

ch 1 : Signature électronique et fonctions de hachage

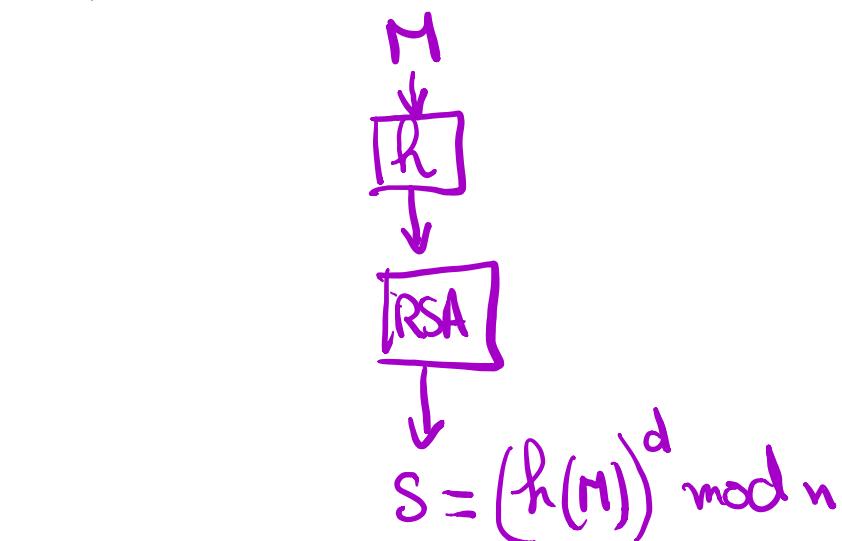
I] Rappels sur RSA

II] Signature RSA et fonctions de hachage

① Problématique de la signature RSA

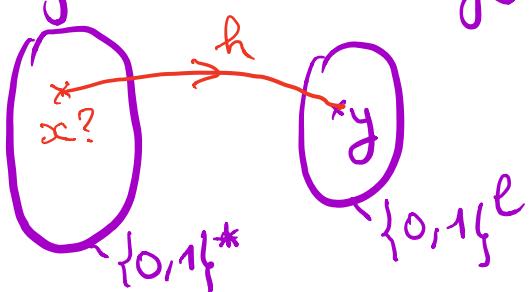
$$\cancel{S = M^d \bmod n}$$

② Paradigme "Hash and sign"



→ Définition d'une fonction de hachage $h: \{0,1\}^* \rightarrow \{0,1\}^l$

prob 1 :



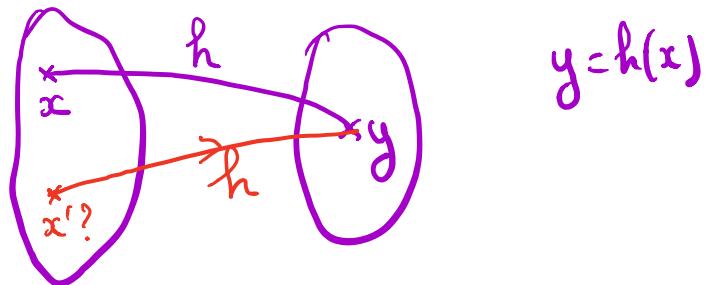
Algorithme pour résoudre le pb 1 : complexité
générique

$$\mathcal{O}(2^l)$$

→ on prend $l \geq 80$

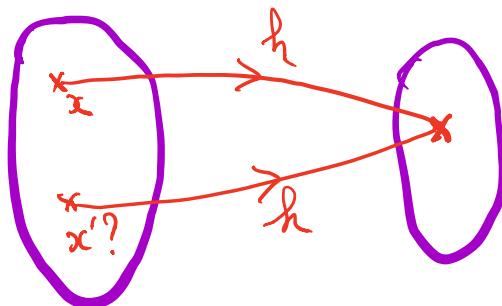
Idem pour le pb 2

prob 2 :



$$y = h(x)$$

prob 3 (h à collisions fortes difficiles)



Algorithme générique :

Générer aléatoirement des valeurs

$$x_1, x_2, x_3, \dots, x_k$$

et calculer leurs images

$$y_1 = h(x_1), y_2 = h(x_2), \dots, y_k = h(x_k)$$

jusqu'à obtenir une égalité

du type : $y_i = y_j \quad (i < j)$

Analyse de cet algorithme : P = probabilité qu'une telle égalité apparaisse

(P dépend de k et de l)

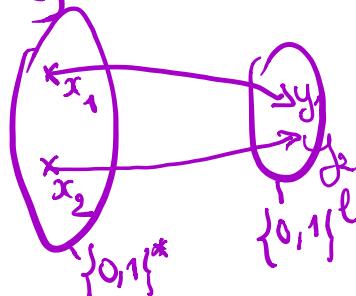
$1 - P$ = probabilité qu'il n'y ait pas de telle égalité

p_1 = probabilité que le 1^e tirage (= génération de x_1) ne provoque pas d'égalité

On a : $p_1 = 1$

p_2 = proba que le 2^e tirage (= génération de x_2) ne provoque pas d'égalité (sachant que le 1^e tirage a déjà été effectué)

$$P_2 = \frac{2^l - 1}{2^l} = 1 - \frac{1}{2^l}$$



p_3 = proba que le 3^e tirage ne provoque pas d'égalité

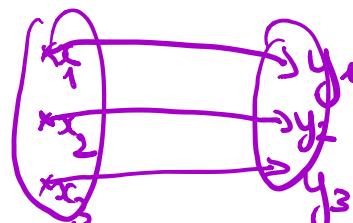
(sachant que les 2 premiers tirages ont déjà été effectués et n'ont pas eux-mêmes provoqué d'égalité)

$$P_3 = \frac{2^l - 2}{2^l} = 1 - \frac{2}{2^l}$$

:

P_k = proba que le k ^e tirage ne provoque pas d'égalité

$$P_k = 1 - \frac{k-1}{2^l}$$



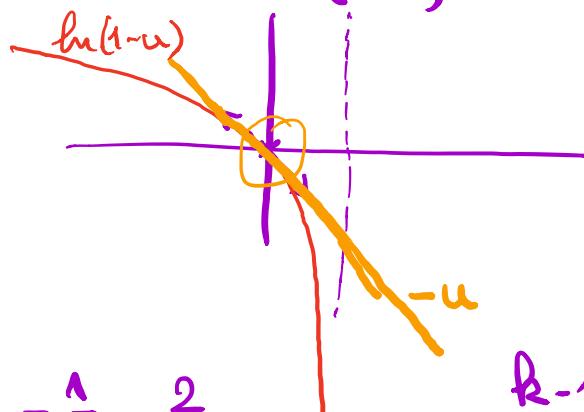
$$\underbrace{1-P}_{\substack{\text{pas de} \\ \text{collision}}} = P_1 \times P_2 \times P_3 \times \cdots \times P_k$$

$$= 1 \times \left(1 - \frac{1}{2^k}\right) \times \left(1 - \frac{2}{2^k}\right) \times \cdots \times \left(1 - \frac{k-1}{2^k}\right)$$

$$\Rightarrow \ln(1-P) = \sum_{u=0}^k \ln\left(1 - \frac{u}{2^k}\right) + \cdots + \ln\left(1 - \frac{k-1}{2^k}\right)$$

pour u petit, $\boxed{\ln(1-u) \approx -u}$

[cf développement limité $\ln(1-u) = -u + o(u)$]



$$\Rightarrow \ln(1-P) \approx -\frac{1}{2^k} - \frac{2}{2^k} - \cdots - \frac{k-1}{2^k}$$

$$\approx -\frac{1}{2^k} \underbrace{[1+2+3+\cdots+(k-1)]}_{= \frac{k(k-1)}{2}}$$

$$\ln(1-P) \approx -\frac{k(k-1)}{2 \times 2^k}$$

" \Rightarrow "

$$1-P \approx e^{-\frac{k(k-1)}{2 \times 2^k}}$$

$$\boxed{P \approx 1 - e^{-\frac{k(k-1)}{2 \times 2^k}}}$$

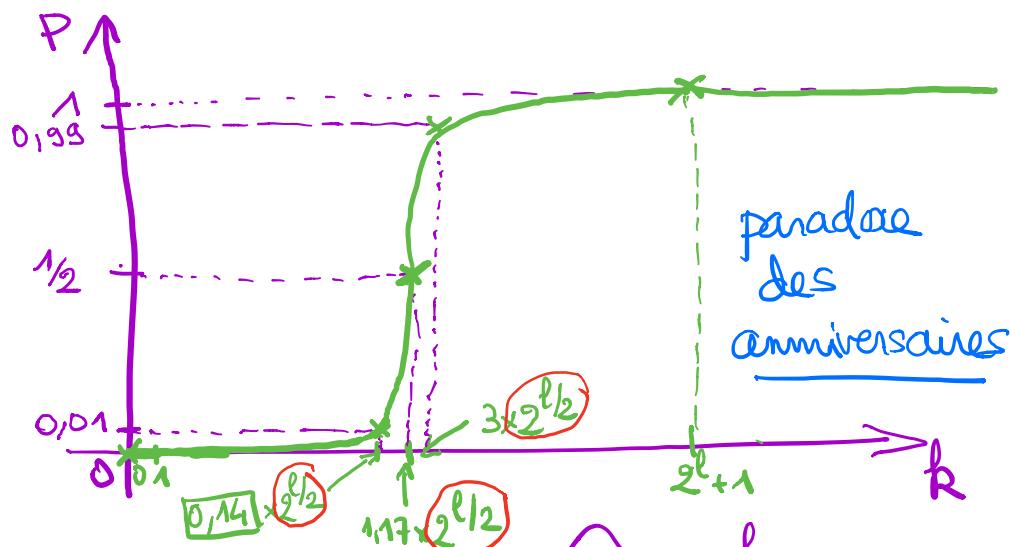
(prob d'obtenir une collision avec k tincages d'atomes)

Remarque: si $k \rightarrow +\infty$, $P \rightarrow 1$
 si $k \rightarrow -\infty$, $P \rightarrow 0$

Inversement $\ln(1-P) \simeq -\frac{k(k-1)}{2 \times 2^k}$

$$\Leftrightarrow \underbrace{\frac{k(k-1)}{\simeq k^2}}_{\simeq k^2} \simeq 2 \times 2^k \times \ln\left(\frac{1}{1-P}\right)$$

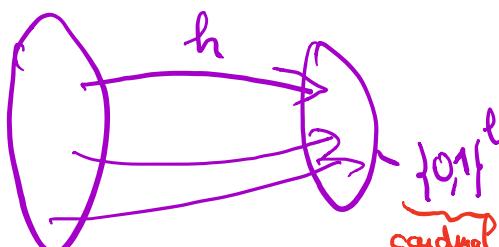
$$\Leftrightarrow \boxed{k \simeq 2^{k/2} \sqrt{2 \ln \frac{1}{1-P}}}$$



$$k=0 \text{ ou } k=1 \Rightarrow P=0$$

$$k > 2^k \Rightarrow P=1$$

$$P = \frac{1}{2} \Leftrightarrow k \simeq 2^{k/2} \sqrt{2 \ln \frac{1}{1-\frac{1}{2}}} \\ = \boxed{2^{k/2} \sqrt{2 \ln 2}} \\ \simeq 1.17$$



$$P = 0,99 \Leftrightarrow k \approx 2^{l/2} \sqrt{2 \ln \frac{1}{1-\frac{99}{100}}} = 2^{l/2} \underbrace{\sqrt{2 \ln 100}}_{\approx 3}$$

$$P = 0,01 \Leftrightarrow k \approx 2^{l/2} \sqrt{2 \ln \frac{1}{1-\frac{1}{100}}} = 2^{l/2} \underbrace{\sqrt{2 \ln \frac{100}{99}}}_{\approx 0,14}$$

Paradoxe des anniversaires

au lieu de considérer une fonction $h : \{0,1\}^k \rightarrow \{0,1\}^l$
 on peut considérer la fonction
 $h : \{\text{personnes}\} \rightarrow \{1, 2, \dots, 365\}$

P = proba de collision pour k personnes

$$\left\{ \begin{array}{l} P \approx 1 - e^{-\frac{k(k-1)}{2 \times 365}} \xrightarrow{\text{cardinal de l'ensemble d'années}} \\ k \approx \sqrt{2 \ln \frac{1}{1-P}} \end{array} \right.$$

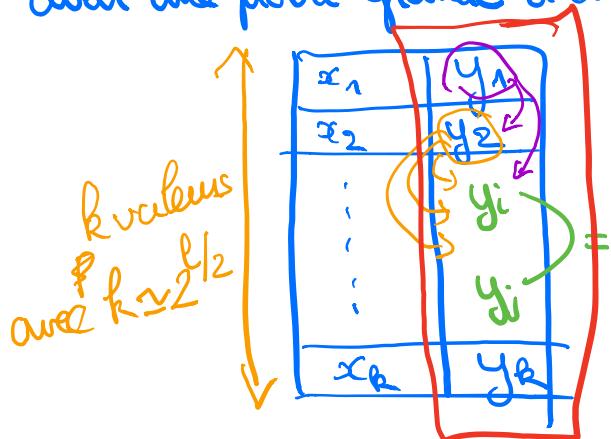
$$P = \frac{1}{2} \Leftrightarrow k \approx 1,17 \times \sqrt{365} \approx 23$$

$$\text{Ex: } k=29 \Rightarrow P \approx 1 - e^{-\frac{29 \times 28}{2 \times 365}} = ? \quad 0,67$$

Algorithme générique pour le pb 3 /: générer aléatoirement

x_1, x_2, \dots, x_R
 jusqu'à calculer $y_i = h(x_i)$
 jusqu'à obtenir une égalité
 du type $y_i = y_j$ ($i < j$)

On a montré qu'il suffit de prendre $k \approx 2^{\ell/2}$ pour avoir une proba grande d'avoir une collision



Méthode naïve
pour trouver
la collision

On parcourt la
liste pour trouver
une valeur égale à y_1

puis:
on parcourt la liste
! y_2

→ complexité : $O(k^2)$

$$(k + (k-1) + (k-2) + \dots = \frac{k(k+1)}{2})$$

$$O(k^2) \approx O(2^\ell)$$

$$(k \approx 2^{\ell/2})$$

Bonne Méthode : on tri l'ensemble des y_i

puis, une fois cet ensemble trié,
la complexité pour trouver une collision
est en $O(k)$

→ complexité : $O(k \ln k)$

(par exemple algorithme "quick sort")

⇒ complexité de l'algo générique pour le ph3

- génération des x_i et des y_i : $O(k)$
- tri des y_i : $\boxed{O(k \ln k)}$
- ~~détection de la collision~~ : $O(k)$

\Rightarrow complexité = $O(k \ln k)$ avec $k \approx 2^{l/2}$

$$= O(2^{l/2} \ln 2^{l/2})$$

$$= \boxed{O(l 2^{l/2})} \text{ car } \ln 2^{l/2} = \frac{l}{2} \ln 2$$

Rémarque : on peut un peu améliorer

$$\rightarrow \boxed{O(2^{l/2})}$$

Consequence : pour avoir une (bonne) fonction de hachage
on choisit l tel que $2^{\frac{l}{2}} \geq 2^{80}$
c'est à dire $\boxed{l \geq 160}$

La taille de sortie d'une fonction de hachage
doit être d'au moins 160 bits
(pour une sécurité de 2^{80})

[ou 256 bits pour une sécurité de 2^{128}]

③ Construction de Merkle-Damgård

(analogie au schéma de Feistel pour le
chiffrement par blocs, et aux modes

d'opération)

Idée: On suppose qu'on a une fonction de compression

$f : \{0,1\}^m \rightarrow \{0,1\}^l$

m fixé
l fixé
avec $m \geq l+2$

qui sont à collisions
fortes difficiles

Question: Comment, à partir de f , construire
une "vraie" fonction de hachage

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

Construction: $x \in \{0,1\}^*$

$$\begin{aligned} x &= x_1 \| x_2 \| x_3 \| \dots \| x_k \\ &= \left(\underbrace{\dots}_{m-l-1 \text{ bits}} \right) = \left(\underbrace{\dots}_{m-l-1 \text{ bits}} \right) \\ y &= y_1 \| y_2 \| y_3 \| \dots \| y_k \| \underbrace{y_{k+1}}_{(m-l-1) \text{ bits}} \\ &= x_k \| 0..0 \\ &\quad \xrightarrow{(m-l-1) \text{ bits}} \text{écriture binaire de } d \\ &\quad \xrightarrow{d \text{ bits "0"}} \end{aligned}$$

on veut "compléter" le dernier bloc

puis on applique une transformation à y

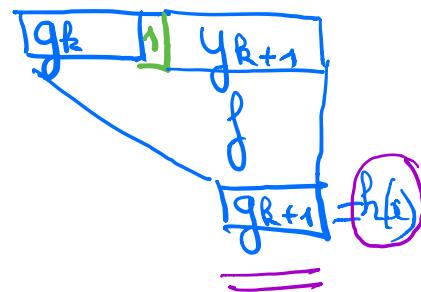
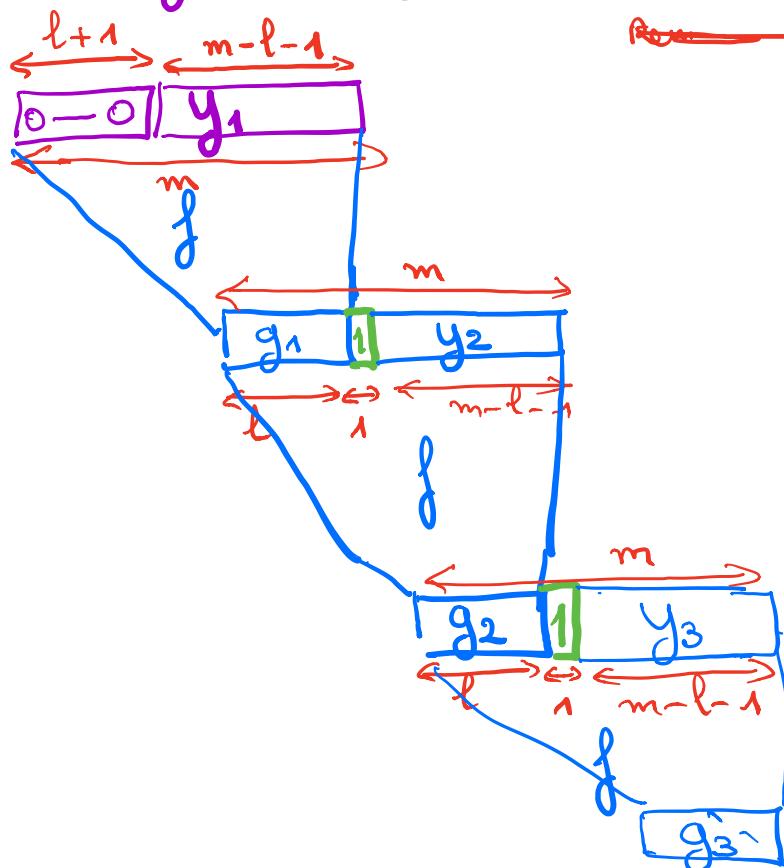
ex: $y_k = x_k \| \overbrace{00000}^{d=5}$

$$y_{k+1} = \underbrace{0\dots0101}_{(m-l-1) \text{ bits}}$$

$$y = y_1 || y_2 || \dots || y_{k+1}$$

Remarque
 $m \geq k+2$
 $\Rightarrow m-k-1 \geq 1$

$$f: \{0,1\}^m \rightarrow \{0,1\}^k$$



On a aussi défini une fonction

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

(à partir de la fonction de compression
 $f: \{0,1\}^m \rightarrow \{0,1\}^k$)

Théorème [Merkle, Damgård, 1989]

Si f est à collisions fâches difficiles
alors h aussi

Démonstration: Supposons que h n'est pas
à collisions fâches difficiles :

on peut trouver x et x' tels que $h(x) = h(x')$
 $\boxed{x \neq x'}$

$$x = x_1 \| \dots \| x_k \rightarrow y = y_1 \| y_2 \| \dots \| y_{k+1}$$

pas forcément
complet

$$x' = x_1 \| \dots \| \underset{\text{pas forcément
complet}}{x_t} \| \dots \| x_{k+1} \rightarrow y' = y'_1 \| y'_2 \| \dots \| y'_{t+1}$$

$$\begin{aligned} h(x) = h(x') &\Leftrightarrow g_{k+1} = g'_{t+1} \\ &\Leftrightarrow f(g_k \| 1 \| y_{k+1}) = f(g'_t \| 1 \| y'_{t+1}) \end{aligned}$$

Si $y_{k+1} \neq y'_{t+1}$, alors on a trouvé une
collision sur f \square

Si $g_k \neq g'_t$, —————— sur f \square

$$\text{Si } y_{k+1} = y'_{t+1} \text{ et } \boxed{g_k = g'_t}$$

$$\boxed{f(g_{k-1} \| 1 \| y_k) = f(g'_{t-1} \| 1 \| y'_t)}$$

Si $g_{k-1} \neq g'_{t-1}$: on a trouvé une collision
sur f \square

Sinon : $g_{k-1} = g'_{t-1}$

$$\underbrace{f(g_{k-2} \parallel 1 \parallel y_{k-1})}_{\neq ?} = f(g'_{t-2} \parallel 1 \parallel y'_{t-1})$$

A quel moment ça s'arrête

1^e cas : on obtient ($t > k$)

$$f(0 \xrightarrow{k+1} \bullet \parallel y_1) = f(g_{t-k} \xrightarrow{k} \bullet \parallel y'_{t-k+1})$$

\neq

→ collision \square

2^e cas : on obtient ($k > t$)

$$f(g_{k-t} \parallel 1 \parallel y_{k-t+1}) = f(0 \xrightarrow{t} \bullet \parallel y'_1)$$

\neq

→ collision \square

3^e cas : on obtient ($k=t$)

$$f(0 \xrightarrow{t} \bullet \parallel y_1) = f(0 \xrightarrow{t} \bullet \parallel y'_1)$$

- si $y_1 \neq y'_1$ → collision \square

- si $y_1 = y'_1$ → on aurait alors
 $y = y'$ et donc $x = z'$
impossible

④ Attaque de Bleichenbacher

Question: Supposons qu'on ait une fonction de hachage h obtenue grâce à la construction de Merkle-Damgård
(ex: MD5, SHA-1)

et qu'on trouve une collision: M et M'
tels que $M \neq M'$ et $h(M) = h(M')$

Soit T et T' deux textes (choisis par l'attaquant)
Est-il possible de fabriquer deux fichiers

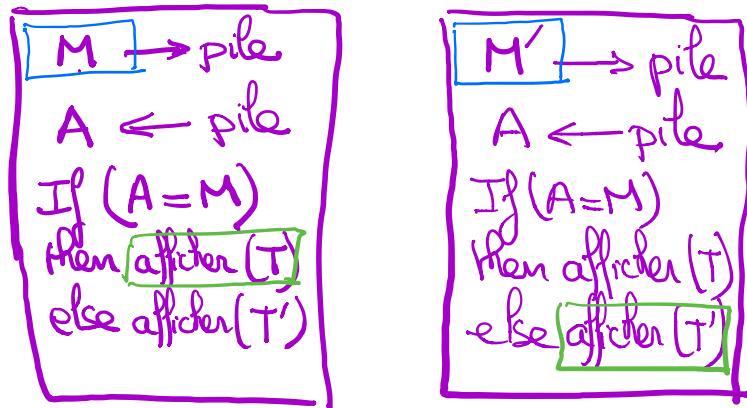
F et F' tels que

- F affiche le texte T
- F' affiche le texte T'
- $h(F) = h(F')$ ($F \neq F'$)

Idée: Bleichenbacher a montré que c'était possible avec des fichiers postscript
($F = \text{nom.ps}$)

Remarque: cela a ensuite été étendu à des fichiers .pdf, .doc, ...

Le langage postscript contient la notion de pile



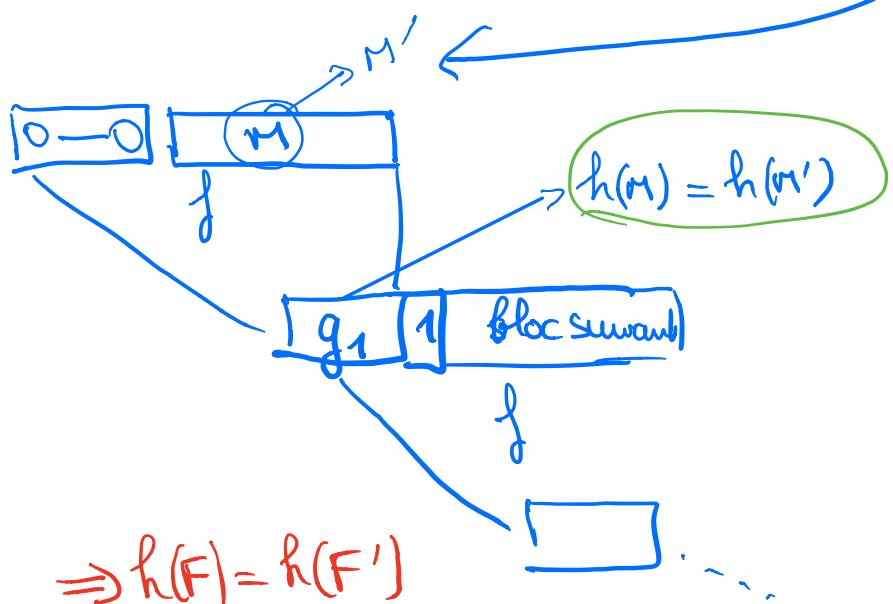
F

F'

- F affiche T
- F' affiche T'

$$h(F) = h(M \parallel \text{suite})$$

$$h(F') = h(M' \parallel \text{suite})$$



$$g_1 = h(m)$$

$$g'_1 = h(m')$$

$$g_1 = g'_1$$

(car M et M'
collisent)

⑤ Fonctions de hachage utilisées en pratique

Nom	Inventeur	Année	l	Attaque, anniversaire	Méilleure attaque connue
MD4 <i>(Message Digest Méthode Damaged)</i>	Rivest	1990	128	$\approx 2^{64}$	1995 : Doffertin → collision
MD5 <i>(Message Digest Méthode Damaged)</i>	Rivest	1991	128	$\approx 2^{64}$	2005 : Wang → collisions (en 2^{24})
SHA-0 <i>(Secure Hash algorithm)</i>	? (NSA)	1993	160	$\approx 2^{80}$	Attaque ?
SHA-1	? (NSA)	1994	160	$\approx 2^{80}$	2004 : Antoine Joux → collision (en 2^{51})
SHA-256 SHA-384 SHA-512 <i>(= Famille SHA-2)</i>	? (NSA)	2002	256 384 512	$\approx 2^{128}$ $\approx 2^{192}$ $\approx 2^{256}$	2006 : Wang → attaque en 2^{69} 2016 : collisions (Stevens,...)
SHA-3 <i>(Keccak)</i> <i>(≠ Hérité Damaged)</i>	Bertoni Daemen Peeters van Assche	2011	224 256 384 512	$\approx 2^{112}$ $\approx 2^{128}$ $\approx 2^{192}$ $\approx 2^{256}$	
<i>Construction épingle</i>					

⑥ Signature RSA en pratique

Méthodes données par des standards

PKCS #1 (publié par RSA Data Security)
 Public Key Cryptographic Standard fondée par les inventeurs de RSA

v1.5 $s = (\mu(M))^d \text{ mod } n$

avec $\mu(M) = \underbrace{\text{padding}}_{\substack{1 \text{ octet} \\ \text{ex: 20 octets}}} \parallel \text{h}(M)$

$\text{padding} \parallel \text{h}(M)$

1024 bits
(= 128 octets)

Vérification: Alice $\xrightarrow{M, s}$ Bob

Bob calcule $s^e \text{ mod } n = \underbrace{\text{padding} \parallel \text{h}(M)}_{1024 \text{ bits}} \parallel \text{BC}$

$\text{h}(M) = ?$

ISO 9796-2

$s = (\mu(M))^d \text{ mod } n$

avec $\mu(M) = \text{GA} \parallel \underbrace{M_1}_{\substack{1 \text{ octet} \\ \text{au plus} \\ 106 \text{ octets}}} \parallel \text{h}(M) \parallel \text{BC}$

où $M = M_1 \parallel M_2$

Alice $\xrightarrow{M_2, s}$ Bob

= Signature avec message recovery $S^e = \text{GA} \parallel \text{M}_1 \parallel \text{h}(M) \parallel \text{BC}$
 (recouvrement de message)

