

Modern Data Encryption

Protecting the Enterprise

Paul Howard

PARANOIA2 WHITE PAPER

PAGE NUMBER

PHINC Consultants

PARANOIA2 WHITE PAPER

PAGE NUMBER

CONTENTS

SECTION	TITLE	PAGE
1	Introduction	3
2	Where Is The Threat	4
3	What Is Encryption?	5
4	Methods Of Encryption	5
5	Block Cipher	6
6	Stream Cipher	6
7	What Is DES?	7
8	DES Modes Of Operation	9
9	Electronic Codebook (ECB)	9
10	Cipher Block Chaining Mode (CBC)	10
11	Cipher Feedback Mode (CFB)	11
12	Output Feedback Mode (OFB)	12
13	Triple DES	13
14	Security Implications Of Key Ciphers	13
15	Securing The Key	14
16	Commercially Available Encryption Devices	16
17	Problems with incorporating DES	17
18	How the Paranoia2 incorporates DES	18

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, which requires protection.

It is common to find data transmissions equating to millions of pounds transferred daily. At all company levels, sensitive information concerning individuals, finances, product or business developments is held and processed in computer systems.

The development of large centralised storage repositories such as enterprise disc subsystems and tape silos, increases the potential threats to personal and corporate privacy. Since data banks often are accessed from remote computer terminals, there is a threat of easy and unauthorised access to sensitive information from any place in the data communications system.

A large amount of effort has been put in place to protect centrally held data which is effectively on-line, i.e. in an enterprise disc subsystem, as this is seen as the most vulnerable access point. However, despite the increasing awareness of the Information Technology industry to the importance of backup and the cost benefits of data archiving to tape, the potential weakness of the security of sensitive data while resident on tape is not being addressed.

Encryption is a tool that may be used in a centralised data pool in a tape environment. It is not a panacea; improper implementation and use of data encryption may only provide an illusion of security. Inadequate understanding of encryption applications and data encryption could deter the utilisation of other required 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 specific data processing applications.

Where is the Threat?

It can be generally accepted that at least some of the data held by a company is sensitive, be it to private individuals, members of staff, business rivals or because of the security legislation in the country or community where the business operates.

The proportion of sensitive data is dependent upon the nature of the business in which the company operates. The prime environments where there will be the highest proportion of sensitive data are Research and Financial. Here the data is the company's most guarded asset, such as new product development in a pharmaceutical company, or highly private financial transactions.

All major companies have a backup strategy, so that in the event of a system failure, the data will be available for restore when the failure has been resolved. All large companies perform this operation to tape.

Tape technology has now advanced to the point where 100 Gbytes of data can be stored on a cartridge small enough to fit into a suit pocket. Many companies operate stringent security procedures in their data centres; visitors must be with pass-holding staff, key-code door locks and the like. All of which are subject to abuse resulting in a security mirage. Even if these security measures are implemented, they are totally reliant upon the loyalty of staff.

The most serious threat to sensitive data has arisen through the need for true Disaster Recovery policies. In the event of a catastrophe, where the entire IT environment is inaccessible, tapes held locally will not be available for restore at an alternative site. Many companies therefore remove the backup and archive tapes to their Disaster Recovery sites or to a remote storage facility. It is during transit from site to site when the tapes are at the highest level of risk, be it from organised theft or misplacement.

Data tapes which have been encrypted will have a higher level of security, should they fall into the public domain or the hands of competitors.

What is Encryption?

The word “encryption” has been coined from the word “cryptography” which is derived from the Greek “kryptos” (hidden) and “graphia” (writing). Encryption is the process of transforming text into an unintelligible form called cipher. Data encryption is the process used to hide the true meaning of data.

Reversing the process of encryption is called decryption. Encryption and decryption comprise the science of cryptography as it is applied to the modern computer.

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 follow a specific and unambiguous set of instructions.

Methods of Encryption

There are two types of cryptosystems: secret key and public key. In secret-key cryptography, also referred to as symmetric cryptography, the same key is used for both encryption and decryption.

The most popular secret-key cryptosystem in use today is known as DES, the Data Encryption Standard. IBM developed DES in the middle 1970's and it has been a United States of America Federal Standard since 1976.

In public-key cryptography, each user has a public key and a private key. The public key is made public whilst the private key remains secret. Encryption is performed with the public key, whilst the private key is used for decryption.

The RSA public-key cryptosystem is the most popular form of public-key cryptography. RSA stands for Rivest, Shamir, and Adleman, the inventors of the RSA cryptosystem.

In both cases the encryption can be applied in either a block cipher or stream cipher.

Block Cipher

A block cipher is a type of symmetric-key encryption algorithm that transforms a fixed-length block of plaintext (unencrypted text) data into a block of ciphertext (encrypted text) data of the same length. This transformation takes place under the action of a user-provided secret key.

Decryption is performed by applying the reverse procedure to the ciphertext block, whilst using the same secret key. The fixed length is called the block size, and for many block ciphers, the block size is 64 bits. In the coming years the block size will increase to 128 bits, as processors become more sophisticated.

Iterated block ciphers encrypt a plaintext block by a process that has several rounds. In each round the same transformation, also known as a round function, is applied to the data using a subkey. The set of subkeys is usually derived from the user-provided secret key by a special function. The set of subkeys is called the key schedule. The number of rounds in an iterated cipher depends on the desired security level and the consequent trade-off with performance. In most cases, an increased number of rounds will improve the security offered by a block cipher, but for some ciphers the number of rounds required to achieve adequate security will be too large for the cipher

to be practical or desirable, other than in a dedicated hardware environment.

Stream Cipher

A stream cipher is a type of symmetric encryption algorithm. Stream ciphers can be designed to be exceptionally fast, much faster than any block cipher. While block ciphers operate on large blocks of data, stream ciphers typically operate on smaller units of plaintext, usually bits.

The encryption of any particular plaintext with a block cipher will result in the same ciphertext when the same key is used. With a stream cipher, the transformation of these smaller plaintext units will vary, depending on when they are encountered during the encryption process. A stream cipher generates what is called a keystream, which is a sequence of bits used as a key. Encryption is accomplished by combining the keystream with the plaintext, usually with the bitwise exclusive-OR operation. The generation of the keystream can be independent of the plaintext and ciphertext, yielding what is termed a synchronous stream cipher, or it can depend on the data and its encryption, in which case the stream cipher is said to be self-synchronizing. Most stream cipher designs are for synchronous stream ciphers.

What Is DES?

The National Institute of Standards & Technology (formerly the National Bureau of Standards) issued the Data Encryption Standard (DES) in 1977 to provide an encryption algorithm for use in protecting Federal unclassified information from unauthorised disclosure or undetected modification during transmission, or whilst in storage.

The standard required NIST to conduct a review every five years to determine whether the cryptographic algorithm specified by the standard should be re-affirmed, revised or withdrawn. The first review resulted in the re-affirmation of the standard in 1983; the standard was re-affirmed in 1988 following a second review; the third review was completed in 1993. FIPS 46-2, which was issued following the third review, re-affirmed the DES until 1998.

The DES is based on the work of IBM and has been adopted as the American National Standard X3.92-1981/R1987. The DES is a publicly known cryptographic algorithm that converts plaintext to ciphertext using a 56-bit key. The same algorithm is used with the same key to convert ciphertext back to plaintext, the process called decryption.

The DES consists of 16 "rounds" of operations that mix the data and key together in a prescribed manner using the fundamental operations of permutation and substitution. The goal is to scramble completely the data and key, so that every bit of the ciphertext depends on every bit of the data, plus every bit of the key (a 56-bit quantity for DES).

Authorised users of encrypted computer data must have the key that was used to encrypt the data in order to decrypt it. The unique key chosen for use in a particular application makes the results of encrypting data using the algorithm unique. Using a different key causes different results. The cryptographic security of the data depends on the security provided for the key used to encrypt and decrypt the data.

The outcome of implementing the DES algorithm is that a cipher text is produced which has one key of a sequence of 2^{56} or 70,000,000,000,000,000 (seventy quadrillion).

Figure 1 below shows the fundamental operation of the DES encryption process, (implementing ECB mode).

A 64 Bit block of data is presented to the encryption engine.

An initial permutation of the block is made. The block is then divided into two 32 bit segments, L Block and R Block. Using a 56 bit derivative of the 64 encryption bit key, a complex non-linear operation (hand icon) is performed on R Block.

The modified R Block is then XORED with L block and the resultant fed to the next R Block register. The unmodified R Block is fed to the next L Block register.

With another 56 bit derivative of the 64 bit key, the same process is repeated.

The sequence is performed a total of 16 times before the L Block and R Block segments are recombined and an inverse of the initial permutation is performed.

The result is the 64 bit ciphertext block.

DES Modes Of Operation

The Federal Information Processing Standard (FIPS) defines four modes of operation for the DES that may be used in a wide variety of applications. The modes specify how data will be encrypted and decrypted.

The modes included in this standard are the **E**lectronic **C**ode**B**ook (ECB) mode, the **C**ipher **B**lock **C**haining (CBC) mode, the **C**ipher **F**eed**B**ack (CFB) mode, and the **O**utput **F**eed**B**ack (OFB) mode.

Electronic Codebook (ECB)

In ECB encryption, a plain text data block (D_1, D_2, \dots, D_{64}) is used directly as the DES input block ($11, 12, \dots, 164$). The input block is processed through a DES device in the encrypt state. The resultant output block ($01, 02, \dots, 064$) is used directly as cipher text (C_1, C_2, \dots, C_{64}) or may be used in subsequent ADP applications.

In ECB decryption, a cipher text block (C_1, C_2, \dots, C_{64}) is used directly as the

DES input block $(I_1, I_2, \dots, I_{64})$. The input block is then processed through a DES device in the decrypt state. The resultant output block $(O_1, O_2, \dots, O_{64})$ is the plain text $(D_1, D_2, \dots, D_{64})$ or may be used in subsequent ADP applications.

The ECB decryption process is the same as the ECB encryption process except that the decrypt state of the DES device is used rather than the encrypt state.

Cipher Block Chaining Mode (CBC)

The message to be encrypted is divided into blocks. In CBC encryption, the first DES input block is formed by exclusive-ORing the first block of a message with a 64-bit initialisation vector (IV), i.e., $(I_1, I_2, \dots, I_{64}) = (IV_1 \oplus D_1, IV_2 \oplus D_2, \dots, IV_{64} \oplus D_{64})$. The input block is processed through a DES device in the encrypt state, and the resulting output block is used as the cipher text, i.e., $(C_1, C_2, \dots, C_{64}) = (O_1, O_2, \dots, O_{64})$.

This first cipher text block is then exclusive-ORed with the second plain text data block to produce the second DES input block, i.e., $(I_1, I_2, \dots, I_{64}) = (C_1 \oplus D_1, C_2 \oplus D_2, \dots, C_{64} \oplus D_{64})$. Note that I and D now refer to the second block. The second input block is processed through the DES device in the encrypt state to produce the second cipher text block. This encryption process continues to "chain" successive cipher and plain text blocks together until the last plain text block in the message is encrypted, see figure 3.

In CBC decryption, the first cipher text block of an encrypted message is used as the input block and is processed through a DES device in the decrypt state, i.e., $(I_1, I_2, \dots, I_{64}) = (C_1, C_2, \dots, C_{64})$. The resulting output block, which equals the original input block to the DES during encryption, is exclusive-ORed with the IV (must be same as that used during encryption) to produce the first plain text block, i.e., $(D_1, D_2, \dots, D_{64}) = (O_1 \oplus IV_1, O_2 \oplus IV_2, \dots, O_{64} \oplus IV_{64})$.

The second cipher text block is then used as the input block and is processed through the DES in the decrypt state and the resulting output block is exclusive-ORed with the first cipher text block to produce the second plain text data block, i.e., $(D_1, D_2, \dots, D_{64}) = (O_1 \oplus C_1, O_2 \oplus C_2, \dots, O_{64} \oplus C_{64})$. Note that again the D and O refer to the second block. The CBC decryption process continues in this manner until the last complete cipher text block has been decrypted.

Cipher text representing a partial data block must be decrypted in a manner as specified for the application.

Cipher Feedback Mode (CFB)

A message to be encrypted is divided into data units each containing K bits ($K = 1, 2, \dots, 64$). In both the CFB encrypt and decrypt operations, an initialisation vector (IV) of length L is used. The IV is placed in the least significant bits of the DES input block with the unused bits set to "0's," i.e., $(I_1, I_2, \dots, I_{164}) - (0, 0, \dots, 0, IV_1, IV_2, IV_L)$.

This input block is processed through the DES device in the encrypt state to produce an output block. During encryption, cipher text is produced by exclusive-ORing a K -bit plain text data unit with the most significant K bits of the output block, i.e., $(C_1, C_2, \dots, C_K) - (D_1 \oplus I_1, D_2 \oplus I_2, \dots, D_K \oplus I_K)$. Similarly, during decryption, plain text is produced by exclusive-ORing a K -bit unit of cipher text with the most significant K bits of the output block, i.e., $(D_1, D_2, \dots, D_K) - (C_1 \oplus I_1, C_2 \oplus I_2, \dots, C_K \oplus I_K)$.

In both cases the unused bits of the DES output block are discarded. In both cases the next input block is created by discarding the most significant K bits of the previous input block, shifting the remaining bits K positions to the left and then inserting the K bits of cipher text just produced in the encryption operation or just used in the decrypt operation into the least significant bit positions, i.e., $(I_1, I_2, \dots, I_{164}) = (I_{[K+1]}, I_{[K+2]}, \dots, I_{164}, C_1, C_2, \dots, C_K)$. This input block is then processed through the DES device in the encrypt state to produce the next output block. This process continues until the entire plain text message has been encrypted or until the entire cipher text message has been decrypted.

The CFB mode may operate on data units of length l through 64 inclusive. K -bit CFB is defined to be the CFB mode operating on data units of length K for $K = 1, 2, \dots, 64$. For each operation of the DES device one K -bit unit of plain text produces one K -bit unit of cipher text or one K -bit unit of cipher text produces one K -bit unit of plain text.

An acceptable alternative for 8-bit CFB when enciphering 7-bit entities using an 8-bit feedback path is to insert a "1" bit in bit position one of the 8-bit feedback path, i.e., $(1, C_1, C_2, \dots, C_7)$. This results in a "1" always being placed in bit location 57 of the DES input block. This alternative is called the 7-bit CFB(a) mode of operation.

Output Feedback Mode (OFB)

A message to be encrypted is divided into data units each containing K bits ($K = 1, 2, \dots, 64$). In both the OFB encrypt and decrypt operations, an initialisation vector (IV) of length L is used. The IV is placed in the least significant bits of the DES input block with the unused bits set to "0's," i.e., $(I_1, I_2, \dots, I_{164}) = (0, 0, \dots, 0, IV_1, IV_2, \dots, IV_L)$. This input block is processed through the DES device in the encrypt state to produce an output block. During encryption, cipher text is produced by exclusive-ORing a K -bit plain text data unit with the most significant K bits of the output block, i.e., $(C_1, C_2, \dots, C_K) = (D_1 \oplus O_1, D_2 \oplus O_2, \dots, D_K \oplus O_K)$.

Similarly, during decryption, plain text is produced by exclusive-ORing a K -bit unit of cipher text with the most significant K bits of the output block, i.e., $(D_1, D_2, \dots, D_K) = (C_1 \oplus O_1, C_2 \oplus O_2, \dots, C_K \oplus O_K)$.

In both cases the unused bits of the DES output block are discarded. In both cases the next input block is created by discarding the most significant K bits of the previous input block, shifting the remaining bits K positions to the left and then inserting the K bits of output just used into the least significant bit positions, i.e., $(I_1, I_2, \dots, I_{164}) = (I_{[K+1]}, I_{[K+2]}, \dots, I_{164}, O_1, O_2, \dots, O_K)$. This input block is then processed through the DES device in the encrypt state to produce the next output block.

This process continues until the entire plain text message has been encrypted or until the entire cipher text message has been decrypted. The OFB mode may operate on data units of length 1 through 64 inclusive. K -bit OFB is defined to be the OFB mode operating on data units of length K for $K = 1, 2, \dots, 64$. For each operation of the DES device one K -bit unit of plain text produces one K -bit unit of cipher text or one K -bit unit of cipher text produces one K -bit unit of plain text.

can be compromised.

1 A *brute force* attack is possible. With enough financial and computer resource, every possible key combination can be generated and tested. But, without knowing exactly what the unencrypted data was in byte or binary terms, how will the perpetrator know when the correct cipher key has been found? Assuming the data was generated by an application with a known file structure, in theory the crack should be easier to achieve. That said the application will have to restore the data before it can be seen to be correct. The crack time has now risen from the time taken to generate and apply the keys to a few cipher blocks, to up to 2^{168} restore operations.

Example: If the data on one cartridge takes 10 minutes to restore. At worst case a tape encrypted using ECB DES3 could take up to 7.1⁴⁵ years to decode.

2 A hardware solution could be designed or a software program written if a portion of unencrypted data is known to be on the tape in encrypted form. A full key sequence could be generated and a comparison to the data on tape made. This would again involve multiple tape passes. To avoid incurring the penalty of multiple tape comparisons the data could be copied to disk and the test data compared, however if there was fifty gigabytes of data on the tape and this was copied and compared at disk level, there could be up to 2^{168} fifty gigabyte comparisons

3 If there is access to the cipher key, the data is compromised.

Securing The Key

It is clear that if an encrypted tape becomes available to an unauthorised party, the danger to the data is minimal, unless there is an "inside" source from whom the encryption key used to generate the tape is available.

Security can be introduced on a procedural basis, however this will not protect from a malicious act other than to highlight that persons actions after the data has entered the public arena, such as in a press disclosure.

If however the data is used in a private manor for business or financial advantage it is unlikely the rightful owner of the data will ever know that security has been compromised.

DES does not have the facility to safeguard against disclosure of the

encryption key, therefore it is necessary for any device which introduces encryption to overcome this problem.

A system whereby both a "User Key" and a "Device Key" must be implemented. In this case a key is input at each session by the user/operator and the encryption unit itself also has an internal key unique to itself. The unit accepts the user key and performs a non linear algorithm with its own key. The resultant product is a composite key that is used to encrypt the data. More to the point data can only be decrypted by a combination of the correct encryption unit combined with the correct user key.

Figure 4 illustrates the principle.

Commercially Available Encryption Devices

During research for this publication it became apparent that almost all of the encryption products on the market place today are of a software nature aimed at streaming communications in particular network and web security.

Those products pertaining to stored data security are almost universally software orientated without a hardware key component, thereby rendering them less effective in the event of malicious actions by personnel with access to the cipher keys.

Of the few remaining products that have a hardware component that ensures the integrity of the encryption, all, apart from one product, are aimed at small systems such as a notebook computer, which can easily be stolen.

At this time only one product on the market is suitable for securing the integrity of data stored, or transported in tape format, which does not rely on software generated encryption of the data stream: PARANOIA.

The PARANOIA range of in-line SCSI encryption units has an integral data key "chip" which combines with the user key to produce a composite key for encryption of the data stream. The range has been developed specifically for the protection of tape subsystems with sensitive data.

Problems with incorporating DES.

The main problem with incorporating DES into a backup environment is the problem of speed. Triple DES requires 48 iterations of an 8-byte data pattern, plus adjustments for the initial permutation, the final permutation, the feed back of data and inclusion of the IV.

Software or firmware solutions are too slow; typically quoted figures are in the order of 6.5 seconds to encrypt a 32-kilobyte block.

Even using dedicated hardware data encryption, because of the number of iterations having to be performed by the hardware, this will also slow any backup.

Data backup is viewed as a laborious, tedious but a necessary part of an IT department. Backup windows are assigned to a minimum, this being the amount of time the IT department has allocated for it's daily backup. The native transfer rate of a backup device is the speed at which a device can physically write data to tape. Modern day backup devices have a native transfer rate in the order of 15 Mbytes/sec. All modern day devices also include compressors, compressing data before it is written to tape. Typically user data compresses in the range of 2 or 3 to 1.

If we take a backup device with a native transfer rate of 10 Mbytes/sec and the data compresses at the rate of 2:1, then over 70 Gigabytes of data can be backed up in 1 hour. The IT department, wanting a 140 Gigabyte daily backup would then assign a backup window of 2 hours.

However encrypted data using any of the above feedback modes does not compress. Using for example a 2 Mbyte block of data of the same value which would compress at 99:1 or greater, will not compress when encrypted using any of the feedback modes.

Most backup devices when confronted with encrypted data go into negative compression in the range of 0.9:1. So the backup window of 2 hours now extends, instead of achieving 20 Mbytes/sec, the rate is cut to 9 Mbytes/sec, extending the backup to over 4 hours.

Because of this negative compression, the data also grows in size. Where once a single tape is required to complete the backup, operator intervention may now be required to insert a second tape to complete the backup.

How the Paranoia2 incorporates DES.

The Paranoia2 is effectively a SCSI-to-SCSI DES converter. It is transparent to the host computer, sited between the SCSI drive and the host computer.

The host and drive interface will operate at SCSI Ultra 2 rates, 80 Mbytes/sec. To overcome the above-mentioned difficulties of the use of DES, SCSI data received from the host is firstly compressed using a hardware compressor, which will compress and decompress data at 75 Mbytes/sec.

The data is then encrypted using a hardware encryption engine. Because as mentioned earlier of the number of iterations needed to be performed by the encryption engine, the maximum encryption rate drops to 20 Mbytes/sec. To overcome this, the P2 uses dual encryption engines.

Whilst one engine is ciphering it's 8 bytes of data, the second engine is receiving it's next 8 bytes of data. This effectively produces a dual encryption engine operating at 40 Mbytes/sec.

Because the compressor is sited before the encryptors, and with the encryption engines ciphering compressed data, with highly compressed data, sustained data rates approaching the speed of the compressor can be achieved.

In laboratory testing, using a slow SCSI device, where the maximum sustained transfer rate for highly compressible data was 6 Mbytes/sec when connected directly to the host. When the P2 was connected between the host and the drive, sustained transfer rates in excess of 50 Mbytes/sec were observed. This was because of the PT's high SCSI transfer rate, and it's ability to compress data at 75 Mbytes/sec.

As indicated earlier, the possibility of de-ciphering the DES key is 72 quadrillion to 1. Because the Paranoia2 uses twin encryption engines, two sets of keys may be entered via the Gui interface. One set of keys for encryption engine 1, the second for encryption engine 2. Compression is always enabled, unless both keys are correct, the data will never de-compress.

To decipher a tape using twin keys and compression, the amount of possibilities to de-cipher the tape is far in excess of 72 quadrillion to 1. Any infiltrator would need to know the compression algorithm, both keys, and the means of splitting data between encryption engines.

A further security issue is included in the Paranoia2: what happens if someone steals a tape and the security keys?

The Paranoia2 also includes a hardware key; keys are assigned per customer. The hardware key can be manufactured at Paranoia2 build time, or manufactured by the customers. The hardware key modifies the GUI key, such that a tape recorded on a specific customer's site cannot be read on another customer's site when using the Paranoia2.

Disclaimer

This document is for information purposes only. This document does not constitute the whole or part of a procedure or plan for the implementation of an encryption system.

Much of the information produced in this document is derived from public domain information sites. Wherever possible the author has attempted to qualify details published. However in the event of any statement being incorrect the author is does not accept liability for any loss of data as a result of statements made herein.

Use of this document

This document has been written with the aim of informing the business community about the principles and uses of data encryption. It may be distributed and used for this purpose by others, on condition that attribution is made and authorship recognised.