Criptografie și Securitate

- Prelegerea 9 -
Construcții practice pentru PRP

Adela Georgescu, Ruxandra F. Olimid

Facultatea de Matematică și Informatică
Universitatea din București

Cuprins

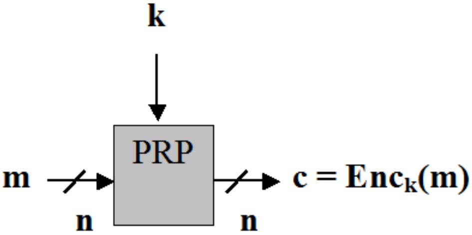
1. Sisteme bloc ca PRP

2. Rețele de substituție - permutare

3. Rețele Feistel

Sisteme bloc ca PRP

- Am văzut că sistemele de criptare bloc folosesc *PRP*;



Sisteme bloc ca PRP

- În criteriile de evaluare pentru adoptarea AES s-a menționat:

  The security provovided by an algorithm is the most important factor... Algorithms will be judged on the following factors...

  The extent to which the **algorithm output is indistinguishable from a random permutation on the input block**.

Sisteme bloc ca PRP

- În criteriile de evaluare pentru adoptarea AES s-a menționat:

  The security provovided by an algorithm is the most important factor... Algorithms will be judged on the following factors...

  The extent to which the **algorithm output is indistinguishable from a random permutation on the input block**.

- Întrebare: Cum se obțin *PRP* în practică?

Sisteme bloc ca PRP

- Ideal ar fi să se utilizeze o permutare aleatoare pe n biți;

Sisteme bloc ca PRP

- Ideal ar fi să se utilizeze o permutare aleatoare pe n biți;
- Ar fi necesari $\approx n \cdot 2^n$ biți pentru stocare;

Sisteme bloc ca PRP

- Ideal ar fi să se utilizeze o permutare aleatoare pe n biți;
- Ar fi necesari $\approx n \cdot 2^n$ biți pentru stocare;
- Pentru $n = 50$ este necesară o capacitate de stocare de $\approx 200\,TB$!

Sisteme bloc ca PRP

- Ideal ar fi să se utilizeze o permutare aleatoare pe n biți;
- Ar fi necesari $\approx n \cdot 2^n$ biți pentru stocare;
- Pentru $n = 50$ este necesară o capacitate de stocare de $\approx 200 TB$!
- Sistemele bloc au în general $n \geq 64$ sau $n \geq 128$ biți;

Sisteme bloc ca PRP

- Ideal ar fi să se utilizeze o permutare aleatoare pe n biți;
- Ar fi necesari $\approx n \cdot 2^n$ biți pentru stocare;
- Pentru $n = 50$ este necesară o capacitate de stocare de $\approx 200 TB$!
- Sistemele bloc au în general $n \geq 64$ sau $n \geq 128$ biți;
- În consecință, NU se poate utiliza o (singură) permutare aleatoare!

Claude Elwood Shannon (1916 - 2001)

- Shannon propune utilizarea mai multor permutări de dimensiune mai mică;
- Acesta este al doilea rezultat al lui Shannon pe care îl studiem, după *securitatea perfectă*.

[Wikipedia]

Rețele de substituție - permutare

- Se construiește permutarea F, pe baza mai multor permutări aleatoare f_i de dimensiune mai mică;

Rețele de substituție - permutare

- Se construiește permutarea F, pe baza mai multor permutări aleatoare f_i de dimensiune mai mică;
- Considerăm F pe 128 biți și 16 permutări aleatoare $f_1, \ldots, f_{16}$ pe câte 8 biți;

Rețele de substituție - permutare

- Se construiește permutarea F, pe baza mai multor permutări aleatoare f_i de dimensiune mai mică;
- Considerăm F pe 128 biți și 16 permutări aleatoare $f_1, \ldots, f_{16}$ pe câte 8 biți;
- Pentru $x = x_1 || \ldots || x_{16}$, $x \in \{0,1\}^{128}$ $x_i \in \{0,1\}^8$:

$$F_k(x) = f_1(x_1) || \ldots || f_{16}(x_{16})$$

Rețele de substituție - permutare

- Se construiește permutarea F, pe baza mai multor permutări aleatoare f_i de dimensiune mai mică;
- Considerăm F pe 128 biți și 16 permutări aleatoare $f_1, \ldots, f_{16}$ pe câte 8 biți;
- Pentru $x = x_1 || \ldots || x_{16}$, $x \in \{0,1\}^{128}$ $x_i \in \{0,1\}^8$:

$$F_k(x) = f_1(x_1) || \ldots || f_{16}(x_{16})$$

- Spunem că $\{f_i\}$ introduc confuzie în F.

Rețele de substituție - permutare

- Întrebare: Considerând capacitatea necesară de stocare, este F fesabilă?

Rețele de substituție - permutare

- Întrebare: Considerând capacitatea necesară de stocare, este F fesabilă?
- Răspuns: DA.

Rețele de substituție - permutare

- Întrebare: Considerând capacitatea necesară de stocare, este F fesabilă?
- Răspuns: DA.
- Fiecare f_i necesită $\approx 8 \cdot 2^8$ biți pentru stocare;

Rețele de substituție - permutare

- Întrebare: Considerând capacitatea necesară de stocare, este F fesabilă?
- Răspuns: DA.
- Fiecare f_i necesită $\approx 8 \cdot 2^8$ biți pentru stocare;
- Sunt 16 funcții $\{f_i\}$, deci în total F necesită pentru stocare $\approx 16 \cdot 8 \cdot 2^8 \approx 32KB$.

Rețele de substituție - permutare

- Întrebare: Este F PRP?

Rețele de substituție - permutare

- Întrebare: Este F PRP?
- Răspuns: NU.

Rețele de substituție - permutare

- Întrebare: Este F PRP?
- Răspuns: NU.
- Fie x și x' 2 intrări care diferă numai prin primul bit:

$$msb(x) \neq msb(x')$$

Rețele de substituție - permutare

- Întrebare: Este F PRP?
- Răspuns: NU.
- Fie x și x' 2 intrări care diferă numai prin primul bit:

$$msb(x) \neq msb(x')$$

- Atunci $F_k(x)$ și $F_k(x')$ diferă numai prin primul byte;

Rețele de substituție - permutare

- Întrebare: Este F PRP?
- Răspuns: NU.
- Fie x și x' 2 intrări care diferă numai prin primul bit:

$$msb(x) \neq msb(x')$$

- Atunci $F_k(x)$ și $F_k(x')$ diferă numai prin primul byte;
- În cazul unei permutări aleatoare, acest lucru nu se întâmplă.

Rețele de substituție - permutare

- F se transformă în PRP în 2 pași:

Rețele de substituție - permutare

- F se transformă în PRP în 2 pași:
  - **Pasul 1**: se introduce difuzie prin amestecarea (permutarea) biților de ieșire;

Rețele de substituție - permutare

- F se transformă în PRP în 2 pași:
  - **Pasul 1**: se introduce difuzie prin amestecarea (permutarea) biților de ieșire;
  - **Pasul 2**: se repetă o rundă (care presupune *confuzie* și *difuzie*) de mai multe ori;

Rețele de substituție - permutare

- F se transformă în PRP în 2 pași:
  - **Pasul 1**: se introduce difuzie prin amestecarea (permutarea) biților de ieșire;
  - **Pasul 2**: se repetă o rundă (care presupune *confuzie* și *difuzie*) de mai multe ori;
- Repetarea *confuziei* și *difuziei* face ca o modificarea unui singur bit de intrare să fie propagată asupra tutoror biților de ieșire;

Rețele de substituție - permutare

- F se transformă în PRP în 2 pași:
  - **Pasul 1**: se introduce difuzie prin amestecarea (permutarea) biților de ieșire;
  - **Pasul 2**: se repetă o rundă (care presupune *confuzie* și *difuzie*) de mai multe ori;
- Repetarea *confuziei* și *difuziei* face ca o modificarea unui singur bit de intrare să fie propagată asupra tutoror biților de ieșire;
- Un exemplu pentru 2 runde și intrarea x:

Rețele de substituție - permutare

- F se transformă în PRP în 2 pași:
  - **Pasul 1**: se introduce difuzie prin amestecarea (permutarea) biților de ieșire;
  - **Pasul 2**: se repetă o rundă (care presupune *confuzie* și *difuzie*) de mai multe ori;
- Repetarea *confuziei* și *difuziei* face ca o modificarea unui singur bit de intrare să fie propagată asupra tutoror biților de ieșire;
- Un exemplu pentru 2 runde și intrarea x:
  - $x' = F_k(x)$;
  - x_1 se obține prin reordonarea biților din x';
  - $x'_1 = F_k(x_1)$;
  - x_2 se obține prin reordonarea biților din x'_1.

Rețele de substituție - permutare

- O rețea de substituție-permutare este o implementare a construcției anterioare de *confuzie-difuzie* în care funcțiile $\{f_i\}$ sunt **fixe** (i.e. nu depind de cheie);

Rețele de substituție - permutare

- O rețea de substituție-permutare este o implementare a construcției anterioare de *confuzie-difuzie* în care funcțiile $\{f_i\}$ sunt **fixe** (i.e. nu depind de cheie);
- $\{f_i\}$ se numesc S-boxes (Substitution-boxes);

Rețele de substituție - permutare

- O rețea de substituție-permutare este o implementare a construcției anterioare de *confuzie-difuzie* în care funcțiile $\{f_i\}$ sunt **fixe** (i.e. nu depind de cheie);
- $\{f_i\}$ se numesc S-boxes (Substitution-boxes);
- S-box-urile rămân în continuare permutări;

Rețele de substituție - permutare

- O rețea de substituție-permutare este o implementare a construcției anterioare de *confuzie-difuzie* în care funcțiile $\{f_i\}$ sunt **fixe** (i.e. nu depind de cheie);
- $\{f_i\}$ se numesc S-boxes (Substitution-boxes);
- S-box-urile rămân în continuare permutări;
- Cum nu mai depind de cheie, aceasta este utilizată în alt scop;

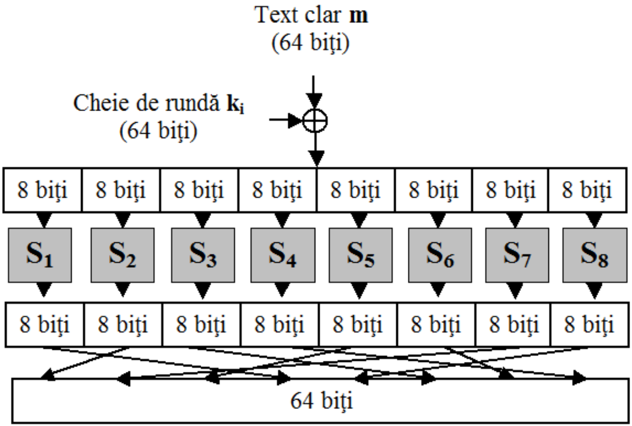
Rețele de substituție - permutare

- O rețea de substituție-permutare este o implementare a construcției anterioare de *confuzie-difuzie* în care funcțiile $\{f_i\}$ sunt **fixe** (i.e. nu depind de cheie);
- $\{f_i\}$ se numesc S-boxes (Substitution-boxes);
- S-box-urile rămân în continuare permutări;
- Cum nu mai depind de cheie, aceasta este utilizată în alt scop;
- Din cheie se obțin mai multe **chei de rundă** (*sub-chei*) în urma unui proces de derivare a cheilor (*key schedule*);

Rețele de substituție - permutare

- O rețea de substituție-permutare este o implementare a construcției anterioare de *confuzie-difuzie* în care funcțiile $\{f_i\}$ sunt **fixe** (i.e. nu depind de cheie);
- $\{f_i\}$ se numesc S-boxes (Substitution-boxes);
- S-box-urile rămân în continuare permutări;
- Cum nu mai depind de cheie, aceasta este utilizată în alt scop;
- Din cheie se obțin mai multe **chei de rundă** (*sub-chei*) în urma unui proces de derivare a cheilor (*key schedule*);
- Fiecare cheie de rundă este XOR-ată cu valorile intermediare din fiecare rundă.

Rețele de substituție - permutare



Rețele de substituție - permutare

- Există 2 principii de bază în proiectarea rețelelor de substituție - permutare:

Rețele de substituție - permutare

- Există 2 principii de bază în proiectarea rețelelor de substituție - permutare:
  - **Principiul 1**: Inversabilitatea S-box-urilor;

Rețele de substituție - permutare

- Există 2 principii de bază în proiectarea rețelelor de substituție - permutare:
  - **Principiul 1**: Inversabilitatea S-box-urilor;
  - **Principiul 2**: Efectul de avalanșă.

Principul 1: Inversabilitatea S-box

- O rețea de substituție-permutare **trebuie** să fie inversabilă;

Principul 1: Inversabilitatea S-box

- O rețea de substituție-permutare **trebuie** să fie inversabilă;
- Dacă fiecare rundă este inversabilă, atunci rețeaua este inversabilă;

Principul 1: Inversabilitatea S-box

- O rețea de substituție-permutare **trebuie** să fie inversabilă;
- Dacă fiecare rundă este inversabilă, atunci rețeaua este inversabilă;
- Într-o rundă:

Principul 1: Inversabilitatea S-box

- O rețea de substituție-permutare **trebuie** să fie inversabilă;
- Dacă fiecare rundă este inversabilă, atunci rețeaua este inversabilă;
- Într-o rundă:
  - XOR este inversabil;

Principul 1: Inversabilitatea S-box

- O rețea de substituție-permutare **trebuie** să fie inversabilă;
- Dacă fiecare rundă este inversabilă, atunci rețeaua este inversabilă;
- Într-o rundă:
    - XOR este inversabil;
    - Permutarea finală a biților de ieșire este inversabilă;

Principul 1: Inversabilitatea S-box

- O rețea de substituție-permutare **trebuie** să fie inversabilă;
- Dacă fiecare rundă este inversabilă, atunci rețeaua este inversabilă;
- Într-o rundă:
  - XOR este inversabil;
  - Permutarea finală a biților de ieșire este inversabilă;
- În concluzie, dacă toate S-box-urile sunt inversabile, atunci rețeaua este inversabilă;

Principul 1: Inversabilitatea S-box

- O rețea de substituție-permutare **trebuie** să fie inversabilă;
- Dacă fiecare rundă este inversabilă, atunci rețeaua este inversabilă;
- Într-o rundă:
  - XOR este inversabil;
  - Permutarea finală a biților de ieșire este inversabilă;
- În concluzie, dacă toate S-box-urile sunt inversabile, atunci rețeaua este inversabilă;
- Principiul 1 - necesitate funcțională (pentru decriptare).

Principul 2: Efectul de avalanșă

- Un singur bit modificat la intrare **trebuie** să afecteze toți biții din secvența de ieșire;

Principul 2: Efectul de avalanșă

- Un singur bit modificat la intrare **trebuie** să afecteze toți biții din secvența de ieșire;
- Efectul de avalanșă apare într-o rețea de substituție-permutare dacă:

Principul 2: Efectul de avalanșă

- Un singur bit modificat la intrare **trebuie** să afecteze toți biții din secvența de ieșire;
- Efectul de avalanșă apare într-o rețea de substituție-permutare dacă:
  1. S-box-urile sunt proiectate a.î. un singur bit schimbat la intrare să schimbe cel puțin 2 biți de la ieșire;

Principul 2: Efectul de avalanșă

- Un singur bit modificat la intrare **trebuie** să afecteze toți biții din secvența de ieșire;
- Efectul de avalanșă apare într-o rețea de substituție-permutare dacă:
  1. S-box-urile sunt proiectate a.î. un singur bit schimbat la intrare să schimbe cel puțin 2 biți de la ieșire;
  2. Permutarea este proiectată a.î. biții de la ieșirea unui S-box să fie împărțiți între intrările în S-box-uri diferite la runda următoare.

Principul 2: Efectul de avalanșă

- Un singur bit modificat la intrare **trebuie** să afecteze toți biții din secvența de ieșire;
- Efectul de avalanșă apare într-o rețea de substituție-permutare dacă:
  1. S-box-urile sunt proiectate a.î. un singur bit schimbat la intrare să schimbe cel puțin 2 biți de la ieșire;
  2. Permutarea este proiectată a.î. biții de la ieșirea unui S-box să fie împărțiți între intrările în S-box-uri diferite la runda următoare.
- Principiul 2 - necesitate de securitate.

Principul 2: Efectul de avalanșă

- Întrebare: De ce sunt suficiente cele 2 proprietăți pentru obținerea efectului de avalanșă?

Principul 2: Efectul de avalanșă

- Întrebare: De ce sunt suficiente cele 2 proprietăți pentru obținerea efectului de avalanșă?
- Presupunem 2 intrări care diferă printr-un singur bit;

Principul 2: Efectul de avalanșă

- Întrebare: De ce sunt suficiente cele 2 proprietăți pentru obținerea efectului de avalanșă?
- Presupunem 2 intrări care diferă printr-un singur bit;
- După Runda 1, ieșirile diferă prin 2 biți (*Proprietatea 1*);

Principul 2: Efectul de avalanșă

- Întrabare: De ce sunt suficiente cele 2 proprietăți pentru obținerea efectului de avalanșă?
- Presupunem 2 intrări care diferă printr-un singur bit;
- După Runda 1, ieșirile diferă prin 2 biți (*Proprietatea 1*);
- La intrarea în Runda 2, biții care diferă sunt intrări în S-box-uri diferite (*Proprietatea 2*);

Principul 2: Efectul de avalanșă

- Întrabare: De ce sunt suficiente cele 2 proprietăți pentru obținerea efectului de avalanșă?
- Presupunem 2 intrări care diferă printr-un singur bit;
- După Runda 1, ieșirile diferă prin 2 biți (*Proprietatea 1*);
- La intrarea în Runda 2, biții care diferă sunt intrări în S-box-uri diferite (*Proprietatea 2*);
- După Runda 2, ieșirile diferă prin 4 biți (*Proprietatea 1*);

Principul 2: Efectul de avalanșă

- Întrabare: De ce sunt suficiente cele 2 proprietăți pentru obținerea efectului de avalanșă?
- Presupunem 2 intrări care diferă printr-un singur bit;
- După Runda 1, ieșirile diferă prin 2 biți (*Proprietatea 1*);
- La intrarea în Runda 2, biții care diferă sunt intrări în S-box-uri diferite (*Proprietatea 2*);
- După Runda 2, ieșirile diferă prin 4 biți (*Proprietatea 1*);
- Urmând acest raționament, după Runda r, ieșirile diferă prin aproximativ 2^r biți;

Principul 2: Efectul de avalanșă

- Întrabare: De ce sunt suficiente cele 2 proprietăți pentru obținerea efectului de avalanșă?
- Presupunem 2 intrări care diferă printr-un singur bit;
- După Runda 1, ieșirile diferă prin 2 biți (*Proprietatea 1*);
- La intrarea în Runda 2, biții care diferă sunt intrări în S-box-uri diferite (*Proprietatea 2*);
- După Runda 2, ieșirile diferă prin 4 biți (*Proprietatea 1*);
- Urmând acest raționament, după Runda r, ieșirile diferă prin aproximativ 2^r biți;
- În concluzie, un sistem bloc cu intrarea de $n = 2^r$ biți obține efectul de avalanșă după cel puțin r runde.

Rețele Feistel

- Se aseamănă rețelelor de substituție-permutare în sensul că păstrează aceleași elementele componente: S-box, permutare, procesul de derivare a cheii, runde;

Rețele Feistel

- Se aseamănă rețelelor de substituție-permutare în sensul că păstrează aceleași elementele componente: S-box, permutare, procesul de derivare a cheii, runde;
- Se diferențiază de rețelele de substituție-permutare prin proiectarea de nivel înalt;

Rețele Feistel

- Se aseamănă rețelelor de substituție-permutare în sensul că păstrează aceleași elementele componente: S-box, permutare, procesul de derivare a cheii, runde;
- Se diferențiază de rețelele de substituție-permutare prin proiectarea de nivel înalt;
- Introduc avantajul major că S-box-urile NU trebuie să fie inversabile;

Rețele Feistel

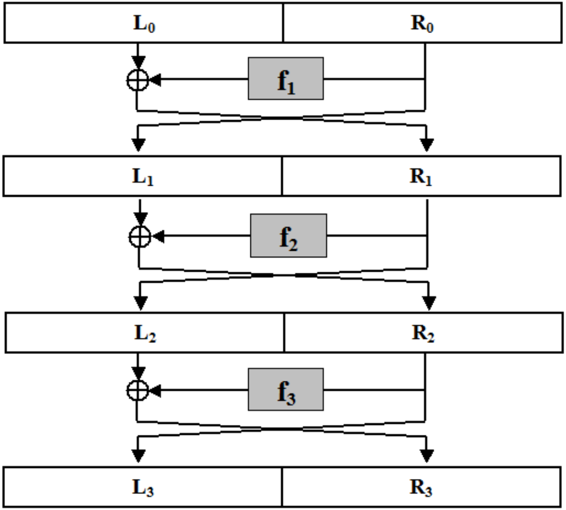
- Se aseamănă rețelelor de substituție-permutare în sensul că păstrează aceleași elementele componente: S-box, permutare, procesul de derivare a cheii, runde;
- Se diferențiază de rețelele de substituție-permutare prin proiectarea de nivel înalt;
- Introduc avantajul major că S-box-urile NU trebuie să fie inversabile;
- Permit așadar obținerea unei structuri *inversabile* folosind elemente *neinversabile*.

Horst Feistel (1915 - 1990)

[Wikipedia]

- Structurile simetrice utilizate în construcția sistemelor bloc poartă numele lui Feistel;
- Munca sa de cercetare la IBM a condus la sistemul de criptare Lucifer și mai târziu la DES.

Rețele Feistel



Rețele Feistel

- Intrarea în runda i se împarte în 2 jumătăți: L_{i-1} și R_{i-1} (i.e. *Left* și *Right*);

Rețele Feistel

- Intrarea în runda i se împarte în 2 jumătăți: L_{i-1} și R_{i-1} (i.e. *Left* și *Right*);
- Ieșirile din runda i sunt:

$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus f_i(R_{i-1})$$

Rețele Feistel

- Intrarea în runda i se împarte în 2 jumătăți: L_{i-1} și R_{i-1} (i.e. *Left* și *Right*);
- Ieșirile din runda i sunt:

$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus f_i(R_{i-1})$$

- Funcțiile f_i depind de cheia de rundă, derivând dintr-o funcție publică $\widehat{f_i}$:

$$f_i(R) = \widehat{f_i}(k_i, R)$$

Rețele Feistel

- Rețelele Feistel sunt inversabile indiferent dacă funcțiile f_i sunt inversabile sau nu;

Rețele Feistel

- Rețelele Feistel sunt inversabile indiferent dacă funcțiile f_i sunt inversabile sau nu;
- Fie (L_i, R_i) ieșirile din runda i;

Rețele Feistel

- Rețelele Feistel sunt inversabile indiferent dacă funcțiile f_i sunt inversabile sau nu;
- Fie (L_i, R_i) ieșirile din runda i;
- Intrările (L_{i-1}, R_{i-1}) în runda i sunt:

$$R_{i-1} = L_i$$

$$L_{i-1} = R_i \oplus f_i(R_{i-1})$$

Construcții

Am omis în cursurile anterioare ultima construcție:

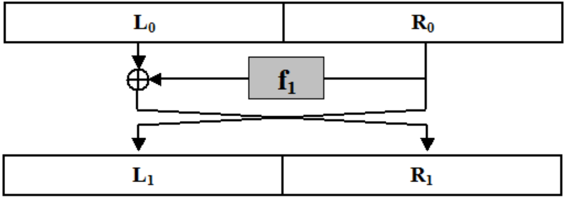
- **PRF** $\Rightarrow$ **PRG** $\surd$
  Pornind de la PRF se poate construi PRG
- **PRG** $\Rightarrow$ **PRF** $\surd$
  Pornind de la PRG se poate construi PRF
- **PRP** $\Rightarrow$ **PRF** $\surd$
  Pornind de la PRP se poate construi PRF
- **PRF** $\Rightarrow$ **PRP**
  Pornind de la PRF se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

$PRF \Rightarrow PRP$

- Revenim acum asupra construcției, când cunoaștem noțiunea de rundă Feistel;

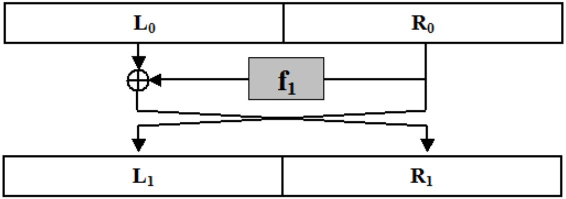
$PRF \Rightarrow PRP$

- Revenim acum asupra construcției, când cunoaștem noțiunea de rundă Feistel;
- Considerăm o singură rundă Feistel pentru care $f_1(x) = F_k(x)$, cu F PRF;



PRF $\Rightarrow$ PRP

- Revenim acum asupra construcției, când cunoaștem noțiunea de rundă Feistel;
- Considerăm o singură rundă Feistel pentru care $f_1(x) = F_k(x)$, cu F PRF;



- Întrebare: Este această construcție PRP?

$PRF \Rightarrow PRP$

- Răspuns: NU.

PRF $\Rightarrow$ *PRP*

- Răspuns: NU.
- Construcția este o permutare pe $\{0,1\}^n$,...

$PRF \Rightarrow PRP$

- Răspuns: NU.
- Construcția este o permutare pe $\{0, 1\}^n$,...
- ... este inversabilă,...

$PRF \Rightarrow PRP$

- Răspuns: NU.
- Construcția este o permutare pe $\{0,1\}^n$,...
- ... este inversabilă,...
- ... dar ieșirea nu este o secvență pseudoaleatoare: primii $n/2$ biți de ieșire sunt egali cu ultimii $n/2$ biți de intrare;

$PRF \Rightarrow PRP$

- Răspuns: NU.
- Construcția este o permutare pe $\{0,1\}^n$,...
- ... este inversabilă,...
- ... dar ieșirea nu este o secvență pseudoaleatoare: primii $n/2$ biți de ieșire sunt egali cu ultimii $n/2$ biți de intrare;
- Se mărește numărul de runde Feistel, păstrând $f_i(x) = F_{k_i}(x)$, unde k_i sunt chei independente;

$PRF \Rightarrow PRP$

- Răspuns: NU.
- Construcția este o permutare pe $\{0,1\}^n$,...
- ... este inversabilă,...
- ... dar ieșirea nu este o secvență pseudoaleatoare: primii $n/2$ biți de ieșire sunt egali cu ultimii $n/2$ biți de intrare;
- Se mărește numărul de runde Feistel, păstrând $f_i(x) = F_{k_i}(x)$, unde k_i sunt chei independente;
- Pentru $i = 3$ se obține PRP;

$PRF \Rightarrow PRP$

- Răspuns: NU.
- Construcția este o permutare pe $\{0, 1\}^n$,...
- ... este inversabilă,...
- ... dar ieșirea nu este o secvență pseudoaleatoare: primii $n/2$ biți de ieșire sunt egali cu ultimii $n/2$ biți de intrare;
- Se mărește numărul de runde Feistel, păstrând $f_i(x) = F_{k_i}(x)$, unde k_i sunt chei independente;
- Pentru $i = 3$ se obține PRP;
- Altfel spus, o rețea Feistel de 3 runde construită folosind 3 PRF $f_1(x) = F_{k_1}(x)$, $f_2(x) = F_{k_2}(x)$ și $f_3(x) = F_{k_3}(x)$ cu k_1, k_2 și k_3 independente este PRP.

Important de reținut!

- Rețele de substituție-permutare
- Rețele Feistel