Compte-rendu pour le TD (n° 2) de l'option Sécurité Informatique

Matthias Gradaive - Antoine Foucault

12 décembre 2014

Sommaire

Introduction

1 Questions

1.1 Question 1

1.1.1 A

L'usage du chiffrement symétrique implique que les utilisateurs partagent une clé commune deux à deux. Dans notre cas on a quatre utilisateurs qui veulent communiquer entre eux de façon sécurisée (chaque utilisateur veut communiquer individuellement avec un autre de façon sécurisée).
Ainsi, chaque utilisateur veut pouvoir communiquer avec les 3 autres de manière sécurisée. Cela implique donc 6 clés au total, pour que chaque utilisateur puisse communiquer de manière sécurisée avec les autres.

1.1.2 B

Il suffit de générer une paire de clé par utilisateur (donc 4 paires dans notre cas). Ainsi l'ensemble des communications est sécurisé entre chaque utilisateur.

1.1.3 C

Dans un système à N utilisateurs, N(N-1) clés symétriques ou N paires de clés asymétriques sont nécessaires pour assurer la confidentialité des communications entre ces utilisateurs.

1.1.4 D

Dans le cas de grands groupes d'utilisateurs il est préférable d'utiliser un système de chiffrement asymétrique. En effet lorsque le nombre d'utilisateurs grandit, on a besoin de beaucoup plus de clés avec un système de chiffrement symétrique qu'avec un système de chiffrement asymétrique (voir questions précédente).
En effet pour rajouter une personne avec l'usage du chiffrement asymétrique, il suffit seulement de lui générer une paire de clés asymétriques, lui renseigner sa clé privée et placer sa clé publique dans un annuaire par exemple. La gestion est donc beaucoup plus simple.

1.2 Question 2

L'implémentation de l'algorithme DSA est effectuée dans le fichier TD.py (ne pas oublier de lire le README).
Les signatures produites par cet algorithme sont probabilistes. En effet, le choix de la clé éphémère « k » peut entraîner plusieurs signatures, et l'algorithme de vérification considère que l'ensemble de ces signatures sont correctes.

1.2.1 A

Après exécution de DSAVerif() avec les paramètres fournis, cette signature est correcte.

1.2.2 B

Après exécution de DSAVerif() avec les paramètres fournis, cette signature est incorrecte.

1.3 Question 3

1.3.1 A

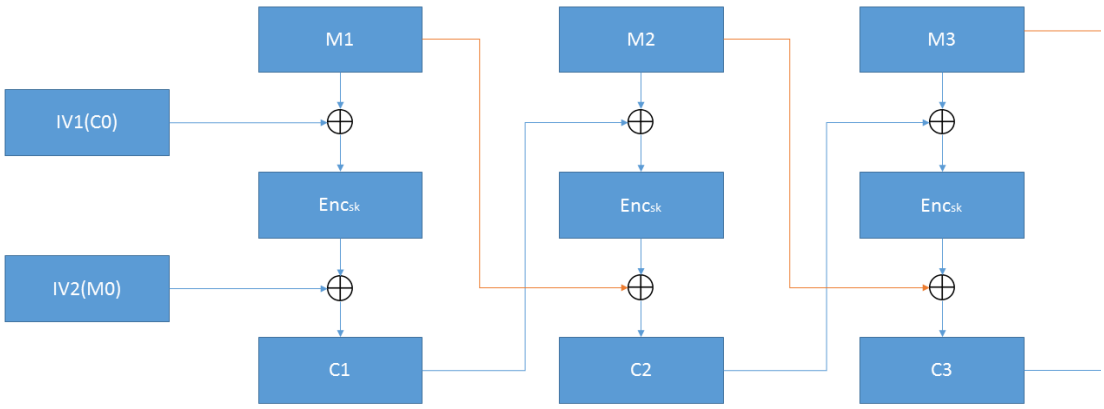

FIGURE 1 – Diagramme présentant le système de chiffrement présenté dans le sujet

1.3.2 B

Voici la formule générale de déchiffrement d'un message : $M_i = C_{i-1}$ XOR $Dec_{sk}(M_{i-1}$ XOR $C_i)$

Pour commencer à déchiffrer le premier message (M_1), Baptiste doit avoir en sa possession les deux vecteurs d'initialisation (IV) qui sont publics. Chaque « tour de déchiffrement » permet ensuite d'obtenir l'information nécessaire au « tour de déchiffrement » suivant (Cette information est M_{i-1}).

1.3.3 C

Le sujet nous indique que le 5^e bloc chiffré a été corrompu lors de la transmission. On cherche donc à savoir l'impact de cette corruption sur l'opération de déchiffrement. Lors du déchiffrement, tous les messages à partir de M_5 seront corrompus. En effet C_5 est utilisé la première fois pour le déchiffrement de M5. Le bloc M_5, résultat du déchiffrement étant incorrect, M_6 le sera également et ainsi de suite...

1.3.4 D

L'usage de MAC pour les blocs chiffrés permettrait d'assurer l'intégrité (et l'authenticité) de chaque bloc envoyé. On peut également réaliser le MAC sur l'ensemble des blocs chiffrés qui ont été envoyés à la fin de la communication.

1.4 Question 4

1.4.1 A

Vrai. Si la taille de l'ensemble d'entrée et supérieur à l'ensemble de sortie, alors il y a obligatoirement des collisions.

1.4.2 B

Vrai. Dans un protocole d'authentification, nous n'avons pas besoin de connaître l'identité de la personne. Par exemple, il suffirait d'avoir la preuve de son appartenance à un groupe qui dispose de certains droits.

1.4.3 C

Ce message ne forme pas un protocole d'échange de clé sécurisé. Amélie communique seulement le hash de la clé et la signature du hash. Cela ne permet pas à Baptiste de récupérer la clé. On ne peut donc pas parler de protocole d'échange de clé.

1.4.4 D

Avec une fonction de hachage à sens unique (non-cryptographique), l'engagement peut être considéré comme camouflant. En effet, l'utilisation du XOR afin de chiffrer la valeur de **i** permet à ce chiffrement d'être inviolable si la valeur de **w** est aléatoire. Par contre l'engagement n'est pas liant. La fonction de hachage n'étant pas cryptographique, il est théoriquement possible pour l'expéditeur (Alice) de calculer deux autres valeurs **x'** et **w'** tel que **H(x' XOR w') = H(x XOR w)**.

Avec une fonction de hachage cryptographique, l'engagement est évidemment encore camouflant. De plus, Alice ne peut plus calculer de nouvelles valeurs de **x'** et **w'** tel que **H(x' XOR w') = H(x XOR w)**. L'engagement est donc à la fois liant et camouflant.

1.5 Question 5

Dans le système présenté pour de cette question, un attaquant pourrait éventuellement deviner le contenu d'un message chiffré, la possibilité des messages clairs se résumant parfois à 2 cas. Cependant nous ne prendrons pas en compte cette remarque pour la résolution des questions.

1.5.1 A

Dans cette situation, Amélie doit s'assurer que son choix de restaurant ne sera pas modifié par Baptiste. Cependant ce choix n'a pas besoin d'être confidentiel. Par conséquent, le chiffrement à clé publique n'est pas nécessaire, les destinataires doivent juste s'assurer que le message n'a pas été modifié et qu'il provient bien d'Amélie (ce qui convient exactement à l'usage d'un MAC).

Amélie a donc juste à envoyer : $\text{rest}_A|\text{MAC}_{sk_{AB}}(\text{rest}_A)|\text{MAC}_{sk_{AC}}(\text{rest}_A)|\text{MAC}_{sk_{AD}}(\text{rest}_A)$

Baptiste peut donc vérifier que le message provient bien d'Amélie et qu'il n'a pas été modifié. Il relaie ensuite le même message aux autres utilisateurs en enlevant (ou pas) sa partie MAC. Ainsi, si Baptiste souhaite modifier le choix d'Amélie, il ne pourra pas changer correctement les MAC destinés aux autres membres et la modification sera détectée.

1.5.2 B

La fête étant surprise, Baptiste ne doit pas être capable de comprendre les messages envoyés par Denis. L'usage de la cryptographie asymétrique est donc justifié ici. De plus, afin d'authentifier et d'assurer l'intégrité du message, l'usage d'un MAC est également utile.

Denis doit donc envoyer deux messages :

À Amélie : $\text{Enc}_{pk_A}(\text{rest}_D)|\text{MAC}_{sk_{AD}}(\text{Enc}_{pk_A}(\text{rest}_D))$
À Christine : $\text{Enc}_{pk_C}(\text{rest}_D)|\text{MAC}_{sk_{CD}}(\text{Enc}_{pk_C}(\text{rest}_D))$

De cette manière Baptiste ne peux pas lire les messages de Denis, tandis que Amélie et Christine peuvent en connaître le contenu (seulement pour le message qui leur est adressé), ainsi qu'en vérifier l'intégrité/authenticité.

1.5.3 C

Dans cette situation on peut considérer Baptiste comme tiers de confiance. Baptiste va recueillir le vote de chaque membre comme un doublon de la forme [Contenu chiffré][MAC associé]. De cette manière, Baptiste est capable d'authentifier les messages qu'il reçoit, de s'assurer qu'ils n'ont pas été modifiés et d'éviter que quelqu'un vote deux fois. De plus, le chiffrement asymétrique assure que les différents votants ne connaîtront pas le choix que les autres ont pu effectuer (et ne seront pas donc influencés) avant le dévoilement des résultats.

Dans le cas où il n'y a pas accès au chifrement asymétrique, il est possible de mettre en place un schéma d'engagement : Chaque utilisateur transmet d'abord le MAC de son choix à Baptiste (de cette façon Baptiste authentifie toujours les messages). Une fois tous les vote reçus, chacun dit à Baptiste le restaurant qu'il a choisi. Ainsi Baptiste peut vérifier si un membre dit la vérité en comparant le MAC de sa réponse avec celui qu'il a précédemment reçu.

1.6 Question 6

Les signatures DSA ne permettent pas de réaliser des échanges confidentiels (chiffrés). Les clés pour les signatures DSA ne peuvent pas être utilisées pour réaliser un chiffrement asymétrique. Nous pouvons alors utiliser l'échange de clé Diffie-Hellman. Cet échange permet à deux personnes de définir une clé qu'ils pourront ensuite utiliser comme clé pour un chiffrement symétrique. Cependant Diffie-Hellman est vulnérable aux attaques man-in-the-middle. Nous pouvons corriger ce problème en signant la valeur envoyée par Alice à Bob (et de même pour Bob vers Alice). Alice et Bob vérifient qu'ils reçoivent bien la valeur envoyée par la bonne personne (en utilisant la partie publique de la clé DSA de l'autre personne) et calculent la clé qu'ils utiliseront pour le chiffrement symétrique. Ils peuvent ensuite s'échanger des messages confidentiels (chiffrés avec la clé symétrique) et authentifiés (en utilisant leurs clés DSA).

Conclusion

Au travers de ce TD, nous avons pu comprendre comment analyser et mettre en place des modes de communication authentifiés et sécurisés. La diversité des situations présentées met en avant la nécessité de comprendre et adapter les outils cryptographiques afin d'éviter des erreurs pouvant mettre en péril une communication.
Une étude de l'algorithme de signature DSA a également permit d'aborder sous l'aspect de la programmation les mécanismes de signature.