Az informatikai biztonság alapjai

Pintér-Huszti Andrea

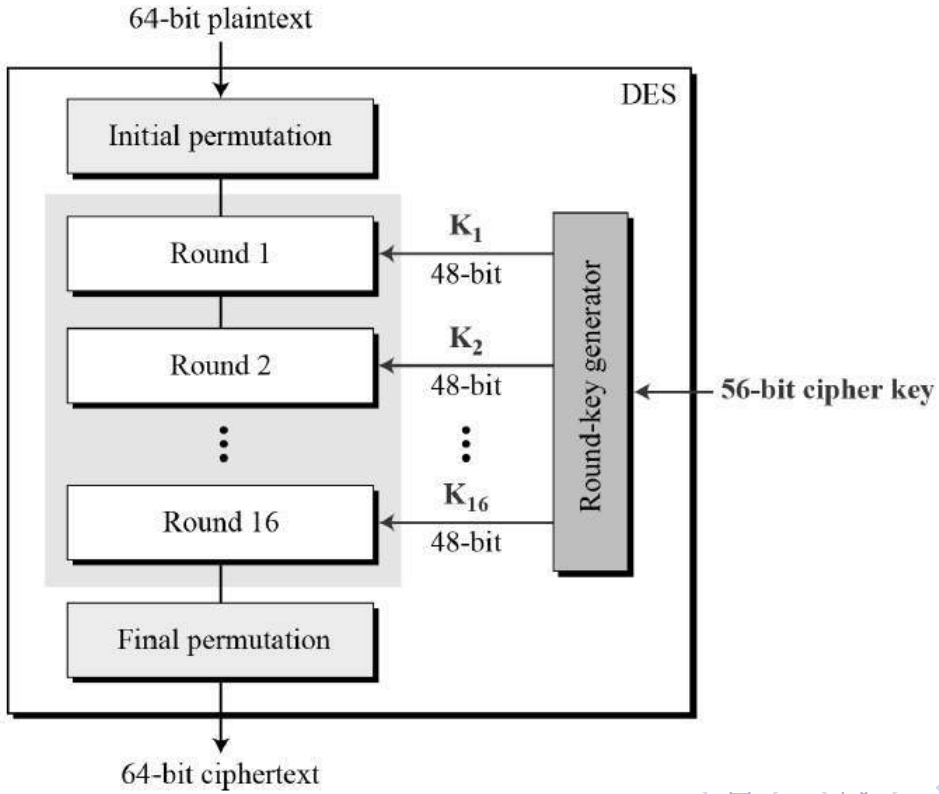
2025. március 23.

Tartalom

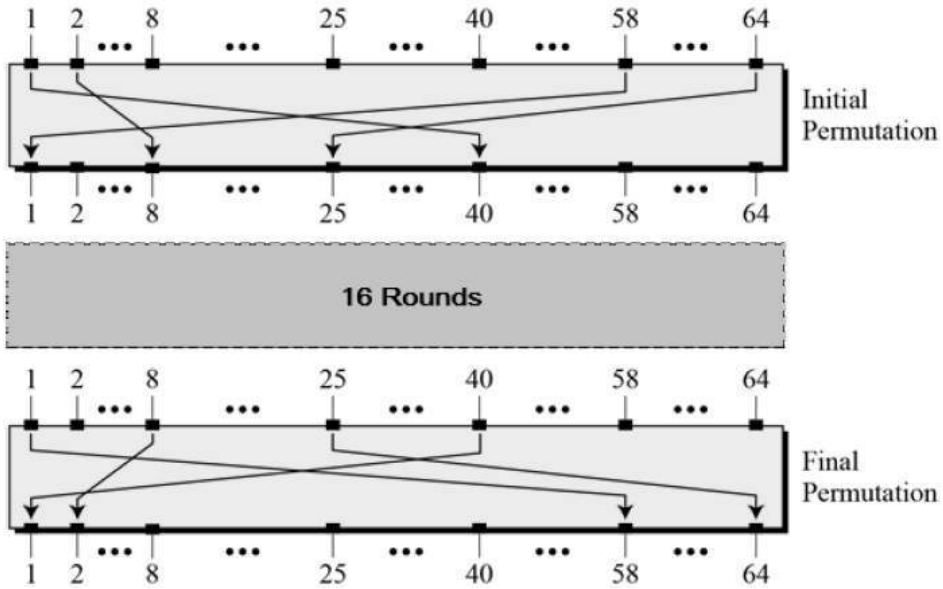
Data Encryption Standard (DES)

Data Encryption Standard (DES)

- $SE = (Key, Enc, Dec)$ szimmetrikus titkosítási séma
- $\mathcal{P} = \mathcal{C} = \{0,1\}^{64}$, $\mathcal{K} = \{0,1\}^{56+8}$, minden $K \in \mathcal{K}$ 56 véletlen bitből és 8 paritás bitből áll
- Key: $K \in \mathcal{K}$ véletlen választása
- Enc:



DES: titkosító algoritmus



Az első lépés egy kulcsfüggetlen permutáció (IP) a 64 bites bemeneten. Az utolsó lépés ennek pontosan az inverz művelete (IP^{-1}, Final permutation).

DES: titkosító algoritmus - Feistel struktúra

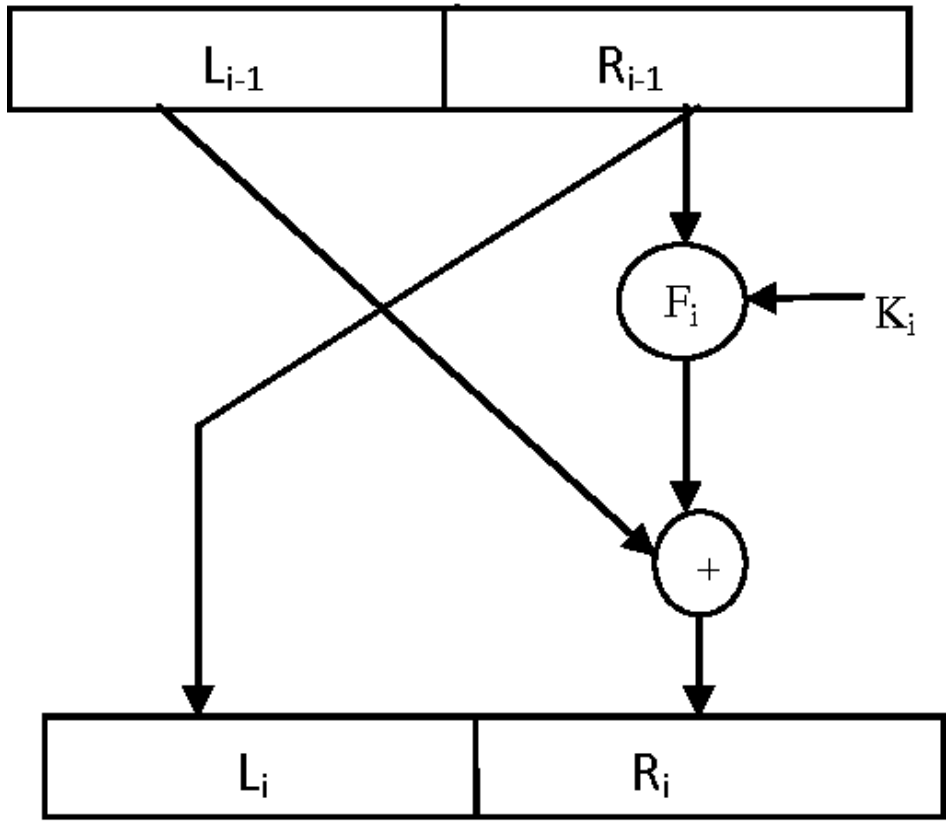


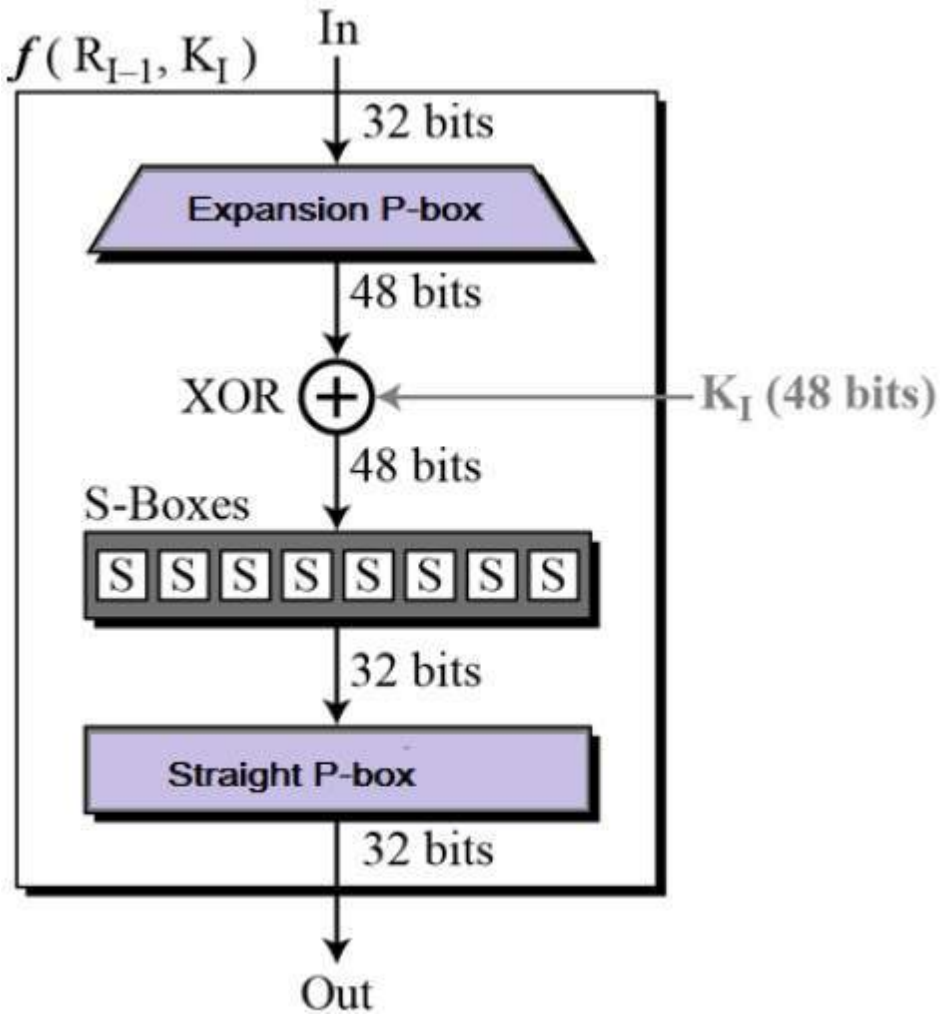
Fig 2: Single Round in Feistel cipher Structure

A két permutáció (IP,IP^{-1}) között 16 kör (Round) hajtódik végre, minden körben a Feistel struktúra fut le.

DES: titkosító algoritmus - Feistel struktúra

- Minden kör két 32 bites bemenetből ugyancsak kettő 32 bites kimenetet produkál.
- A bal oldali kimenet egyszerűen a jobb oldali bemenet másolata. $(L_i = R_{i-1})$
- A jobb oldali kimenetet kizáró vagy (xor) művelettel kapjuk, amit egyrészt a bal oldali bemenet, másrészt a jobb oldali bemenet, valamint az adott körhöz tartozó kulcsérték alapján egy adott f függvénnyel képzett érték között végzünk el. $(R_i = L_{i-1} \oplus f(R_{i-1}, K_i))$

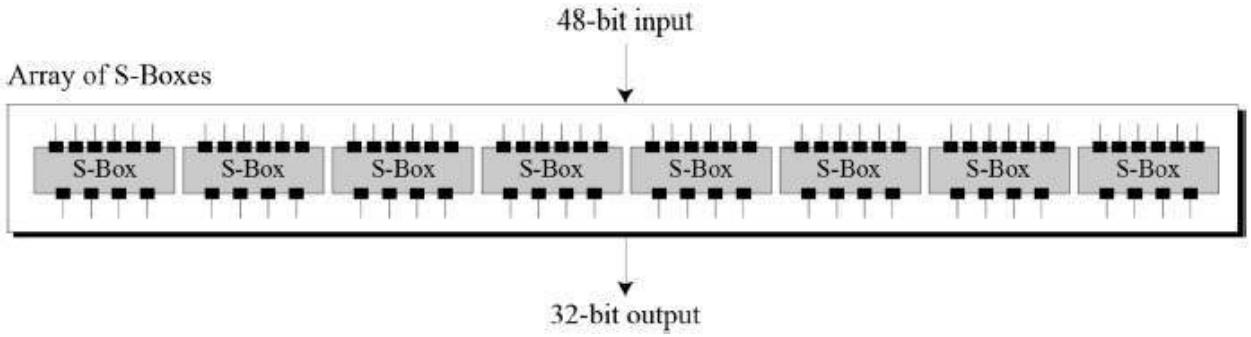
DES: titkosító algoritmus - belső függvény



DES: titkosító algoritmus - belső függvény

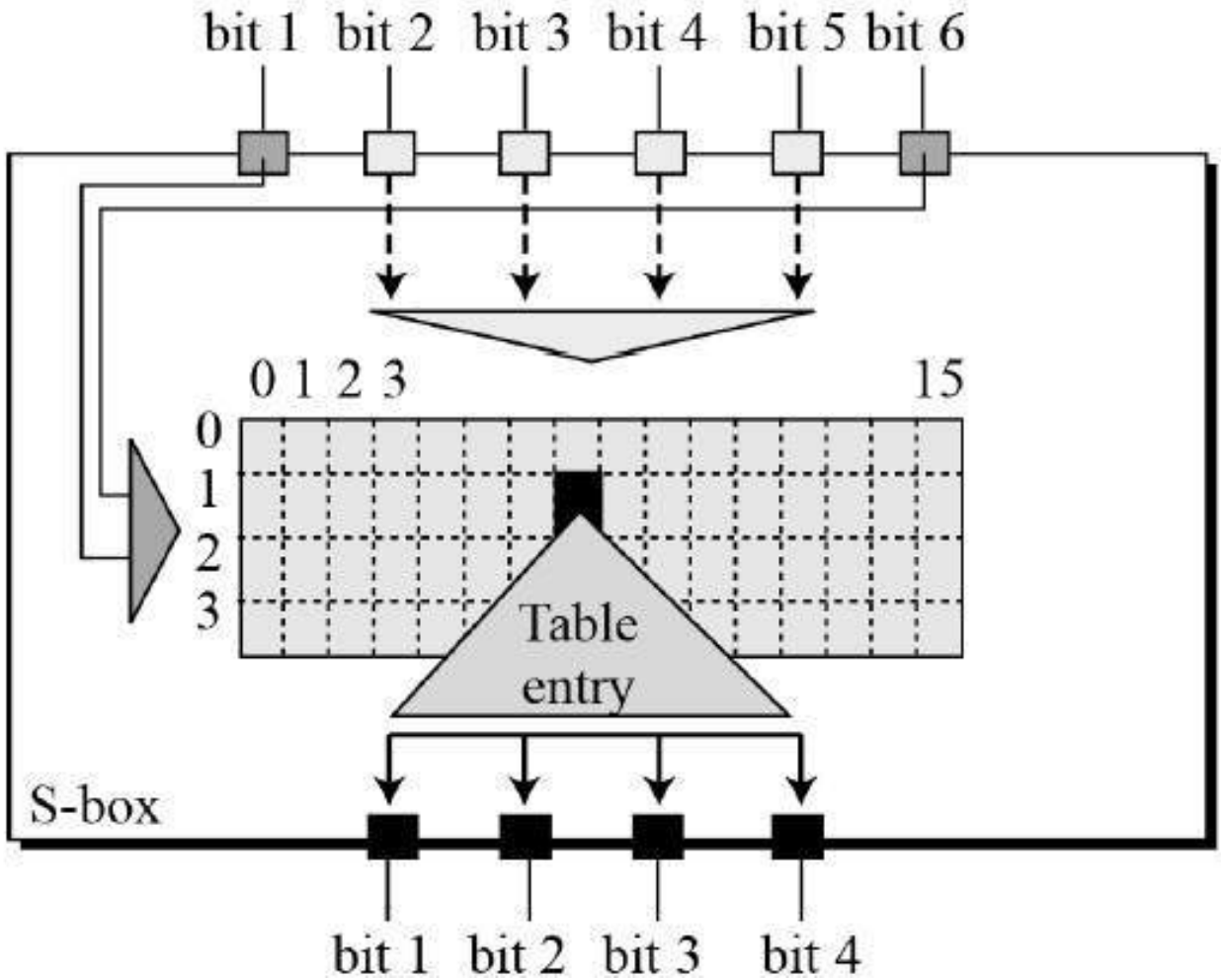
- Első lépésben egy 48 bites számot képzünk a 32 bites jobb oldal kiterjesztésével (Expansion P-box: E, lásd DES táblák segédlet).
- A második lépésben az E kimenete és a K_i bitjei között kizáró vagy műveletet hajtunk végre.
- Az így kapott eredményt 8 db 6 bites csoportra osztjuk, amiket aztán különböző S-dobozokba pumpálunk.
- Az egyes S-dobozok 4 bites kimenetet generálnak.
- Végül az így nyert 32 bitet egy P-dobozon (Straight P-box, lásd DES táblák segédlet) engedjük keresztül.

DES: titkosító algoritmus - S-boxok alkalmazása

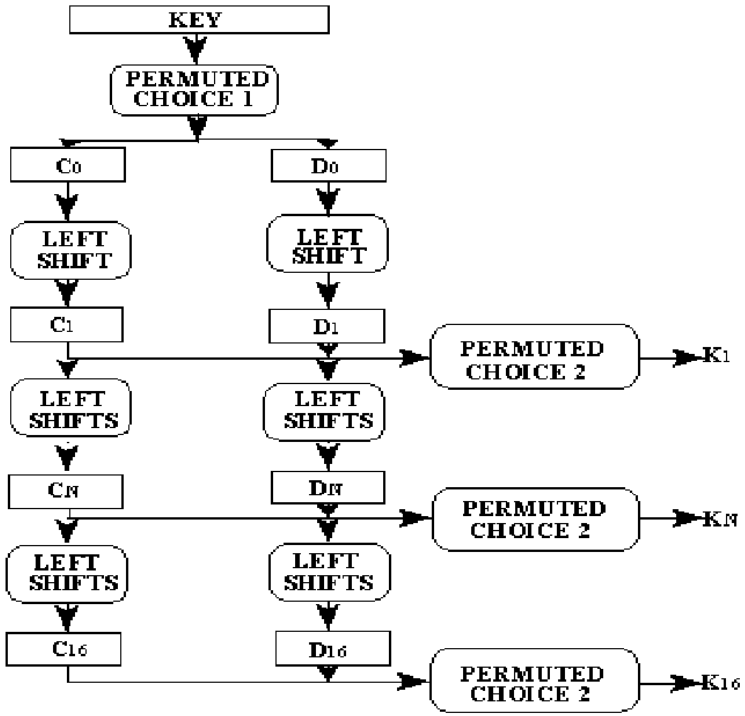


Minden S-doboz 4 sorból és 16 oszlopból álló táblázat (lásd DES táblák segédlet). A bemenet két szélső bitje (1. és 6.) címzi meg a sort, és a középső 4 bitje (2-5. bit) címzi meg az oszlopot. Az így meghatározott cella tartalma az S-doboz kimenete, ami 4 biten ábrázolható.

DES: titkosító algoritmus - S-boxok alkalmazása



DES: Körkulcsok generálása



Körkulcsok generálása

- Mind a 16 iterációs lépésben különböző körkulcsokat használunk. Az algoritmus kezdetekor egy 56 bites permutációt ($PC-1$, lásd DES táblák segédlet) végzünk a kulcson.
- A kulcsot két 28 bites részre particionáljuk, mindkettőt az iteráció sorszámának megfelelő számú bittel (lásd DES táblák segédlet) balra forgatjuk.
- Az 56 bites kulcs egy 48 bites részét minden fokozatban még külön permutáljuk ($PC-2$, lásd DES táblák segédlet).

Visszafejtés

- A visszafejtést és a titkosítást ugyanazzal a kulccsal és körkulcsokkal végezzük.
- A visszafejtő algoritmus megegyezik a titkosító algoritmussal, csak a körkulcsok alkalmazásának sorrendje más.
- Visszafejtésnél $K_{16}, K_{15}, \ldots, K_1$ sorrendet alkalmazunk.

3DES

- Az IBM már 1979-ben felismerte, hogy a DES-kulcs túlságosan rövid (sikeres brute force támadás 1999-ben), és a biztonság növelésének érdekében kidolgoztak egy háromszoros titkosítást használó eljárást. 3DES használata 2023-tól nem javasolt.
- A 3DES algoritmus a DES titkosító vagy visszafejtő algoritmusát hajtja végre háromszor. Az EDE (Encryption-Decryption-Encryption) algoritmus az elterjedt.
- EDE: Három kulcsot és három fokozatot használnak. Az első lépcsőben a nyílt üzenetet a szokott módon a K_1 kulccsal titkosítjuk. Második lépésben visszafejtést végzünk, melyhez a K_2-t használjuk kulcsként. Végül az így kapott eredményt K_3 kulccsal titkosítjuk.

Teljes név: Advanced Encryption Standard (AES), a Rijndael algoritmus egy speciális paraméterezése.

Blokkméret: 128 bit (mindhárom változatnál).

Kulcshosszok: 128, 192, 256 bit.

Körök száma (rounds): 10 (128-bit kulcs), 12 (192), 14 (256).

Alkalmazás: blokk-titkosító, protokollokhoz különböző üzemmódokkal (CBC/CTR/GCM...).

Advanced Encryption Standard (AES)

Ábrázolás — a „state”

A 128 bitet 4 × 4 bájtos mátrixként kezeljük (state).

A műveletek mindig ezen a mátrixon futnak; minden bájt külön cella.

Advanced Encryption Standard (AES)

AES egy körének lépései

Key Schedule (kulcskiterjesztés) – mi ez?

Az AES-ben a key schedule az a folyamat, ami a bemeneti kulcsból (pl. 128/192/256 bit) minden körhöz szükséges körkulcsot (round key) generálja.

Rijndael: Joan Daemen, Vincent Rijmen

- $SE = (Key, Enc, Dec)$ szimmetrikus titkosítási séma
- $\mathcal{P} = \{0,1\}^{128}$
- $\mathcal{C} = \{0,1\}^{128}$
- $\mathcal{K} = \{0,1\}^{k},\ k \in \{128, 192, 256\}$
- Key: véletlenül választunk egy $K \in \mathcal{K}$

1) SubBytes – bájtcsere

Minden állapotmátrixbeli bájtot kicserélünk egy előre meghatározott S-box táblázatból.

Ez nemlineáris lépés, ami megnehezíti a kriptoanalízist.

Egyszerűen: minden byte "átváltozik" egy másik byte-ra a táblázat szerint.

AES - titkosító algoritmus

ShiftRows – sorok eltolása
Az állapot mátrix sorait balra toljuk eltérő mértékben

AddRoundKey – kulcs hozzáadása

Minden bájthoz XOR-oljuk az adott körhöz tartozó 128-bites körkulcsot

AES animáció!

Ez a lépés hozza be a tényleges titkos kulcsot a folyamatban

- Egy kör: SubByte, ShiftRow, MixColumn, AddRoundkey
- Körök száma: $\begin{cases} k = 128, & 10; \\ k = 196, & 12; \\ k = 256, & 14. \end{cases}$

MixColumns – oszlopok összekeverése

Minden oszlopot matematikai mátrixszal keverünk (GF(2^8) műveletekkel).

Ez lineárisan keveri a bájtokat az oszlopban, így egyetlen bájt változása az egész oszlopra hat.

Utolsó körben kihagyjuk