Composition de Cryptographie - 2005

La composition est décomposée en 3 exercices de difficulté croissante + une question bonus.

Il est recommandé aux élèves de bien choisir l'ordre des parties selon leurs compétences et rapidités.

Le support de cours est permis. La majeure partie des questions fait appel au sens pratique d'ingénieur et le suivi du cours en classe, il est recommandé à l'élève de ne pas perdre son temps pour rechercher les solutions dans le support de cours.

1. Déchiffrement avec algorithme symétrique (2 points)

Une fichier de taille N est chiffré par AES-128 en mode CBC par une clé K et un vecteur d'initialisation IV.

a) Ecrire l'équation du bloc chiffré B'_n (n étant le nième bloc) en fonction de B_n, K, IV, B'_{n-1} ?
b) Le fichier chiffré est envoyé à un destinataire. Le destinataire a besoin de savoir la valeur en clair du dernier bloc. Est ce qu'il a besoin de déchiffrer tout le fichier et pourquoi ?
c) Etendre le résultat à un bloc quelconque du fichier chiffré.

2. Générateur de Nombres Aléatoires (3 points)

Un générateur de nombres aléatoires utilise l'implémentation C suivante :

```
static uchar desrandomkey[8]={0x12,0x34, 0x56, 0x78, 0x9a, 0xbc, 0xde, 0xf0};

int my_rand(int lg, char * buffer)
{
int         i, cr;
char * keyptr;
time_t seed;
char in[8];
char out[8];

/*/ utiliser des_ebc*/
   keyptr = des_allockeyp(desrandomkey); // allocation d'un contexte clé DES de desrandomkey
   if (keyptr==NULL)
                        return(-1);
   seed = time(NULL); // seed est initialisée au temps courant
   in[0] =  seed&0xff;
   in[2] =  seed>>8;
   in[4] =  seed>>16;
   in[6] =  seed>>24;
   for (i=0; i< lg; i++)  //lg est la taille du nombre aléatoire en sortie
   {
                        des_ecbbykeyp(1, keyptr, 8, in, out); // déchiffrement DES utilisant desrandomkey
                                                              // sur in et en sortie out
                        buffer[i] = out[seed%8]; // manipulation sur out et copie dans le buffer de sortie
                        memcpy(in, out, 8); // réinjecter out dans in
   }
   des_freekeyp(keyptr); // libérer contexte clé
   return(0);
}
```

a) Est ce que la sortie est aléatoire ?
b) On a besoin d'une valeur aléatoire sur 128 octets. Analyser les cas suivants :
   - my_rand est lancé une seule fois avec une sortie de 128 octets.
   - my_rand est lancé 16 fois avec chaque fois une sortie sur 8 octets et ceci pour une machine rapide et une machine lente.

3. Modes 9796 de Signature et API de sécurité (16 points)

a) En reprenant la définition du mode 9796 et en vous basant sur l'exemple que se passe pour la signature de deux documents totalement différents mais ayant la même valeur hash ?
b) En reprenant la définition du mode 9796-2 et en vous basant sur les exemples de rétablissement partiel ou total que se passe pour la signature de deux documents syntaxiquement différents mais ayant la même valeur hash ?
c) En reprenant la définition du mode 9796-2 et en vous basant sur les exemples de rétablissement partiel quel est le risque pour la signature de deux documents xml des tailles très différentes mais ayant la même valeur hash ?
d) Comment on peut améliorer le schéma 9796-2 ?

Le mode 9796-2(2002) introduit la notion de schéma (sous mode). Le schéma 2 précise que le message M à signer est divisé en deux parties : M1 récupérable et M2. On note par || la concaténation de deux chaînes. (par exemple M = M1 || M2). On suppose que l'algorithme de hash est le SHA1 pour la suite de l'exercice.

e) Implémenter en utilisant les requêtes PKCS11 ou CryptoAPI, SHA1(M2).
f) Une valeur aléatoire S ayant la taille de la valeur hash est générée. Implémenter en utilisant les requêtes CryptoAPI ou PKCS11 , la génération de S.
g) Implémenter en utilisant les requêtes PKCS11 ou CryptoAPI et memcpy : H = SHA1(C || M1 || SHA1(M2) || S) où C est la taille de M1 en bits sur 8 octets.

On note :

- D = 0x01 || M1 || S
- N = g(H) où g est une fonction à sens unique de la même taille que D.
- D' = D XOR N
- D'[0] = D'[0] & 0x7F (mise du bit de pois fort à 0)
- F = D' || H || T où T est le suffixe sur un ou deux octets et indique le type de l'algorithme Hash (implicitement ou explicitement comme pour 9796 & 9796-2)

La signature est appliquée directement sur F.

h) A quoi sert la mise à 0 du bit de poids le plus fort de D' ?
i) Discuter du risque du point c en utilisant ce mode.
j) En supposant qu'un provider CryptoAPI nous permet d'appliquer directement un chiffrement RSA clé privée sur une valeur qui n'est pas une valeur hash, implémenter la signature en utilisant les requêtes CryptoAPI.
k) Un fichier ainsi signé est transmis avec le certificat de signature C. Implémenter la vérification en utilisant les requêtes PKCS11.
l) Après l'application de la clé publique sur la signature décrire le procédé de la vérification de la signature qui nous permet de récupérer la partie (M1) du message M. (4 points).

4. Question Bonus : Signature bancaire (4 points)

Une banque désire offrir à sa clientèle de type entreprises les services bancaires de type virements.

a) Quels types de virements peut elle offrir aux entreprises ?
b) Selon les divers paramètres(type de clientèle, de types de virements, ….) décrire les contrôles fonctionnels/applicatifs qui peuvent être réalisés sur les signatures.