Cryptanalyse : fonctions de hachage

Pierre-Alain Fouque

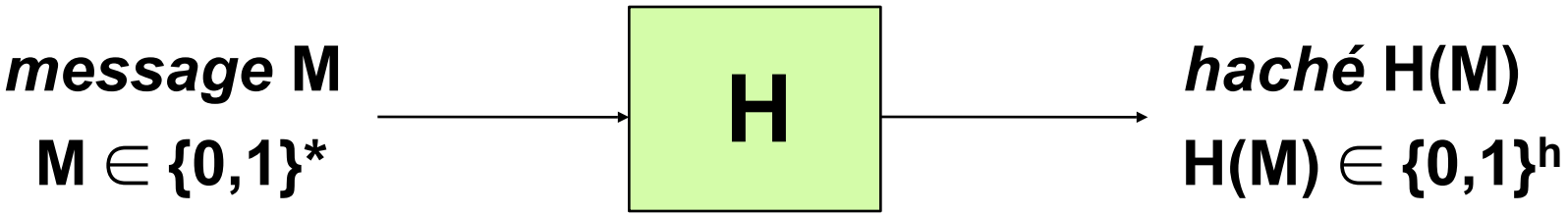
Equipe de Cryptographie

Ecole normale supérieure

Introduction

- Problématiques en crypto
  - Convertir des entrées de taille variable vers une taille fixe (ex : signature, intégrité)
  - Calcul d'empreintes uniques
  - Fonctions « à sens unique »
- Utilisation de fonctions de hachage

Définition



Une fonction de hachage H calcule une empreinte de h bits à partir d'un message arbitraire M

$$H : \{0,1\}^* \rightarrow \{0,1\}^h$$

Sécurité

Pour les fonctions de hachage sans clé :

- Résistance aux collisions

  trouver M1 et M2 tel que H(M1) = H(M2)

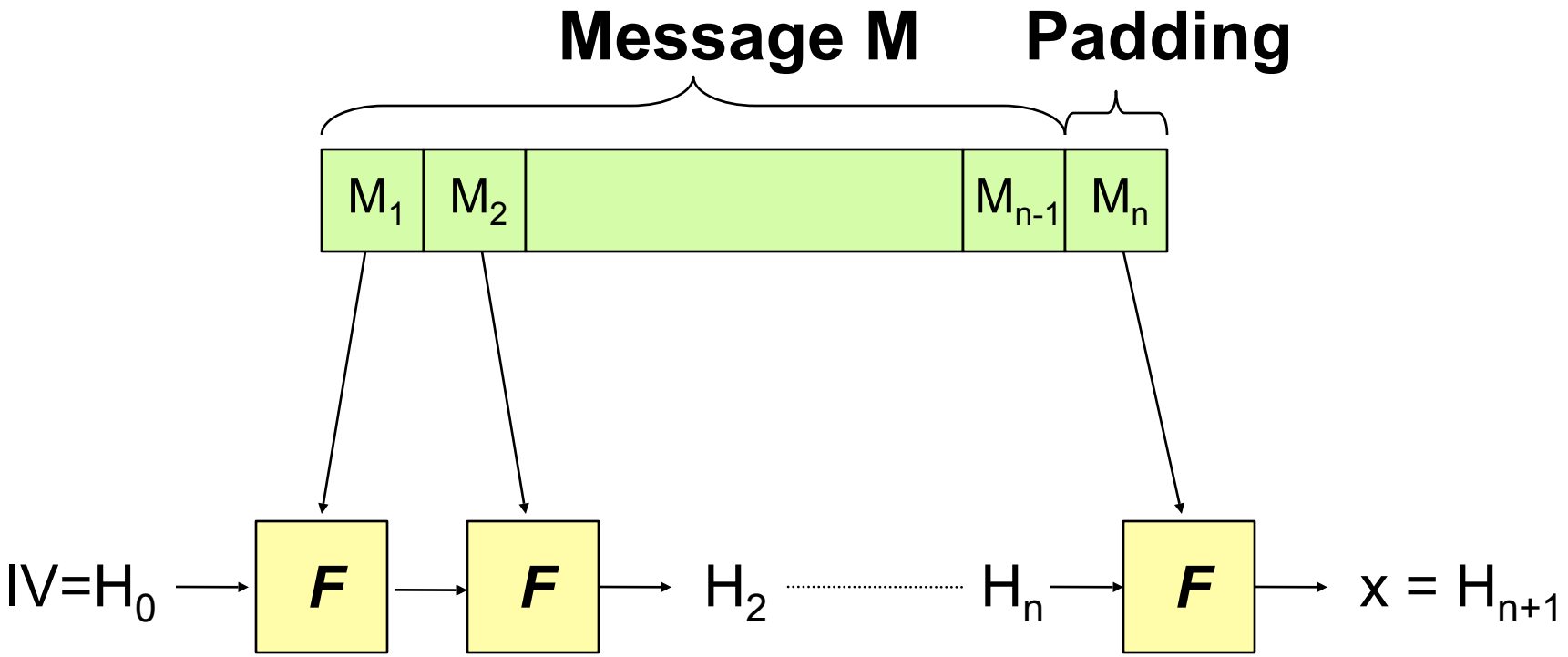
- Résistance aux secondes préimages

  avec M1 donné, trouver M2 tel que H(M1) = H(M2)

- Résistance aux préimages

  avec x donné, trouver M tel que H(M) = x

Fonctions itératives

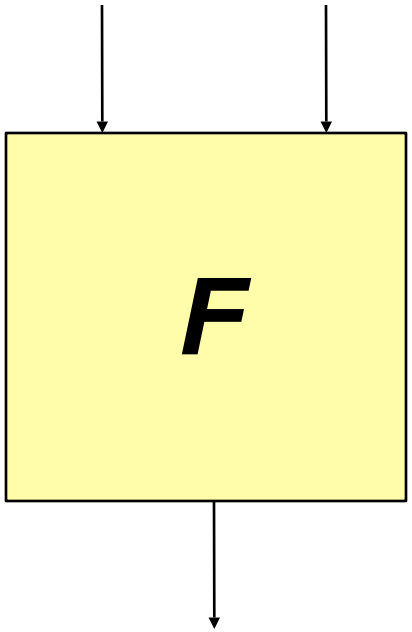


Construction de Merkle-Damgaard
F est généralement appelée Fonction de Compression

Fonction de compression

Haché intermédiaire
(h bits)

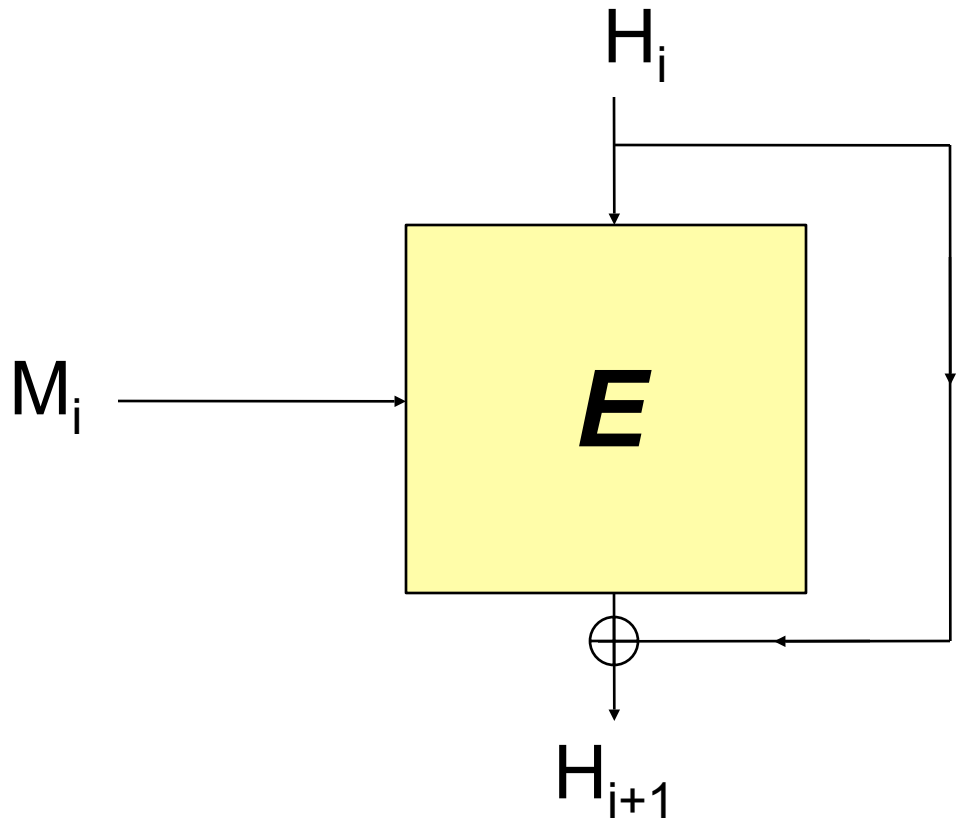
Bloc de message
(n bits)



$F : \{0,1\}^h \times \{0,1\}^n \rightarrow \{0,1\}^h$

Haché intermédiaire
(h bits)

Davies-Meyer



Fonctions classiques

Nom	n	h	Auteur
MD2	128	128	RSA
MD4	512	128	RSA
MD5	512	128	RSA
SHA-1	512	160	NIST
SHA-256	512	256	NIST
RIPEMD	512	128(+)	
HAVAL	512	128(+)	

Cryptanalyse

- Attaques génériques contre les fonctions de hachage
  - Collision
  - Seconde Préimage
- Attaques dédiées contre certaines fonctions
  - SHA-0
  - MD5

Attaque « naïves »

- Résistance aux collisions

  une attaque naïve coûte $2^{h/2}$

- Résistance aux secondes préimages

  une attaque naïve coûte 2^{h}

- Résistance aux préimages

  une attaque naïve coûte 2^{h}

Attaques en collision

- Choisir $2^{h/2}$ messages quelconques

$$x_1, \ldots, x_{h/2}$$

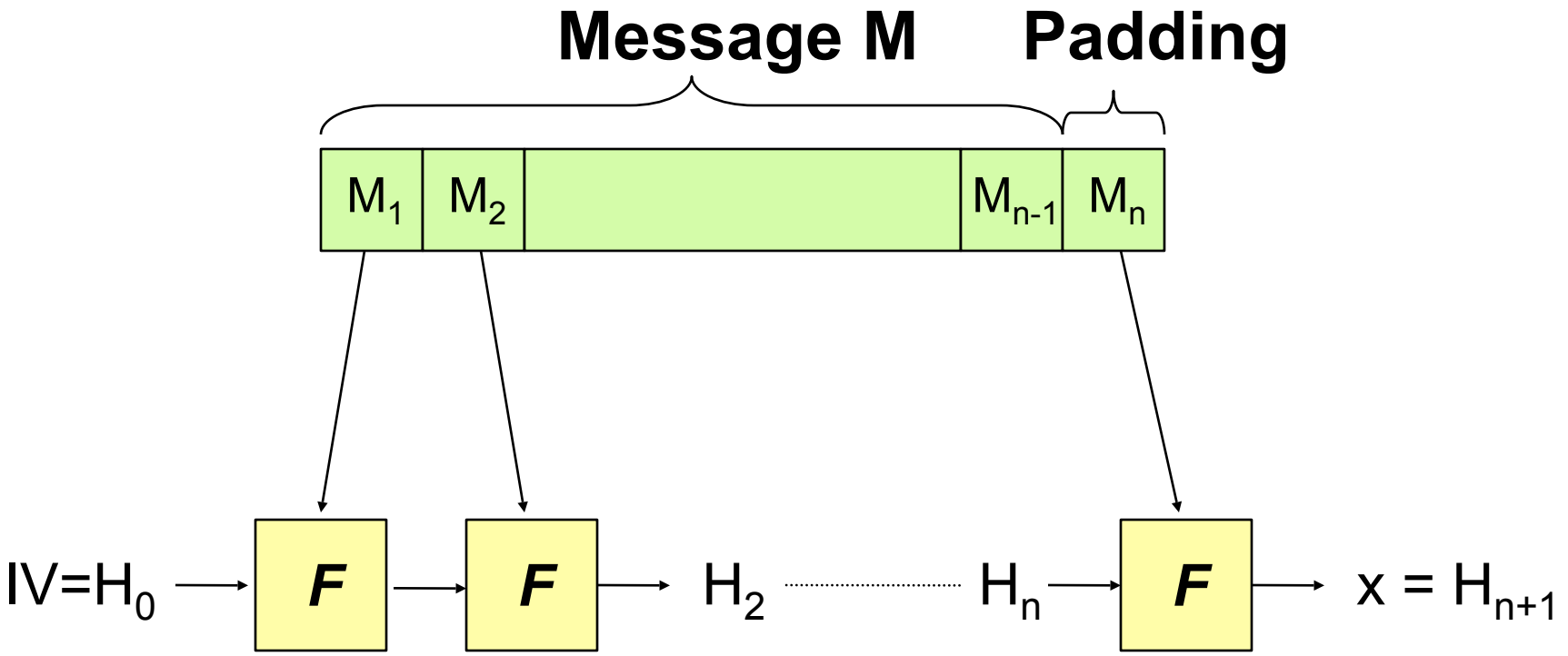
- Calculer les hachés correspondants

$$y_i = H(x_i)$$

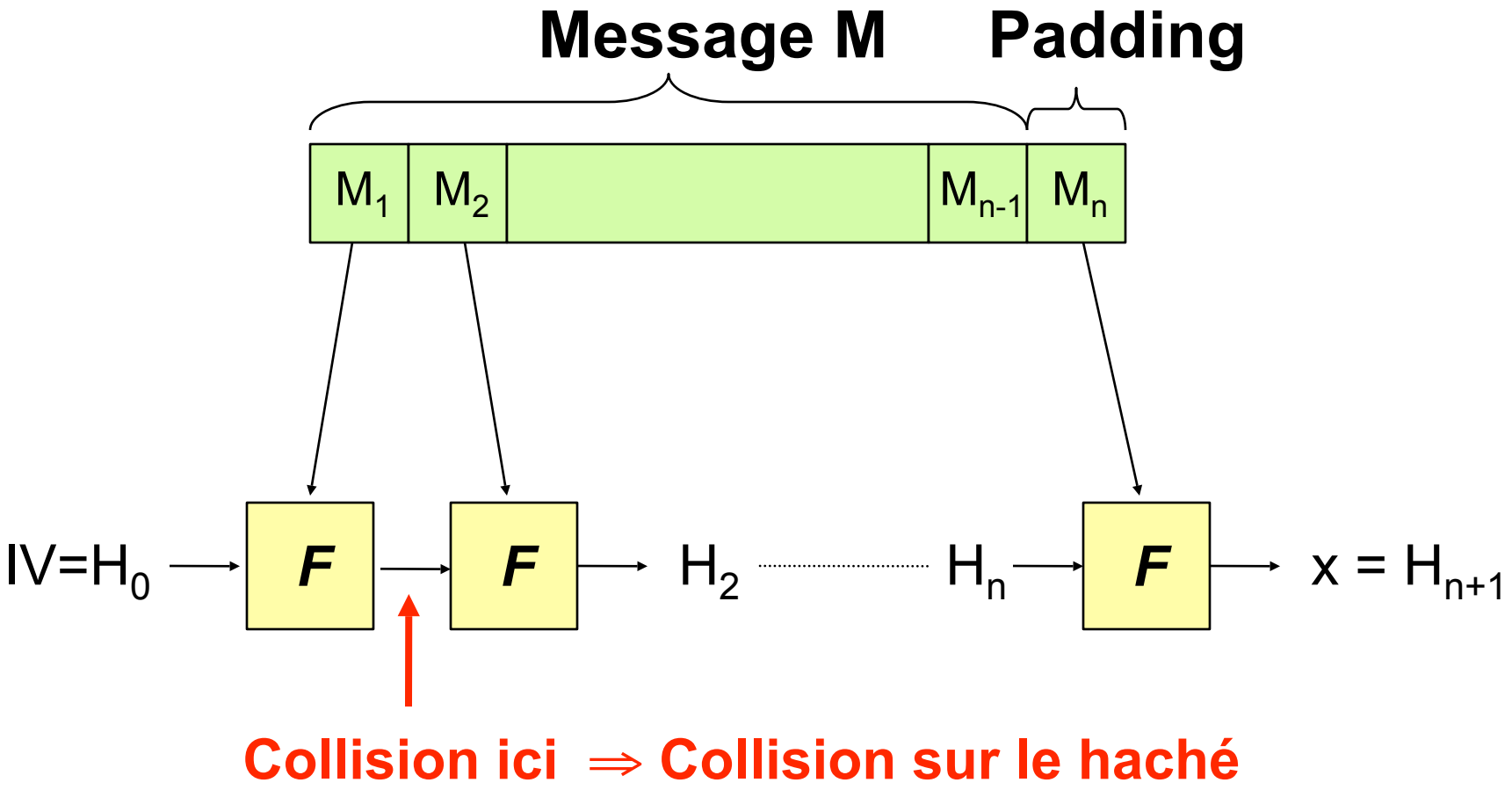
- Chercher (a,b) tel que

$$y_a = y_b$$

F ou H ?



F ou H ?


Message M
Padding
M1
M2
Mn-1
Mn
IV=H0
F
F
H2
Hn
F
x = Hn+1
Collision ici ⇒ Collision sur le haché

Problème

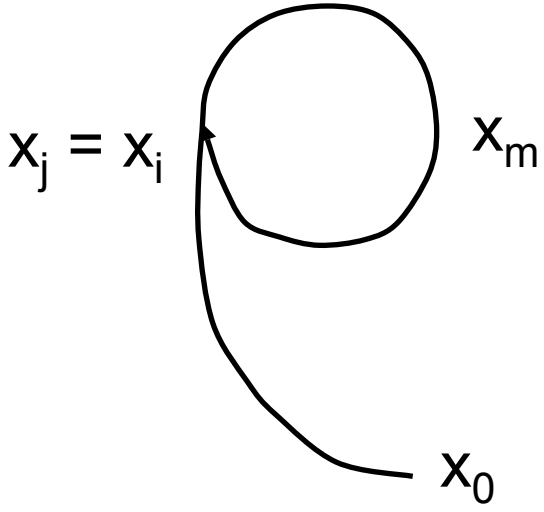
- Pour trouver une collision, il faut :
  - Stocker tous les y_i dans un tableau
  - Trier ce tableau
- Peu efficace (mémoire de $2^{h/2}$)
- Difficile de paralléliser ce calcul

Rho de Pollard

- Soit un x_0 un point quelconque
- Pour simplifier, on note G(x) = F(IV, x)
- Soit $x_{i+1} = G(x_i)$ pour $i=1\ldots m=2^{h/2}$
- $\exists$ (i,j) tel que $x_i = x_j$ avec bonne probabilité

Rho de Pollard

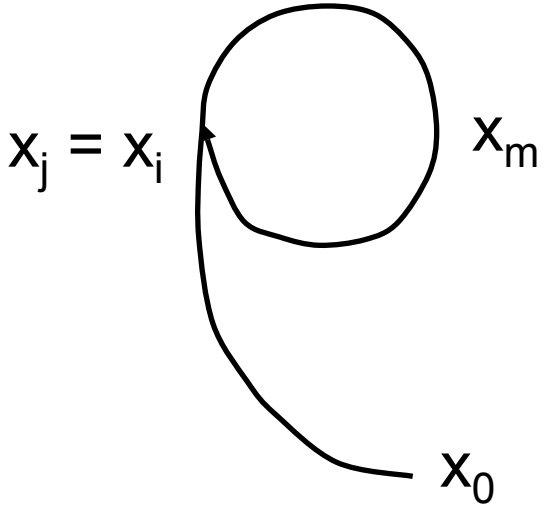
- Soit un x_0 un point quelconque
- Pour simplifier, on note G(x) = F(IV, x)
- Soit $x_{i+1} = G(x_i)$ pour $i=1\ldots m=2^{h/2}$
- ∃ (i,j) tel que $x_i = x_j$ avec bonne probabilité



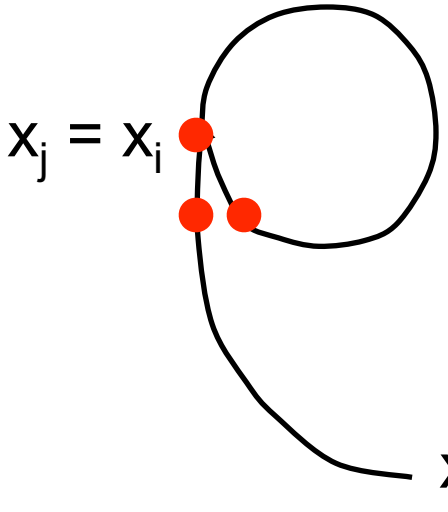
Rho de Pollard

- Soit un x_0 un point quelconque
- Pour simplifier, on note G(x) = F(IV, x)
- Soit $x_{i+1} = G(x_i)$ pour $i=1\ldots m=2^{h/2}$
- ∃ (i,j) tel que $x_i = x_j$ avec bonne probabilité

Soit τ = j - i



Rho de Pollard



$$G(x_{i-1}) = G(x_{j-1})$$

$$x_{i-1} \neq x_{j-1}$$

- On a une collision pour la fonction G (et donc pour la fonction de compression F)
- Il suffit de connaître i et τ

Attaque

1. Calculer x_m
2. Calculer $G^t(x_m)$ jusqu'à observer

$$G^\tau(x_m) = x_m$$

3. Calculer x_τ
4. Calculer $G^t(x_0)$ et $G^t(x_\tau)$ jusqu'à observer

$$G^i(x_0) = G^i(x_\tau)$$

5. Collision entre $a = G^{i-1}(x_0)$ et $b = G^{i-1}(x_\tau)$

Attaque

1. Calculer x_m complexité $m = 2^{h/2}$
2. Calculer $G^t(x_m)$ jusqu'à observer

$$G^\tau(x_m) = x_m$$

3. Calculer x_τ
4. Calculer $G^t(x_0)$ et $G^t(x_\tau)$ jusqu'à observer

$$G^i(x_0) = G^i(x_\tau)$$

5. Collision entre $a = G^{i-1}(x_0)$ et $b = G^{i-1}(x_\tau)$

Attaque

1. Calculer x_m complexité $m = 2^{h/2}$
2. Calculer $G^t(x_m)$ jusqu'à observer

$$G^\tau(x_m) = x_m \quad \text{complexité } \tau \leq m$$

3. Calculer x_τ
4. Calculer $G^t(x_0)$ et $G^t(x_\tau)$ jusqu'à observer

$$G^i(x_0) = G^i(x_\tau)$$

5. Collision entre $a = G^{i-1}(x_0)$ et $b = G^{i-1}(x_\tau)$

Attaque

1. Calculer x_m — complexité $m = 2^{h/2}$
2. Calculer $G^t(x_m)$ jusqu'à observer

$$G^\tau(x_m) = x_m$$ complexité $\tau \leq m$

3. Calculer x_τ
4. Calculer $G^t(x_0)$ et $G^t(x_\tau)$ jusqu'à observer

complexité $i \leq m$

$$G^i(x_0) = G^i(x_\tau)$$

5. Collision entre $a = G^{i-1}(x_0)$ et $b = G^{i-1}(x_\tau)$

Analyse

La complexité de cette attaque est

$$T = O(2^{h/2})$$

$$M \approx 0$$

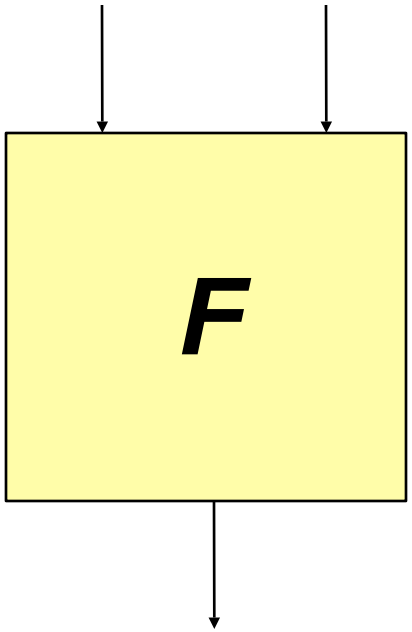
Attaques en préimage

- Entrée : haché x
- Calculer $h_i = H(m_i)$ pour des messages aléatoires
- $h_i = x$ avec probabilité 2^{-h}
- Donc, après 2^h messages, on s'attend à trouver une préimage

Attaques en Préimage contre F

Haché intermédiaire
(h bits)

Bloc de message
(n bits)



$F : \{0,1\}^h \times \{0,1\}^n \rightarrow \{0,1\}^h$

Haché intermédiaire
(h bits)

Attaques en Préimage contre F

$$F(H_i, M_i) = H_{i+1}$$

- Attaque en préimage contre F
  - Entrée = H_{i+1} et H_i
  - Trouver un M_i
- Attaque en pseudo-préimage contre F
  - Entrée = H_{i+1}
  - Trouver un couple (H_i, M_i)

Attaques en Préimage contre F

- Les 2 attaques se transforment en attaques en préimage contre H
- Cela justifie pourquoi il ne faut pas prendre une fonction F simple à inverser, au sens
  - Entrée = M_i et H_{i+1}
  - Retrouver H_i
  - Attaque en pseudo-préimage triviale !

Attaque en préimage H

- Trouver m tq H(m)=h
- Calculer 2^t préimages $n_i=(m_i,h_i)$ si on peut en calculer autant à partir de h
- Calculer 2^{n-t} hachés h'_i en utilisant les messages m'_i à partir de IV
- Trouver une collision i_0 tq $h_{i0}=h'_{i0}$ et $m=m'_{i0}||$ m_{i0} est une préimage de h par H
- Pb: gérer le padding ...

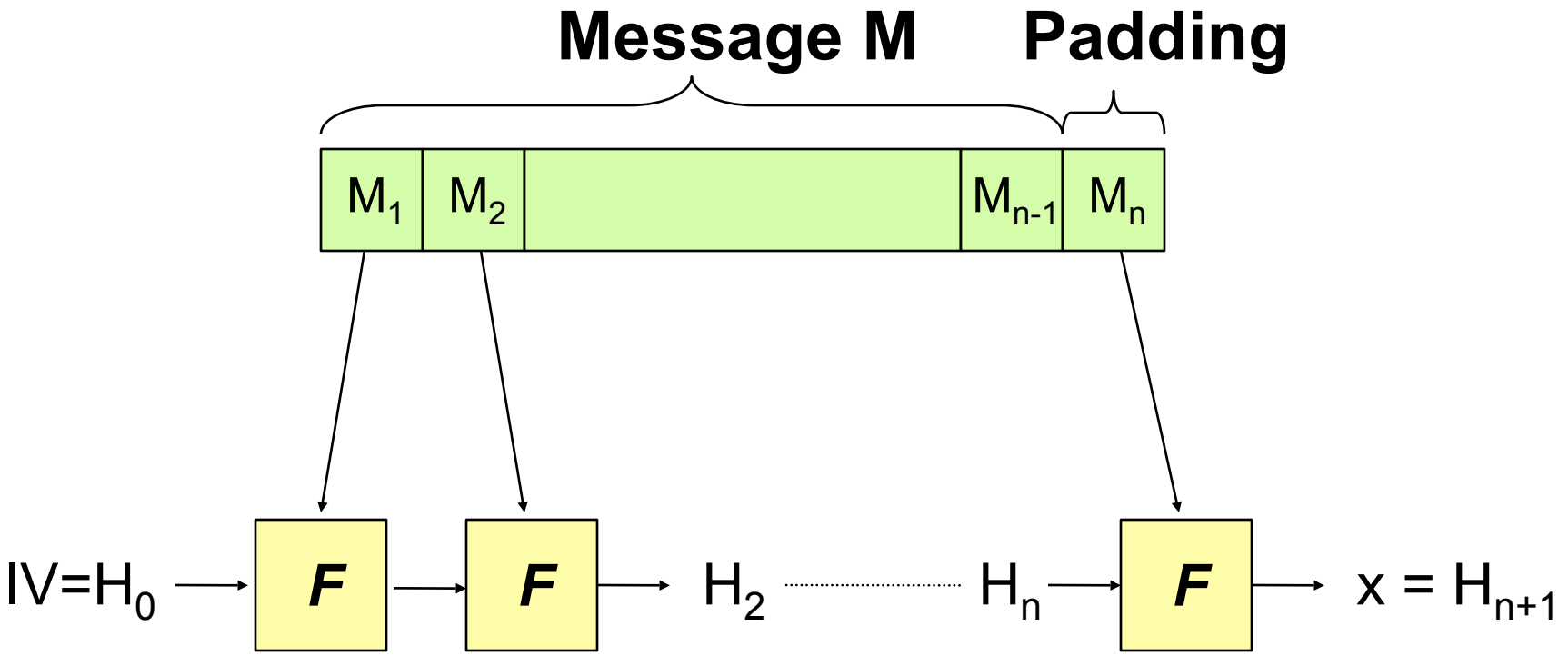
Multicollision

- Obtenir plusieurs messages $m_0,\ldots,m_n$ tq $H(m_0)=H(m_1)=\ldots=H(m_n)=h$ pour un h pas forcément choisi
- Facile sur Merkle-Damgard
- Application: Collision sur la fonction $H(m)=H1(m)||H2(m)$ pour deux fonctions de hachage dont l'une est un Merkle-Damgard

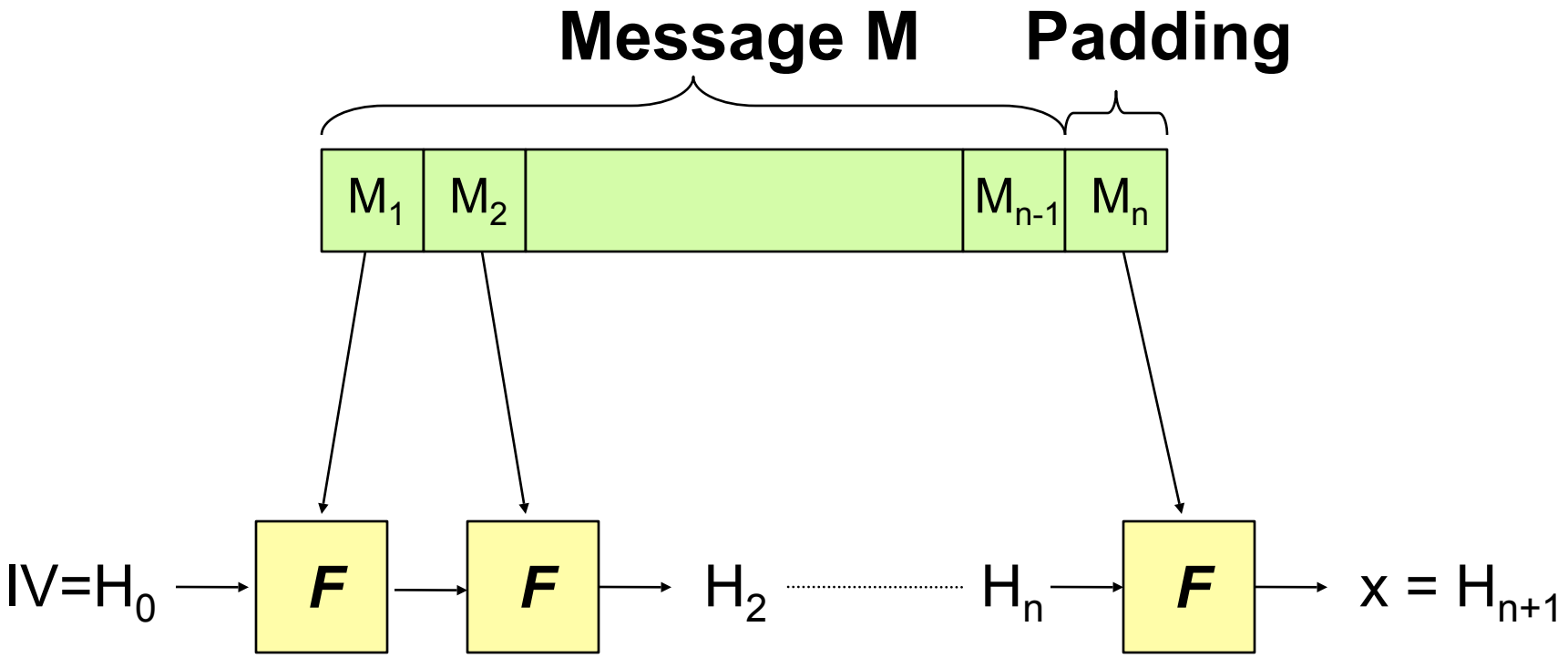
Attaques en seconde préimage

- Entrée : message M et son haché x = H(M)
- Calculer $h_i = H(m_i)$ pour des messages aléatoires
- $h_i = x$ avec probabilité 2^{-h}
- Donc, après 2^h messages, on s'attend à trouver une seconde préimage

Peut-on faire mieux?



Peut-on faire mieux?



On connaît les H_i pour le message M

Attaque pour des messages longs

- On peut essayer d'obtenir l'égalité sur un haché intermédiaire
- Exemple : message de taille $2^{h/2}$ blocs
- Calculer $2^{h/2}$ fois :

$$y_i = F(IV, x_i) \text{ pour des } x_i \text{ aléatoires}$$

- Si $y_a = H_b$, on obtient la seconde préimage

$$M' = (x_a, M_b, M_{b+1}, \ldots, M_n)$$

Problème

$$M = (M_1, \ldots, M_n)$$

et $M' = (x_a, M_b, M_{b+1}, \ldots, M_n)$

n'ont pas la même longueur

⇒ Incompatibilité du dernier bloc !!

⇒ Intérêt de mettre un padding dépendant de la longueur

Nouveaux résultats

Kelsey/Schneier 2004

- Possibilité de faire un compromis avec la taille du message
- Utilisation de points fixes pour accorder les longueurs
- Seconde préimage en 2^{h-k}
- Message de taille 2^{k}

Seconde préimage

- Soit M un message de taille 2^k
- construire un message expandable M'
- c'est un message qui peut avoir une taille variable et avoir des hachés intermédiaires identiques
- connecté M' à M en 2^{n-k} avec un bloc aléatoire
- mettre le message M' à la bonne taille

Attaques dédiées

- SHA-0
- MD5

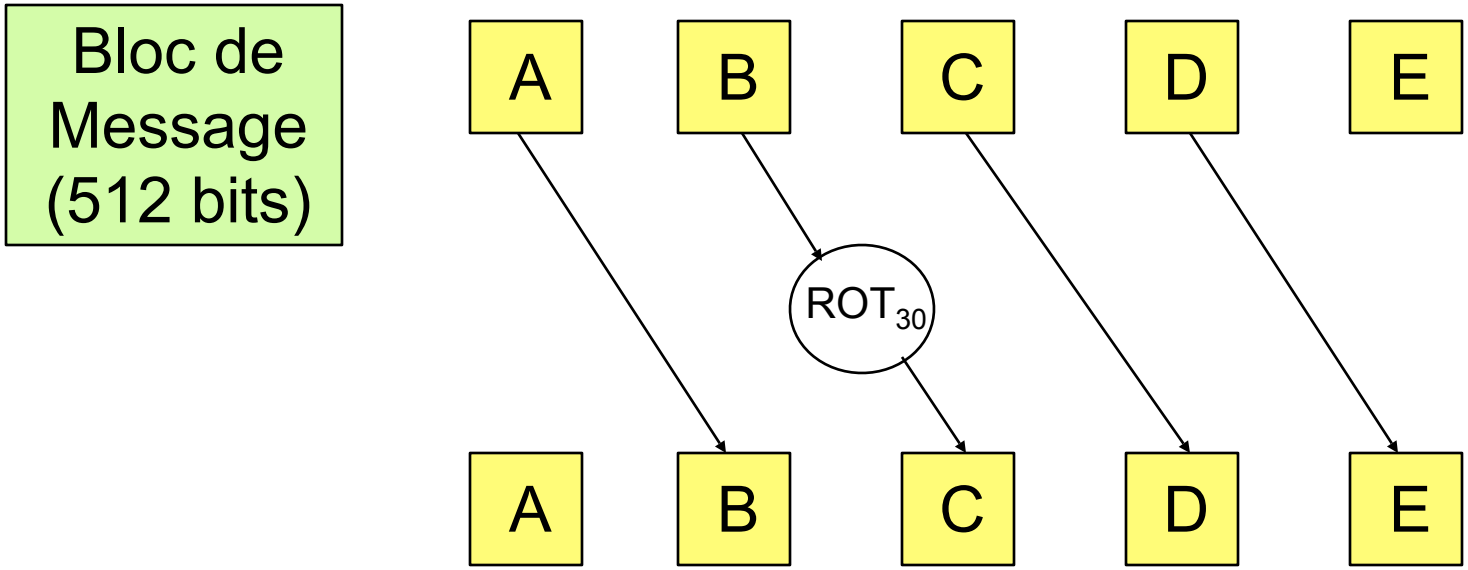
SHA

- SHA-0 publié par le NIST en 1993
- Modifié en 1995 : version SHA-1
- Nouvelle version publiée en 2002, SHA-256

- Fonction de hachage itérative
- Fonction de compression = Feistel généralisé

SHA

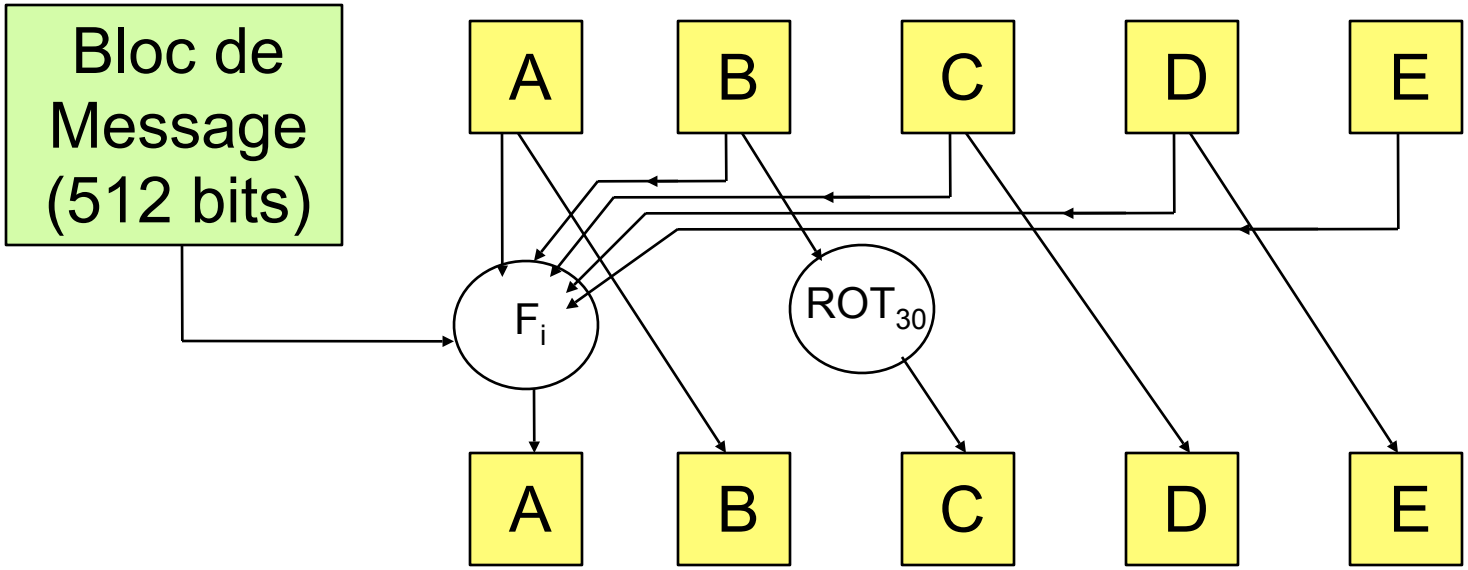
La fonction de compression est constituée de 80 tours :



Tous les objets manipulés font 32 bits

SHA

La fonction de compression est constituée de 80 tours :



Tous les objets manipulés font 32 bits

SHA

5 mots de 32 bits A,B,C,D,E

A = Iv[0]; B = Iv[1]; C = Iv[2]; D = Iv[3]; E = Iv[4];

Pour i = 1, … , 80

{

A = (A<<<5) + f_i(B,C,D) + E + Cst[i] + W[i]

B = A

C = (B<<<30)

D = C

E = D

}

Dérivé du bloc de message

SHA

- La fonction $f_i(B,C,D)$ est une fonction booléenne bit par bit, choisie parmi IF, XOR, MAJORITE
- Les mots de 32 bits W[i] sont dérivés des mots du bloc de message $(M[i])_{i=0,\ldots,15}$ par

$$W[i] = M[i] \text{ pour } i=0 \ldots 15$$

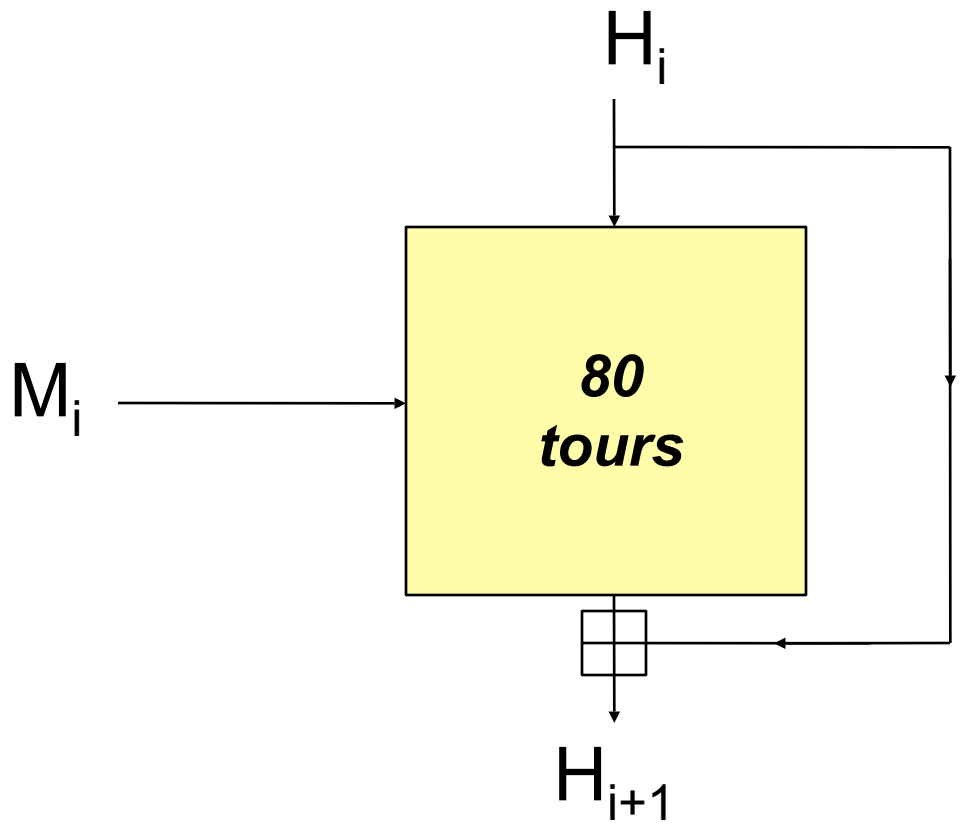
$$W[i] = (W[i-3] \oplus W[i-8] \oplus W[i-14] \oplus W[i-16])_{<<<1}$$

$$\text{pour } i \geq 16$$

Différence avec SHA-0

SHA

Le tour élémentaire est inversible
(Feistel généralisé) ! On applique Davies - Meyer



Attaque différentielle

- Remplacer le bloc de message

$$M = (M_1, \ldots, M_{16})$$

- par $M \oplus \Delta$ où

$$\Delta = (\Delta_1, \ldots, \Delta_{16})$$

- Par extension, on note Δ_i la différence sur W_i
- On peut facilement la calculer car
  - $W_i = M_i$ pour i=0 … 15
  - $W_i = (W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16})$

Observation de base

$$A = (A<<<5) + f_i(B,C,D) + E + Cst[i] + W_i$$

- Si f_i et + sont des XOR, alors
- Si $\Delta_i = 2^{31}$ (différence sur le bit de poids fort)

la différence sur A apparaît en poids fort

- Si $\Delta_{i+1} = 2^4$ (différence sur le 5ème bit)

la différence ne se propage pas au tour suivant

- etc …

Différences

Tour	Δ_i	ΔA	ΔB	ΔC	ΔD	ΔE	Proba
1	2^{31}	2^{31}	0	0	0	0	1
2	2^{4}	0	2^{31}	0	0	0	0.5
3	0	0	0	2^{29}	0	0	0.5
4	0	0	0	0	2^{29}	0	0.5
5	0	0	0	0	0	2^{29}	0.5
6	2^{29}	0	0	0	0	0	0.5

Différences

Tour	Δ_i	ΔA	ΔB	ΔC	ΔD	ΔE	Proba
1	2^{31}	2^{31}	0	0	0	0	1
2	2^{4}	0	2^{31}	0	0	0	0.5
3	0	0	0	2^{29}	0	0	0.5
4	0	0	0	0	2^{29}	0	0.5
5	0	0	0	0	0	2^{29}	0.5
6	2^{29}	0	0	0	0	0	0.5

COLLISION

Problème

- Les Δ_i sur les premiers tours se propagent à cause de

$$W_i = (W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16})$$

$$\Delta_i = (\Delta_{i-3} \oplus \Delta_{i-8} \oplus \Delta_{i-14} \oplus \Delta_{i-16})$$

- Répéter plusieurs fois la différence sur 6 tours en restant compatible avec cette expansion
- Cette expansion agit « bit à bit » (faux dans le cas de SHA-1)

Solution

- On peut trouver une suite $\Delta = (\Delta_i)_{i=1\ldots80}$
  - vérifiant l'expansion
  - collisions locales sur 6 tours ($p=2^{-5}$)
- Essayer des messages aléatoires jusqu'à trouver une collision
  - Probabilité totale < 2^{-80}
  - Mais les 16 premiers tours sont « gratuits » !

Résultats

- 1998 : Chabaud-Joux
  - attaque en 2^{61} opérations
  - Pas faisable en pratique
- 2004 : Biham-Chen
  - Amélioration en 2^{56} («neutral bits»)
- 2004 : Joux
  - Amélioration en 2^{51} (structure itérative)
  - Collision trouvée en août 2004

Chabaud-Joux

- Chabaud et Joux ont proposé une attaque sur une version linéaire de SHA-0
- découverte de bon vecteur de perturbation indiquant où placer les collisions locales
- 2^{16} vecteurs possibles mais seulement 128 vérifiant la contrainte de ne pas avoir de différences dans les 5 premiers et 5 dernières étapes

Approche Probabiliste

- Ensuite, il faut introduire les fonction f_i (IF et MAJ) et l'addition mod 2^{32}
- CJ ont estimé la probabilité que ce qui se passe soit différent de l'attaque sur la fonction linéaire
- Attaque en 2^{61}

IF et XOR

- IF(x,y,z)=y si x=1, 0 sinon et x et x* diffèrent d'1 bit
- Pr(IF(x,y,z)=IF(x*,y,z))=0.5
- Pr(IF(x,y,z)=IF(x,y*,z))=0.5
- Pr(IF(x,y,z)=IF(x,y,z*))=0.5
- Pr(IF(x,y,z)=IF(x,y*,z*))=0
- Pr(XOR(x,y,z)=XOR(x*,y,z))=0
- Pr(XOR(x,y,z)=XOR(x,y*,z))=0
- Pr(XOR(x,y,z)=XOR(x,y,z*))=0
- Pr(XOR(x,y,z)=XOR(x,y*,z*))=1

Approche probabiliste

- Dans les 3 premiers cas, IF se comporte comme XOR avec proba 0.5 alors que dans le dernier cas, IF ne se comportera jamais comme XOR
- Si on rajoute les probabilités des retenues, on peut calculer les complexités

Wang attaque déterministe

- Au lieu d'avoir une approche probabiliste, Wang montre qu'on peut contrôler ce qui se passe dans la fonction de hachage
- Manière plus fine de contrôler en prenant une différence signée $\delta(X,X')=(x_0-x'_0,x_1-x'_1,\ldots,x_{32}-x'_{32})$
- Enfin, même si on impose trop de conditions, celles du début sont facile à remplir en modifiant le message

Conditions sur IF

- IF(X,Y,Z)=(X∧Y)∨(¬X∧Z) - if bit à bit

- if(x,y,z)=if(¬x,y,z) si y=z
- if(x,y,z)=¬if(¬x,y,z) si y=¬z
- if(x,y,z)=if(x,¬y,z) si x=0
- if(x,y,z)=¬if(x,¬y,z) si x=1
- if(x,y,z)=if(x,y,¬z) si x=1
- if(x,y,z)=¬if(x,y,¬z) si x=0
- if(x,y,z)≠if(x,¬y,¬z) dans les autres cas ...

Attaque déterministe

- Enumérer toutes les conditions pour que les fonctions de tours aient un comportement en adéquation avec le modèle
- La complexité est simplement le nombre de conditions

Approche signée

- Différence = soustraction modulo 2^{32}
- Attaque tire parti du comportement déterministe de la fonction IF
- Exprimer le comportement de if en fonction de la différentielle: préserver ou absorber le signe de la différentielle: Si $\delta x \neq 0$, $\delta y = \delta z = 0$, $\delta_{if}(x,y,z) = +\delta x$ si $y = \neg z = 1$, $\delta_{if}(x,y,z) = -\delta x$ si $y = \neg z = 0$, et $\delta_{if}(x,y,z) = +0$ si $y = z$

Attaques multi-blocs

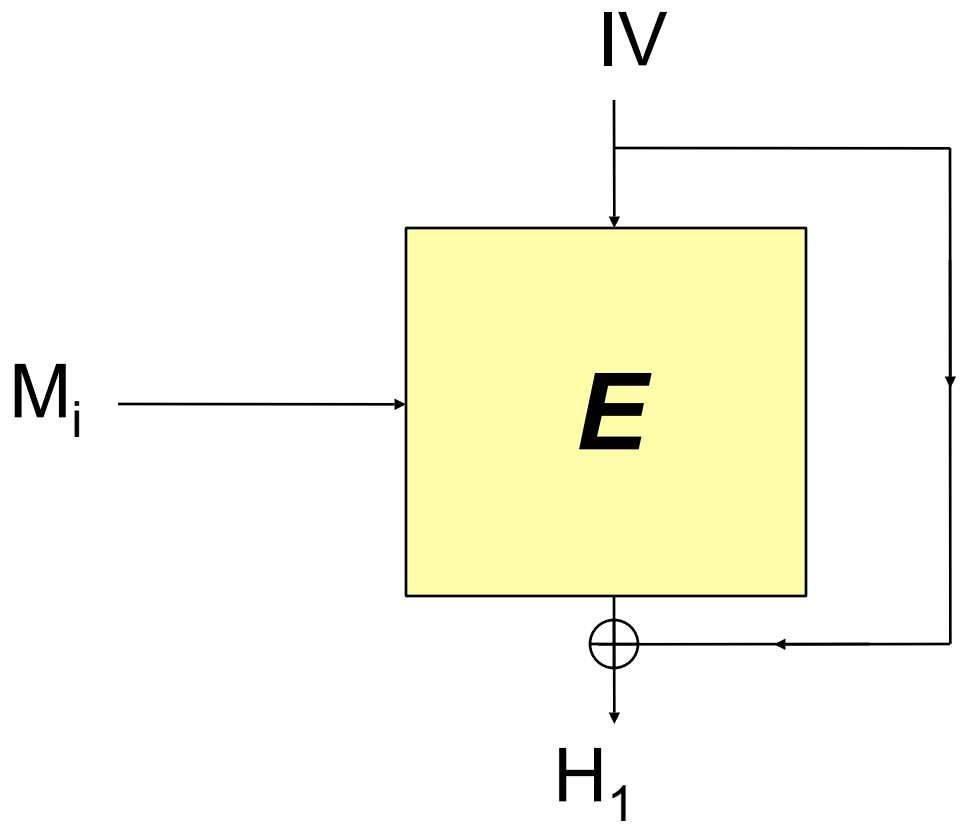
- Utilisation de 2 tours de la fonction de compression

  $F(IV, M_1) = H_1$ $\quad\quad$ $F(IV, M_1') = H_1 \oplus \Delta_1$

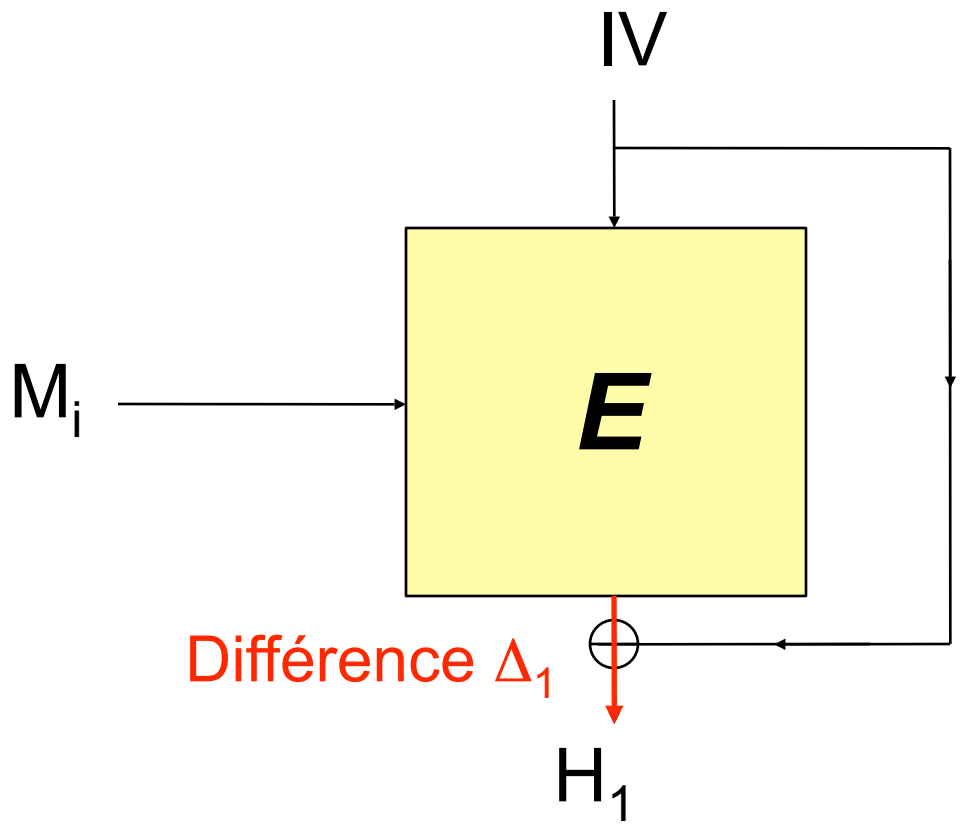
  $F(H_1, M_2) = H_2$ $\quad\quad$ $F(H_1 \oplus \Delta_1, M_2') = H_2$

- La différence est compensée par l'addition finale dans la fonction de compression

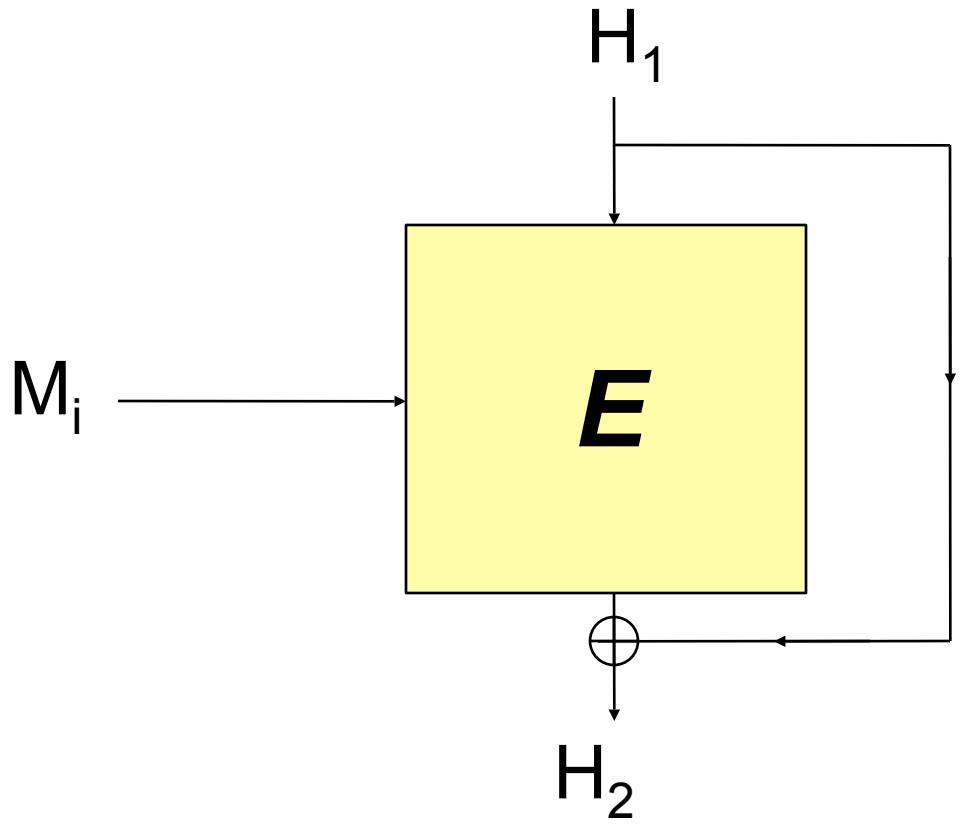
Rappel : Davies-Meyer



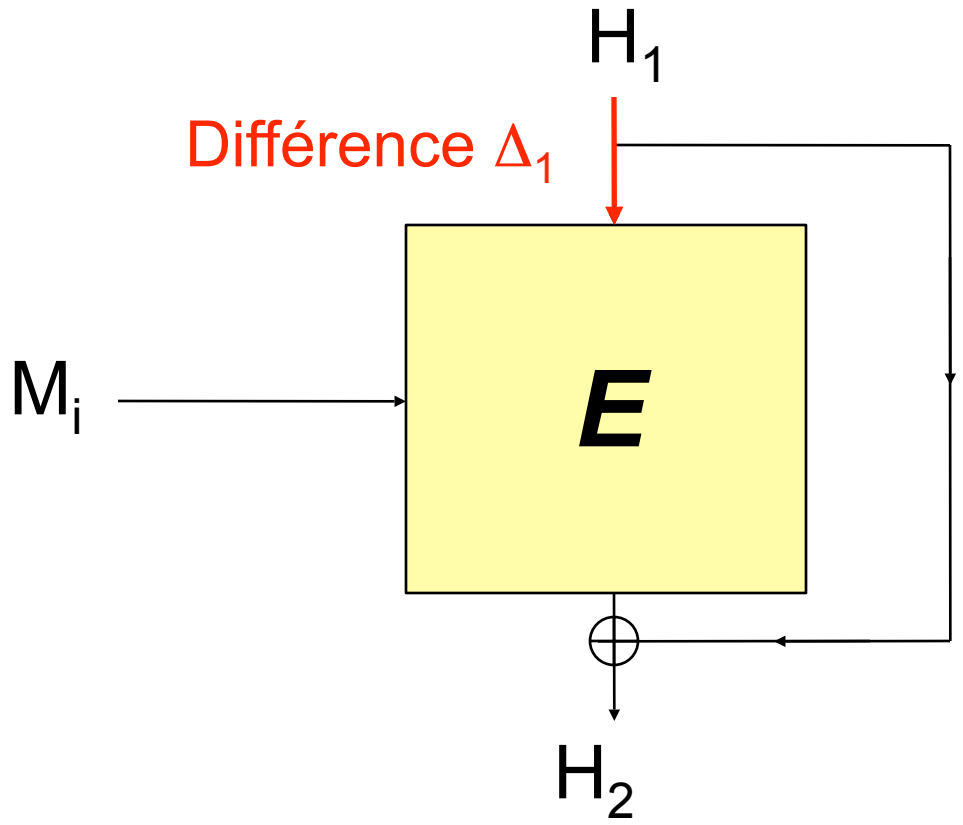
Rappel : Davies-Meyer



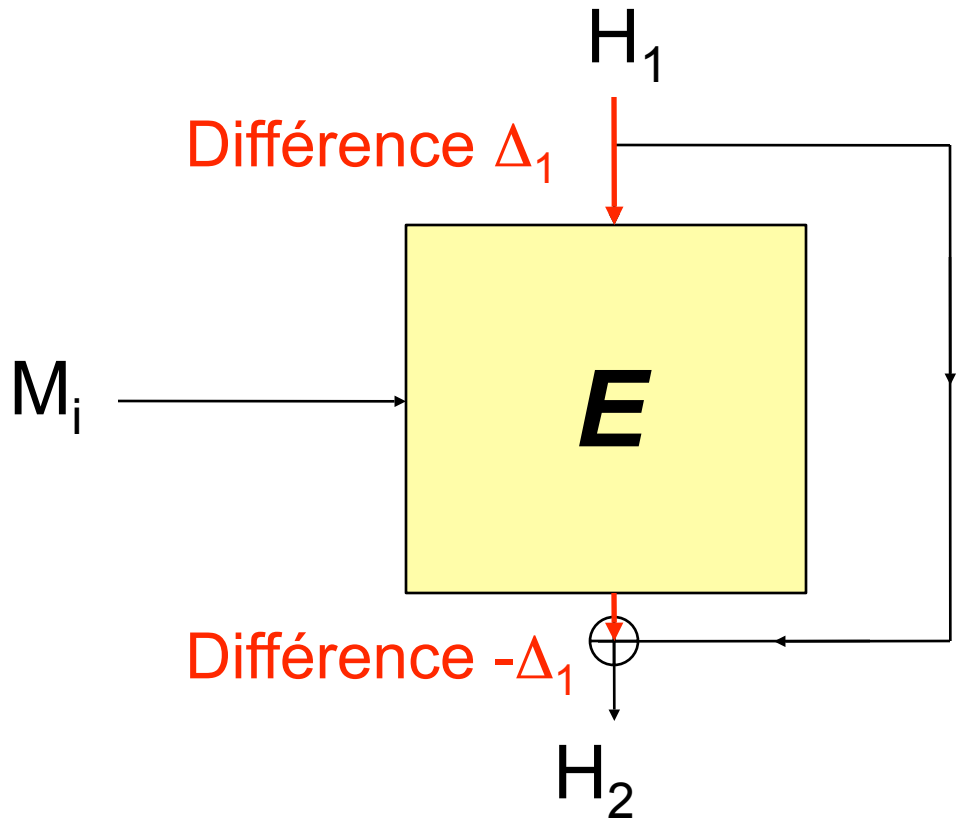
Rappel : Davies-Meyer



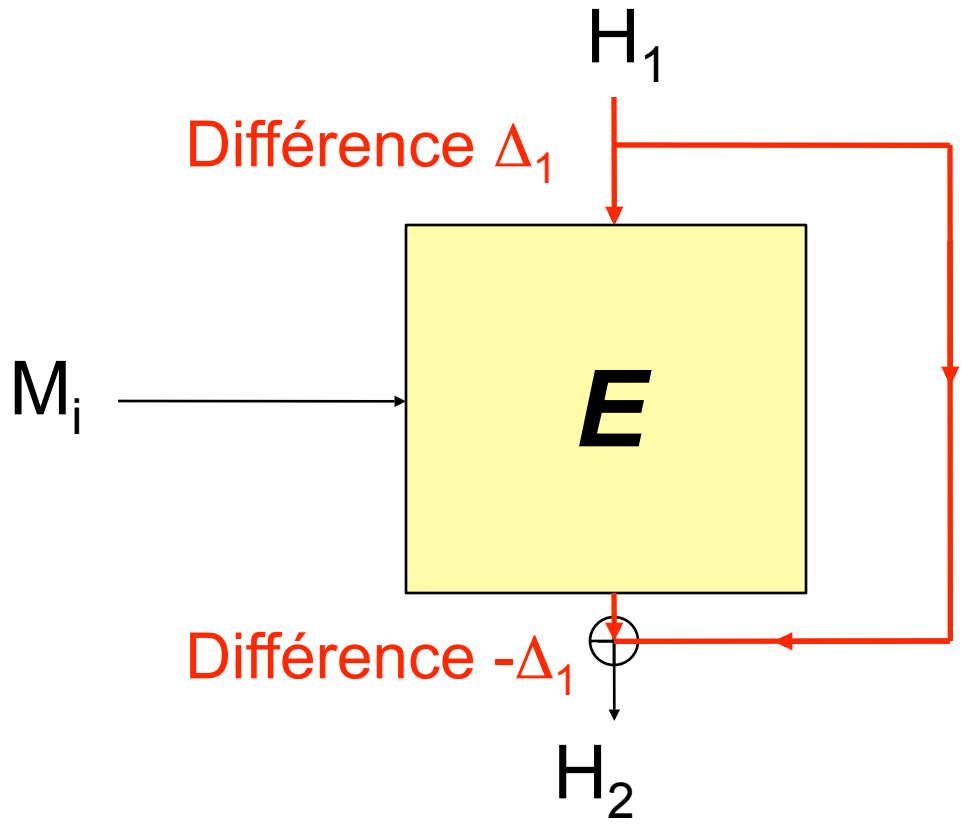
Rappel : Davies-Meyer



Rappel : Davies-Meyer



Rappel : Davies-Meyer



Première collision SHA-0

- On peut relâcher la contrainte que les 5 premières étapes et 5 dernières étapes du vecteur de perturbation n'est pas de différences
- Meilleur vecteur
- Ensuite, on fait une recherche d'une suite de différences qui parte de l'IV et aboutisse à une collision

Résultats

Thursday 12th, August 2004

We are glad to announce that we found a collision for SHA-0.

First message (2048 bits represented in hex):

a766a602 b65cffe7 73bcf258 26b322b3 d01b1a97 2684ef53 3e3b4b7f 53fe3762
24c08e47 e959b2bc 3b519880 b9286568 247d110f 70f5c5e2 b4590ca3 f55f52fe
effd4c8f e68de835 329e603c c51e7f02 545410d1 671d108d f5a4000d cf20a439
4949d72c d14fbb03 45cf3a29 5dcda89f 998f8755 2c9a58b1 bdc38483 5e477185
f96e68be bb0025d2 d2b69edf 21724198 f688b41d eb9b4913 fbe696b5 457ab399
21e1d759 1f89de84 57e8613c 6c9e3b24 2879d4d8 783b2d9c a9935ea5 26a729c0
6edfc501 37e69330 be976012 cc5dfe1c 14c4c68b d1db3ecb 24438a59 a09b5db4
35563e0d 8bdf572f 77b53065 cef31f32 dc9dbaa0 4146261e 9994bd5c d0758e3d

Second message:

a766a602 b65cffe7 73bcf258 26b322b1 d01b1ad7 2684ef51 be3b4b7f d3fe3762
a4c08e45 e959b2fc 3b519880 39286528 a47d110d 70f5c5e0 34590ce3 755f52fc
6ffd4c8d 668de875 329e603e 451e7f02 d45410d1 e71d108d f5a4000d cf20a439
4949d72c d14fbb01 45cf3a69 5dcda89d 198f8755 ac9a58b1 3dc38481 5e4771c5
796e68fe bb0025d0 52b69edd a17241d8 7688b41f 6b9b4911 7be696f5 c57ab399
a1e1d719 9f89de86 57e8613c ec9e3b26 a879d498 783b2d9e 29935ea7 a6a72980
6edfc503 37e69330 3e976010 4c5dfe5c 14c4c689 51db3ecb a4438a59 209b5db4
35563e0d 8bdf572f 77b53065 cef31f30 dc9dbae0 4146261c 1994bd5c 50758e3d

Common hash value (can be found using for example "openssl sha file.bin"
after creating a binary file containing any of the messages)

c9f160777d4086fe8095fba58b7e20c228a4006b

Question ouverte

- Application de la technique à SHA-1 ?
- Problème du <<<1

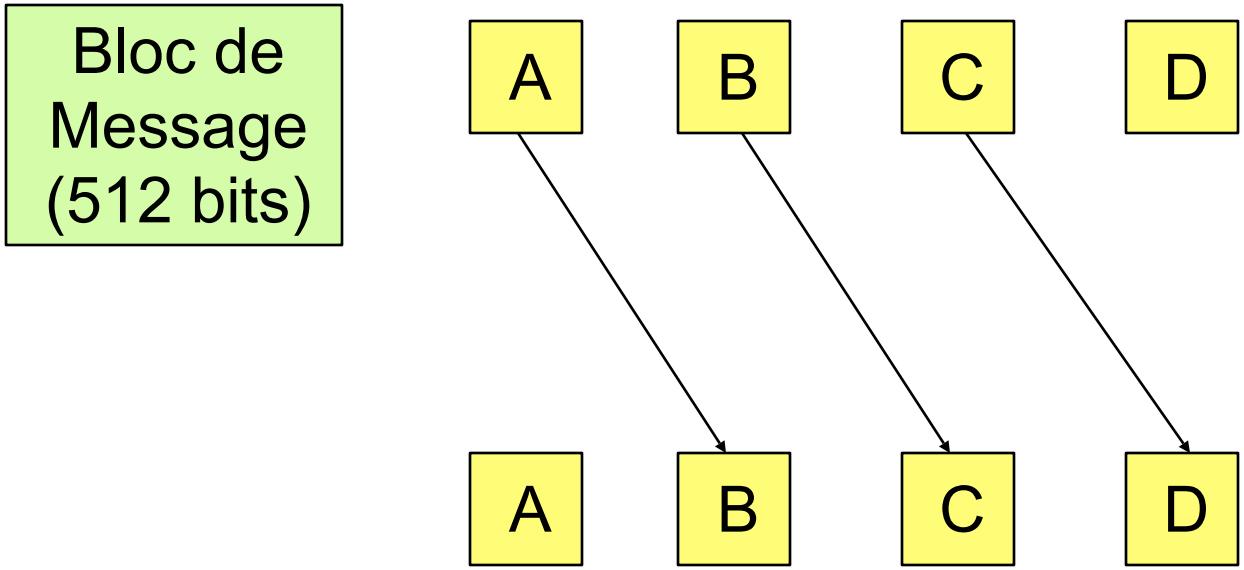
$$W[i] = (W[i-3] \oplus W[i-8] \oplus W[i-14] \oplus W[i-16])_{<<<1}$$

MD5

- Version « renforcée » de MD4 (plus de tours)
- MD4 et MD5 développés par Rivest (RSA Labs)
- MD4 proposé en 1990
- MD5 proposé en 1991
- Très similaire à SHA-1

MD5

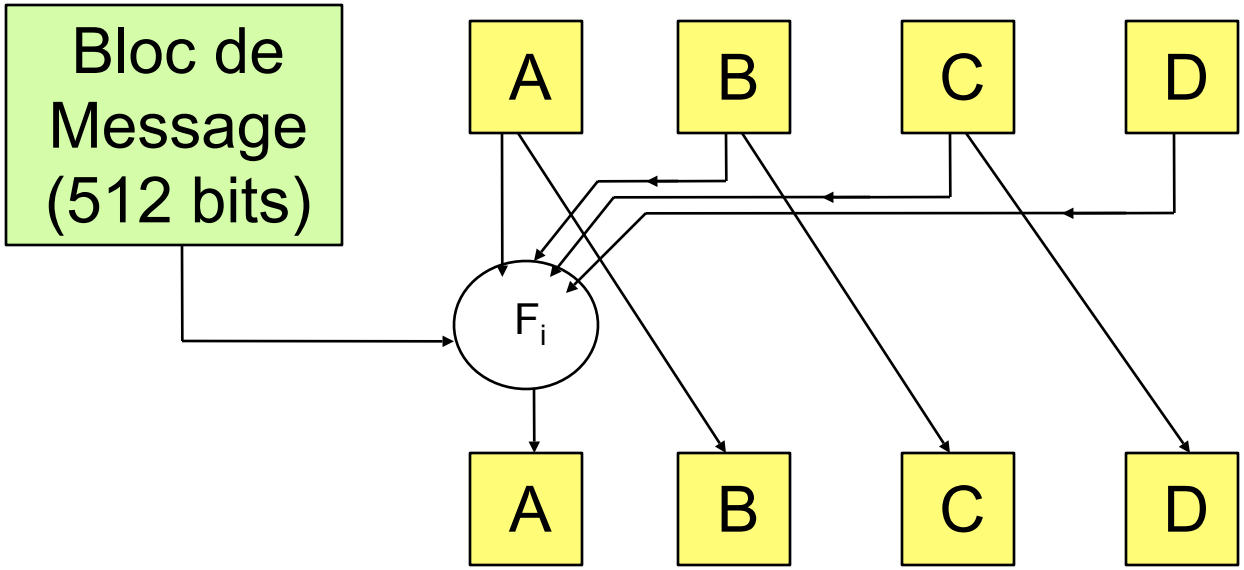
La fonction de compression est constituée de 64 tours :



Tous les objets manipulés font 32 bits

MD5

La fonction de compression est constituée de 64 tours :



Tous les objets manipulés font 32 bits

MD5

- Fonction F_i de la forme

  $F_i\ (A,B,C,D) = A + (D + f_i(A,B,C) + Cst[i] + W[i])<<<s[i]$

- s[i] et Cst[i] sont donnés dans les spécifications

- W[i] est dérivé des mots du bloc de message $(M[i])_{i=0,\ldots,15}$ par simple duplication

Résultats de cryptanalyse

- Den Boer – Bosselaers, 1993
  - Trouvent IV et IV' tel que F(IV, M) = F(IV',M)
  - Pas très utile en pratique
- Dobbertin, 1996
  - Trouve une collision F(IV, M) = F(IV, M') mais IV n'est pas spécifié pour MD5
- Wang et.al., 2004
  - Collision pour MD5 annoncée en août 2004

den Boer-Bosselaers

$A' = D$

$B' = A + (D + f_i(A,B,C) + Cst_i + W_i) <<< s_i$

$C' = B$

$D' = C$

Si $\Delta A = \Delta B = \Delta C = \Delta D = 2^{31}$

alors $\Delta A' = \Delta B' = \Delta C' = \Delta D' = 2^{31}$
avec probabilité 0.5

den Boer-Bosselaers

- Prendre 2 IV ne différent que sur les bit 32
- Les 16 premiers tours sont gratuits
- Les tours 33-64 marchent avec probabilité 1 (utilisation de la fonction XOR)
- Probabilité de succès total 2^{-32}
- Attaque réalisée en pratique

Synthèse

- De nombreuses attaques dédiées contre les fonctions de hachage.
- Les plus populaires :
  - SHA-0 : collision connues
  - MD5 : collision connues
  - MD4 : collisions connues
  - MD2 : collisions, préimages
  - RIPEMD, HAVAL : collisions

En pratique

Fonctions utilisées

- SHA-1
- MD5 (quand la résistance aux collisions n'est pas nécessaire. Ex : certificats, HMAC)
- SHA-256 : nouveaux algorithmes de la famille SHA datant de 2002