Introduction à la cryptographie

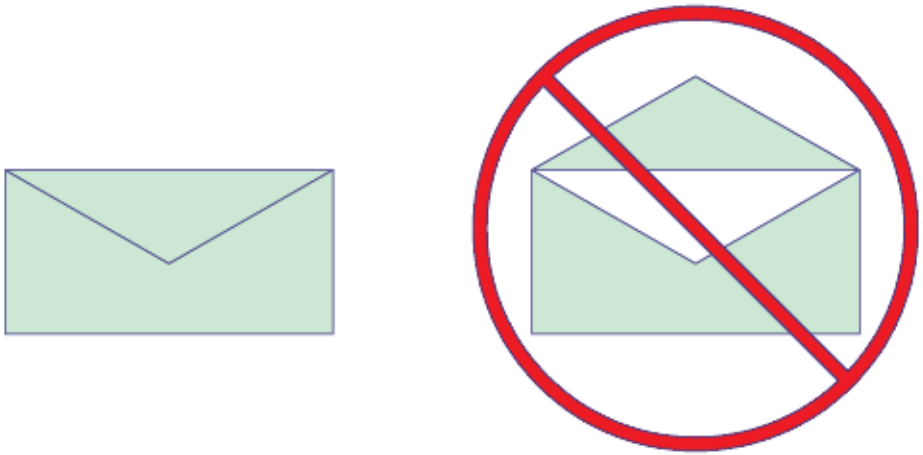
Master Informatique — Semestre 2 — UE optionnelle de 3 crédits

Année 2017-2018

Version du 13 septembre 2017

Les trois concepts de base : intégrité, confidentialité et authentification

Intégrité : Garantir qu'un message (un document, ou encore un fichier) n'a pas subi de modification (aussi bien accidentelle qu'intentionnelle)

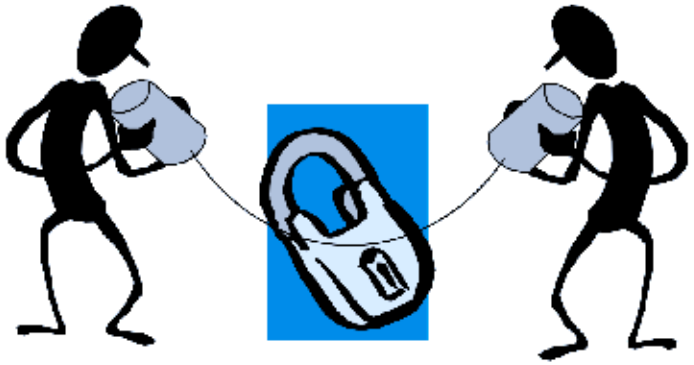


C'est utile notamment pour les téléchargements...

Les trois concepts de base : intégrité, confidentialité et authentification

Deux types de confidentialité :

Archiver des données sur un **support**



Communiquer des données sur un **canal**

de façon qu'un tiers ne puisse en prendre connaissance

Méthode du chiffrement par décalage

Chaque lettre (codée par un octet) subit un décalage constant K

Opération de chiffrement : $C_K(x) = x + K \mod 256$

Opération de déchiffrement : $D_K(y) = y - K \mod 256$

Pour chaque lettre x : $D_K \circ C_K(x) = x$

Cryptanalyse par recherche exhaustive : il suffit d'essayer les 255 clés K possibles.

En pratique, il doit être impossible de fabriquer toutes les clefs possibles puis de les essayer une à une afin de retrouver le texte clair, car on sait détecter de manière automatique qu'un texte a du sens (dans la langue utilisée).

Le nombre de clefs doit donc être redhibitoire.

L'ordinateur le plus puissant au monde (Juin 2016)

Le supercalculateur chinois Sunway TaihuLight affiche une puissance de calcul record de 93 pétaFLOPS, ce qui représente 93×10^{15} (93 millions de milliards) opérations par seconde.

Il rassemble 10,65 millions de cœurs fonctionnant en parallèle...

Parcours de toutes les clefs de n bits

Considérons tout d'abord des clefs de longueur $n = 48$ bits, c'est-à-dire 6 octets. Il y a $2^n = 2^{48} = 256 \times (2^{10})^4 = 256$ milliers de milliards de clefs.

À une fréquence de 1Ghz, il faut $(256 \times 2^{10})^4/1000^3 \simeq 256\,000$ secondes, c'est-à-dire *3 jours*, pour parcourir toutes ces clefs.

Considérons à présent des clefs de longueur $n = 96$ bits, c'est-à-dire 12 octets. Il y a alors $2^n = 2^{96} = 2^6 \times (2^{10})^9 = 64$ milliards de milliards de millards de clefs.
À une fréquence de 10^{17} Hz, il faut $64 \times 10^{27}/10^{17} \simeq 640 \times 10^9$ secondes, c'est-à-dire environ *vingt millénaires*...

2^{96} est bien un chiffre redhibitoire !

Tout ceci est **approximatif** : on néglige ici notamment le temps nécessaire pour déchiffrer le message avec chaque clef et pour tester si le texte obtenu a du sens.

Pourtant, 96 bits, c'est insuffisant !

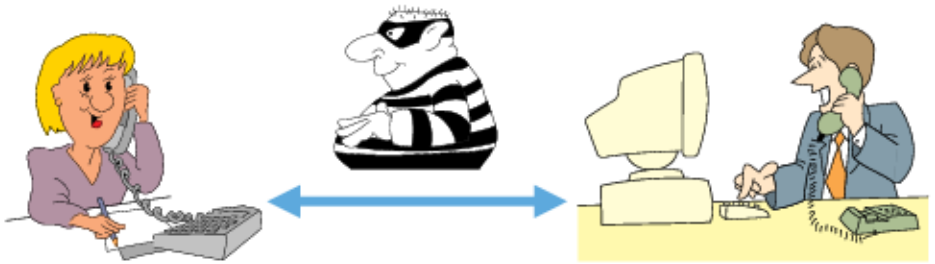
- Il faudrait évidemment prendre en compte la **loi de Moore**.
- Il faudrait aussi envisager la possibilité de calculs **largement distribués sur Internet**.
- La recherche exhaustive mènera au déchiffrement **à coup sûr** : si on essaie seulement 1% des clefs, la probabilité de décrypter est supérieure à 1%, mais ça va 100 fois plus vite !
  - ⇝ Quelle probabilité de succès de l'attaquant peut-on accepter ?
  - ⇝ Quelle est la **durée de confidentialité** requise ?

ANSSI recommande des clefs de 128 bits, au minimum.

Les trois concepts de base : intégrité, confidentialité et authentification

Deux types d'authentification :

① Prouver de façon **interactive** son identité à un interlocuteur



② Attacher, à un message, une preuve **non-interactive** de son origine.

On parle alors de signature électronique

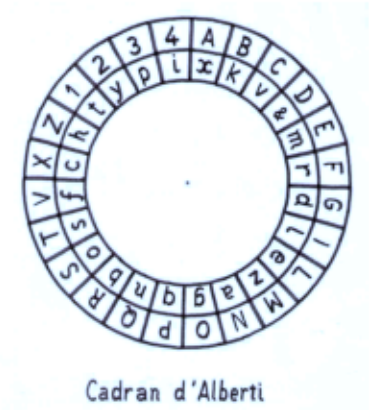
Historique : les 3 âges de la cryptologie selon J. STERN

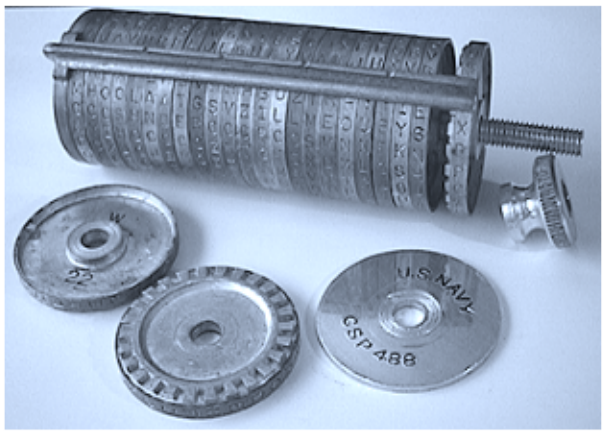
L'âge artisanal : jusqu'en 1918

Mécanismes manuels et secrets fondés sur des **substitutions** et des **permutations**.

Exemples :
- Scytales : **permutation** des lettres du texte
- Chiffrement de César : **substitution** par décalage

MESSAGE $\rightsquigarrow$ PHVVDJH


Cadran d'Alberti



Historique : les 3 âges de la cryptologie selon J. STERN

L'âge technique : de 1919 à 1975

Machines chiffrantes (Enigma)

L'âge paradoxal : de 1976 à nos jours

Invention de la *cryptographie asymétrique*.

Principes de Kerckhoff

— *Le système doit être matériellement, sinon mathématiquement, **indéchiffrable**.*

— *Il faut qu'il **n'exige pas le secret**, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi.*

— *La clef doit pouvoir être **communiquée** et **retenue** sans le recours de notes écrites, et être changée ou modifiée au gré des correspondants.*

— *etc.*

≪ La cryptographie militaire ≫ (1883)

- ✔ *Concepts de base et historique*
- ✔ *Qu'est-ce qu'un chiffre redhibitoire ?*
- ✔ *Des systèmes plus évolués*
- ☛ *Différents types de chiffrement*

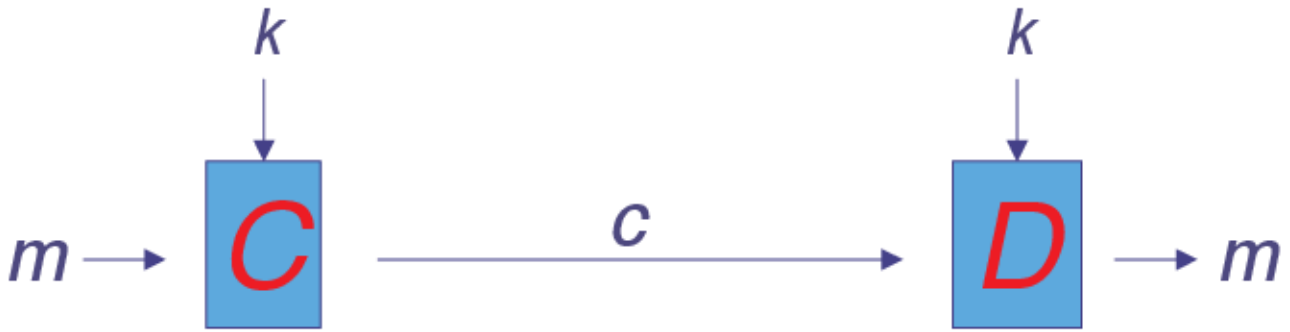
Exemple générique

Chiffrement symétrique (à clef secrète)

— Algorithme de chiffrement : C

— Algorithme de déchiffrement : D

sont **publics**.

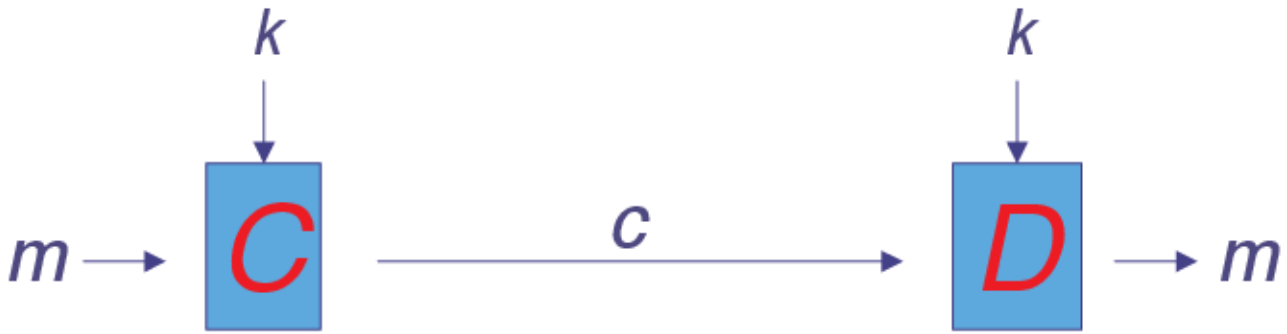


Comment définir la sécurité ? Ce système est sûr s'il est impossible « pratiquement » de retrouver le **message** *m* à partir du **chiffré** *c* sans connaître la clef *k*.

Selon l'ANSSI, la taille de la clef secrète doit être supérieure à 128 bits.

La cryptographie symétrique

On dit qu'un système de chiffrement est symétrique lorsque la même clef *k* est utilisée pour le chiffrement et le déchiffrement.



Ces procédés sont souvent des *heuristiques* à base de **permutations** (diffusion) et **substitutions** (confusion).

La cryptographie symétrique peut être facilement implémentée au niveau matériel afin d'obtenir un **débit très rapide**.

Deux types de chiffrements symétriques

① **Par blocs (≪ block cipher ≫)** : Décomposition du message en blocs

— DES (IBM-NBS)

- normal : clé de 56 bits, blocs de 64 bits
- renforcé (triple-DES) : clé de 112/168 bits

— IDEA : clé de 128 bits, blocs de 64 bits

— **AES (RIJNDAEL : Daemen-Rijmen)**

clefs de 128, 196 ou 256 bits, blocs de 128 bits.

② **Par flot (≪ stream cipher ≫)** : chiffre un flux de données à l'aide d'une clef aussi longue que le texte, appelée parfois *générateur pseudo-aléatoire*

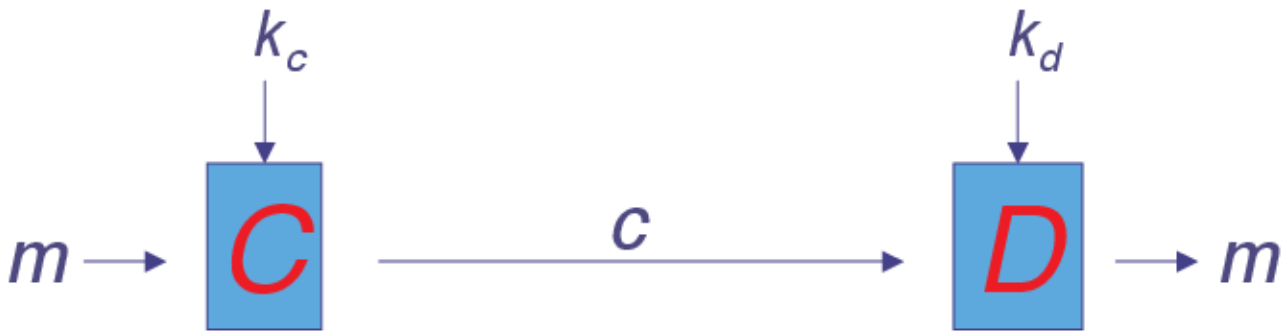
— Le masque jetable, inventé par Gilbert Vernam en 1917.

— RC4 : chiffre octet par octet (inventé par Rivest)

Autre type de chiffrement : le chiffrement asymétrique

Deuxième mode de chiffrement : le chiffrement asymétrique

- Algorithme de chiffrement : C $\rightsquigarrow$ public
- Algorithme de déchiffrement : D $\rightsquigarrow$ public
- **Deux clefs** :
  - chiffrement : k_c $\rightsquigarrow$ **publique** !
  - déchiffrement : k_d $\rightsquigarrow$ privée



Les lois de Kerckhoff sont ainsi poussées à bout !

Sécurité ? Il doit être « pratiquement » impossible de retrouver m à partir de c sans connaître la clef de déchiffrement k_d.

Chiffrement asymétrique : principaux exemples

— **RSA** (Rivest-Shamir-Adleman, 1978)

lié au problème de la *factorisation des nombres*

— **Merkle-Hellman** (1978)

lié au problème du « *sac à dos* » (tous cassés...)

— **Mc Eliece** (1978)

est une application de certains *codes correcteurs d'erreurs*

— **El Gamal** (1985)

lié au problème du *logarithme discret*

- ✔ *Concepts de base et historique*
- ✔ *Qu'est-ce qu'un chiffre redhibitoire ?*
- ✔ *Des systèmes plus évolués*
- ✔ *Différents types de chiffrement*
- ☛ *Symétrique vs asymétrique*

Cryptographie asymétrique : le nombre et l'échange de clefs

En **cryptographie asymétrique**, si n personnes veulent communiquer secrètement 2 à 2, il leur faut en tout $2 \times n$ clés : chacun fabrique une paire de clefs et diffuse sa clef publique.

Si ces personnes utilisent en revanche un chiffrement symétrique, il leur faut une clé secrète pour chaque couple de correspondants, c'est-à-dire en tout $n \times (n-1)/2$ clés.

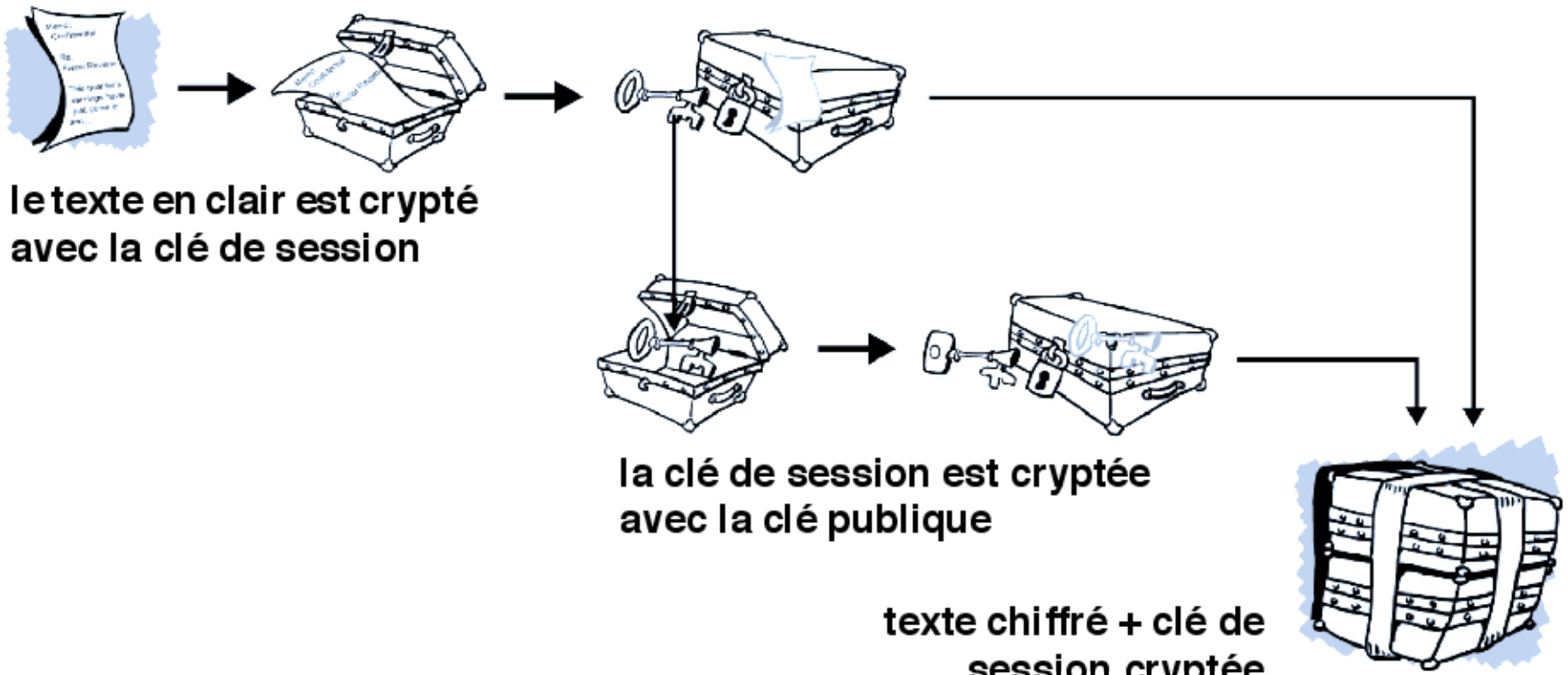
Elles doivent en outre à résoudre le problème de **l'échange des clefs**.

Cependant **la cryptographie symétrique est beaucoup plus rapide** : entre 100 et 1000 fois plus rapide selon les cryptosystèmes, et ses clés sont plus courtes : autour d'une centaine de bits au lieu d'au moins un millier.

Cryptographie symétrique et asymétrique

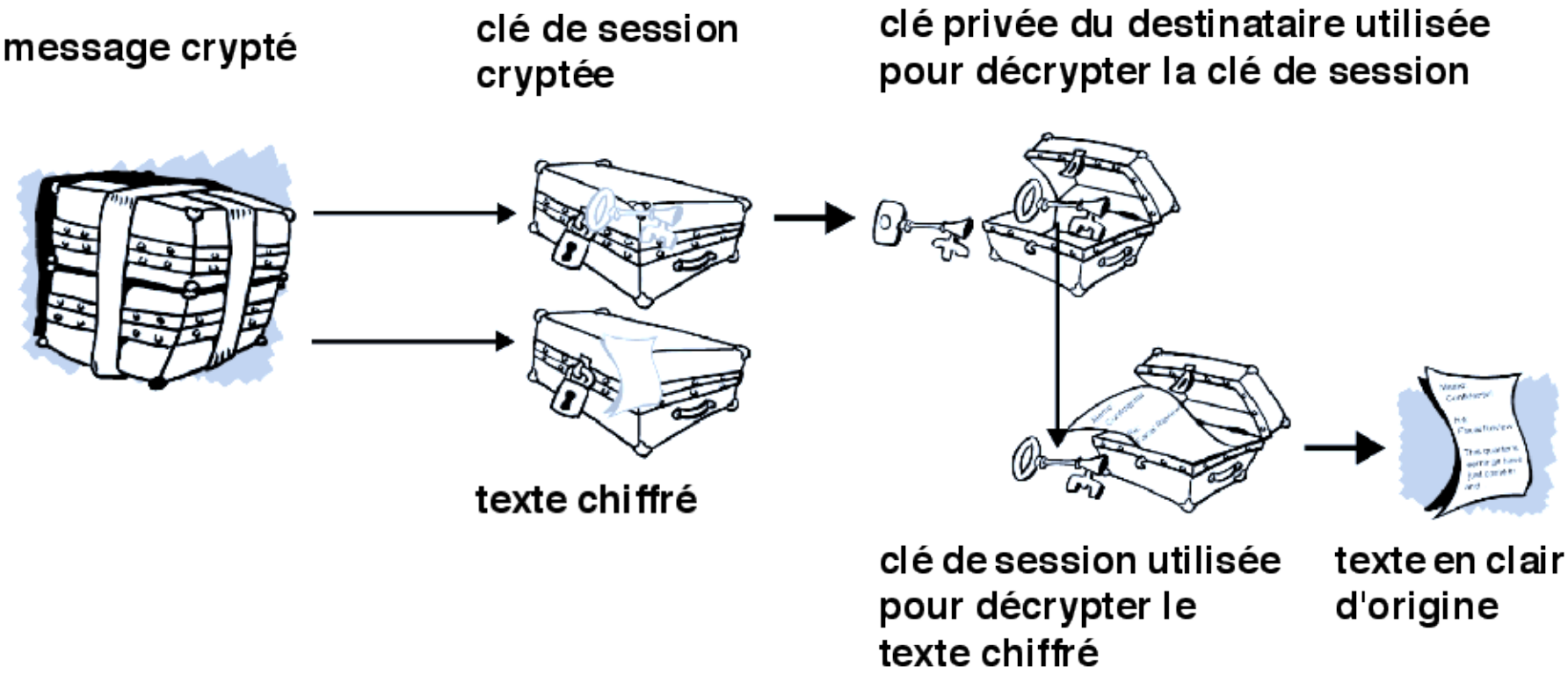
Le principe de l'envoi en PGP (Pretty Good Privacy)

Une **clef de session** nouvelle est fabriquée à chaque envoi.



Cryptographie symétrique et asymétrique

Le principe de la réception en PGP



Aux dernières nouvelles, PGP résiste encore à la NSA.

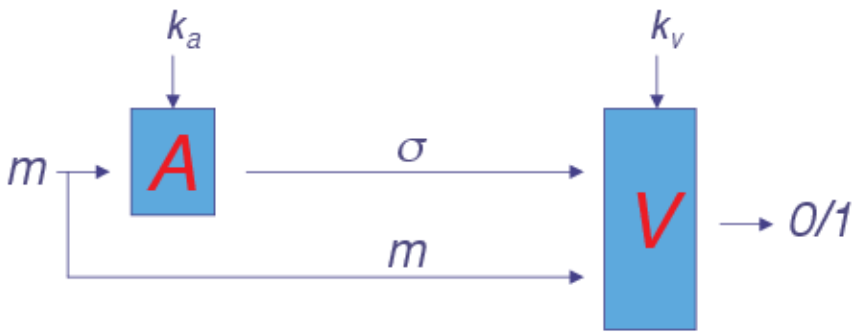
Merci pour l'info, M. Snowden !

- ✔ *Concepts de base et historique*
- ✔ *Qu'est-ce qu'un chiffre redhibitoire ?*
- ✔ *Des systèmes plus évolués*
- ✔ *Différents types de chiffrement*
- ✔ *Symétrique vs asymétrique*
- ☛ *Qu'est-ce qu'une signature électronique ?*

Authentification non-interactive et non-artisanale d'un message

Il s'agit d'attacher à un message une preuve non-interactive de son origine.

- Algorithme d'authentification : A
- Algorithme de vérification : V
- Deux clefs : signature privée : k_a et vérification publique : k_v



Sécurité ? Impossible de produire un (σ, m) valide sans connaître k_a

Exemples

- RSA (Rivest-Shamir-Adleman, 1978)
- El Gamal (1985) et variantes : Schnorr (1989), **DSA** (1994)

Non-répudiation et signature

La **non-répudiation** est un mécanisme qui empêche de nier un contrat.

Elle consiste par exemple à prouver qu'un message a bien été émis par son expéditeur (ou a bien été reçu par son destinataire). L'auteur d'un message ne peut alors nier l'avoir écrit (ou transmis).

Cette fonctionnalité doit donc permettre à un *tiers* de juger un conflit éventuel entre l'expéditeur et le destinataire.

La **signature électronique** est un mécanisme qui garantit l'*authentification* de l'expéditeur, l'*intégrité* des données et la *non-répudiation*.

Références bibliographiques

(1) Jacques Stern, **La science du secret**, Odile Jacob, coll. ≪ Sciences ≫, 5 janvier 1998, 203 pages.

(2) Jean-Guillaume Dumas, Jean-Louis Roch, Éric Tannier et Sébastien Varrette, **Théorie des codes : Compression, cryptage, correction** Collection ≪ Sciences Sup ≫, Dunod, 2007, 352 pages.

(3) A.J. Menezes, P.C. van Oorschot et S.A. Vanstone, **Handbook of Applied Cryptography**, CRC Press, 1996, 816 pages. Disponible en ligne.

(4) William Stallings, **Cryptography and Network Security**. Prentice Hall, 2003, 681 pages.

Une question de chance

Master Informatique — Semestre 2 — UE optionnelle de 3 crédits

Année 2017-2018

Version du 13 septembre 2017

Rappel (bon à savoir)

Combien de fois en moyenne faut-il lancer un dé à 6 faces avant d'obtenir un 3 ?

6 fois, en moyenne !

Car la probabilité de gagner *en un coup* est $\frac{1}{6}$.

Cette propriété permet d'établir le nombre moyen de tentatives nécessaires en moyenne avant de gagner au loto, ou d'établir une connexion TCP/IP sur un réseau perturbé.
Elle sera utilisée à plusieurs reprises dans la suite.

Justification (1/2)

Soit q la probabilité de gagner en un coup.

La probabilité de devoir jouer j fois avant de gagner est $(1-q)^{j-1} \times q$: en effet il faut perdre $j-1$ fois puis gagner.

Le nombre moyen de jets avant de gagner est donc

$$m = q \times \sum_{j=1}^{\infty} j \times (1-q)^{j-1}$$

On peut montrer que $m = 1/q$ de plusieurs manières.

Justification (2/2)

On rappelle d'abord le résultat suivant : pour $-1 < x < 1$,

$$f(x) = \sum_{j=0}^{\infty} x^j = \frac{1}{1-x}$$

C'est d'une certaine manière une extrapolation de la formule bien connue au lycée de la somme des termes d'une suite géométrique.

Par conséquent

$$f'(x) = \sum_{j=1}^{\infty} j \times x^{j-1} = \frac{1}{(1-x)^2}$$

si on ne se pose pas trop de questions, ou si l'on est savant. On en déduit $m = q/(1-(1-q))^2 = q/q^2 = 1/q$.

Justification alternative (si la première ne suffit pas...)

On pose à nouveau $x = 1 - q$. On a $m = q \times r$ où

$$r = \sum_{j=1}^{\infty} j \times x^{j-1} = \sum_{k=0}^{\infty} (k+1) \times x^k$$

en posant $k = j - 1$. On obtient alors

$$r = 1 + x \times \sum_{k=1}^{\infty} k \times x^{k-1} + \sum_{k=1}^{\infty} x^k = x \times r + 1 + \sum_{k=1}^{\infty} x^k = x \times r + \frac{1}{1-x}$$

D'où

$$r \times (1 - x) = \frac{1}{1-x}$$

Par conséquent, $m = q/q^2 = 1/q$.

Fonctions de hachages cryptographiques

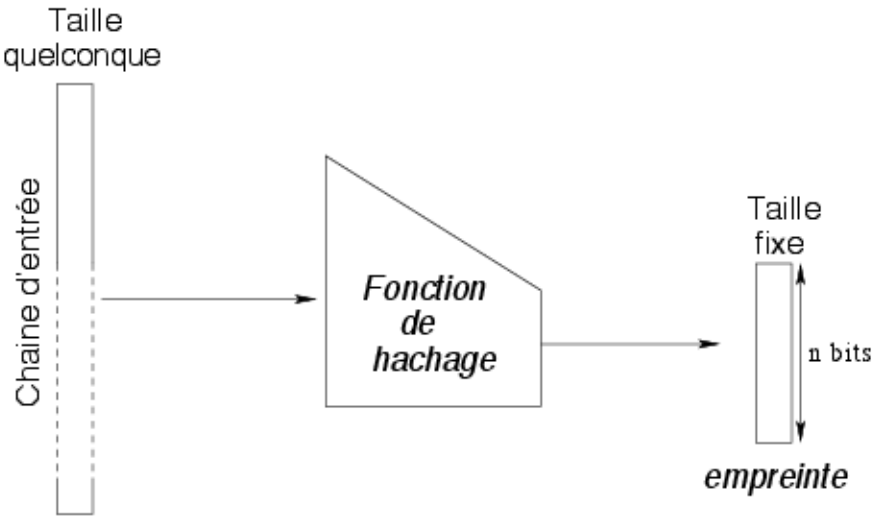
Master Informatique — Semestre 2 — UE optionnelle de 3 crédits

Année 2017-2018

Version du 13 septembre 2017

Rôle d'une fonction de hachage

Le rôle d'une *fonction de hachage* est de transformer un message x de taille quelconque en un résumé court $h(x)$ de taille fixe.



L'image d'un message par une fonction de hachage s'appelle le *résumé*, le *condensé*, l'*empreinte*, ou encore le message *haché*.

Magie des fonctions de hâchage cryptographiques (1/2)

En pratique, une fonction de hâchage *cryptographique* doit garantir qu'une altération (accidentelle ou intentionnelle) du document x modifiera la valeur du résumé $h(x)$ qui en résulte.

Surprenant, non ?

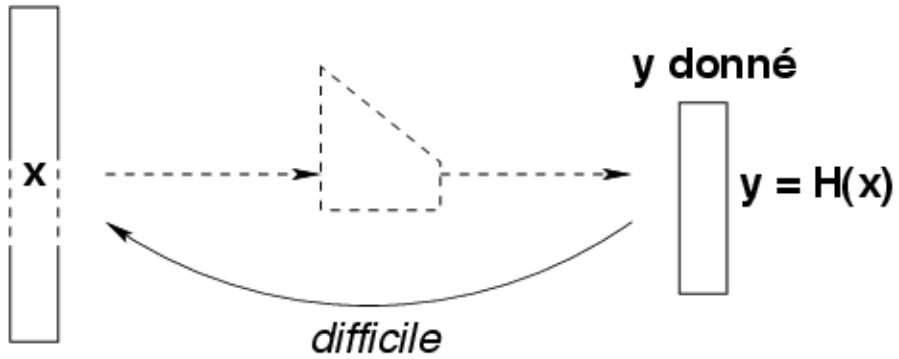
Exemples de résumés générés par la commande UNIX `md5sum`

cette phrase se termine par a	`30aa5e7e92a8c71cc10fe9eeebe01a3b`
cette phrase se termine par e	`d7e1076cbe1d660e14f384683d82cc06`
cette phrase se termine par 3	`e93dde961b27878ae858c8e2f696aa7f`
cette phrase se termine par *	`828b883f6a0ec783181a6475c0aeefd2`

Le résumé comporte 32 caractères hexadécimaux, soit 128 bits.

Magie des fonctions de hâchage cryptographiques (2/2)

Plus formellement, on exige d'une fonction de hâchage *cryptographique* qu'elle soit *résistante à la détermination d'une préimage*.



Étant donné un résumé y, il est **difficile** de calculer un x qui lui correspond.

Il faut donc que le résumé soit assez long !

Explication (un peu grossière)

Considérons un résumé y de longueur l bits et essayons de produire un texte x tel que $y = H(x)$, c'est-à-dire de calculer une *pré-image* de y.

- On choisit un texte x au hasard ;
- On calcule son résumé $H(x)$;
- Si $y = H(x)$ on s'arrête, sinon on recommence avec un autre x.

La probabilité de réussite *en un coup* vaut $1/2^l$ (car il y a 2^l résumés différents).

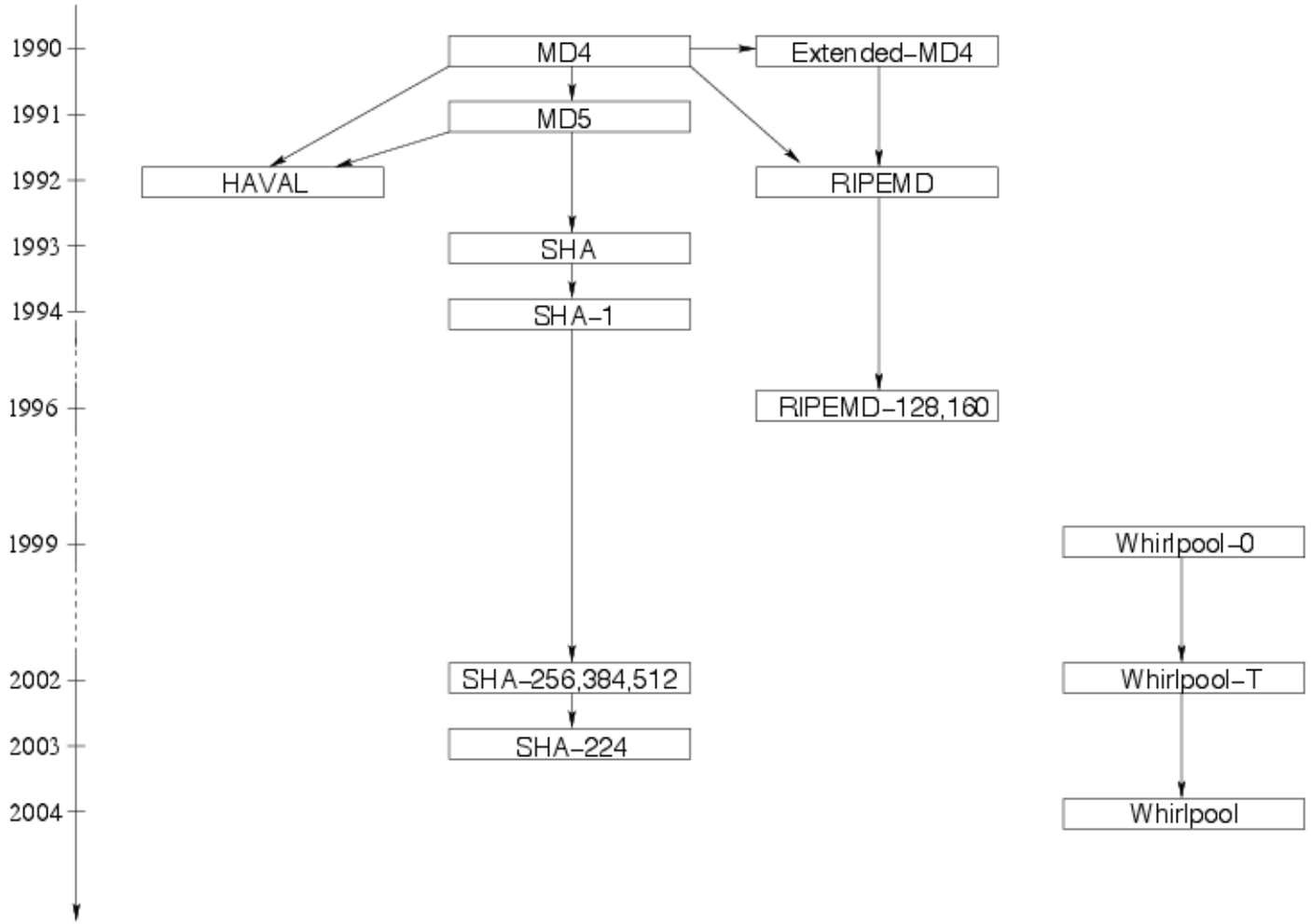
Il suffit donc *en moyenne* de 2^l tentatives pour trouver un x qui convient...

2^l doit être rédhibitoire !

En reprenant la discussion sur les longueurs de clefs, il faut a priori que la longueur l du résumé soit supérieure à 128 bits.

Cependant, l'attaque des anniversaires conduit l'ANSSI à recommander d'utiliser des résumés dont la longueur *deux fois supérieure*, c'est-à-dire **256 bits au minimum**.

Historique des principales fonctions de hachages



La fonction de hachage SHA-3 (aussi appelée « Keccak ») est issue de la « NIST hash function competition » qui s'est conclue en octobre 2012.

Exemples de fonctions de hachage

Exemples de fonctions de hachage fréquemment utilisées

- MD-2, MD-4 (Rivest, 1990, 128 bits)
- MD-5 (Rivest, 1991, 128 bits) très utilisée dans le monde UNIX
- SHA (NIST, 1992, 160 bits)

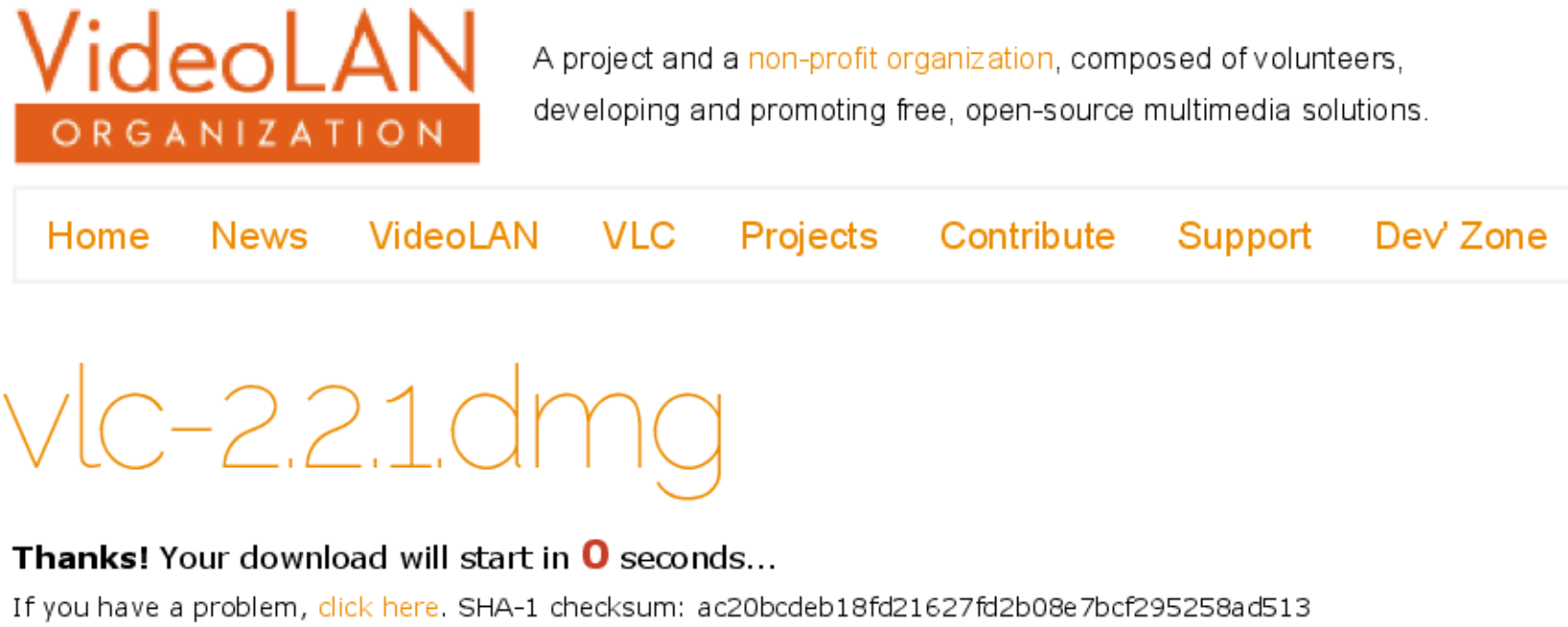
  abandonné à cause d'une « faiblesse technique »
- SHA-1 (NIST, 1994, 160 bits)
- SHA-256/384/512 (NIST, 2000) : condensés sur 256, 384 ou 512 bits
- Whirlpool (512 bits, calculs dans $\mathbb{F}_{256}$)

On recommande de nos jours au moins 256 bits.

✔ *Définition et exemples*

☛ *Propriétés cruciales*

Intégrité des téléchargements



`http://get.videolan.org/vlc/2.2.1/macosx/vlc-2.2.1.dmg`

Il suffit de calculer le résumé du fichier téléchargé et de le comparer au résumé indiqué pour être sûr que le fichier est intègre.

Il est *très improbable* qu'un fichier corrompu ait le même résumé que le fichier original.

Chiffrement des mots-de-passe

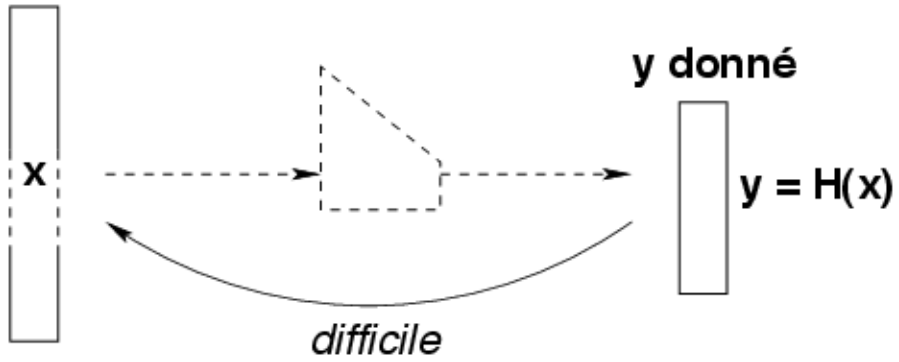
Un **mot-de-passe** est en général nécessaire pour accéder à un système informatique (compte étudiant ou site Internet payant).

Pour des raisons de sécurité, les mots-de-passe des utilisateurs ne sont jamais stockés en clair dans un fichier ; le résumé est stocké à sa place. On dit, improprement, qu'ils sont ≪ chiffrés. ≫

Pour accéder au système, l'utilisateur doit entrer son mot-de-passe : si le résumé stocké correspond au résumé du mot-de-passe entré, l'accès au système est accordé.

Autrefois, sous Unix, le fichier **`/etc/passwd`**, en lecture pour tous, listait les résumés MD5 des mots-de-passe de chaque utilisateur. Désormais, ces résumés MD5 (ou autre) sont stockés dans le fichier **`/etc/shadow`**, qui n'est lisible que par **`root`**.

Préservation des mots-de-passe



Étant donné un résumé y, il est **difficile** de calculer un x qui lui correspond.

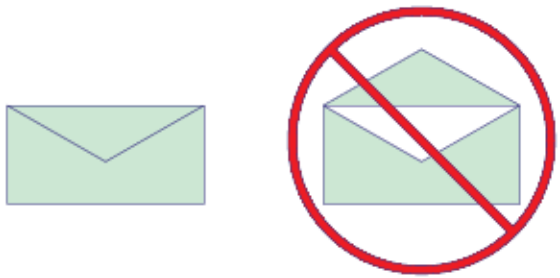
`root` ne doit pas être en mesure de trouver mon mot-de-passe utilisateur *(ou un mot-de-passe équivalent)*, même s'il connaît son résumé. Sinon, il pourra s'introduire à ma place dans un autre système

- qui utilise la même technique d'accès
- et pour lequel j'utilise (bêtement) le même mot-de-passe.

L'usage de MD5 doit être proscrit !

Retour à une propriété fondamentale : l'intégrité

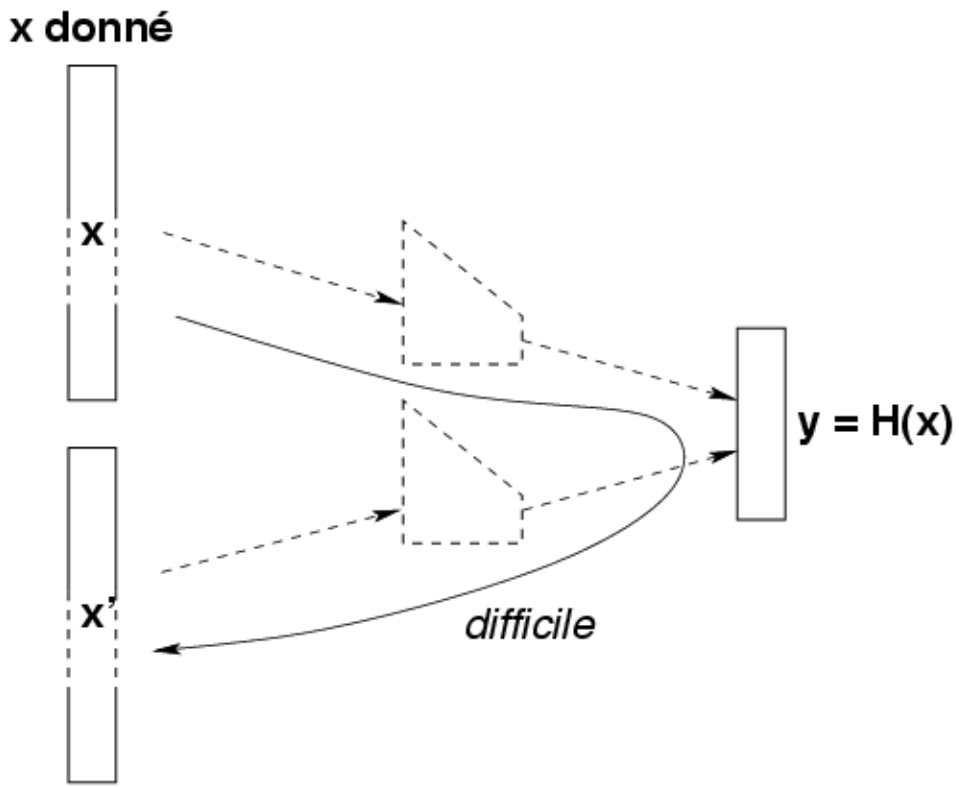
Intégrité : Garantir qu'un message (un document, ou encore un fichier) n'a pas subi de modification (aussi bien accidentelle qu'intentionnelle)



Comment assurer l'intégrité de son disque dur ?

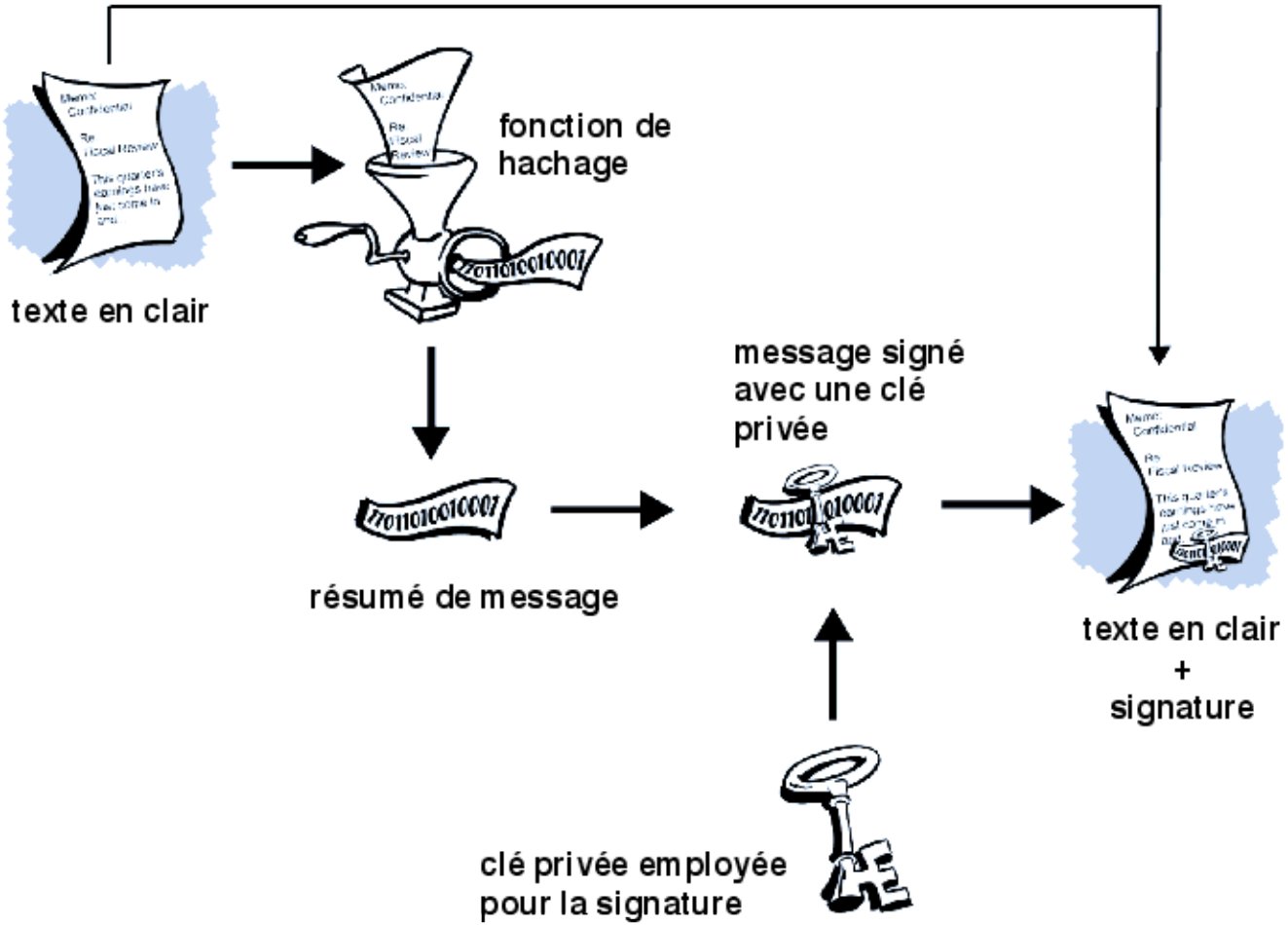
Nouvelle exigence : Résistance à la seconde préimage

Pour garantir l'intégrité, il faut que la fonction de hâchage utilisée soit résistante au calcul d'une seconde préimage.



Étant donné un texte x, il est **difficile** de calculer un autre texte x' qui a le même résumé.

Application plus cruciale : la signature électronique

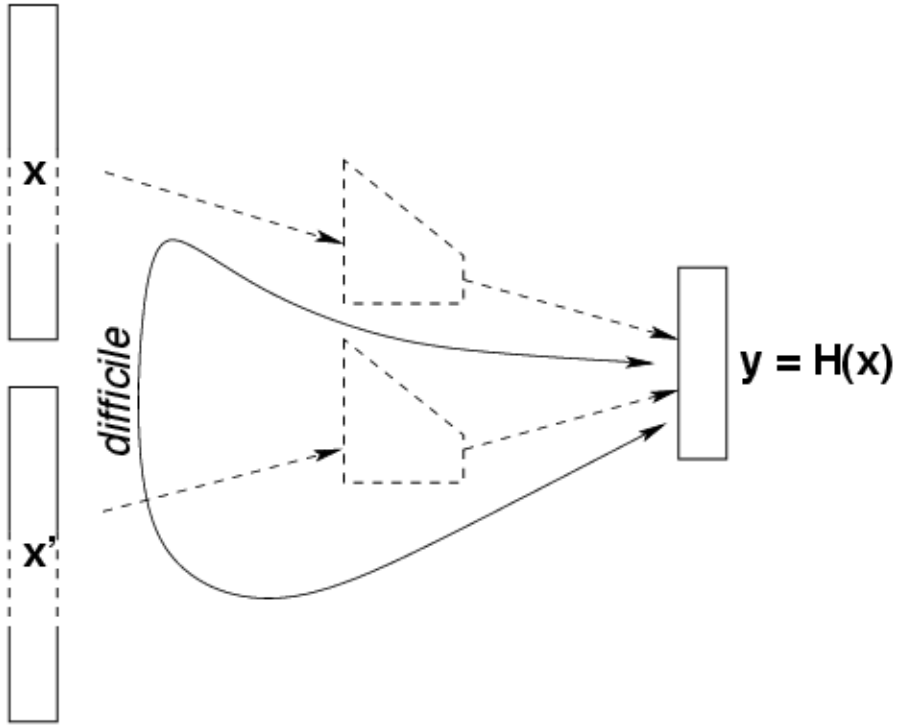


À nouveau ici, la fonction de hâchage doit être résistante au calcul d'une seconde préimage.

Pourquoi ?

Exigence supplémentaire : Résistance à la collision

La fonction de hâchage doit aussi être résistante aux collisions :



Il est **difficile** de calculer deux textes x et x' qui ont le même résumé.

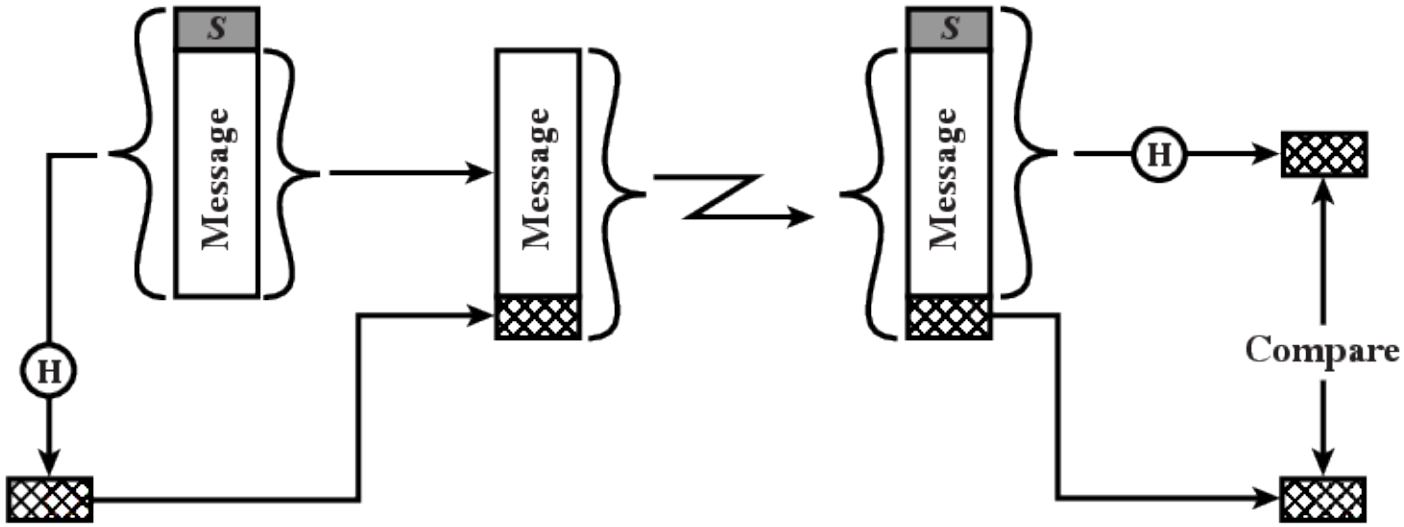
Sinon, il sera possible de faire signer un texte anodin x et d'obtenir un faux document x' correctement signé.

- ✔ *Définition et exemples*
- ✔ *Propriétés cruciales*
- ☛ *Autres applications*

Travaux pratiques : authentification non-interactive d'un message

Un HMAC (pour « keyed-hash message authentication code ») est un moyen de vérifier simultanément l'intégrité de données et l'authenticité d'un message. Ce système s'appuie sur une **clef secrète partagée** entre l'émetteur et le récepteur du message.

Le message transmis (en clair) est accompagné d'un code obtenu en calculant le résumé du message auquel on accole la clef secrète S.



La taille de la clef secrète doit être redhibitoire (au moins 128 bits).

Chiffrement par mot de passe (PBE pour « Password Based Encryption »)

Lorsque l'on souhaite ne garder aucune trace électronique d'une clef, pour parer par exemple à un vol éventuel, on utilise un **mot-de-passe** mémorisable à partir duquel la clef peut être regénérée à volonté.

Le mot-de-passe ne peut être utilisé directement comme clef, car il comporte souvent des motifs liés à la langue sur lesquels des attaques peuvent s'appuyer.

Un moyen simple de générer une clef à partir d'un mot-de-passe est de lui appliquer une fonction de hâchage.

MD5, par exemple, fournira une clef de 128 bits...

Chiffrement par mot de passe : PKCS #5

Compte tenu que le nombre de mots-de-passe est en pratique limité, l'*attaque par dictionnaire* consiste à construire toutes les clefs possibles, afin de les tester exhaustivement sur les messages interceptés.

La technique PBE vise à rendre plus coûteuse et plus systématique la fabrique de la table des clefs. Elle s'appuie sur deux paramètres *transmis en clair* avec le message chiffré :

- le *sallage aléatoire* qui masque le mot-de-passe : une attaque exhaustive devra, pour chaque sel, générer l'ensemble des clefs possibles, avant de les tester.
- le *nombre d'itérations* de la fonction de hâchage : la fabrique de la clef pour un mot-de-passe donné est alors plus coûteuse en temps, ce qui n'est pas génant pour les utilisateurs légitimes, mais l'est pour les adversaires qui doivent fabriquer des millions de clefs.

Ce qu'il faut retenir

① La cryptographie moderne distingue trois problématiques fondamentales : **intégrité**, **confidentialité**, **authentification**.

② Il existe deux grandes familles de chiffrements, toutes deux indispensables en pratique : **symétrique** ou **asymétrique**.

③ Les **fonctions de hâchages** sont utilisées pour garantir l'*intégrité* ou l'*authentification* des documents électroniques, mais aussi la *confidentialité* des mots-de-passe utilisateurs. Elles interviennent aussi dans systèmes de *signatures électroniques* (à clef publiques).

④ Il faut au moins **128 bits** pour une clef secrète mais, comme nous allons l'expliquer, **256 bits** pour un résumé.

Attaque des anniversaires

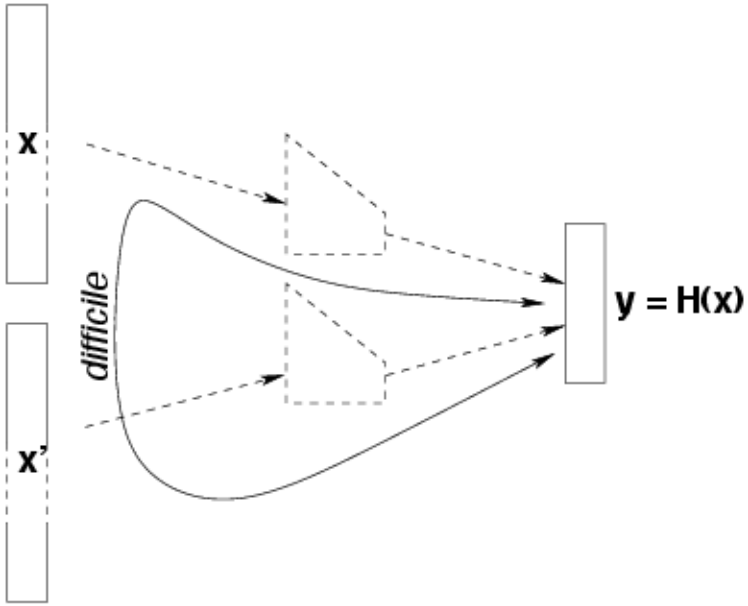
Master Informatique — Semestre 2 — UE optionnelle de 3 crédits

Année 2017-2018

Version du 13 septembre 2017

Principe

L'**attaque des anniversaires** consiste à faire signer à un tiers un **message correct** mais pour lequel on connait un **message frauduleux** ayant la même empreinte.



Problème sous-jacent :

- Fonction de hachage $H : \{0,1\}^t \rightarrow \{0,1\}^n$ avec $t > n$.
- k messages aléatoires : $x_1, ..., x_k$ et leur résumés : $H(x_1), ..., H(x_k)$

La probabilité $p(k)$ d'obtenir une collision vaut

$$p(k) = 1 - \frac{(2^n)!}{(2^n - k)!} \cdot \frac{1}{2^{n \times k}} \geqslant 1 - e^{-\frac{k(k-1)}{2^{n+1}}}$$

Calcul de $p(k)$

$1 - p(k)$ est la probabilité qu'il n'y ait pas de collision.

Le nombre de tirages différents est $(2^n)^k = 2^{n \times k}$.

Le nombre de tirages sans collision est

$$2^n \times (2^n - 1) \times (2^n - 2) \times \ldots \times (2^n - k + 1) = (2^n)!/(2^n - k)!$$

Ainsi

$$1 - p(k) = \frac{(2^n)!/(2^n - k)!}{2^{n \times k}}$$

Approximation de $p(k)$

Le nombre N de tirages sans collision est

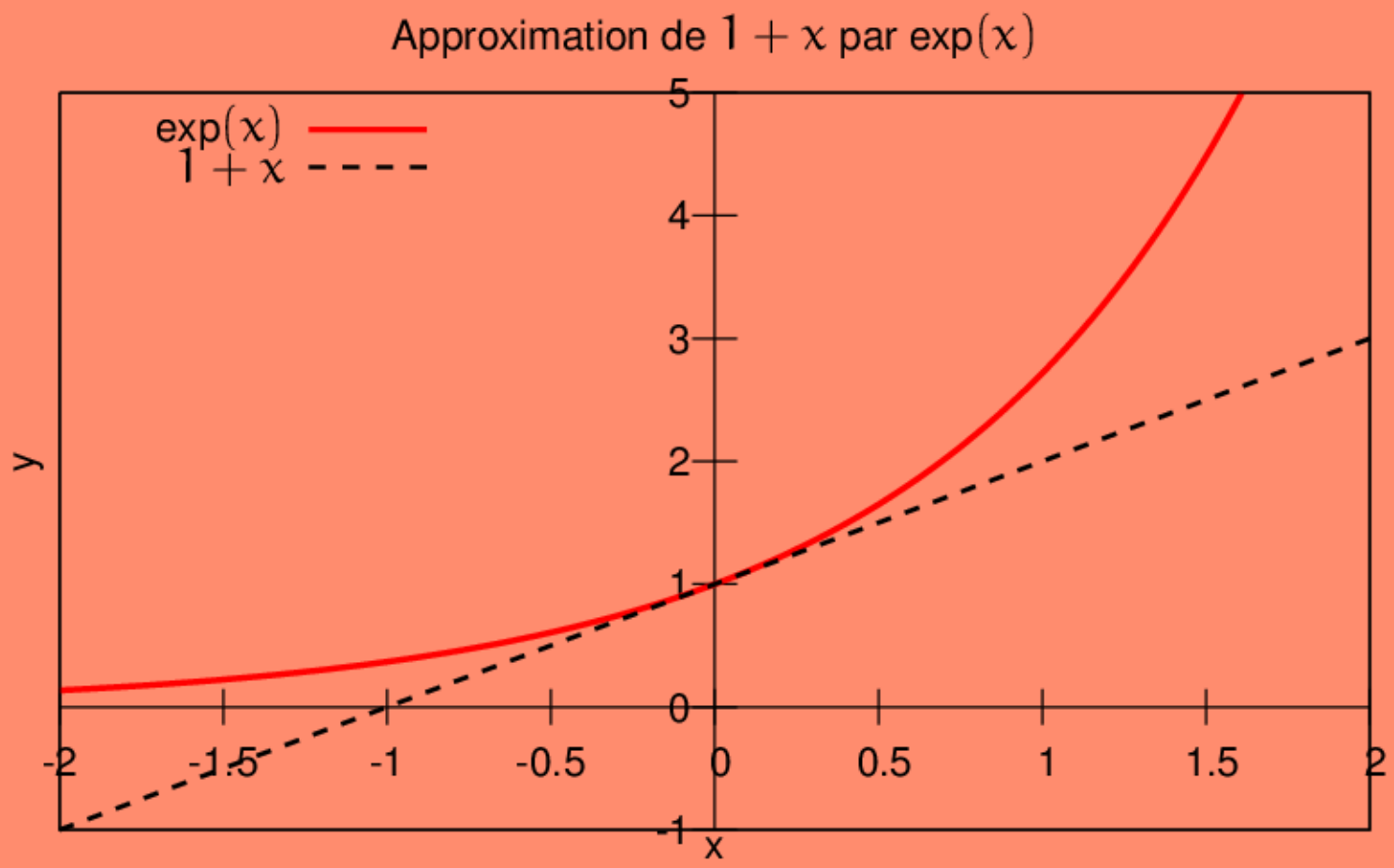
$$\begin{aligned} N &= 2^n \times (2^n - 1) \times (2^n - 2) \times \ldots \times (2^n - k + 1) \\ &= (2^n)^k \times (1 - \tfrac{1}{2^n}) \times (1 - \tfrac{2}{2^n}) \times \ldots \times (1 - \tfrac{k-1}{2^n}) \end{aligned}$$

Or $1 + x \sim e^x$ lorsque x est proche de 0.

Plus précisement, on connait le développement limité $e^x = 1 + x + o(x)$ pour x voisin de 0.

De plus, la courbe de l'exponentielle montre aussi que $1 + x \leqslant e^x$.

Approximation de $p(k)$



Approximation de $p(k)$

Donc

$$\begin{aligned} N &= (2^n)^k \times (1 - \tfrac{1}{2^n}) \times (1 - \tfrac{2}{2^n}) \times \ldots \times (1 - \tfrac{k-1}{2^n}) \\ &\leqslant (2^n)^k \times e^{-\frac{1}{2^n}} \times e^{-\frac{2}{2^n}} \times \ldots \times e^{-\frac{k-1}{2^n}} \\ &\leqslant 2^{n \times k} \times \prod_{i=1}^{i=k-1} e^{-\frac{i}{2^n}} \\ &\leqslant 2^{n \times k} \times e^{-\sum_{i=1}^{i=k-1} \frac{i}{2^n}} \\ &\leqslant 2^{n \times k} \times e^{-\frac{1}{2^n} \sum_{i=1}^{i=k-1} i} \\ &\leqslant 2^{n \times k} \times e^{-\frac{k(k-1)}{2^{n+1}}} \end{aligned}$$

D'où

$$\begin{aligned} 1 - p(k) &\leqslant 2^{n \times k} \times e^{-\frac{k(k-1)}{2^{n+1}}} \times \frac{1}{2^{n \times k}} \\ &\leqslant e^{-\frac{k(k-1)}{2^{n+1}}} \end{aligned}$$

✔ *Calcul et approximation de* $p(k)$

☛ *Pour quels* k *a-t-on* $p(k) \geqslant 1/2$ *?*

Application

Problème sous-jacent :

- Fonction de hachage $H : \{0,1\}^t \longrightarrow \{0,1\}^n$ avec $t > n$.
- k messages aléatoires $x_1, ..., x_k$

La probabilité d'obtenir une collision vaut

$$p(k) \geqslant 1 - e^{-\frac{k(k-1)}{2^{n+1}}}$$

À partir de quel k, a-t-on $p \geqslant 1/2$?

$$p \geqslant 0.5 \text{ si } k \geqslant 2^{(n+1)/2}$$

Il faut donc que $2^{n/2}$ soit un nombre redhibitoire.

On conseille de nos jours au moins $n \geqslant 256$ bits.

Justification

On a $p(k) \geqslant 1/2$ si $1 - p(k) \leqslant 1/2$, c'est-à-dire $\frac{1}{1-p(k)} \geqslant 2$.

D'autre part on sait que $p(k) \geqslant 1 - e^{-\frac{k(k-1)}{2^{n+1}}}$, c'est-à-dire

$$\frac{1}{1-p(k)} \geqslant e^{\frac{k(k-1)}{2^{n+1}}}$$

Ainsi on a $p(k) \geqslant 1/2$ dès que $e^{\frac{k(k-1)}{2^{n+1}}} \geqslant 2$.

Supposons que $k - 1 \geqslant 2^{(n+1)/2}$. Alors $e^{\frac{k(k-1)}{2^{n+1}}} \geqslant e^1 > 2$.

Bémol

Les formules présentées ici évaluent le risque de pouvoir trouver une collision simplement en tirant des messages au hasard et en calculant leur résumé.

Elle suppose que l'ensemble des messages aléatoires et de leur résumé est stocké jusqu'à ce qu'une collision soit trouvée.

L'espace de stockage peut être redhibitoire !

En pratique, *une telle collision a peu de chance d'être exploitable* car les deux messages obtenus doivent être **significatifs**. De plus, l'un doit être **acceptable** ou anondin et l'autre **bénéfique** pour la personne malveillante.

Stratégie de Yuval

1. L'attaquant choisit un message frauduleux et un message anodin que la victime acceptera de signer.
2. Il génère ensuite $2^{n/2}$ variantes du message anodin (en le modifiant très légèrement) ; idem avec le message frauduleux.
3. La probabilité qu'un des messages frauduleux et qu'un des messages anodins aient la même empreinte est supérieure à $0,5$.
4. L'attaquant fait signer cette variante du message anodin par la victime.
5. Il récupère la signature du message et l'accole à la variante du message frauduleux.