S E N S ' 2 0 0 6
Second Scientific Conference with International Participation
SPACE, ECOLOGY, NANOTECHNOLOGY, SAFETY
14 – 16 June 2006, Varna, Bulgaria

ОПИСАНИЕ И ПРИЛОЖЕНИЕ НА СТАНДАРТИЗИРАНИЯ АЛГОРИТЪМ ЗА ШИФРИРАНЕ НА ДАННИ (DES)

Адриана Бороджиева

Русе 7017, ул. „Студентска" № 8, Русенски университет „Ангел Кънчев",
e-mail: aborodjieva@ecs.ru.acad.bg

Ключови думи: криптосистеми, шифриране на данни, стандарт.

Резюме. *Статията разглежда подробно стъпките на стандартизирания алгоритъм за шифриране на данни DES (Data Encryption Standard), който все още е най-масово използваният шифър в света. Днес алгоритъмът DES е загубил значимостта си като стандарт, най-вече заради недостатъчната си дължина на ключа от 56 бита и уязвимостта от диференциален и линеен криптоанализ. Обаче, DES остава най-обстойно анализираният криптографски алгоритъм в литературата, тъй като обхваща в себе си основния подход за проектиране на сигурни блокови шифри и на негова основа се изгражда алгоритъмът Triple-DES, намиращ широко приложение в криптосистемата PGP.*

ВЪВЕДЕНИЕ

Стандартизираният алгоритъм за шифриране на данни DES (*Data Encryption Standard*) е най-обстойно анализираният криптографски алгоритъм в света. Той все още е и най-масово използваният шифър в практиката. Разработен е от екип на IBM и е възприет за федерален стандарт от NIST (National Institute of Standards and Technology) през ноември 1976г., а през януари 1977г. се публикува и официалното му описание. Под названието *DEA* (*Data Encryption Algorithm*), DES е одобрен за стандарт от ANSI през 1981г. Препоръчва се за използване и от Американската асоциация на банките (ABA). Вграден е в много хардуерни и софтуерни продукти [1].

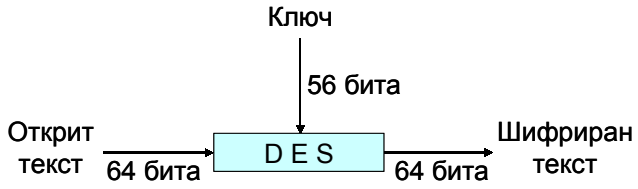
Днес базовият алгоритъм DES е загубил значимостта си като стандарт, най-вече заради недостатъчната си дължина на ключа от 56 бита и уязвимостта от диференциален и линеен криптоанализ [1]. Обаче, на негова основа е изграден алгоритъмът *Triple-DES*, намиращ широко приложение в криптосистемата *PGP* (*Pretty Good Privacy*, буквално "*твърде добра секретност*") – програма за обезпечаване на секретността, създадена от Филип Цимерман (Philip Zimmerman) и публикувана през 1991г. като безплатно програмно осигуряване [2].

Алгоритъмът Triple-DES, прилагащ схемата за трикратно шифриране EDE (Encrypt-Decrypt-Encrypt) с DES при използване на два или три различни 56-битови ключа, става популярен в отговор на нарасналите изчислителни мощности, за които 56-битовата дължина на ключа на DES е вече преодолима. При Triple-DES, въпреки, че проблемът с малката дължина на ключа се решава успешно, сложността и времето за шифриране и дешифриране значително нарастват. Поради високата сигурност, липсата на патентни ограничения и базирането на стандарта DES, Triple-DES е подходящ за приложения, при които времето за реализация не е критично [1].

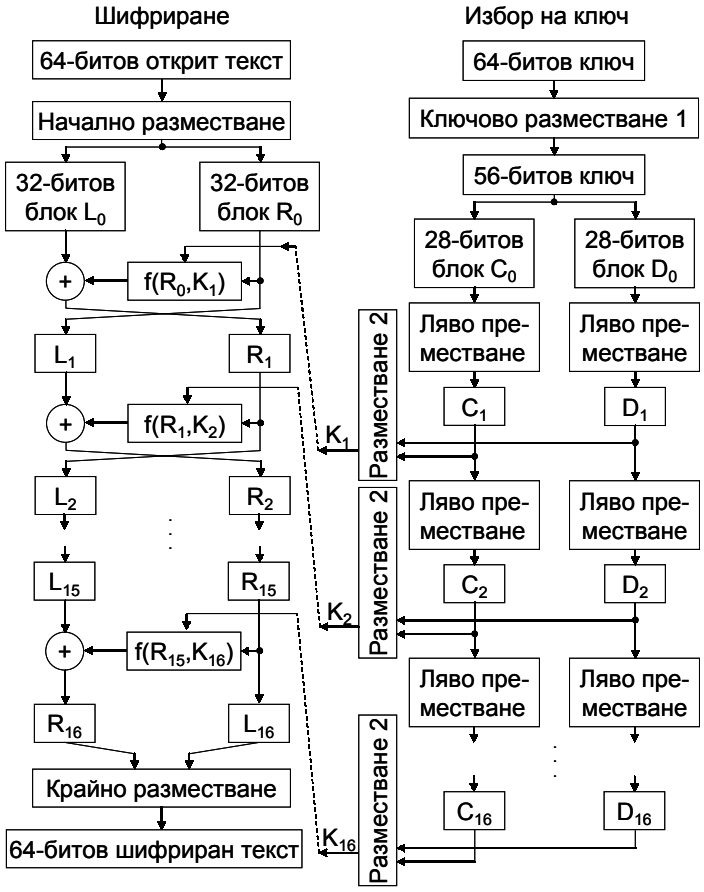
ОПИСАНИЕ НА АЛГОРИТЪМА DES

На фиг.1 е представен *стандартът за шифриране на данни* (*Data EncryptionStandard – DES*) във вид на блокова система за шифриране, която има

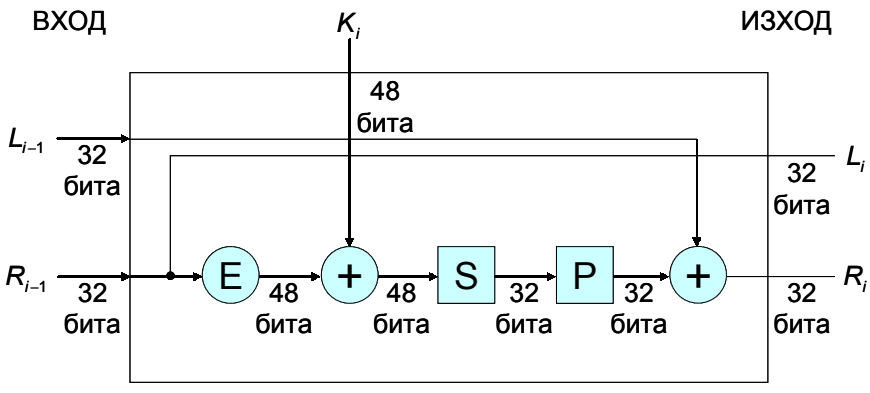
азбука от 2^{64} символа (от гледна точка на система „вход-изход”). Входен блок от 64 бита, явяващ се в тази азбука символ на открития текст, се заменя с нов символ на шифрирания текст. Функциите на системата са показани във вид на блокова диаграма на фиг.2.



Фиг.1. Стандарт за шифриране на данни (DES) във вид на блокова система за шифриране



Фиг.2. Функции на стандарта за шифриране на данни



Фиг.3. Стандартен съставен блок

Алгоритъмът на шифриране започва с началното разместване на 64 бита от открития текст, описано в табл.1, която се чете отляво надясно и отгоре надолу така, че след разместването битовете $x_1, x_2, \ldots, x_{64}$ стават $x_{58}, x_{50}, \ldots, x_7$.

След това начално разместване започва основната част на алгоритъма за шифриране, състояща се от 16 итерации, които използват стандартен блок, показан на фиг.3. Трябва да се отбележи, че и този алгоритъм, също като повечето блокови алгоритми, се базира на отдавна известната идея на Хорст Фейстел, при която обработваните блокове предварително се разделят на две части *L* (*лява част*) и *R* (*дясна част*) и процедурите за шифриране и дешифриране на всеки блок се изразяват в итеративни преобразувания на тези части.

За преобразуването на 64 бита входни данни в 64 бита изходни данни, определени като 32 бита на лявата и 32 бита на дясната половина, съставният стандартен блок, показан на фиг.3, използва ключ от 48 бита.

Изходът на всеки стандартен блок става вход на следващия стандартен блок. Входните 32 бита на дясната половина R_{i-1} без изменение се подават на изхода и стават 32 бита на лявата половина L_i. Тези R_{i-1} бита с помощта на таблицата на разширението (табл.2) се „разширяват" и се преобразуват в 48 бита, след което се сумират по модул 2 с 48-те бита на ключа. Както и в случая с таблицата за началното разместване, таблицата на разширението се чете също отляво надясно и отгоре надолу.

Табл.1. Начално разместване

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Табл.2. Таблица на разширението

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

Дадената таблица разширява битовете $R_{i-1} = x_1, x_2, \ldots, x_{32}$ в битове:

(1) $\quad (R_{i-1})_E = x_{32}, x_1, x_2, \ldots, x_{32}, x_1.$

Всъщност битовете, обозначени в първата и последната колони на таблицата на разширението, са разрядите, които се използват два пъти, за разширение от 32 бита до 48 бита.

По-нататък $(R_{i-1})_E$ се сумират по модул 2 с i-тия ключ, изборът на който се описва по-късно, а резултатът се разделя на осем 6-битови блока: $B_1, B_2, \ldots, B_8$, т.е.

(2) $\quad (R_{i-1})_E \oplus K_i = B_1, B_2, \ldots, B_8.$

Всеки от осемте 6-битови блока B_j се използва като вход за функцията на изображение на S-блока, връщащ 4-битов блок $S_j(B_j)$. По такъв начин, входните 48 бита, с помощта на функцията на S-блока, се преобразуват в 32 бита. Функциите на изображение на S-блока S_j са дефинирани в табл.3.

Преобразуването $B_j = b_1, b_2, b_3, b_4, b_5, b_6$ се изпълнява по следния начин: $b_1 b_6$ формира търсения (необходимия) ред, а $b_2 b_3 b_4 b_5$ – необходимия стълб. Например, ако $b = 110001$, то преобразуването S_1 връща стойността от ред 3 (десетичният еквивалент на двоичното число 11) и стълб 8 (десетичният еквивалент на двоичното число 1000), т.е. числото 5 (в двоичен запис 0101). 32-битовият блок, получен на изхода на S-блока, се „разбърква" с използване на таблицата на разместването (табл.4). Както и другите таблици, P-таблицата се чете отляво надясно и отгоре надолу така, че в резултат на разместването на битовете $x_1, x_2, \ldots, x_{32}$ се получава $x_{16}, x_7, \ldots, x_{25}$. 32-битовият изход на P-таблицата се сумира по модул 2 с 32 бита на лявата половина L_{i-1}, образувайки 32 бита на дясната половина R_i.

Алгоритъмът на стандартния блок може да бъде представен по следния начин:

(3) $\quad L_i = R_{i-1},$

(4) $\quad R_i = L_{i-1} \oplus f(R_{i-1}, K_i).$

Тук $f(R_{i-1}, K_i)$ обозначава функционалното съотношение, включващо описаните по-горе разширение, преобразуване в S-блока и разместване. След 16 итерации в такива стандартни блокове данните се разместват съгласно крайното (окончателно)

обратно разместване, описано в табл.5, където, както и по-рано, входните битове се четат отляво надясно и отгоре надолу.

Табл.3. Функции на изображение на S -блока

Ред	Стълб																S_i
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	S_1
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	S_2
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	S_3
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	S_4
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	S_5
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3	
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	S_6
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6	
3	4	3	2	12	9	5	15	0	11	14	1	7	6	0	8	13	
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1	S_7
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2	
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12	
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7	S_8
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2	
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8	
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11	

Табл.4. Таблица на разместването

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

Табл.5. Крайно (окончателно) разместване

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

За дешифриране се прилага същият алгоритъм, но ключовата последователност, използвана в стандартния блок, се избира в обратния ред. Трябва да се отбележи, че значението на $f(R_{i-1},K_i)$, което може да бъде изразено също и чрез изхода на i-тия блок като $f(L_i,K_i)$, прави процеса на дешифриране възможен.

Изборът на ключ става също в течение на 16 итерации. Входният ключ се състои от 64-битов блок с 8 бита по четност в разряди 8, 16, …, 64. Разместващият избор 1 „изхвърля" битовете по четност и размества останалите 56 бита съгласно табл.6. Изходът на дадената процедура се дели на два елемента – C и D, всеки от които се състои от 28 бита. Изборът на ключ преминава през 16 итерации, изпълнявани за създаване на различни множества от 48 ключови битове за всяка итерация на шифриране. Блоковете C и D последователно се изместват съгласно следните изрази:

(5) $\quad C_i = LS_i(C_{i-1}) \quad$ и $\quad D_i = LS_i(D_{i-1})$.

Тук LS_i е ляво циклично изместване на една позиция при итерации №1, 2, 9 и 16 и на две позиции при останалите итерации. След последователността C_i, D_i се размества съгласно разместващия избор 2, показан в табл.7. В резултат се получава ключовата последователност K_i, която се използва в i-тата итерация на алгоритъма за шифриране.

Табл.6. Кръгово разместване

57	49	41	33	25	17	9
1	58	50	42	14	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

Табл.7. Ключово разместване 2

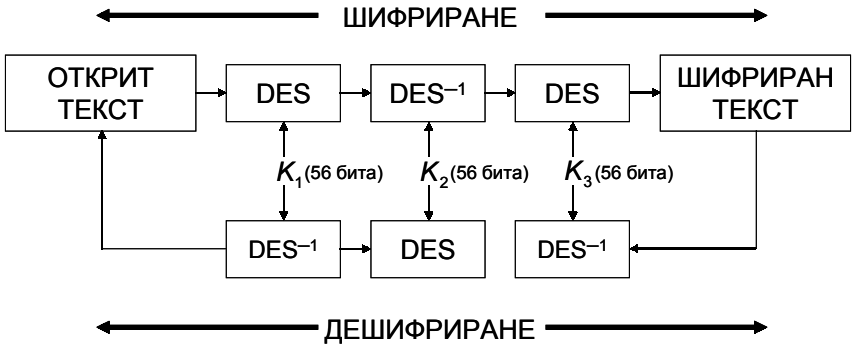
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Основен недостатък на този метод за реализиране на DES като блокова система за шифриране, наричан *метод на шифроващата книга*, се явява това, че при използване на един и същ ключ даденият блок на входния открит текст винаги ще дава един и същ изходен шифриран блок. Друг метод за шифриране, известен като *метод на шифриране с обратна връзка*, довежда до шифриране на отделни битове, а не на символи, което дава и поточното шифриране. В една такава система за шифриране с обратна връзка, шифрирането на сегмент на открития текст зависи *не* само от ключа и текущите данни, но и от някои предшестващи данни.

В края на 70-те години широко са се обсъждали два спорни момента, свързани с DES. Първият касае дължината на ключа – някои изследователи са считали, че 56 бита не са напълно достатъчни, за да се изключи взлом на криптосистемата. Вторият момент касае вътрешната структура на S-блоковете, които IBM никога не са произвеждали. Агенцията за национална безопасност (АНБ) на САЩ, която била привлечена за тестване на алгоритъма DES, поискала тази информация да не се обсъжда публично. Критиките се опасяват, че АНБ е участвала в проектирането на тези схеми и сега са способни да „проникнат" в кое да е съобщение, шифровано съгласно DES. В настоящия момент стандартът DES вече не се явява приемлив избор, осигуряващ надеждно шифриране, тъй като търсенето на 56-битов ключ с помощта на нескъпи компютърни методи се явява работа за няколко дена. Той, обаче, става основа на алгоритъма Triple-DES, описан по-долу, който намира широко приложение за шифриране на съобщения в криптосистемата PGP, версия 5.0.

Разработени са няколко варианта на DES, например DES-X, CRYPT(3), GDES и др. DES-X, предложен от проф. Рон Ривест от Масачузетския технологичен университет и фирма RSA Data Security, има по-висока устойчивост срещу атака на грубата сила, както и срещу диференциален и линеен криптоанализ. CRYPT(3) се използва за изчисляване на отражения (например, цифрови сигнатури) на паролите за достъп във версии на Unix, като тези отражения се съхраняват от сървърите вместо действителните пароли и с това се осигурява по-голяма сигурност. GDES (Generalized DES) оперира с променлива дължина на блоковете, които могат да съдържат q на брой 32-битови подблока, а и броят на операциите h също е променлив. Затова, в общия случай, този алгоритъм се обозначава като $GDES(q,h)$ – стойностите $q=2$ и $h=16$ съответстват на DES. GDES е неустойчив срещу диференциален криптоанализ – дори при $q=8$ и $h=63$, GDES се оказва по-слаб от DES [1].

ОПИСАНИЕ НА АЛГОРИТЪМА TRIPLE-DES



Фиг.4. Шифриране и дешифриране с помощта на Triple-DES

При използване на алгоритъма *Triple-DES*, откритият текст се пропуска през алгоритъма DES три пъти, като втората операция се провежда в режим на дешифриране (фиг.4).

Трите операции се реализират с помощта на различни 56-битови ключа, което е равносилно на използването на 168-битов ключ.

Алгоритъмът Triple-DES се разработва в отговор на нарасналите изчислителни мощности, за които 56-битовата дължина на ключа на DES е вече преодолима. Въпреки, че проблемът с малката дължина на ключа е решен успешно при алгоритъма Triple-DES, като негов основен недостатък може да се посочи значителното нарастване на сложността и времето за шифриране и дешифриране, което го прави удачен само за приложения, при които времето за реализация не е критично [1].

Освен Triple-DES, популярни през последните години стават и други блокови криптографски алгоритми, например *IDEA* (*International Data Encryption Algorithm*), *CAST* и *Blowfish*, които са включени в различните версии на криптосистемата PGP. Подробности за тези алгоритми могат да се намерят в специализираната литература [1, 2].

ЗАКЛЮЧЕНИЕ

В статията е описан подробно приетият като стандарт алгоритъм за шифриране на данни DES, който все още е най-масово използваният шифър в света и най-обстойно анализираният криптографски алгоритъм в литературата, тъй като обхваща в себе си основния подход за проектиране на сигурни блокови шифри, а и на негова основа е изграден алгоритъмът Triple-DES, намиращ широко приложение в криптосистемата PGP и в приложения, в които времето за шифриране и дешифриране не е решаващ фактор.

ЛИТЕРАТУРА

[1] Антонов, П., С. Малчев. Криптография в компютърните комуникации. Варна, 2000.

[2] Скляр, Б. Цифровая связь. Теоретические основы и практическое применение. Москва, Вильямс, 2003.