RSA-CRT et Signature RSA avec padding affine

Charles Duclos et Chunlong Zhu

24 octobre 2017

Plan

1. RSA-CRT
   - RSA :$m = c^d \mod n$ avec la clé privée d
   - Un cas particulier du théorème des restes chinois (CRT)
   - Le Théorème d'Euler
   - Algorithme
   - $(c^d \mod p, c^d \mod q)$?
   - Complexité
   - Exemple
2. Signature RSA avec padding affine
   - Signature primitive RSA
   - Signature RSA sans fonction de padding
   - Signature RSA avec padding affine
   - Cryptanalyse sur signature RSA avec padding affine

RSA : $m = c^d \bmod n$ avec la clé privée d

- Déchiffrer le texte c en utilisant RSA, avec la clé privée d
- Plus efficace avec CRT (le théorème des restes chinois) pour calculer $m = c^d \bmod n$.

Un cas particulier du théorème des restes chinois (CRT)

- Théorème : Soit p et q des nombres premiers distincts et $n = p \times q$. Pour toute couple (x_1, x_2) où $0 \leq x_1 < p$ et $0 \leq x_2 < q$, il existe un nombre unique x où $0 \leq x < n$ tel que $x_1 = x \bmod p$, et $x_2 = x \bmod q$.
- Donc tout entier x $(0 \leq x < n)$ peut être exprimé uniquement dans sa représentation CRT (x_1, x_2).

Le Théorème d'Euler

- Théorème : Si n est un entier positif et a est un nombre entier avec $pgcd(a, n) = 1$, alors $a^{\varphi(n)} \equiv 1 \mod n$ où $\varphi(n)$ est l'indicatrice d'Euler.

Algorithme

- 1.Précalculer les valeurs suivantes données p, q avec $p > q$,
  $d_P = (1/e) \bmod (p-1)$
  $d_Q = (1/e) \bmod (q-1)$
  $q_{inv} = (1/q) \bmod p$
  La clé privée devient le quintuplet $(p, q, d_P, d_Q, q_{inv})$.

- 2. Calculer le message m (étant donnée c)
  $m_1 = c^{d_P} \bmod p$
  $m_2 = c^{d_Q} \bmod q$
  $h = q_{inv} \times (m_1 - m_2) \bmod p$
  $m = m_2 + h \times q$

$(c^d \bmod p,\ c^d \bmod q)$?

- Pour récupérer x de sa représentation CRT (x_1, x_2), nous utilisons la formule de Garner.
  $x = x_2 + h \cdot q$, d'où
  $h = ((x_1 - x_2))((1/q) \mod \text{p})) \mod p$
- Après, on utilise le Théorème d'Euler pour réduire l'exposant d modulo $(p-1)$:
  $c^d \bmod p = c^{d \mod \varphi(p)} \bmod p = c^{d \mod (p-1)} \mod p$
  et de même pour la valeur mod q.

$(c^d \mod p,\ c^d \mod q)$?

- On pose d comme un multiple de $\varphi(p)$ plus un reste, $d = k \cdot \varphi(p) + d \mod \varphi(p)$, où k est un nombre entier.
- Par conséquent $c^d = c^{k \cdot \varphi(p) + d \mod \varphi(p)} = \left(c^{\varphi(p)}\right)^k \cdot c^{d \mod \varphi(p)}$
- Par le théorème d'Euler, $c^{\varphi(p)} \equiv 1 \mod p$
- Ainsi, $c^d \equiv 1^k \cdot c^{d \mod \varphi(p)} \equiv c^{d \mod \varphi(p)} \mod p$
  Finalement, puisque p est premier alors $\varphi(p) = p - 1$, et on a le résultat.

Finalement

- On sait que
  $d = e^{-1} \bmod (p-1)$, et
  $d = e^{-1} \bmod (q-1)$.
- Nous calculons la représentation CRT du message (m_1, m_2) :
  $d_P = (1/e) = d \bmod (p-1)$
  $d_Q = (1/e) = d \bmod (q-1)$
  $m_1 = c^{d_P} \bmod p$
  $m_2 = c^{d_Q} \bmod q$
- Donc : $q_{inv} = (1/q) \bmod p$
  $h = q_{inv} \cdot (m_1 - m_2) \bmod p$
  $m = m_2 + h \cdot q$

Complexité

- On sait que la complexité de RSA classique est $O(\log(n)^3)$.
  Par le théorème de restes chinois, on peut réduire la clé n à p et q.
  On sait $O(\log(n)) = O(\log(p \cdot q)) = O(\log(p) + \log(q))$ donc
  $\log(p) = \log(q) = \frac{1}{2}\log(n)$
- Donc, la complexité de RSA-CRT est
  $O((\log(p)^3 + (\log(q)^3) = O((\frac{1}{2}\log(n))^3 + (\frac{1}{2}\log(n))^3)$
  $= O(\frac{1}{4}\log(n))^3$.
- Donc, le RSA-CRT est 4 fois plus rapide que RSA classique.

Exemple

- On sait que
  $p = 137, q = 131, n = 137 * 131 = 17947, e = 3, d = 11787.m = 513$
  on calcule $c = 5133 \mod n = 8363$.

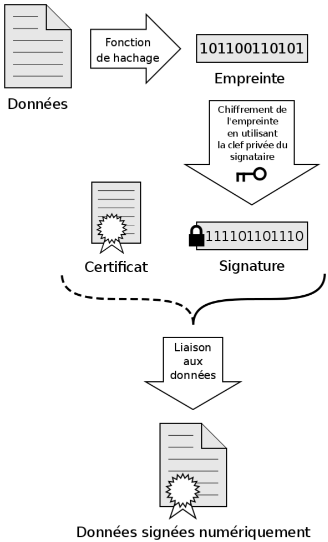
- Par CRT :
  $d_P = e^{-1} \mod (p-1) = d \mod (p-1) = 11787 \mod 136 = 91$
  $d_Q = e^{-1} \mod (q-1) = d \mod (q-1) = 11787 \mod 130 = 87$
  $q_{inv} = q^{-1} \mod p = 131^{-1} \mod 137 = 114$
  $m_1 = c^{d_P} \mod p = 836391 \mod 137 = 102$
  $m_2 = c^{d_Q} \mod q = 836387 \mod 131 = 120$
  $h = q_{inv} \times (m_1 - m_2) \mod p = 114 \times (102 - 120 + 137) \mod 137 = 3$
  $m = m_2 + h \times q = 120 + 3 \times 131 = 513.$

Signature primitive RSA

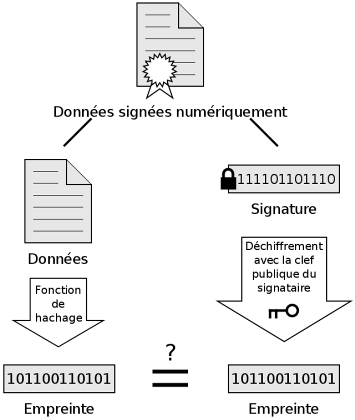
1. Alice souhaite envoyer à Bob un message M dont il puisse vérifier l'authenticité.
2. On suppose qu'Alice et Bob ont procédé à la création de clés, d est la clé privée d'Alice, (n, e) est la clé publique.
3. Alice calcule $S_M = M^d \mod n$ avec sa clé privée d :
4. Alice envoie M et S_M.
5. Bob déchiffre la signature avec la clé publique : $S_M^e \mod n$.
6. Si $S_M^e = M \mod n$, alors Alice est bien l'auteur du message.

Signature RSA

Signature



Vérification



Si les empreintes sont identiques, la signature est valide

Signature RSA sans fonction de padding

- Cas où $m = 1 \mod n$ et $m = 0 \mod n$
- Alice envoie (M, S_M) où $S_M = M^d \mod n$. Soit $\sigma \in (\mathbb{Z}/n\mathbb{Z})^*$, on peut usurper l'identité d'Alice en posant $M = \sigma^e$ et $S_M = \sigma$:
  $S_M^e = \sigma^e = M$.
  On parle de falsification sélective.
- On peut également créer une signature à partir de deux messages M_1 et M_2 et leurs signatures :
  Soient $S_{M_1} = M_1^d \mod n$ et $S_{M_2} = M_2^d \mod n$.
  Dans ce cas, $S_{M_1} \cdot S_{M_2} \mod n$ est une signature valide du message $M_1 \cdot M_2 \mod n$.

Signature RSA avec padding affine

- Pour éviter de signer directement un message M on utilise une fonction de padding, ou remplissage, $\mu(M)$
- On parle de *padding affine* lorsque $\mu(M) = \omega \cdot M + \alpha$, avec $\alpha, \omega \in (\mathbb{Z}/n\mathbb{Z})^*$.
- La signature d'un message M est donc :

$$\mu(M)^d \mod n = (\omega \cdot M + \alpha)^d \mod n$$

Cryptanalyse sur signature RSA avec padding affine

On cherche m_1, m_2, m_3 et m_4 quatre messages distincts de tailles égales au tiers de la taille de n, et tels que :

$$\mu(m_1) \cdot \mu(m_2) = \mu(m_3) \cdot \mu(m_4) \mod n \tag{1}$$

Alors, en utilisant les signatures de m_2, m_3 et m_4 on peut forger la signature de m_1 :

$$\mu(m_1)^d = \frac{\mu(m_3)^d \cdot \mu(m_4)^d}{\mu(m_2)^d} \mod n$$

(1) nous donne :

$$(\omega \cdot m_1 + \alpha) \cdot (\omega \cdot m_2 + \alpha) = (\omega \cdot m_3 + \alpha) \cdot (\omega \cdot m_4 + \alpha) \mod n$$

En posant $P = \alpha \cdot \omega^{-1} \mod n$, on obtient :

$$(P + m_1) \cdot (P + m_2) = (P + m_3) \cdot (P + m_4) \mod n$$

Cryptanalyse sur signature RSA avec padding affine

Soient :

- $t = m_3$
- $y = m_2 - m_3$
- $x = m_1 - m_3$
- $z = m_4 - m_1 - m_2 + m_3$

On peut simplifier l'équation précédente par :

$$x \cdot y = (P + t) \cdot z \mod n \tag{2}$$

On va chercher à déterminer les valeurs de x, y, z et t, tous de tailles égales au tiers de la taille de n.

On obtient deux entiers z et u tels que :

$$P \cdot z = u \mod n \text{ avec } \begin{cases} -n^{\frac{1}{3}} < z < n^{\frac{1}{3}} \\ 0 < u < 2n^{\frac{2}{3}} \end{cases}$$

On peut trouver une bonne approximation de la fraction $\frac{P}{n}$ en la développant en fraction continue. On trouve une solution telle que $|z| < Z$ et $0 < u < U$ si $Z \cdot U > n$, c'est le cas pour $Z = n^{\frac{1}{3}}$ et $U = 2 \cdot n^{\frac{2}{3}}$.
On choisit un entier y tel que $n^{\frac{1}{3}} \leq y \leq 2n^{\frac{1}{3}}$ et pgcd(y, z)= 1. On trouve un entier $t < y$ tel que :

$$t \cdot z = -u \mod y$$

puis on prend

$$x = \frac{u + t \cdot z}{y} \leq 4n^{\frac{1}{3}}$$

On obtient :

$$P \cdot z = u = x \cdot y - t \cdot z \mod n$$

qui correspond à l'équation (2)

Les quatre entiers x, y, z et t étant tous inférieur à $4n^{\frac{1}{3}}$. On retrouve les quatre messages chacun de la taille d'un tiers de la taille de n.

- $m_1 = x + t$
- $m_2 = y + t$
- $m_3 = t$
- $m_4 = x + y + z + t$

Comme $-n^{\frac{1}{3}} < z < n^{\frac{1}{3}}$ et que $y \geq n^{\frac{1}{3}}$ on a que $x + y > 0$, et sachant que $u \geq 0$:

$$x + t = \frac{u + t \cdot (y + z)}{y} \geq 0$$

Ce qui montre que les quatre entiers m_1, m_2, m_3 et m_4 sont positifs, et on a bien :

$$\mu(m_1) \cdot \mu(m_2) = \mu(m_3) \cdot \mu(m_4) \mod n$$

Conclusion

- L'attaque est en temps polynomial et permet une falsification existentielle.
- Il existe une attaque similaire qui permet une falsification sélective mais n'est pas en temps polynomial.
- Cette méthode de padding n'est pas utilisée.

Merci !