L'algorithme LLL : Attaquer un cryptosystème à l’aide de l'algèbre

Problématique : Comment cet algorithme peut-il être performant en cryptanalyse?

I-Algorithme LLL (1982)

A-Objectif

Objectif :

Rechercher le plus court vecteur d'un réseau euclidien en réduisant une base

Réseau euclidien :

$Soit\ (b_1, \dots, b_n)\ une\ base\ de\ L$

$$L = \left\{ \sum_{i=1}^{n} x_i b_i\ , (x_1, \dots, x_n) \in \mathbb{Z}^n \right\}$$

Preuve qu'il existe un plus court vecteur :

$Soit\ E = \{x \in L, \langle x|x \rangle < M, M \in \mathbb{R}\}\ ensemble\ des\ vecteurs\ courts\ de\ L$

Chaque coordonnée de $x \in E$ est bornée

E est donc un ensemble fini

Ainsi, il existe un minimum pour E

CONCLUSION : un plus court vecteur existe

I-Algorithme LLL

B-Procédé

Orthogonalisation de Gram-Schmidt :

Transformation de $B = (b_1, \dots, b_n)$ en $B^* = (b_1^*, \dots, b_n^*)$ orthogonale

Conditions d'arrêt :

- Condition de taille $\forall j > i, \quad |\tilde{\mu_{i,j}}| \leq \frac{1}{2}$
- Condition de Lovász $\forall j \in [\![1, n-1]\!], \delta \left\| b_j^* \right\|^2 \leq \left\| \widetilde{\mu_{j,j+1}} b_j^* + b_{j+1}^* \right\|^2$

Réduction faible :

pour j allant de 2 à n

pour i allant de j-1 à 1

$$b_j \leftarrow b_j - \lfloor \mu_{ij} \rceil b_i$$

Traduction matricielle :

$$U = \begin{array}{c} \begin{array}{ccccc} \overbrace{\quad}^{b_1} & & \cdots & & \overbrace{\quad}^{b_n} \end{array} \\ \begin{pmatrix} 1 & & & & \\ 0 & \ddots & & (\mu_{i,j}) & \\ \vdots & & \ddots & & \\ \vdots & & & \ddots & \\ 0 & \cdots & \cdots & 0 & 1 \end{pmatrix} \end{array} \begin{array}{l} \backslash\, b_1^* \\ \\ \vdots \\ \\ \backslash\, b_n^* \end{array} \qquad \tilde{U} = \begin{array}{c} \begin{array}{ccccc} \overbrace{\quad}^{\tilde{b_1}} & & \cdots & & \overbrace{\quad}^{\tilde{b_n}} \end{array} \\ \begin{pmatrix} 1 & & & & \\ 0 & \ddots & & (\tilde{\mu_{i,j}}) & \\ \vdots & & \ddots & & \\ \vdots & & & \ddots & \\ 0 & \cdots & \cdots & 0 & 1 \end{pmatrix} \end{array} \begin{array}{l} \backslash\, b_1^* \\ \\ \vdots \\ \\ \backslash\, b_n^* \end{array}$$

$$vect < (b_1^*, \dots, b_n^*) > = vect < (\widetilde{b_1^*}, \dots, \widetilde{b_n^*}) >$$

I-Algorithme LLL

B-Procédé

Terminaison de l'algorithme :

Condition de taille respectée intrinsèquement

Condition de Lovász atteignable

- $F = vect < b_1^*, \dots, b_{j-1}^* > \; \Rightarrow F^\perp = vect < b_j^*, \dots, b_n^* >$
- $b_j^* = id(b_j) - p_F(b_j)$
- $p_{F^\perp}(b_{j+1}) = b_{j+1}^* + \mu_{j,j+1} b_j^*$
- $\delta \|p_{F^\perp}(b_j)\|^2 \leq \|p_{F^\perp}(b_{j+1})\|^2$
- Ainsi $\forall j \in [\![1, n-1]\!], \delta \|b_j^*\|^2 \leq \|\widetilde{\mu_{j,j+1}} b_j^* + b_{j+1}^*\|^2$

Algorithme LLL :

Entrée : $B = (b_1, \dots, b_n)$ *une base quelconque du réseau*

Réduire faiblement B

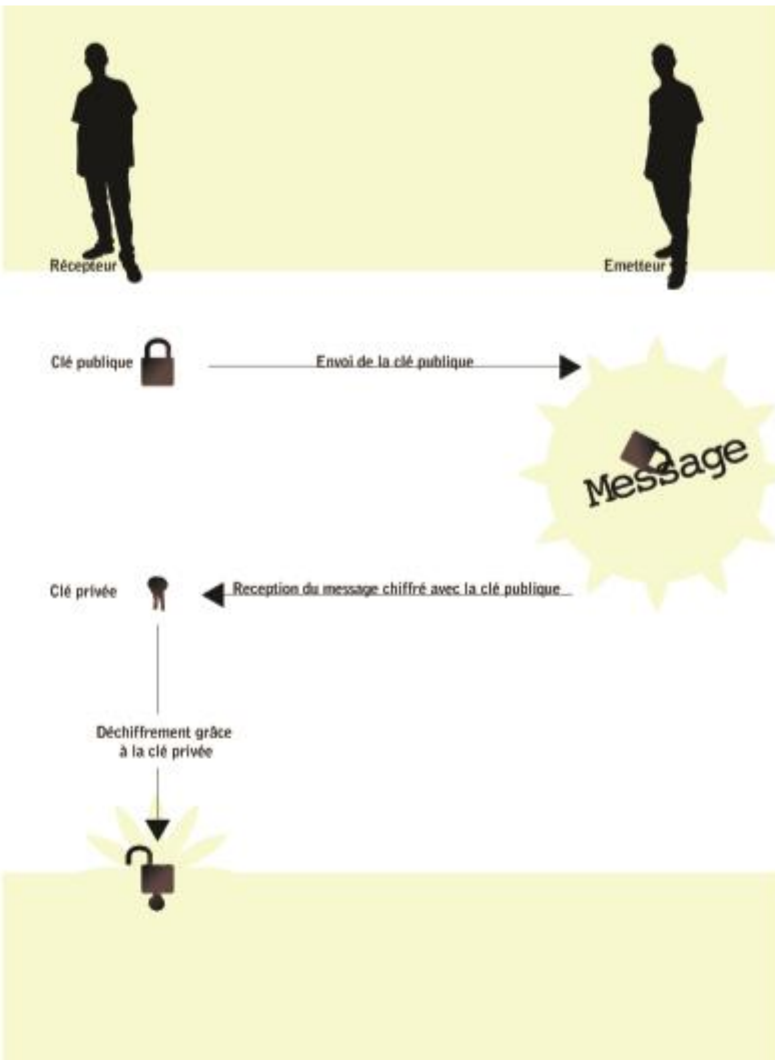
Vérifier que tous les $\tilde{b_j}$ créés respectent la condition de Lovász

Sinon inverser $\tilde{b_j}$ et b_{j+1} et recommencer

Sortie : $\tilde{\mathcal{B}} = (\tilde{b_1}, \dots, \tilde{b_n})$ *une base dite LLL − réduite*

II-Application à la cryptographie

A- Cryptosystème de Merkle-Hellman (1978)



Principe d'un cryptosysteme à clé publique

Clé privée :

- $(a_1, \dots, a_n)$ *une suite supercroissante* (sac à dos)

$$\forall i \in [\![2, n]\!], a_i = \sum_{j=1}^{i-1} a_j$$

- $m \in \mathbb{N}, m > \sum_{i=1}^{n} a_i$
- $w \in [1, m-1], pgcd(m, w) = 1$

Clé publique :

$(b_1, \dots, b_n)$ *telle que* $\forall i \in [\![1, n]\!], b_i \equiv a_i w [m]$

Codage du message :

- Message en clair : $M = (m_1, \dots, m_n)$
  $avec\ \forall i \in [\![1, n]\!], m_i \in \{0,1\}$
- Message codé : $c = \sum_{i=1}^{n} m_i b_i$

« Le problème du sac à dos en cryptographie », Julien Deroff et Rémi Goyat (2007)

II-Application à la cryptographie

B- Réseau de Lagarias-Odlyzko (1985) – Attaque 1

<u>Principe :</u>

- On pose $$G1 = \begin{array}{cc} \overset{n+1}{\longleftrightarrow} & \\ \begin{pmatrix} 1 & & 0 & 0 \\ & \ddots & & \vdots \\ 0 & & 1 & 0 \\ Nb_1 & \cdots & Nb_n & -Nc \end{pmatrix} & \updownarrow \text{n+1} \end{array}$$ $$avec\ N > \sqrt{n}$$ $$Nc = N \sum_{i=1}^{n} m_i b_i$$

- Réduire G1 grâce à l'algorithme LLL

- Déterminer le vecteur colonne $g_j = \begin{pmatrix} g_{1,j} \\ \vdots \\ g_{n+1,j} \end{pmatrix}$ $avec\ \forall i \in [\![1, n]\!], g_{i,j} \in \{0,1\}$, $g_{n+1,j} = 0$

- g_j : bon candidat

<u>Preuve :</u>

$$v = \begin{pmatrix} m_1 \\ \vdots \\ m_n \\ 0 \end{pmatrix} = \sum_{i=1}^{n} m_i g_i + g_{n+1}$$ et $$\|v\| \leq \sqrt{n} \leq \sqrt{n}(Nc)^{1/n}$$

II-Application à la cryptographie

C- Réseau de Coster, La Macchia, Odlyzko et Schnorr (1992) – Attaque 2

Principe :

- On pose $$G2 = \begin{array}{c} \overset{n+1}{\longleftrightarrow} \\ \begin{pmatrix} 1 & & 0 & {}^1/_2 \\ & \ddots & & \vdots \\ 0 & & 1 & {}^1/_2 \\ Nb_1 & \cdots & Nb_n & Nc \end{pmatrix} \end{array} \Big\updownarrow n+1$$ $avec\ N > \frac{\sqrt{n}}{2}$
$$Nc = N \sum_{i=1}^{n} m_i b_i$$
- Réduire G2 grâce à l'algorithme LLL
- Déterminer le vecteur colonne $g_j = \begin{pmatrix} g_{1,j} \\ \vdots \\ g_{n+1,j} \end{pmatrix}$ $avec\ \forall i \in [\![1, n]\!], g_{i,j} \in \left\{-\frac{1}{2}, \frac{1}{2}\right\}$
$$g_{n+1,j} = 0$$
- $g_j + \begin{pmatrix} 1/2 \\ \vdots \\ 1/2 \end{pmatrix}$ et $g_j - \begin{pmatrix} 1/2 \\ \vdots \\ 1/2 \end{pmatrix}$: bons candidats

II-Application à la cryptographie

D- Réseau de Joux et Stern (1991) – Attaque 3

Principe :

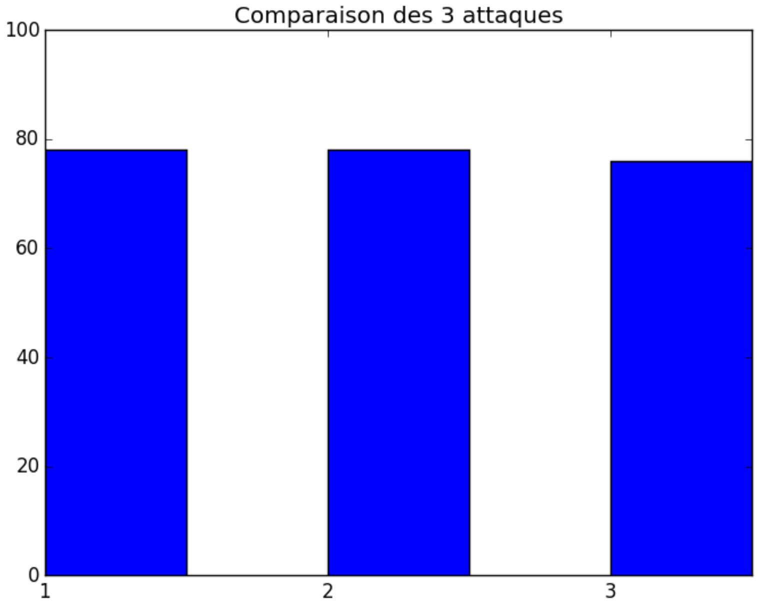
- On pose $G3 = \begin{array}{cc} \text{n+1} & \\ \begin{pmatrix} n+1 & & -1 \\ & \ddots & \\ -1 & & n+1 \\ Nb_1 & \cdots \quad Nb_n & -Nc \end{pmatrix} & \text{n+2} \end{array}$ $\quad avec\ N > n$

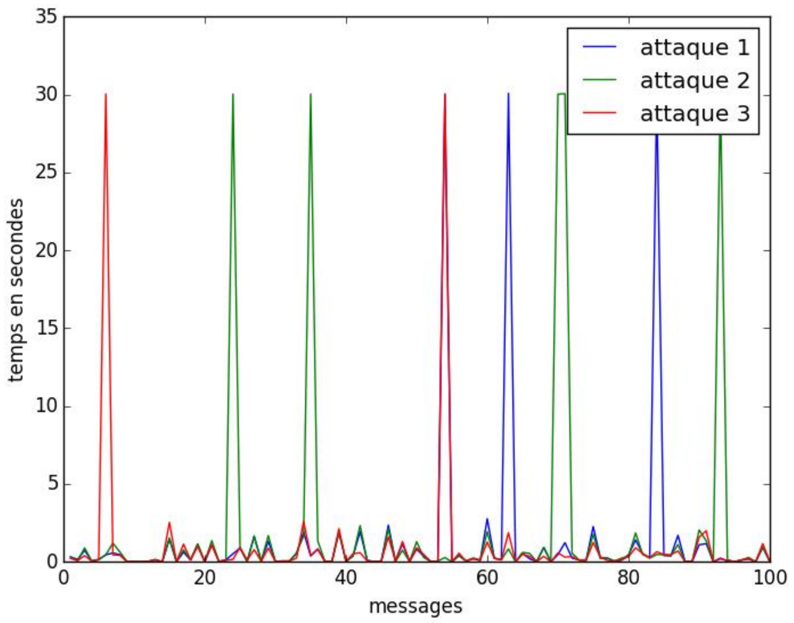
$$Nc = N \sum_{i=1}^{n} m_i b_i$$

- Réduire G3 grâce à l'algorithme LLL
- Déterminer le vecteur colonne $g_j = \begin{pmatrix} g_{1,j} \\ \vdots \\ g_{n+1,j} \end{pmatrix}$ $avec \left| \begin{array}{l} \forall i \in [\![1, n]\!], g_{i,j} \in \{x, y\}, \\ g_{n+1,j} = 0 \end{array} \right. \left| \begin{array}{c} (x, y) \in \mathbb{Z}_+^* \times \mathbb{Z}_-^* \\ x - y = n + 2 \end{array} \right.$
- $\frac{1}{x-y}\left[g_j - \begin{pmatrix} y \\ \vdots \\ y \end{pmatrix}\right]$ et $\frac{1}{x-y}\left[g_j - \begin{pmatrix} x \\ \vdots \\ x \end{pmatrix}\right]$: bons candidats

III-Expériences

A-Comparaison avec des mesures aléatoires



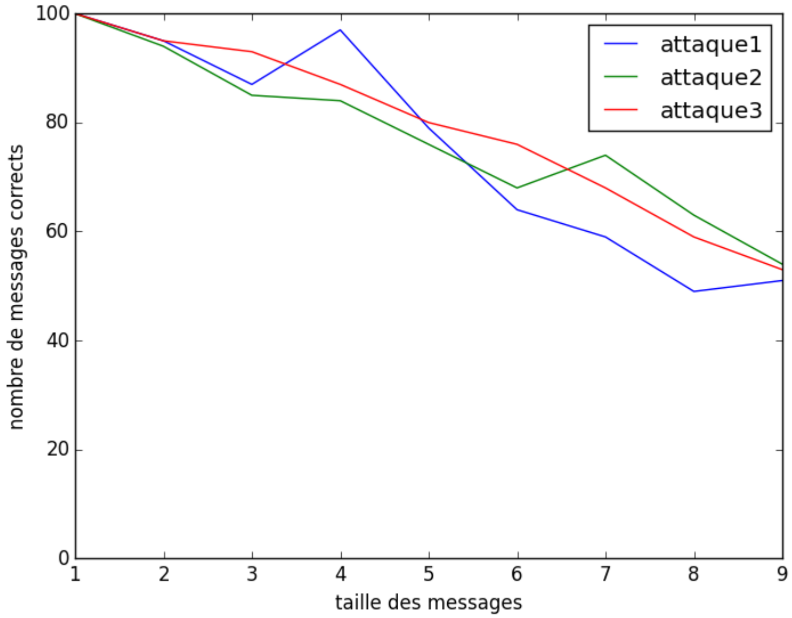


Nombre de messages décryptés

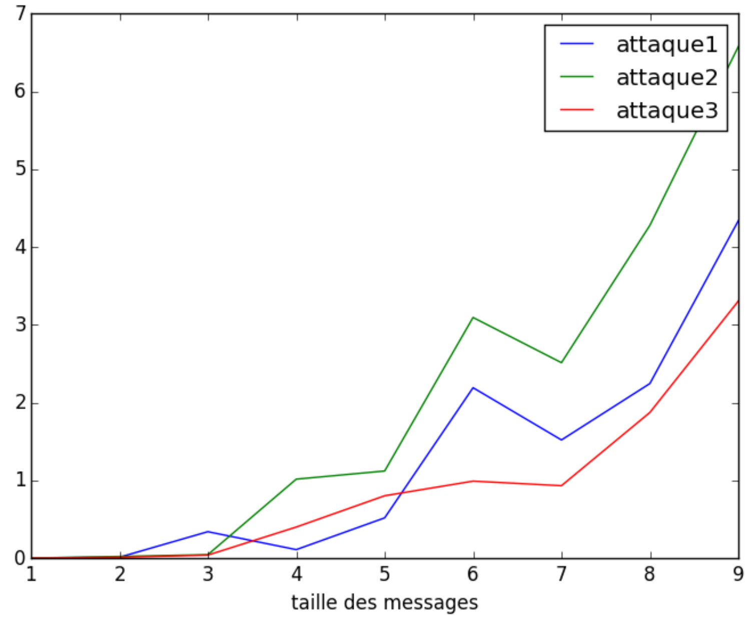
Temps de calcul

III-Expériences

B-Comparaison avec des messages de tailles fixes



Nombre de messages correctement retranscris selon la taille du message



Temps de calcul moyen selon la taille du message

III-Expériences

C- Comparaison en fixant la densité

Densité d'un sac à dos :

Par définition : $d = \frac{n}{\max\limits_{i \in [\![1,n]\!]} \log_2 a_i}$ $pour\ un\ sac$ à $dos\ (a_1, \dots, a_n)$

Théorie :

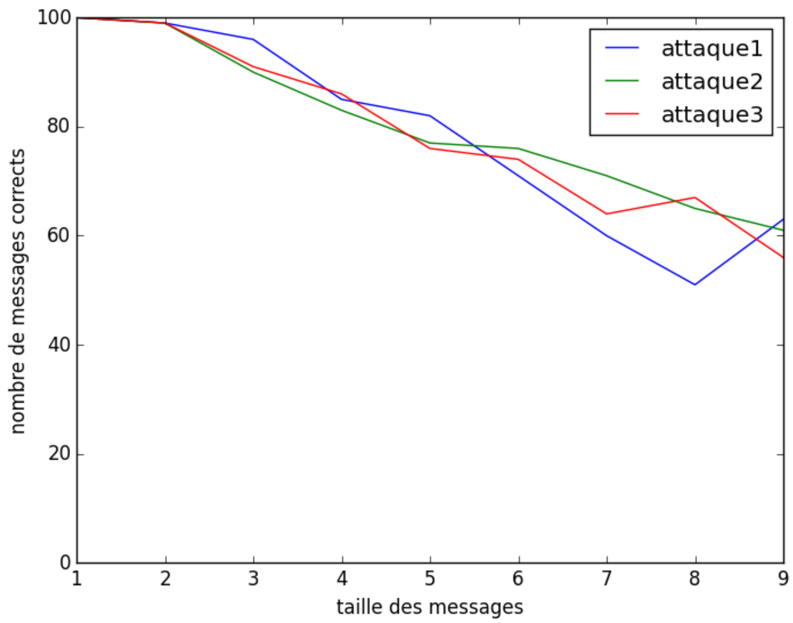
$$Si\ d \leq 1 \Rightarrow\ solution\ unique$$

$$Si\ d \leq 0{,}94 \Rightarrow\ le\ réseau\ de\ Joux\ et\ Stern\ est\ le\ plus\ performant$$
$$Si\ d \leq 0{,}64 \Rightarrow\ le\ réseau\ de\ Lagarias\ et\ Odlyzko\ est\ performant$$

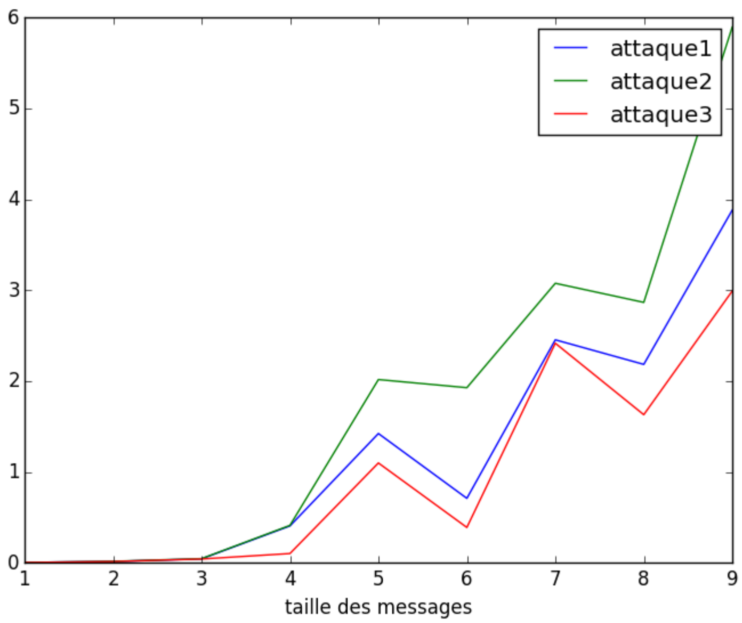
III-Expériences

C- Comparaison en fixant la densité

Cas où $d \leq 0{,}64$:



Nombre de messages correctement retranscris selon la taille du message

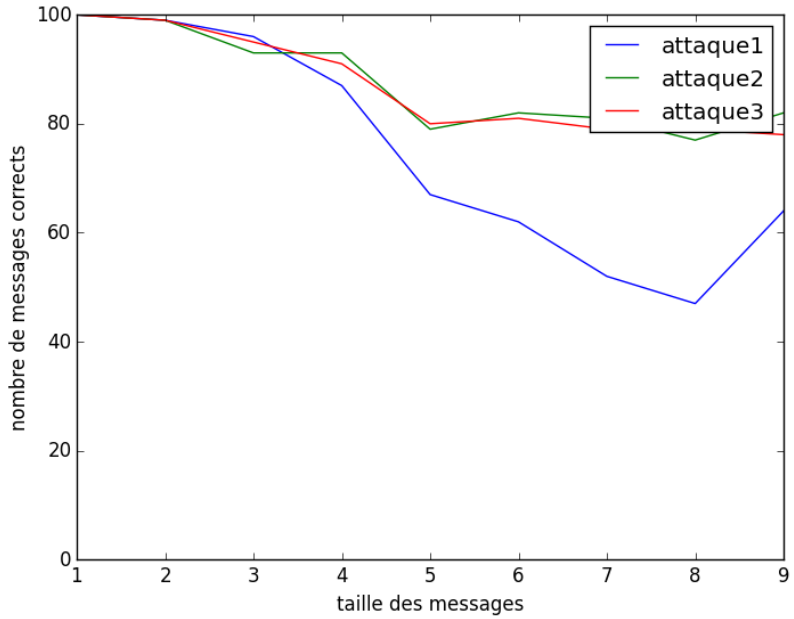


Temps de calcul moyen selon la taille du message

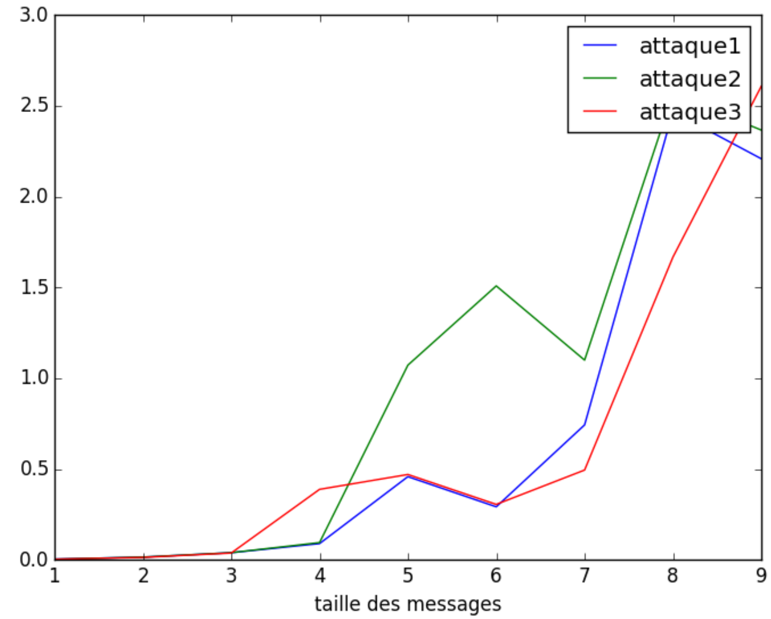
III-Expériences

C- Comparaison en fixant la densité

Cas où $0{,}64 \leq d \leq 0{,}94$:



Nombre de messages correctement retranscris selon la taille du message

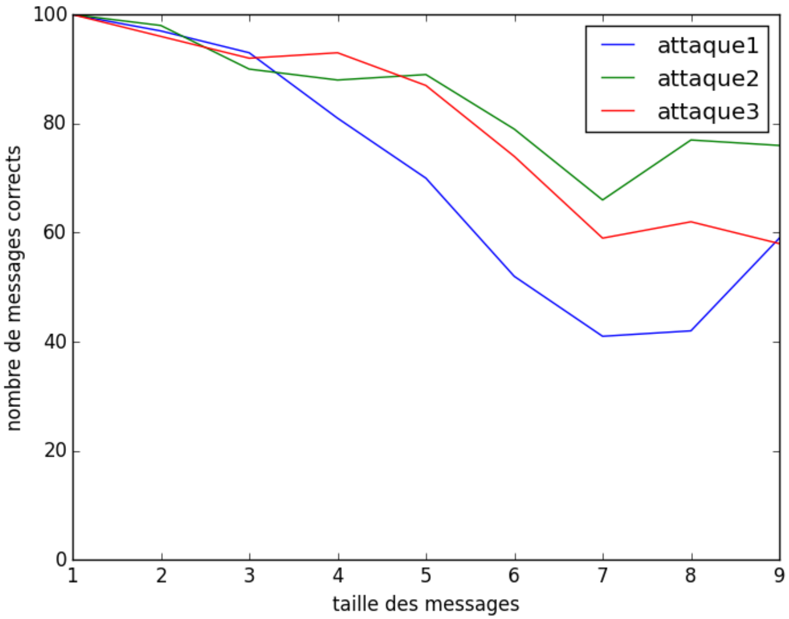


Temps de calcul moyen selon la taille du message

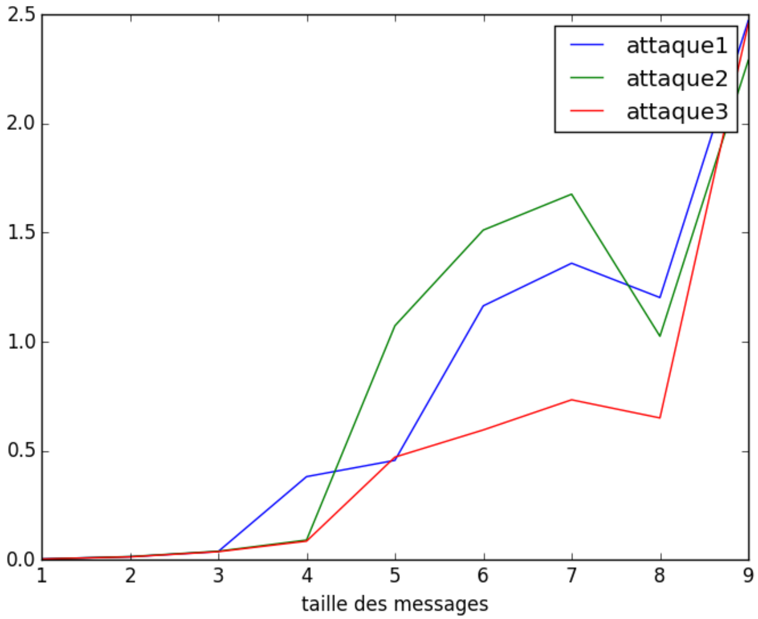
III-Expériences

C- Comparaison en fixant la densité

Cas où $0{,}94 \leq d \leq 1$:



Nombre de messages correctement retranscris selon la taille du message



Temps de calcul moyen selon la taille du message