Introduction à la Cryptologie – LU3IN024

Cryptologie moderne

Jérémy Berthomieu, Valérie Ménissier-Morain

7 avril 2025

Première partie I

Outils de chiffrement moderne

Plan

Bibliothèque implémentant la cryptographie

OpenSSL

Boîte à outils regroupant

- `libcrypto` pour les algos cryptographiques
- `libssl` implémentant le protocole Transport Layer Security (TLS) ainsi que son prédécesseur Secure Sockets Layer (SSL).

☞ Interface commande du terminal `openssl`

programmé en `C`, vulnérabilité Heartbleed trouvée et rapidement patchée en avril 2014, estimée catastrophique de par la proportion de serveurs Web touchée (17% de tous les serveurs du monde) mais pas d'exploitation spectaculaire avérée.

Exemple utilisation OpenSSL

Obtenir l'ensemble des chiffrements possibles

```
% openssl list -cipher-algorithms
AES-128-CBC
AES-128-CBC-HMAC-SHA1
AES-128-CBC-HMAC-SHA256
...
CAMELLIA-128-CBC
CAMELLIA-128-CFB
CAMELLIA-128-CFB1
...
SEED-CFB
SEED-ECB
SEED-OFB
```

Exemple utilisation OpenSSL

Chiffrer (*ENCryption*) / déchiffrer avec AES-256 (en mode CBC) et encodage Base64 (pas de caractères non imprimables)

```
% openssl enc -aes-256-cbc -base64 -in file.pdf -out file.pdf.enc
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
% openssl enc -aes-256-cbc -base64 -pbkdf2 -in file.pdf -out file.pdf.enc
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
% openssl enc -d -aes-256-cbc -base64 -pbkdf2 -in file.pdf.enc > file.pdf
enter aes-256-cbc decryption password:
```

Plan

Controverses sur le chiffrement des données

Le chiffrement a empêché la détection des attentats de Paris, selon la NSA

FBI et Apple s'affrontent sur le chiffrement de l'iPhone (2016-)

Deuxième partie II

L'échange des clés à l'essor de l'informatique

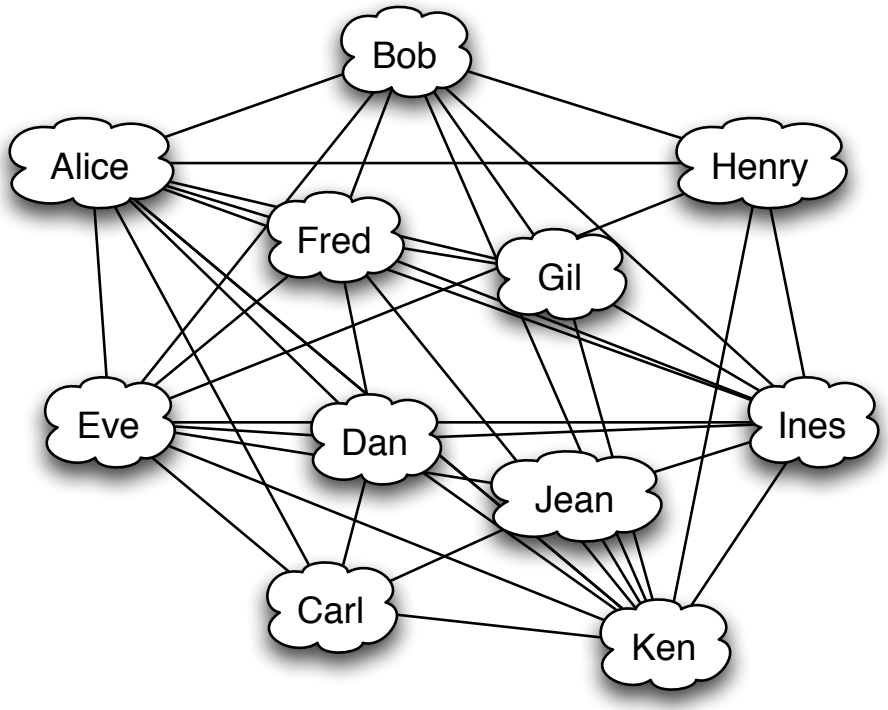
L'informatique et ses réseaux de communication se mettent en place

- 1950 Les premiers ordinateurs commerciaux, cryptographie basée sur la taille des clés, Théorie de Shannon.
- 1970 Réseaux numériques, standardisation du `DES` (chiffrement symétrique) par la NSA
- 1970 L'ordinateur devient un outil pour toutes les entreprises

Problème : l'échange des clés

Problématique

Un nombre énorme de clés à gérer et échanger !



☞ Il faut un moyen sûr d'échanger chaque clé au préalable !

Plan

Échanger une clé sans jamais se rencontrer !

☞ En 1976, Diffie, Hellman et Merkle publient le premier schéma d'échange de clés à l'aide de la notion de clé publique qu'ils énoncèrent.

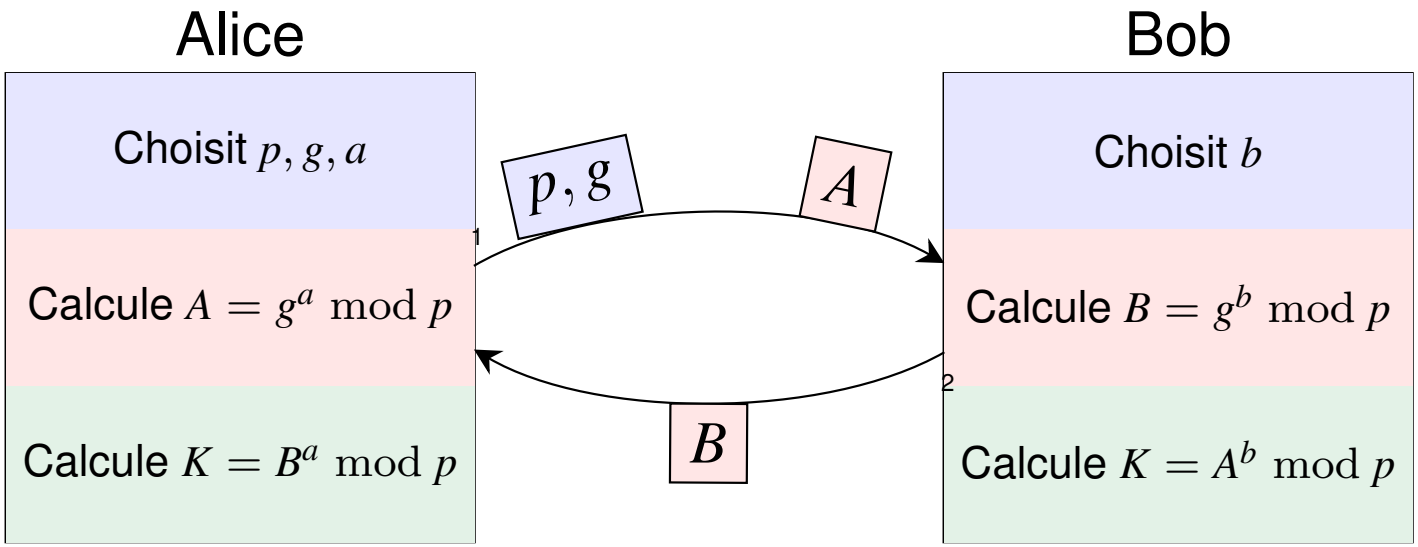
1 INTRODUCTION

We stand today on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where

Échanger une clé sans jamais se rencontrer !

Soit p un nombre premier et g un élément de $\mathbb{Z}/p\mathbb{Z}$ bien choisi. Ces deux éléments sont connus de tous.
Les entiers a et b doivent être tenus secrets.

☞ Nécessite deux transactions sur un canal non sécurisé pour l'échange de la clé commune K.



$$K = A^b \bmod p = (g^a \bmod p)^b \bmod p = g^{ab} \bmod p = (g^b \bmod p)^a \bmod p = B^a \bmod p.$$

Coût de la mise en pratique (cryptographie)

☞ En tout, les deux entités ont à calculer 4 exponentiations mod p.

En utilisant un algorithme d'exponentiation modulaire rapide (voir cours précédent), le coût total dépend de la taille du module p.

Complexité échange de clé par Diffie-Hellman-Merkle

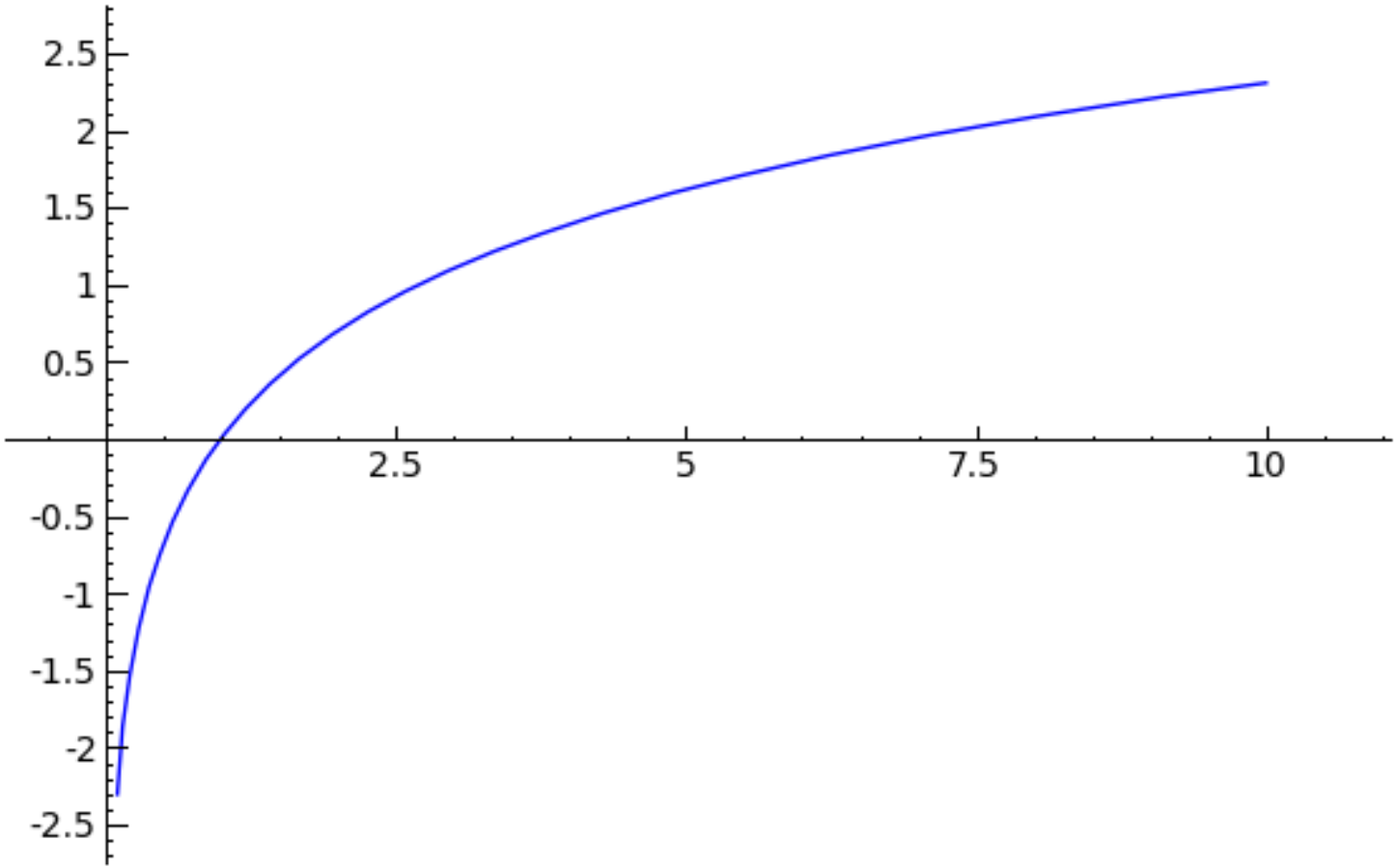
Si l'on souhaite échanger une clé de taille ℓ il faudra faire

$$\mathcal{O}(4\ell^3)$$

opérations binaires.

Sécurité de cette échange (cryptanalyse)

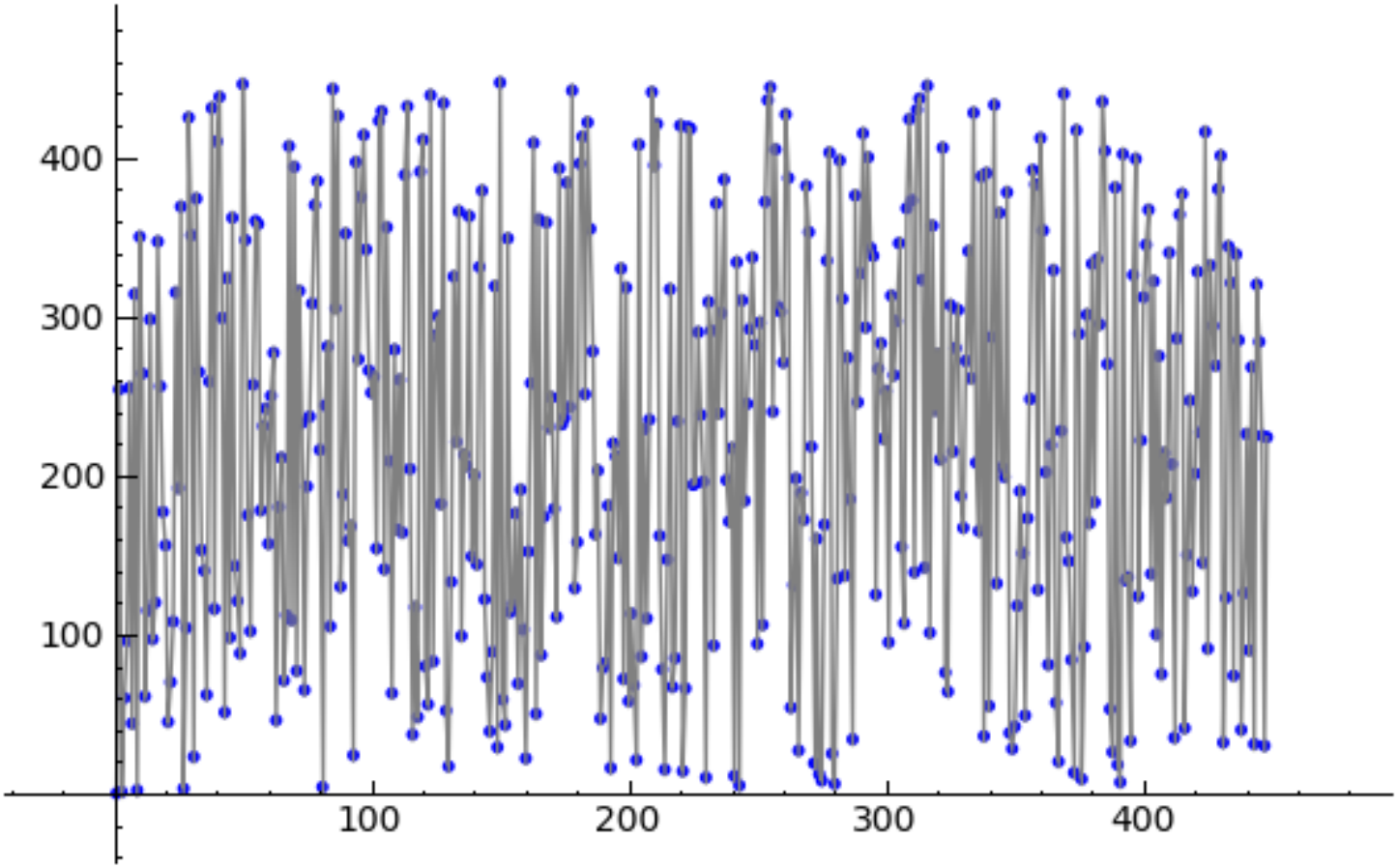
Étant donné $y = e^x \in \mathbb{R}$



Il est facile de calculer x (il suffit de calculer un $\log$ dans $\mathbb{R}$)

Sécurité de cette échange (cryptanalyse)

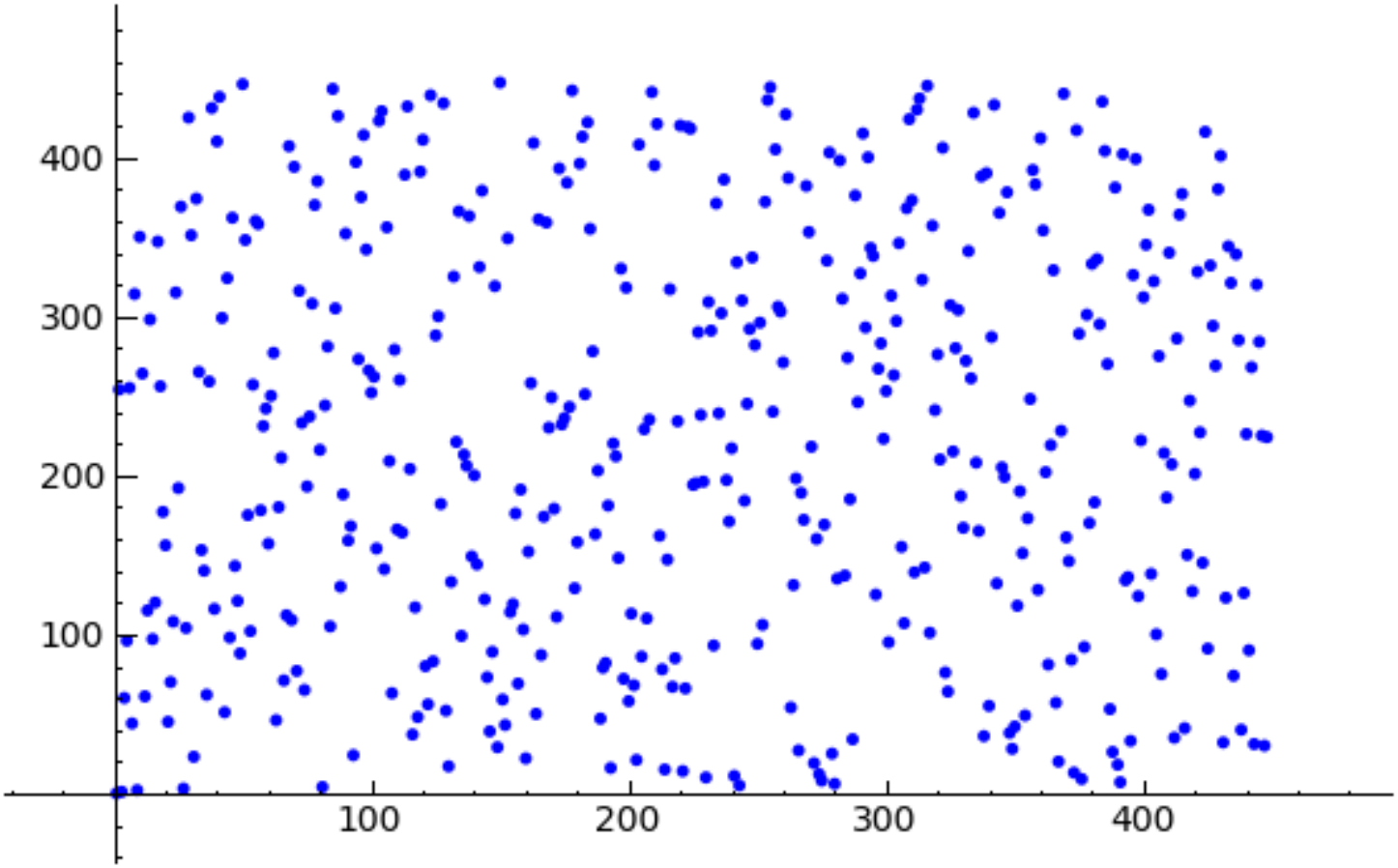
Étant donné $192 = g^x \bmod 449$ ($g = 3$ générateur du groupe cyclique)



Il est difficile de calculer x

Sécurité de cette échange (cryptanalyse)

Étant donné $192 = g^x \bmod 449$ ($g = 3$ générateur du groupe cyclique)



Il est difficile de calculer x

☞ Il s'agit du problème du calcul du logarithme discret (DLP).

Plan

Le DLP : vision générale

Définition

Soit $(G, \circ)$ un groupe fini et $g \in G$. Le problème du logarithme discret (noté DLP) dans le sous-groupe $H = \langle g \rangle$ de G est défini comme suit :

- Étant donné $h \in H$
- Comment retrouver $k \in \mathbb{N}$ tel que $h = g \circ \cdots \circ g$ (k fois) ?

Notations

Pour $g \in G$ et $t \in \mathbb{N}$, on notera l'opération d'*exponentiation* de g en t par :

$$[t]g = g \circ \cdots \circ g, \ t \text{ fois}$$

On peut généraliser à $t \in \mathbb{Z}$ en posant $[-t]g$ l'inverse de $[t]g$.

Le DLP : vision générale

Définition

Soit $(G, \circ)$ un groupe fini et $g \in G$. Le problème du logarithme discret (noté DLP) dans le sous-groupe $H = \langle g \rangle$ de G est défini comme suit :

- Étant donné $h \in H$
- Comment retrouver $k \in \mathbb{N}$ tel que $h = g \circ \cdots \circ g$ (k fois) ?

Notations

Pour $g \in G$ et $t \in \mathbb{N}$ on notera l'opération d'*exponentiation* de g en t par $[t]g = g \circ \cdots \circ g,\ t$ fois. Cette notation permet d'homogénéiser les notations, par exemple :

- Si $(G, \times)$ on a $[t]g = g \times \cdots \times g = g^t$
- Si $(G, +)$ on a $[t]g = g + \cdots + g = t \cdot g$

Le DLP : vision générale

Définition

Soit $(G, \circ)$ un groupe fini et $g \in G$. Le problème du logarithme discret (noté DLP) dans le sous-groupe $H = \langle g \rangle$ de G est défini comme suit :

- Étant donné $h \in H$
- Comment retrouver $k \in \mathbb{N}$ tel que $h = g \circ \cdots \circ g$ (k fois) ?

Définition

Le logarithme discret en base g de h est défini par

$$k = \log_g h \text{ avec } h = [k]g$$

Plan

Brute force attack

Problématique

Étant donnés :

- un groupe fini cyclique $H = \langle g \rangle$ dans lequel on peut calculer *facilement* ;
- un élément h dans H.

On cherche un entier t tel que $h = [t]g$.

☞ On peut se limiter à $t \in \{1, \ldots, |H|\}$.

Recherche exhaustive

On testera $2^{\log |H|}$ valeurs de t différentes.

La complexité de résolution du DLP s'exprime en fonction de $\log |H|$, la taille de $|H|$ (la taille en bits du groupe H)

Exemples où le DLP est facile

Groupe additif modulaire

Ici $H = (\mathbb{Z}/n\mathbb{Z}, +)$ le groupe modulaire additif. On a $H = \langle g \rangle$ et donc $\mathrm{pgcd}(g, n) = 1$.

- Groupe additif : $h = [t]g = t \cdot g \bmod n$
- Ici on peut calculer facilement $\log_g h$.

Preuve de la résolution du DLP dans ce cas

1. $\mathrm{pgcd}(g, n) = 1 \Rightarrow$ on peut calculer $g^{-1} \bmod n$ par Bézout
2. on retrouve t facilement : $\log_g h = h \cdot g^{-1} \bmod n$

La complexité binaire de cette attaque est quadratique en la taille de n (qui est bien la taille de $|H|$).

☞ Pour résoudre facilement un DLP, on peut essayer de se ramener à cet exemple.

Résolution du DLP par transfert additif

Transfert additif

Soit H le groupe cyclique choisi pour support au DLP. Il faut pour un tel transfert :

- isomorphisme entre H et $\mathbb{Z}/|H|\mathbb{Z}$
- qu'il soit explicite
- et que l'on puisse calculer efficacement l'image/image inverse d'un élément par ce dernier

Le DLP sera facile dans H si l'on connait un tel isomorphisme. (On envoie le problème dans $\mathbb{Z}/|H|\mathbb{Z}$ avec l'isomorphisme, on le résout grâce à Bézout et on remonte le résultat dans H toujours grâce à l'isomorphisme.)

Complexité générique

Définition

On appelle groupe générique (ou en boite noire) un groupe dont on ne connaît pas la structure mais avec lequel on peut réaliser des calculs. En particulier, si $(H = \langle g \rangle, \circ)$ et $h \in H$ sont donnés en boite noire, tous les calculs de la forme

$$[a]h \circ [b]g$$

sont possibles et on peut tester les égalités.

Théorème [Shoup 97]

La résolution du DLP dans un groupe générique nécessite au moins $\sqrt{|H|}$ opérations dans H

Existe-t-il un algorithme optimal dans le cas générique ?

☞ Simplification : *connaissance de* $n = |H|$.

Baby-Step Giant-Step de Shanks : idée générale

Soit n l'ordre du groupe $H = \langle g \rangle$ et $h \in H$, on pose $s = \lfloor \sqrt{n} \rfloor + 1$.

Division euclidienne de t par s

L'entier $t = \log_g h$ vérifie $\boxed{1 < t < n}$,

donc il existe un unique couple (q, r) tel que $\boxed{t = q \cdot s + r \text{ avec } 0 \leqslant q, r < s}$

Principe : recherche exhaustive sur q et r d'une collision

$$h = [t]g = [q \cdot s + r]g = [q \cdot s]g \circ [r]g \Rightarrow \boxed{h \circ [-q \cdot s]g = [r]g}$$

donc

Algorithme

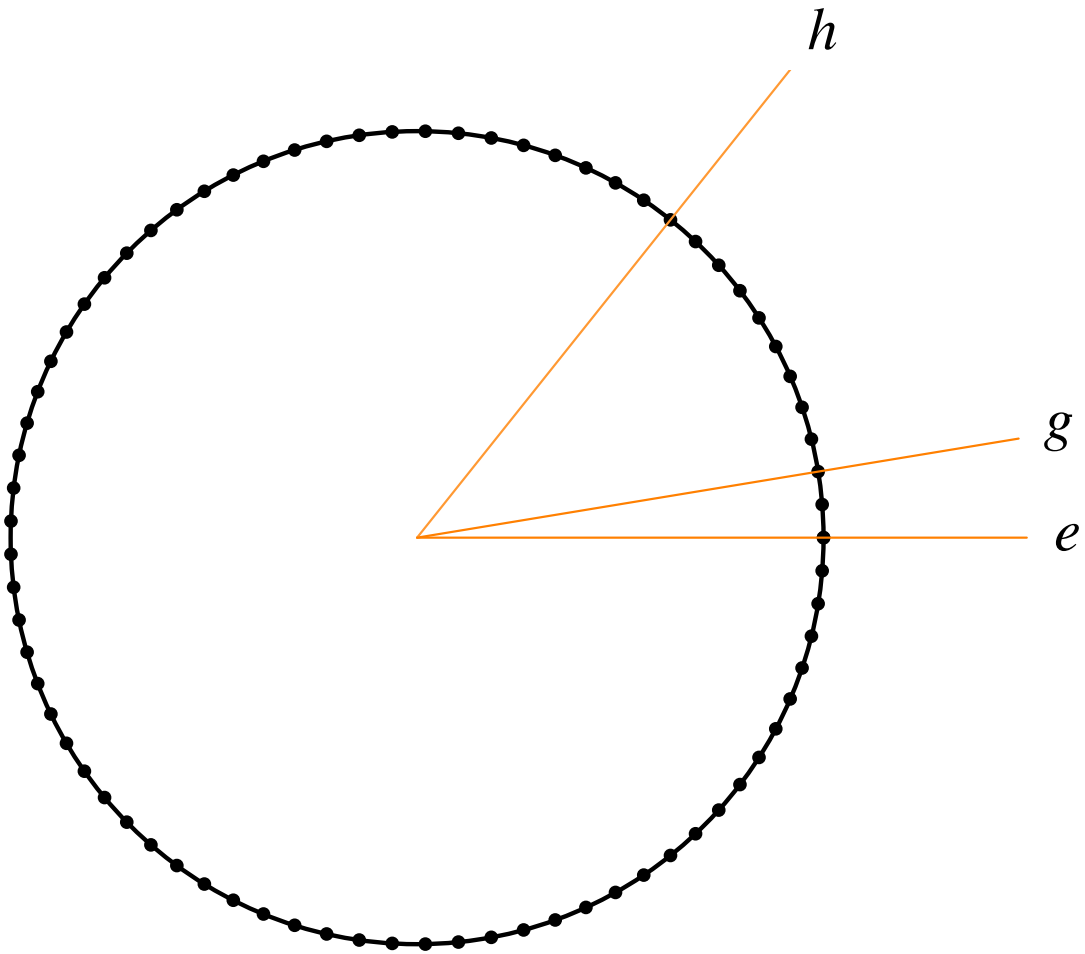
On calcule $s = \lfloor \sqrt{n} \rfloor + 1$.
On énumère les deux suites

GS $\{h \circ [-j \cdot s]g | 0 \leqslant j < s\}$ **BS** $\{[i]g | 0 \leqslant i < s\}$

et il se produit exactement une collision qui nous fournit q er r et finalement $t = qs + r$.

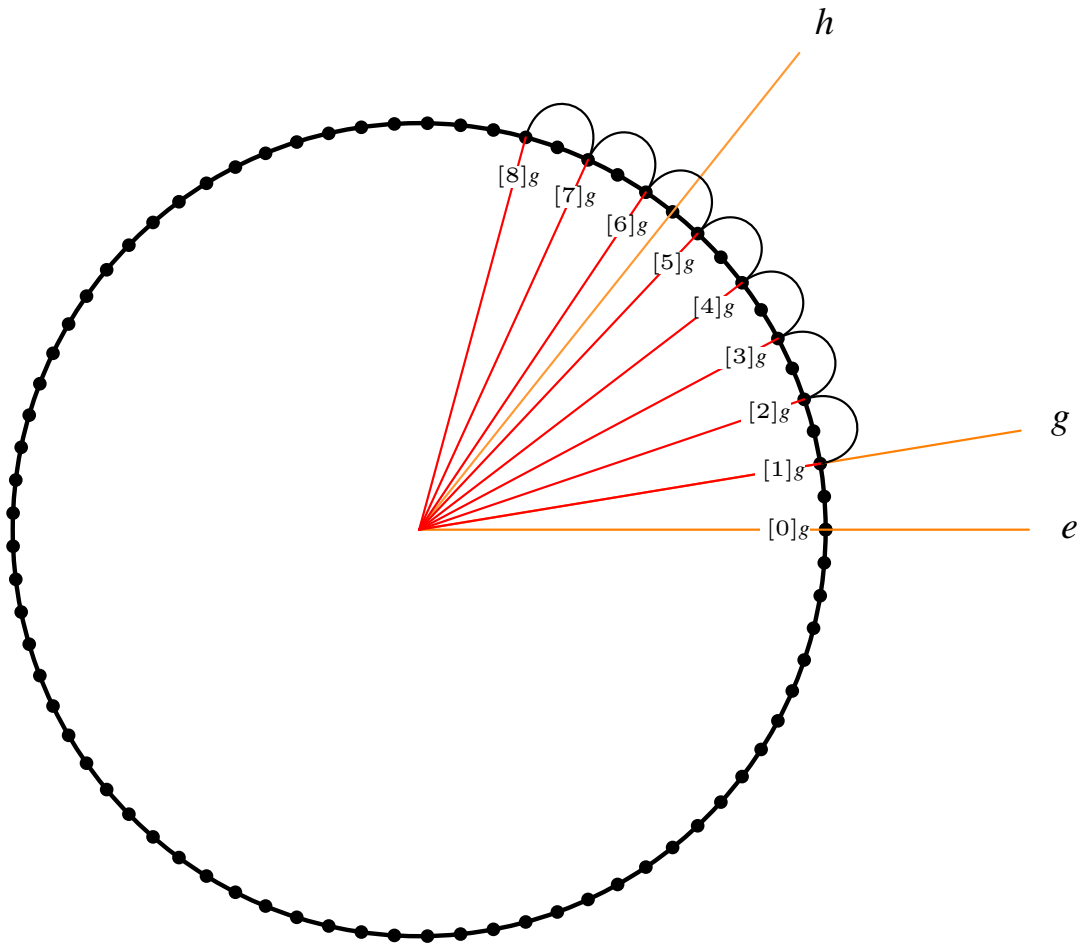
L'existence et l'unicité de la solution découle de l'existence et de l'unicité de (q, r) dans la division euclidienne (cf. TD)

BSGS : imagée dans $(\mathbb{Z}/77\mathbb{Z}, +)$ avec $g = 2$ et $h = 11$



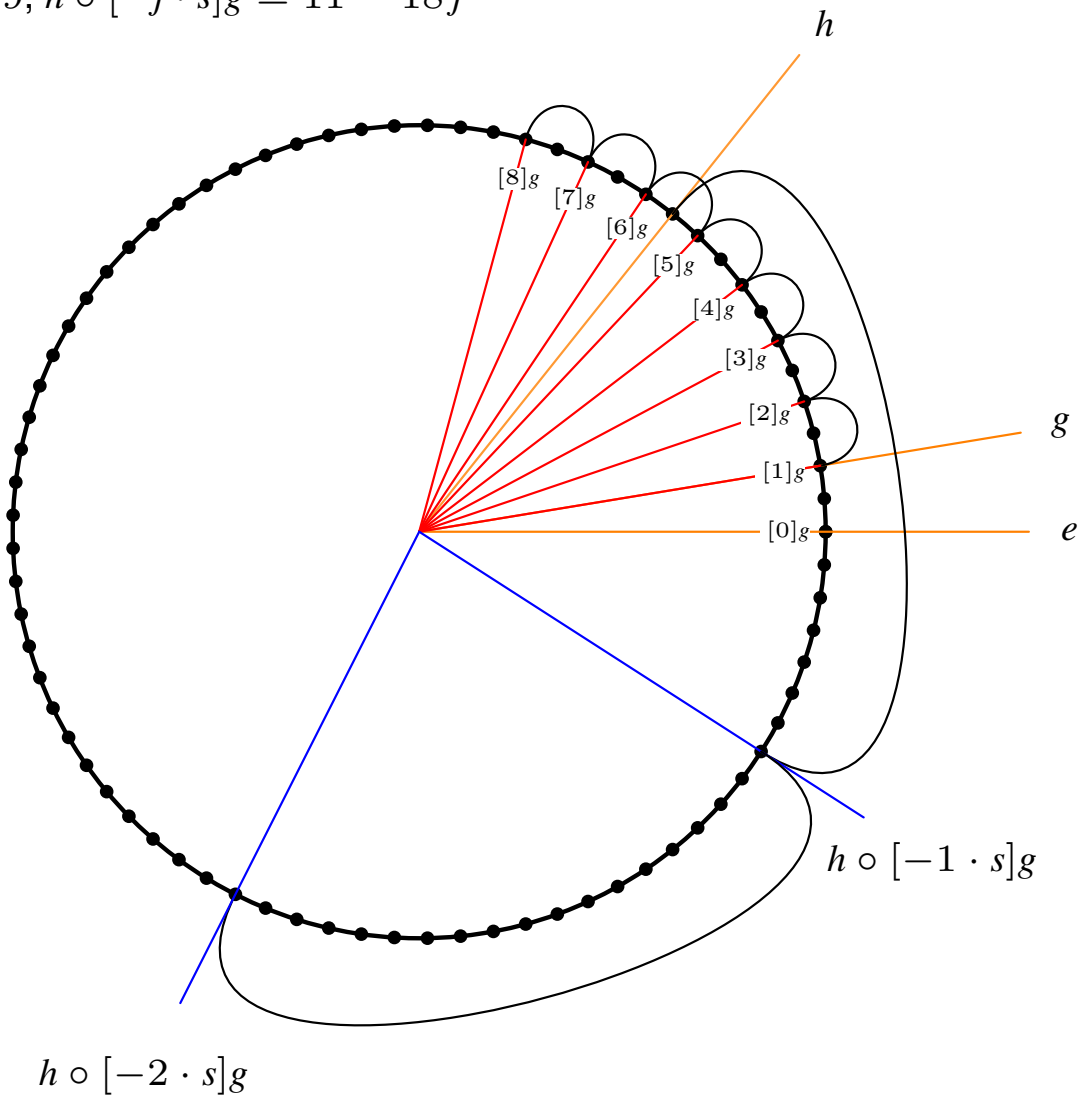
BSGS : imagée dans $(\mathbb{Z}/77\mathbb{Z}, +)$ avec $g = 2$ et $h = 11$

BS $[i]g = 2\,i$



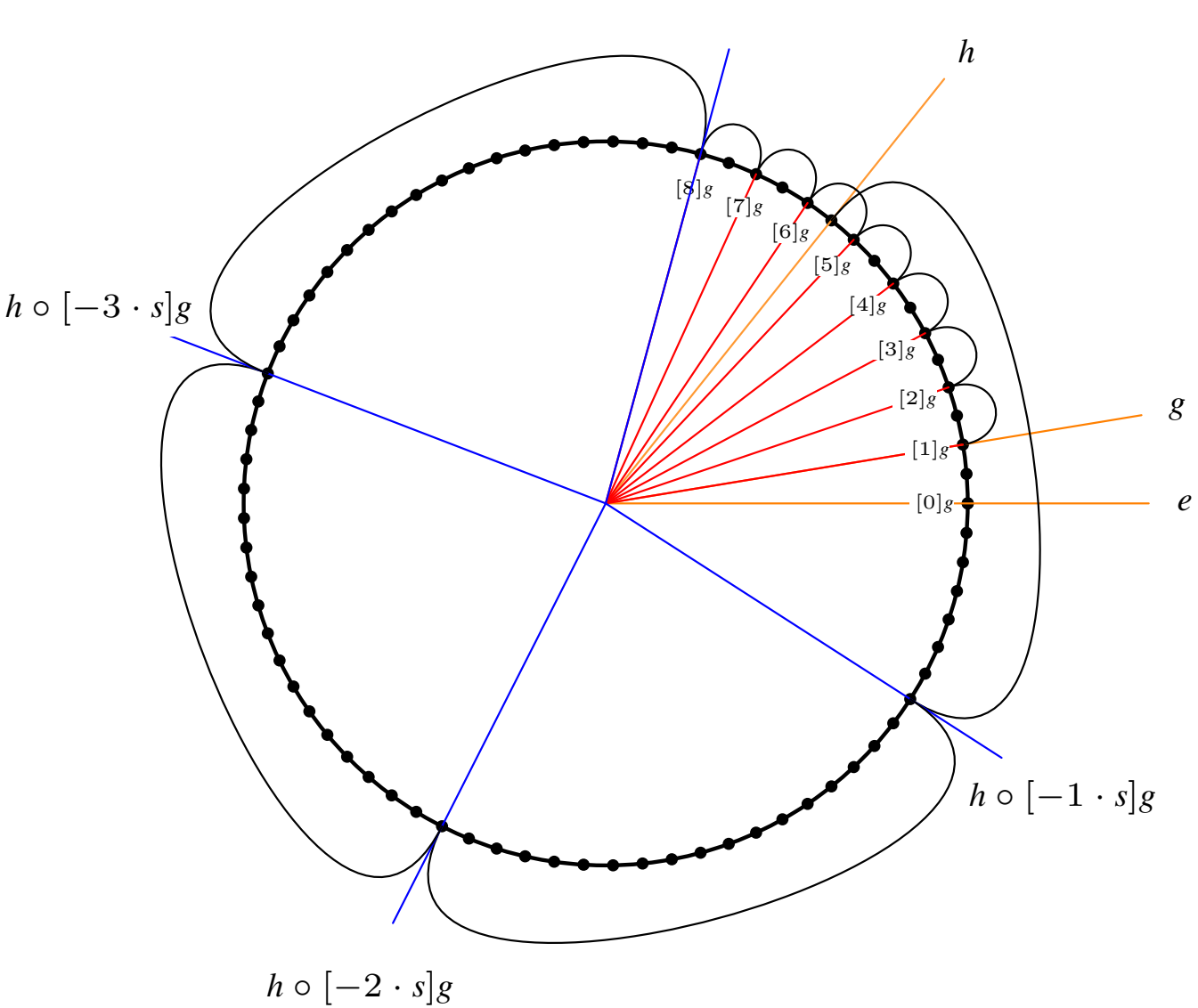
BSGS : imagée dans $(\mathbb{Z}/77\mathbb{Z}, +)$ avec $g = 2$ et $h = 11$

GS $s = \lfloor \sqrt{77} \rfloor + 1 = 9$, $h \circ [-j \cdot s]g = 11 - 18j$



BSGS : imagée dans $(\mathbb{Z}/77\mathbb{Z}, +)$ avec $g = 2$ et $h = 11$

$s = 9$, BS $[8]g = h \circ [-4 \cdot s]g$ GS,
$q = 4,\ r = 8,\ t = qs + r = 4 \times 9 + 8 = 44,\ 2 \times 44 = 11 \bmod 77$ dans le groupe additif $(\mathbb{Z}/77\mathbb{Z}, +)$



BSGS de Shanks dans $(\mathbb{Z}/809\mathbb{Z}^*, \times) \simeq (\mathbb{Z}/808\mathbb{Z}, +)$

On a $g = 3$ et $h = 525$, on veut calculer t tel que $g^t = h \bmod 809$, on prend $s = \lfloor \sqrt{808} \rfloor + 1 = 29$. On a $g^{-s} = 3^{-29} = 523 \bmod 809$.

Baby-steps : $(i, 3^i \bmod 809)$ $(i = 0, \ldots, 28)$:
(0, 1), (1, 3), (2, 9), (3, 27), (4, 81), (5, 243),
(6, 729), (7, 569), (8, 89), (9, 267), (10, 801), (11, 785),
(12, 737), (13, 593), (14, 161), (15, 483), (16, 640), (17, 302),
(18, 97), (19, 291), (20, 64), (21, 192), (22, 576), (23, 110),
(24, 330), (25, 181), (26, 543), (27, 11), (28, 33)

Giant-steps : $(j, 525 \times 523^j \bmod 809)$ $(j = 0, \ldots, 28)$:
(0, 525), (1, 324), (2, 371), (3, 682), (4, 726), (5, 277),
(6, 60), (7, 638), (8, 366), (9, 494), (10, 291), (11, 101),
(12, 238), (13, 697), (14, 481), (15, 773), (16, 588), (17, 104),
(18, 189), (19, 149), (20, 263), (21, 19), (22, 229), (23, 35),
(24, 507), (25, 618), (26, 423), (27, 372), (28, 396)

☞ $t = s \times 10 + 19 = 309$

BSGS de Shanks dans $(\mathbb{Z}/809\mathbb{Z}^*, \times) \simeq (\mathbb{Z}/808\mathbb{Z}, +)$

On a $g = 3$ et $h = 525$, on veut calculer t tel que $g^t = h \bmod 809$, on prend $s = \lfloor \sqrt{808} \rfloor + 1 = 29$. On a $g^{-s} = 3^{-29} = 523 \bmod 809$.

Baby-steps : $(i, 3^i \bmod 809)$ $(i = 0, \ldots, 28)$:
(0, 1), (1, 3), (2, 9), (3, 27), (4, 81), (5, 243),
(6, 729), (7, 569), (8, 89), (9, 267), (10, 801), (11, 785),
(12, 737), (13, 593), (14, 161), (15, 483), (16, 640), (17, 302),
(18, 97), (19, 291), (20, 64), (21, 192), (22, 576), (23, 110),
(24, 330), (25, 181), (26, 543), (27, 11), (28, 33)

Giant-steps : $(j, 525 \times 523^j \bmod 809)$ $(j = 0, \ldots, 28)$:
(0, 525), (1, 324), (2, 371), (3, 682), (4, 726), (5, 277),
(6, 60), (7, 638), (8, 366), (9, 494), (10, 291), (11, 101),
(12, 238), (13, 697), (14, 481), (15, 773), (16, 588), (17, 104),
(18, 189), (19, 149), (20, 263), (21, 19), (22, 229), (23, 35),
(24, 507), (25, 618), (26, 423), (27, 372), (28, 396)

☞ $t = s \times 10 + 19 = 309$

BSGS de Shanks dans $(\mathbb{Z}/809\mathbb{Z}^*, \times) \simeq (\mathbb{Z}/808\mathbb{Z}, +)$

On a $g = 3$ et $h = 525$, on veut calculer t tel que $g^t = h \bmod 809$, on prend $s = \lfloor \sqrt{808} \rfloor + 1 = 29$. On a $g^{-s} = 3^{-29} = 523 \bmod 809$.

Baby-steps : $(i, 3^i \bmod 809)$ $(i = 0, \ldots, 28)$:
(0, 1), (1, 3), (2, 9), (3, 27), (4, 81), (5, 243),
(6, 729), (7, 569), (8, 89), (9, 267), (10, 801), (11, 785),
(12, 737), (13, 593), (14, 161), (15, 483), (16, 640), (17, 302),
(18, 97), (19, 291), (20, 64), (21, 192), (22, 576), (23, 110),
(24, 330), (25, 181), (26, 543), (27, 11), (28, 33)

Giant-steps : $(j, 525 \times 523^j \bmod 809)$ $(j = 0, \ldots, 28)$:
(0, 525), (1, 324), (2, 371), (3, 682), (4, 726), (5, 277),
(6, 60), (7, 638), (8, 366), (9, 494), (10, 291), (11, 101),
(12, 238), (13, 697), (14, 481), (15, 773), (16, 588), (17, 104),
(18, 189), (19, 149), (20, 263), (21, 19), (22, 229), (23, 35),
(24, 507), (25, 618), (26, 423), (27, 372), (28, 396)

☞$t = s \times 10 + 19 = 309$

BSGS : table de hachage

Input : Le générateur g du groupe H d'ordre n et $h \in H \setminus \{e\}$.
Output : Un entier $0 < t < n$ tel que $h = [t]g$.

$s := \lfloor \sqrt{n} \rfloor + 1$;

for $i := 1$ **to** s **do**
    Hash$[[i]g]$:= i; // Table de hachage des Baby-Steps
end for;

$i := 0$; $\gamma := h$;
while true do //Giant-Steps
    if (Hash$[\gamma]$ == NULL) **then**
        $i := i + 1$
        $\gamma := \gamma \circ [-s]g$
    else
        return $i \times s +$ Hash$[\gamma]$;
    end if
end while

BSGS : complexités

☞ Complexité donnée en fonction de $\log |H| = \log n$

- Complexité calcul : $\mathcal{O}(s) = \mathcal{O}(\sqrt{n})$
- Complexité mémoire : $\mathcal{O}(\sqrt{n})$

Remarques

- Le premier algorithme générique de complexité optimale
- Nécessite une place mémoire de taille $\sqrt{n}$ pour la table de hachage
- Nécessite la connaissance de $n = |H|$ mais on peut se contenter d'une surestimation du cardinal de H
- Il existe des variantes de cet algorithme qui permettent de diminuer la complexité en mémoire mais pas en temps de calcul (rho de Pollard par exemple)

Conclusion : complexité DLP groupe générique

☞ Le BSGS est optimal !
☞ De plus, on peut utiliser la décomposition de n pour résoudre le problème encore plus efficacement (attaque basée sur le CRT, voir plus loin).

Pohlig-Hellman

Dans le cas d'un groupe générique H d'ordre $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$ il est possible de résoudre le DLP dans H en $\mathcal{O}(\sum e_i \sqrt{p_i})$ calculs dans G.

Corollaire important

Si l'on souhaite que le DLP soit difficile, il faut choisir un groupe H d'ordre possédant au moins un très grand facteur premier.

Échange DH en pratique

Création du groupe support au DLP (g et p)

```
% openssl genpkey -genparam -algorithm DH -out dhp.pem
.....................................+.....+.....................
.....................................................................
.....................................................+...........
.............................+.....................................+..
..................+........+.....................................
......+.....................+.............................+.........
.....................................................................
.......................+.....................+........+.............
.......................+............+.........................+.....
.....................................+.....................
.....................................................................
.............+.................................+..............+.........
.....................................................................
..............++*++++++*
```

À quoi ressemble ce groupe ?

```
% cat dhp.pem
-----BEGIN DH PARAMETERS-----
MIGHAoGBAP5tnrFgZBE+Rsm5NsQwBjd3O7xvMnogMn9eHIaZ5ktCJJ4xfxFDT+fN
KZZwusYBoR+5a1ctjsPaQZ4mRdxrtwiaHW72B/cAl/joyYo47DUQ40BuLjYGHNXw
+zm8IkzJHnih76J6fRDVRUH6UW1QDrSRWjplXLoc+Kpxhq7rGeLTAgEC
-----END DH PARAMETERS-----
```

Échange DH en pratique

Meilleur affichage

```
% openssl pkeyparam -in dhp.pem -text
-----BEGIN DH PARAMETERS-----
MIGHAoGBAP5tnrFgZBE+Rsm5NsQwBjd3O7xvMnogMn9eHIaZ5ktCJJ4xfxFDT+fN
KZZwusYBoR+5a1ctjsPaQZ4mRdxrtwiaHW72B/cAl/joyYo47DUQ40BuLjYGHNXw
+zm8IkzJHnih76J6fRDVRUH6UW1QDrSRWjplXLoc+Kpxhq7rGeLTAgEC
-----END DH PARAMETERS-----
DH Parameters: (1024 bit)
prime:                                                    p
00:fe:6d:9e:b1:60:64:11:3e:46:c9:b9:36:c4:30:
06:37:77:3b:bc:6f:32:7a:20:32:7f:5e:1c:86:99:
e6:4b:42:24:9e:31:7f:11:43:4f:e7:cd:29:96:70:
ba:c6:01:a1:1f:b9:6b:57:2d:8e:c3:da:41:9e:26:
45:dc:6b:b7:08:9a:1d:6e:f6:07:f7:00:97:f8:e8:
c9:8a:38:ec:35:10:e3:40:6e:2e:36:06:1c:d5:f0:
fb:39:bc:22:4c:c9:1e:78:a1:ef:a2:7a:7d:10:d5:
45:41:fa:51:6d:50:0e:b4:91:5a:3a:65:5c:ba:1c:
f8:aa:71:86:ae:eb:19:e2:d3
generator: 2 (0x2)                                        g
```

Échange DH en pratique

Création des clés privées/publiques à partir de ce groupe

À partir de `dhp.pem` Alice crée ses clés a et A

```
% openssl genpkey -paramfile dhp.pem -out dhkeyA.pem
% openssl pkey -in dhkeyA.pem -text -noout
DH Private-Key: (1024 bit)
private-key:                                          a
4e:51:8f:ae:e6:18:8c:32:58:18:db:35:8d:15:ef:
a1:64:c7:b4:de:ae:99:46:c0:c9:73:00:91:5e:af:
[...]
public-key:                                           A
00:a4:64:57:53:88:51:35:60:67:70:03:0c:e0:d9:
65:4e:96:a1:b0:2c:64:e9:89:bf:7b:4f:98:61:2d:
[...]
prime:                                                p
00:fe:6d:9e:b1:60:64:11:3e:46:c9:b9:36:c4:30:
06:37:77:3b:bc:6f:32:7a:20:32:7f:5e:1c:86:99:
[...]
generator: 2 (0x2)                                    g
```

et Bob fait de même de son côté après avoir reçu `dhp.pem`

```
% openssl genpkey -paramfile dhp.pem -out dhkeyB.pem
```

Échange DH en pratique

Chacun extrait sa clé publique pour l'envoyer à l'autre

Alice envoie A à Bob

```
% openssl pkey -in dhkeyA.pem -pubout -out dhpubA.pem
% openssl pkey -pubin -in dhpubA.pem -text
-----BEGIN PUBLIC KEY-----
MIIBIDCBlQYJKoZIhvcNAQMBMIGHAoGBAP5tnrFgZBE+Rsm5NsQwBjd3O7xvMnog
Mn9eHIaZ5ktCJJ4xfxFDT+fNKZZwusYBoR+5a1ctjsPaQZ4mRdxrtwiaHW72B/cA
l/joyYo47DUQ40BuLjYGHNXw+zm8IkzJHnih76J6fRDVRUH6UW1QDrSRWjplXLoc
+Kpxhq7rGeLTAgECA4GFAAKBgQCkZFdTiFE1YGdwAwzg2WVOlqGwLGTpib97T5hh
LUX3SwJDy4ermJaui93EAfDKFInHVN55DxSh1Ye3l3NMPZK6ppdUCDJFGzLZ/4Z0
QJl3fK6kxbfH1nqwrIP5tVYy1pA5MMzrC2I88SWXzNHUj9eySque7b3V93ZMpX9J
wOJyUQ==
-----END PUBLIC KEY-----
DH Public-Key: (1024 bit)
public-key:                                          A
00:a4:64:57:53:88:51:35:60:67:70:03:0c:e0:d9:
65:4e:96:a1:b0:2c:64:e9:89:bf:7b:4f:98:61:2d:
[...]
prime:                                               p
00:fe:6d:9e:b1:60:64:11:3e:46:c9:b9:36:c4:30:
06:37:77:3b:bc:6f:32:7a:20:32:7f:5e:1c:86:99:
[...]
generator: 2 (0x2)                                   g
```

et Bob fait de même de son côté

```
% openssl pkey -in dhkeyB.pem -pubout -out dhpubB.pem
```

Échange DH en pratique

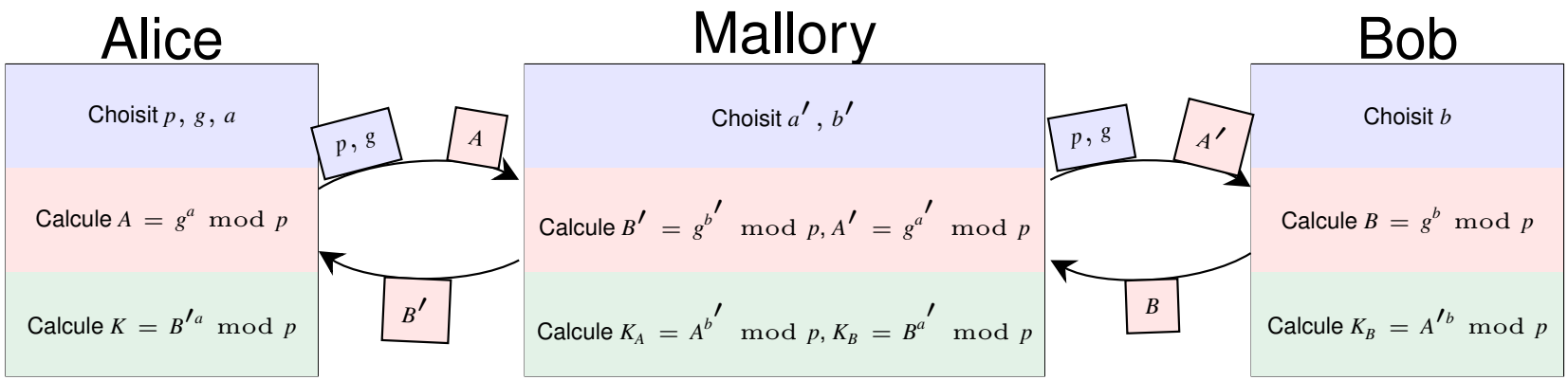
Alice construit la clé partagée K à partir de la clé publique B envoyée par Bob et de sa clé privée a

```
% openssl pkeyutl -derive -inkey dhkeyA.pem -peerkey dhpubB.pem -out
    secret.bin
% xxd secret.bin
00000000: 0281 569b 265a 20da 1910 47a5 dc88 7290  ..V.&Z ...G...r.
00000010: 6f68 418a 3cb0 bdb5 c95d c39b c0ce 5518  ohA.<....]....U.
00000020: c31f d193 aa71 9b6d 1ad6 b95d 4922 3412  .....q.m...]I"4.
00000030: 72f5 9fd0 0a4c abdd 34ad 45d4 9740 1ee7  r....L..4.E..@..
00000040: 179b b2f2 5958 7def 2ee0 e38e 728c f4c9  ....YX}.....r...
00000050: 84c6 1bfa 20b3 b856 b979 d000 343b f502  .... ..V.y..4;..
00000060: 15b5 6df5 55d6 b551 a908 f7a9 7368 2200  ..m.U..Q....sh".
00000070: 4aaf a362 98e9 2bbe cbd7 69ef 1351 470b  J..b..+...i..QG.
```

Bob fait de même de son côté et il doit obtenir la même clé !

Plan

Attaque de l'échange de clé Diffie-Hellman-Merkle



Mallory communique avec Alice avec la clé partagée $K_A = A^{b'} \mod p = B'^a \mod p$,
Mallory communique avec Bob avec la clé partagée $K_B = A'^b \mod p = B^{a'} \mod p$.

Man in the Middle

Mallory peut

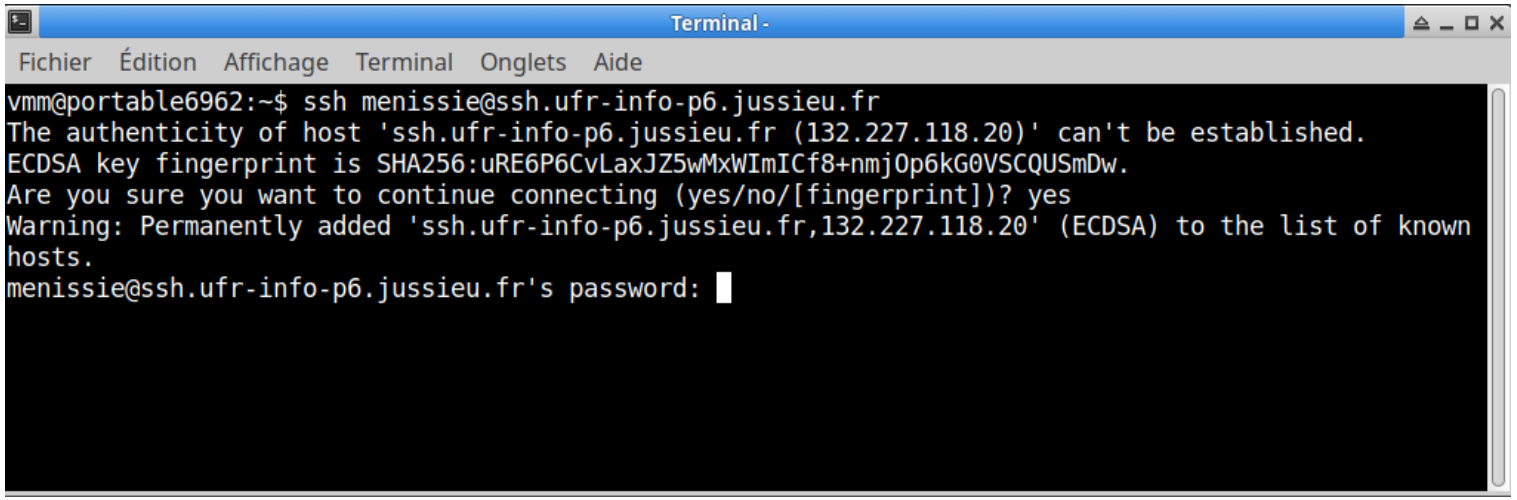
- se contenter d'écouter les conversations d'Alice et Bob
- mais surtout se faire passer pour l'un auprès de l'autre.

Comment y remédier ? Il faut qu'Alice et Bob aient un moyen d'être sûrs de communiquer l'un avec l'autre : l'authentification mais ils ne se connaissent pas encore. . .

Exemple historique (cf. cours 0) : homme d'Élisabeth I^re qui s'immisce dans une discussion entre Marie Stuart et un chevalier fidèle à sa cause et qui provoquera leur perte.

Partager une clé commune

Avec un interlocuteur dont on n'est pas sûr de l'identité



Autre solution ?

Troisième partie III

Chiffrement à Clé Publique

Plan

Idée Générale : La trappe de Diffie-Hellman-Merkle

☞Outre le problème de l'échange des clés privées, le nombre croissant d'échanges augmente le nombre de clés secrètes nécessaires ce qui rajoute un nouveau problème.

☞En 1975 Diffie, Hellman et Merkle proposent les grandes lignes de leur idée théorique pour éviter ce problème.

Définition d'une fonction avec trappe

Une fonction $F : X \to Y$ est à sens unique avec trappe, si :

- F est à sens unique, i.e. elle est difficile à inverser, facile à calculer.
- La connaissance d'une information supplémentaire, appelée la trappe, rend, pour tout $y \in \mathrm{Im}(F)$, le calcul de $x \in X$ tel que $F(x) = y$ réalisable en temps polynomial.

Clé publique : F permet de chiffrer

Clé secrète : la trappe permet de déchiffrer

Premier exemple de chiffrement à clé publique

ElGamal 1985

Ce cryptosystème est basé sur le DLP.

- L'espace des messages est représenté par le groupe $(G, \circ)$.
- Alice tire aléatoirement k et calcule $h = [k]g$
- Alice publie $H = \langle g \rangle$ et h et conserve secrètement k
- Chiffrement : Pour envoyer $x \in G$ à Alice, Bob choisit $t \in \{1, \ldots, |H|\}$ aléatoirement et ne le publie pas. Il calcule

$$y_1 = [t]g \text{ et } y_2 = x \circ [t]h$$

  et envoie le couple (y_1, y_2) à Alice.
- Déchiffrement : Alice calcule $y_2 \circ [-1]([k]y_1)$ et retrouve x.

☞Le DLP est supposé difficile dans $(G, \circ)$

Equivalence DLP, Diffie-Hellman-Merkle, ElGamal ?

Attention !

Le DLP est, a priori, plus fort que les problèmes de Diffie-Hellman-Merkle et ElGamal.

- La résolution du DLP entraine la résolution de Diffie-Hellman-Merkle
- La résolution du DLP entraine la résolution de ElGamal
- Aucune des deux réciproques n'est connue !

☞Il existe peut être des algorithmes qui résolvent plus efficacement Diffie-Hellman-Merkle et/ou ElGamal que le DLP !

ElGamal sur les corps finis

Exemple d'échange ElGamal :
Alice choisit $G = H = \mathbb{Z}/p\mathbb{Z}^*$ avec $p = 2579$ et $g = 2$ qu'elle publie. Elle garde secret $k = 765$ et publie aussi

$$h = 2^{765} \bmod 2579 = 949$$

Bob veut transmettre $x = 1299$ à Alice. Il choisit t aléatoirement $t = 853$ et calcule

$$y_1 = [t]g = 2^{853} \bmod p = 435 \text{ puis}$$
$$y_2 = x \circ [t]h = 1299 \times (949^{853}) \bmod p = 2396$$

et les envoie à Alice. Elle reçoit donc $(435, 2396)$ et calcule alors :

$$x = y_2 \circ [-k]y_1 = 2396 \times (435^{765})^{-1} \bmod 2579 = 1299$$

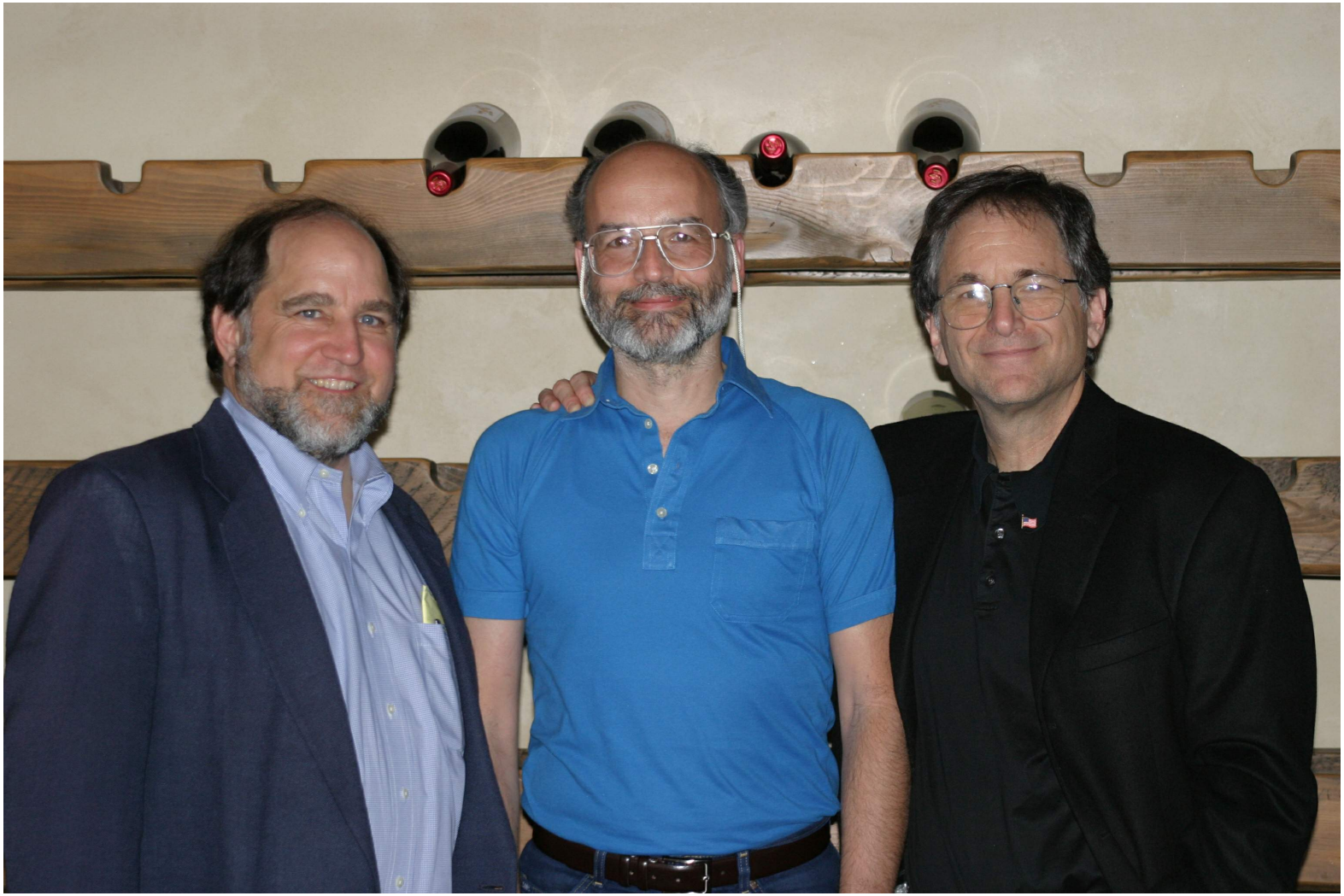
pour retrouver le message de Bob !

Plan

Rivest, Shamir, Adleman

☞1977 première instance d'un cryptosystème à clé publique : RSA.

Rivest, Shamir, Adleman

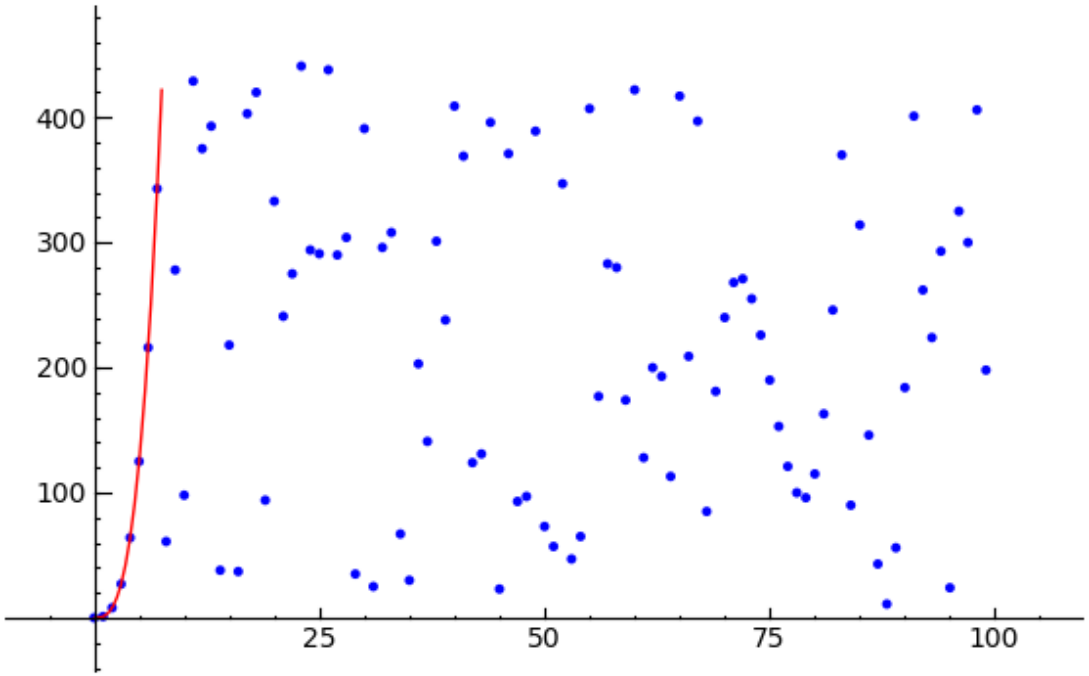


Rivest, Shamir, Adleman

RSA

Basé sur la difficulté de calculer une racine e-ième $\mod$ un entier N de factorisation inconnue

Si on sait factoriser facilement alors ce problème se résout tout aussi facilement. Mais ATTENTION ce ne sont pas des problèmes polynomialement équivalents.



Rivest, Shamir, Adleman

RSA

Soient p et q deux nombres premiers et $n = pq$. On choisit deux entiers e, d tels que $ed = 1 \bmod \varphi(n)$. L'application

$$F_{(n,e)} : \mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}/n\mathbb{Z} \text{ définie par } x \mapsto x^e \bmod n$$

- F est bien à sens unique, difficulté basée sur le calcul de la racine e-ieme modulaire.
- La trappe est la connaissance de d (ou de manière équivalente p et q).

☞on chiffre en faisant $x^e \bmod n$ on déchiffre en calculant $y^d \bmod n$

Preuve : On a $|\mathbb{Z}/n\mathbb{Z}^\times| = \varphi(n) = (p-1)(q-1) \Rightarrow ed = 1 + k \times \varphi(n) \ldots$

Les messages qui ne sont pas inversibles mod n sont dangereux !

Plan

Signer un message / Authentification

☞La signature permet d'authentifier l'expéditeur d'un message

Problème de l'authentification

Étant donné un message m et sa signature s, on doit être capable de vérifier l'expéditeur de m à partir de données publiques.

☞Les problèmes RSA et DLP peuvent être utilisés à cet usage.

Signer avec RSA

☞La signature permet d'authentifier l'expéditeur d'un message

La signature RSA

Soit $N = pq$ un entier RSA et (e, d) un couple d'exposants de chiffrement/déchiffrement choisis par l'expéditeur.

- L'expéditeur publie N et d
- Un message $m \in (\mathbb{Z}/N\mathbb{Z})^\times$ envoyé par l'expéditeur aura pour signature

$$s_m = m^e \mod N$$

- Le destinataire recevra le couple (m, s) et authentifiera l'expéditeur à partir ses clés publiques (N, d) en vérifiant

$$m = s_m^d \mod N$$

C'est l'acte de chiffrer qui devient la brique secrète dans le cadre de la signature.

Fonctions de hachage

Problème signature RSA : la signature est de la taille du message !

Fonction de hachage cryptographique

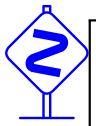
Une fonction de $\{0,1\}^*$ dans $[0, 2^t - 1]$ (avec $t = 160, 256, 512$ etc) est dite de hachage si elle vérifie

- difficile à inverser (construire un message ayant une signature donnée)
- difficile de reconstruire un second message de même signature (seconde préimage)
- la probabilité de trouver deux messages de même signature est très faible (collision)

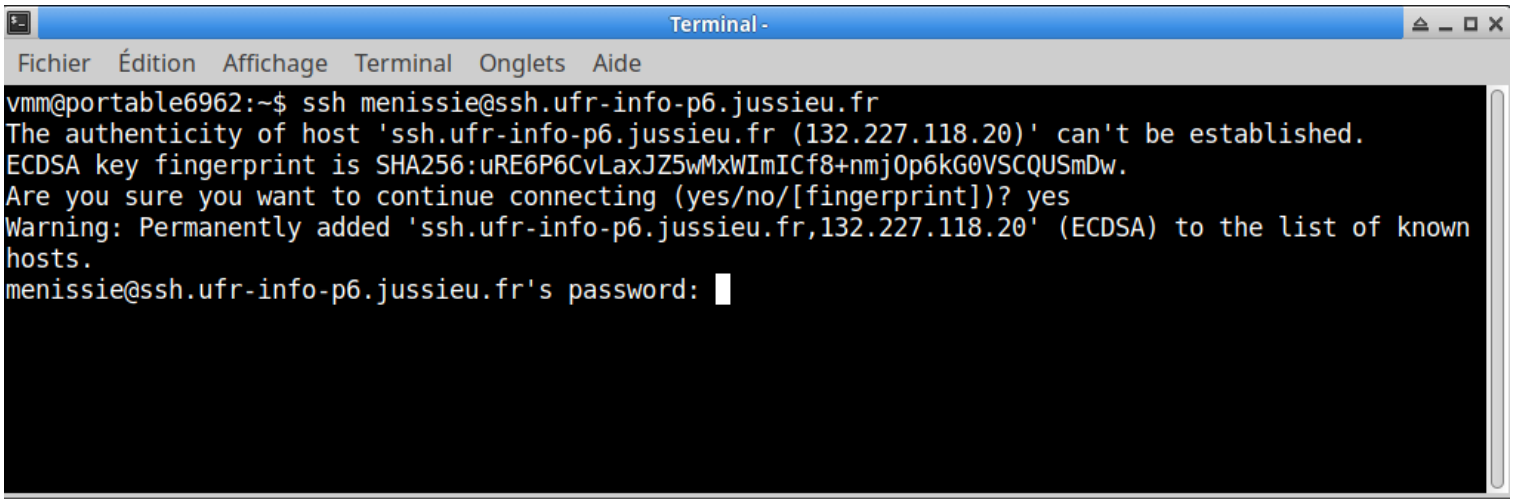
Exemples : MD5, SHA1, SHA2 (SHA-256, SHA-512), SHA3,. . .

☞Permet de vérifier l'intégrité d'un fichier (paquets linux par exemple)
☞Plutôt que signer un message, on signe son empreinte après application d'une fonction de hachage.

Échange de clé et Authentification

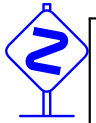


Comment authentifier son interlocuteur au moment du premier échange ?

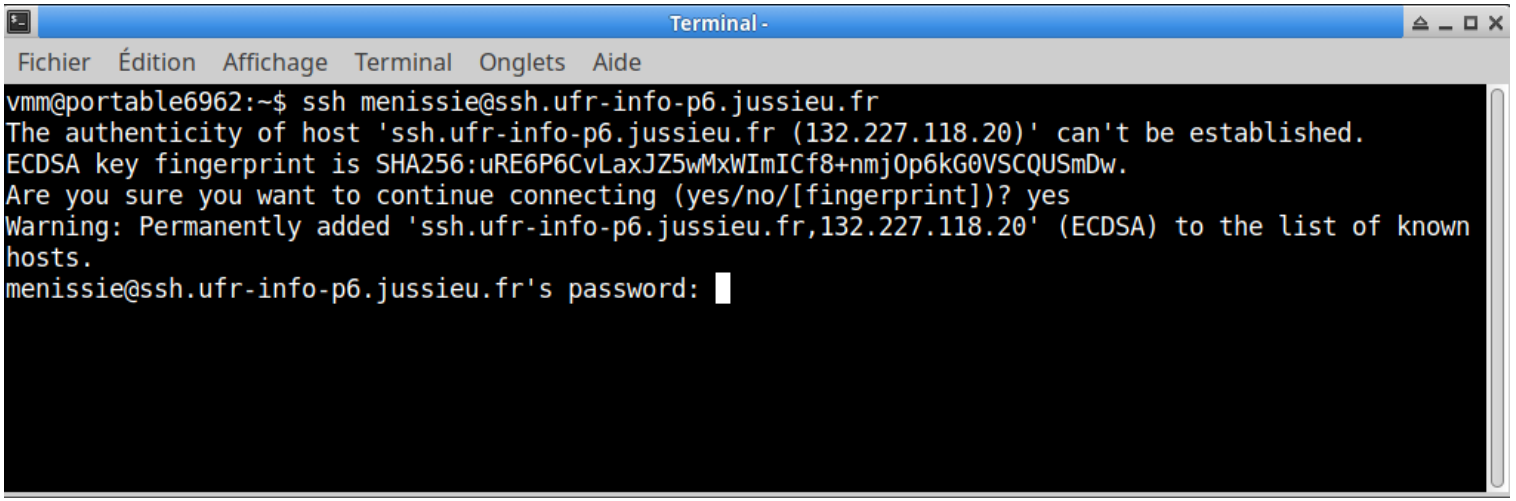


Autre solution ?

Échange de clé et Authentification

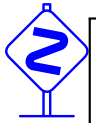


Comment authentifier son interlocuteur au moment du premier échange ?

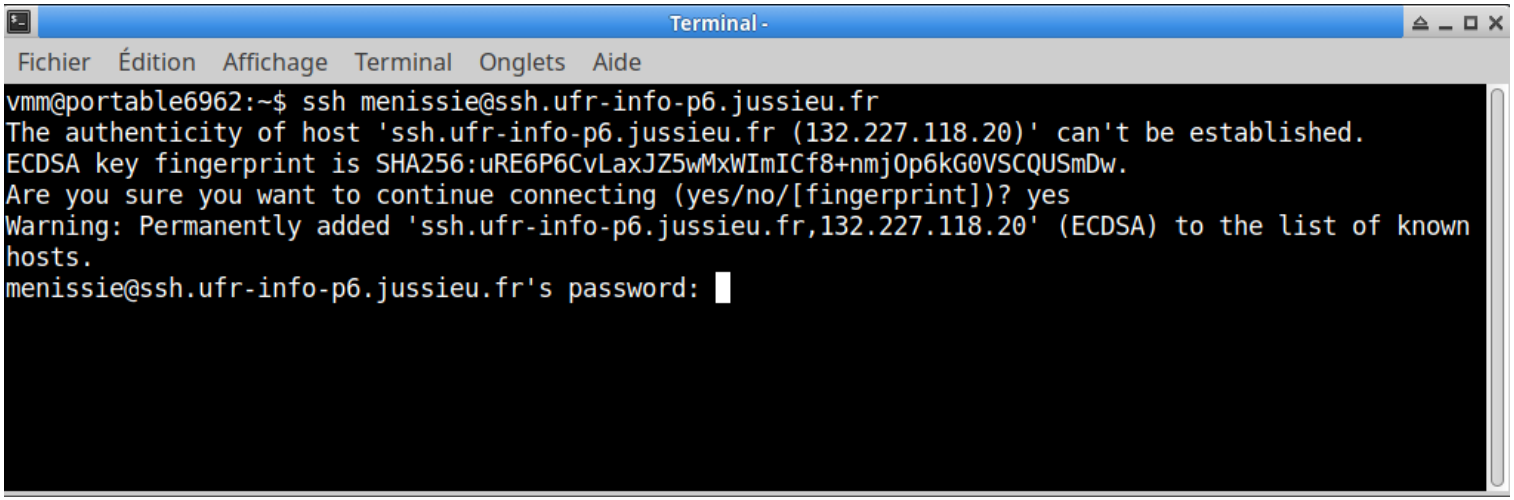


Autre solution ?
Signer les clés publiques ?

Échange de clé et Authentification



Comment authentifier son interlocuteur au moment du premier échange ?



Autre solution ?

Repousse le problème à l'authentification des clés publiques qui permettent de signer !?!!?

Plan

RSA : implémentation chiffrement/vérif. signature

☞Exponentation modulaire

Square and Multiply

Basé sur l'écriture binaire de l'exposant $e = \sum_{i=0}^{l-1} e_i 2^i$:

$$x^e = x^{\sum_{i=0}^{l-1} e_i 2^i} = \prod_{i=0}^{l-1} x^{e_i 2^i}$$

☞On réduit la complexité de l'exponentiation modulo n à $\mathcal{O}((\log n)^3)$ (voir cours précédent).

RSA : implémentation déchiffrement/signature

☞dans ce cas on connaît p et q, idée couper les calculs en deux !

RSA et CRT1

On pré-calcule les valeurs :

- $u_p = q \times (q^{-1} \bmod p) \bmod n$
- $u_q = p \times (p^{-1} \bmod q) \bmod n$
- $d_p = d \bmod (p-1)$
- $d_q = d \bmod (q-1)$

Pour déchiffrer y et obtenir le message x d'origine on utilise le CRT :

1. $x_p = y^{d_p} \bmod p$
2. $x_q = y^{d_q} \bmod q$
3. $x = x_p u_p + x_q u_q \bmod n$

Théorème chinois des restes

Théorème

Soit m_1 et m_2 deux éléments de A un anneau euclidien. Si $\mathrm{pgcd}(m_1, m_2) = 1$ alors nous avons l'isomorphisme d'anneau

$$A/m_1A \times A/m_2A \simeq A/(m_1m_2)A$$

Démonstration existentielle (à lire chez vous) :
Le morphisme d'anneau

$$\begin{array}{rcl} \phi : A & \longrightarrow & A/m_1A \times A/m_2A \\ e & \longmapsto & (e \bmod m_1, e \bmod m_2) \end{array}$$

est surjectif.

Théorème chinois des restes

Théorème

Soit m_1 et m_2 deux éléments de A un anneau euclidien. Si $\mathrm{pgcd}(m_1, m_2) = 1$ alors nous avons l'isomorphisme d'anneau

$$A/m_1A \times A/m_2A \simeq A/(m_1m_2)A$$

Démonstration existentielle (à lire chez vous) :
Le noyau de ϕ est donné par

$$\{e \in A \text{ tq } e = 0 \bmod m_2 \text{ et } e = 0 \bmod m_2\}$$

ainsi le noyau est l'ensemble des multiples de m_1 et m_2. Comme $\mathrm{pgcd}(m_1, m_2) = 1$ on obtient le résultat.

Théorème chinois des restes

Théorème

Soit m_1 et m_2 deux éléments de A un anneau euclidien. Si $\mathrm{pgcd}(m_1, m_2) = 1$ alors nous avons l'isomorphisme d'anneau

$$A/m_1A \times A/m_2A \simeq A/(m_1m_2)A$$

Démonstration constructive :
Comment reconstruire l'élément e dans $A/(m_1m_2)A$ à partir de son image (e_1, e_2) dans $A/m_1A \times A/m_2A$? (l'autre direction est triviale)

Théorème chinois des restes

Théorème

Soit m_1 et m_2 deux éléments de A un anneau euclidien. Si $\mathrm{pgcd}(m_1, m_2) = 1$ alors nous avons l'isomorphisme d'anneau

$$A/m_1A \times A/m_2A \simeq A/(m_1m_2)A$$

Démonstration constructive :
Comment reconstruire l'élément e dans $A/(m_1m_2)A$ à partir de son image (e_1, e_2) dans $A/m_1A \times A/m_2A$? (l'autre direction est triviale)

Pour cela, nous utilisons la relation de Bézout entre m_1 et m_2 :

$$um_1 + vm_2 = 1$$

Théorème chinois des restes

Théorème

Soit m_1 et m_2 deux éléments de A un anneau euclidien. Si $\mathrm{pgcd}(m_1, m_2) = 1$ alors nous avons l'isomorphisme d'anneau

$$A/m_1A \times A/m_2A \simeq A/(m_1m_2)A$$

Démonstration constructive :
Comment reconstruire l'élément e dans $A/(m_1m_2)A$ à partir de son image (e_1, e_2) dans $A/m_1A \times A/m_2A$? (l'autre direction est triviale)

En notant $M = m_1m_2$ il vient alors

$$e = e_1vm_2 + e_2um_1 \bmod M$$

Théorème chinois des restes : algorithme général

On applique récurssivement le procédé de résolution sur les équations du système. Il faut que $\mathrm{pgcd}(m_i, m_j) = 1$ pour $i \neq j$.

$$\left\{\begin{array}{lcll} x & = & e_1 & \bmod m_1 \\ x & = & e_2 & \bmod m_2 \\ & & \vdots & \\ x & = & e_k & \bmod m_k \end{array}\right\} \xrightarrow{\text{CRT}} \left\{\begin{array}{lcll} \boxed{x} & \boxed{=} & \boxed{c} & \boxed{\bmod M = m_1 m_2} \\ x & = & e_3 & \bmod m_3 \\ & & \vdots & \\ x & = & e_k & \bmod m_k \end{array}\right.$$

Théorème chinois des restes : exemple sur $\mathbb{Z}$

Soit à résoudre le système

$$\begin{cases} x &= 2 \bmod 3 \\ x &= 3 \bmod 7 \\ x &= 8 \bmod 19 \end{cases}$$

Les entiers 3, 7 et 19 sont premiers deux à deux. CRT possible.

Initialisation :

- $c \leftarrow 2$
- $M \leftarrow 3$

Calculs :

- $c \leftarrow c \times u \times m_i + e_i \times v \times M$
- $M \leftarrow m_i \times M$
- $c \leftarrow c \bmod M$

avec $um_i + vM = 1$ (relation de Bézout) :

i	e_i	m_i	u	v	c	M	$c \bmod M$
1					2	3	
2	3	7	1	-2	$2 \times 1 \times 7 + 3 \times -2 \times 3 = -4$	$7 \times 3 = 21$	17
3	8	19	10	-9	$17 \times 19 \times 10 + 8 \times -9 \times 21 = 1718$	$19 \times 21 = 399$	122

Les solutions sont de la forme $122 + 399k$ pour $k \in \mathbb{Z}$.

CRT et RSA avec p et q de même taille

Lors du déchiffrement, on connaît p et q et l'on doit calculer

$$x = y^d \bmod n$$

Cette opération prend $\simeq \log d (\log n)^2 \simeq (\log n)^3$ mais on ne se sert pas de la connaissance de p et q !

Mais d'après le CRT on a

$$\mathbb{Z}/n\mathbb{Z} \simeq \mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$$

CRT et RSA avec p et q de même taille

Mais d'après le CRT on a

$$\mathbb{Z}/n\mathbb{Z} \simeq \mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$$

On peut donc faire les calculs à droite :

$$x_p = y^d \bmod p = y^{d \bmod (p-1)} \bmod p \text{ et } x_q = y^{d \bmod (q-1)} \bmod q$$

et on reconstruit la solution modulo n :

$$x = x_p u_p + x_q u_q \bmod n$$

Ici on fait $\simeq 2 \times (\log \sqrt{n})^3 \simeq \frac{1}{4}(\log n)^3$ pour le calcul des exponentiations et $2 \times \log(\sqrt{n})^2 + \log n$ pour la reconstruction.

☞$(\log n)^3 \gg \frac{1}{4}(\log n)^3$

☞Temps d'execution de 2 à 4 fois plus rapide avec le CRT !

Plan

Analyse des Cryptosystèmes

Pourquoi la cryptanalyse ? (rappel)

Valider la sécurité des cryptosystèmes

Établir des standards

Retirer de la circulation des cryptosystèmes/paramètres peu sûrs

Comment ?

Math./Algo. où l'on cherche à "*casser*" les mathématiques sous-jacentes (algorithmique, complexité)

Protocole où l'on cherche une faille dans l'utilisation d'un cryptosystème en pratique

Canaux auxiliaires = Side Channels où l'on cherche à casser le cryptosystème par des biais physiques

Rivest, Shamir, Adleman (RSA)

Rappel

RSA

Soient p et q deux nombres premiers et $n = pq$ et deux entiers e, d tels que $ed = 1 \mod \varphi(n)$. L'application $F_{(n,e)} : \mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}/n\mathbb{Z}$ qui à un élément x fait correspondre $x^e \mod n$.

- F est bien à sens unique, difficulté basée sur la factorisation de n.
- La trappe est la connaissance de p,q et d.

- Chiffrement x : $x^e \mod n$
- Déchiffrement y : $y^d \mod n$

Casser complètement RSA

Problème RSA

Étant donné un entier RSA $n = pq$ $(p < q < 2p)$ on demande de retrouver la clé secrète d

Ce problème est plus difficile que celui où l'on demande de décrypter un message chiffré donné. Ici la résolution de ce problème permet de décrypter n'importe quel message chiffré.

Factorisation et RSA

☞$n = pq$ avec p et q deux premiers distincts.

Lemme

Si l'on connaît la factorisation de n alors on casse complètement RSA.

Preuve

À partir de $n = pq$ on peut calculer $\varphi(n) = (p-1)(q-1)$. À l'aide de l'algorithme d'Euclide étendue (complexité polynomiale) on calcule facilement une relation de Bézout entre e et $\varphi(n)$.
On retrouve alors l'inverse d de e modulo $\varphi(n)$.

☞On peut montrer la réciproque de manière probabiliste.

Il est peut être possible de décrypter un message chiffré donné sans attaquer la clé secrète d ou la factorisation de n, ceci est une question ouverte !

Factorisation : Fermat 160 ?-1665

☞$n = pq$ avec p et q deux premiers de même taille $p < q < 2p$.

Lemme

Si (x, y) est un couple d'entiers vérifiant $x^2 - y^2 = n$ alors $n = (x - y)(x + y)$.

☞L'algorithme peut échouer !

Fermat

Il testait les entiers $x = (\lfloor \sqrt{n} \rfloor + i)$ pour $i = 1, 2, \ldots$ jusqu'à ce que $x^2 - n$ soit un carré parfait. Il développa une méthode pour tester si un entier est un carré parfait basée sur des calculs modulaires ! (voir TD)

☞La complexité reste exponentielle mais est plus rapide que de tester brutalement si les facteurs p et q sont très proches l'un de l'autre.

Factorisation : Fermat 160 ?-1665

Lemme

Si x, y sont des entiers tels que $x^2 - y^2 = n$, alors $n = (x - y)(x + y)$.

$n = 28$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 8$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 21$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 36 = 6^2$ donc $n = (x_3 - 6)(x_3 + 6) = 2 \times 14$.

☞L'algorithme peut échouer !

$n = 30$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 6$ qui n'est pas un carré parfait ;

Factorisation : Fermat 160 ?-1665

Lemme

Si x, y sont des entiers tels que $x^2 - y^2 = n$, alors $n = (x - y)(x + y)$.

$n = 28$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 8$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 21$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 36 = 6^2$ donc $n = (x_3 - 6)(x_3 + 6) = 2 \times 14$.

☞L'algorithme peut échouer !

$n = 30$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 6$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 19$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 34$ qui n'est pas un carré parfait ;
4. $x_4 = 9$, $x_4^2 - n = 51$ qui n'est pas un carré parfait ; $\sqrt{x_4^2 - n} \approx 7.14$

Factorisation : Fermat 160 ?-1665

Lemme

Si x, y sont des entiers tels que $x^2 - y^2 = n$, alors $n = (x - y)(x + y)$.

$n = 28$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 8$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 21$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 36 = 6^2$ donc $n = (x_3 - 6)(x_3 + 6) = 2 \times 14$.

☞L'algorithme peut échouer !

$n = 30$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 6$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 19$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 34$ qui n'est pas un carré parfait ;
4. $x_4 = 9$, $x_4^2 - n = 51$ qui n'est pas un carré parfait ; $\sqrt{51} \approx 7.14$

Factorisation : Fermat 160 ?-1665

Lemme

Si x, y sont des entiers tels que $x^2 - y^2 = n$, alors $n = (x-y)(x+y)$.

$n = 28$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 8$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 21$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 36 = 6^2$ donc $n = (x_3 - 6)(x_3 + 6) = 2 \times 14$.

☞L'algorithme peut échouer !

$n = 30$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 6$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 19$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 34$ qui n'est pas un carré parfait ;
4. $x_4 = 9$, $x_4^2 - n = 51$ qui n'est pas un carré parfait et $\sqrt{51} > 7$.

Factorisation : Fermat 160 ?-1665

Lemme

Si x, y sont des entiers tels que $x^2 - y^2 = n$, alors $n = (x-y)(x+y)$.

$n = 28$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 8$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 21$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 36 = 6^2$ donc $n = (x_3 - 6)(x_3 + 6) = 2 \times 14$.

☞L'algorithme peut échouer !

$n = 30$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 6$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 19$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 34$ qui n'est pas un carré parfait ;
4. $x_4 = 9$, $x_4^2 - n = 51$ qui n'est pas un carré parfait et $\sqrt{51} > 7$.

Factorisation : Fermat 160 ?-1665

Lemme

Si x, y sont des entiers tels que $x^2 - y^2 = n$, alors $n = (x-y)(x+y)$.

$n = 28$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 8$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 21$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 36 = 6^2$ donc $n = (x_3 - 6)(x_3 + 6) = 2 \times 14$.

☞L'algorithme peut échouer !

$n = 30$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 6$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 19$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 34$ qui n'est pas un carré parfait ;
4. $x_4 = 9$, $x_4^2 - n = 51$ qui n'est pas un carré parfait et $\sqrt{51} > 7$.

Factorisation : Fermat 160 ?-1665

Lemme

Si x, y sont des entiers tels que $x^2 - y^2 = n$, alors $n = (x-y)(x+y)$.

$n = 28$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 8$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 21$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 36 = 6^2$ donc $n = (x_3 - 6)(x_3 + 6) = 2 \times 14$.

☞L'algorithme peut échouer !

$n = 30$, on commence avec $x_0 = \lfloor \sqrt{n} \rfloor = 5$ puis

1. $x_1 = 6$, $x_1^2 - n = 6$ qui n'est pas un carré parfait ;
2. $x_2 = 7$, $x_2^2 - n = 19$ qui n'est pas un carré parfait ;
3. $x_3 = 8$, $x_3^2 - n = 34$ qui n'est pas un carré parfait ;
4. $x_4 = 9$, $x_4^2 - n = 51$ qui n'est pas un carré parfait et $\sqrt{51} > 7$.

Crible quadratique et successeurs

Les algorithmes de factorisation actuels sont de complexité sous-exponentielle mais restent inspirés par celle de Fermat !

- Pomerance 1981 : *Quadratic Sieve* $e^{(\log n)^{\frac{1}{2}}(\log\log n)^{\frac{1}{2}}}$.
- Pomerance, Lenstra, etc 1990's : *Alg. Sieve* $e^{(\log n)^{\frac{1}{3}}(\log\log n)^{\frac{2}{3}}}$.

☞Records mondiaux en 2009, 2019 et 2020, l'humanité est capable de factoriser des entiers RSA de taille respectives 768, 795 et 829 bits à l'aide du crible algébrique.

Conclusion : factorisation et RSA

☞La factorisation reste un problème difficile, le meilleur algo connu est de complexité sous-exponentielle !

Peut-on réduire cette complexité de calcul dans le cadre d'une instance RSA mal choisie ?

Plan

Hypothèses de l'attaque

Problème RSA

Étant donné un entier RSA $n = pq$ $(p < q < 2p)$ on demande de retrouver la clé secrète d

Question de départ

> Si le designer choisit une petite clé privée d (exposant de déchiffrement), pour des soucis d'efficacité par exemple, le problème RSA devient-il plus facile à résoudre ?

☞Peut-on en déduire un algorithme de résolution du problème RSA de complexité polynomiale lorsque d est suffisamment petit ?
☞la clé secrète d est toujours de taille au moins 100 bits pour éviter une attaque force brute, mais est-ce suffisant ?

☞Dans les années 1980 cette hypothèse (petit exposant) était vraisemblable car les circuits intégrés étaient bien moins efficaces qu'aujourd'hui.

Ingrédients théoriques

- Approximation diophantienne : trouver efficacement un rationnel approchant un nombre réel donné
- Fractions continues

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\cdots}}}}$$

Fractions continues

Conventions : $\frac{1}{0} = \infty$ et $\frac{1}{\infty} = 0$

Représentation

Pour tout réel α, il existe une suite d'entiers naturels $(a_i)_{i \geqslant 1}$ tel que α puisse se réécrire sous la forme

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\dots}}}}$$

avec $a_0 = \lfloor \alpha \rfloor$

Notation

On dénote $a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\dots}}}}$ généralement par $[a_0, a_1, a_2, a_3, \dots]$

Fractions continues

Conventions : $\frac{1}{0} = \infty$ et $\frac{1}{\infty} = 0$

Notation

On dénote $a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\dots}}}}$ généralement par $[a_0, a_1, a_2, a_3, \dots]$

Définition constructive

Si $\alpha = [a_0, a_1, a_2, a_3, \dots]$ alors $\alpha = a_0 + \frac{1}{[a_1, a_2, a_3, \dots]}$.

☞Lorsque $\exists k$ tq $\forall i > k,\ a_i = 0$ on note $[a_0, a_1, \dots, a_k]$

Fractions continues et rationnels

Proposition

Soit α un réel. On a l'équivalence qui suit

$$\alpha \in \mathbb{Q} \Leftrightarrow \exists k \text{ tq } \alpha = [a_0, a_1, \ldots, a_k].$$

Définition

Soit $\alpha = [a_0, a_1, a_2, \ldots]$ un réel. La suite de rationnels $([a_0, \ldots, a_k])_{k \geqslant 0}$ est appelée la suite des *convergents* (ou réduites) de α.

- ☞ On a $[a_0, a_1, \ldots, a_k] \xrightarrow[k \to \infty]{} \alpha$
- ☞ les convergents sont des approximations rationnelles de α

Fractions continues : exemples

- $\frac{30}{13} = [2, 3, 4]$
- $\frac{1+\sqrt{5}}{2} = [1, 1, 1 \ldots]$
- $\pi \simeq [3, 7, 15, 1, 192]$

Fractions continues : rationnels et Euclide

Proposition

Soit $\alpha = \frac{a}{b}$ un rationnel donné sous sa forme réduite. Une fraction continue (finie) $[a_0, a_1, \ldots, a_n]$ représentant α peut être calculée à l'aide de l'algorithme d'Euclide.

☞Cette fraction continue est la plus courte possible.

La suite des quotients q_i calculés lors de l'algorithme d'Euclide avec pour entrée a et b fournit la représentation $\frac{a}{b} = [q_0, q_1, \ldots, q_n]$ (on fixera $q_i \geqslant 0$ pour $i \geqslant 1$)

Corollaire

Une représentation en fraction continue d'un rationnel $\frac{a}{b}$ peut être calculée en temps

$$\mathcal{O}(\log^2(\max(|a|, |b|))$$

Fractions continues : rationnels et Euclide étendu

Proposition

Soit $\alpha = \frac{a}{b}$ un rationnel donné sous sa forme réduite. Les fractions réduites $\frac{n_k}{d_k}$ représentant les convergents successifs $[a_0, a_1, \ldots, a_k]$ sont toutes calculées au cours d'un algorithme du type Euclide étendu avec a et b pour entrée.

On a $n_0 = a_0$, $d_0 = 1$, $n_1 = a_0 a_1 + 1$, $d_1 = a_1$ et pour $i \geqslant 1$

$$\begin{pmatrix} n_{i+1} & d_{i+1} \\ n_i & d_i \end{pmatrix} = \begin{pmatrix} a_{i+1} & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} n_i & d_i \\ n_{i-1} & d_{i-1} \end{pmatrix}$$

Corollaire

Une représentation en fraction réduite de tous les convergents d'un rationnel $\frac{a}{b}$ peuvent être calculés en temps

$$\mathcal{O}(\log^2(\max(|a|, |b|))$$

Fractions continues : application fondamentale

Soit à calculer une approximation de α une solution du polynôme $x^2 - 5x - 1$. On a $\alpha = 5 + \frac{1}{\alpha}$ et en appliquant récursivement cette relation on obtient

$$\alpha = [5, 5, 5, \ldots]$$

Cette fraction continue est périodique à partir d'un certain rang (on note alors $[\overline{5}]$). La suite de ses convergents fournit des approximations de α (l'erreur diminuant au fur et à mesure que k augmente). Plus généralement on a le résultat suivant (Lagrange, Gauss, Legendre)

Proposition

Une fraction continue représentant un réel α est périodique à partir d'un certain rang k (i.e. $\alpha = [a_0, \ldots, a_{k-1}, \overline{a_k, \ldots, a_{k+\ell}}]$ ssi α est racine d'un polynôme de degré 2 à coefficients entiers.

☞On peut utiliser le même genre de résultat pour calculer des approximations de racine carrée d'entiers.

Fractions continues : théorème fondamentale

Théorème de Legendre (1752-1834)

Soit α un réel et un couple d'entiers (a, b) premiers entre eux et tel que $b > 0$. Si l'inégalité

$$|\alpha - \frac{a}{b}| < \frac{1}{2b^2}$$

est vérifiée alors $\frac{a}{b}$ est la représentation sous forme de fraction réduite d'un convergent de α.

Du théorème de Legendre à l'attaque de Wiener

☞$\varphi(n)$ vérifie, pour un certain entier $k > 0$ premier avec d, l'égalité

$$ed - k\varphi(n) = 1$$

en divisant par d et $\varphi(n)$ on obtient

$$\frac{e}{\varphi(n)} - \frac{k}{d} = \frac{1}{d\varphi(n)}$$

si $2d < \varphi(n)$ alors on a

$$\left|\frac{e}{\varphi(n)} - \frac{k}{d}\right| = \left|\frac{1}{d\varphi(n)}\right| < \frac{1}{2d^2}$$

☞$\frac{k}{d}$ est donc un convergent de $\alpha = \frac{e}{\varphi(n)}$. Si α est connu, en calculant les convergents successifs de α on retrouvera d. Problème, $\varphi(N)$ est *a priori* inconnue ! (sa connaissance suffit pour factoriser n).

☞Idée, remplacer $\varphi(n)$ par une valeur connue proche : n

Attaque de Wiener : approximation de clés secrètes

☞On remplace $\varphi(n)$ par n et la relation précédente devient

$$\frac{e}{n} - \frac{k}{d} = \frac{ed - kn}{dn} = \frac{ed - k\varphi(n) - kn + k\varphi(n)}{dn} = \frac{1 - k(n - \varphi(n))}{dn}$$

Comme $p < q < 2p$ et $n = pq$ on a $n - \varphi(n) = p + q - 1 < 3\sqrt{n}$. Si $e < \varphi(n)$ (c'est *a priori* le cas) alors $k < d$ et on en déduit

$$\left|\frac{e}{n} - \frac{k}{d}\right| = \left|\frac{k(n - \varphi(n)) - 1}{dn}\right| < \frac{3k\sqrt{n}}{dn} = \frac{3k}{d\sqrt{n}} < \frac{3}{\sqrt{n}}$$

☞Pour que les hypothèses du Théorème de Legendre soient vérifiées il suffit que $\frac{3}{\sqrt{n}} < \frac{1}{2d^2}$ c'est-à-dire

$$d < \frac{1}{\sqrt{6}} n^{\frac{1}{4}}$$

Attaque de Wiener : résultat principal

Théorème de Wiener

Soit $n = pq$ un entier RSA avec $p < q < 2p$ et (e, d) vérifiant $ed = 1 \bmod \varphi(n)$. Si l'exposant de déchiffrement d (clé secrète) vérifie

$$d < \frac{1}{\sqrt{6}} n^{\frac{1}{4}}$$

alors on peut casser totalement RSA avec un algorithme de complexité polynomiale en la taille de cette clé secrète d.

☞Par exemple si n fait 1024 bits et d moins de 256 bits, cette attaque fonctionne (et pas la recherche exhaustive !)

Attaque de Wiener : démonstration constructive

1. Soit c le chiffré d'un message aléatoire m : $c = m^e \mod n$.
2. On calcule les convergents $(\frac{n_i}{d_i})_{i=0}^{t}$ de $\frac{e}{n}$ (on a $t = \mathcal{O}(\log(n))$).
3. On teste les d_i jusqu'à obtenir $c^{d_i} = m \mod n$.

On a au plus $t = \mathcal{O}(\log(n))$ tours de boucle à faire.

Comme le coût total de toutes les opérations dans une boucle est polynomial en $\log(n)$ on a bien un algorithme polynomial en la taille de la clé d (puisque $\log(d) = \mathcal{O}(\log(n))$).

Exemple

Soit le problème RSA avec $(n = 160523347, e = 60728973)$.
On chiffre un texte pris au hasard, ici $m = 313$. On a

$$c = 313^{60728973} \bmod 160523347 = 75454098$$

qui nous servira à tester les candidats pour d.

Première étape : On calcule la fraction continue de $\frac{e}{n}$:

$$\frac{e}{n} = [0, 2, 1, 1, 1, 4, 12, 102, 1, 1, 2, 3, 2, 2, 36]$$

Seconde étape : On teste les convergents successifs. On s'aperçoit que le sixième permet de résoudre ce problème RSA. En effet,

$$[0, 2, 1, 1, 1, 4] = \frac{14}{37}.$$

Testons si $\frac{14}{37}$ est bien égal à $\frac{k}{d}$ en calculant $c^d \bmod n$. On a

$$75454098^{37} \bmod 160523347 = 313,$$

donc on a bien retrouvé la clé secrète d.

☞En pratique on calcule directement les convergents successifs.

Tester en Sage

On vous demandera de coder cette attaque dans un langage de bas niveau (`Python` ou mieux `C+GMP`).
Vous pouvez dans un premier temps écrire une version haut niveau en Sage pour vérifier vos programmes.

```
n=160523347
e=60728973
alpha=e/n
alpha
```

```
60728973/160523347
```

```
cv=convergents(alpha)
cv
```

```
[0, 1/2, 1/3, 2/5, 3/8, 14/37, 171/452, 17456/46141, 17627/46
35083/92734, 87793/232061, 298462/788917, 684717/1809895,
1667896/4408707, 60728973/160523347]
```

```
m=313
c=pow(m,e,n)
c
```

```
75454098
```

```
for f in cv:
    d=denominator(f)
    if (pow(c,d,n) == m):
        break
print d
```

```
37
```

Conclusion

Paramètres de RSA mal choisis $\Rightarrow$ la sécurité n'est plus assurée !

☞Il faut absolument suivre les recommandations des instituts gouvernementaux.

☞Pour un attaquant, il faut toujours commencer par cette attaque simple (au cas où !).

Plan

Cryptanalyse Opérationnelle de RSA

☞Différents types de canaux auxiliaires

1. *Social Engineering*
2. Macroscopique
3. Microscopique

Social Engineering

- Repose sur les faiblesses humaines !
  - Faiblesse des mots de passe (nom de l'animal famillier, date d'anniversaire,...)
  - Interrogatoires détournés
  - Taupe
  - Lecture derrière l'épaule
  - Hameçonnage
  - etc.
- Attaque la plus utilisée !
- Contremesure : éducation à la sécurité.
- Les mathématiques ne peuvent rien faire ici !

Canaux Macroscopique

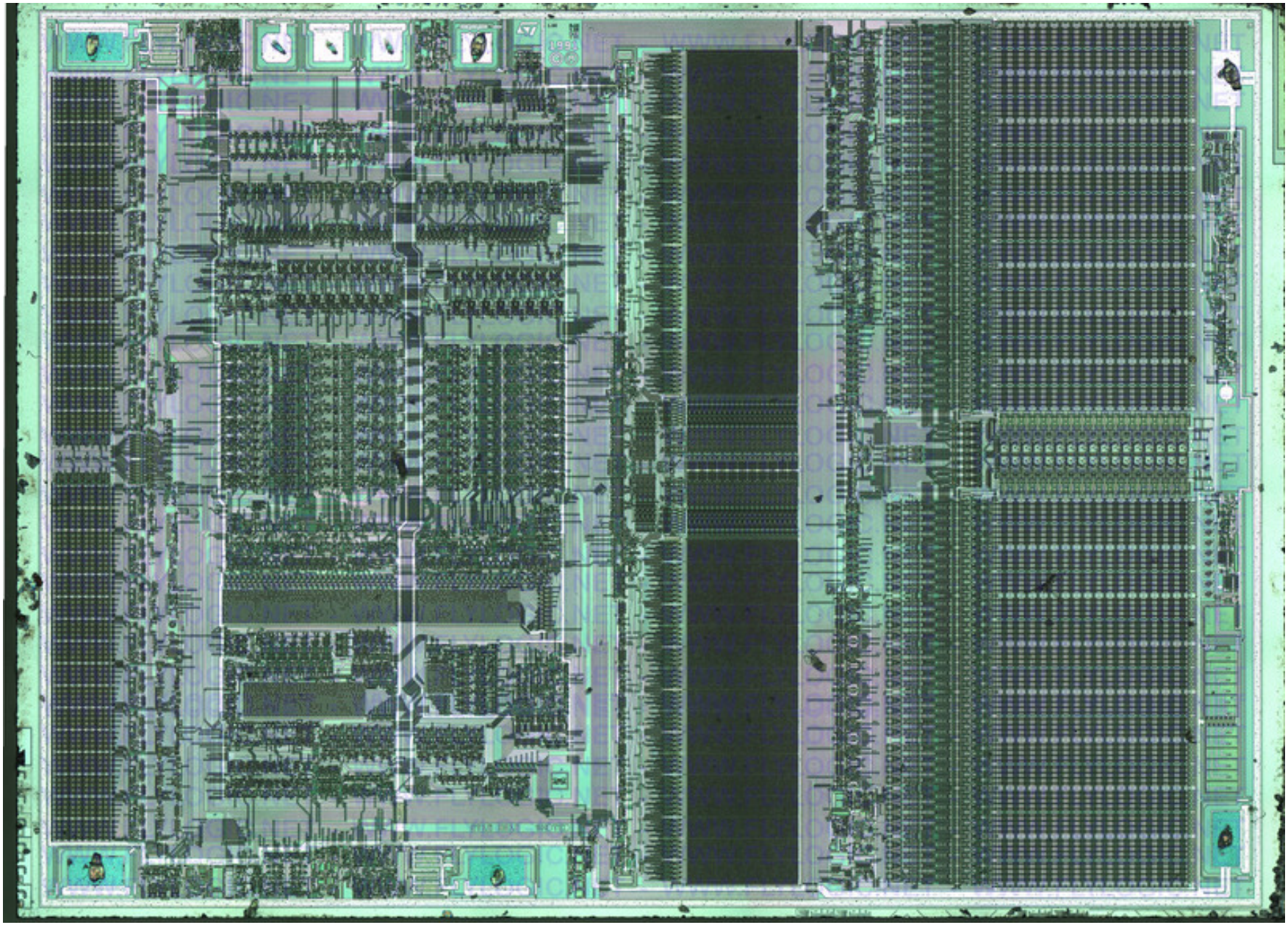
☞Repose sur les *évaporations* d'informations obtenues par phénomènes physiques.

- Emmissions radio des écrans (CRT)
- Claviers mouchards
- Écoute à haute portée
- . . .

☞Les mathématiques tentent de construire des contremesures mais. . .

☞un blindage du matériel sensible est nécessaire (Faraday).

Canaux Microscopiques



Canaux Microscopiques

☞Basé sur les effets de bords de l'arithmétique utilisée

Exponentiation modulaire : attaque de Kocher sur RSA

On calcule $y^d \bmod n$ avec d de w bits.

s_0 := 1

for k:= 0 upto $w-1$ do

  if (k-ième bit de d) = 1 then

    $R_k = (s_k \times y) \bmod n$

  else

    $R_k = s_k$

  end if

  $s_{k+1} = R_k^2 \bmod n$

end for

return R_{w-1}

Canaux Microscopiques

☞Basé sur les effets de bords de l'arithmétique utilisée

Exponentiation modulaire : attaque de Kocher sur RSA

On calcule $y^d \bmod n$ avec d de w bits.

s_0 := 1

for k:= 0 upto $w-1$ do

  if (k-ième bit de d) = 1 then

    $R_k = (s_k \times y) \bmod n$

  else

    $R_k = s_k$

  end if

  $s_{k+1} = R_k^2 \bmod n$

end for

return R_{w-1}

Canaux Microscopiques : attaque de Kocher

☞L'attaquant cherche a retrouver l'exposant d.

Hypothèses de Kocher

1. On peut demander au matériel cryptographique des déchiffrements quelconques avec la clé privée attaquée.
2. On peut mesurer les temps de calcul de manière précise pour chacun des déchiffrement.
3. On peut avoir accès au même matériel et régler nos propres clés.

☞Une attaque par corrélation de Pearson (brute force améliorée)

Canaux Microscopiques : attaque de Kocher

- Attaque par étude statistique
- Étude des corrélations (distributions) entre les différents tests
- Contremesure : rajouter des calculs vides pour équilibrer la charge de travail.

Canaux Microscopiques : attaque par injection de fautes

- ☞A l'aide de radiations on peut changer la nature de quelques bits pendant le calcul
- ☞Utilisable sur les cartes à puces par exemple

Canaux Microscopiques : attaque par injection de fautes

☞A l'aide radiation on peut changer la nature de quelques bits pendant le calcul
☞Utilisable sur les cartes à puces par exemple

Attaque par faute : RSA et CRT

On veut calculer $y^d \bmod n$ rapidement. On connaît $n = pq$.

1. On calcule $S_1 = y^d \bmod p$
2. On calcule $S_2 = y^d \bmod q$
3. On reconstruit par CRT $S = aS_1 + bS_2 \bmod n$

☞On injecte une erreur pendant le premier calcul.

Canaux Microscopiques : attaque par injection d'erreurs

Attaque par faute : RSA et CRT

Supposons que un seul des S_i est faux ici c'est S_1. Supposons aussi que nous avons les résultats juste S et faux $\hat{S}$ issus d'un même message y. Alors

$$S \neq \hat{S} \bmod p \text{ et } S = \hat{S} \bmod q$$

et donc

$$\text{pgcd}(S - \hat{S}, n) = q$$

- ☞Contremesures arithmétiques plus compliquées
- ☞Blindage du matériel pour éviter les fautes.

Conclusions

Canaux auxiliaires

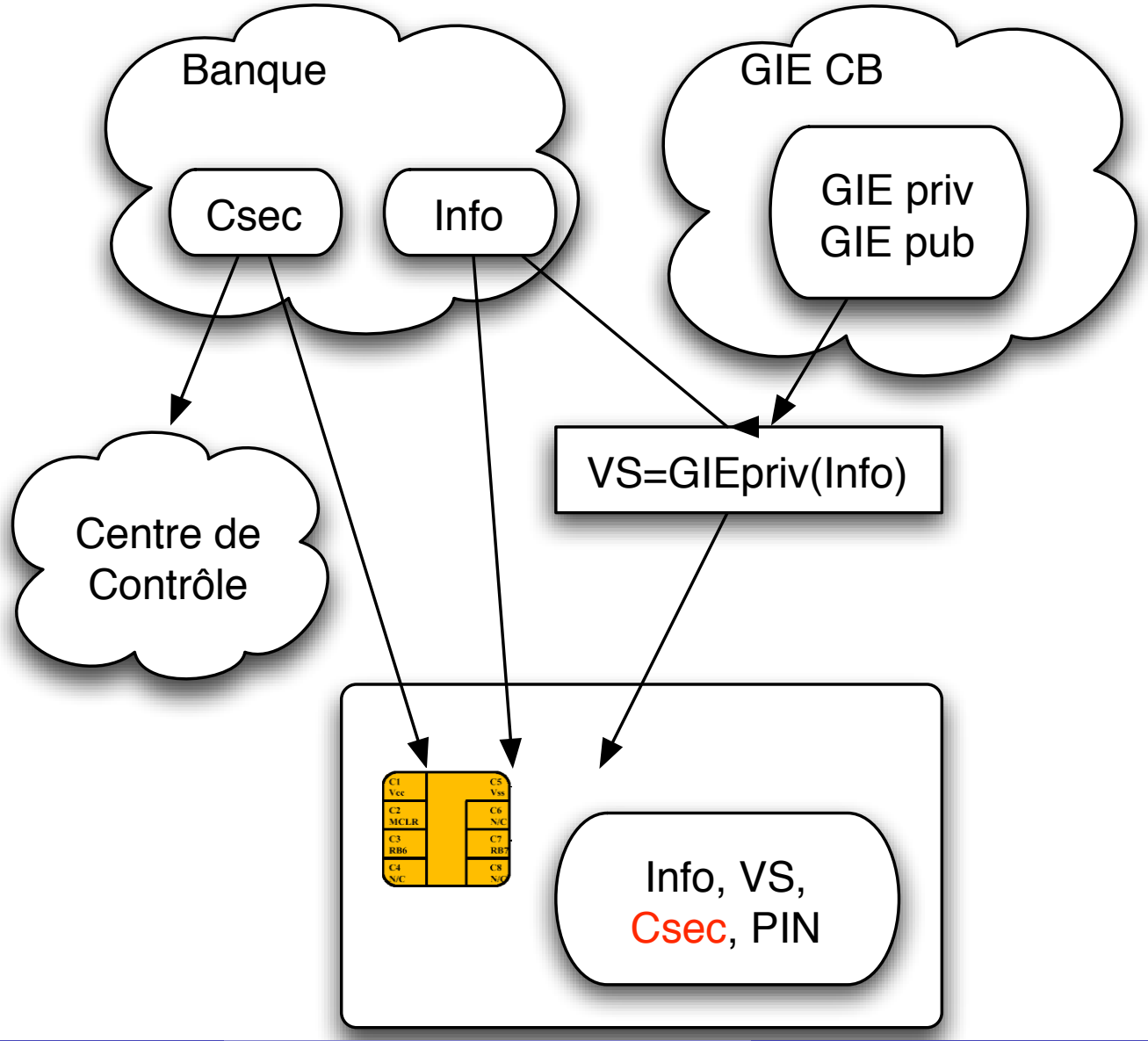
- Autres effets de bord : Thermiques, Radio-electrique,...
- A de forts impacts sur l'électronique contemporaine : téléphonie, carte à puce,...
- Des parades existent mais elles compliquent l'arithmétique sous-jacente.

☞Un cours de M2 du master CCA est consacré a ces problématiques
☞Un teaser vidéo de Tarnovsky !

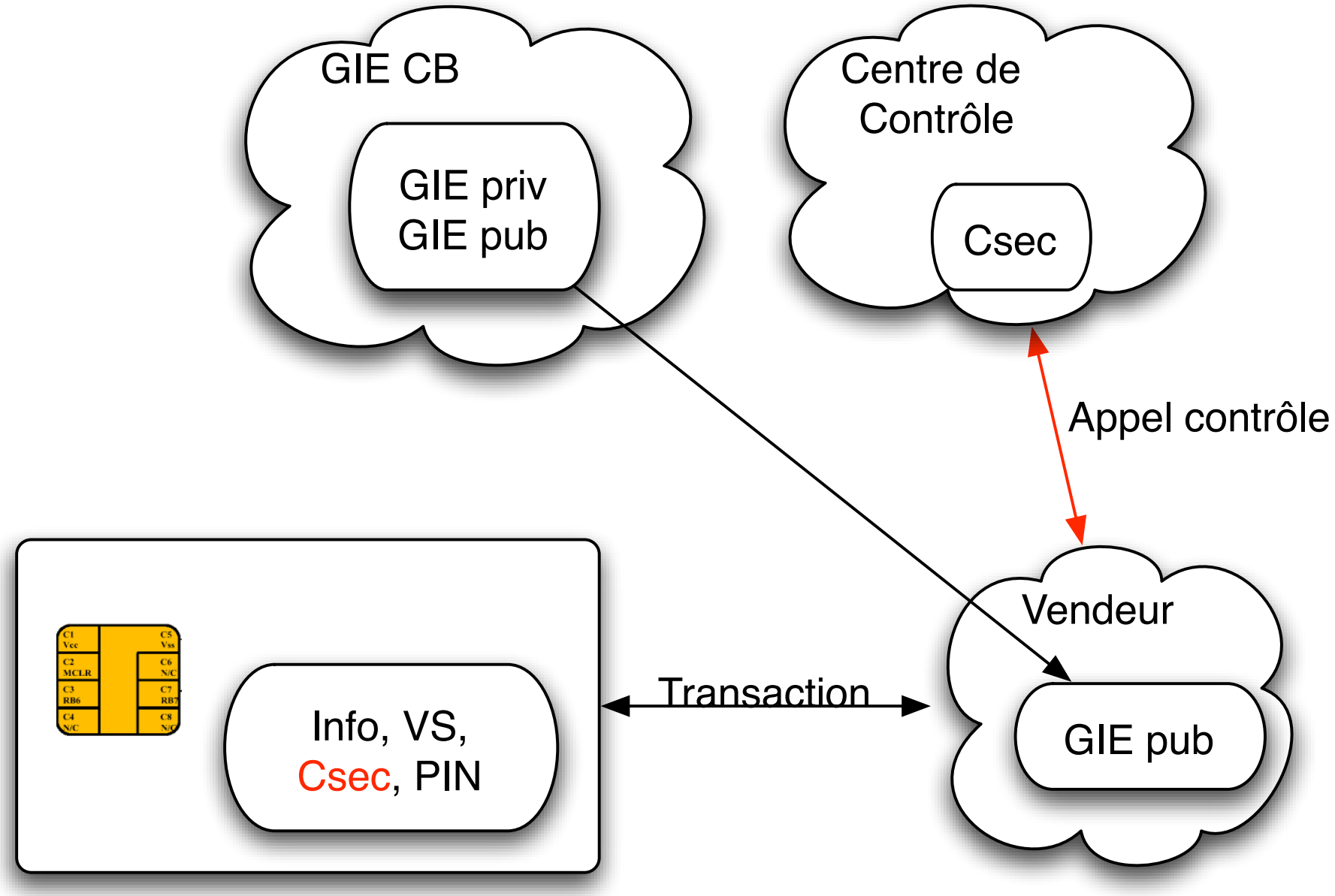
Plan

Création de la carte bancaire

☞La signature RSA permet d'authentifier une transaction



Utilisation de la carte bancaire



Utilisation de la carte bancaire

- Authentification de la carte : le terminal lit Info et VS sur la carte et vérifie la signature à l'aide de GIEpub.
- PIN : le PIN est envoyé à la carte pour calcul en place et vérification.
- Authentification par Csec : appel téléphonique à l'organisme de contrôle pour vérifier de la présence la clé secrète.

Attaque Serge Humpich (1997, factorisation 32 bits), divulgation des YES card (2000)

Quatrième partie IV

Courbe elliptique

Polynômes : Évaluation

Définition

Soit $P = [a_0, \ldots, a_n] \in \mathbb{K}[X]$ un polynôme et e un élément de $\mathbb{K}$ (ou plus généralement d'une $\mathbb{K}$-algèbre i.e un anneau tel qu'il existe un morphisme de $\mathbb{K}$ dans cette algèbre). L'évaluation $P(e)$ de P en e est définie par :

$$\sum_{i=0}^{n} a_i e^i$$

☞Évaluation brute force : On calcule la somme de la définition brutalement, les monômes évalués successivement.

☞n additions et $2n$ multiplications

☞$2n \times \log_2 e$ en mémoire

Polynômes : Évaluation

Définition

Soit $P = [a_0, \ldots, a_n] \in \mathbb{K}[X]$ un polynôme et e un élément de $\mathbb{K}$ (ou plus généralement d'une $\mathbb{K}$-algèbre i.e un anneau tel qu'il existe un morphisme de $\mathbb{K}$ dans cette algèbre). L'évaluation $P(e)$ de P en e est définie par :

$$\sum_{i=0}^{n} a_i e^i$$

Algorithme de Horner $[a_0, \ldots, a_n](e)$ (parenthèses emboîtées) :

$t := a_n$
Pour $i := n - 2$ jusqu'à 0 faire
    $t := a_{i+1} + e \times t$
Fin pour
return $a_0 + e \times t$

Polynômes : Évaluation

Définition

Soit $P = [a_0, \ldots, a_n] \in \mathbb{K}[X]$ un polynôme et e un élément de $\mathbb{K}$ (ou plus généralement d'une $\mathbb{K}$-algèbre i.e un anneau tel qu'il existe un morphisme de $\mathbb{K}$ dans cette algèbre). L'évaluation $P(e)$ de P en e est définie par :

$$\sum_{i=0}^{n} a_i e^i$$

Algorithme de Horner $[a_0, \ldots, a_n](e)$ (parenthèses emboîtées) :

$t := a_n$
Pour $i := n - 2$ jusqu'à 0 faire
  $t := a_{i+1} + e \times t$
Fin pour
return $a_0 + e \times t$

- ☞ n additions
- ☞ n multiplications
- ☞ $n \times \log_2 e$ en mémoire

Plan

Application Principale : Signature DLP

☞Elle repose sur une modification du chiffrement de ElGamal
☞Nécessite l'utilisation d'une fonction de hachage de l'ensemble des messages dans le groupe support du DLP.

La signature DSA sur les corps finis

La génération des clés :

- une fonction de hachage H (e.g. SHA1)
- p, q deux premiers de tailles respectives 1024 et 160 bits tels que $p-1$ est un multiple de q
- g un entier d'ordre q modulo p
- un problème DLP : $h = g^t \bmod p$ (le groupe support est d'ordre q)

Application Principale : Signature DLP

☞Elle repose sur une modification du chiffrement de ElGamal
☞Nécessite l'utilisation d'une fonction de hachage de l'ensemble des messages dans le groupe support du DLP.

La signature DSA sur les corps finis

L'expéditeur publie les clés publiques (p, q, g, h) et conserve t comme clé secrète. La fonction de hachage H est aussi une donnée publique. Pour signer un texte m, l'expéditeur procède comme suite

1. Il choisit un entier k_m au hasard (pour chaque message)
2. $r = (g^{k_m} \bmod p) \bmod q$
3. $s = k_m^{-1}(H(m) + t \times r) \bmod q$
4. si r et s sont non nuls alors la signature de m est le couple (r, s) sinon retourner à la première étape.

☞ATTENTION k_m doit vraiment être choisi au hasard ! (voir TD)

Application Principale : Signature DLP

☞Elle repose sur une modification du chiffrement de ElGamal
☞Nécessite l'utilisation d'une fonction de hachage de l'ensemble des messages dans le groupe support du DLP.

La signature DSA sur les corps finis

Le destinataire vérifiera l'authenticité de l'expéditeur en vérifiant

$$v = r \bmod q$$

où v est calculé comme suit :

1. $u_1 = H(m) \times s^{-1} \bmod q$
2. $u_2 = r \times s^{-1} \bmod q$
3. $v = (g^{u_1} h^{u_2} \bmod p) \bmod q$

☞Alternative à l'utilisation de RSA pour la sécurité embarquée !

Application Principale : Signature DLP

- Elle repose sur une modification du chiffrement de ElGamal
- Nécessite l'utilisation d'une fonction de hachage de l'ensemble des messages dans le groupe support du DLP.

La signature **EC**DSA sur les courbes elliptiques

Peut on faire plus efficace que le DLP sur les corps finis ?

Courbes elliptiques

☞Lieu géométrique !

Sur $\mathbb{K} = \mathbb{F}_p$ avec $p > 3$

Pour a et b dans $\mathbb{K}$ vérifiant

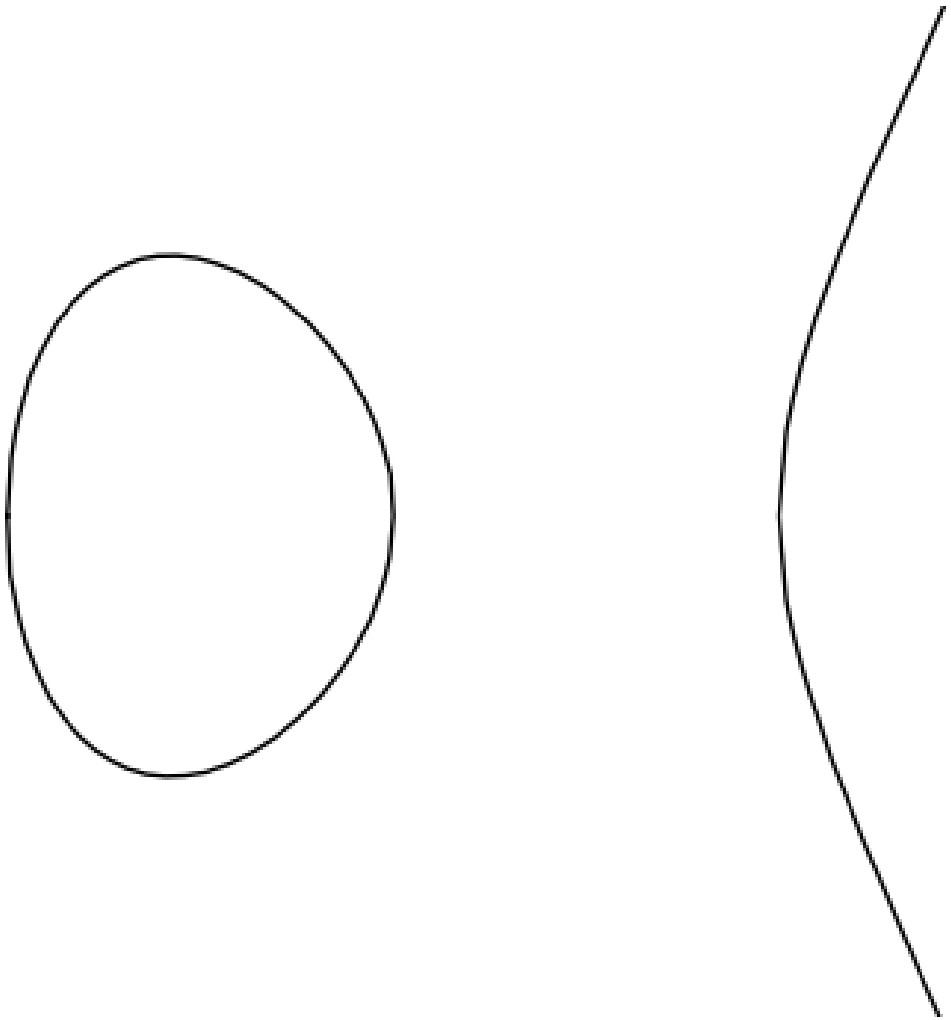
$$\Delta := -16(4a^3 + 27b^2) \neq 0$$

On définit la courbe elliptique $\mathcal{E}_{a,b}$ sur $\mathbb{K}$ par l'ensemble des points $(x, y) \in \mathbb{K}^2$ vérifiant l'équation :

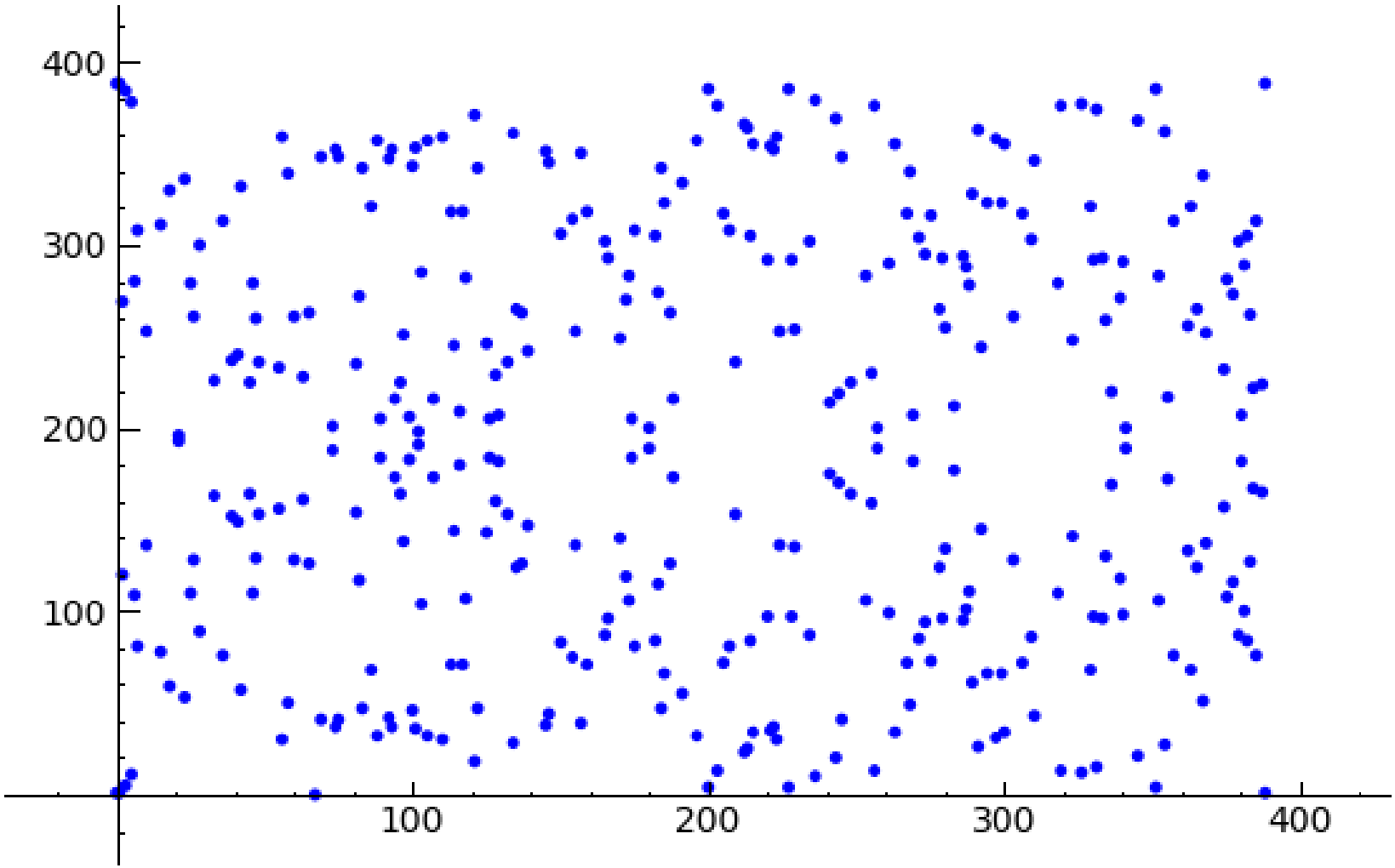
$$y^2 = x^3 + ax + b$$

☞Définition équivalente dans le cas des caractéristiques 2 et 3.

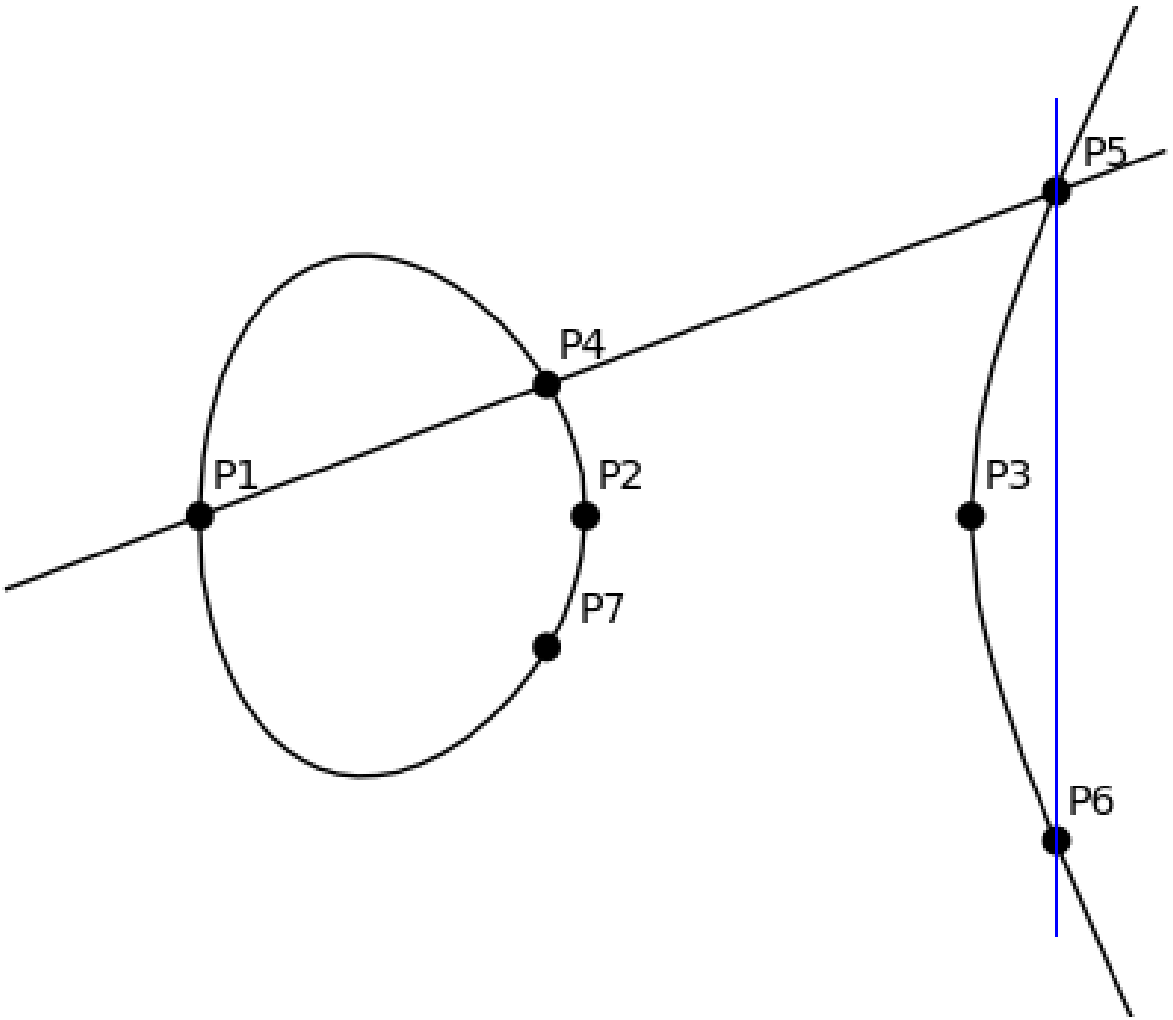
Courbes elliptiques sur $\mathbb{R}^2$



Courbes elliptiques sur $\mathbb{F}_p$, $p = 389$, $y^2 = x^3 - x + 1$



Structure de groupe $E_{a,b} = \mathcal{E}_{a.b} \cup \mathcal{O}$



Structure de groupe $E_{a,b} = \mathcal{E}_{a.b} \cup \mathcal{O}$

Groupe additif commutatif de la courbe

Soit $P = (x_1, y_1), Q = (x_2, y_2) \in \mathcal{E}_{a,b}$

- Point à l'infini $\mathcal{O}$ est l'élément neutre :

$$P + \mathcal{O} = \mathcal{O} + P = P$$

- Si $x_2 = x_1$ et $y_2 = -y_1$ alors $P + Q = \mathcal{O}$
- $P + Q = (x_3, y_3)$ avec

$$x_3 = \lambda^2 - x_1 - x_2$$

$$y_3 = \lambda(x_1 - x_3) - y_1$$

et $\lambda = \frac{3x_1^2 + a}{2y_1}$ si $P = Q$ et $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$ sinon.

☞Ne pas oublier de rajouter le point à l'infini pour former le groupe !

Un exemple concret

On étudie le groupe construit avec $\mathbb{K} = \mathbb{F}_{11}$ et $\mathcal{E}_{1,6} : \ y^2 = x^3 + x + 6$. Pour chaque $x \in \{0, \ldots, 10\}$ on cherche à résoudre une équation quadratique modulo 11. Pour faciliter ce travail, on peut commencer par lister l'ensemble des carrés dans $\mathbb{K}$.

y	$z = y^2 \bmod 11$
0	0
± 1	1
± 2	4
± 3	9
± 4	5
± 5	3

Un exemple concret

On étudie le groupe construit avec $\mathbb{K} = \mathbb{F}_{11}$ et $\mathcal{E}_{1,6} : \ y^2 = x^3 + x + 6$. Pour chaque $x \in \{0, \ldots, 10\}$ on cherche à résoudre une équation quadratique modulo 11. Pour faciliter ce travail, on peut commencer par lister l'ensemble des carrés dans $\mathbb{K}$.

y	$z = y^2 \bmod 11$
0	0
± 1	1
± 2	4
± 3	9
± 4	5
± 5	3

x	$z = x^3 + x + 6 \bmod 11$	y
0	6	
1	8	
2	5	± 4
3	3	± 5
4	8	
5	4	± 2
6	8	
7	4	± 2
8	9	± 3
9	7	
10	4	± 2

Un exemple concret

On étudie le groupe construit avec $\mathbb{K} = \mathbb{F}_{11}$ et $\mathcal{E}_{1,6} : \ y^2 = x^3 + x + 6$. Pour chaque $x \in \{0, \ldots, 10\}$ on cherche à résoudre une équation quadratique modulo 11. Pour faciliter ce travail, on peut commencer par lister l'ensemble des carrés dans $\mathbb{K}$.

y	$z = y^2 \bmod 11$
0	0
± 1	1
± 2	4
± 3	9
± 4	5
± 5	3

x	$z = x^3 + x + 6 \bmod 11$	y
0	6	
1	8	
2	5	± 4
3	3	± 5
4	8	
5	4	± 2
6	8	
7	4	± 2
8	9	± 3
9	7	
10	4	± 2

☞ 12 points rationnels $+$ le point à l'infini $=$ un groupe d'ordre 13, donc cyclique engendré par tout élément autre que le neutre.

Exemple concret

Calculons explicitement avec $\mathbb{K} = \mathbb{F}_{11}$, $\mathcal{E}_{1,6} : \ y^2 = x^3 + x + 6$.
Soit $P = (2, 7)$ on veut calculer $[2]P = P + P$ et $[3]P$.

$$\begin{aligned} \lambda &= (3 \times 2^2 + 1)(2 \times 7)^{-1} \bmod 11 \\ &= 2 \times 3^{-1} \bmod 11 \\ &= 2 \times 4 \bmod 11 \\ &= 8 \end{aligned}$$

$$[2]P = (8^2 - 2 - 2 \bmod 11, 8(2 - 5) - 7 \bmod 11) = (5, 2)$$

Pour $[3]P$ on a $[3]P = [2]P + P$, on calcule l'autre λ

$$\begin{aligned} \lambda &= (7 - 2)(2 - 5)^{-1} \bmod 11 \\ &= 5 \times 8^{-1} \bmod 11 \\ &= 5 \times 7 \bmod 11 \\ &= 2 \end{aligned}$$

$$[3]P = (2^2 - 5 - 2 \bmod 11, 2(5 - 8) - 2 \bmod 11) = (8, 3)$$

Carrés modulo p, symbole de Legendre

☞Étant donné un entier a on ne connaît pas d'algorithme déterministe qui calcule b tel que $a = b^2 \bmod p$ en temps polynomial (en $\log p$).
☞On peut décider qu'un élément est *résidu quadratique* (un carré modulo p) en temps polynomial.

Définition : Symbole de Legendre

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & \text{si } a \text{ est un résidu quadratique} \pmod p\ ; \\ -1 & \text{si } a \text{ n'est pas un résidu quadratique} ; \pmod p\ ; \\ 0 & \text{si } p \text{ divise } a. \end{cases}$$

Propriétés

$$\left(\frac{a_1 a_2}{p}\right) = \left(\frac{a_1}{p}\right)\left(\frac{a_2}{p}\right) \text{ et } \left(\frac{a}{p}\right) = \left(\frac{a \bmod p}{p}\right)$$

Carrés modulo p, symbole de Legendre

☞Étant donné un entier a on ne connaît pas d'algorithme déterministe qui calcule b tel que $a = b^2 \bmod p$ en temps polynomial (en $\log p$).

☞On peut décider qu'un élément est *résidu quadratique* (un carré modulo p) en temps polynomial.

Réciprocité quadratique

Soit p et q deux premiers distincts > 2 (le cas $p = 2$ est trivial)

$$\left(\frac{-1}{p}\right) = \begin{cases} 1 & \text{si } p \equiv 1 \pmod 4 \\ -1 & \text{si } p \equiv -1 \pmod 4 \end{cases},$$

$$\left(\frac{2}{p}\right) = \begin{cases} 1 & \text{si } p \equiv \pm 1 \pmod 8 \\ -1 & \text{si } p \equiv \pm 3 \pmod 8 \end{cases}$$

$$\left(\frac{p}{q}\right) = \begin{cases} \left(\frac{q}{p}\right) & \text{si } p \equiv 1 \pmod 4 \text{ ou } q \equiv 1 \pmod 4 \\ -\left(\frac{q}{p}\right) & \text{si } p \equiv 3 \pmod 4 \text{ et } q \equiv 3 \pmod 4 \end{cases}$$

Structure de groupe pour $\mathbb{K} = \mathbb{F}_p$ avec $p > 3$

Théorème de Hasse

Soit p un nombre premier

$$p + 1 - 2\sqrt{p} \leqslant |E_{a,b}| \leqslant p + 1 + 2\sqrt{p}$$

Théorème de structure

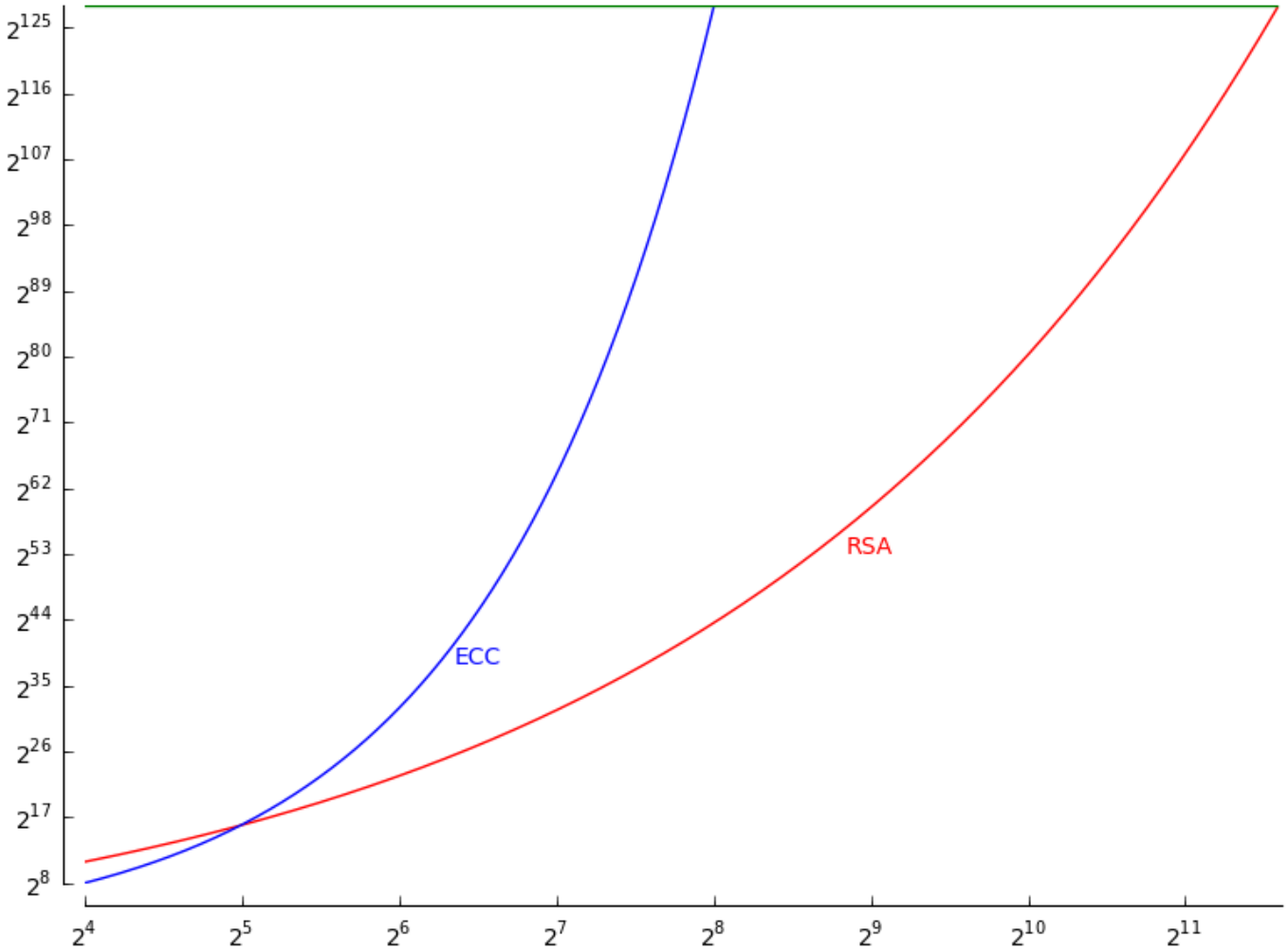
$$(E_{a,b}, +) \simeq (\mathbb{Z}/d_1\mathbb{Z} \times \mathbb{Z}/d_2\mathbb{Z}, +)$$

où d_1 divise d_2 et d_1 divise $p - 1$. En d'autres termes, le groupe construit à partir d'une courbe elliptique est soit cyclique (le cas $d_1 = 1$), soit le produit de deux groupes cycliques.

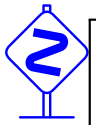
☞Pour la cryptographie on choisit des courbes elliptiques telles que $(E_{a,b}, +)$ est cyclique (le cas $d_1 = 1$) ou au moins contient un grand groupe cyclique (Pohlig-Hellman).

Avantages des courbes elliptiques

- Pas d'attaque connue de complexité sous-exponentielle.
- L'arithmétique peut être rendue très efficace.
- Clefs plus petites pour des sécurités équivalentes.



Retour sur Échange de clé et Authentification



Comment authentifier son interlocuteur au moment du premier échange ?