Cours: Sécurité Informatique

2022-2023

Chapitre 02 : Initiation à la cryptographie

04 - Fonctions de Hachage et Signatures numériques

04 - Fonctions de Hachage et Signatures numériques

Introduction

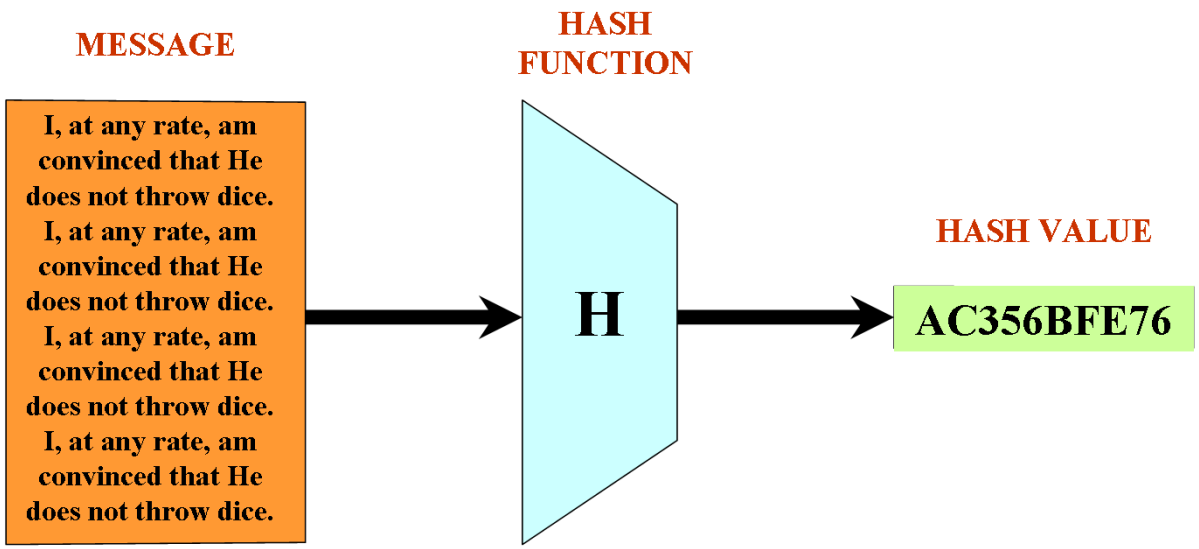
- On nomme **fonction de hachage** une fonction particulière qui, à partir **d'une donnée** fournie en entrée, calcule une ***empreinte*** servant à **identifier rapidement**, la donnée initiale.
  - Calculer l'empreinte d'une donnée ne doit coûter qu'un temps **négligeable.**
- Généralement, une fonction de hachage "**H**" transforme une **entrée** de données "x" d'une dimension variable, et donne comme résultat une **sortie** de données "y" d'une dimension fixe (y = H(x)).
- Évidemment, il doit être impossible de trouver x à partir de y.

Introduction

- Une fonction de hachage doit par ailleurs **éviter** autant que possible les **collisions** (états dans lesquels des données différentes ont une empreinte **identique**).

- La taille des empreintes est dans certains cas bien **plus longue** que celle des données initiales.
  - Un mot de passe dépasse rarement une longueur de 8 caractères, mais son empreinte peut atteindre une longueur de plus de 100 caractères.

Définitions



- Une fonction de hachage doit remplir quelques conditions de base :
  1) L'entrée peut être de dimension variable.
  2) La sortie doit être de dimension fixe.
  3) H(x) doit être relativement facile à calculer.
  4) H(x) doit être **sans collision**.
  5) H(x) doit être **résistance à la préimage (à sens unique)**.
  6) H(x) doit être **résistante à la seconde préimage**.

Définitions

Definition (Collision resistance)

Il est difficile de trouver x et x' tel que

$$h(x) = h(x')$$

- L'attaquant ne doit pas être en mesure de trouver deux messages différents (x; x') tels que H(x) = H(x'), en moins de ($2^{n/2}$) opérations (paradoxe des anniversaires).

Definition (Preimage resistance)

Étant donné une sortie y, il est difficule de trouver x tel que

$$h(x) = y$$

Résistante au préimage = à sens unique

- Soit un haché y donné, l'attaquant ne doit pas être en mesure de trouver un message x tel que H(x) = y, en moins de (2^n) opérations.

Définitions

Definition (2nd preimage resistance)

Étant donné une entrée x, il est difficule de trouver x' tel que

$$h(x') = h(x)$$

- Soit un message x et son haché y correspondant, l'attaquant ne doit pas être en mesure de trouver un message x' ≠ x tel que H(x') = y, en moins de (2^n) opérations.

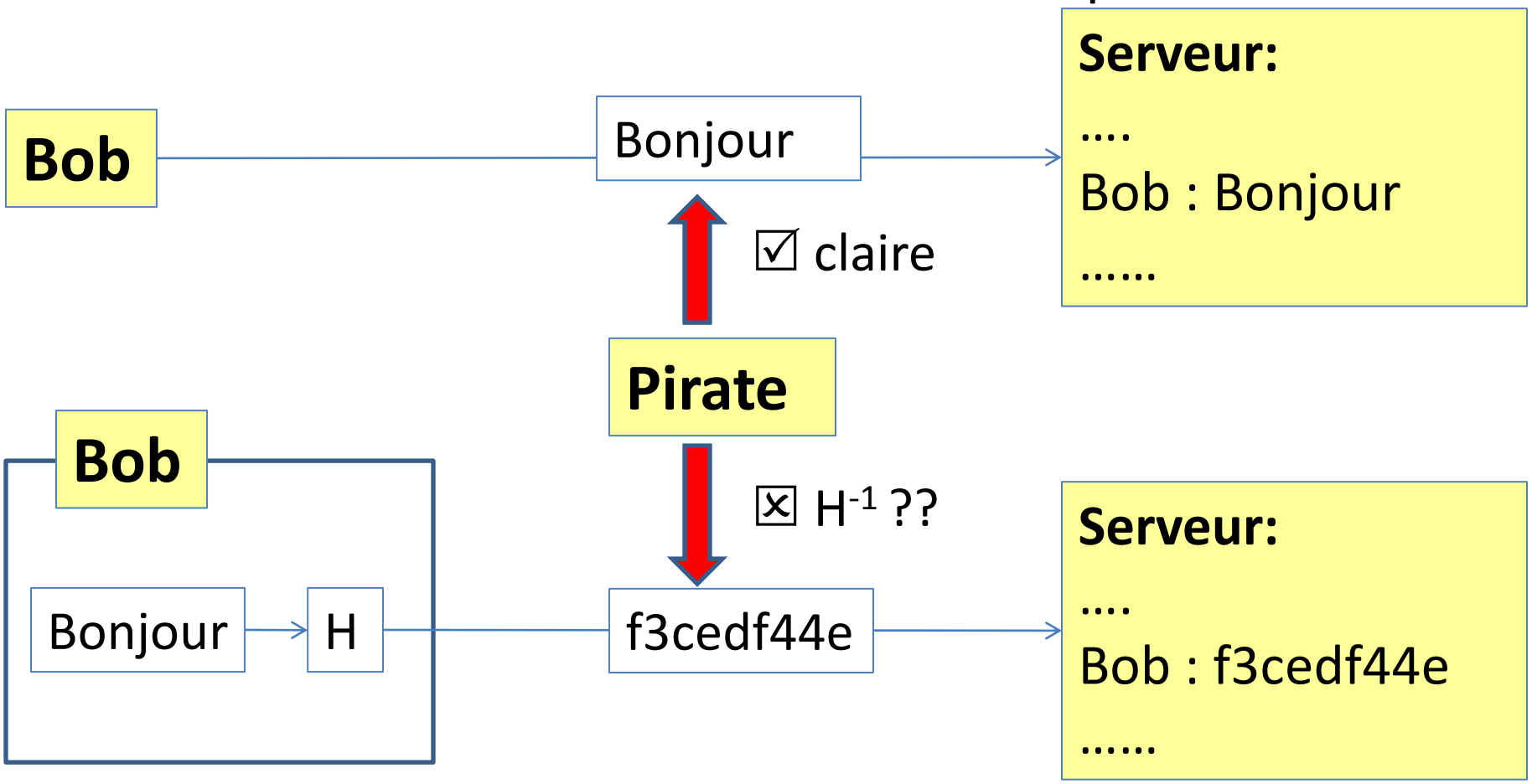
Le paradoxe des anniversaires

- **PROBLÈME:** de combien de personnes doit-on disposer pour avoir une bonne probabilité (P > 0.5) que deux personnes aient la même date d'anniversaire ?
- Il y'a 365 possibilités de dates.
- ... La réponse n'est pas très intuitive!
- **RÉPONSE: seulement 23 personnes pour P > 0.5 !**
- Ceci implique que **statistiquement** est il est relativement fréquent de trouver par une **génération aléatoire** deux messages x et x' qui ont le même haché (empreinte).

  $\Rightarrow$ Le problème de collision doit être pris au sérieux par tout algorithme de hachage.

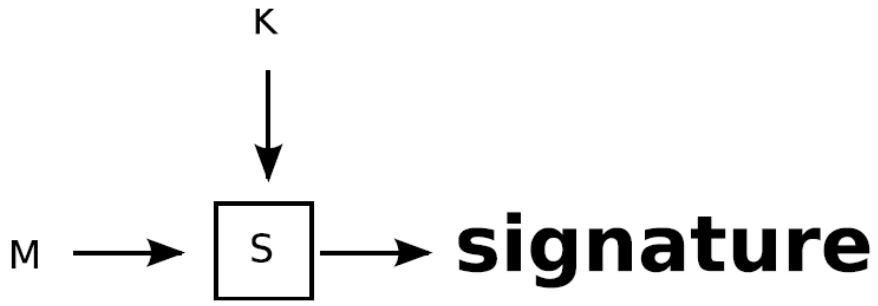
Applications des Fonctions de Hachage

- **Vérification de l'intégrité des fichiers ou des messages.**
- **Protection de mots de passe:** au lieu de stocker tous les mots de passe dans le serveur pour l'authentification d'utilisateurs, il vaut mieux stocker le haché des mots de passe.

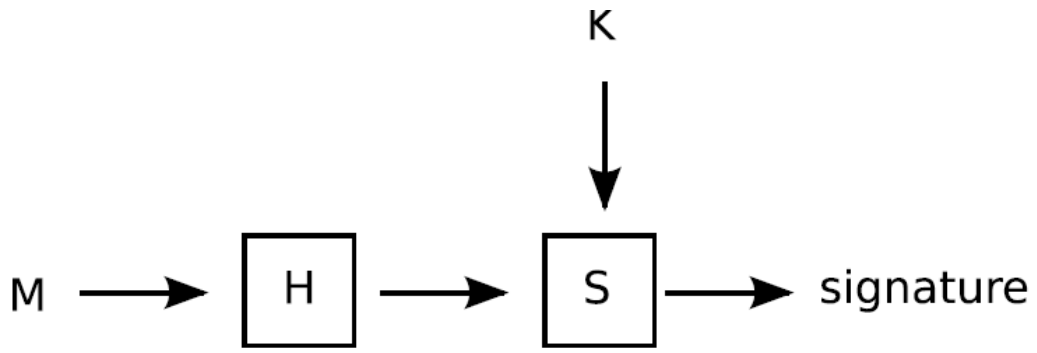


Applications des Fonctions de Hachage

- **Signatures:** L'algorithme de signature est appliqué sur l'empreinte du message M. Les fonctions de hachage permettent dans ce cas d'améliorer la vitesse et la sécurité des schémas de signature.



Taille signature = taille M
Signatures trop grosses, pas pratiques.



Paradigme "Hash-and-Sign", produit de petites signatures. La sécurité dépend alors aussi de H.

Applications des Fonctions de Hachage

- **Confirmation de connaissance/engagement sur une valeur:** si quelqu'un veut prouver qu'il connait un secret sans le révéler dans l'immédiat, il peut publier le haché de ce secret. Une fois le secret révélé, il est facile de vérifier ses dires.
- **Génération de nombres pseudo-aléatoires et dérivation de clés**: $K_i = H(f(K, i))$, K est la clé secrète source.
- De nombreuses autres applications existent pour les fonctions de hachage, notamment dans le domaine d'authentification, identification, et du commerce électronique.

Schéma général

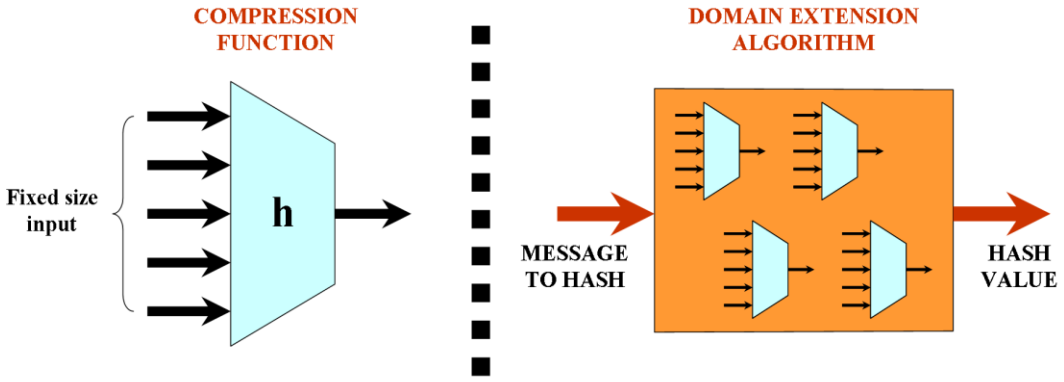
- La majorité des fonctions de hachage sont composées de deux éléments:

1) **Une fonction de compression h:** une fonction dont la taille d'entrée et de sortie est fixe.

Definition (Fonction de compression)

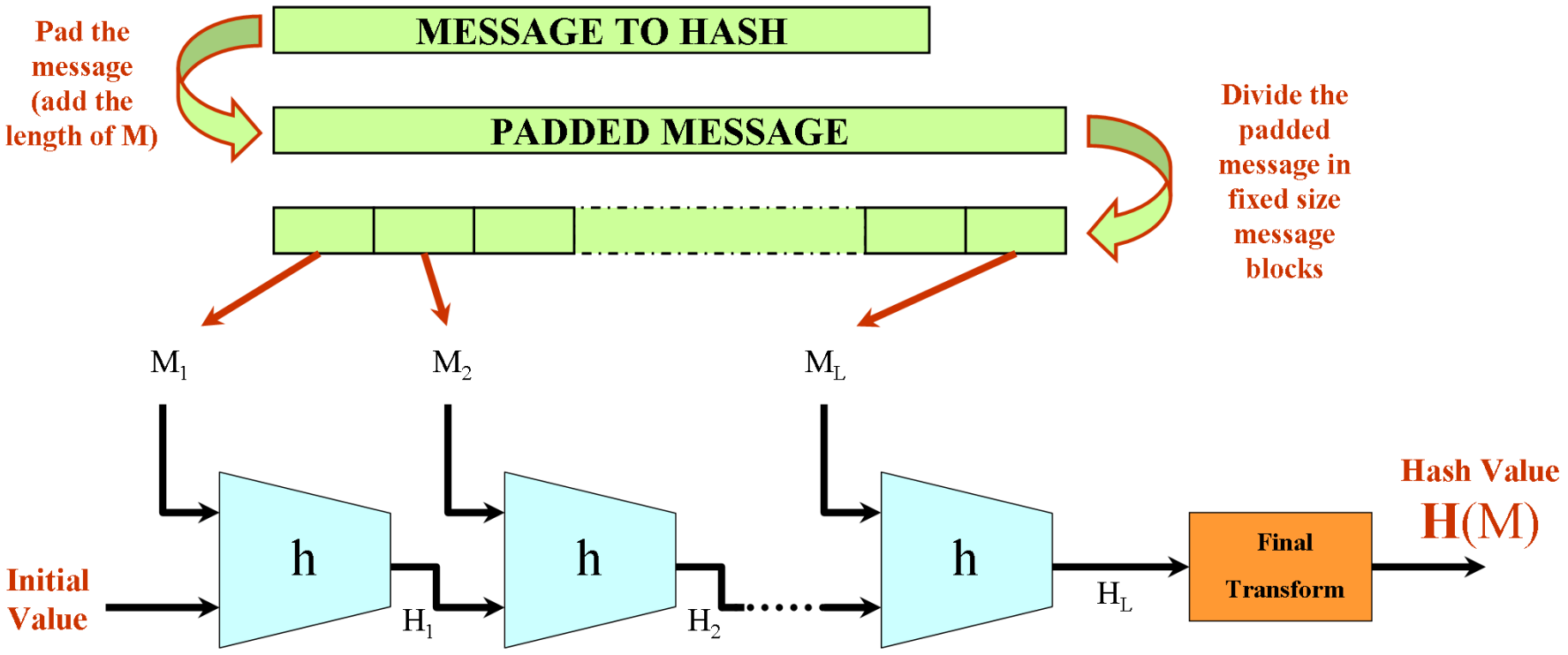
Une fonction $f : \{0,1\}^m \rightarrow \{0,1\}^n$ où $n < m$ est appellée une fonction de compression.

2) **Un algorithme d'extension :** un processus (généralement itératif) utilisant la fonction de compression h pour que la fonction de hachage H puisse hacher des messages de taille arbitraire.



L'algorithme d'extension de Merkle-Damgard

- L'algorithme d'extension le plus connu et le plus utilisé jusqu'à maintenant est l'algorithme itératif de **Merkle-Damgard**.



Les fonctions de compression

Il existe trois grands groupes de fonctions de compression:

- **Ad-hoc**: grande majorité des fonctions utilisées et standardisées (MD, SHA...). Très rapides, mais on ne peut avoir confiance en leur sécurité qu'après une longue analyse par la communauté des cryptographes.
- **Fondées sur un chiffrement par bloc**: un peu plus lentes que les fonctions ad-hoc, elles permettent souvent de prouver leur sécurité en supposant le chiffrement par bloc utilisé comme idéal.
- **Fondées sur un problème difficile:** généralement très lentes, leur sécurité repose totalement sur la difficulté de résoudre un problème difficile (factorisation, résolution de systèmes algébriques, etc.).

Hachage sans clé : **le MD5**

- MD5 (**Message Digest 5**) est une fonction de hachage cryptographique qui calcule, à partir d'un fichier numérique, son **empreinte numérique** (de **128 bits**) avec une probabilité **très forte** que deux fichiers différents donnent deux empreintes différentes.
- Actuellement, MD5 **n'est plus considéré comme sûr** au sens cryptographique.
- On suggère maintenant d'utiliser plutôt des algorithmes tels que SHA-256, RIPEMD-160 ou Whirlpool.

Hachage sans clé : **le MD5**

- Cependant, la fonction MD5 reste encore **largement utilisée** comme outil de **vérification** lors des téléchargements et l'utilisateur peut valider **l'intégrité** de la version téléchargée grâce à l'empreinte.
- Ceci peut se faire avec un programme comme md5sum pour MD5 et sha1sum pour SHA-1.

Hachage sans clé : le MD5

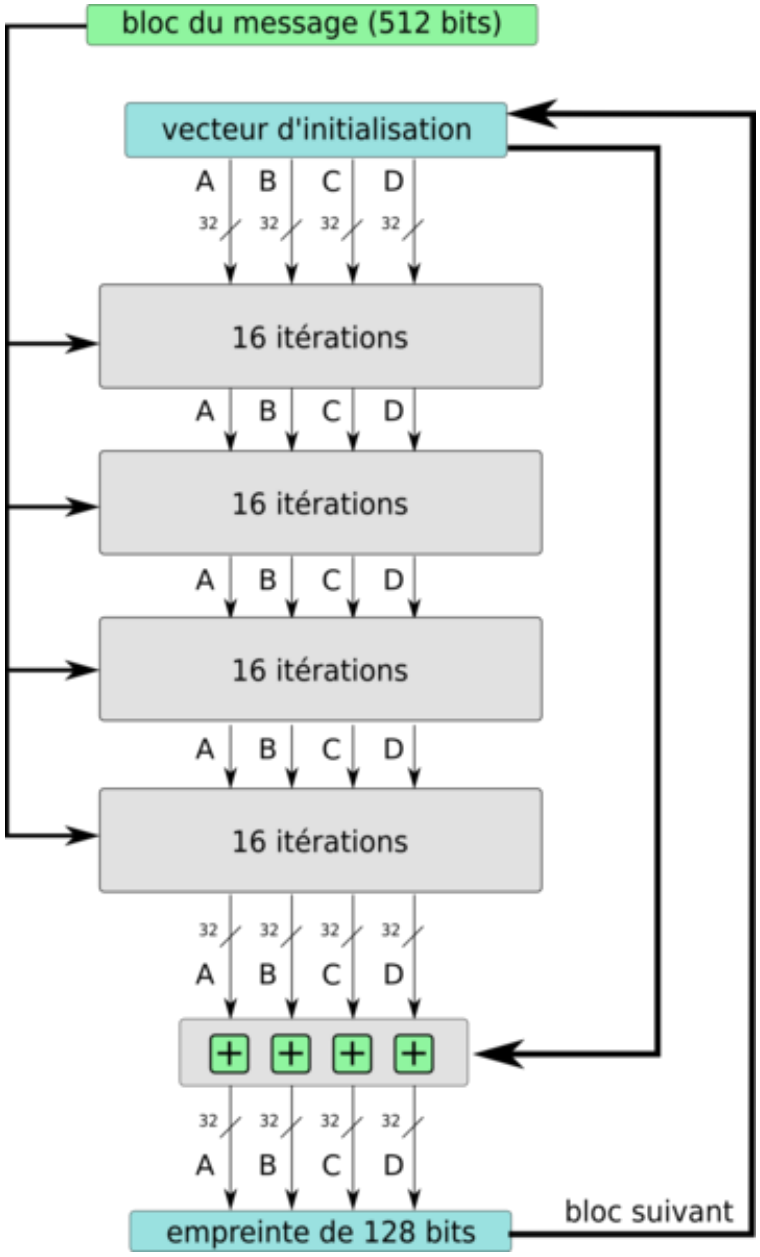
- MD5 travaille avec un message de taille **variable** et produit une empreinte de **128 bits**.
- Le message est divisé en blocs de 512 bits, on applique un **remplissage** de manière à avoir un message dont la longueur est un multiple de 512. Le remplissage se présente comme suit :
  1) On ajoute un '1' à la fin du message.
  2) On ajoute une séquence de '0' (le nombre de zéros dépend de la longueur du remplissage nécessaire).
- Ce remplissage est toujours appliqué, même si la longueur du message peut être divisée par 512. Le message a maintenant une taille en bits multiples de 512.

Hachage sans clé : **le MD5**

- L'algorithme principal travaille avec un état sur 128 bits. Il est lui-même divisé en 4 mots de 32 bits : *A*, *B*, *C* et *D*. Ils sont initialisés au début avec des constantes.
- L'algorithme utilise ensuite les blocs provenant du message à hacher, ces blocs vont modifier l'état interne.
- Les opérations sur un bloc se décomposent en quatre rondes (tours), elles-mêmes subdivisées en 16 itérations similaires basées sur : une fonction non linéaire *F* qui varie selon la ronde, une addition et une rotation vers la gauche.
- Les quatre fonctions non linéaires disponibles sont :

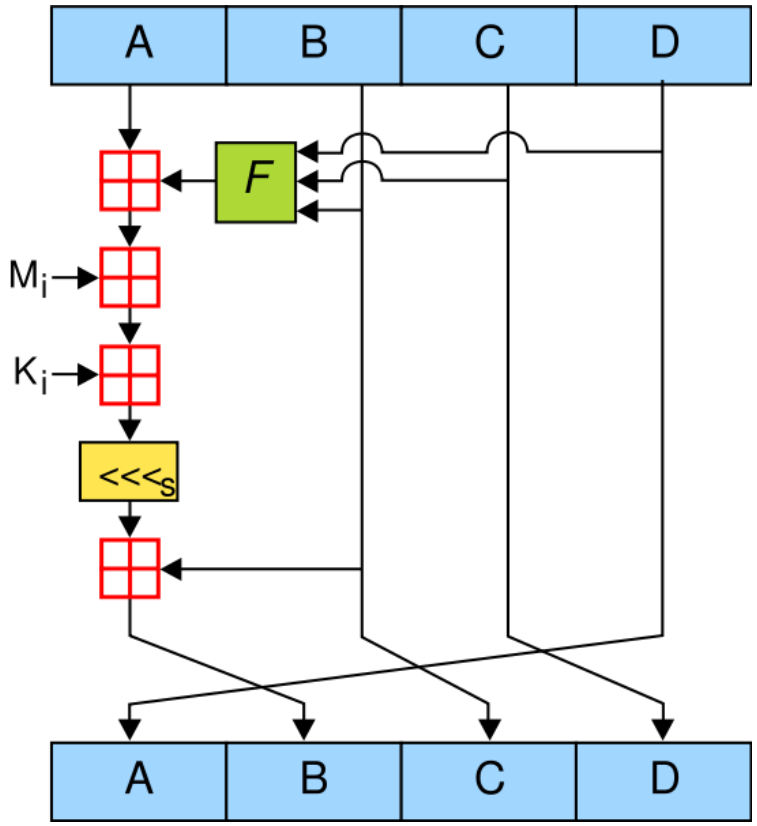
$$F(B,C,D) = (B \wedge C) \vee (\neg B \wedge D)$$
$$G(B,C,D) = (B \wedge D) \vee (C \wedge \neg D)$$
$$H(B,C,D) = B \oplus C \oplus D$$
$$I(B,C,D) = C \oplus (B \vee \neg D)$$

Hachage sans clé : **le MD5**



Hachage sans clé : le MD5

- MD5 comprend 64 blocs de ce type, groupés en **quatre tours** de 16 itérations.
- M_i symbolise un bloc de 32 bits provenant du message à hacher et K_i est une constante de 32 bits, différentes pour chaque itération.



Hachage sans clé : **le MD5**

- Les valeurs initiales de A,B,C et D sont: **A**= 01234567, **B**= 89ABCDEF, **C**= FEDCBA98 et **D**= 76543210.
- Les valeurs de la rotation a chaque étape de 1 à 64 sont:

{7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23,6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21}

- Les valeurs K_i à chaque étape de 1 à 64 sont calculées comme suite :

$$K_i = INT(abs(Sin(i+1))*2^{32})$$

- L'algorithme permet à la fin de trouver une empreinte de 128 bits pour n'importe quel volume de données en entrée.

Hachage sans clé : le SHA-1

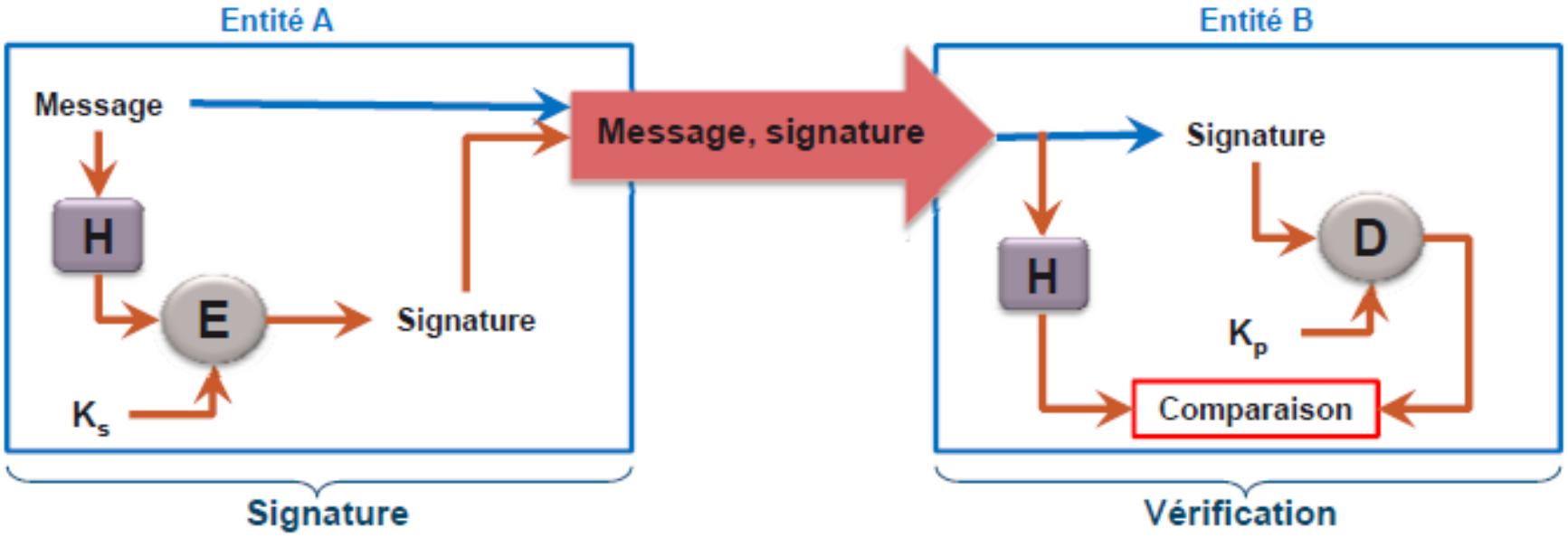
- Le **SHA-1** (Secure Hash Algorithm) est le successeur du SHA-0 qui a été rapidement mis de côté par le **NIST** pour des raisons de sécurité insuffisante.
- Elle produit un résultat (hash ou condensat) de 160 bits (20 octets).
- Le SHA-1 commence par ajouter à la fin du message un bit à 1 suivi d'une série de bits à 0, la séquence ainsi prolongée a une longueur multiple de 512 bits.
- L'algorithme travaille ensuite successivement sur des blocs de 512 bits.
- Pour chaque bloc l'algorithme calcule 80 tours (ou rondes) successifs et applique une série de transformations sur l'entrée.

La Signature Numérique

- Équivalent technologique de la signature manuscrite.
- Permet de signer et chiffrer facilement les documents électroniques.
- Elle assure :
  1) L'intégrité des données.
  2) L'identité de l'expéditeur.
  3) La non-répudiation par l'expéditeur.

La Signature Numérique

- Se base sur une opération inverse d'un chiffrement asymétrique:
  - L'expéditeur possède deux clés: clé publique K_p et clé sécrète K_s.
  - Le destinataire connaît la clé publique de l'expéditeur K_p.
  - La signature électronique a les propriétés suivantes: authentique, infalsifiable, non réutilisable, inaltérable et irrévocable.



La Signature Numérique

- L'algorithme de chiffrement asymétrique est appliqué sur le condensé du message (**utilisation d'une fonction de hachage**), ce qui assure :
  - Efficacité : plus rapide.
  - Compatibilité : taille du message à chiffré est fixe.
  - Intégrité : tout le message en condensé est chiffré.
- Tout algorithme de chiffrement asymétrique peut être utilisé avec n'importe quelle fonction de Hachage pour réaliser un schéma de signature.

La Signature Numérique: ElGamal

- **Paramètres:** p: grand nombre premier et g: générateur du groupe Z_p^*.
- **Génération de clés:** Clé secrète: a | $1 < a < p-2$;

  Clé publique: (p, g, A) | $A=g^a \bmod p$.
- **Signature:** Choisir un nombre aléatoire k | $1 < k < p-1$ et and $pgcd(k, p-1) = 1$;

  Calculer $r = g^k \bmod p$;

  Calculer $s = (H(m) - a.r)k^{-1} \bmod p - 1$ (si s=0 choisir un autre k); La signature du message m est : (r, s).
- **Vérification:**

  Vérifier si $0<r<p$ et $0<s<p-1$

  La signature est valide si $g^{H(m)} \equiv A^r.r^s \bmod p$

La Signature Numérique: DSA

Digital Signature Algorithm (DSA):

- L'algorithme de signature à clé publique DSA a été proposée par le NIST en 1991, il faisait partie de la spécification DSS (Digital Signature Standard).
- Le DSA est une variante de l'algorithme d'ElGamal et de Schnorr.

Génération de clés:

- p : nombre premier de longueur L bits avec $512 \leq L \leq 1024$ et L est divisible par 64.
- q: facteur premier de $p - 1$, de 160 bits
- Choisir $1 < h < p - 1$ aléatoirement | $g = h^{(p-1)/q} \bmod p > 1$
- Générer aléatoirement un x, avec $0 < x < q$
- Clé publique : (p, q, g, y) | $y = g^x \bmod p$
- Clé secrète : x

La Signature Numérique: **DSA**

Signature:

- Choisir un nombre aléatoire k, $1 < k < q$. (Ce k doit être aléatoire, secret, détruit après utilisation et jamais réutilisé).
- Calculer $r = (g^k \bmod p) \bmod q$. (si r=0 choisir un autre k).
- Calculer $s = k^{-1}(H(m) + r.x) \bmod q$ (où H(m) est le haché de m, par exemple avec SHA-256) (si s=0 choisir un autre k).
- La signature du message m est (r, s).

Vérification:

- Vérifier si $0 < r < q$ et $0 < s < q$.
- Calculer $u_1 = H(m).s^{-1} \bmod q$ et $u_2 = r.\ s^{-1} \bmod q$.
- Calculer $v = [(g^{u1}.\ y^{u2}) \bmod p] \bmod q$.
- Si v = r, alors la signature est vérifiée, et l'authentification réalisée.

Conclusion

- Les fonctions de hachage permettent de couvrir un aspect très important de la sécurité informatique, celui de l'intégrité des messages.
- De nouvelles techniques de hachage apparaissent quotidiennement basées sur de nouvelles techniques évoluées (Automates cellulaires, chaos…).
- Le problème de collision constitue le défie sécuritaire majeur de tout algorithme de hachage.
- Les signatures numériques obéissent a un système juridique et nécessitent des lois et des décrets pour régularisé et géré leurs utilisation, ces lois peuvent différent d'un pays a un autre.
- Leurs utilisation nécessite des mécanismes de distribution de signatures et de gestion des certificats.

05 - Les certificats numériques

Les certificats numériques

- Le certificat est une carte d'identité numérique. Il permet **d'associer** une **clé publique** à une **entité** (une personne, une machine, ...) afin d'en assurer la validité.
- Un certificat est délivré par un organisme appelé **autorité de certification** (**CA** : Certification Authority).
- Un certificat est un fichier émis par une CA composé de deux parties, une contenant des informations, l'autre contenant la signature de l'autorité de certification. Il comprend donc :
  - Nom, prénom, adresse email + informations diverses,
  - Clé publique de la personne,
  - Date de validité,
  - Nom de l'autorité de certification,
  - Signature de l'autorité de certification.

Les certificats numériques

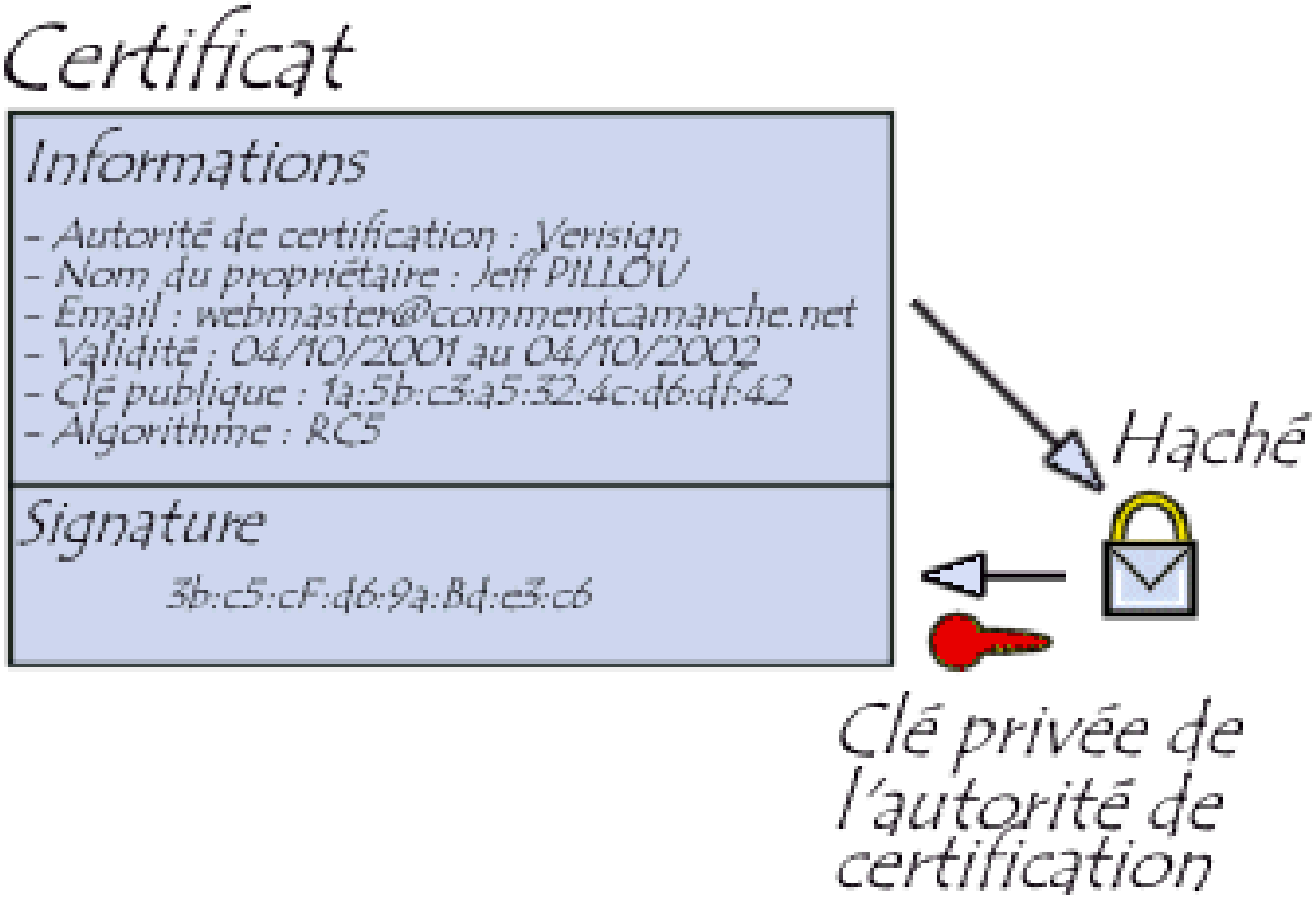
- L'ensemble de ces informations (informations + clé publique du demandeur) est signé par la CA.
- Une fonction de hachage crée une empreinte de ces informations, puis ce condensé est chiffré à l'aide de la clé privée de la CA.
- La vérification du certificat se fait à l'aide de la clé publique de l'autorité de certification et de la date de validité.
- Pour vérifier un certificat, il suffit de connaitre la clé publique de l'autorité émettrice.

Les certificats numériques

- La structure des certificats est normalisée par le standard X.509 de l'UIT, qui définit les informations contenues dans le certificat :
  - La version de X.509 à laquelle le certificat correspond,
  - Le numéro de série du certificat,
  - L'algorithme de chiffrement utilisé pour signer le certificat,
  - Le nom (DN, pour Distinguished Name) de l'autorité de certification émettrice,
  - La date de début de validité du certificat,
  - La date de fin de validité du certificat,
  - L'objet de l'utilisation de la clé publique,
  - La clé publique du propriétaire du certificat,
  - La signature de l'émetteur du certificat.

Les certificats numériques

- Certificat X.509 :

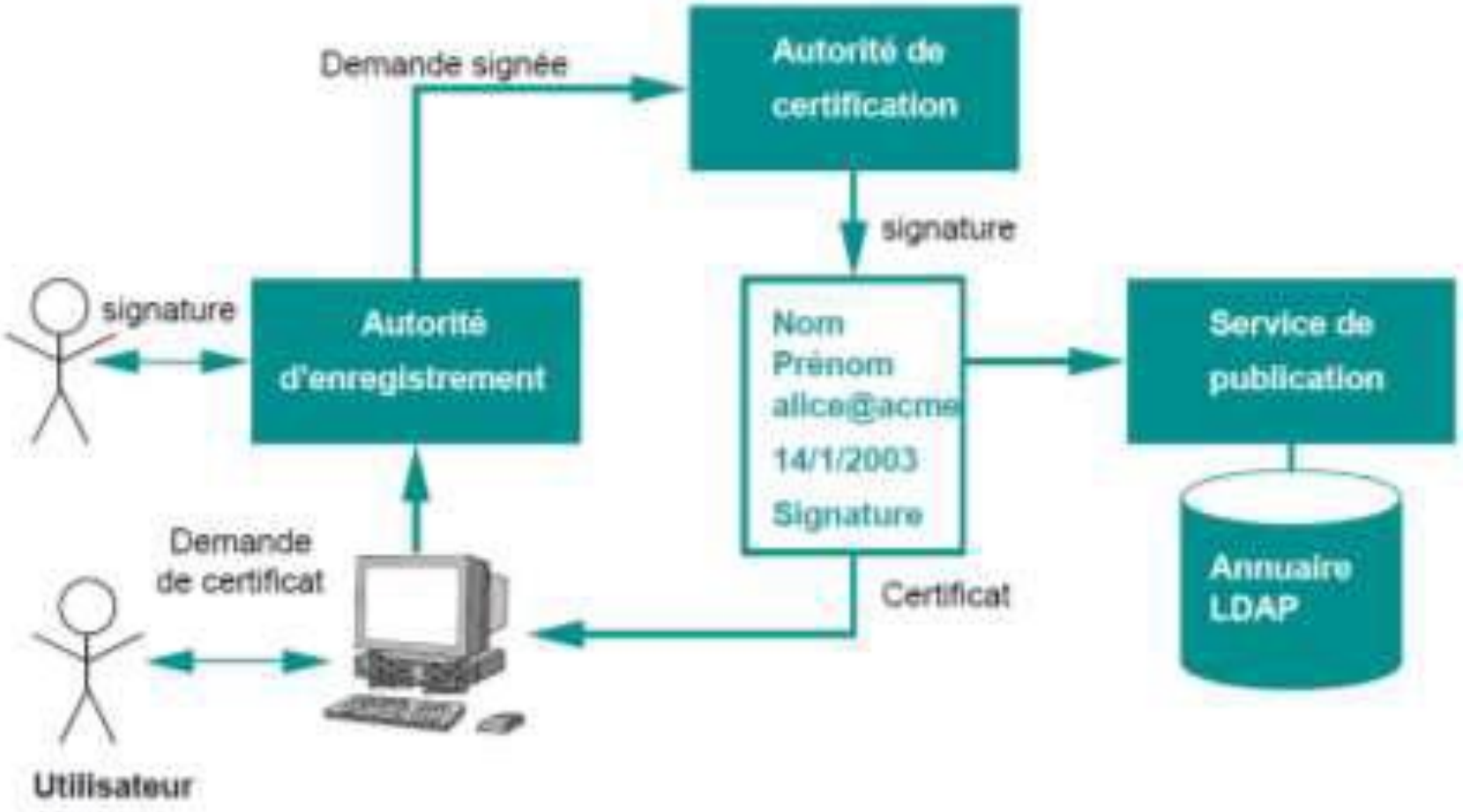


Autorités de certification et PKI

- Une Infrastructure à clés publiques ou Infrastructure de Gestion de Clefs ou encore Public Key Infrastructure (PKI), est un ensemble de composants physiques (ordinateurs, équipements cryptographiques, cartes à puces), de procédures humaines (vérifications, validation) et de logiciels (système et application) en vue de gérer le cycle de vie des certificats électroniques).

Autorités de certification et PKI

- Schéma d'Infrastructure de Gestion de Clés (PKI):



Autorités de certification et PKI

- Une PKI délivre un ensemble de services pour le compte de ses utilisateurs. Parmi eux :
  - Enregistrer et vérifier les demandes de certificats.
    - Autorité d'enregistrement.
  - Créer et distribuer des certificats.
    - Autorité de certification.
  - Vérification de validité de certificats.
    - Autorité de validation.
  - Gérer à tout moment l'état des certificats et prendre en compte leur révocation.
    - Dépôt de listes de certificats révoqués – CRL (Certificate Revocation List).
  - Publier les certificats dans un dépôt.
    - Dépôt de certificats (Annuaire).