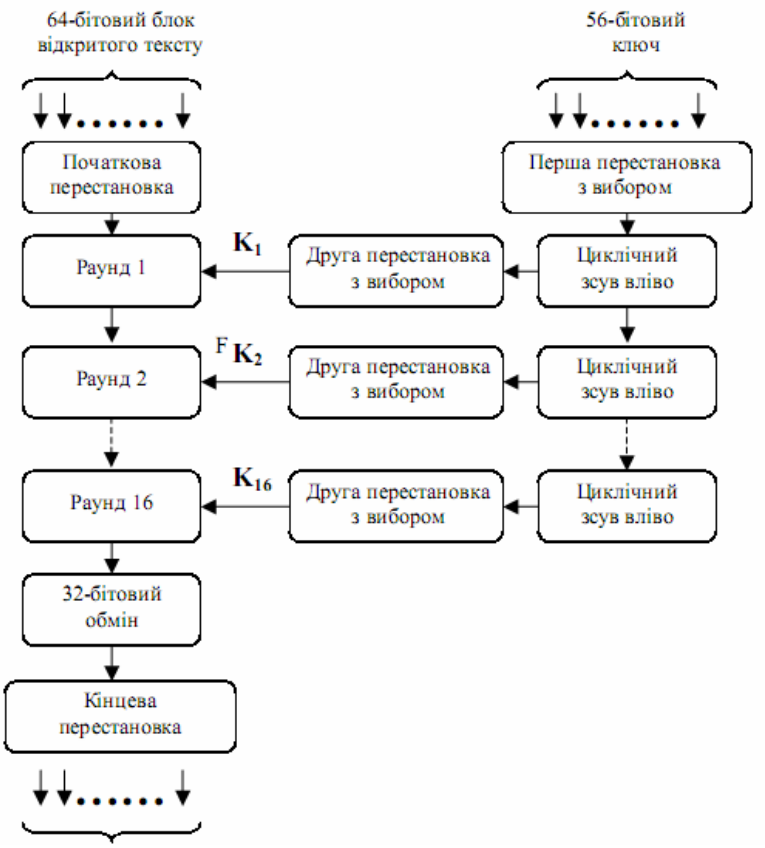
**<u>Лекція №6 (НЕ 2.1, 2 год.)</u>**. Стандарт DES.

***План лекції:***

1. Блок-схема роботи DES.
2. Операція розгортання ключа DES.
3. Криптостійкість DES.

***Зміст лекції:***

Розглянемо детальніше роботу алгоритму. Блок схема подана на наступному рисунку:

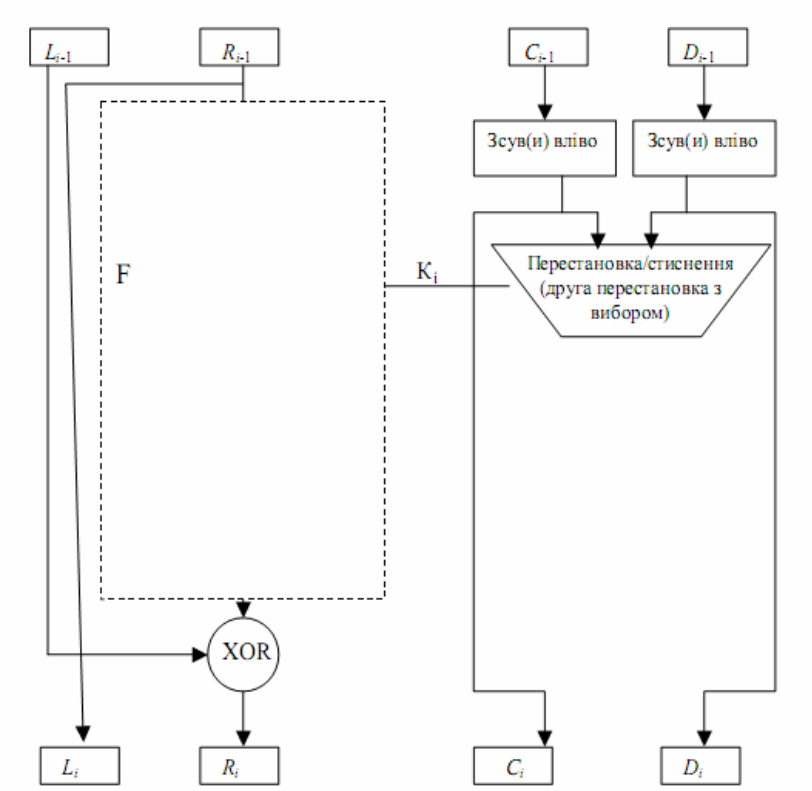



DES працює з 64-бітовим блоком відкритого тексту. Після початкової перестановки блок розбивається на праву й ліву половини довжиною 32 біти. Після цього виконується 16 етапів однакових дій, які називаються цикловою функцією , в яких дані об'єднуються з ключем. Після шістнадцятого етапу права і ліва половини об'єднуються й алгоритм завершується кінцевою перестановкою (оберненою по відношенню до початкової).

На кожному етапі біти ключа зсуваються, а потім із 56 бітів вибираються 48. Права половина даних збільшується до 48 бітів за допомогою перестановки з розширенням, об'єднується з допомогою XOR із 48 бітами зміщеного й переставленого ключа, проходить через 8 S-блоків, утворюючи 32 нових біти, і переставляється знову. Ці чотири операції і виконуються функцією F. Потім результат функції F об'єднується з лівою половиною з допомогою іншого XOR. У результаті цих дій з'являється нова права половина, а стара права половина стає новою лівою. Ці дії повторюються 16 разів, утворюючи 16 етапів DES.

Потім результат функції F об'єднується з лівою половиною з допомогою іншого XOR. У результаті цих дій з'являється нова права половина, а стара права половина стає новою лівою. Ці дії повторюються 16 разів, утворюючи 16 етапів DES.

Тепер розглянемо порядок обробки текстового блока алгоритмом DES.




На цьому рисунку подано один етап DES. Розглянемо його детальніше.

***Початкова перестановка***

Початкова перестановка виконується ще до першого етапу шифрування. Початкова перестановка і відповідна їй кінцева перестановка не впливають на безпеку DES. Оскільки програмна реалізація цієї багатобітової перестановки нелегка, в багатьох програмних реалізаціях DES початкова й кінцева перестановки не використовуються (за таких умов цей алгоритм перестає бути алгоритмом DES).

**Початкова перестановка**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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 |

Таблицю слід читати так: перший біт переставляється на 58-ме місце, другий ñ на 50-те, третій ñ на 42-ге, ..., 64-й ñ на 7-ме.

***Перестановка з розширенням***

Ця операція розширює праву половину даних від 32 до 48 бітів. Оскільки при цьому не просто повторюються певні біти, а й змінюється їхній порядок, ця операція називається перестановкою з розширенням. У неї дві задачі ñ привести розмір правої половини у відповідності з ключем для операції XOR і отримати довший результат, який можна буде стиснути в ході операції підстановки. Проте основний зміст зовсім інший. За рахунок впливу одного

біта на дві підстановки швидше зростає залежність бітів результату від бітів вихідних даних. Це називається лавинним ефектом. DES спроектовано так, щоб якомога скоріше можна було досягти залежності кожного біта шифротексту від кожного біта відкритого тексту й кожного біта ключа.
Перестановку з розширення ще іноді називають Е-блоком. Для кожного 4-бітового вхідного блока перший і четвертий біти являють собою два біти вихідного блока, а другий і третій біти - один біт вхідного блока.

**Перестановка з розширенням**

| | | | | | |
|---|---|---|---|---|---|
| 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 |

***Підстановка з допомогою S-блоків***
Після об'єднання стисненого блока з розширеним блоком із допомогою XOR над 48-бітовим результатом виконується операція підстановки. Підстановки проводяться у восьми блоках підстановки, чи S-блоках. У кожного S-блока 6- бітовий вхід і 4-бітовий вихід, всього використовується вісім різних S-блоків.
48 бітів діляться на вісім 6-бітових підблоки. Кожний окремий підблок обробляється окремим S-блоком: перший підблок ñ S-блоком 1, другий підблок ñ S-блоком 2 і т.д.
Кожен S-блок являє собою таблицю з 2 рядків і 16 стовпців. Кожний елемент в блоці є 4-бітовим числом. За 6 вхідними бітами S-блока визначається, під якими номерами стовпців і рядків шукати вихідні значення.
Вхідні біти в особливий спосіб визначають елемент S-блока. Наприклад, розглянемо 6-бітовий вхід S-блока: b1, b2, b3, b4, b5 і b6. Біти b1 і b6 об'єднуються, утворюючи 2-бітове число від 0 до 3, яке відповідає рядку таблиці. Середні 4 біти, з b2 по b5, об'єднуються, утворюючи 4-бітове число від 0 до 15, яке відповідає стовпцю таблиці.
Наприклад, на вхід шостого S-блока (тобто біти функції XOR з 31 по 36) потрапляє 110011. Перший і останній біт, об'єднуючись, утворюють 11, що відповідає рядку 3 шостого S-блока. Середні 4 біти утворюють 1001, а це стовпець 9 того ж S-блока. Елемент S-блоку 6, який знаходиться на перетині рядка 3 і стовпця 9, ñ 14. Замість 110011 підставляється 1110.
Підстановка з допомогою S-блоків є ключовим етапом DES. Інші дії алгоритму лінійні й легко піддаються криптоаналізу. S-блоки нелінійні, й саме на них більшою мірою, ґрунтується безпека DES.
У результаті цього етапу підстановки отримуються вісім 4-бітових блоки, які знову об'єднуються в єдиний 32-бітовий блок. Цей блок надходить на вхід наступного етапу - підстановки з допомогою Р-блоків.

**S-блоки алгоритму DES**

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $S_1$ | 14 | 4 | 13 | 1 | 2 | 15 | 11 | 8 | 3 | 10 | 6 | 12 | 5 | 9 | 0 | 7 |
| | 0 | 15 | 7 | 4 | 14 | 2 | 13 | 1 | 10 | 6 | 12 | 11 | 9 | 5 | 3 | 8 |
| | 4 | 1 | 14 | 8 | 13 | 6 | 2 | 11 | 15 | 12 | 9 | 7 | 3 | 10 | 5 | 0 |
| | 15 | 12 | 8 | 2 | 4 | 9 | 1 | 7 | 5 | 11 | 3 | 14 | 10 | 0 | 6 | 13 |
| $S_2$ | 15 | 1 | 8 | 14 | 6 | 11 | 3 | 4 | 9 | 7 | 2 | 13 | 12 | 0 | 5 | 10 |
| | 3 | 13 | 4 | 7 | 15 | 2 | 8 | 14 | 12 | 0 | 1 | 10 | 6 | 9 | 11 | 5 |
| | 0 | 14 | 7 | 11 | 10 | 4 | 13 | 1 | 5 | 8 | 12 | 6 | 9 | 3 | 2 | 15 |
| | 13 | 8 | 10 | 1 | 3 | 15 | 4 | 2 | 11 | 6 | 7 | 12 | 0 | 5 | 14 | 9 |
| $S_3$ | 10 | 0 | 9 | 14 | 6 | 3 | 15 | 5 | 1 | 13 | 12 | 7 | 11 | 4 | 2 | 8 |
| | 13 | 7 | 0 | 9 | 3 | 4 | 6 | 10 | 2 | 8 | 5 | 14 | 12 | 11 | 15 | 1 |
| | 13 | 6 | 4 | 9 | 8 | 15 | 3 | 0 | 11 | 1 | 2 | 12 | 5 | 10 | 14 | 7 |
| | 1 | 10 | 13 | 0 | 6 | 9 | 8 | 7 | 4 | 15 | 14 | 3 | 11 | 5 | 2 | 12 |
| $S_4$ | 7 | 13 | 14 | 3 | 0 | 6 | 9 | 10 | 1 | 2 | 8 | 5 | 11 | 12 | 4 | 15 |
| | 13 | 8 | 11 | 5 | 6 | 15 | 0 | 3 | 4 | 7 | 2 | 12 | 1 | 10 | 14 | 9 |
| | 10 | 6 | 9 | 0 | 12 | 11 | 7 | 13 | 15 | 1 | 3 | 14 | 5 | 2 | 8 | 4 |
| | 3 | 15 | 0 | 6 | 10 | 1 | 13 | 8 | 9 | 4 | 5 | 11 | 12 | 7 | 2 | 14 |
| $S_5$ | 2 | 12 | 4 | 1 | 7 | 10 | 11 | 6 | 8 | 5 | 3 | 15 | 13 | 0 | 14 | 9 |
| | 14 | 11 | 2 | 12 | 4 | 7 | 13 | 1 | 5 | 0 | 15 | 10 | 3 | 9 | 8 | 6 |
| | 4 | 2 | 1 | 11 | 10 | 13 | 7 | 8 | 15 | 9 | 12 | 5 | 6 | 3 | 0 | 1 |
| | 11 | 8 | 12 | 7 | 1 | 14 | 2 | 13 | 6 | 15 | 0 | 9 | 10 | 4 | 5 | 3 |
| $S_6$ | 12 | 1 | 10 | 15 | 9 | 2 | 6 | 8 | 0 | 13 | 3 | 4 | 14 | 7 | 5 | 11 |
| | 10 | 15 | 4 | 2 | 7 | 12 | 9 | 5 | 6 | 1 | 13 | 14 | 0 | 11 | 3 | 8 |
| | 9 | 14 | 15 | 5 | 2 | 8 | 12 | 3 | 7 | 0 | 4 | 10 | 1 | 13 | 11 | 6 |
| | 4 | 3 | 2 | 12 | 9 | 5 | 15 | 10 | 11 | 14 | 1 | 7 | 6 | 0 | 8 | 13 |
| $S_7$ | 4 | 11 | 2 | 14 | 15 | 0 | 8 | 13 | 3 | 12 | 9 | 7 | 5 | 10 | 6 | 1 |
| | 13 | 0 | 11 | 7 | 4 | 9 | 1 | 10 | 14 | 3 | 5 | 12 | 2 | 15 | 8 | 6 |
| | 1 | 4 | 11 | 13 | 12 | 3 | 7 | 14 | 10 | 15 | 6 | 8 | 0 | 5 | 9 | 2 |
| | 6 | 11 | 13 | 8 | 1 | 4 | 10 | 7 | 9 | 5 | 0 | 15 | 14 | 2 | 3 | 12 |
| $S_8$ | 13 | 5 | 8 | 4 | 6 | 15 | 11 | 1 | 10 | 9 | 3 | 14 | 5 | 0 | 12 | 7 |
| | 1 | 15 | 13 | 8 | 10 | 3 | 7 | 4 | 12 | 5 | 6 | 11 | 0 | 14 | 9 | 2 |
| | 7 | 11 | 4 | 1 | 9 | 12 | 14 | 2 | 0 | 6 | 10 | 13 | 15 | 3 | 5 | 8 |
| | 2 | 1 | 14 | 7 | 4 | 10 | 8 | 13 | 15 | 12 | 9 | 0 | 3 | 5 | 6 | 11 |

***Підстановка з допомогою Р-блоків***

32-бітовий вихід підстановки з допомогою S-блоків перемішується відповідно до Р-блока. Ця перестановка переміщує кожний вхідний біт в іншу позицію, жоден біт не використовується двічі і жоден не ігнорується. Цей процес називається прямою перестановкою.

**Перестановка з допомогою Р-блоків**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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 |

Нарешті результат перестановки з допомогою Р-блока об'єднується з допомогою XOR з лівою половиною початкового 64-бітового блока. Потім ліва і права половини міняються місцями, і починається наступний етап.

***Кінцева перестановка***

Кінцева перестановка обернена по відношенню до початкової. Слід звернути увагу, що ліва і права половини не міняються місцями після останнього етапу DES, замість цього об'єднаний блок R16 L16 використовується як вхід кінцевої перестановки. Це робиться для того, щоб алгоритм можна було використовувати як для шифрування, так і для дешифрування. Якби ж використовувалася перестановка половинок із наступним циклічним зсувом, результат був би таким самим.

Кінцева перестановка

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 40 | 8 | 48 | 16 | 56 | 24 | 64 | 32 |
| 39 | 7 | 47 | 15 | 55 | 23 | 63 | 31 |
| 39 | 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 |

**2. Операція розгортання ключа**

Спочатку 64-бітовий ключ DES зменшується до 56-бітового ключа відкиданням бітів. 56 бітів ключа переставляються відповідно до таблиці. Після перестановки 56-бітового ключа для кожного з 16 етапів DES генерується новий 48-бітовий підключ. Ці підключі визначаються так. По-перше, 56-бітовий ключ ділиться на дві 28-бітові половинки. Потім половинки циклічно зсуваються вліво на один чи два біти, залежно від етапу.

**Перестановка 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 |
| 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 |

**Кількість бітів зсуву ключа залежно від етапу**

| Етап | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Число | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 1 |

Після зсуву вибирається 48 з 56 бітів. Оскільки при цьому не тільки вибирається підмножина бітів, й змінюється їх порядок, ця операція називається перестановкою зі стисненням (її ще називають перестановкою з вибором). Її результатом є набір із 48 бітів. Наприклад, біт зсунутого ключа в позиції 33 переміщається в позицію 35 результату, а 18-й біт зсунутого ключа відкидається.

**Перестановка зі стисненням**

| | | | | | |
|---|---|---|---|---|---|
| 14 | 17 | 11 | 24 | 1 | 5 |
| 3 | 28 | 15 | 6 | 21 | 10 |
| 23 | 19 | 11 | 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 |

Через зсув для кожного підключа використовується різна підмножина бітів ключа. Кожен біт використовується приблизно в 14 з 16 підключів, хоча не всі

біти використовуються однакову кількість разів.

*Операція розшифрування DES*

Після всіх підстановок, перестановок, операцій XOR і циклічних зсувів можна подумати, що алгоритм дешифрування суттєво відрізняється від алгоритму шифрування і так само заплутаний. Навпаки, різні компоненти були підібрані так, щоб виконувалося дуже корисна властивість: для шифрування й розшифрування використовувався один і той самий алгоритм.

DES дозволяє використовувати для шифрування і розшифрування блока одну функцію. Єдиною відмінністю є те, що ключі повинні використовуватися в зворотному порядку. Тобто, якщо на етапах шифрування використовувалися ключі $K_1,K_2,K_3,\ldots,K_{16}$, то ключами дешифрування будуть $K_{16},K_{15}, K_{14}, \ldots, K_1$. Алгоритм, який створює ключ для кожного етапу, також циклічний. Ключ зсувається вправо, а кількість позицій зсуву становить 0, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1.

**3. Криптостійкість DES**

Обговорюючи DES, неможливо обминути тему безпеки цього алгоритму та можливих атак на нього. Багаторічний досвід експлуатації DES й його відкритість зумовили те, що DES став одним із найпопулярніших алгоритмів із погляду перевірки тих чи інших методів криптоаналізу. За весь час існування алгоритму на нього було здійснено багато атак; при цьому уважно вивчалися та враховувалися його слабкості, виявлені за довгий термін експлуатації. Треба врахувати, що деякі атаки можна реалізувати, лише виходячи з припущення, що зловмисник володіє певними обчислювальними (або часовими) ресурсами. У більшості випадків такі спроби мають лише теоретичний характер, однак не виключено, що з розвитком комп'ютерної техніки та криптології як науки, ці атаки можна буде реалізувати на практиці.

***До основних недоліків DES відносять такі:***

- наявність «слабких» ключів, викликана тим, що для генерування ключової послідовності виконується два незалежних регістри зсуву. Прикладом слабкого ключа може служити 1F1F1F1F0E0E0E0E. При цьому результатом генерування будуть ключові послідовності, однакові з вихідним ключем, в усіх 16 раундах. Існують також різновиди слабких ключів, що дають усього чотири ключові послідовності. Існують також «зв'язані» ключі, які отримуються один з одного інверсією одного біта;
- невелика довжина ключа в 56 бітів. При сучасному рівні розвитку комп'ютерних засобів ця довжина ключа не може забезпечувати потрібного захисту для деяких типів інформації;
- надмірність ключа, що має біти контролю парності. Біхам і Шамір запропонували досить ефективну атаку на реалізацію DES у смарт-картах або банківських криптографічних модулях, що використовують EEPROM- пам'ять для зберігання ключів. Наявність бітів контролю парності дозволяє відновити ключ при втраті частини ключа, викликаній втратою інформації в комірках пам'яті;
- використання статичних підстановок у S-блоках, що, незважаючи на велику кількість раундів, дозволяє криптоаналітикам атакувати цей алгоритм.

Зауважимо, що авторам не відомі успішні атаки на 16-раундовий DES, які використовують останній факт, однак успішні атаки на 8-раундовий DES трапляються. Так, М.Хеллман запропонував атаку на основі робочої станції SUN-4, яка визначає 10 бітів ключа за 10 с. У випадку вибору 512 відкритих текстів імовірність успіху становить 80%, а при виборі 768 відкритих тактів ñ 95%. Відновивши 10 бітів ключа, решту можна знайти методом повного перебору наступних 46 бітів. Отже, враховуючи вищезазначене, можна стверджувати, що сьогодні використання DES для критичної, з погляду секретності, інформації є досить небезпечною справою.