define installation options, and any installation exits and installation-defined callable services to ICSF. Using the ICSF panels, you can view how these options and programs are currently defined. During master key management, you change the status of the key storage registers that contain key parts and the master keys. You can use the ICSF panels to view the status of these hardware registers.You can also use the ICSF panels to deactivate or activate your PCI cryptographic coprocessors and accelerators.

When you check the status of an installation option, an installation exit, or an installation-defined callable service, you may decide to change how you defined the option or program. You must change the information in the installation options data set and restart ICSF to activate the change.

Displaying Administrative Control Functions

To display administrative control functions:

1. Select option 4, ADMINCNTL, on the primary menu panel.

```
CSF@PRIM ----- Integrated Cryptographic Service Facility  ---------
OPTION ===> 4

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

      Licensed Materials - Property of IBM

     5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 106. Primary Panel

The Administrative Control panel appears, which is shown in Figure 107.

```
CSFACF00 ------------- ICSF Administrative Control Functions
COMMAND ===>
          Active CKDS: CRYPTO25.HCR7704.CKDS
          Active PKDS: CRYPTO25.HCR7704.PKDS

To change the status of a control, enter the appropriate character
(E - ENABLE, D - DISABLE) and press ENTER.

         Function                                  STATUS
         --------                                  ------
.  Dynamic CKDS Access                             ENABLED
.  PKA Callable Services                           ENABLED
.  PKDS Read Access                                ENABLED
.  PKDS Write, Create, and Delete Access           DISABLED
```

Figure 107. Administrative Control Functions Panel

On this panel, you can view the following options and their values:

Dynamic CKDS Access (ENABLED or DISABLED)
Specifies whether the dynamic CKDS update services are currently enabled. You can enable or disable these services by placing an 'E' or 'D' before the function on this panel.

Value	Indication
ENABLED	The dynamic CKDS update services are enabled.
DISABLED	The dynamic CKDS update services are disabled.

PKA Callable Services (ENABLED or DISABLED)
Specifies whether the use of PKA callable services is currently enabled. You can enable or disable these services by placing an 'E' or 'D' before the function on this panel.

Value	Indication
ENABLED	PKA callable services are enabled.
DISABLED	PKA callable services are disabled.

PKDS Read Access (ENABLED or DISABLED)
Specifies whether the use of PKDS Read callable service is currently enabled. You can enable or disable this service by placing an 'E' or 'D' before the function on this panel.

Value	Indication
ENABLED	The PKDS Read callable service is enabled.
DISABLED	The PKDS Read callable service is disabled.

PKDS Write, Create, and Delete Access (ENABLED or DISABLED)
Specifies whether the use of PKDS Write, Create, and Delete callable services are currently enabled. You can enable or disable these services by by placing an 'E' or 'D' before the function on this panel.

Value	Indication
ENABLED	The PKDS Write, Create, and Delete callable services are enabled.
DISABLED	The PKDS Write, Create, and Delete callable services are enabled.

Displaying Coprocessor Status

Use the ICSF panels to view the status of the coprocessors. To display coprocessor status:

1. Select option 1, COPROCESSOR MGMT, on the Primary Option panel, as shown in Figure 108.

```
CSF@PRIM ----- Integrated Cryptographic Service Facility  ---------
OPTION ===> 1

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

       Licensed Materials - Property of IBM

     5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 108. Selecting for Coprocessor Status on the Primary Menu Panel

2. The Coprocessor Management panel appears. Refer to Figure 109.

   Attention: If you are running on an IBM @server zSeries 990, this panel has changed. See “TSO panels” on page 220.

```
 CSFCMP00 ---------------- ICSF Coprocessor Management -------------
 OPTION ===> 1

Select the coprocessors to be processed and press ENTER.
Action characters are: A, D, E, R, and S. See the help panel for details.

COPROCESSOR  MODULE ID/SERIAL NUMBER                  STATUS
-----------  -------------------------------          -------

_ A06                                                  ACTIVE
_ A07                                                  ACTIVE
_ C0         E589C396944007A6 5D40369997A386F4         ONLINE
_ C1         0AA379BFD2387960 0367DC04533125FF         ONLINE
_ P00        41-00YE1                                  ONLINE
_ P01        41-00K11                                  ONLINE
_ P02        41-0A355                                  ONLINE
_ P03        41-0BA3F                                  ONLINE
_ P04        41-0RT2T                                  DEACTIVATED
_ P05        41-00342                                  DISABLED
_
```

Figure 109. Coprocessor Management Panel

On this panel, you can view the following options and their values:

Coprocessor
The prefix indicates the type of cryptographic coprocessor. An A represents a PCI Cryptographic Accelerator. A C represents the Cryptographic Coprocessor Feature. A P represents the PCI Cryptographic Coprocessor.

Some servers allow you to partition the processor unit into two sides (side 0 and side 1). The individual central processors, processor storage arrays, and the channel subsystems are associated with side 0 or side 1. The unit on Side 0 is called Coprocessor C0, and the one on Side 1 is called Coprocessor C1.

Module ID/Serial Number
The module ID is the unique 128-bit value that was generated for the CCF during the manufacturing process.The serial number is a number for the PCI Cryptographic Coprocessor.

Status
This field displays the status of the PCICC, the PCICA and the CCF.

State	Indication
ACTIVE (PCICC)	The verification pattern for the SYM-MK matches the verification pattern of the DES master key on the server’s Cryptographic Coprocessor Feature. The hash pattern for the ASYM-MK matches the hash pattern of the Signature Master Key (SMK) register on the server’s Cryptographic Coprocessor Feature. Requests for services can then be routed to either cryptographic coprocessor.
ACTIVE (PCICA)	The PCICA is available for work.
ACTIVE (CCF)	The DES master key is valid.

ONLINE (PCICC)
The PCI Cryptographic Coprocessor is online, but one or both of the master key verification patterns or hash patterns do not match those of the server's Cryptographic Coprocessor Feature. Requests for services cannot be routed to the PCI Cryptographic Coprocessor.

ONLINE (CCF)
The DES master key is not valid.

OFFLINE (PCICC and PCICA)
A PCICC or PCICA may be physically present but it is not available to the operating system. Either it has never been configured online or it has been configured offline by an operator command from the hardware support element.

DISABLED (PCICC and CCF)
The PCI Cryptographic Coprocessor or the Cryptographic Coprocessor Feature has been disabled by the TKE workstation.

DEACTIVATED (PCICC and PCICA)
The PCI Cryptographic Coprocessor or the PCI Cryptographic Accelerator has been deactivated from the Coprocessor Management panel.

TEMP UNAVAILABLE (PCICC and PCICA)
An unexpected error has been returned from the card. The system goes into recovery to try to reset the card. If the reset is successful, the card is usable again. The user will have to press ENTER to refresh the status.

HARDWARE ERROR (PCICC and PCICA)
The reset from a TEMP UNAVAILABLE condition was not successful and the card is unusable.

HARDWARE ERROR (CCF)
A hardware error has been detected.

UNKNOWN: CODE = cccc/ssss (PCICC)
The PCICC has returned an unrecognizable code in response to an attempt to determine its status. The return/reason code appears as the value of CODE.

Changing Coprocessor Status

You can change the status of your PCI cryptographic coprocessors and accelerators, either activating or deactivating them. From the primary menu, select option 1, COPROCESSOR MGMT, and the Coprocessor Management panel is displayed (Figure 110 on page 162).

```
CSFCMP00 ---------------- ICSF Coprocessor Management -------------
OPTION ===> 1

Select the coprocessors to be processed and press ENTER.
Action characters are: A, D, E, R, and S. See the help panel for details.

COPROCESSOR  MODULE ID/SERIAL NUMBER                 STATUS
-----------  ------------------------------          -------

D A06                                                 ACTIVE
_ A07                                                 ACTIVE
_ C0         E589C396944007A6 5D40369997A386F4        ONLINE
_ C1         0AA379BFD2387960 0367DC04533125FF        ONLINE
_ P00        41-00YE1                                 ONLINE
_ P01        41-00K11                                 ONLINE
_ P02        41-0A355                                 ONLINE
_ P03        41-0BA3F                                 ONLINE
_ P04        41-0RT2T                                 DEACTIVATED
_ P05        41-00342                                 DISABLED
_
```

Figure 110. Coprocessor Management Panel

There are action characters that can be entered on the left of the PCI coprocessor or accelerator number.

Character	Indication
D	Makes a PCICC or PCICA unavailable. The status becomes DEACTIVATED. When the request is made, the status of the PCICC/PCICA may be anything except OFFLINE.
A	Makes available a PCICC or PCICA previously deactivated by a 'D' action character. When the request is made, if the PCICC is online and the master keys are correct, the status will be ACTIVE. If the master keys are incorrect, the status will be ONLINE. When the request is made, the status of the PCICA is ACTIVE.

Displaying Coprocessor Hardware Status

You can use the ICSF panels to view the status of the cryptographic coprocessor key registers, the PCI cryptographic coprocessor, the master key verification patterns, and other information about the cryptographic hardware.

When you enter and activate a DES master key, you change the status of the registers. The cryptographic facility contains several key registers. The master key register contains the active DES master key. For the CCF, the auxiliary key register contains either the old DES master key or a new DES master key before it is activated and transferred to the master key register. For the PCICC, there are three registers: one for the old master key, one for the new and one for the current. When you have a PCICC, the old master key is not lost when a new master key is loaded.

In addition, there are also registers for the PKA master keys. When you enter a master key, the Cryptographic Coprocessor Feature or the PCI Cryptographic Coprocessor calculates a verification pattern and a hash pattern for the master key. You can use these patterns to identify master keys.

You can use the panels to display the conditions of the key registers and the verification pattern and hash patterns for the master keys. You may use this information for master key management.

To display coprocessor hardware status:

1. From the Coprocessor Management panel, select the coprocessors to be processed by typing an 'S'.

```
 CSFCMP00 ---------------- ICSF Coprocessor Management -------------
 OPTION ===> 1

Select the coprocessors to be processed and press ENTER.
Action characters are: A, D, E, R, and S. See the help panel for details.

COPROCESSOR  MODULE ID/SERIAL NUMBER                  STATUS
-----------  -------------------------------          -------

_ A06                                                  ACTIVE
_ A07                                                  ACTIVE
S C0         E589C396944007A6 5D40369997A386F4         ACTIVE
_ C1         0AA379BFD2387960 0367DC04533125FF         ONLINE
S P00        41-00YE1                                  ONLINE
_ P01        41-00K11                                  ONLINE
_ P02        41-0A355                                  ACTIVE
_ P03        41-0BA3F                                  ONLINE
_ P04        41-0RT2T                                  DEACTIVATED
_ P05        41-00342                                  DISABLED
_
```

Figure 111. Selecting the coprocessor on the Coprocessor Management Panel

2. The Coprocessor Hardware Status panel appears (Figure 112 on page 164). When more than two coprocessors are requested, the status display can be scrolled left and right to show the other coprocessors. You can scroll to the left using PFKey 10 and to the right with PFKey 11.

```
CSFMKP10 ----------- ICSF - Coprocessor Hardware Status  ----------------
OPTION ===>

                                                    CRYPTO DOMAIN: 0

REGISTER STATUS                COPROCESSOR C0        COPROCESSOR P01
                                                     More:    +
Crypto Serial Number or         :  E589C39694407A60   41-00YE1
     Module Id                  :  5D40C39997A396F0
Status                          :  ACTIVE             ONLINE
DES/Symmetric-Keys Master Key
  New master key register       : FULL                PART FULL
    Verification pattern        : 1972BB5791BB2430    2342352352352352
    Hash pattern                : 0123456789ABCDEF    A17B93C44D24681A
                                : 9691BDA1970BDAA2    806427AAC91221CC
  Old master key register       : EMPTY               EMPTY
    Verification pattern        :
    Hash pattern                :
                                :
  Current master key register   : VALID               VALID
    Verification pattern        : CA6B408A02371B1D    261AAB8A02371705
    Hash pattern                : 41DF774FF81547D0    562A5202F8154331
                                : 090ABC4539727511    4093990AB1202451
PKA Signature/Asymmetric-Keys Master Key
  New master key register       : N/A                 PART FULL
    Hash pattern                :                     234235236236234D
                                :                     5678567856785678
  Old master key register       : N/A                 EMPTY
    Hash pattern                :
                                :
  Current master key register   : VALID               VALID
    Hash pattern                : 9691BDA1970BDAA2    9691BDA1970BDAA2
                                : 1972BB5791BB2430    1972BB5791BB2430
PKA Key Management Master Key register
    Hash pattern                : 123412341241234D    N/A
                                : 5678567856785678
Special Secure Mode             : Enabled             N/A
Environment Control Mask        : FBFEFCF0            N/A
Crypto Configuration Control    : EF569412CD91AB78    N/A
                                : 1F25A78BC88ED77A


 Press ENTER to refresh the hardware status display.
 Press END   to exit to the previous menu.
```

Figure 112. Coprocessor Hardware Status Panel

The coprocessor hardware status fields on this panel contain the following information:

CRYPTO DOMAIN
This field displays the value that is specified for the DOMAIN keyword in the installation options data set at ICSF startup. This is the domain in which your system is currently working. It specifies which one of several separate sets of master key registers you can currently access. A system programmer can use the DOMAIN keyword in the installation options data set to specify the domain value to use at ICSF startup. For more information about the DOMAIN installation option, see page 172.

Crypto Serial Number or Module ID
The serial number is a number for the PCI Cryptographic Coprocessor. The module ID is the unique 128-bit value that was generated for the CCF during the manufacturing process.

Status

This field displays the status of the CCF and the PCICC.

State	Indication
ACTIVE (PCICC)	The verification pattern for the SYM-MK matches the verification pattern of the DES master key on the server's Cryptographic Coprocessor Feature. The hash pattern for the ASYM-MK matches the hash pattern of the Signature Master Key (SMK) register on the server's Cryptographic Coprocessor Feature. Requests for services can then be routed to either cryptographic coprocessor.
ACTIVE (CCF)	The DES master key is valid.
ONLINE (PCICC)	The PCI Cryptographic Coprocessor is online, but one or both of the master key verification patterns or hash patterns do not match those of the server's Cryptographic Coprocessor Feature. Requests for services cannot be routed to the PCI Cryptographic Coprocessor.
ONLINE (CCF)	The DES master key is not valid.

DES/Symmetric-Keys Master KEY

New Master Key Register

This field shows the state of the new master key register.

This key register can be in any of the following states:

State	Indication
EMPTY	You have not entered any key parts for the initial master key, or you have just transferred the contents of this register into the master key register. Or you have RESET the registers. Or you have zeroized the domain from a TKE workstation or the Support Element.
PART FULL	You have entered one or more key parts but not the final key part.
FULL	You have entered an entire new master key, but have not transferred it to the master key register yet.

For the CCF, the new master key is held in an auxiliary key register. This auxiliary key register can contain either a new master key or an old master key. Therefore, a new master key and the old master key cannot coexist.

For the PCICC, there can be an old, new and current master key.

Verification Pattern

When you use the master key panels to enter a new master key, *record the verification pattern* that appears for the master key after the final key part has been entered. You can compare the verification pattern you record with this one to ensure that the key entered and the key in the new master key register are the same.

If your system is using multiple cryptographic coprocessors, you must enter the same master key into all units. If the status of the new master key

registers are valid, the NMK verification patterns for each unit should match, because the patterns verify the same key.

Hash Pattern

If the master key register is not EMPTY, the panel displays a hash pattern for the key. When you enter a new master key, record the hash pattern that appears on the panel. When the master key becomes active, you can compare the hash patterns to ensure that the one you entered and set is in the master key register.

If your system is using multiple cryptographic coprocessors, you enter the same master key into all units. If the status of the new master key registers are valid, the master key register hash patterns for each unit should match, because the patterns verify the same key.

Old Master Key register

This field shows the states of the DES and symmetric keys old master key register.

State	Indication
EMPTY	You have never changed the master key and, therefore, never transferred a master key to the old master key register. Or you have RESET the registers. Or you have zeroized the domain from a TKE workstation or the Support Element.
VALID	You have changed the master key. The master key that was current when you changed the master key was placed in the old master key register.

For the CCF, the old/new master key register is actually the auxiliary master key register. The auxiliary master key register can contain either the new master key or the old master key; therefore a new master key and an old master key cannot coexist at the same time. If an old master key exists, it is lost when you enter a new one.

For the PCICC, there can be an old, new and current master key.

Verification Pattern

When you use the master key panels to enter a new master key, *record the verification pattern* that appears for the master key after the final key part has been entered. You can compare the verification pattern you record with this one to ensure that the key entered and the key in the new master key register are the same.

If your system is using multiple cryptographic coprocessors, you must enter the same master key into all units. If the status of the new master key registers are valid, the DES verification patterns for each unit should match, because the patterns verify the same key.

Hash Pattern

If the master key register is not EMPTY, the panel displays a hash pattern for the key. When you enter a new master key, record the hash pattern that appears on the panel. When the master key becomes active, you can compare the hash patterns to ensure that the one you entered and set is in the master key register.

If your system is using multiple cryptographic coprocessors, you enter the same master key into all units. If the status of the new master key registers

are valid, the master key register hash patterns for each unit should match, because the patterns verify the same key.

Current Master Key register
This field shows the states of the DES and symmetric-keys master key register.

State	Indication
EMPTY	You have never entered and set an initial DES/symmetric-keys master key on the coprocessor. Or you have RESET the registers. Or you have zeroized the domain from a TKE workstation or the Support Element.
VALID	You have entered a new PKA or asymmetric-keys master key on this coprocessor and chosen either the set or change option.

Verification Pattern
When you use the master key panels to enter a new master key, *record the verification pattern* that appears for the master key after the final key part has been entered. You can compare the verification pattern you record with this one to ensure that the key entered and the key in the new master key register are the same.

If your system is using multiple cryptographic coprocessors, you must enter the same master key into all units. If the status of the new master key registers are valid, the NMK verification patterns for each unit should match, because the patterns verify the same key.

Hash Pattern
If the master key register is not EMPTY, the panel displays a hash pattern for the key. When you enter a new master key, record the hash pattern that appears on the panel. When the master key becomes active, you can compare the hash patterns to ensure that the one you entered and set is in the master key register.

If your system is using multiple cryptographic coprocessors, you enter the same master key into all units. If the status of the new master key registers are valid, the master key register hash patterns for each unit should match, because the patterns verify the same key.

PKA Signature/Asymmetric-Keys Master Key

New Master Key register (PCICC only)
This field shows the state of the asymmetric-keys new master key register.

This key register can be in any of the following states:

State	Indication
EMPTY	You have not entered any key parts for the initial asymmetric-keys master key, or you have just transferred the contents of this register into the asymmetric-keys master key register. Or you have RESET the registers. Or you have zeroized the domain from a TKE workstation or the Support Element.
PART FULL	You have entered one or more key parts but not the final key part.

Hash Pattern
If the master key register is not EMPTY, a hash pattern is displayed.

Old Master Key register (PCICC only)
This field shows the states of the asymmetric keys old master key register.

State	Indication
EMPTY	You have never changed the asymmetric-keys master key and, therefore, never transferred an asymmetric-keys master key to the asymmetric-keys old master key register. Or you have RESET the registers. Or you have zeroized the domain from a TKE workstation or the Support Element.
VALID	You have changed the asymmetric-keys master key. The asymmetric-keys master key that was current when you changed the master key was placed in the asymmetric-keys old master key register.

Hash Pattern
If the old asymmetric-keys master key register is valid, the panel displays a hash pattern for the asymmetric-keys old master key.

Current Master Key register
This field shows the states of the PKA signature master key and asymmetric-keys master key register.

State	Indication
EMPTY	You have never entered an initial PKA signature master key or an asymmetric-keys master key on the coprocessor. Or you have RESET the registers. Or you have zeroized the domain from a TKE workstation or the Support Element.
VALID	You have entered a new PKA signature master key or asymmetric-keys master key on this coprocessor.

Hash Pattern
If the PKA signature master key and asymmetric-keys master key registers are valid, the panel displays a hash pattern for the key. When you enter a new PKA signature master key and asymmetric-keys master key, *record the hash pattern* that appears on the panel. When the PKA signature master key and asymmetric-keys master key becomes active, you can compare the hash patterns to ensure that the one you entered and set is in the master key register.

If your system is using other PCI Cryptographic Coprocessors and one or more Cryptographic Coprocessor Features, the asymmetric-keys master key must be the same on all the PCI cards, and must also be the same as the Signature master key in the Cryptographic Coprocessor Feature. If the status of all these cryptographic coprocessors is valid, the MK hash patterns for each unit should match, because the patterns verify the same key.

Note: An audit trail of the hash patterns that the PCI Cryptographic Coprocessor calculates appears in SMF record type 82.

PKA Key Mangement Master Key register (CCF only)

Hash pattern
You have entered a PKA key management master key and the hash pattern for the key register is shown here.

Special Secure Mode (CCF only)
This field shows if the special secure mode is enabled or disabled. Special

secure mode is a lower form of security. This mode allows you to use KGUP to enter clear keys, produce clear PINs, use the secure key import callable service, and initialize the CKDS. Special secure mode is enabled automatically when you send a KGUP request, provided that the SSM installation option is set to YES.

Environment Control Mask (CCF only)
The environment control mask contains controls for a subset of the components for each domain. This field shows the value of this control.

Note: Selected bits can be changed by the TKE workstation.

Crypto Configuration Control (CCF only)
The crypto configuration control contains controls to enable and disable all the major components of the crypto modules. This field shows the value of this control.

See Appendix A, “CCC Bit Assignments”, on page 205 for some selected values.

Note: The CCC cannot be changed.

Displaying Installation Options

Installation options enable you to specify certain modes and conditions to ICSF. For example, if your installation specifies YES for the SSM option, you can enable special secure mode. You specify installation options in the installation options data set. The ICSF startup procedure, specifies the installation options data set to be used for that start of ICSF. The options become active, when you start ICSF. You can use the panels to view each installation option and its current value.

To display installation options:

1. Select option 3, OPSTAT, on the Primary Option panel, as shown in Figure 113 on page 170.

```
CSF@PRIM ----- Integrated Cryptographic Service Facility  ---------
OPTION ===> 3

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

      Licensed Materials - Property of IBM

     5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 113. Selecting the Installation Options on the Primary Menu Panel

The Installation Options panel appears. Refer to Figure 114.

```
CSFSOP00 ----------- ICSF - Installation Options --------------
Enter the number of the desired option above.

  1  OPTIONS  -  Display Installation Options
  2  EXITS    -  Display Installation exits and exit options
  3  SERVICES -  Display Installation Defined Services
```

Figure 114. Installation Options Panel

2. Select option 1, Options, on the Installation Options panel.

   The Installation Option Display panel, which is shown in Figure 115 on page 171, appears.

```
CSFSOP10 ----------- ICSF - Installation Option Display  ROW 1 TO 14 OF 15
COMMAND ===>                                                SCROLL ===> PAGE
       Active CKDS: CRYPTOR2.HCRP230.CKDS
       Active PKDS: CRYPTOR2.HCRP230.PKDS
  OPTION                                                 CURRENT VALUE
 --------                                                -------------
  CHECKAUTH    RACF check authorized callers                 YES
  COMPAT       Allow CUSP/PCF Compatibility                  NO
  COMPENC      Compatibility services encryption algorithm   DES
  DOMAIN       Current domain index or usage domain index    0
  KEYAUTH      Key Authentication in effect                  YES
  SSM          Allow Special Secure Mode                     YES
  TRACEENTRY   Number of trace entries active                599
  USERPARM     User specified parameter data                 USERPARM
  REASONCODES  Source of callable services reason codes      ICSF
  WAITLIST     Source of CICS Wait List if CICS installed    dataset
  PKDSCACHE    PKDS Cache size in records                    32

                Encryption algorithm available                DES, CDMF
****************************** BOTTOM OF DATA ********************************
```

Figure 115. Installation Options Display Panel

This panel displays the keyword for each installation option, a brief description, and the current value of the option. For example, the PKDSCACHE option, which defines the size of the PKDS Cache in records, is currently set at 32.

You may want to change the current value of an installation option. To change and activate an installation option, you must change the option value in the installation options data set and restart ICSF. For integrity reasons, a change of the DOMAIN option also requires a re-IPL of MVS. For a complete description of these installation options and the installation options data set, see *z/OS ICSF System Programmer's Guide*.

The installation options data set that the system uses at ICSF startup contains keywords and their values which specify certain installation options. On this panel, you can view the following options and their values:

Active CKDS: (data-set-name)
This specifies the name of the CKDS the system uses during the startup of ICSF. On the Installation Options Display panel, this data set name is called the active CKDS.

Active PKDS: (data-set-name)
This specifies the name of the PKDS the system uses during the startup of ICSF.

CHECKAUTH(YES or NO)
Indicates whether ICSF performs access control checking of Supervisor State and System Key callers. If you do not specify the CHECKAUTH option, the default is CHECKAUTH(NO).

Value	Indication
YES	ICSF checks Supervisor State and System Key callers.
NO	ICSF does not check Supervisor State and System Key callers, resulting in significant performance enhancement for applications that use ICSF callable services.

COMPAT(YES, NO, or COEXIST)
Indicates whether ICSF is running in compatibility mode, noncompatibility mode,

or coexistence mode with the Cryptographic Unit Support Program (CUSP) or Programmed Cryptographic Facility (PCF). If you do not specify the COMPAT option, the default value is COMPAT(NO).

Value	Indication
YES	ICSF is running in compatibility mode, which means you can run CUSP and PCF applications on ICSF because ICSF supports the CUSP and PCF macros in this mode. You do not have to reassemble CUSP and PCF applications to do this. However, you cannot start CUSP or PCF at the same time as ICSF on the same MVS system.
NO	ICSF is running in noncompatibility mode, which means that you run CUSP applications on CUSP, PCF applications on PCF, and ICSF applications on ICSF. You cannot run CUSP or PCF applications on ICSF, because ICSF does not support the CUSP and PCF macros in this mode. You can start CUSP or PCF at the same time as ICSF on the same z/OS operating system. You can start ICSF and then start CUSP or PCF or you can start CUSP or PCF and then start CSF. You should use noncompatibility mode unless you are migrating from CUSP or PCF to ICSF.
COEXIST	ICSF is running in coexistence mode. In this mode you can run a CUSP or PCF application on CUSP or PCF, or you can reassemble the CUSP or PCF application to run on ICSF. To do this, you reassemble the application against coexistence macros that are shipped with ICSF. In this mode, you can start CUSP or PCF at the same time as ICSF on the same MVS system.

COMPENC(DES or CDMF)

On a system where both DES and CDMF encryption algorithms are available, the COMPENC setting indicates the encryption algorithm for the PCF/CUSP compatibility CIPHER macro.

Value	Indication
DES	The CIPHER macro uses the DES encryption algorithm.
CDMF	The CIPHER macro uses the CDMF encryption algorithm.

DOMAIN(n)

Allows you to access one of several separate sets of master key registers. Each domain contains the following master key registers:

- A master key register that contains the active DES master key
- For the CCF, there is an auxiliary DES master key register that holds either the old or new master key
- If you have a PCICC, there are symmetric master key registers that hold both the old and new master key
- A PKA key management master key register
- A PKA signature master key register
- If you have PCICC, there are ASYM-MK registers for the new, old, and current master key.

You can use domains to have separate master keys for different purposes.

You can use domains in basic mode or with PR/SM logical partition (LPAR) mode. In basic mode, you access only one domain at a time. You can specify a different master key in each domain. For example, you might have one master key for production operations and a different master key for test operations. In

LPAR mode, you can have a different domain for each partition. The number you specify is the number of the domain to be used for this start of ICSF.

Beginning in z/OS V1 R2, 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.

You use the Crypto page of the Customize Activation Profile to assign a usage domain index (0 to 15) to a logical partition and enable cryptographic functions. The DOMAIN number you specify in the installation options data set while running in a partition must be the same number as the usage domain index specified for the partition on the Crypto page. For more information about logical partitions, see *PR/SM Planning Guide*.

To change and activate the other installation options, you must restart ICSF. In compatibility or coexistence mode, to change and activate the DOMAIN option, you must also re-IPL MVS. A re-IPL ensures that a program does not use a key that has been encrypted under a different master key to access a cryptographic service.

KEYAUTH(YES or NO)
Indicates whether or not ICSF should authenticate a key entry after it retrieves one from the in-storage cryptographic key data set. If you do not specify the KEYAUTH option, the default value is KEYAUTH(NO).

Value	Indication
YES	ICSF authenticates the keys. ICSF generates a message authentication code (MAC) for each key entry in the CKDS whenever it creates or updates the key entry. ICSF also performs a MAC verification to ensure that the entry was not changed.
NO	ICSF does not authenticate keys retrieved from the in-storage CKDS. ICSF gains a small enhancement of performance.

SSM(YES or NO)
Indicates whether or not an installation can ever enable special secure mode during the running of ICSF. This mode lowers the security of your system. It allows you to input clear keys by using KGUP, produce clear PINs, use the Secure Key Import callable service and the initial use of Pass Phrase. If you do not specify the SSM option, the default value is SSM(NO).

Value	Indication
YES	Special secure mode is enabled. For z/OS ICSF, SSM(YES) must be specified in order to use KGUP, Secure Key Import callable service, Clear PIN Generate and the initial use of Pass Phrase.
NO	You cannot enable the special secure mode.

TRACEENTRY(n)
Specifies the number, `n`, of trace buffers to allocate for ICSF tracing. `n` is a decimal value. The range of valid values is 100 through 10000.

If you do not specify the TRACEENTRY option, the default value is TRACEENTRY(1000).

USERPARM(value)
Displays the value of an 8-byte field that is defined for installation use. ICSF

stores this value in the CCVT_USERPARM field of the Cryptographic Communication Vector Table (CCVT). An application program or installation exit can examine this field and use it to set system environment information.

REASONCODES(ICSF or TSS)
Specifies which set of reason codes the application interface returns.

Value	Indication
ICSF	ICSF reason codes are returned.
TSS	TSS reason codes are returned.

ICSF is the default.

WAITLIST(value)
Displays the current value of the WAITLIST option. If WAITLIST is coded, the value will be 'dataset' and a second line will contain the name of the specified Wait List data set. If WAITLIST is not coded, the value will be 'default'. If the data set specified by the WAITLIST option cannot be allocated or opened, the value will also be 'default'.

PKDSCACHE(n)
Defines the size of the PKDS Cache in records. The PKDS cache improves performance as it facilitates access to frequently used records. Specify *n* as a decimal value from 0 to 256. If *n* is zero, no cache will be implemented. The default value is 64. PKDSCACHE can be used on OS/390 V2 R10 by applying APAR OW48568.

Encryption algorithm available (DES or DES,CDMF, or CDMF)
Specifies the type of encryption algorithm that is permitted on the current system. This value *cannot* be set or changed; it can only be displayed.

Value	Indication
DES	Only the DES encryption algorithm is available.
DES, CDMF	Both the DES and CDMF encryption algorithms are available.
CDMF	Only the CDMF encryption algorithm is available.

For more information about the ICSF startup procedure and installation options, see *z/OS ICSF System Programmer's Guide*. At any time while you are running ICSF, you can check the current value of these installation options.

The installation exits and installation-defined callable services are also specified in the installation options data set, but they are not displayed on this panel. For a description of how to display the installation exit information, see "Displaying Installation Exits" on page 176. For a description of how to display installation-defined callable service information, see "Displaying Installation-Defined Callable Services" on page 183.

Displaying PCICC Default Roles

Use the ICSF panels to display the default role for the coprocessor. All the access control points enabled will be listed.

1. Select option 1, COPROCESSOR MGMT, on the Primary Option panel, as shown in Figure 116 on page 175.

```
CSF@PRIM ----- Integrated Cryptographic Service Facility  ---------
OPTION ===> 1

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

      Licensed Materials - Property of IBM

     5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 116. Selecting for Coprocessor Status on the Primary Menu Panel

2. The Coprocessor Management panel appears. Refer to Figure 117.

```
CSFCMP00 ---------------- ICSF Coprocessor Management -------------
OPTION ===> 1

Select the coprocessors to be processed and press ENTER.
Action characters are: A, D, E, R, and S. See the help panel for details.

COPROCESSOR  MODULE ID/SERIAL NUMBER                 STATUS
-----------  -------------------------------         -------

_ A06                                                 ACTIVE
_ A07                                                 ACTIVE
_ C0         E589C396944007A6 5D40369997A386F4        ONLINE
_ C1         0AA379BFD2387960 0367DC04533125FF        ONLINE
_ P00        41-00YE1                                 ONLINE
R P01        41-00K11                                 ONLINE
_ P02        41-0A355                                 ONLINE
_ P03        41-0BA3F                                 ONLINE
_ P04        41-0RT2T                                 DEACTIVATED
_ P05        41-00342                                 DISABLED
_
```

Figure 117. Coprocessor Management Panel

3. Select the PCICC by entering an 'R' to the left of the coprocessor. Press enter and the Status Display panel appears (Figure 118 on page 176).

   Note: The default role can be changed with a TKE workstation. See *z/OS ICSF TKE Workstation User's Guide 2000*.

```
CSFCMP30 ---------------- ICSF Status Display -------------
COMMAND ===>

Enabled access control points from the default role for P01, domain 0.

Clear New ASYM Master Key Register
Clear New SYM Master Key Register
Clear PIN Generate Alternate - 3624 Offset
Combine ASYM Master Key Parts
Combine SYM Master Key Parts
DES Key Token Change
Digital Signature Generate
Digital Signature Verify
Encrypted PIN Generate - 3624
Encrypted PIN Generate - GBP
Encrypted PIN Translate - Translate
Encrypted PIN Verify - 3624
Encrypted PIN Verify - GBP
Key Export
Key Generate - OPIM, OPEX, IMEX, etc.
Key Generate - EX, IM, OP
Key Generate - OPIM, OPEX, IMEX, etc.
Key Generate - SINGLE-R
Key Import
Load First ASYM Master Key Part
Load First SYM Master Key Part
PKA Decrypt
PKA Encrypt
PKA Key Generate
PKA Key Import - General
Prohibit Export
Retained Key Delete
Retained Key List
Set ASYM Master Key
SET Block Compose
SET Block Decompose
Symmetric Key Export - PKCS-1.2
Symmetric Key Export - ZERO-PAD
Symmetric Key Generate - PKCS-1.2
Symmetric Key Generate - ZERO-PAD
Symmetric Key Import - PKCS-1.2
Symmetric Key Export - ZERO-PAD
UDX 0x8001
```

Figure 118. Default Role Status Display Panel

Displaying Installation Exits

ICSF provides invocation points where you can use installation exits to perform processing that is specific to your installation. For example, ICSF provides a preprocessing and postprocessing exit invocation for each ICSF callable service. You can write and define an exit to set return codes at postprocessing of a callable service.

You must define each installation exit in the installation options data set. You define the ICSF name for the exit, the load module name of the exit, and the action ICSF

takes if the exit fails. You can use the panels to view the ICSF name for each exit invocation. For a defined exit, you view the exit's load module name and fail options.

ICSF provides the following types of exits:
- ICSF mainline exits
- Key generator utility program exit
- Callable services exits
- Cryptographic Key Data Set (CKDS) Conversion program exit
- Single-record, read-write exit
- CKDS retrieval exit
- Security exits

The mainline exits are called when you start and stop ICSF. The key generator utility program exit is called during key generator utility program processing. The callable services exits are called during each of the callable services. The CKDS conversion program exit is called during conversion of CUSP or PCF CKDS to ICSF CKDS format. The single-record, read-write exit is called when an access to a single record is made to a disk copy of the CKDS. The security exits are called during initialization and stopping of ICSF, during a call to a callable service, and during access of a CKDS entry.

For a detailed description of the ICSF exits, see *z/OS ICSF System Programmer's Guide*.

To display installation exits:

1. Select option 3, OPSTAT, on the Primary Option panel, as shown in Figure 119.

```
CSF@PRIM ---- Integrated Cryptographic Service Facility ---------
OPTION ===> 3

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

       Licensed Materials - Property of IBM

      5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
      US Government Users Restricted Rights - Use, duplication or
      disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 119. Selecting the Installation Options and Hardware Status Option on the Primary Menu Panel

The Installation Options panel appears. Refer to Figure 120 on page 178.

```
CSFSOP00 ----------- ICSF - Installation Options --------
OPTION ===> 2

Enter the number of the desired option above.

  1  OPTIONS  -  Display Installation Options
  2  EXITS    -  Display Installation exits and exit options
  3  SERVICES -  Display Installation Defined Services
```

Figure 120. Installation Options Panel

2. Select option 2, Exits, on the Installation Options panel.
   The first of the Installation Exits Display panels appears. Refer to Figure 121.

```
CSFSOP30 ------ ICSF - Installation Exits Display ---- ROW 1 TO 18 OF 70
COMMAND ===>

ICSF NAME      LOAD MODULE      OPTIONS
-------------  -----------      -------
 CSFAEGN                        *** No Exit Name was specified ***
 CSFAKEX                        *** No Exit Name was specified ***
 CSFAKIM                        *** No Exit Name was specified ***
 CSFAKTR                        *** No Exit Name was specified ***
 CSFATKN                        *** No Exit Name was specified ***
 CSFCKDS                        *** No Exit Name was specified ***
 CSFCKI                         *** No Exit Name was specified ***
 CSFCKM                         *** No Exit Name was specified ***
 CSFCONVX                       *** No Exit Name was specified ***
 CSFCPA                         *** No Exit Name was specified ***
 CSFCPE                         *** No Exit Name was specified ***
 CSFCSG                         *** No Exit Name was specified ***
 CSFCSV                         *** No Exit Name was specified ***
 CSFCTT                         *** No Exit Name was specified ***
 CSFCTT1                        *** No Exit Name was specified ***
 CSFCVE                         *** No Exit Name was specified ***
 CSFCVT                         *** No Exit Name was specified ***
 CSFDCO                         *** No Exit Name was specified ***
 CSFDEC                         *** No Exit Name was specified ***
 CSFDEC1                        *** No Exit Name was specified ***
 CSFDKG                         *** No Exit Name was specified ***
 CSFDKM                         *** No Exit Name was specified ***
 CSFDKX                         *** No Exit Name was specified ***
 CSFDSG                         *** No Exit Name was specified ***
 CSFDSV                         *** No Exit Name was specified ***
 CSFDVPI                        *** No Exit Name was specified ***
 CSFECO                         *** No Exit Name was specified ***
 CSFEDC         USEREDC         NONE - Take no action, if this exit fails
```

Figure 121. First Installation Exits Display Panel

   The Installation Exits Display panel displays the ICSF name for all the possible installation exits your installation can write.

3. Scroll through the screens, to view all of the installation exits.
   The second panel of exits is shown in Figure 122 on page 179.

```
CSFSOP30 ------ ICSF - Installation Exits Display --- ROW 19 TO 36 OF 70
COMMAND ===>

ICSF NAME      LOAD MODULE      OPTIONS
-------------  -----------      -------
 CSFEMK                         *** No Exit Name was specified ***
 CSFENC                         *** No Exit Name was specified ***
 CSFENC1                        *** No Exit Name was specified ***
 CSFEPG                         *** No Exit Name was specified ***
 CSFESECI                       *** No Exit Name was specified ***
 CSFESECK                       *** No Exit Name was specified ***
 CSFESECS                       *** No Exit Name was specified ***
 CSFESECT                       *** No Exit Name was specified ***
 CSFEXIT2                       *** No Exit Name was specified ***
 CSFEXIT3                       *** No Exit Name was specified ***
 CSFEXIT4                       *** No Exit Name was specified ***
 CSFEXIT5                       *** No Exit Name was specified ***
 CSFGKC                         *** No Exit Name was specified ***
 CSFKEX                         *** No Exit Name was specified ***
 CSFKGN                         *** No Exit Name was specified ***
 CSFKGUP                        *** No Exit Name was specified ***
 CSFKIM                         *** No Exit Name was specified ***
 CSFKPI                         *** No Exit Name was specified ***
 CSFKRC                         *** No Exit Name was specified ***
 CSFKRD                         *** No Exit Name was specified ***
 CSFKRR                         *** No Exit Name was specified ***
 CSFKRW                         *** No Exit Name was specified ***
 CSFKTC                         *** No Exit Name was specified ***
 CSFKTR                         *** No Exit Name was specified ***
 CSFKYT                         *** No Exit Name was specified ***
 CSFKYTX                        *** No Exit Name was specified ***
 CSFMDG                         *** No Exit Name was specified ***
```

Figure 122. Second Installation Exits Display Panel

The third panel of exits is shown in Figure 123 on page 180.

```
CSFSOP30 ----- ICSF - Installation Exits Display ---- ROW 37 TO 54 OF 70
COMMAND ===>

ICSF NAME  LOAD MODULE   OPTIONS
-------------  -----------   -------
 CSFMDG1                      *** No Exit Name was specified ***
 CSFMGN                       *** No Exit Name was specified ***
 CSFMGN1                      *** No Exit Name was specified ***
 CSFMVR                       *** No Exit Name was specified ***
 CSFMVR1                      *** No Exit Name was specified ***
 CSFOWH                       *** No Exit Name was specified ***
 CSFOWH1                      *** No Exit Name was specified ***
 CSFPCI                       *** No Exit Name was specified ***
 CSFPEX                       *** No Exit Name was specified ***
 CSFPEXX                      *** No Exit Name was specified ***
 CSFPGN                       *** No Exit Name was specified ***
 CSFPKD                       *** No Exit Name was specified ***
 CSFPKE                       *** No Exit Name was specified ***
 CSFPKG                       *** No Exit Name was specified ***
 CSFPKI                       *** No Exit Name was specified ***
 CSFPKRC                      *** No Exit Name was specified ***
 CSFPKRD                      *** No Exit Name was specified ***
 CSFPKRR                      *** No Exit Name was specified ***
 CSFPKRW                      *** No Exit Name was specified ***
 CSFPKSC                      *** No Exit Name was specified ***
 CSFPKTC                      *** No Exit Name was specified ***
 CSFPKX                       *** No Exit Name was specified ***
 CSFPTR                       *** No Exit Name was specified ***
 CSFPVR                       *** No Exit Name was specified ***
 CSFRKD                       *** No Exit Name was specified ***
 CSFRKL                       *** No Exit Name was specified ***
 CSFRNG        EXITRNG        EXIT - Do not call this exit again, if it fails
 CSFRSWS                      *** No Exit Name was specified ***
 CSFRTC                       *** No Exit Name was specified ***
 CSFSBC                       *** No Exit Name was specified ***
 CSFSBD                       *** No Exit Name was specified ***
```

Figure 123. Third Installation Exits Display Panel

The fourth panel of exits is shown in Figure 124.

```
CSFSOP30 ----- ICSF - Installation Exits Display ---- ROW 55 TO 70 OF 70
COMMAND ===>

ICSF NAME      LOAD MODULE   OPTIONS
-------------  -----------   -------
 CSFSKI                       *** No Exit Name was specified ***
 CSFSKM                       *** No Exit Name was specified ***
 CSFSKY                       *** No Exit Name was specified ***
 CSFSPN                       *** No Exit Name was specified ***
 CSFSRRW                      *** No Exit Name was specified ***
 CSFSSWS                      *** No Exit Name was specified ***
 CSFSYG                       *** No Exit Name was specified ***
 CSFSYI                       *** No Exit Name was specified ***
 CSFSYX                       *** No Exit Name was specified ***
 CSFTCK                       *** No Exit Name was specified ***
 CSFUDK                       *** No Exit Name was specified ***
****************************BOTTOM OF DATA***************************
```

Figure 124. Fourth Installation Exits Display Panel

The system programmer specified the exit identifier, the load-module-name, and the failure option for each exit your installation uses with the EXIT keyword in the installation options data set. On this panel, you can view information about any exit that is specified in the installation options data set. The exit identifier is the ICSF name for the exit.

Table 9 shows the names for some general ICSF exits. Table 10 and Table 11 on page 183 show the ICSF name for each callable service exit.

Table 9. General ICSF Exits and Exit Identifiers

General ICSF Exit	Exit Identifier
Conversion Exit	**CSFCONVX**
Cryptographic Key Data Set Retrieval Exit	**CSFCKDS**
Key Generator Utility Program Exit	**CSFKGUP**
Mainline Exits	**CSFEXIT2, CSFEXIT3, CSFEXIT4, CSFEXIT5**
Security Initialization Exit Point	**CSFESECI**
Security Key Exit Point	**CSFESECK**
Security Service Exit Point	**CSFESECS**
Security Termination Exit Point	**CSFESECT**
Single-record, Read-write Exit Point	**CSFSRRW**

Table 10. Callable Service and its Exit Identifier

Service	Exit Identifier
ANSI X9.17 EDC generate	**CSFAEGN**
ANSI X9.17 Key Export	**CSFAKEX**
ANSI X9.17 Key Import	**CSFAKIM**
ANSI X9.17 Key Translate	**CSFAKTR**
ANSI X9.17 Transport Key Partial Notarize	**CSFATKN**
Clear PIN Encrypt	**CSFCPE**
Clear PIN Generate Alternate	**CSFCPA**
Clear Key Import	**CSFCKI**
Cipher/Decipher	**CSFEDC**
Cipher Text Translate	**CSFCTT**
Cipher Text Translate (with ALET)	**CSFCTT1**
Control Vector Translate	**CSFCVT**
Cryptographic Variable Encipher	**CSFCVE**
Data Key Import	**CSFDKM**
Decode	**CSFDCO**
Decipher	**CSFDEC**
Decipher (with ALET)	**CSFDEC1**
Data Key Export	**CSFDKX**
Digital Signature Generate	**CSFDSG**
Digital Signature Verify	**CSFDSV**
Diversified Key Generate	**CSFDKG**

Table 10. Callable Service and its Exit Identifier (continued)

Service	Exit Identifier
Encode	**CSFECO**
Encipher under Master Key	**CSFEMK**
Encipher	**CSFENC**
Encipher (with ALET)	**CSFENC1**
Encrypted PIN Generate	**CSFEPG**
Key Export	**CSFKEX**
Key Generate	**CSFKGN**
Key Import	**CSFKIM**
Key Part Import	**CSFKPI**
Key Record Create	**CSFKRC**
Key Record Delete	**CSFKRD**
Key Record Read	**CSFKRR**
Key Record Write	**CSFKRW**
Key Test	**CSFKYT**
Key Test Extended	**CSFKYTX**
Key Translate	**CSFKTR**
MAC Generate	**CSFMGN**
MAC Generate (with ALET)	**CSFMGN1**
MAC Verify	**CSFMVR**
MAC Verify (with ALET)	**CSFMVR1**
MDC Generate	**CSFMDG**
MDC Generate (with ALET)	**CSFMDG1**
Multiple Clear Key Import	**CSFCKM**
Multiple Secure Key Import	**CSFSCKM**
One-Way Hash Generate	**CSFOWH**
One-Way Hash Generate (with ALET)	**CSFOWH1**
PCI Interface	**CSFPCI**
PIN Generate	**CSFPGN**
PIN Translate	**CSFPTR**
PIN Verify	**CSFPVR**
PKA Decrypt	**CSFPKD**
PKA Encrypt	**CSFPKE**
PKA Key Generate	**CSFPKG**
PKA Key Import	**CSFPKI**
PKA Key Token Change	**CSFPKTC**
PKDS Record Create	**CSFPKRC**
PKDS Record Delete	**CSFPKRD**
PKDS Record Read	**CSFPKRR**
PKDS Record Write	**CSFPKRW**
Prohibit Export	**CSFPEX**

Table 10. Callable Service and its Exit Identifier (continued)

Service	Exit Identifier
Prohibit Export Extended	**CSFPEXX**
Random Number Generate	**CSFRNG**
Retained Key Delete	**CSFRKD**
Retained Key List	**CSFRKL**
Secure Key Import	**CSFSKI**
Secure Messaging for Keys	**CSFSKY**
Secure Messaging for PINs	**CSFSPN**
SET Block Compose	**CSFSBC**
SET Block Decompose	**CSFSBD**
Symmetric Key Generate	**CSFSYG**
Symmetric Key Import	**CSFSYI**
Symmetric Key Export	**CSFSYX**
Transform CDMF Key	**CSFTCK**
User Derived Key	**CSFUDK**
VISA CVV Service Generate	**CSFCSG**
VISA VISA CVV Service Verify	**CSFCSV**

Table 11. Compatibility and its Exit Identifier

Service	Exit Identifier
Encipher under Master Key	**CSFEMK**
CUSP/PCF GENKEY Service	**CSFGKC**
CUSP/PCF RETKEY Service	**CSFRTC**
Cipher/Decipher	**CSFEDC**

The load module name is the name of the module that contains the exit. The LOAD MODULE column on the panel lists the load module name for each exit. The OPTIONS column on this panel lists the action to occur if the exit fails.

4. To change the module name or failure option of an exit or add a new exit after viewing this panel, access the installation options data set. In the data set, change how you specified an exit or specify a new exit and restart ICSF.

Displaying Installation-Defined Callable Services

ICSF provides callable services to perform cryptographic functions. You can write a callable service to perform a function unique to your installation. In the installation options data set, you must define each installation-defined callable service. You specify a number to identify the service to ICSF, and you specify the load module that contains the service. You can use the panels to view the number and module name for each installation-defined callable service.

Before you can run an installation-defined service, you must do the following:

- Write the service.
- Define the service.
- Write a service stub and link it with your application program.

For more information about writing, defining, and running an installation-defined service, see *z/OS ICSF System Programmer's Guide*.

To display information about installation-defined callable services:

1. Select option 3, OPSTAT, on the Primary Option panel, as shown in Figure 125.

```
CSF@PRIM ----- Integrated Cryptographic Service Facility ---------
OPTION ===> 3

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

       Licensed Materials - Property of IBM

     5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 125. Selecting the Installation Options and Hardware Status Option on the Primary Menu Panel

The Installation Options panel appears. Refer to Figure 126.

```
CSFSOP00 ----------- ICSF - Installation Options --------
OPTION ===> 3

Enter the number of the desired option above.

  1  OPTIONS  -  Display Installation Options
  2  EXITS    -  Display Installation exits and exit options
  3  SERVICES -  Display Installation Defined Services
```

Figure 126. Installation Options Panel

2. Select option 3, Services, on the Installation Options Status panel.

   The Installation Defined Services panel appears. Refer to Figure 127 on page 185.

```
CSFSOP40 -------- ICSF - Installation Defined Services --- ROW 1 TO 8 OF 8
COMMAND ===>

  SERVICE NUMBER        INSTALLATION NAME
  --------------        -----------------

         1               SERVICE1
         3               SERVICE3
         5               SERVICE5
         6               SERVICE6
         8               SERVICE8
        11               SERVICEB
        13               SERVICED
****************************BOTTOM OF DATA************************************
```

Figure 127. Installation-Defined Services Display Panel

The system programmer used the SERVICE keyword in the installation options data set to specify the service-number, the load-module-name, and fail-option for each service. The service number identies the service to ICSF. The load-module-name identifies the module that contains the installation-defined service. The Installation Name column on the panel lists the load-module-name for each installation service.

The panel displays the service number and the corresponding installation name for each installation-defined service that is specified in the installation options data set.

Note: If your installation does not have any installation-defined callable services and you select option 3, the message NO GENERIC SERVICES displays and you remain on the Installation Options panel.

At ICSF start up, you define an installation options data set that contains the options your installation wants to use. The options specify certain modes and conditions on your ICSF system. You specify the keyword and value for each option in the installation options data set. You specify the data set name in the startup procedure. When you start ICSF, the options become active.

Chapter 9. Managing User Defined Extensions on PCI Cryptographic Coprocessors

User Defined Extensions (UDX) support allows you to request implementation of a customized cryptographic callable service. User Defined Extensions are ICSF functions developed for your installation with the help of IBM Global Services. With z/OS Version 1 Release 2 and the @server zSeries Model 900, and with a special contract with IBM, you can develop and load your own UDXs.

Note: A TKE Workstation is required to enable the access control points for UDXs.

You must define your routine to ICSF in the Installation Options Data Set. For more detailed information on the Installation Options Data Set and the new UDX keyword, see *z/OS ICSF System Programmer's Guide*.

The generic service load module is loaded during ICSF startup. Use the ICSF panels to perform UDX authorization processing.

You can perform the following tasks:

- Display a list of UDX ids of all authorized UDXs on a specific PCI cryptographic coprocessor
- Display a list of all PCI cryptographic coprocessors on which a specific UDX is authorized
- Authorize a UDX on any PCI cryptographic coprocessor in the system

```
CSF@PRIM --------- Integrated Cryptographic Service Facility -------
OPTION ===> 9

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

        Licensed Materials - Property of IBM

     5685-051 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.


Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 128. Selecting the UDX MGMT Option on the ICSF Primary Menu Panel

Once you have selected option 9, the following panel is displayed:

```
CSFUDX00 --------- OS/390 ICSF - User Defined Extensions Management -------
OPTION ===>

Enter the number of the desired option.

  1  Display the authorized UDXs for a coprocessor

  2  Display the coprocessors where a UDX is authorized

  3  Authorize a UDX
```

Figure 129. User Defined Extensions Management Panel

Display UDX IDs

A panel similar to Figure 130 is displayed when option 1 is selected.

```
CSFUDX10 ----- ICSF - Authorized UDX Coprocessor Selection   Row 1 to 1 of 6
COMMAND ===>                                                SCROLL===> PAGE

Select the coprocessor to be queried and press ENTER.

  COPROCESSOR        SERIAL NUMBER        STATUS
  -----------        -------------        ------

  P00                41-00YE1             ACTIVE
  P01                41-00K11             ACTIVE
  P02                41-0A355             ACTIVE
  P03                41-0BA3F             ACTIVE
  P04                41-0RT2T             ACTIVE
  P07                41-00B4M             ACTIVE
```

Figure 130. Authorized UDX Coprocessor Selection Panel

Select the PCI Cryptographic Coprocessor you wish to query. Use a / to select the coprocessor. Only one coprocessor can be selected. A panel similar to Figure 131 is displayed.

```
CSFUDX20 --------- ICSF - Authorized UDXs               Row 1 to 1 of 3
COMMAND ===>                                           SCROLL===> PAGE

For PCI Cryptographic Coprocessor P00, the following UDXs are authorized:

  UDX id        Service Module       Comment
  ------        --------------       -------

  XD            UDXSABCD             PIN processing extensions
  XE            UDXSEFGH             Multiple hash generate service
  YH            UDXSIJKL             Secure messaging key generate
 ******************************Bottom of data******************************
```

Figure 131. Authorized UDXs Panel

This panel shows the authorized User Defined Extensions for the coprocessor selected. The UDX id is the two character code. The service module is the z/OS load module specified in the UDX keyword in the ICSF Installation Options Data Set. The comment is also specified in the UDX keyword.

Display Coprocessors for a UDX

This panel is displayed when option 2 is selected from the User Defined Extensions Management Panel.

```
CSFUDX30  --------- ICSF - Coprocessors for Authorized UDXs ----------------
COMMAND ===>

Enter the two character id of the User Defined Extension to be queried.

  UDX id ===>
```

Figure 132. Coprocessors for Authorized UDXs Panel

Use this panel to specify the User Defined Extension id to be queried. A panel similar to Figure 133 appears:

```
CSFUDX40 --------- ICSF - Coprocessors for Authorized UDX      Row 1 to 1 of 3
COMMAND ===>                                                  SCROLL===> PAGE

User Defined Extension XX is authorized on the following coprocessors:

 COPROCESSOR        SERIAL NUMBER        STATUS
  -----------        -------------        ------

  P00                41-00YE1             ACTIVE
  P01                41-00K11             ACTIVE
  P04                41-0RT2T             ACTIVE
 **********************************Bottom of data***************************
```

Figure 133. Coprocessors for Authorized UDXs Panel

Authorize a UDX

A panel similar to Figure 134 on page 190 is displayed when option 3 is selected from the User Defined Extensions Management Panel.

```
CSFUDX50 --------- ICSF - Authorize User Defined Extension --  Row 1 to 1 of 6
COMMAND ===>                                                  SCROLL===> PAGE

UDX id ==>
Password==>

Select the coprocessors to be processed and press ENTER.

  COPROCESSOR        SERIAL NUMBER        STATUS
   -----------        -------------        ------

   P00                41-00YE1             ACTIVE
   P01                41-00K11             ACTIVE
   P02                41-0A355             ACTIVE
   P03                41-0BA3F             ONLINE
   P04                41-0RT2T             ACTIVE
   P07                41-00B4M             ACTIVE
    ******************************Bottom of data***************************
```

Figure 134. Authorize UDXs Panel

If your UDX was developed for your installation by IBM Global Services, you may have been provided a password associated with the UDX.

Use this panel to authorize a specific User Defined Extension on one or more PCI Cryptographic Coprocessors.

Enter the the two character id in the UDX id field. Enter the sixteen hexadecimal characters of the password in the Password field. Use a / to select the coprocessors where the UDX will be authorized.

Chapter 10. Using the Utility Panels to Encode and Decode Data

Encoding data is enciphering data by using a clear key. Decoding data is deciphering data by using the same clear key that enciphered the data. You can use the utility panels to encode and decode data.

Note: ICSF must be active with a valid master key before the encode and decode options may be used. Encode and decode are available only on a DES-capable server or processor. CDMF-only systems cannot use encode and decode.

Encoding Data

To encode data:

1. Select option 5, UTILITY, on the Primary Option panel, and press ENTER. Refer to Figure 135.

```
CSF@PRIM ----- Integrated Cryptographic Service Facility ---------
OPTION ===> 5

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

        Licensed Materials - Property of IBM

     5685-051 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.


Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 135. Selecting the Utilities Option on the Primary Menu Panel

The Utilities panel appears. See Figure 136 on page 192.

```
CSFUTL00 -------------- ICSF - Utilities --------------------------
OPTION ===> 1


Enter the number of the desired option.

  1  ENCODE        -  Encode data
  2  DECODE        -  Decode data
  3  RANDOM        -  Generate a random number
  4  CHECKSUM      -  Generate a checksum and verification and
                      hash pattern
  5  PPKEYS        -  Generate master key values from a pass phrase
```

Figure 136. Selecting the Encode Option on the Utilities Panel

2. Select option 1, Encode, on this panel.
   The Encode panel appears. See Figure 137.

```
CSFECO00 -------------- ICSF - Encode -----------------------------
COMMAND ===>


Enter data below:

  Clear Key       ===>  0000000000000000   Clear Key Value
  Plaintext       ===>  0000000000000000   Data to be encoded
  Ciphertext         :  0000000000000000   Output from the encode
```

Figure 137. Encode Panel

3. In the Clear Key field, enter the clear value of the key you want ICSF to use to encode the data.
4. In the Plaintext field, enter the data in hexadecimal form that you want ICSF to encode.
5. Press ENTER.
   ICSF uses the clear key and the DES algorithm to encode the data. The encoded data is displayed in the Ciphertext field.
6. Press END to return to the Utilities panel.
7. Press END to return to the Primary Option panel.

Decoding Data

To decode data:

1. Select option 5, UTILITY, on the Primary Option panel and press ENTER.
   The Utilities panel appears. See Figure 138 on page 193.

```
CSFUTL00 -------------- ICSF - Utilities ---------------------------
OPTION ===> 2


Enter the number of the desired option.

  1  ENCODE        -  Encode data
  2  DECODE        -  Decode data
  3  RANDOM        -  Generate a random number
  4  CHECKSUM      -  Generate a checksum and verification and
                      hash pattern
  5  PPKEYS        -  Generate master key values from a pass phrase
```

Figure 138. Selecting the Encode Option on the Utilities Panel

2. Select option 2, Decode, on this panel.

   The Decode panel appears. See Figure 139.

```
CSFECO00 -------------- ICSF - Decode ------------------------------
COMMAND ===>


Enter data below:

  Clear Key       ===>  0000000000000000   Clear Key Value
  Ciphertext      ===>  0000000000000000   Data to be decoded
  Plaintext          :  0000000000000000   Output from the decode
```

Figure 139. Decode Panel

3. In the Clear Key field, enter the clear value of the key you want ICSF to use to decode the data. This needs to be the same key value that was used to encode the data.
4. In the Ciphertext field, enter the data in hexadecimal form that you want ICSF to decode.
5. Press ENTER.

   ICSF uses the clear key and the DES algorithm to decode the data. The decoded data is displayed in the Plaintext field.
6. Press END to return to the Utilities panel.
7. Press END to return to the Primary Option panel.

Chapter 11. Using the ICSF Utility Program CSFEUTIL

This chapter contains Programming Interface Information.

ICSF provides a utility program, CSFEUTIL, that performs certain functions that can also be performed using the administrator's panels.

The program that executes CSFEUTIL must be APF-authorized.

The utility can be used for installations with the cryptographic coprocessor feature and the PCI cryptographic coprocessor feature. You can run the utility program to perform the following tasks:

- Reencipher a disk copy of a CKDS
- Change the master key
- Refresh the in-storage CKDS
- Initialize a CKDS and load DES and PKA master keys using a pass phrase

  Restriction: Only loads DES and PKA master keys on the CCF. If you have a PCICC as part of the configuration, the DES and SYM-MK are not loaded.

You invoke the program as a batch job or from another program. To invoke the program as a batch job, use JCL. You specify different parameters on the EXEC statement depending on the task you want the utility program to perform. To invoke the program from another program, use standard MVS linkages like LINK, ATTACH, LOAD, and CALL.

For information about using the utility program to reencipher a disk copy of a CKDS and change the master key, see "Reenciphering a Disk Copy of a CKDS and Changing the Master Key". For information about using the program to refresh the in-storage CKDS, see "Refreshing the In-Storage CKDS Using a Utility Program" on page 196.

Reenciphering a Disk Copy of a CKDS and Changing the Master Key

This section describes how to use the utility program to reencipher a disk copy of a CKDS and to change a master key.

Note: Before performing any function that affects the current CKDS, such as reenciphering, refreshing, or changing the master key, consider temporarily disallowing dynamic CKDS update services. For more information, refer to "Disallowing Dynamic CKDS Updates During KGUP Updates" on page 96.

1. Before you change a master key, you must first reencipher any disk copies of the CKDSs under the new master key in the new master key register.

   You can reencipher a CKDS either using the panels or the utility program.

   Note: In compatibility or co-existence mode, you can use the utility program to reencipher a CKDS but not to change the master key. To change the master key using the utility program, you must be in noncompatibility mode.
2. Invoke the program as a batch job or from another program.

   You pass the same parameters whether you call the program as a batch job or from another program.

3. Pass the names of the CKDSs upon which to perform the task and the name of the task to perform.

   When you invoke the utility program from another program, General Register 1 must contain the address of a data area whose structure is as follows:

   ```
   Bytes 0-1: Length of the parameter string in binary
   Bytes 2-n: The parameter string
   ```

   The parameter string is the same as that which you would specify using the PARM keyword on the EXEC JCL statement if you invoked the program as a batch job.
4. To reencipher a disk copy of a CKDS, pass the following parameters in the following order:
   a. The name of the disk copy of the CKDS to reencipher.
   b. The name of an empty disk copy of the CKDS to contain the reenciphered keys.
   c. The name for the task: REENC.
5. To reencipher the CKDS using JCL, use JCL like the following example:

   ```
   //STEP EXEC PGM=CSFEUTIL,PARM='OLD.CKDS,NEW.CKDS,REENC'
   ```

   The first parameter passed, OLD.CKDS, is the name of the disk copy to reencipher. The second parameter, NEW.CKDS, is the name of an empty disk copy of the CKDS where you want ICSF to place the reenciphered keys.
6. After you reencipher all the disk copies of the CKDSs under the new master key, make the new master key active by changing the master key.

   The utility program activates the new master key and reads a disk copy of a CKDS reenciphered under the new master key into storage.
7. To change a master key, pass the following parameters in the following order:
   a. The name of the disk copy of the CKDS to read into storage.
   b. The name for the task: CHANGE.
8. To change the master key using JCL, use JCL like the following example:

   ```
   //STEP EXEC PGM=CSFEUTIL,PARM='NEW.CKDS,CHANGE'
   ```

   The utility program reads the new master key into the master key register to make that master key active. The program also reads into storage a disk copy of the CKDS that you specify. This CKDS should be reenciphered under the new master key that you are making the current master key. The first parameter passed, NEW.CKDS, is the name of the disk copy of the CKDS that you want ICSF to read into storage.

When you invoke the program as a batch job, you receive the return code in a message when the job completes. You do not receive a reason code with the return code. When the program is invoked from another program, the invoking program receives the reason code in General Register 0 along with the return code in General Register 15. The return codes and reason codes are explained in “Return and Reason Codes for the CSFEUTIL Program” on page 198.

Refreshing the In-Storage CKDS Using a Utility Program

This section describes how to use the CSFEUTIL program to refresh an in-storage CKDS.

1. Invoke the program from a batch job or from another program.
2. You pass the same parameters whether you call the program as a batch job or from another program.

3. Pass the names of the CKDSs to perform the task and the name for the task. When you invoke the utility program from another program, General Register 1 must contain the address of a data area whose structure is as follows:

   ```
   Bytes 0-1: Length of the parameter string in binary
   Bytes 2-n: The parameter string
   ```

   The parameter string is the same as that which you would specify using the PARM keyword on the EXEC JCL statement if you invoked the program as a batch job.
4. To refresh an in-storage CKDS, pass the following parameters in the following order:
   - The name of the disk copy of the CKDS that you want read into storage
   - The name for the task: REFRESH
5. To refresh the CKDS using JCL, use JCL like the following example:

   ```
   //STEP EXEC PGM=CSFEUTIL,PARM='NEW.CKDS,REFRESH'
   ```

   The first parameter passed, NEW.CKDS, is the name of the disk copy of the CKDS that you want read into storage.

When you invoke the program as a batch job, you receive the return code in a message when the job completes. You do not receive a reason code with the return code. When the program is invoked from another program, the invoking program receives the reason code in General Register 0 along with the return code in General Register 15. The return codes and reason codes are explained in “Return and Reason Codes for the CSFEUTIL Program” on page 198.

Loading DES and PKA Master Keys Using a Pass Phrase

This section describes how to use the CSFEUTIL program to load DES and PKA master keys using a pass phrase. This will allow an automated setup of ICSF for an automated electronic delivery process.

The CKDS must be created and empty. See *z/OS ICSF System Programmer's Guide* for this information.

Note: This only initializes the CCF. It will not initialize the PCICC.

The default pass phrase supplied is `Change this Pass Phrase`.

1. Invoke the program from a batch job or from another program.
2. You pass the same parameters whether you call the program as a batch job or from another program.
3. Pass the name of the CKDS to perform the task and the name for the task. When you invoke the utility program from another program, General Register 1 must contain the address of a data area whose structure is as follows:

   ```
   Bytes 0-1: Length of the parameter string in binary
   Bytes 2-n: The parameter string
   ```

   The parameter string is the same as that which you would specify using the PARM keyword on the EXEC JCL statement if you invoked the program as a batch job.
4. To load a pass phrase, pass the following parameters in the following order:
   - The name of the CKDS
   - An optional 16–64 character pass phrase
   - The name for the task: PPINIT

5. To load the pass phrase using JCL (with the default pass phrase), use JCL like the following example:

   ```
   //STEP EXEC PGM=CSFEUTIL,PARM='CSF.CSFCKDS,PPINIT'
   ```

6. To load the pass phrase using JCL (and using your own pass phrase), use JCL like the following example:

   ```
   //STEP EXEC PGM=CSFEUTIL,PARM='CSF.CSFCKDS,different pass phrase,PPINIT'
   ```

When you invoke the program as a batch job, you receive the return code in a message when the job completes. You do not receive a reason code with the return code. When the program is invoked from another program, the invoking program receives the reason code in General Register 0 along with the return code in General Register 15. The return codes and reason codes are explained in “Return and Reason Codes for the CSFEUTIL Program”.

Return and Reason Codes for the CSFEUTIL Program

When you invoke the CSFEUTIL program as a batch job, you receive the return code in a message when the job completes. The meanings of the return codes are as following:

Return Code	Meaning
0	Process successful.
4	Parameters are incorrect.
8	RACF authorization check failed.
12	Process unsuccessful.
72 or 104	CKDS processing has failed.

When the program is invoked from another program, the invoking program receives the reason code in General Register 0 along with the return code in General Register 15. The meaning of the reason codes are as follows:

Return code 8 has the following reason codes:

Reason Code	Meaning
16000	Invoker has insufficient RACF access authority to perform function.

Return code 12 has the following reason codes:

Reason Code	Meaning
16000	Invoker has insufficient access authority to perform function (also applicable to refresh and reencipher).
36000	Unable to change master key. Check hardware status.
36020	Input CKDS is empty or not initialized (authentication pattern in the control record is invalid).
36036	The new master key register for Coprocessor 1 (C1) is not full, but C0 is ready and the current master key is valid.
36040	The new master key register for C0 is not full, but C1 is ready and the current master key is valid.
36044	The master key authentication pattern for the CKDS does not match the authentication pattern of the coprocessors, which are not equal.

36048 The master key authentication pattern for the CKDS does not match the authentication pattern of either of the coprocessors, which are not equal.

36052 A valid new master key is present in C0, but its authentication pattern does not match that of C1 or the CKDS, which are equal.

36056 A valid new master key is present in C1, but its authentication pattern does not match that of C0 or the CKDS, which are equal.

36060 The new master key register(s) is/are not full.

36064 Both new master key registers are full but not equal.

36068 The input CKDS is not enciphered under the current master key.

36076 The new master key register for C0 is not full, but the CPUs are online.

36080 The new master key register for C1 is not full, but the CPUs are online.

36084 The master key register cannot be changed since ICSF is running in compatibility mode.

Return code 72 or 104 has the following reason codes:

Reason Code	Meaning
6008	A service routine has failed. The service routines that may be called are: **CSFMGN** MAC generation **CSFMVR** MAC verification **CSFMKVR** Master key verification
6012	The single-record, read-write installation exit (CSFSRRW) returned a return code greater than 4.
6016	An I/O error occurred reading or writing the CKDS.
6020	The CSFSRRW installation exit abended and the installation options EXIT keyword specifies that the invoking service should end.
6024	The CSFSRRW installation exit abended and the installation options EXIT keyword specifies that ICSF should end.
6028	The CKDS access routine could not establish the ESTAE environment.
6040	The CSFSRRW installation exit could not be loaded and is required.
6044	Information necessary to set up CSFSRRW installation exit processing could not be obtained.
6048	The system keys cannot be found while attempting to write a complete CKDS data set.
6052	For a write CKDS record request, the current master key verification pattern (MKVP) does not match the CKDS header record MKVP.

Note: It is possible that you will receive MVS reason codes rather than ICSF reason codes, for example, if the reason code indicates a dynamic allocation failure. For an explanation of Dynamic Allocation reason codes, see *z/OS MVS Programming: Authorized Assembler Services Guide*.

Chapter 12. Using the ICSF Utility Program CSFPUTIL

This chapter contains Programming Interface Information.

ICSF provides a utility program, CSFPUTIL, that performs certain functions that can also be performed using the administrator's panels.

The utility can be used for installations with the cryptographic coprocessor feature and the PCI cryptographic coprocessor feature. You can run the utility program to perform the following tasks:

- Reencipher a PKDS
- Activate the reenciphered PKDS
- Refresh the PKDS cache

You invoke the program as a batch job or from another program. To invoke the program as a batch job, use JCL. You specify different parameters on the EXEC statement depending on the task you want the utility program to perform. To invoke the program from another program, use standard MVS linkages like LINK, ATTACH, LOAD, and CALL.

For information about using the utility program to reencipher a disk copy of a PKDS, see "Reenciphering a PKDS". For information about using the program to activate the reeenciphered PKDS, see "Activating a Reenciphered PKDS" on page 202. For information about using the program to refresh the PKDS cache, see "Refreshing the PKDS Cache" on page 202.

Reenciphering a PKDS

You can reencipher a PKDS either using the panels or the utility program.

1. Invoke the program as a batch job or from another program.

   You pass the same parameters whether you call the program as a batch job or from another program.

2. Pass the names of the PKDSs upon which to perform the task and the name of the task to perform.

   When you invoke the utility program from another program, General Register 1 must contain the address of a data area whose structure is as follows:

   ```
   Bytes 0-1: Length of the parameter string in binary
   Bytes 2-n: The parameter string
   ```

   The parameter string is the same as that which you would specify using the PARM keyword on the EXEC JCL statement if you invoked the program as a batch job.

3. To reencipher a PKDS, pass the following parameters in the following order:
   a. The name of the PKDS to reencipher.
   b. The name of an empty PKDS to contain the reenciphered keys.
   c. The name for the task: RECIPHER.

4. To reencipher the PKDS using JCL, use JCL like the following example:

   ```
   //STEP EXEC PGM=CSFPUTIL,PARM='OLD.PKDS,NEW.PKDS,RECIPHER'
   ```

   The first parameter passed, OLD.PKDS, is the name of the PKDS to reencipher. The second parameter, NEW.PKDS, is the name of an empty PKDS where you want ICSF to place the reenciphered keys.

5. After you reencipher all the PKDSs under the new master key, activate the PKDS.

When you invoke the program as a batch job, you receive the return code in a message when the job completes. You do not receive a reason code with the return code. The return codes are explained in “Return Codes for the CSFPUTIL Program” on page 203.

Activating a Reenciphered PKDS

You can activate a reenciphered PKDS either using the panels or the utility program.

1. Invoke the program as a batch job or from another program.

   You pass the same parameters whether you call the program as a batch job or from another program.
2. Pass the name of the PKDS upon which to perform the task and the name of the task to perform.

   When you invoke the utility program from another program, General Register 1 must contain the address of a data area whose structure is as follows:

   ```
   Bytes 0-1: Length of the parameter string in binary
   Bytes 2-n: The parameter string
   ```

   The parameter string is the same as that which you would specify using the PARM keyword on the EXEC JCL statement if you invoked the program as a batch job.
3. To activate a reenciphered PKDS, pass the following parameters in the following order:
   a. The name of the PKDS to activate.
   b. The name for the task: ACTIVATE.
4. To activate the PKDS using JCL, use JCL like the following example:

   ```
   //STEP EXEC PGM=CSFPUTIL,PARM='NEW.PKDS,ACTIVATE'
   ```

   The first parameter passed, NEW.PKDS, is the name of the PKDS to activate.

When you invoke the program as a batch job, you receive the return code in a message when the job completes. You do not receive a reason code with the return code. The return codes are explained in “Return Codes for the CSFPUTIL Program” on page 203.

Refreshing the PKDS Cache

This section describes how to use the CSFPUTIL program to refresh a PKDS cache.

1. Invoke the program from a batch job or from another program.
2. You pass the same parameters whether you call the program as a batch job or from another program.
3. When you invoke the utility program from another program, General Register 1 must contain the address of a data area whose structure is as follows:

   ```
   Bytes 0-1: Length of the parameter string in binary
   Bytes 2-n: The parameter string
   ```

The parameter string is the same as that which you would specify using the PARM keyword on the EXEC JCL statement if you invoked the program as a batch job.

4. To refresh a PKDS cache, pass the following parameter:
   - The name for the task: REFRESH
5. To refresh the PKDS cache using JCL, use JCL like the following example:

```
//STEP EXEC PGM=CSFPUTIL,PARM='REFRESH'
```

When you invoke the program as a batch job, you receive the return code in a message when the job completes. You do not receive a reason code with the return code. The return codes are explained in “Return Codes for the CSFPUTIL Program”.

Return Codes for the CSFPUTIL Program

When you invoke the CSFPUTIL program as a batch job, you receive the return code in a message when the job completes. The meanings of the return codes are as following:

Return Code	Meaning
0	Process successful.
4	Partially successful. Job completed but some tokens have not been reenciphered.
8	RACF authorization check failed.
12 or 72	Reencipher, activate or refresh process unsuccessful.

An abend 18F Reason code x’300’ occurs with a JCL error.

Appendix A. CCC Bit Assignments

Following are some of the hardware CCC (crypto configuration control) definitions. You can view these values from the coprocessor hardware status panel (see Figure 112 on page 164). You are not able to change these values.

Note: The CCC applies only to the Cryptographic Coprocessor Feature.

BIT	***Meaning***
6	indicates TKE can be supported.
37 - 38	indicates triple DES and AES are supported.

Bits 80 through 127 (the right-most bits on the hardware status panel) form a pattern indicating the key length that is allowed.

When these bits are `07F7F 0F7F7`, the maximum RSA key management key length is 512 bits.

When these bits are `0FFFF 0FFFF`, the maximum RSA key management key length is 1024 bits.

Appendix B. Control Vector Table

Note: The Control Vectors used in ICSF are exactly the same as documented in CCA and the TSS manuals.

The master key enciphers all keys operational on your system. A transport key enciphers keys that are distributed off your system. Before a master key or transport key enciphers a key, ICSF exclusive ORs both halves of the master key or transport key with a control vector. The same control vector is exclusive ORed to the left and right half of a master key or transport key.

Also, if you are entering a key part, ICSF exclusive ORs each half of the key part with a control vector before placing the key part into the CKDS.

Each type of key on ICSF (except the master key) has either one or two unique control vectors associated with it. The control vector that ICSF exclusive ORs the master key or transport key with depends on the type of key the master key or transport key is enciphering. For double-length keys, a unique control vector exists for each half of a specific key type. For example, there is a control vector for the left half of an input PIN-encrypting key, and a control vector for the right half of an input PIN-encrypting key.

If you are entering a key part into the CKDS, ICSF exclusive ORs the key part with the unique control vector(s) associated with the key type. ICSF also enciphers the key part with two master key variants for a key part. One master key variant enciphers the left half of the key part, and another master key variant enciphers the right half of the key part. ICSF creates the master key variants for a key part by exclusive ORing the master key with the control vectors for key parts. These procedures protect key separation.

Table 12 displays the default value of the control vector that is associated with each type of key. For keys that are double-length, ICSF enciphers a unique control vector on each half. Control vectors indicated with an ″*″ are supported by the CCF.

Table 12. Default Control Vector Values

Key Type	Control Vector Value (Hex) Value for Single-length Key or Left Half of Double-length Key	Control Vector Value (Hex) Value for Right Half of Double-length Key
*AKEK	00 00 00 00 00 00 00 00	
CIPHER	00 03 71 00 03 00 00 00	
CVARDEC	00 3F 42 00 03 00 00 00	
CVARENC	00 3F 48 00 03 00 00 00	
CVARPINE	00 3F 41 00 03 00 00 00	
CVARXCVL	00 3F 44 00 03 00 00 00	
CVARXCVR	00 3F 47 00 03 00 00 00	
*DATA	00 00 00 00 00 00 00 00	
DATAC	00 00 71 00 03 41 00 00	00 00 71 00 03 21 00 00
*DATAM generation key (external)	00 00 4D 00 03 41 00 00	00 00 4D 00 03 21 00 00
*DATAM key (internal)	00 05 4D 00 03 00 00 00	00 05 4D 00 03 00 00 00

Table 12. Default Control Vector Values (continued)

Key Type	Control Vector Value (Hex) Value for Single-length Key or Left Half of Double-length Key	Control Vector Value (Hex) Value for Right Half of Double-length Key
*DATAMV MAC verification key (external)	00 00 44 00 03 41 00 00	00 00 44 00 03 21 00 00
*DATAMV MAC verification key (internal)	00 05 44 00 03 00 00 00	00 05 44 00 03 00 00 00
*DATAXLAT	00 06 71 00 03 00 00 00	
DECIPHER	00 03 50 00 03 00 00 00	
DKYGENKY	00 71 44 00 03 41 00 00	00 71 44 00 03 21 00 00
ENCIPHER	00 03 60 00 03 00 00 00	
*EXPORTER	00 41 7D 00 03 41 00 00	00 41 7D 00 03 21 00 00
IKEYXLAT	00 42 42 00 03 41 00 00	00 42 42 00 03 21 00 00
*IMP-PKA	00 42 05 00 03 41 00 00	00 42 05 00 03 21 00 00
*IMPORTER	00 42 7D 00 03 41 00 00	00 42 7D 00 03 21 00 00
*IPINENC	00 21 5F 00 03 41 00 00	00 21 5F 00 03 21 00 00
*MAC	00 05 4D 00 03 00 00 00	
*MACVER	00 05 44 00 03 00 00 00	
OKEYXLAT	00 41 42 00 03 41 00 00	00 41 42 00 03 21 00 00
*OPINENC	00 24 77 00 03 41 00 00	00 24 77 00 03 21 00 00
*PINGEN	00 22 7E 00 03 41 00 00	00 22 7E 00 03 21 00 00
*PINVER	00 22 42 00 03 41 00 00	00 22 42 00 03 21 00 00

Note: The external control vectors for DATAC, double-length MAC generation and MAC verification keys are also referred to as data compatibility control vectors.

Appendix C. Supporting Algorithms and Calculations

This appendix shows various algorithms and calculations that are used in cryptographic systems.

Checksum Algorithm

To enter a key or a master key manually, you enter key parts. When you enter a key part, you enter two key part halves and a checksum for the key part. The checksum is a two-digit number you calculate using the key part and the checksum algorithm.

After you enter the key part and the checksum, ICSF calculates the checksum for the key part you entered. If the checm you enter and the checm ICSF calculates do not match, you did not enter the key part correctly and should reenter it. Before you enter a key part, you need to calculate the checm. You can use the ICSF utility panels that are described in Chapter 5, “Managing Master Keys”, on page 51 or the checm algorithm that is described in this appendix.

In the checm algorithm, you use the following operations:

- Sum Operation

  The addition table in Figure 140 defines the sum operation. The sum of two hexadecimal digits i and j is the entry at the intersection of the column i and the row j. For example, the sum of A and 6 is C.

Sum	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	0	3	2	5	4	7	6	9	8	B	A	D	C	F	E
2	2	3	0	1	6	7	4	5	A	B	8	9	E	F	C	D
3	3	2	1	0	7	6	5	4	B	A	9	8	F	E	D	C
4	4	5	6	7	0	1	2	3	C	D	E	F	8	9	A	B
5	5	4	7	6	1	0	3	2	D	C	F	E	9	8	B	A
6	6	7	4	5	2	3	0	1	E	F	C	D	A	B	8	9
7	7	6	5	4	3	2	1	0	F	E	D	C	B	A	9	8
8	8	9	A	B	C	D	E	F	0	1	2	3	4	5	6	7
9	9	8	B	A	D	C	F	E	1	0	3	2	5	4	7	6
A	A	B	8	9	E	F	C	D	2	3	0	1	6	7	4	5
B	B	A	9	8	F	E	D	C	3	2	1	0	7	6	5	4
C	C	D	E	F	8	9	A	B	4	5	6	7	0	1	2	3
D	D	C	F	E	9	8	B	A	5	4	7	6	1	0	3	2
E	E	F	C	D	A	B	8	9	6	7	4	5	2	3	0	1
F	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0

Figure 140. Addition Table

- Shift Operation

  The shift table in Figure 141 defines the shift operation. The shift of digit i is denoted by H(i). For example, the shift of 5 is H(5) = E.

i	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
H(i)	0	C	1	D	2	E	3	F	4	8	5	9	6	A	7	B

Figure 141. Shift Table

In the following description of the algorithm, the two hexadecimal digits of the checm are represented by P1 and P2 for the set of 32 hexadecimal digits D(1,2,.....,32). The letter i represents the increment.

To calculate the checm, use the following algorithm:

1. Set i = 0, and set P1 and P2 = 0 (hexadecimal).
2. Let P1 = Sum of P1 and D(i + 1). Let P2 = Sum of P2 and D(i + 2).
3. Let P1 = H(P1). Let P2 = H(P2).
4. Let i = i + 2. If i < 32, go to step 2; otherwise, go to step 5.
5. P1 equals the first checm digit. P2 equals the second checm digit.

Algorithm for Calculating a Verification Pattern

To enter a master key or operational key manually, you enter key parts. After you enter a key part, ICSF displays a verification pattern for that key part on a panel. To verify that you entered the key part correctly, you can use the value of the key part you enter to calculate the verification pattern. Check that the verification pattern you calculate matches the verification ICSF calculates.

To calculate this verification pattern, use the following algorithm:

1. If the key part is an operational key part, exclusive OR the key part with the control vector for the key part's key type. See Appendix B, "Control Vector Table", for a listing of control vectors by key type. If the key part is a master key part, do not exclusive OR it with a control vector.
2. Use the DES algorithm to encrypt the left half of the key part (either master key part or modified operational key part) under the key `4545 4545 4545 4545`.
3. Exclusive OR the result of step 2 with the left half of the key part.
4. Use the result of step 3 as the DES key in the DES algorithm to encrypt the right half of the key part.
5. Exclusive OR the result of step 4 with the right half of the key part.

The resulting 64-bit value is the verification pattern.

The verification pattern for the master key appears on the Coprocessor Selection and Hardware Status panels. If a master key register is full, the panels display the master key verification pattern. The verification patterns for two identical master keys are the same. You can use the verification patterns to verify that master keys in two different key storage units are the same.

ICSF records a master key verification pattern in the SMF record when you enter a master key part or activate a master key. The ICSF SMF record also records a verification pattern when you enter an operational key part.

Algorithm for Calculating an Authentication Pattern

When you initialize a CKDS, ICSF uses the current master key and the authentication pattern algorithm to calculate an authentication pattern for the CKDS. ICSF places the value of the authentication pattern in the header record of the CKDS.

At ICSF startup, ICSF uses the authentication pattern to verify that the master key enciphers the current CKDS specified at ICSF startup. It compares the authentication pattern that is stored in the CKDS with the authentication pattern it calculates for the master key. If the authentication patterns do not match, ICSF startup fails, and ICSF gives you a message that states that the master key is not valid.

To calculate the authentication pattern, ICSF uses the following algorithm:

1. Encrypt the left half of the master key under the key `6767 6767 6767 6767`, using the DES algorithm.
2. Exclusive OR the result of step 1 with the original left half of the key.
3. Use the result of step 2 as the DES key in the DES algorithm to encrypt the right half of the master key.
4. Exclusive OR the result of step 3 with the original right half of the master key.

The resulting 64-bit value is the authentication pattern.

Pass Phrase Initialization Master Key Calculations

The values for the DES and PKA master keys are calculated in the following manner:

1. ICSF appends a two-byte constant, X'AB45', to the pass phrase, and generates the MD5 hash for the string by using an initial hash value of X'23A0BE487D9BD32003424FAAA34BCE00'. The first eight bytes of the result of this calculation become the last eight bytes of the PKA signature master key and the last eight bytes of the calculation become the last eight bytes of the PKA key management master key.
2. ICSF generates the DES master key value by appending a four-byte constant, X'551B1B1B', to the pass phrase, and generating the MD5 hash for the string using the hash that results from Step 1 as the initial hash value.
3. ICSF appends a three-byte constant, X'2A2A88', to the pass phrase and generates the MD5 hash for the string using the output hash of Step 2 as the initial hash value. The result of this calculation becomes the first 16 bytes of PKA signature master key.
4. ICSF appends a one-byte constant, X'94' to the pass phrase, and generates the MD5 hash for the string using the output hash of Step 3 as the initial hash value. The result of this calculation becomes the first 16 bytes of the PKA key management master key.

Note: If the SMK=KMMK option is selected or defaulted, the KMMK is not used.

The MDC–4 Algorithm for Generating Hash Patterns

The MDC-4 algorithm calculation is a one-way cryptographic function that is used to compute the hash pattern of a key part. MDC uses encryption only, and the default key is 5252 5252 5252 5252 2525 2525 2525 2525.

Notations Used in Calculations

The MDC calculations use the following notation:

eK(X) Denotes DES encryption of plaintext X using key K

‖ Denotes the concatenation operation

XOR Denotes the exclusive-OR operation

:= Denotes the assignment operation

T8<1> Denotes the first 8–byte block of text

T8<2> Denotes the second 8–byte block of text, and so on

KD1, KD2, IN1, IN2, OUT1, OUT2
Denote 64–bit quantities

MDC-1 Calculation

The MDC-1 calculation, which is used in the MDC-4 calculation, consists of the following procedure:

```
MDC-1 (KD1, KD2, IN1, IN2, OUT1, OUT2);
   Set KD1mod := set bit 1 and bit 2 of KD1 to "1" and "0", respectively.
   Set KD2mod := set bit 1 and bit 2 of KD2 to "0" and "1", respectively.
   Set F1 := IN1 XOR eKD1mod(IN1)
   Set F2 := IN2 XOR eKD2mod(IN2)
   Set OUT1 := (bits 0..31 of F1) ‖ (bits 32..63 of F2)
   Set OUT2 := (bits 0..31 of F2) ‖ (bits 32..63 of F1)
End procedure
```

MDC-4 Calculation

The MDC-4 calculation consists of the following procedure:

```
MDC-4 (n, text, KEY1, KEY2, MDC);
   For i := 1, 2,...n do
      Call MDC-1(KEY1,KEY2,T8<i>,T8<i>,OUT1,OUT2)
      Set KEY1int := OUT1
      Set KEY2int := OUT2
      Call MDC-1(KEY1int,KEY2int,KEY2,KEY1,OUT1,OUT2)
      Set KEY1 := OUT1
      Set KEY2 := OUT2
   End do
   Set output MDC := (KEY1 ‖ KEY2)
End procedure
```

Appendix D. PR/SM Considerations during Key Entry

If you use logical partition (LPAR) mode provided by the Processor Resource/System Manager (PR/SM), you may have additional considerations when performing the following tasks:

- Entering keys
- Displaying hardware status
- Using the public key algorithm
- Using a TKE Workstation

These additional considerations depend on your processor hardware. For example, LPAR mode permits you to have multiple logical partitions and each logical partition (LP) can have access to the crypto CP for key entry. Therefore, at any given time, multiple LPs can perform key entry procedures.

This appendix gives some basic information on using ICSF in LPAR mode. For more detailed information on configuring and running in LPAR mode, refer to the *PR/SM Planning Guide* and the *S/390 Hardware Management Console Guide*.

Allocating Cryptographic Resources to a Logical Partition

LPs operate independently but can share access to the same cryptographic coprocessor, just as they can share access to I/O devices and any other central processor resources. When you activate the LP, you can specify which cryptographic functions are enabled for that LP. The cryptographic resources available to the LP and the way you allocate them to the LP depends on the server or processor your are using.

Allocating Resources on S/390 Enterprise Servers and S/390 Multiprise

You use the Hardware Master Console tasks to enable various cryptographic functions for an LP. To assign a control domain index and usage domain index and initially enable cryptographic functions for an LP, use the Crypto page of the Customize Activation Profiles task. On the Crypto page you can enable the following functions to the LP:

- Public key algorithm (PKA) function
- Cryptographic functions
  - Special secure mode
  - Public key secure cable (PKSC) and Integrated Cryptographic Service Facility (ICSF)
    - Modify authority (only enabled in one LPAR partition at a time)
    - Query signature controls
    - Query transport controls

These functions are hierarchically applied. For instance, if you do not enable cryptographic functions for the LP, you cannot enable any of the functions below it on the list. To enable basic ICSF functions, you must select the following parameters on the crypto page:

- Usage domain index

  The number you select for usage domain index must match the domain number that is entered in the installation options data set for this LP.
- Enable cryptographic functions

- Enable public key secure cable (PKSC) and Integrated Cryptographic Service Facility (ICSF)

Once an LP is activated, you can then use the Change LPAR Crypto task to change the cryptographic functions that are enabled for that LP. This task has a page for each LP.

Entering the Master Key or Other Keys in LPAR Mode

To perform key entry from the TKE workstation, you must use a logical partition that already has key entry enabled.

In certain situations, ICSF clears the master key registers so the master key value is not disclosed. ICSF clears the master keys in all the logical partitions. The CKDSs and PKDSs are still enciphered under the master keys. To recover the keys in the CKDSs and PKDSs, you must reenter and activate the DES and PKA master keys.

To restore the master keys, first ensure that key entry is enabled for all usage domain indexes for which you need to reenter the master keys. Since multiple domains can have key entry enabled, the domains may already be enabled. Reenter and activate the master key for all usage domain indexes. You can do this either through the Clear Master Key Part Entry panels or the TKE workstation.

Reusing or Reassigning a Domain

In the course of business, you may find it necessary to reuse or reassign a domain that is currently active. If this is the case, there are several steps to perform. It is a good security practice to zeroize the domain secrets, which includes retained keys and master keys.

Run the retained key delete service in the domain to remove them.

You can zeroize the master key with the TKE workstation or with TSO panels. For information on the TKE process, see *z/OS ICSF TKE Workstation User's Guide 2000*.

If you are using the TSO panels, follow the procedure in "Changing Master Keys" on page 74 for your DES and PKA master keys. Your key type should equal DES and the key value should be all zeros.

```
CSFDKE10 -------------- ICSF - Clear Master Key Entry ---------
COMMAND ===>

             CCF DES/PCICC SYM-MK new master key register     : FULL
             CCF Signature/PCICC ASYM-MK master key register  : FULL
             CCF Key management master key register           : FULL


 Specify information below
   Key Type  ===> DES      (DES, SMK, KMMK, ALL-PKA)

   Part      ===> FIRST        (RESET, FIRST, MIDDLE, FINAL)


   Checksum  ===> 00

   Key Value ===> 0000000000000000
             ===> 0000000000000000
             ===> 0000000000000000   (SMK, KMMK and ALL-PKA only)
```

Figure 142. The Clear Master Key Entry Panel

Appendix E. Running HCR7708 on an IBM @server zSeries 990

Support for the IBM @server zSeries 990 has been added. Crypto assist instructions and the optional PCI Cryptographic Accelerator are available on this server. This server does not support the Cryptographic Coprocessor Feature or the PCI Cryptographic Coprocessor.

During the initialization of HCR7708, the startup task determines the type of hardware environment. If the Cryptographic Coprocessor Feature is available, processing continues the same as in the previous release of ICSF and no change is needed for existing Cryptographic Coprocessor Feature configurations. If crypto assist instructions or the optional PCI Cryptographic Accelerator is available, without the Cryptographic Coprocessor Feature for secure cryptography, the machine type will be recognized as the IBM @server zSeries 990.

Running HCR7708 on the IBM @server zSeries 990 differs from running it on the S/390 G5 Enterprise Server, S/390 G6 Enterprise Server, IBM @server zSeries 800 and IBM @server zSeries 900. This section describes the processing differences in the IBM @server zSeries 990 environment. This section does not apply if you are running HCR7708 on a S/390 G5 Enterprise Server, S/390 G6 Enterprise Server IBM @server zSeries 800 or IBM @server zSeries 900.

Operating System Requirements

HCR7708 can be installed on z/OS V1 R3 or z/OS V1 R4 if you are installing on the IBM @server zSeries 990. System SSL and Communication Server applications are the only applications that you will also be able to exploit. System SSL is a web deliverable available with the ICSF web deliverable, HCR7708.

Applications and programs

Applications requiring secure cryptography using encrypted keys will not be able to execute on the IBM @server zSeries 990. All cryptographic keys will appear in the clear.

The following application and programs are not supported on the IBM @server zSeries 990:

- The CICS attachment facility
- The Key Generation Utility Program (KGUP)
- The CKDS Conversion program
- The CSFEUTIL program for CKDS reencipher, refresh, change master key, and passphrase initialization functions
- The PKDS reencipher program (CSFPUTIL)
- The PKDS cache refresh program (CSFPUTIL)
- The PKDS activate program (CSFPUTIL)
- PCF applications
- 4753-HSP applications
- Distributed Key Management System (DKMS)
- Access Method Services Cryptographic option
- UDX (User Defined Extension) support

Callable services

The following services are available when running HCR7708 on a IBM @server zSeries 990:

- Character/Nibble Conversion (CSNBXBC and CSNBXCB)
- Code Conversion (CSNBXEA and CSNBXAE)
- Control Vector Generate (CSNBCVG)
- Decode (CSNBDCO) - This service requires enablement of CP Assist for Cryptographic Functions.
- Encode (CSNBECO) - This service requires enablement of CP Assist for Cryptographic Functions.
- MDC Generate (CSNBMDG and CSNBMDG1) - This service requires enablement of CP Assist for Cryptographic Functions.
- One–Way Hash Generate (CSNBOWH and CSNBOWH1)
- PKA Decrypt (CSNDPKD) - This service requires a PCI Cryptographic Accelerator.
- PKA Encrypt (CSNDPKE) ZERO-PAD formatting only - This service requires a PCI Cryptographic Accelerator.
- PKA Key Token Build (CSNDPKB)
- PKA Public Key Extract (CSNDPKX)
- Symmetric Key Decipher (CSNBSYD and CSNBSYD1) - This service requires enablement of CP Assist for Cryptographic Functions.
- Symmetric Key Encipher (CSNBSYE and CSNBSYE1) - This service requires enablement of CP Assist for Cryptographic Functions.
- X9.9 Data Editing (CSNB9ED)

Installation defined callable services are supported only if you're using clear keys and using one of the above supported callable services.

Callable services that require secure cryptography (the Cryptographic Coprocessor Feature and PCI Cryptographic Coprocessor) to execute, will fail with return code 12 and reason code 8 (Service or algorithm is not available on current hardware).

CKDS and PKDS

The CKDS (Cryptographic Key Data Set) and PKDS (Public Key Data Set) do not have to be allocated on a IBM @server zSeries 990.

CKDS and PKDS labelnames are not supported.

Exits

The following exit types are supported on a IBM @server zSeries 990:

- Mainline exits
- Exits for callable services

  Exits are only supported on available callable services. See "Callable services" for a list of available callable services.
- Security exits

ICSF Setup and Initialization

The following steps do not have to be performed when installing and initializing HCR7708 on a IBM @server zSeries 990:

- ICSF Setup
  - Create the Cryptographic Key Data Set (CKDS)
  - Create the Public Key Data Set (PKDS)
  - MK initialization for SMP/E
- TKE Setup
- Loading Master Keys and Initializing the CKDS through ICSF panels
- Customizing TKE and loading master keys
- CICS-ICSF attachment facility setup

It is normal to see the following messages during the startup of ICSF on a IBM @server zSeries 990:

- Starting ICSF on a IBM @server zSeries 990 without a PCI Cryptographic Accelerator.

  ```
  CSFM001I ICSF INITIALIZATION COMPLETE
  ```

- Starting ICSF on a IBM @server zSeries 990 with a PCI Cryptographic Accelerator. You 'll receive message CSFM411I for each PCI Cryptographic Accelerator you have online.

  ```
  CSFM411I PCI CRYPTOGRAPHIC ACCELERATOR A34 IS ACTIVE
  CSFM001I ICSF INITIALIZATION COMPLETE
  ```

Installation options data set

The following installation options data set keywords are supported on a IBM @server zSeries 990. All other keywords will be processed for syntax only and will otherwise be ignored.

- DOMAIN - On a IBM @server zSeries 990, if a PCI Cryptographic Accelerator is not available, the DOMAIN parameter is not required. If a PCI Cryptographic Accelerator is available, the DOMAIN parameter is required if more than one domain is specified as the usage domain on the PR/SM panels. If only one usage domain is assigned to the LPAR, the DOMAIN parameter is optional.
- CHECKAUTH
- TRACEENTRY
- USERPARM
- EXIT

  Exits are only supported on available callable services. See "Callable services" on page 218 for a list of available callable services.

Note: Optional keyword fields that are not specified will display as blanks on the ISPF panels.

Exploitation

If you are migrating to HCR7708 on a IBM @server zSeries 990 server, System SSL and Communication Server applications are the only applications that you'll be able to migrate. Various security products that support clear keys can continue to execute on an IBM @server zSeries 990, but must be recoded to exploit crypto assist instructions.

Note: There are no migration issues for HCR7708 for existing installations with the following servers: S/390 G5 Enterprise Server, S/390 G6 Enterprise Server, IBM @server zSeries 800 and IBM @server zSeries 900.

Secure Sockets Layer (SSL)

System SSL applications are supported on the IBM @server zSeries 990. SSL defines methods for data encryption, server authentication, message integrity, and client authentication for a TCP/IP connection. Security is provided on the link and callable services that have been enhanced for DES, TDES and SHA-1 services.

TKE workstation

The Trusted Key Entry (TKE) workstation is not needed on the IBM @server zSeries 990 since there are no hardware protected master keys.

TSO panels

Not all utilities are available on an IBM @server zSeries 990. Options 1, 3 and 5 are available. Error messages are issued for the unavailable utilites.

```
CSF@PRIM ------------- Integrated Cryptographic Service Facility ---------
OPTION ===>

Enter the number of the desired option.

  1  COPROCESSOR MGMT    -  Management of Cryptographic Coprocessors
  2  MASTER KEY          -  Master key set or change, CKDS/PKDS processing
  3  OPSTAT              -  Installation options
  4  ADMINCNTL           -  Administrative Control Functions
  5  UTILITY             -  ICSF Utilities
  6  PPINIT              -  Pass Phrase Master Key/CKDS Initialization
  7  TKE                 -  TKE Master and Operational key processing
  8  KGUP                -  Key Generator Utility processes
  9  UDX MGMT            -  Management of User Defined Extensions

      Licensed Materials - Property of IBM

     5694-A01 (C) Copyright IBM Corp. 1990, 2003. All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 143. ICSF Primary Menu Panel

There are new panels in several cases.

If you wish to display coprocessor status on an IBM @server zSeries 990, the following panel will appear:

```
CSFGCMP0 ---------------- ICSF Coprocessor Management -------------
OPTION ===> 1

Select the coprocessors to be processed and press ENTER.
Action characters are: A and D. See the help panel for details.

COPROCESSOR  MODULE ID/SERIAL NUMBER                 STATUS
-----------  -------------------------------         -------

_ A06                                                 ACTIVE
_ A07                                                 ACTIVE
```

Figure 144. Coprocessor Management Panel

If you choose the UTILITY option from the Primary Option panel, the following panel now appears:

```
CSFGUTL0 ---------------- ICSF - Utilities -------------------------
OPTION ===>


Enter the number of the desired option.

  1  ENCODE        -  Encode data
  2  DECODE        -  Decode data
```

Figure 145. ICSF Utilities Panel

Appendix F. Accessibility

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 it 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 Volume 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.

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 verifiy the operations 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
USA

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

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, 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:

IBM Corporation
Mail Station P300
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.

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

Programming Interface Information

This ICSF Administrator's Guide is intended to help the ICSF administrator manage the cryptographic keys.

This book primarily documents information that is NOT intended to be used as a Programming Interface of OS/390 ICSF.

This book also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of OS/390 ICSF. This information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

Programming Interface information

End of Programming Interface information

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

AIX	Personal System/2
ES/3090	Processor Resource/Systems Manager
IBM	PR/SM
IBMLink	RACF
MVS/DFP	S/390
MVS/ESA	S/390 Parallel Enterprise Server
Multiprise	SecureWay
OS/390	Resource Link
Personal Security	3090
zSeries	z/OS

The e-business logo is a trademark of IBM.

The following term is a trademark of another company:

VISA VISA International Service Association

MasterCard
MasterCard International Incorporated

Netscape
Netscape Communications Corporation

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

Index

D

E

F

G

M

N

O

P

R

S

T

U

V

W

Readers' Comments — We'd Like to Hear from You

z/OS
Integrated Cryptographic Service Facility
Administrator's Guide

Publication No. SA22-7521-04

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

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.

Name _____ Address _____

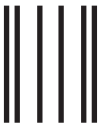
Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
SA22-7521-04

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY 12601-5400

Fold and Tape | **Please do not staple** | Fold and Tape

Cut or Fold
Along Line

Program Number: 5647-A01, 5655–G52

Printed in U.S.A.

SA22-7521-04