Sécurisation des communications

Architecture matérielle, systèmes d'exploitation et réseaux

Exercice 1 *QCM d'activation*

Pour chaque question, une seule réponse est correcte.

1. Le chiffrement symétrique utilise :
   A. deux clés différentes : une publique et une privée
   B. une seule clé, partagée entre l'émetteur et le récepteur
   C. uniquement la clé publique du destinataire
   D. aucune clé, seulement un algorithme secret
2. Quel objectif de sécurité garantit que le message n'a pas été modifié pendant le transport ?
   A. la confidentialité
   B. l'authentification
   C. l'intégrité
   D. la non-répudiation
3. Dans le chiffrement asymétrique, pour envoyer un message confidentiel à Alice, Bob chiffre avec :
   A. la clé privée d'Alice
   B. la clé publique de Bob
   C. la clé publique d'Alice
   D. la clé privée de Bob
4. Le chiffrement de César avec un décalage de 3 transforme la lettre `D` en :
   A. `A`
   B. `G`
   C. `F`
   D. `H`
5. Un certificat numérique délivré par une autorité de certification permet de :
   A. chiffrer le message avec la clé privée du serveur
   B. vérifier que la clé publique appartient bien au serveur annoncé
   C. remplacer le chiffrement asymétrique par du chiffrement symétrique
   D. garantir qu'aucun pirate ne peut intercepter les données

Exercice 2 *Identifier les objectifs de sécurité*

Pour chaque situation, indiquer quel(s) objectif(s) de sécurité (confidentialité, intégrité, authentification) est (sont) menacé(s). Justifier.

1. Un pirate intercepte un email et le lit sans que l'expéditeur ni le destinataire ne s'en aperçoivent.
2. Un pirate modifie le montant d'un virement bancaire pendant son transfert sur le réseau.
3. Un pirate crée un faux site web identique à celui d'une banque pour récupérer les identifiants des clients.
4. Un pirate intercepte un message chiffré, le supprime et le remplace par un autre message (également chiffré avec la bonne clé publique).

Exercice 3 *Chiffrement de Vigenère*

Le chiffrement de Vigenère est une extension du chiffrement de César. Au lieu d'utiliser un décalage unique, on utilise un **mot-clé** qui détermine un décalage différent pour chaque lettre du message.

Par exemple, avec le mot-clé `NSI` (décalages 13, 18, 8) :

Position	**1**	**2**	**3**	**4**	**5**	**6**
Message	S	E	C	R	E	T
Clé (répétée)	N	S	I	N	S	I
Décalage	13	18	8	13	18	8
Chiffré	F	W	K	E	W	B

On appelle **analyse fréquentielle** la technique qui consiste à compter la fréquence de chaque lettre d'un message chiffré et à la comparer à la fréquence des lettres dans la langue du message. En français, la lettre E est de loin la plus fréquente : avec un chiffrement de César, la lettre la plus fréquente du message chiffré correspond donc probablement à E, ce qui donne le décalage. L'exercice « Analyse fréquentielle » met en œuvre cette méthode.

1. Chiffrer le message `PYTHON` avec la clé `NSI`.
2. Écrire une fonction `chiffre_vigenere(message, cle)` qui chiffre un message (en majuscules, sans accent) avec la clé donnée.
3. Écrire la fonction `dechiffre_vigenere(message, cle)` correspondante.
4. Pourquoi le chiffrement de Vigenère est-il plus résistant que celui de César face à l'analyse fréquentielle ?
5. Est-il pour autant incassable ? Justifier.

Exercice 4 *Attaque de l'homme du milieu*

Alice veut envoyer un message confidentiel à Bob en utilisant un chiffrement asymétrique. Elle récupère la clé publique de Bob sur un serveur public. Mais un pirate, Ève, a remplacé la clé publique de Bob par la sienne sur le serveur.

1. Décrire précisément ce qui se passe lorsque Alice chiffre son message et l'envoie.
2. Ève peut-elle lire le message d'Alice ? Justifier.
3. Ève peut-elle faire suivre le message à Bob sans qu'il se doute de quelque chose ? Expliquer comment.
4. Quel mécanisme permet de se prémunir contre cette attaque ? Expliquer son fonctionnement.

Exercice 5 *Analyse fréquentielle*

Le message suivant a été chiffré par substitution mono-alphabétique (chaque lettre est remplacée par une autre, toujours la même) :

```
OH FKLIIUHPHQW HVW XQH PHWKRGH DQFLHQQH
```

On sait que, dans un texte français suffisamment long, les lettres les plus fréquentes sont E, A, S, I, N, T (dans cet ordre approximatif).

1. Compter la fréquence de chaque lettre dans le message chiffré.
2. En déduire que ce message a probablement été chiffré avec un chiffrement de César. Trouver le décalage.
3. Déchiffrer le message.
4. Écrire une fonction Python `casse_cesar(message_chiffre)` qui teste tous les décalages possibles et affiche les 26 résultats.

Exercice 6 *Scénario complet (type bac)*

Cet exercice porte sur la sécurisation des communications.

Clara se connecte au site de sa banque depuis son navigateur. L'adresse du site commence par `https://`.

Partie A — Mise en place de la connexion sécurisée

1. Que signifie le `s` dans `https` ?
2. Quel protocole est utilisé pour sécuriser cette connexion ?
3. Expliquer pourquoi le navigateur de Clara vérifie le certificat du serveur de la banque avant d'établir la connexion. Quel type d'attaque cette vérification permet-elle d'éviter ?
4. Décrire les étapes de l'établissement de la connexion sécurisée entre le navigateur de Clara et le serveur de la banque.

Partie B — Pendant la communication

5. Une fois la connexion établie, le chiffrement utilisé pour échanger les données est-il symétrique ou asymétrique ? Justifier ce choix.
6. Clara effectue un virement de 500 € vers le compte de David. Expliquer comment la signature électronique peut garantir que l'ordre de virement n'a pas été modifié pendant le transfert et qu'il provient bien de Clara.

Partie C — Programmation

7. On souhaite simuler un échange de clé simplifié. Écrire une fonction `genere_cle_session(taille)` qui renvoie une chaîne aléatoire de `taille` caractères parmi les lettres majuscules et les chiffres.
8. Écrire une fonction `xor_chiffre(message, cle)` qui chiffre un message en effectuant un XOR caractère par caractère entre le message et la clé (répétée si nécessaire). On rappelle que `ord(c)` renvoie le code ASCII d'un caractère et `chr(n)` renvoie le caractère de code ASCII n. Pour deux entiers `a` et `b`, l'expression `a ^ b` calcule le OU exclusif (XOR) bit à bit de leurs écritures binaires : par exemple, `6 ^ 3` vaut `5`, car `110` XOR `011` donne `101`. Le XOR de deux caractères `c1` et `c2` s'obtient donc par `chr(ord(c1) ^ ord(c2))`.