Développement Logiciel Cryptographique

Examen de session 1

Février 2018

Préambule

Les supports de cours de l'UE Développement Logiciel Cryptographique **sont les seuls documents autorisés** pour cet examen.

L'usage d'une calculatrice est autorisé. Durée : 1h30

1 Quizz [7 points]

1. Pourquoi n'utilise-t-on pas le mode CRT du RSA pour chiffrer un clair ou vérifier une signature plus rapidement ?

2. Pour calculer un logarithme discret, quel est le principal avantage de la méthode *rho* de POLLARD sur la méthode *baby step - giant step* ?

3. Il est parfois laissé à l'utilisateur la possibilité de choisir la valeur de l'exposant public de sa clé RSA. A-t-il totale liberté dans le choix de cet élément de clé ? Si vous pensez que non, quelles sont selon vous les restrictions, et pourquoi ?

4. Il est souvent considéré que l'exposant privé d'une clé RSA est *full size*. Que signifie ce terme, et pourquoi s'applique-t-il à cet élément de clé ?

5. Écrivez le pseudo-code d'une multiplication scalaire sur courbes elliptiques par la méthode du *doube-and-add de droite à gauche*.

6. Dans l'attaque SQUARE sur l'AES :
   (a) Qu'appelle-t-on une cellule active ?, une cellule équilibrée ?
   (b) Une cellule active est-elle nécessairement équilibrée ? Justifiez.
   (c) Une cellule équilibrée est-elle nécessairement active ? Justifiez.

7. Supposons qu'un déchiffrement RSA standard avec une clé de 1024 bits prenne 120 ms. Combien de temps prendra ce même déchiffrement effectué en mode CRT ? En combien de temps prendra le déchiffrement en mode CRT avec une clé de 2048 bits ? Justifiez vos réponses.

2 Bibliothèque GMP [7 points]

1. Écrivez en langage C une fonction qui prend en entrée deux grands entiers a et b de type `mpz_t` (on supposera que $a < b$), et qui génère pour l'appelant un grand entier n pair aléatoire et uniformément distribué dans l'intervalle $[a, b]$.

2. Le fragment de programme suivant contient (au moins) sept erreurs ou maladresses de programmation. Lesquelles ? Expliquez.

```
...
#include <time.h>
#include "gmp.h"
...
gmp_randstate_t prng;
...
void get_prime(z_prime, unsigned int size) {
  mpz_t z_tmp;
  gmp_randseed_ui(prng, time(NULL));
  mpz_urandomb(z_tmp, prng, size)
  while (mpz_probab_prime_p(z_tmp, 5) != 2)
    mpz_add(z_tmp, z_tmp, 2);
  mpz_clear(z_tmp);
  mpz_set(z_prime, z_tmp);
}
...
int main() {
  ...
  gmp_randinit_default(prng);
  ...
  scanf("%u", size);
  ...
  get_prime(z_p, size);
  get_prime(z_q, size);
  ...
}
```

3 Génération de premiers [6 points]

1. On suppose une génération de premiers consistant à répétitivement tirer au hasard un entier n et à en tester la primalité par un test de MILLER-RABIN à 5 itérations. On considère que le temps mis pour générer un premier de cette manière est essentiellement dominé par les exponentiations modulaires. On note que la densité des premiers autour de x est bien approximée par $1/\ln x$.

   a) Évaluez le nombre moyen de candidats testés pour générer un premier de 512 bits.

   b) Évaluez le nombre total moyen d'itérations effectuées pour générer un premier de 512 bits.

   c) En supposant que la calcul d'une exponentiation modulaire sur des nombres de 512 bits prend 10 μs, évaluez le temps moyen de génération d'un premier de 512 bits.

   d) Que deviennent les réponses aux questions ci-dessus dans le cas d'une génération d'un premier de 1024 bits ?

2. Un nombre de CARMICHAEL peut-il être déclaré composé par un test de FERMAT ? Expliquez.

3. Un nombre de CARMICHAEL peut-il être déclaré premier par un test de MILLER-RABIN ? Expliquez.