FIPS PUB 74

FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION

1981 APRIL 1

U.S. DEPARTMENT OF COMMERCE / National Bureau of Standards

GUIDELINES FOR IMPLEMENTING AND USING THE NBS DATA ENCRYPTION STANDARD

CATEGORY: ADP OPERATIONS
SUBCATEGORY: COMPUTER SECURITY

U.S. DEPARTMENT OF COMMERCE, Malcolm Baldrige, *Secretary*
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Director*

Foreword

The Federal Information Processing Standards Publication Series of the National Bureau of Standards is the official publication relating to standards adopted and promulgated under the provisions of Public Law 89-306 (Brooks Act) and under Part 6 of Title 15, Code of Federal Regulations. These legislative and executive mandates have given the Secretary of Commerce important responsibilities for improving the utilization and management of computers and automatic data processing in the Federal Government. To carry out the Secretary's responsibilities, the NBS, through its Institute for Computer Sciences and Technology, provides leadership, technical guidance and coordination of Government efforts in the development of guidelines and standards in these areas.

Comments concerning Federal Information Processing Standards Publications are welcomed and should be addressed to the Director, Institute for Computer Sciences and Technology, National Bureau of Standards, Washington, DC 20234.

James H. Burrows, *Director*
Institute for Computer Sciences
and Technology

Abstract

The Data Encryption Standard (DES) was published as Federal Information Processing Standards Publication (FIPS PUB) 46 on January 15, 1977 [2]. The DES specifies a cryptographic algorithm for protecting computer data. FIPS PUB 81 [3] defines four modes of operation for the DES which may be employed in a wide variety of applications. These guidelines are to be applied in conjunction with FIPS PUB 46 and FIPS PUB 81 when implementing and using the Data Encryption Standard. They provide information on what encryption is, general guidance on how encryption protects against certain vulnerabilities of computer networks, and specific guidance on the DES modes of operation in data communications applications. When used with the proper administrative procedures and when implemented in accordance with these guidelines, electronic devices performing the encryption and decryption operations of the standard can provide a high level of cryptographic protection to data in computer systems and networks.

Key words: Computer security; cryptography; data integrity; encryption; Federal Information Processing Standards Publication; key distribution; network security; security.

Nat. Bur. Stand. (U.S.), Fed. Info. Process. Stand. Publ. (FIPS PUB) 74, 39 pages.
(1981)
CODEN:FIPPAT

For sale by the National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161.

Federal Information
Processing Standards Publication 74

1981 April 1

ANNOUNCING THE

GUIDELINES FOR IMPLEMENTING AND USING THE NBS DATA ENCRYPTION STANDARD

Federal Information Processing Standards Publications are issued by the National Bureau of Standards pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), Executive Order 11717 (38 FR 12315, dated May 11, 1973), and Part 6 of Title 15 Code of Federal Regulations (CFR).

Name of Guideline: Guidelines for Implementing and Using the NBS Data Encryption Standard (DES).

Category of Guideline: ADP Operations, Computer Security.

Explanation: The selective application of technological and related procedural safeguards is an important responsibility of every Federal organization in providing adequate security to its ADP systems. This publication provides guidelines to be used by Federal organizations when these organizations specify that cryptographic protection is required for sensitive or valuable computer data. Protection of computer data during transmission between electronic components or while in storage may be necessary to maintain the confidentiality and integrity of the information represented by that data. These guidelines are to be applied in conjunction with FIPS PUB 46 and FIPS PUB 81 when implementing and using the Data Encryption Standard.

Approving Authority: U.S. Department of Commerce, National Bureau of Standards (Institute for Computer Sciences and Technology).

Maintenance Agency: U.S. Department of Commerce, National Bureau of Standards (Institute for Computer Sciences and Technology).

Applicability: These guidelines are applicable whenever the DES is used for the cryptographic protection of computer data.

Implementation: These guidelines should be referenced in the formulation of plans by Federal agencies for the encryption of computer data using the DES.

Specifications: Federal Information Processing Standard 74 (FIPS PUB 74), Guidelines for Implementing and Using the NBS Data Encryption Standard (affixed).

Cross Index:

a. FIPS PUB 31, Guidelines to ADP Physical Security and Risk Management.
b. FIPS PUB 39, Glossary for Computer Systems Security.
c. FIPS PUB 41, Computer Security Guidelines for Implementing the Privacy Act of 1974.
d. FIPS PUB 46, Data Encryption Standard.
e. FIPS PUB 48, Guidelines on Evaluation of Techniques for Automated Personal Identification.
f. FIPS PUB 65, Guideline for Automatic Data Processing Risk Analysis.
g. FIPS PUB 81, DES Modes of Operation Standard.

Qualifications: These guidelines provide information which aids in the secure implementation of the DES. In addition it presents the considerations that are necessary when implementing cryptography and key management schemes. Some of the implementations described are not required methods but are for the reader's own information. However, the modes of operation are specified by the DES Modes of Operation Standard (FIPS PUB 81 Cross Index g).

Export Control: Cryptographic devices and technical data regarding them are subject to Federal Government export controls as specified in Title 22, Code of Federal Regulations, Parts 121 through 128. Cryptographic devices implementing these guidelines and technical data regarding them must comply with these Federal regulations.

Patents: Cryptographic equipment implementing these guidelines may be covered by U.S. and foreign patents.

Where to Obtain Copies of the Guideline: Copies of this publication are for sale by the National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161. When ordering, refer to Federal Information Processing Standards Publication 74 (FIPS-PUB-74) and title. When microfiche is desired, this should be specified. Payment may be made by check, money order, or deposit account.

Federal Information Processing Standards Publication 74

1981 April 1

Specifications for

GUIDELINES FOR IMPLEMENTING AND USING THE NBS DATA ENCRYPTION STANDARD

CONTENTS

1. INTRODUCTION

Within the last decade, there has been a vast increase in the accumulation and communication of digital computer data in both the private and public sectors. Much of this information has a significant value, either directly or indirectly, and requires protection. It is common to find data transmissions which constitute monetary transfers of billions of dollars daily. Sensitive information concerning individuals, organizations, and corporate entities is collected by Federal agencies in accordance with statutory requirements and is processed in computer systems. This information requires some type of protection, and cryptographic protection may be specified by the authority responsible for the data. The NBS Data Encryption Standard [2]* must be employed when cryptographic protection is required for unclassified Federal ADP data. The DES Modes of Operation Standard [3] defines the methods or modes in which the DES may be implemented.

The rapid growth of computer data banks increases the potential threats to personal privacy. Since data banks often are accessible from remote computer terminals, there is a threat of easy and unauthorized access to personal information from any place in the data communications system. Such information has typically been scattered in remote locations, controlled under separate auspices, and physically or administratively protected. With a telecommunications network of computer systems, what was previously a laborious job of assembling comprehensive dossiers on individuals may become a simple task. Thus, both valuable and sensitive information require protection against unauthorized disclosure and modification.

Encryption is a tool which may be used in data security applications. It is not a panacea. With improper implementation and use, data encryption may only provide an illusion of security. With inadequate understanding of encryption applications, data encryption could deter the utilization of other needed protection techniques. However, with proper management controls, adequate implementation specifications, and applicable usage guidelines, data encryption will not only aid in protecting data communications but can provide protection for a myriad of specific data processing applications.

2. DATA ENCRYPTION

2.1 What Is Data Encryption?

Data encryption is a process used to hide the true meaning of data. The word "encryption" has been coined from the word "cryptography" which was derived from the ancient Greek words "kryptos" (hidden) and "graphia" (writing). Encryption is the process of transforming text or data into an unintelligible form called cipher. Reversing the process of encryption and transforming the cipher back into its original form is called decryption. Encryption and decryption comprise the science of cryptography as it is applied to the modern computer.

2.2 How Is Data Encryption Achieved?

Data encryption is achieved through the use of an algorithm that transforms data from its intelligible form to cipher. An algorithm is a set of rules or steps for performing a desired operation. An algorithm can be performed by anything that can be taught or programmed to perform a specific and unambiguous set of instructions. Electronic devices which efficiently perform the mathematical steps of the algorithm specified in the Data Encryption Standard (DES) are described in these guidelines.

2.3 Where Should Data Encryption Be Used?

Cryptography (encryption) has historically been used to protect sensitive information during communication. It can be used for protecting computer data transmitted between terminals and computers or between computers. Data is encrypted before transmission and decrypted after it is received. The algorithm used to decrypt the received cipher must be the inverse of the algorithm

*Numbers in brackets indicate references given in section 9.

used to encrypt the transmitted data. In general, a device used to transmit and receive data would contain algorithms for both encryption and decryption.

Encryption can be used between data processing machines and data storage devices such as magnetic tape and magnetic disk. In this application, the data is encrypted before it is written on the storage device and decrypted before it is subsequently read. Data is stored in its cipher form and transformed to plaintext only when it is to be processed within the computer.

Encryption can be used to authenticate the identities of users, terminals, and computers of a data processing system. Passwords have historically been used to differentiate between friend and foe during times of war. Knowledge of the secret password was accepted as authenticating the identity of friends. Unique identification was not necessary and the password was changed for each mission. The DES uses a key, similar to a password, which must be supplied to each group of users of the algorithm. Having the correct key authenticates an individual to a data processing system.

In a similar manner a terminal or a computer may be authenticated as an authorized device of a data processing system. Supplying the correct key to a DES device when requested by the authorization system can authenticate a terminal associated with the device. This authorization system may be a special program or a special computer system which has been established to control access to the resources and data of the overall system. The authorization system must be initialized with the identities and the authentication keys of all authorized users and devices of the system. This system will issue a challenge for proper identification whenever a device or individual wishes to access the system. Similar challenge-response password systems are currently in use for computer user authentication. When combined with data encryption technology, authorization systems can authenticate the claimed identities of users and devices without compromising the passwords or keys by transmitting them through the system.

2.4 When Should Data Encryption Be Used?

Data encryption should be used whenever it is the most cost effective method available to protect the confidentiality or integrity of the data. Confidentiality refers to the accidental or intentional disclosure of data to an unauthorized individual. Integrity refers to data which has not been exposed to accidental or malicious alteration or destruction. Encryption of data prevents unauthorized recipients of the cipher from interpreting its meaning. Encryption can also prevent unauthorized individuals from manipulating the cipher in such a way that the original data is changed in a predetermined manner. To be effective, encryption must cost less than the expected loss (risk) if the protection were not provided. Computation or estimation of costs and risks and the decision to employ cryptographic protection are management functions of the authority responsible for the data. Risk analysis information may be found in FIPS PUB 65 [6].

2.5 Why Is a Data Encryption Standard Necessary?

A data encryption standard is needed to protect sensitive or valuable data within Federal computer systems and networks. Effective sharing of computational facilities and controlled sharing of computer data have been retarded pending development of adequate protection measures. Data encryption techniques are needed for controlling access to sensitive data in multiuser computer systems, for protecting the integrity of transactions in national and international monetary transfer systems, for disguising sensitive data during transmission, and for authenticating the users and devices of distributed computer systems and networks. A myriad of different encryption algorithms would result in a fundamental incompatibility of data communications equipment. Research and development in cryptographic algorithms are difficult areas; redundant and unusable results often occur. Support of several standards would incur a higher cost for the Federal Government. The Data Encryption Standard provides a basic method for more effective computer utilization and a high level of protection for computer data.

The need to interface with the data processing facilities of Federal agencies may make it desirable that private organizations have and be able to use the DES. Since its adoption as a Federal Standard, the DES algorithm has been approved as a standard by the American National Standards Institute [1] and recommended for use by the American Bankers Association [7].

2.6 What Are the Requirements of a DES?

An encryption algorithm must satisfy the following requirements in order to be acceptable as a Federal standard:

1. It must provide a high level of security.
2. It must be completely specified and easy to understand.
3. The security provided by the algorithm must not be based upon the secrecy of the algorithm.
4. It must be available to all users and suppliers.
5. It must be adaptable for use in diverse applications.
6. It must be economical to implement in electronic devices and be efficient to use.
7. It must be amenable to validation.
8. It must be exportable.

The algorithm described in FIPS PUB 46 satisfies all these requirements.

2.7 What Role Has NBS Played in the DES?

NBS has the responsibility for developing Federal Information Processing Standards through Public Law 89-306 and Executive Order 11717. The Institute for Computer Sciences and Technology (ICST) has the responsibility within the NBS to recommend and coordinate standards and guidelines for improved computer utilization and information processing within the Federal Government, as well as for developing the technology needed to support these standards activities. Because of the unavailability of general cryptographic technology outside the national security arena, and because security provisions, including encryption, were needed in unclassified applications involving Federal Government computer systems, NBS initiated a computer security program in 1973 which included the development of a standard for computer data encryption. Since Federal standards impact on the private sector, NBS solicited the interest and cooperation of industry and user communities in this work.

In May 1973, NBS published a notice in the Federal Register (38FR12763) inviting the submission of data encryption algorithms and techniques which might be considered for use in a Federal standard. The responses showed considerable interest in and need for such protection. A second Federal Register solicitation (39FR30961) in August 1974 reiterated the former solicitation and provided a further opportunity to submit data encryption algorithms. Subsequent to the closing of the solicitation, algorithms submitted to NBS were evaluated for technical feasibility as a Federal standard. This document discusses the algorithm which satisfied the requirements of a data encryption standard. It was developed by the International Business Machines Corporation (IBM). IBM made the specifications of the algorithm available to NBS for publication as a Federal Information Processing Standard (FIPS) and has provided nondiscriminatory and royalty free licensing procedures for building electronic devices which implement the algorithm. At the request of NBS, the National Security Agency (NSA) conducted an exhaustive technical analysis of the DES. No shortcuts or secret solutions were found and, as a result, NSA confirmed the soundness of the DES's encryption principle and its suitability to protect unclassified Federal data [8]. NBS published the algorithm in the Federal Register in March 1975 (40FR12067) for public comment and published the proposed standard in the Federal Register in August 1975 (40FR32395) for public comment. In January 1977 the algorithm was published as a Federal standard, FIPS PUB 46 [2].

3. DATA ENCRYPTION METHODS

3.1 Basic Methods

Encryption is a transformation of data from its original, intelligible form to an unintelligible cipher form. Two basic transformations may be used: permutation and substitution. Permutation changes the order of the individual symbols comprising the data. In a substitution transformation, the symbols themselves are replaced by other symbols. During permutation the symbols retain their identities but lose their positions. During substitution the symbols retain their positions but lose their original identities.

The set of rules for a particular transformation is expressed in an algorithm. Basic transformations may be combined to form a complex transformation. In a computer system the symbols of the data are groups of one or more binary digits ("1"s and "0"s) called bits. A group of bits is called a byte. In computer applications the encryption transformation of permutation reorders the bits of the data. The encryption transformation of substitution replaces one bit with another or one byte with another.

3.2 Encoding and Enciphering

Coding or encoding, in a noncryptographic sense, is often used to mean changing from one intelligible form to another. The American Standard Code for Information Interchange (ASCII) and Morse code are examples of noncryptographic codes. Reducing the length of a data element without removing any of its information content is called compression. Expanding the length of a data element is usually done for error detection and correction purposes. Even though the form of the data is changed, no attempt is made to prevent unauthorized decoding. The remainder of this subsection will apply to cryptographic codes that are used to disguise plaintext information and thereby prevent the disclosure of the information to unauthorized parties.

Within basic encryption transformation classes, encoding is usually distinguished from enciphering. A code is a correspondence between codewords and data elements. A data element may be a letter, a syllable, a word, a phrase, or a special symbol. Codebooks generally consist of two sections: one alphabetized on the data elements for use in encoding and the second alphabetized on the code words for use in decoding. Encoding consists of looking up every data element of a message to be transmitted and substituting its codeword equivalent to produce the encoded message. Decoding consists of finding the received codewords in the codebook and replacing them with their equivalent data elements, thus reconstructing the original message.

A codebook may be automated to perform the encode and decode functions as just described or an algorithm may be used to automatically encode and decode without looking up the corresponding values in tables. The latter method is preferred when automation is feasible because encoding and decoding can be performed rapidly, by simply computing the code equivalent each time it is needed rather than storing an enormously large codebook.

Enciphering consists of an algorithmic computation involving the data itself. The original plaintext data may either be used directly in the computation or may be combined with the results of the computation to form cipher. The cipher that results from such a transformation is generally the same length as the original data that is enciphered.

Ciphers may be thought of as operating on data elements of fixed length and codes as operating on data elements of variable length. Another useful distinction is that a code typically operates on linguistic entities (words) while a cipher operates on syntactic entities (letters or groups of letters). In general computer applications, bits or bytes are used in data encryption algorithms without regard to their linguistic content. Thus the computer encryption transformation of a fixed number of bits or bytes is generally called enciphering.

3.3 Block Ciphers

A cipher that is produced by simultaneously transforming a group of message bits into a group of cipher bits is called a block cipher. In general, the groups are the same size.

3.4 Product Ciphers

Combining the basic transformations of permutation and substitution produces a complex transformation termed a product cipher. The characteristics of a product cipher are discussed in "Cryptography and Computer Privacy" [4]. If permutation and substitution operations are applied to a block of data, the resulting cipher is called a block product cipher.

3.5 Recirculating Block Product Cipher

A block product cipher may be constructed by using a permutation operation and a substitution operation alternately and recirculating the output of one pair of operations back into the input for some number of iterations. Each iteration is called a round. A cipher produced in this way is termed a recirculating block product cipher. If a recirculating block product cipher is properly constructed with an unknown key, then the alteration of a single bit of the plaintext block will unpredictably alter each bit of the ciphertext block. Altering a bit of the ciphertext will also result in an unpredictable change to the plaintext block after decryption.

3.6 Characteristics of the DES Algorithm

The DES algorithm is a recirculating, 64-bit, block product cipher whose security is based on a secret key. DES keys are 64-bit binary vectors consisting of 56 independent information bits and eight parity bits. The parity bits are reserved for error detection purposes and are not used by the encryption algorithm. The 56 information bits are used by the enciphering and deciphering operations and are referred to as the *active key.* Active keys are generated (selected at random from all possible keys) by each group of authorized users of a particular computer system or set of data. Each user should understand that the key must be protected and that any compromise of the key will compromise all data and resources protected by that key.

In the encryption computation the 64-bit data input is divided into two halves each consisting of 32 bits. One half is used as input to a complex nonlinear function, and the result is exclusive OR'ed to the other half. (See fig. 5.1.) After one iteration, or round, the two halves of the data are swapped and the operation is performed again. The DES algorithm uses 16 rounds to produce a recirculating block product cipher. The cipher produced by the algorithm displays no correlation to the input. Every bit of the output depends on every bit of the input and on every bit of the active key.

The security provided by the DES algorithm is based on the fact that, if the key is unknown, an unauthorized recipient of encrypted data, knowing some of the matching input data, must perform an unacceptable effort to decipher other encrypted data or recover the key. Even having all but one bit of the key correct does not result in intelligible data.

The only known way of obtaining the key with certainty is by obtaining matched ciphertext and plaintext and then by exhaustively testing keys by enciphering the known plaintext with each key and comparing the result with the known ciphertext. Since 56 independent bits are used in a DES key, 2^{56} such tests are required to guarantee finding a particular key. The expected number of tests to recover the correct key is 2^{55}. At one microsecond per test 1142 years would be required. Under certain conditions (not only knowing matched plaintext and ciphertext but also the complement of the plaintext and the resulting ciphertext) the expected effort would be reduced to 571 years. The possibility of 2^{56} keys (approximately 70 quadrillion) makes the guessing or computing of any particular key very unlikely given that the guidelines for generating and protecting a key provided in this publication are followed. Of course, one can always reduce the time required to exhaust any cryptoalgorithm by having several devices working in parallel. Time is reduced but initial expenses are increased.

An important characteristic of the DES algorithm is its flexibility for usage in various data processing applications. Each cipher block is independent of all others allowing encryption or decryption of a single block in a message or data structure. Random access to encrypted data is therefore possible. The algorithm may be used in this straightforward way to form a block cipher or alternatively used with chaining in which the output of the algorithm depends on previous results of the algorithm. The first technique is called the Electronic Codebook (ECB) mode and the chaining technique has two examples (discussed in these guidelines) called the Cipher Block Chaining (CBC) mode and the Cipher Feedback (CFB) mode. In addition, DES may be used in the Output Feedback (OFB) mode to generate a pseudorandom stream of bits which is exclusive OR'ed to the plaintext bits to form cipher. These will be discussed in 5.3.

The DES algorithm is mathematically a one-to-one mapping of the 2^{64} possible input blocks onto all 2^{64} possible output blocks. Since there are 2^{56} possible active keys, there are 2^{56} possible mappings. Selecting one key selects one of the mappings.

The input to the algorithm is under complete specification of the designer of the cryptographic system and the user of the system. Any pattern of 64 bits is acceptable to the algorithm. The format of a data block may be defined for each application. In the ECB mode, the subfields of each block may be defined to include one or more of the following: a block sequence number, the block sequence number of the last block received from the transmitter, error detecting/correcting codes, control information, date and time information, user or terminal authentication information, or a field in which random data is placed to ensure that identical data fields in different input blocks will result in different cipher blocks. It is recommended that no more than 16 bits be used for known constant values. For example, the same 32-bit terminal identification value should not be used in every block. If it is desired that data blocks in the ECB mode display a sequence dependency, a portion of the last sent or last received block may be incorporated into the block, either as a subfield or exclusive OR'ed to the block itself.

The DES algorithm is composed of two parts: the enciphering (encryption) operation and the deciphering (decryption) operation. The algorithms are functionally identical except that the selected portion of the key used for rounds 1,2,...,16 during the encryption operation are used in the order 16,15,...,1 for the decryption operation. The algorithm uses two 28-bit registers called C and D to hold the 56-bit active key. The key schedule of the algorithm circularly shifts the C and D registers independently, left for encryption and right for decryption. (See fig. 5.3 and table 5.4.) If the bits of the C register are all zeros or all ones (after Permuted Choice 1 is applied to the key) and the bits of the D register are all zeros or all ones, then decryption is identical to encryption. This occurs for four known keys: (0101010101010101), (FEFEFEFEFEFEFEFE), (1F1F1F1F0E0E0E0E), and (E0E0E0E0F1F1F1F1). [Note that the parity bits of the key are set so that each 8-bit byte has odd parity.] It is likely that, in all other cases, data encrypted twice with the same key will not result in plaintext (the original, intelligible data form). This characteristic is beneficial in some data processing applications in that several levels of encipherment can be utilized in a computer network even though some of the keys used could be the same. If an algorithm is its own inverse, then an even number of encryptions under the same key will result in plaintext.

There are certain keys such that for each key K there exists a key K′ for which encryption with K is identical to decryption with K′ and vise versa. K and K′ are called dual keys. Keys with duals were found by examining the equations which must hold in order for two keys to have reversed key schedules. Keys having duals are keys which produce all zeros, all ones, or alternating zero-one patterns in the C and D registers after Permuted Choice 1 has operated on the key. (See fig. 5.3.) These keys are listed below.

	KEY	DUAL
1.	E001E001F101F101	01E001E001F101F1
2.	FE1FFE1FFE0EFE0E	1FFE1FFE0EFE0EFE
3.	E01FE01FF10EF10E	1FE01FE00EF10EF1
4.	01FE01FE01FE01FE	FE01FE01FE01FE01
5.	011F011F010E010E	1F011F010E010E01
6.	E0FEE0FEF1FEF1FE	FEE0FEE0FEF1FEF1
7.	0101010101010101	0101010101010101
8.	FEFEFEFEFEFEFEFE	FEFEFEFEFEFEFEFE
9.	E0E0E0E0F1F1F1F1	E0E0E0E0F1F1F1F1
10.	1F1F1F1F0E0E0E0E	1F1F1F1F0E0E0E0E

The first 6 keys have duals different than themselves, hence each is both a key and a dual giving 12 keys with duals. The last four keys equal their duals, and are called self-dual keys. These are the four previously discussed keys for which double encryption equals no encryption, i.e., the identity mapping. The dual of a key (which has a dual) is formed by dividing the key into two halves of eight hexidecimal characters each and circular shifting each half by two characters. No other keys are known to exist which have duals.

Data may be decrypted first and then encrypted (rather than encrypted and then decrypted) and result in plaintext. Plaintext may be encrypted several times and then decrypted the same number of times with the same key and result in plaintext. Similarly, data may be encrypted successively by different keys and decrypted sucessively by the same keys to produce the original data, if the decryption operations are performed in the proper (inverse) order. If $D_1(E_1(P)) = P$ is read "Encrypting plaintext with Key 1 and then decrypting the result with Key 1 yields the plaintext," then the following are true:

1. $E_1(D_1(P)) = P$
2. $E_1(E_1(P)) = P$ for self-dual keys
3. $D_1(D_1(E_1(E_1(P)))) = P$
4. $E_1(E_1(D_1(D_1(P)))) = P$
5. $D_1(D_2(E_2(E_1(P)))) = P$
6. $D_1(D_2(...(D_j(E_j...(E_2(E_1(P)...) = P$
7. $E_1(E_2(...(E_j(D_j...(D_2(D_1(P)...) = P$
8. $E_2(E_1(P)) = P$ for dual keys
9. $D_2(D_1(P)) = P$ for dual keys

but in general the following is not true:

10. $D_2(D_1(E_2(E_1(P)))) = P.$

4. SECURITY THREATS REDUCED THROUGH ENCRYPTION

Encryption may be implemented in a computer system in order to combat several possible threats to the security of computer data. These threats are generally categorized as transmission threats and storage threats. Security against these threats is generally termed communication security (COMSEC) or file security (FILESEC). The DES algorithm can be used in both applications but the key will be handled differently. The generation, distribution, protection, and destruction of cryptographic keys are generically referred to as key management and are discussed in section 6.

4.1 Transmission Threats

Encryption can be used to prevent the disclosure of data and to detect the modification of transmitted data. Encryption will not combat the threats of accidental or deliberate destruction. Encrypted data can be lost or destroyed as easily as unencrypted data. Adequate backup facilities or copies must be provided to recover from the destruction of either encrypted or unencrypted data. In addition, destruction or loss of the key used to encrypt data is equivalent to the loss or destruction of the data itself.

The following is a list of threats that are countered with the encryption of transmitted data:

1. **Spoofing:** Spoofing is the threat of accepting a false claim of identity. Spoofing by a computer system penetrator is a serious threat at many places in a computer system. The computer's data communication system is especially vulnerable to spoofing. The identities of terminals, computers, and users can often be simulated so that the receiving device cannot discern a true identity from a falsely claimed identity. Data encryption can be used for authentication by requiring that a unique encryption key be associated with each identity. Successful communication using this key mutually authenticates the holders of the key (provided that the key has not been compromised) and thus prevents spoofing. If the key is not known, false messages cannot be correctly generated and entered into the system and hence message spoofing is prevented.

2. **Misrouting:** The threat of misrouting is directly proportional to the complexity of the communication system and inversely proportional to the reliability of its components. A simple message routing indicator scheme combined with encryption of the routing indicator may be used to detect misrouting, but prevention can only be accomplished with dedicated lines and permanent connections. In any but geographically local systems, the prevention of misrouting is not economically feasible. However, data encryption can prevent the unauthorized use of misrouted data.

3. **Passive Wiretapping (Monitoring):** Monitoring of messages during data transmission can occur all along the transmission path in any of several ways. Wiretapping or radio reception of the transmitted data are the most common methods. The transmission is not delayed or altered, only monitored or copied. This threat is difficult to combat in any way other than physically protecting the transmission path or encrypting the data. Plaintext is also vulnerable to monitoring due to radiation, conduction, and acoustic pickup during input and output operations. These threats are prevalent in high voltage CRT terminals, electrically connected devices, and mechanical printing or punching devices. Encryption protects the plaintext from disclosure. The encryption devices should be designed to be an integral part of the original source equipment and the final destination equipment whenever possible. The data encryption devices themselves must be physically protected and designed to minimize electronic emanations.

4. **Active Wiretapping:** With this type of communication threat the communication line is broken, a high speed receiver-transmitter is installed, and the intercepted data is retransmitted unchanged until a special "looked for" event causes the tapping mechanism to modify the data so as to have false information accepted as valid. Communications will be slightly delayed while the data is being modified but this delay is often not detectable because other variable length delays are already in the communication system. Encryption prevents the penetrator from intelligently modifying the cipher so that the decrypted plaintext is ungarbled (i.e., readable and acceptable). Special precautions must be utilized to prevent either the playback threat or the substitution threat. The former consists simply of copying a valid encrypted message and playing it back (retransmitting it) to the unsuspecting receiver. If the key has not been changed, the receiver will correctly decrypt the message and may accept it. For certain types of messages (funds deposits, merchandise orders, etc.), this could have disastrous results. The substitution threat consists of replacing blocks or characters of ciphertext with other blocks or characters without actually deciphering the data or having the key. The perpetrator substitutes the cipher of known plaintext. This can be accomplished in the block mode if each block is totally independent from all others, and no other block or message authentication system is used.

4.2 Storage Threats

In addition to combatting threats to computer data security during transmission among terminals and computers, the DES may be used effectively for protecting computer data during storage, but the system implementation will be different in the two cases. In the transmission case, the cryptographic key must be available at the two participating locations simultaneously and may be destroyed when that transmission is complete. In the storage case, the key need be at only one location but must be retained for reuse when the data is to be retrieved and used. The computer system or the user must be able to provide the key at the appropriate place and at the appropriate time.

The following is a list of threats that are countered with the encryption of stored data:

1. **Theft:** Encryption of stored computer data provides protection against the disclosure of stolen data. Data may be stolen from on-line devices (disks, mass storage devices, etc.) by unauthorized access, or from off-line devices (magnetic tape, cards, disk packs, etc.) by physically removing the device and reading it on another computer system. In addition if there is a threat of a computer data storage facility or a computer center being taken over by force, bulk encryption of all data using a common key which is easily erased from the encryption device effectively renders the data unreadable and unusable by destroying the key. This key must be kept in a physically secure location (safe, etc.) so that it may be reentered into the encryption device when the facility has been made secure again. User controlled encryption of private data files renders the data unreadable to other system users.

2. **Residue:** Data that is left on magnetic media and not erased after it is no longer needed is called residue. Erasing computer data on magnetic storage media may be a very time consuming process. Overwriting data which is to be discarded in a shared system can use a significant amount of input and output time if done as standard practice. Data recovered by simply reading discarded data that was not destroyed is considered to be "scavenged." If sensitive data is always stored on the media in an encrypted form, tapes and disk packs may be returned to their supplier when no longer

needed or the "scratched" data tapes may be reused without erasing. Merely destroying the key precludes use of the data. System failures during the erasing of magnetic media are no longer a concern if the media are encrypted. Encryption of stored data with the user's private key obviates the need for clearing temporary storage after use.

3. **Remanence:** Remanence is the magnetic flux remaining in a magnetic substance after the magnetic force has been removed. In some magnetic storage media, data stored for a long period of time on the media can remain at a lower signal intensity level even after the media have been erased. Encryption of all sensitive data stored on such media removes this threat and such storage media may be released for general usage rather than destroyed. It should be noted that for unclassified computer data, this is a very insignificant threat and encryption should not be justified for this reason alone.

4. **Addressing Failure:** Random access magnetic storage media have a physical addressing mechanism which positions the data under the reading heads and transfers the data. Software data access methods generally have a complex data structure associated with the stored data to optimize access to it. Both of these mechanisms have a small, but nonzero, probability of failure. Encrypting the data by combining the location of the data with the key can prevent accidental reading of the wrong data. Applications of this type in the system will depend greatly on the implementation of the DES device in the proper place in the system architecture.

5. IMPLEMENTATION OF THE ALGORITHM

A cryptographic system comprises many components, e.g., a cryptographic algorithm, a key management system, an applications interface, a maintenance procedure, and a user training program. Section 5 discusses the basic implementation of the DES algorithm in electronic devices and methods of interfacing it to particular applications.

A hardware implementation of the DES algorithm is described and a software interface is outlined. The device performs the mathematical transformation described in the DES. The software interface provides control functions to the device, receives status information from the device, and implements the Cipher Block Chaining (CBC), Cipher Feedback (CFB), or Output Feedback (OFB) modes of operation discussed in 5.3. This approach provides a flexible mechanism for use in many data processing environments, but it may not provide adequate efficiency or security in all cases. For example, special hardware may be required for very high speed or error sensitive applications.

5.1 Basic Implementation

Basic implementation refers to the embodiment of the DES algorithm. FIPS PUB 46 specifies that electronic hardware is required for the basic implementation.

5.1.1 Electronic Devices

The NBS DES algorithm specifies the encryption of 64 bits of data into a 64-bit cipher based on a 56-bit active key, and the decryption of a 64-bit cipher block into a 64-bit data block based on a 56-bit active key. The steps and the tables of the algorithm are completely specified and no options to the basic algorithm are contained in the DES. However, there are many ways to incorporate the algorithm into a cryptographic system and the implementation used will depend on the application. A recommended method is to implement the basic DES algorithm in a special purpose electronic device and then control it from a programmable computer (e.g., a microprocessor). Some of the issues involved in the application of the DES are: how is the input formatted, is the data itself or a different 64-bit value used as input to the algorithm, how is the key generated and distributed, and how often is the key changed?

Implementation of the DES algorithm in special purpose electronic devices provides the following economic and security benefits:

1. Efficiency of algorithm operation is much higher in specialized electronic devices.
2. Basic implementation of the algorithm in specialized LSI electronic devices which can be used in many applications and environments should result in cost savings to the user through high volume production.
3. Functional operation of the device may be tested and validated independently of the environment in which it is used.
4. An encryption key may be entered directly into the device without appearing elsewhere in the computer system.
5. Unauthorized modification of the algorithm is very difficult in such a device.
6. Independent devices may encipher the data simultaneously and the output may be tested before the cipher is transmitted.
7. The control and data paths, to and from the device, may be controlled and monitored.

For these reasons, implementation in special purpose devices (electronic devices or read only memories) is required by FIPS PUB 46.

5.1.2 Basic Implementation Control Functions

Several control functions must be available in the basic implementation of the algorithm. The actual controls that are provided in an electronic implementation will vary according to the technology used and the packaging available. The following discussion presents a set of controls designed and implemented by the NBS technical staff in two identical hardware devices being used in the NBS Data Encryption Testbed. The two DES test units were designed and built in medium scale integration (MSI) TTL logic. The Data Encryption Testbed based on these units is described in 5.5.

Control lines are used to provide control signals to the DES device; status lines are used to monitor the condition of the DES device; data lines are used to input and output the plain and enciphered data. In the NBS implementation, eight data input lines and eight data output lines are used. Both the data and key needed by the algorithm are entered via the data lines in 8-bit bytes. Similarly, when the encryption or decryption operation is complete, the plaintext or ciphertext is sequentially read from the device in 8-bit bytes.

CONTROL LINES

1. Data/Key—Enter data (0) or enter key (1).
2. Encipher/Decipher—Encipher data (0) or decipher data (1).
3. Plain/Cipher—Enter plain key (0) or enter enciphered key (1).
4. Reset except key (1)—Clears all internal registers except key register.
5. Reset (1)—Clears all internal registers.
6. Input ready (1)—Input lines are ready to be read into the DES device.
7. Output accepted (1)—Output lines have been read by the controlling device.

STATUS LINES

1. Busy (1)—Device is busy and cannot input or output.
2. Parity error (1)—Key being entered has a parity error.
3. Control error (1)—The control last given to the DES is incorrect.
4. Output ready (1)—Output lines are ready to be read.
5. Input accepted (1)—Input lines have been read.

The NBS implementation is designed for use as an encryption testbed device and for use as a DES validation device. The testbed has been designed to develop control procedures for DES devices in various applications and for different communications protocols. For demonstration purposes, digital displays of data, control and status are provided on the front panel of the device. Two units have been constructed to provide a test facility for data communications. The NBS DES device is capable of either enciphering or deciphering a block of data in nine microseconds, once the data has been loaded. In addition, it takes a minimum of twenty microseconds to either load or unload the device.

A separate unit was built to operate the DES device manually. This unit has two sets of 16 rotary thumbwheel switches: 16 for the data and 16 for the key. Each switch has 16 positions: hexadecimal digits 0-9 and A-F. These allow 64-bit entry of key, plaintext, and cipher into the DES device. The test unit also contains control buttons and binary switches to provide the control signals necessary for operating the DES. The test unit is only used for off-line demonstrations of the DES devices and for maintenance testing.

5.2 Secondary Implementation

The secondary implementation consists of the control mechanisms which govern the operation of the basic implementation. It is also responsible for implementing the CBC, CFB, and OFB modes of operation which are discussed in section 5.3. Each NBS DES device is connected to a microprocessor computer with a multiline cable as a parallel interface. This interface contains the data input and output lines, the control lines, and the status lines. The DES device input lines and the control lines are connected to output ports of the microprocessor. The DES device output lines and the status lines are connected to input ports of the microprocessor. The DES device looks like a simple input-output device to the microprocessor.

5.2.1 Secondary Implementation Control Functions

A DES device must be contained in a control environment that conforms to the requirements of a particular application. This environment includes electrical power, control and status lines, data lines for input and output, and the capability of providing other special services that will depend on the application. One such service is to collect and enter the data into the DES primary device in accordance with the data format and communication protocol specifications. Another service is to receive the output from the DES device and then present it to the communication system.

In any encrypted communications application other than link encryption (i.e., cryptographic protection of a communication line or path having no intermediary nodes), addressing and related control information must be available in an unencrypted form. Separating sensitive information from control information is a very crucial security task of the secondary device.

5.2.2 Error Handling

Errors associated with the primary encryption device should be detected and handled by the secondary device. Physical tampering detectors (vibration or intrusion sensors) may be used to detect physical tampering or unauthorized access to the encryption unit. Sensors which detect abnormal changes in the electrical power or the temperature may be used to monitor physical environment changes which could cause a security problem. However, the major requirement for error detection or correction involves the application itself. The type of error control utilized will depend on the sensitivity of the data and the application. The method selected may range from no error handling capability for some systems to full redundancy of encryption devices in other systems. Errors may be ignored when detected or the entire system may be immediately shutdown. Errors which could compromise the plaintext or key should never be ignored.

5.3 Modes of Operation

The DES algorithm specifies a mathematical transformation of a 64-bit input block to a 64-bit output block using a key. Specific examples of this transformation are given in NBS Special Publication 500-20 [5]. $E_K(I) = O$ and $D_K(O) = I$ are read "Enciphering the input I using key K results in output O" and "Deciphering the output O using key K results in input I." Given the same I and K, the same O always results. Likewise, given the same O and K, the same I results.

If the input at time t is called I_t, then the output is called O_t. A sequence of input blocks to the DES may be denoted as $I_1, I_2, I_3, \ldots, I_n$. The outputs are similarly denoted as $O_1, O_2, O_3, \ldots, O_n$.

The DES specifies only the functions E and D. Other considerations will define the input and how the output is used. Many different possibilities exist but the application generally dictates which ones are feasible. In order to provide compatibility between devices which are able to communicate, four modes of operation are specified in FIPS PUB 81.

5.3.1 The Electronic Codebook (ECB) Mode

The simplest mode of operation, the Electronic Codebook (ECB), is the DES algorithm specified in FIPS PUB 46. The ECB mode is shown in figures 5.1 through 5.3, and tables 5.1 through 5.4. In the ECB mode of operation, the algorithm is independent of time and is called a memoryless system. Given the same data and the same key, the resultant cipher will always be the same. This characteristic should be considered when designing a cryptographic system using the ECB mode. The output block O_t is not dependent on any of the previous inputs, $I_1, I_2, \ldots, I_{t-1}$. It is important to note that the full 64-bits of the O_t must be available in order to obtain the original input I_t.

A general guideline for using the DES in this mode is that all possible inputs should be allowed and used whenever possible. Since the security of the data in this mode is based on the number of inputs in the code book, this number should be maximized whenever possible. In particular this mode should never be used for enciphering single characters (e.g., 8-bit ASCII characters by entering them in a fixed 8-bit position and filling the other 56 bits with a fixed number). Two to the 64th inputs are possible in this mode and as large a subset as feasible should be used. Random information should be used to pad small blocks and the random information discarded when the block is deciphered.

Data should be entered into the input register so that the first character of input appears on the left, the second character to the right of it, etc., and the last character on the far right. Using shift register technology, the characters should enter on the right and be shifted left until the register is full. Similarly, the output of the DES should be taken from left to right when being transmitted or stored in character serial mode. Using shift register technology, the characters should exit from the left and the register shifted left until the register is empty.

5.3.2 The Cipher Block Chaining (CBC) Mode

A method of using the DES algorithm in which the blocks of cipher are chained together is called the Cipher Block Chaining (CBC) mode. Figure 5.4 demonstrates how the CBC mode is used to encrypt a message. The input to the DES at time t is defined to be the exclusive or (represented by $\oplus$) of the data at time t and the cipher at time t–1. The cipher at time 0 is defined to be a quantity called the initialization vector or IV. The CBC mode requires complete blocks of 64 bits until the final block is to be enciphered.

The final (terminal) data block of a message or record may not contain exactly 64 bits when processing in the CBC mode. When this occurs, either the terminal block must be padded to 64 bits or the terminal block must be enciphered in a way that yields the same number of bits as the input. The first technique is called padding and the second is called truncation.

When a sequence of characters is being enciphered and the terminal block contains less than the maximum number of characters (e.g., eight in the case of 8-bit characters), then padding may be used to format the final input block in the following way. Suppose P padding characters are needed to fill out the block. If P equals one, the character representing the number one should be put in the last byte position. If P is greater than one, the character representing the number P should be put in the last byte and zeros should be put in the remaining P–1 byte positions. (See fig. 5.4.) In most coding schemes, the last three bits of the character representing a digit are the same as the binary representation of the digit (e.g., the ASCII representation of the character 4 is a hexadecimal 34). One bit may be used in the header block of a message packet to signify a padded message (i.e., that the final block of the packet is padded) or some other method must be devised.

Truncation may be used in the CBC mode when the number of cipher bits must be the same as the number of input bits. It may be necessary that an enciphered tape contain the same number of records and the same number of characters per record as the unenciphered tape. This requirement also occurs in some message switching systems in which the record length is fixed. In these cases the following method can be used to encipher the terminal block which does not contain 64 bits.

The short terminal block is enciphered by encrypting the previous cipher block in the ECB mode and exclusive OR'ing the result to the terminal data block. (See fig. 5.5.) The receiver must detect the short cipher block and perform the same operation, i.e., encrypt the previous complete cipher block and perform the exclusive OR operation to obtain the original plaintext block. If a short terminal block contains B bits, then the leftmost B bits of the enciphered cipher block are used. This technique normally provides adequate security for the final block, but it should be noted that if the last B bits of plaintext are known to an active wiretapper, he or she may alter the last B bits of cipher so that they will decrypt to any desired plaintext. This is because, if only the last B bits are altered, the same value will be exclusive OR'ed to the short cipher block upon decryption.

One or more bit errors within a single cipher block will affect the decryption of two blocks (the block in which the error occurs and the succeeding block). If the errors occur in the t^{th} cipher block, then each bit of the t^{th} plaintext block will have a average error rate of 50 percent. The $(t+1)^{st}$ plaintext block will have only those bits in error which correspond directly to the cipher bits in error, and the $(t+2)^{nd}$ plaintext block will be correctly decrypted. Thus, the CBC mode synchronizes itself one block after the error.

5.3.3 The Cipher Feedback (CFB) Mode

The Cipher Feedback (CFB) mode of operation may be used in applications which require chaining to prevent substitution or where blocks of 64 bits cannot be used efficiently. Most computer data that are to be transmitted or stored are coded in 6- to 8-bit codes. FIPS PUB 1 [9] requires the use of the 7-bit ASCII code for interchange. In many communications protocols the units of data are bits or characters rather than blocks. The Cipher Feedback Mode of using the DES satisfies a requirement for encrypting data elements of length K where $1 \leqslant K \leqslant 64$.

The CFB mode of operation is shown in figure 5.6. The input to the DES algorithm is not the data itself but rather the previous 64 bits of cipher. The first encryption uses an initialization vector (IV) as its I_0 input. In the CFB mode both the transmitter and the receiver of data use only the encryption operation of the DES. The output at time t is the 64-bit block O_t. The cipher at time t is produced by exclusive OR'ing the K bits of plaintext P_t to the leftmost K bits of O_t. This cipher C_t is transmitted and also is entered on the right side of the input register after the previous input is shifted left K bit positions. The new input is used for the next encipherment.

A 64-bit IV is generated at time 0 and put into the input register. From that time on, the cipher text will depend on this initial input. In order to fill the receiver's input register, one of two events must occur:

1. The receiver must independently generate the identical initial fill.
2. The transmitter must transmit sufficient data to fill the receiver's input register.

A guideline is that the transmitter generates a pseudorandom number (48 to 64 bits) and transmits it as the IV. The transmitter and the receiver shall use this number (with the high order bits of the 64-bit DES input padded with "0" bits if necessary) as the 64-bit IV. Using a higher number of bits provides higher security but also results in higher transmission overhead. It is desirable that no two messages enciphered with the same key use the same IV. The DES may be used as a pseudorandom number generator to generate the IV. Start-stop (asynchronous) communications devices should transmit the IV as characters with appropriate start-stop bits appended.

In the CFB mode, errors within a K-bit unit of cipher will affect the decryption of the garbled cipher and also the decryption of succeeding cipher until the bits in error have been shifted out of the DES input block. The first affected K-bit unit of plaintext will be garbled in exactly those places where the cipher is in error. Succeeding decrypted plaintext will have an average error rate of 50 percent until all errors have been shifted out of the input block. Assuming no additional errors are encountered during this time, the correct plaintext will then be obtained. Thus, the CFB mode is self-synchronizing.

[illegible]-81

The CFB mode of operation is also useful for the encryption of stored data. For maximun efficiency 64-bit data elements are used. If the terminal data block does not contain a full 64 bits of data, the remaining bits are padded before encryption. However, the cipher block may be truncated so that only the cipher bits corresponding to the unpadded bits are used. In this case the number of cipher bits will equal the number of data bits.

When using the K-bit CFB mode the last K bits of cipher can be altered by an active wiretapper, who knows the last K bits of plaintext, so that the final K bits will decrypt to any desired K bits of plaintext. This is the same threat that applies to the CBC mode with terminal block truncation. If this is a significant threat, it is recommended that the final K bits of plaintext be a function of the previous plaintext bits (i.e., a parity or sum check).

5.3.4 The Output Feedback (OFB) Mode

The Output Feedback (OFB) mode like the CFB mode operates on data units of length K where K is an integer from 1 to 64. However, the OFB mode does not chain cipher from one time to the next. A one bit error in cipher text causes only one bit of the decrypted plaintext to be in error. Therefore, this mode can be useful in applications where no error propagation is required.

Figure 5.7 illustrates the OFB mode. The first encryption uses an initialization vector (IV) as its I_0 input, and both the transmitter and receiver use only the encryption operation of the DES. The cipher at time t is produced by exclusive OR'ing the K bits of plaintext to the leftmost K bits of the output O_t. The same K bits of the DES output block are fed back to the right side of the input register after the previous input is shifted left K bit positions, and the new input is used for the next encipherment.

The output of the OFB mode is independent of both plaintext and cipher. Therefore, the OFB mode does not have the self-synchronization property of the CBC and CFB modes. If synchronization is lost then a new IV must be established between the transmitter and receiver.

5.3.5 Relationship of CBC and 64-bit CFB

Like CBC, the CFB mode of operation can be used to encrypt 64-bit blocks. In this case the entire 64 bits of O_t are exclusive OR'ed with 64 bits of plaintext at each encryption time. This is called the 64-bit CFB mode of operation.

Let M1 be a 64-bit CFB machine with key schedule, KS $= (K_1,K_2,...,K_{16})$, on each of the 16 encryption rounds. (Figure 5.3 shows the generation of a DES key schedule.) In CFB mode the same schedule is also used for decryption. Let M2 be a CBC machine with a key schedule of KR $= (K_{16},K_{15},...,K_1)$ for encryption (i.e., the DES decipher operation), and $(K_1,K_2,...,K_{16})$ for decryption (i.e., the DES encipher operation). If M1 encrypts the 64-bit plaintext blocks P_1, P_2, and P_3 with initialization vector IV to form cipher C_1, C_2, and C_3; then M2 will encrypt P_3, P_2, and P_1 with initialization vector C_3 to form cipher C_2, C_1, IV. Similarly while M1 will decrypt C_1, C_2, and C_3 (using initialization vector IV) to P_1, P_2, and P_3; M2 will decrypt C_2, C_1, and IV (using initialization vector C_3) to P_3, P_2, and P_1. Thus by reversing (IV,C_1,C_2,C_3) to (C_3,C_2,C_1,IV) we may decrypt cipher generated by M1 with M2.

To see that the above statements are true let E[S](X) represent the encryption of X in the ECB mode using key schedule S, and let D[S](X) be the ECB decryption of X under schedule S. Note that S is the key schedule and not the key itself. Decryption uses the key schedule in the reverse order of encryption. Thus, $E[KS](X) = D[KR](X)$. The encryption of P_1, P_2, and P_3 by M1 using IV may be described by three equations.

$$\begin{aligned} P_1 \oplus E[KS](IV) &= P_1 \oplus O_1 = C_1 \\ P_2 \oplus E[KS](C_1) &= P_2 \oplus O_2 = C_2 \\ P_3 \oplus E[KS](C_2) &= P_3 \oplus O_3 = C_3 \end{aligned}$$

O_1, O_2, and O_3 represent ECB encryption, with key schedule KS, of inputs IV, C_1, and C_2 respectively. $\oplus$ is a 64-bit exclusive or operator. The encryption of P_3, P_2, and P_1 by M2 using C_3 as the initialization vector may also be described by three equations.

$$\begin{aligned} E[KR](P_3 \oplus C_3) &= E[KR](O_3) &= D[KS](O_3) &= C_2 \\ E[KR](P_2 \oplus C_2) &= E[KR](O_2) &= D[KS](O_2) &= C_1 \\ E[KR](P_1 \oplus C_1) &= E[KR](O_1) &= D[KS](O_1) &= IV \end{aligned}$$

By reversing the key schedules, the inputs, and the outputs we have obtained equivalent machines. Similar equations may be derived for decryption, and the relationship holds for an arbitrary length stream of 64-bit plaintext blocks.

5.4 CBC and CFB for Data Authentication

The DES can be used for data (message) authentication. A Message Authentication Code (MAC) is computed as a cryptographic function of the data (message). The MAC is then stored or transmitted with the data. Only those knowing the secret key can recompute the MAC for the received data and verify that the data has not been modified by comparing the computed MAC with the stored or transmitted MAC. An unauthorized recipient of the data who does not possess the key cannot modify the data and generate a new MAC to correspond with the modified data. This technique is useful in applications which require maintaining data integrity but which do not require protecting the data from disclosure. For example, computer programs may be stored in plaintext form with a computed MAC appended to the program file. The program may be read and executed without decryption. However, when the integrity of the program is questioned, a MAC can be computed on the program file and compared with the one stored in the file. If the two MAC's compare, and the cryptographic key used to generate the MAC has been protected, then the program file has not been modified.

A MAC may be computed using either the CBC or the CFB mode. In CBC authentication, a message is encrypted in the normal CBC manner but the cipher is discarded. If the number of data bits is not a multiple of 64, then the last data bit is appended with zeros on the right to form an integral number of blocks. The most significant M bits of the final output block are used as the MAC.

In CFB authentication, a message is encrypted in the normal CFB manner except that the cipher text is discarded. After encrypting the final K bits of data and feeding the resulting cipher text back into the DES input block, the DES device is operated one more time and the most significant M bits of the resulting DES output block are used as the MAC.

In both CBC and CFB authentication, a MAC should be used that is as long as practical. Since a MAC is an error detection code (which is computed using cryptographic techniques), a long MAC is desirable. Bit manipulation within a message will be detectable with a probability of $1-(1/2^M)$. Saying that a message is authentic or concluding that it has not been modified is based upon this probability. The proposed Federal Standard 1026 requires M to be at least 24 for Federal telecommunication applications. Financial transaction application standards are recommending M to be 32. Application designers should select M to optimize security and efficiency requirements.

In ADP communications security applications a message numbering and verifying system should be used to detect the insertion of false messages, the deletion of valid messages, and the replay of previously valid messages. The combined use of a Message Identifier (MID) and a MAC achieves these security objectives and protects against modification. If the data source MAC and the data destination MAC are in agreement and if the MID agrees with the value expected by the receiver, then the message is accepted. The MID should be unique and deterministic for each message transmitted between a sender and receiver. The uniqueness may be achieved through the use of a nonrepeating binary counter.

5.5 System Implementation

FIPS PUB 46 specifies that the basic implementation of the DES be done in hardware. However, the type of hardware used and the placement of the hardware will depend on the system and the requirements for speed and security. The DES device may also be interfaced to a computer system and an application program. This subsection will describe one possible implementation and the software interface used on the experimental Data Encryption Testbed at NBS. The mention of the specific product brands does not constitute or imply an NBS endorsement.

The two TTL implementations of the DES and the two PROLOG microprocessor computers have been interfaced to an asynchronous communication line between a computer (PDP 11/45) and a terminal (CRT ASCII TTY compatible). The line will operate at 300, 1200 and 2400 bits per second. Seven-bit ASCII characters with a parity bit are transmitted in an 11-bit, start-stop format (one bit for start and two bits for stop). RS-232C electrical and mechanical interfaces are used at all Data Terminal Equipment (DTE) and Data Circuit-terminating Equipment (DCE) interfaces. Universal Asynchronous Receive/Transmit (UART) chips are used to receive and transmit data on both sides of each of the PROLOG computers. A full duplex communication system is supported with only a small delay encountered at the PROLOG computer.

When the two PROLOG/DES units are inserted into the communication line, the line is divided into three parts. (See fig. 5.8.) One part is between the terminal and the TSU (Terminal Security Unit), the second is between the TSU and the CSU (Computer Security Unit), and the third is between the CSU and the computer. The data is in plaintext form on parts 1 and 3 and is in ciphertext form on part 2. It is assumed that the terminal and the TSU are colocated in a secure facility and that the CSU and the computer are colocated in a secure facility.

The PROLOG computers have 1K of Read Only Memory (ROM) and 2K of Random Access Memory (RAM). Programs are written for the PROLOG computers on the PDP 11/45 using a UNIX operating system. NBS personnel have written a cross assembler program on the PDP 11 to assemble the programs of several microprocessors. The cross assembler is written in the C programming language and outputs a listing of the assembled program and a core image to the PDP 11 files.

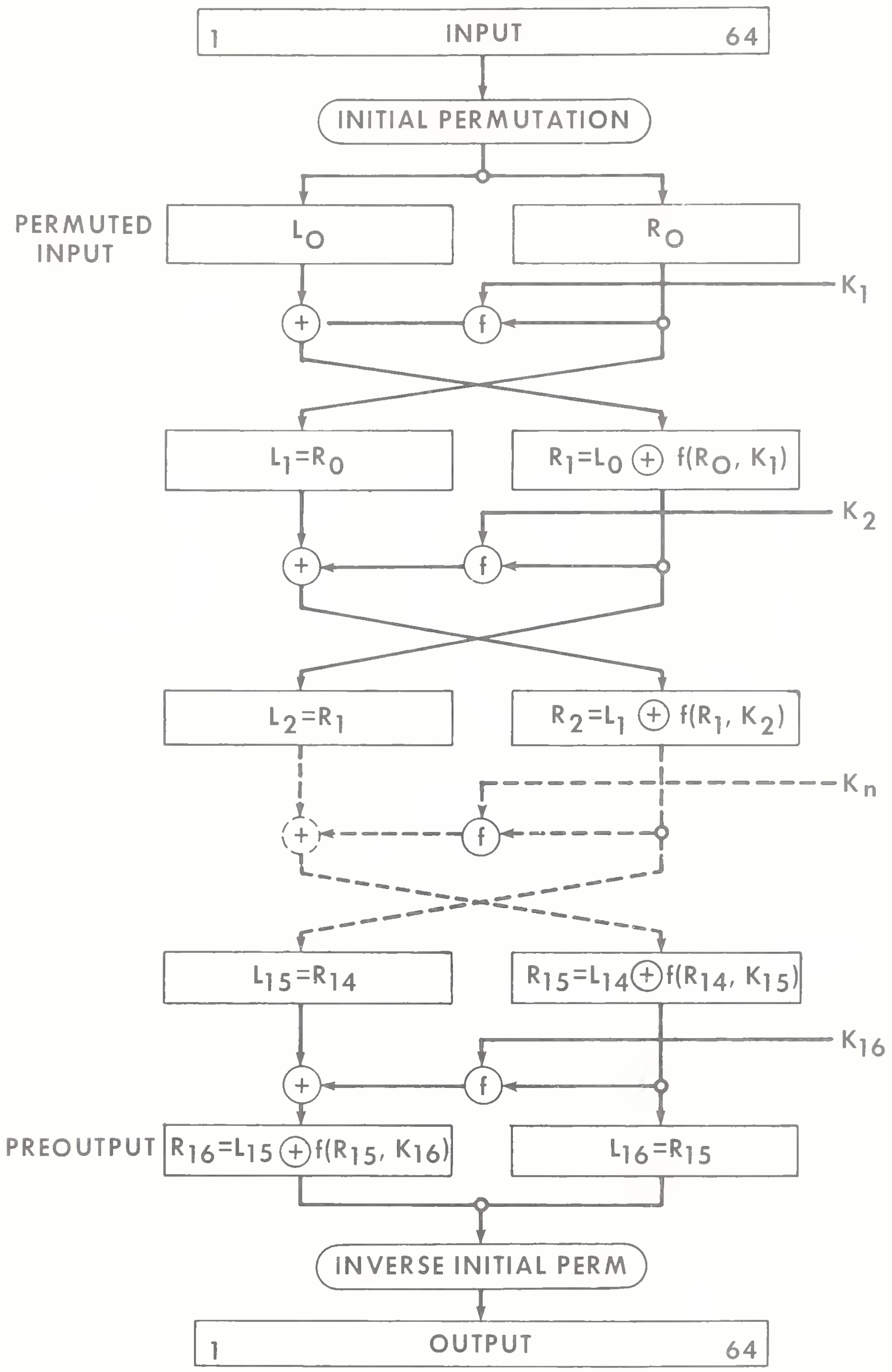


Figure 5.1. Electronic Codebook (ECB) Mode—Enciphering Computation.

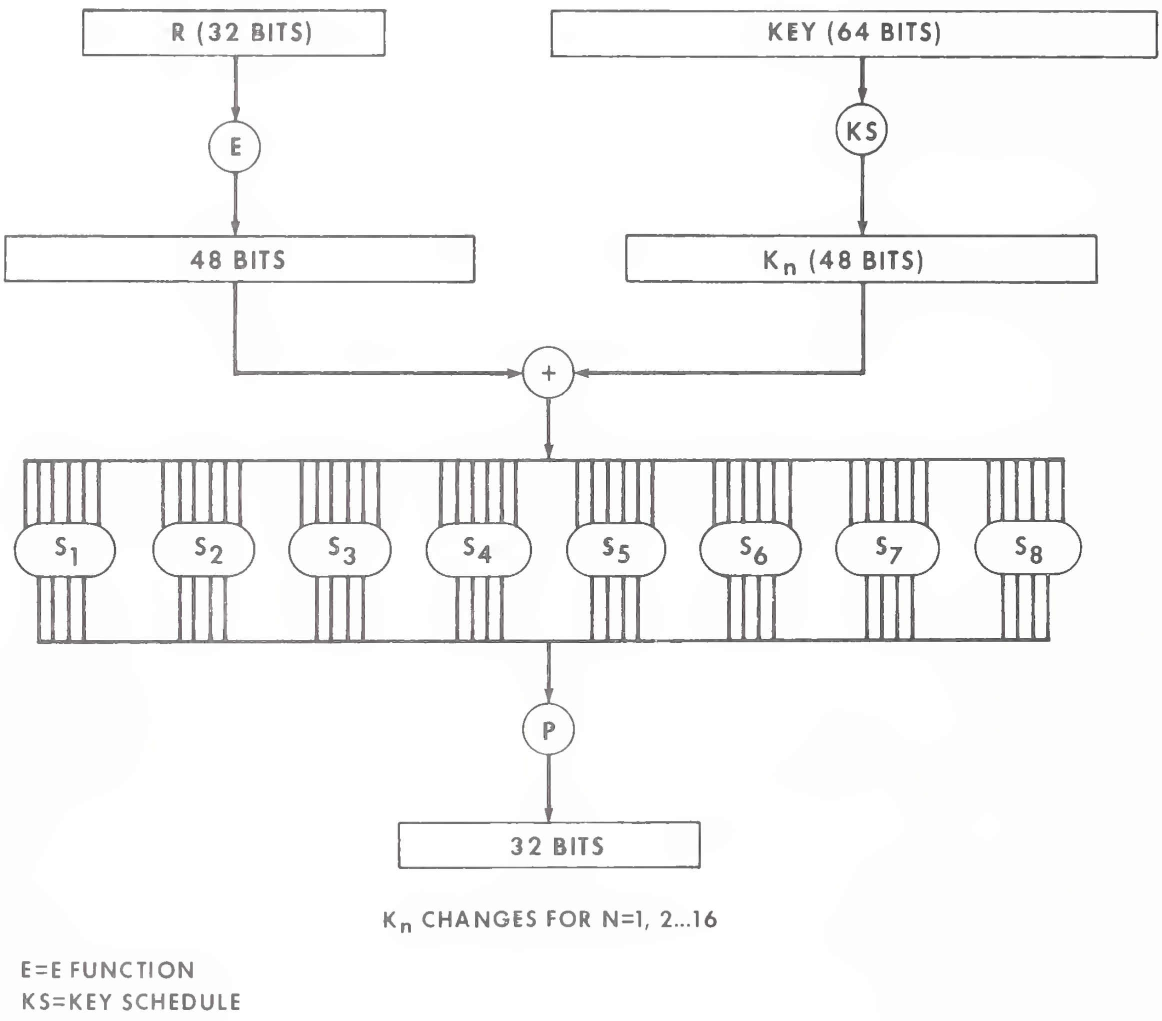


Figure 5.2. Electronic Code book (ECB) Mode—Calculation of f(R,K).

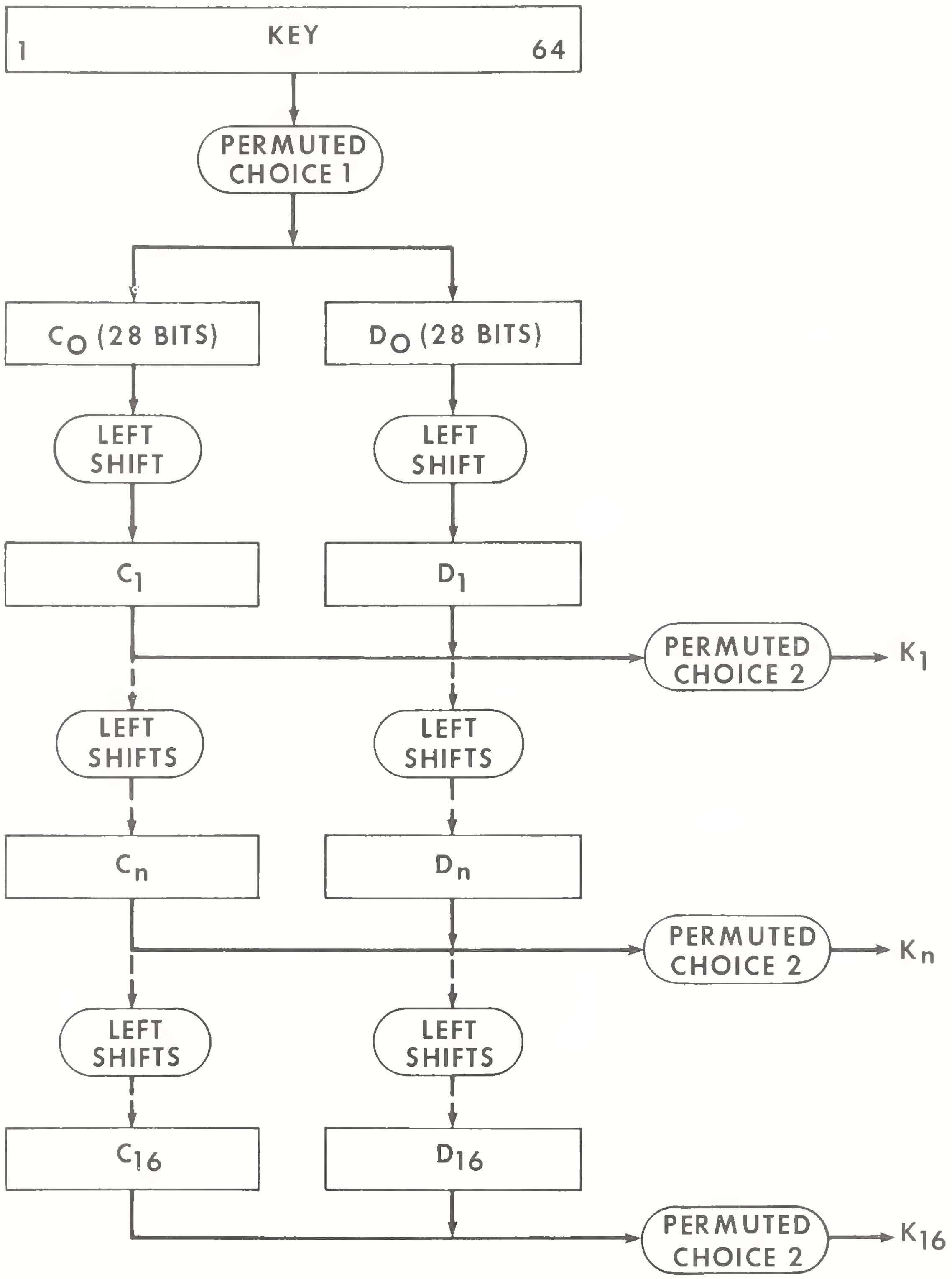


Figure 5.3. Electronic Codebook (ECB) Mode—Key Schedule (KS) Calculation.

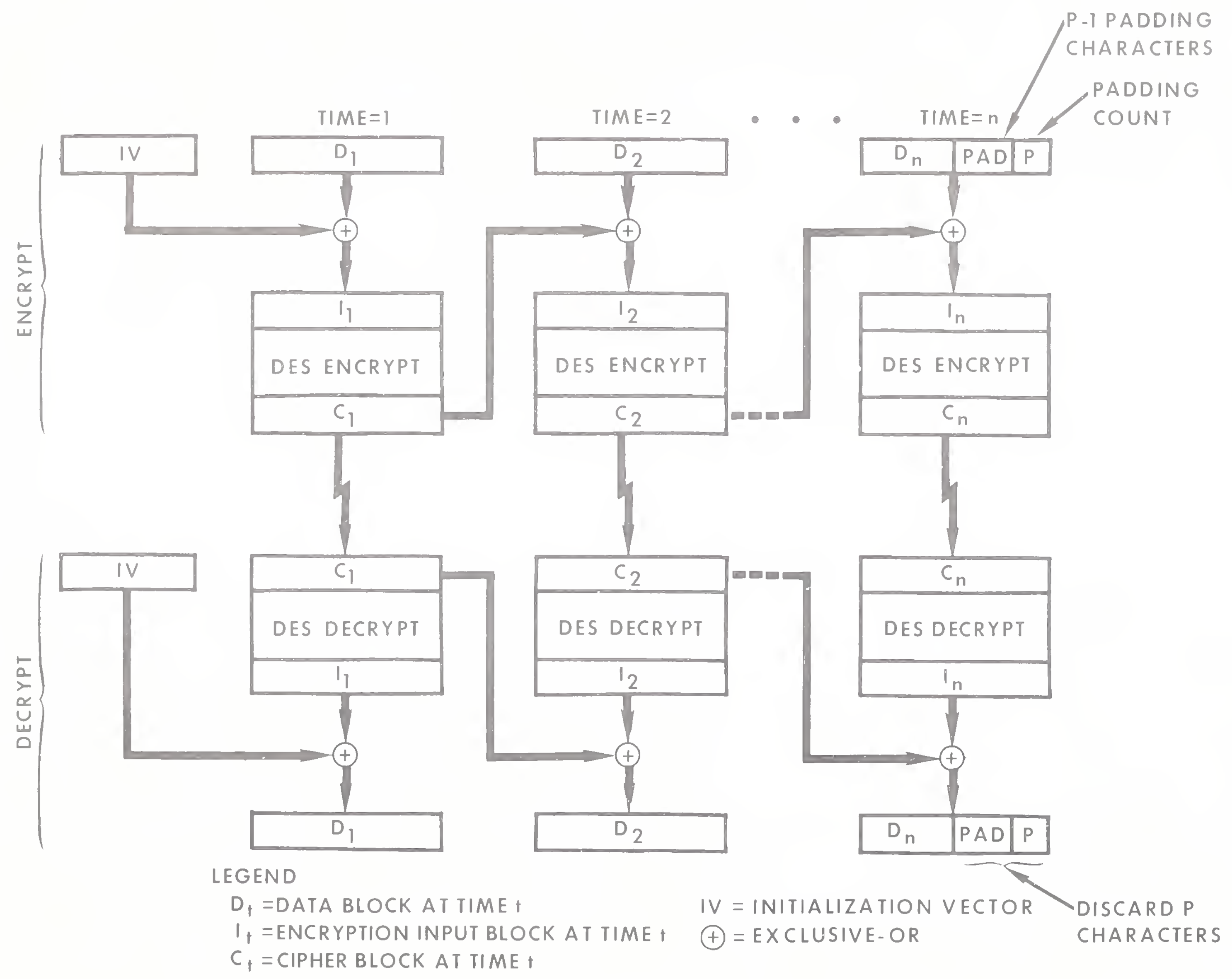
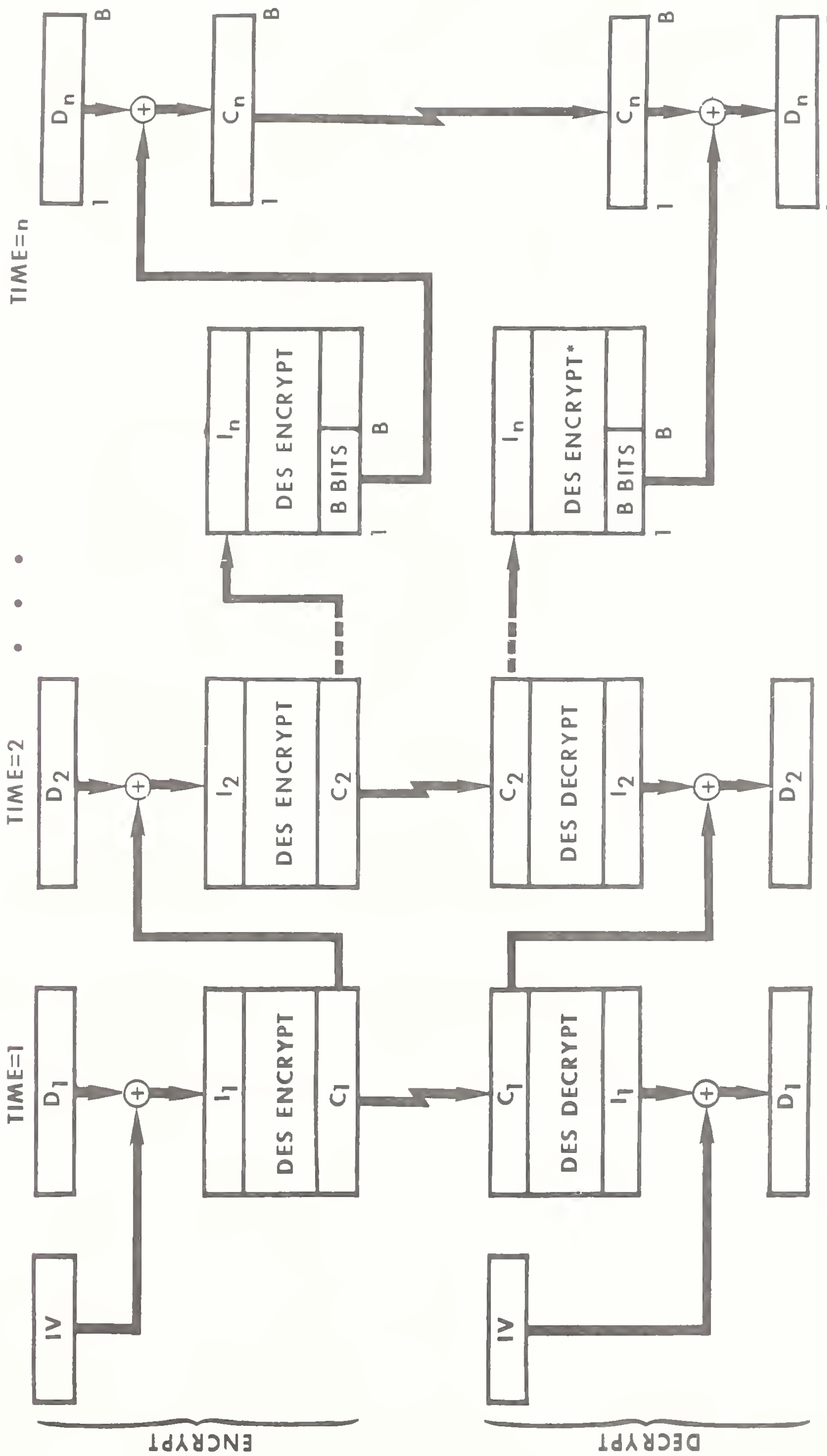


Figure 5.4. Cipher Block Chaining (CBC) Mode—With Terminal Block Padding.



LEGEND
D_t =DATA BLOCK AT TIME t
I_t =ENCRYPTION INPUT BLOCK AT TIME t
C_t =CIPHER BLOCK AT TIME t

IV = INITIALIZATION VECTOR
⊕ = EXCLUSIVE-OR

*NOTE MODE CHANGE

Figure 5.5. Cipher Block Chaining (CBC) Mode—With Terminal Block Truncation.

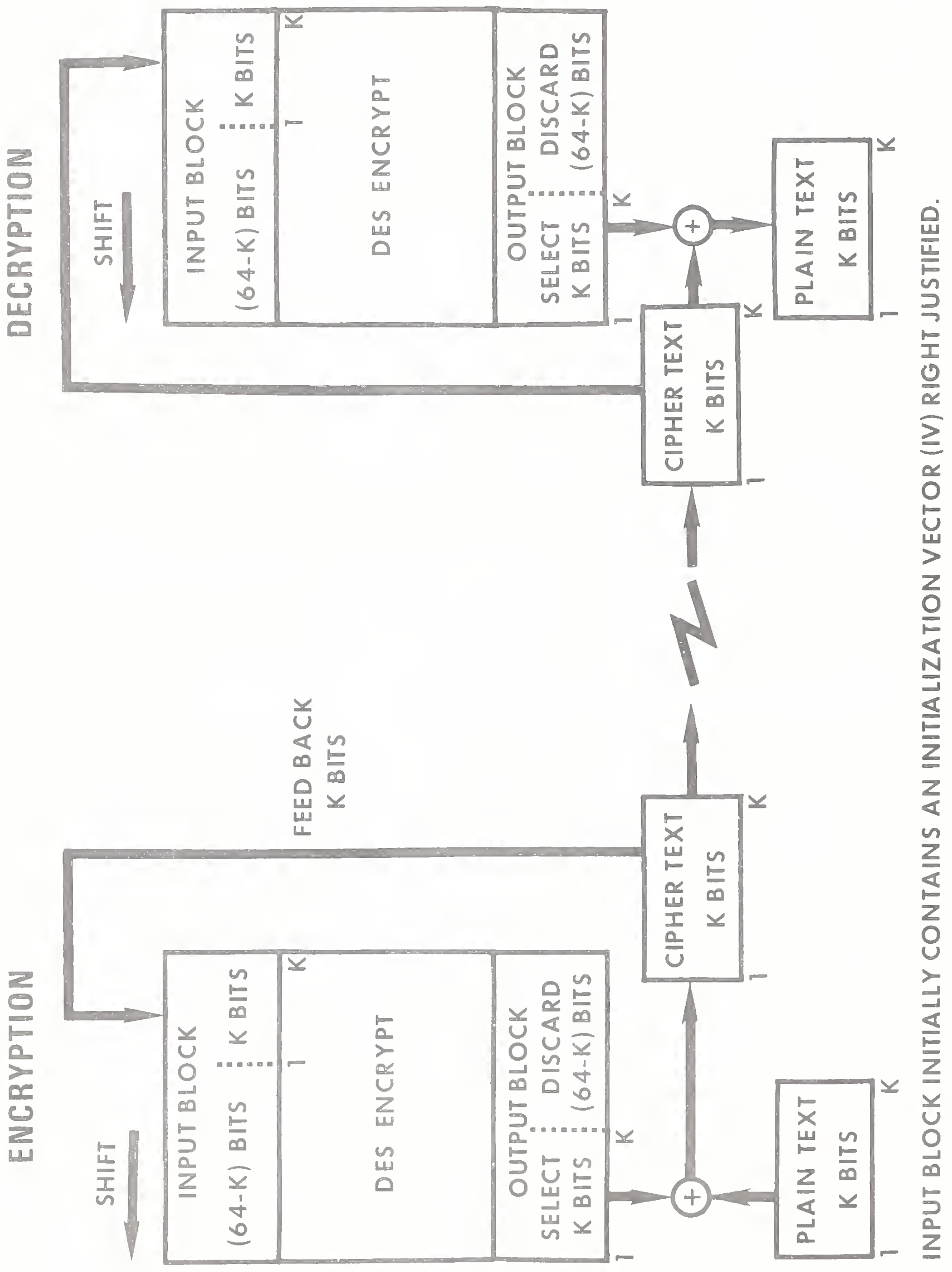


Figure 5.6. K-Bit Cipher Feedback (CFB) Mode.

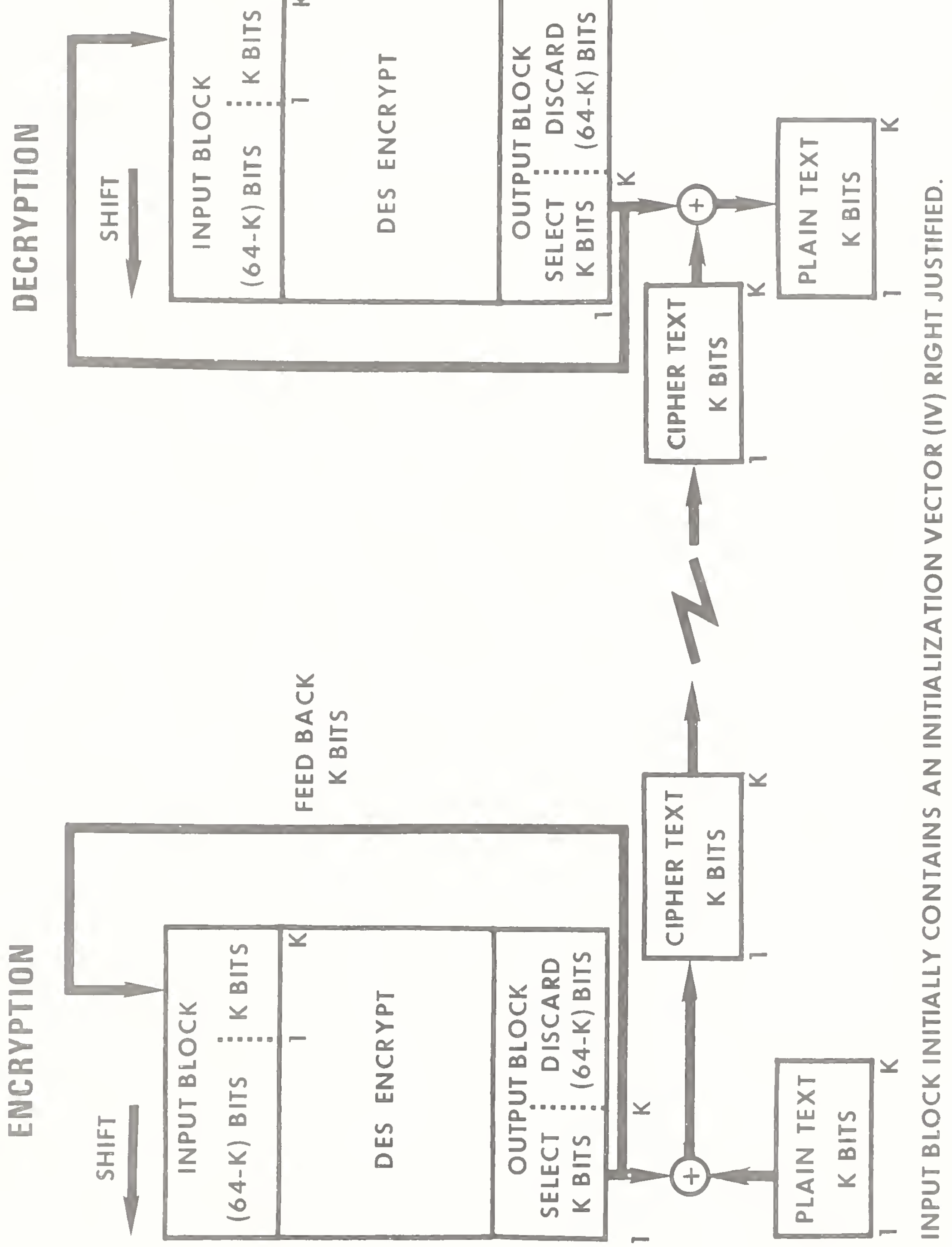


Figure 5.7. K-Bit Output Feedback (OFB) Mode.

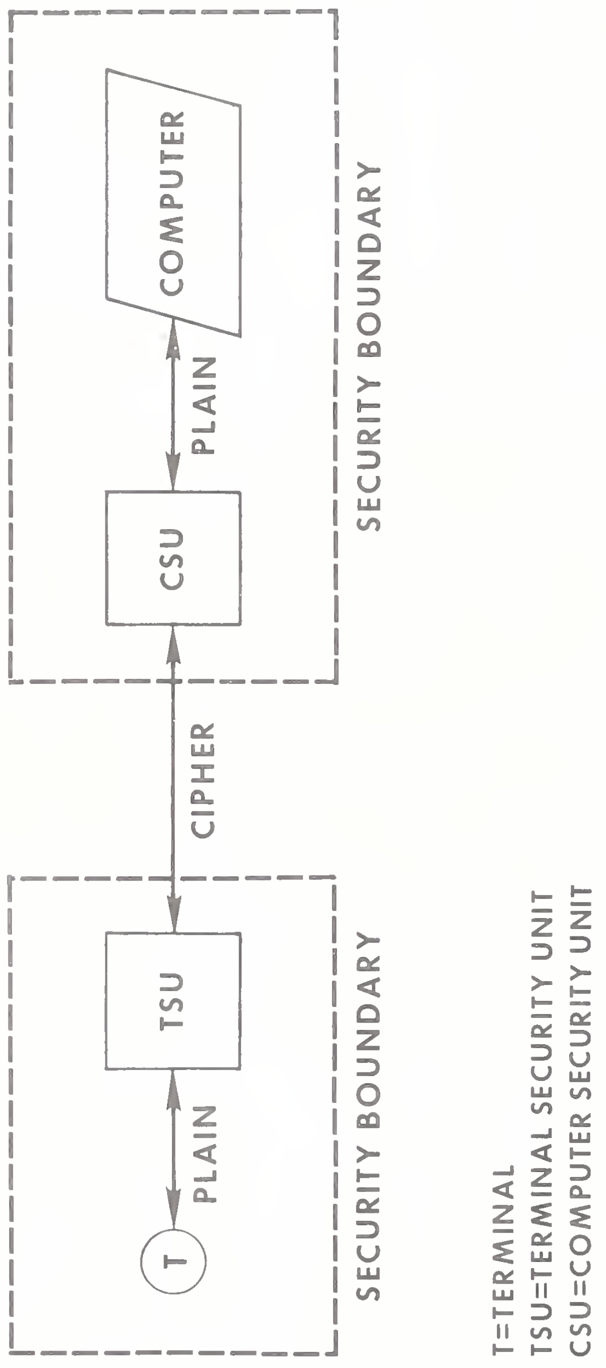


Figure 5.8. NBS Data Encryption Testbed.

Table 5.1 Electronic Codebook (ECB) Mode—E Bit-Selection Table

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Let E denote the function which takes a block of 32 bits as input and yields a block of 48 bits as output. The 48 bits of output, written as 8 blocks of 6 bits each, are obtained by selecting the bits from the input according to the above table. Thus the first 3 bits of E(R) are the bits in positions 32, 1, and 2 of R while the last 2 bits of E(R) are the bits in positions 32 and 1.

Table 5.2 Permuted Choice 1

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

The table has been divided into two parts, with the first part determining how the bits of C_0 are chosen. The bits of KEY are numbered 1 through 64. The bits of C_0 are respectively bits 57, 49, 41, ..., 44 and 36 of KEY, with the bits of D_0 being bits 63, 55, 47, ..., 12 and 4 of KEY.

Table 5.3 Permuted Choice 2

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

The first bit of K_n is the 14th bit of C_nD_n, the second bit the 17th, and so on with the 47th bit the 29th, and the 48th bit the 32nd.

Table 5.4 Left Shift Table

Iteration Number	Number of Left Shifts
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Successive C and D values are formed according to the above table. For example, C_3 and D_3 are obtained from C_2 and D_2 respectively, by two left shifts, and C_{16} and D_{16} are obtained from C_{15} and D_{15}, respectively, by one left shift. In all cases, by a single left shift is meant a rotation of the bits one place to the left, so that after one left shift the bits in the 28 positions are the bits that were previously in positions 2, 3, ..., 28, 1.

6. KEY MANAGEMENT

Management of the cryptographic keys used to protect data is of utmost importance to the security of the data. This chapter will provide guidance on how to generate, distribute, and protect keys.

There are at least three types of keys: data-encrypting keys, key-generating keys, and key-encrypting keys. When keys are stored in an encrypted form, the security of the keys is equivalent to the security of the key which was used for the encryption. Keys should be encrypted when stored in a less than fully secure medium and when transmitted over unprotected channels. In any cryptographic key system there has to be at least one unencrypted key. This key is often called a master key. The master key is the sole protector of all the information protected by each of the keys encrypted under the master. Thus, a master key is more valuable than any of the data-encrypting keys which it protects.

6.1 Key Generation and Protection

A DES cryptographic key consists of 64 bits, 56 of which are used by the algorithm (forming the active key) and 8 of which are used to detect errors within the key. If the 64 bits are numbered from left to right (1, 2, ..., 64), bits (8, 16, 24, ..., 64) are used for parity checking of each 8-bit byte. The parity bits should be set to the complement of the modulo 2 sum of the previous seven bits. Thus the modulo 2 sum of the entire eight bits is always 1.

Certain fundamental guidelines should be followed in generating keys. Every bit of the active key should be generated or selected at random. Every possible combination of bits in the active key should have equal probability of being selected, and each key should be generated independently of every other key. The security provided by each of the possible 2^{56} keys is the same although, in certain situations, the dual keys noted earlier may be undesirable because of the characteristic of the algorithm which makes the encrypt and decrypt functions identical for these keys. Repeating a short key to make a 56-bit key severely decreases security. A key made by repeating four hexadecimal characters, such as 29FB, four times to produce a 16-character DES key (including odd parity) provides only $2^{14}/2^{56} = 10^{-12.64}$ of the security of a fully independent key. A 56-bit key made from 8 decimal digits, each coded in 7 bits, reduces the security to $10^{8}/2^{56} = 10^{-8.85}$ of its maximum level.

The useful lifetime of a key will depend on the requirements and the environment of the application. A new key should be generated and used when any event occurs that may have compromised the existing key. A new key should also be generated and used periodically in the event that an undetected compromise has occurred. A system with low requirements for security and high costs of key change may change the key monthly. A system with medium security requirements may change the key weekly. High security requirements may dictate the need for changing the key daily or even more often. The method and cost of key distribution must be considered whenever a key management system is designed. Manual techniques and automated techniques are discussed in 6.2.

Unencrypted keys must always be physically protected to prevent unauthorized individuals from gaining knowledge of their values. Encrypted keys may also require physical protection if an unauthorized individual could in some way use an encrypted key to spoof system users. Physical protection of keys is often considered the weak link in the security of a cryptographic system. It is possible to design cryptographic algorithms to meet any specified level of security. This level may be measured in dollars or years of computer time required to recover plaintext. But it is difficult to quantify the effort required to subvert physical security. In many cases, a guard or courier could be bought for much less money than the amount required to mount a cryptographic attack.

6.2 Key Distribution

Key distribution is perhaps the most critical operation in a complex cryptographic system. Generating a "good key" for the DES is a relatively simple task. However, distributing this key to all the authorized users or devices may require the greatest amount of planning in the design and operation of a secure communication system. Since key distribution techniques depend on the particular application, this subsection will treat two basic applications of data encryption separately.

6.2.1 Communication Security

A requirement for communication security based on encryption is to have the decryption key available wherever decryption is authorized. Each authorized user of the key must be authenticated before the key is distributed and the transmitter of the key should be authenticated by the receiver before the previous key is discarded.

Manual methods of key distribution are commonly used where the security requirements allow long lifetimes for keys or where there are only a few devices using the same key. Duplicated key lists are often distributed by certified mail or by courier. These lists usually contain a set of keys to be sequentially used and specifications as to when to use them. In case of a possible compromise of a key, alternative keys are presented. In case of a possible compromise of a list, alternative lists are distributed with instructions for use. Machine readable storage media, such as punched cards, paper or magnetic tape, or magnetic striped cards may be used. Special key loading devices such as electronic memory chips, electronic modules, battery operated key loaders, etc. may also be used. Keys are generally inserted at the designated time into an encryption device by a security officer who physically unlocks the device and manually or electronically enters the key. Cryptographic devices generally have some form of physical protection against theft or tampering.

Automated methods of distributing a key may also be used. In general, a key to be used for a terminal-computer connection or transaction may be generated, distributed to the communicating devices via a secure path, and then destroyed at the conclusion of the connection or transaction. The secure path may be a dedicated path for distributing keys or it may be a path that is established within the network that is protected by a key used only for that purpose. The latter is considered more feasible in a general computer network. The key-encrypting key should be manually distributed or distributed outside the normal communication network.

Specific methods to be used in key distribution must be based on the characteristics of the network being protected. The value of the data being transmitted and the anticipated threats to the data are important factors. It must be emphasized that the protection provided through the use of the DES is no better than the protection provided to the key. It should be assumed that if a particular key is lost or compromised that any data protected by that key is also compromised. Provisions should always be made so that the key can be changed in an orderly and timely manner if its compromise is suspected.

6.2.2 File Security

Encryption protection may be provided for data to be stored in files. This protection is called file security. Data files may have many different structures and they may be stored on various storage media. It is very important that the use of encryption be evaluated with respect to the anticipated threats to the data. Only certain types of threats can be prevented or deterred in general. Theft of storage media will not permit the thief to read the plaintext corresponding to the encrypted data. However, unless a backup exists, the data will also be unavailable to the owner. Physically secured computers or computers with secure operating systems are required to protect the plaintext data while it is resident in the computer itself. Encryption does not solve the computer security problem, but it may reduce its magnitude and provide increased flexibility.

The distribution problem for encryption keys in file security applications is different than that in communication security applications. The former requires that only one copy of the key be available when the data is encrypted before it is stored. However, the key used to protect the data must be associated with the data and securely stored until the data is to be used. File security in this application simply reduces the amount of data requiring physically secure storage to the key itself. The key can only be discarded when the data is reencrypted under a new key, when the data is decrypted and no longer requires cryptographic protection, or when the data is no longer needed.

Encryption may also be used in another file security application which is analogous to a secure data vault. The computer facility is used to store data that was encrypted at a terminal and which can only be decrypted at a terminal. The encrypted data cannot be processed within the computer, but encrypted data may be stored and retrieved by location, by surrounding unencrypted data, or by a related unencrypted index. Users may encrypt selected fields of sensitive data at the terminal before it is sent to the computer. The user must store or remember the key used. When the data is to be retrieved and used at the terminal, it is decrypted just before it is printed. The disadvantage of this application is that the computational capability of the computer cannot be fully used because its sensitive data is always encrypted.

6.3 Key Destruction

When keys are no longer needed for encryption or decryption, they should be destroyed. Even after a key is destroyed the information which it protects often continues to be sensitive. One should always assume that the cipher has been exposed to unauthorized, untrusted, individuals. It is therefore necessary that the remains of the destroyed key contain no information which would aid an adversary in the reconstruction of the key.

7. TRANSPARENCY IN COMMUNICATIONS PROTOCOLS

A protocol is a procedural standard or a discipline for maintaining order. It is an agreement to follow an established set of rules. A communications protocol is a set of rules for a group of cooperating users which will allow them to communicate effectively. Transparency is an attribute of a communication protocol that describes the flexibility of the protocol for allowing changes which do not affect the rest of the protocol. This chapter presents some of the issues of adding encryption to a communication system.

A computer network can be described in terms of communications protocols, configurations, code sets and operational procedures. A protocol specifies the control procedures of the network (e.g., connection establishment, flow control, error control). The configuration specifies the topology and participating equipment of the network. The code set specifies the bit patterns of the user data and the control information. Finally, the operational procedures specify the administrative aspects of the network: when the network is available, how people will get access to the data and services of the network, how connections between communicating devices are to be established, etc.

The fundamental use of encryption in communications has historically been to hide the meaning of messages from the enemy. However, encryption can provide additional benefits. In some communication systems it is desirable to hide the fact that a message is sent at a particular time. This is called traffic flow security. It may also be desirable to assure that a message is received unaltered. A more recent requirement of some communication systems is for the receiver of a message to be able to "prove" to a third party that he did, in fact, receive the message from the transmitter. The protocols of a communication system will depend greatly on the security requirements as well as the physical properties of the system.

7.1 Transparent Use of Encryption

A goal of adding cryptographic protection into an existing data network is to make its use transparent to the other functions of the network. How well this goal is met will depend on the characteristics of the network and at what point in the development of the network cryptographic protection is incorporated. Cryptography should be incorporated into the design phase as soon as possible. The security objective of performing encryption at the place of origination of a message and not performing decryption until the message reaches its ultimate destination often makes complete transparency more difficult to achieve.

Cryptographic devices may generally be placed at the ends of a simple communication link with little difficulty. Transparent operation of the devices can be achieved by encrypting everything that leaves one end and decrypting it upon arrival at the other end. Since there are not any devices in the path between the cryptographic devices that are sensitive to the information being transmitted, control information need not be separated from data. Both synchronous and asynchronous transmissions may be protected in this way. The only requirement for transparency is that the data entering the encryption device must exit the decryption device at the other end of the communication line with an acceptable delay. All links of a network may use the same key, or different keys may be used for each link. As a rule, network users will not know that the data is encrypted from the operational response of the network.

More complex communication systems make use of network control devices to route data to the intended receiver. Control information for such systems must be in plaintext wherever it is used by a network control device. The control device must be able to differentiate between plaintext and ciphertext if both are contained in the data stream. The differentiation can be implicit or explicit. An example of the former is the separation of data from control by position, and an example of the latter is to reserve special codes for control. It is generally easier to add encryption onto a communication network which implicitly separates data and control information. In either case separation of control information from data is necessary before encrypting in all but the simplest link encryption application. Separating control information from data in order to achieve cryptographic transparency in end-to-end encryption applications is necessary and is often difficult if encryption is performed after the communication control information is added.

Data transparency requires that encrypted data which have the same codes as control characters not be interpreted as legitimate control. In Binary Synchronous Communications (BISYNC) transparent mode, valid control characters are indicated by a two-character sequence consisting of "DLE" followed by the control character. If the cipher results in a "DLE" character, a "DLE DLE" is sent for the single "DLE" and the extra "DLE" is removed before decryption.

Section 8 describes an alternate method of achieving transparency in which control characters are never generated in the encryption process. This method requires special operations for mapping data characters only onto data characters.

Cipher text transparency is generally easy to achieve in packet or message communication protocols because the data is implicitly separated from control information. Control information is typically added after the user data is encrypted. Traffic flow security is generally not provided in such networks but link encryption of data and control may be utilized in those networks where the amount of message traffic is considered sensitive. This requires that the encryption device continues to produce cipher which is transmitted even though no messages are being sent.

7.2 Nontransparent Use of Encryption

When code transparency is not required in a communication system, adding cryptography to the system is generally easier. The designer need not be concerned about the output of the encryption process since no device in the system that is sensitive to the code exists in the path between the encryption and decryption devices. However, if the encryption device in some way controls the decryption device, then the control must be provided by 1) control signals sent outside the data path; 2) special control codes that are detected by the decrypting device in the decrypted plaintext; or 3) special control codes in the ciphertext.

In any communication application of encryption other than the simplest implementation of link encryption, a certain degree of nontransparency will be unavoidable. Procedures must be established for entering the key at the proper time, errors must be handled in some way, and encrypted data must be recognized.

7.3 Communication Standards Based on the DES

Standards are necessary to assure that terminals and computers which use the DES are able to communicate. It is assumed that for any two devices to communicate in an encrypted mode, they must first be able to communicate in an unencrypted mode. This requirement establishes many of the parameters of communications protocols such as the code, the synchronization mode, the message protocol, the line speed, the channel capacity, the error control and the connection control. The use of DES in communications requires the specification of the following additional parameters:

a. Mode of encryption/decryption
b. Initialization
c. Synchronization
d. Error control
e. Buffering
f. Key management

Federal and American National Standards Institute (ANSI) standards efforts have been initiated to define appropriate specifications for these parameters in several communications protocols. The Federal standards are being drafted in a subcommittee of the Federal Telecommunications Standards Committee. One of the standards being prepared is expected to be issued as Federal Standard 1026. It specifies interoperability and security related requirements for communication security devices implementing the DES. Federal Standard 1027 will be a companion document which specifies the minimum physical and electrical security features of devices implementing the DES. ANSI cryptographic standards efforts at the time of this publication include:

1. ANSI X9A3: Security Standards of Consumer Initiated Electronic Financial Transactions;
2. ANSI X9E8: Financial Message Authentication Standard;
3. ANSI X3T1: Standards for Use of the Data Encryption Algorithm.

In addition, the International Organization for Standardization has established a working group to address data encryption.

8. USING DES TO MAP A CHARACTER SET ONTO ITSELF

In certain applications it is desirable that only valid plaintext characters appear as cipher. For example, special control characters are often used to designate headers, synchronization bits, and the beginning or ending of ciphertext. If control characters can also appear randomly as cipher, then it is difficult to distinguish between intended plaintext control characters and cipher. One solution is to stuff redundant characters into the transmitted data (to indicate control characters) thereby adding additional overhead. Also, in situations where cipher characters are to be printed, no unprintable characters can be permitted in cipher. A character is defined to be valid if it is not used as a control character and invalid if it may be used for control. For example, a character which indicates a carriage return is invalid. A problem arises since presently defined DES modes of operation map K-bit characters onto K-bit characters. If the number of members in a valid plaintext character set is not a power of two, then invalid characters will appear in cipher. A modification is proposed which permits the encryption of a character set of arbitrary size onto itself. Therefore, valid characters are always encrypted to valid characters. The modification is discussed as it applies to specific examples as well as to the general problem.

8.1 Example I (Digits)

In this subsection we will consider a solution for the problem of enciphering digits onto digits. Later subsections will apply the solution to other cases.

Consider DES as used in the Cipher Feedback (CFB) mode. (See fig. 5.6.) K bits of the 64-bit DES output are exclusive OR'ed with a K-bit plaintext character to form cipher. Suppose that one desires to encipher the digits, 0 through 9. Four-bit characters are required to represent the 10 digits, the first 10 character representations correspond to the digits, and the remaining 6 are invalid. (See table 8.1.) Even if only valid plaintext characters are enciphered, DES in the CFB mode will produce cipher characters which may be invalid.

8.1.1 Solution

Let FO denote the 64 bits of the final DES output. Instead of exclusive OR'ing the first four bits of FO with the four-bit plaintext character, add the two values modulo 10 (base 10). The modulo 10 sum of the digits A and B is the remainder of A + B divided by 10. X is congruent to Y modulo 10 ($X \equiv Y$ modulo 10) if and only if $X - Y = 10m$ for some integer m. Thus A + B is congruent to a valid cipher character. For example, suppose that FO = 1101..... and that one wished to encipher 0011. Since $0011 + 1101 = 10000 \equiv 0110$ modulo 10, 0110 is the resultant cipher. The input register to the DES function will contain exactly 16 valid cipher characters, so 10^{16} distinct input register combinations are possible.

8.1.2 Decryption

The decryption algorithm is similar to the encryption algorithm except that the first FO character is subtracted modulo 10 from cipher to form plaintext. Using the values from the previous example, $0110 - 1101 = -0111 \equiv 0011$ modulo 10. The algorithms are inverses of each other because the FO generated by the decrypting device is the same as the FO generated by the encrypting device.

Let P be a valid plaintext character and G be the first character of FO. Let C be the corresponding cipher character.

$$\begin{aligned} C &\equiv (P + G) \text{ modulo } 10. \\ C &= P + G - 10m. \\ P &= C - G + 10m. \\ P &\equiv (C - G) \text{ modulo } 10. \end{aligned}$$

Decryption is unique even though the first character of FO may not be an integer modulo 10 (i.e., a base 10 digit).

Since G is not necessarily a valid character, there is a bias on cipher which depends on the plaintext. If the plaintext is flat (randomly generated), for example, the cipher is also flat, but if several plaintext zeros are encrypted there is a bias towards zero through five in the cipher. This problem can be overcome by selecting G from FO in a manner which virtually assures that G is evenly distributed over the digits. Subsection 8.5 provides a solution which will render an insignificant bias in most applications.

One might consider encrypting the digits as follows: Exclusive OR(XOR) the first FO character with the plaintext character and then use the result modulo 10. The trouble with this solution is that it does not decrypt correctly. Suppose that FO = 0101..... and that plaintext is 1000. 0101 XOR $1000 = 1101 \equiv 0011$ modulo 10. Therefore 0011 would be taken as cipher. But 0011 XOR 0101 = $0110 \not\equiv 1000$ modulo 10. Decryption would not produce the correct plaintext.

8.2 Example II (Alphanumerics)

The USA Standard Code for Information Interchange (ASCII), with b7 as the high-order bit and b1 as the low-order bit, appears in table 8.2. Suppose one desires to encipher the 96 characters whose binary representations range from 0100000 to 1111111. These 96 characters may be mapped into the integers modulo 96 by subtracting 0100000 from their ASCII representations. Let $\leftrightarrow$ symbolize this mapping. Then

$$\begin{array}{rcccccl} \text{SP} & = & 0100000 & \leftrightarrow & 0000000 & = & 0, \\ ! & = & 0100001 & \leftrightarrow & 0000001 & = & 1, \\ \cdot \\ \cdot \\ \cdot \\ \text{DEL} & = & 1111111 & \leftrightarrow & 1011111 & = & 95. \end{array}$$

If we wish to encipher the character, n, and the first character of FO is }, then cipher is formed using the following equations.

$$\begin{aligned} &n = 1101110 \leftrightarrow 1001110. \\ &\} = 1111101. \\ &\text{cipher} \leftrightarrow (1001110 + 1111101) \text{ modulo } 96 \equiv 0001011. \\ &\text{cipher} = (0001011 + 0100000) = 0101011 = +. \end{aligned}$$

One must remember to translate the plaintext to an integer modulo 96 before addition and then to translate the result back to a valid character after addition. Nine characters may be held in the 64-bit input register. The number of possible input register settings is, $96^9 = 6.92 \times 10^{17}$. Note that since the length of a character (7 bits) does not evenly divide the length of the input register (64 bits) the first bit of the input register is always fixed to zero.

8.3 Example III (General Solution)

The proposed method may be used as a general solution. Suppose one has an N character alphabet. Let K be such that $2^{K-1} < N \leqslant 2^K$. Then one must be satisfied that $N^{[64/K]}$ (the number of possible input register combinations) is sufficiently large where [X] is the greatest integer $\leqslant X$. For security reasons, it is recommended that $N^{[64/K]}$ be at least $2^{48} \cong 10^{14.4}$.

If the characters are contiguous, then a simple translation will map them onto the integers modulo N before addition is performed; and after addition, the inverse will map back to valid characters (as previously discussed in 8.2). If the characters are not contiguous, then conversion tables can be used to make the transformations to and from the integers modulo N. Consider the USA Standard Code (ASCII) presented in table 8.2. Suppose that the only valid characters are: A, B, C, F, H, I, M, N, O, P, U, V, and Z. In this case N=13 and K=4. The number of possible inputs at each encryption is, $13^{16} = 6.65 \times 10^{17}$.

If the set of possible characters is not too large, then for each possible character the conversion table will list its modulo N value, if it is valid, or an invalid indicator, if it is invalid. This table could be used to determine whether or not a character is valid as well as to map it to its corresponding modulo N value. (See table 8.3.)

If the character set is too large other possibilities exist. A conversion table could be made which just covers the range from the first to the last valid character. In this case characters which are found to be less than 1000001 and greater than 1011010 are invalid. For the others, subtract 1000000 and use the result as an index to the table. (See table 8.4.)

Another possibility is to store the binary representations and modulo N values for only valid characters. Searching, hashing, or some other method must be used to find the correct location of the character being looked up. (See table 8.5.)

Once the modulo N sum of the plaintext and K bits of FO have been found, another table (the inverse of table 8.5) is required to convert back to the binary representation. This table need only have one entry for each integer modulo N. The integer modulo N is incremented and the result is used as an index to find the corresponding cipher character.

8.4 Solution for Plaintext Bias

When the ciphertext bias produced by the use of invalid characters from FO is unacceptable, only valid characters should be selected from FO. Consider the example where the digits are to be mapped onto themselves. The first four bits of FO will be valid with probability 10/16. If the first four bits form a valid character they may be used for the addition to plaintext. If they are invalid consider the second four bits. If the second four bits form a valid character they may be added to the plain text to form cipher. Repeat this procedure until either a valid cipher is formed or until all 16 four-bit characters of FO have been examined and each one is found to be invalid. The latter event, called the default condition, will occur with probability $(6/16)^{16} = .000000153$. In this case the value to be added to plaintext can be arbitrarily selected as 1001 (9). A new FO is generated for each character to be enciphered.

If the bits of FO are statistically random then, as long as the default condition is not employed, the cipher should also be random. The default condition is definitely nonrandom, but since it should only occur with probability .000000153 the ciphertext will be near random. In fact, frequency counts would have to be done on very large amounts of data before the slight bias would be detectable. Using a Chi-square test would require data from more than 10^{13} encryptions before one could expect to detect nonrandomness. Of course, if the plaintext is flat random, no bias will be found on cipher.

In general if one has a character set of N members and K is such that $2^{K-1} < N \leqslant 2^K$, then one must be satisfied that $((2^K-N)/2^K)^{[64/K]}$, where [X] is the greatest integer $\leqslant$X, is sufficiently small.

Table 8.1 Digit to Character Conversion Table

	0	↔	0000	
	1	↔	0001	
	2	↔	0010	
valid digits	3	↔	0011	valid characters
	4	↔	0100	
	5	↔	0101	
	6	↔	0110	
	7	↔	0111	
	8	↔	1000	
	9	↔	1001	
	10	↔	1010	
	11	↔	1011	
invalid digits	12	↔	1100	invalid characters
	13	↔	1101	
	14	↔	1110	
	15	↔	1111	

Table 8.2 USA Standard Code for Information Interchange*

Bits b7 b6 b5					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b4	b3	b2	b1	COLUMN → ROW ↓	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	´	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	¦
1	1	0	1	13	CR	GS	−	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

*Reprinted with permission of the American National Standards Institute.

Table 8.3 All Characters of Table 8.2

Entry	Binary	Modulo N Value
1	0000000 (NUL)	17 (invalid character)
2	0000001 (SOH)	17
.	.	.
65	1000000 (@)	17
66	1000001 (A)	0 (valid character ↔ 0 modulo 13)
67	1000010 (B)	1
68	1000011 (C)	2
69	1000100 (D)	17
70	1000101 (E)	17
71	1000110 (F)	3
.	.	.
91	1011010 (Z)	12
92	1011011 ([)	17
.	.	.
128	1111111 (DEL)	17

Table 8.4 Valid Character Range

Entry	Binary	Modulo N Value
1	1000001 (A)	0 (valid character ↔ 0 modulo 13)
2	1000010 (B)	1
3	1000011 (C)	2
4	1000100 (D)	17
.	.	.
25	1011001 (Y)	17
26	1011010 (Z)	12

Table 8.5 Valid Characters Only

Entry	Binary	Modulo N Value
1	1000001 (A)	0
2	1000010 (B)	1
3	1000011 (C)	2
4	1000110 (F)	3
5	1001000 (H)	4
.	.	.
13	1011010 (Z)	12

9. REFERENCES

[1] Data Encryption Algorithm (DEA), American National Standards Institute ANSI X3.92.

[2] Data Encryption Standard, National Bureau of Standards (U.S.), Federal Information Processing Standards Publication (FIPS PUB) 46, National Technical Information Service, Springfield, VA (1977).

[3] DES Modes of Operation, National Bureau of Standards (U.S.), Federal Information Processing Standards Publication (FIPS PUB) 81, National Technical Information Service, Springfield, VA (1980).

[4] Feistel, Horst, Cryptography and Computer Privacy, Scientific American, Vol. 228 No. 5, May 1973, pages 15-23.

[5] Gait, Jason, Validating the Correctness of Hardware Implementations of the NBS Data Encryption Standard, NBS Special Publication 500-20, Revised September 1980.

[6] Guideline For Automatic Data Processing Risk Analysis, National Bureau of Standards (U.S.), Federal Information Processing Standards Publication (FIPS PUB) 65, National Technical Information Service, Springfield, VA (1979).

[7] Management and Use of Personal Identification Numbers, ABA Bank Card Standard, Aids from ABA catalog number 207213 (1979).

[8] National Security Agency Memorandum for the Members, Former United States Communications Security Board, Serial: N/0817 (7 July 1978).

[9] USA Standard X3.4-1968, Standard Code for Information Interchange, Federal Information Processing Standards Publication (FIPS PUB) 1, United States of America Standards Institute, 10 East 40th Street, New York, New York 10016 (November 1968).

U.S. DEPARTMENT OF COMMERCE
National Technical Information Service
5285 Port Royal Road
Springfield, Virginia 22161

OFFICIAL BUSINESS

POSTAGE AND FEES PAID
U.S. DEPARTMENT OF COMMERCE
COM-211

U.S.MAIL

3rd Class Bulk Rate