Les Fonctions de Hashage en Cryptographie

Module : Cryptographie et Sécurité Informatique

Ayoub DEBBAGH & Ouassim CHAKIR

Université Moulay Ismail
Faculté des Sciences et Techniques, Errachidia
Filière : Cycle d'ingénieurs en Génie Informatique
Option : Génie Logiciel

Encadré par : Pr. F. AMOUNAS

December 3, 2024

Sommaire

Définition et Contexte

- **Qu'est-ce qu'une fonction de hachage ?**
  - Une fonction mathématique qui convertit des données de taille variable en une empreinte de taille fixe.
  - Fondement essentiel dans la sécurité des systèmes numériques.

- **Pourquoi sont-elles importantes ?**
  - Garantissent l'intégrité et l'authenticité des données.
  - Réduisent les grandes quantités de données en condensés pratiques.

Rôle des Fonctions de Hachage en Cryptographie

- **Applications principales :**
  - Vérification de l'intégrité des données (e.g., fichiers, messages).
  - Signature électronique pour garantir l'identité de l'expéditeur.
  - Sécurisation des mots de passe dans les bases de données.

- **Avantages clés :**
  - Rapide à calculer et efficace.
  - Fournit une empreinte unique pour chaque donnée.

Processus de Hachage Étape par Étape

1. **Entrée des données :** Texte, image ou fichier.
2. **Prétraitement :** Transformation de l'entrée en une séquence binaire.
3. **Application de l'algorithme :** Une série de calculs mathématiques est effectuée.
4. **Sortie :** Une empreinte unique de taille fixe.

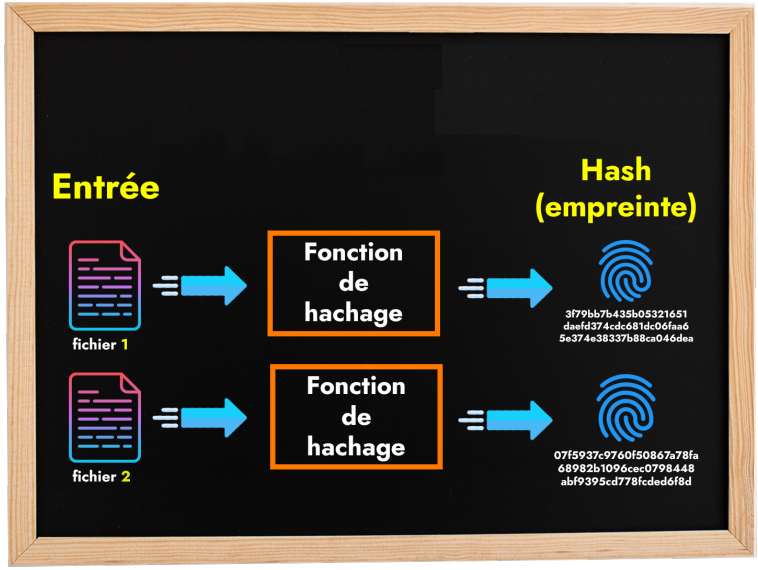


Figure 1: Processus de hachage

Transformation des Entrées en Sorties de Taille Fixe

- **Fonction de hachage :**
  - Un algorithme mathématique qui transforme une donnée d'entrée de taille variable en une empreinte de taille fixe.
  - Exemple : Une chaîne de caractères ou un fichier produit un haché de 256 bits.
- **Propriétés de la transformation :**
  - La sortie est toujours de taille fixe, quel que soit la taille de l'entrée.
  - La transformation est déterministe (même entrée = même sortie).

Illustration d'un Exemple Simple

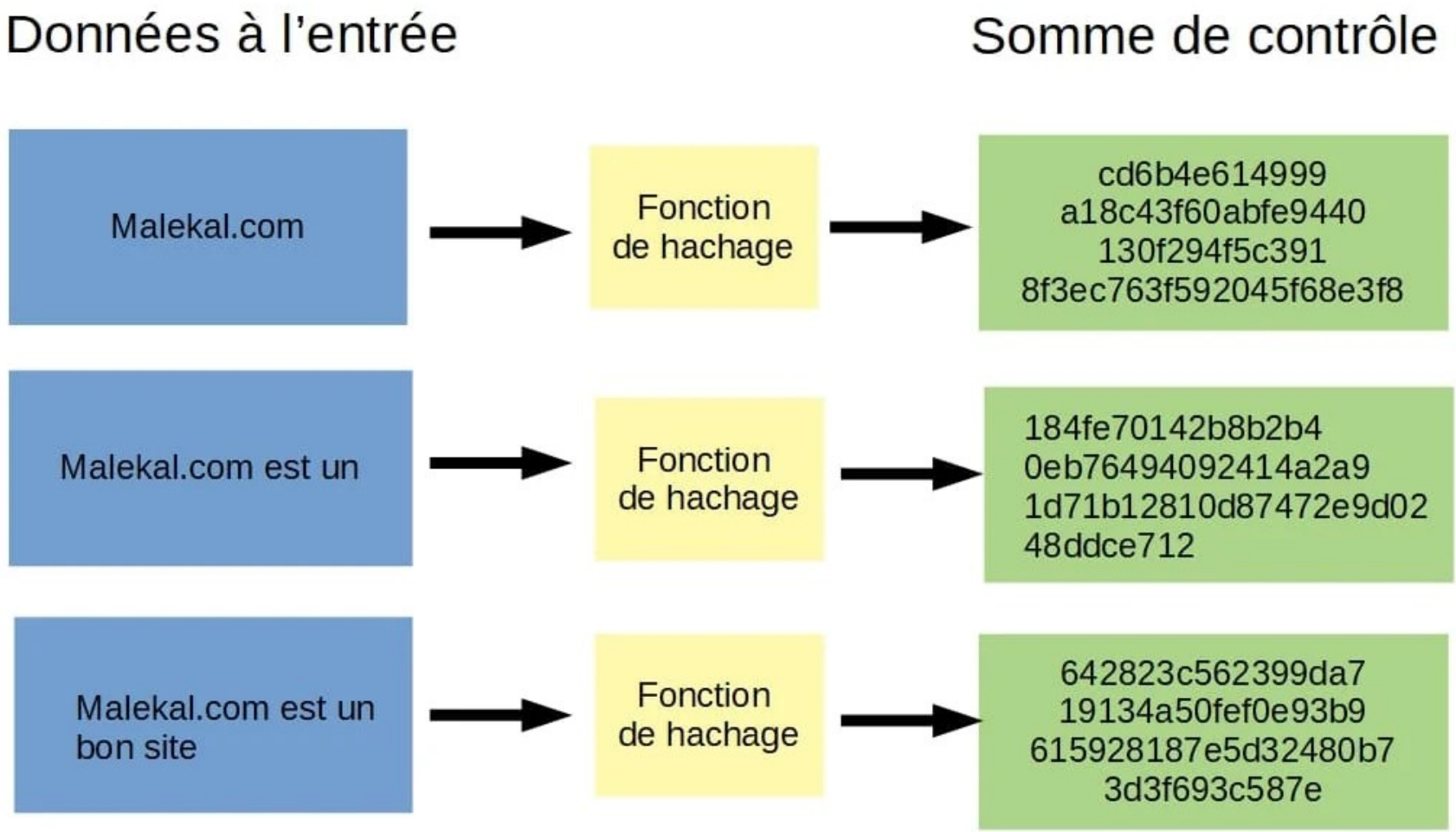


Figure 2: Exemple illustratif avec différentes entrées et leurs empreintes.

Les Propriétés des Fonctions de Hashage

- **Déterminisme:** Même message = même hash.
- **Rapidité:** Calcul rapide.
- **Résistance aux collisions:** Difficile de trouver deux messages avec le même hash.
- **Résistance à l'inversion:** Impossible de retrouver l'entrée.
- **Avalanche:** Petite modification → grand changement.

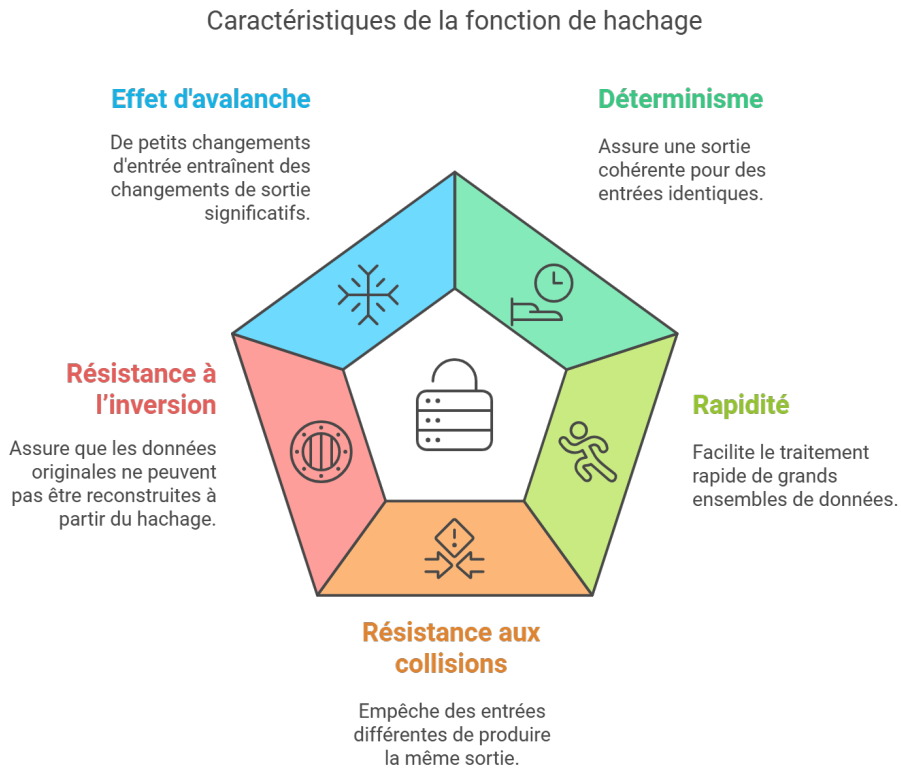


Figure 3: Les propriétés des fonctions de hashage

Les Applications en Cryptographie

- **Intégrité des données :** Le hashage permet de vérifier que les données n'ont pas été altérées pendant la transmission ou le stockage. *Exemple :* Utilisé dans les téléchargements de fichiers pour vérifier leur intégrité avec une empreinte hash.
- **Signature numérique :** Une fonction de hashage est utilisée pour créer une empreinte du message, qui est ensuite chiffrée avec une clé privée. Cela garantit l'authenticité et l'intégrité des documents. *Exemple :* Utilisé dans les certificats SSL/TLS pour sécuriser les communications web.
- **Gestion des mots de passe :** Les mots de passe sont stockés sous forme d'empreintes hashées dans les bases de données. Cela empêche de révéler les mots de passe originaux en cas de fuite de données. *Exemple :* Hashage avec un sel (salt) pour renforcer la sécurité contre les attaques par force brute.

Les Applications en Cryptographie (Suite)

- **Preuve de travail (Proof of Work) :** Dans les cryptomonnaies comme Bitcoin, le hashage est utilisé pour résoudre des problèmes mathématiques complexes qui valident les transactions.
  Exemple : Les mineurs calculent un hash répondant à certaines contraintes pour ajouter un bloc à la blockchain.
- **Hachage des messages (HMAC - Hash-based Message Authentication Code) :** Utilisé pour assurer l'intégrité et l'authenticité des messages à l'aide d'une clé secrète. Cela empêche qu'un message soit altéré ou falsifié pendant sa transmission.
  Exemple : Utilisé dans les protocoles de sécurité comme IPsec et SSL/TLS.

Introduction à MD5

- MD5 (*Message Digest 5*) est une fonction de hachage cryptographique créée en 1991.
- Elle prend une entrée de taille variable et produit une empreinte (hash) de 128 bits.
- Historiquement utilisée pour la vérification d'intégrité des données et les signatures numériques.
- Aujourd'hui, elle est considérée comme vulnérable aux attaques de collision et a été remplacée par des fonctions plus sécurisées comme SHA-256.

Aperçu du Processus de MD5

Le processus de hachage MD5 comprend les étapes suivantes :

1. Préparation des données (*Padding* et *Appendage de la longueur*).
2. Initialisation des variables.
3. Transformation en blocs de 512 bits.
4. Application de la fonction MD5 (4 étapes de transformation).
5. Construction du hash final.

Étape 1 : Préparation des Données

- **Ajout de Padding** : Les données sont complétées pour atteindre une longueur multiple de 512 bits.
  - Ajoutez un bit `1`, suivi de plusieurs bits `0`.
  - Exemple pour `"abc"` : `01100001 01100010 01100011` devient `...100000`
- **Ajout de la longueur** : La longueur du message original est ajoutée à la fin du bloc.

Étape 2 : Initialisation des Variables

MD5 utilise quatre variables de 32 bits initialisées comme suit :

$$A = 0x67452301$$
$$B = 0xefcdab89$$
$$C = 0x98badcfe$$
$$D = 0x10325476$$

Ces valeurs sont utilisées pour initialiser le processus de transformation.

Étape 3 : Découpage en Blocs de 512 bits

- Le message est divisé en blocs de 512 bits.
- Chaque bloc est ensuite traité indépendamment.
- Exemple pour "abc" après padding :

```
01100001 01100010 01100011 ...  [padding + longueur]
```

Étape 3 : Découpage en Blocs de 512 bit (Suite)

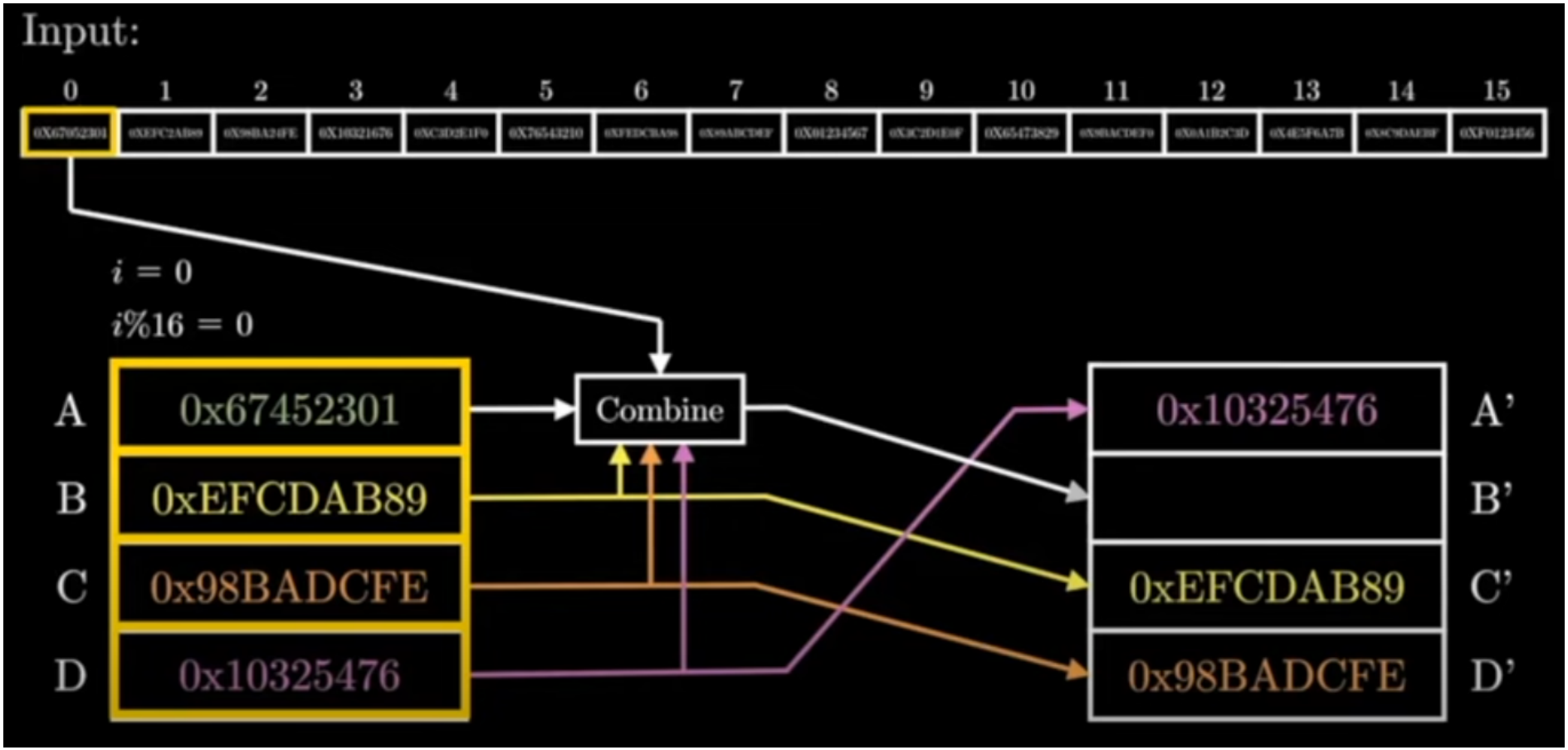


Figure 4: Fonctionnement du MD5

Étape 4 : Fonction de Transformation (1/2)

Chaque bloc est transformé à l'aide de quatre fonctions non linéaires (F, G, H, I) :

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

Étape 4 : Fonction de Transformation (2/2)

Les quatre variables (A, B, C, D) sont mises à jour à chaque tour :

1. Calculez une valeur temporaire basée sur F, G, H ou I.
2. Ajoutez-la à la variable courante.
3. Effectuez une rotation circulaire.

Ce processus est répété 64 fois.

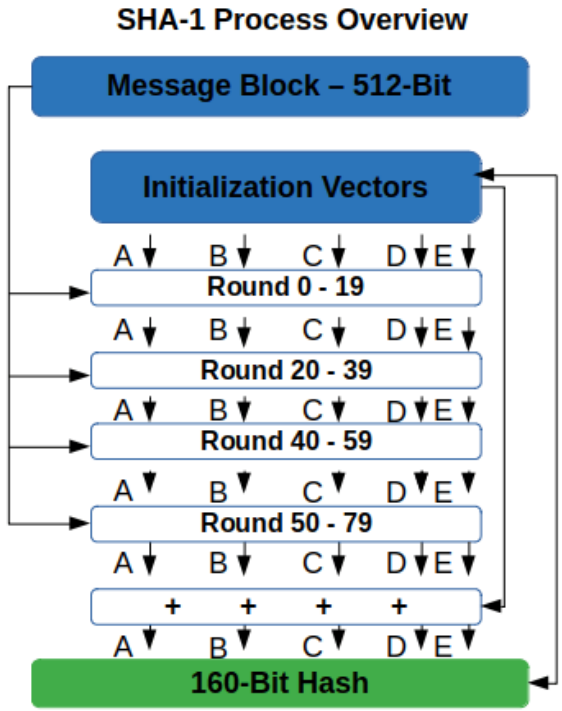
Étape 5 : Construction du Hash Final

- Après avoir traité tous les blocs, les variables A, B, C et D sont concaténées pour former le hash final.
- Exemple pour `"abc"` :

`900150983cd24fb0d6963f7d28e17f72`

SHA-1 : Introduction

- **Nom complet :** Secure Hash Algorithm 1.
- **Développé par :** NSA (National Security Agency) en 1993.
- **Caractéristiques principales :**
  - Algorithme de hachage cryptographique.
  - Génère une empreinte unique de **160 bits**.
- **Statut actuel :** Déprécié depuis 2017 en raison de failles de sécurité.



Processus de SHA-1.

SHA-1 : Fonctionnement

1. **Entrée :** Une donnée de taille variable (texte, fichier, etc.).
2. **Prétraitement :**
   - Divise les données en blocs de **512 bits**.
   - Ajoute des bits de remplissage pour atteindre une longueur multiple de 512.
3. **Processus principal :**
   - Les blocs sont traités individuellement à l'aide de transformations mathématiq
   - Chaque bloc modifie l'état interne pour produire un haché final.
4. **Sortie :** Une empreinte unique de **160 bits**.

SHA-1 : Propriétés et Utilisations

- **Propriétés :**
  - **Déterminisme :** Même entrée = même sortie.
  - **Effet avalanche :** Une petite modification de l'entrée entraîne un changement significatif dans la sortie.
  - **Taille fixe :** Empreinte de 160 bits.
- **Utilisations principales :**
  - Signatures numériques.
  - Certificats SSL/TLS (avant 2017).
  - Vérification d'intégrité des fichiers.

SHA-1 : Faiblesses et Vulnérabilités

- **Collisions :**
  - Il est possible de trouver deux entrées différentes produisant le même haché.
  - Une collision a été démontrée en 2017 par Google et CWI.
- **Impact des failles :**
  - Compromet la fiabilité des signatures numériques.
  - Rend l'algorithme inadapté aux applications critiques.
- **Remplacement recommandé :** SHA-2 et SHA-3.

SHA-1 : Exemple d'Utilisation

Exemple 1 :
Entrée : "Bonjour le monde"
Sortie SHA-1 :
`d3486ae9136e7856bc42212385ea797094475802`

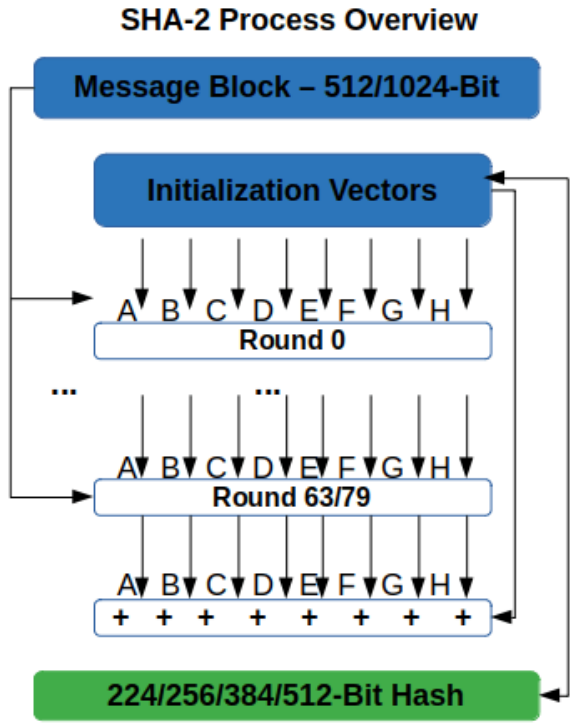
Exemple 2 :
Entrée : "Bonjour le monde." *(avec un point ajouté à la fin)*
Sortie SHA-1 :
`f0438a08e54d7d4ed6a8773bdcd95e33a469ff17`

Observation :

- Une modification minime dans l'entrée, comme l'ajout d'un point, génère un haché totalement différent.
- Cela illustre clairement l'effet d'avalanche, une propriété essentielle des fonctions de hachage.

SHA-2 : Introduction

- **Nom complet :** Secure Hash Algorithm 2.
- **Développé par :** NSA (National Security Agency) en 2001.
- **Caractéristiques principales :**
  - Successeur de SHA-1, avec une sécurité renforcée.
  - Génère une empreinte unique de taille variable : **224, 256, 384, ou 512 bits**.
- **Statut actuel :** Standard recommandé pour les applications de cryptographie modernes.



Vue générale de SHA-2.

SHA-2 : Fonctionnement

1. **Entrée :** Une donnée de taille variable (texte, fichier, etc.).
2. **Prétraitement :**
   - Divise les données en blocs de **512 bits** (SHA-256) ou **1024 bits** (SHA-512).
   - Ajoute des bits de remplissage pour atteindre une longueur multiple de la taille du bloc.
3. **Processus principal :**
   - Les blocs sont traités individuellement par des opérations mathématiques (addition modulaire, XOR, rotations, etc.).
   - Chaque bloc modifie l'état interne (IVs) pour produire un haché final.
4. **Sortie :** Une empreinte unique de taille fixe.

SHA-2 : Propriétés et Utilisations

- **Propriétés :**
  - **Sécurité accrue :** Résistance aux collisions et aux attaques par force brute.
  - **Taille flexible :** Empreintes de 224, 256, 384, ou 512 bits.
  - **Effet avalanche :** Une petite modification de l'entrée entraîne un changement significatif dans la sortie.
- **Utilisations principales :**
  - Cryptographie moderne (certificats SSL/TLS, blockchain, signatures numériques).
  - Vérification d'intégrité des fichiers et bases de données.
  - Sécurisation des mots de passe.

SHA-2 : Variantes

- **SHA-224 :** Empreinte de 224 bits.
- **SHA-256 :** Empreinte de 256 bits, la variante la plus couramment utilisée.
- **SHA-384 :** Empreinte de 384 bits.
- **SHA-512 :** Empreinte de 512 bits, utilisée pour les systèmes nécessitant une sécurité maximale.

SHA-2 : Faiblesses et Limites

- **Consommation de ressources :**
  - SHA-2 est plus exigeant en termes de puissance de calcul que SHA-1.
  - Moins adapté aux systèmes embarqués ou limités en ressources.
- **Menace théorique :**
  - Bien que sûr aujourd'hui, SHA-2 pourrait être vulnérable à des attaques futures, comme les attaques quantiques.
- **Remplacement :** SHA-3 est recommandé pour les nouvelles applications nécessitant une sécurité à long terme.

SHA-2 : Exemple Pratique

Exemple 1 :
Entrée : "Bonjour le monde"
Sortie SHA-256 :

`a2c26b46b68ffc68ff99b453c1d30413413422a5cfc5e46676b5d7ed56385cc9`

Exemple 2 :
Entrée : "bonjour le monde" *(avec une minuscule au lieu de "B")*
Sortie SHA-256 :

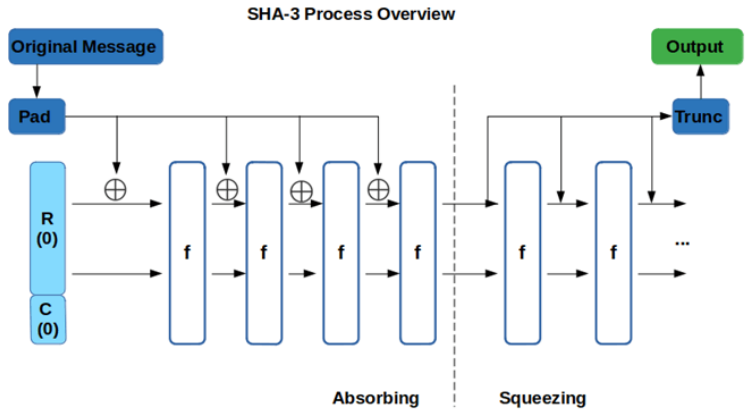
`5f16f4acbf8f6e4c8d6a5e54bff64b894a3d0ff5ea8df7382451bce429c4d2b1`

Observation :

- Une modification minime dans l'entrée, comme changer la lettre majuscule "B" en minuscule "b", génère un haché totalement différent.
- Cela illustre clairement l'effet d'avalanche, une propriété clé des fonctions de hachage cryptographiques.

SHA-3 : Introduction

- **Nom complet :** Secure Hash Algorithm 3.
- **Développé par :** NIST (National Institute of Standards and Technology) en 2015.
- **Caractéristiques principales :**
  - Basé sur l'algorithme Keccak.
  - Fournit une sécurité renforcée contre les attaques modernes.
- **Statut actuel :** Recommandé pour les applications nécessitant une sécurité avancée.



Vue générale de SHA-3.

SHA-3 : Fonctionnement

- **Entrée :** Une donnée de taille variable (texte, fichier, etc.).
- **Principe :**
  - Utilise une construction par **éponge cryptographique**.
  - Comporte deux phases : **absorption** et **extraction**.
- **Sortie :** Une empreinte unique de taille variable (224, 256, 384, ou 512 bits).

SHA-3 : Propriétés et Utilisations

- **Propriétés :**
  - **Sécurité accrue :** Résistance aux attaques modernes comme les attaques par différentiel.
  - **Flexibilité :** Peut produire des hachés de tailles variées.
  - **Différence avec SHA-2 :** Indépendant des constructions de compression utilisées par SHA-2.
- **Utilisations principales :**
  - Applications IoT et systèmes embarqués nécessitant une sécurité accrue.
  - Cryptographie moderne, blockchain et signatures numériques.
  - Validation de données et gestion des mots de passe.

SHA-3 : Exemple d'Utilisation

Exemple 1 :
Entrée : "Bonjour le monde"
Sortie SHA-3 (256 bits) :

`79f57f45a73ed5615b7a4f7257f4b3a6ac4bcd7d9cf5e6b7263f787f`

Exemple 2 :
Entrée : "Bonjour le Monde." *(modification de la casse et ajout d'un point)*
Sortie SHA-3 (256 bits) :

`a60c6517a56d6b0a6f1c9e6f8e503f730034ec7c77eb65f8a74a32b2f2e6f556`

Observation :

- Une modification minime dans l'entrée, comme un changement de casse et l'ajout d'un point, génère un haché totalement différent.
- Cela illustre clairement l'effet d'avalanche.

Whirlpool : Introduction

- **Nom complet :** Whirlpool.
- **Développé par :** Vincent Rijmen et Paulo Barreto en 2000.
- **Caractéristiques principales :**
  - Basé sur un algorithme cryptographique avancé appelé **AES** (Advanced Encryption Standard).
  - Produit une empreinte de **512 bits**.
  - Conçu pour être sécurisé contre les attaques modernes.
- **Statut actuel :** Utilisé dans des applications nécessitant une sécurité élevée.

Whirlpool : Fonctionnement Technique

- **Étapes principales :**
  1. **Prétraitement :**
     - Divise les données en blocs de **512 bits**.
     - Ajoute des bits de remplissage pour atteindre une longueur multiple de 512.
  2. **Transformation par blocs :**
     - Chaque bloc est traité via un réseau de substitution-permutation (SPN).
     - Utilise une matrice de transformation linéaire et des substitutions non linéaires.
  3. **Sortie finale :** Une empreinte unique de **512 bits**.
- **Propriétés de sécurité :**
  - Résistant aux collisions et aux attaques de force brute.
  - Forte diffusion grâce à ses transformations non linéaires.

Whirlpool : Exemple d'Utilisation

Exemple 1 :
Entrée : "Bonjour le monde"
Sortie Whirlpool :
`19fa61d75522a4668c6425aab7df61e49c7ac088f29a9e27b58f3a81...`

Exemple 2 :
Entrée : "Bonjour Le Monde" *(changement de casse)*
Sortie Whirlpool :
`4a0c8d667a8cdd1d8c6d7f29b58f7a818e2497a88f29a9e27b58c642...`

Observation :

- Une modification minime dans l'entrée génère une empreinte totalement différente.
- Illustre l'effet d'avalanche.

Whirlpool : Propriétés et Limites

- **Propriétés :**
  - Taille fixe de l'empreinte : **512 bits**.
  - Résistance élevée contre les attaques cryptographiques.
  - Algorithme robuste basé sur AES.
- **Limites :**
  - Consommation de ressources : Plus exigeant que les algorithmes comme SHA-1.
  - Adoption limitée : Moins répandu que SHA-2 ou SHA-3.
- **Applications :** Sécurisation des bases de données, signatures numériques, etc.

Étude Comparative

Fonction	**Taille**	**Collisions**	**Utilisation**	**Statut**
MD5	128 bits	Faible	Vérification	Déprécié
SHA-1	160 bits	Faible	SSL	Déprécié
SHA-2	224-512 bits	Forte	Cryptographie	Sécurisé
SHA-3	224-512 bits	Très forte	Sécurité avancée	Sécurisé
RIPEMD	128-160 bits	Moyenne	Alternative	Rare
Whirlpool	512 bits	Très forte	Sécurité élevée	Sécurisé

Conclusion

- Les fonctions de hachage jouent un rôle fondamental en cryptographie, assurant la sécurité des systèmes numériques.
- Elles garantissent trois propriétés clés :
  - **Intégrité :** Protection contre les modifications non autorisées des données.
  - **Authenticité :** Vérification de l'identité des expéditeurs et destinataires.
  - **Confidentialité :** Sécurisation des informations sensibles.
- Les avancées comme SHA-2 et SHA-3 offrent des solutions robustes face aux menaces modernes.
- Le choix de la fonction de hachage doit être adapté au contexte d'utilisation, en favorisant les algorithmes éprouvés et sécurisés.
- En perspective, le développement de nouvelles technologies (e.g., cryptograp post-quantique) renforcera davantage la sécurité des systèmes.

Références I

CommentOuvrir.com, *Comprendre les bases des fonctions de hachage*.
Disponible sur: https://commentouvrir.com/info/comprendre-les-bases-des-fonctions-de-hachage/

SealPath Blog, *Types de chiffrement : Guide*.
Disponible sur: https://www.sealpath.com/fr/blog/types-de-chiffrement-guide/

IDEMIA, *La cryptographie expliquée*.
Disponible sur: https://www.idemia.com/fr/cryptographie

V. Rijmen et P. Barreto, *The WHIRLPOOL Hashing Function*.
Disponible sur: https://www2.seas.gwu.edu/~poorvi/Classes/CS381_2007/Whirlpool.pdf

National Institute of Standards and Technology (NIST), *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*.
Disponible sur: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf

Questions ?

Merci ! Des questions ?