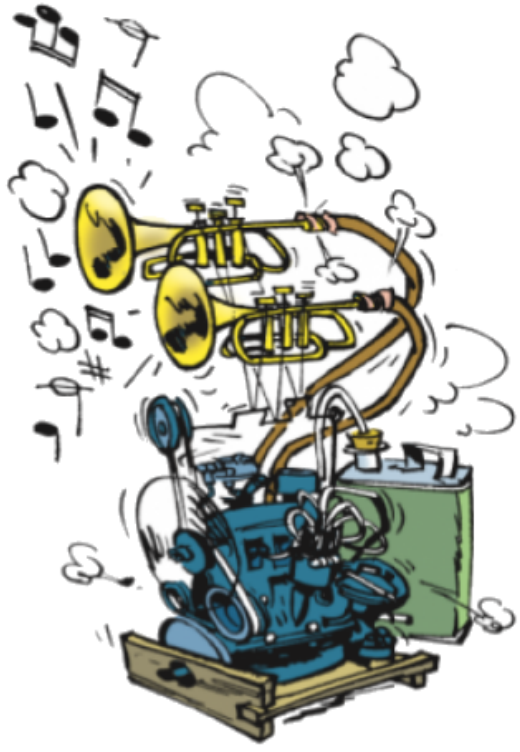
Qu'est-ce qu'une attaque cryptographique ?



Charles Bouillaguet

SORBONNE UNIVERSITÉ

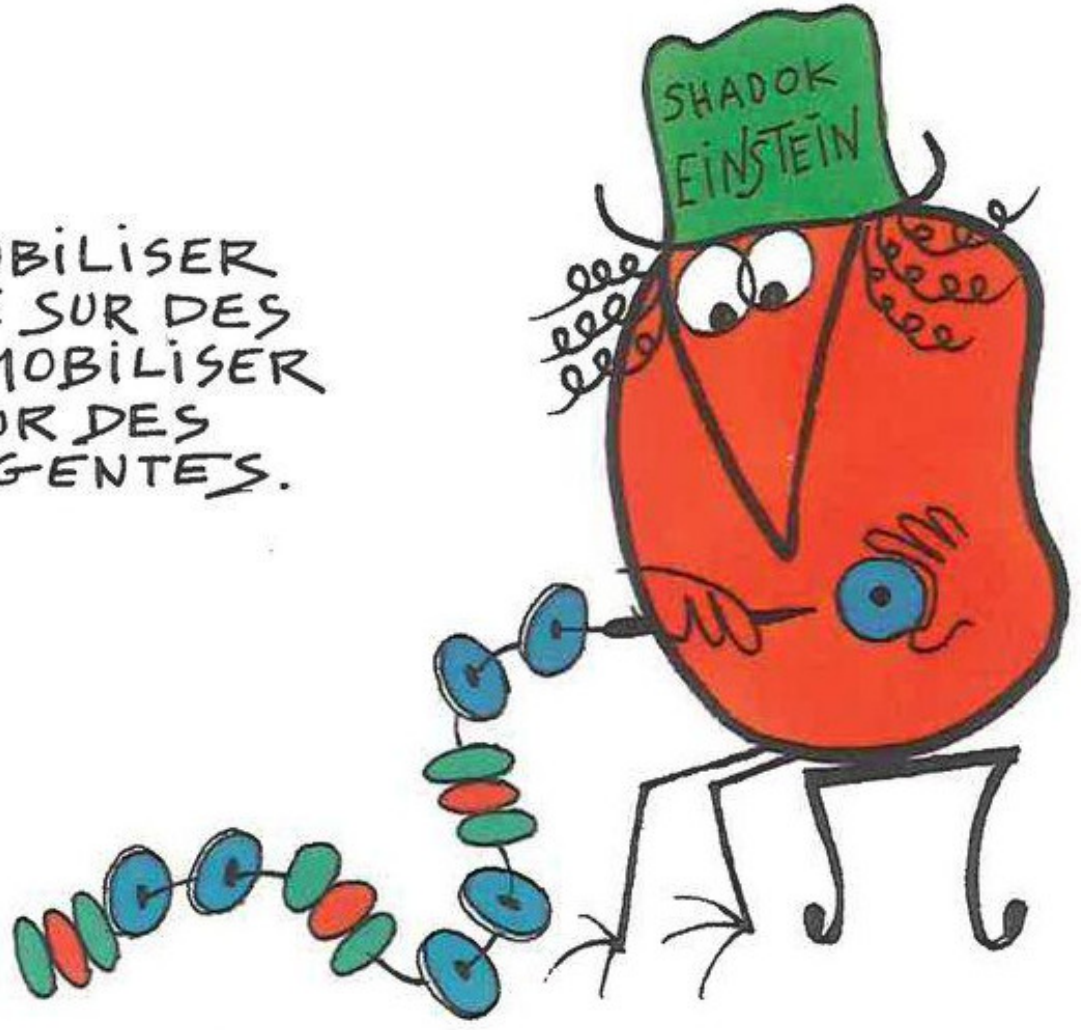
cnrs

LIP6

22 juin 2022

Pourquoi cette question ?!?

IL VAUT MIEUX MOBILISER SON INTELLIGENCE SUR DES CONNERIES QUE MOBILISER SA CONNERIE SUR DES CHOSES INTELLIGENTES.



Utilisateurs normaux

- Utilisent des mécanismes cryptographiques pour **garantir des propriétés de sécurité**
  - Confidentialité
  - Integrité
  - Authenticité

Adversaires

- Veulent ☠ **briser** ☠ la sécurité
  - Lancent des **attaques** contre les mécanismes cryptographiques

Definition

Une **attaque** cryptographique est un **algorithme** dont l'exécution réussie **brise** la sécurité (avec bonne probabilité).

Il y a (presque) **toujours** des attaques ☹

- Essayer toutes les valeurs possibles des secrets
- Exécuter un algorithme de factorisation des grands entiers
- Écrire une instance de SAT et lancer un solveur
- ...

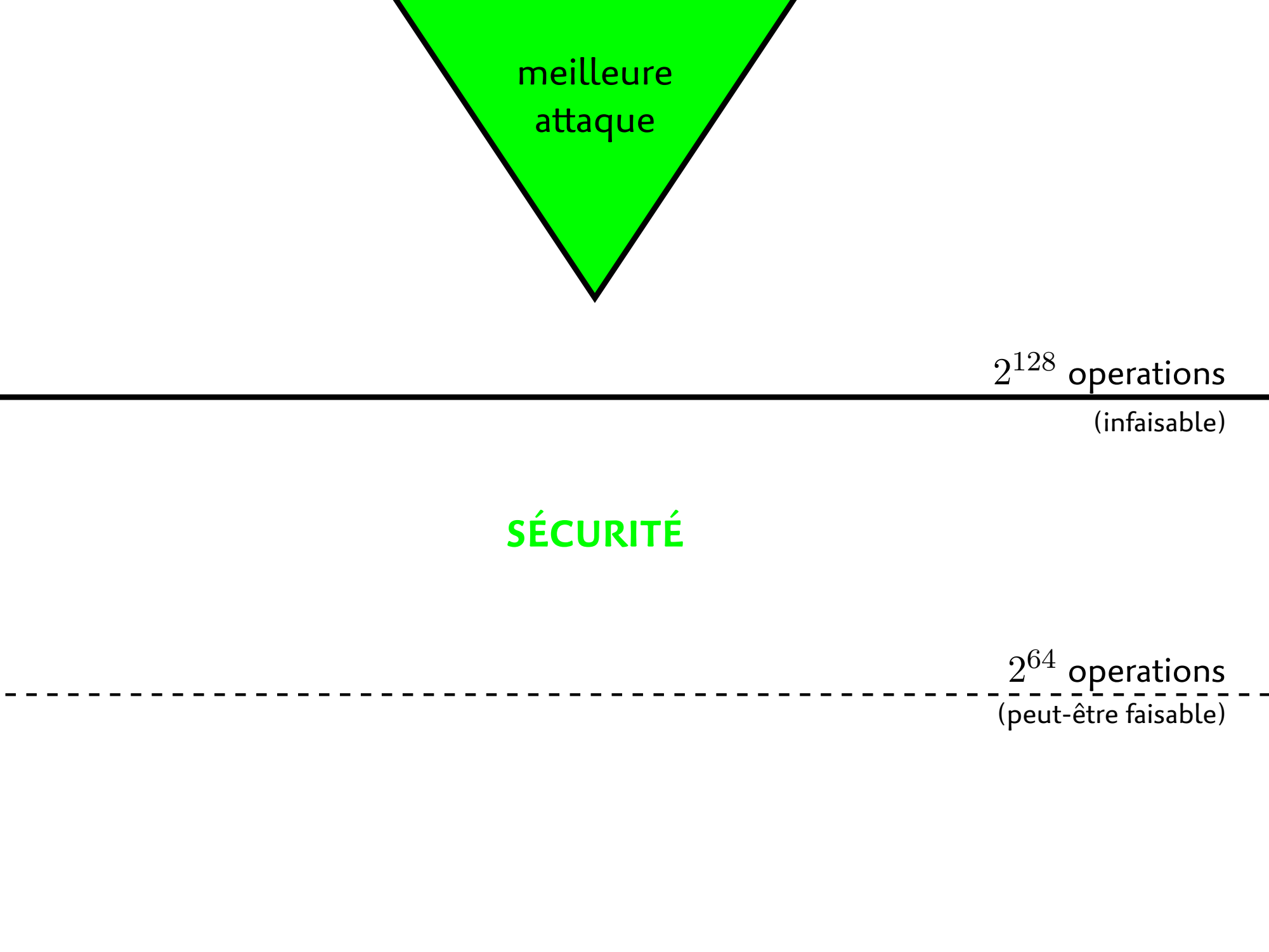
(En principe) elles sont **inoffensives** ☺

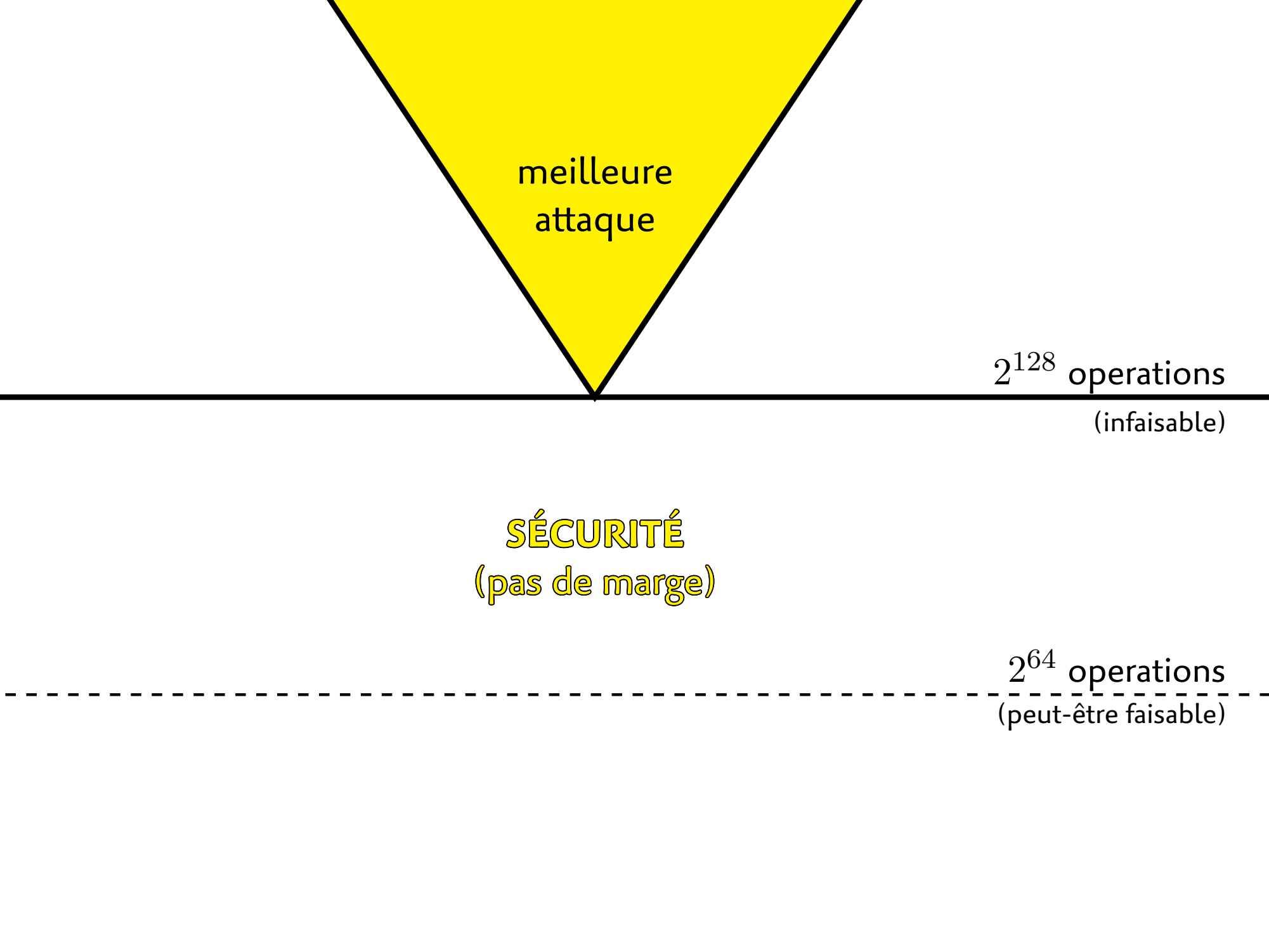
→ La **tailles des clefs** et des autres paramètres des algorithmes cryptographique sont choisies pour que l'exécution des attaques (connues) soit **trop coûteuse** !

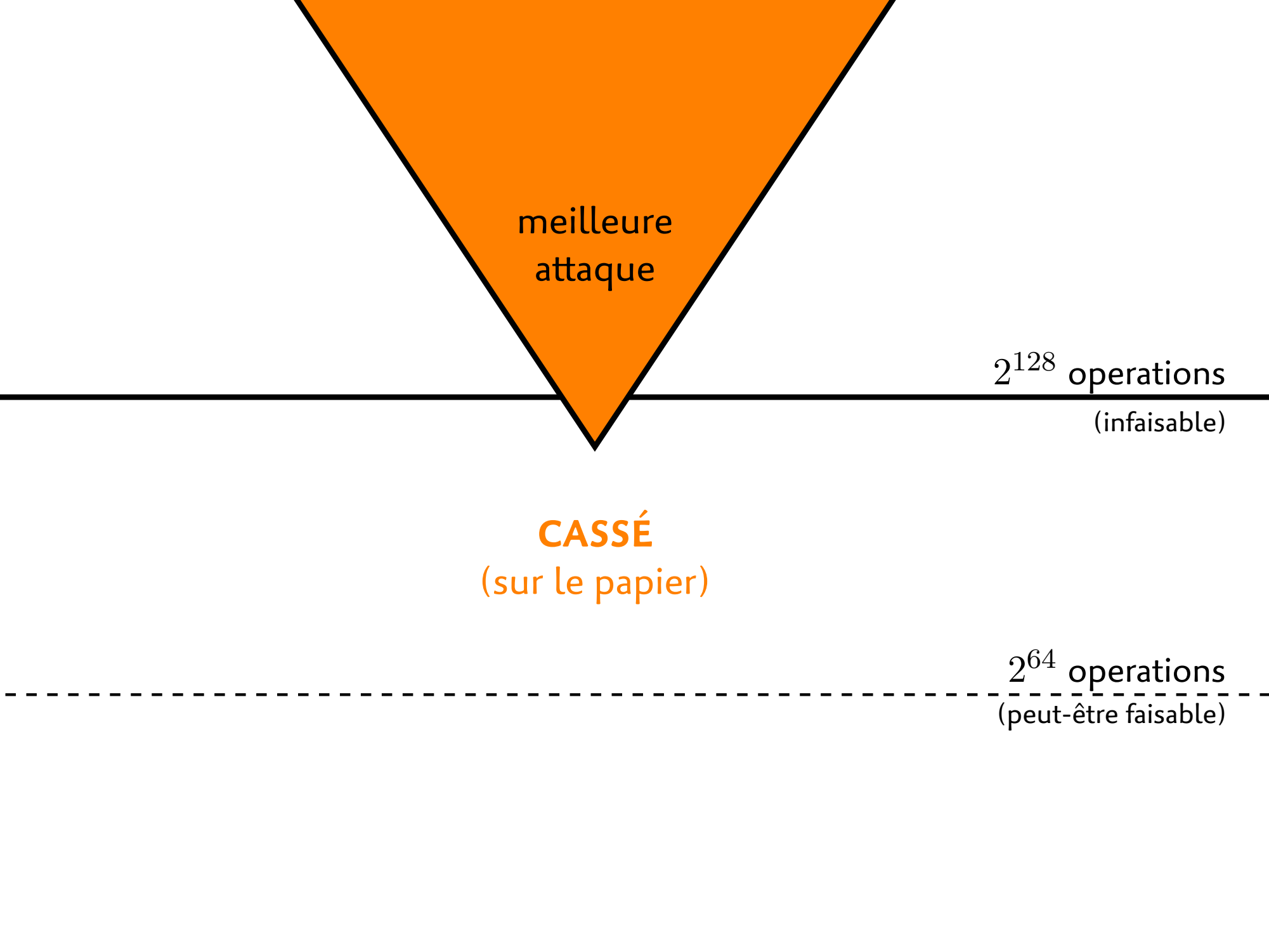
→ trop d'opérations $\Longrightarrow$ infaisable $\Longrightarrow$ securité

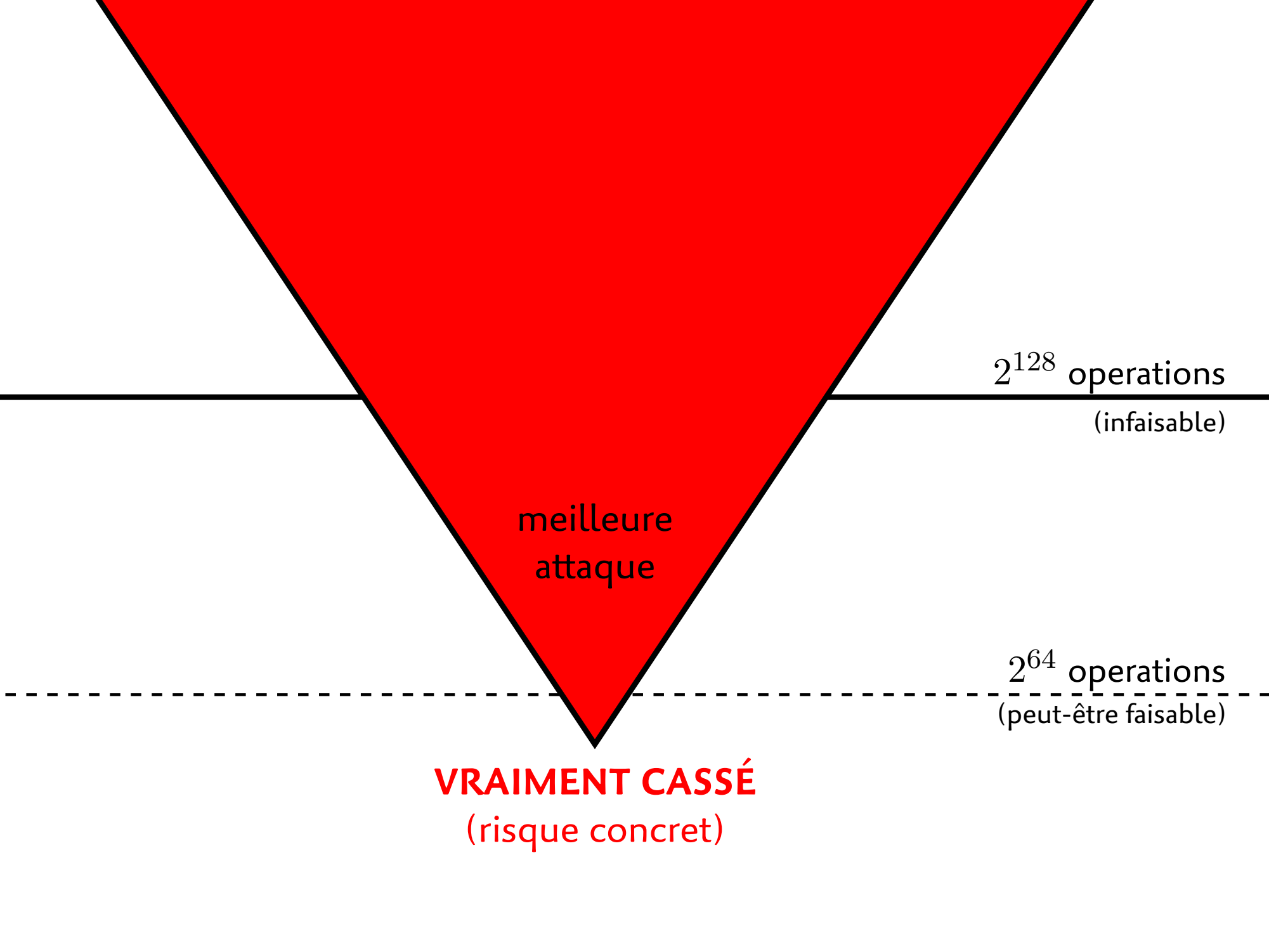
- 2^{128} opérations = trop
- L'AES a donc des clefs de 128 bits

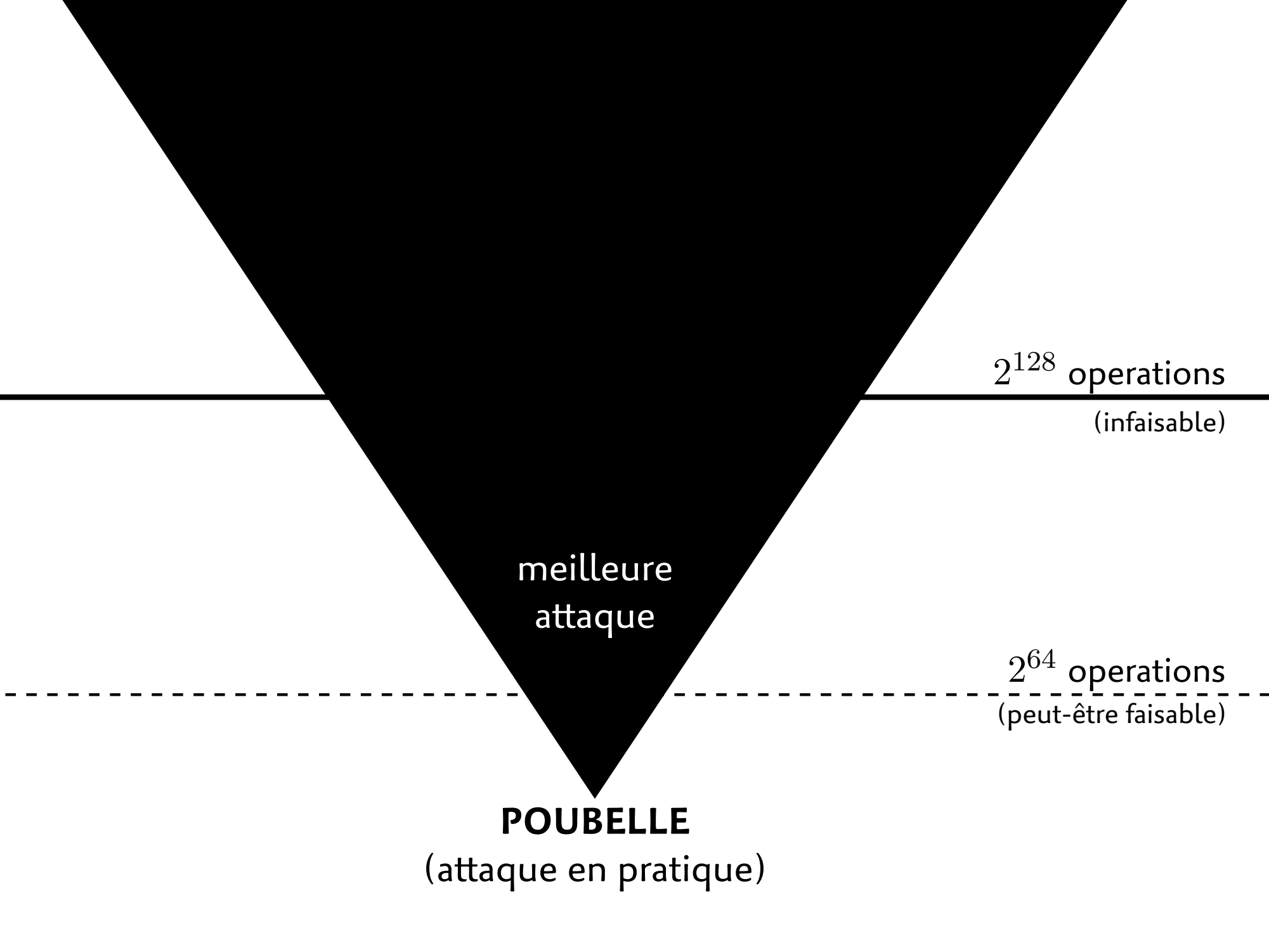
Mais on n'est pas à l'abri de l'invention de nouvelles attaques...


meilleure
attaque
2^{128} operations
(infaisable)
SÉCURITÉ
2^{64} operations
(peut-être faisable)


meilleure
attaque
2^{128} operations
(infaisable)
SÉCURITÉ
(pas de marge)
2^{64} operations
(peut-être faisable)

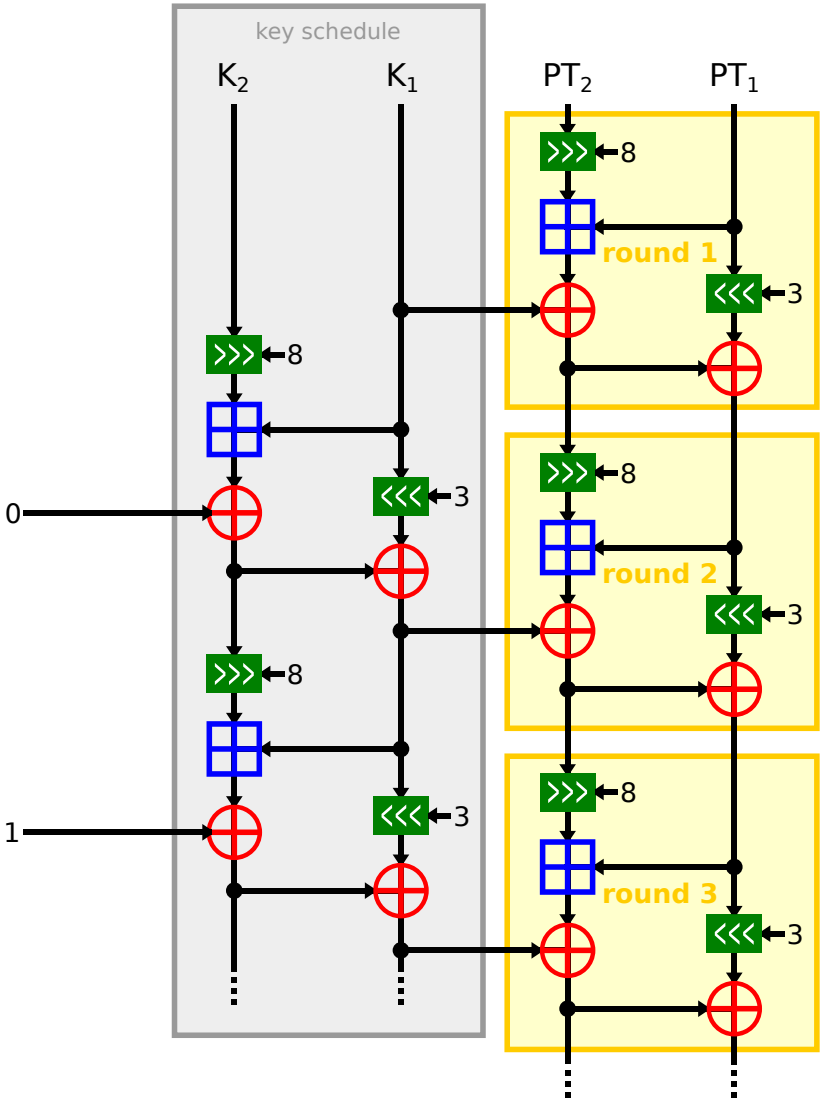

meilleure
attaque
2^{128} operations
(infaisable)
CASSÉ
(sur le papier)
2^{64} operations
(peut-être faisable)


2^{128} operations
(infaisable)
meilleure
attaque
2^{64} operations
(peut-être faisable)
VRAIMENT CASSÉ
(risque concret)


2^{128} operations
(infaisable)
meilleure
attaque
2^{64} operations
(peut-être faisable)
POUBELLE
(attaque en pratique)

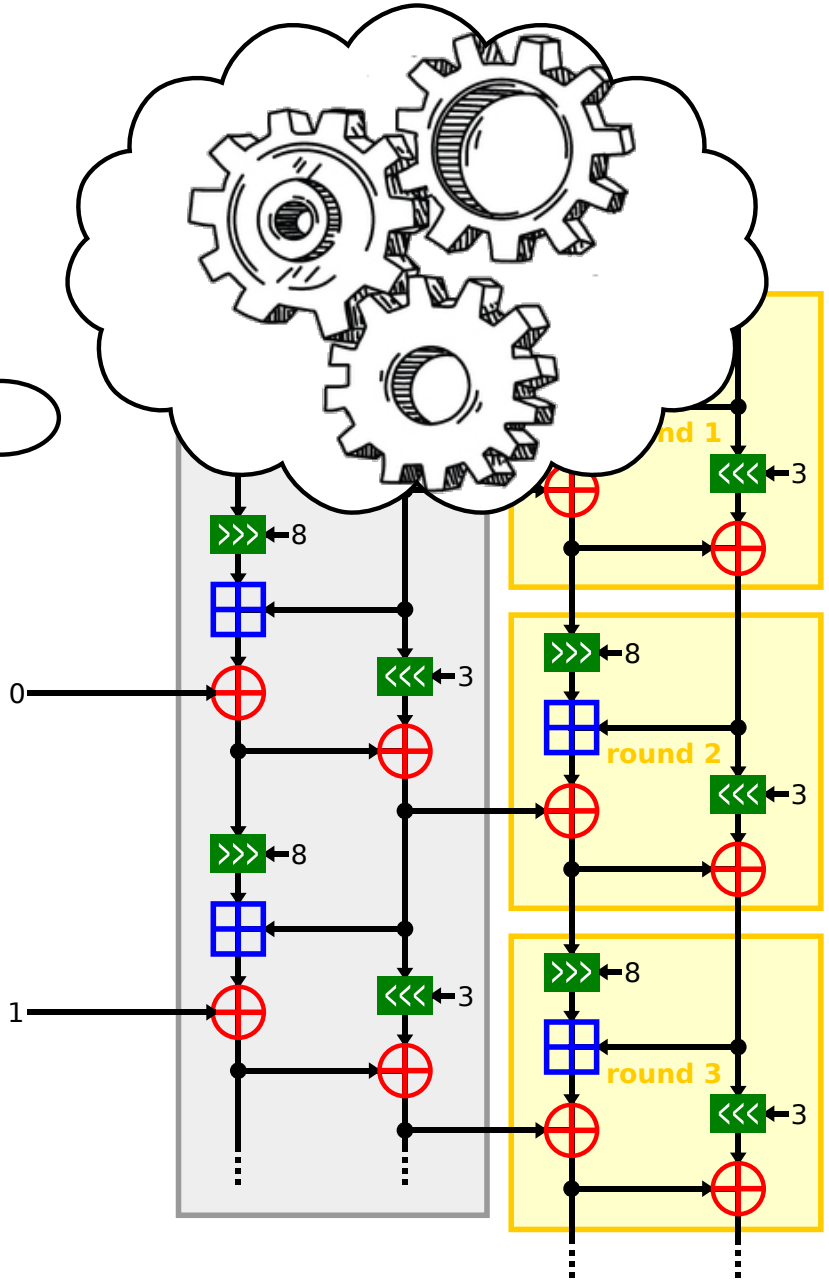
La cryptanalyse

Cryptanalyste



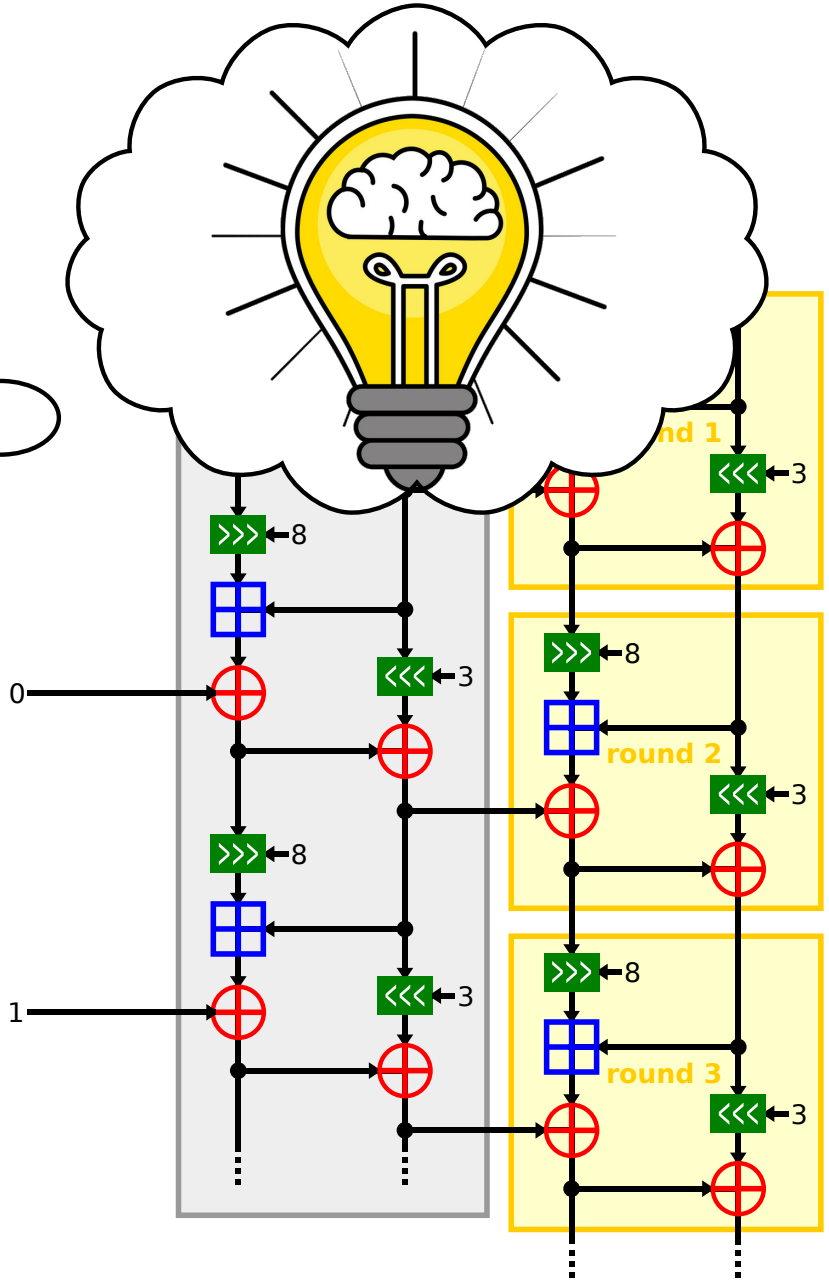
La cryptanalyse

Cryptanalyste



La cryptanalyse

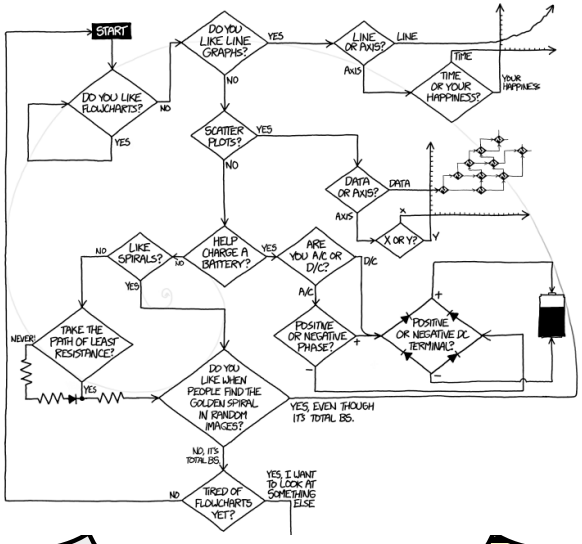
Cryptanalyste

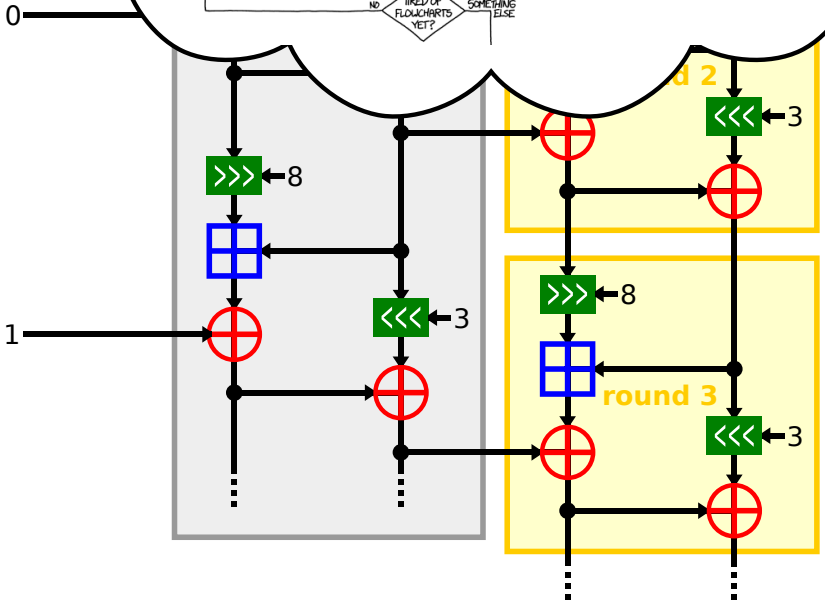


La cryptanalyse

Cryptanalyste

Attaque :





Cas #1 : l'attaque peut être programmée et exécutée

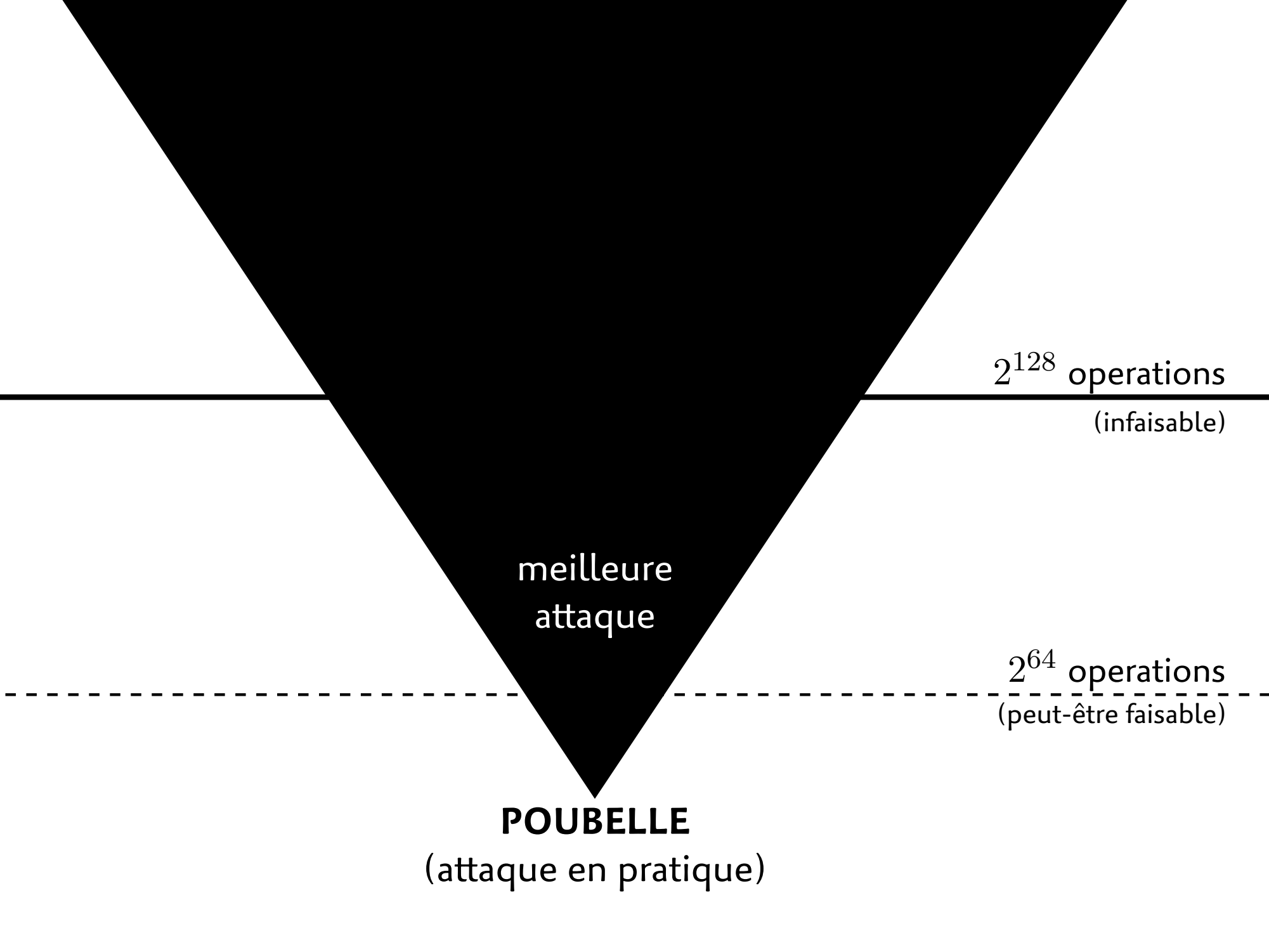
Cryptanalyste

```
int j = 0;
for (int i = j + 1; i < l; i++) {
    vector swap = A[i][j] & (~A[j][j]);
    for (int k = j; k < v; k++) {
        vector xor = swap & (A[i][k] ^ A[j][k]);
        A[i][k] ^= xor;
        A[j][k] ^= xor;
    }
    vector xor = swap & (b[i] ^ b[j]);
    b[i] ^= xor;
    b[j] ^= xor;
}
for (int j = 0; j < v - 1; j++) {
    vector add = A[j+1][j];
    for (int k = j + 1; k < v; k++)
        A[j+1][k] ^= add & A[j][k];
    b[j+1] ^= add & b[j];
    for (int i = j + 2; i < l; i++) {
        vector add = A[i][j];
        A[i][j+1] ^= add & A[j][j+1];
        vector swap = A[i][j+1] & (~A[j+1][j+1]);
        vector xor = swap & (A[i][j+1] ^ A[j+1][j+1]);
        A[i][j+1] ^= xor;
        A[j+1][j+1] ^= xor;
        for (int k = j + 2; k < v; k++) {
            A[i][k] ^= add & A[j][k];
            vector xor = swap & (A[i][k] ^ A[j+1][k]);
            A[i][k] ^= xor;
```

Cas #1 : l'attaque peut être programmée et exécutée

Crypta te

```
int j = 0;
for (int i = j + 1; i < l; i++) {
    vector swap = A[i][j] & (~A[j][j]);
    for (int k = j; k < v; k++) {
        vector xor = swap & (A[i][k] ^ A[j][k]);
        A[i][k] ^= xor;
        A[j][k] ^= xor;
    }
    vector xor = swap & (b[i] ^ b[j]);
    b[i] ^= xor;
    b[j] ^= xor;
}
for (int j = 0; j < v - 1; j++) {
    vector add = A[j+1][j];
    for (int k = j + 1; k < v; k++)
        A[j+1][k] ^= add & A[j][k];
    b[j+1] ^= add & b[j];
    for (int i = j + 2; i < l; i++) {
        vector add = A[i][j];
        A[i][j+1] ^= add & A[j][j+1];
        vector swap = A[i][j+1] & (~A[j+1][j+1]);
        vector xor = swap & (A[i][j+1] ^ A[j+1][j+1]);
        A[i][j+1] ^= xor;
        A[j+1][j+1] ^= xor;
        for (int k = j + 2; k < v; k++) {
            A[i][k] ^= add & A[j][k];
            vector xor = swap & (A[i][k] ^ A[j+1][k]);
            A[i][k] ^= xor;
```

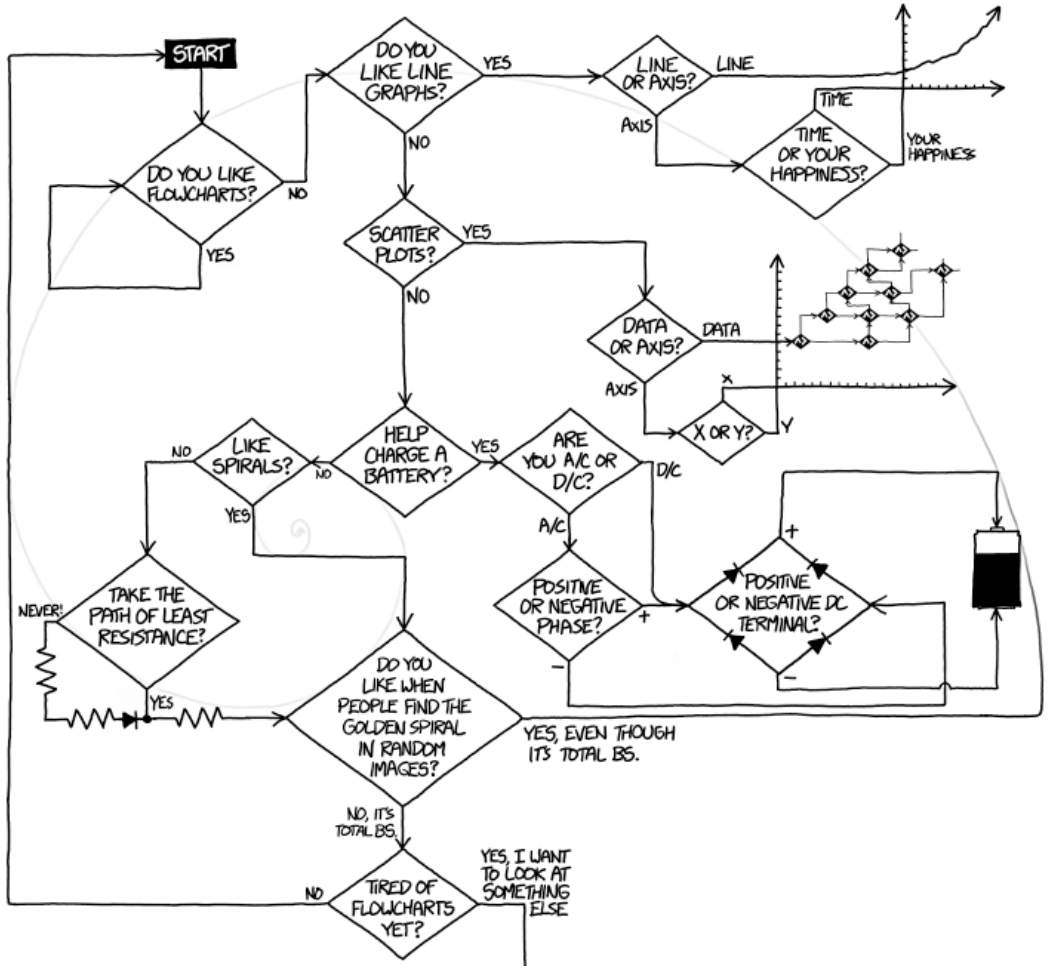

2^{128} operations
(infaisable)
meilleure
attaque
2^{64} operations
(peut-être faisable)
POUBELLE
(attaque en pratique)

Quelques exemples

- 2007 Collision sur `MD5` avec préfixes choisis
  - $\rightsquigarrow$ faux certificats
- 2007 Signatures pirates sur `SFLASH`
  - ... en passe de standardisation par NESSIE
- 2010 Factorisation 768 bits
  - Taille des clefs dans les badges VIGIK...
- 2017 Collision pour `SHA-1`
  - ... toujours utilisé dans `git`
- 2020 Signature pirate sur `LUOV`
  - Candidat 2ème tour compétition post-quantique du NIST
- 2021 Déchiffrement `GPRS` pirate
  - *Stay tuned for the details*
- 2022 Signature pirate sur `Rainbow`
  - Candidat 3ème tour compétition post-quantique du NIST

Cas #2 : l'attaque est trop coûteuse pour le monde réel

Cryptanalyste

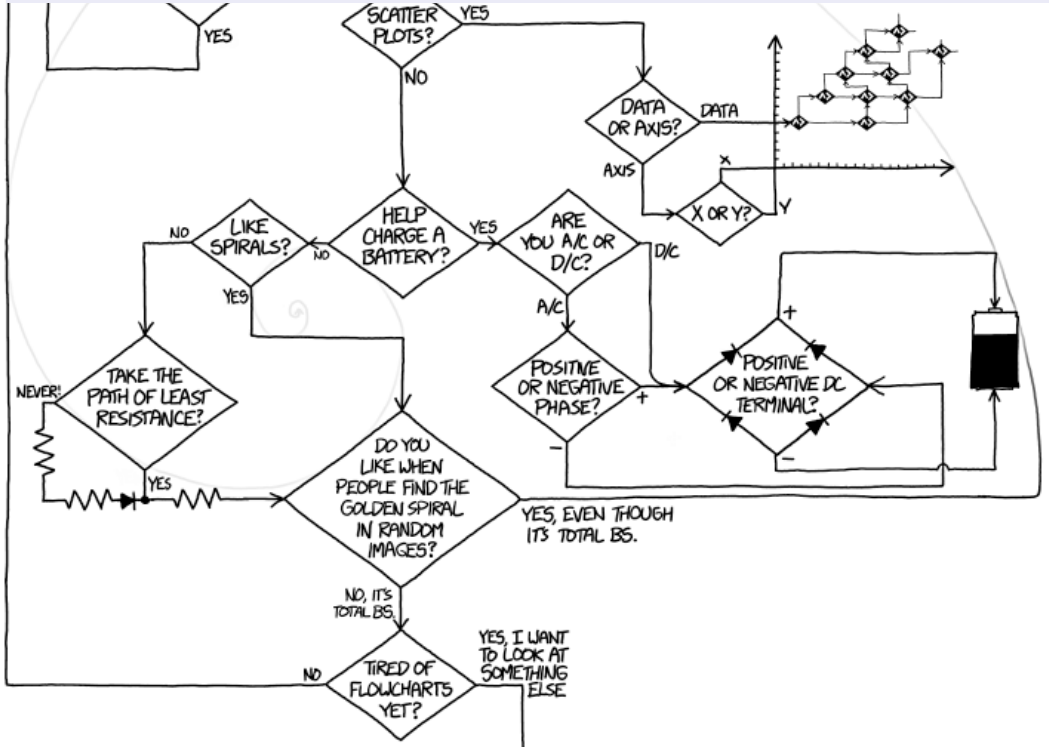


Cas #2 : l'attaque est trop coûteuse pour le monde réel

Cryptanalyste

Théorème

Cet algorithme s'exécute en moins de 2^x operations et $x < 128$

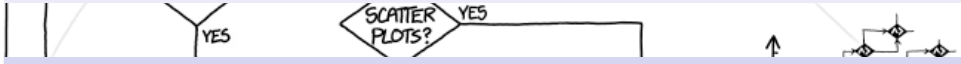


Cas #2 : l'attaque est trop coûteuse pour le monde réel

Cryptanalyste

Théorème

Cet algorithme s'exécute en moins de 2^x operations et $x < 128$



Démonstration.

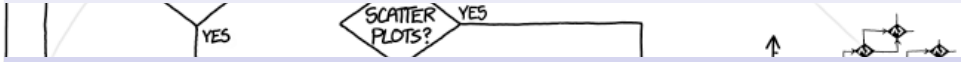
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. □

Cas #2 : l'attaque est trop coûteuse pour le monde réel

Crypta te

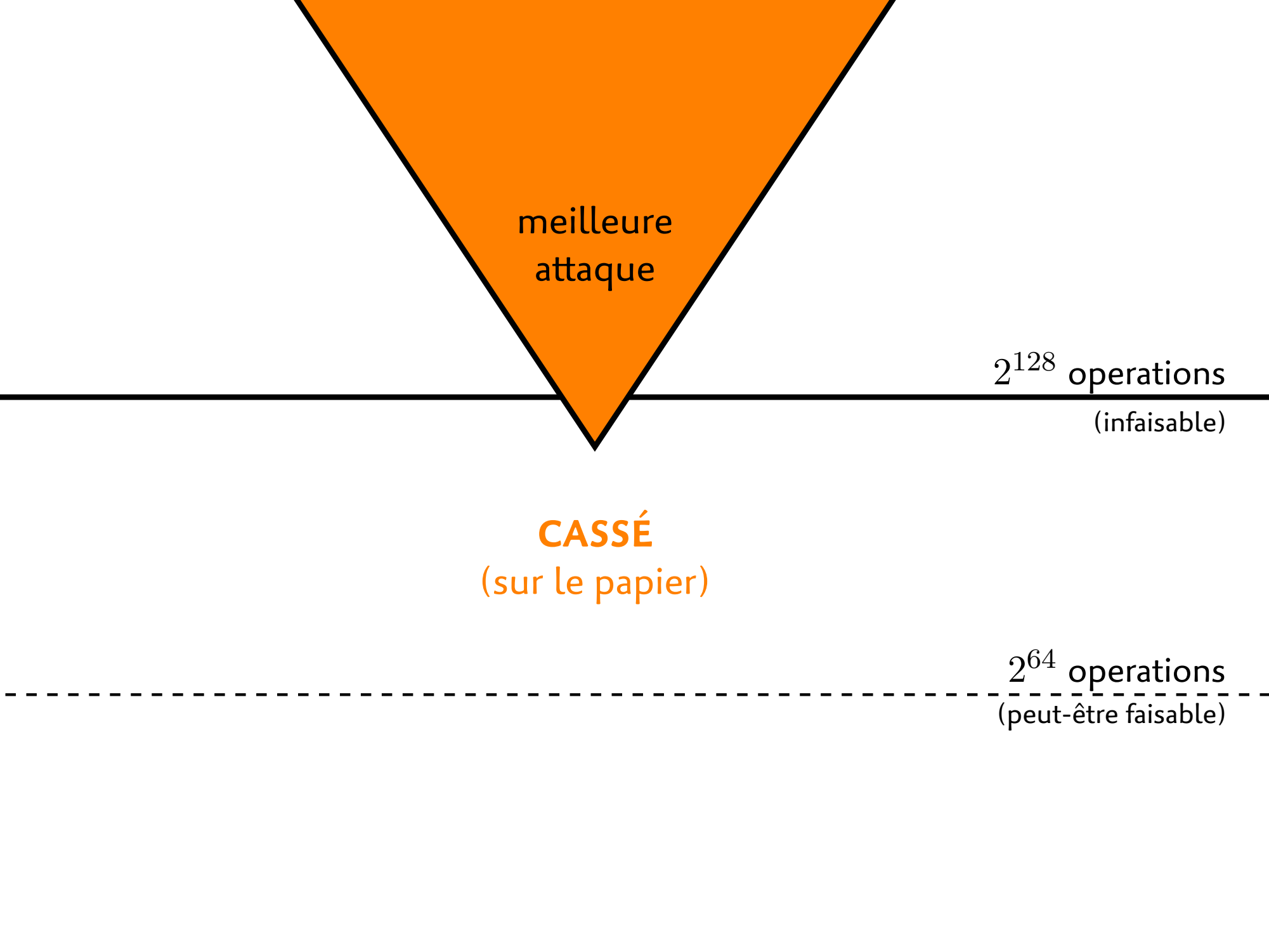
Théorème

Cet algorithme s'exécute en moins de 2^x operations et $x < 128$



Démonstration.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. □


meilleure
attaque
2^{128} operations
(infaisable)
CASSÉ
(sur le papier)
2^{64} operations
(peut-être faisable)

La sécurité est (presque tout le temps) **calculatoire**

- L'étude du **coût des calculs** joue un **rôle central** dans toute la cryptologie
  - Pour définir la sécurité
  - Pour établir la dangerosité des attaques

Rigueur mathématique

- **Modèle de calcul** abstrait
  - Historiquement : λ-calcul, machine de Turing, ...
  - Plus simple que le vrai matériel
- **Idées algorithmiques abstraites**
  - formulées dans le « langage » du modèle
- **Fonction de coût**
  - Donne le « coût » de l'exécution dans le modèle

Qu'est-ce qu'un « bon » modèle de calcul ?

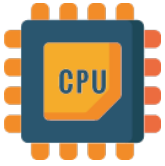
(pour la cryptologie)

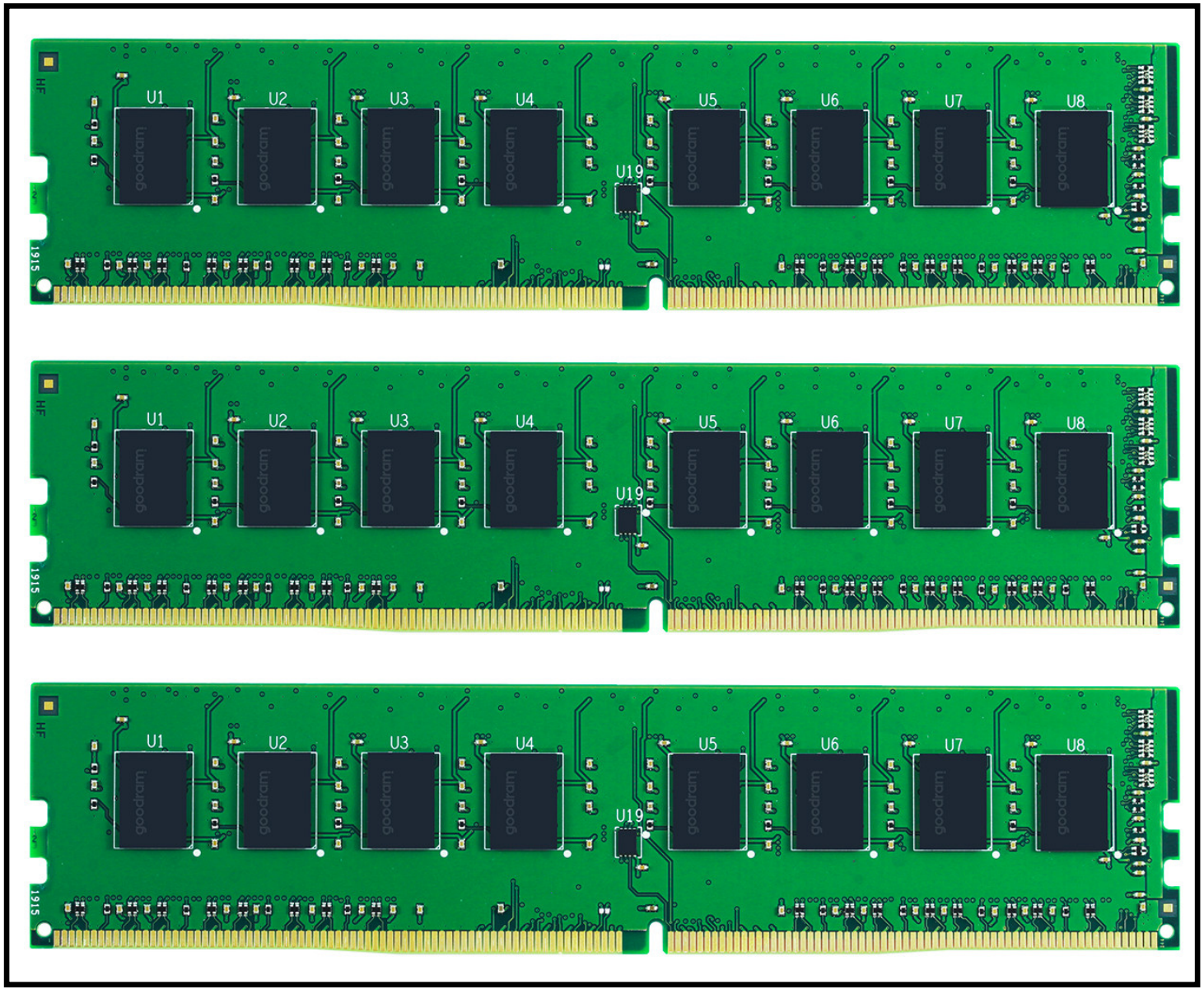
Cahier des charges

1. Simple
   - Pas d'opération bizarre/inutile
     - (~~`call/cc`~~, ~~`regex`~~, ...)
2. Expressif
   - `+`,`-`,`*`,`/`,`if`, `for`, `while`,..., (~~turing machine~~, ~~λ-calcul~~)
3. Abstrait
   - ~~*Starting from Ivy Bridge processors, there is an undocumented hardware next-page TLB prefetcher for virtual 4K pages*~~ (**NON**)
4. Fidèle au monde réel
   - « Réaliste »

La **Random Access Machine**

Le modèle de calcul usuel

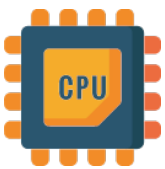


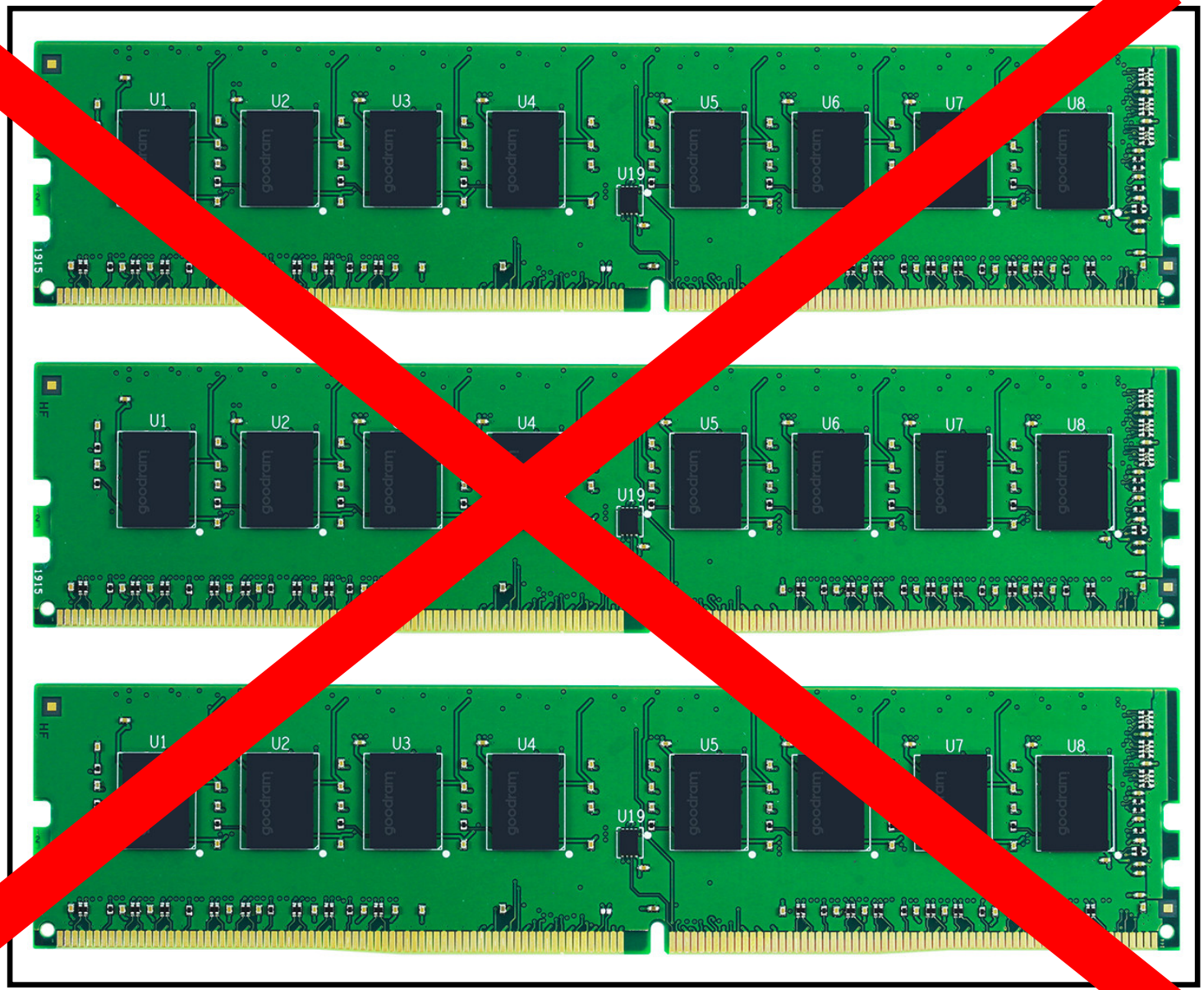


Fonction de coût :
opérations
élémentaires

La **Random Access Machine**

Le modèle de calcul usuel

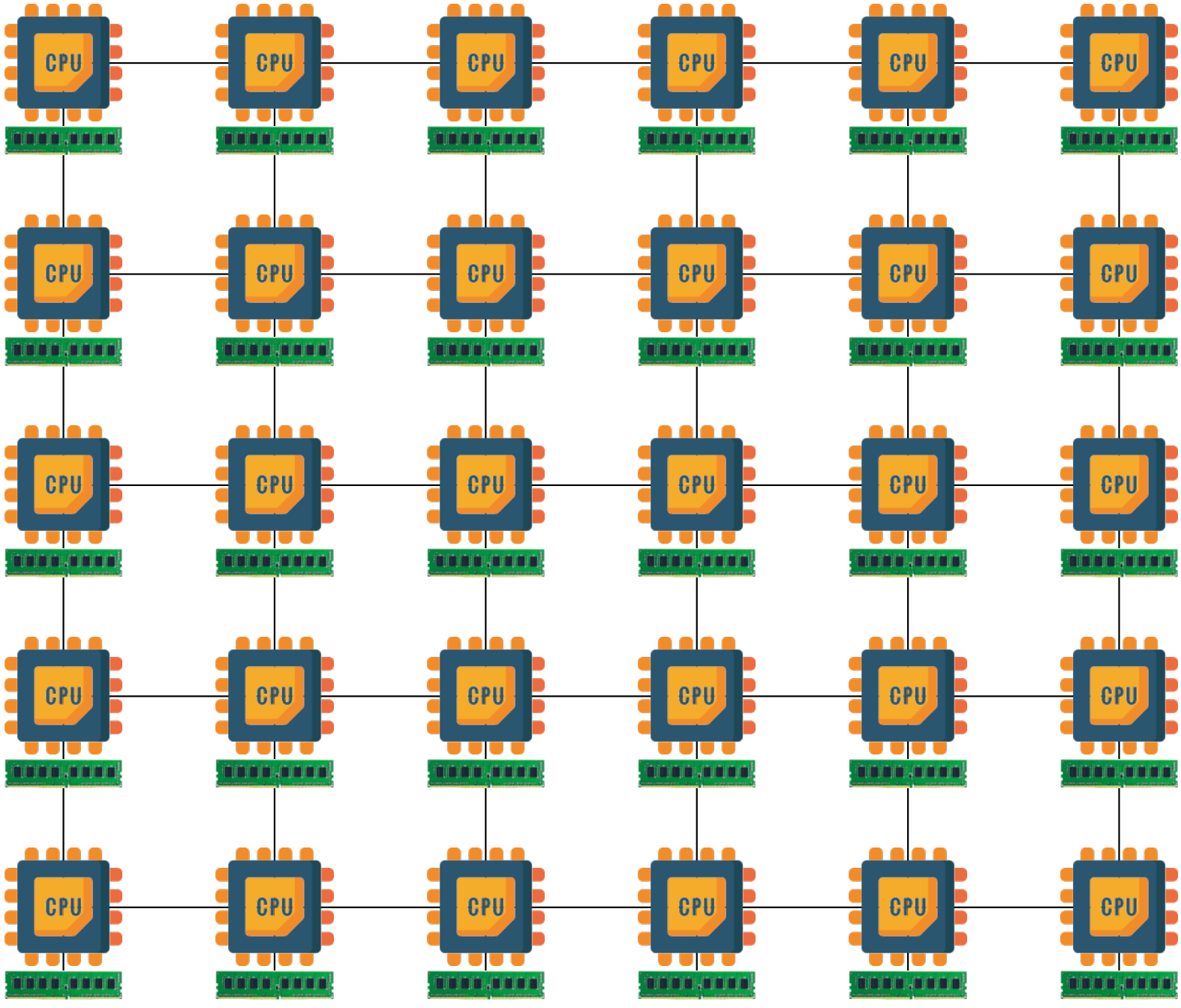




Fonction de coût :
opérations
élémentaires

Random Access Machine

Dans la vraie vie, gros calcul =



Effets **pervers** du modèle de la *Random Access Machine*

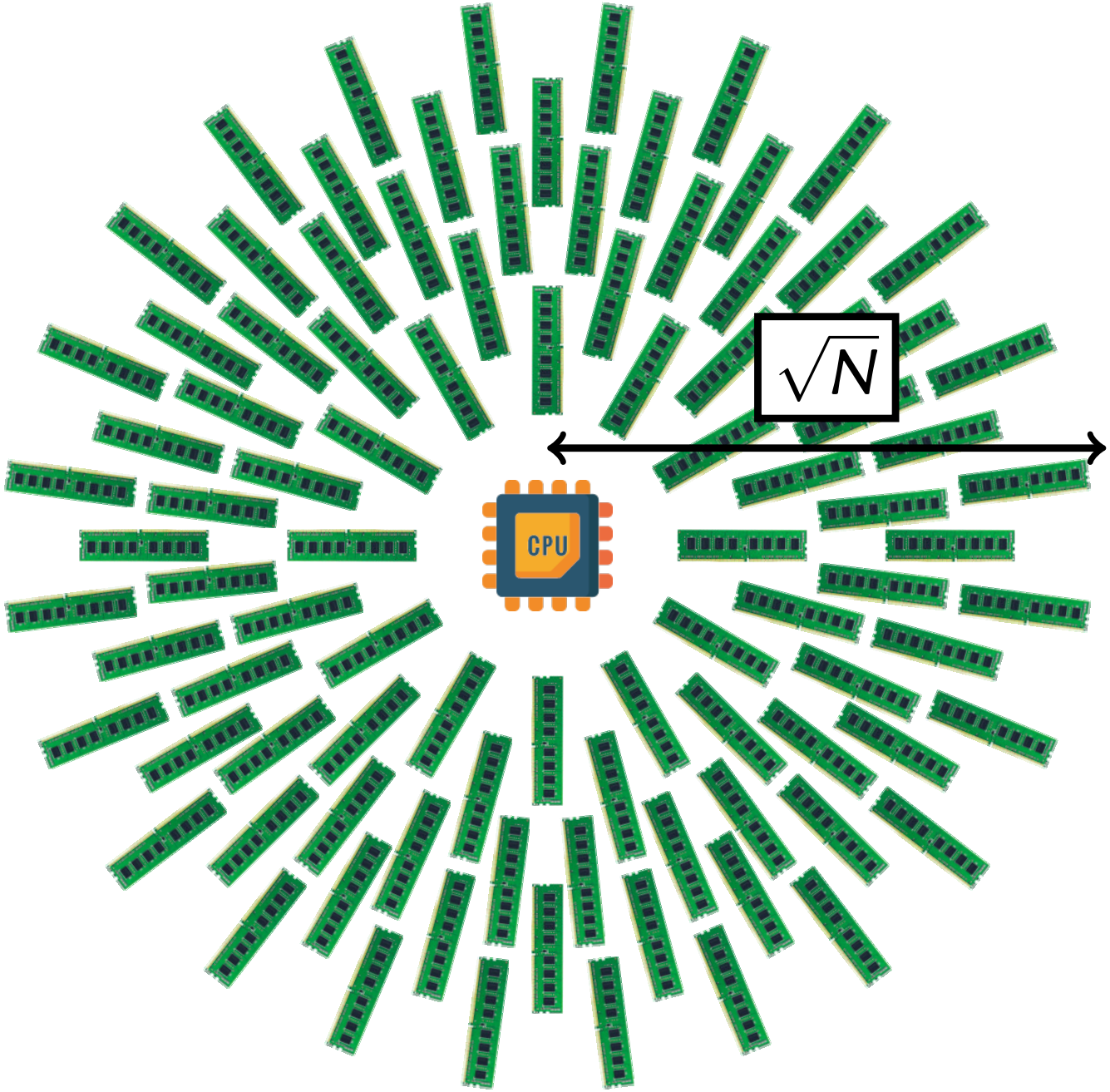
Problème #1 : sous-estime le coût des calculs

- Accès en temps $\mathcal{O}(1)$ à une mémoire arbitrairement grande ?
- Empiriquement faux
- Physiquement impossible

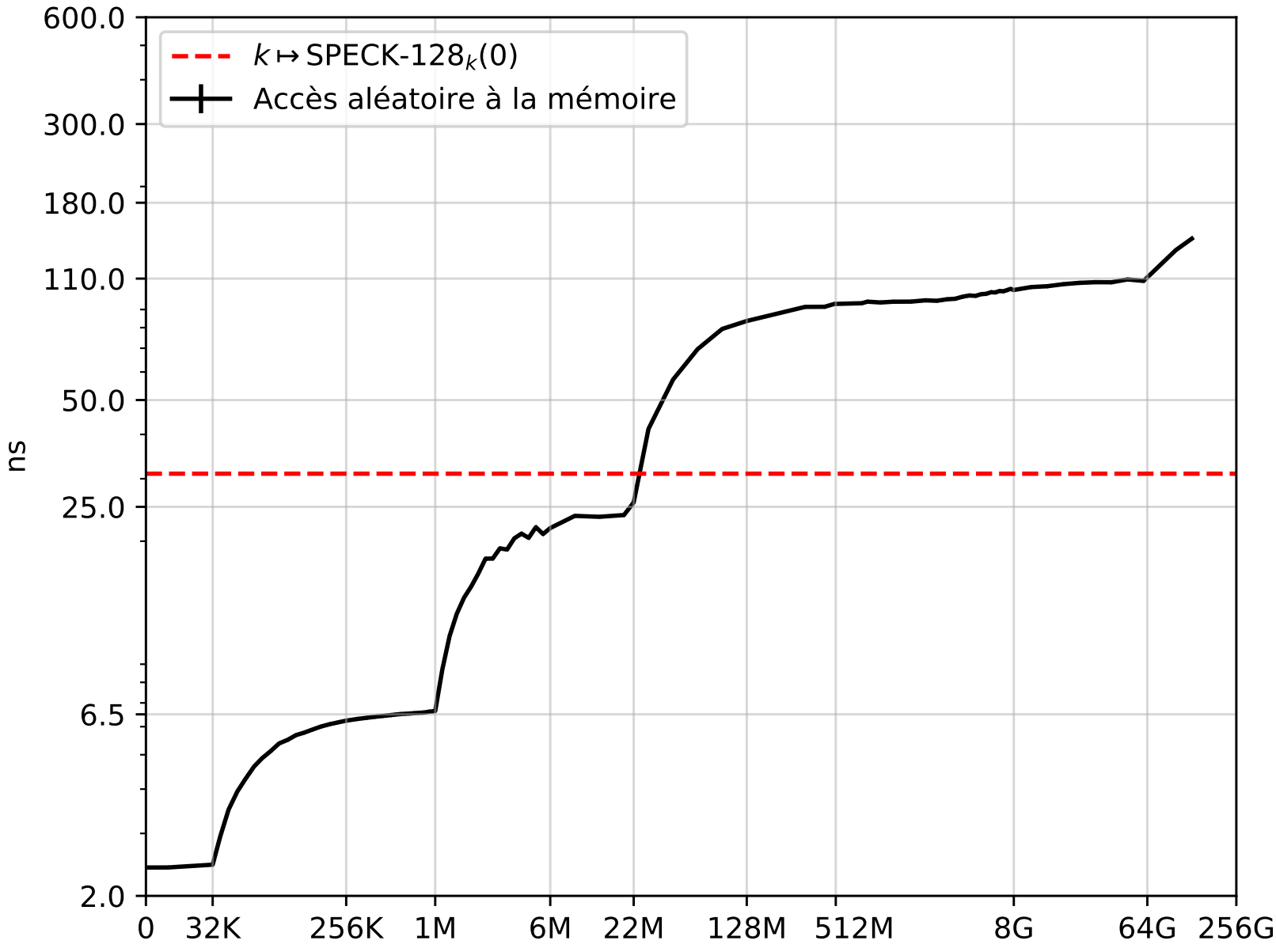
Problème #2 : favorise des algorithmes « théoriques »

- Tendance à utiliser + **[mémoire]** pour faire - **[# opérations]**
- Uniquement séquentiel

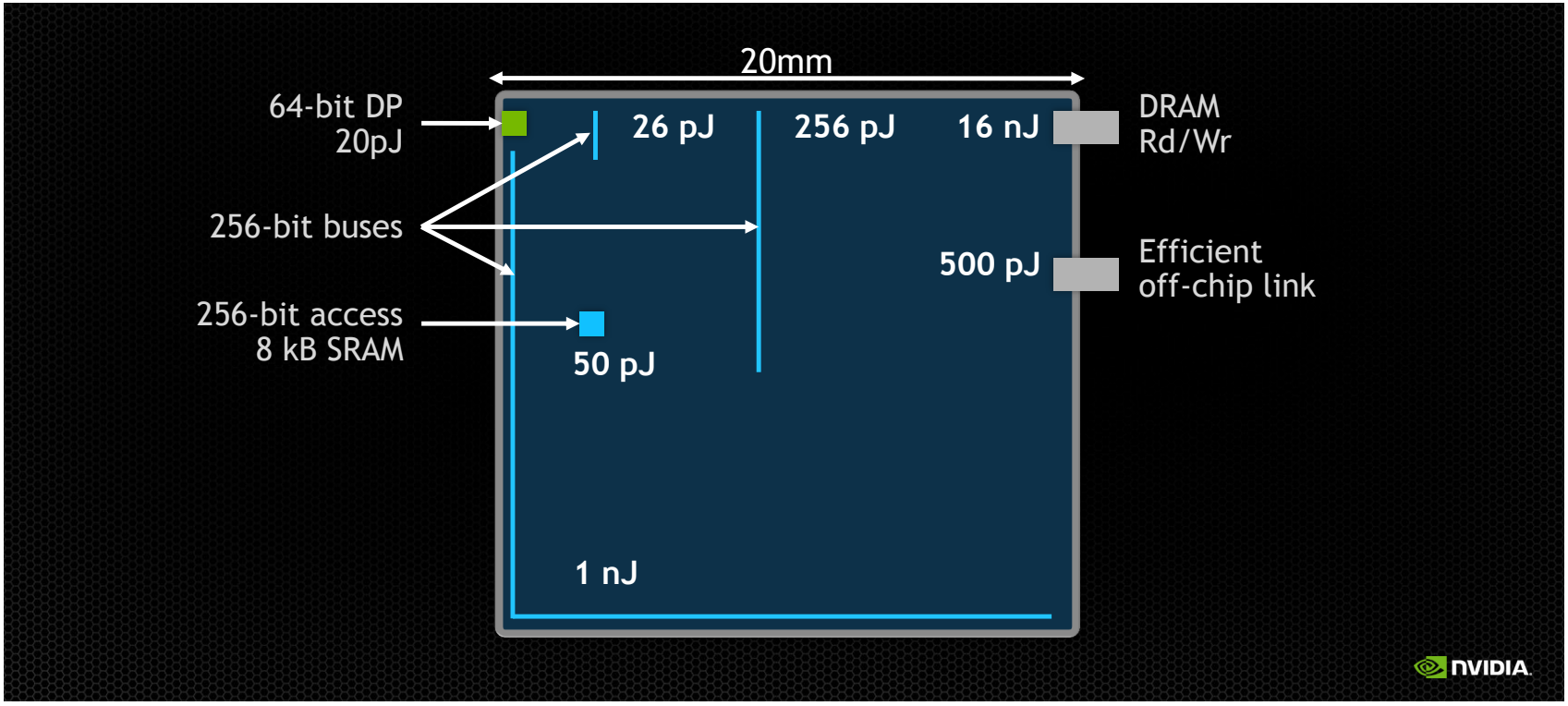
Accéder à la mémoire est couteux



Accéder à la mémoire est couteux



Déplacer des données est (énergétiquement) couteux



(image : Bill Dally, NVIDIA, « the path to exascale »)

Consommation énergétique sur un CPU normal

- Lire la RAM = 10× multiplication flottante

Classic McEliece, NTRU Prime, Rainbow

Finalistes de la compétition « Post-Quantique » du NIST

Classic McEliece 3rd-round submission, §8.2

[...] *Subsequent ISD variants have reduced the number of bit operations considerably below* 2^{256}. *However none of these analyses took into account the costs of memory access.*

NTRU Prime 3rd-round submission, §6.6

[we] estimate the cost of each access to a bit within N bits of memory as the cost of $N^{0.5}/2^5$ *bit operations.*

$\Longrightarrow$ « Nouveau » modèle de calcul

Coût = [# **opérations locales**] + $\sqrt{N}\times$ [# **accès mémoire**]

Resucée du « *Hierarchical Memory Model* (HMM) »

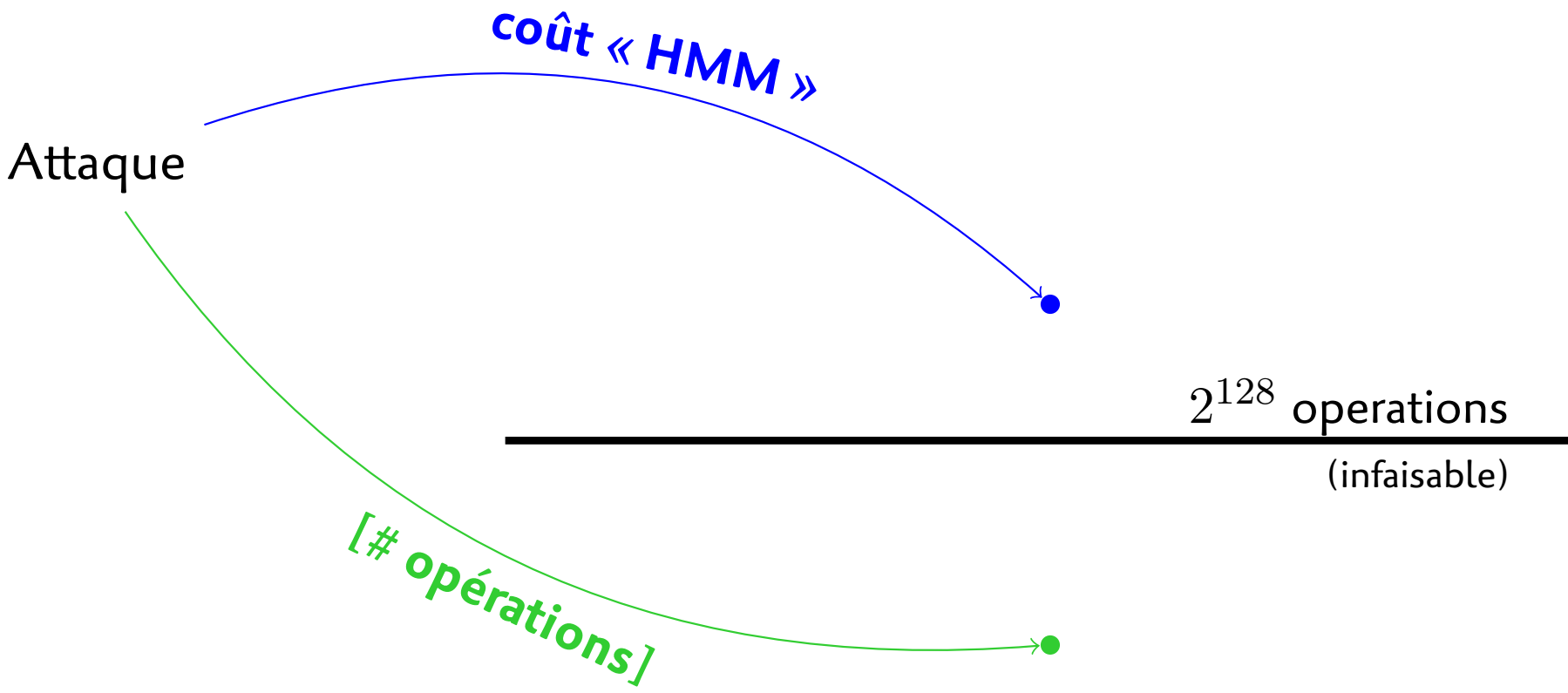
- Aggarwal, Alpern, Chandra, Snir (**1987**)
- Accéder à `M[i]` coûte $f(i)$
  - Par exemple $f(i) = \log_2 i$,
  - Ou $f(i) = i^\alpha$ (avec par ex. $\alpha = 1/2$)

Theorem

Avec $f(i) = i^\alpha$, la recherche dichotomique coûte $\Omega\left(n^\alpha\right)$ et le tri coûte $\Omega\left(n^{1+\alpha}\right)$.

$\Longrightarrow$ Hierarchical Memory Model

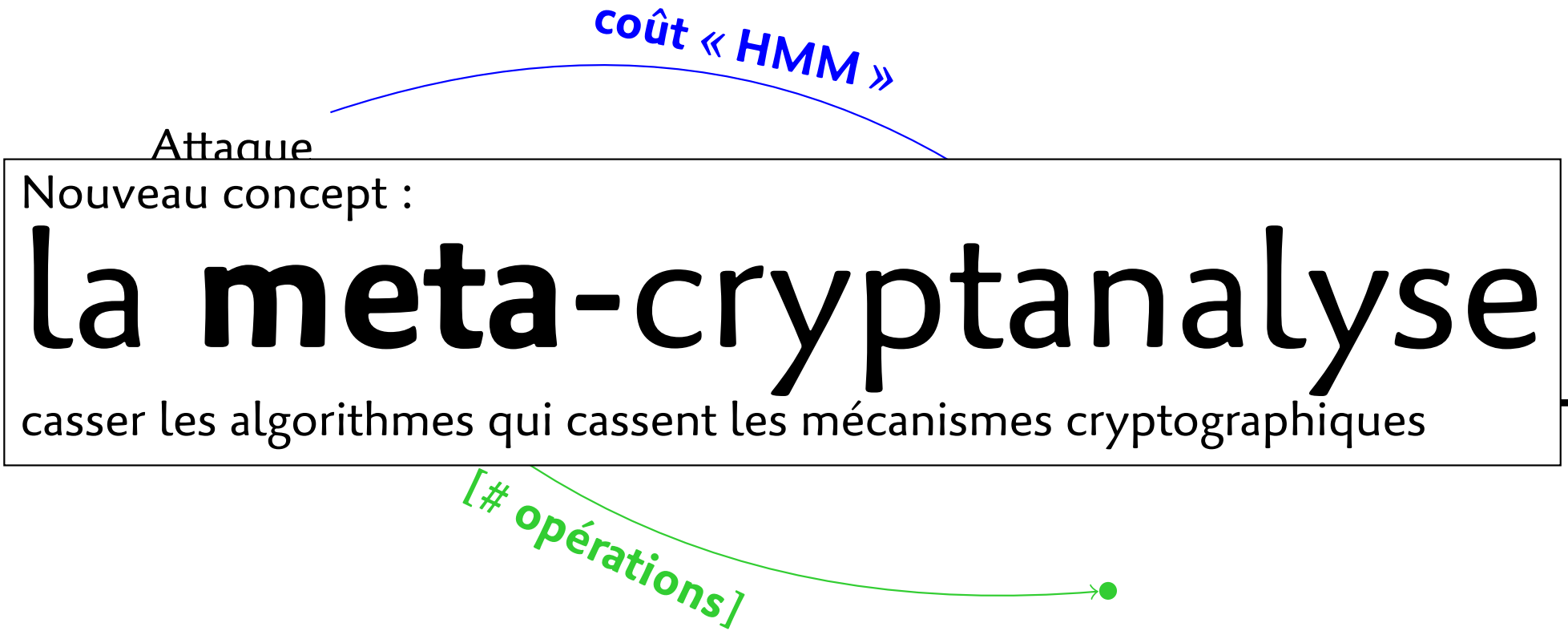
Coût = [# **opérations locales**] + $\sqrt{N}\times$ [# **accès mémoire**]



- [Comment/Est-ce que] la communauté se positionne ?

$\Longrightarrow$ Hierarchical Memory Model

Coût = [# **opérations locales**] + $\sqrt{N}\times$ [# **accès mémoire**]



- [Comment/Est-ce que] la communauté se positionne ?

Effets **pervers** du modèle de la *Random Access Machine*

Problème #1 : sous-estime le coût des calculs

- Accès en temps $\mathcal{O}(1)$ à une mémoire arbitrairement grande ?
- Empiriquement faux
- Physiquement impossible

Problème #2 : favorise des algorithmes « théoriques »

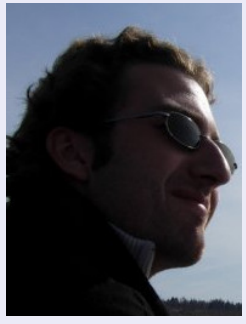
- Tendance à utiliser + **[mémoire]** pour faire - **[# opérations]**
- Uniquement séquentiel

La *Random Access Machine* est séquentielle

La meilleure attaque sur l'AES-128 (récupération de clef)

Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting
(EUROCRYPT 2013)


Patrick Derbez


Jérémy Jean


Pierre-Alain Fouque

- 7 tours de l'AES-128, 2^{100} opérations, mémoire de taille 2^{100}
  - invalide dans le HMM

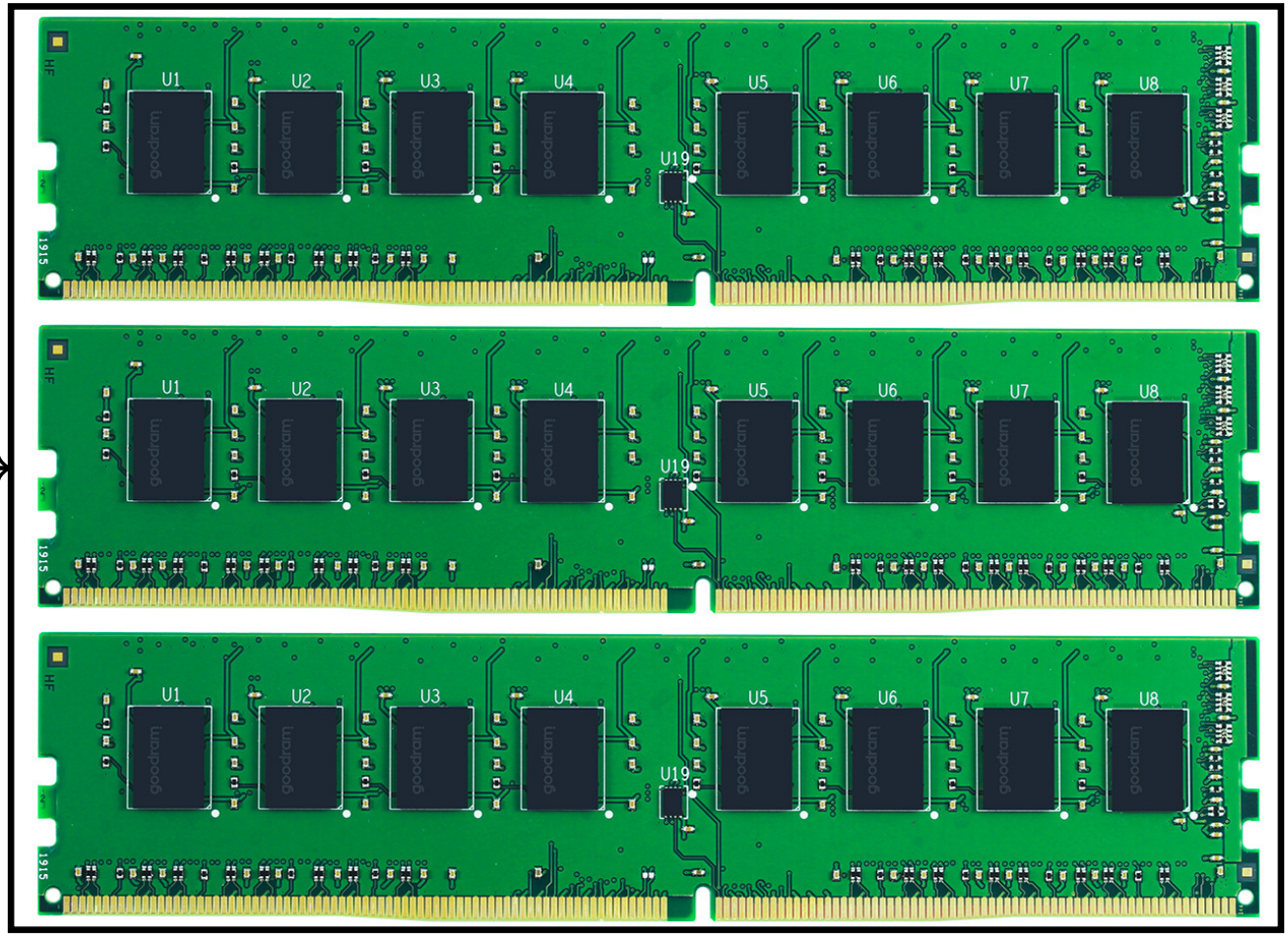
La *Random Access Machine* est séquentielle

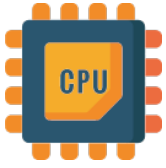
La meilleu de clef)

- 7 tour taille 2^{100}
  - inv

Attaque sophistiquée

$M = 2^{100}$

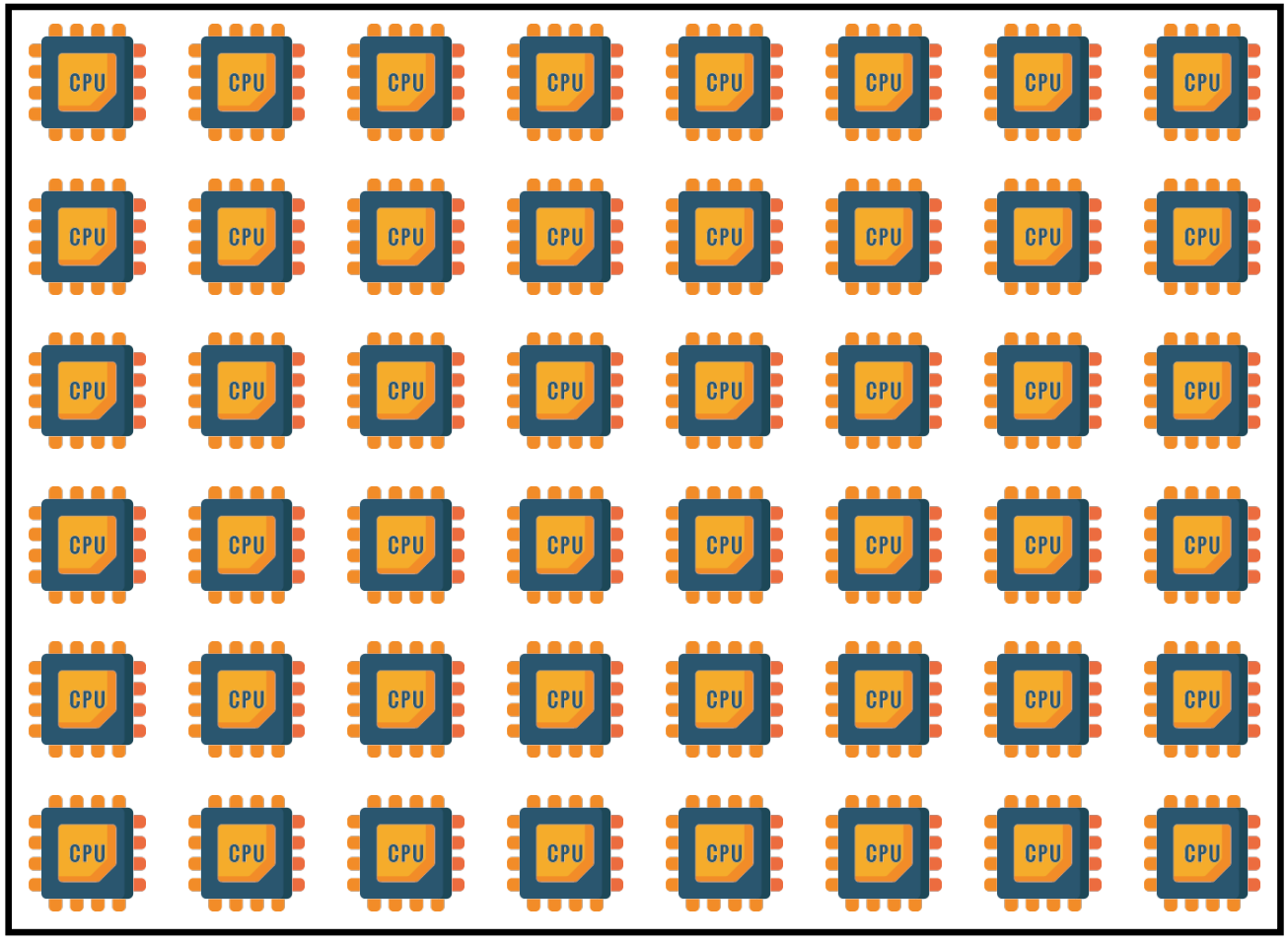




$T = 2^{100}$

Recherche exhaustive

2^{100} processeurs



~~$T = 2^{100}$~~

$T = 2^{28}$

Difficile d'échapper à cet argument

Une attaque qui demande 2^{100} blocs de mémoire...

- Nécessite énormément de hardware
- Transformation : **stockage** $\rightsquigarrow$ **calculs**
- Et hop ! On a les moyens de faire la recherche exhaustive
- (et plus on consomme de mémoire, plus ça s'aggrave)

not so sure anymore

Résumé provocateur

On s'intéresse à des attaques

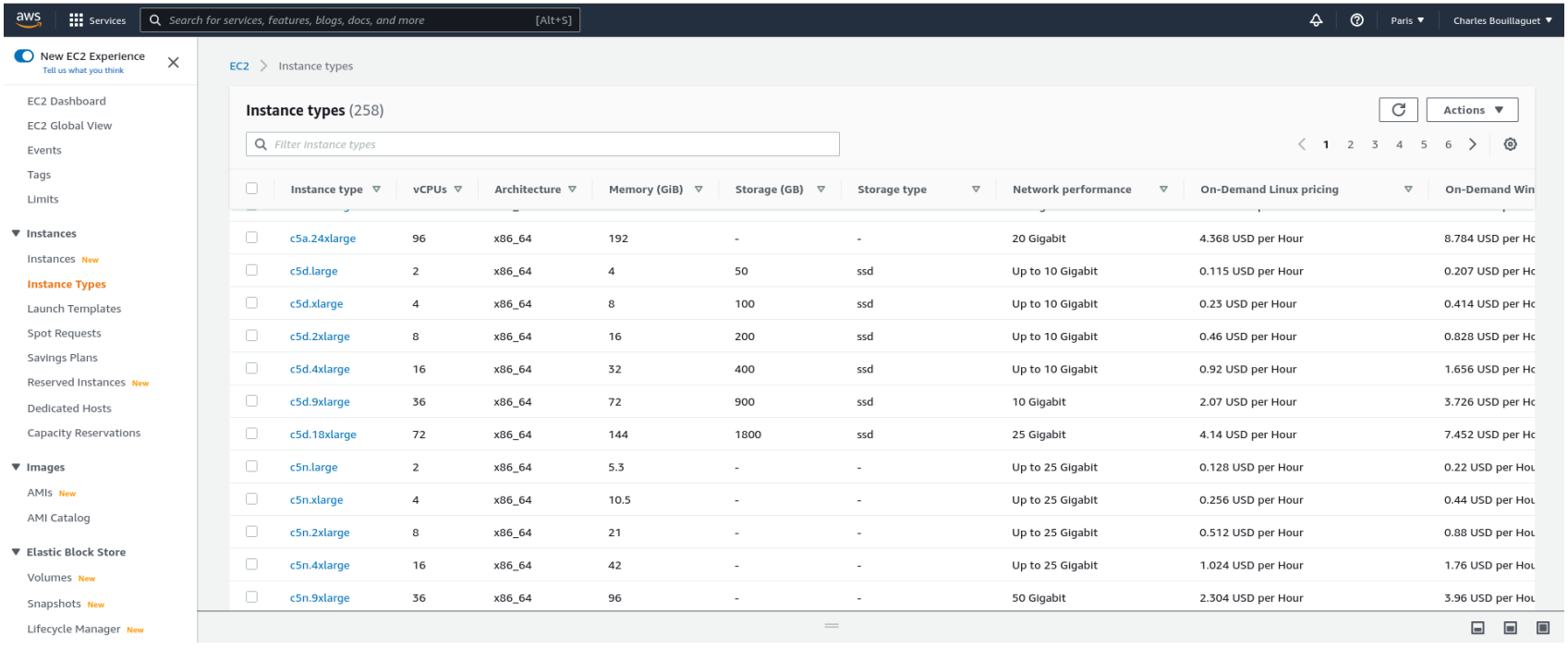
- qui nécessitent dans le fond d'avoir les moyens de faire la recherche exhaustive
- alors qu'on considère que la recherche exhaustive est impossible

Une autre manière de regarder le même problème

Une autre manière de regarder le même problème

Nouveau modèle de calcul : le *cloud* !

- On *loue* des machines virtuelles
- Fonction de coût = $$$



C'est très différent !

Beaucoup de mémoire $\rightsquigarrow$ beaucoup de VMs $\rightsquigarrow$ $$$$$$$$$$$

Using the Cloud to Determine Key Strengths Triennial Update

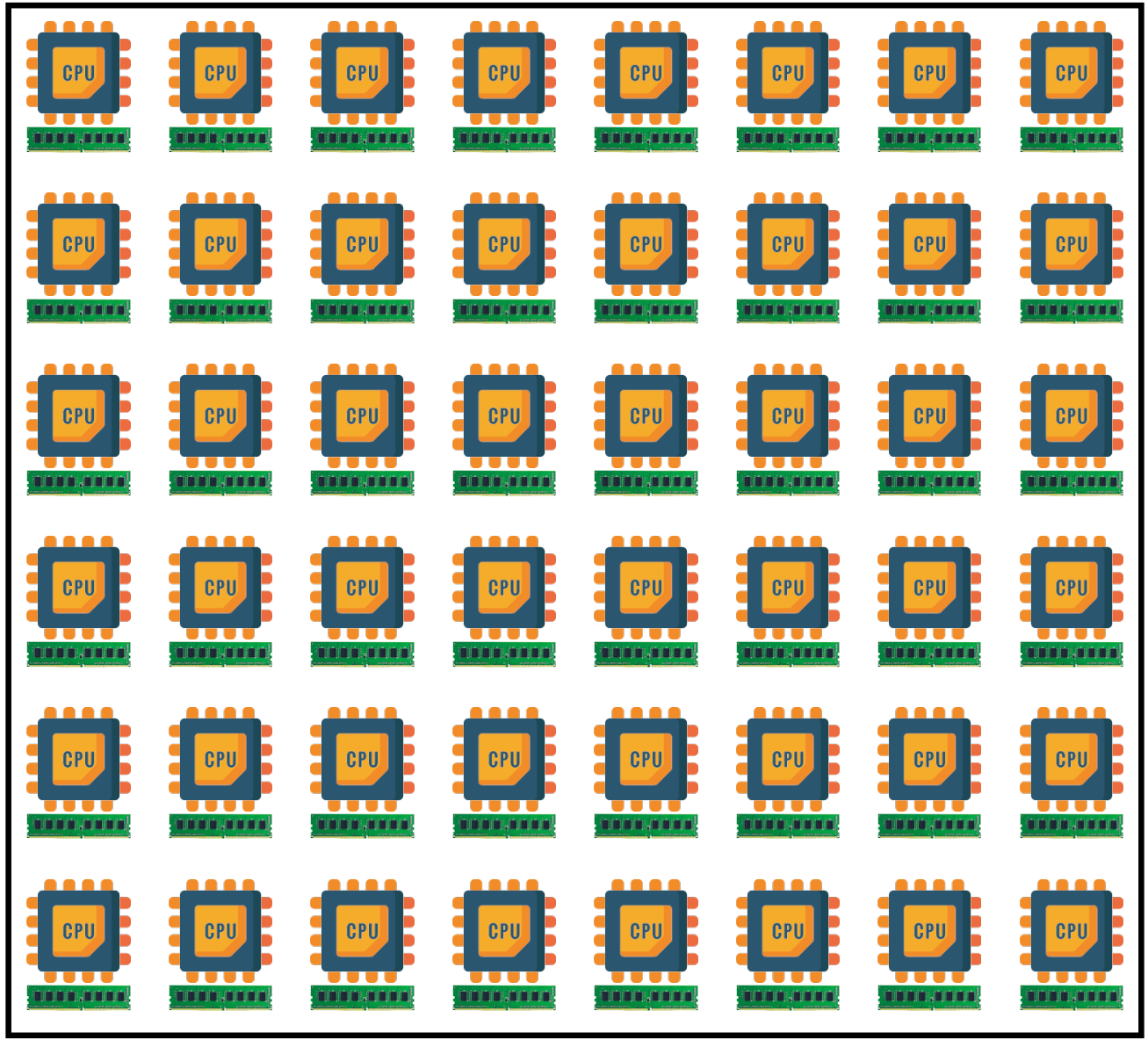
Marguerite Delcourt[1,2], Thorsten Kleinjung[1], Arjen K. Lenstra[1], Shubhojyoti Nath[1,3], Dan Page[4], and Nigel P. Smart[5]

[1] EPFL IC IINFCOM LACAL, Station 14, CH-1015 Lausanne, Switzerland.
[2] EPFL IC IINFCOM LCA2, Station 14, CH-1015 Lausanne, Switzerland.
[3] Indian Institute of Technology Kanpur, India.
[4] Dept. Computer Science, University of Bristol, Merchant Venturers Building, Woodland Road, Bristol, BS8 1UB, United Kingdom.
[5] COSIC, Departement Elektrotechniek (ESAT), Kasteelpark Arenberg 10 - bus 2440, 3001 Leuven, Belgium.

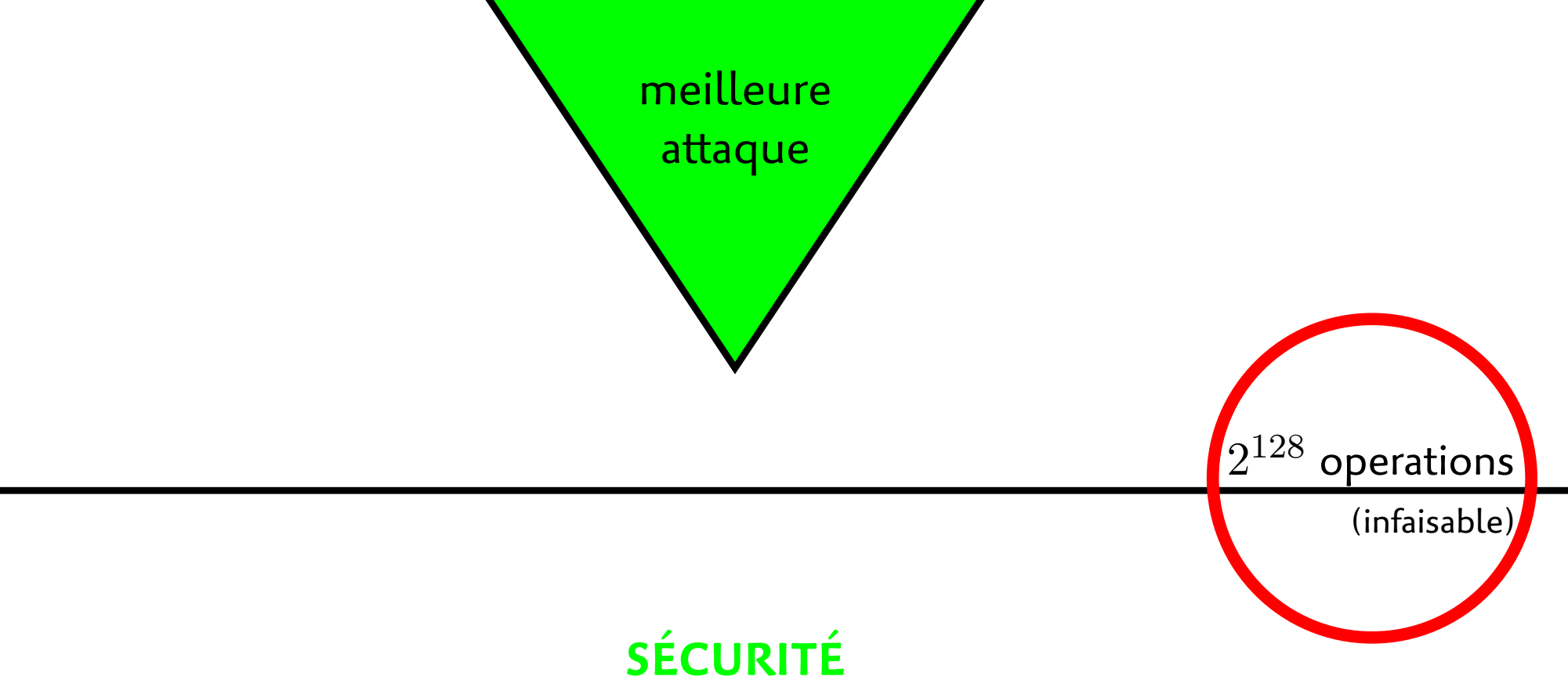
Abstract. We develop a new methodology to assess cryptographic key strength using cloud computing, by calculating the true economic cost of (symmetric- or private-) key retrieval for the most common cryptographic primitives. Although the present paper gives the current year (2018), 2015, 2012 and 2011 costs, more importantly it provides the tools and infrastructure to derive new data points at any time in the future, while allowing for improvements such as of new algorithmic approaches. Over time the resulting data points will provide valuable insight in the selection of cryptographic key sizes. For instance, we observe that the past clear cost-advantage of total cost of ownership compared to cloud-computing seems to be evaporating.[6]

Sur le cloud : meilleur que la force brute ?

2^{100} processeurs + 2^{100} mémoire



$\$\$ = ???$


meilleure
attaque
2^{128} operations
(infaisable)
SÉCURITÉ

2^{128} opérations maxi ?

On se met d'accord sur des *limites* :

- **[Temps]** $\leq 10^{10}$ s
- **[Taille de la machine]** $\leq 10^{10}$ CPUs
- **[# op/s]** $\leq 10^{10}$ opérations / s (par CPU)

De manière logique :

- **[# op]** $\leq$ **[# op/s]** $\times$ **[Temps]** $\times$ **[Taille de la machine]**

On en conclut :

- **[# opérations]** $\leq 10^{30}$ $(\approx 2^{100})$

2^{128} opérations maxi ?

On se met d'accord sur des *limites* :

- **[Temps]** $\leq 10^{10}$ s
- **[Taille de la machine]** $\leq 10^{10}$ CPUs
- **[# op/s]** $\leq 10^{10}$ opérations / s (par CPU)

De manière logique :

- $\underbrace{\textbf{[\# op]}}_{f(\mathcal{A})} \leq$ **[# op/s]** $\times \underbrace{\textbf{[Temps]} \times \textbf{[Taille de la machine]}}_{g(\mathcal{A})}$

On en conclut :

- **[# opérations]** $\leq 10^{30}$ $(\approx 2^{100})$
- **[Temps]** $\times$ **[Taille de la machine]** $\leq 10^{20}$ CPU·s

Ça rappelle de vieux souvenirs

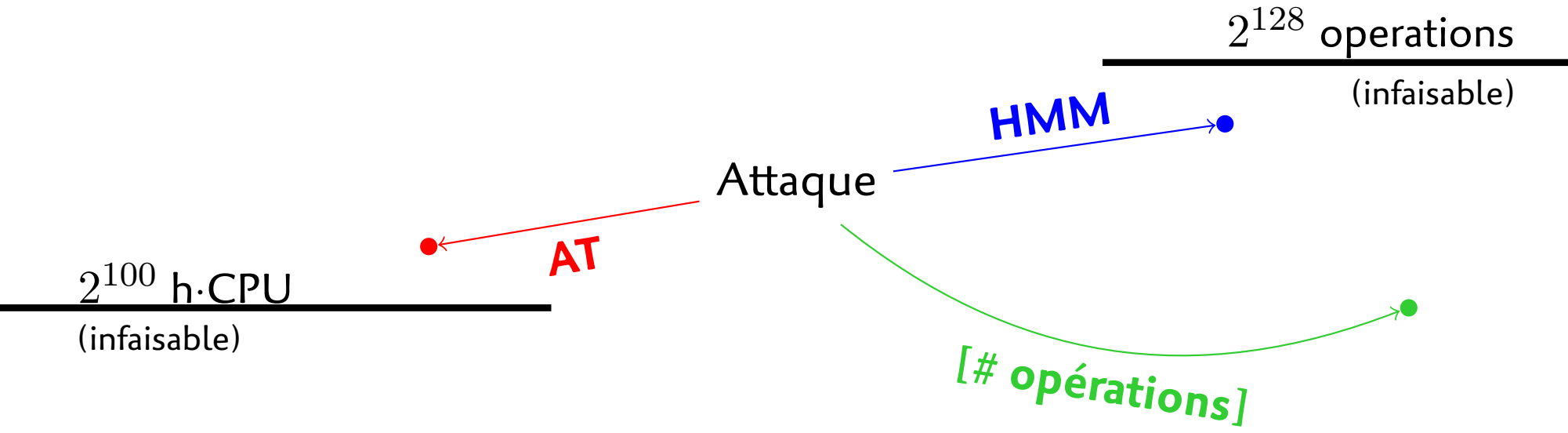
Modèle VLSI (1970–1980)

- Circuit planaire
- Coût = [**Aire**] $\times$ [**Temps**], limite naturelle

Modèle *cloud*/VLSI (« AT »)

- **Le plus réaliste** (à mon avis)
  - $\approx$ prix de la location sur un *cloud* public
  - $\approx$ # heures-CPU dans un centre de calcul
  - $\approx$ énergie nécessaire
- Algorithme séquentiel $\rightsquigarrow$ coût = TM
  - Pour baisser, il faut paralléliser
- Coût des communications
- Coûts **asymptotiquement plus élevés** que [# **ops**] pour
  - Tri, FFT, multiplication, intersection, jointure, ...

Conséquence inévitable de tout ceci



Résumons

Modèle de calcul usuel

- Random Access Machine
- Coût = [# **opérations**], limite naturelle

Modèle HMM (1987, revival avec le NIST)

- Random Access Machine
- Coût = [# **opérations locales**] + $\sqrt{N}\times$ [# **accès mémoire**]

Modèle VLSI (1970–1980, revival avec le cloud)

- Circuit planaire
- Coût = [**Taille de la machine**] $\times$ [**Temps**], limite naturelle

Résumons

Modèle de calcul usuel

- Random Access Machine
- Coût = [# **opérations**] limite naturelle

Modèle HMM (1[illegible]97, revival avec le NIST)

- Random Access Machine
- Coût = [# **opérations locales**] + $\sqrt{N}$ [# **accès mémoire**]

Modèle VLSI (1970–198[illegible] revival avec le cloud)

- Circuit planaire
- Coût = [**Taille de la machine**] [**Temps**] limite naturelle

C'est l'heure de la démarche scientifique

On a des **modèles** en compétition

- Censés décrire le **monde réel**
- Peut-on les valider expérimentalement ?

Exemple simple : le tri

- Composant usuel d'attaques cryptographiques
- Modèle **RAM** : $T \approx n \log_2 n$
- Modèle **HMM** : coût $\approx n^{1+\alpha}$ (quel α ?)
- Modèle **VLSI** : coût $\approx n^{1.5}$ (en 2D)

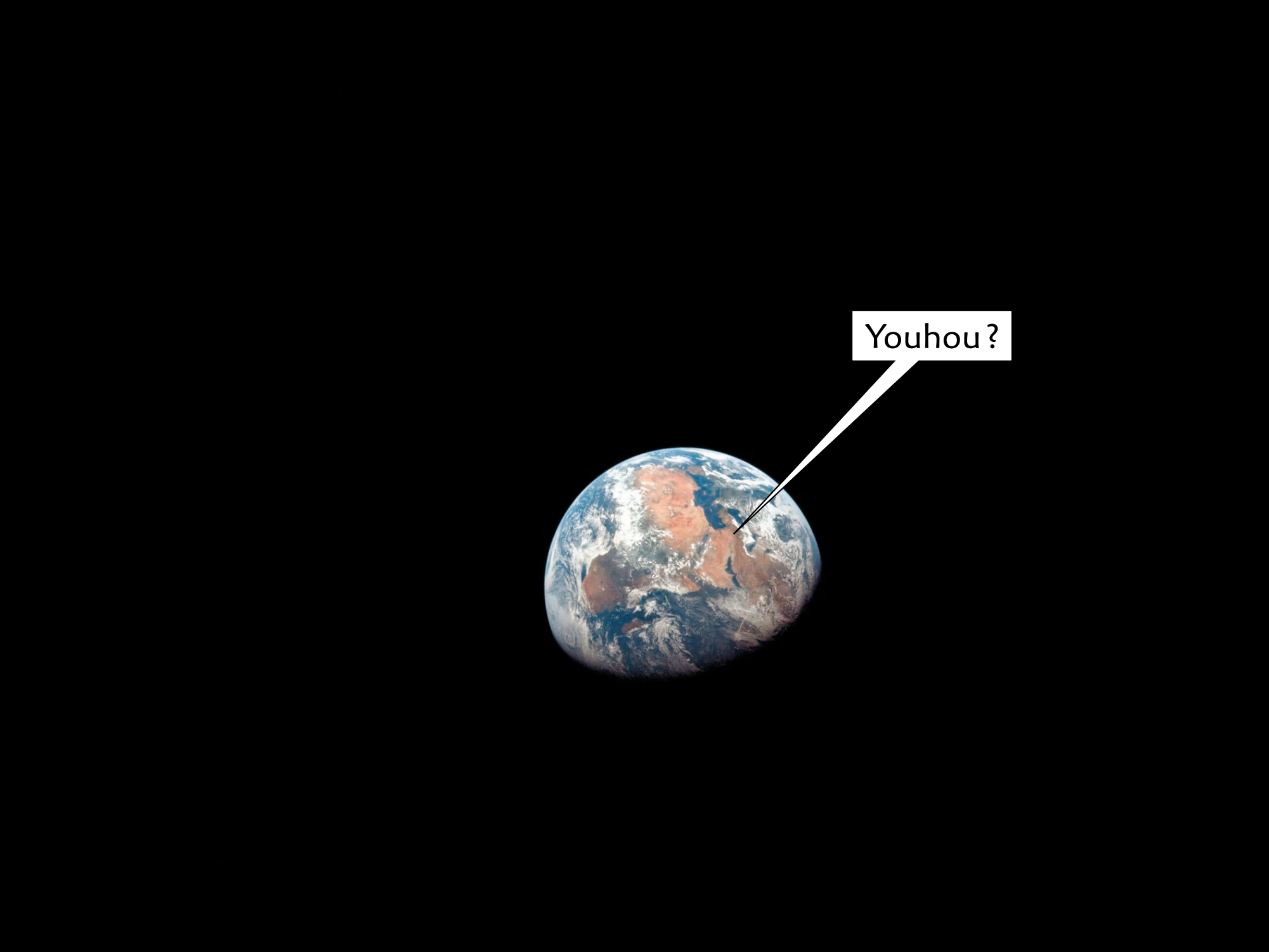
Quel modèle a raison ?

Conclusion

- Problèmes du modèle RAM
  - Personne ne les conteste
- Tout le monde continue à utiliser le modèle RAM (!)
  - Éducation, habitude, mimétisme, ...
  - Fiable du point de vue de la sécurité
- Tant qu'on ne se confronte pas au vrai matériel
  - On ne peut pas percevoir le décalage « théorie $\leftrightarrow$ pratique »
  - Communauté crypto plutôt centrée sur la théorie

> *Mon bon ami, toute théorie est sèche,*
> *et l'arbre précieux de la vie est fleuri.*

- Attaques dans le modèle RAM
  - $\rightarrow$ `/dev/null` ?
  - Non. Mais *moins fortes* que celles dans le modèle VLSI


Youhou ?