Nivelul prezentare

Scopul securitatii

- confidentialitatea
  - informaţia este disponibilă doar utilizatorilor autorizaţi
- integritatea
  - informaţia poate fi modificată doar de utilizatorii autorizaţi sau în modalitatea autorizată (mesajul primit nu a fost modificat în tranzit sau măsluit)
- disponibilitatea
  - accesul la informaţie al utilizatorilor autorizaţi nu este îngrădit (opusul este denial of service)

Probleme derivate

- autentificarea
  - determinarea identităţii persoanei cu care schimbi mesaje înainte de a dezvălui informaţii importante
- controlul accesului
  - protectia impotriva accesului ne-autorizat
- non-repudierea
  - transmitatorul nu poate nega transmiterea unui mesaj pe care un receptor l-a primit

Metode de rezolvare

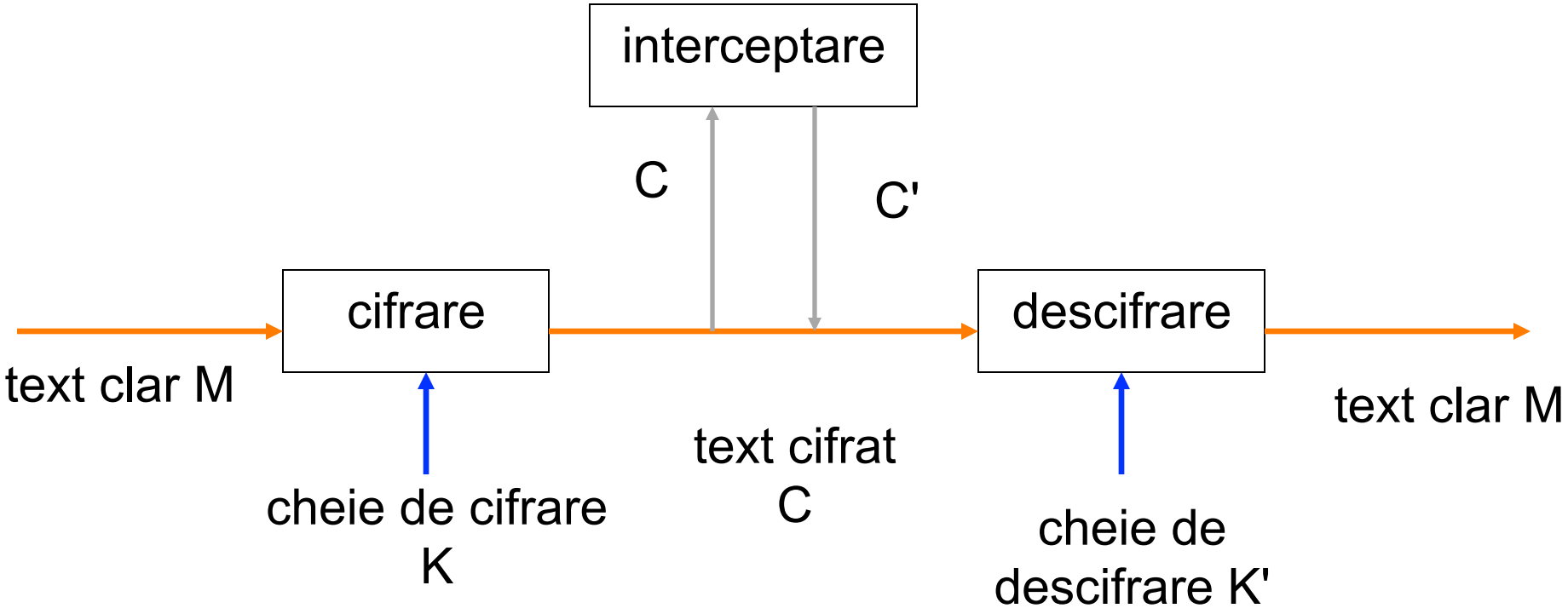
- Organizare
  - Algoritmi de criptare si hash
  - Mecanisme de securitate
    - **criptare, rezumare (hash), semnatura digitala**
  - Servicii si protocoale de securitate

- Securitatea in ierarhia de protocoale
  - considerata initial in nivelul prezentare al ISO OSI
  - este distribuita, in realitate, diverselor nivele
    - fizic – tuburi de securizare a liniilor de transmisie
    - legatura de date – legaturi criptate
    - retea – ziduri de protectie (firewalls), IPsec
    - transport – end-to-end security
    - aplicatie – autentificarea, non-repudierea

Alte aspecte

- Politici de securitate.
- Control software (antivirus).
- Control hardware:
  - Cartele inteligente;
  - Biometrie.
- Control fizic (protecţie).
- Educaţie.
- Măsuri legale.

Modelul de bază al criptării



confidentialitatea - intrusul să nu poată reconstitui M din C (să nu poată descoperi cheia de descifrare K‘).

integritatea - intrusul să nu poată introduce un text cifrat C', fără ca acest lucru să fie detectat (sa nu poată descoperi cheia de cifrare K).

Definiţii

- Spargerea cifrurilor = criptanaliză.
- Proiectarea cifrurilor = criptografie.
- Ambele sunt subdomenii ale criptologiei.
- Transformarea F realizată la cifrarea unui mesaj:

F : {M} x {K} -> {C}, unde:

  - {M} este mulţimea mesajelor;
  - {K} este mulţimea cheilor;
  - {C} este mulţimea criptogramelor.

- Operaţii:
  - Cifrarea: $C = E_k(M)$.
  - Descifrarea: $M = D_{k'}(C)$.
- Conotaţie de ordin practic!

Problema criptanalistului

- Criptanaliză cu **text cifrat cunoscut**; se cunosc:
  - Un text cifrat;
  - Metoda de criptare;
  - Limbajul textului clar;
  - Subiectul;
  - Anumite cuvinte din text.
- Criptanaliză cu **text clar cunoscut**; se cunosc:
  - Un text clar;
  - Textul cifrat corespunzător;
  - Anumite cuvinte cheie (login).
- Criptanaliză cu **text clar ales**; se cunosc:
  - Mod cifrare anumite porţiuni de text;
  - Exemplu pentru o bază de date - modificare / efect.

Caracteristicile sistemelor secrete

- sistem **neconditionat sigur**
  - rezistă la orice atac, indiferent de cantitatea de text cifrat interceptat
  - ex. one time pad
- **computational sigur** sau **tare**
  - nu poate fi spart printr-o analiză sistematică cu resursele disponibile.
- sistem **ideal**
  - indiferent de volumul textului cifrat care este interceptat, o criptogramă nu are o rezolvare unică, ci mai multe, cu probabilităti apropriate

Cerinţe criptosisteme cu chei secrete

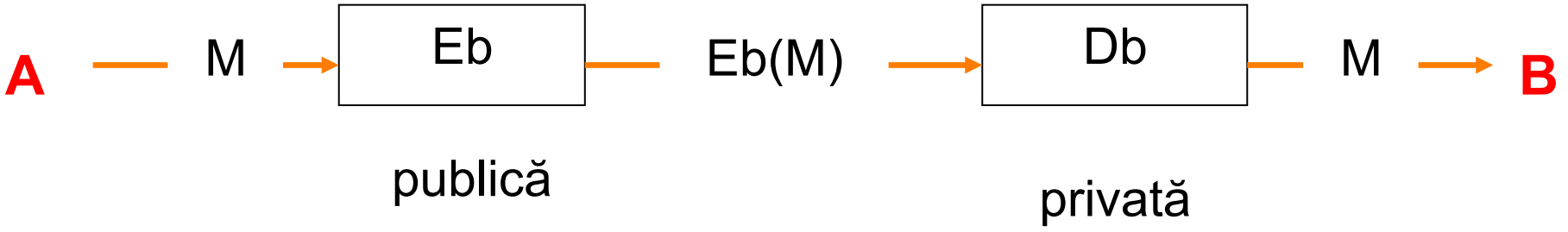
- Cerinţe generale:
  - Cifrare şi descifrare eficiente pentru toate cheile.
  - Sistem uşor de folosit (chei de transformare).
  - Securitatea să depindă de chei, nu de algoritm.
- Cerinţe specifice pentru confidenţialitate: să fie imposibil computaţional ca un criptanalist să determine sistematic:
  - Transformarea D_k din C, chiar dacă ar cunoaşte M.
  - M din C (fără a cunoaste D_k).
- Cerinţe specifice pentru integritate: să fie imposibil computaţional ca un criptanalist să determine sistematic:
  - Transformarea E_k, din C, chiar dacă ar cunoaşte M.
  - Cifrul C' astfel ca $D_k(C')$ să fie un mesaj valid (fără a cunoaşte E_k).

Modelul criptografic cu chei publice

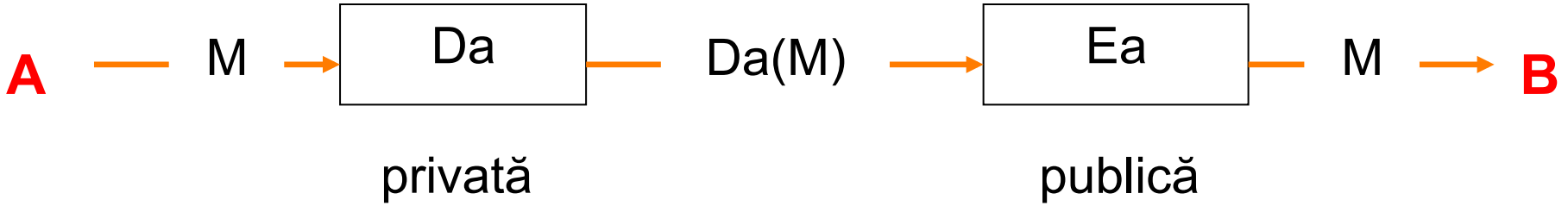
- Sistemele criptografice:
  - Simetrice.
  - Asimetrice:
    - Propuse de Diffie şi Hellman în 1976.
    - Chei diferite de cifrare E şi descifrare D.
    - Nu se pot deduce (uşor) una din alta, mai precis:
      - D(E(M)) = M;
      - Este extrem de greu să se deducă D din E;
      - E nu poate fi "spart" prin criptanaliză cu text clar ales.
- Într-un sistem asimetric, fiecare utilizator U:
  - Face publică cheia (transformarea) E_u de cifrare.
  - Păstrează secretă cheia (transformarea) D_u de descifrare.
- Schema de integritate / autentificare:
  - Condiţia necesară este ca transformările E_a şi D_a să comute, adică

$$E_a(D_a(M)) = D_a(E_a(M)) = M.$$

Schema de confidentialitate



Schema de integritate

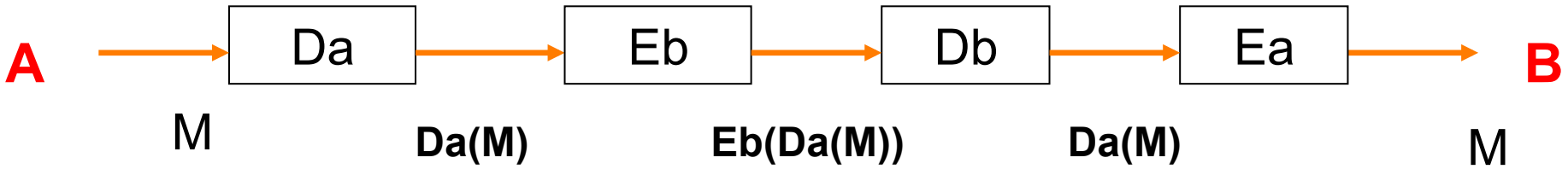


Se asigură:

confidentialitate doar B, care are cheia privata Db poate intelege mesajul M

integritate A are garantia că nimeni no poate modifica mesajul M deoarece nu cunoaste cheia privata Da

Schema de autentificare

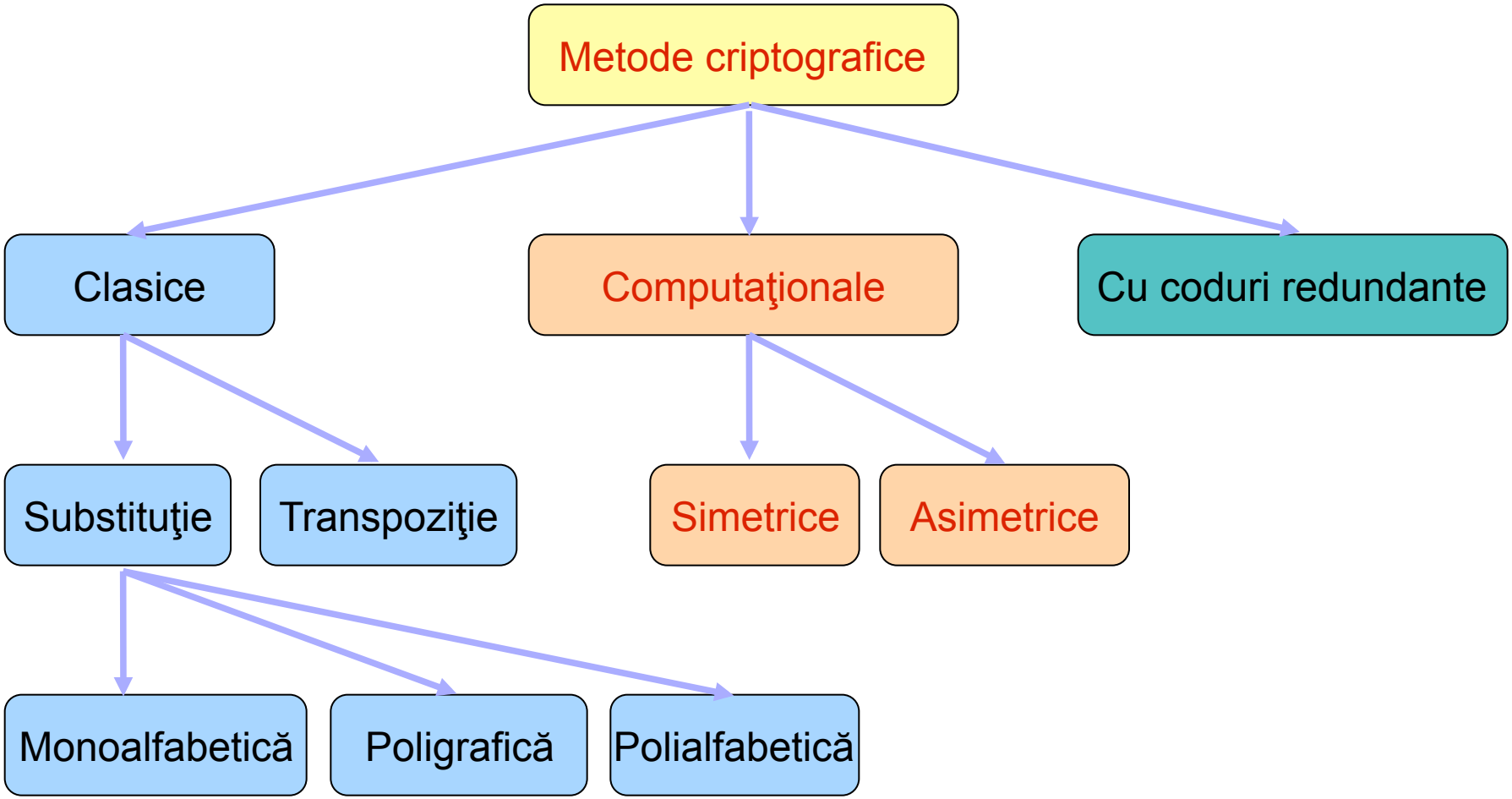


autentificare — B are garantia că A este sursa mesajului

(semnătură digitală - se poate semna doar rezumat)

ne-repudiere — folosind perechea Da(M) si M

Clasificare generală



Utilizarea TI în evaluarea algoritmilor criptografici

Fie S o sursa de informatii

- care transmite mesajele $X_1,..., X_n$
- cu probabilitatile $p(X_1), ..., p(X_n)$, ptr. care $\Sigma_{i=1,n}\ p(X_i) = 1$.

Entropia este cantitatea medie de informatie transmisa de sursa

$$H(X) = \Sigma_{i=1,n}\ p(X_i)*\log\ (1/p(X_i))$$

- $\log(1/p(X_i))$ = cantitatea de informatie primita la receptia lui X_i.
- baza logaritm = 2 → cantitatea masurata in numar de biti

Exemplu

- aruncarea monedei – cap sau pajura
- probabilitati egale
- informatie 1 bit: $H(X) = -\Sigma_{i=1,2}\ (1/2)*\log\ (1/2) = 1$

Utilizarea TI (2)

Entropia = cantitatea de incertitudine inlaturata (in medie) de un mesaj

Exemplu:

o sursa poate trimite n = 4 mesaje, cu probabilitati egale p(X) = 1/4

$$H(X) = \Sigma_{i=1,4} (1/4)*\log (4) = 2$$

fiecare mesaj inlatura o incertitudine de 2 biti

(inainte nu se stia care din cele 4 mesaje va fi primit)

Entropia depinde de distributia probabilitatior mesajelor

- H(X) maxim când $p(X_1) = p(X_2) = ... = p(X_n) = 1/n$.
- H(X) descreşte când distribuţia mesajelor se restrânge.
- H(X) = 0 când $p(X_i) = 1$ pentru un mesaj i.

Entropie conditionata - Echivocitatea

Exemplu:

- mesajele X sunt conditionate de mesaje Y

  Fie m=4 şi p(Y) = 1/4 pentru fiecare Y.

  Fiecare Y restrânge X:

  dupa Y1: urmeaza X1 sau X2,

  dupa Y2: urmeaza X3 sau X4,

  dupa Y3: urmeaza X2 sau X3,

  dupa Y4: urmeaza X4 sau X1.

Problema:

- cum se calculeaza entropia lui X ?

Entropie conditionata – Echivocitatea (2)

Dat fiind Y din mulţimea mesajelor $Y_1,..., Y_n$ cu $\Sigma_{i=1,n}\ p(Y_i) = 1$,

- fie: $p_Y(X)$ probabilitatea mesajului X condiţionat de Y.

  $p(X,Y)$ probabilitatea mesajelor X şi Y luate împreună:

  $$p(X,Y) = p_Y(X)*p(Y).$$

- Echivocitatea este entropia lui X condiţionat de Y:

  $$H_Y(X) = \Sigma_{X,Y}\ p(X,Y)*\log\ (1/p_Y(X))$$

  $$H_Y(X) = \Sigma_{X,Y}\ p_Y(X)*p(Y)\ \log\ (1/p_Y(X))$$

  $$= \Sigma_Y\ p(Y)\ \Sigma_X\ p_Y(X)*\log\ (1/p_Y(X)).$$

Pentru exemplu:

Echivocitatea este: $H_Y(X) = 4\ (\ (1/4)\ 2\ (1/2)\ \log\ 2) = \log\ 2 = 1.$

H(X) = 2 ➔ cunoaşterea lui Y reduce incertitudinea lui X la un bit.

Confidenţialitatea perfectă

Fie:

- M texte clare cu probabilitatea p(M), Σ_M p(M) = 1.
- C criptograme, cu probabilitatea p(C), Σ_C p(C) = 1.
- K chei cu probabilitatea p(K), Σ_K p(K) = 1.
- $p_C(M)$ probabiltiatea să se fi transmis M când se recepţionează C.

Confidenţialitatea perfectă <=> $p_C(M) = p(M)$.

Fie $p_M(C)$ probabilitatea să se recepţioneze C când s-a transmis M:

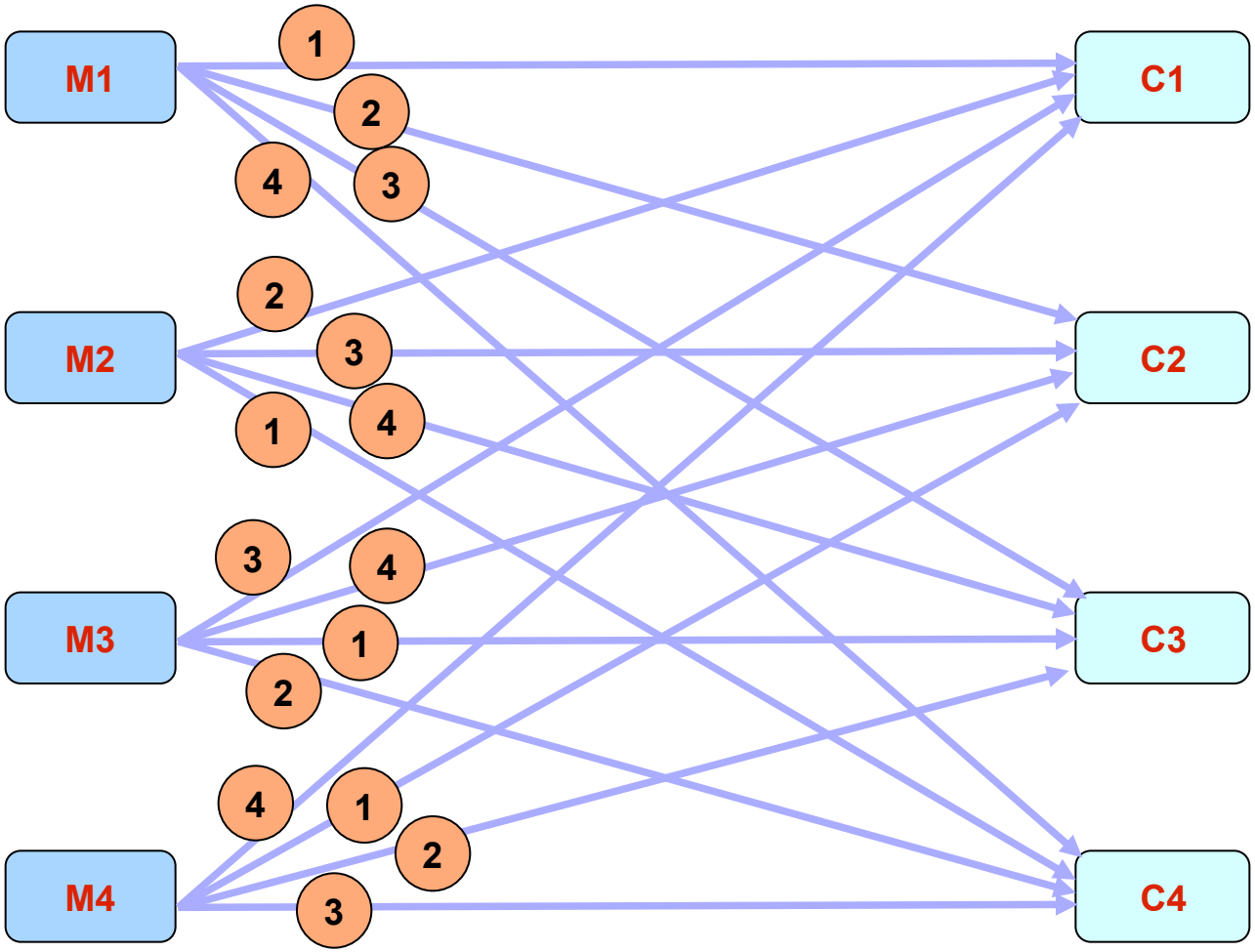
$p_M(C) = \Sigma_{k,\ Ek(M)=C}\ p(k)$.

- Confidenţialitatea perfectă:

  $p_M(C) = p(C)$, pentru toate M şi orice C.

Confidenţialitatea perfectă

- Confidenţialitatea perfectă este posibilă dacă se folosesc chei la fel de lungi ca mesajele codificate.



Distanţa de unicitate

- "Spargerea" confidenţialităţii depinde de cantitatea de criptograme de care intrusul dispune
  - cantitatea de incertitudine în K cunoscand C, exprimată ca:

$$H_C(K) = \Sigma_C\, p(C) \Sigma_K\, p_C(K) \log\,(1/p_C\,(K))$$

- Dacă $H_C(K)=0$ nu există incertitudine şi cifrul se poate sparge.
- Când creşte lungimea N a textelor cifrate echivocitatea scade.
- Distanţa de unicitate:
  - Cel mai mic N pentru care $H_C(K)$ este foarte apropiat de 0.
- Cifru neconditionat sigur:
  - $H_C(K)$ nu se apropie niciodată de 0.

Calcul aproximativ distanţă unicitate

Pentru un limbaj, consideram mulţimea mesajelor de lungime N

- cu entropia H(X)
- fiecare mesaj este o secventa de N simboluri dintr-un alfabet A
- alfabetul are L simboluri
- nu toate combinatiile de N simboluri au sens
  - ex. anumite succesiuni de consoane, digrame, trigrame, etc.

Rata limbajului este entropia pe simbol:

$$r = H(X) / N$$

- r = numar de biti pentru un simbol → total 2^r simboluri
- pentru limba engleză r = 1 ... 1.5 biti pe litera

- Numarul mesajelor de lungime N, cu sens este 2^{rN}

Calcul aproximativ distanţă unicitate (2)

Pentru toate combinatiile posibile, se definește

Rata absolută a limbajului (entropia pe simbol):

- $R = \log L = \Sigma_{i=1,L} (1/L) \log (L)$
- pentru limba engleză R = log 26 = 4.7 biţi pe literă

- Numarul de mesaje posibile, de lungime N este 2^{RN}

Redundanţa D este

- D = R - r
- D = 3.2 ... 3.7 în limba engleză.
- se datorează cuvintelor fără sens

Calcul aproximativ distanţă unicitate (3)

- Ipoteze:
  - Sunt 2^{RN} mesaje posibile de lungime N, din care 2^{rN} au sens.
  - Toate mesajele cu sens au aceeaşi probabilitate, $1/2^{rN}$.
  - Toate mesajele fără sens au probabilitate 0.
  - Sunt $2^{H(K)}$ chei cu probabilităţi egale.
  - Cifrul este aleator:
    - Pentru fiecare k şi C, descifrarea $D_K(C)$ este variabilă aleatoare independentă uniform distribuită pe toate mesajele, cu sau fără sens.

Calcul aproximativ distanţă unicitate (4)

- Fie criptograma $C = E_K(M)$.
  - Criptanalistul are de ales între $2^{H(K)}$ chei, doar una este corectă.
  - Rămân $2^{H(K)}-1$ chei care pot da soluţie falsă

    (Adica acelaşi C se obţine criptând un alt mesaj M' cu înţeles, cu o cheie diferita de K)
  - cu aceeaşi probabilitate $q = 2^{rN} / 2^{RN} = 2^{-DN}$

    D = R-r este redundanţa limbajului
  - Numărul de soluţii false F:

    $F = (2^{H(K)} -1)q = (2^{H(K)} -1)\, 2^{-DN} \approx 2^{H(K)-DN}$

    conditia de unicitate → $\log F = H(K)-DN = 0$

    → $N = H(K) / D$ Interpretare!

Cifrarea prin substitutie

Cifrul lui Cezar (substitutie monoalfabetică)

```
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
| | | | | | | | | | | | | | | | | | | | | | | |
D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
```

textul clar: **CRIPTOGRAFIE**

text cifrat: **FULSWRJUDILH**

Relatia de calcul

$c[i] = (m[i] + 3) \bmod 26$

In general

$c[i] = (a.m[i] + b) \bmod n.$

Substitutia polialfabetică (Vigenere)

foloseste 36 de cifruri Cezar si o cheie de cifrare de lungime l

fiecare litera din cheie = substitutul literei A din textul clar

Exemplu: cheia POLIGRAF

POLIGRAFPOLIGRAGPOLIGRAFPOLIGRAFPOLI

AFOSTODATACANPOVESTIAFOSTCANICIODATA

PTZAZFDFIONITGOATGEQGWOXIQLVOTITSOEI

Cifrarea prin substituţie

- Cifrul Beaufort:
  - Cifrare: c[i] = (k[i] - m[i]) mod n.
  - Descifrare: m[i] = (k[i] - c[i]) mod n
- Substituţia poligrafică:
  - Un grup de n litere este înlocuit cu un alt grup de n litere.

Analiza cifrării prin substituţie

- Substituţie monoalfabetică:
  - N = H(K) / D = log n! / D
  - Pentru limba engleză:
    - N = log 26! / 3.2 = 27.6
- Substituţie periodică cu perioada d:
  - Sunt s^d chei posibile pentru fiecare substiţutie simplă:
    - N = H(K) / D = log s^d / D = (d*log s) / D
  - Pentru cifrul Vigenere s = 26:
    - N = d * 4.7 / 3.2 = 1.5 d

Cifrarea prin transpozitie

Modifică ordinea caracterelor. Uzual:

- textul clar dispus în liniile succesive ale unei matrice si
- parcurgerea acesteia după o anumită regulă pentru stabilirea noii succesiuni de caractere.

Exemplu

- caracterele dispuse pe linii sunt citite pe coloane,
- ordinea coloanelor este dată de ordinea alfabetică a literelor unei chei.

`cheie:` `POLIGRAF`

`ordine:` `76543812`

text clar: **AFOSTODATACANPOVESTIAFOSTCANICIO**

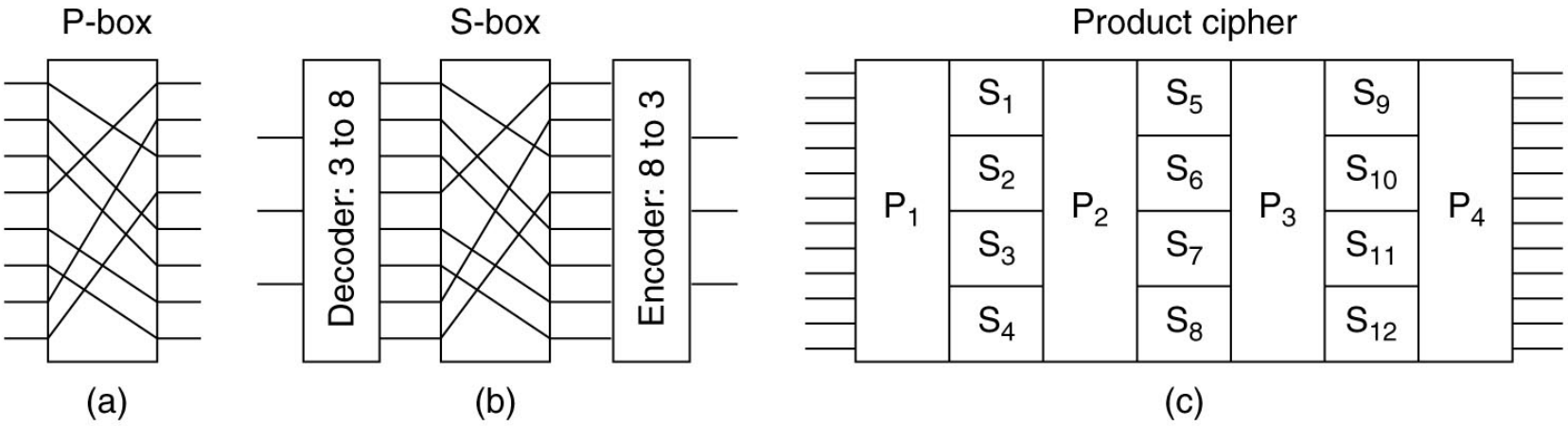
POLIGRAF
AFOSTODA
TACANPOV
ESTIAFOS
TCANICIO

text cifrat: **DOOIAVSOTNAISAINOCTAFASCATETOPFC**

Analiza cifrării prin transpoziţie

- Pentru spargerea cifrului:
  - Cifrul permută caracterele cu o perioadă fixă d.
  - Sunt d! permutări posibile.
  - Toate sunt echiprobabile.
- H(K) = log d!
  - N = H(K) / D = log d! / D
  - N = d log (d/e) / D
- Pentru d = 27 şi D = 3.2 rezultă:
  - N = 27.9

Cifruri produs

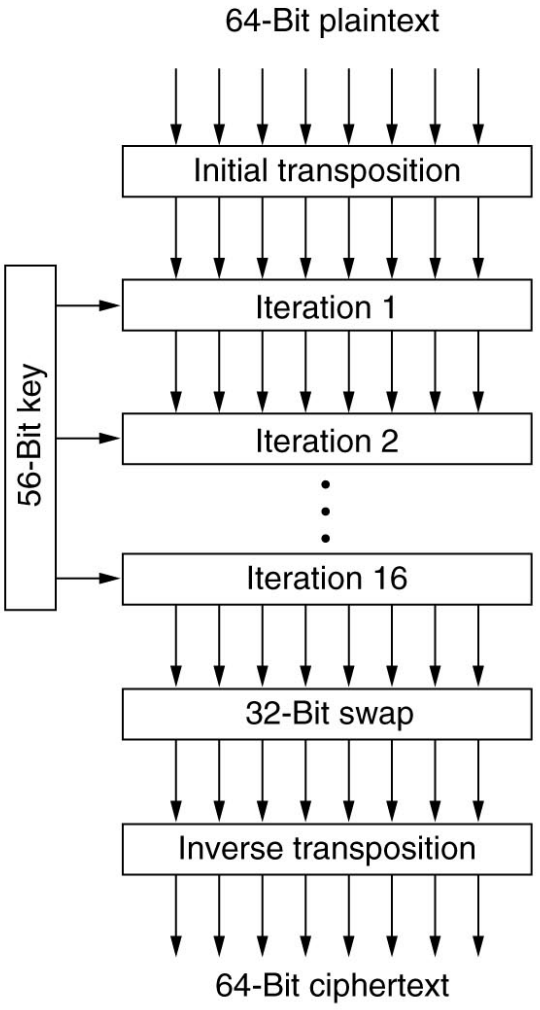


Principii pentru a obţine o securitate mai mare:

- compune două cifruri "slabe", complementare
  - P-box – permutare - asigură difuzia
  - S-box – substitutie - asigură confuzia
- repetă aplicarea permutării şi substituţiei

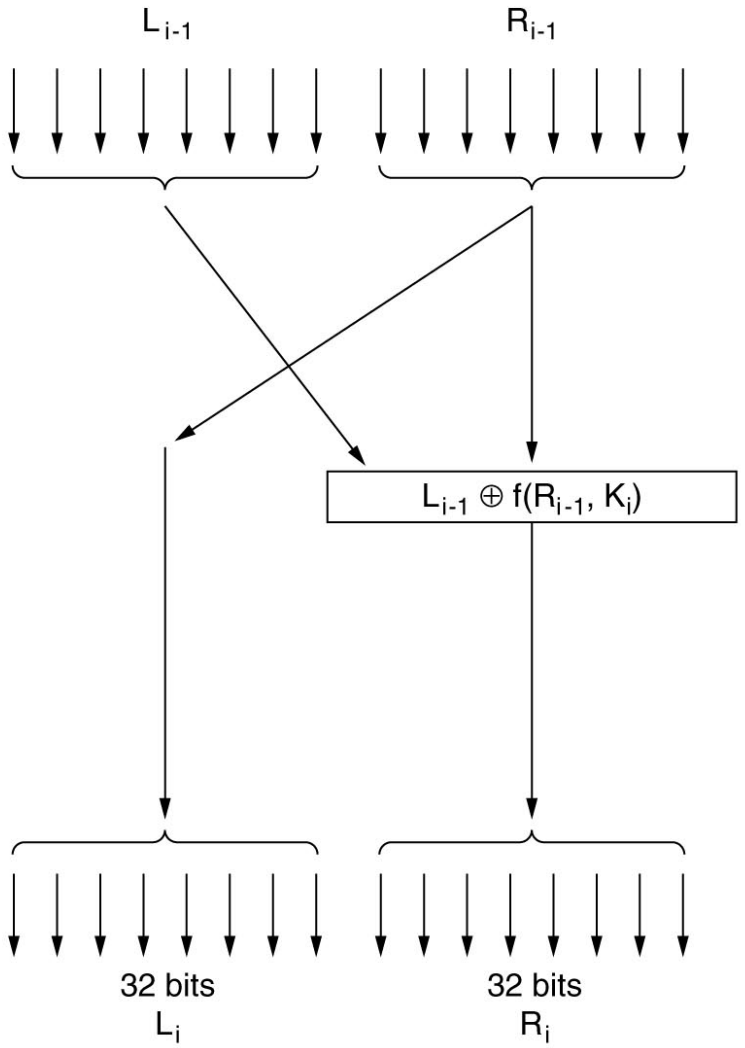
DES (Data Encryption Standard)

Schema generală



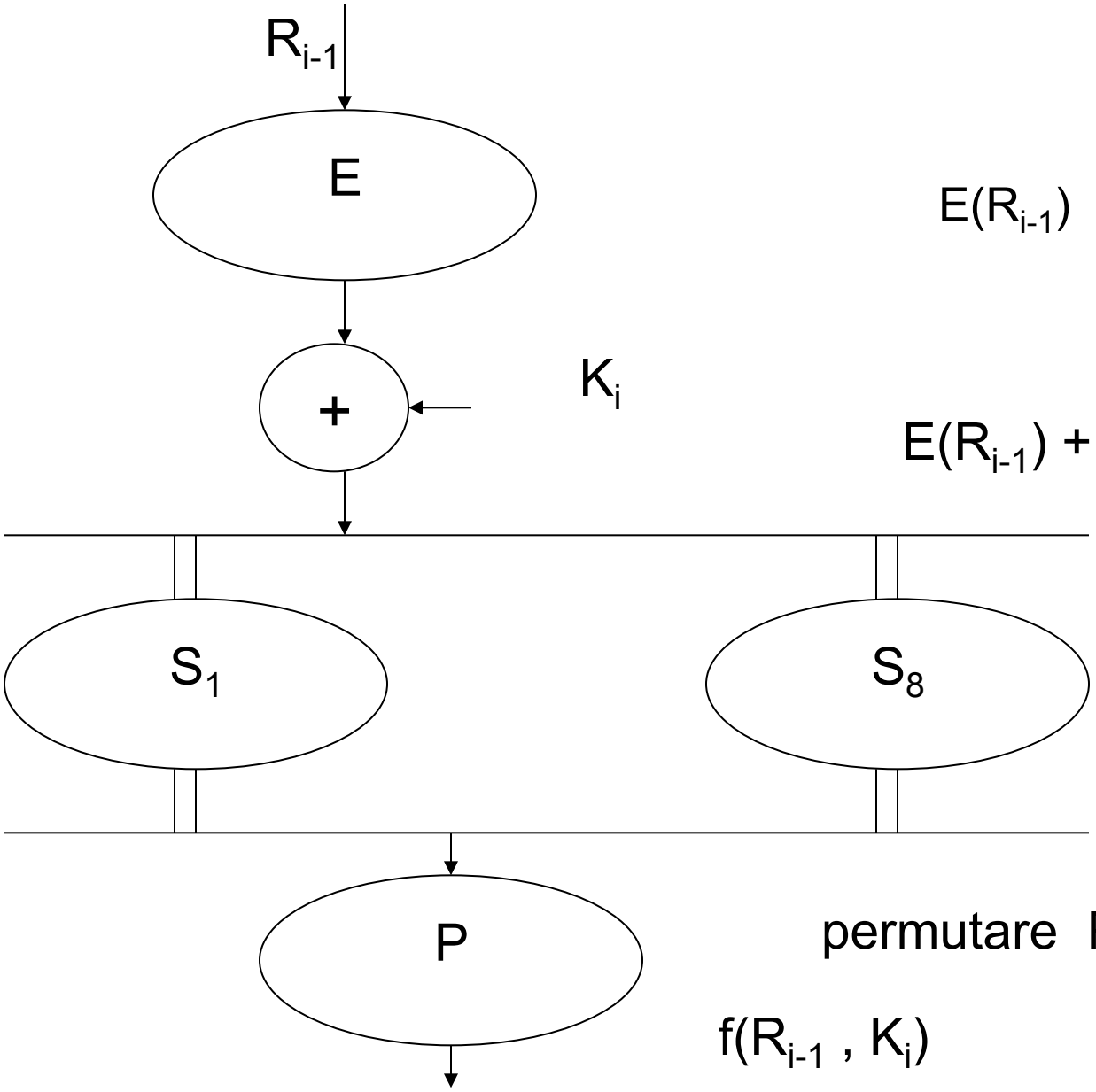
(a)

O iteraţie



(b)

Calculul lui $f(R_{i-1},k_i)$



32 biti

$E(R_{i-1})$

expandare la 48 de biti prin repetarea unora

$E(R_{i-1}) + Ki => B_1B_2...B_8$

8 blocuri a 6 biti sunt intrari in S-boxes

$S_j(B_j)$

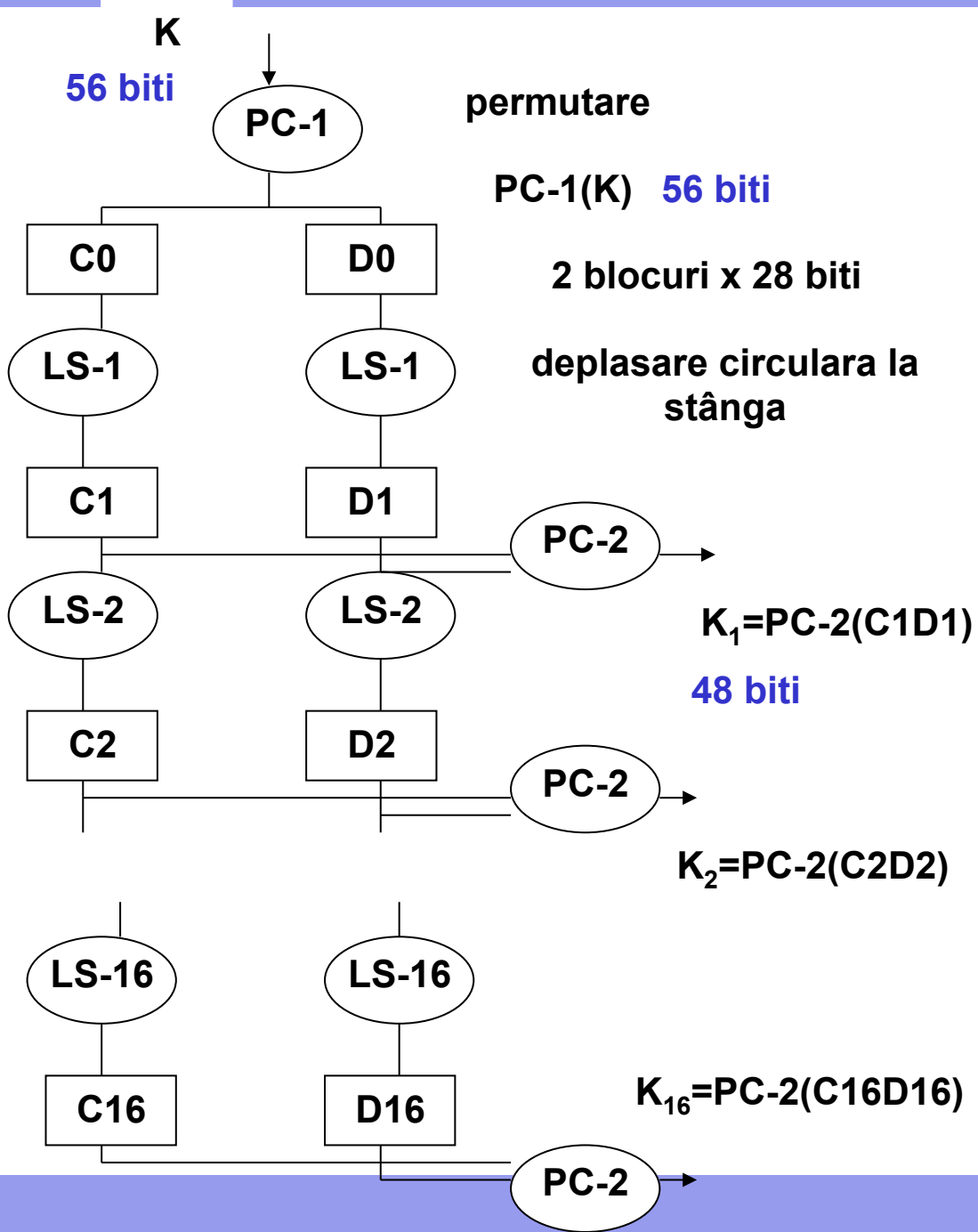
Fiecare Sj produce un cod de 4 biti

permutare $P(S_1(B_1), ... S_8(B_8))$

$f(R_{i-1} , K_i)$

32 biti

Calculul cheilor



permutare

PC-1(K) 56 biti

2 blocuri x 28 biti

deplasare circulara la stânga

(cu 1 sau 2 biti,
in functie de numarul ciclului)

K_1=PC-2(C1D1)

48 biti

K_2=PC-2(C2D2)

K_{16}=PC-2(C16D16)

Comentarii

- Transpoziţiile, expandările, substituţiile, permutările sunt defininte în DES
- Acelaşi algoritm folosit la criptare si decriptare

La criptare: $L_j = R_{j-1}$

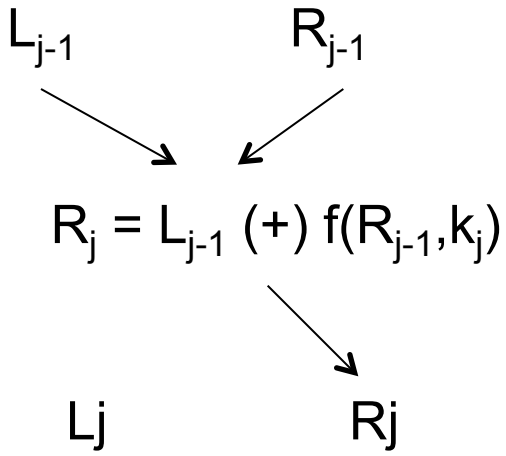
$R_j = L_{j-1}$ (+) $f(R_{j-1},k_j)$

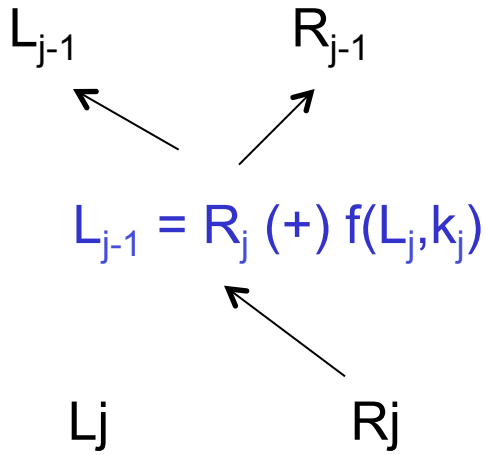
De unde: $R_{j-1} = L_j$

$L_{j-1} = R_j$ (+) $f(R_{j-1},k_j)$

şi $L_{j-1} = R_j$ (+) $f(L_j,k_j)$

Decriptare = ordine inversă criptarii (cu cheile în ordinea $k_{16} – k_1$)





Comentarii (2)

- Elementele cheie ale algoritmului nu au fost făcute publice
  - Controverse
    - Există trape care să uşureze decriptarea de către NSA?

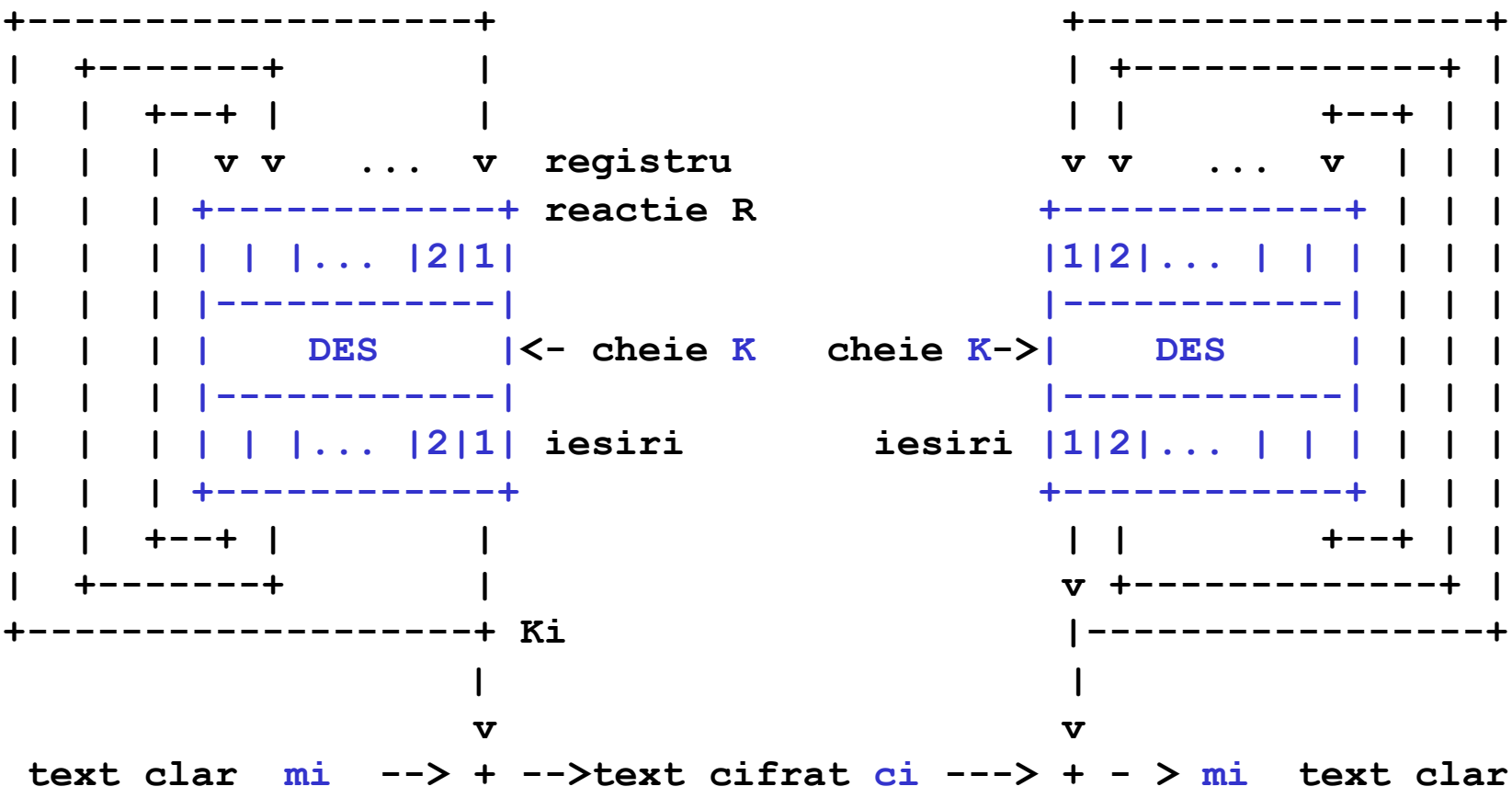
      NSA declară că NU.
    - Descoperirea şi folosirea unei astfel de trape de un criptanalist răuvoitor
    - Urmarea – unele detalii despre S-box au fost dezvăluite de NSA

Comentarii (3)

- Număr de iteraţii – suficiente pentru difuzie?
  - Experimental, după 8 iteraţii nu se mai văd dependenţe ale biţilor de ieşire de grupuri de biţi din intrare
- Lungimea cheii
  - Cheie DES de 56 biţi spartă prin forţă brută (4 luni * 3500 maşini) în 1997
  - Dar, nu au fost raportate deficienţe în algoritm
  - Triple DES “măreşte” lungimea cheii

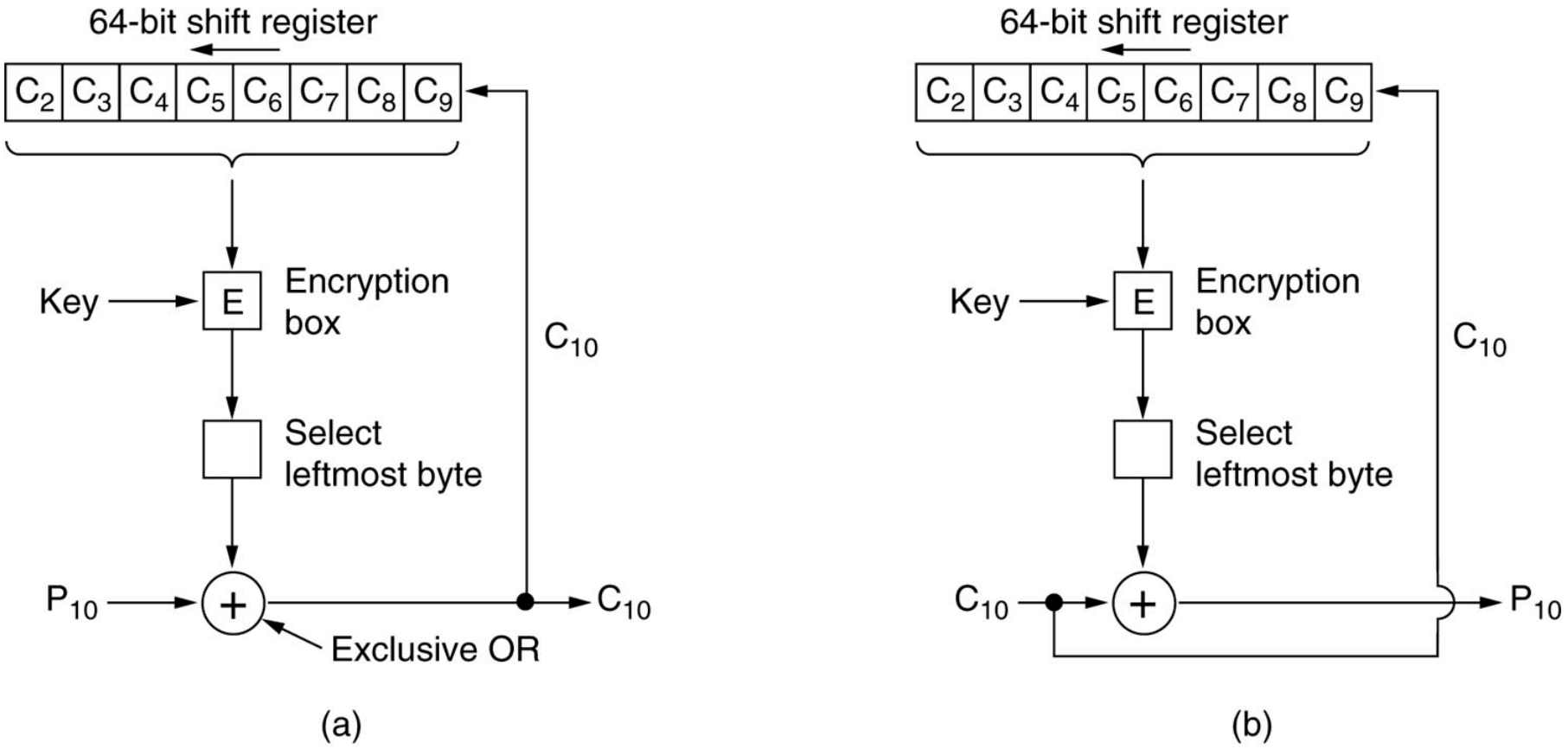
Cifrarea secvențială

Sistem secvential sincron cu reactie bloc (OFB - Output Fead Back)



Foloseste un Initialization Vector ca prima intrare in R
Nu trebuie refolosita aceeasi pereche (K, IV)

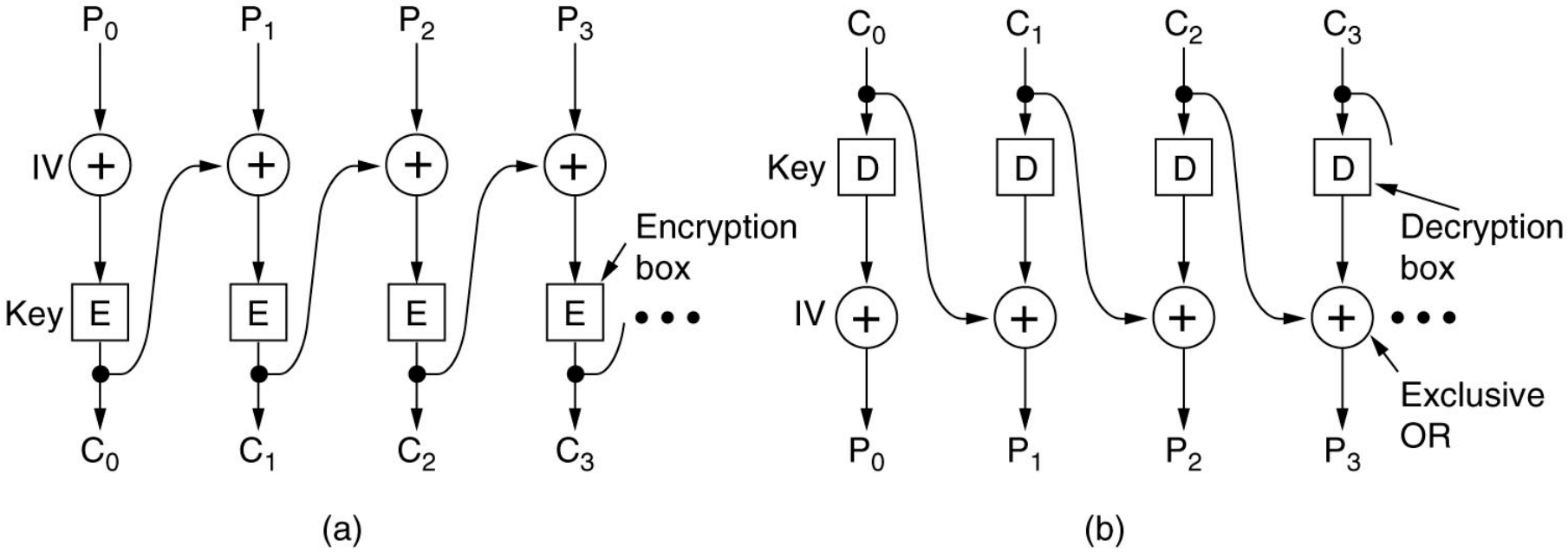
CFB - K-bit Cifer-Feed Back



Foloseste un Initialization Vector ca prima valoare in Registrul de deplasare

O eroare de un bit in criptograma conduce la decriptarea eronata a 8 octeti

CBC - Cipher Block Chaining



Key – cheie secreta

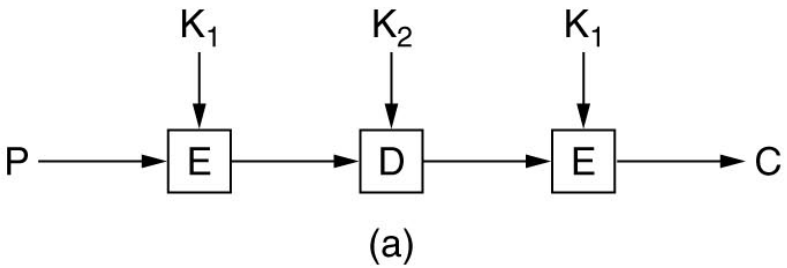
IV – Initialization Vector

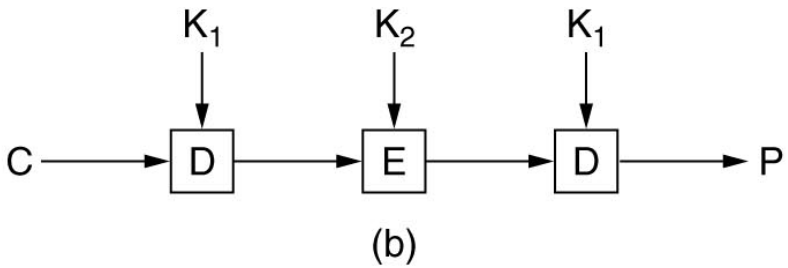
ales aleator, acelasi pentru criptare si decriptare

folosit pentru combinarea cu primul bloc

Avantaj: acelasi text clar repetat in mesaj va fi criptat diferit

Triplu DES





AES – Advanced Encryption Standard

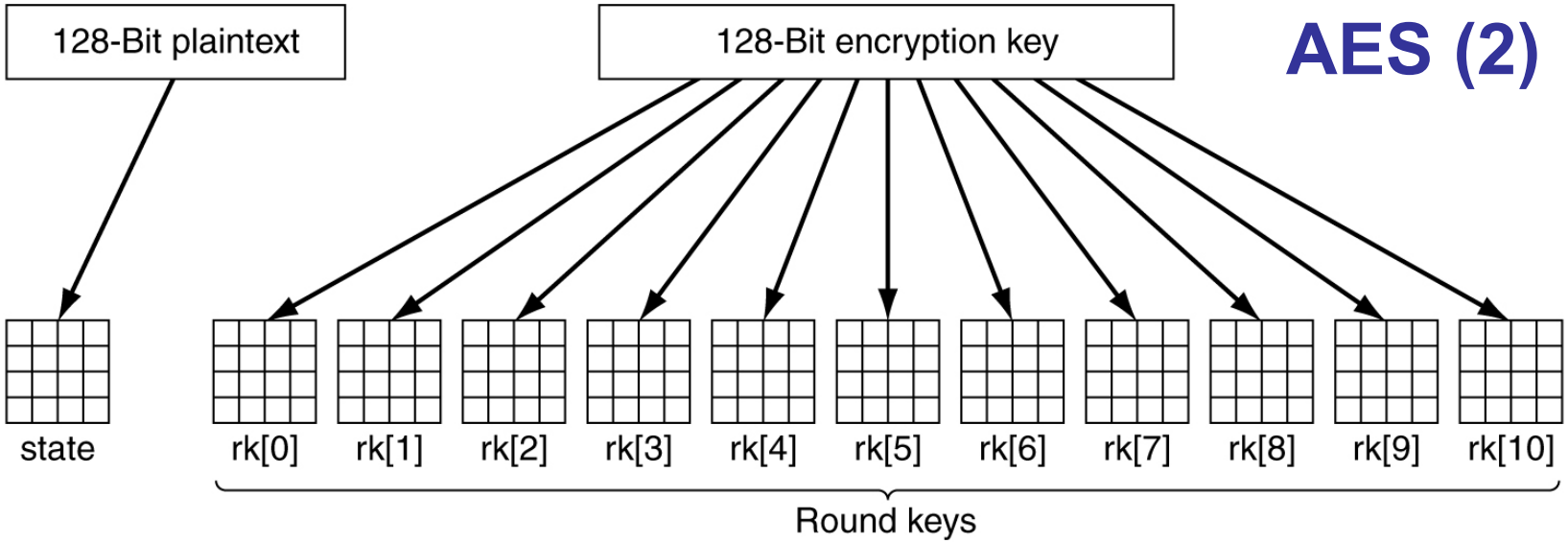
Regulile concursului organizat de NIST (ianuarie 1997) erau:

1. Algoritmul trebuie să fie un cifru bloc simetric.
2. Tot proiectul trebuie sa fie public
3. Trebuie să fie suportate chei de 128, 192, şi de 256 biţi
4. Trebuie să fie posibile atât implementări hardware cât şi software
5. Algoritmul trebuie să fie public sau oferit cu licenţă nediscriminatorie.

Finaliştii şi scorurile lor au fost următoarele:

1. Rijndael (din partea lui Joan Daemen şi Vincent Rijmen, 86 voturi)
2. Serpent (din partea lui Ross Anderson, Eli Biham şi Lars Knudsen, 59 voturi)
3. Twofish (din partea unei echipe condusă de Bruce Schneier, 31 voturi)
4. RC6 (din partea RSA Laboratories, 23 voturi)
5. MARS (din partea IBM, 13 voturi)

AES (2)



n runde (n=10 pentru cheie de lungime 128; 12/ 192, 14/256)

Bloc 128 biţi = matrice 4*4 octeţi – state

Operaţii pe coloane sau pe linii; 4 operaţii pe rundă

substitute – la nivel octet, foloseşte tabel substituţie

rotate_rows – prin deplasare circulară la stânga la nivel octet

```
1   5   9  13          1   5   9  13
2   6  10  14    →     6  10  14   2
3   7  11  15         11  15   3   7
4   8  12  16         16   4   8  12
```

mix_columns – elementele unei coloane sunt înmulţite cu o matrice

$$\begin{vmatrix} s'0i \\ s'1i \\ s'2i \\ s'3i \end{vmatrix} = \begin{vmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{vmatrix} \begin{vmatrix} s0i \\ s1i \\ s2i \\ s3i \end{vmatrix}$$

xor_roundkey_into_state – adaugă o cheie rk[i]

Rijndael definint în câmp Galois $G(2^8)$ prin polinomul $P = x^8+x^4+x^3+x+1$

număr = coeficienţii unui polinom

Ex. $23 = 10111_{(2)}$ este polinomul $1*x^4+0*x^3+1*x^2+1*x+1$

$x^4+ x^2+ x+1$

adunarea coeficienţilor făcută modulo 2

înmulţirea făcută ca la polinoame, dar modulo P

Ex. $(x^3+1)*(x^4+x) = x^7+x^4+x^4+x = x^7+x$

Algoritmul AES (3)

```
#define LENGTH 16                          /* # bytes in data block or key */
#define NROWS 4                            /* number of rows in state */
#define NCOLS 4                            /* number of columns in state */
#define ROUNDS 10                          /* number of iterations */
typedef unsigned char byte;                /* unsigned 8-bit integer */

rijndael(byte plaintext[LENGTH], byte ciphertext[LENGTH], byte key[LENGTH])
{
  int r;                                   /* loop index */
  byte state[NROWS][NCOLS];                /* current state */
  struct {byte k[NROWS][NCOLS];} rk[ROUNDS + 1];      /* round keys */

  expand_key(key, rk);                     /* construct the round keys */
  copy_plaintext_to_state(state, plaintext);   /* init current state */
  xor_roundkey_into_state(state, rk[0]);   /* XOR key into state */

  for (r = 1; r <= ROUNDS; r++) {
       substitute(state);                  /* apply S-box to each byte */
       rotate_rows(state);                 /* rotate row i by i bytes */
       if (r < ROUNDS) mix_columns(state);     /* mix function */
       xor_roundkey_into_state(state, rk[r]);  /* XOR key into state */
  }
  copy_state_to_ciphertext(ciphertext, state);   /* return result */
}
```

Comentarii

- Nu au fost probleme la utilizare
- Experimental – difuzie bună
- Metodă bazată pe algebră (câmpuri Galois)
  - substituţii şi mixare coloane folosesc operaţii cu sens în teoria algebrică (nu simple tabele greu de explicat)
  - autorii nu au oferit argumente matematice
  - nu sunt suspectate trape sau scurtături ascunse

Cifrarea prin functii greu inversabile

- functii greu inversabile
  - cunoscînd x este usor de calculat f(x)
  - calculul lui x din f(x) este foarte dificil.
- adaptare:
  - calculul lui x din f(x) trebuie să fie o problemă intratabilă doar pentru criptanalist
  - nu pentru destinatarul autorizat care
    - **are cheia sau**
    - **dispune de o trapă ce face problema usor de rezolvat.**
- problemă intratabilă - nu există un algoritm de rezolvare în timp polinomial.
- Metode
  - algoritmi exponentiali
  - problema rucsacului.

Algoritmi exponentiali – RSA

In RSA (Rivest, Shamir si Adleman):

Criptarea si decriptarea se fac prin functii exponentiale

Criptarea se face prin calculul

$$C = (M^e) \bmod n$$

unde **(e, n)** reprezintă cheia de criptare.

M este un bloc de mesaj (valoare întreagă între 0 si n-1)

C este criptograma.

Decriptarea se face prin calculul

$$M = (C^d) \bmod n$$

unde **(d, n)** este cheia de decriptare

Algoritmi exponentiali – RSA

Conditia: functiile de criptare si decriptare trebuie sa fie inverse una alteia:

$$(M^e \bmod n)^d \bmod n = M$$

Conditia poate fi indeplinita daca

- e este un intreg relativ prim cu $\Phi(n)$

  $\Phi(n)$ este Functia lui Euler

  adica nr de întregi pozitivi <n relativ primi cu n

- d este inversul multiplicativ al lui e modulo $\Phi(n)$

$$e*d \bmod \Phi(n) = 1$$

- n este produsul a doua numere prime, $n = p * q$

  caz in care $\Phi(n) = (p-1)(q-1)$

Metoda RSA

1. Se aleg două numere prime *p* şi *q*,

   (de obicei de 1024 biţi).

2. Se calculează $n = p \times q$

   şi $z = (p - 1) \times (q - 1)$.

3. Se alege *d* un număr relativ prim cu *z*

   d poate fi un numar prim care satisface

   d > (p-1) si d > (q-1)

4. Se găseşte *e* astfel încât $e \times d = 1 \bmod z$.

5. (e, n) este cheia de criptare.

   (d, n) este cheia de decriptare.

Motivatie

Functia lui Euler

$\Phi(n)$ = nr de întregi pozitivi <n relativ primi cu n

daca p prim => $\Phi(p) = p-1$.

daca $n = p*q$ cu p, q prime atunci

$$\Phi(n) = \Phi(p) * \Phi(q) = (p-1)\,(q-1)$$

T. (Euler). Pentru orice a si n cu $(a,n) = 1$ avem

$$a^{\Phi(n)} \bmod n = 1$$

T. (cifrare). Date fiind e si d care satisfac

$$ed \bmod \Phi(n) = 1$$

si un mesaj $M \in [0, n-1]$, avem

$$(M^e \bmod n)^d \bmod n = M$$

Dem. $ed \bmod \Phi(n) = 1 \Rightarrow ed = t\,\Phi(n) + 1$ ptr. un anumit t.

$(M^e \bmod n)^d \bmod n = M^{ed} \bmod n$

$= M^{\,t\,\Phi(n)\,+\,1} \bmod n$

$= M * M^{\,t\,\Phi(n)} \bmod n = M(M^{\,t\,\Phi(n)} \bmod n) \bmod n$

$= M((M^{\,\Phi(n)} \bmod n)^t \bmod n) \bmod n$

$= M((1)^t \bmod n) \bmod n$

$= (M.1) \bmod n = M$

Ex:

Alegem p = 3 şi q = 11, rezultând n = 33 şi z = 20

Alegem d = 7 (7 şi 20 nu au factori comuni)

***e* poate fi găsit din 7 × *e* = 1 mod 20, care dă *e* = 3.**

Plaintext (P)			Ciphertext (C)		After decryption	
Symbolic	Numeric	P^3	P^3 (mod 33)	C^7	C^7 (mod 33)	Symbolic
S	19	6859	28	13492928512	19	S
U	21	9261	21	1801088541	21	U
Z	26	17576	20	1280000000	26	Z
A	01	1	1	1	01	A
N	14	2744	5	78125	14	N
N	14	2744	5	78125	14	N
E	05	125	26	8031810176	05	E

Sender's computation — Receiver's computation

Comentarii

Cheia de criptare (e,n) se face publica

Problema:

- cunoasterea lui (e,n) sa nu permita deducerea lui d

Securitate pastrata:

- p si q sunt numere prime foarte mari
- p si q sunt pastrate secrete

Prin simetrie, cifrarea si descrifrarea sunt comutative si mutual inverse

$$(M^e \bmod n)^d \bmod n = M$$

=> RSA utilizată ptr confidentialitate si autentificare.

Nu au fost identificate atacuri reusite cu RSA

Metoda MH (Merkle si Hellman)

Problema rucsacului

Se dau ponderile A = (a[1], a[2], …, a[m])

Se cere determinarea lui X = (x[1],x[2],...x[m]) cu elemente binare, a.i.

$$C = \Sigma_{i=1,m} \; x[i] * a[i]$$

Găsirea unei solutii = backtracking

=> număr operatii care creste exponential cu m.

O solutie **x** poate fi verificată prin cel mult m operatii de adunare

Varianta rucsac simplu a problemei (trapa):

dacă A satisface proprietatea de dominantă (este o secventa super-crescatoare), adică

$$a[i] > \sum_{j=1,i-1} a[j]$$

atunci problema poate fi rezolvată în timp liniar.

Ex.

text clar	**1**	**0**	**1**	**0**	**0**	**1**
rucsac	**1**	**2**	**5**	**9**	**20**	**43**
text cifrat	**1**		**5**			**43**

suma = 49 reprezinta criptograma

decriptarea ?

Cheie publica: secventa (oarecare) de intregi

Cheie secreta: secventa super-crescatoare

Contributia Merkle si Hellman

conversie secventa (oarecare) ⇔ secventa super-crescatoare

Solutia: aritmetica modulara

rucsac simplu A = [a1, a2, …, am]

rucsac greu G = [g1, g2, …,gm]

se obtine prin calcule gi = w * ai mod n

Ex.

rucsac simplu A = [1, 2, 4, 9], w = 15, n = 17

1*15 mod 17 = 15 mod 17 = 15

2*15 mod 17 = 30 mod 17 = 13

4*15 mod 17 = 60 mod 17 = 9

9*15 mod 17 = 135 mod 17 = 16

rucsac greu G = [15, 13, 9, 16]

Obs. inmultirea mod n strica proprietatea de dominanta

Conditii:

Toate numerele din G trebuie sa fie distincte intre ele

Conversia inversa de la G la A trebuie sa produca o solutie unica

Impun restrictii asupra lui n si w; ex.:

w = 3; n = 6

x	**3*x**	**3*x mod 6**
1	3	3
2	6	0
3	9	3
4	12	0
5	15	3
6	18	0

w = 3; n = 5

x	**3*x**	**3*x mod 5**
1	3	3
2	6	1
3	9	4
4	12	2
5	15	0
6	18	3

Cerinte:

1. n trebuie sa fie mai mare decat suma tuturor ai
2. w si n trebuie sa fie prime intre ele (se alege n prim)

   => w are un invers multiplicativ w^{-1} ($w * w^{-1} = 1 \bmod n$)

Criptare

Obtine criptograma C din textul clar P prin $C = G * P$

unde G este rucsacul greu, $G = w * A \bmod n$ (adica $g_i = w * a_i \bmod n$)

Ex: P = [1,0,1,0], G = [15,13,9,16] ➔ C = 15+9 = **24**

Decriptare

Receptorul cunoaste A,w, n si, bineinteles, G

Deoarece $C = G * P = w * A * P \bmod n$, rezulta

$$w^{-1} * C = w^{-1} * G * P = w^{-1} * w * A * P \bmod n = A * P \bmod n$$

din care P se afla prin rucsac simplu

Ex. A = [1, 2, 4, 9], w = 15, n = 17, C = **24**

$w^{-1} = 15^{-1} = 8 \bmod 17$ ➔ $w^{-1} * C = 8 * 24 = 192 \bmod 17 =$ **5**

A = [1, 2, 4, 9] => P = [1, 0, 1, 0]

Comentarii

Metoda pare sigura

S-au gasit metode de atac prin ocolire rucsac greu in anumite cazuri

Oricum, algoritmul MH este greu de utilizat