Cours 6. Théorie des nombres et hypothèses cryptographiques

HAI709I – Cryptographie

Bruno Grenet

Université de Montpellier – Faculté des Sciences

Introduction

Comment construire des primitives cryptographiques ?

Cryptographie symétrique

- Théorie :
  - Générateur pseudo-aléatoire $\Rightarrow$ chiffrement symétrique
  - Chiffre par bloc $\Rightarrow$ chiffrement symétrique
  - Fonction de compression $\Rightarrow$ fonction de hachage
  - …
- Pratique :
  - Générateurs pseudo-aléatoires : LFSR, …
  - Chiffre par bloc : AES, …
  - Fonctions de hachage : SHA-3, …
  - …

$\rightarrow$ Les hypothèses (théorie ou pratique) sont-elles réalistes ?

Hypothèses de difficulté

Problèmes algorithmiques considérés comme *difficiles* à résoudre

Objectifs

- Construire toute la cryptographie symétrique sur très peu d'hypothèses
- Définir une cryptographie *asymétrique*

Type de problèmes

- Souvent issus de la théorie des nombres :
  - Factorisation des entiers
  - Logarithme discret dans un groupe
- Cryptographie *post-quantique* :
  - Systèmes polynomiaux
  - Codes correcteurs
  - Problèmes *bruités*

1. Groupes

2. Entiers modulaires, RSA et factorisation

3. Algorithmes de théorie des nombres

Notion de groupe

Définition

Un groupe $(G, \star)$ est un ensemble et une opération binaire vérifiant

- *(loi interne)* pour tout $g, h \in G$, $g \star h \in G$
- *(identité)* il existe $e \in G$ tel que $g = g \star e = e \star g$
- *(inverses)* pour tout g, il existe h tel que $g \star h = h \star g = e$
- *(associativité)* pour tout $g_1, g_2, g_3 \in G$, $(g_1 \star g_2) \star g_3 = g_1 \star (g_2 \star g_3)$

Un groupe est abélien ou commutatif si

- *(commutativité)* pour tout $g, h \in G$, $g \star h = h \star g$

Pour nous

- On n'utilisera que des groupes *abéliens*
- (Très) souvent : $\star = \times$ (groupe *multiplicatif*) $\rightarrow$ neutre 1
- Parfois $\star = +$ (groupe *additif*) $\rightarrow$ neutre 0

Exemples et contre-exemples

Groupes

$(\mathbb{R}, +)$ $(\mathbb{R}\setminus\{0\}, \times)$

$(\mathbb{Z}, +)$ $(\mathbb{Q}\setminus\{0\}, \times)$

$(\mathbb{Q}, +)$ $(\mathbb{C}, +)$ $(\mathbb{C}\setminus\{0\}, \times)$

$(\mathbb{Z}/n\mathbb{Z}, +)$

(Polynômes, +)

Matrices avec +/×

Non-groupes

$(\mathbb{Z}\setminus\{0\}, \times)$

$(\mathbb{N}, +)$

Ordre d'un groupe – ordre d'un élément

Hypothèse : groupe multiplicatif ($\star = \times$)

Définitions

- L'ordre d'un groupe est son nombre d'éléments.
- L'ordre d'un élément $x \in G$ est le plus petit entier m tel que $x^m = 1$ ($= x \times x \times \cdots \times x$), éventuellement $+\infty$

Sous-groupe et sous-groupe engendré

- Un sous-groupe H de G est un groupe tel que $H \subset G$
- Si $x \in G$, l'ensemble $G_x = \{x^n : n \geq 0\}$ est le sous-groupe engendré par x

$(2\mathbb{Z}, +)$ s-gpe de $(\mathbb{R}, +)$

L'ordre d'un élément x est l'ordre du sous-groupe engendré par x

Théorème de Lagrange

Théorème (admis, mais pas difficile !)

Soit G un groupe et H un sous-groupe de G. Alors $|H|$ divise $|G|$.

Corollaire

Soit $x \in G$. Alors $x^{|G|} = 1$.

Preuve directe du corollaire

$X = \{ g \cdot x : g \in G \}$ $\quad X \subseteq G$

$m_x : g \mapsto g \cdot x$ est une bijection

$\hookrightarrow$ $m_x^{-1} = m_{x^{-1}}$ $\rightarrow$ $g \xmapsto{m_x} g \cdot x \xmapsto{m_{x^{-1}}} g \cdot x \cdot x^{-1} = g$

$\Rightarrow$ $|X| = |G|$

$\prod_{g \in G} g = \prod_{y \in X} y = \prod_{g \in G} (g \cdot x) = x^{|G|} \prod_{g \in G} g$ $\Rightarrow$ $1 = x^{|G|}$

1. Groupes

2. Entiers modulaires, RSA et factorisation

3. Algorithmes de théorie des nombres

Calculs *modulo n*

$\mathbb{Z}/n\mathbb{Z}$ est l'ensemble $\{0, \dots, n-1\}$ *muni des opérations modulo n*

Opérations additives *modulo n*

- L'addition *modulo n* de a et b est $(a+b) \bmod n$
- L'opposé *modulo n* de a est $(-a) \bmod n$ $= n - a$

$\rightarrow$ Pour tout n, $(\mathbb{Z}/n\mathbb{Z}, +)$ est un groupe (fini !)

Opérations multiplicatives *modulo n*

- La multiplication *modulo n* de a et b est $(a \times b) \bmod n$
- L'inverse *modulo n* de a est $b \in \mathbb{Z}/n\mathbb{Z}$ tel que $a \times b = 1$

$\rightarrow$ Est-ce que l'inverse *modulo n* existe ? Est-ce $(\mathbb{Z}/n\mathbb{Z}, \times)$ est un groupe ?

Remarque

On note (généralement) les opérations sans le « mod n »

Algorithme d'Euclide étendu

EuclideEtendu(a, b)

1. Si $b = 0$: renvoyer $(a, 1, 0)$ ✓
2. $(q, r) \leftarrow$ DivisionEuclidienne(a, b)
3. $(d, u_1, v_1) \leftarrow$ EuclideEtendu(b, r) → $d = bu_1 + rv_1$
4. Renvoyer $(d, v_1, u_1 - qv_1)$ → $av_1 + b(u_1 - qv_1) = bu_1 + (a - bq)v_1 = d$ ✓

Propriété (*coefficients de Bézout*)

EuclideEtendu(a, b) renvoie (d, u, v) tels que pgcd$(a, b) = d = au + bv$

Conséquence

- Si pgcd$(a, n) = 1$, il existe u, v tels que $au + nv = 1 \rightarrow a \times u \bmod n = 1$
- Si a a un inverse *modulo* n, alors pgcd$(a, n) = 1$

$a \in \mathbb{Z}/n\mathbb{Z}$ est inversible si et seulement si pgcd$(a, n) = 1$

$\mathbb{Z}/n\mathbb{Z}$ et $\mathbb{Z}/p\mathbb{Z}$

Cas p premier

Si p est premier, $\text{PGCD}(a, p) = 1$ pour tout $a \neq 0$

- Tous les éléments non nuls de $\mathbb{Z}/p\mathbb{Z}$ sont inversibles
- $(\mathbb{Z}/p\mathbb{Z} \setminus \{0\}, \times)$ est un groupe (fini)

$\mathbb{Z}/5\mathbb{Z} \setminus \{0\}$

$1^{-1} = 1 \quad 2^{-1} = 3 \quad 3^{-1} = 2 \quad 4^{-1} = 4$

Cas n non premier

Si k divise n, $\text{PGCD}(k, n) = k$

- Certains éléments non nuls de $\mathbb{Z}/n\mathbb{Z}$ ne sont pas inversibles
- $(\mathbb{Z}/n\mathbb{Z} \setminus \{0\}, \times)$ n'est **pas** un groupe

$\mathbb{Z}/6\mathbb{Z} \setminus \{0\}$. $1^{-1} = 1 \quad 5^{-1} = 5$

2, 3, 4 non inversibles

Groupes des inversibles

Si a, b sont inversibles dans $\mathbb{Z}/n\mathbb{Z}$, alors $a \times b$ aussi $\rightarrow (a \times b)^{-1} = a^{-1} \times b^{-1}$

Conséquence

Soit $(\mathbb{Z}/n\mathbb{Z})^\times$ l'ensemble des éléments inversibles de $\mathbb{Z}/n\mathbb{Z}$

- $((\mathbb{Z}/n\mathbb{Z})^\times, \times)$ est un groupe (*groupe des inversibles de* $\mathbb{Z}/n\mathbb{Z}$)
- $((\mathbb{Z}/n\mathbb{Z})^\times, +)$ n'est pas un groupe

Remarque

- $(\mathbb{Z}/p\mathbb{Z})^\times = \mathbb{Z}/p\mathbb{Z} \setminus \{0\}$

RSA

- Méthode de *chiffrement à clef publique*
  - Une clef *publique* pour chiffrer, connue de tout le monde
  - Une clef *privée* pour déchiffrer, connue uniquement de son propriétaire
- Version présentée ici non sûre, mais idée principale

Principe

- Génération des clefs :
  - On choisit deux premiers p, q aléatoires
  - On calcule $N = p \times q$ et $\varphi(N) = (p-1) \times (q-1)$
  - On choisit $e \in (\mathbb{Z}/\varphi(N)\mathbb{Z})^\times$, aléatoire $\rightarrow$ *clef publique* (e, N)
  - On calcule $d = e^{-1} \in \mathbb{Z}/\varphi(N)\mathbb{Z} \rightarrow$ *clef privée*
- Chiffrement d'un message clair $m \in \mathbb{Z}/N\mathbb{Z}$: $c \leftarrow m^e$ $(= m \times m \times \cdots \times m)$
- Déchiffrement d'un message chiffré c : $\tilde{m} \leftarrow c^d$

Remarque

- Travail avec deux « $\mathbb{Z}/n\mathbb{Z}$ » : $n = N$ et $n = \varphi(N)$

Justification de RSA

$d = e^{-1} \bmod \varphi(N)$

Pourquoi ça marche ?

- $\tilde{m} = c^d = (m^e)^d = m^{e \times d} \overset{(*)}{=} m^{1+k\varphi(N)} = m \times (m^{\varphi(N)})^k = m \times 1^k = m$
- Avec la clef publique : il suffit de calculer m^e dans $\mathbb{Z}/N\mathbb{Z} \to$ exponentiation rapide
- Avec la clef privée : il suffit de calculer $m^d \to$ idem

Lemme

L'ordre de $((\mathbb{Z}/N\mathbb{Z})^\times, \times)$ est $\varphi(N) = (p-1)(q-1)$ où $N = pq$

On veut mq le nb d'éléments inversibles dans $\mathbb{Z}/N\mathbb{Z}$ est $\varphi(N)$.

a inversible $\Leftrightarrow$ PGCD$(a, N) = 1$

Or pour tout a PGCD$(a, N) = \begin{cases} 1 \\ p \\ q \end{cases}$

Inversibles : $pq - p - q + 1$

$= (p-1)(q-1)$

Non inversibles $= \{0, p, 2p, \dots, (q-1)p, q, 2q, \dots, (p-1)q\} \to p-1+q-1+1 = p+q-1$

RSA est-il sûr ?

Comment attaquer RSA ?

- Un attaquant connaît e, N et c et cherche m tel que $m^e = c$ dans $\mathbb{Z}/N\mathbb{Z}$
- Exemples d'approches pour l'attaquant :
  - Factoriser N, calculer $\varphi(N)$ et inverser e dans $\mathbb{Z}/\varphi(N)\mathbb{Z} \to d$
  - Calculer directement $\varphi(N)$ sans factoriser, puis inverser $e \to d$
  - Trouver directement m sans calculer $\varphi(N)$ ni d

Hypothèse RSA

- Expérience RSA :
  - Protocole : produit (N, e, d) et tire uniformément $c \in (\mathbb{Z}/N\mathbb{Z})^\times$
  - Attaquant : étant donné N, e et c, calcule $m \in (\mathbb{Z}/N\mathbb{Z})^\times$
  - Succès de l'attaquant si $m^e = c$
- Hypothèse RSA : pour tout APP, $\Pr[m^e = c] \leq \text{negl}(\log N)$

La cryptographie basée sur RSA repose sur l'hypothèse RSA

Lien avec la factorisation

Hypothèse de factorisation

- Expérience de factorisation
  - Protocole : produit (N, p, q) avec $N = p \times q$, p, q premiers
  - Attaquant : étant donné N, calcule (p', q')
  - Succès de l'attaquant si $p' \times q' = N$
- Hypothèse de factorisation : pour tout APP, $\Pr[N = p' \times q'] \leq \mathrm{negl}(\log N)$

Proposition

- L'hypothèse RSA implique l'hypothèse de factorisation
- Contraposée : si la factorisation est facile, on sait casser RSA

Réciproque : problème ouvert

- *On ne sait pas* si factoriser est nécessaire pour casser RSA

1. Groupes

2. Entiers modulaires, RSA et factorisation

3. Algorithmes de théorie des nombres

Exponentiation rapide

Algorithme

Entrée : Un élément $x \in G$, un entier n
Sortie : x^n dans G

1. $y \leftarrow x$; $t \leftarrow 1$
2. Tant que $n > 0$:
3. Si n est impair :
4. $t \leftarrow t \times y$
5. $n \leftarrow n - 1$
6. $y \leftarrow y \times y$
7. $n \leftarrow n/2$
8. Renvoyer t

Propriétés

- Correction : invariant $t \times y^n = x^{n_{\text{init}}}$
- Complexité : $O(\log n)$ opérations dans G

Application

Primalité et génération de nombres premiers

Tests de primalité

- Algorithme de Agrawal-Kayal-Saxena : déterministe, complexité poly($\log n$)
- Algorithme de Miller-Rabin : probabiliste, complexité $O(\log^3 n)$
- …

Trouver un nombre premier

- Tirer des entiers (impairs), jusqu'à tomber sur un nombre premier
- Théorème des nombres premiers :
  - il y a env. $n/\log n$ nombres premiers $\leq n$
  - algorithme probabiliste de complexité $O(\log^4 n)$ (espérance)
- Pas d'algorithme déterministe connu

Algorithmes de factorisation

Algorithme *naïf*

- Tester tous les diviseurs possibles, par ordre croissant
- Si N a un facteur p ($\neq 1, N$), $p \leq \sqrt{N}$
- Algorithme en temps $O(\sqrt{N}\text{poly}\log N)$

Algorithmes rapides

- Méthode « $p-1$ » de Pollard $\rightarrow$ efficace si $N = pq$ et $p-1$ a des *petits* facteurs
- Méthode « rho » de Pollard, de complexité $O(N^{\frac{1}{4}}\text{poly}\log N)$
- Crible quadratique, de complexité $2^{O(\log^{\frac{1}{2}} N \log^{\frac{1}{2}} \log N)}$
- Crible algébrique, de complexité $2^{O(\log^{\frac{1}{3}} N \log^{\frac{2}{3}} \log N)}$

Remarques

- Records de factorisation : RSA-250 en 2020 (829 bits)
- Meilleurs algorithmes non polynomiaux… mais *sous-exponentiels*
- Algorithme quantique polynomial (Shor)

Conclusion

Groupes

- Notion mathématique indispensable pour la cryptographie asymétrique
- Principaux groupes utilisés :
  - Dans ce cours : $((\mathbb{Z}/n\mathbb{Z})^\times, \times)$ donc le cas particulier $((\mathbb{Z}/p\mathbb{Z})^\times, \times)$
  - Ailleurs : points d'une courbe (hyper-)elliptique, isogénies, …

Entiers modulaires, RSA, factorisation

- Cryptosystème RSA basé sur les propriétés de $\mathbb{Z}/n\mathbb{Z}$
  - *Attention : RSA tel que présenté n'est pas sûr !*
- Si la factorisation est facile, on sait casser RSA… mais la réciproque n'est pas connue

Algorithmes en théorie des nombres

- Test de primalité, tirage aléatoire de nombres premiers
  - Algorithmes polynomiaux (*en le nombre de bits !*)
- Factorisation
  - Pas d'algorithme (non quantique) polynomial connu
  - Meilleurs algorithmes *sous-exponentiels* en env. $2^{\log^{\frac{1}{3}} N}$

À suivre

Cryptographie à clef publique

- Échange de clefs : comment se mettre d'accord sur une clef secrète ?
- Chiffrement asymétrique : comment ne pas avoir à se mettre d'accord ?
- Signatures : équivalent des MACs, mais sans partage de clef

Hypothèses de difficulté

- Autre hypothèse que RSA $\rightarrow$ logarithme discret
  - permet l'échange de clefs
  - et le chiffrement asymétrique
- Comment rendre RSA sûr ?