Activité découverte : Cryptographie - SHA-256

---

1. Introduction au hachage cryptographique

Activité découverte : Comprendre une fonction de hachage simple

L'objectif ici est que les élèves manipulent directement une fonction de hachage très simple en Python et en tirent eux-mêmes des conclusions sur ses limites.

👉 **`Voir NBJ pour les programmes Python`**

Questions pour guider l'exploration

1. Observez les résultats obtenus. Que remarquez-vous ?
2. Que se passe-t-il si vous changez l'ordre des lettres dans un mot ?
3. Pouvez-vous trouver des cas où deux mots différents donnent le même résultat ? *(notion de collision)*
4. Pourquoi ce type de fonction n'est-il pas adapté à la cryptographie ?
5. Comment pourrait-on améliorer cette fonction pour éviter les collisions ?

Objectifs cachés

- Faire émerger la notion de collision.
- Montrer qu'une fonction de hachage doit être sensible à la moindre modification.
- Faire réfléchir sur l'importance de la longueur fixe du résultat.

Réponses et explications

1. Les nombres obtenus sont toujours compris entre 0 et 255, peu importe la longueur du texte.
2. La valeur reste la même si les lettres sont juste permutées (ex: "AB" et "BA" ont le même hash), ce qui est un problème en cryptographie.
3. Oui, dès que la somme ASCII modulo 256 est identique, une collision apparaît. Ex : "AB" et "BA" ont le même hash.
4. Les collisions sont fréquentes. La fonction est trop simple et on peut retrouver l'entrée à partir de la sortie.
5. Ajouter des transformations non linéaires, allonger la taille du hash généré, utiliser des opérations plus complexes comme XOR ou rotations.

---

2. SHA-256 : Fonctionnement et découverte

Activité pratique : Manipuler SHA-256 en Python

L'élève va découvrir comment fonctionne SHA-256 en expérimentant avec des valeurs simples.

👉 **`Voir NBJ pour les programmes Python`**

Questions pour guider l'exploration

1. Observez les hash obtenus. Que remarquez-vous sur leur longueur ?
2. Testez des entrées légèrement modifiées. Que constatez-vous ? *(ex : "Bonjour" et "bonjour")*
3. Pourquoi parle-t-on d'effet avalanche en cryptographie ?
4. SHA-256 est-il réversible ? Pourquoi ?
5. Quelles sont les propriétés essentielles d'une bonne fonction de hachage cryptographique ?

Objectifs cachés

- Comprendre que SHA-256 produit toujours une sortie de 256 bits.
- Observer l'effet avalanche : un changement minime dans l'entrée change totalement la sortie.
- Introduire le concept d'irréversibilité et d'unicité des empreintes.
- Montrer que SHA-256 garantit l'intégrité des données.

Réponses et explications

1. Les hash obtenus ont toujours la même longueur (64 caractères hexadécimaux, soit 256 bits).
2. Un changement minime (ex : majuscule/minuscule) entraîne un hash totalement différent (effet avalanche).
3. L'effet avalanche signifie qu'un petit changement d'entrée entraîne une modification complète du hash.
4. SHA-256 est irréversible car il ne permet pas de retrouver l'entrée d'origine à partir du hash.
5. Une bonne fonction de hachage doit être rapide, déterministe, résistante aux collisions et irréversible.

---

3. L'énigme "Route 666" et la combinatoire

Présentation du problème

Un chanteur de rock a publié sur Twitter le hash SHA-256 de son numéro de téléphone, en défiant les internautes de le retrouver.

Indices donnés :

- Le numéro commence par **001** (code national USA).
- Le code de zone est **555**.
- Le préfixe est **666**.
- Il reste à deviner les **4 derniers chiffres**.
- SHA-256 du numéro :
  9dd8646d336e4dc5b08b5f15e3fe6980e645f1f96e79862b54460e4d21287819

Les élèves doivent trouver une **stratégie** pour résoudre ce problème et comprendre la **difficulté d'une attaque par force brute**.

Questions pour guider l'exploration

1. Combien de combinaisons possibles faut-il tester ?
2. Quelle stratégie permettrait d'optimiser la recherche ?
3. Combien de temps faudrait-il à un ordinateur pour tester toutes les possibilités ?
4. Comment cette attaque illustre-t-elle l'importance du choix des mots de passe ?

Objectifs cachés

- Faire travailler la notion de combinatoire et de calcul de complexité.
- Comprendre comment une attaque par force brute fonctionne en pratique.
- Sensibiliser aux bonnes pratiques de sécurité (choix des mots de passe robustes).

Réponses et explications

1. Il y a 104=1000010^4 = 10 000104=10000 combinaisons possibles (de 0000 à 9999).
2. Si on connaît des indices sur les chiffres les plus probables (ex : préférences du chanteur, chiffres significatifs), on peut tester ces combinaisons en premier. De plus, une approche distribuée permettrait de diviser les tests entre plusieurs ordinateurs.
3. Un ordinateur classique peut tester environ 1,5 million de hachages par seconde. Puisqu'il n'y a que 10 000 combinaisons possibles, la recherche peut se faire en quelques millisecondes.
4. Un mot de passe court (ex : 4 chiffres) peut être facilement brisé par force brute. Plus la longueur du mot de passe augmente, plus il devient difficile à casser (ex : un mot de passe de 8 chiffres nécessiterait 108=10000000010^8 = 100 000 000108=100000000 tests). L'utilisation d'un salage cryptographique (ajout d'une valeur aléatoire au hash) empêche ce type d'attaque.

---

4. Expérience pratique : Briser le code

Activité : Coder une attaque par force brute et analyser le temps de calcul

L'objectif est que les élèves écrivent un programme en Python pour tester toutes les combinaisons possibles et mesurer le temps d'exécution, puis réalisent plusieurs essais pour analyser comment ce temps évolue.

👉 `Voir NBJ pour les programmes Python`

Questions pour guider l'exploration

1. **Observez les temps de calcul obtenus pour 4, 5 et 6 chiffres. Que remarquez-vous ?**
2. **Si on ajoute un chiffre supplémentaire, le temps de calcul double-t-il, triple-t-il ?**
3. **Pouvez-vous tracer un tableau avec les temps obtenus pour mieux visualiser la croissance ?**
4. **Pourquoi est-il difficile d'utiliser une attaque par force brute sur un mot de passe de 8 caractères ?**

Objectifs cachés

- Faire manipuler les élèves pour qu'ils **découvrent par eux-mêmes** que le temps de calcul **n'augmente pas de manière linéaire**.
- Montrer qu'une augmentation **modérée** de la longueur du code testé entraîne une **forte augmentation** du temps de calcul.
- Faire comprendre **intuitivement** pourquoi un **mot de passe long** est beaucoup plus sécurisé qu'un **mot de passe court**.

Réponses et explications

1. Le temps de calcul augmente très rapidement à chaque ajout d'un chiffre.
2. Non, il augmente **bien plus rapidement** que cela.
3. Exemples de résultats attendus (variables selon la machine utilisée) :

Nombre de chiffres	Temps (secondes)
4	0.002
5	0.025
6	0.260

En observant ces résultats, on constate que le temps **croît très rapidement**.

4. Chaque caractère supplémentaire **multiplie considérablement** le nombre de possibilités. Un mot de passe de **8 chiffres** nécessiterait **10^8 = 100 000 000** tests, ce qui prendrait un **temps très long** sur un ordinateur classique. C'est pourquoi il est **essentiel** d'avoir des mots de passe longs et d'utiliser des techniques de **salage** et **hachage sécurisé** pour protéger les données.

---

5. Histoire et principes fondamentaux de la cryptographie

Activité 1 : Construction d'une frise chronologique

Objectif
Permettre aux élèves de visualiser l'évolution de la cryptographie à travers les âges en identifiant des événements clés.

Consignes

1. Utiliser l'outil en ligne Frise Chronos pour créer une frise chronologique interactive.
2. Rechercher et sélectionner au moins 15 événements marquants liés aux méthodes de cryptage et à leurs applications concrètes.
3. Inclure dans la frise les événements suivants, en les complétant avec d'autres découvertes ou applications significatives :

Date	Système cryptographique	Explication
-500 av. J.-C.	**Scytale spartiate**	Premier système connu de chiffrement par transposition.
-150 av. J.-C.	**Carré de Polybe**	Chiffrement basé sur une grille de substitution.
50 av. J.-C.	**Chiffre de César**	Décalage des lettres selon une clé secrète.
IXe siècle	**Analyse fréquentielle (Al-Kindi)**	Première méthode de cassage des chiffres de substitution.
1467	**Disque chiffrant (Alberti)**	Introduction du chiffrement polyalphabétique.
1553	**Chiffre de Bellaso**	Première apparition d'un chiffrement polyalphabétique avec clé répétée.
1586	**Chiffre de Vigenère**	Perfectionnement du chiffrement polyalphabétique.

1863	**Attaque de Kasiski**	Première méthode systématique pour casser Vigenère.
1917	**Masque jetable (Vernam)**	Seul chiffrement prouvé comme incassable.
1940	**Décryptage d'Enigma (Turing)**	Avancée majeure en cryptanalyse durant la Seconde Guerre mondiale.
1976	**Cryptographie asymétrique (Diffie-Hellman)**	Introduction des clés publiques et privées.
1977	**RSA (Rivest, Shamir, Adleman)**	Premier système de chiffrement à clé publique largement adopté.
1991	**PGP (Zimmermann)**	Logiciel de chiffrement de communication utilisé dans les emails.
2001	**AES (Advanced Encryption Standard)**	Nouveau standard de chiffrement remplaçant DES.
2015	**SHA-3**	Nouvelle fonction de hachage standardisée par le NIST.

Ressources

- Site pour créer la frise : Frise Chronos
- Références historiques : Histoire de la cryptologie - Wikipédia

Activité 2 : Exposé sur une méthode de chiffrement historique

Objectif

Approfondir la compréhension des principes mathématiques derrière les méthodes de chiffrement historiques en réalisant un exposé détaillé.

Consignes

1. Choisir une des deux méthodes de chiffrement suivantes :
    - Chiffre de César
    - Chiffre de Vigenère
2. Réaliser un exposé présentant :

    - Le contexte historique de la méthode choisie.
    - Le principe mathématique du chiffrement et du déchiffrement.
    - Un exemple chiffré illustrant le processus.
    - Les forces et faiblesses de la méthode en termes de sécurité.

Présentation

- L'exposé peut être présenté sous forme de diaporama, de poster ou d'une démonstration en classe avec des exemples chiffrés/déchiffrés.
- Encourager les élèves à illustrer leur explication avec des chiffres et des lettres pour bien comprendre l'algorithme sous-jacent.

Objectifs pédagogiques de cette partie

- Comprendre l'évolution de la cryptographie et son impact sur l'histoire.
- Mettre en perspective les méthodes de chiffrement avec les outils modernes de cryptographie.
- Savoir expliquer un principe mathématique appliqué à la sécurité informatique.
- Développer une démarche de recherche et de communication sur un sujet scientifique.

---

6. Prolongements et discussion finale

Cette dernière partie vise à ouvrir la réflexion sur la cryptographie moderne et à sensibiliser les élèves aux enjeux de la cybersécurité.

6.1 SHA-3 et les limites du SHA-256

SHA-256 : Un standard de sécurité, mais jusqu'à quand ?

SHA-256 est actuellement considéré comme sûr, mais comme tous les algorithmes cryptographiques, il pourrait devenir vulnérable à l'avenir, notamment avec l'essor des ordinateurs quantiques.

SHA-3 : Une nouvelle génération de hachage

En 2015, le NIST a sélectionné SHA-3 comme une alternative aux fonctions de hachage de la famille SHA-2. SHA-3 repose sur une architecture totalement différente (Keccak) et est conçu pour résister à des attaques plus avancées.

Questions pour guider l'exploration

1. Pourquoi le NIST a-t-il jugé nécessaire de développer SHA-3 alors que SHA-256 n'est pas encore cassé ?
2. En quoi le fonctionnement de SHA-3 diffère-t-il fondamentalement de SHA-256 ?
3. Que pourrait-il se passer si un jour on trouvait une collision dans SHA-256 ?

Réponses et explications

1. Le NIST a anticipé l'émergence de nouvelles menaces et a voulu proposer un standard de remplacement avant qu'une faille soit découverte.
2. SHA-3 utilise une structure éponge différente de la structure classique de SHA-256 (basée sur les transformations Merkle-Damgård).
3. Une collision dans SHA-256 signifierait qu'il ne garantit plus l'unicité des empreintes numériques, compromettant ainsi son usage dans la signature électronique et la protection des données.

6.2 Importance des bonnes pratiques en cybersécurité

Les élèves découvrent ici comment protéger leurs propres données grâce à des pratiques simples mais essentielles.

Consignes : Discussion et réflexion en classe

1. Pourquoi faut-il utiliser des mots de passe longs et uniques ?

- Plus un mot de passe est long, plus il est difficile à deviner par force brute.
- Un mot de passe unique évite la compromission de plusieurs comptes en cas de fuite de données.

2. Qu'est-ce que le "salage" des mots de passe ?

- Il s'agit d'ajouter une valeur aléatoire unique avant de calculer le hachage d'un mot de passe, afin d'éviter les attaques par dictionnaire et par pré-calcul (rainbow tables).

3. Pourquoi utiliser un gestionnaire de mots de passe ?

- Un gestionnaire stocke des mots de passe longs et complexes de manière sécurisée, évitant ainsi aux utilisateurs de réutiliser le même mot de passe partout.

4. L'authentification multifactorielle : une protection supplémentaire

- En ajoutant un deuxième facteur d'authentification (ex : code SMS, application mobile), on sécurise l'accès aux comptes même si le mot de passe est volé.

Objectifs cachés :

- Sensibiliser les élèves aux menaces numériques et à la nécessité d'adopter des comportements sécurisés.
- Expliquer les bonnes pratiques de gestion des mots de passe et des accès numériques.
- Montrer l'importance de l'anticipation des risques en cybersécurité.

6.3 Conclusion : Cryptographie, entre science et sécurité

La cryptographie est au cœur de notre monde numérique :

- Stockage sécurisé des mots de passe,
- Authentification sur les sites web,
- Chiffrement des communications (WhatsApp, Signal),
- Protection des transactions financières (cartes bancaires, blockchain).

Question finale pour encourager la réflexion :
👉 Selon vous, quels seront les défis de la cryptographie dans 20 ans ?

Prolongements possibles :

- Faire des recherches sur l'ordinateur quantique et la cryptographie post-quantique.
- Étudier des protocoles cryptographiques avancés (courbes elliptiques, zero-knowledge proof).