Securitatea Sistemelor Informatice

- Curs 7.2 -
Funcții hash

Adela Georgescu

Facultatea de Matematică și Informatică
Universitatea din București
Anul universitar 2022-2023, semestrul I

Funcții Hash

- Întrebare: Ați auzit vreodată de funcții hash? În ce context?

Funcții Hash

- Întrebare: Ați auzit vreodată de funcții hash? În ce context?
- Acestea primesc ca argument o secvență de *lungime variabilă* pe care o comprimă într-o secvență de *lungime mai mică, fixată*;

Funcții Hash

- Întrebare: Ați auzit vreodată de funcții hash? În ce context?
- Acestea primesc ca argument o secvență de *lungime variabilă* pe care o comprimă într-o secvență de *lungime mai mică, fixată*;
- Utilizarea clasică a funcțiilor hash este în domeniul *structurilor de date*;

Funcții Hash

- Întrebare: Ați auzit vreodată de funcții hash? În ce context?
- Acestea primesc ca argument o secvență de *lungime variabilă* pe care o comprimă într-o secvență de *lungime mai mică, fixată*;
- Utilizarea clasică a funcțiilor hash este în domeniul *structurilor de date*;
- Să luăm un exemplu...

Funcții Hash

- Considerăm o mulțime de elemente de dimensiune mare care trebuie stocată (într-un tablou);

Funcții Hash

- Considerăm o mulțime de elemente de dimensiune mare care trebuie stocată (într-un tablou);

Adresă
Batiștei nr.17
Academiei nr.23
Tudor Arghezi nr.103
Nicolae Bălcescu nr.10
C.A.Rosetti nr.7
Hristo Botev nr.35
...

Funcții Hash

- Considerăm o mulțime de elemente de dimensiune mare care trebuie stocată (într-un tablou);

Adresă
Batiștei nr.17
Academiei nr.23
Tudor Arghezi nr.103
Nicolae Bălcescu nr.10
C.A.Rosetti nr.7
Hristo Botev nr.35
...

- Ulterior, elementele trebuie să fie ușor accesibile;

Funcții Hash

- Considerăm o mulțime de elemente de dimensiune mare care trebuie stocată (într-un tablou);

Adresă
Batiștei nr.17
Academiei nr.23
Tudor Arghezi nr.103
Nicolae Bălcescu nr.10
C.A.Rosetti nr.7
Hristo Botev nr.35
...

- Ulterior, elementele trebuie să fie ușor accesibile;
- Întrebare: Cum procedăm?

Funcții Hash

- **Varianta 1.** Elementele se stochează la rând;

Funcții Hash

- **Varianta 1.** Elementele se stochează la rând;

Index	Adresă
1	Batiștei nr.17
2	Academiei nr.23
3	Tudor Arghezi nr.103
4	Nicolae Bălcescu nr.10
5	C.A.Rosetti nr.7
6	Hristo Botev nr.35
...	... ...

Funcții Hash

- **Varianta 1.** Elementele se stochează la rând;

Index	Adresă
1	Batiștei nr.17
2	Academiei nr.23
3	Tudor Arghezi nr.103
4	Nicolae Bălcescu nr.10
5	C.A.Rosetti nr.7
6	Hristo Botev nr.35
...	... ...

- Dar o căutare necesită un algoritm de complexitate ...

Funcții Hash

- **Varianta 1.** Elementele se stochează la rând;

Index	Adresă
1	Batiștei nr.17
2	Academiei nr.23
3	Tudor Arghezi nr.103
4	Nicolae Bălcescu nr.10
5	C.A.Rosetti nr.7
6	Hristo Botev nr.35
...	... ...

- Dar o căutare necesită un algoritm de complexitate ... $O(n)$;

Funcții Hash

- **Varianta 2.** Tabloul de elemente este sortat.

Funcții Hash

- **Varianta 2.** Tabloul de elemente este sortat.

Index	Adresă
...	... ...
17	Academiei nr.23
...	... ...
120	Batiștei nr.17
...	... ...
223	C.A.Rosetti nr.7
...	... ...
401	Hristo Botev nr.35
...	... ...
503	Nicolae Bălcescu nr.10
...	... ...
696	Tudor Arghezi nr.103
...	... ...

Funcții Hash

- **Varianta 2.** Tabloul de elemente este sortat.

Index	Adresă
...	... ...
17	Academiei nr.23
...	... ...
120	Batiștei nr.17
...	... ...
223	C.A.Rosetti nr.7
...	... ...
401	Hristo Botev nr.35
...	... ...
503	Nicolae Bălcescu nr.10
...	... ...
696	Tudor Arghezi nr.103
...	... ...

- În acest caz o căutare necesită un algoritm de complexitate ...

Funcții Hash

- **Varianta 2.** Tabloul de elemente este sortat.

Index	Adresă
...	... ...
17	Academiei nr.23
...	... ...
120	Batiștei nr.17
...	... ...
223	C.A.Rosetti nr.7
...	... ...
401	Hristo Botev nr.35
...	... ...
503	Nicolae Bălcescu nr.10
...	... ...
696	Tudor Arghezi nr.103
...	... ...

- În acest caz o căutare necesită un algoritm de complexitate ... $O(\log n)$;

Funcții Hash

- **Varianta 3.** Se folosește o funcție hash H și fiecare element x se stochează la indexul $H(x)$;

Funcții Hash

- **Varianta 3.** Se folosește o funcție hash H și fiecare element x se stochează la indexul $H(x)$;

Index	Adresă
...	... ...
14	Tudor Arghezi nr.103
...	... ...
29	Batiștei nr.17
...	... ...
113	C.A.Rosetti nr.7
...	... ...
365	Academiei nr.23
...	... ...
411	Nicolae Bălcescu nr.10
...	... ...
703	Hristo Botev nr.35
...	... ...

Funcții Hash

- **Varianta 3.** Se folosește o funcție hash H și fiecare element x se stochează la indexul $H(x)$;

Index	Adresă
...	... ...
14	Tudor Arghezi nr.103
...	... ...
29	Batiștei nr.17
...	... ...
113	C.A.Rosetti nr.7
...	... ...
365	Academiei nr.23
...	... ...
411	Nicolae Bălcescu nr.10
...	... ...
703	Hristo Botev nr.35
...	... ...

- Căutarea devine optimă pentru că se realizează în ...

Funcții Hash

- **Varianta 3.** Se folosește o funcție hash H și fiecare element x se stochează la indexul $H(x)$;

Index	Adresă
...	... ...
14	Tudor Arghezi nr.103
...	... ...
29	Batiștei nr.17
...	... ...
113	C.A.Rosetti nr.7
...	... ...
365	Academiei nr.23
...	... ...
411	Nicolae Bălcescu nr.10
...	... ...
703	Hristo Botev nr.35
...	... ...

- Căutarea devine optimă pentru că se realizează în ... $O(1)$!

Funcții Hash

- În exemplul precedent:
  - H(”Tudor Arghezi nr.103”) = 14;
  - H(”Batiștei nr.17”) = 29;
  - ...

Funcții Hash

- În exemplul precedent:
  - H("Tudor Arghezi nr.103") = 14;
  - H("Batiștei nr.17") = 29;
  - ...
- Analizăm, pe rând, cele 3 operații: **căutare**, **adăugare**, **ștergere**;

Funcții Hash

- În exemplul precedent:
  - H("Tudor Arghezi nr.103") = 14;
  - H("Batiștei nr.17") = 29;
  - ...
- Analizăm, pe rând, cele 3 operații: **căutare**, **adăugare**, **ștergere**;
- Pentru **căutarea** adresei *Edgar Quinet nr.35*, se calculează H("Edgar Quinet nr.35");

Funcții Hash

- În exemplul precedent:
  - H("Tudor Arghezi nr.103") = 14;
  - H("Batiștei nr.17") = 29;
  - ...
- Analizăm, pe rând, cele 3 operații: **căutare**, **adăugare**, **ștergere**;
- Pentru **căutarea** adresei *Edgar Quinet nr.35*, se calculează H("Edgar Quinet nr.35");
- Presupunând că H("Edgar Quinet nr.35") = 79, atunci se verifică ce este stocat pe poziția 79;

Funcții Hash

- Pentru **introducerea** adresei *Nicolae Filipescu nr.31*, se calculează H("Nicolae Filipescu nr.31");

Funcții Hash

- Pentru **introducerea** adresei *Nicolae Filipescu nr.31*, se calculează H("Nicolae Filipescu nr.31");
- Presupunând că H("Nicolae Filipescu nr.31") = 153, atunci se introduce valoarea *Nicolae Filipescu nr.31* la indexul 153;

Funcții Hash

- Pentru **introducerea** adresei *Nicolae Filipescu nr.31*, se calculează H("Nicolae Filipescu nr.31");
- Presupunând că H("Nicolae Filipescu nr.31") = 153, atunci se introduce valoarea *Nicolae Filipescu nr.31* la indexul 153;
- Pentru **ștergerea** adresei *Hristo Botev nr.35*, se calculează H("Hristo Botev nr.35");

Funcții Hash

- Pentru **introducerea** adresei *Nicolae Filipescu nr.31*, se calculează H("Nicolae Filipescu nr.31");
- Presupunând că H("Nicolae Filipescu nr.31") = 153, atunci se introduce valoarea *Nicolae Filipescu nr.31* la indexul 153;
- Pentru **ștergerea** adresei *Hristo Botev nr.35*, se calculează H("Hristo Botev nr.35");
- Se obține H("Hristo Botev nr.35") = 401 și se eliberează această celulă;

Funcții Hash

- Întrebare: Ce se întâmplă dacă pentru 2 valori $x \neq x'$, $H(x) = H(x')$?

Funcții Hash

- Întrebare: Ce se întâmplă dacă pentru 2 valori $x \neq x'$, $H(x) = H(x')$?
- Răspuns: În acest caz, apare o coliziune - ambele valori se vor stoca în aceeași celulă.

Funcții Hash

- În criptografie, o funcție hash rămâne o funcție care comprimă secvențe de lungime variabilă în secvențe de lungime fixă, insă:

Funcții Hash

- În criptografie, o funcție hash rămâne o funcție care comprimă secvențe de lungime variabilă în secvențe de lungime fixă, insă:
- Dacă în contextul structurilor de date este *preferabil* să se minimizeze coliziunile, în criptografie acest lucru este **impus**;

Funcții Hash

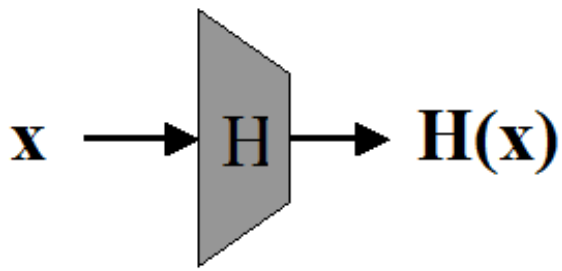
- În criptografie, o funcție hash rămâne o funcție care comprimă secvențe de lungime variabilă în secvențe de lungime fixă, insă:
- Dacă în contextul structurilor de date este *preferabil* să se minimizeze coliziunile, în criptografie acest lucru este **impus**;
- Dacă în contextul structurilor de date coliziunile apar *arbitrar* (valorile sunt alese independent de funcția hash), în criptografie coliziunile trebuie evitate chiar dacă sunt căutate **în mod voit** (de către adversar).

Funcții Hash

- Întrebare: Există funcții hash fără coliziuni?

Funcții Hash

- Întrebare: Există funcții hash fără coliziuni?
- Răspuns: NU! Funcțiile hash (cel puțin cele interesante d.p.d.v criptografic) comprimă, deci funcția nu poate fi injectivă;



Funcții Hash

- În aceste condiții, o funcție hash impune ca determinarea unei coliziuni să devină dificilă;

Funcții Hash

- În aceste condiții, o funcție hash impune ca determinarea unei coliziuni să devină dificilă;
- Considerăm funcțiile hash de domeniu infinit și ieșire de lungime fixată $l(n)$, unde $l(n)$ este un polinom în parametrul de securitate n:

$$H : \{0,1\}^* \rightarrow \{0,1\}^{l(n)}$$

Experimentul $Hash^{coll}_{\mathcal{A},H}(n)$

- Considerăm experimentul $Hash^{coll}_{\mathcal{A},H}(n)$;

Experimentul $Hash^{coll}_{\mathcal{A},H}(n)$

- Considerăm experimentul $Hash^{coll}_{\mathcal{A},H}(n)$;
- Adversarul $\mathcal{A}$ indică 2 valori $x, x' \in \{0,1\}^*$;

Experimentul $Hash^{coll}_{\mathcal{A},H}(n)$

- Considerăm experimentul $Hash^{coll}_{\mathcal{A},H}(n)$;
- Adversarul $\mathcal{A}$ indică 2 valori $x, x' \in \{0,1\}^*$;
- Se definește valoarea experimentului $Hash^{coll}_{\mathcal{A},H}(n) = \mathbf{1}$ dacă $x \neq x'$ și $H(x) = H(x')$;

Experimentul $Hash^{coll}_{\mathcal{A},H}(n)$

- Considerăm experimentul $Hash^{coll}_{\mathcal{A},H}(n)$;
- Adversarul $\mathcal{A}$ indică 2 valori $x, x' \in \{0,1\}^*$;
- Se definește valoarea experimentului $Hash^{coll}_{\mathcal{A},H}(n) = \mathbf{1}$ dacă $x \neq x'$ și $H(x) = H(x')$;
- Altfel, $Hash^{coll}_{\mathcal{A},H}(n) = \mathbf{0}$.

Rezistența la coliziuni

Definiție

$H : \{0,1\}^* \to \{0,1\}^{l(n)}$ se numește rezistentă la coliziuni (collision-resistant) dacă pentru orice adversar PPT $\mathcal{A}$ există o funcție neglijabilă negl așa încât

$$Pr[Hash^{coll}_{\mathcal{A},H}(n) = 1] \leq negl(n).$$

Securitatea funcțiilor hash

- În practică, rezistența la coliziuni poate fi dificil de obținut;

Securitatea funcțiilor hash

- În practică, rezistența la coliziuni poate fi dificil de obținut;
- Pentru anumite aplicații sunt utile noțiuni mai relaxate de securitate;

Securitatea funcțiilor hash

- În practică, rezistența la coliziuni poate fi dificil de obținut;
- Pentru anumite aplicații sunt utile noțiuni mai relaxate de securitate;
- Există 3 nivele de securitate:

Securitatea funcțiilor hash

- În practică, rezistența la coliziuni poate fi dificil de obținut;
- Pentru anumite aplicații sunt utile noțiuni mai relaxate de securitate;
- Există 3 nivele de securitate:

    1. Rezistența la coliziuni: este cea mai puternică noțiune de securitate și deja am definit-o formal;

Securitatea funcțiilor hash

- În practică, rezistența la coliziuni poate fi dificil de obținut;
- Pentru anumite aplicații sunt utile noțiuni mai relaxate de securitate;
- Există 3 nivele de securitate:

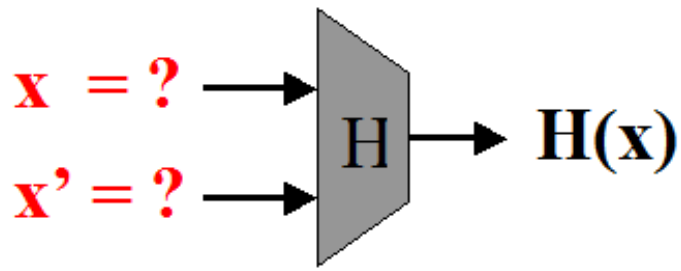
  1. Rezistența la coliziuni: este cea mai puternică noțiune de securitate și deja am definit-o formal;
  2. Rezistența la a doua preimagine: presupune că fiind dat x este dificil de determinat $x' \neq x$ a.î. $H(x) = H(x')$

Securitatea funcțiilor hash

- În practică, rezistența la coliziuni poate fi dificil de obținut;
- Pentru anumite aplicații sunt utile noțiuni mai relaxate de securitate;
- Există 3 nivele de securitate:

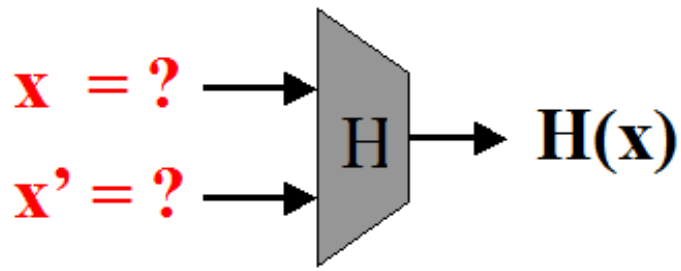
1. Rezistența la coliziuni: este cea mai puternică noțiune de securitate și deja am definit-o formal;
2. Rezistența la a doua preimagine: presupune că fiind dat x este dificil de determinat $x' \neq x$ a.î. $H(x) = H(x')$
3. Rezistența la prima preimagine: presupune că fiind dat $H(x)$ este imposibil de determinat x.

Rezistența la coliziuni



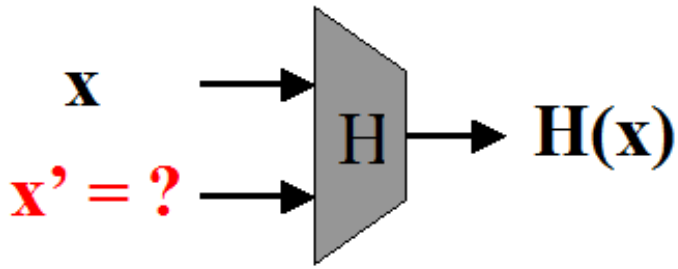
- **Provocare:** Se cer 2 valori $x \neq x'$ a.î. $H(x) = H(x')$;

Rezistența la coliziuni



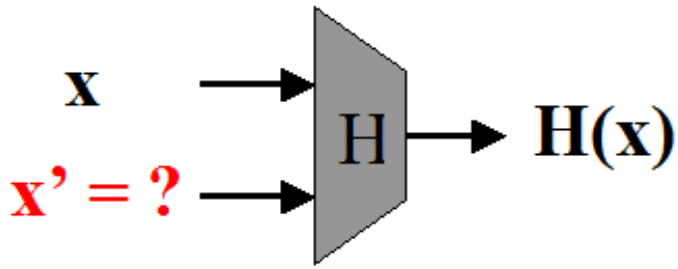
- **Provocare:** Se cer 2 valori $x \neq x'$ a.î. $H(x) = H(x')$;
- **Atac**: "Atacul zilei de naștere" necesită $\approx 2^{l(n)/2}$ evaluări pentru H.

Rezistența la a doua preimagine



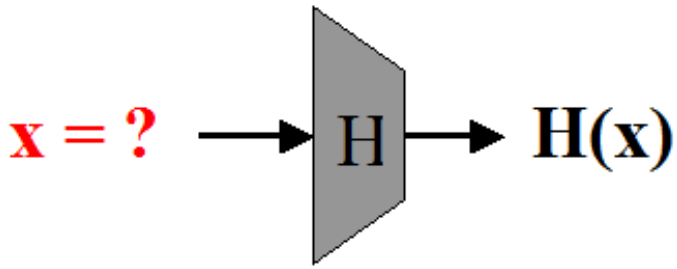
- **Provocare:** Fiind dat x, se cere x' a.î. $H(x) = H(x')$;

Rezistența la a doua preimagine



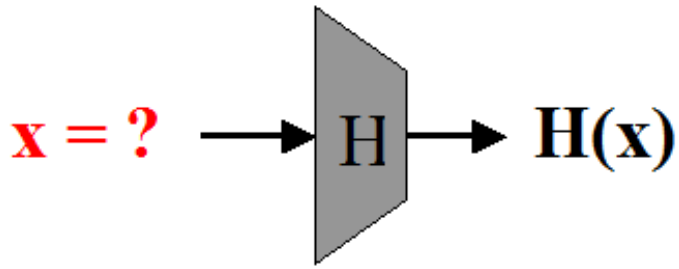
- **Provocare:** Fiind dat x, se cere x' a.î. $H(x) = H(x')$;
- **Atac**: Un atac generic necesită $\approx 2^{l(n)}$ evaluări pentru H.

Rezistența la prima preimagine



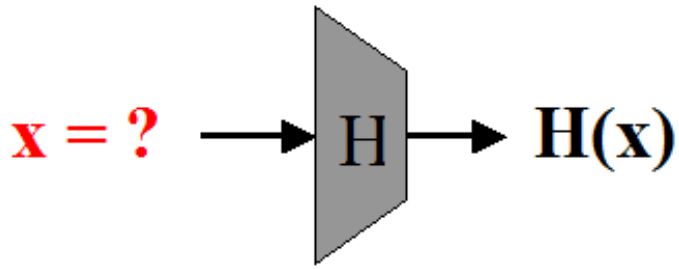
- **Provocare:** Fiind dat $y = H(x)$, se cere x a.î. $H(x) = y$;

Rezistența la prima preimagine



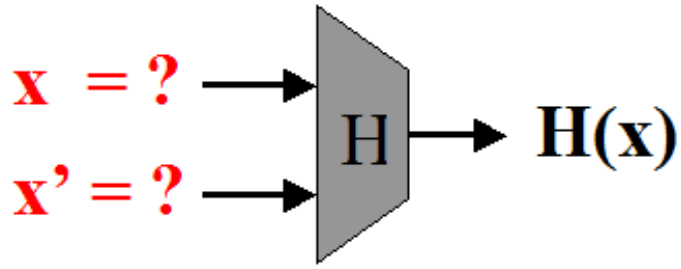
- **Provocare:** Fiind dat $y = H(x)$, se cere x a.î. $H(x) = y$;
- O astfel de funcție se numește și *calculabilă într-un singur sens (one-way function)*;

Rezistența la prima preimagine



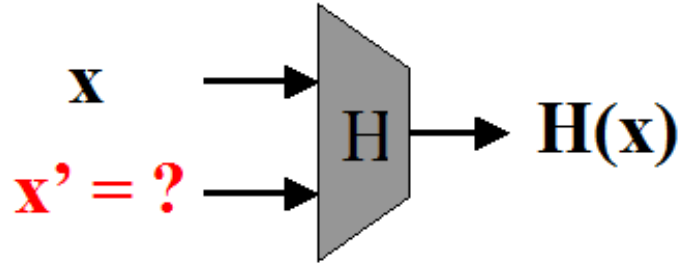
- **Provocare:** Fiind dat $y = H(x)$, se cere x a.î. $H(x) = y$;
- O astfel de funcție se numește și *calculabilă într-un singur sens (one-way function)*;
- **Atac**: Un atac generic necesită $\approx 2^{l(n)}$ evaluări pentru H.

Atacuri în practică

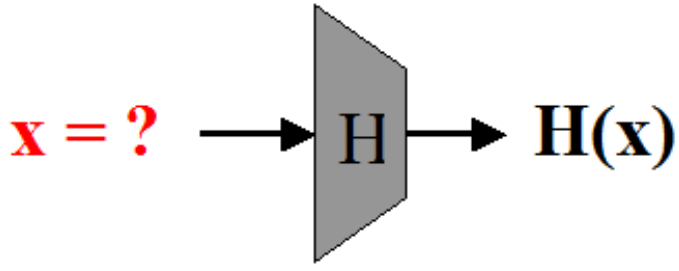


$\{x\}$ = documente originale
$\{x'\}$ = documente false
cineva este de acord să semneze electronic documentul original, însă semnătura devine valabilă și pentru un document fals

---



documentul x care este semnat electronic poate fi înlocuit de documentul x'



dacă se cunoaște cheia de sesiune k_i calculată din cheia master k
$x = H(k||i)$, atunci se determină cheia k

Securitatea funcțiilor hash

- Întrebare: De ce o funcție care satisface proprietatea de **rezistență la coliziuni** satisface și proprietatea de **rezistență la a doua preimagine**?

Securitatea funcțiilor hash

- Întrebare: De ce o funcție care satisface proprietatea de **rezistență la coliziuni** satisface și proprietatea de **rezistență la a doua preimagine**?
- Răspuns: Pentru x fixat, adversarul determină $x' \neq x$ pentru care $H(x) = H(x')$, deci găsește o coliziune;

Securitatea funcțiilor hash

- Întrebare: De ce o funcție care satisface proprietatea de **rezistență la coliziuni** satisface și proprietatea de **rezistență la a doua preimagine**?
- Răspuns: Pentru x fixat, adversarul determină $x' \neq x$ pentru care $H(x) = H(x')$, deci găsește o coliziune;
- Întrebare: De ce o funcție care satisface proprietatea de **rezistență la a doua preimagine** satisface și proprietatea de **rezistență la prima imagine**?

Securitatea funcțiilor hash

- Întrebare: De ce o funcție care satisface proprietatea de **rezistență la coliziuni** satisface și proprietatea de **rezistență la a doua preimagine**?
- Răspuns: Pentru x fixat, adversarul determină $x' \neq x$ pentru care $H(x) = H(x')$, deci găsește o coliziune;
- Întrebare: De ce o funcție care satisface proprietatea de **rezistență la a doua preimagine** satisface și proprietatea de **rezistență la prima imagine**?
- Răspuns: Pentru x oarecare, adversarul calculează $H(x)$, o inversează și determină x' a.î. $H(x') = H(x)$. Cu probabilitate mare $x' \neq x$, deci găsește o a doua preimagine.

Atacul "zilei de naștere"

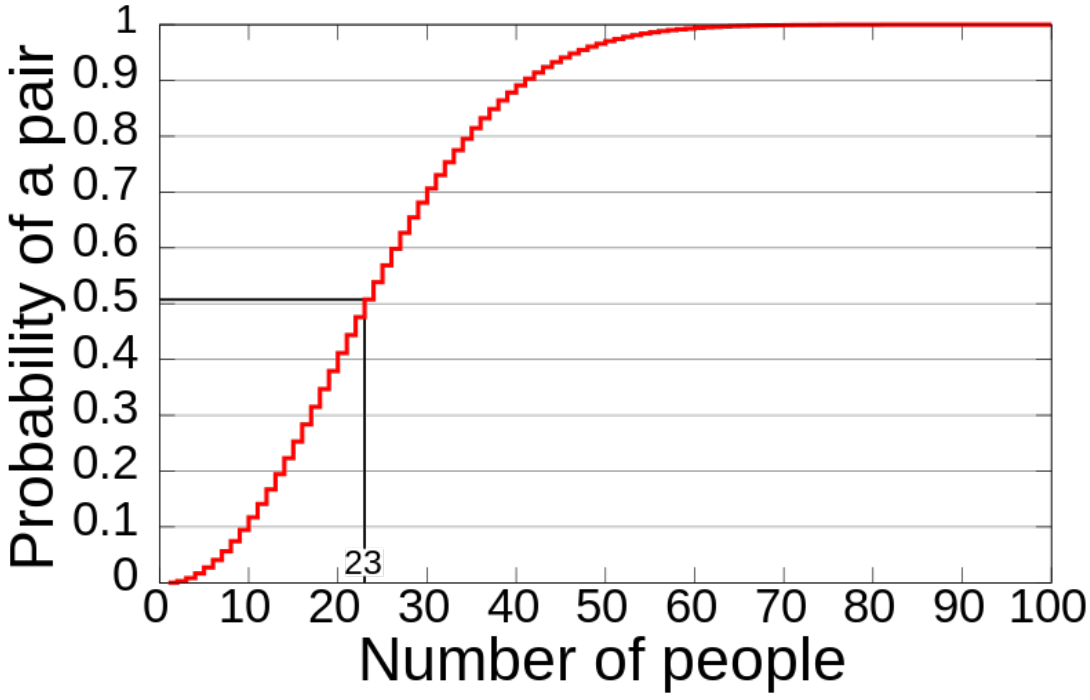
- Analizăm posibilitatea de a determina o coliziune pornind de la un exemplu clasic: *paradoxul nașterilor*;

Atacul "zilei de naștere"

- Analizăm posibilitatea de a determina o coliziune pornind de la un exemplu clasic: *paradoxul nașterilor*;
- Întrebare: Care este dimensiunea unui grup de oameni pentru ca 2 dintre ei să fie născuți în aceeași zi cu probabilitate 1/2 ?

Atacul "zilei de naștere"

- Analizăm posibilitatea de a determina o coliziune pornind de la un exemplu clasic: *paradoxul nașterilor*;
- Întrebare: Care este dimensiunea unui grup de oameni pentru ca 2 dintre ei să fie născuți în aceeași zi cu probabilitate 1/2 ?
- Răspuns: 23!



[Wikipedia]

Atacul "zilei de naștere"

- Generalizând, considerăm o mulțime de dimensiune n și q elemente uniform aleatoare din această mulțime $y_1, \ldots, y_q$;

Atacul "zilei de naștere"

- Generalizând, considerăm o mulțime de dimensiune n și q elemente uniform aleatoare din această mulțime $y_1, \ldots, y_q$;
- Atunci pentru $q \geq 1.2 \times 2^{n/2}$ probabilitatea să existe $i \neq j$ a.î. $y_i = y_j$ este $\geq 1/2$.

Atacul "zilei de naștere"

- Generalizând, considerăm o mulțime de dimensiune n și q elemente uniform aleatoare din această mulțime $y_1, \ldots, y_q$;
- Atunci pentru $q \geq 1.2 \times 2^{n/2}$ probabilitatea să existe $i \neq j$ a.î. $y_i = y_j$ este $\geq 1/2$.
- Aceast rezultat conduce imediat la un atac asupra funcțiilor hash cu scopul de a determina coliziuni:

Atacul "zilei de naștere"

- Generalizând, considerăm o mulțime de dimensiune n și q elemente uniform aleatoare din această mulțime $y_1, \ldots, y_q$;
- Atunci pentru $q \geq 1.2 \times 2^{n/2}$ probabilitatea să existe $i \neq j$ a.î. $y_i = y_j$ este $\geq 1/2$.
- Aceast rezultat conduce imediat la un atac asupra funcțiilor hash cu scopul de a determina coliziuni:
  - Adversarul alege $2^{n/2}$ valori x_i;

Atacul "zilei de naștere"

- Generalizând, considerăm o mulțime de dimensiune n și q elemente uniform aleatoare din această mulțime $y_1, \ldots, y_q$;
- Atunci pentru $q \geq 1.2 \times 2^{n/2}$ probabilitatea să existe $i \neq j$ a.î. $y_i = y_j$ este $\geq 1/2$.
- Aceast rezultat conduce imediat la un atac asupra funcțiilor hash cu scopul de a determina coliziuni:
  - Adversarul alege $2^{n/2}$ valori x_i;
  - Calculează pentru fiecare $y_i = H(x_i)$;

Atacul "zilei de naștere"

- Generalizând, considerăm o mulțime de dimensiune n și q elemente uniform aleatoare din această mulțime $y_1, \ldots, y_q$;
- Atunci pentru $q \geq 1.2 \times 2^{n/2}$ probabilitatea să existe $i \neq j$ a.î. $y_i = y_j$ este $\geq 1/2$.
- Aceast rezultat conduce imediat la un atac asupra funcțiilor hash cu scopul de a determina coliziuni:
  - Adversarul alege $2^{n/2}$ valori x_i;
  - Calculează pentru fiecare $y_i = H(x_i)$;
  - Caută $i \neq j$ cu $H(x_i) = H(x_j)$;

Atacul "zilei de naștere"

- Generalizând, considerăm o mulțime de dimensiune n și q elemente uniform aleatoare din această mulțime $y_1, \ldots, y_q$;
- Atunci pentru $q \geq 1.2 \times 2^{n/2}$ probabilitatea să existe $i \neq j$ a.î. $y_i = y_j$ este $\geq 1/2$.
- Aceast rezultat conduce imediat la un atac asupra funcțiilor hash cu scopul de a determina coliziuni:
  - Adversarul alege $2^{n/2}$ valori x_i;
  - Calculează pentru fiecare $y_i = H(x_i)$;
  - Caută $i \neq j$ cu $H(x_i) = H(x_j)$;
  - Dacă nu găsește nici o coliziune, reia atacul.

Atacul "zilei de naștere"

- Generalizând, considerăm o mulțime de dimensiune n și q elemente uniform aleatoare din această mulțime $y_1, \ldots, y_q$;
- Atunci pentru $q \geq 1.2 \times 2^{n/2}$ probabilitatea să existe $i \neq j$ a.î. $y_i = y_j$ este $\geq 1/2$.
- Aceast rezultat conduce imediat la un atac asupra funcțiilor hash cu scopul de a determina coliziuni:
  - Adversarul alege $2^{n/2}$ valori x_i;
  - Calculează pentru fiecare $y_i = H(x_i)$;
  - Caută $i \neq j$ cu $H(x_i) = H(x_j)$;
  - Dacă nu găsește nici o coliziune, reia atacul.
- Cum probabilitatea de succes a atacului este $\geq 1/2$, atunci numărul de încercări este ≈ 2.

Transformarea Merkle-Damgård

- Numim funcție de compresie o funcție hash rezistentă la coliziuni de intrare de lungime fixă;

Transformarea Merkle-Damgård

- Numim funcție de compresie o funcție hash rezistentă la coliziuni de intrare de lungime fixă;
- Întrebare: Intuitiv, ce se construiește mai ușor, H_1 sau H_2 ?

$$H_1 : \{0,1\}^{m_1} \to \{0,1\}^{n_1}, m_1 > n_1, m_1 \approx n_1$$

$$H_2 : \{0,1\}^{m_2} \to \{0,1\}^{n_2}, m_2 \gg n_2$$

Transformarea Merkle-Damgård

- Numim funcție de compresie o funcție hash rezistentă la coliziuni de intrare de lungime fixă;
- Întrebare: Intuitiv, ce se construiește mai ușor, H_1 sau H_2 ?

$$H_1 : \{0,1\}^{m_1} \to \{0,1\}^{n_1}, m_1 > n_1, m_1 \approx n_1$$

$$H_2 : \{0,1\}^{m_2} \to \{0,1\}^{n_2}, m_2 \gg n_2$$

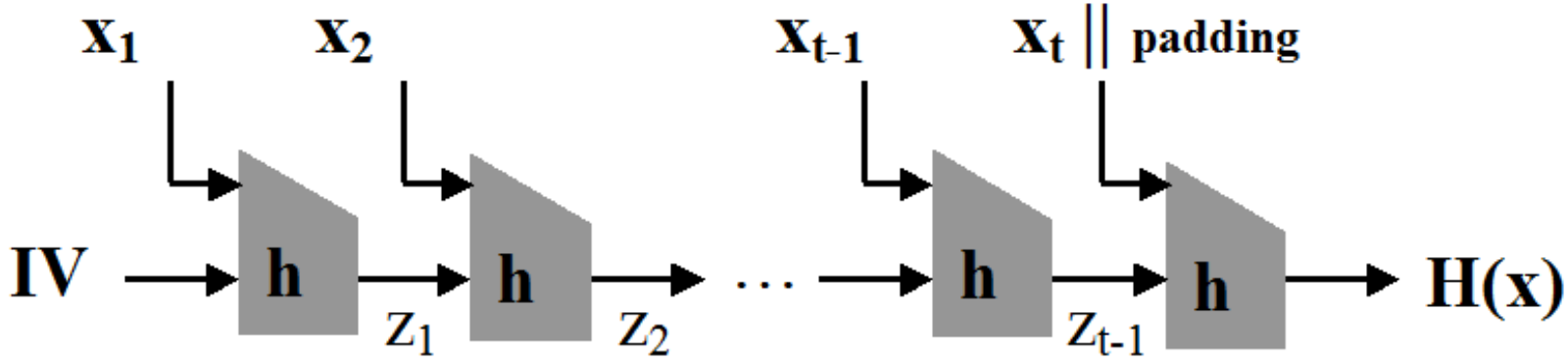
- Răspuns: Cu cât domeniul și codomeniul funcției sunt mai apropiate ca dimensiune, numărul de coliziuni scade $\Rightarrow$ coliziunile devin mai dificil de determinat (considerăm că funcția distribuie valorile uniform aleator).

Transformarea Merkle-Damgård

- Scopul este să construim o funcție hash (cu intrare de lungime variabilă), pornind de la o funcție de compresie (de lungime fixă);

Transformarea Merkle-Damgård

- Scopul este să construim o funcție hash (cu intrare de lungime variabilă), pornind de la o funcție de compresie (de lungime fixă);
- Pentru aceasta se aplică transformarea Merkle-Damgård;



Notații

- h = o funcție de compresie de dimensiune fixă
- $x = x_1||x_2||\ldots||x_{t-1}||x_t$ = valoarea de intrare
- IV = vector de inițializare
- padding = 100...0|| lungimea mesajului

Transformarea Merkle-Damgård

Teoremă

Dacă h prezintă rezistență la coliziuni, atunci și H prezintă rezistență la coliziuni.

Transformarea Merkle-Damgård

Teoremă

Dacă h prezintă rezistență la coliziuni, atunci și H prezintă rezistență la coliziuni.

- Intuitiv, dacă există $x \neq x'$ a.î. $H(x) = H(x')$, atunci trebuie să existe $< z_{i-1}, x_i > \neq < z'_{i-1}, x'_i >$ a.î. $h(z_{i-1}||x_i) = h(z'_{i-1}||x'_i)$;

Transformarea Merkle-Damgård

Teoremă

Dacă h prezintă rezistență la coliziuni, atunci și H prezintă rezistență la coliziuni.

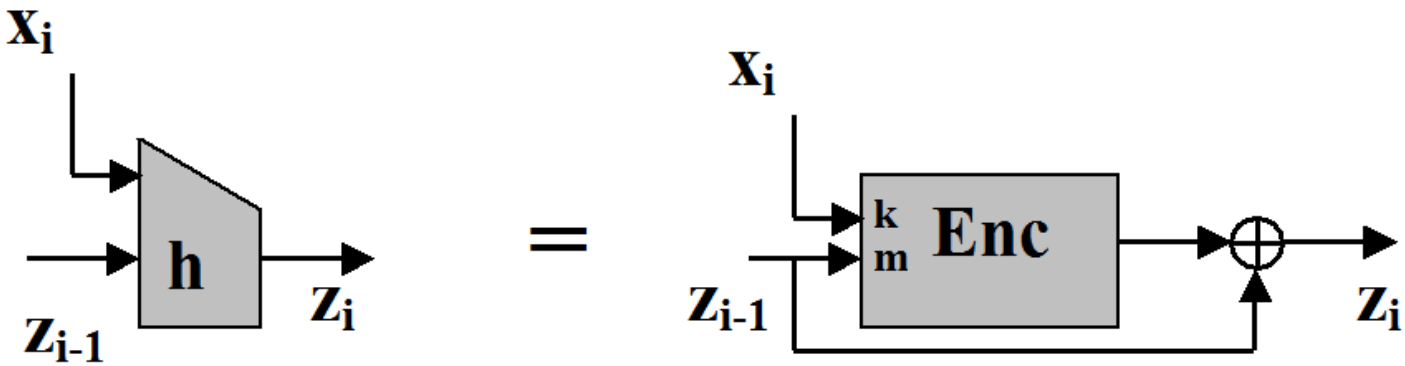
- Intuitiv, dacă există $x \neq x'$ a.î. $H(x) = H(x')$, atunci trebuie să existe $< z_{i-1}, x_i > \neq < z'_{i-1}, x'_i >$ a.î. $h(z_{i-1}||x_i) = h(z'_{i-1}||x'_i)$;
- Altfel spus, dacă se găsește o coliziune pentru H se găsește o coliziune și pentru h.

Transformarea Merkle-Damgård

- Rămâne să prezentăm o construcție pentru h;

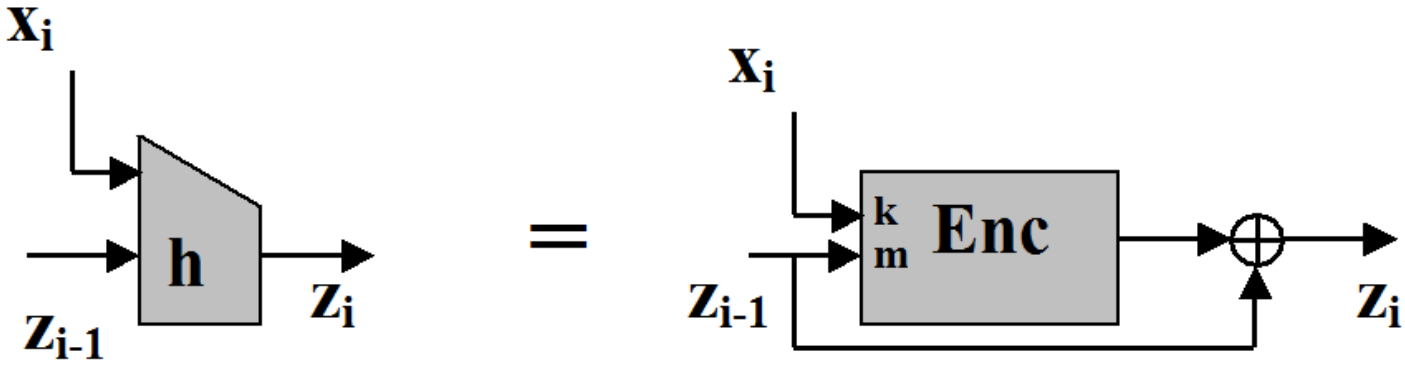
Transformarea Merkle-Damgård

- Rămâne să prezentăm o construcție pentru h;
- Construcția Davies-Meyer:



Transformarea Merkle-Damgård

- Rămâne să prezentăm o construcție pentru h;
- Construcția Davies-Meyer:



- *Enc* este un sistem bloc care criptează z_{i-1} cu cheia x_i:

$$h(z_{i-1}||x_i) = Enc_{x_i}(z_{i-1}) \oplus z_{i-1}$$

Exemple

- MD5 (Message Digest 5)
  - definit în 1991 de R.Rivest ca înlocuitor pentru MD4
  - produce o secvență hash de 128 biți
  - nesigur din 1996
  - utilizat în versiuni mai vechi de Moodle

Exemple

- MD5 (Message Digest 5)
  - definit în 1991 de R.Rivest ca înlocuitor pentru MD4
  - produce o secvență hash de 128 biți
  - nesigur din 1996
  - utilizat în versiuni mai vechi de Moodle

- SHA (Secure Hash Algorithm)
  - o familie de funcții hash publicate de NIST
  - SHA-0 și SHA-1 sunt nesigure
  - SHA-2 prezintă 2 variante: SHA-256 și SHA-512
  - SHA-3 adoptat în 2012 pe baza Keccak

Exercitii funcții hash

- Este $H(x) = x(mod\ N)$ o funcție hash rezistenta la coliziuni? Daca da, explicati de ce, daca nu, gasiți o coliziune.

Exercitii funcții hash

- Este $H(x) = x(mod\ N)$ o funcție hash rezistenta la coliziuni? Daca da, explicati de ce, daca nu, gasiți o coliziune.
- Raspuns: Funcția nu e rezistenta la coliziuni: pentru orice $x < N$ si orice $a \in \mathbb{N}$, x si $aN + x$ formeaza o coliziune.

Exercitii funcții hash

- Este $H(x) = x(mod\ N)$ o funcție hash rezistenta la coliziuni? Daca da, explicati de ce, daca nu, gasiți o coliziune.
- Raspuns: Funcția nu e rezistenta la coliziuni: pentru orice $x < N$ si orice $a \in \mathbb{N}$, x si $aN + x$ formeaza o coliziune.
- Este $H(x) = ax + b(mod\ N)$ cu $(a, N) = 1$ o functie hash rezistenta la coliziuni? Daca da, explicați de ce, daca nu, gasiți o coliziune.

Exercitii funcții hash

- Este $H(x) = x(mod\ N)$ o funcție hash rezistenta la coliziuni? Daca da, explicati de ce, daca nu, gasiți o coliziune.
- Raspuns: Funcția nu e rezistenta la coliziuni: pentru orice $x < N$ si orice $a \in \mathbb{N}$, x si $aN + x$ formeaza o coliziune.
- Este $H(x) = ax + b(mod\ N)$ cu $(a, N) = 1$ o functie hash rezistenta la coliziuni? Daca da, explicați de ce, daca nu, gasiți o coliziune.
- Raspuns: Funcția nu e rezistenta la coliziuni: cautam x_1 si x_2 asa incat $H(x_1) = H(x_2)$ adica $ax_1 + b = ax_2 + b(mod\ N)$. Obținem $a(x_1 - x_2) = 0(mod\ N)$. Cum (a,N) obținem ca $x_1 - x_2 = 0(mod\ N)$.