Cryptosystèmes, algorithmes et longueur de clés

Daniel.Augot@inria.fr,Anne.Canteaut@inria.fr,Morain@lix.polytechnique.fr

13 octobre 1999

Contenu de l'exposé

1. Rappel clé secrète, clé publique.
2. DES.
   (a) Historique et paramètres.
   (b) Description.
   (c) Attaques connues.
3. RSA
   (a) Description
   (b) Attaques
   (c) Coût de la factorisation.

4. Elliptic Curve cryptosystems
   (a) Le groupe des points d'une courbe elliptique.
   (b) Le logarithme discret et Diffie-Hellmann.
   (c) Attaques.
5. Trois records
   (a) descracker
   (b) RSA-155.
   (c) ECC2-97.

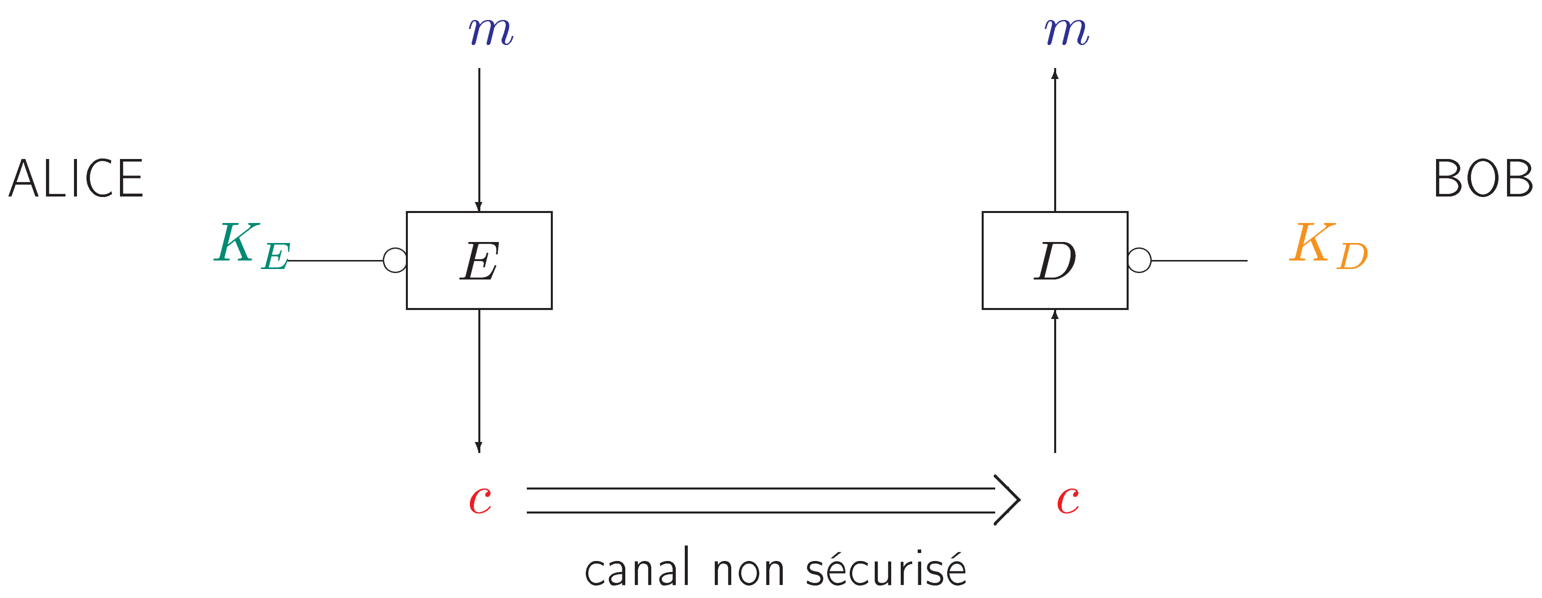
Hors sujet : trappes cachées, ordinateurs quantiques etc.

Seulement des faits établis et publiés.

Pas de prospective.

I. Clé publique, clé secrète, attaques

Chiffrement



Clef secrète / Clef publique

Chiffrement à clef secrète :

Les deux utilisateurs partagent la même clef.

$$\xrightarrow{E_{K_{AB}}(m)}$$

K_{AB} K_{AB}

$$D_{K_{AB}} \circ E_{K_{AB}} = Id$$

Chiffrement à clef publique :

Chaque utilisateur possède une clef publique et une clef secrète.

$$\xrightarrow{E_{P_B}(m)}$$

P_B S_B

$$D_{S_B} \circ E_{P_B} = Id$$

Propriétés requises

Les fonctions E et D sont toujours supposées connues. La sécurité repose uniquement sur le secret de la clef.

- **Attaque à chiffré seul** L'attaquant connaît le chiffré c.

  Il veut retrouver le texte clair m (ou mieux, la clef K_D).

- **Attaque à clair connu** L'attaquant connaît certains couples clairs/chiffrés.

  Il veut retrouver la clef K_D.

- **Attaque à clair choisi** L'attaquant connaît certains couples clairs/chiffrés correspondant à des messages clairs de son choix. Il veut retrouver la clef K_D.

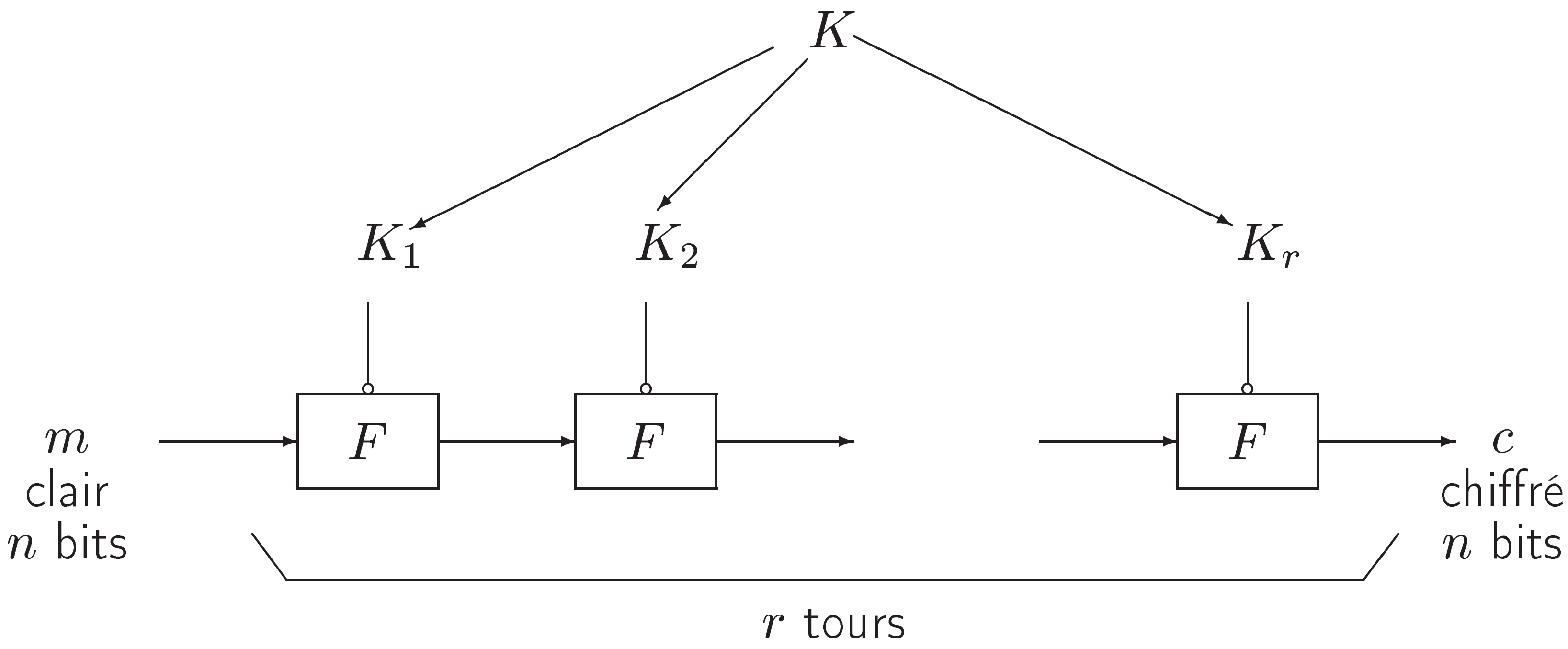
En clé publique, clair connu et clair choisi sont gratuits.

II. Data Encryption Standard

DES

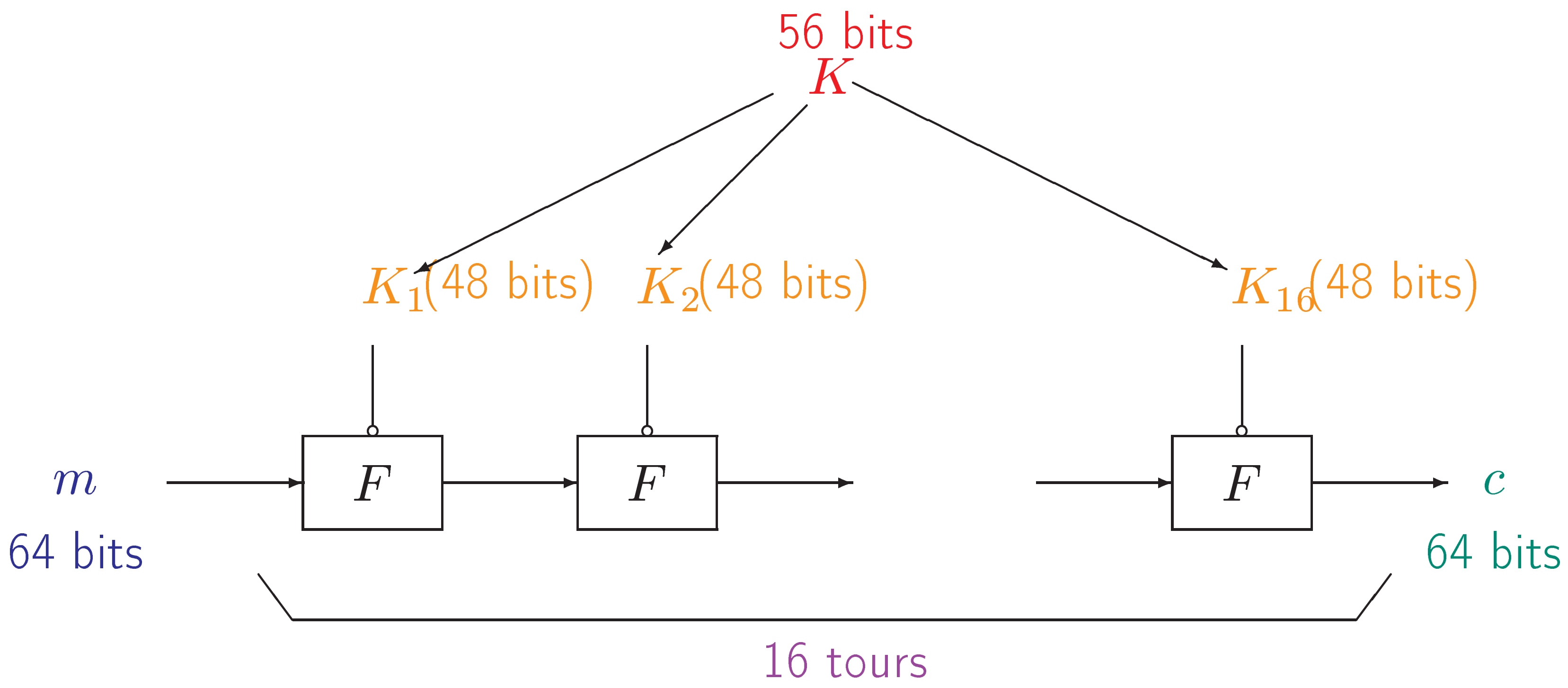
- Standard du NIST en 1977.
- Conçu par IBM (lucifer), retouché par la NSA.
- N'a succombé à aucunes attaques autres que la recherche exhaustive.
- Paramètres :
  - chiffrement par bloc.
  - clair et chiffré de 64 bits.
  - clé de 64 bits, dont seulement 56 sont utilisés.
- Recherche exhaustive en 2^{55} en moyenne.

Chiffrements itératifs par blocs

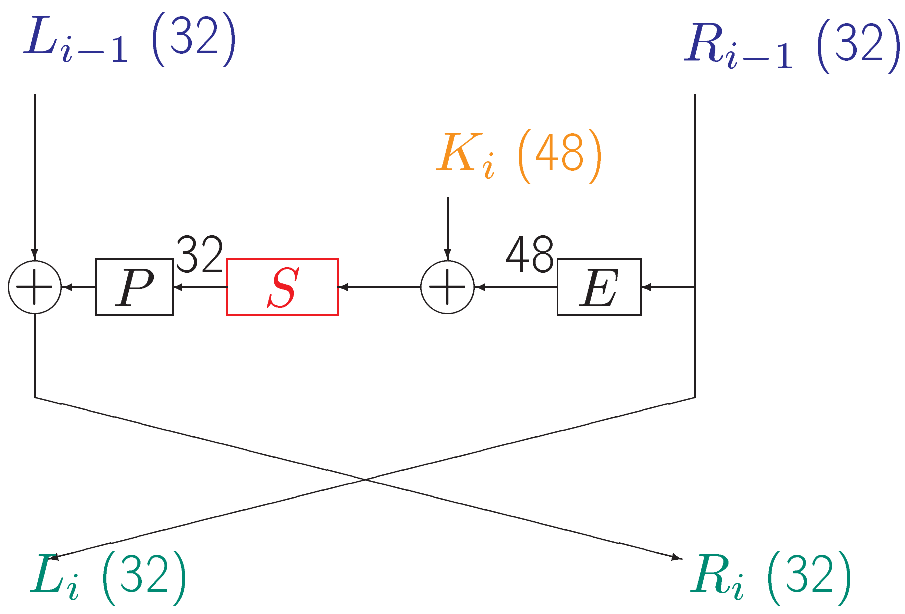


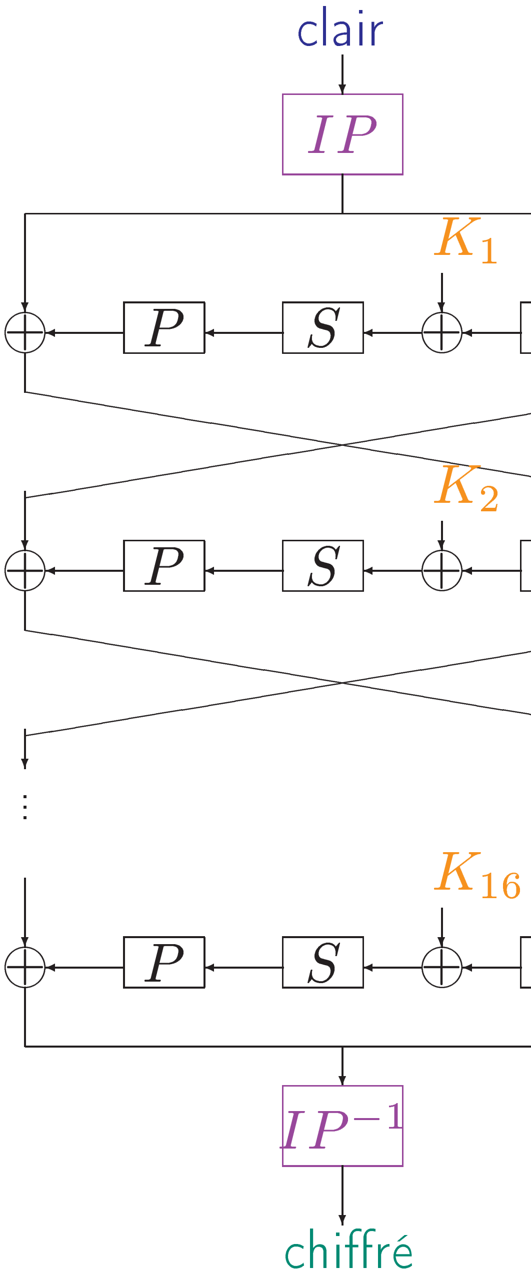
où F_K est une permutation sur les mots de n bits

Schéma du DES

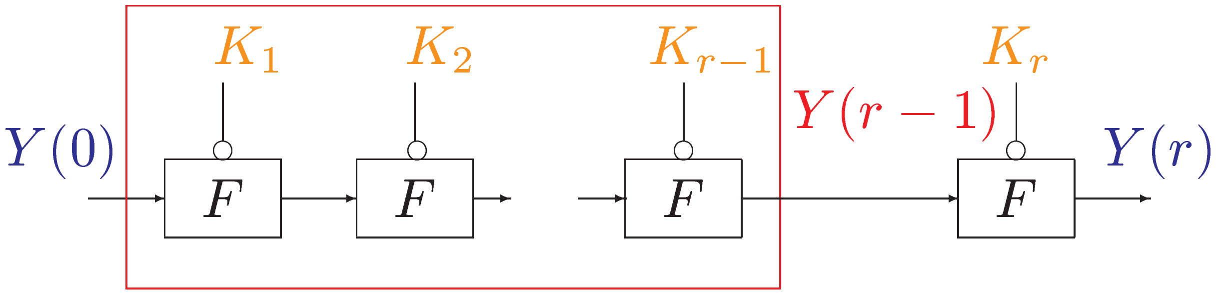


Fonction itérée du DES




clair
IP
K_1
P
S
E
K_2
P
S
E
K_{16}
P
S
E
IP^{-1}
chiffré

Cryptanalyses sur la dernière clé



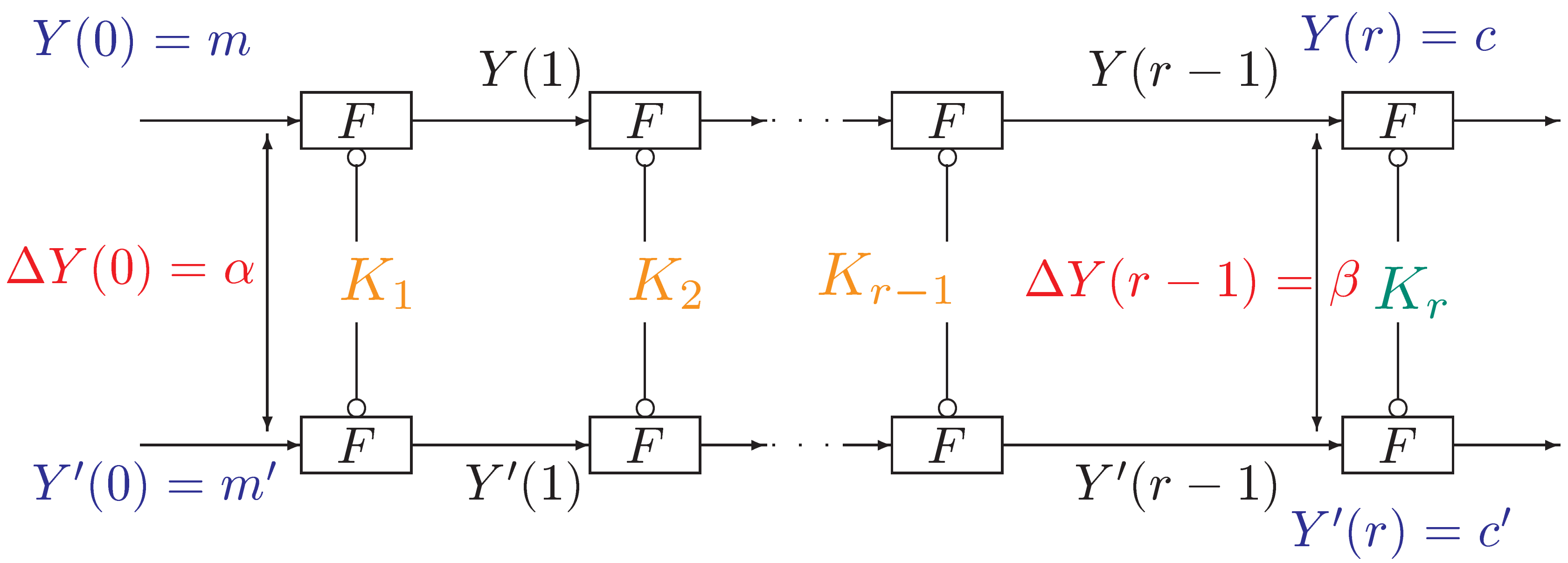
Biais statistique dans la distribution de $(Y(0), Y(r-1))$

On calcule $Y(r-1) = F^{-1}(Y(r), K_r)$ en faisant varier K_r

comparaison

valeur de K_r

Cryptanalyse différentielle



Succés

`FEAL` Japon 87-88. Blocs de 64 bits, clés de 128 bits.

Cryptanalyse différentielle : `FEAL-4` : 8 couples clairs/chiffrés. `FEAL-16` : ≤ 2000 couples clairs/chiffrés.

`LUCIFER` Ancêtre du DES. Bloc, clé 64 bits.

Cryptanalyse différentielle sur 8 tours : 60 couples couples clairs/chiffrés.

`KHUFU` Merkle 90. blocs 64 bits, 128 bits de clé.

cryptanalyse linéaire : 2^{43} clairs/chiffrés.

Résistance du DES

Pour le DES :

Cryptanalyse différentielle :

Nombre de couples clairs-chiffrés (choisis) $= 2^{47}$

Cryptanalyse linéaire :

Nombre de couples clairs-chiffrés (connus) $= 2^{43}$

Si on remplace les boîtes $\mathbb{S}$ par d'autres, ca s'effondre.

Autres systèmes de chiffrement à clef secrète

- **Triple DES à 2 clefs :** taille des blocs : 64 bits, taille de la clef : 112 bits.

$$c = E_{K_1}(D_{K_2}(E_{K_1}(m))) \qquad m = D_{K_1}(E_{K_2}(D_{K_1}(c)))$$

- **IDEA** : développé par Lai et Massey en 1990.
  taille des blocs : 64 bits, taille de la clef : 128 bits.

- **AES** (Advanced Encryption Standard) (1997-2001)
  processus de standardisation d'un nouvel algorithme de chiffrement à clef secrète
  (actuellement 15 candidats)
  taille des blocs : 128 bits, taille de la clef : 128 / 192 / 256 bits.

III. RSA

Calcul de racines modulo pq

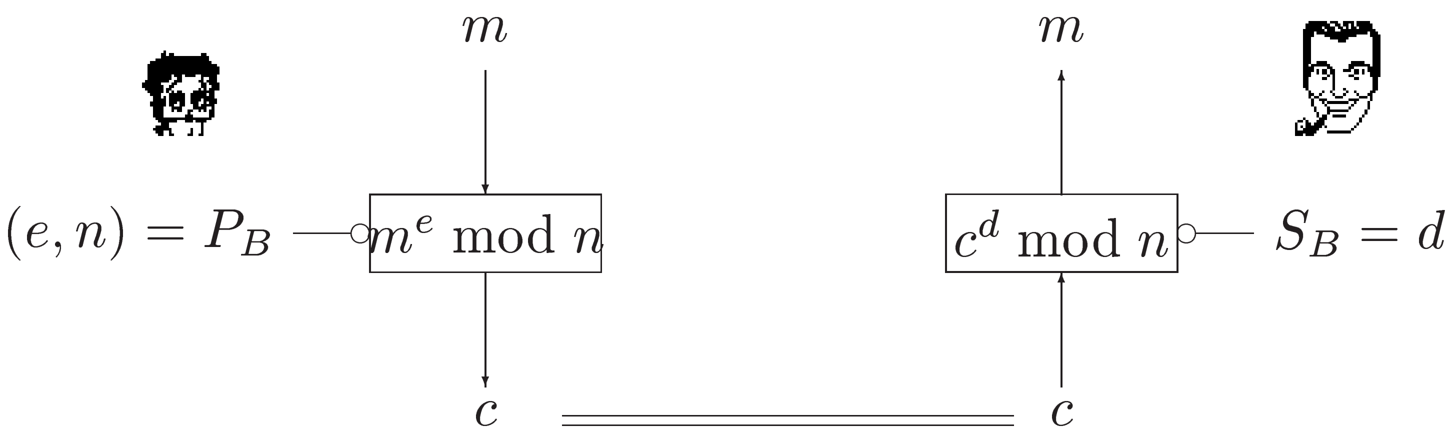
Soit $n = pq$ où p et q sont deux entiers premiers.

Soit $e \in \mathbf{Z}/n\mathbf{Z}$ tel que $\mathrm{pgcd}(e, (p-1)(q-1)) = 1$, et $y \in (\mathbf{Z}/n\mathbf{Z})^*$.

Trouver $x \in (\mathbf{Z}/n\mathbf{Z})^*$ tel que $x^e \bmod n = y$.

- **Quand on connaît p et q :** facile.
- **Quand on ne connaît pas p et q :** difficile.

Le chiffrement à clef publique RSA [Rivest - Shamir - Adleman 78]



Bob choisit deux grands nombres premiers p et q et un entier e premier avec $(p-1)(q-1)$.

Il calcule d tel que $ed \bmod (p-1)(q-1) = 1$.

RSA : précautions et théorie

Quelques précautions :

- Il faut que p et q soient à peu près de même taille.
- Il faut que $(p-1)$ et $(q-1)$ aient un grand facteur premier, par exemple $\frac{p-1}{2}$ et $\frac{q-1}{2}$ premiers.
- et beaucoup d'autres. . .

Remarques théoriques :

- Du point de vue théorique, il n'est pas prouvé qu'il n'existe pas d'algorithme polynomial pour factoriser.

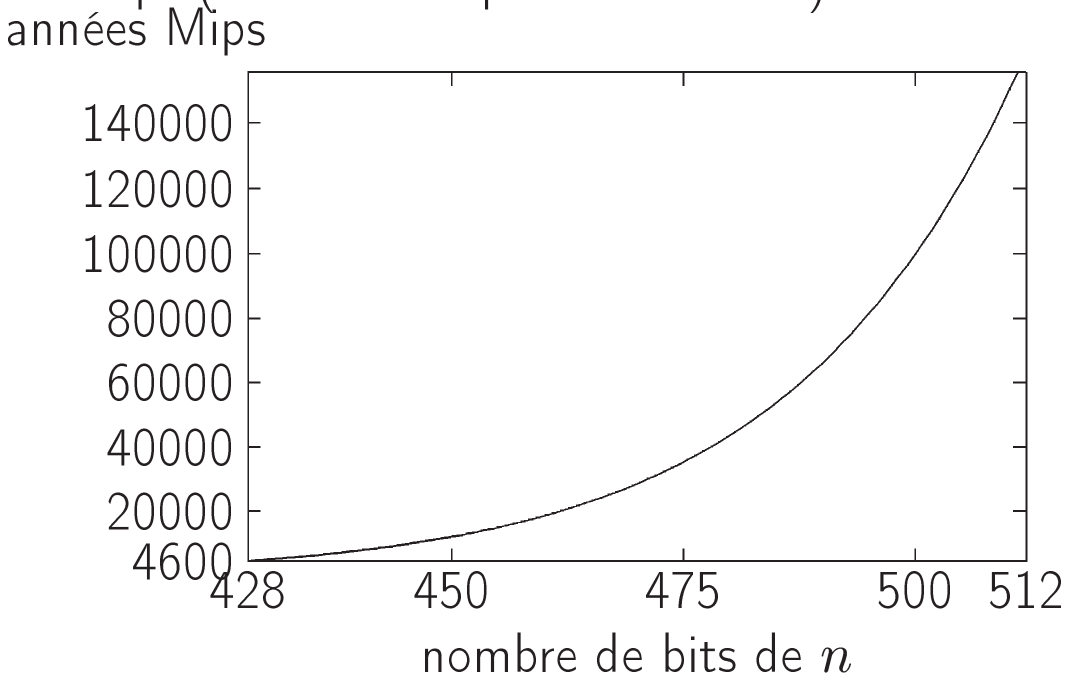
– Il n'est pas non plus prouvé que casser RSA est équivalent à factoriser.

– Retrouver l'exposant secret est équivalent à factoriser.

Garbage : Taille de la clef publique

Factorisation d'un nombre n de 428 bits (129 chiffres) en 1995.

$\approx$ 4 600 années Mips (un Pentium pendant 46 ans)



Garbage : Taille de la clef publique (2)

nb de bits de n	années Mips	temps sur 100 millions de Pentium
428	$4\ 600$	14,5 secondes
512	$1,7 \cdot 10^{5}$	8,5 minutes
700	$1,5 \cdot 10^{8}$	5,6 jours
1024	$2,2 \cdot 10^{11}$	228 ans
1500	$1,7 \cdot 10^{17}$	17 millions d'années
2048	$6,1 \cdot 10^{21}$	610 000 millions d'années

$\Longrightarrow$ On utilise pour n des nombres de 512, 1024 ou 2048 bits.

Factorisation

Divers problèmes :

- factoriser des nombres composés avec des petits facteurs,
- factoriser des nombres spéciaux,
- factoriser des nombres avec deux grands facteurs.

On considère ce dernier cas.

- Jusqu'en 70 : complexité exponentielle.
- Premier algo non exponentiel : Fractions Continues.

Complexité

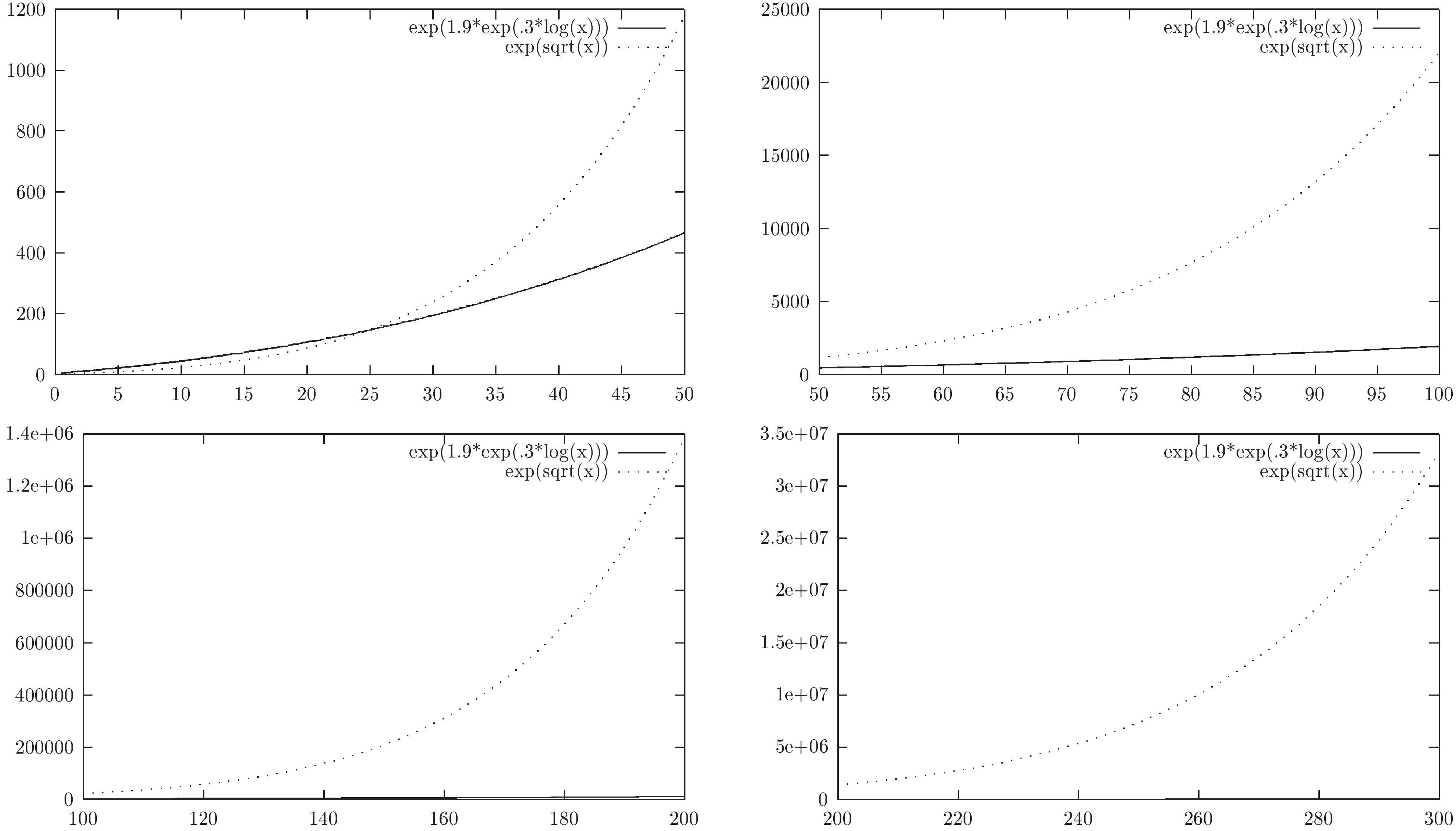
Notation :

$$L_{a,b}(n) = \exp\left((b+o(1)(n^a \log(n)^{1-a})\right).$$

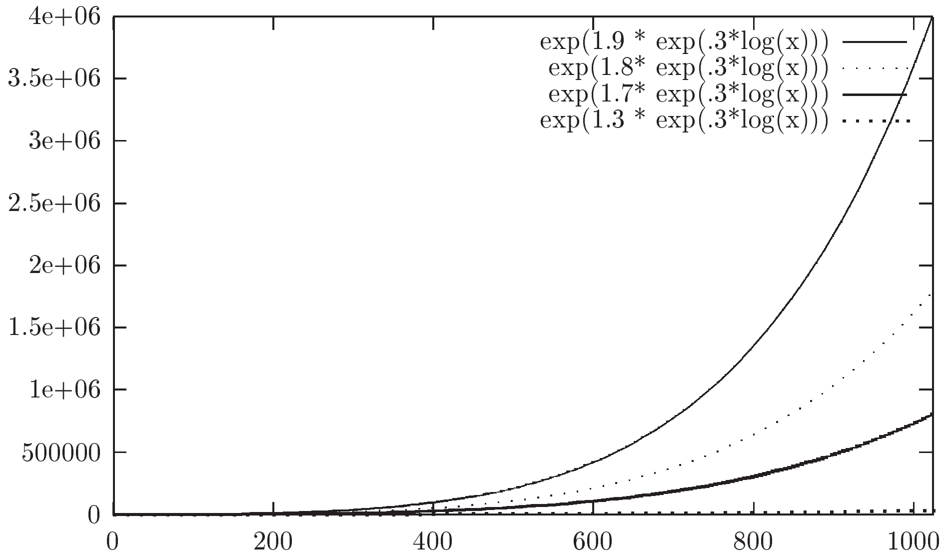
$L_{0,b}$ = polynomial, $L_{1,b}$ = purement exponentiel.

Algorithme	Complexité	Année
CFRAC	$L_{1/2,\sqrt{2}}$	1970
QS	$L_{1/2,1}$	80's
MPQS	$L_{1/2,1}$	80's
NFS	$L_{1/3,1.9}$	80's

La factorisation a un coût *sous-exponentiel* (conjecturé).


exp(1.9*exp(.3*log(x)))
exp(sqrt(x))
0
200
400
600
800
1000
1200
0
5
10
15
20
25
30
35
40
45
50
exp(1.9*exp(.3*log(x)))
exp(sqrt(x))
0
5000
10000
15000
20000
25000
50
55
60
65
70
75
80
85
90
95
100
exp(1.9*exp(.3*log(x)))
exp(sqrt(x))
0
200000
400000
600000
800000
1e+06
1.2e+06
1.4e+06
100
120
140
160
180
200
exp(1.9*exp(.3*log(x)))
exp(sqrt(x))
0
5e+06
1e+07
1.5e+07
2e+07
2.5e+07
3e+07
3.5e+07
200
220
240
260
280
300

Constante en haut



Structures de ces algorithmes

On cherche X et Y tel que $X^2 = Y^2 \bmod N$. On obtient ainsi $\gcd(X - Y, N)$ qui est facteur (pas toujours) de N.

On calcule des relations (collecte)

$$x_k^2 = \prod_{p \in B} p_1^{e_1} \dots p_m^{e_m} \bmod N \quad \text{où B est base de petits premiers}$$

On résout le système (sur F_2)

$$\begin{aligned} \sum \epsilon_k (e_1, \dots, e_m) = (0, \dots, 0) (\mathrm{mod} 2) &= (2v_0, \dots, 2v_m) \text{ d'où} \\ \left(\prod x_k^{\epsilon_k}\right)^2 &= \left(\prod p_k^{v_k}\right)^2. \end{aligned}$$

Nature des algorithmes

Une phase plus ou moins distribuée, où un nombre énorme de relations est collecté.

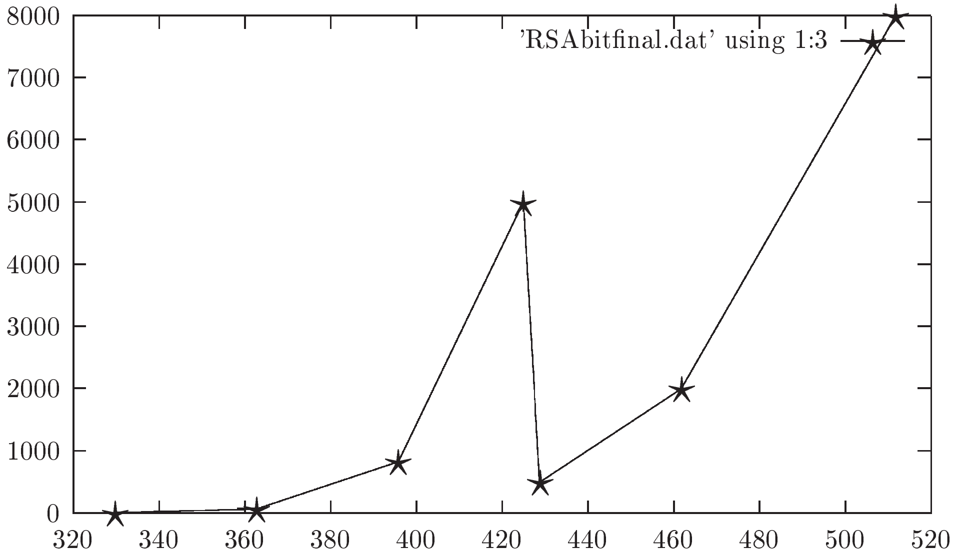
Une résolution de système linéaire, très creux et très grand.

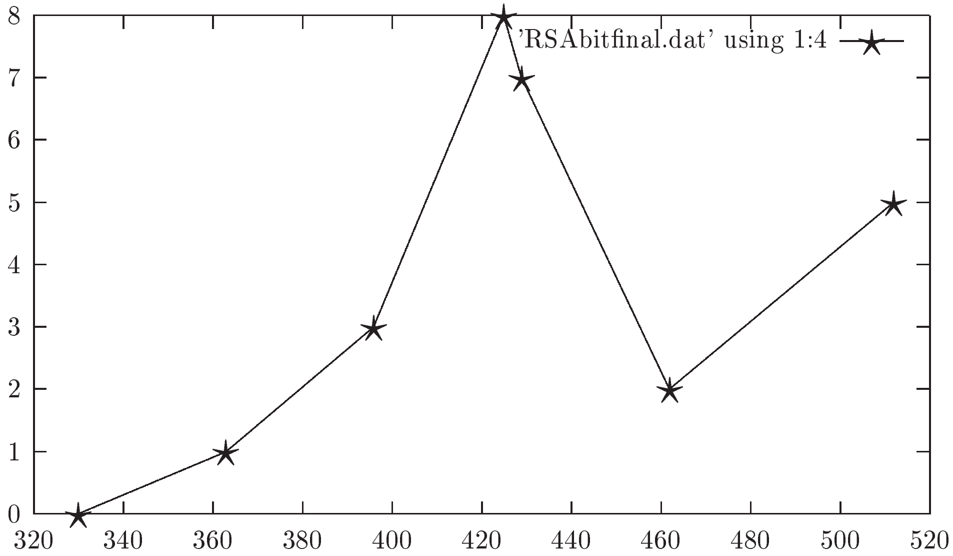
Grande technicité (Existence et Unicité d'implémentation ?).

Beaucoup de réglages (une Ferrari à régler).

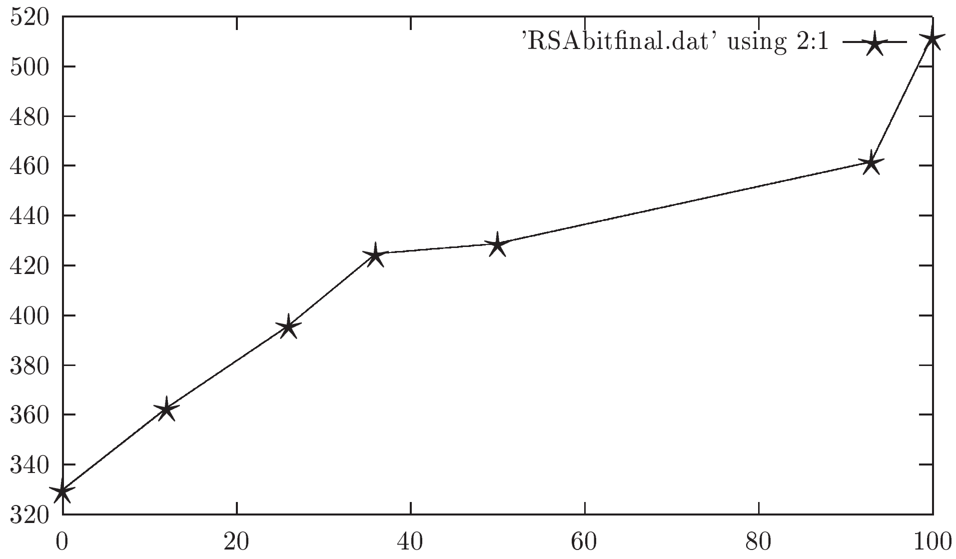
Beaucoup de sous-algorithmes non triviaux.

savoir-faire plutôt qu'un algorithme (Peut-on factoriser sans Montgomery ?).


'RSAbitfinal.dat' using 1:3
8000
7000
6000
5000
4000
3000
2000
1000
0
320
340
360
380
400
420
440
460
480
500
520


'RSAbitfinal.dat' using 1:4
8
7
6
5
4
3
2
1
0
320
340
360
380
400
420
440
460
480
500
520

Epoque de réalisation



IV. Courbes elliptiques

Courbes elliptiques : équations de Weierstrass.

Equation :

$$y^2 = x^3 + ax + b. \quad \text{avec } 4a^3 + 27b^2 \neq 0.$$

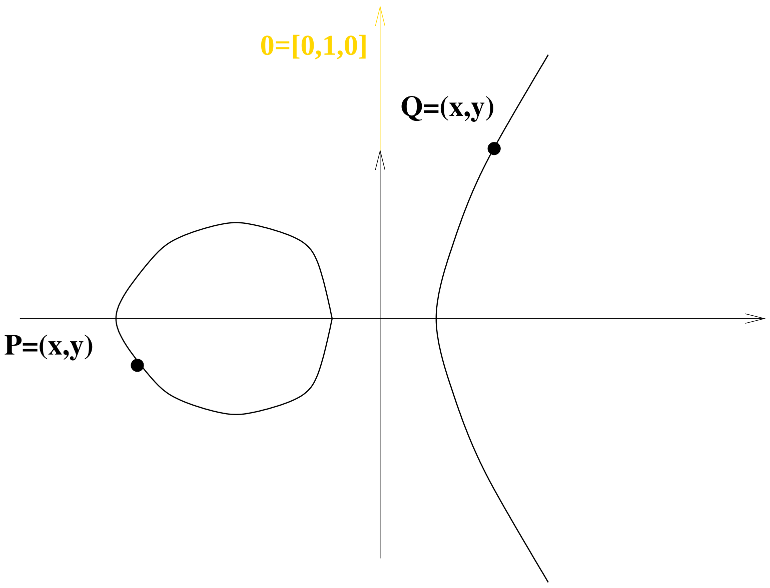
caractéristique 2 :

$$y^2 + xy = x^3 + a_2x^2 + a_6$$

- $K = R$ (cf. dessins),
- $K = F_p$ (nombre fini de points),
- $K = F_{2^n}$ idem,

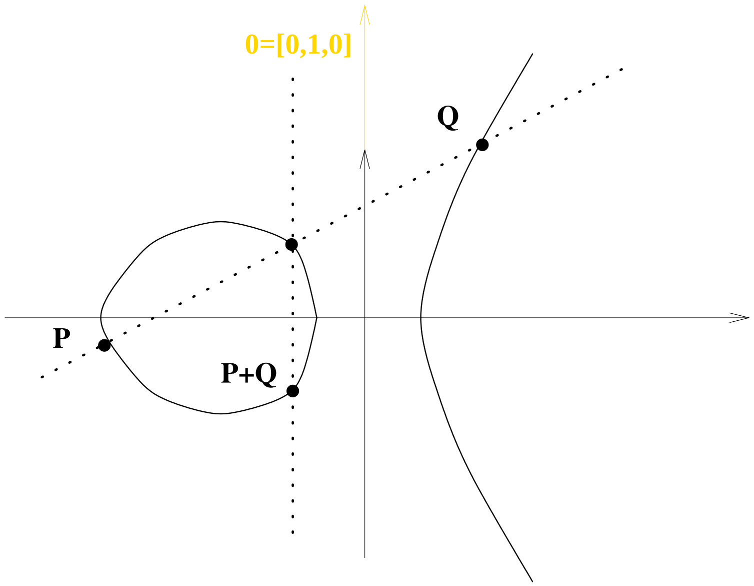
Courbes elliptiques : points.

$E(K)$ est l'ensemble des points $P = (x, y) \in K^2$ vérifiant l'équation de Weierstrass, plus le point $O = [0, 1, 0] \in P^2$ à l'infini.

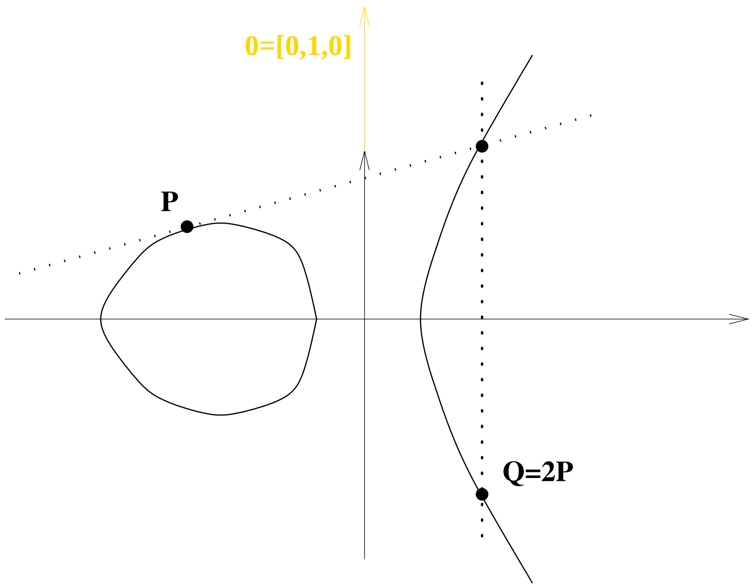


Courbes elliptiques : addition.

$$(x_1, y_1) + (x_2, y_2) = (x_3, y_3) \text{ avec } \begin{cases} x_3 = \lambda^2 - x_1 - x_2 \\ y_3 = \lambda(x_1 - x_3) - y_1 \end{cases}, \ \lambda = \frac{y_2 - y_1}{x_2 - x_1}$$



$$(x_1, y_1) + (x_2, y_2) = (x_3, y_3) \text{ avec } \begin{cases} x_3 = \lambda^2 - x_1 - x_2 \\ y_3 = \lambda(x_1 - x_3) - y_1 \end{cases}, \ \lambda = \frac{3x_1^2 + a}{2y_1}$$



Groupe des points d'une courbe elliptique sur F_q.

Théorème (Hasse). $|\#E(F_q) - (q+1)| \leq 2\sqrt{q}$.

Si q est grand, alors le groupe a un cardinal de l'ordre de q.

Exemples :

- $K = F_q$ est un nombre premier de 160 bits,
- $K = F_{2^n}$, avec n de 160 bits.

On utilise ce groupe pour faire du *logarithme discret*.

Diffie-Hellmann, ElGamal. . .

Protocoles en signature : ECDSA

Domaine : $E(F_q)$ une courbe elliptique, P un point de base, n l'ordre de P.

Clé d'Alice : $Q = sP$, s secret, Q public.

Signature

1. Tirer k aléatoire, calculer $R = kP$,
2. soit x l'abscisse $x \bmod n$ de R.
3. Calculer $d = k^{-1}(e + sx) \bmod n$.
4. (x, d) est la signature.

Vérification

1. Calculer $\begin{cases} u_1 = d^{-1}e \bmod n, \\ u_2 = d^{-1}x \bmod n. \end{cases}$
2. Calculer $u_1P + u_2Q = (x_0, y_0)$.
3. Accepter si $x_0 = x$.

$$u_1P + u_2Q = d^{-1}eP + d^{-1}xQ = d^{-1}(e + sx)P = kP$$

Attaques par collision

Soit $(G, +)$ un groupe, et P, Q les données d'un problème de logarithme discret, avec $Q = mP$, et m à trouver.

On construit une marche aléatoire $u_{n+1} = f(u_n)$ itérée de la manière suivante :

$$u_0 = a_0 P + b_0 Q$$
$$u_{k+1} = u_k + T_{\text{random}}$$

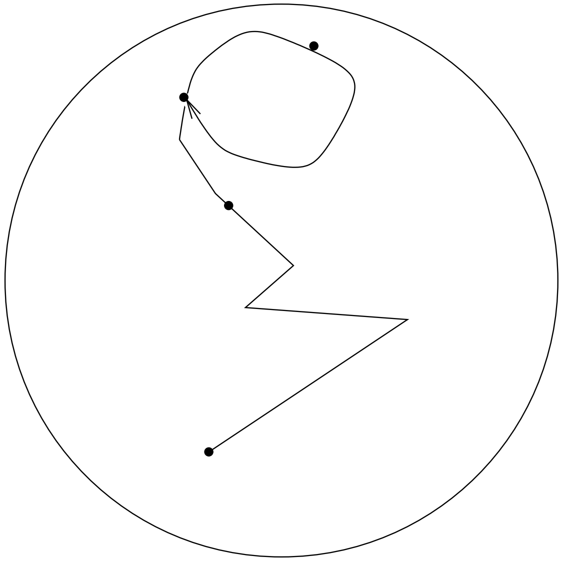
où T_{random} est tiré aléatoirement dans $\{T_1, \ldots, T_B\}$, B petit.

La structure additive de la marche permet de connaître (a_k, b_k) tel que $u_k = a_k P + b_k Q$.

Dés qu'on obtient une collision, $u_k = u_l$, alors $a_k P + b_k Q = a_l P + b_l Q$.

On en déduit :

$$m = \frac{a_k - a_l}{b_k - b_l}.$$



Dans le cas aléatoire, on trouve une collision en $\sqrt{\frac{\pi \# G}{2}}$ itérations. (Flajolet-Odlysko).

Les coureurs cyclistes :

On évite de garder un mémoire tous les points déjà parcourus en itérant deux suites, l'une deux fois plus rapide que l'autre.

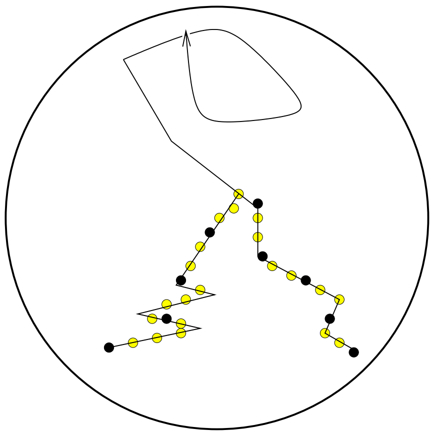
$$\begin{cases} u_k = f(u_k) \\ u_{2k} = f(f(u_k)) \end{cases}$$

Si G est de taille M

complexité de l'attaque : $O(\sqrt{M})$ en temps, O($\texttt{rien}$) en mémoire.

traduit en bit-size : $O(2^{\frac{n}{2}})$, où G est de bit-size m.

Méthode ρ parallèle



Complexité de l'attaque : $O(\sqrt{N}/m)$, $O(\sqrt{N})$ en mémoire.

Points distingués : $O\left(\frac{\sqrt{N}}{m} + \frac{1}{\theta}\right)$, mémoire $O\left(\theta\sqrt{N}\right)$.

Conséquences : longueur de clés.

Des précautions à prendre (éviter les courbes supersingulières, les courbes anomales)

Les seules attaques possibles sont les attaques génériques :

méthode ρ, de complexité $2^{n/2}$.

méthode ρ parallèle, de complexité $\frac{2^{n/2}}{m}$ (m processeurs).

Pour factoriser, la complexité est sous exponentielle.

Certicom estime qu'un groupe de taille 160 bits fournit une sécurité équivalente à RSA-1024.

Tailles

Hasse : $|\#E(F_q) - (q+1)| \leq 2\sqrt{q}$. Pour $\#E(F_q) \approx 2^n$, il faut $q \approx 2^n$.

En *domaine* constant, une clé publique est un point (x, y). Par *compression de point,* il suffit de stocker x, plus un bit pour indiquer quelle racine y de $y^2 = x^3 + ax + b$ est l'ordonnée du point.

Une clé publique a 161 bits.

Une signature ECDSA est constituée de deux entiers (r, s) de 160 bits, soit 320 bits, idem que DSA.

Pour la même résistance, RSA a 1024 bits de clé publique et de signature.

V. Records

DES Cracker

- Hardware spécial pour casser le DES par recherche exhaustive.

- Puces "Deep-Crack" calculent en parallèle.

- Algo
  - Recherche d'octets crédibles (69 caractères : $\frac{69}{256} \approx \frac{1}{4}$. Sur 8 octets : $1/65536 = 2^{-16}$, il reste 2^{40} clés.
  - Deuxième passe sur un deuxième message chiffré : il reste 2^{24} clés.

- Vitesse :
  - 16 cycles d'horloge pour chiffrer. Fréquence 40 Mhz : 2,5 Mkey/s.

– 24 unités sur une puce.

– “scalable” : n puces divisent le temps par n.

On met les puces dans des cartes dans des chassis.

Device	How Many In Next Device	Keys/Sec	Days/avg search
Search Unit	24	2,500,000	166,800
Chip	64	60,000,000	6,950
Board	12	3,840,000,000	109
Chassis	2	46,080,000,000	9.05
EFF DES Cracker		92,160,000,000	4.524

– 90 millards de clés à la seconde, avec moins de 1536 puces.

– 56 heures pour un DES.

Coût et ressources

The whole project was budgeted at about US \$ 210,000. Of this, \$80,000 was used to design, integrate, and test the EFF DES Cracker. The other \$130,000 was for materials including chips, boards and all other components on the boards, card cages, power supplies, cooling, and a PC. The software for controlling the EFF DES Cracker was written separately as a volunteer project that took 4-5 weeks. The entire project was completed within about eighteen months, with much of that time being used for preliminary research. The core team contained fewer than ten people, none of whom worked full-time on the project. The final cost came in at well under \$250,000.

RSA155

Algorithme : NFS

124 millions de relations obtenues en 3.7 mois (dont 39 millions de doublons).

8000 MIPS.YEAR

"Extrapolation using the asymptotic complexity formula for NFS would predict approximately 14000 mips years for RSA-155."

Matrice de 6,6 millions de lignes et de colonnes (61 "un" par ligne en moyenne).

Amélioration qualitative sur NFS.

Contigences

- 9 semaines de sélection de polynômes.
- 3 mois et 1/2 de collecte de relations.
- 224 heures de temps CPU et 3.2 Go de mémoire sur un Cray C916.
- 40 heures : 4 jobs racines carrées sur 4 SGI Origin 2000 (300 Mhz).

160	175-400 MHz SGI and Sun workstations
8	250 MHz SGI Origin 2000 processors
120	300-450 MHz Pentium II PCs
4	500 MHz Digital/Compaq boxes

ECC2-97

Contact : Robert.Harley@inria.fr

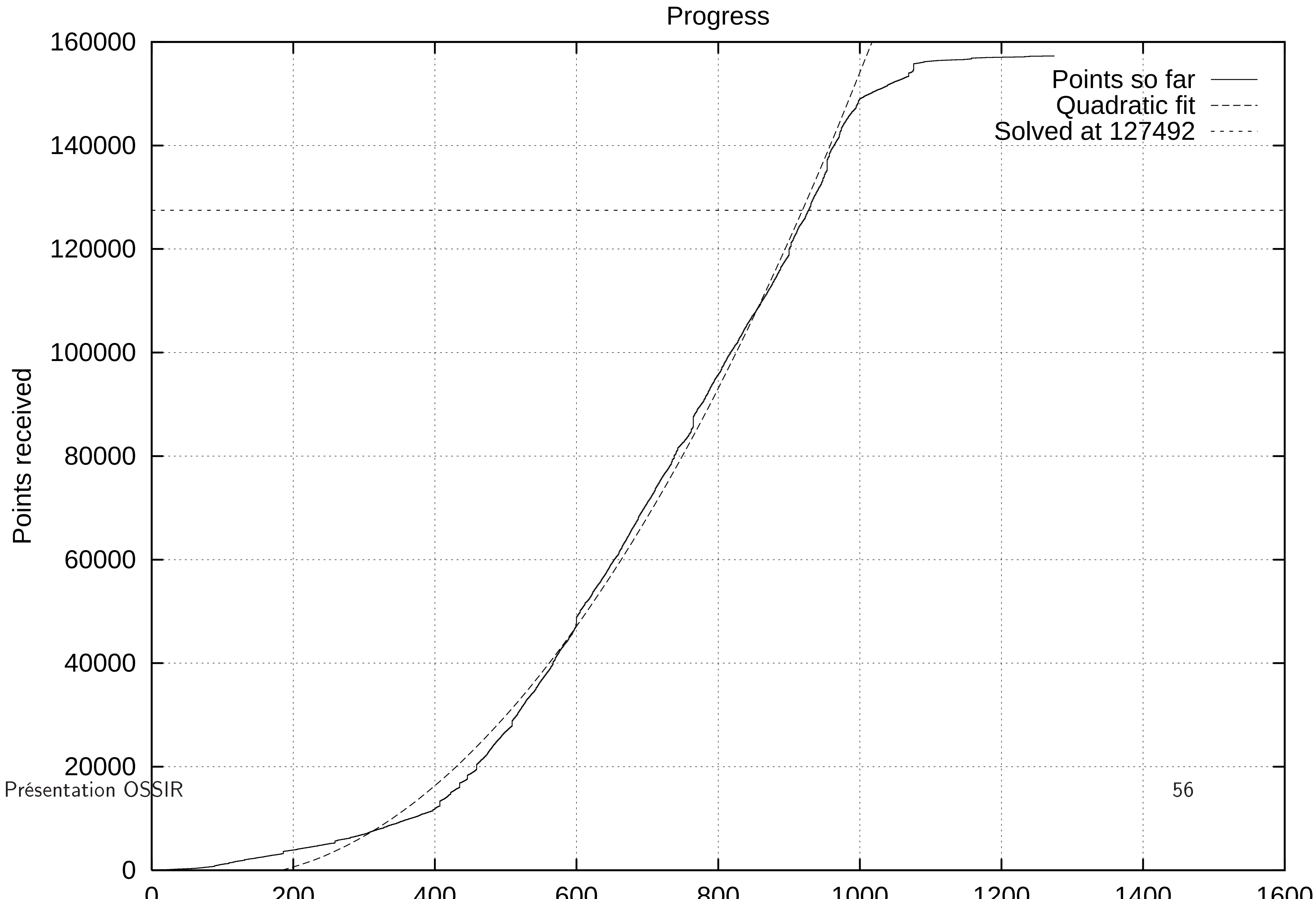
28 Septembre 1999.

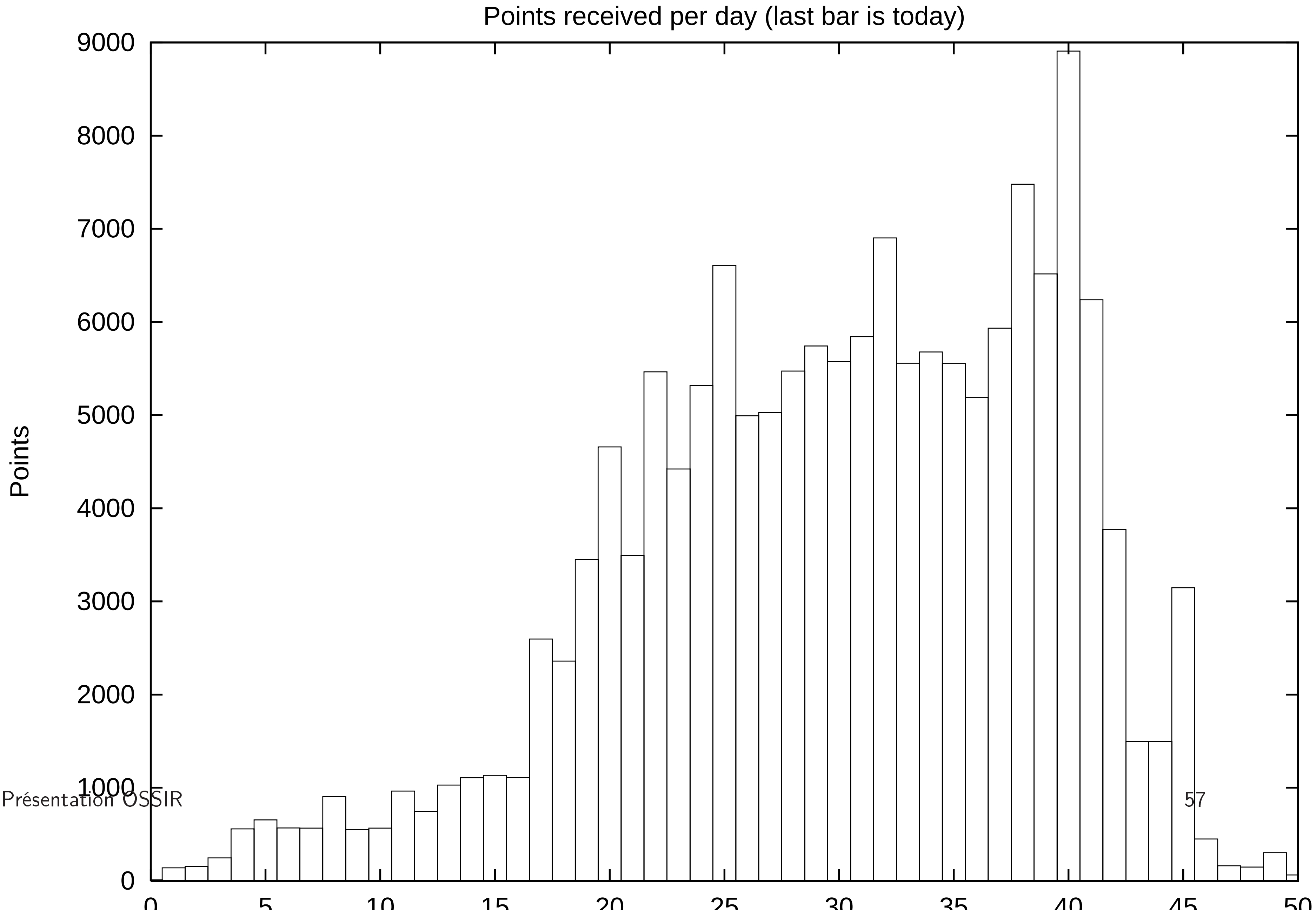
40 jours de calcul (3 fois moins que RSA155).

740 ordinateurs.

16000 MIPS.YEAR (2 fois plus que RSA155).

Chance : 1/3 seulement du nombre de calcul estimé : une chance sur 10 que celà se produise.


Progress
Points received
Points so far
Quadratic fit
Solved at 127492
0
20000
40000
60000
80000
100000
120000
140000
160000
0
200
400
600
800
1000
1200
1400
1600


Points received per day (last bar is today)
Points
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
0
5
10
15
20
25
30
35
40
45
50

Conclusion

– les problèmes difficiles mais pas trop sont atteignables peu à peu. . .

– à faible coût. . .

– dans un contexte ouvert.

La nature chaotique et rapide des progrès des machines ne permet pas de comparer les complexités.

	RSA 155		ECC2-97		DES
MIPS	RSA 155	$\xrightarrow{\times 6}$	ECC2-97	$\xrightarrow{\times 6}$	DES
Elapsed Time	RSA 155	$\xleftarrow{\times 4}$	ECC2-97	$\xleftarrow{\times 15}$	DES

Ce sont les paramètres des accords de Wassenaar !

```
5. A. 2. a. 1. a. A "symmetric algorithm" employing a key length in
                        excess of 56 bits; or

               b. An "asymmetric algorithm" where the security of the algorithm is
                  based on any of the following:

                      1. Factorisation of integers in excess of 512 bits (e.g., RSA);
                      2. Computation of discrete logarithms in a multiplicative group of
                      a finite field of size greater than 512 bits (e.g., Diffie-Hellman
                      over Z/pZ); or
                      3. Discrete logarithms in a group other than mentioned in
                      5.A.2.a.1.b.2. in excess of 112 bits (e.g., Diffie-Hellman over an
                      elliptic curve);
```