Criptografie și Securitate

- Prelegerea 9.1 -
Data Encryption Standard - DES

Adela Georgescu, Ruxandra F. Olimid

Facultatea de Matematică și Informatică
Universitatea din București

Cuprins

DES - Data Encryption Standard

- 1970 - Horst Feistel proiectează Lucifer, o familie de cifruri bloc, la IBM, cu lungime cheie = 128 biți și lungime bloc = 128 biți;

DES - Data Encryption Standard

- 1970 - Horst Feistel proiectează Lucifer, o familie de cifruri bloc, la IBM, cu lungime cheie = 128 biți și lungime bloc = 128 biți;
- 1973 - NIST inițiază o cerere pentru propuneri în vederea standardizarii cifrurilor bloc în SUA;
  IBM trimite o variantă de Lucifer.

DES - Data Encryption Standard

- 1970 - Horst Feistel proiectează Lucifer, o familie de cifruri bloc, la IBM, cu lungime cheie = 128 biți și lungime bloc = 128 biți;
- 1973 - NIST inițiază o cerere pentru propuneri în vederea standardizarii cifrurilor bloc în SUA;
  IBM trimite o variantă de Lucifer.
- 1976 - NIST adoptă Lucifer modificat ca standard DES cu lungime cheie = 56 biți și lungime bloc = 64 biți.

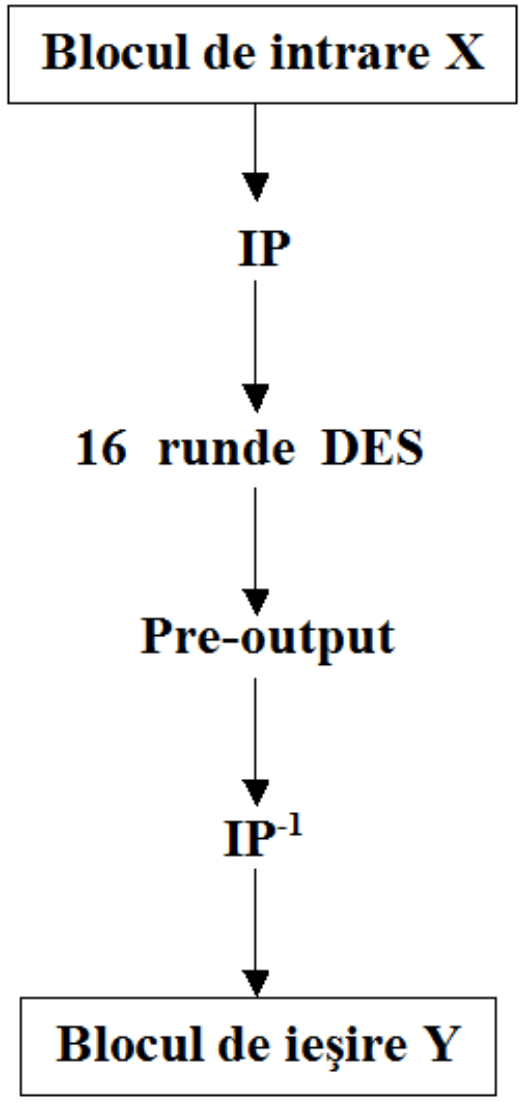
DES - Data Encryption Standard

- 1970 - Horst Feistel proiectează Lucifer, o familie de cifruri bloc, la IBM, cu lungime cheie = 128 biți și lungime bloc = 128 biți;
- 1973 - NIST inițiază o cerere pentru propuneri în vederea standardizarii cifrurilor bloc în SUA;
  IBM trimite o variantă de Lucifer.
- 1976 - NIST adoptă Lucifer modificat ca standard DES cu lungime cheie = 56 biți și lungime bloc = 64 biți.
- 1997 - DES este spart prin cautare exhaustivă (forță brută).

DES - Data Encryption Standard

- 1970 - Horst Feistel proiectează Lucifer, o familie de cifruri bloc, la IBM, cu lungime cheie = 128 biți și lungime bloc = 128 biți;
- 1973 - NIST inițiază o cerere pentru propuneri în vederea standardizarii cifrurilor bloc în SUA;
  IBM trimite o variantă de Lucifer.
- 1976 - NIST adoptă Lucifer modificat ca standard DES cu lungime cheie = 56 biți și lungime bloc = 64 biți.
- 1997 - DES este spart prin cautare exhaustivă (forță brută).
- 2001 - NIST adoptă Rinjdael ca noul standard AES în locul lui DES.

DES


Blocul de intrare X
IP
16 runde DES
Pre-output
IP-1
Blocul de ieşire Y

Descriere DES

- DES este o rețea de tip Feistel cu 16 runde și o cheie pe 56 biți;

Descriere DES

- DES este o rețea de tip Feistel cu 16 runde și o cheie pe 56 biți;
- Procesul de derivare a cheii (*key schedule*) obține o sub-cheie de rundă k_i pentru fiecare rundă pornind de la cheia master k;

Descriere DES

- DES este o rețea de tip Feistel cu 16 runde și o cheie pe 56 biți;
- Procesul de derivare a cheii (*key schedule*) obține o sub-cheie de rundă k_i pentru fiecare rundă pornind de la cheia master k;
- Funcțiile de rundă $f_i(R) = f(k_i, R)$ sunt derivate din aceeași funcție principală $\widehat{f}$ și nu sunt inversabile;

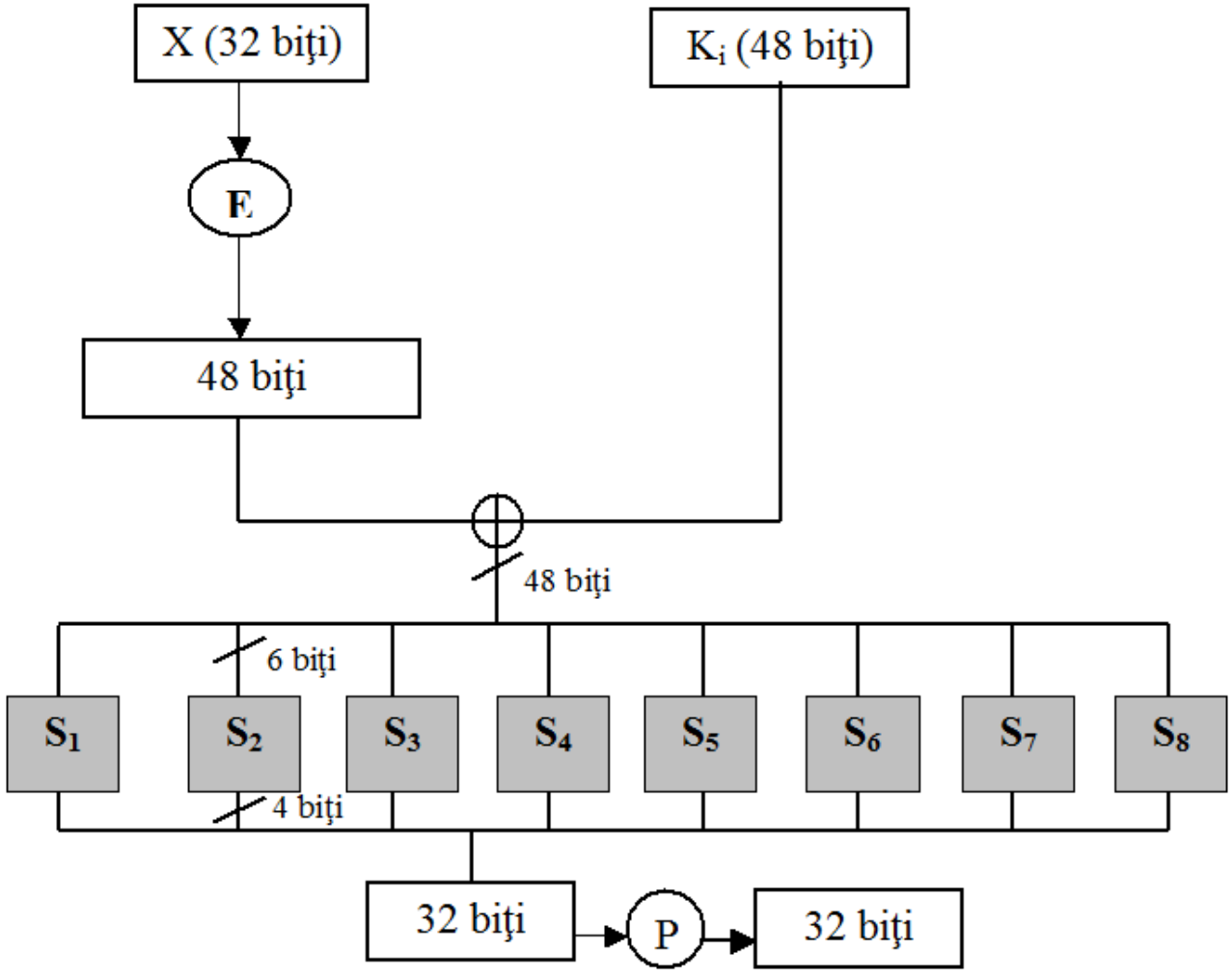
Descriere DES

- DES este o rețea de tip Feistel cu 16 runde și o cheie pe 56 biți;
- Procesul de derivare a cheii (*key schedule*) obține o sub-cheie de rundă k_i pentru fiecare rundă pornind de la cheia master k;
- Funcțiile de rundă $f_i(R) = f(k_i, R)$ sunt derivate din aceeași funcție principală $\widehat{f}$ și nu sunt inversabile;
- Fiecare sub-cheie k_i reprezintă permutarea a 48 biți din cheia master;

Descriere DES

- DES este o rețea de tip Feistel cu 16 runde și o cheie pe 56 biți;
- Procesul de derivare a cheii (*key schedule*) obține o sub-cheie de rundă k_i pentru fiecare rundă pornind de la cheia master k;
- Funcțiile de rundă $f_i(R) = f(k_i, R)$ sunt derivate din aceeași funcție principală $\widehat{f}$ și nu sunt inversabile;
- Fiecare sub-cheie k_i reprezintă permutarea a 48 biți din cheia master;
- Întreaga procedură de obținere a sub-cheilor de rundă este fixă și cunoscută, singurul secret este cheia master .

Funcția $f(k_i, R)$



Funcția $f(k_i, R)$

- Funcția $\widehat{f}$ este, în esență, o rețea de substituție-permutare cu doar o rundă.

Funcția $f(k_i, R)$

- Funcția $\widehat{f}$ este, în esență, o rețea de substituție-permutare cu doar o rundă.
- Pentru calculul $f(k_i, R)$ se procedează astfel:

Funcția $f(k_i, R)$

- Funcția $\widehat{f}$ este, în esență, o rețea de substituție-permutare cu doar o rundă.
- Pentru calculul $f(k_i, R)$ se procedează astfel:
  1. R este expandat la un bloc R' de 48 biți cu ajutorul unei funcții de expandare $R' = E(R)$.

Funcția $f(k_i, R)$

- Funcția $\widehat{f}$ este, în esență, o rețea de substituție-permutare cu doar o rundă.
- Pentru calculul $f(k_i, R)$ se procedează astfel:
  1. R este expandat la un bloc R' de 48 biți cu ajutorul unei funcții de expandare $R' = E(R)$.
  2. R' este XOR-at cu k_i iar valoarea rezultată este împărțită în 8 blocuri de câte 6 biți.

Funcția $f(k_i, R)$

- Funcția $\widehat{f}$ este, în esență, o rețea de substituție-permutare cu doar o rundă.
- Pentru calculul $f(k_i, R)$ se procedează astfel:
  1. R este expandat la un bloc R' de 48 biți cu ajutorul unei funcții de expandare $R' = E(R)$.
  2. R' este XOR-at cu k_i iar valoarea rezultată este împărțită în 8 blocuri de câte 6 biți.
  3. Fiecare bloc de 6 biți trece printr-un SBOX diferit rezultând o valoare pe 4 biți.

Funcția $f(k_i, R)$

- Funcția $\widehat{f}$ este, în esență, o rețea de substituție-permutare cu doar o rundă.

- Pentru calculul $f(k_i, R)$ se procedează astfel:
  1. R este expandat la un bloc R' de 48 biți cu ajutorul unei funcții de expandare $R' = E(R)$.
  2. R' este XOR-at cu k_i iar valoarea rezultată este împărțită în 8 blocuri de câte 6 biți.
  3. Fiecare bloc de 6 biți trece printr-un SBOX diferit rezultând o valoare pe 4 biți.
  4. Se concatenează blocurile rezultate și se aplică o permutare, rezultând în final un bloc de 32 biți.

Funcția $f(k_i, R)$

- Funcția $\widehat{f}$ este, în esență, o rețea de substituție-permutare cu doar o rundă.
- Pentru calculul $f(k_i, R)$ se procedează astfel:
  1. R este expandat la un bloc R' de 48 biți cu ajutorul unei funcții de expandare $R' = E(R)$.
  2. R' este XOR-at cu k_i iar valoarea rezultată este împărțită în 8 blocuri de câte 6 biți.
  3. Fiecare bloc de 6 biți trece printr-un SBOX diferit rezultând o valoare pe 4 biți.
  4. Se concatenează blocurile rezultate și se aplică o permutare, rezultând în final un bloc de 32 biți.
- **De remarcat:** Toată descrierea lui DES, inclusiv SBOX-urile și permutările sunt publice.

SBOX-urile din DES

- Formează o parte esențială din construcția DES;

SBOX-urile din DES

- Formează o parte esențială din construcția DES;
- DES devine mult mai vulnerabil la atacuri dacă SBOX-urile sunt modificate ușor sau dacă sunt alese aleator

SBOX-urile din DES

- Formează o parte esențială din construcția DES;
- DES devine mult mai vulnerabil la atacuri dacă SBOX-urile sunt modificate ușor sau dacă sunt alese aleator
- Primul și ultimul bit din cei 6 de la intrare sunt folosiți pentru a alege linia din tabel, iar biții 2-5 sunt folosiți pentru coloană; output-ul va consta din cei 4 biți aflați la intersecția liniei și coloanei alese.

S_5		Cei 4 biţi din mijloc															
		0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
Biţii din margine	**00**	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
	01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
	10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
	11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011

SBOX-urile din DES

- Formează o parte esențială din construcția DES;
- DES devine mult mai vulnerabil la atacuri dacă SBOX-urile sunt modificate ușor sau dacă sunt alese aleator
- Primul și ultimul bit din cei 6 de la intrare sunt folosiți pentru a alege linia din tabel, iar biții 2-5 sunt folosiți pentru coloană; output-ul va consta din cei 4 biți aflați la intersecția liniei și coloanei alese.

S_5		Cei 4 biţi din mijloc															
		0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
Biţii din margine	**00**	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
	01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
	10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
	11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011

- Modificarea **unui bit** de la intrare întotdeauna afectează cel puțin **doi biți** de la ieșire.

Efectul de avalanșă

- DES are un puternic efect de avalanșă generat de ultima proprietate menționată mai sus și de permutările folosite;

Efectul de avalanșă

- DES are un puternic efect de avalanșă generat de ultima proprietate menționată mai sus și de permutările folosite;
- Pentru a vedea aceasta, vom urmări diferența între valorile intermediare dintr-un calcul DES a două valori de intrare care diferă printr-un singur bit;

Efectul de avalanșă

- DES are un puternic efect de avalanșă generat de ultima proprietate menționată mai sus și de permutările folosite;
- Pentru a vedea aceasta, vom urmări diferența între valorile intermediare dintr-un calcul DES a două valori de intrare care diferă printr-un singur bit;
- Notăm cele două valori cu (L_0, R_0) si (L_0', R_0') unde $R_0 = R_0'$;

Efectul de avalanșă

- DES are un puternic efect de avalanșă generat de ultima proprietate menționată mai sus și de permutările folosite;
- Pentru a vedea aceasta, vom urmări diferența între valorile intermediare dintr-un calcul DES a două valori de intrare care diferă printr-un singur bit;
- Notăm cele două valori cu (L_0, R_0) si (L_0', R_0') unde $R_0 = R_0'$;
- După prima rundă, valorile (L_1, R_1) si (L_1', R_1') încă diferă printr-un singur bit, deși acum diferența e în partea dreaptă;

Efectul de avalanșă

- DES are un puternic efect de avalanșă generat de ultima proprietate menționată mai sus și de permutările folosite;
- Pentru a vedea aceasta, vom urmări diferența între valorile intermediare dintr-un calcul DES a două valori de intrare care diferă printr-un singur bit;
- Notăm cele două valori cu (L_0, R_0) si (L_0', R_0') unde $R_0 = R_0'$;
- După prima rundă, valorile (L_1, R_1) si (L_1', R_1') încă diferă printr-un singur bit, deși acum diferența e în partea dreaptă;
- În a doua rundă DES, R_1 si R_1' trec prin funcția $\hat{f}$;

Efectul de avalanșă

- Presupunând că bitul în care R_1 si R_1' diferă nu este duplicat în pasul de expandare, ele încă diferă printr-un bit înainte de aplicarea SBOX-ului;

Efectul de avalanșă

- Presupunând că bitul în care R_1 si R_1' diferă nu este duplicat în pasul de expandare, ele încă diferă printr-un bit înainte de aplicarea SBOX-ului;
- După SBOX, valorile intermediare diferă în cel puțin *doi* biți;

Efectul de avalanșă

- Presupunând că bitul în care R_1 si R_1' diferă nu este duplicat în pasul de expandare, ele încă diferă printr-un bit înainte de aplicarea SBOX-ului;
- După SBOX, valorile intermediare diferă în cel puțin *doi* biți;
- (L_2, R_2) și (L_2', R_2') diferă în *trei* biti: 1-bit diferență între L_2 și L_2' și 2-biți diferență între R_2 și R_2';

Efectul de avalanșă

- Presupunând că bitul în care R_1 si R_1' diferă nu este duplicat în pasul de expandare, ele încă diferă printr-un bit înainte de aplicarea SBOX-ului;
- După SBOX, valorile intermediare diferă în cel puțin *doi* biți;
- (L_2, R_2) și (L_2', R_2') diferă în *trei* biti: 1-bit diferență între L_2 și L_2' și 2-biți diferență între R_2 și R_2';
- Permutarea din $\hat{f}$ împrăștie diferența dintre R_2 și R_2' în diferite regiuni ale lor;

Efectul de avalanșă

- Presupunând că bitul în care R_1 si R_1' diferă nu este duplicat în pasul de expandare, ele încă diferă printr-un bit înainte de aplicarea SBOX-ului;
- După SBOX, valorile intermediare diferă în cel puțin *doi* biți;
- (L_2, R_2) și (L_2', R_2') diferă în *trei* biti: 1-bit diferență între L_2 și L_2' și 2-biți diferență între R_2 și R_2';
- Permutarea din $\hat{f}$ împrăștie diferența dintre R_2 și R_2' în diferite regiuni ale lor;
- La următoarea rundă fiecare bit care diferă va fi folosit ca intrare într-un SBOX diferit, rezultând o diferență de 4 biți între jumătățile drepte ale valorilor intermediare;

Efectul de avalanșă

- Presupunând că bitul în care R_1 si R_1' diferă nu este duplicat în pasul de expandare, ele încă diferă printr-un bit înainte de aplicarea SBOX-ului;
- După SBOX, valorile intermediare diferă în cel puțin *doi* biți;
- (L_2, R_2) și (L_2', R_2') diferă în *trei* biti: 1-bit diferență între L_2 și L_2' și 2-biți diferență între R_2 și R_2';
- Permutarea din $\hat{f}$ împrăștie diferența dintre R_2 și R_2' în diferite regiuni ale lor;
- La următoarea rundă fiecare bit care diferă va fi folosit ca intrare într-un SBOX diferit, rezultând o diferență de 4 biți între jumătățile drepte ale valorilor intermediare;
- Efectul este exponențial și după 7 runde toți cei 32 biți vor fi modificați.

Efectul de avalanșă

- După 8 runde toți biții din jumătatea stângă vor fi modificați;

Efectul de avalanșă

- După 8 runde toți biții din jumătatea stângă vor fi modificați;
- DES are 16 runde deci efectul de avalanșa este atins foarte repede;

Efectul de avalanșă

- După 8 runde toți biții din jumătatea stângă vor fi modificați;
- DES are 16 runde deci efectul de avalanșa este atins foarte repede;
- Deci DES aplicat pe două intrări similare întoarce ieșiri complet diferite și independente;

Efectul de avalanșă

- După 8 runde toți biții din jumătatea stângă vor fi modificați;
- DES are 16 runde deci efectul de avalanșa este atins foarte repede;
- Deci DES aplicat pe două intrări similare întoarce ieșiri complet diferite și independente;
- Efectul se datorează și permutărilor alese cu grijă (este verificat faptul că permutări *aleatoare* nu oferă același efect puternic de avalanșă).

Atacuri pe DES cu număr redus de runde

- Pentru a înțelege securitatea DES vom studia mai întâi comportametul lui DES pe un număr redus de runde (maxim 3);

Atacuri pe DES cu număr redus de runde

- Pentru a înțelege securitatea DES vom studia mai întâi comportametul lui DES pe un număr redus de runde (maxim 3);
- Sigur, DES cu 3 runde nu este o funcție pseudoleatoare pentru că efectul de avalanșă încă nu e complet;

Atacuri pe DES cu număr redus de runde

- Pentru a înțelege securitatea DES vom studia mai întâi comportametul lui DES pe un număr redus de runde (maxim 3);
- Sigur, DES cu 3 runde nu este o funcție pseudoleatoare pentru că efectul de avalanșă încă nu e complet;
- Vom arăta câteva atacuri cu text clar care găsesc cheia k;

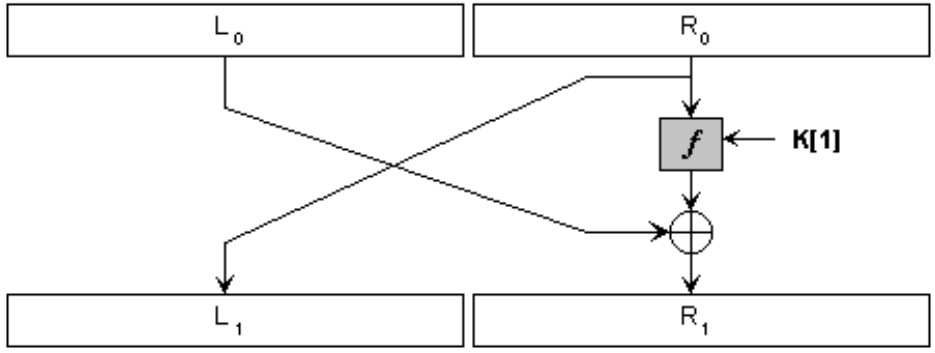
Atacuri pe DES cu număr redus de runde

- Pentru a înțelege securitatea DES vom studia mai întâi comportametul lui DES pe un număr redus de runde (maxim 3);
- Sigur, DES cu 3 runde nu este o funcție pseudoleatoare pentru că efectul de avalanșă încă nu e complet;
- Vom arăta câteva atacuri cu text clar care găsesc cheia k;
- Adversarul are deci acces la perechi de forma $\{(x_i, y_i)\}$ cu $y_i = DES_k(x_i)$;

Atacuri pe DES cu număr redus de runde

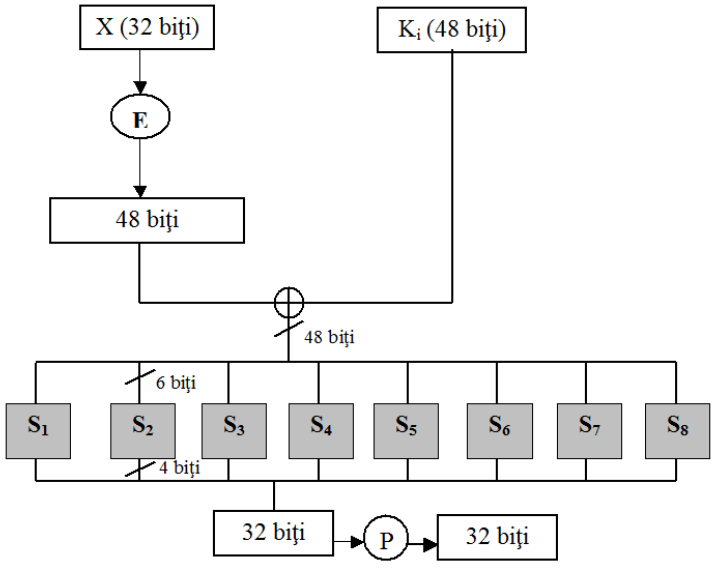
- Pentru a înțelege securitatea DES vom studia mai întâi comportametul lui DES pe un număr redus de runde (maxim 3);
- Sigur, DES cu 3 runde nu este o funcție pseudoleatoare pentru că efectul de avalanșă încă nu e complet;
- Vom arăta câteva atacuri cu text clar care găsesc cheia k;
- Adversarul are deci acces la perechi de forma $\{(x_i, y_i)\}$ cu $y_i = DES_k(x_i)$;
- În descrierea atacurilor, ne vom concentra asupra unei singure perechi (x, y).

DES cu o singură rundă



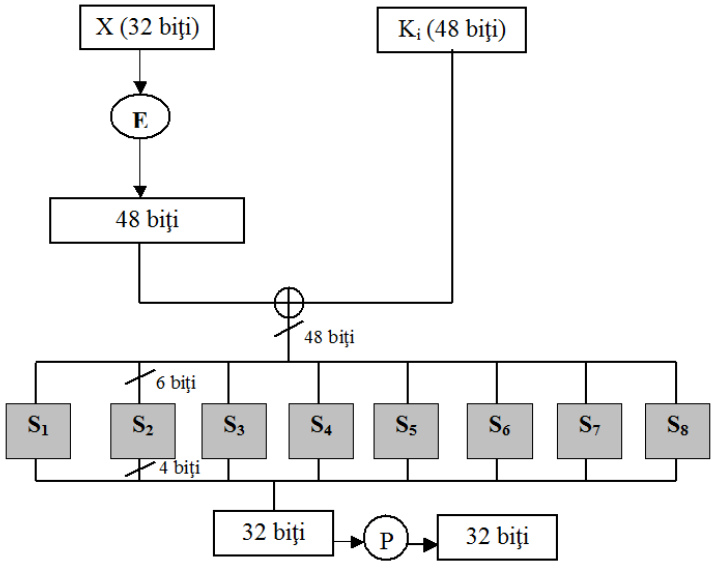
- Cunoaștem $x = (L_0, R_0)$ și $y = (L_1, R_1)$ unde
  - $L_1 = R_0$
  - $R_1 = L_0 \oplus f_1(R_0)$
- De asemenea, $f_1(R_0) = R_1 \oplus L_0$

DES cu o singură rundă



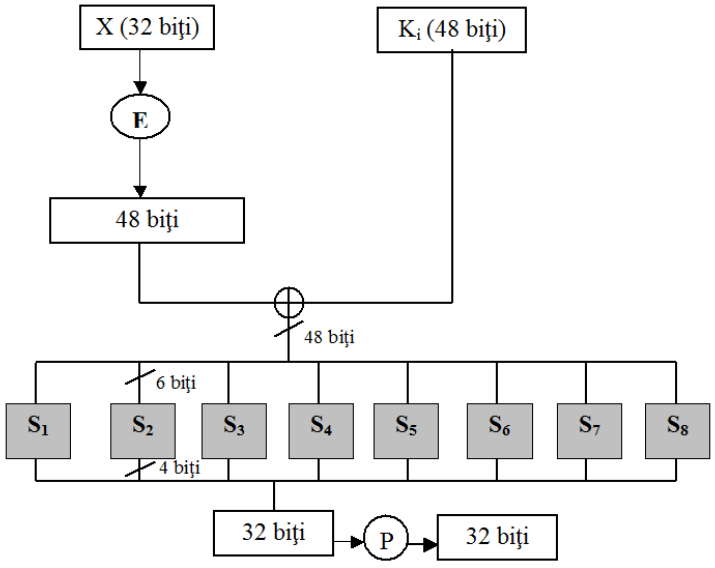
- Aplicând $P^{-1}(R_1 \oplus L_0)$ obținem o valoare intermediară care reprezintă ieșirea din cele 8 SBOX-uri;

DES cu o singură rundă



- Aplicând $P^{-1}(R_1 \oplus L_0)$ obținem o valoare intermediară care reprezintă ieșirea din cele 8 SBOX-uri;
- Intrarea în SBOX-uri este $E(R_0) \oplus k_1$; R_0 este cunoscut, la fel ieșirea din SBOX-uri;

DES cu o singură rundă



- Aplicând $P^{-1}(R_1 \oplus L_0)$ obținem o valoare intermediară care reprezintă ieșirea din cele 8 SBOX-uri;
- Intrarea în SBOX-uri este $E(R_0) \oplus k_1$; R_0 este cunoscut, la fel ieșirea din SBOX-uri;
- Pentru fiecare ieșire din SBOX, există 4 valori posibile ale porțiunii corespunzătoare de 6 biți din cheia k_1 care ar conduce la acea valoare.

DES cu una sau două runde

- Atac DES cu o singură rundă

DES cu una sau două runde

- Atac DES cu o singură rundă
  - Am redus numărul cheilor posibile de la 2^{48} la $4^8 = 2^{16}$;

DES cu una sau două runde

- Atac DES cu o singură rundă
  - Am redus numărul cheilor posibile de la 2^{48} la $4^8 = 2^{16}$;
  - Acum se pot verifica pe rând toate variantele și recupera complet cheia.

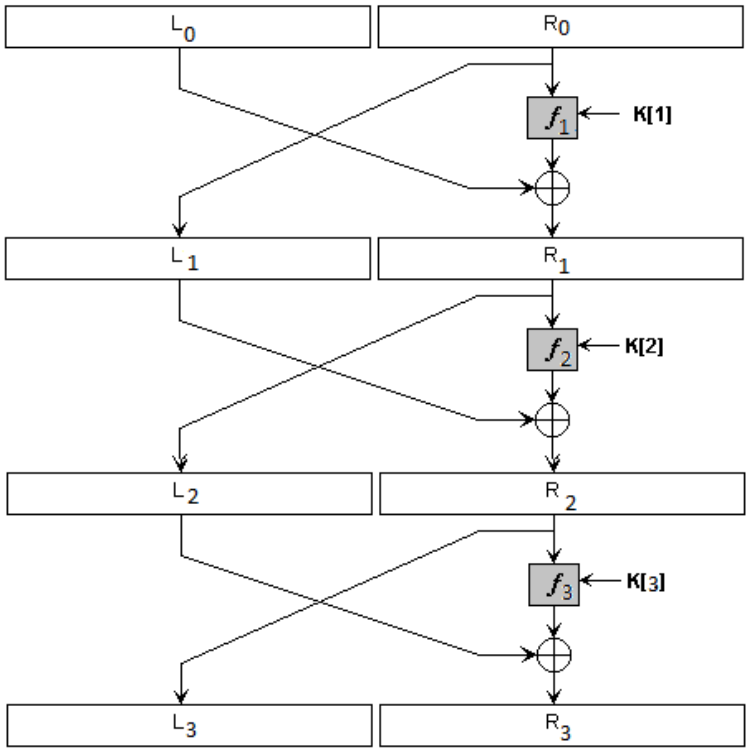
DES cu una sau două runde

- Atac DES cu o singură rundă
  - Am redus numărul cheilor posibile de la 2^{48} la $4^8 = 2^{16}$;
  - Acum se pot verifica pe rând toate variantele și recupera complet cheia.
- Atac DES cu două runde

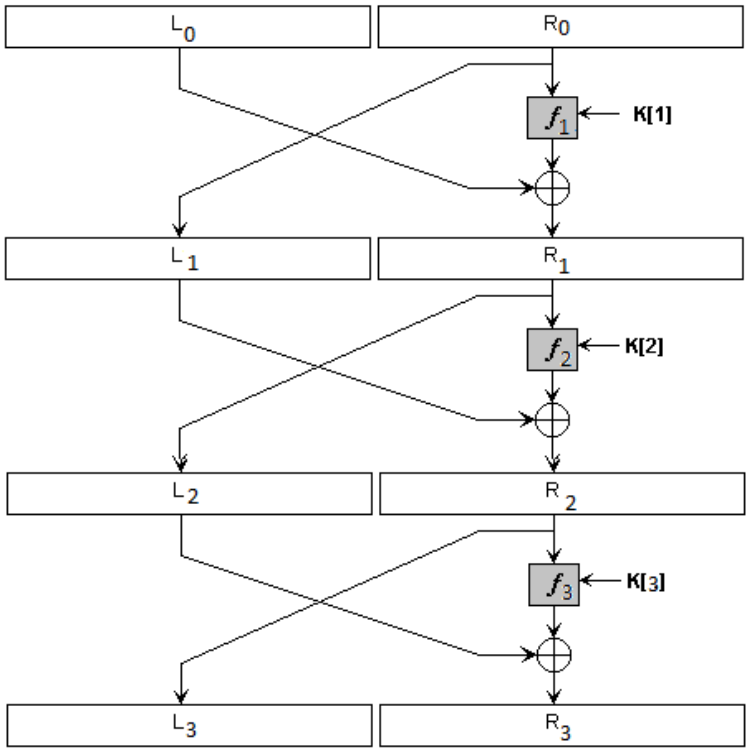
DES cu una sau două runde

- Atac DES cu o singură rundă
  - Am redus numărul cheilor posibile de la 2^{48} la $4^8 = 2^{16}$;
  - Acum se pot verifica pe rând toate variantele și recupera complet cheia.
- Atac DES cu două runde
  - Atacul găsește cheile k_1 și k_2 în timp $2 \cdot 2^{16}$ atunci când se cunoaște o pereche text clar/text criptat.

DES cu trei runde

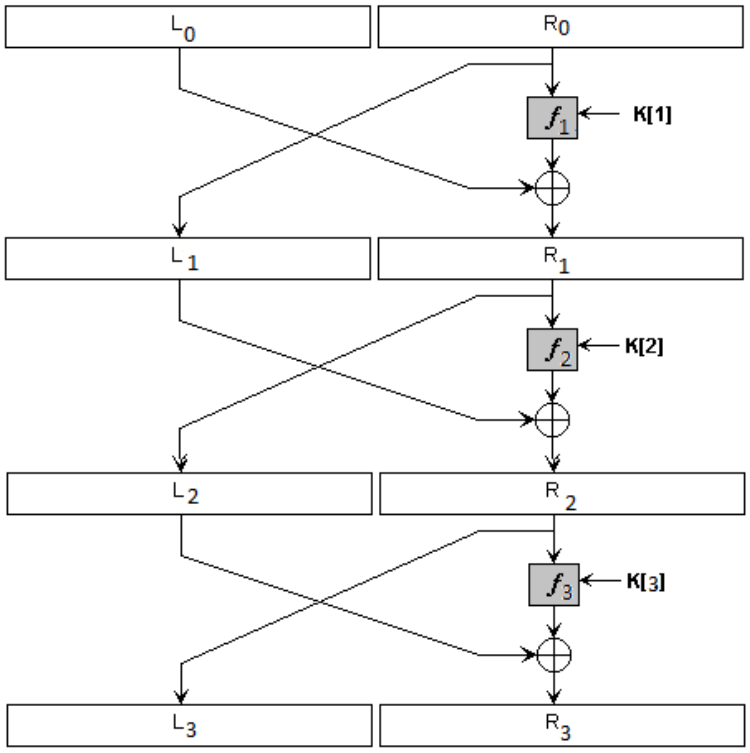


DES cu trei runde



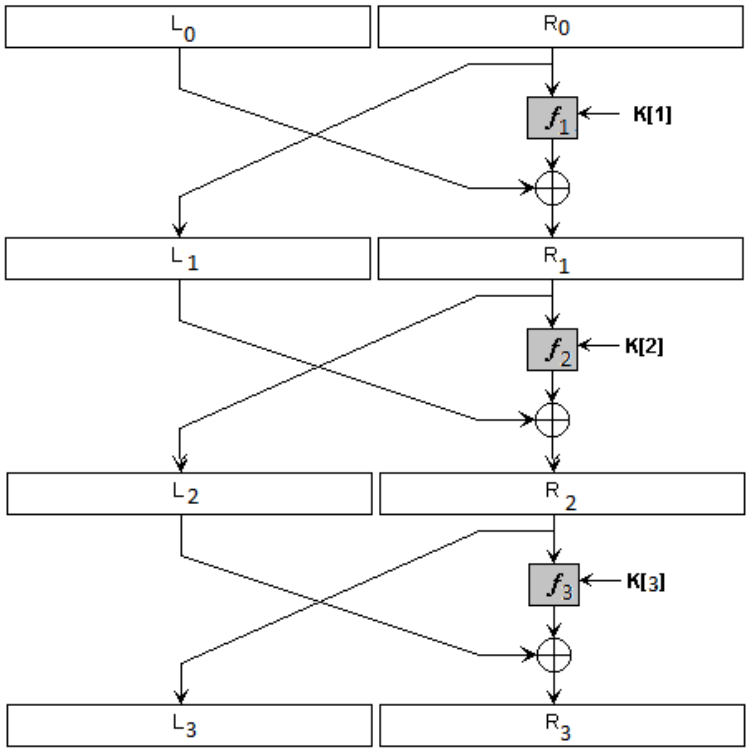
- Nu se cunosc R_1, L_2 și deci nici toate intrarile/ieșirile din fiecare funcție f;

DES cu trei runde



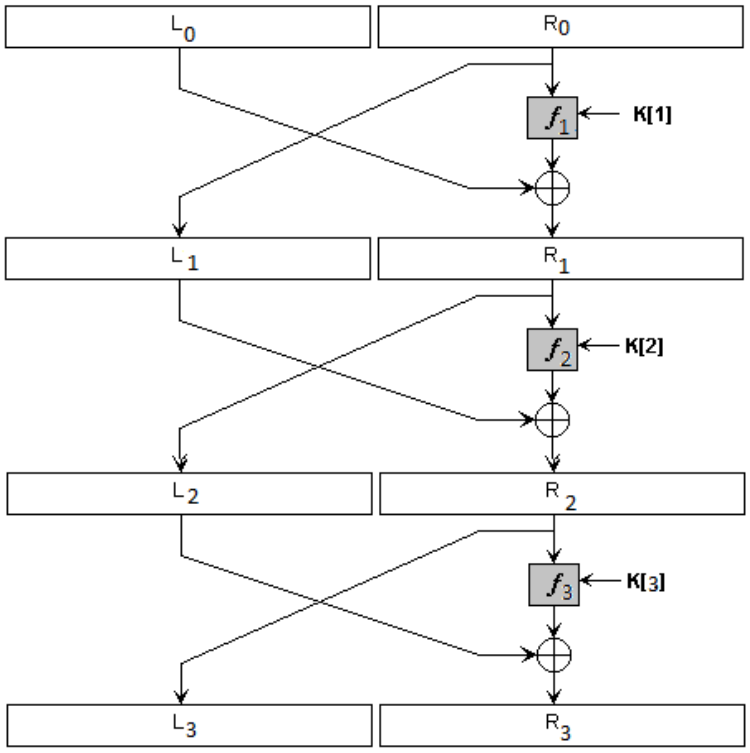
- Nu se cunosc R_1, L_2 și deci nici toate intrarile/ieșirile din fiecare funcție f;
- Atacul anterior nu funcționează; un nou atac va explora relațiile dintre ieșirile respectiv intrările în f_1 și f_3;

DES cu trei runde



- Nu se cunosc R_1, L_2 și deci nici toate intrarile/ieșirile din fiecare funcție f;
- Atacul anterior nu funcționează; un nou atac va explora relațiile dintre ieșirile respectiv intrările în f_1 și f_3;
- Atacul traversează spațiul cheilor pentru fiecare jumatăte a cheii master și, în timp $2 \cdot 2^{28}$ va produce câte 2^{12} variante pentru fiecare jumătate a cheii;

DES cu trei runde



- Nu se cunosc R_1, L_2 și deci nici toate intrarile/ieșirile din fiecare funcție f;
- Atacul anterior nu funcționează; un nou atac va explora relațiile dintre ieșirile respectiv intrările în f_1 și f_3;
- Atacul traversează spațiul cheilor pentru fiecare jumatăte a cheii master și, în timp $2 \cdot 2^{28}$ va produce câte 2^{12} variante pentru fiecare jumătate a cheii;
- Complexitatea timp totală este $2 \cdot 2^{28} + 2^{24} < 2^{30}$.

Securitatea sistemului DES

- Incă de la propunerea sa, DES a fost criticat din două motive:

Securitatea sistemului DES

- Incă de la propunerea sa, DES a fost criticat din două motive:
  1. Spațiul cheilor este prea mic făcând algoritmul vulnerabil la forță brută;

Securitatea sistemului DES

- Incă de la propunerea sa, DES a fost criticat din două motive:
  1. Spațiul cheilor este prea mic făcând algoritmul vulnerabil la forță brută;
  2. Criteriile de selecție a SBOX-urilor au fost ținute secrete și ar fi putut exista atacuri analitice care explorau proprietățile matematice ale SBOX-urilor, cunoscute numai celor care l-au proiectat.

Securitatea sistemului DES

- Incă de la propunerea sa, DES a fost criticat din două motive:
  1. Spațiul cheilor este prea mic făcând algoritmul vulnerabil la forță brută;
  2. Criteriile de selecție a SBOX-urilor au fost ținute secrete și ar fi putut exista atacuri analitice care explorau proprietățile matematice ale SBOX-urilor, cunoscute numai celor care l-au proiectat.
- Cu toate acestea....

Securitatea sistemului DES

- Incă de la propunerea sa, DES a fost criticat din două motive:
  1. Spațiul cheilor este prea mic făcând algoritmul vulnerabil la forță brută;
  2. Criteriile de selecție a SBOX-urilor au fost ținute secrete și ar fi putut exista atacuri analitice care explorau proprietățile matematice ale SBOX-urilor, cunoscute numai celor care l-au proiectat.
- Cu toate acestea....
  - După 30 ani de studiu intens, cel mai bun atac practic rămâne doar o căutare exhaustivă pe spațiul cheilor;

Securitatea sistemului DES

- Incă de la propunerea sa, DES a fost criticat din două motive:
  1. Spațiul cheilor este prea mic făcând algoritmul vulnerabil la forță brută;
  2. Criteriile de selecție a SBOX-urilor au fost ținute secrete și ar fi putut exista atacuri analitice care explorau proprietățile matematice ale SBOX-urilor, cunoscute numai celor care l-au proiectat.
- Cu toate acestea....
  - După 30 ani de studiu intens, cel mai bun atac practic rămâne doar o căutare exhaustivă pe spațiul cheilor;
  - O căutare printre 2^{56} chei este fezabilă azi (dar netrivială);

Securitatea sistemului DES

- Incă de la propunerea sa, DES a fost criticat din două motive:
  1. Spațiul cheilor este prea mic făcând algoritmul vulnerabil la forță brută;
  2. Criteriile de selecție a SBOX-urilor au fost ținute secrete și ar fi putut exista atacuri analitice care explorau proprietățile matematice ale SBOX-urilor, cunoscute numai celor care l-au proiectat.
- Cu toate acestea....
  - După 30 ani de studiu intens, cel mai bun atac practic rămâne doar o căutare exhaustivă pe spațiul cheilor;
  - O căutare printre 2^{56} chei este fezabilă azi (dar netrivială);
  - In 1977, un calculator care să efectueze atacul într-o zi ar fi costat 20.000.000$;

Securitatea sistemului DES

- Primul atac practic a fost demonstrat în 1997 când un număr de provocări DES au fost propuse de RSA Security și rezolvate;

Securitatea sistemului DES

- Primul atac practic a fost demonstrat în 1997 când un număr de provocări DES au fost propuse de RSA Security și rezolvate;
- Prima provocare a fost spartă în 1997 de un proiect care a folosit sute de calculatoare coordonate prin Internet; a durat 96 de zile;

Securitatea sistemului DES

- Primul atac practic a fost demonstrat în 1997 când un număr de provocări DES au fost propuse de RSA Security și rezolvate;
- Prima provocare a fost spartă în 1997 de un proiect care a folosit sute de calculatoare coordonate prin Internet; a durat 96 de zile;
- A doua provocare a fost spartă anul următor în 41 de zile;

Securitatea sistemului DES

- Primul atac practic a fost demonstrat în 1997 când un număr de provocări DES au fost propuse de RSA Security și rezolvate;
- Prima provocare a fost spartă în 1997 de un proiect care a folosit sute de calculatoare coordonate prin Internet; a durat 96 de zile;
- A doua provocare a fost spartă anul următor în 41 de zile;
- Impresionant a fost timpul pentru a treia provocare: *56 de ore*;

Securitatea sistemului DES

- Primul atac practic a fost demonstrat în 1997 când un număr de provocări DES au fost propuse de RSA Security și rezolvate;
- Prima provocare a fost spartă în 1997 de un proiect care a folosit sute de calculatoare coordonate prin Internet; a durat 96 de zile;
- A doua provocare a fost spartă anul următor în 41 de zile;
- Impresionant a fost timpul pentru a treia provocare: *56 de ore*;
- S-a construit o mașină în acest scop, *Deep Crack* cu un cost de 250.000$

Securitatea sistemului DES

Figure: *Deep Crack-construită pentru căutare exhaustivă DES în 1998*

Securitatea sistemului DES

Figure: *Deep Crack-construită pentru căutare exhaustivă DES în 1998*

- Ultima provocare a fost spartă în 22 de ore (efort combinat de la ultimele două provocări);

Securitatea sistemului DES

Figure: *Deep Crack-construită pentru căutare exhaustivă DES în 1998*

- Ultima provocare a fost spartă în 22 de ore (efort combinat de la ultimele două provocări);
- Atacurile prin forță brută pe DES au devenit un studiu de caz în încercarea de a micșora costurile;

Securitatea sistemului DES

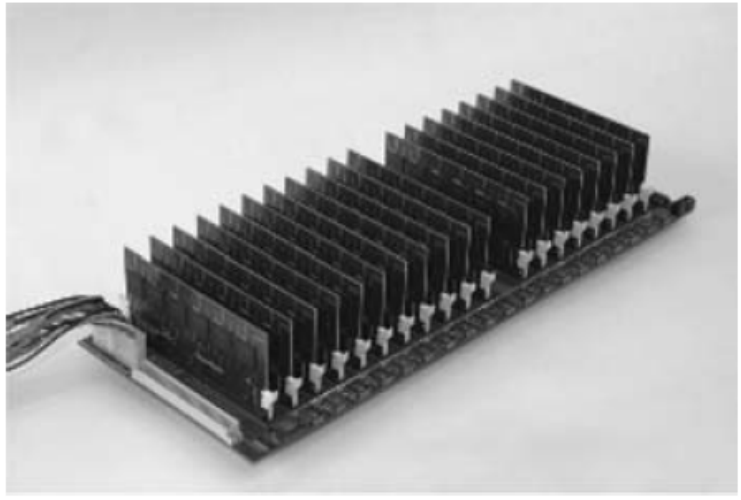


Figure: COPACABANA-Cost Optimized Parallel Code-Breaker

- In 2006, o echipă de cercetători de la universitățile Bochum și Kiel din Germania a construit mașina COPACABANA (*Cost Optimized Parallel Code-Breaker*) care permite găsirea cheii DES cu un timp de căutare mediu de mai puțin de 7 zile, la un cost de 10.000$.

Securitatea sistemului DES

- O altă problemă a sistemului DES, mai puțin importantă, este lungimea blocului relativ scurtă (64 biți);

Securitatea sistemului DES

- O altă problemă a sistemului DES, mai puțin importantă, este lungimea blocului relativ scurtă (64 biți);
- Securitatea multor construcții bazate pe cifruri bloc depinde de lungimea blocului;

Securitatea sistemului DES

- O altă problemă a sistemului DES, mai puțin importantă, este lungimea blocului relativ scurtă (64 biți);
- Securitatea multor construcții bazate pe cifruri bloc depinde de lungimea blocului;
- În modul de utilizare CTR, dacă un atacator are 2^{27} perechi text clar/text criptat, securitatea este compromisă cu probabilitate mare;

Securitatea sistemului DES

- O altă problemă a sistemului DES, mai puțin importantă, este lungimea blocului relativ scurtă (64 biți);
- Securitatea multor construcții bazate pe cifruri bloc depinde de lungimea blocului;
- În modul de utilizare CTR, dacă un atacator are 2^{27} perechi text clar/text criptat, securitatea este compromisă cu probabilitate mare;
- Concluzionând, putem spune că insecuritatea sistemului DES nu are a face cu structura internă sau construcția in sine (care este remarcabilă), ci se datorează numai lungimii cheii prea mici.

Criptanaliză avansată

- La sfarsitul anilor ′80, Biham și Shamir au dezvoltat o tehnică numită criptanaliza diferențială pe care au folosit-o pentru un atac împotriva DES;

Criptanaliză avansată

- La sfarsitul anilor ′80, Biham și Shamir au dezvoltat o tehnică numită criptanaliza diferențială pe care au folosit-o pentru un atac împotriva DES;
- Atacul presupune complexitate timp 2^{37} (memorie neglijabilă) dar cere ca atacatorul să analizeze 2^{36} texte criptate obținute dintr-o mulțime de 2^{47} texte clare alese;

Criptanaliză avansată

- La sfarsitul anilor ′80, Biham și Shamir au dezvoltat o tehnică numită criptanaliza diferențială pe care au folosit-o pentru un atac împotriva DES;
- Atacul presupune complexitate timp 2^{37} (memorie neglijabilă) dar cere ca atacatorul să analizeze 2^{36} texte criptate obținute dintr-o mulțime de 2^{47} texte clare alese;
- Din punct de vedere teoretic, atacul a fost o inovație, dar practic e aproape imposibil de realizat;

Criptanaliză avansată

- La sfarsitul anilor ′80, Biham și Shamir au dezvoltat o tehnică numită criptanaliza diferențială pe care au folosit-o pentru un atac împotriva DES;
- Atacul presupune complexitate timp 2^{37} (memorie neglijabilă) dar cere ca atacatorul să analizeze 2^{36} texte criptate obținute dintr-o mulțime de 2^{47} texte clare alese;
- Din punct de vedere teoretic, atacul a fost o inovație, dar practic e aproape imposibil de realizat;
- La inceputul anilor ′90, Matsui a dezvoltat criptanaliza liniară aplicată cu succes pe DES;

Criptanaliză avansată

- La sfarsitul anilor '80, Biham și Shamir au dezvoltat o tehnică numită criptanaliza diferențială pe care au folosit-o pentru un atac împotriva DES;
- Atacul presupune complexitate timp 2^{37} (memorie neglijabilă) dar cere ca atacatorul să analizeze 2^{36} texte criptate obținute dintr-o mulțime de 2^{47} texte clare alese;
- Din punct de vedere teoretic, atacul a fost o inovație, dar practic e aproape imposibil de realizat;
- La inceputul anilor '90, Matsui a dezvoltat criptanaliza liniară aplicată cu succes pe DES;
- Deși necesită 2^{43} texte criptate, avantajul este că textele clare nu trebuie să fie alese de atacator, ci doar cunoscute de el;

Criptanaliză avansată

- La sfarsitul anilor '80, Biham și Shamir au dezvoltat o tehnică numită criptanaliza diferențială pe care au folosit-o pentru un atac împotriva DES;
- Atacul presupune complexitate timp 2^{37} (memorie neglijabilă) dar cere ca atacatorul să analizeze 2^{36} texte criptate obținute dintr-o mulțime de 2^{47} texte clare alese;
- Din punct de vedere teoretic, atacul a fost o inovație, dar practic e aproape imposibil de realizat;
- La inceputul anilor '90, Matsui a dezvoltat criptanaliza liniară aplicată cu succes pe DES;
- Deși necesită 2^{43} texte criptate, avantajul este că textele clare nu trebuie să fie alese de atacator, ci doar cunoscute de el;
- Problema însă rămâne aceeași: atacul e foarte greu de pus în practică.

Criptanaliza diferențială

- Cataloghează diferențe specifice între texte clare care produc diferențe specifice în textele criptate cu probabilitate mai mare decât ar fi de așteptat pentru permutările aleatoare;

Criptanaliza diferențială

- Cataloghează diferențe specifice între texte clare care produc diferențe specifice în textele criptate cu probabilitate mai mare decât ar fi de așteptat pentru permutările aleatoare;
- Fie un cifru bloc cu blocul de lungime n și $\Delta_x, \Delta_y \in \{0,1\}^n$;

Criptanaliza diferențială

- Cataloghează diferențe specifice între texte clare care produc diferențe specifice în textele criptate cu probabilitate mai mare decât ar fi de așteptat pentru permutările aleatoare;
- Fie un cifru bloc cu blocul de lungime n și $\Delta_x, \Delta_y \in \{0,1\}^n$;
- Spunem că diferențiala (Δ_x, Δ_y) apare cu probabilitate p dacă pentru intrări aleatoare x_1, x_2 cu

$$x_1 \oplus x_2 = \Delta_x$$

și o alegere aleatoare a cheii k

$$Pr[F_k(x_1) \oplus F_k(x_2) = \Delta_y] = p$$

Criptanaliza diferențială

- Cataloghează diferențe specifice între texte clare care produc diferențe specifice în textele criptate cu probabilitate mai mare decât ar fi de așteptat pentru permutările aleatoare;
- Fie un cifru bloc cu blocul de lungime n și $\Delta_x, \Delta_y \in \{0,1\}^n$;
- Spunem că diferențiala (Δ_x, Δ_y) apare cu probabilitate p dacă pentru intrări aleatoare x_1, x_2 cu

$$x_1 \oplus x_2 = \Delta_x$$

și o alegere aleatoare a cheii k

$$Pr[F_k(x_1) \oplus F_k(x_2) = \Delta_y] = p$$

- Pentru o funcție aleatoare, probabilitatea de apariție a unei diferențiale nu e mai mare decât 2^{-n};

Criptanaliza diferențială

- Cataloghează diferențe specifice între texte clare care produc diferențe specifice în textele criptate cu probabilitate mai mare decât ar fi de așteptat pentru permutările aleatoare;
- Fie un cifru bloc cu blocul de lungime n și $\Delta_x, \Delta_y \in \{0,1\}^n$;
- Spunem că diferențiala (Δ_x, Δ_y) apare cu probabilitate p dacă pentru intrări aleatoare x_1, x_2 cu

$$x_1 \oplus x_2 = \Delta_x$$

și o alegere aleatoare a cheii k

$$Pr[F_k(x_1) \oplus F_k(x_2) = \Delta_y] = p$$

- Pentru o funcție aleatoare, probabilitatea de apariție a unei diferențiale nu e mai mare decât 2^{-n};
- La un cifru bloc slab, ea apare cu o probabilitate mult mai mare;

Criptanaliza diferențială

- Dacă o diferențială există cu probabilitate $p \gg 2^{-n}$, cifrul bloc nu mai este permutare pseudoaleatoare;

Criptanaliza diferențială

- Dacă o diferențială există cu probabilitate $p \gg 2^{-n}$, cifrul bloc nu mai este permutare pseudoaleatoare;
- Ideea este de a folosi multe diferențiale cu p ușor mai mare decât 2^{-n} pentru a găsi cheia secretă folosind un atac cu text clar ales;

Criptanaliza diferențială

- Dacă o diferențială există cu probabilitate $p \gg 2^{-n}$, cifrul bloc nu mai este permutare pseudoaleatoare;
- Ideea este de a folosi multe diferențiale cu p ușor mai mare decât 2^{-n} pentru a găsi cheia secretă folosind un atac cu text clar ales;
- Criptanaliza diferențială a fost folosită cu succes pentru a ataca cifruri bloc (altele decât DES si AES), de pildă FEAL-8;

Criptanaliza liniară

- Metoda consideră relații liniare între intrările și ieșirile unui cifru bloc;

Criptanaliza liniară

- Metoda consideră relații liniare între intrările și ieșirile unui cifru bloc;
- Spunem că porțiunile de biți $i_1, \cdots, i_l$ și ${i_1}', \cdots, {i_l}'$ au distanța p dacă pentru orice intrare aleatoare x și orice cheie k

$$Pr[x_{i_1} \oplus \cdots \oplus x_{i_l} \oplus y_{i'_1} \oplus \cdots \oplus y_{i'_l} = 0] = p$$

unde $y = F_k(x)$.

Criptanaliza liniară

- Metoda consideră relații liniare între intrările și ieșirile unui cifru bloc;
- Spunem că porțiunile de biți $i_1, \cdots, i_l$ și ${i_1}', \cdots, {i_l}'$ au distanța p dacă pentru orice intrare aleatoare x și orice cheie k

$$Pr[x_{i_1} \oplus \cdots \oplus x_{i_l} \oplus y_{i'_1} \oplus \cdots \oplus y_{i'_l} = 0] = p$$

unde $y = F_k(x)$.

- Pentru o funcție aleatoare se așteaptă ca $p = 0.5$;

Criptanaliza liniară

- Metoda consideră relații liniare între intrările și ieșirile unui cifru bloc;
- Spunem că porțiunile de biți $i_1, \cdots, i_l$ și $i_1', \cdots, i_l'$ au distanța p dacă pentru orice intrare aleatoare x și orice cheie k

$$Pr[x_{i_1} \oplus \cdots \oplus x_{i_l} \oplus y_{i_1'} \oplus \cdots \oplus y_{i_l'} = 0] = p$$

  unde $y = F_k(x)$.
- Pentru o funcție aleatoare se așteaptă ca $p = 0.5$;
- Matsui a arătat cum se poate folosi o diferență p mare pentru a sparge complet un cifru bloc;

Criptanaliza liniară

- Metoda consideră relații liniare între intrările și ieșirile unui cifru bloc;
- Spunem că porțiunile de biți $i_1, \cdots, i_l$ și $i_1', \cdots, i_l'$ au distanța p dacă pentru orice intrare aleatoare x și orice cheie k

$$Pr[x_{i_1} \oplus \cdots \oplus x_{i_l} \oplus y_{i_1'} \oplus \cdots \oplus y_{i_l'} = 0] = p$$

  unde $y = F_k(x)$.
- Pentru o funcție aleatoare se așteaptă ca $p = 0.5$;
- Matsui a arătat cum se poate folosi o diferență p mare pentru a sparge complet un cifru bloc;
- Necesită un număr foarte mare de perechi text clar/text criptat.

Creșterea lungimii cheii

- Singura vulnerabilitate practică DES este cheia scurtă;

Creșterea lungimii cheii

- Singura vulnerabilitate practică DES este cheia scurtă;
- S-au propus diverse metode de a construi un sistem bazat pe DES care să folosească o cheie mai lungă;

Creșterea lungimii cheii

- Singura vulnerabilitate practică DES este cheia scurtă;
- S-au propus diverse metode de a construi un sistem bazat pe DES care să folosească o cheie mai lungă;
- Nu se recomandă schimbarea structurii interne întrucât securitatea sistemului ar putea fi afectată;

Creșterea lungimii cheii

- Singura vulnerabilitate practică DES este cheia scurtă;
- S-au propus diverse metode de a construi un sistem bazat pe DES care să folosească o cheie mai lungă;
- Nu se recomandă schimbarea structurii interne întrucât securitatea sistemului ar putea fi afectată;
- Solutie alternativă: considerăm DES o cutie neagră care implementează un cifru bloc "perfect" cu o cheie pe 56 biți.

Criptare dublă

- Fie F un cifru bloc (in particular ne vom referi la DES); definim un alt cifru bloc F' astfel

$$F'_{k_1,k_2}(x) = F_{k_2}(F_{k_1}(x))$$

cu k_1, k_2 chei independente;

Criptare dublă

- Fie F un cifru bloc (in particular ne vom referi la DES); definim un alt cifru bloc F' astfel

$$F'_{k_1,k_2}(x) = F_{k_2}(F_{k_1}(x))$$

  cu k_1, k_2 chei independente;
- Lungimea totală a cheii este 112 biți, suficient de mare pentru căutare exhaustivă;

Criptare dublă

- Fie F un cifru bloc (in particular ne vom referi la DES); definim un alt cifru bloc F' astfel

$$F'_{k_1,k_2}(x) = F_{k_2}(F_{k_1}(x))$$

  cu k_1, k_2 chei independente;
- Lungimea totală a cheii este 112 biți, suficient de mare pentru căutare exhaustivă;
- Insă, se poate arăta un atac în timp 2^{56} unde $|k_1| = 56 = |k_2|$ (fața de $2^{2 \cdot 56}$ cât necesită o căutare exhaustivă);
- Atacul se numeste meet-in-the-middle;

Atacul meet-in-the-middle

- Iată cum funcționează atacul dacă se cunoaște o pereche text clar/text criptat (x, y) cu $y = F_{k_2}(F_{k_1}(x))$:

Atacul meet-in-the-middle

- Iată cum funcționează atacul dacă se cunoaște o pereche text clar/text criptat (x, y) cu $y = F_{k_2}(F_{k_1}(x))$:
  1. Pentru fiecare $k_1 \in \{0,1\}^n$, calculează $z := F_{k_1}(x)$ și păstrează (z, k_1);

Atacul meet-in-the-middle

- Iată cum funcționează atacul dacă se cunoaște o pereche text clar/text criptat (x, y) cu $y = F_{k_2}(F_{k_1}(x))$:
  1. Pentru fiecare $k_1 \in \{0,1\}^n$, calculează $z := F_{k_1}(x)$ și păstrează (z, k_1);
  2. Pentru fiecare $k_2 \in \{0,1\}^n$, calculează $z := F_{k_2}^{-1}(y)$ și păstrează (z, k_2);

Atacul meet-in-the-middle

- Iată cum funcționează atacul dacă se cunoaște o pereche text clar/text criptat (x, y) cu $y = F_{k_2}(F_{k_1}(x))$:
  1. Pentru fiecare $k_1 \in \{0,1\}^n$, calculează $z := F_{k_1}(x)$ și păstrează (z, k_1);
  2. Pentru fiecare $k_2 \in \{0,1\}^n$, calculează $z := F_{k_2}^{-1}(y)$ și păstrează (z, k_2);
  3. Verifică dacă există perechi (z, k_1) și (z, k_2) care coincid pe prima componentă;

Atacul meet-in-the-middle

- Iată cum funcționează atacul dacă se cunoaște o pereche text clar/text criptat (x, y) cu $y = F_{k_2}(F_{k_1}(x))$:
  1. Pentru fiecare $k_1 \in \{0,1\}^n$, calculează $z := F_{k_1}(x)$ și păstrează (z, k_1);
  2. Pentru fiecare $k_2 \in \{0,1\}^n$, calculează $z := F_{k_2}^{-1}(y)$ și păstrează (z, k_2);
  3. Verifică dacă există perechi (z, k_1) și (z, k_2) care coincid pe prima componentă;
  4. Atunci valorile k_1, k_2 corespunzătoare satisfac

$$F_{k_1}(x) = F_{k_2}^{-1}(y)$$

adică $y = F'_{k_1,k_2}(x)$

Atacul meet-in-the-middle

- Iată cum funcționează atacul dacă se cunoaște o pereche text clar/text criptat (x, y) cu $y = F_{k_2}(F_{k_1}(x))$:
  1. Pentru fiecare $k_1 \in \{0,1\}^n$, calculează $z := F_{k_1}(x)$ și păstrează (z, k_1);
  2. Pentru fiecare $k_2 \in \{0,1\}^n$, calculează $z := F_{k_2}^{-1}(y)$ și păstrează (z, k_2);
  3. Verifică dacă există perechi (z, k_1) și (z, k_2) care coincid pe prima componentă;
  4. Atunci valorile k_1, k_2 corespunzătoare satisfac

$$F_{k_1}(x) = F_{k_2}^{-1}(y)$$

  adică $y = F'_{k_1,k_2}(x)$

- Complexitatea timp a atacului este $O(2^n)$.

Criptare triplă

- Există două variante:

Criptare triplă

- Există două variante:
  1. Trei chei independente - k_1, k_2 și k_3 iar

$$F'_{k_1,k_2,k_3} = F_{k_3}(F_{k_2}^{-1}(F_{k_1}(x)))$$

Criptare triplă

- Există două variante:
  1. Trei chei independente - k_1, k_2 și k_3 iar

$$F'_{k_1,k_2,k_3} = F_{k_3}(F^{-1}_{k_2}(F_{k_1}(x)))$$

  2. Două chei independente - k_1 și k_2 iar

$$F'_{k_1,k_2} = F_{k_1}(F^{-1}_{k_2}(F_{k_1}(x)))$$

Criptare triplă

- Există două variante:
  1. Trei chei independente - k_1, k_2 și k_3 iar

$$F'_{k_1,k_2,k_3} = F_{k_3}(F^{-1}_{k_2}(F_{k_1}(x)))$$

  2. Două chei independente - k_1 și k_2 iar

$$F'_{k_1,k_2} = F_{k_1}(F^{-1}_{k_2}(F_{k_1}(x)))$$

- F' este ales astfel pentru a fi compatibil cu F atunci când cheile sunt alese $k_1 = k_2 = k_3$;

Criptare triplă

- Există două variante:
  1. Trei chei independente - k_1, k_2 și k_3 iar
  $$F'_{k_1,k_2,k_3} = F_{k_3}(F^{-1}_{k_2}(F_{k_1}(x)))$$
  2. Două chei independente - k_1 și k_2 iar
  $$F'_{k_1,k_2} = F_{k_1}(F^{-1}_{k_2}(F_{k_1}(x)))$$
- F' este ales astfel pentru a fi compatibil cu F atunci când cheile sunt alese $k_1 = k_2 = k_3$;
- Prima variantă are lungimea cheii $3n$ dar cel mai bun atac necesită timp 2^{2n} (funcționează atacul meet-in-the-middle);

Criptare triplă

- Există două variante:
  1. Trei chei independente - k_1, k_2 și k_3 iar
  $$F'_{k_1,k_2,k_3} = F_{k_3}(F^{-1}_{k_2}(F_{k_1}(x)))$$
  2. Două chei independente - k_1 și k_2 iar
  $$F'_{k_1,k_2} = F_{k_1}(F^{-1}_{k_2}(F_{k_1}(x)))$$
- F' este ales astfel pentru a fi compatibil cu F atunci când cheile sunt alese $k_1 = k_2 = k_3$;
- Prima variantă are lungimea cheii $3n$ dar cel mai bun atac necesită timp 2^{2n} (funcționează atacul meet-in-the-middle);
- A doua variantă are lungimea cheii $2n$ si cel mai bun atac necesită timp 2^{2n}.

Triplu-DES (3DES)

- Se bazează pe tripla invocare a lui DES folosind două sau trei chei;

Triplu-DES (3DES)

- Se bazează pe tripla invocare a lui DES folosind două sau trei chei;
- Este considerat sigur și în 1999 l-a înlocuit pe DES ca standard;

Triplu-DES (3DES)

- Se bazează pe tripla invocare a lui DES folosind două sau trei chei;
- Este considerat sigur și în 1999 l-a înlocuit pe DES ca standard;
- 3DES este foarte eficient în implementarile hardware (la fel ca și DES) dar totuși lent în implementari software;

Triplu-DES (3DES)

- Se bazează pe tripla invocare a lui DES folosind două sau trei chei;
- Este considerat sigur și în 1999 l-a înlocuit pe DES ca standard;
- 3DES este foarte eficient în implementarile hardware (la fel ca și DES) dar totuși lent în implementari software;
- Este încă folosit la scară largă, fiind considerat un cifru bloc puternic;

Triplu-DES (3DES)

- Se bazează pe tripla invocare a lui DES folosind două sau trei chei;
- Este considerat sigur și în 1999 l-a înlocuit pe DES ca standard;
- 3DES este foarte eficient în implementarile hardware (la fel ca și DES) dar totuși lent în implementari software;
- Este încă folosit la scară largă, fiind considerat un cifru bloc puternic;
- Este popular în aplicațiile financiare și în protejarea informațiilor biometrice din pașapoartele electronice;

Triplu-DES (3DES)

- Singurele dezavantaje ar fi lungimea mică a blocurilor și faptul că este destul de lent fiindcă aplică DES de trei ori;

Triplu-DES (3DES)

- Singurele dezavantaje ar fi lungimea mică a blocurilor și faptul că este destul de lent fiindcă aplică DES de trei ori;
- Acestea au dus la înlocuirea lui ca standard cu AES;

Triplu-DES (3DES)

- Singurele dezavantaje ar fi lungimea mică a blocurilor și faptul că este destul de lent fiindcă aplică DES de trei ori;
- Acestea au dus la înlocuirea lui ca standard cu AES;
- DES-X este o variantă DES care rezistă mai bine (decât DES) la forța brută;

Triplu-DES (3DES)

- Singurele dezavantaje ar fi lungimea mică a blocurilor și faptul că este destul de lent fiindcă aplică DES de trei ori;
- Acestea au dus la înlocuirea lui ca standard cu AES;
- DES-X este o variantă DES care rezistă mai bine (decât DES) la forța brută;
- DES-X folosește două chei suplimentare k_1, k_2:

$$DESX_{k,k_1,k_2} = k_2 \oplus DES_k(x \oplus k_1)$$

Important de reținut!

- DES a fost sistemul simetric dominant de la mijlocul anilor ′70 până la mijlocul anilor ′90;

Important de reținut!

- DES a fost sistemul simetric dominant de la mijlocul anilor ′70 până la mijlocul anilor ′90;
- DES cu cheia pe 56 biți poate fi spart relativ ușor astăzi prin forță brută;

Important de reținut!

- DES a fost sistemul simetric dominant de la mijlocul anilor ′70 până la mijlocul anilor ′90;
- DES cu cheia pe 56 biți poate fi spart relativ ușor astăzi prin forță brută;
- Insă, este foarte greu de spart folosind criptanaliza diferențială sau liniară;

Important de reținut!

- DES a fost sistemul simetric dominant de la mijlocul anilor ′70 până la mijlocul anilor ′90;
- DES cu cheia pe 56 biți poate fi spart relativ ușor astăzi prin forță brută;
- Insă, este foarte greu de spart folosind criptanaliza diferențială sau liniară;
- Pentru 3DES nu se cunoaste nici un atac practic.