Il DES e gli algoritmi di cifratura a blocchi

Leonardo Miliani | 27 dicembre 2007

Molti usano la crittografia in campo informatico senza sapere da dove essa tragga origine. Ebbene, il viaggio che iniziamo oggi ci porterà ad analizzare quelle che possiamo considerare le sue vere radici: il primo algoritmo che prenderemo in considerazione è visto un po' come il capostipite degli algoritmi di crittazione, nonché uno dei più anziani ancora in circolazione: il **DES**, acronimo di **Data Encryption Standard**. Esaminandolo vedremo anche i principali modi d'uso di un algoritmo a blocchi nonché alcune tra le più diffuse funzioni crittografiche, così da avere un'infarinatura a livello generale che arricchirà il bagaglio culturale di ogni utente di computer alle prime armi. Siccome l'argomento è molto lungo da trattare, ho diviso l'articolo in 2 parti: nella prima, che potete leggere qui di seguito, viene presentato il DES, mostrando come è nato e come è strutturato. La storia del DES inizia nel 1972, quando l'**NSA*** (la *National Security Agency*, l'organizzazione di intelligence americana preposta alla sicurezza dei dati governativi) chiese al National Bureau of Standards, NBS, adesso divenuto **NIST** (*National Institute of Standards and Tecnology*), di sviluppare un algoritmo di crittazione da adottare come standard per la cifratura dei documenti governativi riservati. Fu indetto un primo bando di concorso, che però vide lo scarto di tutti i codici presentati perché non rispondenti ai criteri stabiliti dall'NSA. Ne fu quindi indetto un secondo: fu allora che *IBM* presentò un algoritmo di cifratura denominato **DEA** (**Data Encryption Algorithm**), basato su un suo precedente codice, il *Lucifer*. Il DEA piacque subito perché rispondeva ai suddetti criteri. Però doveva subire ancora qualche lavoro di affinamento. E fu così che IBM lavorò sull'algoritmo fino a presentare la versione che fu alla fine adottata dal Governo Americano nel 1977 con il nome ufficiale di **Data Encryption Standard**, abbreviato in DES. Nel 1983 fu riconfermato come standard, e così anche nel 1988 e nel 1993. Nel 1999, dopo alcuni tentativi di crack portati a termine nel corso dei precedenti 12 mesi, il Governo Americano decise di adottarne una versione irrobustita denominata **Triple-DES**, in cui la cifratura si otteneva eseguendo 3 volte il DES con 3 chiavi differenti (per un totale di 168 bit complessivi). Solo nel 2002 il NIST ha selezionato come standard un nuovo algoritmo, il **Rijndael**, divenuto poi l'**Advanced Encryption Standard**, abbreviato **AES**, per pensionare il vecchio DES. Ma dobbiamo arrivare al 2004 affinché una nota decreti l'abbandono ufficiale dell'algoritmo da parte del Governo USA in favore del suo più forte ed efficiente successore: si conclude solo allora la sua onorata carriera ventennale.

Analisi generale del DES

Come riportato nella scheda introduttiva visibile a lato, il DES è un algoritmo di cifratura a blocchi a chiave simmetrica. Spieghiamo questi termini… Un **algoritmo di cifratura a blocchi** è un algoritmo che tratta i dati in blocchi di dimensioni predefinite: nel caso del DES, ogni ciclo opera su 64 bit di dati. Se la lunghezza del messaggio da cifrare non corrisponde ad un multiplo della lunghezza del blocco, bisogna eseguire un'operazione di *padding* (o riempimento): bisogna cioè aggiungere dei bit senza significato alla fine del testo da cifrare per far sì che la sua lunghezza diventi *modulo lung_blocco* (vale a dire che la divisione della lunghezza del testo del messaggio per la lunghezza del blocco che può essere trattato dall'algoritmo deve dare zero). Esempio: se il messaggio è lungo 100 bit, per eseguire il DES bisogna portarne la lunghezza a 128 bit [64*2] aggiungendo 28 bit nulli (128 mod 64 = 0). Un **algoritmo a chiave simmetrica**,

SCHEDA	
Nome	DES (Data Encryption Standard)
Data di rilascio	1975
Sviluppatore	IBM
Tipo	algoritmo a blocchi (block cipher)
Chiave	Simmetrica
Dimensioni chiave	56 bit
Dimensioni blocco	64 bit
Struttura interna	Rete di Feistel S-Boxes Key-schedule
Cicli	16

Tabella 1: Le principali caratteristiche del DES

detto anche a chiave privata, è un algoritmo in cui la chiave di cifratura è la stessa della chiave di decifratura. Bisogna fare in modo che la chiave resti il più possibile segreta: non si deve in nessun modo divulgarla pubblicamente, altrimenti tutti quelli che ne entrano in possesso possono decifrare il messaggio cifrato. Tornando al DES fu, come detto, sviluppato partendo dal Lucifer, che adottava una chiave a 256 bit. IBM propose inizialmente per il DES una chiave a 128 bit ma l'NSA prima "suggerì" l'adozione di una chiave a 64 bit e poi fece introdurre una modifica all'algoritmo in modo che venissero in pratica usati solo **56 bit**; gli altri 8 bit dovrebbero essere utilizzati per operazioni di controllo della parità ma sono generalmente scartati. Come altri algoritmi a blocchi, il DES non è utilizzabile per lavorare su un testo di lunghezza arbitraria dato che è nato per operare su un blocco di dati di lunghezza prefissata. Bisogna utilizzarlo in una delle cosiddette "modalità operative", vale a dire un sistema di codifica basato sull'algoritmo in questione.

Le modalità operative del DES

Secondo i documenti ufficiali che segnano le linee guida dello standard DES, le modalità utilizzabili per questo algoritmo sono: **Electronic CodeBook** (ECB), **Cipher Block Chaining** (CBC), **Cipher FeedBack** (CFB), **Output FeedBack** (OFB). Esistono anche altre modalità molto diffuse, il Vettore di Inizializzazione, VI, e la modalità Counter (CTR), che però non approfondisco in quanto non usate con il DES. Vediamo brevemente le differenze fra di esse (*Ci* è il blocco di dati cifrato, *E* l'operazione di cifratura, *K* è la chiave, *Pi* il blocco di dati in chiaro, *k* il numero di blocchi da cifrare):

- **ECB**
  E' questa la modalità di utilizzo più semplice di un algoritmo a blocchi, ma è anche la più insicura! Tecnicamente si tratta di prendere un blocco di dati e di cifrarlo separatamente con la chiave:
  $C_i = E(K, P_i)$ *con* $i = 1...k$
  Perché è insicura? Perché lo stesso blocco di dati cifrato con la stessa chiave fornirà sempre lo stesso risultato! Quindi, NON va mai utilizzata!

- **CBC**
  La modalità CBC (concatenamento dei blocchi) è la più utilizzata. Si tratta di effettuare uno XOR fra il blocco di dati in chiaro da cifrare ed il blocco di dati cifrati precedentemente, in modo da introdurre un fattore di casualità che renda più difficoltosa l'analisi del testo cifrato ed evitando i problemi dell'ECB:
  $C_i = E(K, P_i \text{ XOR } C_{i-1})$ *con* $i=1..k$

- **CFC**
  Questa modalità, molto simile alla precedente, trasforma di fatto l'algoritmo a blocchi in uno a flusso (stream cipher) auto-sincronizzante. L'operazione di XOR viene eseguita tra l'attuale blocco di dati in chiaro ed il blocco di dati cifrato precedentemente:
  $C_i = E(C_{i-1}) \text{ XOR } P_i$ *con* $i=1..k$
  La modalità CFC può essere utilizzata per cifrare/decifrare blocchi di dati da un flusso continuo dove serve bassa latenza, come ad esempio un canale di *streaming*: questo perché l'algoritmo opera utilizzando un blocco di dati già cifrato (che può perciò essere mantenuto in memoria tramite una *cache*) e quello in chiaro appena passatogli.

- **OFB**
  A differenza delle modalità finora mostrate, l'OFB opera in maniera diversa, non usando mai il messaggio diviso a blocchi ma trasformandolo in un flusso pseudo-casuale di bit denominato *keystream*, il quale viene poi combinato per cifrare il testo in chiaro. Questa modalità trasforma l'algoritmo a blocchi in un vero e proprio algoritmo a flusso sincronizzato (s*ynchronous stream cipher*). L'algoritmo viene inizializzato con un **VI**, **Vettore di Inizializzazione** (che può essere un contatore o un valore generato casualmente,

l'importante è che non venga mai utilizzato lo stesso VI più di una volta con la stessa chiave, altrimenti si possono avere gli stessi problemi di sicurezza della modalità ECB):

$K_0 = VI$

$K_i = E(K, K_{i-1})$ *con i=1..k*

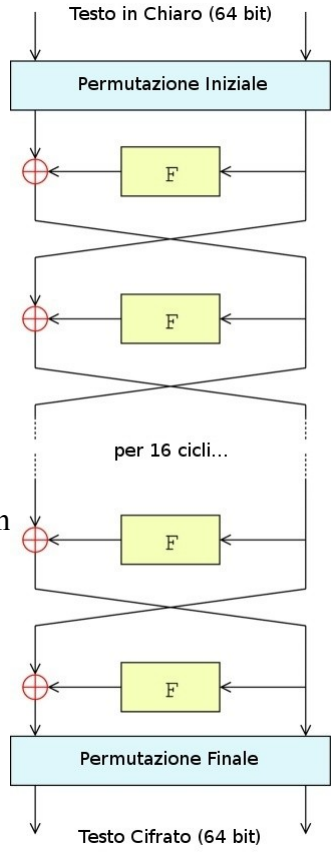
$C_i = P_i$ XOR K_i

Questa modalità offre alcuni vantaggi: non c'è bisogno di nessuna operazione di padding (riempimento) perché la keystream è lunga quanto il messaggio stesso, per cui se anche l'ultima parte di esso non riempie tutto un blocco di dati, vengono cifrati esclusivamente i bit esistenti; la decifratura corrisponde esattamente alla cifratura perciò non c'è bisogno di scrivere 2 funzioni differenti per le 2 operazioni.

-

Analisi strutturale

La struttura del DES è molto lineare (come si vede dalla Figura 1): ogni blocco di dati in chiaro (lungo 64 bit) passa attraverso 16 identici cicli del processo di crittazione (denominati *"rounds"* in inglese). Alla fine si hanno 64 bit di dati cifrati. La *Permutazione Iniziale* e la *Permutazione Finale* sono 2 processi inversi (la Permutazione Finale fa esattamente le stesse operazioni di quella Iniziale ma in ordine inverso) che non hanno crittograficamente nessun significato: si pensa che siano stati inseriti per poter caricare e scaricare più facilmente i blocchi dati nell'hardware degli anni '70 e contemporaneamente per rallentare la velocità del DES se eseguito in software. Il simbolo rosso ⊕ indica uno **XOR,** cioè un OR esclusivo. La **funzione Feistel** (rappresentata dal blocco giallo con la lettera F) rappresenta il cuore del DES: riceve in input metà del blocco di dati, che viene cifrato con parte dei bit della chiave. L'output della funzione viene poi ricombinato con l'altra metà del blocco di dati ancora in chiaro e ne vengono invertite le posizioni prima del ciclo successivo (la metà di destra diventa la metà di sinistra e viceversa). L'ultimo ciclo non inverte le 2 metà, così che i dati abbiano alla fine la stessa posizione di partenza: questa caratteristica della struttura del Feistel rende i processi di cifratura e decifratura molto simili. Ma vediamo in dettaglio le varie strutture interne alla base del DES.



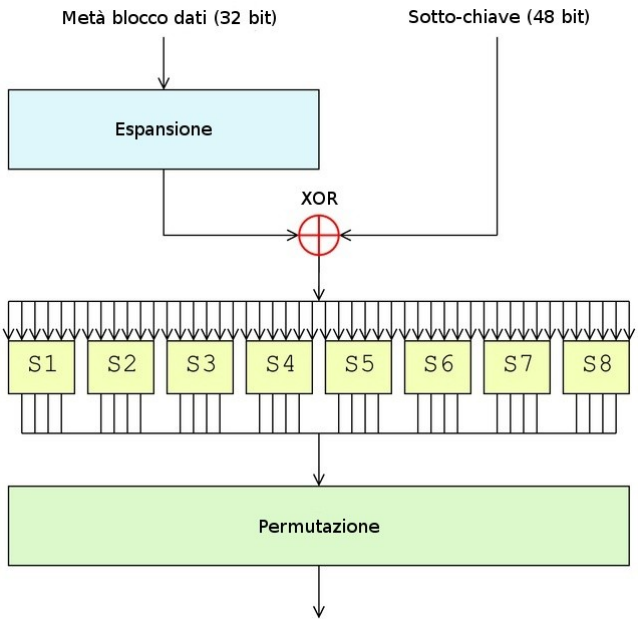
La funzione Feistel

Questa funzione, nota anche come **Rete di Feistel**, prende il nome dall'omonimo crittografo che lavorava in IBM e che aveva inventato questo algoritmo come cifrario a sé stante. La prima utilizzazione della funzione di Feistel si ebbe nel Lucifer, il progenitore del DES. Generalizzando, la funzione Feistel opera nel seguente modo: prende un blocco di dati ed una chiave come input; effettua delle operazioni di riordinamento dei bit (chiamate Permutation Boxes o *P-Boxes*), applica delle semplici funzioni non lineari di sostituzione dei bit (indicate come Substitution Boxes o S-Boxes), e mescola linearmente i bit con operazioni di XOR. Alla fine del ciclo si ottiene il dato cifrato in output.

La particolarità della rete di Feistel sta nel fatto che le funzioni di cifratura e decifratura sono spesso molto simili, se non proprio identiche: in questo ultimo caso, la funzione può essere usata anche per decifrare il messaggio, utilizzandolo come input: si otterrà come output il testo in chiaro. La funzione Feistel del DES esegue 4 operazioni (vedi figura):

- **Espansione:**
  viene preso metà blocco dei dati da elaborare (32 bit), che viene poi espanso a 48 bit

effettuando delle semplici operazioni di duplicazione dei suoi bit;

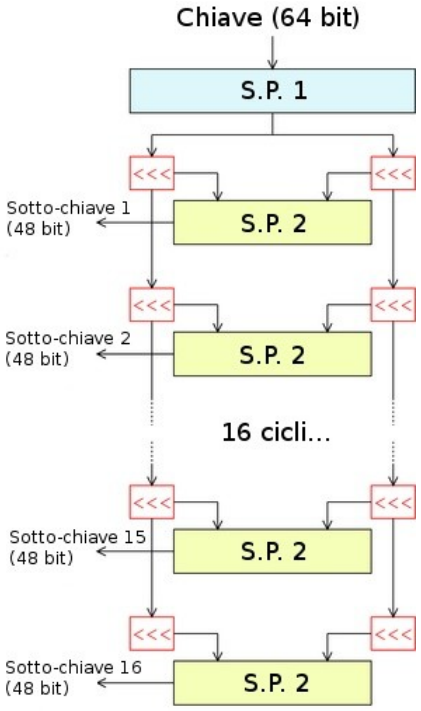


- **Mescolamento**:
  l'output dell'operazione di espansione viene combinato con una sotto-chiave di 48 bit (ottenuta mediante un procedimento denominato *key-schedule*, gestione della chiave, descritto più avanti e che fornisce 16 sotto-chiavi diverse, una per ogni ciclo del DES), utilizzando una semplice operazione di OR esclusivo (XOR);
- **Sostituzione:**
  dopo il mescolamento con la sotto-chiave, il blocco viene diviso in 8 porzioni di 6 bit ciascuna e poi processato da 8 *S-Boxes* (indicate con Sx in figura). Ogni S-box sostituisce i 6 bit in ingresso con 4 bit scelti arbitrariamente da una tabella di sostituzione. Senza l'operazione di sostituzione la funzione Feistel sarebbe lineare e quindi facilmente violabile.
- **Permutazione:**
  l'ultima parte della funzione ricostruisce il blocco di dati riassemblando i bit ricevuti come output dalle S-Boxes.

Il key-schedule

Il *key-schedule* è un algoritmo che genera le sotto-chiavi a partire dalla chiave primaria (vedi figura). Dei 64 bit originali ne vengono selezionati solo 56 tramite la funzione S.P. 1 (Scelta Permutata 1); i restanti 8 bit vengono scartati (quasi sempre) o utilizzati come bit di controllo parità. I 56 bit vengono poi divisi in 2 parti di 28 bit ciascuna, ognuna delle quali viene poi trattata in maniera separata dall'algoritmo. Durante i 16 cicli, il key-schedule ruota, a seconda del ciclo, entrambe le metà così generate di 1 o 2 bit verso sinistra (i segni <<< in rosso nello schema) e poi seleziona con la funzione S.P. 2 (Scelta Permutata 2) i 48 bit da fornire come sotto-chiave alla funzione Feistel: 24 bit sono prelevati dalla pipeline di sinistra e 24 bit da quella di destra. Grazie alla rotazione, ogni sotto-chiave è formata da una diversa combinazione di bit (ogni bit è usato in 14 sotto-chiavi su 16).

In caso di decrittazione, il key-schedule genera le sotto-chiavi in ordine inverso e ruotando i bit verso destra.



La scomoda "presenza" dell'NSA e la robustezza del DES

La scarsa robustezza del DES deriva dalla limitata lunghezza della sua chiave. Molte illazioni sono state fatte circa la scelta eseguita dall'NSA di imporre una chiave così corta, quando già il Lucifer operava con chiavi a 256 bit. La più ricorrente di esse è la diceria secondo la quale l'NSA era all'epoca già in possesso (unica al mondo) di elaboratori elettronici talmente potenti in grado di violare un testo cifrato con una chiave a 56 bit.

Un'altra voce molto diffusa è quella secondo cui le S-Boxes contenessero una sorta di *backdoor*,

vale a dire uno schema nascosto che poteva permettere all'NSA di decifrare qualunque messaggio protetto con il DES. Secondo i "sostenitori del complotto", l'NSA fece pressioni affinché l'algoritmo che doveva essere utilizzato come standard fosse sicuro verso tutti gli attacchi allora noti al grande pubblico ma non verso i loro… Qui però entriamo nella trama di un film di spionaggio. La verità sembra essere che l'NSA lavorò al rafforzamento delle S-Boxes con tecniche segrete proprio perché aveva imposto una chiave molto corta. Difatti uno sviluppatore del DES, *Alan Konheim*, affermò che:

"spedimmo le S-Boxes a Washington. Ce le rispedirono completamente differenti."

A sostegno di questa teoria c'è il fatto che le S-Boxes del DES sono risultate molto robuste contro la **crittanalisi differenziale****, una tecnica crittanalitica che è stata scoperta solo agli inizi degli anni '90. Secondo i crittanalisti le S-Boxes mostrano una robustezza assolutamente inusuale e "studiata" contro questo tipo di attacco: esse sono nate, cioè, proprio per resistere ad un attacco che ufficialmente all'epoca nessuno conosceva!
Steven Levy, giornalista americano specializzato in informatica e crittografia, ha pubblicato nel 2001 un libro intitolato *"*Crypto*"* dove riporta le dichiarazioni di alcuni crittografi che lavorarono al progetto DES. Secondo quanto scritto in quel testo i ricercatori IBM scoprirono la tecnica di attacco denominata crittanalisi differenziale già nel 1974 ma la tennero segreta su ordine dell'NSA. *Don Coppersmith*, crittografo IBM che lavorò alle S-Boxes del DES, ha dichiarato infatti a Levy che la decisione di IBM di secretare quei documenti

"fu presa perché la [crittoanalisi differenziale] poteva essere uno strumento molto potente, usato per violare un sacco di schemi, e c'era la convinzione che quelle informazioni rilasciate al pubblico dominio potessero compromettere seriamente la sicurezza nazionale".

Walter Tuchman, capo del progetto DES e sviluppatore del Triple-DES, ha rilasciato invece questo commento:

"Loro [l'NSA, N.d.A.] ci chiesero di marcare tutti i nostri documenti [relativi alla crittoanalisi differenziale, N.d.A.] con la dicitura 'Confidenziale'… Apponemmo poi un numero su ognuno di essi e li chiudemmo in un luogo sicuro, perché erano considerati come 'Classificati' dal governo USA. Ci dissero di farlo, così noi lo facemmo."

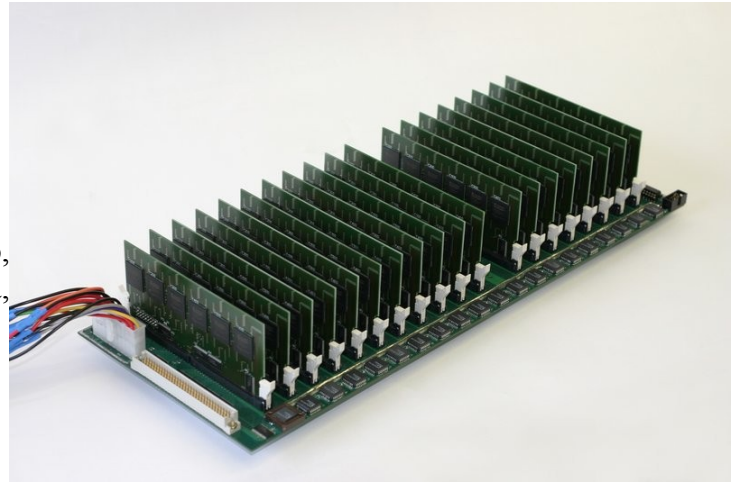
Attacchi al DES e sua violazione
Alla fine degli anni '80 l'unico attacco noto al DES era quello condotto con la **forza bruta** (*brute force*): sfruttare gli elaboratori elettronici per provare tutte le possibili combinazioni di caratteri e ricostruire così la chiave di cifratura. Ma la potenza di calcolo dei computer di allora non era sufficiente per portare a termine un simile attacco al DES in un tempo ragionevole né i costi erano abbordabili.

Nel 1977, infatti, *Whitfield Diffie* e *Martin Hellman* (i "padri" della crittografia a chiave pubblica) proposero un calcolatore in grado di scovare una chiave DES in un solo giorno. Peccato che realizzarlo costasse "appena" 20 milioni di dollari! Nel 1993 la cifra era scesa ad 1 milione di dollari, stando al "preventivo" presentato dal crittanalista canadese Michael J. Wiener per una sua macchina che sarebbe stata capace di violare il DES in 7 ore.
Contemporaneamente, agli inizi degli anni '90 comparivano le prime crittanalisi del DES che contemplassero anche altri tipi di attacco oltre alla forza bruta. I primi a lavorare in tal senso furono *Eli Biham* e *Adi Shamir*, due crittoanalisti israeliani che nel 1990 scoprirono (pubblicamente) l'attacco a crittanalisi differenziale: nel 1992 in un loro documento affermavano che con tale attacco si poteva violare il DES, ma serviva conoscere però almeno 2^{47} testi in chiaro, un valore

assolutamente irrealistico. Nel 1994 il crittanalista giapponese *Mitsuru Matsui* scriveva che con un attacco condotto con la **crittanalisi lineare** (una tecnica da lui inventata agli inizi del 1990 e che si basa sulla ricerca di approssimazioni affini al funzionamento di un algoritmo crittografico) si poteva violare il DES, ma servivano però 2^{43} testi in chiaro noti. Anche in questo caso l'attacco restava a livello teorico.

Il progredire della tecnica ha portato la potenza elaborativa dei calcolatori a crescere in maniera esponenziale (secondo la ben nota *legge di Moore*, la potenza di calcolo raddoppia ogni 18 mesi). E così gli attacchi a forza bruta si facevano negli anni sempre più efficaci, fino a quando, nel 1997, il Progetto **DESCHALL** riusciva a violare per la prima volta ed in circa 96 giorni di calcolo un testo cifrato con il DES (ed a portarsi a casa il premio di 10.000 dollari messo in palio da *RSA Security*, che aveva indetto una gara per riuscire a forzare il vecchio algoritmo) utilizzando un software che girava durante i tempi di *idle* su di una rete di centinaia di migliaia di computer collegati via internet. Nel 1998 la *Electronic Frontier Foundation* costruiva una macchina denominata **EFF DES cracker** (1800 CPU per un costo di 250.000 dollari), poi ribattezzata *Deep Crack*, che recuperava una chiave DES in poco più di 2 giorni; una seconda versione migliorata effettuava l'operazione in circa 22 ore.

Nel marzo del 2007 le Università di Bochum e Kiel, in Germania, hanno assemblato **COPACABANA** (vedi figura a lato), una macchina composta da 120 chip FPGA di tipo XILINX Spartan3-1000 operanti in parallelo, comunemente disponibili in commercio, assemblati su 20 DIMM da 6 chip l'una, ed appositamente programmata per lavorare sul DES. Con soli 10.000 dollari di spesa, è stato possibile recuperare una chiave DES in circa 6,4 giorni di calcolo.



Conclusioni

Il DES, come si è visto, è stato un vero pioniere della crittografia. Grazie ad esso schiere di studiosi si sono cimentati per anni nello studio della sua struttura e della sua robustezza: ed è proprio studiando il modo di violarlo che è stato dato un enorme impulso a quella scienza informatica denominata crittoanalisi che, all'epoca della pubblicazione del DES, vedeva fra le sue fila pochissimi studiosi. Il DES, nel bene o nel male, ha talmente segnato la scena crittografica che anche oggi, a più di 20 anni di distanza dalla sua comparsa sulla scena, è utilizzato come termine di paragone per qualsiasi algoritmo a blocchi di nuova concezione. Non solo, ma grazie alle debolezze scovate nel DES ed al modo per attaccarlo crittografi di tutto il mondo hanno via via perfezionato la tecnica rilasciando algoritmi di crittazione sempre più efficienti, rapidi e sicuri. E' grazie anche al DES, insomma, se oggi la crittografia ha raggiunto i livelli che le competono.
Bruce Schneier, uno dei "guru" della crittografia mondiale, ha dichiarato:

Il DES è il più importante algoritmo mai realizzato. Dato che aveva il pedigree dell'NSA, era ritenuto universalmente "sicuro". *E' stato anche il più studiato algoritmo di cifratura mai inventato e molti crittografi* "sono andati a scuola" *sul DES. In quasi tutti i nuovi algoritmi di cifratura in uso oggi è possibile trovare le loro radici che affondano nel DES, ed i documenti che analizzano differenti aspetti del DES continuano anche ai giorni nostri ad essere pubblicati.*

D’altro canto, gli acciacchi della vecchiaia non si nascondono, ed il DES non è da meno. Come si è visto, la tecnologia oggi esistente permette di recuperare la chiave di cifratura del DES con hardware alla portata di tutti: pertanto, l’uso di questo algoritmo deve essere limitato al puro scopo accademico. Per la protezione “seria” dei dati è meglio affidarsi senza nessun dubbio a qualcosa d’altro e che sia, nel contempo, analizzato come e quanto il DES (leggi: AES).

*: ‘NSA è un’agenzia di intelligence americana, al pari di CIA ed FBI, che è stata fondata nel 1952 con lo scopo preciso di proteggere tutte le trasmissioni governative e militari e, contemporaneamente, di controllare tutte le trasmissioni civili. Non c’è in America una telefonata, una e-mail o qualunque altro scambio di dati effettuato con qualsiasi canale che non venga monitorato dall’NSA.

**: la crittoanalisi differenziale è lo studio di come le differenze presenti in un testo fornito in input ad un algoritmo di crittazione influenzino il suo output. Studiando i risultati ottenuti si riesce a ricostruire la struttura interna delle trasformazioni e permutazioni effettuate, scoprendo dove l’algoritmo non manifesta comportamenti casuali, e sfruttando queste proprietà per ricostruire la chiave segreta.