ESA/390 Integrated Cryptographic Facility: An overview

by P. C. Yeh
R. M. Smith, Sr.

This paper reviews the objectives of the Enterprise Systems Architecture/390™ (ESA/390™) Integrated Cryptographic Facility. It presents the cryptographic key-management scheme, summarizes key elements and unique characteristics of the facility, and describes the physical security provided by the first ESA/390 implementation.

Cryptography is an effective method of protecting information while it is being transmitted through a communication link or while it is stored in a medium vulnerable to unauthorized access. Cryptographic operations can also be used for processing message authentication codes (MACs) and personal identification numbers (PINs) in a financial-transaction environment.

As the connectivity of computer networks and the quantity and value of information processed by computers increases, concerns have grown about the threat of disclosure or modification, done accidentally or intentionally, of sensitive data. Computer users have demanded high-speed cryptographic functions for bulk encryption to provide network and database security.

Also, because of the pervasive use of PINs at automated teller machines and point-of-sale terminals, and the increasing use of electronic funds transfer among banks and wholesale institutions, the financial industry has become more security conscious and has started to demand high-performance and high-security computer systems to support many types of financial transactions.

This paper describes the Enterprise Systems Architecture/390™ (ESA/390™) Integrated Cryptographic Facility (ICRF), which is a CPU-integrated implementation of DEA-based cryptographic operations. The Data Encryption Algorithm (DEA) is a Federal Information Processing Standard[1] and an American National Standard.[2]

Objectives

The overall objective is to provide a DEA-based cryptographic facility on System/390™ machines to support bulk encryption and financial-transaction environments with high performance and security and various levels of compatibility. Following is a description of specific requirements and objectives in more detail.

High performance is a key requirement for the ICRF. With more and more automated teller machines or point-of-sale terminals attached to a host, performance of cryptographic operations at the host has become a constraint that prevents the achievement of acceptable transaction rates at several major installations. To achieve the per-

formance requirement set for the mid-1990s, the objective for the first high-end implementation was 1000 transactions per second.[3] A CPU-integrated approach was needed to achieve this goal.

Compatibility with previous IBM cryptographic products was another important consideration. Applications written for the ICRF must be interoperable with existing IBM cryptographic products. That is, it must be possible to interchange data keys encrypted under a key-encrypting key as well as to interchange encrypted data. Application programs written for the Cryptographic Unit Support Program (CUSP)[4] and the Programmed Cryptographic Facility (PCF)[5] must be portable, at the source-code level, to the Integrated Cryptographic Service Facility (ICSF);[6] that is, an application program written for CUSP or PCF, after being recompiled or reassembled, must operate correctly under ICSF. It must also be possible to convert a cryptographic key data set (CKDS) from the form used by CUSP to that used by ICSF.

Overall security of a computer system depends largely on system integrity.

Interoperability and program portability with the IBM 4753 Network Security Processor was also required. In this case, interoperability made the interchange of keys necessary for many new key types. Program portability was extended to a larger set of functions, including operations for MAC processing, PIN processing, and key distribution. Accomplishing this extension required a joint effort to develop a Common Cryptographic Architecture (CCA).[7] CCA is intended to apply to all future IBM cryptographic products.

The facility must be capable of processing MACs and PINs for financial-transaction environments. Ciphertext translation and automated key-generation and key-distribution functions were required. A secure means of installing the master key and initial key-encrypting keys using dual control was also required. Additionally, dynamic, transparent master-key change must be provided to enhance usability and availability.

Enhanced overall security, particularly internal security, was also an important requirement. It is imperative that there be a way to minimize the amount of software, particularly application programs, that must be trustworthy.

Cryptography is an effective mechanism for protecting external data but is not a total solution to the general data security problem in a computer system. The overall security of a computer system depends largely on system integrity, which is normally achieved by means of authorization and access controls. For example, if there were no control of the use of the cryptographic facility and the use of keys, the overall system security could hardly be enhanced by cryptography.[8] System integrity is the only ultimate protection.

In some cases, cryptography can be used to enhance system integrity so that the amount of software that must be trustworthy may be reduced, though it cannot be totally eliminated. Key handling using control vectors[9,10] is an example of enhancing internal integrity. It is provided mainly to reduce the degree of trust required in some internal software that uses or controls the use of the cryptographic facility.

Security is measured relatively. Different installations have different security requirements, making it difficult to set up a security objective satisfying all users. However, a security objective is needed to achieve a balanced design and to provide a guideline for making detailed technical decisions. The following security objectives are based on targeted market environments and applications. These objectives are also based on the assumption that an operating system with reasonable authorization or access controls is to be used in conjunction with the cryptographic facility.

1. Provide a cryptographic facility on general-purpose computer systems for commercial applications (as contrasted with military applications).
2. Perform all cryptographic operations by the cryptographic facility within a physically secure boundary.
3. Eliminate any practical possibility of any machine component, such as hardware or microprogram, outside the physically secure bound-

ary subverting the security of the cryptographic facility.

4. Eliminate any practical possibility of any program (application or privileged) deriving the master key or encrypted cryptographic keys in the clear (not encrypted) outside the cryptographic facility. This objective applies to the manipulation and use of the cryptographic functions only in the normal mode and not the special-security mode. The special-security mode is used to enable several functions that require keys or PINs to be in the clear and that need tight control.
5. Eliminate any practical possibility of application programs (as contrasted to privileged programs) subverting the intended security.

Applications

The ICRF has a number of CPU synchronous functions to support the major applications now described.

Data secrecy. To protect the secrecy of data, high-speed encryption and decryption functions are included. These functions provide both fast response time for short messages and high throughput for bulk data. They use the cipher-block-chaining (CBC) mode of operation.[11]

Message authentication. A message authentication code (MAC)[12] is a cryptographic checksum that can be used to verify messages. A MAC can be applied to a message, which is either encrypted or in the clear, from originator to recipient.

A MAC is generated for a message using a secret cryptographic key and is sent with the message by the originator. The recipient performs MAC verification. A MAC is generated for the received message using the same cryptographic key. The generated MAC is compared with the received MAC. If they match, it is highly likely that the message is genuine and has been received without modifications. The probability of detecting a change in the received data is dependent on the number of bits in the MAC.

MACs can be used to ensure the integrity of network communications between systems and also to ensure the integrity of stored data. High-performance MAC-generation and MAC-verification functions are provided to encourage such applications. These functions support a 32-, 48-, or 64-bit MAC. The MAC-verification function performs the entire verification process within a physically secure boundary and only indicates the verification result as being either successful or unsuccessful.

PIN verification. A secret personal identification number (PIN)[13] is usually used to authenticate the holder of a debit card or credit card in an electronic-funds-transfer system. The PIN is basically the cardholder's electronic signature, and its secrecy is of the utmost importance.

Generally speaking, the PIN can be a random number assigned by the card issuer or can be cryptographically derived from some information about the cardholder, such as a primary account number. Random PINs are normally placed in a 64-bit formatted block, called the PIN block, and then the PIN blocks are encrypted and stored in a PIN database for PIN verification. Derived PINs need not be stored and can be regenerated at verification time, based on a PIN-generation key.

A PIN may travel through several cryptographic switching nodes before being verified.

After being entered at an automated teller machine or point-of-sale terminal for host PIN verification, the PIN is placed in a PIN block, and the block is encrypted and sent to the host.

If PIN verification is done using a PIN database, the received PIN block is compared with an appropriate entry from the database.

If PIN verification is done using a cryptographic algorithm, the received PIN block is deciphered and compared with a generated PIN. The generated PIN is derived using the appropriate information, algorithm, and cryptographic keys. If the generated PIN and received PIN match, verification is successful. The ICRF performs the entire algorithmic verification process within the phys-

ically secure boundary and only indicates whether the result is successful or unsuccessful.

The ICRF provides high-performance PIN-verification functions for several algorithms. A number of commonly used PIN-block formats are supported by the PIN-verification functions.

Although algorithmic PIN verification would normally dictate the PIN that a cardholder must use, a PIN offset may be included to permit cardholders to select their own PINs. In this case, a PIN offset, which may be recorded on the card and need not be kept secret, specifies the relationship between the derived PIN and the selected PIN.

PIN translation. In some situations, a PIN entered by a cardholder may travel through several cryptographic switching nodes before it reaches the system that performs PIN verification. Normally, the cryptographic key encrypting the PIN block is changed at each switching node, and, depending on the capability at the next node, the PIN-block format may be changed. High-performance PIN-translation functions are provided to change the PIN-block format and encrypting key without disclosing the PIN outside of the physically secure boundary.

The PIN-translation functions can also be used in some other scenarios. For example, if PIN verification is done using a PIN database, the function can be used to convert, if necessary, the format of the received PIN block and the cryptographic key encrypting the received block to match the ones used by the PIN database.

Message translation. If the message originator and recipient do not share a cryptographic key, but each of them shares a secret key with a third node, this third node may re-encipher passing messages to allow encrypted messages to be transmitted between the originator and recipient. To meet this requirement, a high-speed ciphertext-translation function is provided to decrypt and re-encrypt the message without disclosing the clear message outside of the physically secure boundary.

Key management. A secret cryptographic key, normally a key-encrypting key (KEK), must be installed in two DEA systems before each one can communicate with the other by using cryptography. This initial key installation is usually performed by a manual procedure.

To minimize the number of keys that must be manually installed, key-management functions were designed to support a key-distribution center (KDC). With a KDC, each system would only need to manually install one KEK, which is shared by the KDC. Two systems that do not share a key could obtain a shared KEK from the KDC and then start cryptographic communications. For that purpose, a number of high-security key-management functions are provided to perform on-line key generation, key import, and key export for all types of keys.

Basic cryptographic concepts

This section presents some basic concepts and describes the ICRF implementation of key handling, including protection of cryptographic keys, degree of key separation, enforcement of control vectors, and conversion of key states.

Since the DEA is a key-controlled algorithm, the security of protected data depends on the security of the cryptographic key. Cryptographic keys are usually protected by encipherment under other keys, called key-encrypting keys.

A 128-bit master key is used by the ICRF to protect other keys in the system. The security of the master key is achieved by storing it in nonvolatile storage inside a physically secure boundary. Other KEKs are also 128 bits in length and are mostly used to protect cryptographic keys being transmitted on external links or being stored on an external or internal medium.

Although legitimate users will not misuse keys, an inside adversary may attempt to do so to subvert system security. To eliminate undesirable exposures caused by misusing keys, the ESA/390 ICRF controls key usage by means of key types based on the control vector concept.[9,10]

Key types. The ICRF specifies the intended usage of a cryptographic key by assigning a key type to the key. Thus, key type is used to achieve key separation. When a key is generated or imported, the type of the key is declared and remains unchanged thereafter. For this discussion, ten key types are described. The prescribed uses and the key length for keys of each type are explained as follows:

1. Data-encrypting key: This key is 64 bits and is used to encrypt or decrypt data.
2. Data-translation key: This key is 64 bits and is used only for the ciphertext-translation function. A data-translation key at an intermediate system is also a data-translation key at another system if the other system is also an intermediate one, or it is a data-encrypting key at another system if the other system is the message originator or recipient.
3. MAC-generation key: This key is 64 bits and is used by the message originator to generate MACs.
4. MAC-verification key: This key is 64 bits and is used by the message receiver to verify MACs. The MAC-generation key at the message originator is the MAC-verification key at the message receiver.
5. Input PIN-encrypting key: This key is 128 bits and is used at a receiving system to protect PIN blocks sent to this system from another system.
6. Output PIN-encrypting key: This key is 128 bits and is normally used at a sending system to protect PIN blocks sent from this system to another system. Two systems must share a common key to securely transmit PIN blocks; the key is an input PIN-encrypting key at the receiving system and is an output PIN-encrypting key at the sending system.
7. PIN-generation key: This key is 128 bits and is used to algorithmically generate PINs or PIN offsets.
8. PIN-verification key: This key is 128 bits and is used to algorithmically verify PINs. For algorithmic PIN-generation and -verification processes, the PIN-generation key in the clear form is equal to the PIN-verification key in the clear form.
9. Exporter KEK: This key is 128 bits and is used at a sending system to protect keys of any type that are sent from this system to another system.
10. Importer KEK: This key is 128 bits and is used at a receiving system to protect keys of any type that are sent from another system to this system or to protect keys that are stored internally or externally and can be imported to this system later. Two systems must share a common KEK for exchanging keys of any type; the key is an importer KEK at the receiver and is an exporter KEK at the originator.

Control vectors. The ICRF uses control vectors to specify key types and to control the intended usage of cryptographic keys. To implement the ICRF in the space available, it was necessary to make design tradeoffs between the circuitry used for control vector granularity, for performance, and for other functions. The resulting design was to implement a subset of the allowable control vector combinations as described in Reference 10. This subset, which was chosen to reduce complexity and maximize performance, consists of a set of control vectors that are predetermined constants. A control vector is assigned to each key type. To provide compatibility with the IBM 3848 Cryptographic Unit Support Program, all zeros was chosen as the control vector for data-encrypting keys. Also, two variant constants were included to assist in converting a CUSP-3848-type CKDS that is encrypted using 3848-type variants to the control vectors used by the ICRF.

This control-vector subset and the associated key types are used by the IBM Common Cryptographic Architecture (CCA) to ensure system interoperability and program portability for current and future IBM products.

Key-encrypting-key derivatives. A KEK derivative is computed by exclusive-ORing of the 128-bit KEK with an appropriate control vector. Keys of different types are encrypted under different KEK derivatives. The ESA/390 implementation of the control vector enforcement uses an implicit control vector table. The table containing control vectors resides within the physically secure boundary.

The function code of each cryptographic function defines the control vectors that are to be used to generate the appropriate KEK derivatives. If an encrypted key is misused in an unintended cryptographic function, the derivative of the specified KEK assumed by the operation is different from the one actually used to protect the key, and the key cannot be correctly retrieved by the operation.

When a key is exported or imported by using the key-management functions, the same control vector is used to obtain the KEK derivative that protects the key externally as is used to obtain the master-key derivative that protects the key in the system. Thus, the type of the key is not changed.

The type of a key is determined by the control vector used to obtain the KEK derivative that protects the key.

Key states. *The state of a key is determined by the key type of the KEK whose derivative encrypts the key.* Three different states of a cryptographic key are defined: operational, exportable, and importable.

A key encrypted under a master-key derivative is said to be in the operational state. Normal cryptographic functions only accept keys in the operational state of the system. A key in the operational state can be converted into the exportable state.

A key encrypted under an exporter KEK derivative is said to be in the exportable state. A key in this state is ready to be sent to the system that shares the same exporter KEK. A key in the exportable state of a system cannot be converted into the importable or operational state of the system.

A key encrypted under an importer KEK derivative is said to be in the importable state. A key in the importable state can be converted into the operational state.

If a key is sent from one system to another, the key is sent in the exportable state with respect to the sending system and is received in the importable state with respect to the receiving system.

Complementary key types. Since the DEA is a symmetric algorithm, two systems using cryptography to communicate with one another must share the same key. This key is normally used at one system for a particular function and at the other system for a different function. In the ICRF these two functions performed at different systems using the same key are called complementary functions. The corresponding key types for complementary functions are called complementary key types. Thus, two copies of the same key used in complementary functions must have complementary types.

Table 1 summarizes the complementary key types. Note that a data-encrypting key and a data-translation key can be the complementary type of itself. Note also that two copies of a key encrypted as complementary types normally reside in different systems.

Table 1 Complementary key types

Data-encrypting key	Data-encrypting key
Data-encrypting key	Data-translation key
Data-translation key	Data-translation key
MAC-generation key	MAC-verification key
Input PIN-encrypting key	Output PIN-encrypting key
PIN-generation key	PIN-verification key
Exporter KEK	Importer KEK

When two copies of a key are generated as complementary types in a system, they normally are generated in appropriate states to ensure that they cannot both be converted into the operational state on that system; otherwise key-usage control is not achieved, and security exposures may exist.

Key management

A DEA-based cryptographic system requires an effective mechanism for the secure generation, distribution, and installation of cryptographic keys. The control-vector scheme is a high-security method for the key creator to control the usage of the keys by the key receiver; thus, it is a secure means for network key management. Once a key of a particular type is created, the state of the key may be changed subsequently, but the type of the key normally remains unchanged. This section discusses the ICRF key-management functions.

Key generation. Although cryptographic keys may be generated manually by tossing coins or throwing dice, this section only describes automated mechanisms for generating keys by using functions provided by the ICRF.

A DEA-based pseudo-random-number generator is provided within the physically secure boundary. The generated number may be used for many cryptographic purposes, for example, as a cryptographic key.

The generate-complementary-keys function is used to generate two copies of a key encrypted as complementary types. One copy is in the importable state, and the other is in the exportable state. Keys of all supported key types can be generated in either state.

The generated key in the exportable state can be sent to another system sharing the same exporter

KEK. The generated key in the importable state can be converted into the operational state so that the key can be used by this system. Once this key is in the operational state, it can be further converted into the exportable state by using an exporter KEK and then can be distributed to the system sharing this exporter KEK.

Cryptographic keys may be distributed manually or automatically.

Key distribution. Cryptographic keys may be distributed manually or automatically. Automated distribution of keys usually involves keys in the encrypted form. Manual distribution is mainly used for keys in the clear form.

When cryptographic keys in clear are distributed manually, the keys are normally split into two or more parts, called key parts; each part has the same length as the complete key. The complete key is obtained by exclusive-ORing all key parts.

Encrypted keys in the exportable state are ready for automated distribution. The generate-complementary-keys function can be used to generate cryptographic keys of any type in the exportable state. The function can be used for both key generation and key distribution.

Cryptographic keys of any type in the operational or importable state can be distributed by automated means to any other system if an exporter KEK is shared between this system and the other system. Importable keys can first be converted into the operational state by means of the re-encipher-to-master-key function. The re-encipher-from-master-key function can be used to convert an operational key of any type into the exportable state. These functions do not change the type of the key being exported or imported.

The re-encipher-from-master-key function is useful to distribute one key to multiple systems. For example, it may be desirable to send the same key to several systems within an enterprise so that all of those systems can back up the originating one. As another example, a card issuer must distribute the same PIN-verification key to other financial institutions so that they can perform an algorithmic PIN-verification process.

Key installation. The master key can only be manually installed; all types of other keys may be installed manually or automatically.

A dual-key-entry procedure is used to manually enter cryptographic keys in the clear form. The procedure requires that the clear key be split into two or more parts. Each key part is separately entered, and the entered key parts are subsequently combined to obtain the complete key. The procedure is described in more detail later in the paper.

For automatic key import, the re-encipher-to-master-key function can be used to convert keys of any type in the importable state into the operational state. The encipher-under-master-key function is provided to convert a clear key into a data-encrypting key in the operational state.

When a large number of clear keys are to be imported for key types other than the data-encrypting-key type, an automatic mechanism is needed. The encipher-under-importer-key function is provided to convert a clear key into a key of any key type in the importable state. The function encrypts the clear-key value using an appropriate derivative of an importer KEK. The key produced by the function can be converted into the operational state by means of the re-encipher-to-master-key function. The encipher-under-importer-key function needs to be tightly controlled because a malicious program could use it to produce known keys of any type. Repeated execution of this function could produce keys of different types but with the same clear value. A special-security mode is defined to enable this function and others having similar characteristics. A physical key is required to activate the special-security mode.

Dynamic master key change

One requirement is to provide dynamic master key change in a way that is transparent to applications. This section briefly describes functions for supporting the master-key-change process

and mechanisms for detecting, reporting, and verifying master key change. There is only one level of master key conversion; that is, keys encrypted under derivatives of a master key cannot be recovered if the master key is changed twice before conversion occurs.

To facilitate master key change, two registers, in addition to the master key register, are available in the ICRF to hold the contents of the new and old master keys.

The re-encipher-to-new-master-key function reenciphers keys of any type from a derivative of the current master key to the same derivative of the new master key; the function provides a smooth transition of the master key change and is used by the cryptographic support program to convert the cryptographic key data set (CKDS) before the master key in hardware is changed.

The ICRF consists of a number of cryptographic functions.

The re-encipher-from-old-master-key function reenciphers keys of any type from a derivative of the old master key to the same derivative of the current master key. This function allows keys to be converted after the master key in hardware has been changed. The function is used for keys that are not in the CKDS—some cryptographic keys may be kept by applications. Also a master key change may occur between the suspension and resumption of a long cryptographic operation, such as encryption of bulk data.

A master-key-version-number (MKVN) register in the ICRF keeps track of the master key currently being used. For functions using the master key, a referenced MKVN is specified by the program. This reference is compared with the contents of the MKVN register. If they do not match, the operation is rejected. After detection, the cryptographic support program converts the key using the re-encipher-from-old-master-key function and then resumes the operation. This action is performed transparently to application programs.

The set-master-key function causes the current master key to become the old master key and the new master key to become the current master key. Functions are provided for generating a verification pattern for the new master key, the current master key, and the old master key. These patterns are made by using a cryptographic one-way function.[14] They can be used by the cryptographic support program to determine hardware status after the initial program load.

Structure of the ICRF

This section summarizes major hardware components of the ICRF. Also included are descriptions of some unique characteristics of the facility.

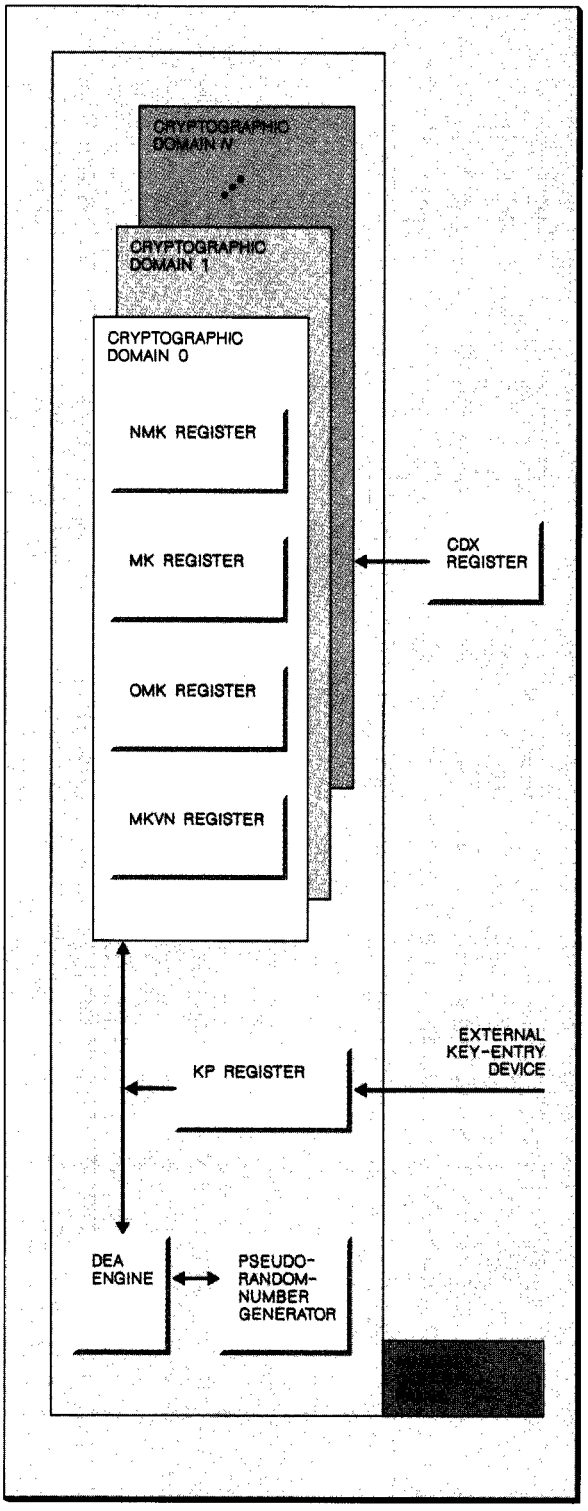
The ICRF consists of a number of cryptographic functions. To achieve high-performance and high-security objectives, all cryptographic functions were defined to be privileged, CPU-synchronous functions. The facility includes a DEA engine, a pseudo-random-number generator, a manual-control panel, and a number of registers. Information about master keys and some internal secret data, such as cryptographic keys and the pseudo-random-number generator seed, is preserved in nonvolatile storage while the main power of the machine is off.

The facility is physically protected by a tamper-resistant secure enclosure. Secret information, such as clear keys or clear PINs, or secret intermediate results are always kept inside the physically secure boundary of the ICRF.

ICRF registers. The ICRF includes the following registers:

- The master key (MK) register contains the current master key used by normal cryptographic functions.
- The new master key (NMK) register contains the new master key or new master key parts for dynamic master key change.
- The old master key (OMK) register contains the old master key for dynamic master key change.
- The key part (KP) register is used as a buffer for manually importing cryptographic keys from an external key-entry device, such as a key pad.
- The master key version number (MKVN) regis-

Figure 1 Integrated Cryptographic Facility



ter is provided to ensure the integrity and to control the use of the master key.

Cryptographic domains. The ICRF supports multiple, independent cryptographic domains to achieve high protection and isolation among systems running on the same machine. The MK, NMK, OMK, and MKVN registers are replicated for each cryptographic domain.

Systems using different cryptographic domains are cryptographically isolated from each other because they use different master keys. A mechanism is provided to prevent unauthorized systems from accessing a cryptographic domain.

When the machine is operating in the Processor Resource/Systems Manager™ (PR/SM™) mode,[15] a cryptographic domain can be designated for each PR/SM partition so that systems running in different partitions can achieve the same degree of isolation and protection as that of physically separate machines, each with a different master key. When the machine is operating in the native mode, different systems running on the machine at different times by means of initial program loading (IPL) can also use different cryptographic domains to achieve a high degree of isolation.

A cryptographic-domain-index (CDX) register is provided in each CPU containing the ICRF to designate a cryptographic domain used by cryptographic functions. The register contents are changeable only by privileged programs.

Figure 1 shows the structure of the ICRF.

Protection of cryptographic domain. The contents of the MKVN register in a cryptographic domain and an authorization pattern are jointly used to control the use of the cryptographic domain.

The MKVN register contents are reset to zeros by IPL. Afterward, the register must be set to a nonzero value before normal cryptographic functions can be performed. The register contents can be set to a nonzero value only if the program can supply the correct authorization pattern. The authorization pattern is the result of a cryptographic one-way function using the master key. The program can obtain the authorization pattern of a master key only when the master key is in the NMK register. Thus, after the contents of the NMK

register have been moved into the MK register, other programs cannot obtain the authorization pattern. The authorization pattern is then protected by the owner and is unknown to other programs so that no other program can use the associated master key.

Manual-control panel. A manual-control panel is provided for clearing secret quantities, disabling the ICRF, and controlling the use of certain special cryptographic functions. The panel includes the following manual controls:

- Reset: While this control is on, a second reset control is enabled, causing all secret quantities in the ICRF to be set to zeros. This control consists of two physical switches to reduce the chance of accidental operation of the reset function.
- Special-security mode: If this control is on, the encipher-under-importer-key function and PIN-generation functions are enabled. Additional controls are provided under PR/SM to enable the special-security mode only for selective PR/SM partitions.
- Disable: With this control on, all cryptographic functions are disabled for all cryptographic domains.
- Operational key part 1 (OKP1): This control allows a function to be enabled for importing the first key part of an operational key.[16]
- Operational key part 2 (OKP2): While this control is on, a function is enabled for importing the second or subsequent key part of an operational key.
- New master key part 1 (NMKP1): Operation of this control permits a function to be enabled for importing the first key part of a new master key.
- New master key part 2 (NMKP2): With use of this control, a function is enabled for importing the second or subsequent key part of a new master key.

Only one control of the OKP1, OKP2, NMKP1, and NMKP2 can be turned on at any time. Additional controls under PR/SM allow only one PR/SM partition at a time to perform manual key entry.

Setting the above controls requires physical keys. Entering the first key part requires different physical keys than it does to enter the remaining key parts of an operational key or a master key.

Dual-key entry

The ICRF provides a secure means for manually installing the master keys and initial KEKs using the dual-key-entry process. This section presents a brief description of the process and a summary of its security aspects.

The dual-key-entry process requires that the clear key be split into two or more parts; each part has the same length as the complete key. Each key part is separately entered by means of a manual key-entry device, and the key parts are combined to form the complete key by exclusive-ORing the corresponding bits of the key parts.

Master-key entry. For master-key entry, the key parts are combined in the register for the new master key. After having been entered, the key parts of a new master key never leave the physically secure boundary.

A verification pattern is provided for each newly entered key part. The pattern is computed using a cryptographic one-way function and can be used to verify whether the key part has been correctly entered.

Operational-key entry. Unlike master-key parts, it is impractical to maintain the partially completed parts of operational keys within the physically secure boundary. It may be required, for example, for one courier to enter the first part of several different keys and then have a different courier enter the second part. Before a partially completed key part leaves the physically secure boundary, it is exclusive-ORed with the control vector for the intended key type and is then encrypted using a special master key derivative. This value is returned to the program for subsequent combine operations.

Each function associated with an operational key entry requires that the intended key type of the ultimate key be specified. Thus, to obtain a meaningful result, the program must specify the same intended key type when the partial key part comes out of one step and then again when it is entered into the next step.

If the key part to be combined is the final key part, the result is encrypted under the master key derivative obtained by using the control vector for the key type of the ultimate key. The encrypted

quantity is the ultimate key in the operational state and is returned to the program.

A verification pattern derived by means of a cryptographic one-way function is also provided for each key part when it is imported.

Security aspects of dual-key entry. Security of the manual key-entry process is critical to the security of the entire system. Much effort was devoted to the design to ensure that no single person could subvert the security of the system by misusing the dual-key-entry process.

Security of the manual key-entry process is critical to the security of the entire system.

Since the manual key-entry process involves interaction with a program, the design of a secure key-entry process must take into consideration that someone may attempt to compromise the system by making subtle changes to the program.

A summary of major characteristics of the key-entry process with explanations of some of the original security concerns follows.

Dual-key entry is provided and two physical keys are required to enter a complete key: one physical key to enter the first key part and another physical key to enter subsequent key parts. Duality ensures that one person does not enter both parts.

Different manual controls are required for importing a master-key part than for importing an operational key part. This differentiation prevents the program from importing a master key as an operational key or vice versa.

Changing the setting of the manual controls clears the contents of the key-part register, thus preventing the program from stealing a key part previously entered. This situation could happen, for example, if the previous dual-key-entry process failed in the middle because of a transient hardware failure.

The contents of the key-part register are reset at first use so that dual-key-entry functions cannot be retried by the program. Resetting prevents the program from importing an operational-key part twice, as different types. For example, if those functions are allowed to be retried, key parts for a MAC-verification key could be additionally imported by the program as a MAC-generation key. The program then has the potential to forge messages with valid MACs.

As another example, if the program is permitted to combine a particular key part multiple times, then the program can cancel out the result by combining it twice. Thus, the security of a key with multiple key parts could be reduced.

Each key part of an operational key can only be imported once. The combine-key-parts function allows only one key part to be specified by the program and requires the other key part to be in the key-part register.

If all key parts could be independently imported by means of the import-key-part function, and the combine-key-parts function were allowed to accept both key parts specified by the program, the program could specify a key part of any type for both arguments and obtain a known key of the key type. The clear value of the resulting key is zero because exclusive-ORing a value with itself is zero.

The verification pattern produced by dual-key-entry functions for importing operational key parts is computed based on the value resulting from exclusive-ORing the newly entered key part with the control vector, specifying the key type of the ultimate key. Thus, the pattern can be used not only to verify whether the key part has been correctly entered but also to verify whether the correct intended key type was specified.

If this capability of detecting the key type of the imported key part were not provided, the program could subvert system security without being caught.

With all of the above, the program might still subvert system security by tricking couriers to enter each key part of an operational key twice. For

example, after the program imported a key part with the intended type, the program could display a message to the courier requesting the same key part to be re-entered because the previous one was lost due to a hardware transient error. If the courier does as requested, the program then could import the key part with a different type. To be absolutely sure about the security, couriers should never enter any key part twice on the same system. With this guideline, a secure manual key-entry procedure can be developed such that no single party, the key-entry program or any courier, can compromise system security.

Physical security

A significant amount of engineering effort and hardware cost has been devoted to prevent physical probing and intrusion, based on the assumption that the machine may be left unattended. In this section, we briefly describe major aspects of physical security provided by the first ICRF implementation.

Physical access control. Physical keys are required to utilize the manual-control panel; a physical key is also required to open the machine chest for accessing cryptographic components by service personnel.

Tamper-resistant design. The ICRF is tamper-protected. Whenever tampering is detected, a tamper indicator is turned on and all secret quantities are cleared to zeros. The facility becomes nonoperational until the tamper indication is cleared by manual intervention. Tamper detection is active regardless of whether the main power of the machine is on or off.

Tamper-resistant cables are used for connections between cryptographic components. The cable consists of parallel wires and several layers of metal shielding to protect transmission from eavesdropping. The cable is also protected against physical intrusion, such as breaking or cutting the cable, which, when detected, triggers the tamper indicator. Protection is also provided against attacks utilizing low temperature and ionizing radiation.

Each ICRF includes a tamper-protected thermal conduction module (TCM), which constitutes the volatile portion. Sensors that trigger the tamper indicator should an opening be detected are built inside the module.

The nonvolatile portion of an ICRF uses card-on-board technology and is packaged in a box that is thoroughly wrapped by tamper-detection wires. The box is then placed in a bigger container filled with liquid epoxy. After the epoxy congeals, it is difficult to break or resolve the resulting epoxy brick without triggering the tamper indicator.

Extensive hardware implementation. All cryptographic functions are performed by hard-wired circuitry inside the physically secure boundary. The CPU microprogram, which is used to interpret instructions and resides outside the security boundary, does not perform any cryptographic primitives or operations. The microprogram has no more ability than the control program to perform cryptographic attacks on system security. This design approach reduces implementation flexibility but enhances overall system security.

No scan out. Scan rings have been used extensively for error analysis in large integrated design. Normally, when hardware errors are detected, the service processor temporarily stops all hardware clocks and scans out the hardware internal status.

Scan rings are implemented in each ICRF for debugging during development and manufacturing cycles. However, the scan capability is disabled before the machine is shipped, and is designed in such a way that the capability cannot be enabled without raising a tamper condition that clears all secret quantities in the ICRF.

Concluding remarks

As far as we know, this product is the first commercially available one that implements cryptography in the CPUs of a general-purpose high-speed mainframe. This integrated approach maximizes performance and also has many advantages over the channel-attached approach: No channel is tied up; there is inherent high physical security of the facility because it is not moveable; and performance is automatically enhanced as CPU speed is increased in the future. This approach did impose some constraints on implementation flexibility. For example, it is easier to provide programmability for channel-attached devices than for mainframe CPU facilities. Therefore, channel-

attached devices could easily be modified to support new functions, such as additional PIN-verification algorithms.

Control vectors were used for key separation to discourage the misuse of cryptographic keys. Interoperability and compatibility with the IBM 4753 Network Security Processor were achieved by using the same set of control vector constants as defined in CCA. The degree of key separation was determined with careful investigation so as to provide a balanced system and to best fit the design criteria and business environments.

Acknowledgments

Many individuals have contributed to the concept and development of this product. Among them, Walter F. Bankowski, Brian B. Moore, and Julian Thomas initiated the project and established hardware design guidelines. Chris J. Holloway and Robert J. Rosenthal defined PIN-processing environments and customer requirements. Ernest T. Zooper constantly supplied market requirements. Stephen M. Matyas guided general cryptographic direction and materialized the control vector concept; Don B. Johnson frequently reviewed ICRF security aspects and coordinated the development of ICRF with the IBM Common Cryptographic Architecture. Randall J. Easter, John T. Matcham, and Vincent A. Spano, who were the chief hardware implementers, provided implementation feedback. Gina Bourbeau, Lucina L. Green, Michael J. Kelly, and R. Craig Larson specified software design criteria; Peter H. Gum, Roger E. Hough, Sandy L. Rankin, Steve J. Schmandt, and Devon Yu supplied PR/SM support requirements. Dennis G. Abraham, Ramesh K. Karne, Carl H. Meyer, An V. Le, Russ Prymak, and John D. Wilkins suggested security improvements.

Cited references and notes

1. Federal Information Processing Standard Publication 46, Data Encryption Standard, National Bureau of Standards (now NIST), U.S. Department of Commerce, Washington (January 1977).
2. American National Standard X3.92-1981, Data Encryption Algorithm, American National Standards Institute, New York (December 31, 1981).
3. The transaction here means the IMS fast path financial transaction consisting of thousands of instructions and several I/O and cryptographic operations.
4. The IBM Cryptographic Unit Support Program (CUSP) is the support software for the IBM 3848 Cryptographic Unit.
5. The IBM Programmed Cryptographic Facility (PCF) is the software-only version of 3848-CUSP. It simulates the 3848 Cryptographic Unit functions and provides the CUSP application program interface.
6. The IBM Integrated Cryptographic Service Facility (ICSF) is the support software for the ESA/390 Integrated Cryptographic Facility (ICRF).
7. D. B. Johnson et al., "Common Cryptographic Architecture, Cryptographic Application Programming Interface," *IBM Systems Journal* **30**, No. 2, 130–150 (1991, this issue).
8. R. E. Lennon, "Cryptography Architecture for Information Security," *IBM Systems Journal* **17**, No. 2, 138–150 (1978).
9. B. O. Brachtl, S. M. Matyas, C. H. Meyer, *Controlled Use of Cryptographic Keys via Generating Station Established Control Values*, U.S. patent 4,850,017 (July 18, 1989).
10. S. M. Matyas, "Key Handling with Control Vectors," *IBM Systems Journal* **30**, No. 2, 151–174 (1991, this issue).
11. American National Standard X3.106-1983, Mode of DEA Operations, American National Standards Institute, New York (1983).
12. American National Standard X9.9-1986, American National Standard for Financial Institution Message Authentication (Wholesale), American Bankers Association, Washington (August 15, 1986).
13. American National Standard X9.8-1982, Personal Identification Number (PIN) Management and Security, American National Standards Institute, New York (January 14, 1982).
14. A cryptographic one-way function has the property that, given the output, cryptographic key, and the algorithm of the function, it is impractical to derive the input value. It is also impractical to derive two or more inputs resulting in the same output.
15. PR/SM is a hardware feature that allows the resources of a machine to be dynamically shared among multiple, independent system control programs running simultaneously.
16. The term "operational key" is used here to mean a clear key to be entered as a key of any type in the operational state.

Phil C. Yeh *IBM Data Systems Division, P.O. Box 950, Poughkeepsie, New York 12602.* Dr. Yeh is a senior engineer in the Enterprise Systems Central Architecture department. He received an M.S. in computer science and a Ph.D. in electrical engineering from the University of Illinois at Urbana-Champaign in 1977 and 1981, respectively. In 1981, he joined IBM at the Poughkeepsie laboratory, where he has worked on several assignments in architecture. He has three issued patents, two patents on file, and has published several technical papers.

Ronald M. Smith, Sr. *IBM Data Systems Division, P.O. Box 950, Poughkeepsie, New York 12602.* Mr. Smith is a Senior Technical Staff Member in the Enterprise Systems Central Architecture department. He received a B.E.E. in electrical engineering from the Ohio State University in 1957. He joined

IBM in Endicott in 1957 and moved to Poughkeepsie in 1961. He worked on assignments in circuit design, central processor design, and programming before joining Central Systems Architecture in 1966. He has ten patents, four patents on file, and 13 published invention disclosures.

Reprint Order No. G321-5430.