Лабораторна робота №4

Симетричні криптосистеми. Алгоритм DES

DES (Data Encryption Standard) - симетричний алгоритм шифрування, в якому один ключ використовується як для шифрування, так і для розшифрування даних. DES розроблений фірмою IBM і затверджений урядом США в 1977 році як офіційний стандарт (FIPS 46-3). DES має блоки по 64 біт і 16 циклову структуру мережі Фейстеля, для шифрування використовує ключ з довжиною 56 біт. Алгоритм використовує комбінацію з нелінійних (S-блоки) і лінійних (перестановки E, IP, IP-1) перетворень. Для DES рекомендовано декілька режимів:

- режим електронної кодової книги (ECB - Electronic Code Book),
- режим зчеплення блоків (CBC - Cipher Block Chaining),
- режим зворотного зв'язку по шіфротекста (CFB - Cipher Feed Back),
- режим зворотного зв'язку по виходу (OFB - Output Feed Back).

У 1972 році, після проведення дослідження потреб уряду США в комп'ютерній безпеці, американське Національне Бюро Стандартів (тепер Національний Інститут Стандартів і Технологій) визначило необхідність в загальноурядовому стандарті шифрування некритичної інформації. 15 травня 1973, після консультації з АНБ (Агентством національної безпеки), НБС оголосило конкурс на шифр, який задовольнить суворим критеріям проекту, але жоден конкурсант не забезпечував виконання всіх вимог. Другий конкурс був початий 27 серпня 1974. Представлений IBM шифр Lucifer, який був заснований на ранньому алгоритмі Хорста Фейстеля, визнали прийнятним.

17 березня 1975 запропонований алгоритм DES було видано у Федеральному Реєстрі. У наступному році було проведено 2 відкритих симпозіуму по обговоренню цього стандарту, де піддалися жорсткій критиці зміни, внесені АНБ до алгоритму: зменшення початкової довжини ключа і S-блоки (блоки підстановки), критерії проектування яких не розкривалися. АНБ підозрювалося в свідомому ослабленні алгоритму з метою, щоб АНБ могло легко переглядати зашифровані повідомлення. Сенатом США було проведено перевірку дій АНБ, результатом якої стала заява, в якій йшлося про те, що в процесі розробки DES АНБ переконало IBM, що зменшеної довжини ключа буде достатньо для всіх комерційних додатків, що використовують DES, допомагало в розробці S-перестановок, а також, що остаточний алгоритм DES був кращим, на їх думку, алгоритмом шифрування, був позбавлений статистичної або математичної слабкості. Також було виявлено, що АНБ ніколи не втручалося в розробку цього алгоритму.

Частина підозр у прихованій слабкості S-перестановок була знята в 1990, коли були опубліковані результати незалежних досліджень Елі Біхама (Eli Biham) і Аді Шаміра (Adi Shamir) по диференціальному криптоаналізу - основному методу злому блочних алгоритмів шифрування із симетричним ключем. S-блоки алгоритму DES виявилися більш стійкими до атак, ніж ті, що задавали випадково.

DES є блоковим шифром. Вхідними даними для блокового шифру є блок розміром n біт та k-бітний ключ. На виході, після застосування шифрувального перетворення, отримуємо n-бітний зашифрований блок. Блокові шифри реалізуються шляхом багаторазового застосування до вихідних блоків деяких базових перетворень. Навіть незначні відмінності вхідних даних як правило призводять до істотної зміни результату.

Блокове шифрування на основі мережі Фейстеля

Відповідного до простого опису цього методу розглядається можливість шифрування деякої інформації у двійковому вигляді (знаходиться у пам’яті комп’ютера, на іншому

пристрої, у файлі).

Алгоритм шифрування полягає в тому, що інформація розбивається на блоки однакової (фіксованої) довжини. Отримані блоки називаються вхідними. У випадку, якщо довжина вхідного блоку менша за розмір, що обраний алгоритм шифрування здатен шифрувати одночасно (розмір блоку), то блок збільшується будь-яким засобом. Як правило довжина блоку є ступеню двійки, наприклад, 64 біта або 128 біт.

Розглянемо операції, що виконуються тільки з одним блоком. У процесі шифрування з іншими блоками виконуються такі самі операції.

Обраний блок поділяється на дві частини (підблоки) однакового розміру — на ліву (L_0) та праву (R_0). «Лівий підблок» (L_0) змінюється функцією F з використанням раундового ключа K_0 : $x = F(L_0, K_0)$.

Результат додається по модулю 2 («XOR») до «правого підблоку» R_0 : $x = x \otimes R_0$

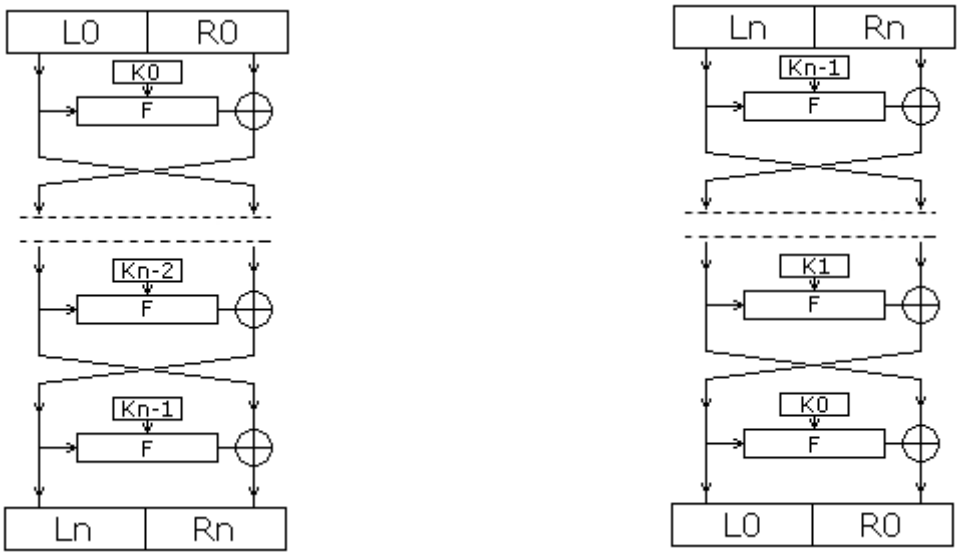
Результат використовується у наступному раунді в якості «лівого підблоку» L_1 : $L_1 = x$.

«Лівий підблок» L_0 поточного раунду буде використовуватись у наступному раунді в якості «правого підблоку» R_1 : $R_1 = L_0$

За певним математичним правилом обчислюється раундовий ключ K_1 — той, що буде використовуватись у наступному раунді.

Наведені операції виконуються N-1 раз, де N — кількість раундів в обраному алгоритмі шифрування. При переході від одного раунду (етапу) до наступного змінюються ключі: K_0 замінюється на K_1 , K_1 — на K_2 і т.д.

Дешифрування інформації відбувається подібно до шифрування, з тією різницею, що ключі ідуть у зворотньому порядку - від (N-1) -го ключа до першого.



Шифрування Дешифрування

Рис.1. Шифрування та дешифрування мережею Фейстеля

Для реалізації алгоритму необхідно використовувати певні функції. У своїх роботах Хорст Фейстель замість терміну «функція» використовував термін «блок» тому, що мова йшла про блоковий шифр і припускалось, що s- та p-блоки будуть цифровими мікросхемами (цифровими блоками). Таким чином було описано два типи блоків для виконання перетворень (функцій $F(L_i, K_i)$):

блок підставки (s-блок, англ. s-box);

блок перестановок (p-блок, англ. p-box).

Взагалі, будь-яке двійкове перетворення над блоком даних фіксованої довжини може

бути реалізовано у вигляді s-блоку. Тому, що існує певна складність побудови N-розрядного s-блоку при великих значеннях N, на практиці застосовують більш прості конструкції.

В алгоритмі DES використовуються пряме перетворення мережею Фейстеля в шифруванні і зворотне перетворення мережею Фейстеля у дешифруванні.

Схема шифрування алгоритму DES

DES є блочним шифром - дані шифруються блоками по 64 біти. На вхід алгоритму подається 64 бітний блок явного тексту, а на його виході отримується 64-бітний блок шифрограми. Під час шифрування і дешифрування використовується один і той самий алгоритм (за винятком дещо іншого шляху утворення робочих ключів).

Безпечність алгоритму базується на безпечності ключа. Ключ має довжину 56 біт. Як правило, у вихідному вигляді ключ має довжину 64 біти, де кожний 8-й біт є бітом паритету. Ці контрольні біти можуть бути винесені в останній байт ключа. Ключем може бути довільна 64-бітна комбінація, яку може бути змінено у будь-який момент часу. Деякі комбінації вважаються слабкими ключами, тому що можуть бути легко визначені.

На найнижчому рівні алгоритм представляється поєднанням двох базових технік шифрування: перемішування і підставки. Цикл алгоритму, з яких складається DES, є комбінацією цих технік, де в якості об'єктів перемішування виступають біти тексту, ключа і блоків підставок. На рис.2 показано загальну роботу алгоритму DES.

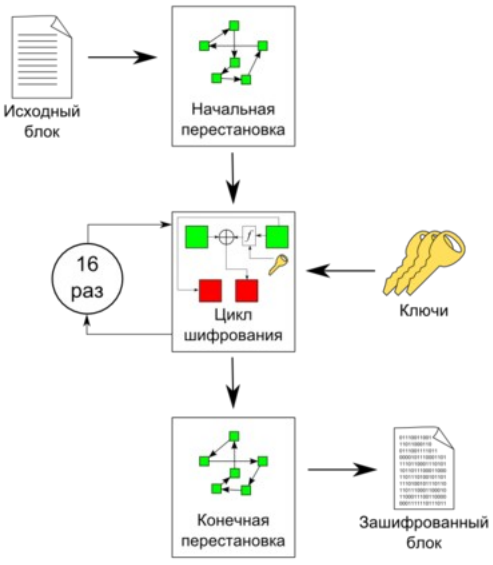


Рис.2. Алгоритм DES

Початкова перестановка

Оригінальний текст розбивається на блоки по 64 біти (8 байт). Кожний оригінальний 64-бітний блок даних подається на вхід алгоритму і над його бітами виконують початкову перестановку відповідно до наступної таблиці:

Таблиця 1. Початкова перестановка IP.
Перший і третій рядок є нумерацією біт в блоці, другий та четвертий — номерами біт в оригінальному блоці

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

Таким чином, після початкової перестановки IP, першими трьома бітами блоку стануть 58, 50, 42 біти вхідного блоку, а його 3 останніми бітами - 23, 15 та 7.

Цикли шифрування

Над отриманим після початкової перестановки 64-бітовим блоком виконують 16-циклів перетворення Фейстеля, які наведено вище. У 16-циклах перетворення Фейстеля ключову роль шифрування грає функція F — функція Фейстеля. Для обчислення функції F використовуються функція розширення E, перетворення S (складається з 8 перетворень S-блоків) і перестановка P (рис.3).

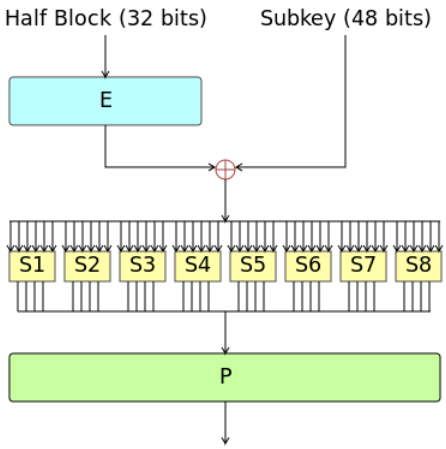


Рис.3. Схема обчислення функції F.

Аргументами функції F є 32-бітовий вектор та 48-бітовий ключ k_i, який є результатом перетворення 56-бітового початкового ключа шифру k. В якості функції використовується функція розширення E, яка розширює вхідний 32-бітовий вектор до 48-бітового вектора E шляхом повторення деяких біт відповідно до наступної таблиці:

Номера бітів	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Біти входу	_	1	2	3	4	_	_	5	6	7	8	_	_	9	10	11	12	_	_	13	14	15	16	_
Біти на виході	32	1	2	3	4	5	4	5	6	7	8	9	8	9	10	11	12	13	12	13	14	15	16	17

Номера бітів	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
Біти входу	_	17	18	19	20	_	_	21	22	23	24	_	_	25	26	27	28	_	_	29	30	31	32	_
Біти на виході	16	17	18	19	20	21	20	21	22	23	24	25	24	25	26	27	28	29	28	29	30	31	32	1

У верхньому рядку наведено біти вхідного вектору, у нижньому — розширеного. Повторення номерів у нижньому рядку позначає повторення відповідних біт. Видно, що розширений вектор складається з блоків по 6 біт, які йдуть по порядку, а кожний наступний блок починається з останніх двох біт попереднього блоку. Відмінним є лише перший 6-бітний блок, який починається бітом 32, та останній 6-бітний блок, який закінчується бітом з номером 1.

Отриманий після перестановки вектор E складається по модулю 2 (по-бітова логічна операція XOR) з відповідним ключем k_i. Результат цього перетворення буде мати таку саму довжину, як E та ключ k_i, а саме 48-біт. Для подальших перетворень необхідно повернутись до блоку довжиною у 32 біта, для чого E представляється у вигляді восьми послідовних блоків B_1, B_2, ... B_8:

$$E = B_1 B_2 \ldots B_8$$

Кожен блок B_i є 6-бітовим. Далі кожен з цих блоків буде трансформуватись у новий 4 бітовий блок B'_i за допомогою відповідних S_i перетворень. Індекс i вказує на відповідність блоку B_i та перетворення S_i.

Таблиця 3. Перетворення S_i

a ↓	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	←***b***
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	10	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	

Розглянемо приклад роботи перетворення. Припустимо, що утворено наступний блок

$B_3 = 101111$.

Значить для отримання B'_3 буде використовуватись перетворення S_3.

Перший і останній розряди B_3 є двійковим записом числа $\boldsymbol{a}$, тому $0 \leqslant a \leqslant 3$.

4 розряди із середини B_3 представляють число $\boldsymbol{b}$, тому $0 \leqslant b \leqslant 15$.

Рядки таблиці S_3 є числом $\boldsymbol{a}$, тому нумеруються від 0 до 3. Стовпці таблиці S_3 є числом $\boldsymbol{b}$, тому нумеруються від 0 до 15. Пара чисел ($\boldsymbol{a}$, $\boldsymbol{b}$) визначає число, що знаходиться на перетині рядка $\boldsymbol{a}$ і стовпця $\boldsymbol{b}$. Двійкове подання цього числа є B'_3. Для нашого випадку $a = 11_2 = 3$, $b = 0111_2 = 7$, таким чином отримуємо для S_3 пару (3, 7), а на перетині відповідного рядку та стовпця знаходиться число 7, що дає двійкове значення для $B'_3 = 0111$.

Після виконання подібних перетворень для усіх $\boldsymbol{B_i}$ буде утворено 32 бітовий блок $B'_1B'_2B'_3B'_4B'_5B'_6B'_7B'_8$. Останньою операцією, що буде виконано у функції F є перестановка P, яка задається наступним чином:

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

Де у верхньому рядку вказано нумерацію біт у вихідному блоці функції F, а у нижньому — біти вхідного блоку ($B'_1B'_2B'_3B'_4B'_5B'_6B'_7B'_8$).

Генерування ключів k_i

Ключі k_i отримують із початкового ключа k (64 біт = 8 байтів або 8 символів у ASCII) у такий спосіб. Вісім бітів, що знаходяться у позиціях 8, 16, 24, 32, 40, 48, 56, 64, додаються в ключ k так, щоб кожен байт мав непарне число одиниць. Це необхідно для виявлення помилок при обміні та зберіганні ключів. Потім роблять перестановку для розширеного ключа (крім доданих бітів 8, 16, 24, 32, 40, 48, 56, 64). Така перестановка визначена таблицею 5.

Таблиця 5. Початкова перестановка біт ключа

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	C_0
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	D_0

Ця перестановка визначається двома блоками C_0 і D_0 по 28 біт кожний. Наступні значення блоків C_i, D_i (i = 1,2,3 ...) отримують із попередніх значень блоків C_{i-1}, D_{i-1} за допомогою одного або двох лівих циклічних зміщень відповідно до наступної таблиці:

Таблиця 6. Циклічний зсув для ключа

і-й раунд	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Число зсувів	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Ключ k_i, $i = 1, \ldots 16$ складається із 48 біт, вибраних з бітів вектору C_iD_i (56 біт) згідно з таблицею 7. Перший і другий біти у k_i є 14, 17 вектору C_iD_i

Таблиця 7

і-й біт раундового ключа	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
номери біт з векторів C_iD_i	14	17	11	24	1	5	3	28	15	6	21	10	23	19	12	4	26	8	16	7	27	20	13	2

і-й біт раундового ключа	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
номери біт з векторів C_iD_i	41	52	31	37	47	55	30	40	51	45	33	48	44	49	39	56	34	53	46	42	50	36	29	32

Кінцева перестановка

Після 16-ти раундів застосовується кінцева перестановка біт. Вона позначається як IP^{-1} і є зворотньою до перестановки IP. Так 58-й біт на вході початкової перестановки IP переходить у позицію 1 (і т.д.), а при кінцевій перестановці 1-ий вхідний біт переводиться у позицію 58. Розташування біт кінцевої перестановки наведено у таблиці 8.

Таблиця 8. Зворотня перестановка IP^{-1}

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

33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
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

Нумерацію біт у кінцевому блоці наведено у першому та третьому рядках, а номера біт — у другому та четвертому.

Дешифрування

При дешифруванні даних всі дії виконуються у зворотньому порядку. У 16 циклах дешифрування, на відміну від шифрування із допомогою прямого перетворення мережею Фейстеля, використовується зворотнє перетворення мережею Фейстеля (див. рис.1).

$$R_{i-1} = L_i$$
$$L_{i-1} = R_i \otimes F(L_i, k_i)$$

Ключі k_i, i = 1, ..., 15 такі самі, що і при шифруванні, тільки вони застосовуються у зворотньому до шифрування порядку. Функція F, перестановки IP та IP^{-1} такі самі, як і в процесі шифрування.

Практичне завдання

1. Необхідно зашифрувати перші вісім літер свого прізвища, імені та по-батькові за латинською транслітерацією за допомогою алгоритму DES. В якості пароля взяти слово «password». Для зменшення кількості обчислень в алгоритмі DES слід обмежитись лише одним раундом.

2. Написати програмне забезпечення, що реалізує алгоритм DES. Перевірити його роботу для наступних прикладів:

приклад 1

Текст (HEX)	01 23 45 67 89 AB CD EF
Ключ (HEX)	FE FE FE FE FE FE FE FE
Шифр (HEX)	**6D CE 0D C9 00 65 56 A3**

приклад 2

Текст (HEX)	`00 00 00 00 00 00 00 00`
Ключ (HEX)	`00 00 00 00 00 00 00 00`
Шифр (HEX)	**`8C A6 4D E9 C1 B1 23 A7`**

приклад 3

Текст (HEX)	`01 23 45 67 89 AB CD EF`
Ключ (HEX)	`FE DC BA 98 76 54 32 10`
Шифр (HEX)	**`ED 39 D9 50 FA 74 BC C4`**

Приклад 4: перевірити результати першого завдання.