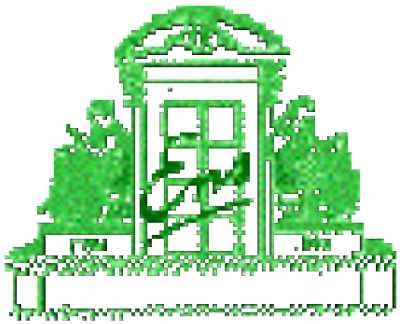
Initiation à la cryptographie

Cryptanalyse

Pierre-Alain FOUQUE
Équipe de Cryptographie
École normale supérieure



Plan du cours

- Cryptanalyse RSA
  - Signature d petit
  - Broadcast RSA
- Cryptographie multivariée
- Attaque par canaux auxiliaires sur RSA

Signature RSA

- d petit (accélération exponentiation modulaire)
- $d<(1/3).N^{1/4}$
- N bonne approximation de $\phi(N)$
- Chercher une bonne approximation de $e/\phi(N)$
- On a: $|e/N - k/d|<1/(2d^2)$

Fraction continuée

- Obtenir les meilleures approximations rationnelles d'un nombre x
  - ω Étant donné Q, elles calculent p'/q' tq ∀p,q, $0<q\leq Q$, $|p'/q'-x|<|p/q-x|$
- Notation: $[a_0,a_1,\ldots,a_n]$ pour

$$a_0+1/(a_1+1/(a_2+1/(\ldots a_{n-1}+1/a_n)))$$

- a_k sont appelés quotients partiels
- Pour k=0,..,n, les convergents sont $[a_0]=a_0/1$, $[a_0,a_1]=(a_1a_0+1)/a_1$,

Fraction continuée

- $[a_0,a_1,\ldots,a_{n-1},a_n]=[a_0,a_1,\ldots,a_{n-1}+1/a_n]$
- $[a_0,a_1,..,a_n]=a_0+1/[a_1,..,a_n]=[a_0,[a_1,..,a_n]]$
- Convergents successifs:
  - $p_0=a_0$, $q_0=1$, $p_1=a_0a_1+1$, $q_1=a_1$
  - $p_i=a_ip_{i-1}+p_{i-2}$, $q_i=a_iq_{i-1}+q_{i-2}$ et $[a_0,a_1,\ldots,a_i]=p_i/q_i$
- $p_iq_{i-1}-p_{i-1}q_i = (-1)^{i-1}$
- $p_iq_{i-2}-p_{i-2}q_i=(-1)^ia_i$

Fraction continuée simple

- Simple: si $a_1,\ldots,a_n \in \mathbb{N}$
- $x_i = p_i/q_i$ alors x_{2i} est strict. croiss. x_{2i+1} strict. décroissante et $x_{2i} \leq x_n \leq x_{2i+1}$
- $q_i \geq q_{i-1}$ et $q_i \geq i$
- Soit x réel: $a_0 = \lfloor x \rfloor$, $x = a_0 + \xi_0$ $0 \leq \xi_0 < 1$
- Si $\xi_0 \neq 0$, $a'_1 = 1/\xi_0$ et $a_1 = \lfloor a'_1 \rfloor$, $a'_1 = a_1 + \xi_1$ $0 \leq \xi_1 < 1$, …

FC d'un rationnel

- Un rationnel est représenté FC simple
- Algo:
  - Si x est entier, $\xi_0=0$ et $x=a_0$
  - Sinon $x=h/k$ avec h,k entiers et $k>1$
    $h/k=a_0+\xi_0=0$ et $h=ka_0+\xi_0 k$ où a_0 quotient et $\xi_0 k$ est le reste de la division eucli. de h par k
  - Si $\xi_0 k \neq 0$, $a'_1=1/\xi_0=k/k_1$ et $k/k_1=a_1+\xi_1$ et $k=a_1k_1+\xi_1k_1$ et a_1 est le quotient et $k_2=\xi_1k_1$ le reste de la division de k par k_1.
- C'est donc l'algorithme d'Euclide

Propriétés

- Si x est représentable par une FC simple avec un nbre pair (impair) de convergents, il l'est aussi par un nbre impair (pair)
- Si $x=(P\delta+R)/(Q\delta+S)$ où $\delta>1$, et P,Q,R et S sont des entiers, $Q>S>0$ et $PS-QR=\pm 1$, alors R/S et P/Q sont consécutifs de la FC de x
- Si $|p/q-x|<1/2q^2$, alors p/q est un convergent de x

Signature RSA

- Chercher les convergents de e/N
- Il y en a log(N) et parmi eux, il y a k/d
- Tester pour chacun si d est le bon en chiffrant et déchiffrant par exemple
- Extension attaque:
  - $d<N^{0.292}$ en utilisant des réseaux
  - Quand p et q sont proches $|p-q|<N^{\beta}$ pour $\beta\in]1/4,1/2[$ et $d=N^{\delta}$, alors $\delta<3/4-\beta$ (exos)

Broadcast RSA

- Soit
  - $c_1 = m^3 \bmod N_1$,
  - $c_2 = m^3 \bmod N_2$,
  - $c_3 = m^3 \bmod N_3$
- Trouver m ?

Généralisation

- Étant donné $(a_i m+b_i)^3 \bmod N_i$, pour $i=1,\ldots,k$ avec a_i et b_i connus, est-il possible de calculer m en temps polynomial ? $((2^{|ti|}m+t_i)^3 \bmod N_i)$
- $P_i(x)=0 \bmod N_i$ pour $i=1,\ldots,k$, où P_i de degré $\leq d$, est-il possible de retrouver toutes les solutions si $k>d(d+1)/2$ et $\min(N_i)>2^{d^2}$?

Réseaux

- $L=\{y : y=\sum_{i=1}^{n} a_i b_i,\ a_i \in Z\}$ où b_i vecteurs linéairement indépendants $\in R^n$
- b_i base de L et n est sa dimension
- Déterminant: $\det(L)=|\det(B)|$ $B=\mathrm{mat}(b_i)$
- Il est indépendant de la base
- Longueur du plus court vecteur non nul est notée λ_1
- Minkowski: $\lambda_1 \leq \gamma_n^{1/2}(\det(L))^{1/n}$ où γ_n est la constante de Hermite. Pour $n>8$, $\gamma_n \leq n$.

LLL algorithme

- Pb. Appro: Trouver un vecteur non nul $\leq 2^{(n-1)/2}\lambda_1$ de L en temps polynomial ?

- Th (LLL): étant donné L comme une base de vecteur entier de longueur $\leq B$, on peut trouver un vecteur b en temps $O(n^6(\log B)^3)$ tq $||b||_2 \leq 2^{(n-1)/4}(\det(L))^{1/n}$

Une seule équation

- Soit $N=\prod_{i=1}^{n} N_i$ et $n=\min(N_i)$
- Pb: Étant donné $\sum_{j=0}^{d} a_{ij}x^j = 0 \bmod N_i$, $i=1,\ldots,k$. Si le système a une solution $x<n$, et N_i sont premiers entre eux, retrouver x ?
- Soit $u_j<N$, $u_j=\delta_{ij} \bmod N_i$ ($\delta_{ij}=1$ ssi $i=j$)
- $0 = \sum_{j=0}^{d} x^j \sum_{i=1}^{k} u_i a_{ij} = \sum_{j=0}^{d} x^j c_j \bmod N$

Lemme

- Lemme: si $|c_j|<N/((d+1)n^j)$, et qu'il existe $c_j \neq 0$, on peut trouver tous les x tq $x<n$ et $\sum_{j=0}^{d} c_j x^j = 0 \bmod N$ en temps $O((d\log N)^3)$

- Conditions du lemme improbable d'être satisfaite en général

Théorème

- Étant donné un ensemble d'équations $\sum_{j=0}^{d} a_{ij}x^j = 0 \bmod N_i$, $i=1,\ldots,k$ où N_i premiers entre eux, et $\text{pgcd}((a_{ij})_{j=0}^{d},N_i)=1$, $\forall i$.
- On peut trouver tous les $x<n$, en temps $O(d^6(\log N)^3)$ si

$$N > n^{d(d+1)/2}.2^{(d+2)(d+1)/4}.(d+1)^{d+1}$$

où $\text{pgcd}((a_{ij})_{j=0}^{d},N_i)$ est le pgcd des $d+2$ nombres

Preuve

- Pour appliquer le lemme, on peut multiplier l'équation par S
- On veut que $|Sc_i \bmod N| < N/(n^i(d+1))$
- L réseau de dimension d+2
  - $b_1 = (c_0, nc_1, n^2c_2, \ldots, n^dc_d, 1/(d+1))$
  - $b_2 = (N, 0, 0, \ldots, 0, 0)$
  - $b_3 = (0, nN, 0, \ldots, 0, 0)$
  - $b_4 = (0, 0, n^2N, \ldots, 0, 0)$
  - ....
  - $b_{d+2} = (0, 0, 0, \ldots, n^dN, 0)$

Application

- Envoyer des messages linéairement liés avec RSA n'est pas sûr quand e est petit
- Envoyer plus de e(e+1)/2 messages permet à un adversaire de retrouver les messages quand

$$N_i > 2^{(e+2)(e+1)/4}(e+1)^{e+1}$$

Cryptographie multivariée

- Utiliser des polynômes à plusieurs variables
- $F(x)=x^{q^\theta+1}$ dans $GF(q^n)$
- P=ToFoS où S et T sont deux matrices inversibles sur GF(q) de taille n×n
- F peut se représenter par n équations quadratiques
- P aussi car S et T sont des changements de variables linéaires

Chiffrement/déchiffrement

- Pour chiffrer, calculer $c=P(m)$
- Pour déchiffrer,
  - on calcule $y=T^{-1}(c)$
  - si $pgcd(q^{\theta}-1,q^{n}-1)=q^{pgcd(\theta,n)}-1=1$, F est inversible.
    Si $h=1/(q^{\theta}+1) \bmod q^{n}-1$, $x=F^{-1}(y)=y^{h}$ dans $GF(q^{n})$
  - Calculer $m=S^{-1}(x)$
- Intérêts:
  - Pas d'algorithme en temps polynomial s'il existe des ordinateurs quantiques (problème NP-dur même si q=2)
  - Rapide sur des cartes à puces bas coût (sans coprocesseur crypto)

Attaque

- $B=F(A)=A^{q\theta+1}$
- $B^{q\theta-1} = A^{(q\theta+1)(q\theta-1)}$ donc $B^{q\theta-1} = A^{q^2\theta-1}$
- $AB^{q\theta} = A^{q^2\theta}B$
- Il existe n équations bilinéaires entre le chiffré et le clair
  - En prenant $n \times n(n-1)/2$ couples clair/chiffrés, retrouver ces équations
  - Pour déchiffrer, il reste à résoudre un système linéaire

Attaque par canaux auxiliaires sur RSA

- Attaque très efficace: retrouve des clés
- DFA: Faute pendant calcul m^d mod N
- SPA: Regarder la courbe de consommation du calcul de m^d mod N et si on sait distinguer les carrés des multiplications, on lit les bits de d
- DPA: Prendre plusieurs courbes et faire des statistiques (cf. DES)

Implémentations

$Exp_1(M,e,N)$:

```
{R=M;
 for (i=n-2 downto 0)
    {R=R^2 mod N;
     if (e_i==1)
            R=R.M mod N;
    }
 return R;
}
```

$Exp_2(M,e,N)$:

```
{R=1;S=M;
 for (i=0 to n-1)
    {if (e_i==1)
            R=R.S mod N;
     S=S^2 mod N;
    }
 return R;
}
```

Single Exp, Multiple Data (SEMD)

- La carte utilise un exposant secret
- On a le droit de modifier une seule fois l'exposant secret avec une valeur qu'on connaît (une seule autre carte)

- On compare les courbes de consommations pour plusieurs messages
  - Si on a le même bit dans les deux exposants, alors en soustrayant et moyennant, on obtient une corrélation entre les courbes
  - Sinon, on a du bruit blanc

Multiple Exp., Single Data (MESD)

- Il se peut que la courbe de consommation dépendent de la valeur des opérandes et pas seulement du bit de l'exposant
- On va deviner bit par bit les valeurs de d en modifiant l'exposant et en regardant si on a corrélation entre l'exposant secret et celui qu'on trouve petit à petit

Zero exp., Multiple Data (ZEMD)

- Deviner chaque bit petit à petit
- Consommation dépend poids de Hamming des opérandes
- Supposons qu'on connaisse les i bits de poids fort et on devine le bit (i+1) de d (d_g)
- Pour déterminer ce bit:
  - Calcul de M^{dg} : si un octet de M bien particulier contient un fort HW, mettre la courbe dans S_h sinon dans S_l
  - Calculer Moyenne de $S_l - S_h$. Si bon pari, corrélation forte, sinon bruit blanc

Contre-mesures DPA

- Masquage du message par un random r^e mod N et division par r à la fin du calcul peut éviter certaines attaques DPA, mais pas toutes (pas SEMD)

- Éviter d'utiliser le même exposant
  - Générer λ aléatoire de m bits (20 ou 32) et calculer $M^{d+\lambda\phi(N)}$ mod N

Windowing Algorithms

- Les contre-mesures rendent les algorithmes plus lents, donc on utilise des algorithmes d'exponentiations plus efficaces
- Square-and-multiply par fenêtre de 4 bits
  - Stocker M^a mod N pour $a \in \{1,\ldots,15\}$ et suivant la valeur des blocs de 4 bits, élever 4 fois au carré et multiplier par la valeur de M^a qui convient
- Autres algorithmes par fenêtre glissante (CLNW et VLNW)
  - On peut ici apprendre tous les blocs qui contiennent des zéros consécutifs et qu'ensuite, on commence par un « 1 »

Intérêt

- L'attaque SPA classique ne fonctionne pas si on ne sait pas distinguer les opérandes des multiplications,
  - cependant, on sait quand il y a 4 zéros consécutifs (proba 1/16)
  - en moyenne sur les algorithmes par fenêtre glissante, on apprend 40% des bits
  - pas besoin de protection SPA ?
  - des contre-mesures sont utilisées alors qu'aucune attaque est connue pour plus de sécurité ?
- Que se passe-t-il si la randomisation de l'exposant est utilisée sur un algorithme par fenêtre ?

Résultats classiques sur RSA

- N et $\phi(N)$ ont presque la moitié de leurs bits en commun (bits de poids forts = MSB)
  - $|N-\phi(N)|<3\sqrt{N}$
  - N une bonne approximation de $\phi(N)$
- Si e est petit, on apprend les MSB de d
  - Si e=3, $\overline{d}=(1+2N)/3$ bonne approximation de d
  - Si $e=2^{16}+1$, on sait qu'une bonne approximation de d est $(1+kN)/e$ où $k<e$ mais on ne connaît pas exactement la valeur de k

Attaque e=3

- On suppose qu'on connaît pour plusieurs randomisation de d, $d_j = d + \lambda_j \times \phi(N)$, une fraction 1/r de bits non consécutifs de d_j
- Étape 1: Retrouver pour chaque d_j, la valeur λ_j utilisée
  - on connaît bits de poids fort de d et $\phi(N)$
- Étape 2: Retrouver les bits de $\phi(N)$ en utilisant λ_j des bits de poids faibles vers les bits de poids forts

Étape 1: Trouver λ_j pour tout d_j

- Notons $\overline{d}$ l'approximation de d (on a presque la moitié des bits de poids fort de d)
- On peut calculer une approximation de

$$d_j = d + \lambda_j \times \phi(N) \approx \overline{d} + \lambda_j \times N$$

  qui est une égalité sur les MSB s'il n'y a pas de propagation de retenue $\Rightarrow$ $n/2-\alpha$ bits
- En remplaçant $\phi(N)$ par N et d par $\overline{d}$, on peut calculer pour les 2^m valeurs de j: $\overline{d}_j = \overline{d} + j \times N$
- Pour le bon j, on a presque les n/2 MSB de $\overline{d}_j$
- Tester la valeur j où les bits de $\overline{d}_j$ calculé correspondent aux bits observés de d_j

Étape 2: Retrouver $\phi(N)$ et d

- On va retrouver la clé secrète d par paquets de 8 bits des LSB vers MSB
- On fait une recherche exhaustive sur le bloc de 8 bits de $\phi(N)$
- On devine les 8 bits de $\phi' = \phi(N) \bmod 2^8$
- On en déduit $d = (1+2\phi')/3 \bmod 2^8$
- Si des bits de d_j connus sont dans les 8 premiers, on calcule $d_j = d + \lambda_j \times \phi' \bmod 2^8$
- S'il y a des bits connus de d_j parmi les 8 qui ne sont identiques, on rejette la valeur ϕ' et on passe à la suivante

Conclusion

- Cryptanalyse est nécessaire pour bien construire un schéma
- Il est important de voir que même si peu d'information fuit, on peut retrouver les informations secrètes

Probabilité de succès étape 1

- On veut obtenir les λ_j pour toutes les mesures (ω tests)
- Bad_j est l'événement qu'on ait associé un mauvais j à λ_j
- Si on estime que chaque bit est aléatoire et indépendant des autres, alors

  $$Pr[\text{Bad comp. j et } \lambda_j] = (1/2)^{(n/2-\alpha)/r}$$
- Comme on a 2^m valeurs de j pour chaque test, $Pr[\exists\ 1 \le j \le \omega: Bad_j] = 2^m\omega/2^{(n/2-\alpha)/r}$
- Pour n=1024, r=5, m=32, α=10 et ω=64, la probabilité de succès est au moins $1-1/2^{63}$

Probabilité de succès étape 2

- En pratique, l'attaque fonctionne très bien et pour chaque bloc de 8 bits, il ne reste que 1 ou 2 choix possibles pour $\phi(N)$
- De plus, si on fait le mauvais choix, au coup suivant on s'en aperçoit car aucun bloc n'est possible
- Si on estime que chaque bit est aléatoire et indépendant des autres,

  alors le nombre moyen de faux candidat pour un bloc de $\phi(N)$ est $(1/2^{8/r})^{8\omega/r} \times 2^8$ qui tend vers 0 quand ω augmente

Signature

- Pour contrer l'attaque précédente, enlever des équations par exemple r tq $q^r > 2^{80}$
- Pb: pour déchiffrer, il faudrait faire une recherche exhaustive sur les équations manquantes
- Utilisation en signature seulement
- Prendre pgcd(θ,n)>1 pour accélérer les calculs

Invariant

- Calcul de la différentielle
  - $DF(a,x)=F(a+x)-F(x)-F(a)=ax^{q\theta}+xa^{q\theta}$
  - C'est une équation bilinéaire en a et x
- Invariant:
  $DF(\xi a,x)+DF(a,\xi x)=(\xi+\xi^{q\theta})DF(a,x)$
- Si ξ tq $\xi \in Ker(x \rightarrow x+x^{q\theta})$ alors

  $DF(\xi a,x) = DF(a,\xi x)$ si caract(q)=2

Utilisation de l'invariant

- Soit M appli. linéaire tq

$$DF(M(a),x) + DF(a,M(x)) = 0$$

Alors $M(x)=\xi x$ dans $GF(q^n)$

- Recherche N: $DP(N(a),x)+DP(a,N(x))=0$
- $N=S^{-1}oMoS$
- Si on compose P avec N, on obtient un autre système de n-r équations
- Si on prend r équations dans ce deuxième système et qu'on les rajoute aux n-r premières de P, on obtient n équations