Criptografie și Securitate

- Prelegerea 8 -
Sisteme de criptare bloc

Adela Georgescu, Ruxandra F. Olimid

Facultatea de Matematică și Informatică
Universitatea din București

Cuprins

1. Definiție

2. Construcție

3. Moduri de utilizare

4. Exemple

Criptografia simetrică

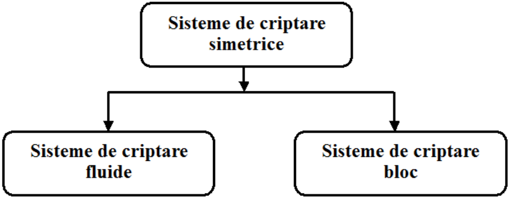
- Am studiat sisteme simetrice care criptează **bit cu bit** - sisteme de criptare fluide;

Criptografia simetrică

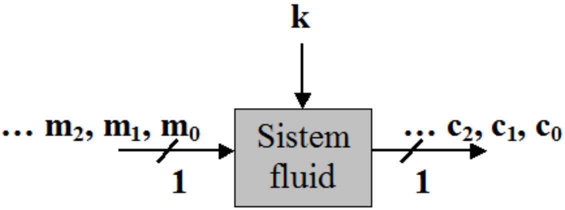
- Am studiat sisteme simetrice care criptează **bit cu bit** - sisteme de criptare fluide;
- Vom studia sisteme simetrice care criptează **câte n biți simulan** - sisteme de criptare bloc;

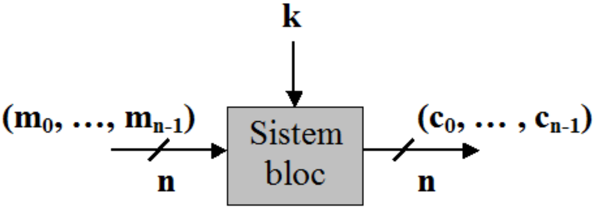
Criptografia simetrică

- Am studiat sisteme simetrice care criptează **bit cu bit** - sisteme de criptare fluide;
- Vom studia sisteme simetrice care criptează **câte n biți simulan** - sisteme de criptare bloc;



Sisteme bloc vs. sisteme fluide





Sisteme bloc vs. sisteme fluide

... d.p.d.v. al modului de criptare:

Sisteme fluide

- criptarea biților se realizează individual
- criptarea unui bit din textul clar este independentă de orice alt bit din textul clar

Sisteme bloc

- criptarea se realizează în blocuri de câte n biți
- criptarea unui bit din textul clar este dependentă de biții din textul clar care aparțin aceluiași bloc

Sisteme bloc vs. sisteme fluide

... d.p.d.v. *tradițional*, în practică:

Sisteme fluide

- necesități computaționale reduse
- utilizare: telefoane mobile, dispozitive încorporate, PDA
- par să fie mai puțin sigure, multe sunt sparte

Sisteme bloc

- necesități computaționale mai avansate
- utilizare: internet
- par să fie mai sigure, prezintă încredere mai mare

Sisteme bloc

- Introducem noțiunea de permutare pseudoaleatoare sau PRP *(PseudoRandom Permutation)*

Sisteme bloc

- Introducem noțiunea de permutare pseudoaleatoare sau PRP *(PseudoRandom Permutation)*
- În analogie cu ce știm deja:
  - **PRP** sunt necesare pentru construcția **sistemelor bloc**

Sisteme bloc

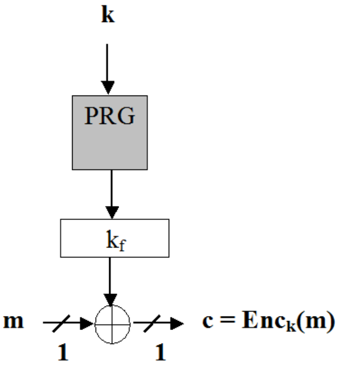
- Introducem noțiunea de permutare pseudoaleatoare sau PRP *(PseudoRandom Permutation)*
- În analogie cu ce știm deja:
  - **PRP** sunt necesare pentru construcția **sistemelor bloc**

asa cum

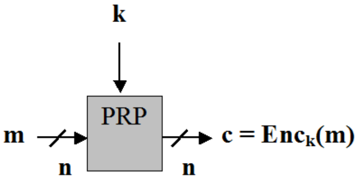
  - **PRG** sunt necesare pentru construcția **sistemelor fluide**

Sisteme bloc

Sisteme fluide



Sisteme bloc



PRP

- Ramâne să definim noțiunea de permutare pseudoaleatoare sau PRP *(PseudoRandom Permutation)*;

PRP

- Ramâne să definim noțiunea de permutare pseudoaleatoare sau PRP *(PseudoRandom Permutation)*;
- Acesta este o funcție **deterministă** și **bijectivă** care pentru o cheie fixată produce la ieșire o **permutare** a intrării ...

PRP

- Ramâne să definim noțiunea de permutare pseudoaleatoare sau PRP *(PseudoRandom Permutation)*;
- Acesta este o funcție **deterministă** și **bijectivă** care pentru o cheie fixată produce la ieșire o **permutare** a intrării ...
- ... **indistinctibilă** față de o permutare aleatoare;

PRP

- Ramâne să definim noțiunea de permutare pseudoaleatoare sau PRP *(PseudoRandom Permutation)*;
- Acesta este o funcție **deterministă** și **bijectivă** care pentru o cheie fixată produce la ieșire o **permutare** a intrării ...
- ... **indistinctibilă** față de o permutare aleatoare;
- În plus, atât funcția cât și inversa sa sunt **eficient calculabile**.

PRP

Definitie

O permutare pseudoaleatoare definită peste $(\mathcal{K}, \mathcal{X})$ *este o funcție bijectivă*

$$F : \mathcal{X} \times \mathcal{K} \rightarrow \mathcal{X}$$

care satisface următoarele proprietăți:

1. *Eficiență:* $\forall k \in \mathcal{K}, x \in \mathcal{X}, \exists$ *algoritmi determiniști PPT care calculează* $F_k(x)$ *și* $F_k^{-1}(x)$
2. *Pseudoaleatorism:* $\forall$ *algoritm PPT* $\mathcal{D}$, $\exists$ *o funcție neglijabilă negl a.î.:*

$$|Pr[D(r) = 1] - Pr[D(F_k(\cdot)) = 1]| \leq negl(n)$$

unde $r \leftarrow^R Perm(X), k \leftarrow^R \mathcal{K}$

Notații

- $F_k(x) = F(k, x)$
  o cheie este în general (aleator) aleasă și apoi fixată
- $Perm(X) =$ mulțimea tuturor funcțiilor bijective de la $\mathcal{X}$ la $\mathcal{X}$
- $\mathcal{X} = \{0, 1\}^n$
- $\mathcal{D} =$ *Distringuisher* care are acces la *oracolul* de evaluare a funcției

PRF

- Introducem noțiunea de funcție pseudoaleatoare sau PRF *(PseudoRandom Function)*...

PRF

- Introducem noțiunea de funcție pseudoaleatoare sau PRF *(PseudoRandom Function)*...
- ... ca o generalizare noțiunii de **permutare pseudoaleatoare**;

PRF

- Introducem noțiunea de funcție pseudoaleatoare sau PRF *(PseudoRandom Function)*...
- ... ca o generalizare noțiunii de **permutare pseudoaleatoare**;
- Acesta este o funcție **cu cheie** care este **indistinctibilă** față de o funcție aleatoare (cu același domeniu și mulțime de valori).

PRF

Definitie

O funcție pseudoaleatoare definită peste $(\mathcal{K}, \mathcal{X}, \mathcal{Y})$ *este o funcție bijectivă*

$$F : \mathcal{X} \times \mathcal{K} \rightarrow \mathcal{Y}$$

care satisface următoarele proprietăți:

1. *Eficiență:* $\forall k \in \mathcal{K}, x \in \mathcal{X}, \exists$ *algoritm PPT care calculează* $F_k(x)$
2. *Pseudoaleatorism:* $\forall$ *algoritm PPT* $\mathcal{D}$, $\exists$ *o funcție neglijabilă negl a.î.:*

$$|Pr[D(r) = 1] - Pr[D(F_k(\cdot)) = 1]| \leq negl(n)$$

unde $r \leftarrow^R Func(X, Y), k \leftarrow^R \mathcal{K}$

Notații

- $F_k(x) = F(k, x)$
  o cheie este în general (aleator) aleasă și apoi fixată
- $Func(X, Y) =$ mulțimea funcțiilor de la $\mathcal{X}$ la $\mathcal{Y}$
- $\mathcal{X} = \{0, 1\}^n$, $\mathcal{Y} = \{0, 1\}^n$
  considerăm în general că *PRF păstrează lungimea*
- $\mathcal{D} =$ *Distringuisher* care are acces la *oracolul* de evaluare a funcției

$PRP \subseteq PRF$

- Întrebare: De ce PRF poate fi privită ca o generalizare a PRP?

PRP $\subseteq$ *PRF*

- Întrebare: De ce *PRF* poate fi privită ca o generalizare a *PRP*?
- Răspuns: *PRP* este *PRF* care satisface:

PRP ⊆ *PRF*

- Întrebare: De ce *PRF* poate fi privită ca o generalizare a *PRP*?
- Răspuns: *PRP* este *PRF* care satisface:
  1. $\mathcal{X} = \mathcal{Y}$
  2. este inversabilă
  3. calculul funcției inverse este eficient

Construcții

- **PRF** $\Rightarrow$ **PRG**
  Pornind de la PRF se poate construi PRG

Construcții

- **PRF** $\Rightarrow$ **PRG**
  Pornind de la PRF se poate construi PRG
- **PRG** $\Rightarrow$ **PRF**
  Pornind de la PRG se poate construi PRF

Construcții

- **PRF $\Rightarrow$ PRG**
  Pornind de la PRF se poate construi PRG
- **PRG $\Rightarrow$ PRF**
  Pornind de la PRG se poate construi PRF
- **PRP $\Rightarrow$ PRF**
  Pornind de la PRP se poate construi PRF

Construcții

- **PRF $\Rightarrow$ PRG**
  Pornind de la PRF se poate construi PRG
- **PRG $\Rightarrow$ PRF**
  Pornind de la PRG se poate construi PRF
- **PRP $\Rightarrow$ PRF**
  Pornind de la PRP se poate construi PRF
- **PRF $\Rightarrow$ PRP**
  Pornind de la PRF se poate construi PRP

Construcții

- **PRF $\Rightarrow$ PRG**
  Pornind de la PRF se poate construi PRG
- **PRG $\Rightarrow$ PRF**
  Pornind de la PRG se poate construi PRF
- **PRP $\Rightarrow$ PRF**
  Pornind de la PRP se poate construi PRF
- **PRF $\Rightarrow$ PRP**
  Pornind de la PRF se poate construi PRP

Întrebare: Care dintre aceste construcții este trivială?

Construcții

Răspuns: **PRP** $\Rightarrow$ **PRF**

Construcții

Răspuns: **PRP $\Rightarrow$ PRF**

PRP este o particularizare a *PRF* : $\mathcal{X} \times \mathcal{K} \rightarrow Y$ care satisface:

1. $\mathcal{X} = \mathcal{Y}$
2. este inversabilă
3. calculul funcției inverse este eficient

Construcții

- **PRF $\Rightarrow$ PRG**
  Pornind de la PRF se poate construi PRG
- **PRG $\Rightarrow$ PRF**
  Pornind de la PRG se poate construi PRF
- **PRP $\Rightarrow$ PRF** $\surd$
  Pornind de la PRP se poate construi PRF
- **PRF $\Rightarrow$ PRP**
  Pornind de la PRF se poate construi PRP

$PRF \Rightarrow PRG$

- Considerăm $F : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ PRF;

PRF $\Rightarrow$ PRG

- Considerăm $F : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ PRF;
- Construim $G : \mathcal{K} \rightarrow \{0,1\}^{nl}$ *PRG* sigur:

$$G(k) = F_k(0)||F_k(1)||\ldots||F_k(l-1)$$

$PRF \Rightarrow PRG$

- Considerăm $F : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ PRF;
- Construim $G : \mathcal{K} \rightarrow \{0,1\}^{nl}$ PRG sigur:

$$G(k) = F_k(0)||F_k(1)||\ldots||F_k(l-1)$$

- Întrebare: De ce este G sigur?

$PRF \Rightarrow PRG$

- Considerăm $F : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ PRF;
- Construim $G : \mathcal{K} \rightarrow \{0,1\}^{nl}$ *PRG* sigur:

$$G(k) = F_k(0)||F_k(1)||\ldots||F_k(l-1)$$

- Întrebare: De ce este G sigur?
- Răspuns: $F_k(\cdot)$ este *indistinctibilă* față de o funcție aleatoare

$PRF \Rightarrow PRG$

- Considerăm $F : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ PRF;
- Construim $G : \mathcal{K} \rightarrow \{0,1\}^{nl}$ *PRG* sigur:

$$G(k) = F_k(0)||F_k(1)||\ldots||F_k(l-1)$$

- Întrebare: De ce este G sigur?
- Răspuns: $F_k(\cdot)$ este *indistinctibilă* față de o funcție aleatoare $\Rightarrow G(k)$ este *indistinctibilă* față de o secvență aleatoare de lungime ln.

$PRF \Rightarrow PRG$

- Considerăm $F : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ PRF;
- Construim $G : \mathcal{K} \rightarrow \{0,1\}^{nl}$ *PRG* sigur:

$$G(k) = F_k(0)||F_k(1)||\ldots||F_k(l-1)$$

- Întrebare: De ce este G sigur?
- Răspuns: $F_k(\cdot)$ este *indistinctibilă* față de o funcție aleatoare $\Rightarrow G(k)$ este *indistinctibilă* față de o secvență aleatoare de lungime ln.
- Avantaj: Construcția este *paralelizabilă*

Construcții

- **PRF** $\Rightarrow$ **PRG** $\surd$
  Pornind de la PRF se poate construi PRG
- **PRG** $\Rightarrow$ **PRF**
  Pornind de la PRG se poate construi PRF
- **PRP** $\Rightarrow$ **PRF** $\surd$
  Pornind de la PRP se poate construi PRF
- **PRF** $\Rightarrow$ **PRP**
  Pornind de la PRF se poate construi PRP

$PRG \Rightarrow PRF$

- Considerăm $G : \mathcal{K} \rightarrow \mathcal{K}^2$ PRG cu $|\mathcal{K}| = n$:

$$G(k) = (G_0(k), G_1(k))$$

$G_0(k)$ și $G_1(k)$ sunt prima și a doua jumătate a ieșirii din PRG

$PRG \Rightarrow PRF$

- Considerăm $G : \mathcal{K} \rightarrow \mathcal{K}^2$ PRG cu $|\mathcal{K}| = n$:

$$G(k) = (G_0(k), G_1(k))$$

$G_0(k)$ și $G_1(k)$ sunt prima și a doua jumătate a ieșirii din PRG

- Construim $F : \mathcal{K} \times \{0, 1\} \rightarrow \mathcal{K}$ PRF:

$$F_k(0) = G_0(k), F_k(1) = G_1(k)$$

$F_k(0)$ întoarce prima jumătate a secvenței pseudoaleatoare $G(k)$

$F_k(1)$ întoarce a doua jumătate a secvenței pseudoaleatoare $G(k)$

$PRG \Rightarrow PRF$

- Considerăm $G : \mathcal{K} \rightarrow \mathcal{K}^2$ PRG cu $|\mathcal{K}| = n$:

$$G(k) = (G_0(k), G_1(k))$$

  $G_0(k)$ și $G_1(k)$ sunt prima și a doua jumătate a ieșirii din PRG

- Construim $F : \mathcal{K} \times \{0, 1\} \rightarrow \mathcal{K}$ PRF:

$$F_k(0) = G_0(k), F_k(1) = G_1(k)$$

  $F_k(0)$ întoarce prima jumătate a secvenței pseudoaleatoare $G(k)$

  $F_k(1)$ întoarce a doua jumătate a secvenței pseudoaleatoare $G(k)$

- Întrebare: De ce este F sigură?

$PRG \Rightarrow PRF$

- Considerăm $G : \mathcal{K} \rightarrow \mathcal{K}^2$ PRG cu $|\mathcal{K}| = n$:

$$G(k) = (G_0(k), G_1(k))$$

$G_0(k)$ și $G_1(k)$ sunt prima și a doua jumătate a ieșirii din PRG

- Construim $F : \mathcal{K} \times \{0,1\} \rightarrow \mathcal{K}$ *PRF*:

$$F_k(0) = G_0(k), F_k(1) = G_1(k)$$

$F_k(0)$ întoarce prima jumătate a secvenței pseudoaleatoare $G(k)$

$F_k(1)$ întoarce a doua jumătate a secvenței pseudoaleatoare $G(k)$

- Întrebare: De ce este F sigură?
- Răspuns: $G(k)$ este *indistinctibilă* față de o secvență aleatoare de lungime $2n$

$PRG \Rightarrow PRF$

- Considerăm $G : \mathcal{K} \rightarrow \mathcal{K}^2$ PRG cu $|\mathcal{K}| = n$:

$$G(k) = (G_0(k), G_1(k))$$

  $G_0(k)$ și $G_1(k)$ sunt prima și a doua jumătate a ieșirii din PRG

- Construim $F : \mathcal{K} \times \{0,1\} \rightarrow \mathcal{K}$ *PRF*:

$$F_k(0) = G_0(k), F_k(1) = G_1(k)$$

  $F_k(0)$ întoarce prima jumătate a secvenței pseudoaleatoare $G(k)$

  $F_k(1)$ întoarce a doua jumătate a secvenței pseudoaleatoare $G(k)$

- Întrebare: De ce este F sigură?
- Răspuns: $G(k)$ este *indistinctibilă* față de o secvență aleatoare de lungime $2n$
  $\Rightarrow G_0(k)$ și $G_1(k)$ sunt *indistinctibile* față de o secvență aleatoare de lungime n

$PRG \Rightarrow PRF$

- Considerăm $G : \mathcal{K} \to \mathcal{K}^2$ PRG cu $|\mathcal{K}| = n$:

$$G(k) = (G_0(k), G_1(k))$$

  $G_0(k)$ și $G_1(k)$ sunt prima și a doua jumătate a ieșirii din PRG

- Construim $F : \mathcal{K} \times \{0,1\} \to \mathcal{K}$ *PRF*:

$$F_k(0) = G_0(k), F_k(1) = G_1(k)$$

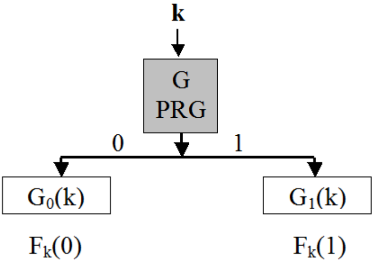
  $F_k(0)$ întoarce prima jumătate a secvenței pseudoaleatoare $G(k)$

  $F_k(1)$ întoarce a doua jumătate a secvenței pseudoaleatoare $G(k)$

- Întrebare: De ce este F sigură?
- Răspuns: $G(k)$ este *indistinctibilă* față de o secvență aleatoare de lungime $2n$
  $\Rightarrow G_0(k)$ și $G_1(k)$ sunt *indistinctibile* față de o secvență aleatoare de lungime n
  $\Rightarrow$ pentru $k \leftarrow^R \{0,1\}$, $F_k(\cdot)$ este *indistinctibilă* față de o funcție aleatoare.

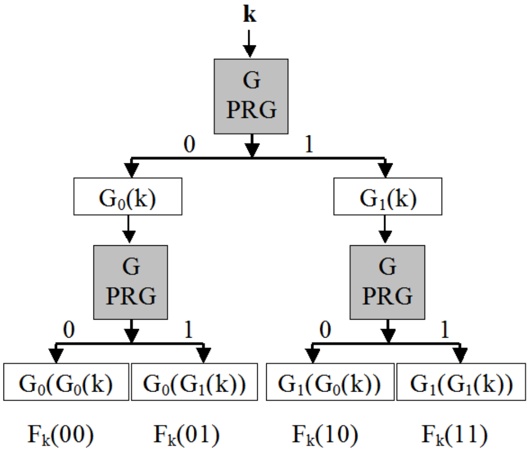
$PRG \Rightarrow PRF$

- Construcția pentru un singur bit de intrare...



$PRG \Rightarrow PRF$

- ...se poate generaliza pentru un număr oarecare de biți



$PRG \Rightarrow PRF$

- Construcția poate fi reprezentată ca un arbore binar cu cheia k rădăcină;

$PRG \Rightarrow PRF$

- Construcția poate fi reprezentată ca un arbore binar cu cheia k rădăcină;
- Pentru un nod de valoare k', copilul stâng ia valoarea $G_0(k')$ și copilul drept ia valoare $G_1(k')$;

$PRG \Rightarrow PRF$

- Construcția poate fi reprezentată ca un arbore binar cu cheia k rădăcină;
- Pentru un nod de valoare k', copilul stâng ia valoarea $G_0(k')$ și copilul drept ia valoare $G_1(k')$;
- Valoarea funcției $F_k(x) = F_k(x_0, \ldots, x_{n-1})$ este obținută prin parcurgerea arborelui în funcție de x ;

$PRG \Rightarrow PRF$

- Construcția poate fi reprezentată ca un arbore binar cu cheia k rădăcină;
- Pentru un nod de valoare k', copilul stâng ia valoarea $G_0(k')$ și copilul drept ia valoare $G_1(k')$;
- Valoarea funcției $F_k(x) = F_k(x_0, \ldots, x_{n-1})$ este obținută prin parcurgerea arborelui în funcție de x ;
- Adâncimea arborelui este *liniară* în n (n);

$PRG \Rightarrow PRF$

- Construcția poate fi reprezentată ca un arbore binar cu cheia k rădăcină;
- Pentru un nod de valoare k', copilul stâng ia valoarea $G_0(k')$ și copilul drept ia valoare $G_1(k')$;
- Valoarea funcției $F_k(x) = F_k(x_0, \ldots, x_{n-1})$ este obținută prin parcurgerea arborelui în funcție de x ;
- Adâncimea arborelui este *liniară* în n (n);
- Dimensiunea arborelui este *exponențială* în n (2^n);

PRG $\Rightarrow$ *PRF*

- Construcția poate fi reprezentată ca un arbore binar cu cheia k rădăcină;
- Pentru un nod de valoare k', copilul stâng ia valoarea $G_0(k')$ și copilul drept ia valoare $G_1(k')$;
- Valoarea funcției $F_k(x) = F_k(x_0, \ldots, x_{n-1})$ este obținută prin parcurgerea arborelui în funcție de x ;
- Adâncimea arborelui este *liniară* în n (n);
- Dimensiunea arborelui este *exponențială* în n (2^n);
- NU se utilizează în practică din cauza performanței scăzute.

Construcții

- **PRF** $\Rightarrow$ **PRG** $\surd$
  Pornind de la PRF se poate construi PRG
- **PRG** $\Rightarrow$ **PRF** $\surd$
  Pornind de la PRG se poate construi PRF
- **PRP** $\Rightarrow$ **PRF** $\surd$
  Pornind de la PRP se poate construi PRF
- **PRF** $\Rightarrow$ **PRP**
  Pornind de la PRF se poate construi PRP

$PRF \Rightarrow PRP$

Teorema (Luby-Rackoff '85)

Dacă $F : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ *este PRF, se poate construi* $F' : \mathcal{K} \times \{0,1\}^2 \rightarrow \{0,1\}^2$ *PRP.*

$PRF \Rightarrow PRP$

Teorema (Luby-Rackoff '85)

Dacă $F : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ este PRF, se poate construi $F' : \mathcal{K} \times \{0,1\}^2 \rightarrow \{0,1\}^2$ PRP.

- Construcția folosește runde **Feistel**, pe care le vom prezenta într-un curs ulterior.

Moduri de utilizare

- Să continuam cu ceva mai practic...

Moduri de utilizare

- Să continuam cu ceva mai practic...
- Întrebare: Ce se întâmplă dacă lungimea mesajului clar este **mai mică** decât dimensiunea unui bloc?

Moduri de utilizare

- Să continuam cu ceva mai practic...
- Întrebare: Ce se întâmplă dacă lungimea mesajului clar este **mai mică** decât dimensiunea unui bloc?
- Răspuns: Se completează cu biți: $\mathbf{1\ 0 \ldots 0}$;

Moduri de utilizare

- Să continuam cu ceva mai practic...
- Întrebare: Ce se întâmplă dacă lungimea mesajului clar este **mai mică** decât dimensiunea unui bloc?
- Răspuns: Se completează cu biți: $\mathbf{1\ 0 \ldots 0}$;
- Întrebare: Ce se întâmplă dacă lungimea mesajului clar este **mai mare** decât lungimea unui bloc?

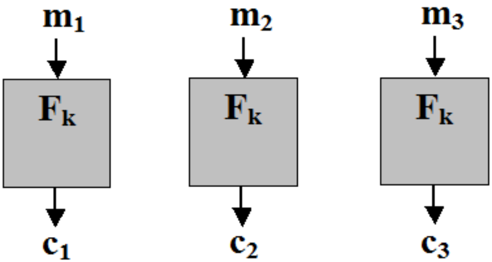
Moduri de utilizare

- Să continuam cu ceva mai practic...
- Întrebare: Ce se întâmplă dacă lungimea mesajului clar este **mai mică** decât dimensiunea unui bloc?
- Răspuns: Se completează cu biți: $\mathbf{1\ 0 \ldots 0}$;
- Întrebare: Ce se întâmplă dacă lungimea mesajului clar este **mai mare** decât lungimea unui bloc?
- Răspuns: Se utilizează un **mod de operare** (ECB, CBC, OFB, CTR);

Moduri de utilizare

- Să continuam cu ceva mai practic...
- Întrebare: Ce se întâmplă dacă lungimea mesajului clar este **mai mică** decât dimensiunea unui bloc?
- Răspuns: Se completează cu biți: $\mathbf{1\ 0 \ldots 0}$;
- Întrebare: Ce se întâmplă dacă lungimea mesajului clar este **mai mare** decât lungimea unui bloc?
- Răspuns: Se utilizează un **mod de operare** (ECB, CBC, OFB, CTR);
- Notăm cu F_k un sistem de criptare bloc (i.e. PRP) cu cheia k fixată.

Modul ECB (Electronic Code Book)



Modul ECB (Electronic Code Book)

- Pare modul cel mai **natural** de a cripta mai multe blocuri;

Modul ECB (Electronic Code Book)

- Pare modul cel mai **natural** de a cripta mai multe blocuri;
- Pentru decriptare, F_k trebuie să fie **inversabliă**;

Modul ECB (Electronic Code Book)

- Pare modul cel mai **natural** de a cripta mai multe blocuri;
- Pentru decriptare, F_k trebuie să fie **inversabliă**;
- Este **paralelizabil**;

Modul ECB (Electronic Code Book)

- Pare modul cel mai **natural** de a cripta mai multe blocuri;
- Pentru decriptare, F_k trebuie să fie **inversabliă**;
- Este **paralelizabil**;
- Este **determinist**, deci este **nesigur**;

Modul ECB (Electronic Code Book)

- Pare modul cel mai **natural** de a cripta mai multe blocuri;
- Pentru decriptare, F_k trebuie să fie **inversabliă**;
- Este **paralelizabil**;
- Este **determinist**, deci este **nesigur**;
- Întrebare: Ce informații poate să ofere modul de criptare ECB unui adversar pasiv?

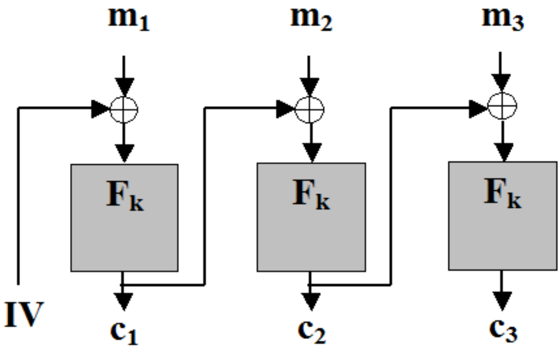
Modul ECB (Electronic Code Book)

- Pare modul cel mai **natural** de a cripta mai multe blocuri;
- Pentru decriptare, F_k trebuie să fie **inversabliă**;
- Este **paralelizabil**;
- Este **determinist**, deci este **nesigur**;
- Întrebare: Ce informații poate să ofere modul de criptare ECB unui adversar pasiv?
- Răspuns: Un adversar pasiv detectează repetarea unui bloc de text clar pentru că se repetă blocul criptat corespunzător;

Modul ECB (Electronic Code Book)

- Pare modul cel mai **natural** de a cripta mai multe blocuri;
- Pentru decriptare, F_k trebuie să fie **inversabliă**;
- Este **paralelizabil**;
- Este **determinist**, deci este **nesigur**;
- Întrebare: Ce informații poate să ofere modul de criptare ECB unui adversar pasiv?
- Răspuns: Un adversar pasiv detectează repetarea unui bloc de text clar pentru că se repetă blocul criptat corespunzător;
- Modul ECB NU trebuie utilizat în practică!

Modul CBC (Cipher Block Chaining)



Modul CBC (Cipher Block Chaining)

- IV este o ales în mod aleator la criptare;

Modul CBC (Cipher Block Chaining)

- IV este o ales în mod aleator la criptare;
- IV se transmite în clar pentru ca este necesar la decriptare;

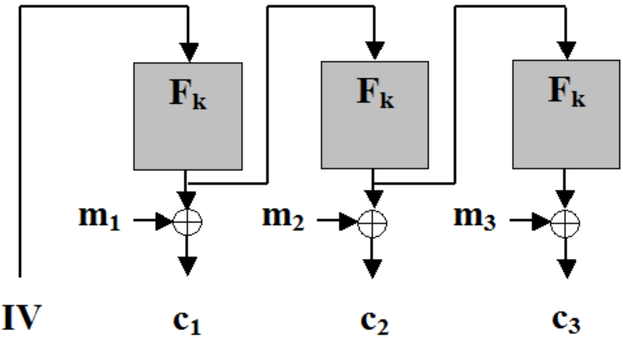
Modul CBC (Cipher Block Chaining)

- IV este o ales în mod aleator la criptare;
- IV se transmite în clar pentru ca este necesar la decriptare;
- Pentru decriptare, F_k trebuie să fie **inversablіă**;

Modul CBC (Cipher Block Chaining)

- IV este o ales în mod aleator la criptare;
- IV se transmite în clar pentru ca este necesar la decriptare;
- Pentru decriptare, F_k trebuie să fie **inversabliă**;
- Este **secvențial**, un dezavantaj major dacă se poate utiliza procesarea paralelă.

Modul OFB (Output FeedBack)



Modul OFB (Output FeedBack)

- Generează o secvență pseudoaleatoare care se XOR-ează mesajului clar;

Modul OFB (Output FeedBack)

- Generează o secvență pseudoaleatoare care se XOR-ează mesajului clar;
- *IV* este o ales în mod aleator la criptare;

Modul OFB (Output FeedBack)

- Generează o secvență pseudoaleatoare care se XOR-ează mesajului clar;
- *IV* este o ales în mod aleator la criptare;
- *IV* se transmite în clar pentru ca este necesar la decriptare;

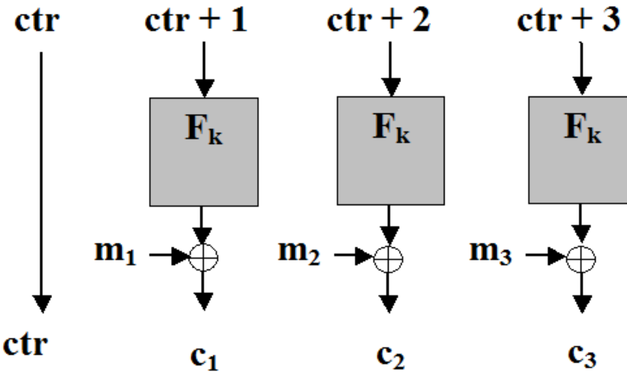
Modul OFB (Output FeedBack)

- Generează o secvență pseudoaleatoare care se XOR-ează mesajului clar;
- IV este o ales în mod aleator la criptare;
- IV se transmite în clar pentru ca este necesar la decriptare;
- F_k nu trebuie neapărat să fie inversabliă;

Modul OFB (Output FeedBack)

- Generează o secvență pseudoaleatoare care se XOR-ează mesajului clar;
- IV este o ales în mod aleator la criptare;
- IV se transmite în clar pentru ca este necesar la decriptare;
- F_k nu trebuie neapărat să fie inversabliă;
- Este **secvențial**, însă secvența pseudoaleatoare poate fi pre-procesată anterior decriptării.

Modul CTR (Counter)



Modul CTR (Counter)

- Generează o secvență pseudoaleatoare care se XOR-ează mesajului clar;

Modul CTR (Counter)

- Generează o secvență pseudoaleatoare care se XOR-ează mesajului clar;
- *ctr* este o ales în mod aleator la criptare;

Modul CTR (Counter)

- Generează o secvență pseudoaleatoare care se XOR-ează mesajului clar;
- *ctr* este o ales în mod aleator la criptare;
- *ctr* se transmite în clar pentru ca este necesar la decriptare;

Modul CTR (Counter)

- Generează o secvență pseudoaleatoare care se XOR-ează mesajului clar;
- *ctr* este o ales în mod aleator la criptare;
- *ctr* se transmite în clar pentru ca este necesar la decriptare;
- F_k nu trebuie neapărat să fie inversabliă;

Modul CTR (Counter)

- Generează o secvență pseudoaleatoare care se XOR-ează mesajului clar;
- *ctr* este o ales în mod aleator la criptare;
- *ctr* se transmite în clar pentru ca este necesar la decriptare;
- F_k nu trebuie neapărat să fie inversabliă;
- Este **paralelizabil**;

Modul CTR (Counter)

- Generează o secvență pseudoaleatoare care se XOR-ează mesajului clar;
- *ctr* este o ales în mod aleator la criptare;
- *ctr* se transmite în clar pentru ca este necesar la decriptare;
- F_k nu trebuie neapărat să fie inversabliă;
- Este **paralelizabil**;
- În plus, secvența pseudoaleatoare poate fi pre-procesată anterior decriptării.

Exemple

- DES (Data Encryption Standard):
  - propus de IBM
  - adoptat ca standard NIST în 1976
    (lungime cheie = 64 biți, lungime bloc = 64 biți)
  - spart prin căutare exhaustivă în 1997

Exemple

- DES (Data Encryption Standard):
  - propus de IBM
  - adoptat ca standard NIST în 1976
    (lungime cheie = 64 biți, lungime bloc = 64 biți)
  - spart prin căutare exhaustivă în 1997

- AES (Advanced Encryption Standard):
  - algoritmul *Rijndael* propus de J. Daemen și V.Rijman
  - adoptat ca standard NIST în 2001
    (lungime cheie = 128, 192, 256 biți, lungime bloc = 128 biți)

Important de reținut!

- Sisteme bloc vs. sisteme fluide
- Noțiunile de PRP, PRF
- Construcții specifice PRG, PRF, PRP
- Transpunerea sistemelor bloc în practică - moduri de operare: ECB, CBC, OFB, CTR