Composition de Sécurité logique - 2007

La composition est décomposée en deux parties :

- Une partie d'étude de cas réel
- API de sécurité et Carte à Puce

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 font 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. Etude de Cas Réel. Serveur Bourse en Ligne (16 points)

La Société Bourse-en-Ligne du Wouroudistan offre via son serveur www.bourseonline.wou:

- A ses membres non-clients : la consultation des divers cours avec un décalage de 15 minutes, des forums de discussion pour chaque action et la gestion d'un portefeuille virtuel d'actions.
- à sa clientèle : divers services boursiers incluant le passage d'ordres , gestion de compte, compte bancaire, cours des actions sans décalage horaire, ……

a) Citer deux risques que peuvent subir le serveur ainsi que les conséquences.(4 lignes)
b) Citer les protections à implémenter. (2 lignes)

Accès Membre Non Client :

c) Chaque membre non-client est identifié par un identifiant (username) et un mot de passe. Est-ce que cette protection est nécessaire et pour quelles raisons ? (4 lignes max)
d) Chaque personne qui désire devenir membre, précise dans un formulaire Web un identifiant à utiliser, son mot de passe, son email et quelques autres informations. Citer deux risques liés à cette saisie (hors https) et les protections contre ces risques. (4 lignes max)
e) Pour les membres non-clients l'accès se fait par http uniquement. Est-ce-que c'est suffisant et pourquoi ? (2 lignes max)
f) La participation au forum se fait sans aucun contrôle. Discuter de l'existence éventuelle de risques et de protections dans la participation au forum. (6 lignes max)

Accès Client :

g) L'identifiant (RRRRRRRRSS) du client que la société Bourse-En-Ligne donne à chaque client et est formé de deux parties :
- Une partie racine de 8 chiffres (RRRRRRRR) qui est séquentielle pour chaque nouveau client (notée r[] dans la fonction ci-après)
- Une partie suffixe (SS) de 2 chiffres résultants d'un calcul (notée s[] ci-arpès) par la fonction suivante :

```
// Pour les calculs les chiffres de la racine sont considérés des entiers et non pas la représentation ascii:
int Sum=0 ;
For (i=0 ; i< 8 ; i+=2)
{
        Sum = Sum+  r[i]*10 + r[i+1]; // Exemple '67" donnent 6*10 + 7 = 67
        Sum=Sum%100;
}
s[0] = Sum/10 ;
s[1] = Sum%10 ;
```

A quoi peut servir le suffixe de l'identifiant ?

h) En calculant les suffixes pour les deux racines : 12345678 et 14325876 faire une conclusion sur la robustesse de la fonction précédente.

i) Les clients et les membres non-clients accèdent du même formulaire de login pour la connexion au serveur :

Après la saisie de l'identifiant et en cliquant sur le champ de saisie du mot de passe, le serveur connaît directement s'il a affaire à un client ou à un membre non-client.

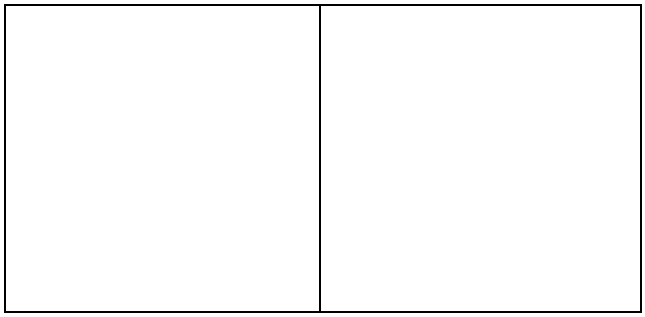
Comment le serveur est capable de différencier les deux types de connexion ?

j) En présence d'un client, et après avoir cliqué sur le champ mot de passe le formulaire de saisie du login change à :

Où le client doit saisir son mot de passe en cliquant avec sa souris sur le clavier virtuel.

Contre quel(s) risque(s) protège le clavier virtuel ?

k) A chaque connexion par le client, un nouveau clavier virtuel est présenté au client :



Contre quel(s) risque(s) protège la modification du clavier virtuel à chaque connexion?

l) Certaines opérations comme les virements du compte du membre vers un compte bancaire nécessitent l'utilisation de la grille d'authentification (format carte de crédit) . Pour rappel le serveur pose une question aléatoire sous la forme d'un numéro de cellule (contenant un nombre aléatoire dépendant du client et que le serveur possède) , et le client doit répondre par le contenu de la cellule.

	A	B	C	D	E
1	302	456	…	…	…
2	…	….	…	…	…
3	….	…	…	….	…
4	…	…	…	789	321

Pour quelles raisons le nombre de cellules (ou lignes et colonnes) ne peut pas être faible ?

m) Pour quelles raisons le nombre de cellules (ou lignes et colonnes) ne peut pas être grand ?

n) Comment peut-on améliorer la sécurité offerte par la grille en jouant sur le contenu de chaque cellule et en prenant en compte la question précédente ?

o) En supposant que chaque grille possède 10 colonnes et 6 lignes et que chaque cellule contient un nombre qui peut être présenté par un integer de taille 4 octets et que le nombre des clients est de 200000, calculer la taille de la liste des grilles des clients sur le serveur.

Opérations :

p) Le serveur reçoit 40 000 hits/minute. Chaque hit est estimé à 20 koctets transmis. Définir la vitesse de la ligne du serveur.

q) Le serveur est managé en local par une connexion SSH (port 22). Définir les règles de filtrage du firewall en utilisant un plan d'adressage interne au réseau de la société 192.168.1.x.

r) Chaque transaction boursière (achat/vente) est identifiée par un identifiant unique. Que proposez vous comme format d'identifiant ?

2. PKCS11 et Carte à Puce (9 points)

Dans un projet d'échange de fichiers sécurisés, une vingtaine de clients sont concernés. La PKI ne nécessite pas l'utilisation de certificats. Chaque client est identifié par une identité unique de 8 caractères. Chaque client possède une carte à puce: sur laquelle seront inscrits une clé privée RSA (S,N) et sa clé publique (N,P) qui sont identifiés respectivement par des fichiers d'identité C101 et C102. La taille de la clé est de 1024 bits.

a) *Pourquoi l'utilisation de certificats n'est pas obligatoire ?*
b) *Comment les clés publiques peuvent être identifiées chez le serveur de la banque d'une façon unique par client ?*
c) *Comment les clés publiques peuvent être échangées ?*
d) *Définir le Mapping de la carte en prenant en compte tous les identifiants.*
e) *Choisir une carte parmi les cartes suivantes : 1ko, 4 ko et 16 ko.*
f) *A partir des attributs PKCS11, proposer un template pour faire la recherche de la clé publique et la clé privée?*
g) *A partir des attributs PKCS11, proposer un template pour obtenir la valeur de la clé publique?*
h) *A partir des attributs PKCS11, proposer un template pour créer un objet de type clé publique?*
i) L'accès à la carte se fait par un lecteur PINPAD (à clavier intégré). La présentation du PIN se fait par un ordre transparent de classe 0xA0 avec instruction=0x20 et le buffer de l'ordre transparent à envoyer est formé de deux zones de syntaxe : T,L,V où T est le type, L (un octet) est la longueur de la valeur V.

1. Zone Formatage de PIN. T=0x52

- La zone valeur a les octets suivants :
  - Contrôle Byte :

    bit1-bit0 : PIN coding
    00 = BCD
    01 = characters ASCII
    10 = Format 2 PIN Block
    11 = RFU

    bit3-bit2 : 00 = usage future

    bit7-bit4 : length of PIN to be presented on the PIPAD

  - Insertion position Byte: This position is used by the reader to insert the entered PIN into the data to the smart card. Count start with 1 (i.e. insertion position 1 is the first byte)

  - Octets PIN à envoyer à la carte par le lecteur: Le PIN saisi au clavier sera inséré à la position indiquée par l'Insertion Position Byte. **Faire attention au format du fichier PIN.**

2. Zone temporisation de saisie. T=0x80.
   - La zone valeur est constituée d'un octet qui indique le temps en secondes.

Sachant que le fichier PIN contient 8 octets formates de la façon suivante:
0xFF,0xFF, 0xFF,0xFF, 0xFF,0xFF,P_1P_2,P_3P_4 avec $P_1P_2P_3P_4$ les 4 digits du PIN de la carte

Définir la requête transparente à envoyer au lecteur pour la saisie sécurisée du PIN.

Valeurs PKCS11 à utiliser dans la question 2 :

Attributs	Certaines valeurs de classes d'attributs
#define CKA_CLASS 0x00000000 #define CKA_TOKEN 0x00000001 #define CKA_PRIVATE 0x00000002 #define CKA_LABEL 0x00000003 #define CKA_APPLICATION 0x00000010 #define CKA_VALUE 0x00000011 #define CKA_CERTIFICATE_TYPE 0x00000080 #define CKA_ISSUER 0x00000081 #define CKA_SERIAL_NUMBER 0x00000082 #define CKA_KEY_TYPE 0x00000100 #define CKA_SUBJECT 0x00000101 #define CKA_ID 0x00000102 #define CKA_SENSITIVE 0x00000103 #define CKA_ENCRYPT 0x00000104 #define CKA_DECRYPT 0x00000105	#define CKO_DATA 0x00000000 #define CKO_CERTIFICATE 0x00000001 #define CKO_PUBLIC_KEY 0x00000002 #define CKO_PRIVATE_KEY 0x00000003 #define CKO_SECRET_KEY 0x00000004 #define CKO_VENDOR_DEFINED 0x80000000

#define CKA_WRAP 0x00000106 #define CKA_UNWRAP 0x00000107 #define CKA_SIGN 0x00000108 #define CKA_SIGN_RECOVER 0x00000109 #define CKA_VERIFY 0x0000010A #define CKA_VERIFY_RECOVER 0x0000010B #define CKA_DERIVE 0x0000010C #define CKA_START_DATE 0x00000110 #define CKA_END_DATE 0x00000111 #define CKA_MODULUS 0x00000120 #define CKA_MODULUS_BITS 0x00000121 #define CKA_PUBLIC_EXPONENT 0x00000122 #define CKA_PRIVATE_EXPONENT 0x00000123 #define CKA_PRIME_1 0x00000124 #define CKA_PRIME_2 0x00000125 #define CKA_EXPONENT_1 0x00000126 #define CKA_EXPONENT_2 0x00000127 #define CKA_COEFFICIENT 0x00000128 #define CKA_PRIME 0x00000130 #define CKA_SUBPRIME 0x00000131 #define CKA_BASE 0x00000132 #define CKA_VALUE_BITS 0x00000160 #define CKA_VALUE_LEN 0x00000161	