Cette photo est-elle authentique ?

Clé publique, clé privée et signature numérique face aux deepfakes

Mise en situation. Les IA génératives fabriquent des photos, des vidéos et des voix si réalistes qu'on ne peut plus se fier à ses yeux : ce sont les **deepfakes** (« hypertrucages »). Pour les repérer, on peut entraîner une autre IA à détecter les faux, mais les faussaires améliorent sans cesse leurs outils.
Une autre idée consiste à **prouver le vrai** : l'appareil photo « signe » chaque image au moment de la prise de vue, et n'importe qui peut ensuite vérifier que l'image n'a pas été modifiée. Cette signature repose sur une invention mathématique des années 1970 : la **cryptographie à clé publique**.

1. Une clé publique, une clé privée

Définition

En **cryptographie à clé publique**, chaque personne (ou chaque appareil) possède une **paire de clés** liées mathématiquement :

- une **clé publique**, que l'on donne à tout le monde ;
- une **clé privée**, que l'on garde secrète et que l'on ne transmet jamais.

Cette paire sert à deux choses :

- **chiffrer** : n'importe qui peut chiffrer un message avec la clé publique de Bob, mais seule la clé privée de Bob permet de le déchiffrer (comme un cadenas ouvert que tout le monde peut fermer, mais que seul Bob peut rouvrir) ;
- **signer** : Léa signe un document avec sa clé privée, et n'importe qui peut vérifier la signature avec la clé publique de Léa (comme un sceau que seule Léa peut apposer, mais que tout le monde peut reconnaître).

Exemple

Le cadenas affiché par le navigateur sur un site en `https` utilise ce principe : le site présente sa clé publique, et le navigateur s'en sert pour établir une connexion chiffrée.

▶ À vous de jouer !

1. Pour chaque situation, indiquer quelle clé est utilisée, et à qui elle appartient.
   a) Ali envoie un message secret à Bob.
   b) Bob lit le message secret envoyé par Ali.
   c) L'appareil photo de Léa signe une photo.
   d) Un journal vérifie que la photo vient bien de l'appareil de Léa.
2. Pourquoi la clé privée de l'appareil photo ne doit-elle jamais sortir de l'appareil ?

Coup de pouce

Ce qui doit pouvoir être fait par **tout le monde** utilise la clé publique ; ce qui ne doit être fait que par **une seule personne** utilise sa clé privée.

2. L'empreinte d'une image

Définition

On ne signe pas l'image entière, mais son **empreinte** : un nombre calculé à partir de **tous** les pixels. Si un seul pixel change, l'empreinte change.
Dans cette activité, une image est résumée par 6 pixels (une vraie photo en a des millions). Son empreinte, notée h, est le **reste de la division euclidienne** de la somme des pixels par 33.

Exemple

Si la somme des pixels vaut 100 : $100 = 33 \times 3 + 1$, donc l'empreinte vaut $h = 1$.

120	45	200
30	90	17

Photo A

120	45	210
30	90	17

Photo B

La photo B est la photo A retouchée : un pixel a été modifié.

▶ À vous de jouer !

1. Calculer la somme des pixels de la photo A, puis son empreinte h.
2. Calculer l'empreinte h de la photo B. Que constate-t-on ?

Coup de pouce

Pour trouver le reste de la division de 502 par 33, on cherche le plus grand multiple de 33 inférieur ou égal à 502 : $33 \times 15 = 495$.

3. Signer une image

Définition

L'appareil photo de Léa possède :

- la **clé publique** : le nombre $n = 33$ et le nombre $e = 3$;
- la **clé privée** : le nombre $d = 7$, caché dans l'appareil.

Pour signer une image d'empreinte h, l'appareil calcule le reste de la division de h^7 par 33 : c'est la **signature** s.
Pour ne pas manipuler de trop grands nombres, on multiplie par h **une étape à la fois**, et on ne garde que le reste de la division par 33 après chaque multiplication.

Exemple

Signature d'une image d'empreinte $h = 2$: on part de 1 et on multiplie 7 fois par 2.

Étape	1	2	3	4	5	6	7
Calcul	$1 \times 2 = 2$	$2 \times 2 = 4$	$4 \times 2 = 8$	$8 \times 2 = 16$	$16 \times 2 = 32$	$32 \times 2 = 64$	$31 \times 2 = 62$
Reste	2	4	8	16	32	31	29

Par exemple, à l'étape 6 : $64 = 33 \times 1 + 31$, on garde 31. La signature vaut $s = 29$.

▶ À vous de jouer !

1. L'appareil de Léa signe la photo A, d'empreinte $h = 7$. Recopier et compléter le tableau des 7 étapes, puis donner la signature.
2. Pourquoi un faussaire qui connaît seulement la clé publique ($n = 33$ et $e = 3$) ne peut-il pas faire ce calcul ?

Coup de pouce

Étape 2 : $7 \times 7 = 49 = 33 \times 1 + 16$, on garde 16. Étape 3 : $16 \times 7 = 112\ldots$

4. Vérifier une signature

Définition

Pour vérifier qu'une image d'empreinte h porte bien la signature s de l'appareil de Léa, on utilise sa **clé publique** : on calcule le reste de la division de s^3 par 33 (en trois étapes).

- Si on retrouve l'empreinte h de l'image, la signature est **valide** : l'image vient de l'appareil de Léa et n'a pas été modifiée.
- Sinon, la signature est **refusée**.

Exemple

Image d'empreinte 2 et de signature 29 : $1 \times 29 = 29$, puis $29 \times 29 = 841 = 33 \times 25 + 16$, puis $16 \times 29 = 464 = 33 \times 14 + 2$. On retrouve 2 : la signature est valide.

100	60	180
40	70	25

Photo C

▶ À vous de jouer !

Un journal reçoit trois photos accompagnées d'une signature :

- la photo A, avec la signature 28 ;
- la photo B (la photo A retouchée), avec la même signature 28 ;
- la photo C, un deepfake fabriqué par Ève, avec une signature 13 qu'elle a inventée.

1. Calculer le reste de la division de 28^3 par 33, en trois étapes.
2. Vérifier la signature de la photo A, puis celle de la photo B. Que conclure ?
3. Calculer l'empreinte h de la photo C, puis vérifier sa signature. Que conclure ?

Coup de pouce

$28 \times 28 = 784$ et $33 \times 23 = 759$.

5. Programmer la vérification en Python

Exemple

En Python, `a % b` donne le reste de la division euclidienne de `a` par `b`. Par exemple, `502 % 33` vaut 7.

```
def empreinte(pixels):
    total = 0
    for p in pixels:
        total = total + p
    return total % 33

def puissance_mod(a, k):
    r = 1
    for i in range(k):
        r = r * a % 33
    return r

def verifier(pixels, s):
    return puissance_mod(s, 3) == empreinte(pixels)
```

▶ À vous de jouer !

1. Que renvoie `empreinte([120, 45, 200, 30, 90, 17])` ?
2. Donner les valeurs successives de `r` lors de l'appel `puissance_mod(7, 7)`. Que renvoie cet appel ?
3. Que renvoient `verifier([120, 45, 200, 30, 90, 17], 28)` et `verifier([120, 45, 210, 30, 90, 17], 28)` ?
4. Écrire une fonction `signer(pixels)` qui renvoie la signature d'une image, avec la clé privée $d = 7$.

Coup de pouce

`r * a % 33` calcule d'abord `r * a`, puis le reste de la division par 33 : c'est exactement une étape du tableau de la partie 3.

6. La signature face aux deepfakes : forces et limites

Remarque

Ce principe est utilisé en vrai. Le standard **C2PA** (« Content Credentials ») ajoute aux images un historique signé : appareil utilisé, retouches, utilisation d'une IA. Depuis 2025, les smartphones Pixel 10 de Google signent chaque photo prise par l'appareil, et Google Photos signale les retouches faites avec une IA ; en 2026, Apple a présenté une signature des photos dès le capteur. Depuis le 2 août 2026, le règlement européen sur l'IA (*AI Act*) impose aussi aux fournisseurs d'IA générative de marquer les contenus qu'ils produisent, de façon lisible par une machine.
Les vrais systèmes utilisent les mêmes calculs (on parle d'algorithme **RSA**), mais avec des nombres d'environ 600 chiffres au lieu de 33.

▶ À vous de jouer !

Vrai ou faux ? Justifier.

1. Une photo sans signature est forcément un deepfake.
2. Une photo dont la signature est valide montre forcément une scène réelle.
3. Si on fait une capture d'écran d'une photo signée, la capture porte la même signature.
4. Un système de signature peut contenir des failles.

7. Pour aller plus loin

★ Question étoilée

Ève veut faire passer une photo truquée pour une photo de l'appareil de Léa.

1. Ève modifie deux pixels de la photo A : 120 devient 125 et 45 devient 40. Calculer l'empreinte de la photo obtenue. La signature 28 est-elle encore valide ? Qu'en conclure sur notre fonction d'empreinte ?
2. Ève connaît une image d'empreinte 7 et sa signature 28. Elle teste les clés privées possibles $d = 1, 2, 3, \ldots$ en calculant le reste de la division de 7^d par 33, jusqu'à retrouver 28. Combien d'essais lui faut-il ? Pourquoi cette attaque est-elle impossible avec une vraie clé ?

À retenir

- En cryptographie à clé publique, chacun possède une **clé publique** (connue de tous) et une **clé privée** (secrète). On chiffre avec la clé publique du destinataire ; on signe avec sa propre clé privée, et tout le monde vérifie avec la clé publique.
- On signe l'**empreinte** d'un fichier : si un seul pixel change, l'empreinte change et la signature est refusée.
- Face aux deepfakes, la signature permet de **prouver l'origine** d'une image (standard C2PA, obligation de marquage de l'*AI Act*).
- Une signature valide ne prouve pas que la scène est vraie, et l'absence de signature ne prouve pas qu'une image est fausse : il faut toujours croiser les sources.