Dimension algorithmique et chiffrement post-quantique

Gabriel Chênevert

12 octobre 2018

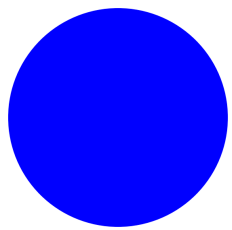
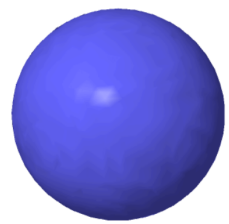
Dimension de Hausdorff

Notion formulée par Felix Hausdorff en 1918

Dimension de Hausdorff

Intuition

Mesure d'une boule de rayon r, dimension d :

segment	disque	boule	hyperboule
			?
$2r$	πr^2	$\frac{4\pi r^3}{3}$	$\frac{\pi^2 r^4}{2}$

Comportement sous dilatation :

$$\mathfrak{m}(\alpha A) = \alpha^d \mathfrak{m}(A)$$

Dimension de Hausdorff

Notion de contenu

$$\mathfrak{m}_d(A) := \inf \left\{ \sum_i r_i^d \;\middle|\; A \text{ couvert par des boules de rayon } r_i \right\}$$

Définition

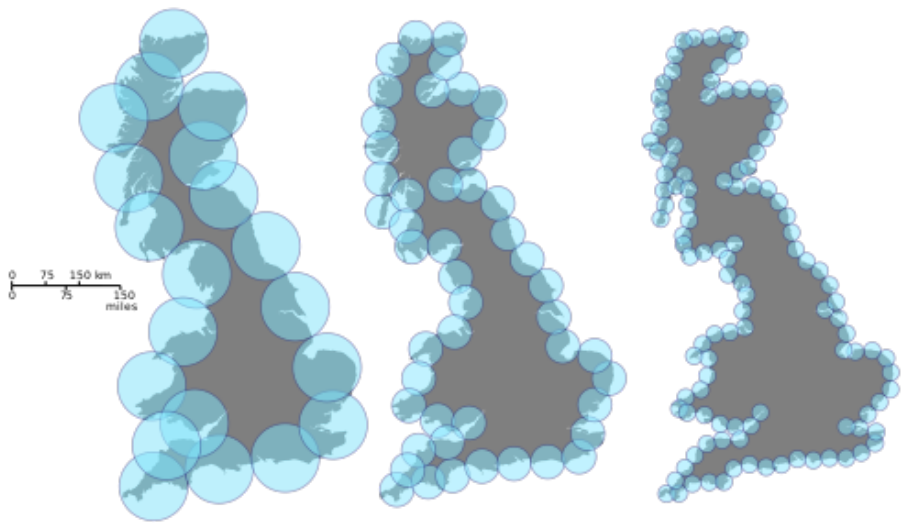
$$\dim A := \sup\{d \,|\, \mathfrak{m}_d(A) = +\infty\} = \inf\{d \,|\, \mathfrak{m}_d(A) = 0\}$$

Dimension de Hausdorff

Dans les cas simples

Si $N(r)$ désigne le nombre de boules de rayon r nécessaires pour couvrir A,

$$\dim A = \lim_{r \to 0} \frac{\log N(r)}{\log 1/r}.$$



Dimension algorithmique

Étant donné un algorithme A, on considère pour $n \in \mathbf{N}$ le nombre maximal A_n d'étapes de calcul effectuées par A sur une entrée de taille n.

Définition

$$\dim A := \lim_{n \to \infty} \frac{\log A_n}{\log n}$$

En particulier : si $A_n \sim C\, n^d$, alors $\dim A = d$.

Mais aussi par exemple $A_n = C\, n^d\, \log n\, (\log \log n)^4$

Dimension algorithmique

Exemples

- Énumérer les points à coordonnées entières dans une d-boule : dimension d
- Test de primalité naïf : dimension $\frac{1}{2}$
- Multiplication matricielle : dimension comprise entre 2 et 3
  - algorithme naïf : 3
  - Strassen : 2,807
  - Coppersmith-Winograd : 2,376

Dimension algorithmique

Algorithme d'Euclide

Le cas le plus défavorable est le calcul du PGCD de deux termes successifs dans la suite de Fibonacci : on obtient

$$A_n = C \log_\phi(n) \qquad \text{où} \qquad \phi = \frac{1+\sqrt{5}}{2}$$

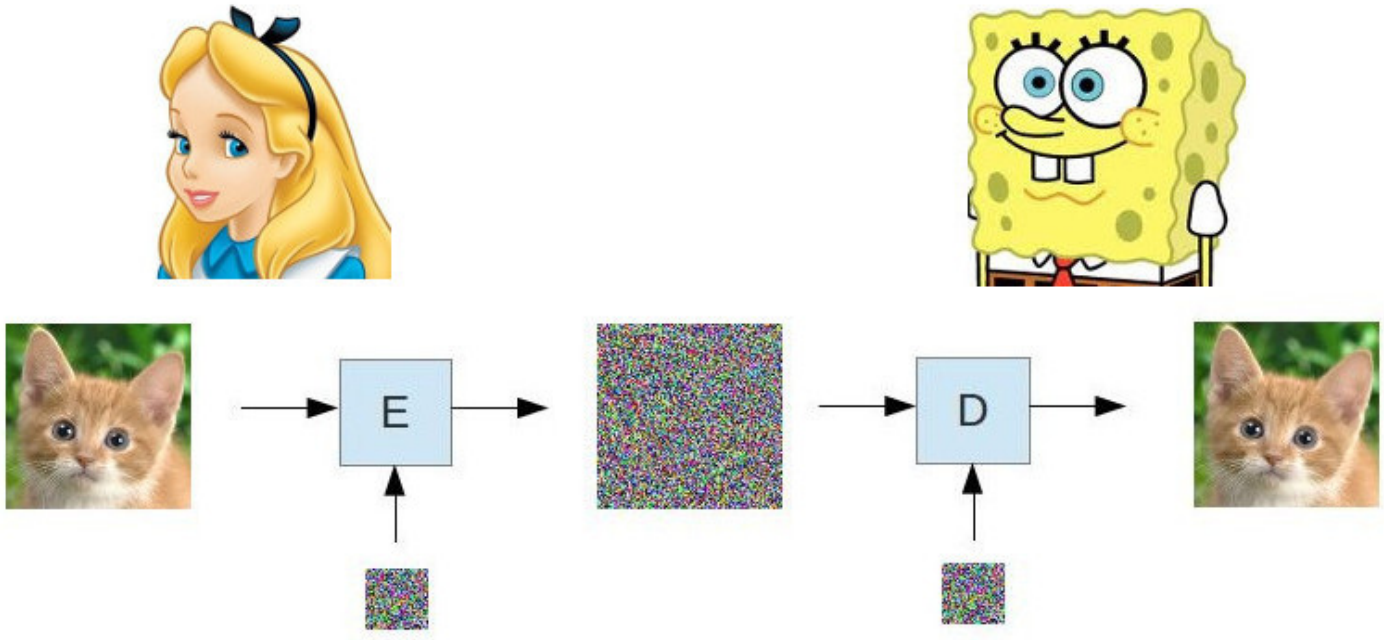
$$\Longrightarrow \dim A = 0$$

Disons plutôt

$$\log \dim A = 1.$$

Cryptographie

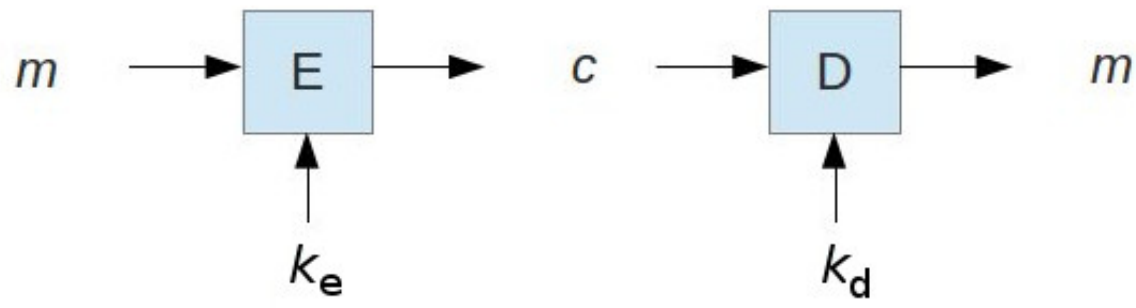
Chiffrement symétrique (AES)



- E et D : dimension finie par rapport à la *taille* de la clé
- attaque : dimension finie par rapport au *nombre* de clés

Cryptographie

Chiffrement asymétrique



- k_e publique, k_d privée : chiffrement à clé publique
- k_e privée, k_e publique : signature numérique

Cryptographie

RSA (1977)

Notons $a \underset{n}{\equiv} b$ pour dire que $n \mid (b - a)$.

Étant donné un (grand) entier n :

$$\begin{cases} E(e, m) \underset{n}{\equiv} m^e \\ D(d, c) \underset{n}{\equiv} c^d \end{cases}$$

Pour que ça fonctionne : on demande que

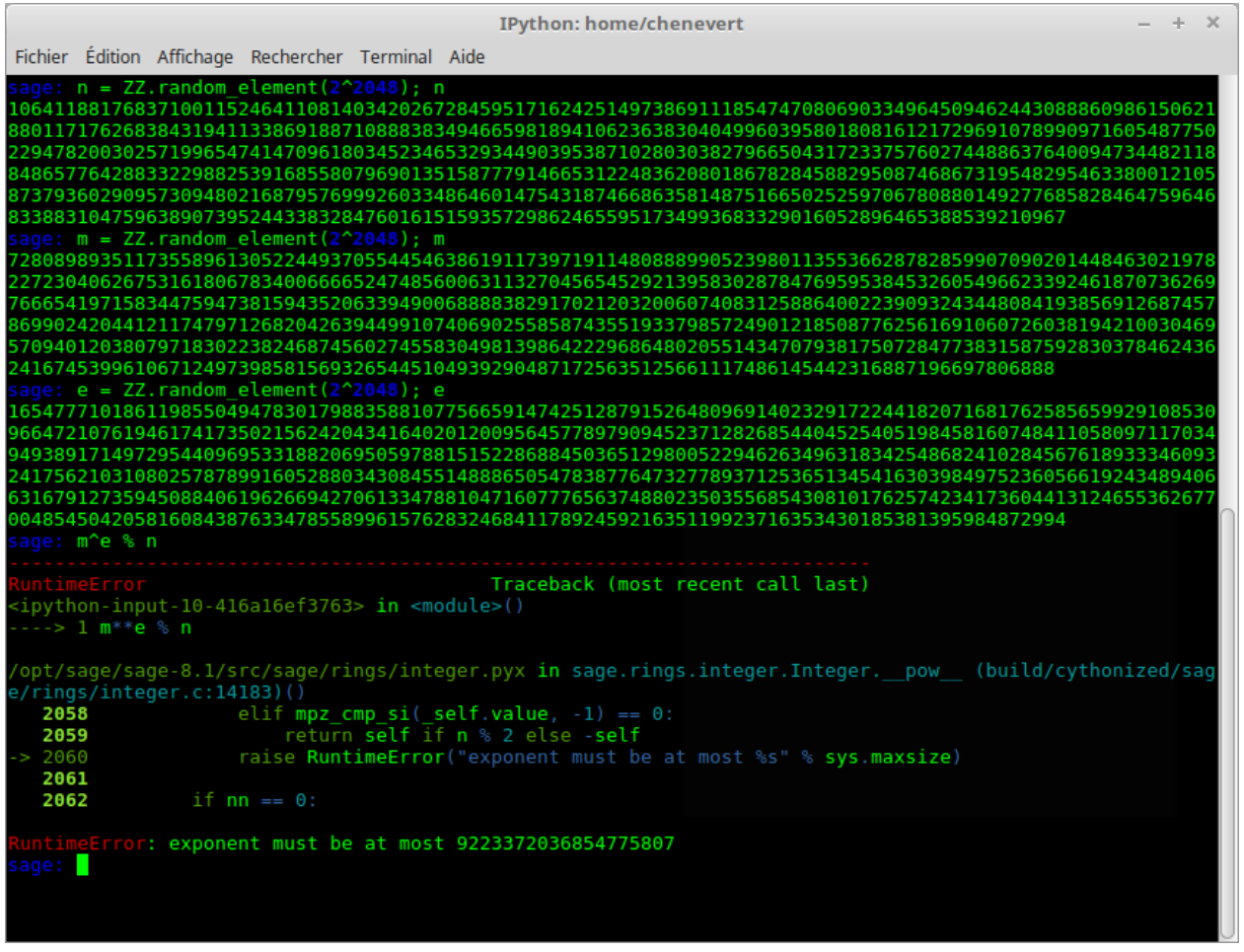
$$de \underset{\phi(n)}{\equiv} 1$$

où $\phi(n)$ est le nombre d'entiers entre 1 et n sans facteur commun avec n (*indicatrice d'Euler*)

Cryptographie

Exponentiation modulaire

Algorithme naïf d'exponentiation modulaire ($m^e \,\%\, n$) : dim $= \infty$!



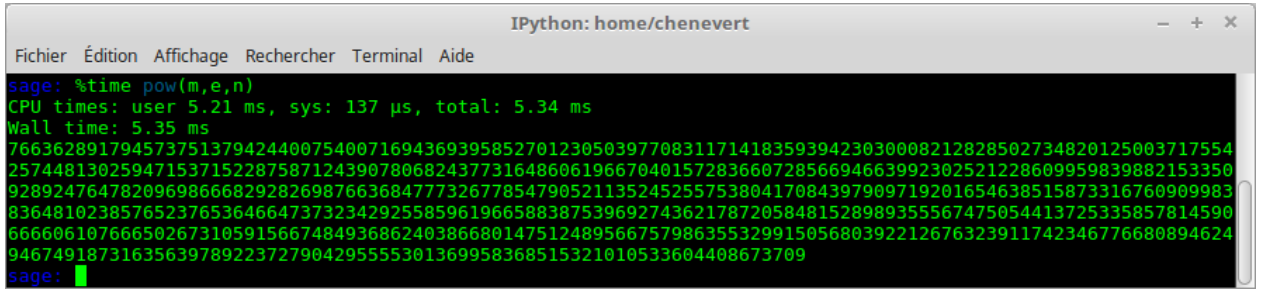
Cryptographie

Exponentiation rapide

Mais :

- pour la multiplication d'entiers de ℓ chiffres,
  - algorithme naïf : dimension 2
  - algorithme de Karatsuba : dimension 1,585
  - algorithme de Schönage-Strassen / Fürer : dimension 1

- l'exponentiation modulaire peut être réalisée à l'aide de $\log_2(e)$ mises au carré et multiplications modulaires.



Cryptographie

Attaquer RSA

Connaissant d ou e, on obtient l'autre comme coefficient de Bézout à l'aide de l'algorithme d'Euclide :

$$de + k\phi(n) = 1.$$

On a donc intérêt à ce que le calcul de $\phi(n)$ soit **long**, pour cela on prend $n = pq$ avec p et q deux grands nombres premiers.

Meilleure attaque à ce jour : *factoriser* n puis en déduire

$$\phi(n) = (p-1)(q-1).$$

Cryptographie

Notation L

Pour $d \in \mathbf{R}$ et $\alpha \in [0, 1]$,

$$L_\alpha(n) := \exp\left(d\,(\log n)^\alpha\,(\log\log n)^{1-\alpha}\right).$$

- $L_0(n) = (\log n)^d$
- $L_1(n) = n^d$
- en général $L_\alpha(n)$ est en quelque part entre les deux.

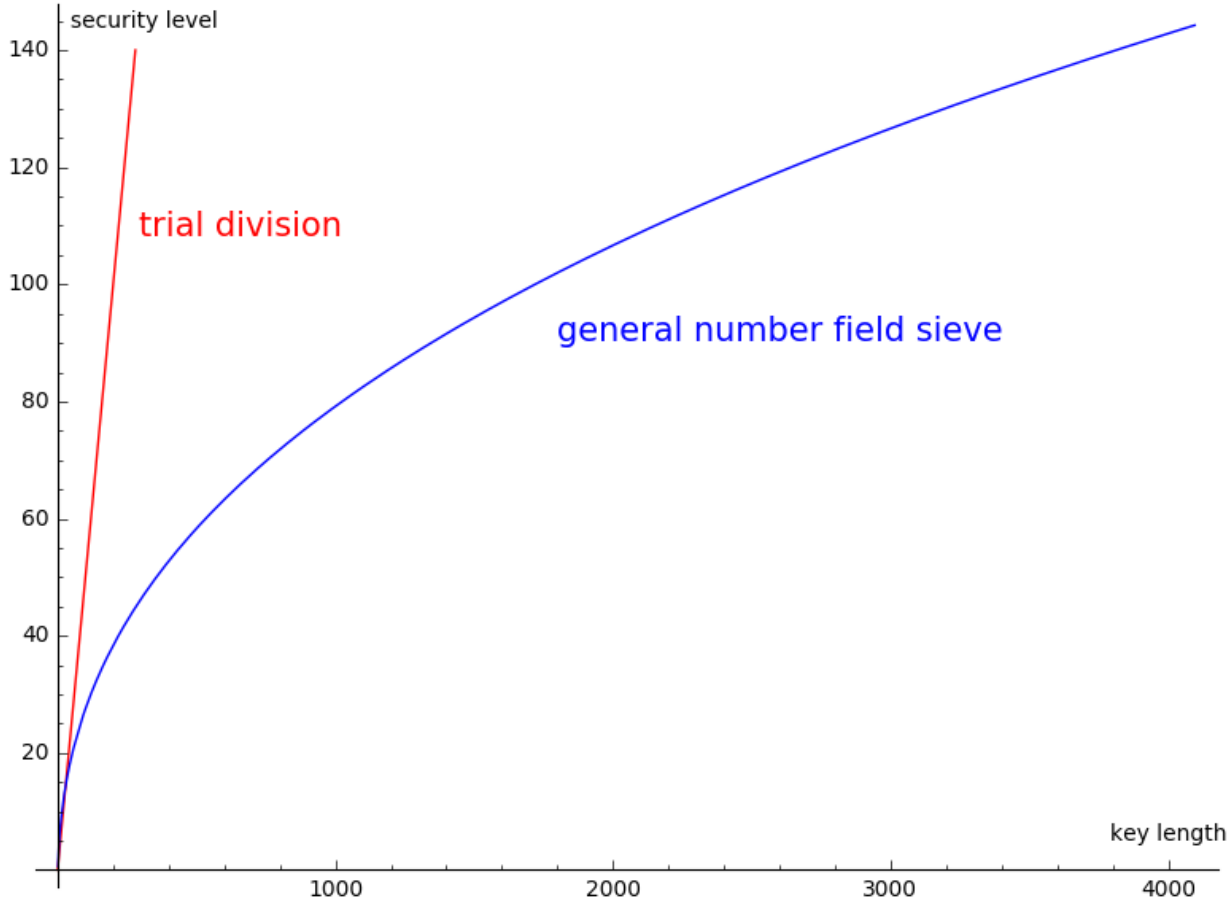
Cryptographie

Factorisation

- Recherche exhaustive de facteur : L_1 avec $d = \frac{1}{2}$
- Crible quadratique : $L_{\frac{1}{2}}$ avec $d = 1$
- Crible du corps de nombres généralisé : $L_{\frac{1}{3}}$ avec $d \approx 1{,}923$.

Cryptographie

Crible du corps de classe généralisé



Cryptographie

Logarithme discret

Autre façon d'utiliser l'exponentiation modulaire : $x \underset{n}{\equiv} g^y$

Problème du logarithme discret : retrouver $y = \ll \log_g(x) \gg$

Même complexité que la factorisation dans le cas arithmétique (DSA, Diffie-Hellman)

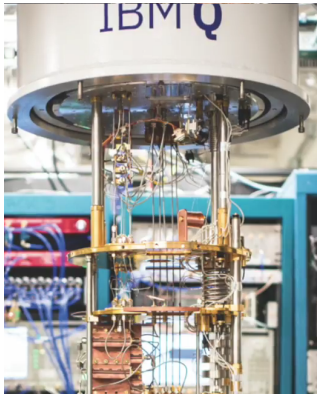
Mais : $\sqrt{n}$ dans un groupe quelconque

$\Longrightarrow$ utilisation de courbes elliptiques (ECDSA, ECDH)

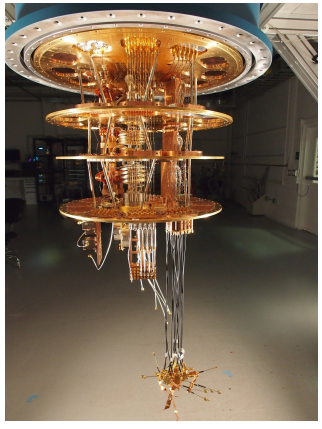
Menace quantique

État de l'art

Recuit simulé



IBM : 50 qubits



Google : 70 qubits

Menace quantique

Algorithme de Grover

Menace quantique

Algorithme de Grover

Étant donné une fonction « quelconque » $f : A \to B$ avec $|A| = n$ et $b \in f(A)$, la recherche d'une préimage $a \in A$ telle que $f(a) = b$ prend classiquement n évaluations.

Un ordinateur quantique pourrait en trouver une (*probablement*) avec seulement $\sqrt{n}$ évaluations de f.

$\Longrightarrow$ doubler la taille des clés pour le chiffrement symétrique

Menace quantique

Algorithme de Shor

Factorise un entier n en nombre d'opérations

$$(\log n)^2(\log\log n)(\log\log\log n)$$

$\Longrightarrow$ $\log \dim = 2$!

Basé sur la capacité d'un ordinateur quantique à découvrir facilement la période d'une fonction périodique.

Résoud aussi le problème du logarithme discret.

Menace quantique

Algorithme de Shor

~~RSA~~ ~~(EC)DSA~~ ~~(EC)DH~~

Chiffrement post-quantique

Il est nécessaire de réfléchir dès maintenant à des alternatives asymétriques efficaces et sécurisées, même face à un éventuel attaquant quantique.

4 pistes principales à ce jour :

- à base de fonctions de hachage
- à base de codes correcteurs
- à base de réseaux
- équations quadratiques à plusieurs variables.

Processus de standardisation en cours par le NIST.

Chiffrement post-quantique

Chiffrement à base de réseaux

On se donne un réseau $\Lambda = \{ \sum_i a_i \mathbf{v}_i \mid a_i \in \mathbf{Z} \}$ où $\mathbf{v}_1, \ldots, \mathbf{v}_n$ sont des vecteurs linéairement indépendants dans $\mathbf{R}^n$.

Chiffrement : on ajoute à un message $\mathbf{v} \in \Lambda$ un petit bruit $\mathbf{e} \in \mathbf{R}^n$:

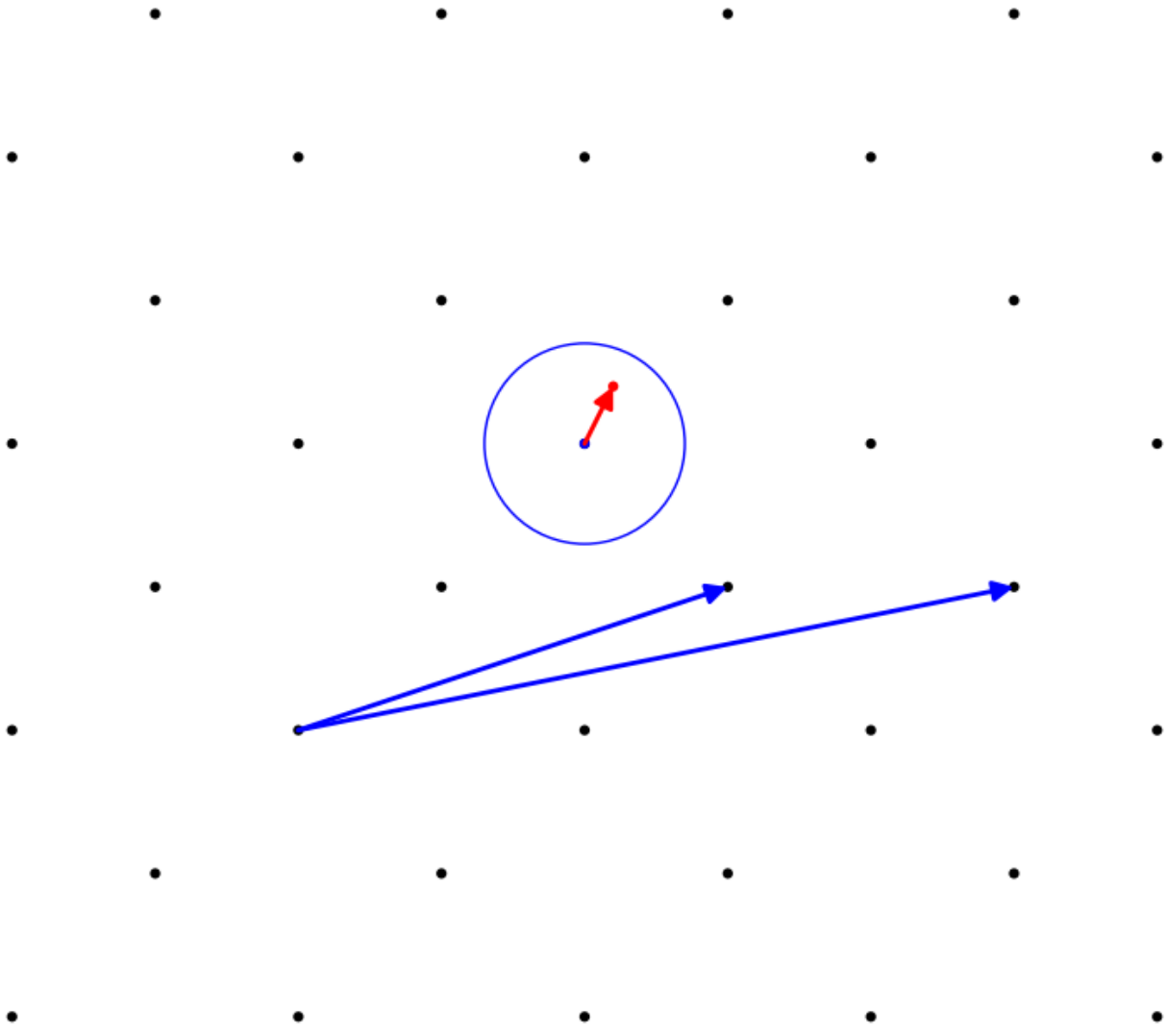
$$\mathbf{c} = \mathbf{v} + \mathbf{e}.$$

Déchiffrement : étant donné $\mathbf{c}$, on déchiffre au vecteur $\mathbf{v}$ du réseau le plus proche.

Problème algorithmiquement difficile sans connaître une base adaptée.

Chiffrement post-quantique

Chiffrement à base de réseaux



Merci

et bon congrès !