CHAPTER ELEVEN

Applying Cryptography to Electronic Funds Transfer Systems—Personal Identification Numbers and Personal Keys

One essential requirement of an electronic funds transfer (EFT) system is that institutions must be able to join together in a common EFT network (defined as an *interchange*) such that the EFT security of each institution is independent of the security measures implemented at other institutions. Another requirement is that the process of identification or verification of a user must involve a secret value, commonly called a Personal Identification Number (PIN) which is, on the average, only 4 to 6 digits long.

To discuss EFT security from a more general viewpoint, two terms associated with personal verification are defined: an authentication parameter (AP) and a personal authentication code (PAC). An AP is a function of secret and nonsecret user-supplied information as well as nonsecret system-supplied information. A PAC is a function of the user's identifier (ID), AP, and a secret system-supplied authentication key, KA. A quantity similar to PAC is used in message authentication and is defined as a message authentication code (MAC). Examples of personal verification and message authentication are provided to illustrate the use of AP, PAC, and MAC.

After developing a set of EFT security requirements, implementations based on PIN/system keys and PIN/personal keys are discussed. It is shown that neither implementation satisfies all of the stated requirements, although the PIN/system key approach does provide adequate protection for current EFT systems.

An implementation incorporating PINs, personal and system keys (defined as a *hybrid* key management), and an intelligent secure card is discussed next. This approach, which meets the stated requirements to a higher degree, offers the potential for increased security in future EFT applications.

A glossary of terms and abbreviations is provided at the end of this chapter.

BACKGROUND

Many techniques for cryptographic authentication are used in EFT systems and in those systems being evaluated by major financial institutions and their

vendors. The purpose here is to suggest some additional techniques for consideration in the development of these systems.

Every day EFT systems electronically transfer billions of dollars between institutions and individuals. Such transactions (e.g., deposits and withdrawals) cannot be processed safely unless user identities can be validated securely and the correct, unaltered transmission of messages between network nodes (terminals, computers, etc.) can be assured.

The process of validating user identities is called *personal authentication*, *personal verification*, or *personal identification*, whereas the process of validating messages is called *message authentication*. The term personal verification is used throughout this chapter specifically to address validation of secret quantities supplied by a system user. (If a user is verified on the basis of a PIN only, the term PIN validation is commonly used.)

A user is normally provided with an embossed, magnetic stripe identification card (bank card) containing an institution identification number, the card's expiration date, and a *primary account number* (PAN).[1] The institution at which the customer opens his account, and which provides the user with a bank card, is called the *issuer*. At an entry point to the system, information on the user's bank card is read into the system and the user enters a secret quantity called the *personal identification number* (PIN). If the cardholder has supplied the correct PIN and if the balance in the account is sufficient to permit the transaction and if that type of transaction is allowed for that account, the system authorizes the funds transfer.

Consider the network configuration shown in Figure 11-1. The entry point at which transaction requests are initiated, such as a *point of sale* (POS) terminal or an *automated teller machine* (ATM), is defined as an *EFT terminal*. An institution's computer facility, which also happens to manage the connected EFT terminals, is referred to as a *host processing center* (HPC). The three HPCs shown in Figure 11-1 are interconnected via an intelligent *switch*. The switch, which can be another HPC, establishes connections between the HPCs so that information can be routed in the network efficiently. A *communications control unit* (CC), an independent device positioned in the path between an HPC and its associated EFT terminals and between an HPC and adjacent network nodes, is responsible for managing data transmissions over the communications links. Similarly, EFT terminals are assumed to provide complementary support for link management functions. Theoretically, and assumed here, the CC has the capacity to verify system users and data.

The HPC that first acts on information entered at an EFT terminal is the *acquirer* (acquiring HPC).[2] A user who initiates a transaction at an EFT terminal may be a customer of a local institution (HPC X, in which case the acquirer is also the issuer) or a remote (distant) institution (HPC Y or HPC Z). If a user can initiate transactions at an entry point not controlled by the issuer, the supporting network is called an *interchange system*.

[1] The American National Standard Institute's (ANSI) standard magnetic stripe format is given in Appendix C.

[2] The acquirer is normally the HPC associated with the EFT terminal at which PIN and card information are entered.

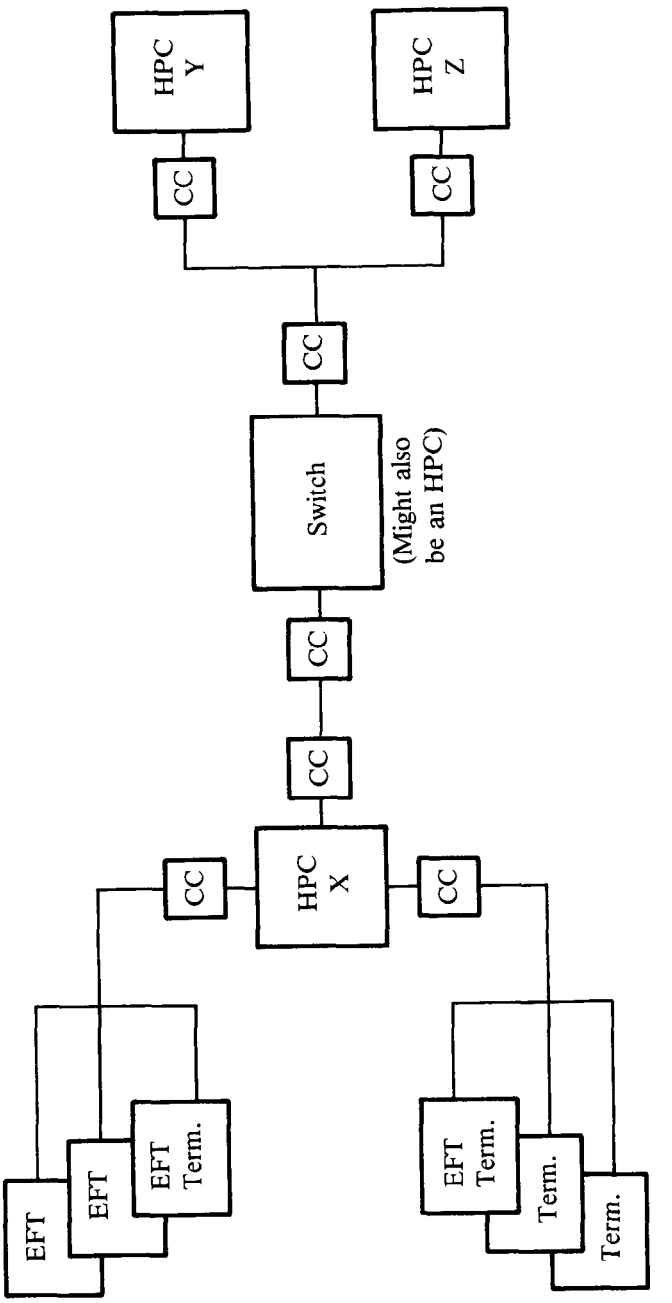


Figure 11-1. Example of an EFT Network Supporting Interchange

For example, consider a simple transaction in which cryptography is not employed. A customer wishes to use a bank card to pay a grocery bill of $35.00 and to receive an additional $50.00 in cash. Assume that the grocer's account is with institution X and the customer's account is with institution Y. The customer's card is inserted into the EFT terminal, either by the customer or by an employee of the retailer attending the EFT terminal, and the customer enters his PIN via a suitable entry device such as a keyboard or a PIN pad, which looks and operates much like a hand-held calculator. Similarly, the grocer enters a transfer request for $85.00 to be transferred from the customer's account to the grocer's account ($35.00 for the groceries plus the $50.00 to be given to the customer).

The information entered at the EFT terminal is assembled into a *debit request* message. This message or *transaction request*, which includes the customer's PIN, his account number (PAN), and suitable routing information, is then sent via the acquirer (HPC X) and switch to the issuer (HPC Y).

Upon receiving the debit request, HPC Y verifies that the PIN correlates properly with the customer's PAN, and that the customer's account balance is sufficient to cover the $85.00 transfer. If the PIN check fails, the user is normally given at least two more chances to enter the correct PIN. If after the additional trials the PIN is still rejected, HPC Y sends a negative reply to HPC X. If the PIN is correct but the account balance is insufficient to cover the transfer, HPC Y denies the debit request by sending a negative reply or *debit refusal* (insufficient funds) message to HPC X. A message is then sent via the network to the grocer's EFT terminal indicating to the grocer that the funds transfer has been disapproved.

If the debit request is approved, HPC Y records the debit request, reduces the customer's account balance by $85.00, and transmits a positive reply or *debit authorization* message back to HPC X. Upon receiving the debit authorization HPC X takes two actions. A message is sent to the grocer's EFT terminal, indicating to the grocer that the funds transfer has been approved. Then HPC X credits the grocer's account with $85.00. This completes the transaction. (Although other protocols are possible, the one described above will be assumed throughout the present discussion.)

When personal verification is performed by the issuer's HPC (or the HPC of another designated node), the process is said to operate in the *on-line* mode. If personal verification is performed by the terminal, the process is said to operate in the *off-line* mode. If only the terminal and communications controller are involved, the process is said to operate in the *off-host* mode.

In an on-line environment, the highest level of data security can be provided. However, if the system is to be kept available even when the HPCs are not operating (e.g., during weekends, holidays, maintenance, etc.), an off-line or off-host mode of operation must be supported. In general, there is insufficient storage to maintain a verification table at the EFT terminals, and users must be verified without benefit of the HPC's files (of hundreds of millions of characters) which contains the PAN/PIN information of each user. Therefore, off-line and off-host authentication techniques must contend with this limitation, which ultimately results in a decrease in security.

SECURITY EXPOSURES IN EFT SYSTEMS

An EFT system is defined to have the following four components:

1. Communication links
2. Computers
3. Terminals
4. Bank cards

Security considerations for each of these are discussed below (see also reference 1, where part of the material has been taken).

Communication Link Security

Communication links are highly vulnerable to interception of messages by a number of techniques which permit passive (listening), and/or active (data alteration/substitution) attacks. Where there are public telephone line connections between computers and terminals, and that is most common, one normally needs a physical connection to intercept the information. With satellite or microwave transmissions, on the other hand, a physical connection is not required since an appropriate antenna allows the communications channel to be breached between the sending and receiving stations.

If certain data were altered, illegitimate authorization of a transaction could occur. For example, money could be diverted to the wrong institution or account, transaction amounts could be changed, or a debit refusal message could be converted into a debit authorization message. Message authentication techniques eliminate these exposures. They allow the receiver to determine where the message originated, if it is current, what its destination is, and, most importantly, if it has been altered.

Computer Security[3]

Today, time-sharing, real-time interactive terminal communications, and computer-to-computer data links are all common. These technological innovations, coupled with the growth of computer usage, have increased the opportunities for computer abuse. Access to the computer can be gained through a remote terminal or other peripheral device such as a card reader. Thus programs or data stored in or being processed by a computer system could be copied, altered, replaced, or even destroyed. To cope with these problems, a combination of physical security, procedural protection methods, and cryptography can be used.

Cryptography alone does not solve the computer security problem. Other methods are required such as access control, store and fetch protection, and the like. This contrasts with communication security where cryptography may indeed be the only method needed to provide protection.

[3] The security exposures discussed here apply in general to programs or data stored in or processed by a communications control unit or terminal control unit.

Terminal Security

Whenever cipher keys reside in terminals, some physical security is mandatory. Without it, an opponent may be able to probe for a key or change its value. Therefore, both the integrity of nonsecret parameters and the confidentiality of secret parameters must be preserved. In well-designed systems, considerable time would be required to probe for a key successfully.

However, sufficient time to probe for a key would be available if an opponent could steal a terminal. (Note that a resident key is normally stored in volatile storage maintained under battery power so that the key will not be erased if main power is interrupted.) Elaborate interlocks designed to detect penetration and to erase the terminal's key(s) automatically could be defeated with enough time and resources. In that case, previous data encrypted or transformed with the secret terminal-resident key would be exposed. On the other hand, removal of a terminal is likely to be detected and the proper response is to invalidate the terminal and key in the supporting network. This would protect future encrypted data.

Because of the trend toward employment of large numbers of inexpensive terminals (which may be installed in relatively nonsecure locations), the secrecy afforded terminal-resident cryptographic keys may be very little. An inexpensive terminal cannot have elaborate or sophisticated defenses, and penetration of the terminal without its physical removal from the premises becomes increasingly more difficult to defend against.

Even a public-key cryptosystem (PKC) (see Chapters 2 and 9 and references 2 and 3), employing public keys in the EFT terminals, would not provide a secure solution if there were no physical security at the entry points. A PKC would eliminate the need to keep certain (public) keys secret, but it would still be necessary to protect the integrity of these public keys. Otherwise, an attack is possible wherein an opponent replaces the installed public key with a key of his or her own choosing. The opponent, knowing the corresponding secret key, could then produce forged verification information that would be accepted by the terminal. In addition, a public key would only allow the terminal to authenticate transaction response messages received from the issuer. A secret key would still be required for generation of the MACs on transaction request messages sent to the issuer. Hence, even the public-key approach requires a secret key at the entry point.

> *It may appear that the solution to the exposed terminal problem is to use only externally supplied secret keys (i.e., to use personal keys instead of system keys.) However, it will be shown later that personal keys alone will not lead to a secure implementation.*

When transactions are conducted entirely at an EFT terminal (such as a cash-issuing terminal operating in the off-line mode), only personal verification is required—message authentication between the EFT terminal and the issuer is, by definition, not needed. In that case, a public-key cryptosystem with a public key installed in the EFT terminal would suffice to permit personal verification. However, unless the integrity of the public key can be ensured, the system could be attacked.

EFT Terminals in Nonsecure Environments

Ordinarily, information entered into or stored within an EFT terminal would be protected as a consequence of the physical security routinely provided by the owner of an establishment, e.g., the retailer, wherein an EFT terminal is installed. However, if the retailer or an employee of the retailer becomes an opponent, the secrecy and integrity of information at the entry point no longer can be maintained (see the section entitled EFT Security Requirements). Although its physical surroundings (the building, locked doors, employees of the retailer), to some degree, protect an EFT terminal from outsiders who are not authorized to have access, they do not protect an EFT terminal from insiders who are authorized to have access (e.g., the retailer, clerks, cashiers, and sales personnel employed by the retailer).

Those with authorized access to the retailer's premises could subvert security in several ways. For example, by

1. Illicitly reading (skimming) card information.
2. Probing the EFT terminal for secret keys.
3. Replacing keys, algorithms, and hardware devices with parameters, procedures, and devices under the control of the opponent.
4. Tapping (obtaining electronically) information entered at the EFT terminal.
5. Tapping information sent to the issuer.

The insider has an advantage over the outsider simply in terms of the time available to carry out attacks. But the threats are even more insidious because of the insider's ability to coordinate one activity with another (e.g., skimming bank cards and simultaneously tapping the output line).

Fake Equipment Attack

Instead of using indirect methods to obtain the PIN, an opponent can recover PINs directly by subverting the entry process. For example, an opponent could replace the PIN pad[4] and terminal with devices that will display or print the entered values, PIN and personal key (KP), to the opponent stationed out of view. Each PIN and KP so obtained is recorded and reentered into the real terminal which is kept hidden. Upon receiving the transaction response from the issuer, the opponent sends a comparable message to the bogus terminal, leading the user to believe the transaction was completed without interference.

This fake equipment attack points out the vulnerability of entering card and PIN information into devices whose security and integrity cannot be assured. It clearly demonstrates once again that cryptography does not provide the solution to the problem of protecting secret information if that information can be attacked before it is entered into the system. It also

[4] This is a device attached to the terminal, often with an integrated encryption capability, specifically designed to facilitate PIN entry.

argues strongly in favor of a design in which secret user information need not be exposed at the entry point. One approach already suggested involves an intelligent secure card (see the sections entitled Bank Card Security and Personal Key Approach with an Intelligent Secure Card). Since a secret component (KP) is stored on the card, and the card can neither be read nor skimmed by the opponent (the retailer, in this case), its use prevents the exposure at the entry point. The protocol should be such that there is no need to transfer secret card information to another device during normal operation (i.e., transformations involving the secret information are performed directly on the card).

Bank Card Security

The most convenient method of identification or authentication currently in use by financial institutions encompasses something the customer has, a bank card, and something the customer knows, a PIN. The unique correspondence of the account number contained on the card's magnetic stripe and the PIN memorized by the customer serves to identify the customer. Possession of the card without knowledge of the PIN, or knowledge of the PIN without the corresponding card, is insufficient for an imposter to gain access to the system.

Magnetic Stripe Card

Security exposures exist today because it is relatively easy to counterfeit or duplicate magnetic stripe cards. Special knowledge of the card's recorded data is not required to duplicate a card and there are several methods of transferring data from one card to another. Moreover, it does not matter if information on the card is encrypted or in the clear; cryptography does not protect against this exposure.

Two methods of duplicating encoded data previously recorded on the magnetic stripe of a bank card are *skimming* and *buffer recording* [1]. One technique for skimming involves placing a piece of recording tape over the magnetic stripe of a good card and applying heat (e.g., from a common household iron). The recording tape is then placed over the blank stripe of another card and heat is again applied. With this technique, it is possible to produce several duplicate cards without seriously degrading the quality of recorded information on either the original or the duplicate card.

Buffer recording produces a duplicate card of higher quality, but the method is more complex and more expensive than skimming. An electromagnetic reader (a device similar to a tape recorder) and buffer storage are required. Data, read from the card, are stored in the buffer. Later, the data can be read from the buffer and written on a blank card.

Duplication is more readily detectible if cards are constructed with some random property that changes from card to card and which is subsequently verified as part of the users transaction processing [1]. One such technique involves two sets of interleaved magnetic bars printed on the card's inner core. This forms a protective magnetic fingerprint with no two cards being alike. In addition, cards can be constructed of heat- and pressure-sensitive materials that invalidate a card if there are attempts to alter or duplicate it.

But mechanisms that discourage card duplication also increase the cost of both the card and the card reader. If only a few properties are added, the card is less expensive to read but relatively easy to duplicate. As more and more random properties are added, the card becomes more expensive to read and more difficult to duplicate. Furthermore, the special properties must be checked each time the card is read; otherwise a counterfeit card without these properties could be used instead.

Intelligent Secure Card

Recent advances in technology have made it possible to embed a microprocessor on a plastic card permitting identification and authentication computations to be performed directly on the card rather than in the logic of the system entry point device. In addition, small storage arrays permit important customer account information to be stored on the card, thus providing automated record keeping functions equivalent to those provided by a savings passbook. The net effect is to produce a card that is intelligent as well as secure [4, 5]. The intelligent secure card is also referred to as chip card and smart card.

IDENTIFICATION AND AUTHENTICATION OF SYSTEM USERS

A primary objective in the design of a practical authentication method is to find an inexpensive, practicable technique that is difficult to penetrate. The problem is that of properly balancing security against human factors and cost.

User characteristics are grouped into two sets: *transferable*, which means they could be forged, and *nontransferable*, which implies they cannot be forged.

Transferable User Characteristics

Verification can involve something a person has (a magnetic stripe identification card) or knows (a password). Passwords are commonly used, but they are not very secure. They may be compromised without the user, the institution's management, or its auditors knowing.

Nontransferable User Characteristics

Methods of verification involve testing for a unique personal characteristic such as something a person is (voiceprint, fingerprint, hand geometry) or something a person does (handwritten signature). One method, based on a handwritten signature, makes use of a signature pen capable of measuring pen pressure and pen acceleration. Experiments have shown that these two parameters are unique in the process of writing a signature [6].

When a person signs his or her name, the acceleration and pressure motions

are not consciously controlled. This can be demonstrated by the close match between a person's signatures written with eyes open and those written with eyes closed. Further development work, however, is needed to bring handwritten electronic signature verification into everyday use.

For the present, a password used in conjunction with a magnetic stripe card and authentication processing performed at an HPC represent the most practical means for achieving personal verification.

REQUIREMENTS FOR PERSONAL VERIFICATION AND MESSAGE AUTHENTICATION

The problems to be solved with cryptography are:

1. Verification of system users, referred to as personal verification.
2. Verification of data (to check for origin, true content, timeliness, and destination), referred to as message authentication.

It must be recognized that

> *applications requiring personal verification very frequently require message authentication, and since both processes are closely related and neither must be allowed to weaken the security of the other, it is prudent to seek a cryptographic solution to both problems simultaneously.*

For example, when the entry node must take some action in response to a transaction request (e.g., to dispense cash or not), the node must receive a message notifying it that the user has or has not been accepted (Figure 11-2). For detection of an attack where a "User Not OK" (or negative) reply is changed to a "User OK" (or positive) reply, message authentication must be used. In this application, the best personal verification scheme could be circumvented were there no message authentication.

The personal verification process starts with the user providing *personal verification information*. This can be categorized in the approaches discussed here as user-remembered information (e.g., a PIN or a *password*) and user-supplied information stored on the bank card (e.g., a primary account number, PAN, or a cryptographic key). If a cryptographic key is stored on the bank card, it is referred to as a *personal key* (KP).

The PAN is also referred to as a *personal identifier*, or *identifier* (ID). Using a unique ID is better than using, for example, the name of a person, since people's names are not always unique. In the discussion that follows, it is assumed that an opponent has knowledge of user IDs. This is a practical assumption, since ordinarily no special precautions are taken to ensure their secrecy (i.e., they are treated as nonsecret quantities). Obtaining them would not be too difficult, since they are frequently used for identification and auditing purposes and are transmitted, stored, and printed on documents in unencrypted form.

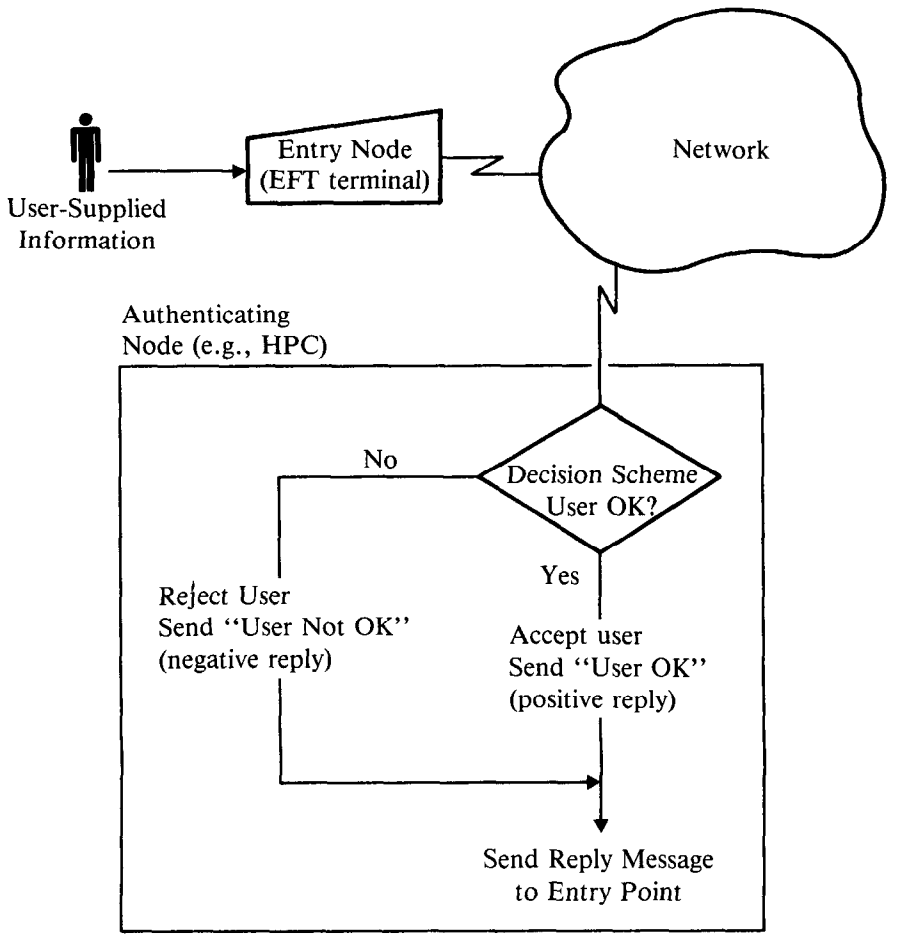
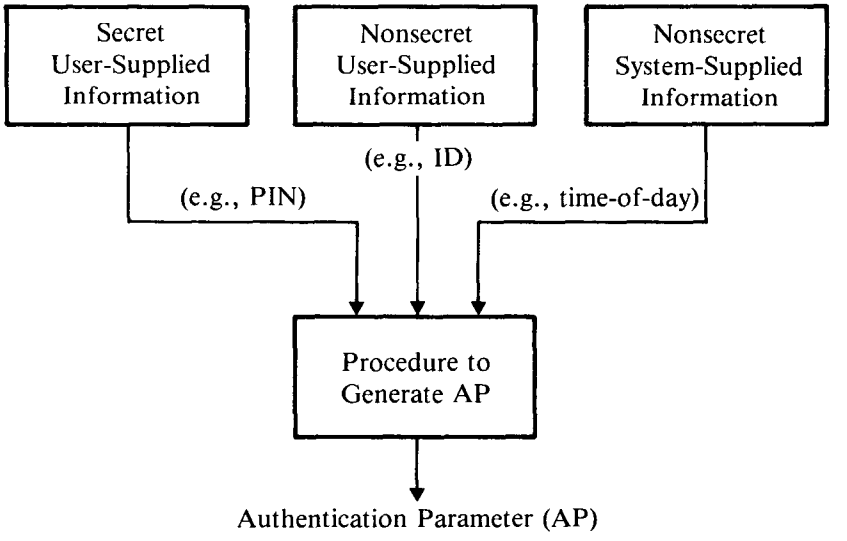


Figure 11-2. The Personal Verification Process

Authentication Parameter

The following discussion of several possible methods for performing personal verification assumes that part or all of the user-supplied verification information is first subjected to a transformation at the entry point. This process creates another quantity defined as a *personal authentication parameter*, or *authentication parameter* (AP) for short (Figure 11-3). An AP is a function satisfying the following conditions:

1. It is always a function of secret user-supplied information, which may or may not be shared with the issuer.
2. It may or may not be a function of other nonsecret user-supplied and/or system-supplied information.
3. It is a function only of the information specified in conditions 1 and 2



Note: Proper coupling of nonsecret with secret information can lead to a quantity more resistant to analysis.

Figure 11-3. Transformation of User-Supplied Verification Information to a Personal Authentication Parameter at the Entry Point

above (i.e., AP does not depend on secret information known to anyone other than the user or the user and the issuer).[5]

It is fundamental to any cryptographically secure procedure for personal verification that the secret information supplied by the user (which is used by the issuer to authenticate the user) be transformed at the entry point under some cryptographic process to ensure that the secret user-supplied information cannot be ascertained by the unauthorized. It is significant that the secret information used in the computation of AP is known only to the user or to the user and issuer. This is a requirement if personal verification in an interchange environment is to be achieved exclusively between the user and issuer.

For personal verification, AP can be used either alone (by storing an appropriate reference,[6] AP of reference, in a verification table) or in conjunction with other information related to AP via a special cryptographic key defined solely for authentication purposes. The nonsecret information in the computation of AP may be time-invariant (e.g., the primary account number), time-variant (e.g., a time-of-day clock reading), or both. If AP is time-invariant, an AP of reference may be precomputed and stored in a verification

[5] In a discussion of PIN/system key approaches to EFT security, it is likely that the definition of AP would be broadened, allowing AP to depend additionally on secret system-supplied information (e.g., a secret system key).

[6] A reference is a quantity uniquely related to the item to be verified or authenticated. It is a parameter computed or designated, and used, by the authenticator as part of the authentication process.

table at the issuer. If AP is time-variant, then an AP of reference must be dynamically computed.

If AP depends on time-variant information as well as on the transaction request message, it can also be used as a *message authentication code* (MAC). Message authentication (via the MAC) is used primarily to detect stale or bogus messages inserted into the communications path and fraudulently modified messages traversing nonsecure communications systems. However, in such a situation, AP can serve a dual purpose—i.e., it can be used for personal verification as well as message authentication.

The function defining AP may be simple or complex. For example, AP may merely define a process of concatenation of parameters, or it may involve successive encryptions and decryptions, such that it is computationally unfeasible to invert the process (or one-way function) and find secret user-supplied information from AP. The developed set of security requirements places additional constraints on the specification of AP.[7]

Some specifications for AP presently under consideration by the American National Standards Institute (ANSI) technical committees X9 are AP = PIN and AP = PIN||ID, where PIN||ID denotes the concatenation of PIN and ID. If secret information (a 56-bit personal key, KP) is also stored on the bank card, then a better choice for the authentication parameter is AP = $E_{PIN \oplus KP}(ID)$ as discussed later in this chapter. (Note that $\oplus$ denotes modulo 2 addition and $E_K(X)$ and $D_K(Y)$ define encipherment of X with key K and decipherment of Y with key, K, respectively.)

Personal Authentication Code

In cases where a copy of AP can be safely stored in a verification table at the authenticating node, or where it is possible for the authenticator to compute an AP of reference either by recreating the information that is used to compute each user's AP or by safely storing that information in a table at the authenticating node, personal verification can be based solely on ID and AP. However, if the integrity of the verification table or the secrecy and integrity of stored information used to compute the AP of reference cannot be ensured, or if it is not possible to recreate the information used to compute each user's AP of reference (e.g., if PINs and KPs are selected independently), then personal verification can be based on ID, AP, and a *personal authentication code* (PAC). PAC is a function satisfying the following conditions:

1. It is a function of ID, secret user-supplied information, a secret key known only to the issuer (or authenticator), and possibly other non-secret information.
2. It does not depend on secret information known to anyone other than the user and the issuer.

[7]For example, AP must be a one-way function of the input parameters which define it, and the secret user-supplied information and AP must each contain on the order of 56 independent bits.

The distinction between AP and PAC is as follows. Either a part of the secret information used in the computation of AP is known only to the user, or, all the secret information used in the computation of AP is shared between the user and issuer. On the other hand, it is always the case that a part of the secret information used in the computation of PAC is known only to the issuer.

There are two approaches for implementing personal authentication codes in an EFT system. In the first approach, PACs are stored directly on the magnetic stripe of the user's bank card thereby eliminating the need for a verification table at the issuer. When needed, the PAC is supplied to the system by the user, and together with AP it is forwarded to the *authenticator* (synonymous with authenticating node). In the second approach, the PACs are stored in a verification table in the issuer's system and only AP is sent to the issuer. In either case, the correct (ID, AP, PAC) relationship is checked via the authentication key, KA.

The authentication parameter and personal authentication code are introduced to broaden and generalize many of the concepts. At the same time, they allow different EFT systems to be discussed using a common, consistent terminology.

Personal Verification Using AP Only

Consider an AP-authenticating procedure that uses a verification table at the issuer, and therefore is an on-line verification method. Assume, for this example, that the quantity to be verified is $AP = E_{KP \oplus PIN}(ID)$, where KP is a 56-bit personal key, and a copy of AP is stored in the verification table of the issuer's HPC during the initialization process. Later, during the verification process, the quantities KP*, PIN*, and ID are entered at an entry point of the system by a user who wishes to be authenticated.[8] At the entry point, $AP^* = E_{KP^* \oplus PIN^*}(ID)$ is generated and transmitted (together with ID) to the authenticator. If the stored AP of reference (indexed in the verification table by ID) agrees with the received AP*, the authenticator concludes that $KP^* \oplus PIN^* = KP \oplus PIN$ and the user's identity is ID (as claimed). Otherwise, the user is rejected.

For the approach to be secure, it must not be possible for an opponent to violate the integrity of the verification table (e.g., by overwriting a stored value of AP with another value of AP). If this were possible, the opponent could define his own personal key (KP*) and PIN* for some existing ID, generate $AP^* = E_{KP^* \oplus PIN^*}(ID)$, and overwrite AP with AP* in the verification table. Subsequently, the opponent could supply KP*, PIN*, and ID at an entry point and be accepted by the system as the user whose identifier is ID.

It is assumed, although not shown at this point, that time-invariant AP values (as described in the example above) are included in the transaction

[8] KP* and PIN* denote the personal key and PIN entered by the user and KP and PIN denote the personal key and PIN (of reference) stored in the system. If the user is legitimate and does not make an error in entering his personal key and PIN, then $KP^* \oplus PIN^*$ equals $KP \oplus PIN$.

request messages sent from the EFT terminal to the issuer, and message authentication is implemented such that the issuer can validate the content, timeliness, sender, and intended receiver of all received messages (see the section entitled Message Authentication Using MAC). Without message authentication, a procedure for personal verification based on time-invariant APs could be attacked in the following manner. A microprocessor is connected to the communications line of the EFT terminal to be attacked. The opponent initiates a transaction at the designated terminal using a bogus PIN and KP and an ID corresponding to a previously intercepted value of AP. The microprocessor is programmed to intercept the transaction request message and replace the bogus value of AP (computed by the terminal from the bogus values of PIN and KP) with the (correct) previously intercepted value of AP. The opponent, masquerading under the assumed ID, is thus validated by the issuer.

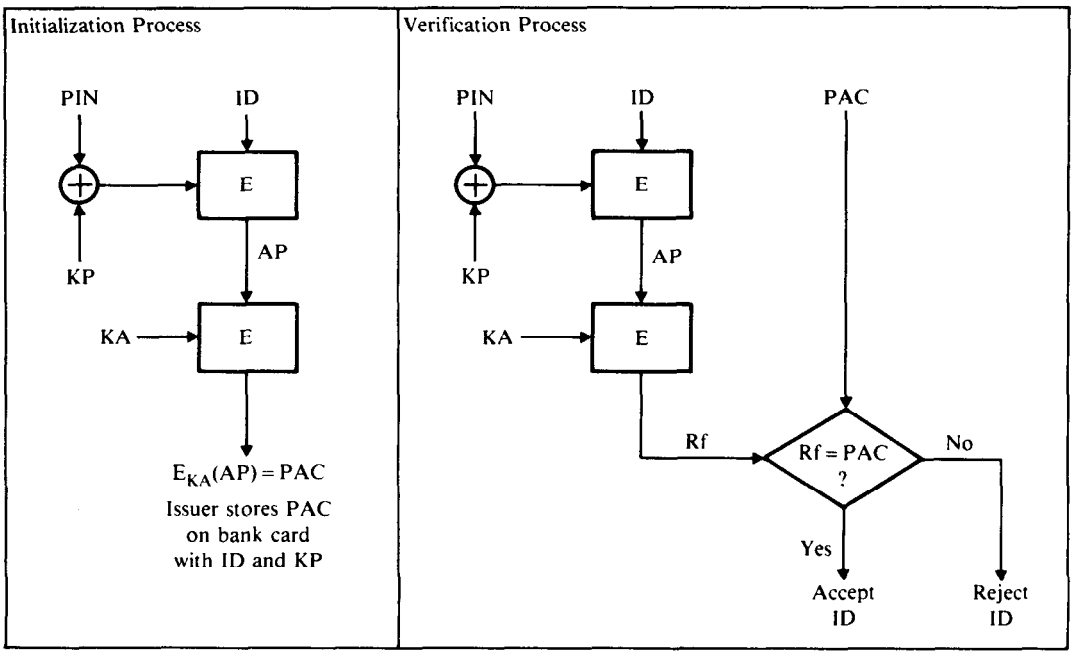
Personal Verification Using AP and PAC

An alternative procedure which does not use a verification table is also possible. Consider the on-line case where $AP = E_{KP \oplus PIN}(ID)$, $PAC = K_{KA}(AP)$, and KA is an issuer-controlled, secret authentication key. At the time KP and PIN are created, the issuer (who has KA) computes and stores PAC on the user's bank card (Figure 11-4). Since KA is a secret key, only the issuer can create valid PAC values. During the verification process, both AP and PAC are transmitted to the issuer. This permits the issuer to authenticate AP by enciphering it under KA and testing the result for equality with PAC (Figure 11-4).

To verify the user, the authenticator creates a dynamic *reference* (Rf), or more specifically, a PAC of reference, defined by $Rf = E_{KA}(AP)$. This reference is then compared with the received PAC. If Rf = PAC,[9] the authenticator concludes the following: The received quantities AP and PAC are properly related via the secret authentication key. Since these corresponding quantities could only be generated by someone who knows (or has access to) the secret authentication key, AP is accepted as genuine. Since AP depends on KP, PIN, and ID, authentication of AP also authenticates the triple (KP, PIN, ID) provided that message authentication is also employed to ensure that the transmitted ID is unchanged (i.e., one can conclude that AP was computed from a valid triple).[10] On the other hand, if AP or PAC or both do not have the correct values, then $Rf \neq PAC$ (with high probability) and hence this condition can be detected (with high probability).

[9] It is assumed that AP and PAC are part of the transaction request message whose content, timeliness, sender, and intended receiver are authenticated by the issuer.

[10] PAC is really only coupled to AP. Authentication of AP does not by itself ensure that the claimed ID is the ID in the triple (KP, PIN, ID) that was used to compute AP. Message authentication is also required to ensure that the claimed ID has not been changed. Coupling AP and ID to PAC is discussed in the section Personal Verification with Independent PINs and Personal Keys (Equation 11-4).



Legend:

AP:	Authentication Parameter	64 Bits
ID:	User Identifier	$\leq$ 64 Bits
KA:	System Authentication Key	56 Bits
KP:	Personal Key	56 Bits
PAC:	Personal Authentication Code	64 Bits (Truncation Possible)
PIN:	Personal Identification Number	$\leq$ 56 Bits
RF:	Reference	64 Bits (Truncation Possible)

Figure 11-4. A Method for Achieving Personal Verification

Message Authentication Using a MAC

To simplify the discussion of message authentication, it is assumed that data secrecy is not required, i.e., a message (M) is sent in the clear and therefore can be read by an opponent.[11] Furthermore, it is assumed that M consists of only one 64-bit data block. The message authentication code (MAC), defined here as quantity $E_K(M \oplus Z)$, is produced by the modulo 2, addition of M with a nonsecret initializing vector (Z) and encipherment of the result with a secret authentication key (K).[12] The initializing vector, Z (used to

[11] If secrecy and authentication are desired at the same time, the data are first encrypted and then one of the authentication methods described below is applied to the resulting ciphertext.

[12] Typically, a message would exceed 64 bits and would contain identification information such as the user's ID, the computed value of AP, a PAC value (if used), as well as other data such as the transaction code and the transaction amount. In that case, a MAC may be generated by using a form of chained block encryption wherein Z is added modulo 2 to the first block of plaintext and each block of ciphertext is added modulo 2 to the next block of plaintext.

introduce time-dependent information into the authentication procedure), may be established between the communicants as part of the session initialization process.[13] By transmitting M and MAC to the receiver, the authenticity of M can be checked by comparing the received MAC for equality with a system-generated MAC of reference (Figure 11-5).

There is a close analogy between personal verification where a PAC is transmitted to the authenticator together with AP (Figure 11-4) and message authentication where a MAC is transmitted to the authenticator together with the corresponding message. (The procedure in Figure 11-4 is actually the same as Figure 11-5 if AP is replaced by $M \oplus Z$, PAC is replaced by MAC, and KA is replaced by K.) The difference is that PAC is precomputed whereas MAC is dynamically computed.

To create a reference (Rf), or more specifically a MAC of reference, the authenticator encrypts $M \oplus Z$ with K using the same procedure as the sender. This reference is then compared with the received MAC. If Rf = MAC, the authenticator concludes the following: The received quantities M and MAC are properly related via the secret authentication key. Since these corresponding quantities could only be generated by someone who knows (or has access to) the secret authentication key, M will be accepted as genuine. On the other hand, if M or MAC or both do not have the correct values, then $Rf \neq MAC$ (with high probability) and hence this condition can be detected (with high probability).

Although prevented from generating a proper MAC for an arbitrary M, an opponent could present a previously intercepted message and MAC. To permit detection of such an event, the MAC must be time-variant—assured here by the quantity Z. It could also be assured by including a unique message sequence number in M, in which case quantity Z (Figure 11-5) would not be needed. In either case, the receiver could then detect stale messages injected into the communication path as well as deleted messages.

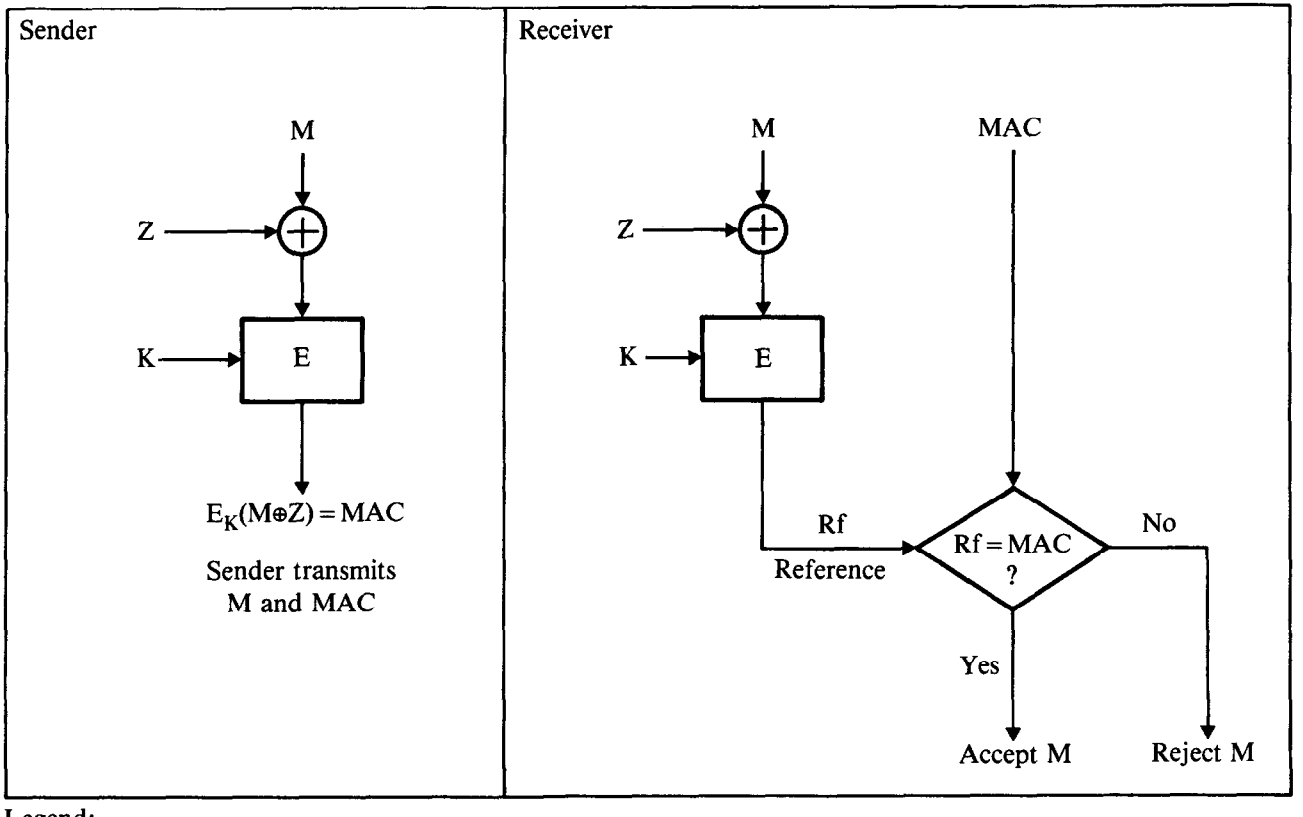
In the discussion that follows, personal verification is based on an AP value received in the transaction request message M. Message authentication is based on a MAC attached to the transaction request message. AP is used to validate the originator of the message, whereas MAC is used to validate the content of the message (including the received AP).

EFT Security Requirements[14]

The following EFT security requirements assume that the system must be secure from both insiders (those that have access to privileged system interfaces and internal functions of the system) and outsiders (legitimate users or opponents who have access only to external system interfaces).

[13] It is assumed that the process of establishing Z is such that an outsider cannot alter or predict its value. Otherwise, he may be able to find M* and Z* such that $M^* \oplus Z^* = M \oplus Z$. He could then change $[M, E_K(M \oplus Z)]$ to $[M^*, E_K(M \oplus Z)] = [M^*, E_K(M^* \oplus Z^*)]$ which would authenticate when Z* were used as the initializing vector.

[14] ©1981 IEEE. The section on EFT security requirements is reprinted in part from the authors contribution to the Proceedings of the IEEE 1981 Symposium on Security and Privacy, April 27–29, 1981, Oakland, California [7].



Legend:

K:	Authentication Key	56 Bits
M:	Message	64 Bits
MAC:	Message Authentication Code	64 Bits (Truncation Possible)
Rf:	Reference	64 Bits (Truncation Possible)
Z:	Initializing Vector	64 Bits

Figure 11-5. A Method for Achieving Message Authentication

Although the subject of auditability is not specifically addressed in this chapter, it should be understood that auditability is a prime objective of any well-designed security system. Commonly recognized auditing practices include accountability, cross checks, and sensitivity checks. The EFT security requirements listed below do not obviate the need to implement supportive security practices (e.g., auditing practices, access control, and physical security).

It is not possible to meet every one of the requirements. In some cases, particularly in the case of requirements 1 through 3, they should be considered goals more than requirements. Requirements 4 through 17 assume that cryptography is used (when appropriate) to protect the privacy of data and to authenticate data and users.

Requirement 1. The process of entering information at an EFT terminal must be protected; i.e., the integrity and secrecy of information entered into the EFT terminal must not be compromised as a consequence of the entry process. Furthermore, the information flow within a terminal must be protected.

Requirement 2. Information stored permanently or temporarily within an EFT terminal must be protected; i.e., the integrity of public information and the secrecy as well as integrity of private information must be assured.

Requirement 3. System managers and maintainers of the system (insiders) must be denied the opportunity to misuse the system accidentally or intentionally.

Requirement 4. Messages and system users must be authenticated.

Without personal verification, an opponent who may be a legitimate system user could pose as any system user. Without message authentication, an opponent could alter messages or inject previously communicated messages back into the communications network. This in turn could cause the system to provide some resource to the opponent for which he lacks authorization. Thus, an EFT system (and, for that matter, any communications system) will function securely only if users and messages are genuine.

In implementing authentication methods, the system must provide security measures (other than cryptography) to prevent subversion (e.g., bypassing) of authentication procedures. To the extent that subversion is prevented, the system is said to have the property of integrity.

Requirement 5. The security of the personal verification process implemented at one institution must not depend on security measures implemented at other institutions.

The requirement is based on the principle that a well-designed procedure for personal verification should be such that a user can be authenticated by the issuer without exposing or disclosing (to others) the secret information used in that process and without depending on others, including terminals, terminal control units, communications control units, and Electronic Data Processing (EDP) systems in the network (except the issuer's HPC) to pro-

tect the secrecy of that information. The personal verification process must be such that it is unaffected by a breach in security at another node.[15]

The requirement can be satisfied only if the cryptographic parameters, keys, and operations used to authenticate users are controlled and managed by the cardholder and card issuer. In that case, each issuer or institution can specify the security level it desires, independent of other institutions.

Verification could also be performed by some other node, such as a switch designated by the issuer. In this case, the security of the personal verification process would depend on security measures implemented at the designated node. Although employing the services of a designated node should be an option available to the issuer, it should never be a requirement in order to transact business in an interchange.

Requirement 6. Each user must have only one set of user-supplied verification information, and it must be possible to initiate personal verification with this one set of information at any entry point to the system.

Otherwise, institutions could not join together in an interchange unless users were required to know or possess a different set of verification information for each different institution in the system. This would be very impractical.

Requirement 7. The personal verification process must involve user remembered PINs and that process must be secure even if PINs have only four digits.

The PIN is the means by which the EFT system prevents a lost, stolen, or forged bank card from being used at an entry point by someone other than the true cardholder. However, a limitation which any design must cope with is that of human factors. It has been determined that, on the average, people cannot reliably remember a PIN which is longer than six digits. In addition, people prefer PINs as short as possible. In today's EFT systems, the PINs in use are normally between four and six digits long, which is a compromise between usability and security. It is important that this limitation be kept in mind when verification schemes are designed. (Normally, PIN length is a parameter that the issuer can select.) For instance, since verification is based on PIN and card information, it follows that PIN exhaustion (trying one PIN after the other) at the entry point interface is always feasible if card information is available, although six-digit PINs provide greater protection against such attacks than do four-digit PINs. However, exhaustive attacks should not be feasible anywhere else in the system (e.g., within the nodes via a programming interface).

[15] Security at the node designated to perform verification can be obtained through the use of a cryptographic facility (Chapter 4) or security module [8] (see also Implementation of Fraud Prevention Techniques, Chapter 10). With such an approach, important authentication information does not exist in the clear except within the protected confines of hardware. However, additional security measures are needed to prevent unauthorized users from directly exercising the cryptographic facility or security module.

Requirement 8. It must be unfeasible to derive user-remembered information solely from a bank card.

Since bank cards may become lost or stolen, the strength of the personal verification process would be diminished if a PIN could be derived from card information. For the same reason, an opponent who creates a bogus card should not also be able to create a valid corresponding PIN.

Requirement 9. Data can be authenticated only if sufficient nonsubvertible redundant information related to the data to be authenticated is introduced.

Assuming that the sender can generate arbitrary information (e.g., messages, passwords, etc.), the authenticator would have to treat any arbitrary pattern of bits as valid. In simple terms, data cannot always be authenticated by testing only the data. To authenticate random data, the authenticator must be provided with some additional, redundant information that is related to the information to be authenticated and cannot be subverted by an opponent.

The required redundant information may either be an integral part[16] of the data to be authenticated or it may be separate from it (defined here as an authentication code). In a communications system, analogously error-free transmission over a noisy channel requires the introduction of redundancy. Although, in this case, the redundancy is a countermeasure against the introduction of random (unintelligent) noise. In cryptographic applications, one is additionally concerned with the effects of deliberate tampering (intelligent noise) introduced by an intruder.

Requirement 10. Personal verification and message authentication require a reference to be available to the authenticator at the time authentication takes place. This reference, or the process used to generate the reference, must be defined and agreed upon in advance. It must be such that the integrity of the reference (and sometimes its secrecy) or the process that generates the reference or both can be assured by the authenticator.

The authentication process requires a comparison of two quantities (directly or indirectly). If the two quantities are equal, or correlate properly, the quantity to be authenticated is considered genuine; otherwise, it is not. At least one of these two quantities must be determined by the authenticator, and in this discussion, that quantity is called the reference.

Requirement 11. When a personal authentication code is used, it must be a function of user ID, the authentication parameter associated with

[16] For example, consider the (not necessarily practical) case where messages are formed by using only a character set of 36 symbols (A thru Z and 0 thru 9) and each character is represented by 8 bits. In that case, only 36 out of a possible 256 plaintext characters are used. A received message is accepted only if each decrypted character is in the set of 36 valid plaintext characters.

that ID, and a secret authentication key managed, controlled, and known only to the authenticator (issuer, switch, or terminal).

The EFT procedure or protocol must ensure that forged values of PAC will be unacceptable to the authenticator. It must not be possible to subvert the process of personal verification by supplying forged parameters or altering or replacing stored system parameters. This can be accomplished with a secret authentication key that relates PAC to the secret user-supplied information used for authentication. The relationship may actually be one that relates PAC to AP, and hence relates PAC to secret user-supplied information indirectly since AP depends on secret user-supplied information. The key permits valid PACs to be created only by those so authorized.

If a verification table is used by the authenticator and the table is such that stored information cannot be changed, or cannot be changed without detection, then a copy of the user's secret user-supplied information, or AP value, can be stored directly in the table. In that case, a secret authentication key and PAC values are not needed.

Requirement 12. Information included in a transaction request message sent from the entry point to the issuer, which the issuer will use to validate the identity of the user initiating the transaction, must be formed from secret user-supplied information. It may or may not also be formed from nonsecret information, but it must not be formed from secret information known to anyone other than the user or the user and issuer.

This is the only way in which the personal verification process at one institution can be completely isolated from other institutions (see Requirement 5). It also implies that secret cipher keys used in the personal verification process by each institution are not shared with other institutions. Note that an EFT system in which PINs are protected via system keys would not meet this requirement.

Requirement 13. Knowledge of a transaction request message sent from an entry point to the issuer, which includes information that the issuer will use to validate the identity of the user initiating that transaction, must not allow an exposure of secret user-supplied information. And it must not permit equivalent user-supplied information to be derived that would allow an opponent masquerading as a customer of some financial institution to be verified by the system.[17]

Since users may initiate EFT transactions at any entry point in an interchange, information included in the transaction message which the issuer will use to

[17] Requirement 13, however, applies only if user-supplied verification information can be forged or duplicated, e.g., information the user knows (PIN) or has (personal key). On the other hand, if a user is identified by something he does (signature) or is (fingerprint), then duplication of that information may be difficult. Therefore, secrecy in this latter case is not critical. But a solution with PINs, or PINs and personal keys, is pursued here because such an approach is more apt to be used in the foreseeable future.

authenticate that user (e.g., an AP value or a MAC, whose computation involves secret user-supplied information) will often be transmitted through networks under the control of someone other than the issuer. A well-designed procedure for personal verification should not assume that the secrecy or integrity of transaction request messages can be maintained by others except the user and issuer. Therefore, knowledge of data contained in the transaction request message (including the generated MAC) should not expose secret user-supplied information nor should it permit equivalent user-supplied information to be derived that would permit an opponent to be verified by the system.

Requirement 14. Secret user-remembered information (PIN) must be supplemented by additional secret user-supplied information.[18] The information sent from an entry point to the issuer, which the issuer will use to validate the identity of a user, must be a one-way function of the PIN and additional secret user-supplied information.

With short PINs (e.g., 4 to 6 digits), there are insufficient independent secret bits available for computing an authentication parameter (AP) that will prevent recovery of the PIN via exhaustive methods. To overcome this, additional independent secret bits must be made available. However, because of requirement 12, the additional independent secret bits must be user-supplied. At the same time, because of the short PIN, the magnitude and random nature of the additional required secret bits precludes them from being committed to memory. Therefore, since each user has a bank card for initiating transactions at the entry point, it is assumed that the additional secret user-supplied bits (defined to be KP for personal key) are stored on the bank card. Thus, the card must provide storage for an additional quantity to serve as KP (a 56-bit key is assumed). The security issues involved with storing a KP on the bank card (e.g., whether a KP stored on a bank card can be maintained as a secret parameter) are taken up later (see the section entitled Threats to the Secrecy of a Key Stored on a Magnetic Stripe Card).

Since the information used by the issuer to validate the identity of a user must not depend on secret information other than that supplied by the user (requirement 12), and knowledge of that information must not expose secret user-supplied information or allow equivalent user-supplied information to be determined (requirement 13), it follows that this information must be a one-way function of user-supplied information.

A function f is a one-way function *if, for any argument x in the domain of f, it is easy to compute the corresponding value y = f(x); yet for almost all y in the range of f, it is computationally infeasible, given a value of y and knowledge of f, to calculate any x whatsoever with the property that f(x) = y. It is important to note that a function is defined which is not invertible from a computational point of view, but whose*

[18] It is assumed here that verification is based on something the user has (data stored on a magnetic stripe identification card) or knows (a password). Since these data can be forged, they must be kept secret. However, this would not apply if a nonforgeable input parameter were employed (e.g., a fingerprint or the dynamics of a handwritten signature).

noninvertibility is entirely different from that normally encountered in mathematics. A function f is normally called "noninvertible" when the inverse of a point y is not unique; i.e., there exist distinct points x1 and x2 such that f(x1) = y = f(x2). This is not the sort of inversion difficulty that is required here. Rather, it must be overwhelmingly difficult, given a value y and knowledge of f, to calculate any x whatsoever with the property that f(x) = y [2].

The complexity of DES makes it suitable for designing one-way functions. About 56 independent bits of secret input information are needed to obtain a one-way function with DES.[19] The nonsecret output of the one-way function must also be about 56 bits. In subsequent discussions, the one-way function approach will also be defined as the *noninvertible mode;* all other approaches will be referred to as *invertible modes.*[20]

An example of a one-way function of secret user-supplied information is $AP = E_{KP \oplus PIN}(ID)$. Advantage is taken here of the fact that, for a strong algorithm, knowledge of plaintext (ID) and corresponding ciphertext (AP) does not permit deduction of the key (KP ⊕ PIN). Thus KP and PIN cannot be obtained from AP and ID. Also a KP* and PIN* cannot be deduced such that $AP = E_{KP^* \oplus PIN^*}(ID)$. In that case, AP is a one-way function of KP and PIN.

A more detailed discussion of AP values and one-way functions is given in Appendix D. For the remainder of this chapter, the discussion will be concerned mainly with AP values that are one-way functions of the user's ID and secret user-supplied information, although the computation may also involve nonsecret system-supplied information.

Requirement 15. For message authentication, the authentication code must be a function of the message, a secret authentication key, and time-dependent information. The authentication key must always be known by or accessible to the sender.

Since messages are unpredictable, as far as information content is concerned, message authentication codes (MACs) cannot be precalculated; they must be calculated dynamically. Each MAC is transmitted to the authenticator together with the data that produce it. An opponent is prevented from generating a MAC, since a secret quantity (unknown to the opponent) is used in its generation. Since the procedure or cryptographic algorithm is public, as assumed here, that secret quantity must be a cryptographic key.

Furthermore, the MAC must be time-dependent to permit detection of

[19] As a rule, with approximately 32 independent key bits, the key can be recovered relatively easily using exhaustive methods. As the number of key bits increases, the key may or may not be recoverable depending upon the sophistication and resources of an opponent.

[20] Two other definitions that are closely related but should not be confused with the definitions of invertible and noninvertible functions are those of reversible and irreversible encryption [9]. *Reversible encryption* is defined as a cryptographic transformation of plaintext to ciphertext such that the ciphertext can be converted back to the original plaintext. *Irreversible encryption* is defined as a cryptographic transformation of plaintext to ciphertext such that the ciphertext cannot be converted back to the original plaintext by other than exhaustive methods.

previously transmitted data (stale messages). Requirement 15 differs slightly from requirement 11 because the PAC is time-invariant. Recall that personal verification information (e.g., PINs) remains constant for relatively long periods of time.

> *Requirement 16.* Time dependence in the message authentication code requires that a common time reference be established between the communicants. Furthermore, the receiver must be able to determine the reference's validity independently.

As previously discussed, time dependence allows the receiver to detect whether messages have been deleted or prevented from arriving, and whether stale messages (messages recorded on a prior occasion) have been injected back into the transmission path. But time-dependent quantities per se are not enough to ensure that the receiver rejects stale messages. In addition, the receiver must be able to establish independently the validity of the time reference. A weak procedure would result if the time reference were supplied to the receiver by the sender. In that case an opponent could also do the same and trick the receiver into accepting a previously sent message.

There are two ways in which a time reference could be established between the sender and the receiver: The sender could request a time reference (e.g., a randomly generated quantity) from the receiver, in which case the reference is under exclusive control of the receiver. Or the sender and receiver could maintain a common time reference (e.g., a sequential counter), in which case the reference is under control of both the sender and receiver. For example, a time reference stored on the bank card in a writeable storage element could be automatically updated with each use of the card. The issuer, on the other hand, could track the time reference by storing it in the verification table. The user and issuer could also establish a time reference via some means not under their direct control (e.g., by using a date and time-of-day).

> *Requirement 17.* The security of the message authentication process implemented at one institution must not depend on security measures implemented at other institutions.

The requirement is based on the principle that a well-designed procedure for message authentication should be such that the process of authenticating transaction requests sent from the user to the issuer, and the process of authenticating transaction responses sent from the issuer to the originating terminal, can be effected without exposing the secret information used in these processes and without depending on other institutions and network nodes to protect the secrecy of information used in these processes. This means that message authentication between the user and issuer and between the issuer and originating terminal must be unaffected by the security or lack thereof at any other EDP system or terminal in the interchange.

Comments on the EFT Security Requirements

The reader may note that the six security principles recommended by Kaufman and Auerbach [10] are a subset of the requirements developed here. The basic idea of employing one-way functions for personal verification was discussed in references 11 and 12. The reader may also note that some requirements are derived from preceding requirements. This allows the requirements to be developed in an orderly and progressive manner.

Although there is bound to be some disagreement over what constitutes a true security requirement, the intent here has been to develop a set of requirements that will tend toward maximizing security. In the end, financial institutions and designers and developers of cryptographic systems must weigh their own EFT security requirements against those developed here and decide which are mandatory, which are only desirable, and which are possibly unnecessary. One must always balance the probability and gravity of harm, should it occur, against the cost of implementing sufficient measures to prevent that harm.

PERSONAL VERIFICATION IN THE ON-LINE MODE

There are many ways of using PINs and personal keys to achieve personal verification. Several different designs and design tradeoffs are considered next.

In the design of a procedure for personal verification, the PINs and personal keys may be considered as parameters of the problem, where "secret" and "time-invariant" are attributes of these parameters. The significance of secret versus nonsecret and time-variant versus time-invariant parameters in the design of cryptographic systems has already been discussed. There is, however, another parameter attribute that is important to the design of personal verification procedures, i.e., whether the parameter is *independent* or *dependent*. An independent parameter is one whose value does not depend on any other parameter. It may be arbitrarily selected by the user, systems personnel, or the system. A dependent parameter is one whose value depends on one or more other parameters. Its value is derived from the parameter or parameters upon which it depends.

Independent PINs and personal keys can provide greater security than dependent PINs and personal keys. An independent parameter is not automatically compromised as the result of a compromise of other parameters in a cryptographic system, whereas a dependent parameter is always compromised whenever the secret parameter or parameters and algorithm (if secret) used to compute the parameter are compromised. With an independent parameter, every bit of the parameter must be stored. On the other hand, a dependent parameter can be computed dynamically as needed and thus storage requirements may be substantially reduced. For example, PINs could be derived from the corresponding IDs using a secret system key. This

permits the system to regenerate PINs rather than store them in a table. However, such a choice of dependent parameters might be unacceptable because users may want to select their own PINs, and institutions may want to provide this option to their customers.

Note that in the examples below, it is assumed (although not specifically shown) that authentication parameters are sent to the issuer in the transaction request messages and message authentication techniques are used to prevent replay of intercepted AP values.

Personal Verification with Dependent PINs and Dependent Personal Keys

PINs and personal keys can be made dependent variables, for example, by deriving them from users' IDs via a *PIN-generating key* (KPN) and a *personal-key-generating key* (KPG) as indicated by the following relationships:

$$PINi = E_{KPN}(IDi) \tag{11-1}$$

$$KPi = D_{KPG}(IDi) \tag{11-2}$$

Encipherment with KPN and decipherment with KPG permit the use of only a single secret key, KPN = KPG.

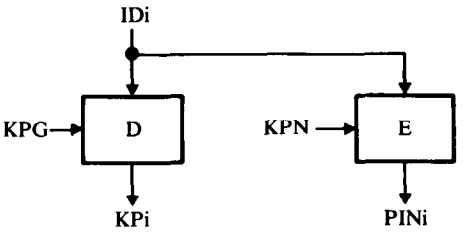
During the initialization process, the issuer selects KPN and KPG and then produces and issues personal identification numbers, personal keys, and bank cards to each user. ID and KP are stored on the bank card, whereas PIN must be remembered by the user. KPN and KPG are retained by the issuer so that each user's PIN and personal key can be regenerated during the process of personal verification.

A procedure that could be used for verification of a user is as follows (Figure 11-6). User i enters IDi, KPi, and PINi at an entry point (EFT terminal). The EFT terminal computes $APi = E_{KPi \oplus PINi}(IDi)$ and sends IDi and APi to the issuer (via an interchange if necessary). The issuer computes a PINi of reference and a KPi of reference (via Equations 11-1 and 11-2) from the received IDi and the stored values of KPN and KPG. IDi and the derived values of PINi and KPi are then used to compute an APi of reference, or reference Rfi for short. Rfi and the received APi are compared for equality. If Rfi = APi, then IDi is accepted (i.e., the identity of the user is accepted as IDi); otherwise, IDi is rejected.

Deriving PINs and personal keys from seed keys, KPN and KPG, has the disadvantage that it is awkward to reissue new PINs and personal keys.[21] Changing KPN or KPG causes every user's PIN or personal key to change, and thus requires the issuer to reissue a PIN and a bank card to each user and update the associated account file in the issuer's data base. Changing a user's identifier affects only that one user, but requires the issuer to close and open a new account. Compromise of both KPN and KPG allows a global attack against all users whose PINs and KPs were generated with these keys.

[21] To change a user's PIN, one could define the ith updated PIN as the ith encipherment of ID under KPN. However, in that case the system must be able to track the value of i (e.g., by storing it in a verification table).

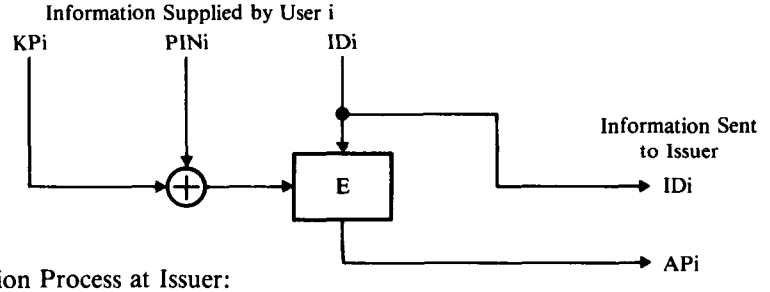
Initialization Process at Issuer:



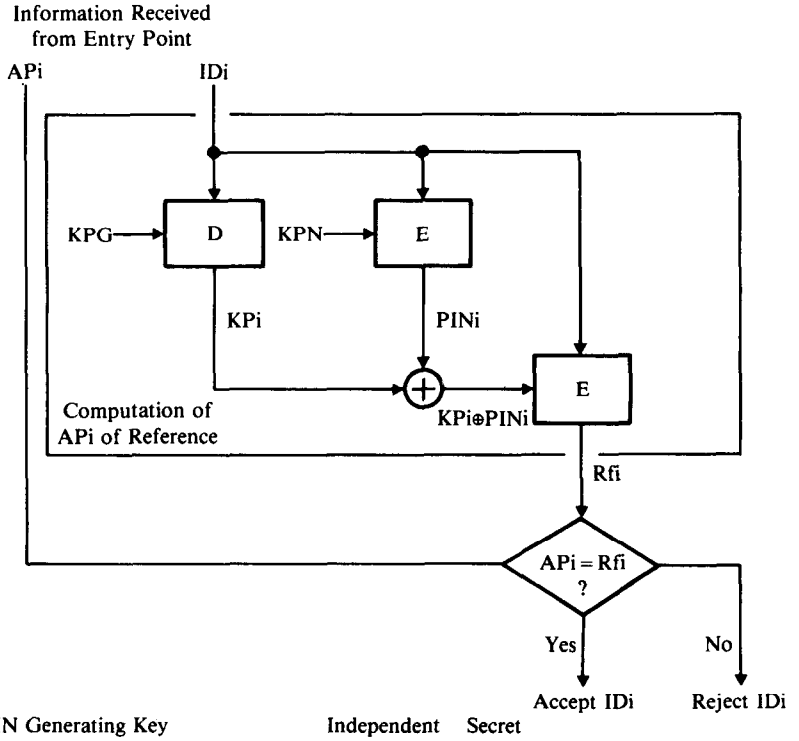
Stored on bank card

Note: Issuer selects KPG and KPN and computes PINi and KPi for each user.

Initiation of Personal Verification at Entry Point:



Verification Process at Issuer:



Legend:

KPN:	PIN Generating Key	Independent	Secret
KPG:	Personal-Key Generating Key	Independent	Secret
IDi:	User i's Identifier	Independent	Nonsecret
PINi:	User i's PIN	Dependent	Secret
KPi:	User i's Personal Key	Dependent	Secret
APi:	User i's Authentication Parameter	Dependent	Nonsecret

Figure 11-6. Personal Verification using a PIN Generating KEY (KPN) and a Personal-Key Generating Key (KPG)

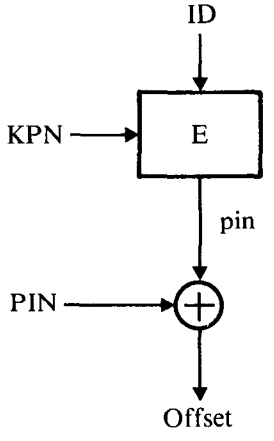
The verification procedure does have the advantage that there is no requirement for a personal authentication code (PAC) to be stored on the bank card or in a verification table, i.e., only ID, AP, and the secret system keys are needed.

If banks wish to allow users to choose their own PINs, the scheme could be redefined as follows. From the user's specified PIN, a quantity (defined as the offset) is generated such that the offset equals the modulo 2 addition of PIN and $E_{KPN}(ID)$, where $E_{KPN}(ID)$ is a system-generated personal identification number (defined as pin, Figure 11-7). The offset, which is equal to PIN $\oplus$ pin, is recorded on the user's bank card. When a user is identified to the system, the entered PIN is added modulo 2 to the offset to reproduce the correct dependent value of $E_{KPN}(ID)$ = pin.[22] Thereafter, pin and KP are used together as the basis for user verification.

A disadvantage of this method is that PIN $\oplus$ offset is a constant (C = pin) as long as ID and KPN are not changed. For example, if an opponent ever obtained the PIN and offset, he could calculate C. From then on, the particular ID is compromised even if the user changes his PIN. Note that changing PIN will only change the offset but will not affect pin. Furthermore, since the offset must be stored on the bank card, storage for an additional bit pattern is now required.

Personal Verification with Independent PINs and Independent Personal Keys

When PINs and personal keys are independent variables, they cannot be dynamically regenerated. Therefore, PIN and Personal key generating keys cannot generally be used securely. (The procedure described in Figure 11-7



KPN: pin Generating Key

Figure 11-7. Personal Verification Using a PIN Offset

[22] With current magnetic stripe cards, the modulo 2 addition is performed in the EFT terminal. With an intelligent secure card, the operation would be performed on the card.

employed a PIN generating key, and independent PINs were allowed by defining an offset. However, such a scheme has certain security weaknesses, as discussed above.) Hence, a different procedure for personal verification must be used.

One approach is for the issuer to store each user's independently calculated authentication parameter, $AP = E_{KP \oplus PIN}(ID)$, in a verification table. Selecting a particular entry in the table is accomplished by using ID as shown in Figure 11-8. Thus, the dynamic computation producing APi of reference, as shown above in Figure 11-6, is replaced by one in which IDi is used to look up the corresponding APi of reference in a table.

Storage of information in a verification table at the issuer can be avoided if an equivalent amount of information is distributed among the users and stored on the bank cards. This additional information, defined here as a personal authentication code (PAC), can be computed using the relationship

$$PACi = E_{KA}(APi) = E_{KA}(E_{KPi \oplus PINi}(IDi)) \qquad (11\text{-}3)$$

where KA is a secret *authentication key* known only to the issuer.

A user is now verified when the correct (AP, PAC) pair is supplied. However, the procedure relates PAC to ID only indirectly. Although AP is checked via the procedure, ID itself is not explicitly checked. Thus if the authenticator receives IDj, APi, and PACi instead of IDi, APi, and PACi, there is no way to determine that IDi was changed to IDj—the generated PACi of reference will check only that APi and PACi are a valid pair.

This can be remedied by coupling PAC to AP and ID be defining PAC as follows:

$$PACi = E_{KA}(E_{KA}(IDi) \oplus APi) \qquad (11\text{-}4)$$

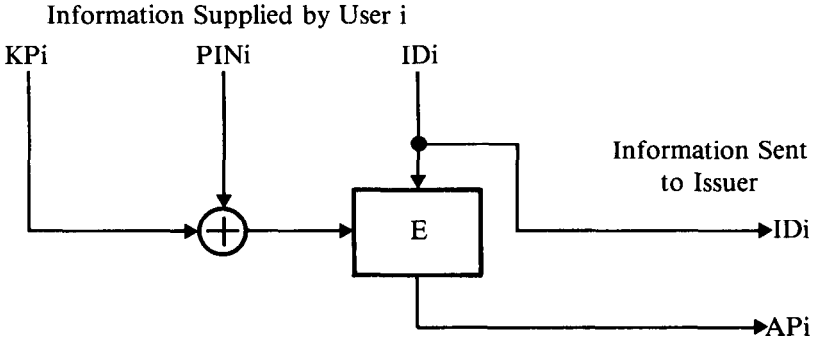
If IDi and APi are 8-byte blocks, PACi can be thought of as being produced by encrypting IDi and APi using a form of ciphertext feedback. Thus, PACi validates both APi and the correspondance of APi to IDi. IDi and APi are accepted as valid only if they generate, via KA, a PACi of reference equal to the received PACi.

Since PAC is precomputed, personal verification based solely on the triple (ID, AP, PAC) *without any additional message authentication* is by definition exposed to an active attack wherein previously recorded information is used to modify a transmitted message. For example, an intruder who has previously intercepted the triple (IDi, APi, PACi) can masquerade as user i by entering IDi and any bogus values for KPi, PINi, and PACi at the entry point and then replacing the bogus triple (IDi, AP*, PAC*) transmitted in the transaction request message with the previously recorded triple (IDi, APi, PACi) via an active attack.

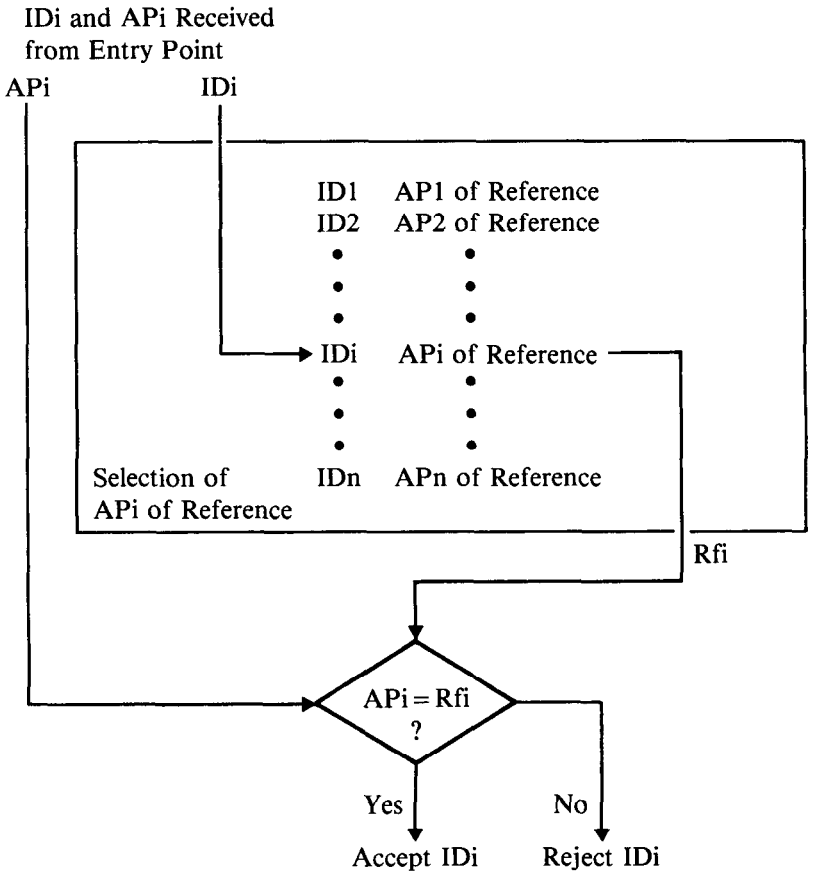
Message authentication, which provides a defense against message alteration and the introduction of stale messages, solves the problem. In that case, the relationship between ID and AP is now checked via the MAC and, therefore, personal verification can be based on either Equation 11-3 or 11-4.

During the initialization process (Figure 11-9), PINs are selected by either

Initiation of Personal Verification at Entry Point (Computation on the Intelligent Secure Card):



Verification Process at Issuer:

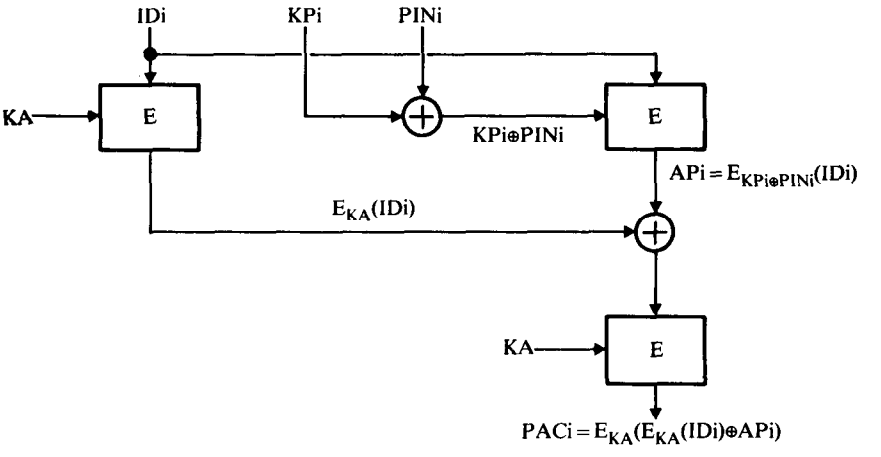


Legend:

IDi:	User i's Identifier	Independent	Nonsecret
PINi:	User i's PIN	Dependent	Secret
KPi:	User i's Personal Key	Dependent	Secret
APi:	User i's Authentication Parameter	Dependent	Nonsecret

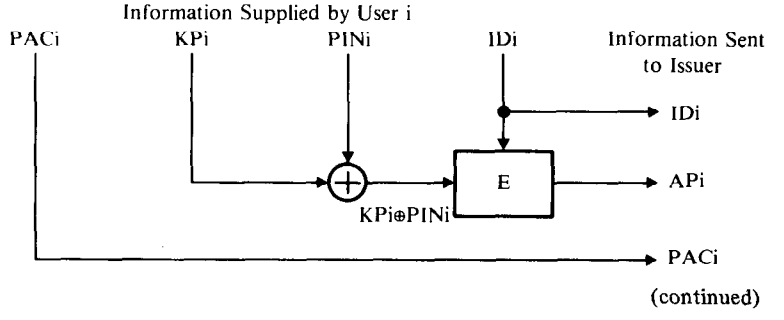
Figure 11-8. Personal Verification using Table Lookup

Initialization Process at Issuer:



Notes: IDi, KPi, and PACi are recorded on the bank card of the ith user. Issuer or user selects PINi. Issuer selects KPi, records it on the bank card, and computes APi. Issuer also selects KA and computes PACi.

Initiation of Personal Verification at the Entry Point:



(continued)

Legend:

KA:	Authentication Key	Independent	Secret
IDi:	User i's Identifier	Independent	Nonsecret
PINi:	User i's PIN	Dependent	Secret
KPi:	User i's Personal Key	Dependent	Secret
APi:	User i's Authentication Parameter	Dependent	Nonsecret
PACi:	User i's Personal Authentication Code	Dependent	Nonsecret

Figure 11-9. Personal Verification Using a Secret Authentication Key and Personal Authentication Codes (cont'd next page)

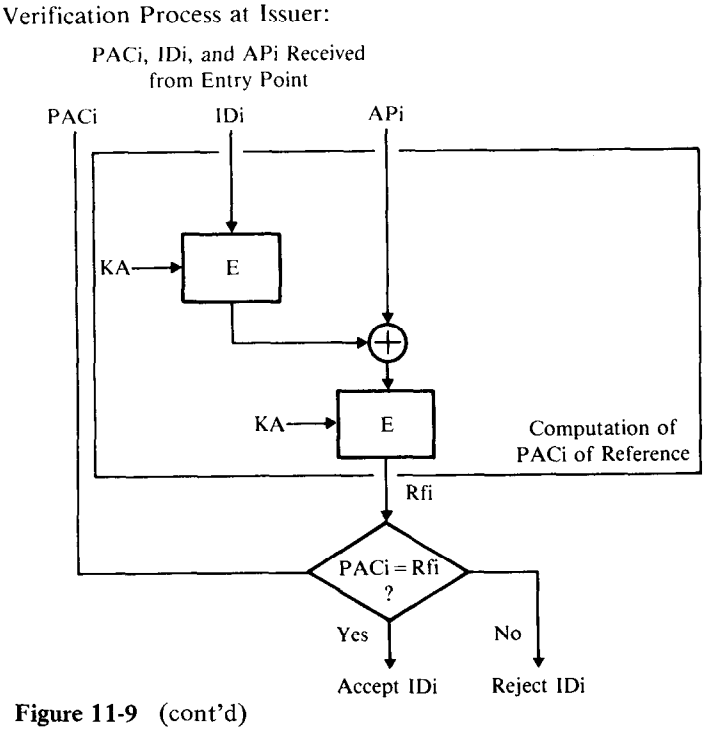


Figure 11-9 (cont'd)

the user or the issuer whereas personal keys are selected by the issuer (to prevent misuse of KPs, see the section entitled Objections to the PIN/Personal Key Approach). The issuer also selects KA and computes APi = $E_{KPi \oplus PINi}(IDi))$ and PACi = $E_{KA}(E_{KA}(IDi) \oplus APi$ for each user. KA is retained by the issuer so that each user's IDi and APi can be checked against PACi, which is stored on the bank card.

The process of user verification (Figure 11-9) requires user i to enter IDi, KPi, PINi, and PACi at an entry point (EFT terminal). The EFT terminal computes API = $E_{KPi \oplus PINi}(IDi))$ and sends IDi, APi, and PACi to the issuer—via an interchange if necessary. The issuer computes a PACi of reference, or Rfi for short (via Equation 11-3 or 11-4, as appropriate), from the received values of IDi and APi using the stored value of KA. Next, Rfi and the received PACi are compared for equality. If Rfi = PACi, then (IDi, APi) is accepted as valid and the identity of the user is accepted as IDi. Otherwise, (IDi, APi) is rejected and the user claiming IDi is denied system services.

The approaches in Figures 11-6 and 11-9 are similar in some respects and different in others. Neither requires a verification table, since their references are generated using one or more system keys. In the first case, KPG and KPN are used to generate personal keys and PINs, and in the second case, KA is used to compute PAC. Both approaches provide comparable security because they are equally vulnerable as far as compromise of system keys is concerned. In either implementation, compromise of the system keys KPG and KPN, or KA, will allow global attacks against any user of the system.

The major difference in the two approaches is that an additional quantity,

PAC, must be stored on the bank card when independent KPs and PINs are used. Independence is therefore achieved at the expense of extra storage on the bank card.

Although only three examples of on-line personal verification have been given, variations of these, as well as other designs, are possible. For example, a different KA could be defined for each user, in which case a verification table would be required to store the KAs. Personal authentication codes could also be used to provide cryptographic separation among institutions in an interchange supporting off-line verification. For example, each institution could define two authentication keys for off-line verification. A first key, KAlocal, could be used only to verify an institution's own users (i.e., it would not be shared with other institutions). A second key, KAremote, could be shared and used by all other institutions to verify that institution's users. Such a protocol would require two PACs to be stored on the bank card, one calculated using KAlocal and the other calculated using KAremote. Each entry point would be required to store its KAlocal key and the KAremote key for each other institution in the interchange.

Minimizing Card Storage Requirements

In any scheme not requiring a verification table, the specification of at least one authentication key is mandatory. If, in addition, the PINs are selected independently, additional storage must be provided on the card for a personal authentication code, as in the method of Figure 11-9. In that method, KP as well as PIN are independent variables and KP and PAC are stored on the bank card. Since security requires that KP be selected by the issuer to eliminate misuse of personal keys (see section Objections to the PIN/Personal Key Approach), it is not really necessary to make KP an independent variable. It is shown next that it is possible, by giving up the freedom to choose KP independently, to reduce storage requirements on the bank card by making KP dependent on PAC and accepting a shorter PAC length (on the order of 16 to 32 bits). Assume that

$$AP = E_{KP \oplus PIN}(ID) \tag{11-5}$$

$$PAC = \text{leftmost m bits of } E_{KA}(E_{KA}(ID) \oplus AP)) \tag{11-6}$$

Storage on the bank card can be reduced by generating PAC dynamically from KP.

$$PAC = \text{leftmost m bits of } E_{KP}(ID) \tag{11-7}$$

The price paid for this advantage is that only certain KPs will satisfy all three equations (11-5, 11-6, and 11-7). Furthermore, due to the complexity of the equations, only trial-and-error methods (i.e., key exhaustion) can be used to generate proper KPs. If the PAC length is m bits, the average number of iterations to find an acceptable KP is 2^{m-1}. To make exhaustion feasible, m should be chosen less than 32.

For analysis of the generation of KPs by the issuer, let

$$\text{APtrial} = E_{\text{KPtrial} \oplus \text{PIN}}(\text{ID}) \tag{11-8}$$

$$C1 = \text{the leftmost m bits of } E_{KA}(E_{KA}(\text{ID} \oplus \text{APtrial})) \quad (= \text{PACtrial}) \tag{11-9}$$

$$C2 = \text{the leftmost m bits of } E_{\text{KPtrial}}(\text{ID}) \tag{11-10}$$

Due to Equation 11-7, the constraint C1 = C2 is now introduced.

For the generation of KPs that satisfy the condition C1 = C2, trial values of KP (KPtrial) are used for a selected PIN with the corresponding ID to generate a trial AP according to Equation 11-8. A trial PAC (C1 = PACtrial) is then computed from the trial AP using Equation 11-9. A trial value of C2 is also calculated using Equation 11-10. The KPtrial is accepted only if C1 = C2; otherwise, the KPtrial is rejected and the process is repeated (Figure 11-10). Assuming that the probability of a match (C1 = C2) at each trial is 2^{-m}, it takes an average of $2^m/2$ trials to find an acceptable KP.

Once an acceptable KP is found, the initialization process is complete. To initiate the verification process at the entry point requires that AP = $E_{KP \oplus PIN}(ID)$ and PAC = the leftmost m bits of $E_{KP}(ID)$ be generated and transmitted together with the ID to the issuer (Figure 11-11). At the authenticating node (the issuer), the authentication key, KA, is used to generate a PAC of reference, i.e.,

$$\text{PAC of Reference} = \text{leftmost m bits of } E_{KA}(E_{KA}(\text{ID}) \oplus \text{AP})$$

Only if this quantity is equal to the received PACi will the user (IDi) be accepted. Otherwise, the user will be rejected (Figure 11-11).

This approach requires, again, proper message authentication. Otherwise, any intercepted triple (ID, AP, PAC) sent to the authenticating node would authenticate the user (Figure 11-11). An opponent could thus masquerade as the user associated with the intercepted triple.

But even if proper message authentication is in place, an opponent could attempt to impersonate another user (say user i) by entering IDi (assumed known) and using a method of trial and error, as follows: an arbitrary personal key and PIN (i.e., KP* and PIN*) are entered so that the entry point calculates AP* = $E_{KP^* \oplus PIN^*}(IDi)$ and PAC* = $E_{KP^*}(IDi)$ (Figure 11-11). At the authenticating node, Rf* = $E_{KA}(IDi) \oplus AP^*$) is calculated and checked for equality with the leftmost m bits of PAC*. Since the probability is 2^{-m} that these (fake) quantities are indeed identical, about 2^{m-1} trials are required for the attack to succeed. In the described attack the opponent must manually enter the trial values of KP* and PIN*, which means that the time per trial is measured in seconds. The value of m is selected so that $2^{m-1} \times$ (time per trial) is sufficiently large.

A more serious threat arises if insiders are able to repeatedly exercise the operation that calculates the PAC of reference (Rf in Figure 11-11). In that case, the opponent selects a trial key, KPtrial, and calculates a set of APs by

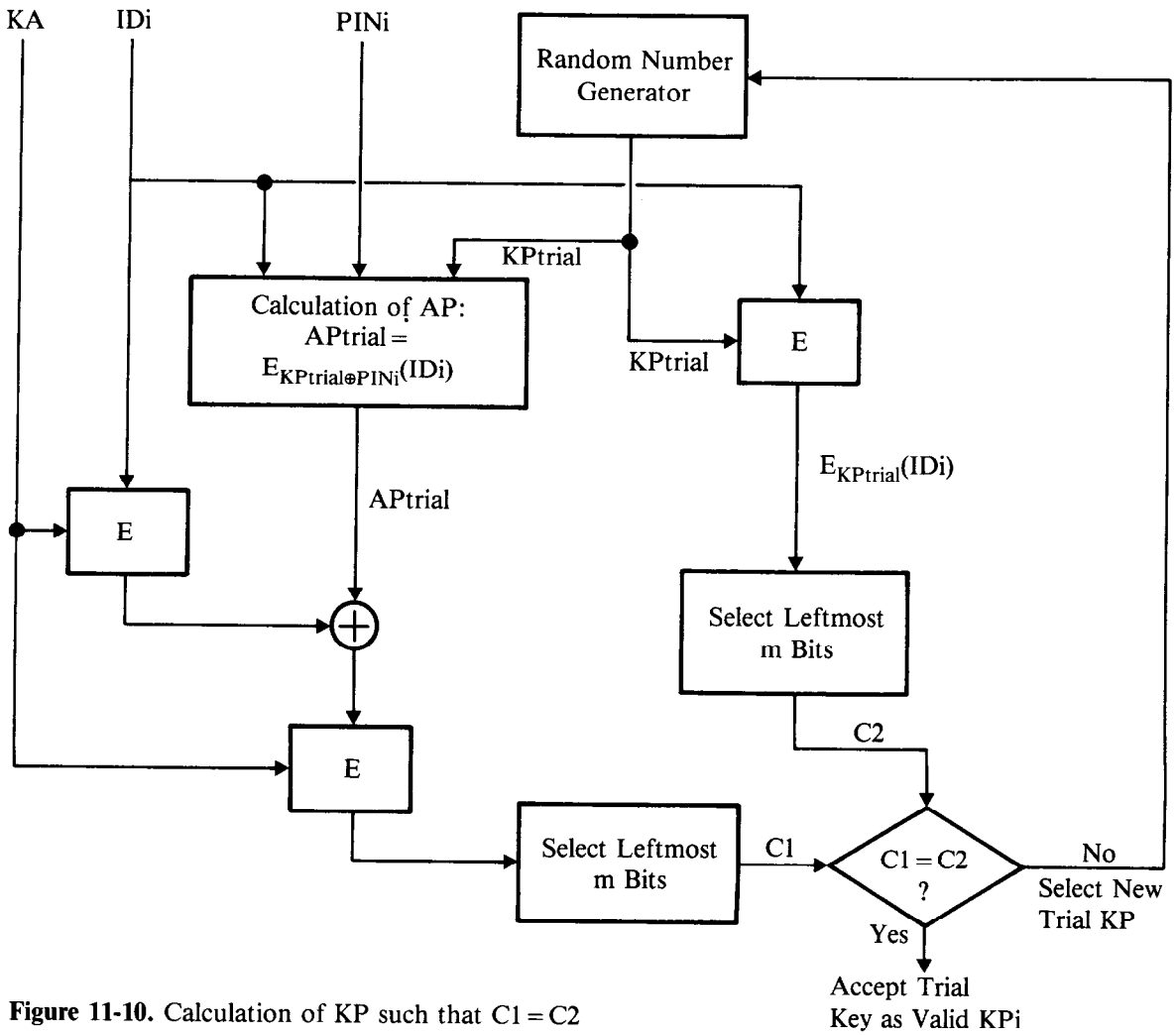


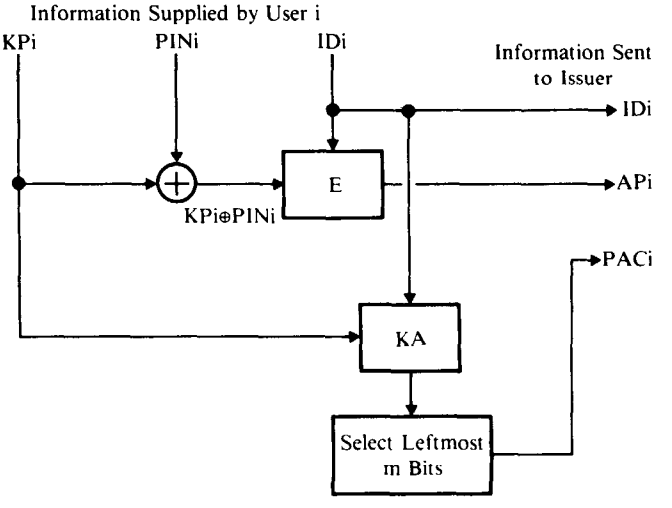
Figure 11-10. Calculation of KP such that C1 = C2

exhausting PIN space (i.e., $\{APj = E_{KPtrial \oplus PINj}(IDi); j = 1,2, \ldots, n\}$ where n is the number of possible PIN combinations). In addition, PACtrial = $E_{KPtrial}(IDi)$ is calculated. The set of APs is then used together with IDi and PACtrial to calculate the set $Q = \{Rfj; j = 1, 2, \ldots, n\}$, where $Rfj = E_{KA}(IDi) \oplus APj$. If Q contains an element Rfj such that Rfj = PACtrial, then the attack succeeds, because in that case an equivalent PIN and KP are found for the IDi. If no match is found, the procedure is repeated using a different KPtrial. This insider attack will succeed only if references can be calculated at will. One way to detect and thwart such an attack is to log all unsuccessful validations. Since most of the time (in an honest environment) the validations will succeed, the statistic of unsuccessful validations is a useful audit tool. The lesson to be learned from this threat analysis is the following: whenever the number of trials required by the issuer to generate certain parameters is small, extreme care must be taken to ensure that an opponent cannot benefit from this efficient computational procedure in the same way. By restricting the opponent to attack the entry point, the described method for reducing card storage can be made sufficiently strong. The value of m is thus chosen so that the issuer can carry out the required

Initialization Process at Issuer:

Notes: Issuer or user selects PINi. Issuer selects KA, generates KPi (as in Figure 11-10), and records KPi on the bank card.

Initiation of Personal Verification at Entry Point:



Legend:

KA:	Authentication Key	Independent	Secret
IDi	User i's Identifier	Independent	Nonsecret
PINi:	User i's PIN	Independent	Secret
KPi:	User i's Personal Key	Dependent	Secret
APi:	User i's Authentication Parameter	Dependent	Nonsecret
PACi:	User i's Personal Authentication Code	Dependent	Nonsecret

Figure 11-11. Personal Verification Using a Secret Authentication Key and Personal Authentication Codes

Verification Process at Issuer:

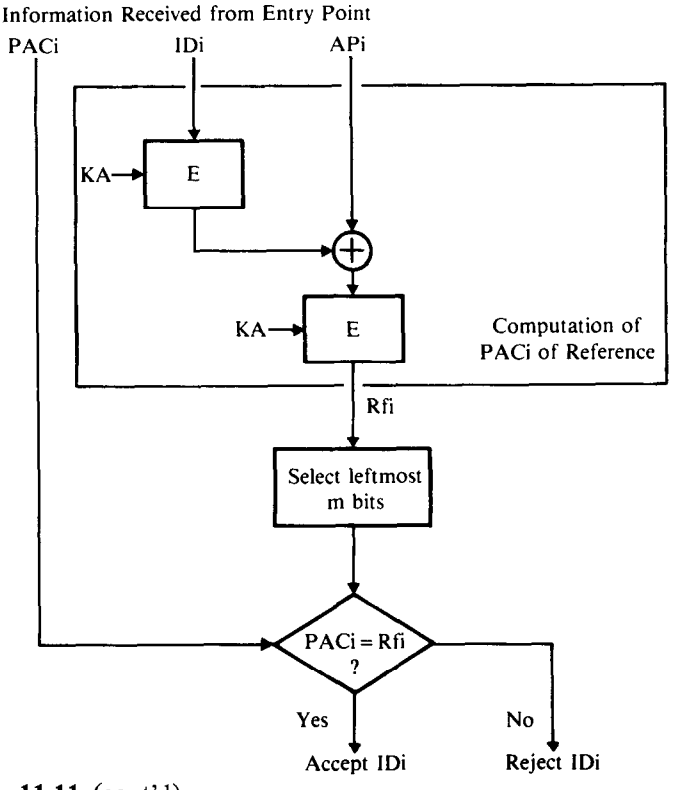


Figure 11-11 (cont'd)

computations efficiently using a high-speed computer whereas an opponent is forced to use a much slower procedure of trial and error involving a manual entry of trial parameters at the entry point.

PERSONAL VERIFICATION IN THE OFF-LINE AND OFF-HOST MODES

When personal verification is performed by a terminal, the process is said to operate in the off-line mode. When a terminal and a communications control unit (Figure 11-1) cooperate to perform personal verification, the process is said to operate in the off-host mode. In the off-line mode, the entry point must additionally assume the role of issuer. It must perform personal verification and manage transaction requests, although it is relieved of other tasks such as opening new accounts and assigning and selecting PINs.

The discussion in this section demonstrates that personal verification using the on-line mode is more secure than either the off-line or off-host modes of operation. The reason for this is that the on-line mode does not involve widely distributed keys as do the other modes. With the off-line and off-host modes, it is not possible to isolate the personal verification process to a single institution. To show why security is reduced in this case, methods

are discussed wherein personal verification is performed by a node different from the issuer's node.

Cryptographic system designs for off-line and off-host banking transactions are more straightforward than those for comparable on-line banking transactions. For instance, in an off-line mode, there is no requirement for message authentication, since the entire procedure takes place at the entry point. However, because a terminal and a communications control unit do not, in general, have storage for a verification table, the possible designs are more limited. The following discussion assumes, therefore, that insufficient storage is available for a verification table. Hence, personal verification must be based on testing a preestablished relationship between the ID and secret information supplied by the user.

By definition, when personal verification is performed in the off-line or off-host mode, a widely distributed authentication key must be used to test the relationships among ID, secret user-supplied information, and PAC. If that key is a secret key (using the DES), its compromise globally affects the entire verification procedure. If that key is a public key (using a PKC), compromising its integrity compromises only the verification procedure at that one off-line or off-host location:

> *An attack against a PKC succeeds if the public key at the entry point is changed by an opponent who then enters a PAC calculated with the corresponding secret key. If the secret key used to compute user PACs is compromised, a global attack against all users is possible (as would be the case with a conventional algorithm). The advantage of the PKC, however, is that the secret key is not stored at the entry point.*

Because the authentication key (whether it is a public or a private key) is widely distributed, the soundness of the off-line and off-host modes is questionable. In an interchange environment, for example, the authentication keys may be stored in thousands of terminals. However, the approach may be used safely in networks where the number of institutions and EFT terminals is small and where the EFT terminals are vault-like units. With such an implementation, protecting the secrecy or integrity of a key in the EFT terminal may be less of a problem than providing similar protection to a verification table or authentication key located at a host.

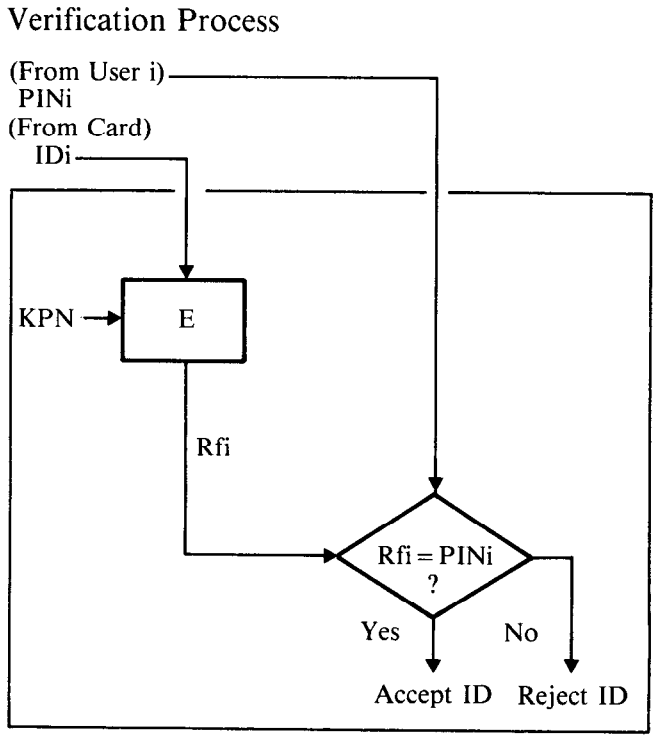
An important requirement of any off-line or off-host personal verification procedure is that a compromise of either of these modes should not compromise or severely weaken security in the on-line mode. (In the discussion it is assumed that both modes are implemented in the network.) In other words, compromising a secret key used in the off-line or off-host mode should not jeopardize secret user-supplied information essential to on-line personal verification.

Personal Verification with System-Selected PINs Employing a PIN Generating Key

The on-line mode can be separated cryptographically from either the off-line or off-host mode if dependent PINs and dependent personal keys are used. For example, assume the on-line scheme illustrated in Figure 11-6,

where on-line personal verification is based on PINs and personal keys derived from other keys (KPN and KPG). One way to achieve separation between on-line and off-line personal verification, in this case, would be to base off-line personal verification on PIN only and to base on-line personal verification on PIN as well as KP. This could be achieved by storing only KPN in the terminal (Figure 11-12) and by storing KPN as well as KPG at the issuer's HPC (Figure 11-6). In that case, a compromise of KPN in the off-line mode (resulting in the exposure of PINs) would not compromise the on-line mode since personal keys are still secure.

During the verification process (Figure 11-12), user i enters IDi and PINi at an EFT terminal in which KPN has been installed. The EFT terminal computes $E_{KPN}(IDi) = Rfi$ (i.e., the PIN of reference), and compares Rfi and PINi for equality. If Rfi = PINi, then IDi is accepted; otherwise, if Rfi ≠ PINi, IDi is rejected.



KPN = PIN Generating Key
Rfi = PIN of Reference

Notes: To initialize the system, the issuer selects KPN and computes $PINi = E_{KPN}(IDi)$ for each user (ID1, ID2, ..., IDn). If KPN is compromised a valid PIN can be generated for any given ID.

Figure 11-12. An Example of Off-Line Personal Verification with System Generated PINs

Personal Verification with User-Selected PINs Employing Offsets

Consider a design based on independent PINs which uses an offset field on the card and satisfies the relation

$$E_{KPN}(IDi) = PINi \oplus offseti$$

where KPN is a secret system-managed PIN generating key (see also Figure 11-7).

In this case, KPN, IDi, and PINi are independent variables, whereas E_{KPN} (IDi) and offseti are dependent variables. Verification takes place as illustrated in Figure 11-13. The security weakness here (as discussed before) is that the user-related PIN added to the offset results in a constant, C, as long as ID and KPN remain fixed.

Personal Verification with User-Selected PINs Employing PACs

In a more secure approach, different personal authentication codes (PAC and PACoff) can be used for on-line and off-line (or off-host) transactions, respectively. This is accomplished by using different secret authentication keys (KA and KAoff). For example, one could define

$$PACi = E_{KA}(E_{KA}(IDi) \oplus APi)$$
$$PACoffi = E_{KAoff}(E_{KAoff}(IDi) \oplus APi)$$

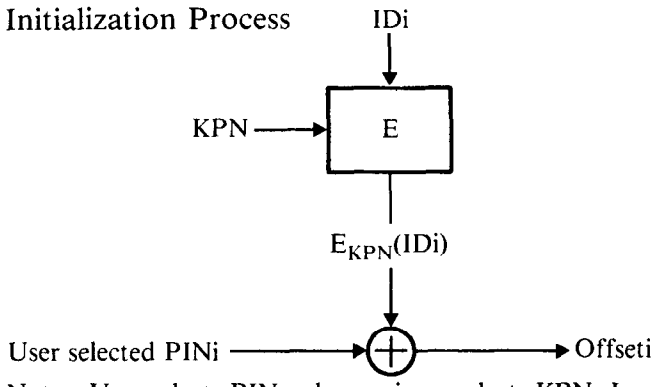
where

$$APi = E_{KPi \oplus PINi}(IDi)$$

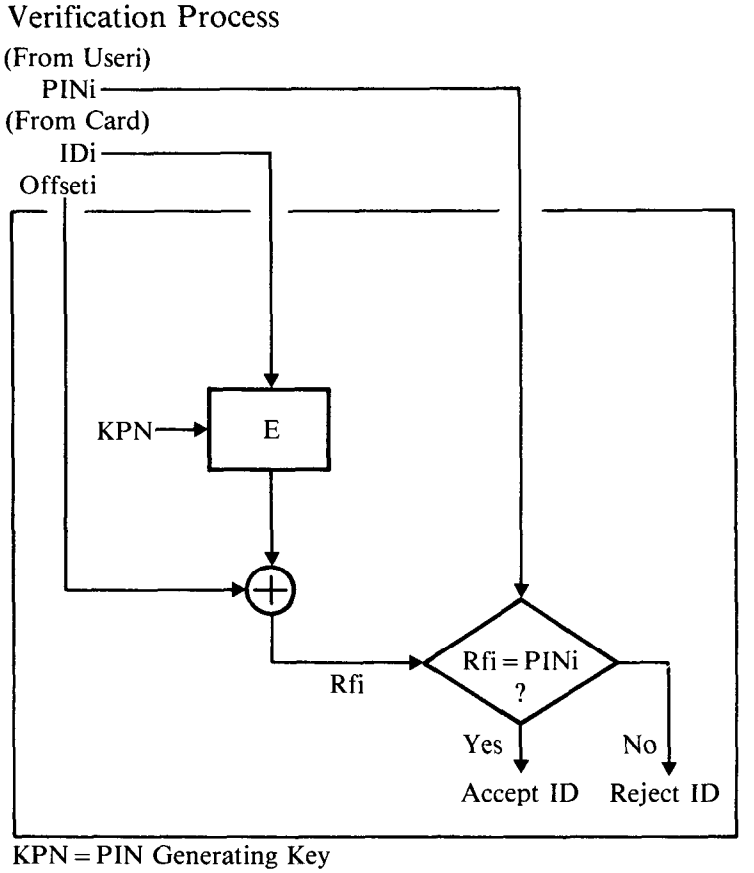
The on-line verification mode using KA and PACi is shown in Figure 11-9. The off-line and off-host modes use the same procedure as the on-line mode except that KA is replaced by KAoff and PAC is replaced with PACoff (Figure 11-14). KA is stored at the issuer's HPC for on-line verification whereas KAoff is stored in the appropriate terminals and communication controllers to allow off-line and off-host verification.

If KAoff is compromised, a valid PACoff can be generated for any given set of values (ID, KP, PIN) thus compromising the off-line mode. However, the actual KP and PIN cannot be determined, and thus the on-line mode is still secure. Furthermore, since KA is unavailable, PAC cannot be evaluated.

A saving in card space could be achieved by basing on-line personal verification on an authentication parameter stored in a verification table (Figure 11-8). This would eliminate the need to store a personal authentication code associated with on-line verification on the bank card. Storage requirements on the bank card could be reduced further by basing off-line personal verification on a personal authentication code (PACoff) which is related to KP. This idea is discussed above in the section Minimizing Card Storage Requirements.



Notes: User selects PIN, whereas issue selects KPN. Issuer computes offset for each user by adding, module 2, the user-selected PIN and the value E_{KPN}(ID).



KPN = PIN Generating Key
Rfi = PIN of Reference
Note: If KPN is compromised, actual PINs can be generated.

Figure 11-13. An Example of Off-Line Personal Verification with User-Generated PINs Employing an Offset

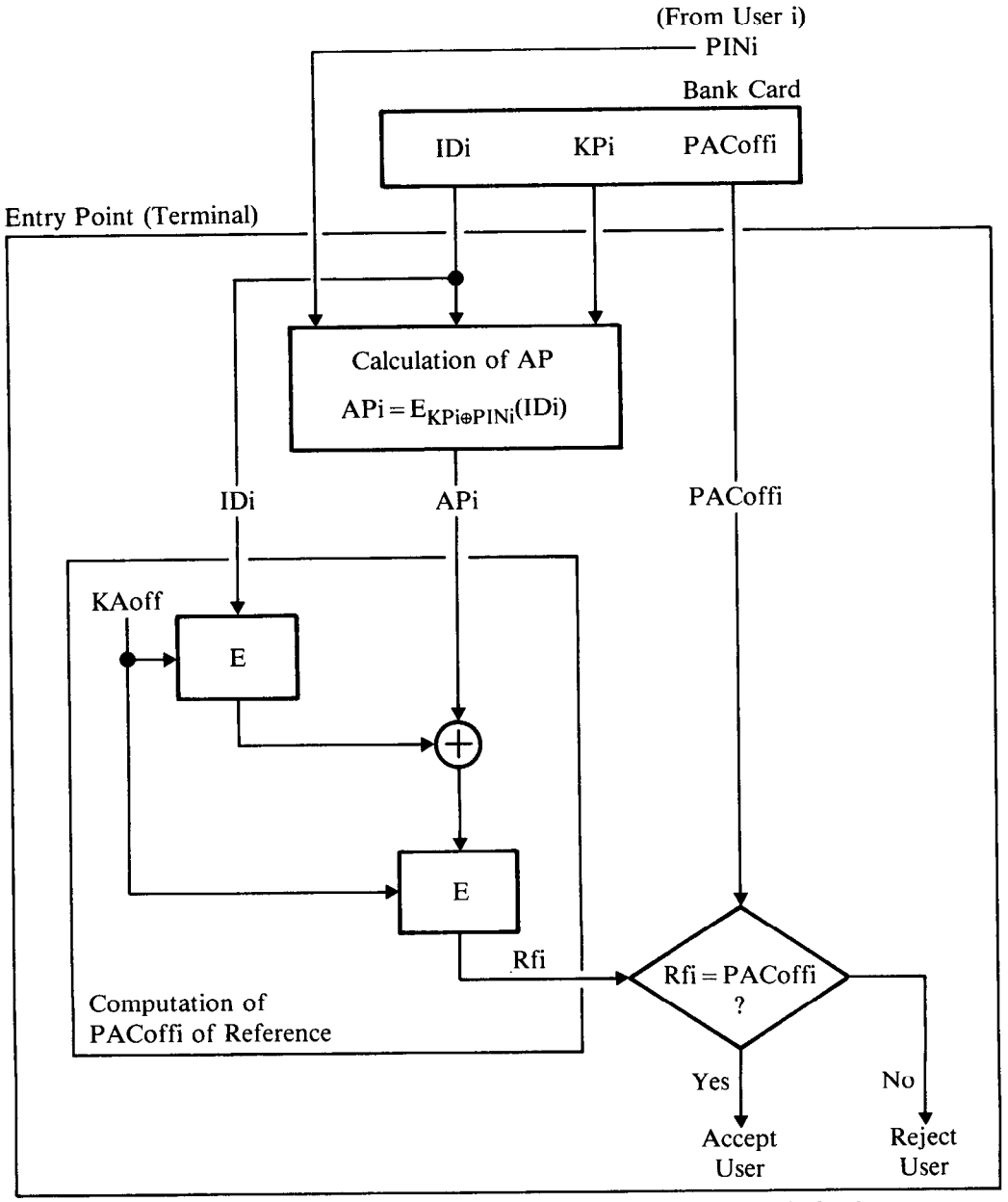


Figure 11-14. Off-Line Personal Verification - PACoffi stored on Bank Card

GUIDELINES FOR CRYPTOGRAPHIC DESIGNS

The EFT security requirements developed above are used in the remainder of the discussion to develop good cryptographic designs. These EFT security requirements may be used effectively to compare and establish trade-offs in different designs (e.g., key management approaches based on system keys only, or personal keys only, or a hybrid approach involving a combination of both).

In the following discussion of methods used to satisfy the various requirements, the focus is on requirements 1, 2, 5, 12, 14, and 17—which are concerned with the integrity of secret information entered into a system, integrity and secrecy of terminal stored information, isolation of the personal verification process, generation of user verification data, one-way functions of secret user-supplied information, and isolation of the message authentication process, respectively. Since requirements 12 and 14 follow from requirement 5, it suffices to compare the different methods with respect to requirements 1, 2, 5, and 17 which are repeated below for the reader's convenience.

> *Requirement 1.* The process of entering information at an EFT terminal must be protected, i.e., the integrity and secrecy of information entered into the EFT terminal must not be compromised as a consequence of the entry process. Furthermore the information flow within a terminal must be protected.
>
> *Requirement 2.* Information stored permanently or temporarily within an EFT terminal must be protected, i.e., the integrity of public information and the secrecy as well as integrity of private information must be assured.
>
> *Requirement 5.* The security of the personal verification process implemented at one institution must not depend on security measures implemented at other institutions.
>
> *Requirement 17.* The security of the message authentication process implemented at one institution must not depend on security measures implemented at other institutions.

To satisfy requirement 1, the following must be true:

1. The entry paths for the PIN and user-supplied key into and within the terminal are not exposed during normal system operation.
2. The entry paths for terminal resident keys into and within the terminal are not exposed during the key loading process.
3. During the entry process, secret card information is not exposed even if bank cards are temporarily in the possession of others.

Requirement 1 is independent of cryptography. It requires that security measures be employed at the entry point to reduce the probability that the PIN and secret card information (e.g., a personnel key, KP) can be ascertained during the entry process. To prevent, or at least make it difficult for card

information to be ascertained while the card is in the possession of someone other than the card owner (a bank teller or store clerk), requires a particular approach in the design of the bank card, i.e., the introduction of an intelligent secure card (discussed below) capable of performing computations that involve secret information stored on the card.

To satisfy requirement 2, one must assure that

1. Permanently stored terminal keys are not exposed.
2. Temporarily stored PINs and terminal keys (active for the duration of a transaction) are not exposed.

Keys stored in a terminal can be protected only by physical security measures. Hence, requirement 2 is also independent of cryptography. To avoid the problem of key security at the terminal altogether, one might search for a method which does not require a cryptographic key resident in the terminal. This leads to the idea of using a key supplied by the user, i.e., a personal key. It happens, however, that such an approach is not secure unless a bank card with the following properties could be introduced: counterfeit cards could not be manufactured and information stored on genuine cards could not be read or altered. Such a bank card is defined below as an ideal intelligent secure card. It is concluded, however, that the requirements of the ideal bank card are unattainable at a reasonable cost with current technology.

An intelligent secure card capable of achieving realistic security objectives is defined a practical intelligent secure card (or intelligent secure card for short). It has the property that card information is secure while the card is temporarily in the possession of another person or device for the purpose of transacting business. However, use of an intelligent secure card does not obviate the need to protect the integrity and sometimes the secrecy of information transferred between the card and terminal.

Requirements 1 and 2 can therefore be satisfied partially through the introduction of an intelligent secure card incorporating a personal key. Card information, in this case, is not entered into the EFT terminal. The PIN is not assumed to be entered directly on the card, although, if it were, requirements 1 and 2 could be fully satisfied. Therefore, except for PIN entry, the intelligent secure card is equivalent to a miniature EFT terminal carried by its owner.

Computationally the card would contain the following elements: a one-chip microprocessor, ROS (read only store) containing executable programs, a small RAM (random access memory) for storage of intermediate results (which could be on the microprocessor chip), and a nonvolatile memory for the storage of customer unique information (e.g., account number, secret card key or parameter, and PIN-related information) and dynamic information (e.g., current balance, number of transactions, and a one-up counter or message sequence number). The nonvolatile memory is partitioned and protected so that

1. A portion of the memory can be written to only by the issuing institution during card personalization. Thereafter, secret information stored

in this memory can be read only by the card, whereas nonsecret information stored in this memory can be read by both the card and EFT terminal.

2. A part of the memory can be written to and read only by the card.

Power and timing information would be provided externally by the EFT terminal, although fully self-contained devices are not precluded. Electrical connections for communication between the card and terminal are also provided.

One important design goal is to achieve isolation among institutions with regard to personal verification (requirement 5) and message authentication (requirement 17). A design based solely on PINs and system keys requires a high degree of trust among institutions, since cryptographic transformations are required to translate data from encryption under one key to encryption under another key. This means that PINs or PIN-related information occurs in the clear at nodes not under the control of the issuer. The potential for exposure is minimized if this information occurs in the clear only in secure hardware. But the fact remains that the issuer must trust other institutions to implement secure hardware and maintain its integrity. Hence requirements 5 and 17 cannot be satisfied with a design based solely on PINs and system keys. Details are discussed in the section The PIN/System Key Approach.

Requirement 5 can be satisfied with a design that increases the number of combinations of secret user-supplied information by introducing (in addition to PIN) a personal key (KP) written on an intelligent secure bank card. With PIN and KP, the number of independent bits of secret user-supplied information is sufficient to defend against direct search and dictionary attacks. (Designs based only on PIN are vulnerable to these attacks.) A design based only on KPs and PINs (no system keys) is described in the section The PIN/Personal Key Approach. Specifically, it is shown that KP enhances personal verification and authentication of transaction request messages by satisfying requirement 5. However, the EFT terminal cannot use personal keys to authenticate transaction response messages unless it can be assured that these keys are known only to the issuer (i.e., not even known to the legitimate user). An opponent who enters a fake personal key at an EFT terminal could, for example, inject bogus transaction response messages into the communication line that the EFT terminal would accept as valid. Elimination of these exposures would be possible, and thus requirements 5 and 17 could be satisfied, if a pure personal key approach were used together with an ideal intelligent secure card. But, as a practical matter, the properties of an ideal intelligent secure card are unattainable, and therefore requirement 17 cannot be satisfied with only a personal key approach. *As a consequence, use of a personal key does not eliminate the requirement for a terminal resident key.*

In the section The PIN/Personal Key/System Key (Hybrid Key Management) Approach Using an Intelligent Secure Card, an implementation is discussed that uses a combination of personal and system keys (defined as *hybrid key management*) together with an intelligent secure card capable of

achieving realistic security objectives (defined as a practical intelligent secure card in contrast to the ideal intelligent secure card described above). In particular, requirement 5 can be satisfied completely and requirement 17 can be satisfied to a high degree. To satisfy requirement 17 completely requires that electronic digital signatures be implemented (see the section entitled Security Enhancements With Digital Signatures).

The different methods can now be classified as follows:

Method 1. PIN/System Key Approach
Method 2. PIN/Personal Key Approach
Method 3. PIN/Personal Key/System Key Approach

In addition, the following categories can be defined:

Category a. Using a magnetic stripe bank card
Category b. Using a practical intelligent secure card
Category c. Using an ideal intelligent secure card

Hence, altogether there are nine different approaches defined, i.e., 1a, 1b, 1c, 2a, . . . , 3b, 3c. Tables 11-1 and 11-2 indicate the effectiveness of each method with respect to the requirement of separating personal verification and message authentication among institutions (requirements 5 and 17), respectively. Since requirements 1 and 2 require physical security in all approaches, a separate table is not provided for these.

Details of the different designs are discussed in the sections The PIN/System Key Approach, The PIN/Personal Key Approach, and The PIN/Personal Key/System Key (Hybrid Key Management) Approach Using an Intelligent Secure Card. From these discussions it can be concluded that the combination 3b presents a realistic security solution.

Before presenting the details of different designs, some of the fundamental threats to PIN secrecy are highlighted. Furthermore, key management requirements common to all designs are investigated.

Threats to PIN Secrecy

Observation of the PIN

In any nationwide EFT environment, there will be a very large number of PIN-using terminals located in nonsecure locations. The line from the PIN-using terminal could be tapped and one or more cameras or video recorders could be positioned to observe and record customers entering their PINs into the EFT terminal or PIN pad device. This observation is synchronized with reading the corresponding transaction sent over the communication line to establish a relationship between the observed PIN and intercepted account number. After a period of time, a significant number of PINs could be obtained from unsuspecting customers who were not particularly careful during the PIN entry process. A computer could then correlate the recovered PINs with the intercepted transactions to determine the customer's account number and

Method 1			Method 2			Method 3		
a	b	c	a	b	c	a	b	c
Requirement 5: Comparison of Methods Excluding the Entry Point								
not satisfied	not satisfied	not satisfied	satisfied	satisfied	satisfied	satisfied	satisfied	satisfied
Common keys between nodes are necessary.			Secret verification information is processed only at the end points.					
Requirement 5: Comparison of Methods at the Entry Point								
not satisfied	not satisfied	not satisfied	not satisfied	satisfied	satisfied	not satisfied	satisfied	satisfied
Key at entry point is not controlled by the issuer.			Secrecy of input information (PIN and KP) not controlled by the issuer.	Card information (KP) not exposed to others; fake KP and misuse of KP not possible.	Secrecy of card information (KP) guaranteed.	Secrecy of input information (PIN and KP) not controlled by the issuer.	Card information (KP) not exposed to others; fake KP and misuse of KP not possible.	Secrecy of card information (KP) guaranteed.

Table 11-1. Security Properties of Different Cryptographic Methods—Separation of Personal Verification Process.

Method 1			Method 2			Method 3		
a	b	c	a	b	c	a	b	c
Requirement 17: Comparison of Methods Excluding the Entry Point								
not satisfied	not satisfied	not satisfied	satisfied	satisfied	satisfied	satisfied	satisfied	satisfied
Common keys between nodes are necessary.			Secret verification information is processed only at the end points.					
Requirement 17: Comparison of Methods at the Entry Point								
not satisfied	not satisfied	not satisfied	not satisfied	not satisfied	satisfied*	not satisfied	partly satisfied	satisfied*
Key at entry point is not controlled by issuer.			Secrecy of input information not controlled by the issuer. Threat of KP misuse; threat of fake KP.	Threat of KP misuse since user could determine his KP.	Secrecy of card information guaranteed.	Secrecy of input information not controlled by the issuer.**	Threat of KP misuse since user could determine his KP. But attack requires subversion of system keys.	Secrecy of card information guaranteed.

Note: Although the PIN/personal key approach (method 2) provides good separation between institutions, it is not an acceptable solution unless an ideal intelligent secure card is used (method 2c). The security exposure is due to the threat of KP misuse and the use of fake KPs.

*It is assumed that the signal path between the intelligent secure bank card and the terminal is not subverted.

**There is also the threat of KP misuse and the use of fake KPs. But a successful attack is dependent upon subversion of system keys.

Table 11-2. Security Properties of Different Cryptographic Methods—Separation of Message Authentication Process

other information necessary to produce a counterfeit card. This would allow fraud to be perpetrated against each corresponding account.

A suggested defense against this threat is to encrypt the account number and all other information that links the PIN to the cardholder. This prevents the production of a counterfeit card from intercepted information. However, financial institutions may find that encryption of card information conflicts with some other system requirement. For example, it may be necessary for certain intermediate nodes, which do not have an encryption/decryption capability, to read the information contained in the transaction messages.

Bugging of Input Information at EFT Terminals

A bug placed in an EFT terminal allows all PINs entered into that terminal to be intercepted. This, in turn, allows an opponent to successfully masquerade as any one of the users whose PINs are intercepted, thus allowing fraud to be perpetrated against each of the corresponding accounts.

Insertion of Fake Equipment

The most insidious fraud threat is one that uses fake equipment. A dishonest merchant may induce unsuspecting cardholders to use EFT terminals with fake PIN pads. Although there are many variations on this attack (see Countering the Fake Equipment Threat, Chapter 10), some of which can be defended against, there is one that offers no apparent practical defense (no matter what method is employed, personal keys or system keys) if card information and PIN are entered into the retailer's equipment.

In this attack, the merchant replaces the EFT terminal and PIN pad with devices that will display or print the entered PIN at a work station. Every PIN so obtained is automatically recorded. The card information is automatically written on an unused card supplied by the opponent and the PIN and card are entered into the real PIN pad and EFT terminal, which are also hidden. Upon receiving the transaction response from the issuer, the same response is sent to the fake terminal. In this way, the cardholder is made to think that the transaction has completed normally. PINs and counterfeit cards obtained in this manner are then used to commit fraud against the corresponding accounts.

There is no apparent practical defense against the above fake equipment attack. Financial institutions must therefore be willing to accept this threat, realizing that it is impossible to develop protection systems that completely eliminate all risks of fraud.

Key Management Requirements

Fundamental to any key management approach is the requirement (10) that: Personal verification and message authentication require a reference to be available to the authenticator at the time the authentication takes place. This reference, or the process used to generate the reference, must be defined and agreed to in advance. It must be such that the integrity of the reference (and sometimes its secrecy) and/or the process that generates the reference, can be assured by the authenticator.

Thus to check a received MAC at the issuer (associated with the transaction request message) an authentication key must either be stored or dynamically created at the issuer. The same requirement exists at the entry point in order to provide checking procedures for the MAC associated with the transaction response message sent from the issuer to the entry point.

To provide a check for message timeliness, required for detection of the replay of stale messages, the MACs must be time-dependent. Furthermore, for detection of the replay of stale messages and MACs, the time dependence must be controlled by the authenticating node.

This can be accomplished by using a universal time reference, T, e.g., a time-of-day (TOD) clock, which logically ties all nodes together. Here, each node must access T internally since otherwise T is not under the authenticating node's control.

Another way of establishing a common time reference is for the authenticating node to generate a random quantity which is transmitted to the sending node (i.e., the node desiring to send a message to the authenticating node). The sending node places this received random value in the message, generates a MAC on the message, and sends the message and MAC to the authenticating node.

To continue the discussion of authentication concepts: consider two nodes, A and B, which use a common message authentication key, Kauth. This key can be either static (in which case Kauth is stored permanently at both nodes) or dynamic (in which case Kauth may be generated by one or both of the nodes). Figure 11-15 summarizes the concepts discussed thus far.

For the case where Kauth is randomly generated, a key-encrypting key (KNC, or node communication key) must be used to encrypt Kauth so that it may be transmitted safely to the other node where it must be established.

For example, if one node generates Kauth, it can be sent in the form E_{KNC}(Kauth) to the other node. Details of protocols which incorporate the concepts of Figure 11-15 are suggested in Figures 11-16 through 11-19. These figures specifically show that initiation protocols are needed before regular communication can start, unless when a universal time reference is used in conjunction with a static Kauth. If created dynamically, Kauth could be generated either at node A or node B. Another possibility discussed in reference 13 is for Kauth to be formed from random quantities generated at node A and node B.

The notation MAC(argument) is used to show specifically the quantities that MAC depends on. For example, MAC(key, data) is interpreted as the leftmost m bits of the last block of E_{key}(data), where $m \leqslant 64$ is selected by the user and for convenience is omitted from the notation.

Choosing a particular design approach, i.e., system keys, personal keys, or both, affects the way references can be established (discussed below). After one decides how references are established, the major remaining effort must focus on the problem of how best to achieve separation of the personal verification and message authentication processes among different institutions. The details are discussed under the general topic of key management in each of the separate sections covering the PIN/system key approach, the

Authentication Key	Time Reference	
	Universal	System Generated
Kauth	time-of-day clock	Ta generated at node A Tb generated at node B
Static, i.e., Kauth is permanently stored at both nodes	MACa,b and MACb,a are functions of Kauth, TOD, and data sent between node A and node B.	MACa,b is a function of Kauth, Tb, and data sent from node A to node B. MACb,a is a function of Kauth, Ta, and data sent from node B to node A.
	No initiation protocol required.	Initiation protocol required to send Ta from node A to node B and Tb from node B to node A.
Dynamic, i.e., Kauth is randomly generated	MACa,b and MACb,a are functions of Kauth, TOD, and data sent between node A and node B.	MACa,b is a function of Kauth, Tb, and data sent from node A to node B. MACb,a is a function of Kauth, Ta, and data sent from node B to node A.
	Initiation protocol required to establish common authentication key (Kauth) between node A and node B.	Initiation protocol required to send Ta from node A to node B and Tb from node B to node A. In addition, a common authentication key (Kauth) must be established between node A and node B.

Legend:

IDa:	Identifier of node A
IDb:	Identifier of node B
Kauth:	Message authentication key
Ma,b:	Message sent from node A to node B
Mb,a:	Message sent from node B to node A
MACa,b:	Message authentication code for Ma,b
MACb,a:	Message authentication code for Mb,a
Ta:	System generated time reference at node A
Tb:	System generated time reference at node B
TOD:	Universal time reference stored at node A and node B

Figure 11-15. Concepts Associated with Authentication

Time Reference: Permanently stored time reference at both nodes, e.g., time-of-day clock (TOD)

Data sent from node A to node B:

TOD, IDa, IDb, Ma,b, MACa,b(Kauth,TOD,IDa,IDb,Ma,b)

Data sent from node B to node A:

TOD, IDb, IDa, Mb,a, MACb,a(Kauth,TOD,IDb,IDa,Mb,a)

Legend:

IDa:	Identifier of node A
IDb:	Identifier of node B
Kauth:	Message authentication key
Ma,b:	Message sent from node A to node B
Mb,a:	Message sent from node B to node A
MACa,b:	Message authentication code for Ma,b
MACb,a:	Message authentication code for Mb,a
TOD:	Universal time reference stored at node A and node B

Figure 11-16. Message Authentication—Universal Time Reference and Static Authentication Key

PIN/personal key approach, and the PIN/system key/personal key (or hybrid key management) approach.

One particular method that assures separation of the personal verification process among institutions uses a secret personal key stored on the card. A system with personal keys faces new threats.

Threats to the Secrecy of a Key Stored on a Magnetic Stripe Card

One major threat to the key-on-the-card (magnetic stripe card) is that cards can be lost and stolen, and information on the card can be copied. Furthermore, information entered into nonsecure terminals can be bugged, and unsuspecting customers can be induced to enter card and PIN information into fake equipment.

However, not all of these exposures present the same threat to EFT security, as will be seen from the discussion below. Note also that many of the exposures could be eliminated with an intelligent secure card (see also the sections entitled Bank Card Security, and Personal Key Approach with an Intelligent Secure Card).

A PIN is analogous to a user-remembered combination to a combination lock; a card with a secret personal key (KP) stored on it is analogous to a physical key to a key lock. It is true that physical keys can be lost, stolen, and duplicated, and therefore, that cards with secret keys written on them would be subject to the same exposures. Yet, keys and locks have been proven to be practical, useful, and worthwhile, even though they do not provide their users with perfect or absolute security. The same would be true of

Time Reference: Randomly generated time-variant quantities (Ta at node A and Tb at node B) establish origin of time reference, e.g., Ta and Tb are incremented by one for each message sent

Initiation protocol to exchange Ta and Tb:

From node A to node B: Ta, IDa, IDb
From node B to node A: Tb, IDb, IDa

Data sent from node A to node B:

Tb+i, IDa, IDb, Ma,b, MACa,b(Kauth,Tb+i,IDa,IDb,Ma,b)

Data sent from node B to node A:

Ta+j, IDb, IDa, Mb,a, MACb,a(Kauth,Ta+j,IDb,IDa,Mb,a)

Legend:

IDa:	Identifier of node A
IDb:	Identifier of node B
Kauth:	Message authentication key
Ma,b:	Message sent from node A to node B
Mb,a:	Message sent from node B to node A
MACa,b:	Message authentication code for Ma,b
MACb,a:	Message authentication code for Mb,a
Ta:	System generated time reference at node A
Tb:	System generated time reference at node B
i:	Message sequence number for Ma,b
j:	Message sequence number for Mb,a

Note: It must not be possible to influence the generation of Ta and Tb externally. Otherwise, stale messages associated with a previously used Ta and Tb can be inserted.

Figure 11-17. Message Authentication–System-Generated Time Reference and Static Authentication Key

bank cards with secret keys stored on them. Information stored on the card would be protected as a consequence of the physical security routinely provided to the card by its holder. Once users were aware of the security implications of exposing card information, it is assumed that they would take the necessary precautions to protect their cards, and by so doing, they would protect the information stored thereon.

Lost Cards

Loss of a card is probably the most common way in which card information becomes exposed, yet this represents the least serious threat to EFT security. An opponent is unlikely to launch an attack by first searching for a lost card, and people who find lost cards are unlikely to be motivated to tap the communications line and recover PINs using cryptographic methods.

Time Reference: Permanently stored time reference at both nodes, e.g., time-of-day clock (TOD)

Initiation protocol to exchange time-variant authentication key (Kauth) generated at node B

From node A to node B: IDa, IDb
From node B to node A: IDb, IDa, E_{KNC}(Kauth)

Data sent from node A to node B:

TOD, IDa, IDb, Ma,b, MACa,b(Kauth,TOD,IDa,IDb,Ma,b)

Data sent from node B to node A:

TOD, IDb, IDa, Mb,a, MACb,a(Kauth,TOD,IDb,IDa,Mb,a)

Legend:

IDa:	Identifier of node A
IDb:	Identifier of node B
Kauth:	Message authentication key
KNC:	Node communication key (key-encrypting key)
Ma,b:	Message sent from node A to node B
Mb,a:	Message sent from node B to node A
MACa,b:	Message authentication code for Ma,b
MACb,a:	Message authentication code for Mb,a
TOD:	Universal time reference stored at node A and node B

Figure 11-18. Message Authentication—Universal Time Reference and Dynamic Authentication Key

It is assumed that the cardholder would notify the issuer upon discovering that he has lost his card. The issuer would then invalidate the account and disallow further transactions until a new card had been issued. Thus the time during which fraud could be committed against an account would be relatively short.

Stolen Cards

Stolen cards are not much of a threat either, although here it must be assumed that the opponent is motivated and capable of attacking the system (which would include the tapping of communication lines, interception of authentication pattern values, and recovery of PINs). However, AP values cannot be intercepted after the cards are stolen, since the cards themselves are needed to initiate transactions. In addition, it is assumed that reissued cards would use different KPs, thus making the APs different.

For stolen cards to be of significant value, they must be stolen from selected individuals whose transaction request messages have been previously

Time Reference: Randomly generated time-variant quantities (Ta at node A and Tb at node B) establish origin of time reference, e.g., Ta and Tb are incremented by one for each message sent

Initiation protocol to exchange Ta and Tb as well as Kauth:

From node A to node B: Ta, IDa, IDb
From node B to node A: Tb, IDb, IDa, E_{KNC}(Kauth)

Data sent from node A to node B:

Tb+i, IDa, IDb, Ma,b, MACa,b(Kauth,Tb+i,IDa,IDb,Ma,b)

Data sent from node B to node A:

Ta+j, IDb, IDa, Mb,a, MACb,a(Kauth,Ta+j,IDb,IDa,Mb,a)

Legend:

IDa:	Identifier of node A
IDb:	Identifier of node B
Kauth:	Message authentication key
KNC:	Node communication key (key-encrypting key)
Ma,b:	Message sent from node A to node B
Mb,a:	Message sent from node B to node A
MACa,b:	Message authentication code for Ma,b
MACb,a:	Message authentication code for Mb,a
Ta:	System generated time reference stored at node A
Tb:	System generated time reference stored at node B
i:	Message sequence number for Ma,b
j:	Message sequence number for Mb,a

Note: It must not be possible to influence the generation of Ta and Tb externally. Otherwise, stale messages associated with a previously used Ta and Tb can be inserted. Since Kauth represents time-variant information generated at node B, it is not necessary also to generate Tb at node B. In that case, Ta could be used in place of Tb to generate MACb,a.

Figure 11-19. Message Authentication—System-Generated Time Reference and Dynamic Authentication Key

intercepted. For example, the opponent could tap the communications line from an EFT terminal and accumulate a file of transaction request messages for many different cardholders. (It is assumed that the cryptographically transformed PINs are sent in the transaction request messages.) By stealing the cards of one or more of these people, a computer could then be used to obtain the associated PINs. This would allow fraud to be committed against each account until such time as the card is reported stolen or missing and the issuer updates his data base to reject the invalid card.

Copying Card Information

Copying card information presents a serious threat provided that the card can be read without the cardholder's knowledge. In that case, the issuer has no basis for updating his data base to reject the invalid card. In effect, the opponent has enough time in which to attack the PIN (e.g., by intercepting the user's AP value and recovering the PIN using exhaustive methods, see the section Objections to the PIN/Personal Key Approach Using a Magnetic Stripe Card).

A particular scenario for obtaining card information follows: In any nationwide EFT environment, there is probably a very large number of nonsecure EFT terminals used without PINs for the purchase of merchandise. The same card used in ATMs and other PIN-using terminals could be used in these nonsecure terminals. Thus it is not especially difficult to bug a nonsecure terminal to read and record the key from the card along with the normal magnetic stripe information.

Dishonest retailers or employees who routinely handle customers' cards in the process of transacting business represent another threat. For example, it may be common practice for the cardholder to present his or her card to a clerk, cashier, or salesperson who enters it into the EFT-terminal's card reader. Before returning the card to the cardholder, a skillful opponent could easily skim information from many cards without being observed. The line from the retailer's PIN-using terminal would also be tapped to intercept AP values. After a time, information from a significant number of cards could be skimmed.

Bugging of Input Information at EFT Terminals

Information (PINs and KPs) entered into a terminal could be exposed to a bugging attack. A suggested defense is to interlock the terminal to an alarm so that any penetration of the device causes the alarm to be triggered. However, inexpensive terminals cannot have sophisticated alarm systems. Thus, for inexpensive terminals located in nonsecure retailer environments, there is an increased potential that card information and PINs will be exposed to a bugging attack. For the same reason, terminal-resident keys are also vulnerable in such an environment. Thus, the threat is not restricted to personal keys.

Insertion of Fake Equipment

The fake equipment threat described earlier to obtain PINs (see Threats to PIN Secrecy) is equally effective for obtaining KPs. There is no apparent practical defense to this threat if secret information is entered into a device not under the control of the customer.

THE PIN/SYSTEM KEY APPROACH

In the PIN/system key approach, personal verification is based solely on a secret PIN entered into the EFT terminal by the customer. The PIN is

often combined with other nonsecret information such as the cardholder's ID and this combined information is encrypted under a secret system-supplied key.[23] When the ID is coupled to the PIN, attacks against the PIN must take into account the ID. The work factor for certain dictionary attacks against the PIN is increased (see also Appendix D). The ANSI-proposed method for PIN encryption is described in Appendix E.

System keys are used to generate the MACs required for message authentication (Table 11-3). A separate dynamically generated transaction key KTR is used for MAC generation between each logically adjacent pair of nodes (e.g., the terminal and the acquirer, the acquirer and the switch, and the switch and the issuer). The keys KTR1 and KTR2 are used to distinguish further between the MACs generated on transaction request and transaction response messages, respectively. The subscripts x,y denote the sender (x) and the receiver (y). For example, KTR1term,acq represents a transaction key shared by the terminal and the acquirer, and is used by the terminal (the sender) to generate a MAC on a transaction request message sent to the acquirer (the receiver).

It is assumed that the transaction keys are generated at one node (either sender x or receiver y) and transmitted to the other node (receiver y or sender x, as the case may be) encrypted under a permanently installed system key shared by the two nodes (Table 11-3). Transaction keys transmitted between a terminal and the acquirer would be encyrpted under a terminal master key, KMT. Transaction keys transmitted between the acquirer and the switch, and between the switch and the issuer, would be encrypted under an interchange key, KI. The flows of information from an EFT terminal to the issuer and from the issuer back to the EFT terminal (including the keys and MACs used in the message authentication process) are shown in Tables 11-4 and 11-5. The PIN/system key approach is discussed in greater detail in the section Implementation of Fraud Prevention Techniques, in Chapter 10. Some variations on the key management approach shown in Tables 11-4 and 11-5 are listed below.

1. KTR1term,acq = KTR2acq,term
   KTR1acq,sw = KTR2sw,acq
   KTR1sw,iss = KTR2iss,sw
2. KTR1term,acq = KTR1acq,sw = KTR1sw,iss
   KTR2iss,sw = KTR2sw,acq = KTR2acq,term
3. Both 1 and 2 are satisfied (i.e., a single transaction key KTR is used).

In case 1, a different transaction key is established between each logically adjacent pair of nodes, and this key is used to generate the MACs for both Mreq and Mresp. In case 2, a first transaction key (KTR1) is established among the nodes (term, acq, sw, and iss) for the purpose of generating the MAC on Mreq and a second transaction key (KTR2) is established among

[23]If entered into a PIN pad with encryption capability, the PIN may be encrypted under a secret PIN-pad key. However, a terminal key is still required unless the PIN-pad key can also be used to generate the MAC's which are required for message authentication.

System User	Bank Card	System Nodes: EFT Terminal	Acquirer's Host	Switch's Host	Issuer's Host
		Permanently Installed Master Keys			
none	none	KMT	KMHacq	KMHsw	KMHiss
		Permanently Installed Interchange Keys			
none	none	none	KIacq,sw	KIacq,sw KIsw,iss	KIsw,iss
		Dynamically Generated Keys Used for MAC Generation on the Transaction Request Message, Mreq			
none	none	KTR1term,acq	KTR1term,acq KTR1acq,sw	KTR1sw,iss KTR1acq,sw	KTR1sw,iss
		Dynamically Generated Keys Used for MAC Generation on the Transaction Response Message, Mresp			
none	none	KTR2acq,term	KTR2acq,term KTR2sw,acq	KTR2iss,sw KTR2sw,acq	KTR2iss,sw

Note: Keys used for personal verification at the issuer and switch are not shown.

Legend:

KMT: Terminal master key
KMH: Host master key
KI: Interchange key
KTR1: Transaction key for MAC generation on transaction request message (Mreq)
KTR2: Transaction key for MAC generation on transaction response message (Mresp)

Table 11-3. Keys Used for Message Authentication–PIN/System Key Approach

		System Nodes			
System User	Bank Card	EFT Terminal	Acquirer's Host	Switch's Host	Issuer's Host
1 Enter PIN into terminal.	2 Enter card into terminal and read card information.	3 Generate AP. 4 Formulate Mreq which includes time dependent information from acquirer (TODacq) and terminal (Tterm). 5 Generate MAC1term,acq with KTR1term,acq. 6 Send Mreq and MAC1term,acq to acquirer.	7 Check received MAC1term,acq with KTR1term,acq of reference and TODacq of reference; retain Tterm randomly generated by terminal. 8 Check for correct destination. 9 Generate MAC1acq,sw with KTR1acq,sw. 10 Send Mreq and MAC1acq,sw to switch.	11 Check received MAC1acq,sw with KTR1acq,sw of reference and TODsw of reference. 12 Check for correct destination. 13 Generate MAC1sw,iss with KTR1sw,iss. 14 Send Mreq and MAC1sw,iss to issuer.	15 Check received MAC1sw,iss with KTR1sw,iss of reference and TODiss of reference. 16 Verify user. 17 Decide if Mreq is to be honored. 18 Formulate Mresp which includes time information stored at issuer (TODiss) as well as time information generated by the terminal (Tterm).

Note: It is assumed that the acquirer periodically sends time-of-day information (TODacq) to the terminals in its domain. The terminal, on the other hand, generates random information (Tterm) and sends it to the acquirer. This can be done as part of the initiation protocol (Figure 11-15). The TOD stored at the other network host nodes (TODsw at the switch and TODiss at the issuer) is assumed to be equal to TODacq within an allowable range (ΔTOD).

The integers 1–18 in the table show the sequence of steps in the transaction.

Table 11-4. Information Flow from Terminal to Issuer—PIN/System Key Approach

System User	Bank Card	System Nodes: EFT Terminal	Acquirer's Host	Switch's Host	Issuer's Host
	32 Eject card from terminal.	29 Check received MAC2acq,term with KTR2acq,term of reference and Tterm of reference.	25 Check received MAC2sw,acq with KTR2sw,acq of reference and TODacq of reference.	21 Check received MAC2,iss,sw with KTR2iss,sw of reference and TODsw of reference.	18 Formulate Mresp which includes time information stored at issuer (TODiss) as well as time information generated by the terminal (Tterm).
		30 Decide if Mresp is to be accepted or rejected.	26 Check for correct destination.	22 Check for correct destination.	19 Generate MAC2iss,sw with KTR2iss,sw.
		31 If Mresp is accepted, process transaction; otherwise, abort transaction request.	27 Generate MAC2acq,term with KTR2acq,term.	23 Generate MAC2sw,acq with KTR2sw,acq.	20 Send Mresp and MAC2iss,sw to switch.
			28 Send Mresp and MAC2acq,term to terminal.	24 Send Mresp and MAC2sw,acq to acquirer.	

Note: It is assumed that the acquirer periodically sends time-of-day information (TODacq) to the terminals in its domain. The terminal, on the other hand, generates random information (Tterm) and sends it to the acquirer. This can be done as part of the initiation protocol (Figure 11-15). The TOD stored at the other network host nodes (TODsw at the switch and TODiss at the issuer) is assumed to be equal to TODacq within an allowable range (ΔTOD).

The integers 18–32 in the table show the sequence of steps in the transaction.

Table 11-5. Information Flow from Issuer to Terminal—PIN/System Key Approach

the nodes for the purpose of generating a MAC on Mresp. In case 3, a single transaction key KTR is established among the nodes for the purpose of generating the MACs on Mreq and Mresp. In each case, the protocols vary slightly depending on the method used to generate, transmit, and intitialize the various transaction keys.

Key Management Considerations for PIN/System Key Approach

A major objective of the PIN/system key approach is to provide a key management scheme which is transparent to the user (i.e., it does not require keys to be supplied by the user). This means that system keys are used to encrypt PINs and generate message authentication codes. However, the keys at the entry points are not generally known to the issuer (final destination). Thus as messages are routed through the network, encrypted PINs must be decrypted and reencrypted and MACs must be generated.

Sharing of Secret Keys

The PIN/system key approach requires limited sharing of keys so that an institution may recover (decrypt) encrypted PINs and regenerate MACs. Hence financial institutions must be willing to exchange and use interchange keys. On the other hand, key management designs must consider the constraint that financial institutions are unwilling to share, as a condition of joining the interchange, all of their secret keys (e.g., terminal master keys, host master keys) with other institutions.

Cryptographic Translations

In addition to the cryptographic operation or operations required at the EFT terminal to transform PINs and generate MACs, one or more cryptographic operations are required in the security module of the acquiring institution (or designated node) and in the security module of the switch (if used) to allow PINs and MACs traversing the system to be transformed and regenerated, respectively, so that ultimately they are in a form that can be comprehended by the issuing institution.[24]

PIN Translation at the Issuer

If PIN validation is coupled with MAC validation, then PIN translation at the issuer is not required. However, if PIN validation is separate from MAC validation, then a PIN translation is likely to be required at the issuer to transform the PIN of reference, the received PIN, or both, into a form that will allow them to be compared. There are different reasons for this. For example, the issuer may not wish to share the key under which the PIN of reference is encrypted. Or the received PIN may be encrypted under a terminal-generated key, and in turn this key may be transmitted to the issuer

[24]In the PIN/system key approach discussed in Chapter 10, the PIN and MAC must undergo translation at a comparable number of points in the network.

encrypted under the issuer's interchange key. Ultimately, each different key management scheme will have its own different requirement for PIN translation at the issuer.

Protection Against Misrouted Data

Since PINs must be transformed from encryption under one key to encyrption under another key as messages are routed through the system, PIN information may be misrouted accidentally or intentionally. If an opponent were to cause PINs to be translated to encryption under a known key, these PINs would be exposed. Thus the PIN/system key approach must incorporate methods to detect misrouting, thus assuring that PINs are routed only to the proper destinations.

Defending Against the Misrouting Attack

If a PIN/system key approach to EFT security is improperly designed, an intentional misrouting attack may be possible. For example, an opponent might alter the destination bank ID to one of his own choosing. Thus at a selected system node (e.g., the switch) a PIN could be translated from encryption under an interchange key KIi to encryption under an interchange key KIx rather than the intended interchange key KIj. The opponent, who knows KIx, could then recover all misrouted PINs.

A defense against the misrouting attack can be provided by coupling the destination bank ID to the appropriate destination interchange key (see also reference 14). The procedure ensures that the indicated destination bank ID is the one designated at the creation of the transaction request message. At the sender this is made possible by calculating a message authentication code on the information requiring protection (i.e., the transaction request message which contains the bank ID, time-variant information such as a time stamp, and the encrypted PIN).

Before a TRANSLATE operation is attempted, a MAC of reference is generated for the received data = [T, BIDj, E_{KIi}(PIN)], where T denotes the time variant information, BIDj denotes the destination bank ID, and E_{KIi}(PIN) denotes the PIN encrypted under interchange key KIi. The MAC of reference is then compared for equality with the received MAC. If the two MACS are identical, the TRANSLATE operation is enabled. Otherwise an error condition is noted and the TRANSLATE operation is inhibited.

Using the Data Encryption Standard (DES) in the Cipher Block Chaining (CBC) mode (Chapter 2 and Figure 11-20) will suffice to detect as little as a one-bit change in the entire message. Note that MAC does not have to be 64 bits long. A smaller number of bits could be used with a corresponding loss of error detection capability (i.e., the probability of detecting an error is decreased).

The CBC mode can also be used if both secrecy and authentication are required. If two different keys are employed (i.e., one key for encryption and another for MAC generation), a strong procedure is effected. However, this is not the case if only one key is employed. An insecure procedure results if data are

first encrypted as shown in Figure 2-17 and a MAC is generated on the encrypted data employing the CBC mode as indicated in principle in Figure 11-20. With known plaintext and matching ciphertext, a string of ciphertext blocks and MAC can be constructed that will pass the authentication check, although the recovered plaintext will be "garbage" (suggested by D. Coppersmith, IBM Thomas J. Watson Research Center, Yorktown Heights, N.Y.). If the MAC is generated first, followed by encryption, a change in ciphertext will not generally be propagated to the MAC field due to the self-synchronizing property of the CBC mode. *Thus, message authentication is not achieved if the same key is used for MAC generation and message encryption.*

Two implementations to detect misrouting are discussed. In the first approach (case 1), interchange keys are stored in the clear in secure hardware and are used directly during execution of the TRANSLATE operation (Figure 11-20).

In the second approach (case 2), the interchange keys are stored in encrypted form outside the secure hardware and are supplied to the TRANSLATE operation as additional parameters. This is a more economical approach, since less storage is required within the secure hardware.

The interchange keys could, for example, be stored encrypted under some unique variant of the master key (KMx) residing in the secure hardware (see Chapters 4 and 5). As a result, there is a table of encrypted keys like this one:

BID1	$E_{KMx}(KI1)$
BID2	$E_{KMx}(KI2)$
⋮	⋮
BIDj	$E_{KMx}(KIj)$
⋮	⋮
BIDn	$E_{KMx}(KIn)$

However, the TRANSLATE operation cannot distinguish between one encrypted key and another, i.e., between $E_{KMx}(KIi)$ and $E_{KMx}(KIj)$, when they are not stored within the confines of the secure hardware. Therefore, a test is needed to ensure that the correct BIDj, $E_{KMx}(KIj)$ pair is used in the TRANSLATE operation.

In devising such a test, advantage can be taken of the fact that the correct bank ID (BIDj) already resides within the secure hardware (Figure 11-20), provided that the MAC check was successfully completed. Using the methods for validating time-invariant data discussed in the section Authentication of Time-Invariant Data, in Chapter 8, a validation pattern (VP) which is a function of this BID, say $VP = E_{BID}(BID)$, is defined. The VP, in turn, is linked to a test pattern (TP) and the quantity to be checked, $E_{KMx}(KIj)$, as shown in Figure 11-21.

The table of encrypted keys is now extended by including the test pattern for each encrypted key, i.e.,

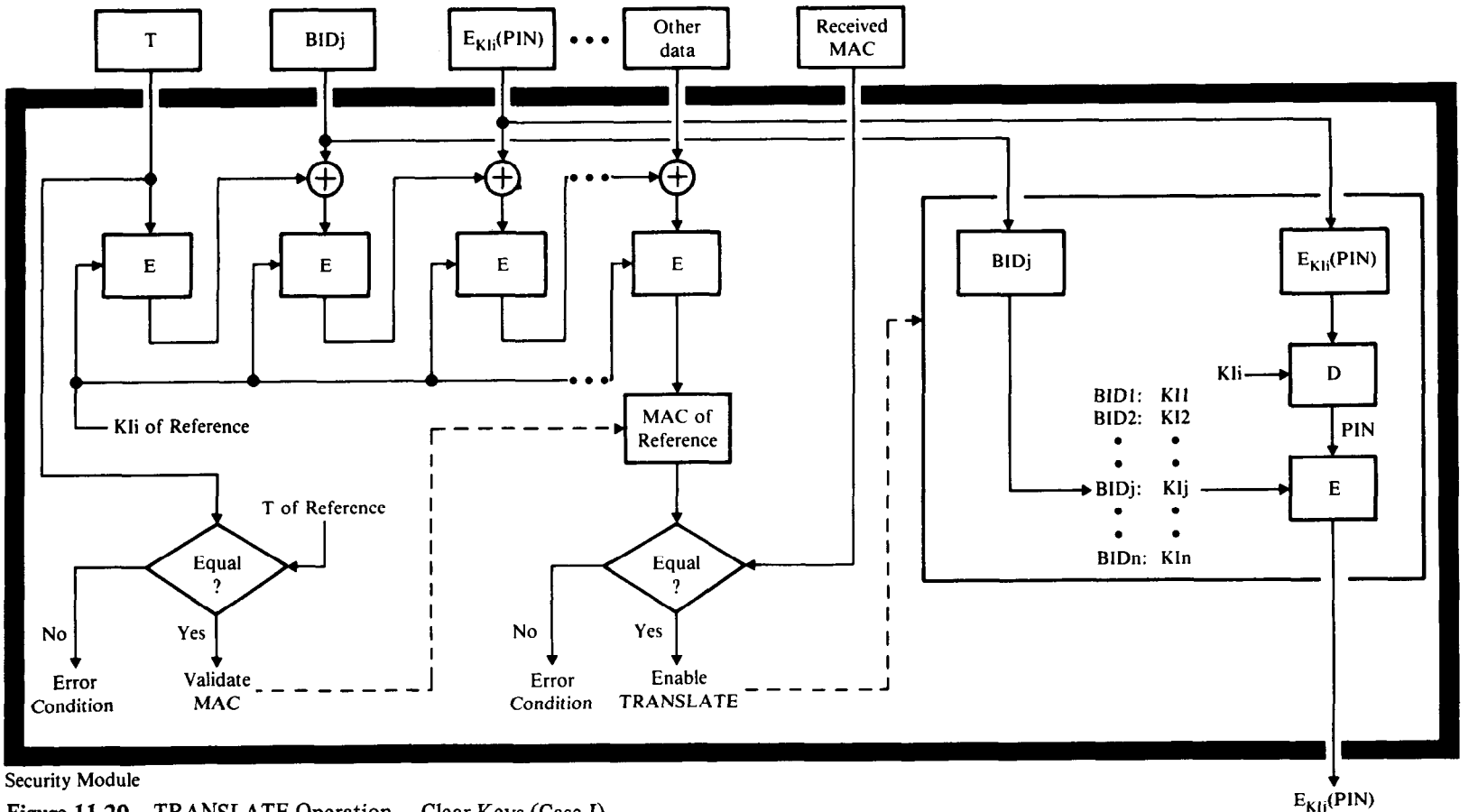
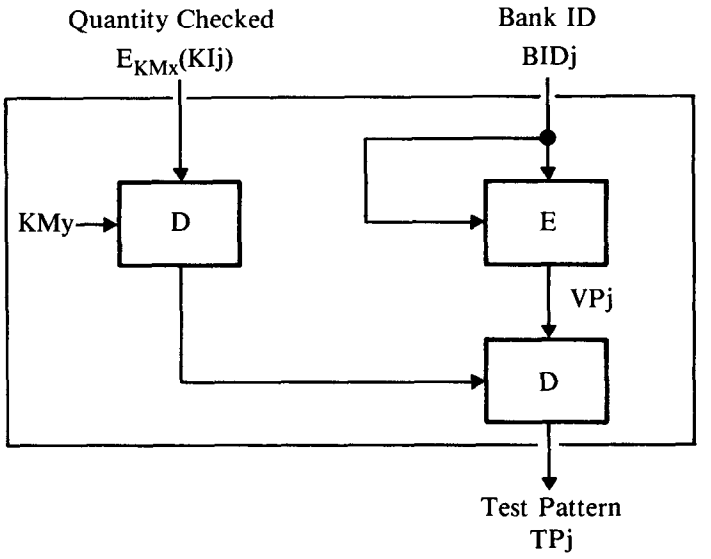


Figure 11-20. TRANSLATE Operation – Clear Keys (Case I)



Note: Test pattern is generated under secure conditions and makes use of a special variant of the Master Key (KMy).

Figure 11-21. Generation of Test Pattern

BID1	$E_{KMx}(KI1)$	TP1
BID2	$E_{KMx}(KI2)$	TP2
⋮	⋮	⋮
BIDj	$E_{KMx}(KIj)$	TPj
⋮	⋮	⋮
BIDn	$E_{KMx}(KIn)$	TPn

The encrypted key can now be authenticated using the method indicated in Figure 11-22.

After the checks for content (via MAC), timeliness (via T and MAC), and proper translate key (via TP) have been successfully completed (all in secure hardware), the TRANSLATE operation is enabled and allowed to execute (again in secure hardware, Figure 11-23). If any of the checks fail, the TRANSLATE operation is not enabled.

The approach described here blocks misrouting attacks by enabling the TRANSLATE operation only after the proper bank ID/key relationship has been established. Another defense against the misrouting attack is to transform the personal verification information at the entry point so that the resulting value is a one-way function of the input information. This is achieved by supplementing PINs with personal keys. Misrouting attacks are, in that case, ineffective because the personal key provides end-to-end authentication. An additional advantage is that a simpler key management is achieved [see the section PIN/Personal Key/System Key Approach (Hybrid Key Management) Using an Intelligent Secure Card].

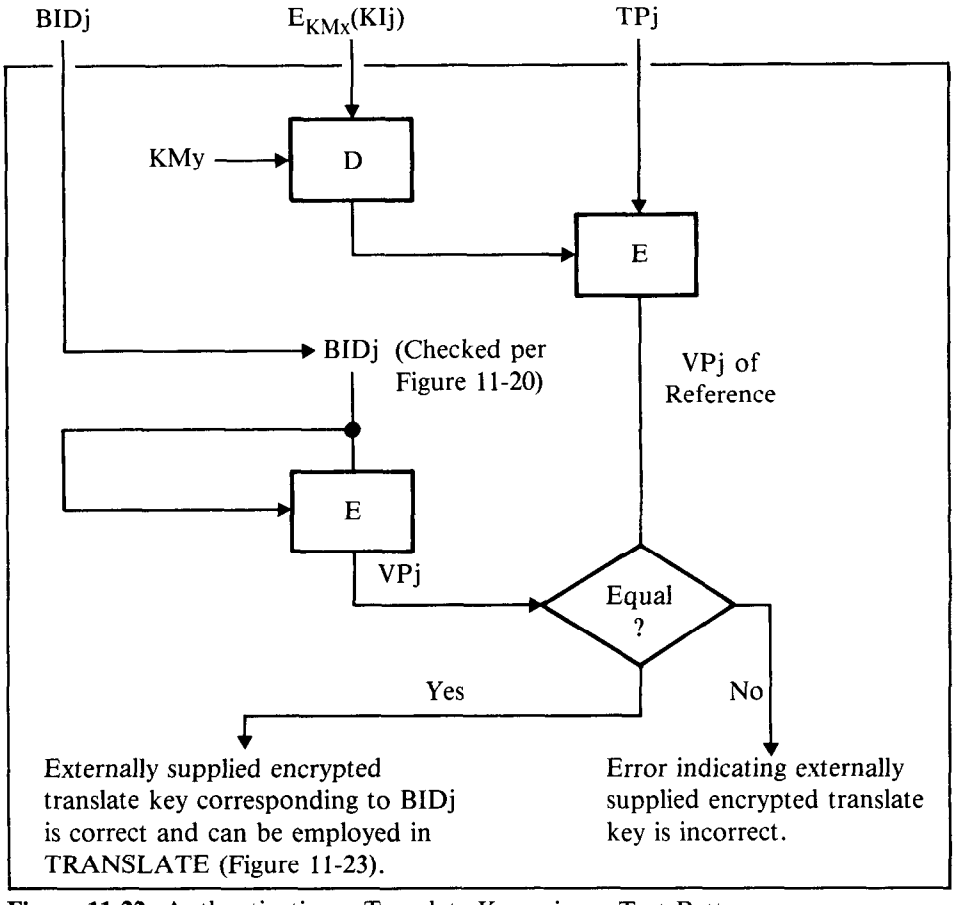


Figure 11-22. Authenticating a Translate Key using a Test Pattern

When personal verification information, i.e., PIN, is protected exclusively via system-controlled keys, key management must be specifically designed to prevent misrouting attacks. The described methods solve the problem by coupling routing information (e.g., bank IDs) with corresponding cryptographic keys via a message authentication code. Only after the proper bank ID/key relationship has been established is the TRANSLATE operation enabled. The TRANSLATE operation works with clear keys stored in secure hardware (case 1) or with encrypted keys stored externally (case 2). But, in the latter case, the encrypted keys must also be coupled to their respective identifiers, which is accomplished here by introducing a checking procedure based on stored test patterns.

Methods to prevent misrouting of PINs could also be extended to include messages. As an illustration of this, consider the case where the MAC on the transaction request message is generated at the EFT terminal using a resident terminal key. At the acquiring institution, the received MAC is replaced with a new MAC generated under KIi. Likewise, at the switch, the MAC generated under KIi is replaced with a MAC generated under KIj. Thus, the generation

of a valid MAC on a misrouted message is prevented by enabling the MAC generation operation (as was suggested with the TRANSLATE operation) only if the validated bank identifier BIDj received in the transaction request message agrees with the bank identifier corresponding to the supplied encrypted key $E_{KMx}(KIj)$.

A PIN/System Key Approach for Noninterchange

By definition, a secure PIN/system key approach can be devised for a noninterchange (local) environment, since the acquirer and the issuer are one and the same. Although the processes of personal verification and authentication of transaction request messages involve the user, the EFT terminal, and the HPC, the EFT terminal and the HPC are components of the issuer. Therefore, personal verification and authentication of transaction request messages involve only the user and the issuer.

Authentication of transaction response messages involves only the issuer and EFT terminal. Under the established protocol, the decision to approve or disapprove transactions is made by the issuer; EFT terminals merely respond to the commands received from the issuer's HPC in the transaction response messages. Hence, each institution controls its own EFT security.

A PIN/System Key Approach for Interchange

The EFT security achieved implicitly with a PIN/system key approach in a noninterchange environment is not achieved in an interchange environment. In an interchange environment, the acquirer and issuer may represent different financial institutions, and users may therefore interact with EFT terminals not owned or managed by the issuer. Thus, each institution must trust that:

1. The secret keys it shares with other institutions, or with a switch (if used), will be adequately protected.
2. Each other institution will implement appropriate security measures in its terminals and PIN entry devices to protect PINs.
3. PINs—which may be either included in the message in encrypted form or used in the computation of the MAC—messages and their corresponding MACs will be transmitted via the network to their proper destinations in accordance with an agreed-upon protocol ensuring both PIN secrecy and message integrity.

It follows from these statements that institutions must trust one another and, if used, they must trust the switch. Complete independence, isolation, and separation among the institutions with respect to personal verification and message authentication are thus unattainable with the PIN/system key approach.

Transaction response messages are not merely commands to be acted upon by the originating EFT terminal. Each institution must agree that in its role as an acquiring institution, it will honor transaction requests on the basis

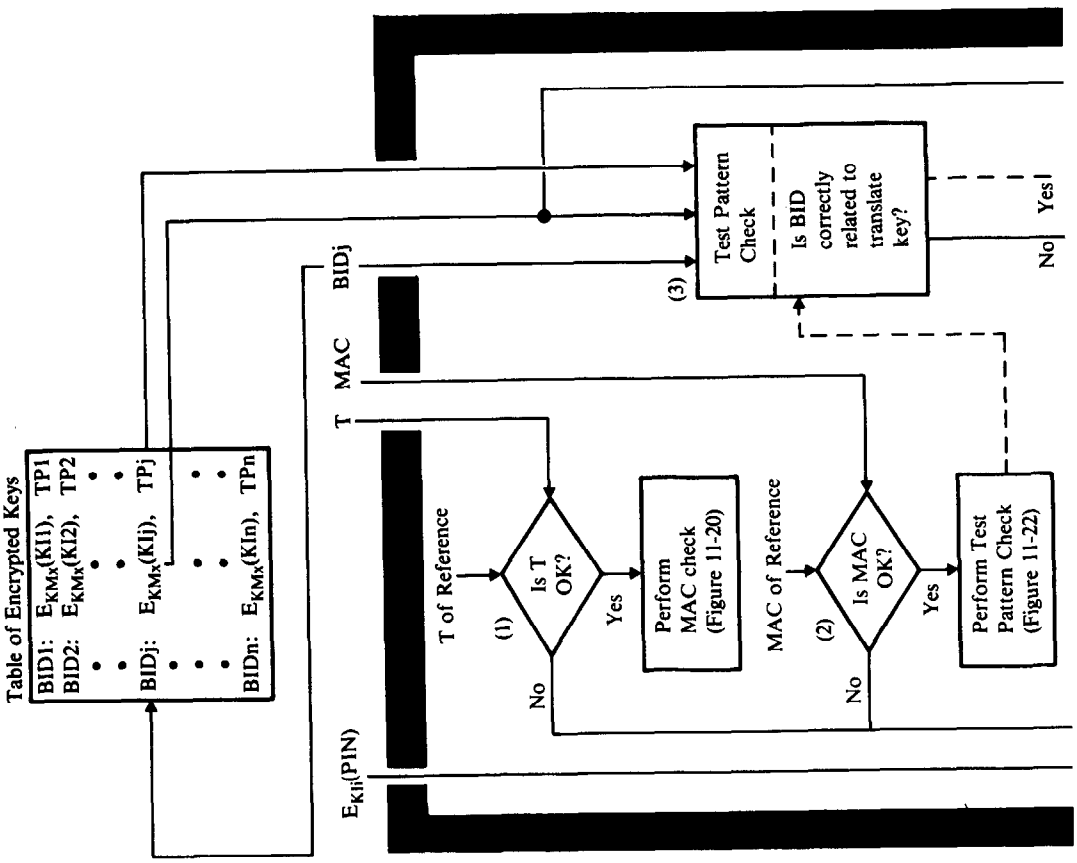
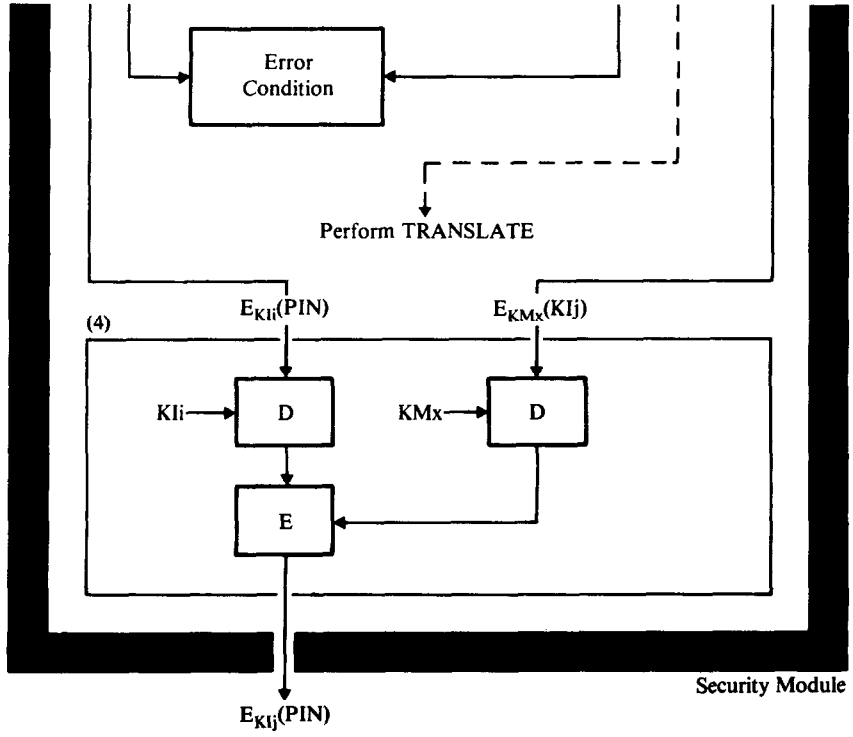


Table of Encrypted Keys
BID1: E_{KMx}(KI1), TP1
BID2: E_{KMx}(KI2), TP2
BIDj: E_{KMx}(KIj), TPj
BIDn: E_{KMx}(KIn), TPn
BIDj
MAC
T
T of Reference
(1)
Is T OK?
No
Yes
Perform MAC check (Figure 11-20)
MAC of Reference
(2)
Is MAC OK?
No
Yes
Perform Test Pattern Check (Figure 11-22)
(3)
Test Pattern Check
Is BID correctly related to translate key?
No
Yes
E_{KII}(PIN)



TRANSLATE is enabled only if all the following checks are successful:

1. T for timeliness (Figure 11-20)
2. MAC for content (Figure 11-20)
3. TP for correct translate key (Figure 11-22)

Figure 11-23. TRANSLATE Operation – Encrypted Keys (Case 2)

of receiving a valid transaction response message and MAC from the issuing institution. Message authentication permits the originating EFT terminal to detect bogus response messages that may be injected into the communication line. However, the cryptographic equivalent of a signed message or digital signature (Chapter 9) would be required for the acquirer to have incontrovertible proof that transactions have been authorized by the issuer.

If institutions are willing to join together in an interchange such that each institution trusts all other institutions and the switch, each institution is willing to share secret keys with other institutions or, at least, share one secret key with a switch, and each institution is willing to implement a key management that will allow PINs, messages, and generated MACs to be sent throughout the system in a protected manner in accordance with the established protocol, then an acceptable PIN/system key approach to EFT security is possible.

Disadvantages of the PIN/System Key Approach

Compromise of a System Key Allows Global Attacks Against PINs

The obvious disadvantage of using system keys to protect PINs is that a compromise of any one of these keys will reveal a large number of PINs. (On the other hand, a knowledge of one personal key will reveal at most only the associated PIN for that one account number.) System keys must therefore be provided an especially high degree of security.

For increased security, the PINs could be encrypted with a continually changing key such that the new key is a one-way function of the old key. The purpose of changing the key at time T is to protect traffic encrypted under a previous key, since the new key does not reveal the keys used at times previous to T. As a consequence, PINs cannot be obtained from intercepted encrypted PINs at times previous to T, even if the new key is compromised. One method for achieving this property is to use a decimal counter which holds a cycle count [15]. This counter is incremented whenever a new key is desired, preferably after every transaction. The incremented cycle count, concatenated with the device's identity, is encrypted under the old key, and the resulting cipher is used as a new key. The HPC and security module can track this operation, provided the current cycle count for each transaction is known. This can be accomplished by including the current cycle count in each transaction request sent to the issuer.

Although this method works in principle, a synchronization requirement is introduced, since the communicating nodes must track the keys even in the presence of system errors. This may be unacceptable in some system designs.

Exposure of Keys at the Entry Point

The suggested protection against terminal intrusion, which could lead to an exposure of the secret terminal key, is to interlock the terminal key so that any penetration of the device causes the key to be erased, thus making the device inoperative. However, the manufacture of tamper-proof terminals,

although perhaps possible, could lead to unacceptable, increased costs to the respective financial institutions. Even under the assumption that the secret terminal key is interlocked, so that any penetration of the device causes the key to be erased, it must be realized that with enough time and sophisticated equipment the interlock can generally be defeated. Then the terminal key could be recovered.

Key Management is Not Robust

The term robust is used frequently in statistical analyses where the assumption is often made that the underlying probability distributions are Gaussian or normal (i.e., follow a bell-shaped curve). But such an assumption is often incorrect. The question then is: How are the results affected by a deviation from the original assumption of a normal distribution? If deviations do not seriously affect the results, one says the method is robust.

In the PIN/system key approach, key management must defend against dictionary and misrouting attacks. However, weak key management at one network node could expose the PINs of many users, and therefore the security of the entire network could be jeopardized (i.e., a weakness at one point in the system could affect the security of the entire system). Therefore one can say that the PIN/system key approach is not robust.[25]

Advantages of the PIN/System Key Approach

1. Although separation of the authentication process among institutions is not possible with the PIN/system key approach, it nevertheless achieves many of the EFT security requirements stated previously, and provides a strong defense against threats posed by outsiders (those with access only to external system interfaces).
2. The PIN/system key approach can be implemented using existing technology and is in compliance with present bank card format standards and emerging PIN management standards.
3. With secret keys installed in the EFT terminals, there is automatically something with which the terminal can be interlocked (i.e., the secret terminal key) so that any penetration of the device causes the key to be erased and the terminal to become inoperative. Without a secret key, the terminal would have to be interlocked to an alarm or hardware-disabling device.

Conclusion: Although the PIN/system key approach does not satisfy all of the stated EFT security requirements, it does provide a reasonable level of protection, using existing technology and current banking practices and standards, for present EFT systems and those planned for the near future.

[25]A hybrid key management approach is less vulnerable to global exposures as discussed in the section The PIN/Personal Key/System Key (Hybrid Key Management) Approach using an Intelligent Secure Card. Such an approach is thus more robust. In other words, security weaknesses introduced at some nodes in the network do not have a major effect on the overall security of the network.

THE PIN/PERSONAL KEY APPROACH

The personal key approach attempts to improve the process of personal verification by eliminating the need for a secret key in the EFT terminal. Instead a personal key is recorded on each customer's bank card. The key, together with the cryptographic algorithm, is then used to encipher the PIN and generate the MACs needed to authenticate the transaction request and transaction response messages.

The personal key could be stored on either a magnetic stripe card or an intelligent secure card. An intelligent secure card offers greater security, although the magnetic stripe card can provide a migration path to a system incorporating personal keys stored on intelligent secure cards. A PIN/personal key approach using a magnetic stripe card is described first. An approach in which the personal key is stored on an intelligent secure card is described in the section Personal Key Approach with an Intelligent Secure Card.

Description of a PIN/Personal Key Approach Using a Magnetic Stripe Card

Assume that cardholders are authenticated on the basis of an authentication parameter, defined as $AP = E_{KP \oplus PIN}(ID)$, a copy of which is stored in a verification table in the issuer's HPC.[26] Message authentication between the user and issuer, and between the issuer and originating EFT terminal, is based on a MAC produced from the message using a transaction key (KTR) computed dynamically in the EFT terminal from KP, PIN, and ID, as follows: $KTR = D_{KP \oplus PIN}(ID)$.[27] Since $E_{KP \oplus PIN}(ID)$ was already used for an authentication parameter, the decipher operation is used to define KTR. A copy of each user's KTR is also stored in the issuer's verification table.

When a customer initiates a transaction and his card is read, the card information (including KP and ID) is transferred to the terminal and his PIN is entered via a suitable entry device. The PIN and KP are Exclusive-ORed to produce an intermediate key, which is used in turn to produce AP and KTR by enciphering and deciphering the ID, respectively. A transaction request message is then formed, which consists of a time stamp supplied by the acquirer's HPC (TODacq), a message sequence number supplied by the terminal,

[26] The verification table could be eliminated by defining a personal authentication code, $PAC = E_{KA}(AP)$, which is stored on the bank card. In such a case, the issuer would need to store only KA. (See also Figure 11-9.)

[27] A one-way function of KP, PIN, and ID (e.g., $KTR = D_{KP \oplus PIN}(ID)$) has an advantage over a function that is not one-way (e.g., $KTR = KP \oplus PIN$). If KTR were defined as $KP \oplus PIN$, then knowledge of KTR would permit an equivalent KP and PIN (say KP* and PIN*) to be derived such that $KP^* \oplus PIN^* = KP \oplus PIN$. Therefore, unauthorized entry to the system could be gained at any EFT terminal by using KP* and PIN*. On the other hand, if KTR is a one-way function of KP and PIN, there is no practical way to devise KPs and PINs that could be used to gain entry to the system. In that case, an opponent would be forced to conduct an active attack wherein a previously intercepted AP value (corresponding to KTR) is inserted into a bogus message and a MAC is then generated for the message using the compromised KTR.

the user's account number, the user's authentication parameter, the terminal's ID, the transaction type, and the transaction data. The time stamp allows the issuer to check the timeliness of the transaction request message. The message sequence number, which is returned in the transaction response message, allows the EFT terminal to check the timeliness of the response. (See also Figures 11-15 through 11-19 for a discussion of time stamps.) A MAC is then generated for the transaction request message using KTR, and the transaction request message and MAC are sent to the issuer.[28]

The issuer validates the message using the KTR of reference stored in his data base filed under the user's identifier (ID). A MAC of reference is computed from the message and the KTR of reference. The MAC of reference is then checked for equality with the received MAC. The time stamp in the received message is also checked against the time stamp of reference to ensure that the message is current (not a stale message). If both tests succeed, the message is validated. The time stamp of reference is then replaced by the received time stamp.

The received AP value is next compared for equality with the AP of reference also filed under the user's identifier (KD) in the issuer's data base.[29] If the two AP values are equal, the issuer concludes that the secret data supplied by the user (KP and PIN) are properly related to the claimed ID. If the requested transaction can be honored, a positive response is sent to the originating terminal; otherwise, a negative response is sent.

The positive and negative responses could consist of request for transaction granted and request for transaction not granted, respectively (where "transaction" denotes the transaction request message repeated in its entirety). A MAC is also generated for the transaction response message using KTR, and the transaction response message and generated MAC are sent to the EFT terminal.

Upon receipt of the response message, the terminal generates a MAC of reference from the stored KTR of reference and the received message. The received MAC is then compared for equality with the MAC of reference. The message sequence number in the received message is also compared for equality with the message sequence number of reference stored previously in the EFT terminal (at the time it was generated). If both the MACs and message sequence numbers are equal, the response is validated; otherwise, it is not. If a validated positive response is received, the terminal honors the

[28] Since KTR is a function of KP and PIN, MAC is by definition an authentication parameter. Thus, personal verification could be based on the MAC (i.e., an AP value in the message is unnecessary). In the proposed ANSI standard for PIN management and security [11], PIN validation is treated separately from message authentication. For this reason, personal verification and message authentication are treated separately here. An EFT system in which authentication of the transaction request message and personal verification are based solely on the MAC is described in the section The PIN/Personal Key/System Key (Hybrid Key Management) Approach Using an Intelligent Secure Card.

[29] If PAC = E_{KA}(AP) is stored on the bank card, verification takes place as follows. The received value of AP is encrypted with the issuer's KA of reference to generate a PAC of reference. If PAC of reference equals the received PAC, the user is accepted; otherwise, he is rejected.

transaction. If either an invalid or negative response is received, the transaction is denied.

Key Management Considerations for PIN/Personal Key Approach

The major objective of the PIN/Personal Key approach is to devise a key management which minimizes the need for system keys. This means that personal authentication information is a function of KP and PIN only. This also means that message authentication codes associated with transaction request messages are a function of KP (but not a function of system keys). Thus to check a received MAC (associated with the transaction request message) the issuer needs a dynamically computed MAC of reference. This in turn means that KP must be stored or recreated at the issuer dynamically.

The requirement to generate a MAC of reference exists also at the entry point. This allows the MAC associated with the transaction response message sent from the issuer to the entry point to be checked.

A reference must be available at both the issuer and the entry point to permit MACs to be checked. With the PIN/Personal Key approach, that reference would be KP or a secret value related to KP and perhaps PIN. The important thing is that the integrity and secrecy of that reference be assured. It is relatively easy to satisfy this requirement at the issuer where strict security procedures can be enforced. This means that the processes of personal verification and authentication of transaction request messages can be isolated to the respective issuing institutions. However, the situation is different at the entry point. A secret reference cannot be stored in the EFT terminal, since doing so would conflict with the intent of the personal key approach. The personal key approach attempts to eliminate the storage of a secret key (or parameter) in the EFT terminal and thus eliminate the need to manage and maintain the secrecy of terminal resident keys (or parameters). Since storing a reference in the EFT terminal is effectively the same as storing a key, a practical solution is not achieved. The only remaining alternative is to store the reference on the card. However, the price paid for doing this is that the process of authentication of transaction response messages cannot be isolated among the respective institutions.

Advantages of the PIN/Personal Key Approach

Increased Number of Combinations of Secret User-Supplied Information

With an increased number of combinations of secret user-supplied information, discovery of a PIN and personal key via attacks using exhaustive methods are computationally infeasible. Trial and error methods at the entry point interface where the user supplies his information or exhaustive methods performed on the system via a programming interface, are effectively thwarted.

End-To-End Protection Between the User and Issuer

In an interchange environment, a user-supplied key provides true end-to-end cryptographic protection between the user and issuer. Secret interchange

keys shared with other institutions or with a switch are unnecessary. Also, cryptographic transformations at the acquirer, the switch, or other intermediate network nodes to decipher data under one key and reencipher them under another key are unnecessary. Thus data are not exposed in the HPC of other institutions, or even in the security modules of those institutions.

In summary, the PIN/personal key approach does not require

1. Sharing of secret keys
2. Cryptographic translation of data
3. Protection against data misrouting
4. PIN translation at the issuer

as would be the case with the PIN/system key approach. This reduces the complexity of key management.

Objections to the PIN/Personal Key Approach Using a Magnetic Stripe Card

Although a magnetic stripe card can be used for storing KP, there are several objections that favor the intelligent secure card as the storage medium for such a KP. For example, a key stored on the magnetic stripe card could be compromised via skimming or bugging. In addition, a key stored on the magnetic stripe card offers no protection against certain active fraud threats as discussed below in the section Exposures Due to Misuse of Personal Keys and Fake Personal Keys.

A Key on the Magnetic Stripe Card Cannot be Protected

One of the major objections raised against storing a key on the magnetic stripe card is that the key cannot be adequately protected, i.e., it cannot be maintained as a secret in a practical EFT system. In a nationwide EFT environment, there is probably a very large number of nonsecure terminals used without PINs and cryptography for the purchase of merchandise. In such an environment a key on the bank card would be exposed. (See also the section above entitled Threats to the Secrecy of a Key Stored on a Magnetic Stripe Card.)

If a cardholder's personal key should become compromised, the associated PIN can also be ascertained with only a small additional effort provided that a transaction request message initiated by the cardholder can be intercepted in the network domain where PINs are used. Consider this case: assume that cardholders are authenticated on the basis of an authentication parameter (AP) included in the transaction request message, where AP is defined as $AP = E_{KP \oplus PIN}(ID)$. With a compromised KP and the corresponding intercepted AP and ID, the PIN can be recovered using a method of direct search. With a four-digit PIN there would be 10,000 combinations, namely, 0000, 0001, . . . , 9999. Starting with the first value, each successive value is tested to see if it is the correct PIN. This is done by Exclusive-ORing the trial PIN with the compromised KP, encrypting ID under this intermediate value, and

comparing the result for equality with AP. The trial PIN is therefore the actual PIN in question if the computed value of AP is equal to the intercepted value of AP.

If AP is not sent, the system can still be attacked since the intercepted MAC is a function of KP and PIN. In this case a trial KTR is generated by Exclusive-ORing the trial PIN with the compromised KP and decrypting ID under this intermediate value. The trial KTR is accepted as valid if the MAC generated from the intercepted message (using the trial KTR) equals the corresponding intercepted MAC.

A Key on the Magnetic Stripe Card Must be Shared with the Terminal

By definition, a secret user-supplied key must be used to achieve personal verification and authentication of transaction request messages between the cardholder and issuer such that the cardholder and issuer are completely isolated from all other users, programs, and devices in the EFT system. Although a personal key is required to achieve isolation, it is not by itself sufficient. Isolation is achieved only if the secret personal key is not exposed, disclosed, or shared with others. A key written on a magnetic stripe card must always be read into the EFT terminal, since the terminal contains the cryptographic algorithm. Therefore, in an interchange, the process of personal verification is not isolated to the cardholder and issuer: the secrecy of KP depends additionally on security measures implemented in the acquiring institution's terminals.

Exposure Due to Misuse of Personal Keys and Fake Personal Keys

Although the personal key can be used to generate MACs on EFT transaction request messages and transaction response messages, these MACs are based on a KP and PIN supplied and known to the cardholder. Thus the cardholder himself is able to launch active attacks by generating valid MACs for arbitrary transaction response messages. This action is referred to as a *misuse of personal key attack.* An opponent who supplies a bogus personal key to the EFT terminal can also forge transactions. This action is referred to as a *fake personal key attack.*

Fraud could be perpetrated against the system by initiating a transaction at an EFT terminal using any (bogus or valid) KP and PIN known to the cardholder. A microprocessor previously placed in the communication line between the EFT terminal to its host could be programmed to intercept and prevent all opponent-initiated transactions from reaching the issuer. The microprocessor would then generate a fraudulent response message and valid MAC and send them to the EFT terminal. The EFT terminal would respond as though it were in communication with the issuer, when in fact it would be in communication with the opponent's microprocessor.

The attack illustrates why the process of authenticating transaction response messages must be based on secret information (a secret key) known only to the issuer or to the issuer and originating terminal, but not to the cardholder.

No Interlocking with KP

When a secret terminal key is employed, the suggested protection against bugging and probing is to interlock the terminal key so that any penetration of the device causes the key to be erased and the device to become inoperative. In the personal key approach, there is no secret key to interlock. In that case, a defense against penetration of the terminal must be provided by other methods, e.g., using an integrated alarm system. In addition to increasing cost, alarms may malfunction or become inoperative or be intentionally bypassed during an attack. Moreover, an indication of the alarm's ineffectiveness is not necessarily obvious.

Personal Key Approach with an Intelligent Secure Card

Although several objections have been raised with regard to the personal key, most of these are objections to storing the key on the magnetic stripe card. A valid objection to the personal key, however, is that *authentication of transaction response messages must not be based on a key known to the cardholder (legitimate user or opponent).* Otherwise, the system is exposed to active fraud threats wherein valid MACs would be generated on fraudulent transaction response messages.

An Ideal Intelligent Secure Card

The objection to the "personal key approach," stated above, could be largely overcome by employing an intelligent secure card with the following properties:

1. Secret information stored on the card cannot be probed or read.
2. It is not possible to manufacture counterfeit cards.
3. It is not possible to write a bogus key on a genuine card.

An intelligent secure card with the above properties is defined here as an *ideal intelligent secure card* ("ideal" mainly because the properties of the card are unattainable with present technology).

The intelligent secure card must have some identifying property or feature that could be checked at the time of its use to distinguish it from a bogus card. Otherwise, an opponent may be able to manufacture inexpensive bogus cards which do not have the properties of the intelligent secure cards, but nevertheless satisfy the interface requirements of the entry point.

With an ideal intelligent secure card, all cryptographic operations would be performed on the card using KP. KP would be used to encrypt and protect the PIN and to generate MACs on the transaction request and transaction response messages.

Loss and theft of cards and copying card information present no threat to the secrecy of KP since secret information stored on the card cannot be ascertained due to property 1. The fake personal key attack is blocked since

		System Nodes			
System User	Ideal Intelligent Secure Bank Card	EFT Terminal	Acquirer's Host (Inst. X)	Switch's Host	Issuer's Host (Inst. Y)
		Permanently Installed Keys			
none	KP	none	none	none	KMHiss
		Keys Used for MAC Generation on the Transaction Request Message, Mreq			
none	KTRcard,iss (dynamically generated from KP and PIN)	none	none	none	KTRcard,iss for each member of institution Y (dynamically generated or stored)
		Keys Used for MAC Generation on the Transaction Response Message, Mresp			
none	KTRcard,iss (dynamically generated from KP and PIN)	none	none	none	KTRcard,iss for each member of institution Y (dynamically generated or stored)

Note: Keys associated with personal verification at the issuer (and perhaps the switch) are not shown.

Legend:

KMH: Host master key
KTR: Transaction key (used for message authentication, e.g., KTR1card,iss = $D_{KP \oplus PIN}(ID)$)
KP: Personal Key

Table 11-6. Keys Defined for the PIN/Personal Key Approach Using an Intelligent Secure Card

manufacturing or changing a bank card is considered not possible due to properties 2 and 3. Cardholders would not be given their personal keys (only the issuer would know the KPs), thus blocking misuse of personal keys by legitimate users. In effect, the issuer would determine all card information, including the user's personal key. Cardholders would be prevented from obtaining their KPs since they cannot read the information due to property 1.

Furthermore, because all cryptographic operations would be performed on the card, it is unnecessary to transmit KP to the entry point. Thus, KP would not be exposed at the terminal. The ideal intelligent secure card would also eliminate the need for storing a key in the terminal. Hence, an extremely simple key management could be implemented. No additional system keys would be needed with the possible exception of an authentication key at the issuer if personal authentication codes are used.

There is, however, one remaining exposure with this approach. After all checking has been done on the bank card, the terminal must be informed of the outcome (positive or negative). But since there are no keys stored in the terminal, data communications between the card and terminal cannot be authenticated. Therefore the integrity of this communication path must be assured independently. Otherwise, a negative response could be changed to a positive response (again allowing fraud to be committed).

A summary of the keys required with the PIN/personal key approach is provided in Table 11-6. A description of the PIN/personal key approach when used in conjunction with an intelligent secure card is summarized in Tables 11-6 and 11-7. The tables show the flows of information from the card to the issuer (via the EFT terminal) and from the issuer to the EFT terminal (via the card), and include a description of the keys and MACs used in the message authentication process.

Comparing Tables 11-6, 11-7, and 11-8 with those of the PIN/system key approach (Tables 11-3, 11-4, and 11-5), one observes that a simpler key management is achieved with the PIN/personal key approach.

The personal key approach provides adequate EFT security if the requirements of an ideal intelligent secure card are met. However, the approach has one major drawback; it is unlikely that an attractively priced card meeting these requirements can be produced with current technology. Despite this drawback, the intelligent secure card does offer the potential for improved EFT security if used in conjunction with personal and system keys, as discussed in the section The PIN/Personal Key/System Key (Hybrid Key Management) Approach Using an Intelligent Secure Card.

A Practical Intelligent Secure Card

The requirements for an ideal intelligent secure card are unattainable for the following reasons. First, it is unlikely that probing for card information can be prevented. With enough time and resources, information on the card could be recovered. Second, since institutions must be able to arrange for the manufacture of cards, an opponent must be assumed to have the same opportunity.

A more realistic objective would be to make it prohibitively expensive for an opponent to obtain only a few cards, by forcing him to assume the total cost and burden of becoming a manufacturer. Since the opponent would

		System Nodes			
System User	Ideal Intelligent Secure Bank Card	EFT Terminal	Acquirer's Host	Switch's Host	Issuer's Host
1 Enter PIN and transfer to card via terminal.	2 Generate Tcard and transfer to terminal.	3 Read card information and formulate Mreq which includes TODacq and Tcard.	8 Forward received Mreq and MAC1card,iss to switch.	9 Forward received Mreq and MAC1card,iss to issuer.	10 Check received MAC1card,iss with KTRcard,iss of reference and TODiss of reference.
	5 Compute MAC1card,iss with KTRcard,iss.	4 Send Mreq to intelligent secure card.			11 Verify user.
	6 Send Mreq and MAC1card,iss to terminal.	7 Forward received Mreq and MAC1card,iss to acquirer.			12 Decide if Mreq is to be honored. 13 Formulate Mresp which includes Tcard.

Note: It is assumed that the acquirer periodically sends time-of-day information (TODacq,term) to the terminals in its domain. The card also generates time-variant information (Tcard) which is transmitted to the issuer. The TOD stored at the other network host nodes (TODsw at the switch and TODiss at the issuer) is assumed to be equal to TODacq within an allowable range (ΔTOD).

The integers 1–13 in the table show the sequence of steps in the transaction.

Table 11-7. Information Flow from Card to Issuer—PIN/Personal Key Approach with Intelligent Secure Card

		System Nodes			
System User	Ideal Intelligent Secure Bank Card	EFT Terminal	Acquirer's Host	Switch's Host	Issuer's Host
	19 Check received MAC2iss,card with KTR of reference and Tcard of reference. 20 Decide if Mresp should be accepted or rejected. 21 Notify terminal to process transaction if Mresp is accepted; otherwise notify terminal to abort transaction request*.	18 Forward received Mresp and MAC2iss,card to card. 22 Initiate action based on decision made by intelligent secure card.	17 Forward received Mresp and MAC2iss,card to terminal.	16 Forward received Mresp and MAC2iss,card to acquirer.	13 Formulate Mresp which includes Tcard. 14 Generate MAC2iss,card using KTR. 15 Send Mresp and MAC2iss,card to intelligent secure card via switch, acquirer, and terminal.

*This response to the terminal cannot be authenticated since there is no terminal resident key.

The integers 13–21 in the table show the sequence of steps in the transaction.

Table 11-8. Information Flow from Issuer to Terminal–PIN/Personal Key Approach with Intelligent Secure Card

normally need only a few cards, the cost per card would be very high. It seems much more reasonable that an intelligent secure card could be manufactured (designed and mass-produced) with the property that secret information stored on the card could not be read, skimmed, or copied during periods when the card is used (and exposed) routinely to transact business. It is assumed that secret information stored on the card would be secure against reading, skimming, or copying even if the card is unwittingly entered into a fake or modified terminal under the control of an opponent, or if, as a part of the procedure for transacting business, the cardholder gives his card to a dishonest merchant (or employee of the merchant) who, in turn, surreptitiously enters the card into a special reading device hidden from view. An intelligent secure card with these properties is defined here as a *practical intelligent secure card,* or *intelligent secure card,* for short.

To summarize, the following properties are assumed for the intelligent secure card.

1. Secret information stored on the card cannot be probed or read by personnel or equipment handling the card during routine business transactions. Sophisticated techniques and expensive equipment would be required to probe or write secret card information, although it is assumed that this could be accomplished in a laboratory environment.
2. It is very expensive to manufacture counterfeit cards on a small scale.
3. It is very expensive to write a bogus key on a genuine card.

The intelligent secure card, as assumed here therefore, only defends against attacks of short duration that do not injure or destroy the card or the secret information stored thereon. It is assumed that a destroyed, injured, or nonfunctional card would be promptly reported to the issuing institution, and that the issuing institution would invalidate the corresponding account and either reissue a new card or reinitialize the existing card with a new key (as appropriate). Likewise, it is assumed that lost and stolen cards would be promptly reported to the issuing institution and that a similar action would be taken by the issuing institution to invalidate the accounts and reissue new cards to the affected cardholders.

However, since one must assume that an opponent could manufacture bogus cards and write bogus keys on them, the PIN/Personal key approach is still exposed to a fake personal key attack. Since one must assume also that a legitimate user can determine his KP if he is willing to overcome the obstacles identified in item 1 above, the exposure to misuse of KPs also exists. For these reasons, the intelligent secure card does not overcome the basic objection to the personal key approach stated previously; namely, authentication of transaction response messages must not be based on a key known to the cardholder (legitimate user or opponent).

This objection (to the personal key approach) can be overcome by basing authentication of transaction response messages on a secret terminal key in addition to a personal key (i.e., the key management employs both personal and system keys). Such hybrid key management used together with an intel-

ligent secure card offers the potential for increased security in future EFT applications.

THE PIN/PERSONAL KEY/SYSTEM KEY (HYBRID KEY MANAGEMENT) APPROACH USING AN INTELLIGENT SECURE CARD

Discussed here is a system which provides a higher level of security than either the PIN/system key or the PIN/personal key approach. From a security point of view, it is thus a preferred solution. The approach combines the features of an intelligent secure card (see the section Personal Key Approach with an Intelligent Secure Card) with that of hybrid key management based on both system keys and personal keys.[30] For reasons of completeness, some of the ideas and terms discussed above are repeated here.

Hybrid key management used together with the intelligent secure card solves the following problems individually associated with the PIN/system key and PIN/personal key approaches, respectively:

1. The PIN/system key approach does not provide isolation of institutions as far as personal verification and message authentication are concerned, although it is an acceptable solution.
2. The PIN/personal key approach in combination with an intelligent secure card, although it provides a higher degree of isolation for personal verification than does the PIN/system key approach, is subject to misuse of KPs and fake KPs. It is, by itself, an unacceptable solution.

With a combination of both approaches (i.e., PIN/system key and PIN/personal key), personal verification can be isolated among institutions and the threats of misused and fake KPs are greatly reduced. As shown below, an attack will succeed only if system keys are subverted and personal keys are manipulated at the same time. In addition, the end-to-end message authentication procedure based on KP and PIN is combined with personal verification eliminating the need for generation of a separate authentication parameter for personal verification.

Although the hybrid approach combines two key management schemes (system and personal key), it is actually less complex than the PIN/system key approach. Rerouting attacks are of no concern because of the personal key, which eliminates some of the functions needed in the PIN/system key approach. It is also more robust since security exposures occurring at intermediate nodes have only a limited effect on overall security.

[30] An increase in security over the PIN/system key approach can be achieved by coupling a hybrid key management with a magnetic stripe card. Storing a personal key on the magnetic stripe card allows a migration path to a hybrid key management approach coupled with an intelligent secure card. The details of such an approach are omitted, although the reader should have no difficulty in adapting the hybrid key management approach described here to work with a magnetic stripe card.

Description of a Hybrid Key Management Approach

The system discussed here is composed of host processing centers (HPCs) and EFT terminals, interconnected in an EFT network supporting interchange.[31] Each network node has a DES cryptographic capability either integrated into the node or contained in a separate dedicated device called a *security module*[32] attached to the node via a secure, local cable. Each security module has a set of cryptographic operations that may be invoked by the supporting device or HPC via a defined interface. The cryptographic operations perform data encryption and decryption and key translations necessary to the management of EFT transactions. No clear cryptographic keys ever exist outside the security module, except during periods when they are initially generated or entered into the system.

Keys stored in a security module are protected by implementing adequate physical security measures and/or providing a set of interlocks that will erase all secret information if penetration of the security module or containing device is detected.

It is assumed (as in the discussion of the PIN/personal key approach, Table 11-6), that a transaction key (a dynamically created key used solely for authentication, denoted by KTR1), is used to generate the MAC on the transaction request message (Mreq). KTR1 is a one-way function of the PIN, the personal key, and the user identifier, so that each user is assigned a different value of KTR1. (Other variations are possible in which different KTR1 keys are generated for the same user on successive transactions, but are omitted from the discussion.) End-to-end authentication is made possible by storing a copy of each user's KTR1 at the issuer. At the entry point, KTR1 is dynamically created from user-supplied information. Since the MAC depends only on secret user-supplied information and other nonsecret information, it is by definition an authentication parameter (AP). Thus authentication of the transaction request message and personal verification are integrated into one procedure.

The response message is authenticated on the basis of a time-variant key KSTR generated randomly at the issuer and transmitted to the terminal in the form $E_{Knode}(E_{KTR2}(KSTR))$ (i.e., doubly encrypted under two keys, KTR2 and Knode). KTR2 is defined by the one-way function $KTR2 = D_{KTR1}(Tcard)$, where KTR1 is the same key used to generate the MAC on the transaction request message and Tcard is a nonsecret time-variant quantity generated by the intelligent secure card. Knode is a time-variant system key shared between two logically adjacent network nodes (e.g., the terminal and acquirer, the acquirer and switch, and the switch and issuer). Thus as KSTR is routed from the issuer to the terminal it is encrypted and reencrypted successively under different Knode keys. For example, Knode

[31] For simplicity, the control units shown in Figure 11-1 are omitted.

[32] The security module provides the same function as a cryptographic facility (see The Cryptographic Facility, Chapter 4). The term "security module" is used here to maintain consistency with the discussion in Chapter 10 of the PIN/System key approach.

would equal the terminal master key KMT on the link between the acquirer and terminal.

At the time Tcard is generated, KTR2 is also generated by the intelligent secure card. KTR2 and Tcard are then sent to the terminal where KTR2 is saved for later use in decrypting KSTR and Tcard is forwarded to the issuer in the transaction request message. The issuer generates KTR2 from Tcard (which it receives in Mreq) and KTR1 (which is stored in the issuer's data base). In a sense, KTR2 is nothing more than a time-variant personal key with the property that knowledge of KTR2 does not reveal KP. Note also that KTR2 is not routed through the system (i.e., the terminal and issuer establish KTR2 without involving other system nodes or depending on these nodes to protect the secrecy of KTR2).

At the terminal, the encrypted KSTR arrives in the form $E_{KMTacq,term}E_{KTR2}(KSTR)$. It is decrypted first with KMT and second with KTR2 to recover KSTR. Mresp is authenticated by generating a MAC of reference from Mresp and KSTR and comparing it for equality with the received MAC.[33]

The Reason for Doubly Encrypting KSTR

The explanation of why KSTR is doubly encrypted under KTR2 and Knode rather than being encrypted only under Knode is now provided. If KSTR is encrypted under both KTR2 and Knode, the message authentication process associated with transaction response messages can be subverted (i.e., bogus messages and MACs acceptable to the terminal could be generated) only if both keys are compromised. An opponent must therefore compromise a Knode key, or some other system key that would allow a Knode key to be determined, and learn the value of KP on some intelligent secure card. With knowledge of KP it would be an easy matter to calculate the value of KTR1 for a bogus PIN and ID entered at the terminal. The value of Tcard could also be intercepted from the transmitted transaction request message, which would allow KTR2 to be calculated. With knowledge of KTR2 and Knode, the opponent (using an active wiretap) could then intercept and block the transaction response message and send a fraudulent transaction response message and MAC to the terminal in its place that would be accepted as valid.

The suggested method of doubly encrypting KSTR under KTR2 and Knode provides additional security (over a method of encrypting KSTR under Knode) only if the cost or work factor to read or write a key on a card or manufacture bogus cards is significant. If the card's defenses can be overcome easily in a laboratory at a low or moderate cost, encryption of KSTR under KTR2 does not add significantly to the security of KSTR. In such a case, the protocol could be modified so that Tcard and KTR2 are not generated by the intelligent secure card and Tcard is eliminated from the transaction request message. At the issuer, KSTR is encrypted only under Knode. Otherwise, the protocol is the same.

[33] There is no need to protect the encrypted value of KSTR with a MAC since, if changed, KSTR will not decrypt correctly and an incorrect MAC of reference will be generated.

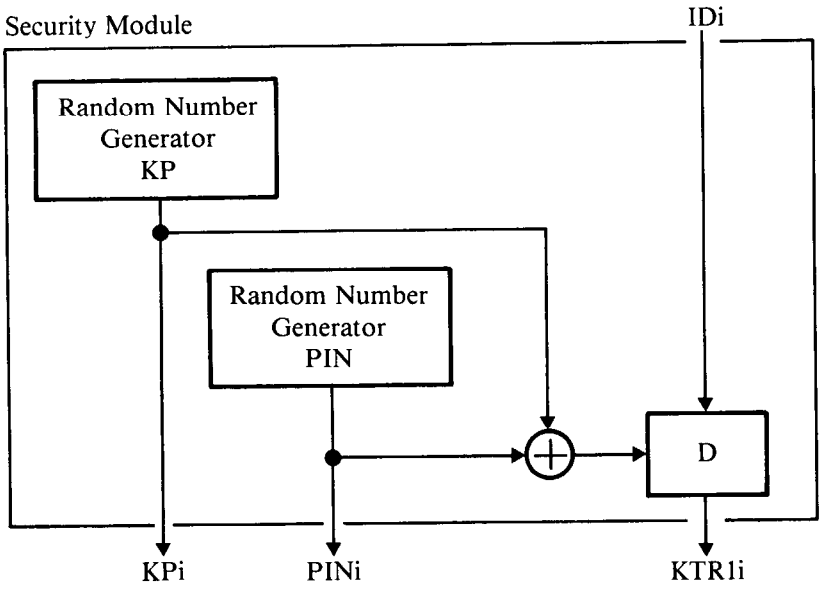


Figure 11-24. Generation of KP, PIN, and KTR1 in Issuer's Security Module - Initialization Process

PIN and KP Selection

While there are several ways in which PINs and personal keys could be selected, it will be assumed that they are produced by the issuer using the security module as a generator of pseudo-random numbers. The generated PIN is printed on a PIN mailer and the corresponding KP is written on the bank card. The PIN mailer and bank card are then forwarded to the designated customer. KP and PIN are also Exclusive-ORed to form the intermediate value KP ⊕ PIN, which is then used as a key to decipher ID and generate a transaction key, KTR1 = $D_{KP \oplus PIN}(ID)$.[34] The generation of KP, PIN, and KTR1 is shown in Figure 11-24. KTR1 is stored in the data base of the issuer's HPC encrypted under a variant of the host master key.

PIN and KP Validation

Each time the customer initiates a transaction at an EFT terminal, the entered PIN is transferred to the bank card where it is Exclusive-ORed with KP to form the intermediate value KP ⊕ PIN (Figure 11-25), which in turn is used with ID to generate KTR1. KTR1 is then used to generate a MAC on the transaction request message. A time-of-day clock value (time stamp) is included in the message to ensure that it is time-variant. (A separate authentication parameter is not transmitted in the message since personal verification is combined with message authentication by basing the MAC on KP and PIN.)

At the issuer, the message is validated using a corresponding KTR1 of

[34] For an explanation of the advantage of making KTR1 a one-way function of KP, PIN, and ID, see footnote 27.

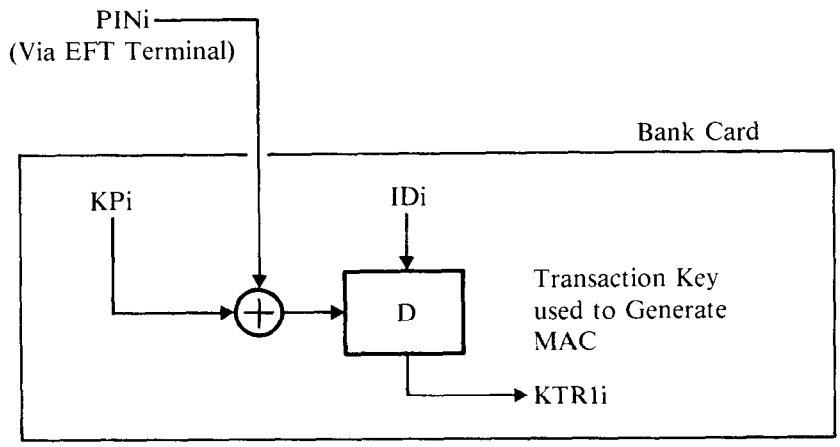


Figure 11-25. Regeneration of the Transaction Key on the Bank Card - Part of the Verification Process

reference which is filed under the user's identifier (ID) in the data base of the issuer's HPC.[35] The KTR1 of reference and the received message are then used to generate a MAC of reference. The MAC of reference and the received MAC are compared for equality. If they are equal, the issuer concludes that the content of the message is correct and that the secret information (KP and PIN) used in the computation of the MAC is properly related to the claimed ID thus verifying the user at the same time. By verifying that the time stamp in the received message correlates properly with the corresponding reference maintained by the issuer, the issuer can determine that the received message is not a stale message. More details are provided below in the sections A Hybrid Key Management Approach for Noninterchange, and A Hybrid Key Management Approach for Interchange.

System Key Generation

The keys for an institution's terminals are produced by that institution's security module, using the module as a generator of pseudo-random numbers. The keys are transferred from the module to a secure device (e.g., a printer) and they are then transported and installed in the appropriate EFT terminals. Each terminal master key is also encrypted under a variant of the host master key (derived in the issuer's security module) and then stored in the data base of the issuer's HPC (filed under the terminal identifier, TID). It is assumed that each terminal in the network has a unique identifier TID, and that the TID is included in each transaction request message.

Key Management Considerations for the Hybrid Approach

The major objective of the PIN/personal key/system key approach is to use externally supplied keys as well as system keys (but only to the extent that

[35] It is assumed that the integrity of the data base can be guaranteed by the issuer's HPC. Various cryptographic techniques can be used to achieve data base integrity (see Authentication of Time-Invariant Data, Chapter 8).

they are of maximum use). Since it is secure to base the required reference at the issuer for MAC checking of the transaction request message entirely on KP and PIN (as discussed above for the PIN/personal key approach), this approach is also used in a hybrid key management scheme. As a consequence, the personal verification process is separated among institutions. On the other hand, since KP alone does not provide a secure method for MAC checking of the transaction response message, the required reference at the entry point is also based, in addition to KP, on a system key. This requires the presence of a terminal resident key (e.g., a terminal master key). In addition, keys used for MAC checking at intermediate nodes must be shared among institutions and thus the message authentication process cannot be isolated to each individual institution. However, greater separation can be achieved by coupling message authentication of transaction response messages to PIN and KP. This is achieved in the implementation discussed here by introducing the end-to-end transaction session key, KSTR, which is routed from the issuer to the terminal encrypted under both KTR2 = D_{KTR1}(Tcard) and Knode. (A still higher degree of separation can be achieved if digital signatures are used as discussed in the section below, Security Enhancements with Digital Signatures.)

In summary, KTR1 is used to generate the MAC on the transaction request message, KSTR is used to generate the MAC on the transaction response message, and KTR2 is one of the keys used to encrypt KSTR for transmission from the issuer to the entry point.

In an interchange environment, the issuer has no knowledge of the acquirer's terminal master keys. In that case, an interchange key (KI), in addition to KTR2, is used to encrypt and forward transaction session keys from one institution to another, e.g., from one institution to a switch, from the switch to an institution, or from one institution to another institution. It is assumed here that interchange keys are generated on a bilateral basis, so that two institutions (or an institution and its EFT switch) can share a common interchange key. An Interchange key is just one special form of node key (Knode). A terminal master key is also a node key.

The MAC generated on the transaction response message using KSTR has the advantage that it can be sent to the originating EFT terminal without undergoing any cryptographic transformation. Only the key (Knode) which protects E_{KTR2}(KSTR) changes as the transaction response message traverses the network.[36]

Hybrid Key Management Approach for Noninterchange

Each time a customer initiates a transaction at an EFT terminal, the customer's card is inserted into the terminal, or suitable read/write device attached to the terminal, and the customer enters his PIN via a suitable entry device (PIN pad or keyboard). The PIN is routed to the card where it is Exclusive-ORed with KP thus generating KTR1 in the manner described

[36] As discussed above, $E_{Knode}E_{KTR2}$(KSTR) is routed back to the terminal which has knowledge of KTR2. Encrypting with Knode, where Knode changes from node to node, assures that attacks which manipulate KP will not succeed.

above (Figure 11-25). The card also generates a random number, Tcard, which is decrypted under KTR1 to produce KTR2. KTR2 and Tcard are then routed to the terminal where KTR2 is stored for later use in authenticating the transaction response message.

Based on the customer's request, the terminal formats a transaction request message, which consists of a time stamp (TOD, time-of-day), time-variant information generated by the terminal (Tterm, a message sequence number), time-variant information generated by the card (Tcard), the user ID, the terminal ID, the transaction type, and the transaction data (Figure 11-26). The time stamp is obtained from the acquirer (which is also the issuer since a local transaction is described),[37] at the request of the EFT terminal. A different request could be made for each customer-initiated transaction. Another possibility is that the acquirer periodically sends time information to the terminal.

The purpose of the time-stamp is to provide the terminal with time-variant data that can be used in the preparation of the transaction request message. This permits the issuer (which, in this case, is the acquirer) to detect stale transaction request messages that may be injected into the communication path. It is assumed that the issuer's HPC maintains a time-of-day clock that can be read to obtain a time stamp.

The purpose of the message sequence number, generated by the EFT terminal, is to provide the issuer with time-variant data that can be used in the preparation of the transaction response message. This permits the EFT terminal to detect stale transaction response messages injected into the communication path. (Without authentication of the transaction response message, a replay attack is possible.) The message sequence number is incremented on each new transaction request message, independent of the customer initiating the transaction.

The purpose of the time-variant information generated by the intelligent secure card is to allow the terminal and issuer to establish a time-variant key, KTR2 = D_{KTR1}(Tcard), without involving other system nodes or relying on these nodes to protect the secrecy of KTR2. In this manner the terminal

Time Stamp (TOD)	Time-Variant Information Controlled By Card (Tcard)	Time-Variant Information Controlled By Terminal (Tterm)	User Identifier (ID)	Terminal Identifier (TID)	Transaction Type	Transaction Data

Figure 11-26. Transaction Request Message Formatted at the EFT Terminal

[37] To eliminate the need for the terminal (acting on behalf of the user) to request a time stamp from the issuer, the issuer could also record a current "time reference" for the user in his HPC's data base. For example, a message sequence number obtained from a one-up counter on the bank card could be used in place of the time stamp.

and issuer share a key that allows end-to-end authentication of Mresp without jeopardizing the security of KTR1, KP, or PIN.

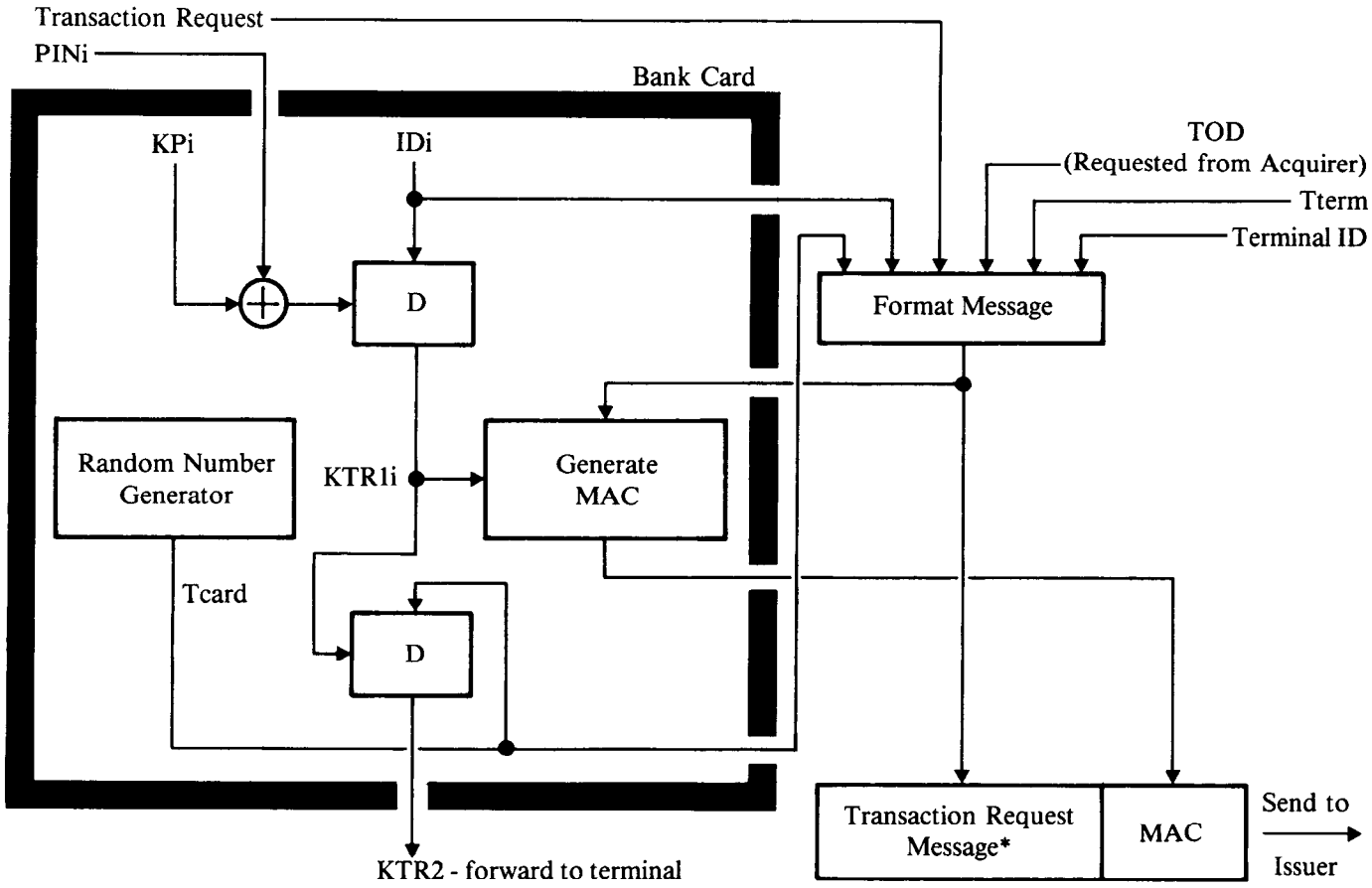
Once the transaction request message has been formatted, it is transferred to the bank card where a MAC is generated using KTR1. The MAC is then returned to the terminal. Generation of a transaction request message and corresponding MAC is shown in Figure 11-27. The message and MAC are then sent to the issuer.

As mentioned earlier, the issuer validates the message using a corresponding KTR1 of reference which is filed under the user's ID in the data base of the issuer's HPC. The KTR1 of reference and the received message are then used to generate a MAC of reference, and the generated MAC of reference is compared for equality with the received MAC. The time stamp is also checked for currency. If both MACs are equal and the time stamp is within the prescribed bounds, the message and user are validated. Message authentication at the issuer's HPC is shown in Figure 11-28.

If the received MAC and time stamp are valid and the transaction request can be honored, the issuer's HPC formats and sends a positive transaction response message to the originating terminal. Otherwise, if any one of the conditions is not met, a negative transaction response message is sent to the originating terminal. Only the positive response is important to the discussion, since a negative response could be defined as any response other than a positive response, e.g., a random bit pattern.

A positive transaction response message consists of a doubly encrypted transaction session key $E_{KMT}E_{KTR2}(KSTR)$ and a MAC generated on the transaction request message using KSTR. The procedure for generating a positive transaction response message (Figure 11-29) is as follows: Using the received TID, the HPC identifies the corresponding encrypted terminal master key in its data base and passes this to the security module along with the received transaction request message. The security module first generates a random number, defined as a transaction session key (KSTR), which it then uses to generate a MAC on the transaction request message. Next, KTR1 is recovered from the issuer's data base and used together with the received value of Tcard to generate $KTR2 = D_{KTR1}(Tcard)$. Finally, the encrypted terminal master key is read from the issuer's data base, decrypted, and used with KTR2 to doubly encrypt KSTR. The doubly encrypted transaction session key, $E_{KMT}E_{KTR2}(KSTR)$, and MAC are returned to the issuer's HPC whereupon they are sent to the originating terminal.

At the terminal, the procedure for validating the transaction response message (Figure 11-30) is as follows. The doubly encrypted transaction session key, $E_{KMT}E_{KTR2}(KSTR)$, is decryted under the terminal master key (KMT) resident in the terminal. Next $E_{KTR2}(KSTR)$ is decrypted under the value of KTR2 forwarded previously from the card to the terminal. KSTR is used to generate a MAC of Reference on the original transaction request message which is assumed to have been saved in the terminal. The MAC of reference is then compared for equality with the received MAC. If the two MACs are equal, the EFT terminal honors the requested transaction; otherwise, the EFT terminal informs the customer that the requested transaction has been



*A copy of the transaction request message is retained by the EFT terminal.

Figure 11-27. Generation of the Transaction Request Message and MAC at the EFT Terminal

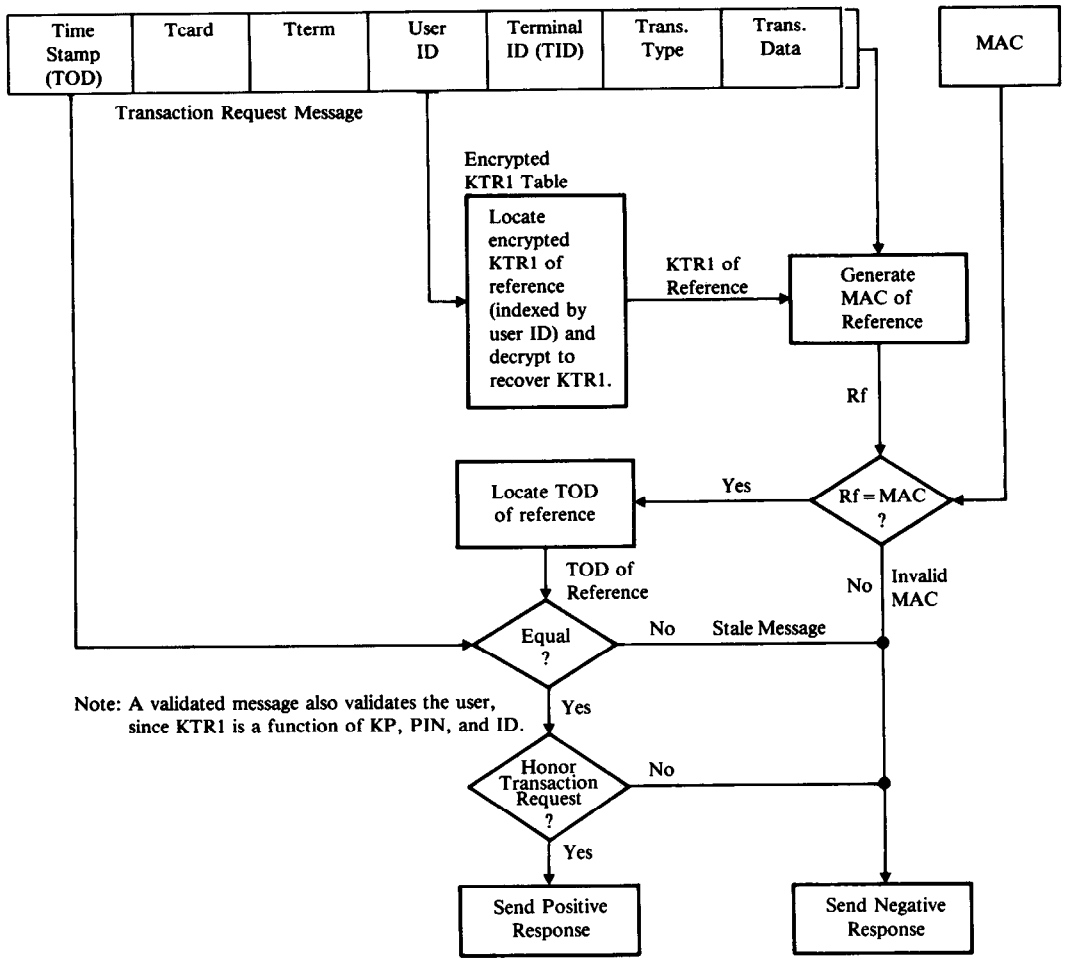


Figure 11-28. Message Authentication at the Issuer's EDP System

disallowed. In any case, the EFT terminal sends a message to the issuer, informing the issuer of the final disposition of the requested transaction. A MAC for this final message can be generated securely based on KSTR.

Hybrid Key Management Approach For Interchange

An interchange transaction originates in the same manner as does a local transaction. The message and MAC based on KTR1 are sent to the issuer via the acquirer and switch. It is assumed that the proper message routing can be determined from information contained in the message.[38] (Current bank card standards call for an institution code to be the first several digits of the personal account number [16].)

[38] Once the destination has been determined, the network routing information will appear in the message's header.

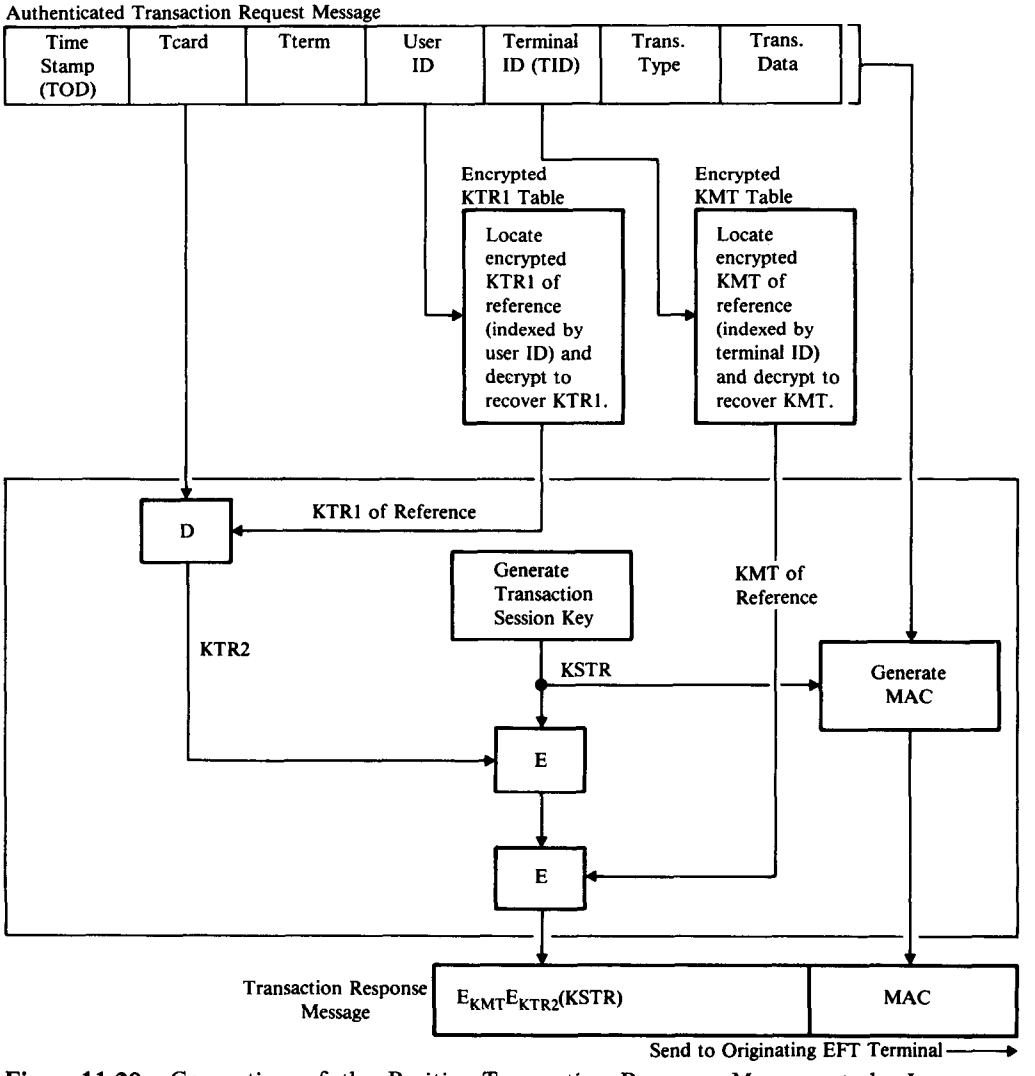


Figure 11-29. Generation of the Positive Transaction Response Message at the Issuer

The issuer validates the message using the KTR1 of reference stored in its data base, as previously described. The time stamp is also checked for currency (i.e., that the clock readings do not vary by more than some fixed limit). In checking for timeliness, it is assumed that the clocks of all institutions are synchronized and do not vary by more than small amounts. Where transactions cross time zones, either automatic adjustments are assumed to be present or a common fixed time is used (e.g., Universal Mean Time).

If the received MAC and time stamp are valid and the transaction request can be honored, the issuer's HPC formats and sends a positive transaction response message to the originating terminal. Otherwise, a negative transaction response message (not discussed here) is sent to the originating terminal.

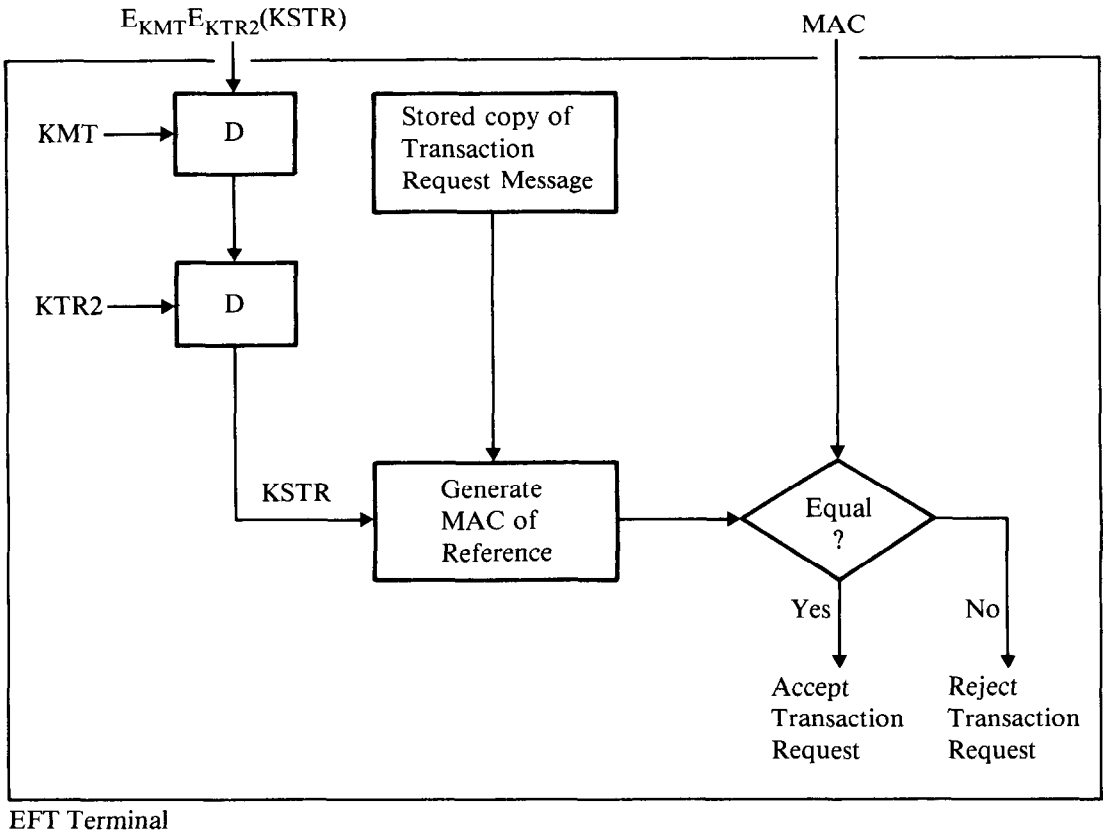


Figure 11-30. Message Authentication at the EFT Terminal

A positive transaction response message consists of a doubly encrypted transaction session key (KSTR) and a MAC generated on the transaction request message using KSTR. The procedure for generating a positive transaction response message is the same as in the case of a local transaction, except that E_{KTR2}(KSTR) is encrypted under a key specified by the appropriate node (an interchange key, KI) rather than a terminal master key. In that case, $E_{KIiss,sw}E_{KTR2}$(KSTR) and MAC are produced and sent to the switch. At the switch, $E_{KIiss,sw}E_{KTR2}$(KSTR) is translated from encipherment under KIiss,sw (the issuer's interchange key) to encipherment under KIsw,acq (the acquirer's interchange key). $E_{KIsw,acq}E_{KTR2}$(KTSR) and MAC are then sent to the acquirer, where $E_{KIsw,acq}E_{KTR2}$(KSTR) is again translated from encipherment under KIsw,acq (the acquirers interchange key) to encipherment under KMTacq,term (the originating terminal's master key).[39] $E_{KMTacq,term}E_{KTR2}$(KSTR) and MAC are then sent to the originating terminal. At the terminal, the procedure for validating the transaction response message is the same as in the case of a local transaction.

[39] To thwart message misrouting attacks, the terminal could use KTR2 to generate a MAC on the routing information contained in the message's header.

The general ideas as to the keys to be used as well as how data flows to and from the issuer are shown in Tables 11-9, 11-10, and 11-11.

Cryptographic Considerations for an Intelligent Secure Card

Use of an intelligent secure card does require some additional consideration in the design of secure procedures for personal verification and message authentication. Computation of AP on the bank card poses a problem that is not present when AP is computed inside the EFT terminal.

An authentication parameter computed inside an EFT terminal may or may not involve time-variant information, but an authentication parameter computed on the card *must always involve time-variant information.* The following example explains the reason. A transformation applied to secret user-supplied information inside a terminal (such as a one-way function or encryption under a secret terminal key) prevents an intercepted output from being reentered (in its original form) as input to the terminal. Of course, the issuer must trust that the terminal (acting on its behalf) will perform this transformation with integrity. When an intelligent secure card is used, a similar transformation performed on the card does not achieve the same end. In this case, the issuer cannot trust the cardholder, since the cardholder may be an opponent intent on committing fraud against the system. The issuer also has no way to ensure that the card computes the authentication parameter. A bogus card could store an intercepted value and read it out of storage and pass it to the terminal whenever prompted. To prevent this, the issuer must require the authentication parameter to be a function of time-variant information. This forces the intelligent secure card to compute AP dynamically, and thus prevents the described attack. In the implementation discussed above, this time variance is achieved with time of day (TOD) information.

Security Enhancements with Digital Signatures

If institutions do not wish to share secret keys for purposes of authenticating response messages, digital signatures based on either a conventional or public-key algorithm could be used (see Digital Signatures, Chapter 9). A public-key algorithm, however, is more practical, since there are no restrictions on the number of signed messages that can be sent and received using a single pair of keys. With a conventional algorithm, each digital signature must be validated using a separate validation parameter (or pattern of bits).

Consider a digital signature procedure for authenticating transaction response messages which is based on a public key algorithm. Each institution would have a public and private key, PK and SK, respectively. The public key is shared with each other institution and published in a major newspaper to allow each public key to be independently validated.

In such an approach, the issuer transforms a received transaction request using his secret key and sends the response to the originating acquirer. The acquirer recovers the transaction request using the issuer's public key (which could be stored in the HPC's data base). Since the issuer's secret key is not

System User	Intelligent Secure Bank Card	System Nodes: EFT Terminal	Acquirer's Host (Inst. X)	Switch's Host	Issuer's Host (Inst. Y)
		Permanently Installed Master Keys			
none	KP	KMT	KMHacq	KMHsw	KMHiss
		Permanently Installed Interchange Keys			
none	none	none	KIacq,sw	KIacq,sw KIsw,iss	KIsw,iss
		Keys Used for MAC Generation for Transaction Request Message, Mreq			
none	KTR1card,iss (dynamically generated from KP and PIN)	none	none	none	KTR1card,iss for all members of institution Y

Keys Used to Protect the Transaction Session Key Used for MAC Generation of the Transaction Response Message, Mresp					
none	KSTR received from issuer; KTR2card,iss dynamically generated from KTR1card,iss and Tcard, i.e., D_{KTR1}(Tcard)	KMTacq,term	KMTacq,term KIsw,acq	KIiss,sw KIsw,acq	KIiss,sw; KSTR randomly generated by issuer; KTR2card,iss regenerated based on Tcard and KTR1card,iss

Note: Keys associated with personal verification at the issuer (and perhaps the switch) are not shown.

Legend:

KMT: Terminal master key
KMH: Host master key
KTR1: Message Authentication Key for Request Messages (Mreq) (e.g., KTR1 = $D_{KP \oplus PIN}$(ID))
KTR2: Message Authentication Key for Response Messages (Mresp)
KP: Personal Key
KI: Interchange key

Table 11-9. Keys Referenced in the Hybrid Approach

		System Nodes			
System User	Intelligent Secure Bank Card	EFT Terminal	Acquirer's Host	Switch's Host	Issuer's Host
1 Enter PIN and transfer to card via terminal.	2 Generate Tcard and transfer to terminal.	3 Read card information and formulate Mreq which includes Tterm, Tcard, and TODacq,term.	10 Forward received Mreq and MAC1card,iss to switch.	11 Forward received Mreq and MAC1card,iss to issuer.	12 Check received MAC1card,iss with KTR1card,iss of reference and TODiss of reference.
	5 Compute MAC1card,iss with KTR1card,iss.	4 Send Mreq to intelligent secure card.			13 Verify user.

	8	14
6	Store Mreq and KTR2	Decide if Mreq is to be
Generate KTR2	in terminal.	honored.
(randomly based		
on Tcard, i.e.,	9	
KTR2 =	Forward received Mreq,	
D_{KTR1}(Tcard)).	MAC1card,iss to acquirer.	
7		
Send Mreq,		
MAC1card,iss		
and KTR2 to		
terminal.		

Note: It is assumed that the acquirer periodically sends time-of-day information (TODacq) to the terminals in its domain. The terminal, on the other hand, generates random information (Tterm) and sends it to the issuer. This can be done as part of the initiation protocol (Figure 11-15). The card also generates time-variant information (Tcard) which is transmitted to the issuer. The TOD stored at the other network host nodes (TODsw at the switch and TODiss at the issuer) is assumed to be equal to TODacq within an allowable range (ΔTOD). The integers 1–14 in the table show the sequence of steps in the transaction.

Table 11-10. Information flow from Terminal to Issuer–Hybrid Approach

System User	Intelligent Secure Bank Card	System Nodes: EFT Terminal	Acquirer's Host	Switch's Host	Issuer's Host
		25 Decrypt $E_{KMTacq,term}(Q)$ with KMTacq,term to obtain $Q = E_{KTR2}(KSTR)$.	23 Translate received $E_{KIsw,acq}(Q)$ to $E_{KMTacq,term}(Q)$.	21 Translate received $E_{KIss,sw}(Q)$ to $E_{KIsw,acq}(Q)$.	14 Decide if Mreq is to be honored. 15 Generate KSTR randomly.
		26 Decrypt Q with KTR2 of reference to obtain KSTR of reference.	24 Send MAC2iss,card and $E_{KMTacq,term}(Q)$ to terminal.	22 Send MAC2iss,acq and $E_{KIsw,acq}(Q)$ to acquirer.	16 Compute MAC2iss,term on received Mreq with KSTR.
		27 Generate MAC2iss,term of reference on stored Mresp with KSTR of reference.			17 Generate KTR2 using KTR1card,iss of reference and received Tcard.

	28 Process transaction if MAC2iss,term of reference equals the received MAC2iss,term; otherwise, abort transaction.	
		18 Encrypt KSTR with KTR2 to obtain Q = E_{KTR2}(KSTR).
		19 Encrypt Q with KIiss.
		20 Send MAC2iss,card and $E_{KIiss,sw}$(Q) to switch.

Note: A check for the correct destination is not necessary since KSTR can only be correctly recovered at the node with the proper KTR1.

The integers 14–28 in the table show the sequence of steps in the transaction.

Table 11-11. Information Flow from Issuer to Terminal—Hybrid Approach

shared with the acquirer, an improvement in security is obtained. Furthermore, the message authentication process is now isolated among institutions. Thus requirement 17 is satisfied.

A MAC based on an acquirer-generated time-variant key (KSTR) is next produced and the MAC and KSTR encrypted under the terminal master key, KMT, is forwarded to the originating terminal. The terminal generates a MAC of reference using the stored KMT, KSTR, and the message of reference, and compares the result for equality with the received MAC. If the two MACs were equal, the terminal honors the request; otherwise not.

In effect, a transaction request transformed under the issuer's secret key provides the acquirer with the equivalent of a signed message authorizing the transaction. The acquirer logs all such signed messages until after they have been cleared via normal accounting methods between respective institutions.

With such an approach, the acquirer has proof of authorization from the issuer, and the issuer need not fear that unwarranted and unprovable claims will be brought against him from other acquirers. The terminal responds only to the orders given it by its owning institution (the acquirer), whereas the acquirer directs the terminal to honor a transaction request only after receiving a signed transaction response message from the issuer.

Advantages

The PIN/personal key/system key approach using an intelligent secure card satisfies the EFT security requirement to a much higher degree than either the PIN/system key or the PIN/personal key approaches. The intelligent secure card

prevents skimming of KP information during routine operations.

It is assumed that cardholders will take appropriate steps to protect their cards during periods of nonuse and that they will promptly report lost or stolen cards. Computations involving KP are performed on the card which means that KP is not read into the EFT terminal. Therefore,

KP is not exposed to a fake equipment attack,
KP is not exposed to probing or bugging of EFT terminals.

Except for the PIN, which is assumed to be entered into the EFT terminal where it exists momentarily before being transferred to the intelligent secure card, secret user-supplied information used in the process of personal verification is known only to the cardholder and issuer. Because KP and PIN together have more than 56 independent secret bits,

exhaustive attacks (trying all PIN and KP combinations) at the point of entry are infeasible,
dictionary and exhaustive attacks (on the system) are infeasible,
a one-way function of PIN and KP is possible,

which implies that it is not possible to deduce PIN and KP or to determine

equivalent values of PIN and KP from information transmitted throughout the system, and

there is no need to involve or depend on encryption under secret system keys.

The result is that personal verification and authentication of transaction *request* messages can be isolated to a very high degree, since only the cardholder and issuer are involved. True end-to-end cryptographic protection is thus achieved between the cardholder and issuer, and requirement 5 is satisfied. System keys (in addition to personal keys) are used for the authentication of transaction response messages, which means that

EFT terminals are not exposed to a misuse of a personal key attack or a fake personal key attack.

However, complete isolation with regard to the authentication of transaction *response* messages (requirement 17) is achieved only if a scheme for digital signatures is used by the institutions in the interchange network. With digital signatures,

no sharing of secret keys among institutions or with a switch would be required, no cryptographic translations of data as they traverse the network would be required, and the acquirer would have an electronically signed receipt for each transaction request authorized by the issuer.

KEY MANAGEMENT CONSIDERATIONS—SYMMETRIC VERSUS ASYMMETRIC ALGORITHMS

Frequently, the argument is made that key management is simplified with public-key (asymmetric) algorithms as opposed to conventional (symmetric) algorithms like the DES.[40] To prove or disprove such a general statement, however, is a nontrivial problem.

It must be recognized at the outset that to initialize a system employing symmetric algorithms requires a secure path to distribute the secret keys (e.g., by courier). A system employing asymmetric algorithms also needs a secure path for it's secret keys. But to distribute public keys requires only a channel with integrity (i.e., it must be assured that the correct public keys are distributed). An asymmetric system will obviously not be much simpler than a symmetric system if the number of secret keys to be distributed is comparable. Thus it will depend on the particular application if key management will or will not be simpler with a public-key algorithm.

In the EFT design discussed below (see A Cryptographic System Using an Intelligent Secure Card and A Public-Key Algorithm) an implementation is suggested which uses only public keys at system entry points. Secret keys are required at host nodes and on bank cards in the form of personal keys. Such

[40] The term asymmetric indicates that the encrypting and decrypting keys are different whereas the term symmetric indicates that the encrypting and decrypting keys are (basically) the same.

an implementation offers an advantage over the hybrid key management approach discussed above since it provides a higher degree of isolation among institutions. Before embarking further into EFT system designs, some general ideas are worth examining.

In particular, it is important to distinguish between implementations where (1) cryptographic authentication alone or authentication as well as secrecy is required and (2) where secrecy but not authentication is required. In the former case it is not generally clear if there is a major difference in key management complexity between both approaches. However, in the latter case (secrecy without authentication) the public-key approach definitely results in simpler key management. Comparisons are made here in terms of the number of keys stored in the system. The details of how the system must be initiated are not given. Thus the final verdict of which approach is simpler may very well depend on the protocols which must be used to initialize the system.

Authentication With and Without Secrecy

An implementation that comes to mind first is probably one where n users in the system wish to communicate with each other using personal keys. System involvement is thus minimized in such an approach.

In the asymmetric (e.g., RSA) approach, each user defines his own secret key and corresponding public key. Since the integrity of the public keys must be assured (otherwise authentication is not possible) they will most likely be stored at a system node defined as the key distribution center (KDC).[41] The process of storing public keys requires that users are identified before a public key is accepted by the KDC. (Otherwise an opponent could masquerade as a legitimate system user.) Once this process is completed, the KDC has the added responsibility to route public keys to the appropriate system entry point in such a way that they can be authenticated by the requester. This requires the presence of a secret key belonging to and stored within the KDC. Consequently, a secret key (SKu, or universal secret key) and corresponding public key (PKu) are defined by the KDC.

If user i wants to communicate with user j, he would request user j's public key, PKj, from the KDC. To assure that correct (current, not stale) PKs are received, a handshake protocol between the requesting user and the KDC is defined wherein user i sends a random number, RNi, together with other information, to the KDC in the form IDi, IDj, . . . , $E_{PKu}(D_{SKi}(IDi,RNi))$. The KDC would then look up user i's public key based on IDi obtained from the request message to recover RNi (using SKu and PKi). The quantity $E_{PKi}(D_{SKu}(IDi, PKj, RNi))$ is created next and sent to user i who subsequently deciphers with SKi and enciphers with PKu to recover PKj and RN. If the

[41] One might also consider an approach in which the set of public keys are published in a directory, eliminating the need for a KDC. But this requires that users input data (i.e., with the RSA algorithm on the order of 200 decimal digits each, see Chapter 2). In addition, a practical method must be found to periodically update the directory. Furthermore, the integrity of the public keys must be assured. A KDC solves all of these problems very efficiently.

Key Distribution Center		
ID1;	PK1;	$D_{SKu}(ID1, PK1)$
ID2;	PK2;	$D_{SKu}(ID2, PK2)$
⋮	⋮	⋮ ⋮
IDn;	PKn;	$D_{SKu}(IDn, PKn)$

Note: The KDC stores its secret key, SKu, separate from the above table (i.e., in secure hardware). The signatures $D_{SKu}(IDi, PKi)$, i = 1,2, . . . , n, allow the stored public keys to be authenticated.

User Identification Card
IDi · · · user i's identification
PKu · · · public key of KDC
PKi · · · user i's public key
SKi · · · user i's secret key

Figure 11-31. Personal Key Approach with Asymmetric Algorithm (RSA)–Information Stored in the System and on the User Identification Card

recovered RN is identical to the RN originally generated (and presumed saved) by user i, user i concludes that PKj was in fact sent from the KDC and thus is user j's public key. This protocol is repeated by user j who must later obtain user i's public key to participate in a meaningful conversation. The information needed by the KDC and the information supplied by the user (as read from a magnetic stripe on an appropriate identification card, for example), is shown in Figure 11-31. The above suggested handshake protocol is illustrated in Figure 11-32.

A personal key approach using the DES could be implemented by also using a KDC but storing each user's secret personal key KP instead of storing

User i obtains PKj as:

IDi, IDj, $E_{PKu}(D_{SKi}(IDi, RNi))$ ⟶ to KDC
IDi, IDj, $E_{PKi}(D_{SKu}(IDi, PKj, RNi))$ ⟵ from KDC

User i subsequently recovers PKj using SKi and PKu

User j obtains PKi as:

IDj, IDi, $E_{PKu}(D_{SKj}(IDj, RNj))$ ⟶ to KDC
IDj, IDi, $E_{PKj}(D_{SKu}(IDj, PKi, RNj))$ ⟵ from KDC

User j subsequently recovers PKi using SKj and PKu

Figure 11-32. Personal Key Approach with Asymmetric Algorithm (RSA)–Protocol to Establish Authenticated Public Keys

Key Distribution Center	
ID1;	Encrypted KP1
ID2;	Encrypted KP2
⋮	⋮
IDn;	Encrypted KPn

Note: The KDC stores its secret master key used to encrypt personal keys separate from the above table (i.e., in secure hardware)

User Identification Card
IDi · · · user i's identification
KPi · · · user i's secret personal key

Figure 11-33. Personal Key Approach with Symmetric Algorithm (DES)–Information Stored in the System and on the User Identification Card

public keys. The information stored at the KDC and on each user's bank card is shown in Figure 11-33. These secret keys could then be used to securely distribute and authenticate a session key (KS) randomly generated by the KDC (Figure 11-34). To determine that a received session key has indeed originated with the KDC, a random number is generated by the user and sent, encrypted under his personal key, to the KDC. Only if the KDC returns that same random number together with a session key encrypted under the user's personal key, will the session key be accepted as genuine.

The key management requirements of RSA and DES are not much different as far as the user-supplied input information is concerned. The KDC must, in the former case, assure the integrity of n public keys and the integrity and secrecy of its secret key (SKu). In the latter case the integrity

User i obtains KS as:

IDi, IDj, E_{KPi}(IDi, RNi) ⟶ to KDC
IDi, IDj, E_{KPi}(IDi, RNi, KS) ⟵ from KDC

User i recovers KS using KPi

User j obtains KS as:

IDj, IDi, E_{KPj}(IDj, RNj) ⟶ to KDC
IDj, IDi, E_{KPj}(IDj, RNj, KS) ⟵ from KDC

User j recovers KS using KPj

Figure 11-34. Personal Key Approach with Symmetric Algorithm (DES)–Protocol to Establish Authenticated Session Keys

	Asymmetric (RSA) Algorithm	Symmetric (DES) Algorithm
Number of secret keys per user	1	1
Number of public keys per user	1	none
Number of secret keys in KDC	1	n user keys 1 KDC master key
Number of public keys in KDC	n	none

Table 11-12. Required Number of Keys for Asymmetric and Symmetric Algorithms—Personal Key Approach

and secrecy of n user keys and the master key of the KDC must be assured. Either approach requires a secure system node (the KDC). These conclusions are summarized in Table 11-12.

Let it next be assumed that n nodes in a network communicate with each other such that cryptography is transparent to the user (i.e., the user is not required to provide cryptographic parameters). Let it furthermore be assumed that secrecy and authentication are required. This can be achieved (using for example the RSA algorithm) by deciphering with the sender's secret key and enciphering with the receiver's public key as shown in An Approach Using Public Key-Algorithms, Chapter 9.

In a symmetric system, authentication and secrecy are automatically achieved by defining one secret key and performing one operation only (i.e., encryption). Starting with a symmetric system, let KCi,j (where KC denotes a communication key) define the secret key which operates on messages sent from node i to node j. For a three node network, six secret keys (as shown in Figure 11-35) must be defined for complete node to node communication assuming each node manages keys independently of each other node.

For a n node network there are $2(n-1)$ keys per node required (i.e., $n-1$ keys to operate on data sent from one node to the $n-1$ other nodes and $n-1$ keys to operate on data received by one node from the $n-1$ other nodes). Hence there are a total of $2n(n-1)$ keys in the system. Since some identical keys are stored in different nodes (e.g., KCi,j appears in node i and node j), there are only $n(n-1)$ different keys in the network.

If different keys are not needed to protect data flowing in opposite directions between two nodes, then KCi,j may be identical to KCj,i. This reduces the number of keys required at each node from $2(n-1)$ to $(n-1)$. The total number of keys in the network then becomes $n(n-1)$ and the total number of different keys in the network becomes $n(n-1)/2$.

Let an asymmetric system (e.g., the RSA algorithm) be discussed next and let SKi define the secret key used at node i to operate on (decipher) data sent to any other node. The corresponding public key, PKi, is made available

Node 1	Node 2	Node 3
KC1,2 →	KC1,2	
KC2,1 ←	KC2,1	
KC1,3 →		KC1,3
KC3,1 ←		KC3,1
	KC2,3 →	KC2,3
	KC3,2 ←	KC3,2

The arrow indicates data flow direction and the corresponding entries indicate the keys to be used.

Figure 11-35. Symmetric Algorithm—Keys Required to Achieve Authentication and Secrecy in a Three Node Network

to other system nodes to authenticate messages received from node i. Communications from node i to node j take the form $E_{PKj}D_{SKi}(Xi,j)$ where Xi,j are the data sent from node i to node j whose integrity and secrecy must be maintained. For a three node network, three secret and three public keys must be defined as shown in Figure 11-36.

In an n node network, each node stores one secret key and (n − 1) public keys. There are thus a total number of n secret keys in the network and (n − 1)n public keys. The total number of different keys is less because the public key of node i appears in (n − 1) nodes. Hence there are n different secret keys and n different public keys in the network.

If only authentication (not secrecy) is required, the data Xi, j sent from node i to node j are of the form $D_{SKi}(Xi, j)$ instead of $E_{PKj}(D_{SKi}(Xi, j))$. At the receiver (node j), PKi is used to authenticate Xi, j. Hence the same keys defined above for authentication and secrecy are required for authentication even when secrecy is not required. A summary is given in Table 11-13.

Node 1	Node 2	Node 3
PK2;SK1 →	PK1;SK2	
SK1;PK2 ←	SK2;PK2	
PK3;SK1 →		PK1;SK3
SK1;PK3 ←		SK3;PK1
	PK3;SK2 →	PK2;SK3
	SK2;PK3 ←	SK3;PK2

Note: Secrecy as well as integrity of the SKs must be assured whereas for the PKs only integrity must be assured.

The arrow indicates data flow direction and the corresponding entries indicate the keys to be used.

Figure 11-36. Asymmetric Algorithm—Keys Required to Achieve Authentication and Secrecy in a Three-Node Network.

	Asymmetric (RSA) Algorithm	Symmetric (DES) Algorithm
Number of secret keys per node	1	2(n-1) w/ unidirectionality (n-1) w/o unidirectionality
Number of public keys per node	(n-1)	none
Total number of secret keys in network	n	2n(n-1) w/ unidirectionality n(n-1) w/o unidirectionality
Total number of public keys in network	n(n-1)	none
Total number of different secret keys in network	n	n(n-1) w/ unidirectionality n(n-1)/2 w/o unidirectionality
Total number of different public keys in network	n	none

Note: It is assumed that all n nodes of the network communicate with each other without user involvement.

Table 11-13. Required Number of Keys for Asymmetric and Symmetric Algorithms—Transparent Case where Each Node Stores the Required Keys

From Table 11-13 one could easily conclude that key management becomes less complex with asymmetric systems since fewer keys are managed. To make a true comparison, however, system design concepts must also be considered. Most likely an approach where each node stores a set of KCs will not be used. Instead, one would define one secret key per node (KNCi for node i) and store all n of the required keys in a common key distribution center (KDC). Thus the KDC would be the trusted node in the system and would be called upon to generate and distribute session keys to nodes requesting to communicate with one another. For example, a randomly generated session key (KS) would be distributed to nodes i and j in the form $E_{KNCi}(KS)$ and $E_{KNCj}(KS)$, respectively.

The KDC could also be used with an asymmetric algorithm. In that case the KDC would store all n public keys and route them to the system nodes as required. Based on the number of keys stored at the KDC (Table 11-14), one cannot conclude which system (asymmetric or symmetric) is easier to implement. Other investigators analyzing asymmetric and symmetric cryptosystems have concluded that protocols in both systems are strikingly similar [17, 18].

Secrecy Without Authentication

In this section the question to be addressed is: What security penalty (if any) is there if permanently installed keys are not used? The rationale is that each node could generate its public and secret keys dynamically. For ex-

	Asymmetric (RSA) Algorithm	Symmetric (DES) Algorithm
Number of secret keys per system node	1	1
Number of public keys per system node	1	none
Number of secret keys in KDC	1	n user keys 1 KDC master key
Number of public keys in KDC	n	none

Table 11-14. Required Number of Keys for Asymmetric and Symmetric Algorithms–User Transparent Case with Key Distribution Center

ample, the public key generated at node i could be sent over a nonsecure channel to node j. Node j could then encrypt messages for node i with node i's public key (PKi). Only node i, having generated the corresponding secret key SKi can decrypt such a message. Thus, off hand it seems that an acceptable solution has been found for key distribution and key initialization. (For example, the keys required in the three node network shown in Figure 11-36 can easily be generated on demand.)

But there is a major difference between the static situation where keys at each node are defined in an initialization process and the dynamic situation where keys are generated as they are needed. Due to the fact that the keys listed in Figure 11-36 are defined ahead of time as part of system initialization, the corresponding nodes are coupled. After initialization, the sender (node i) has no control over the use of a key by another node (node j). This is in contrast to the case where the keys are generated dynamically. As a consequence there is no way for a node to check the identity of the sender, since the public key sent to the receiving node could have originated with any system node, including a bogus node. The sender, on the other hand, does not really know who is using his personal key to encrypt data addressed to him. To take advantage of these security weaknesses requires, however, an active attack since data on a communications line must be altered enroute. Therefore the described method of dynamically generating keys in lieu of initializing the system with predetermined keys provides data security if the opponent has only the capacity to eavesdrop.

To illustrate the consequences of implementing a cryptographic system without authentication, consider the following case. To send a secret document from A to B, let it be locked in a suitcase. To start with, A seals the suitcase with lock A, which only A can open. After B receives the suitcase, B in turn seals the suitcase with lock B, which only B can open. The doubly locked suitcase is then returned to A. Upon receipt, A removes (his) lock A. The suitcase, still locked with lock B, is returned to B whereupon B removes his lock and retrieves the secret document (Figure 11-37).

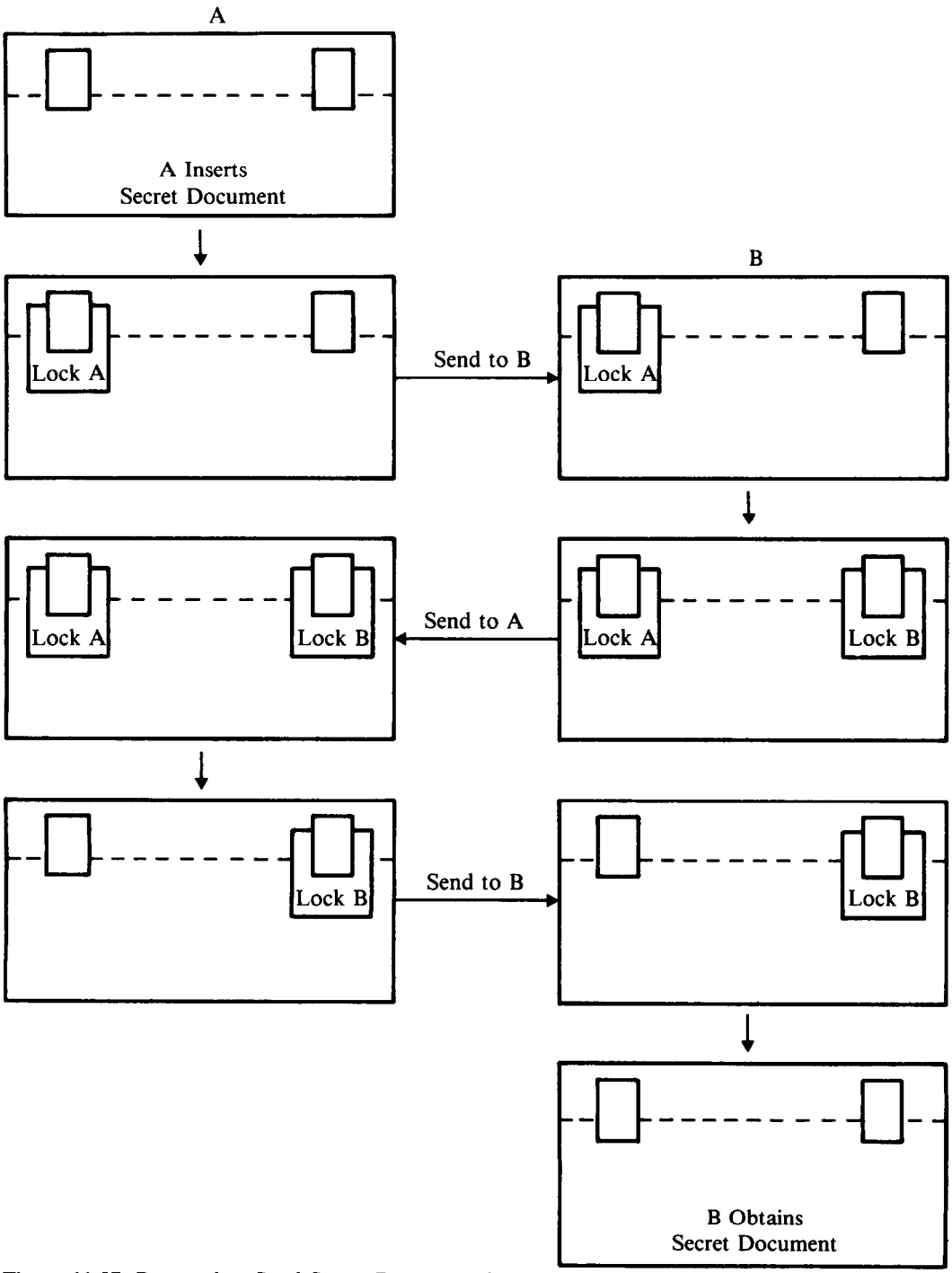


Figure 11-37. Protocol to Send Secret Document from A to B

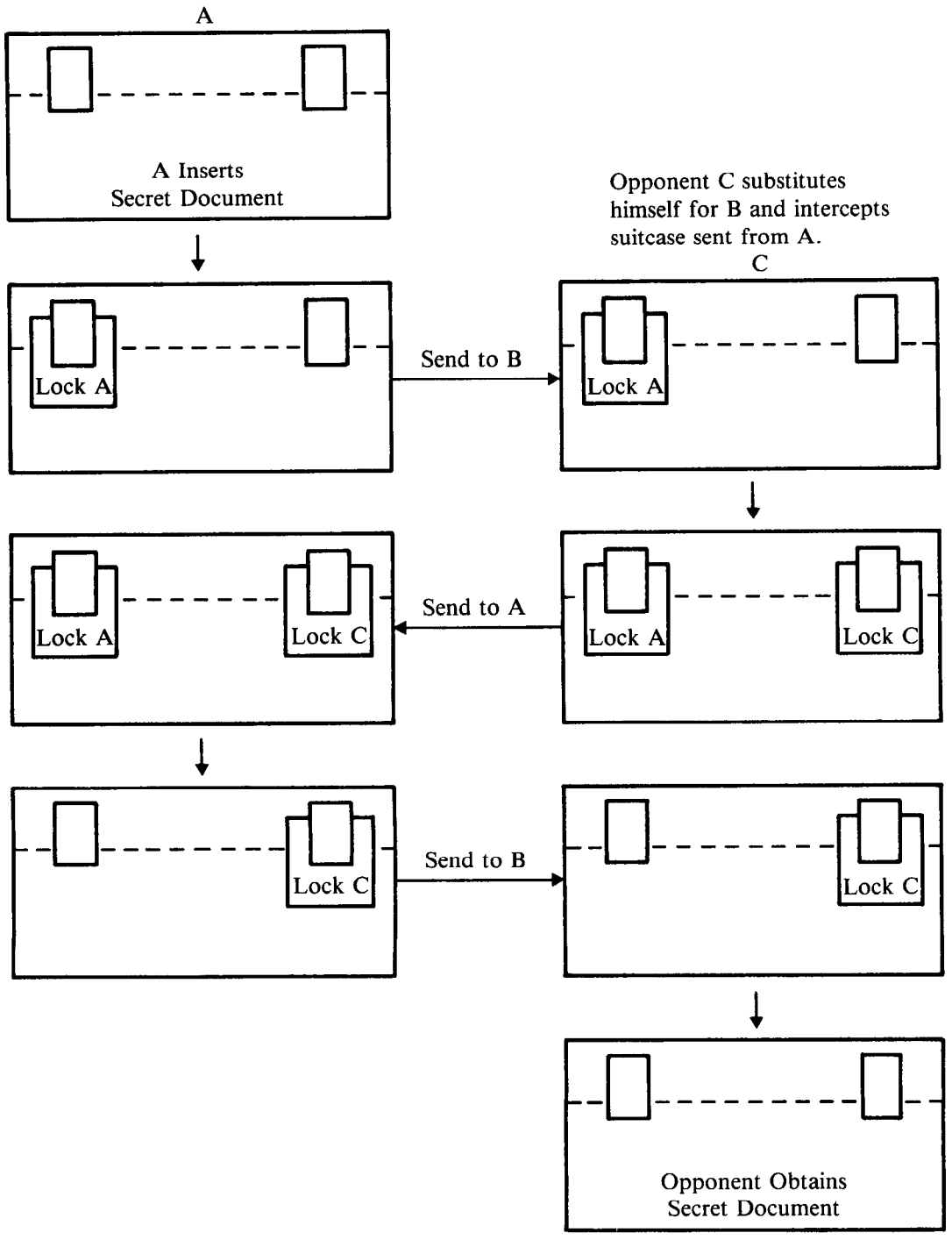


Figure 11-38. Interception of Secret Document by Opponent

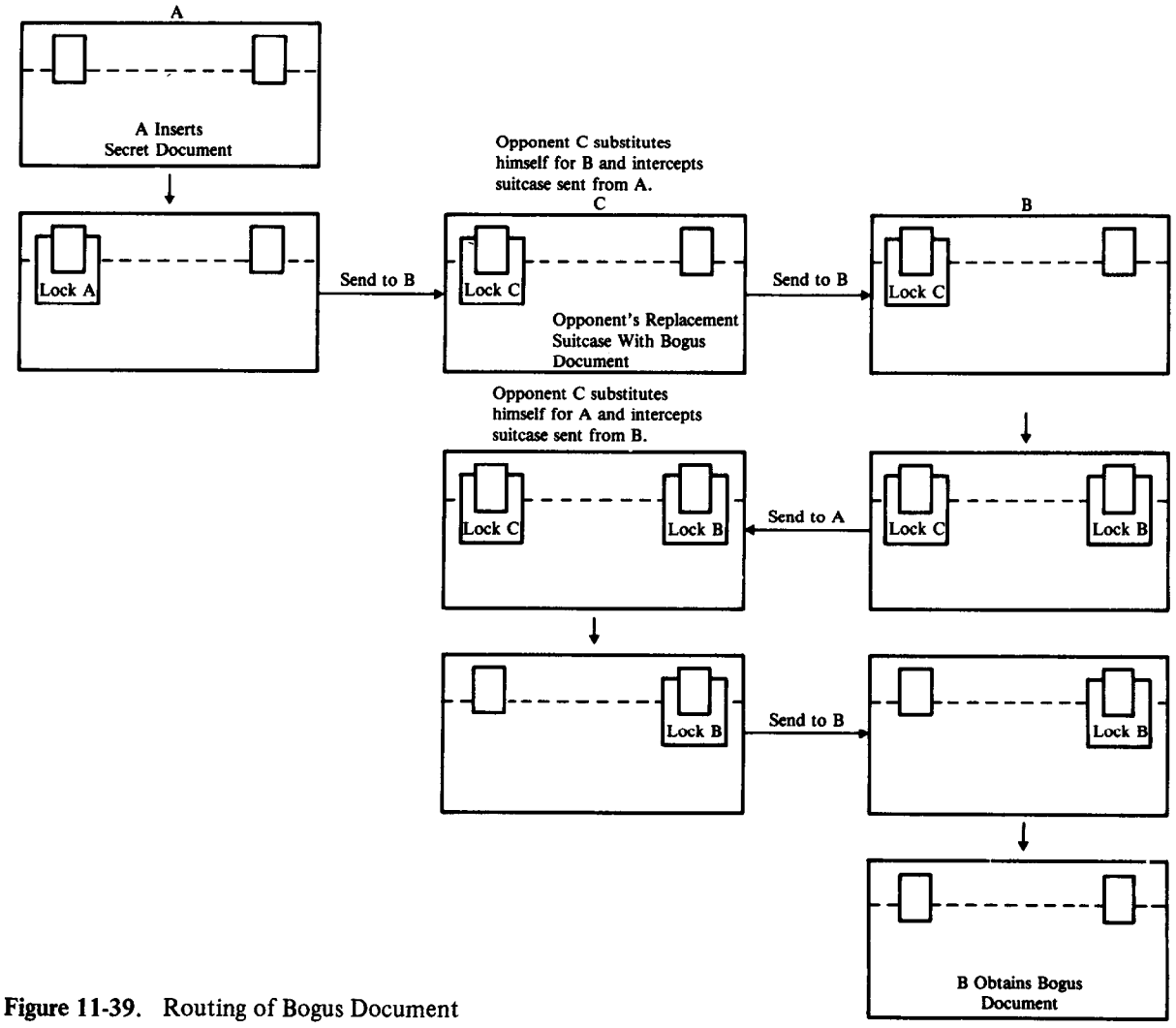


Figure 11-39. Routing of Bogus Document

There are two basic attacks which may be successful in this instance: rerouting and masquerading.

Since A cannot distinguish between B's lock and one supplied by an opponent (due to lack of authentication), it is possible for an opponent to intercept the suitcase and apply his own lock, thus preventing B from applying his lock. As a result the suitcase, sealed with the opponent's lock would be returned to A for removal of A's lock. Once A's lock is removed, the opponent again intercepts the suitcase, removes his lock, and thereby obtains the secret document (Figure 11-38).

In the second attack the opponent again intercepts the suitcase (with A's lock in place) only this time the suitcase and secret document are discarded. The opponent replaces the suitcase with one of his own, complete with a bogus secret document, locks the suitcase with his lock, and forwards the suitcase to B. The protocol requires B to apply his lock and return the suitcase to A, which B does. The opponent again intercepts the suitcase, removes his lock, and returns the case to B. B eventually opens his lock and removes the bogus secret document thinking it originated with A (Figure 11-39). The opponent (posing as B) also sends a bogus suitcase to A to prevent A from detecting the deception.

Figures 11-38 and 11-39 illustrate two attacks against a public-key cryptosystem implementing the particular communication protocol described in Figure 11-37. In general, a public-key cryptosystem's communication protocol is always exposed to active attack if that cryptosystem does not employ preinitialized keys. Although, for applications in which active attacks are considered not a threat, the tradeoff between security and key management complexity may be an attractive one.

A CRYPTOGRAPHIC SYSTEM USING AN INTELLIGENT SECURE CARD AND A PUBLIC-KEY ALGORITHM

The PIN/personal key approach, which was shown to be nonsecure, demonstrates the need for having system keys installed in the EFT terminals. This is true whether a conventional (symmetric) or public-key (asymmetric) algorithm is used. If a conventional algorithm is employed, the terminals must store secret keys. If a public-key algorithm is employed, the terminals may store secret keys, public keys, or both, although, typically, the terminals would store only public keys. (These public keys would be used to authenticate transaction response messages.) When a public-key algorithm is used, one therefore attempts to improve the process of message authentication by eliminating the need for a secret key in the EFT terminal. The requirement for keeping the necessary terminal resident keys secret and assuring their integrity in a conventional approach is thus replaced by the requirement of assuring only the integrity of public terminal resident keys in a public-key approach.

Authentication of transaction request messages, on the other hand, always requires a secret key at the entry point. This key must be supplied by the system user, since otherwise the terminal would have to store a secret key (which is to be avoided).

The main features of the system are thus as follows. Personal verification and authentication of transaction request messages (sent from the terminal to the issuer) are based on a secret user-supplied key derived from the PIN and secret card information and a corresponding public user key stored at the issuer (different for each of the institutions's customers)[42]. Authentication of transaction response messages is based on a secret key available in the security module of the issuer's EDP system and a corresponding public key established at each terminal.

In the described system there is total isolation of the personal verification processes of the various institutions and almost total isolation in the authentication of transaction requests sent from the user to the issuer. These procedures are effected and established solely between the user and the issuer. No secret keys involved in the processes are exposed at the entry point, although the integrity of the public keys must be assured. In an interchange, the acquirer and switch merely act as network routing points to pass nonsecret information to the issuer.

In addition, there is a digital signature capability for transaction responses. The acquirer honors a request from the cardholder only after the issuer has sent a signed message to the acquirer (or to the originating terminal) authorizing the transaction. (Note that this advantage is also obtained with the DES/public-key approach discussed in the section Security Enhancements with Digital Signatures.)

Description of a Public-Key Management Approach

The system discussed here is composed of host processing centers and EFT terminals, interconnected in an EFT network supporting interchange. Each network node has a public-key cryptographic capability that is either integrated into the node or contained in a security module attached to the node via a secure, local cable. Each security module has a set of cryptographic operations that may be invoked by the supporting device or EDP system via a defined interface. No clear cryptographic keys ever exist outside the security module except during periods when they are initially generated or entered into the system.

Each customer is provided with an intelligent secure bank card, which has an installed public-key algorithm and storage for secret and nonsecret information (e.g., keys, encrypted keys, and account-related information). Keys stored in a security module are protected by implementing adequate physical security measures and/or providing a set of interlocks that will erase all secret information if penetration of the security module or containing device is detected.

A secret user key (SKc, where c stands for customer) is used to generate a quantity (DGSreq) which will enable the issuer to authenticate the transaction

[42] PIN secrecy depends entirely on maintaining the secrecy of certain card information (i.e., it is not achieved because of the public-key algorithm). If relevant card information were available to an opponent, the PIN could be derived easily from information in an intercepted transaction request message. The intelligent secure card provides the means to adequately protect card information.

request message, Mreq. In the conventional approach this quantity was called a MAC. Since a public-key approach provides a digital signature capability, the term DGS is used instead of MAC. To demonstrate the parallelism between the public-key and conventional algorithm approaches, assume that (Mreq, DGSreq) is routed to the issuer and define DGSreq as

$$\text{DGSreq} = D_{SKc}[\text{CE(Mreq)}]$$

where CE(Mreq) represents the compressed encoding of Mreq. As discussed in Chapter 9, CE(M) is a one-way function of M and can be generated with publicly known keys for symmetric as well as asymmetric algorithms. In the implementation discussed here it suffices to specify one public key for generating CE(M) without, at the same time, specifying the corresponding secret key. This is so because the sender and receiver use the same procedure for generating CE(M). To check the signature, the issuer generates the compressed encoding of the received message and compares it with E_{PKc}(DGSreq) (i.e., the received DGSreq encrypted under PKc). If both quantities agree Mreq is accepted; otherwise, not.

This check on DGSreq can be used also to verify the user (e.g., if SKc is defined as SKc = SKc* $\oplus$ PIN, where SKc* is a secret parameter stored on the card). Since the digital signature (DGS) is now also a function of PIN, personal verification as well as message authentication are combined in one procedure. [The same idea was used in the hybrid key management approach by defining KTR = $D_{KP \oplus PIN}$(ID) and is repeated here for the sake of uniformity in the discussion.]

A digital signature on the response message (DGSresp) is generated using a secret key SKb, uniquely defined for each institution (where b stands for bank). Thus,

$$\text{DGSresp} = D_{SKb}[\text{CE(Mresp)}]$$

To check the received Mresp at the terminal, the public key, PKb, corresponding to SKb, must be available. In the implementation suggested here, PKb and its digital signature D_{SKu}[CE(IDb, PKb)] are stored on the bank card. The key SKu (where u stands for universal) is a key known only to a trusted node or key distribution center. By storing the corresponding public key, PKu, in all terminals, PKb can be authenticated at the terminal before it is used to authenticate Mresp. Authentication of PKb is achieved by enciphering D_{SKu}[CE(IDb, PKb)] with PKu and checking that the result is equal to the compressed encoding of (IDb, PKb), where PKb is the received public key and IDb is the known bank identifier.

To initialize the operation, each institution produces a public and private key-pair (PKb, SKb) for its own use. The private bank key (SKb) is retained in the institution's security module. The public bank key (PKb) is distributed to each other institution in the interchange, and it is also sent (e.g., via a courier) to a key distribution center or designated trusted party. Public keys are distributed securely to assure their integrity (e.g., to avoid masquerading attacks).

Prior to receiving each institution's PKb, the key distribution center produces its own public and private key pair (PKu, SKu) for the purpose of interchange. The secret key, SKu, is employed to generate a digital signature for PKb in the form $D_{SKu}[CE(IDb, PKb)]$. The user supplies this quantity together with IDb and PKb to the terminal where PKu resides. This enables the terminal to check the DGS and thus authenticate PKb.

The public key of the key distribution center (PKu) and the public bank keys, together with their DGSs, are distributed to each of the respective institutions (banks). SKu is also stored in a safe or vault for recovery purposes. Securely maintaining SKU will also allow other institutions to later join the interchange. The key distribution center can also publish PKu (e.g., in a major newspaper like the New York Times), which will allow each institution to validate the received PKu independently.

Upon receipt of this information from the key distribution center, each issuer validates PKu (by comparing the received PKu with the published PKu) and transfers PKu to its security module where it can be safely stored and used as necessary. The public keys of each bank and their corresponding digital signatures (computed from SKu) are stored in the data base of the institution's EDP system (Figure 11-40).

The issuer also generates a DGS for each users public key, PKc, with the aid of SKb (e.g., $D_{SKb}[CE(IDc, PKc)]$). The quantities IDc, PKc, and the DGS for PKc, are then stored in the data base of the institution's EDP system (Figure 11-40) and written on the user's bank card (Figure 11-41). A copy of the institution's public bank key PKb and signature $D_{SKu}[CE(IDb, PKb)]$ are also written on the user's bank card. Each institution also installs PKu in each of its terminals (Figure 11-42).

PIN Selection

For this discussion, PINs are assumed to be produced by the issuer using the security module as a generator of pseudorandom numbers. The generated PIN is printed on a PIN mailer and the mailer is sent to the customer. PINs may also be encrypted under a PIN master key and stored off-line for purposes of backup.

Generation of the User's Public and Private Keys

The issuer will produce (in addition to PIN) a public and private key pair (PKc, SKc) for each customer. The user's private key and PIN are Exclusive-ORed to produce a secret card parameter, SKc* (i.e., $SKc^* = SKc \oplus PIN$), which is then written on the user's intelligent secure card.

Validation of the User's PIN and Card Key

Each time the customer initiates a transaction at a terminal, the customer's PIN is entered via a PIN pad or keyboard and transferred to the card. (If the card has its own keyboard, the PIN would not be exposed in the terminal.) On the card, PIN is Exclusive-ORed with the secret card parameter (SKc*) to produce the user's private key, SKc, and SKc is then used to generate a DGS

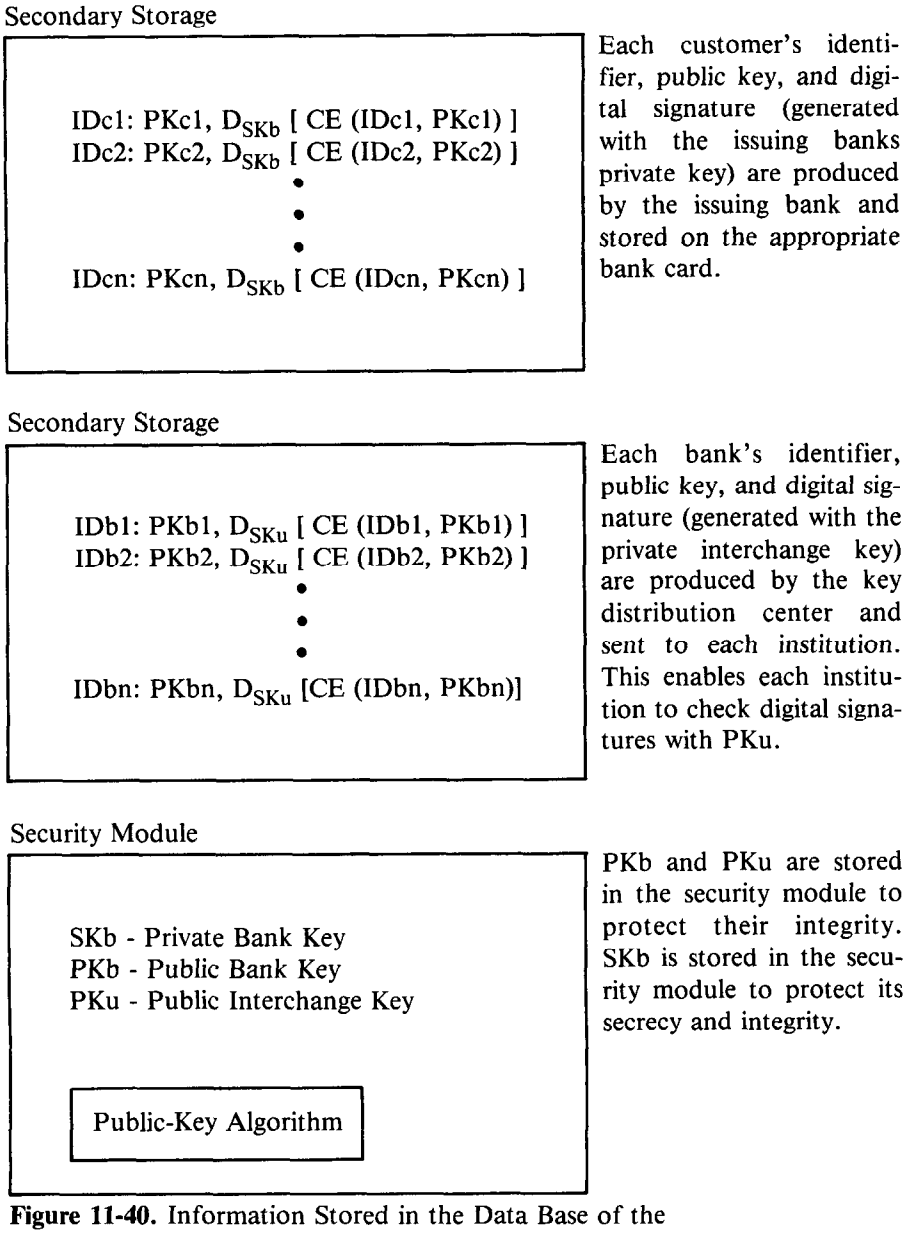


Figure 11-40. Information Stored in the Data Base of the Issuer's EDP System and in the Issuer's Security Module

on the transaction request message via the public-key algorithm. A time-of-day (TOD) clock obtained from the acquirer via the terminal (as described earlier) is included in the message to ensure that it is time-variant.

At the issuer, the corresponding PKc of reference, which is filed under the user's identifier, is read from the EDP system's data base[43] and used to

[43] See footnote 35.

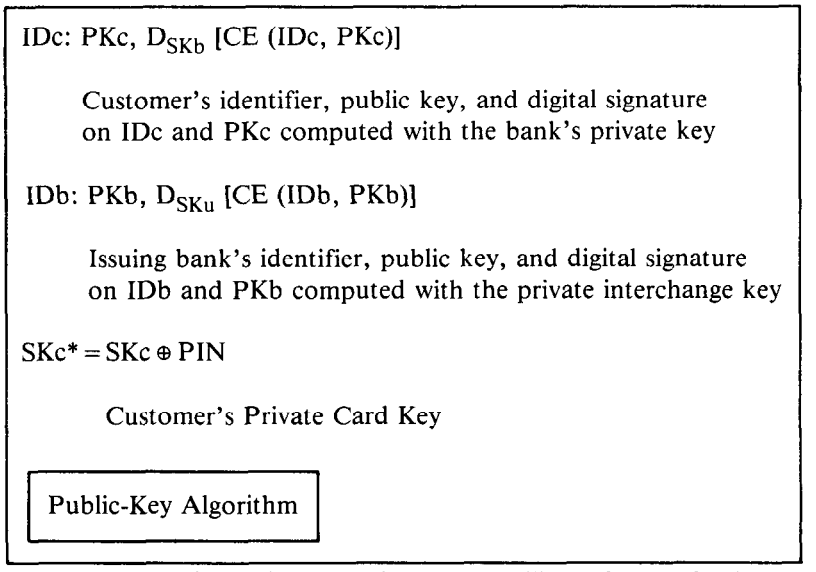


Figure 11-41. Information Stored on the Intelligent Secure Card

encrypt the received DGSreq. This result is compared for equality with the compressed encoding calculated on the received message Mreq. Furthermore, the received TOD is checked for currency against a TOD of reference stored in the issuer's EDP system. If both tests succeed, the issuer concludes that the content of the message is correct, the message is not a stale message, and the secret information entered by the customer at the entry point (SKc* and PIN) is properly related to the claimed ID.

Key Management Considerations for Asymmetric Algorithms

To take maximum advantage of the public key idea, secret terminal resident keys should be unnecessary. Only the public keys permanently stored in the terminals are needed to authenticate transaction response messages. To authenticate transaction request messages, however, requires a secret key at the entry point, although a public key can be used at the destination where these messages are checked. Such a secret key must therefore be supplied by the user since, by definition, no secret key should be stored in the terminal permanently. Such a design therefore dictates a personal key ap-

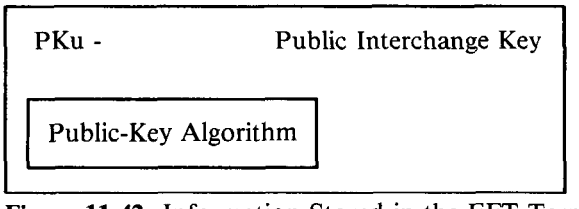


Figure 11-42. Information Stored in the EFT Terminal

proach, where the personal key is equal to the secret key required in the public key approach. The corresponding public key stored at the issuer is used to authenticate request messages.

The most straightforward approach for authenticating transaction response messages would be to store PKu at each EFT terminal and store SKu at each cooperating institution's HPC. A digital signature computed on Mresp at any institution using SKu could be checked at any terminal using PKu. However, this has the disadvantage that institutions share the secret key SKu—an undesirable situation. If SKu should become compromised, an opponent could generate a valid digital signature on any transaction response message.

To avoid sharing secret information and thereby realize separation among the institutions, each subscriber bank can define its own keys (PKb, SKb, where b represents bank). The secret key SKb is used by an individual institution to compute a digital signature on the transaction response message, Mresp. At the EFT terminal, the public key PKb is used to authenticate Mresp and its digital signature. However, such a solution requires that the public key of each institution be available at the entry point. Due to storage limitations (at EFT terminals), such an approach is impractical if there are a large number of subscriber institutions in the interchange network.

The disadvantage of storing a common SKu at each institution in the interchange, or of storing the PKb of each bank in each EFT terminal, can be avoided as follows. A trusted system node or key distribution center is designated to manage the universal secret key, SKu. The individual institutions still define their own (SKb, PKb) key pairs. To establish the correct PKb at an arbitrary entry point, the trusted node generates a digital signature on (IDb, PKb) using SKu. The terminals in which PKu is stored authenticate PKb by encrypting D_{SKu}[CE(IDb, PKb)] with PKu and comparing the result for equality with the compressed encoding of the supplied (IDb, PKb) stored on the bank card. Note that D_{SKu}[CE(IDb, PKb)] is also stored on the bank card. Thus, a hierarchical public-key approach is used in which a universal secret key, SKu, is the dominant system key. This key, known only to the trusted node, provides the means (via the digital signature) for each institution to authenticate the public keys of each other institution (PKb1, PKb2, . . . , etc.). Thus each PKb is checked before being used to authenticate transaction response messages at the entry point. Secret user keys, on the other hand, are used (in conjunction with PINs) to generate digital signatures on the transaction request messages, which in turn allows the issuer (who has the corresponding public keys) to verify users and authenticate transaction request messages.

Off-Line Use

Figure 11-43 illustrates an offline transaction. The customer's card is placed in a card read/write device coupled to or integrated within the terminal. The customer then enters his PIN via a suitable entry device (PIN pad or keyboard). The PIN is transferred to the card where it is Exclusive-ORed with the secret parameter SKc* on the card to form the user's private key, SKc, namely: SKc = SKc* ⊕ PIN. SKc is temporarily stored on the card, i.e., until transaction processing is complete.

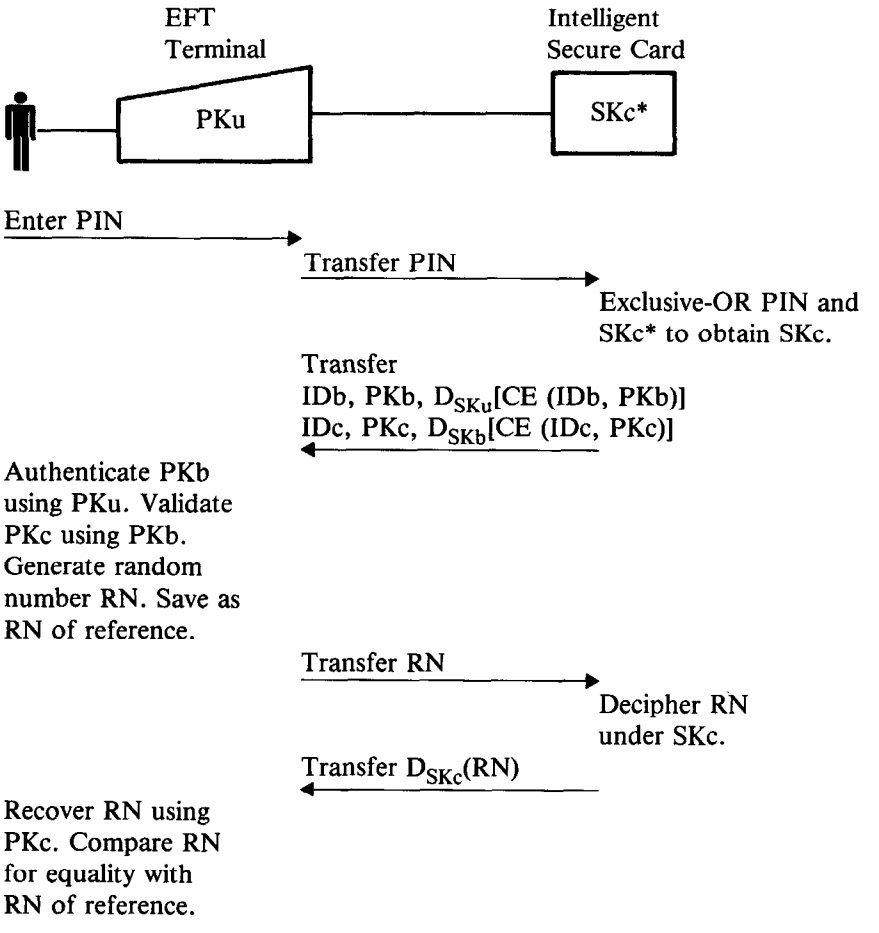


Figure 11-43. Off-Line Use

The two public keys, PKc and PKb, which were originally written on the card by the issuer, are then transferred from the card to the terminal together with their respective digital signatures. This enables the terminal to authenticate PKb and PKc with the aid of PKu (stored in the terminal).

To validate the user, the terminal performs a handshake with the card, as follows: The terminal generates a random number RN, which it transfers to the card and requests that the card decipher the random number under SKc. Upon deciphering the random number under SKc, the card transfers the result to the terminal. The terminal then enciphers the received value under PKc (which it previously authenticated) to recover RN. The recovered value of RN is compared for equality with the RN of reference. If the two quantities are equal, the terminal concludes that (1) the card is capable of cryptographic operations and (2) the card generated the proper SKc. Since SKc is a function of the secret card parameter (SKc*) as well as PIN, personal verification is achieved. The transaction request is then honored according to whatever established limits and prior protocols and arrangements have been implemented.

A variation of the method is for each terminal to also store the public key,

PKb, of the local bank. In that case, if a customer performs an off-line transaction at a terminal belonging to his own bank, the terminal can authenticate PKc directly from $D_{SKb}[CE(IDc, PKc)]$ and avoid the intermediate step of authenticating PKb from $D_{SKu}[CE(IDb, PKb)]$.

On-Line Use in Interchange and Noninterchange

In describing both local and interchange transactions it will be noted that there is no difference in the protocols. Only message routing is different, since different parts of the interchange network are traversed.

The data and computing capability of the customer's card are made available to the terminal (Figure 11-44) when the card is inserted into a suitable read/write device. The customer then enters his PIN via a suitable entry device, the PIN is routed to the card, and the PIN is Exclusive-ORed with the secret card parameter SKc* to form the user's private key, SKc, which is temporarily stored on the card.

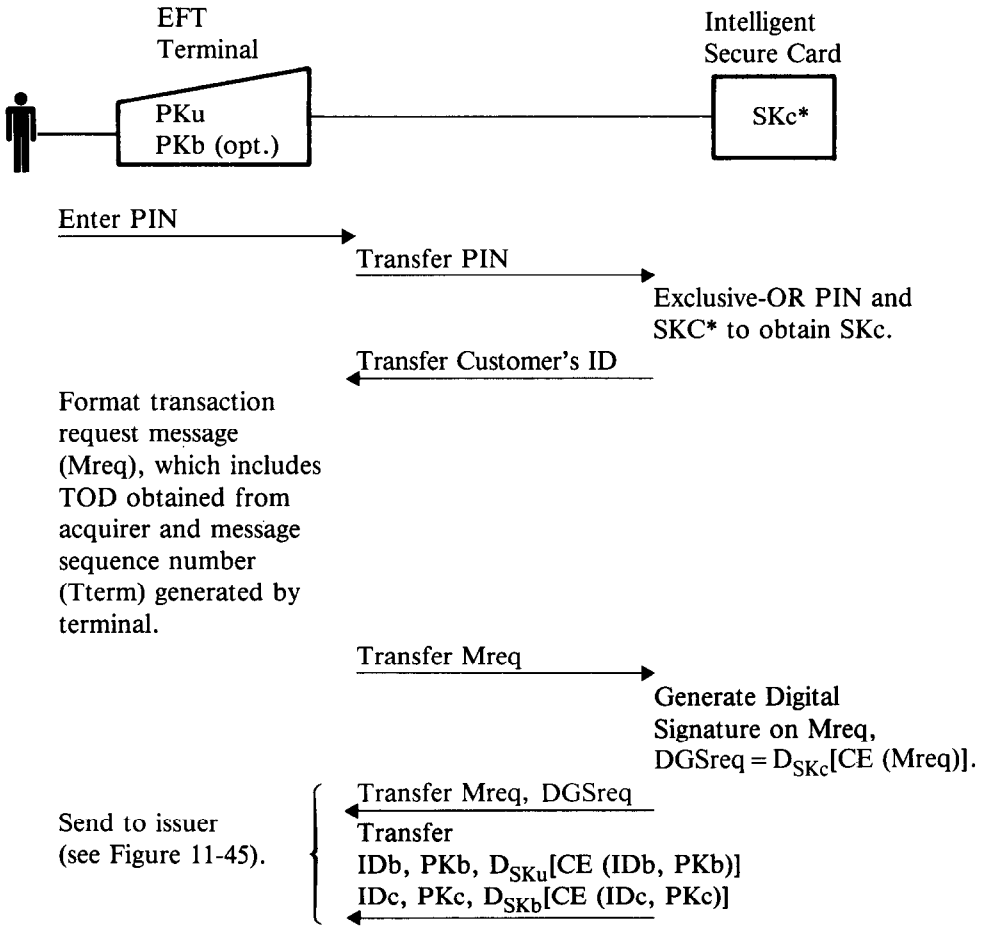


Figure 11-44. On-Line Use—EFT Terminal

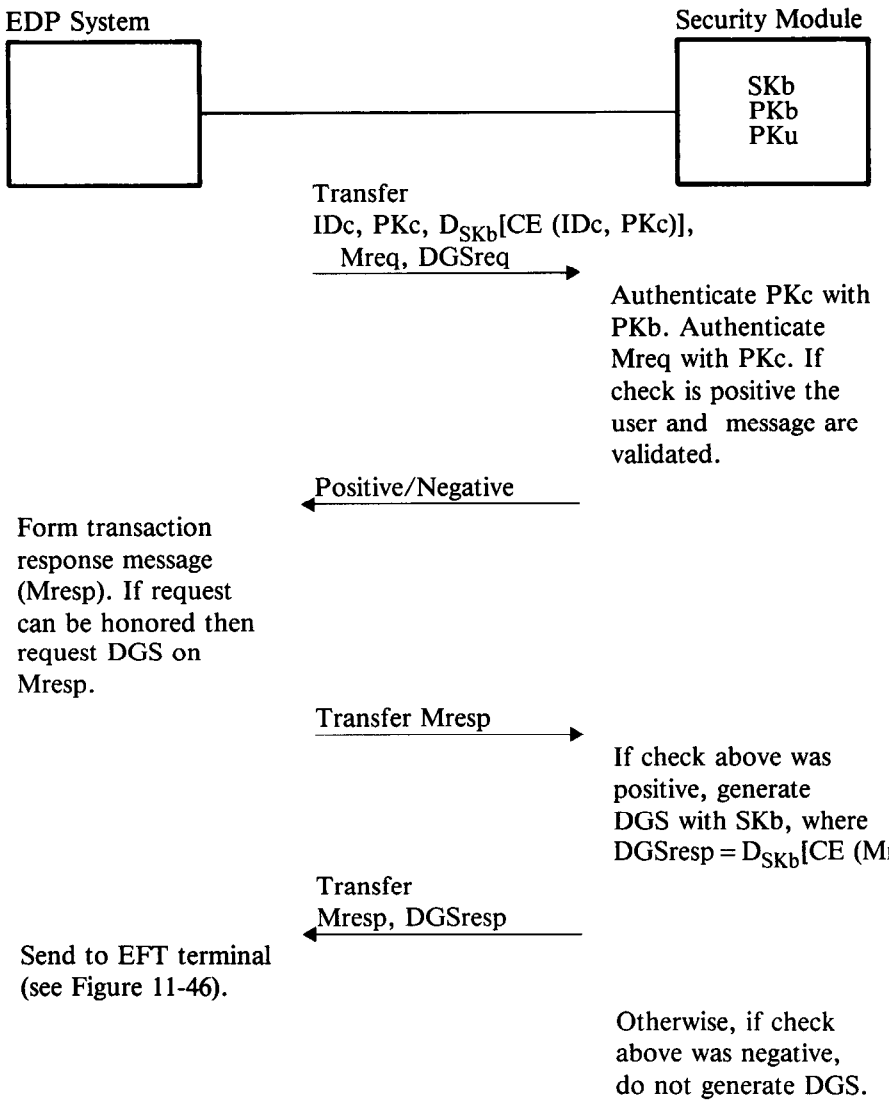
To allow the issuer to validate the transaction request message, a DGS is generated using SKc (i.e., $D_{SKc}[CE(Mreq)]$). This is accomplished by transferring the assembled message to the card where the compressed encoding of Mreq is generated and in turn deciphered under SKc. The message and signature are returned to the terminal for transmittal to the issuer (identical to the acquirer for a local transaction, different from the acquirer for interchange). If it is desired to allow intermediate nodes to authenticate and read the message, the quantities IDb, PKb, $D_{SKu}[CE(IDb, PKb)]$ and IDc, PKc, $D_{SKb}[CE(IDc, PKc)]$ can be read from the card and sent together with the transaction request message.

At the issuer (Figure 11-45), the corresponding PKc of reference is stored in the EDP system's data base, for example, in the form IDc, PKc, $D_{SKb}[CE(IDc, PKc)]$. Prior to its use, PKc of reference is authenticated using PKb. The received message is then authenticated by generating its compressed encoding and comparing the result for equality with the received DGSreq encrypted under PKb (i.e., with $E_{PKb}(DGSreq) = E_{PKb}[D_{SKb}(CE(Mreq))]$). If they are identical, and if the TOD checks, then the issuer concludes that the content of the message is correct, the message is not stale, and the secret information supplied by the user, SKc* and PIN, is properly related to the claimed ID. If the requested transaction can be honored, a positive response is sent to the originating terminal. Otherwise, a negative response is sent to the originating terminal.

To send a response, the issuer generates a digital signature on the response message Mresp using SKb (i.e., $DGSresp = D_{SKb}[CE(Mresp)]$). Mresp and DGSresp are then sent to the EFT terminal. A positive response could consist of sending back the request message (i.e., Mresp = Mreq and $DGSresp = D_{SKb}[CE(Mreq)]$). This will be assumed here. A negative response can be anything other than a positive response.

To authenticate the response message, the terminal reads IDb, PKb, $D_{SKu}[CE(IDb, PKb)]$ from the card and validates PKb using PKu (stored in the terminal). (If PKb is stored in the terminal, the prior step can be eliminated). The received DGSresp is then enciphered under PKb and the result is compared for equality with the compressed encoding of the Mreq of reference. If the two quantities are equal, the terminal honors the transaction; otherwise, the transaction is denied (Figure 11-46).

A summary of the keys required with the described public key approach is provided in Table 11-15. Tables 11-16 and 11-17 show the flows of information from the card to the issuer (via the EFT terminal) and from the issuer to the EFT terminal. Comparing these tables with those of the hybrid approach (Tables 11-9 through 11-11), one observes that only public quantities are shared among institutions. Thus a higher degree of isolation is achieved with the public key approach. On the other hand, it must be realized that the approach requires a secret universal key. If that key becomes compromised, the security of the entire system is lost. The described public key approach depends, therefore, on one node or key distribution center which is trusted by everyone in the system, and that one party controls and manages the universal secret key.



Note: To reduce information to be sent, DGSresp can be defined as DGSresp = D_{SKb} [CE (Mreq)]. In this case the response message is identical to the request message and does not have to be sent. Thus only DGSresp is returned to the terminal as a positive response.

Figure 11-45. On-Line Use—Issuer's EDP System

Additional Comments

In the described approach, PIN secrecy depends on SKc* secrecy. If SKc* were known to an opponent, PIN could be derived easily using only PKc (nonsecret and assumed available). Since PIN has relatively few combinations, and assuming SKc* (Figure 11–43) is available, an opponent could

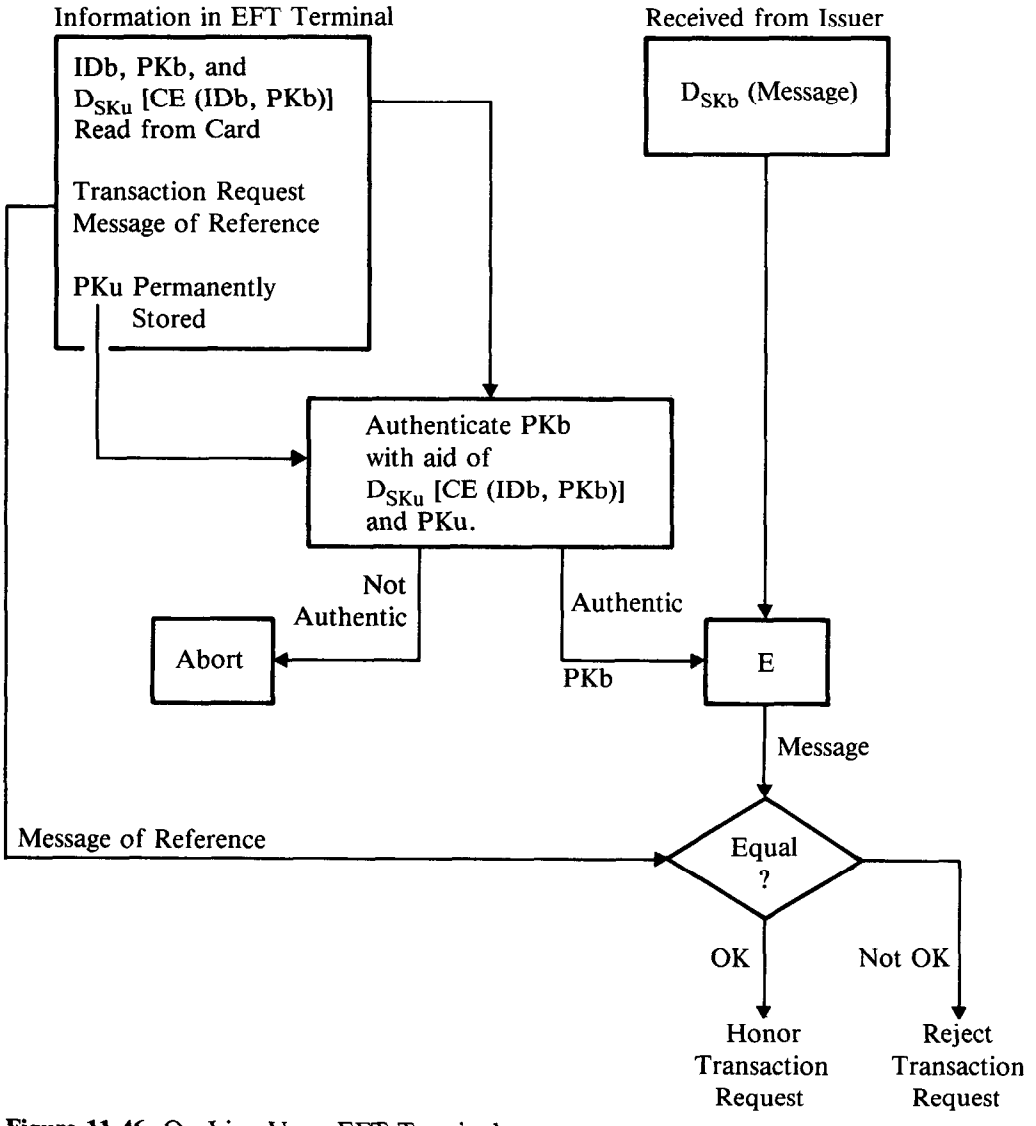


Figure 11-46. On-Line Use—EFT Terminal

enumerate all possible candidates for SKc via the relation SKctrial = SKc* ⊕ PINtrial. SKc is determined by finding the SKctrial satisfying the relation $D_{SKctrial}E_{PKc}(X) = X$.

In an off-line environment, PIN-related information must always be stored on the card, and thus this type of attack will always succeed if card information is available to an opponent. Defense against the attack is therefore based on the high level of data protection afforded by the intelligent secure card. *Hence, strictly speaking requirement 8 (see EFT Security Requirements) cannot be satisfied when a public-key algorithm is used in an off-line environment (i.e., PIN information can always be obtained from card information).*

System User	Intelligent Secure Bank Card	System Nodes: EFT Terminal	Acquirer's Host (Inst. X)	Switch's Host	Issuer's Host (Inst. Y)
		Permanently Installed Keys/Parameters			
none	KP SKc* PKc PKb	PKu	none	none	SKMiss PKMiss
		Keys Used for Generating and Authenticating Digital Signatures (DSG) on the Transaction Request Messages, Mreqs			
none	SKc (Dynamically Generated from SKc* and PIN, e.g., SKc = SKc* ⊕ PIN)	none	none	none	PKc is stored for all members of institution Y.

Keys Used for Generating and Authenticating Digital Signatures on the Transaction Response Messages, Mresps					
none	PKb (Authenticated Using IDb and $D_{SKu}[CE(IDb, PKb)]$, Also Stored on Bank Card)	PKu (Used to Authenticate PKb)	none	none	SKb is defined by institution Y.

Note: Keys associated with personal verification at the issuer (and, perhaps, the switch) are not shown.

Legend:

SKM: Secret host master key
PKM: Public host master key
SKb: Secret institution (bank) key
PKb: Public institution (bank) key
SKc: Secret user (customer) key
SKc*: Secret card parameter
PKc: Public user (customer) key
SKu: Secret universal key
PKu: Public universal key
IDb: Institution (bank) identifier
IDc: User (customer) Identifier

Table 11-15. Keys Defined for the Public Key Approach Using an Intelligent Secure Card

		System Nodes			
System User	Intelligent Secure Bank Card	EFT Terminal	Acquirer's Host	Switch's Host	Issuer's Host
1 Enter PIN and transfer to card via terminal.	2 Generate SKc = SKc* ⊕ PIN. 3 Read IDb,PKb, D_{SKu}[CE(IDb,PKb)] IDc, PKc and D_{SKu}[CE(IDc,PKc)] from card and transfer to terminal. 7 Compute DGSreq with SKc (i.e., D_{SKc}[CE(Mreq)]). 8 Send Mreq and DGSreq to terminal.	4 Authenticate PKb with PKu and PKc with PKb. 5 Read card information and formulate Mreq which includes TODacq and Tterm. 6 Send Mreq to intelligent secure card. 9 Forward received Mreq and DGSreq to acquirer.	10 Forward received Mreq and DGSreq to switch.	11 Forward received Mreq. and DGSreq to issuer.	12 Check received DGSreq with PKc of reference and TODsw of reference. 13 Verify user. 14 Decide if Mreq is to be honored. 15 Formulate Mresp which includes Tterm.

Note: It is assumed that the acquirer periodically sends time-of-day information (TODacq,term) to the terminals in its domain. The TOD stored at the other network host nodes (TODsw at the switch and TODiss at the issuer) is assumed to be equal to TODacq within an allowable range (ΔTOD). The terminal also generates time-variant information (Tterm) which is transmitted to the issuer.

The integers 1–15 in the table show the sequence of steps in the transaction.

Table 11-16. Information Flow from Card to Issuer—Public Key Approach with Intelligent Secure Card

System User	Intelligent Secure Bank Card	System Nodes: EFT Terminal	Acquirer's Host	Switch's Host	Issuer's Host
		20 Check received DGSresp with PKb of reference and Tterm of reference. 21 Decide if Mresp should be accepted or rejected. 22 Notify terminal to process transaction if Mresp is accepted; otherwise, abort transaction request.	19 Forward received Mresp, and DGSresp to terminal.	18 Forward received Mresp, and DGSresp to acquirer.	15 Formulate Mresp which includes Tterm. 16 Generate DGSresp on Mresp using SKb. 17 Send Mresp and DGSresp to intelligent secure card via switch, acquirer, and terminal.

The integers 15–22 in the table show the sequence of steps in the transaction.

Table 11-17. Information Flow from Issuer to Terminal—Public Key Approach with Intelligent Secure Card

In an on-line environment, this exposure can be avoided by decoupling SKc and PIN (i.e., by not making SKc a function of PIN). The relation SKc = SKc* ⊕ PIN was used in the present discussion for the sake of uniformity with the approach described in the preceding section, the PIN/Personal Key/System (Hybrid Key Management) Approach Using an Intelligent Secure Card. For example, a better approach would be to define SKc = SKc* and transmit PIN ‖ RN (i.e., PIN concatenated with a random number generated on the card) encrypted under PKb. This approach requires an additional PIN verification step at the issuer, but has the advantage that knowledge of any one of the parameters SKc or PIN does not reveal information about the other.

CONCLUDING REMARKS

The purpose of this chapter has been to suggest various techniques that may be used for cryptographic authentication in future EFT systems. For the present and near term, it appears that PIN/system key-based EFT systems using magnetic stripe cards will predominate. In such systems, a reasonable level of protection can be achieved with existing technology and current banking practices and standards.

In future systems, a nominal increase in security can be achieved if personal keys are combined with the present PIN/system key designs (e.g., using hybrid key management). However, a significant increase in security is achievable if a hybrid key management is implemented in conjunction with intelligent secure cards.

Finally, further enhancements in security are possible by introducing public-key encryption at financial institutions, thereby providing a means for the issuer to give the acquirer an electronically signed receipt for each transaction request authorized by the issuer.

GLOSSARY

For the analysis of the authentication process, the following quantities are defined.

AP = authentication parameter
CC = communications controller
DES = data encrypting standard
DGS = digital signature
HPC = host processing center
ID = user identifier
KA = authentication key
KC = communication key
KDC = key distribution center
KI = interchange key

Knode = node key
KP = personal cryptographic key
KPG = personal key generating key used to generate KP from ID
KPN = PIN generating key used to generate PIN from ID
KMT = terminal master key
KNC = secondary communication key
KSTR = transaction session key
KT = resident terminal key
KTR = transaction key
MAC = message authentication code
PAC = personal authentication code
PAN = primary account number
PIN = personal identification number
PK = public key in a public-key cryptosystem
PKC = public-key cryptosystem
RN = random number
Tcard = time-variant information generated by bank card
TID = terminal identifier
TOD = time of day
Tterm = time-variant information generated by terminal
SK = secret key in a public-key cryptosystem
TR = transaction request
Rf = reference
Z = initializing vector

The notation $E_K(X) = Y$ defines encipherment of the quantity X under the cipher key K, resulting in ciphertext Y. The notation $D_K(Y) = X$ defines decipherment of Y under cipher key K, resulting in plaintext X.

REFERENCES

1. *Introduction to EFT Security,* Division of Management Systems and Economic Analysis, Federal Deposit Insurance Corporation, Washington, DC (August 1976).
2. Diffie, W. and Hellman, M. E., "New Directions in Cryptography," *IEEE Transactions on Information Theory,* **IT-22**, No. 6, 644–654 (1976).
3. Rivest, R. L., Shamir, A. and Adelman, L., "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM,* **21**, No. 2, 120–126 (1978).
4. Hirsch, P., "French Bring 'Smart' Credit Card to U.S.," *Computerworld,* **16**, No. 43, 1, 8 (October 20, 1980).
5. Orr, W., "The Chip Card is Here, but Where is it Going?," *ABA Banking Journal,* **72**, No. 9, 93–95 (1980).
6. Herbst, N. M. and Liu, C. N., "Automatic Signature Verification Based on Accelerometry," *IBM Journal of Research and Development,* **21**, No. 3, 245–253 (1977).

7. Meyer, C. H., Matyas, S. M., and Lennon, R. E., "Required Cryptographic Authentication Criteria for Electronic Funds Transfer Systems," *Proceedings of the 1981 Symposium on Security and Privacy,* IEEE Computer Society, Oakland, CA, 89–98 (April 1981).
8. Campbell, C. M., Jr., "A Microprocessor-Based Module to Provide Security in Electronic Funds Transfer Systems," *Proceedings COMPCON* **79**, 148–153 (1979).
9. *American National Standard for Personal Identification Number Management and Security, Draft Standard,* American National Standards Institute, Technical Committee X9.A3, Revision 5 (November 5, 1980).
10. Kaufman, D. and Auerbach, K., "A Secure National System for Electronic Funds Transfer," *AFIPS Conference Proceedings 1976 NCC,* **46**, 129–138 (June 1976).
11. Evans, A., Kantrowitz, W., and Weiss, E., "A User Authentication System Not Requiring Secrecy in the Computer," *Communications of the ACM,* **17**, No. 8, 437–442 (1974).
12. Purdy, G. B., "A High Security Log-in Procedure," *Communications of the ACM,* **17**, No. 8, 442–445 (1974).
13. Lennon, R. E. and Matyas, S. M., "Cryptographic Key Distribution Using Composite Keys," *Conference Record, 1978 National Telecommunications Conference,* **2**, 26.1.1–26.1.6 (December 1978).
14. Matyas, S. M. and Meyer, C. H., "Cryptographic Authentication Techniques in Electronic Funds Transfer Systems," *Proceedings of the National Electronics Conference,* **35**, Chicago, 309–314 (October 1981).
15. *PIN Manual: A Guide to the Use of Personal Identification Numbers in Interchange,* MasterCard International, Inc. (formerly Interbank Card Association), New York (1980).
16. Proposed American National Standard X4.16, *Magnetic Stripe Encoding for Financial Transaction Cards,* American National Standards Institute, X4 (Draft, October 1980).
17. Kent, S. T., "Comparison of Some Aspects of Public-Key and Conventional Cryptosystems," *Conference Record of the 1979 International Conference on Communications,* **1**, Boston, 04.3.1–04.3.5 (June 1979).
18. Needham, R. M. and Schroeder, M. D., "Using Encryption for Authentication in Large Networks of Computers," *Communications of the ACM,* **21**, No. 12, 993–999 (1978).

Other Publications of Interest

19. Lennon, R. E. and Matyas, S. M., "Cryptographic PIN Processing in EFT Systems," *Proceedings COMPCON* **79**, 142–147 (September 1979).
20. Meyer, C. H. and Matyas, S. M., "Some Cryptographic Principles of Authentication in Electronic Funds Transfer Systems," *Proceedings of the Seventh Data Security Symposium,* Mexico City, Mexico, 73–88 (October 1981).