Codage - Cryptographie

Emmanuel Jeandel (emmanuel.jeandel@lif.univ-mrs.fr)
http://www.lif.univ-mrs.fr/~ejeandel/enseignement.html

28 mars 2011

1 Partages de Secret

Q 1) Trouver un protocole pour que Alice confie un secret à Bob, Carole et David de sorte que la seule façon de reconstituer le secret est qu'ils se réunissent ensemble.

Q 2) Alice décide de confier également le secret à Elise, de sorte qu'elle puisse remplacer Bob ou Carole, mais pas les deux. Etablir un protocole permettant un tel comportement.

Q 3) Trouver un protocole pour que Alice confie un secret à Bob, Carole, et David de sorte que la seule façon de reconstituer le secret soit que deux d'entre eux se réunissent ensemble.

On suppose maintenant que Alice a donné, d'une façon qu'on ne connait pas, un secret à Bob, Carole, David et Eve, de sorte que trois d'entre eux doivent se réunir ensemble pour le retrouver.

Q 4) Alice est morte. Un nouveau collègue, Fabrice, est arrivé. Bob et les autres décident de partager le secret avec Fabrice, de sorte que, comme avant, il suffit que trois d'entre eux se réunissent pour retrouver le secret. Expliquer comment faire. (Il n'est pas nécessaire, ni conseillé, de retrouver le secret x)

Q 5) Eve s'avère en fait être une espionne. Bob, Carole, David et Fabrice vont donc changer leurs secrets de sorte que Eve ne puisse plus les utiliser. Que vont-ils faire ?

Q 6) Bob, Carole, David et Fabrice trouvent que la sécurité est trop élevée. Ils décident donc que maintenant il suffit que deux d'entre eux se réunissent pour retrouver le secret. Que vont-ils faire ?

Définition 1

> Un schéma à seuil (k, n) est un dispositif qui permet à Alice de confier son secret x à n personnes, de sorte que k quelconques d'entre elles soient capables de le retrouver en se réunissant ensemble.

La première question est donc un schéma $(3, 3)$, la troisième un schéma $(2, 3)$. On admettra qu'on peut trouver des schémas (k, n) pour tous n et k.

Variante

Alice, Bob, Carole et David cherchent à générer un secret y de sorte que
- Personne ne connait le secret
- Le secret peut être retrouvé si les 4 personnes se réunissent ensemble

Q 7) Etablir un protocole.

On veut maintenant pouvoir retrouver le secret si 3 des 4 personnes se réunissent ensemble.

Q 8) Etablir un protocole.

2 Transmettre un secret

Alice possède deux secrets X_0 et X_1. Bob veut récupérer un des deux secrets X_i mais ne veut pas réveler à Alice lequel. On suppose qu'Alice dispose d'une paire (clé publique,clé privée) dont Bob ne connait évidemment que la clé publique. Voici le début du protocole

- Alice envoie deux nombres aléatoires A_0 et A_1 à Bob
- Bob choisit un nombre k, encrypte k pour obtenir K et envoie $Y = K + A_i$ à Alice
- Alice décrypte $Y - A_0$ pour obtenir B_0 et $Y - A_1$ pour obtenir B_1 et envoie $B_0 + X_0$ et $B_1 + X_1$ à Bob

Q 1) Finir le protocole. (on pourra faire deux cas suivant si $i = 0$ ou $i = 1$)

Q 2) Donner pour $i = 0$ ce qui est en possession d'Alice, Bob et d'un observateur extérieur à la fin de l'exécution du protocole.

Q 3) Vérifier :

- Qu'à la fin du protocole, Bob connaît bien X_i.
- Que Alice ne peut pas savoir ce que Bob veut connaître
- Que Bob ne peut apprendre qu'un seul secret

Q 4) Généraliser le protocole si Bob veut acheter un secret parmi 3.

Q 5) Montrer que le protocole ne marche plus si on enlève les étapes de cryptage et décryptage du message.

3 RSA

On rappelle ici rapidement le principe de RSA.
- On se donne deux nombres premiers p et q
- On fabrique $n = p \times q$.
- On choisit un nombre e premier avec $(p-1) \times (q-1)$.
- On calcule d tel que $ed \equiv 1 \mod (p-1)(q-1)$

On peut alors crypter x avec e en calculant $x^e \mod n$ et décrypter le résultat y en faisant $y^d \mod n$, et réciproquement (on peut crypter avec d et décrypter avec e).

La clé privée est (d, n), la clé publique (e, n).

On rappelle que $x^t \equiv x \mod n$ si $t \equiv 1 \mod \phi(n)$ et que si p et q sont premiers, alors $\phi(pq) = (p-1)(q-1)$

Q 1) Vérifier rapidement qu'on retrouve bien le résultat original en le cryptant avec d puis décryptant avec e.

Q 2) Vérifier que si quelqu'un connait p, q, ou $(p-1)(q-1)$, il peut trouver la clé secrète.

Q 3) Décrypter le message 11 sachant qu'il a été encrypté par la clé $(7, 55)$

Q 4) On suppose qu'on a choisi p et q très proches (par exemple $|p-q| < 1000$). Expliquer comment trouver rapidement p et q à partir de n (poser $p - q = \epsilon$). Essayer avec $n = 64507$.

Signatures

Supposons que Evil ait récupéré le message x, encrypté avec la clé publique d'Alice (ce qu'on note x^D). Evil veut trouver le message x.

Pour cela, elle va demander à Alice de crypter un ou deux messages de son choix. Seulement, si elle demande à Alice de crypter le message x^D, Alice va s'en apercevoir.

Q 5) Proposer un protocole pour qu'Evil décrypte le message x sans qu'Alice s'en rende compte.

Q 6) Expliquer comment cette méthode peut être appliquée pour faire signer à Alice un document sans qu'elle sache ce qu'elle signe (on considère que Alice, pour signer un document, doit le crypter avec sa clé privée)

Même modulo

On suppose que deux personnes ont le même n, mais des entiers d, e différents.

Q 7) Montrer que le cryptage est commutatif (crypter un document par d_1 puis d_2 est équivalent à le crypter par d_2 puis d_1)

Q 8) Montrer qu'en pratique, si un message est crypté avec d_1 et avec d_2 (de façon à obtenir deux documents), on peut retrouver le message d'origine à partir des deux versions cryptées.

Généralisation

Q 9) Généraliser RSA pour qu'il y ait trois clés k_1, k_2, k_3, de sorte que :
- Si on crypte avec k_1, il faut k_2 et k_3 pour décrypter
- Si on crypte avec k_2, il faut k_1 et k_3 pour décrypter
- Si on crypte avec k_3, il faut k_2 et k_1 pour décrypter

4 Diffie-Helmann

On suppose donnés deux entiers g et p (p est premier) que tout le monde connaît (pas seulement Alice et Bob).

Voici le protocole

- Alice choisit un entier x et calculer $X = g^x \mod p$, et le transmet à Bob
- Bob choisit un entier y, calculer $Y = g^y \mod p$ et le transmet à Alice

On suppose qu'il est difficile (même impossible), étant donné $Z = g^z \mod p$ de retrouver z.

Q 1) Expliquer rapidement pourquoi Alice et Bob connaissent $C = g^{xy} \mod p$ tous les deux, mais que personne d'autre ne peut le trouver.

Q 2) À quoi sert un tel protocole ?

Q 3) Généraliser le protocole pour qu'à la fin Alice, Bob et Carole connaissent tous trois un nombre C sans que quiconque qui les observe ne soit capable de trouver C en espionnant les communications.

On suppose que Denis a une machine qui lui permet, étant donné $Z = g^z \mod p$, de retrouver z.

Evil veut utiliser Denis pour casser l'algorithme Diffie-Hellman. Cependant, s'il demande directement à Denis de retrouver x à partir de $X = g^x$, Denis va se rendre compte que g^x correspond au message échangé par Alice, et ne va donc pas répondre.

Q 4) Donner une méthode pour que Evil puisse retrouver x connaissant $X = g^x$ avec l'aide de Denis, mais sans que Denis ait la connaissance de X ou de x. On pourra commencer ainsi : Evil choisit un nombre aléatoire r et envoie $X \times g^r$ à Denis.

5 TP

On se propose en TP de coder la décryption RSA comme on l'utilise en pratique (mais en simplifiant tout de même un peu).

5.1 Génération de la clé

Les clés publiques et privées seront générés par les commandes :

```
openssl genrsa -out private.pem 192
openssl rsa -in private.pem -pubout -out public.pem
```

La clé privée (resp. publique) se trouve alors dans private.pem (resp. public.pem)

En faisant une clé de 192 bits, vous ne pourrez donc crypter que des chaînes dont la taille est *exactement* $192/8 = 24$

Q 1) Fabriquez-vous une paire clé publique-clé privée.

5.2 Utilisation de la clé

Pour crypter le fichier toto.txt avec la clé publique, on fera

```
openssl rsautl -encrypt -inkey public.pem -pubin -raw -in toto.txt -out toto.crypt
```


Attention, ceci ne marchera que si le fichier toto.txt fait *exactement* 24 caractères.

Pour décrypter le fichier toto.crypt on peut utiliser

```
openssl rsautl -decrypt -inkey private.pem  -raw -in toto.crypt
```

5.3 Premier programme

Q 2) Écrire un programme qui prend un texte de exactement 24 caractères sur son entrée standard et le décrypte sur la sortie.

On utilisera donc l'algorithme RSA :

- Considerer le texte de 24 caractères non pas comme un texte, mais comme un nombre x (donc en base 256)
- Calculer $y = x^d \mod n$
- Renvoyer y, en le considérant comme un texte et non un nombre.

5.3.1 Trouver d et n

On peut retrouver la clé privée (d et n) à partir du fichier `private.pem` en utilisant la commande

```
openssl rsa -in private.pem -text -noout | tr -d ':'
```

n est ce qui est appelé "modulus" dans la sortie, et d ce qui est appelé "private exponent". Les deux nombres sont écrit en base 16 (hexadecimal).

5.3.2 Ecriture du programme

Le programme est très simple à écrire, mais nécessite la manipulation de grands entiers (c'est à dire pas des entiers `int`, bien trop petits pour stocker nos valeurs). Pour cela, on utilisera les facilités que nous offre la bibliothèque `GMP` (`http://gmplib.org/`).

Pour écrire un programme utilisant GMP, il faut ajouter

```
#include <gmp.h>
```

au début du programme C et le compiler avec

```
gcc -o toto -lgmp toto.c
```

Voilà un rapide descriptif des fonctions de la bibliothèque dont vous pourrez avoir besoin

- Les entiers GMP sont du type `mpz_t`. Avant de pouvoir les utiliser, il faut les initialiser, en utilisant la fonction `mpz_init`.
- Si on veut rentrer une chaîne encodée en base 16 dans un entier x, on utilisera
  `mpz_set_str(x, "A5F513E75676AB81", 16);`
- Si on veut convertir une chaîne encodée en base 256, la technique précédente ne marche pas. Il faut utiliser
  `mpz_import(x,12, 1, 1, 1, 0, t);`
  On ne va pas s'attarder sur la signification des paramètres. Le deuxième paramètre (ici 12) spécifie le nombre de caractères du tableau t à lire. Le résultat est alors dans x.
- Si on veut convertir un nombre x en une chaîne de caractères t on écrira symétriquement
  `mpz_export(t,NULL, 1, 1, 1, 0, x);`
- Enfin, la fonction
  `mpz_powm(a,b,c,d);`
  met dans l'entier a le résultat de $b^c \mod d$.

5.4 Deuxième programme

Le professeur a encrypté un message avec une clé trop faible, de seulement 40 bits (il s'agit donc d'un message de 5 caractères). Vous trouverez le message crypté et la clé publique du message sur la page web du cours.

Q 3) Décrypter le message.

Pour décrypter le message, il vous faut donc retrouver e, sachant que vous connaissez d et n (qu'on peut obtenir en tapant `openssl rsa -in public-prof.pem -pubin -text -noout`)

Il vous faut donc :

- Retrouver p et q tel que $n = p \times q$.
- Calculer e tel que $ed \equiv 1 \mod (p-1)(q-1)$.
- Réutiliser le programme précédent avec la nouvelle valeur de e et n

Pour calculer e, on va utiliser l'algorithme d'Euclide étendu récursif, qui sur l'entrée a, b avec $a > b$ premiers entre eux, renvoie (x, y) tel que $ax + by = 1$:

- Si $b = 1$, renvoyer $(1, (1 - a))$.
- Sinon
  - écrire $a = qb + r$.
  - Calculer $(x, y) = Euclide(b, r)$
  - Renvoyer $(y, x - q \times y)$

Attention, il sera nécessaire d'utiliser le type `long` plutôt que le type `int` pour que le programme marche correctement. On affichera les valeurs avec `printf("%ld", x);`

Hint : La valeur de e finit par 7 et le plus petit des deux nombres premiers finit par 9.