Crypto Symétrique
TD Fonctions de hachage

Exercice I

Cet exercice concerne la méthodologie de la sécurité prouvable vue en CM, ainsi que les notions de sécurité des fonctions de hachage.

1. En regardant les jeux de sécurité présentés en cours, formalisez le jeu de sécurité correspondant à la propriété de la résistance aux deuxième préimages;
2. Pouvez-vous comparer votre jeu de sécurité avec les trois jeux formalisant la résistance aux collisions ?
3. Peut-on utiliser nos algorithmes pour trouver des collisions (vus en cours) pour casser cette propriété ?

Exercice II

Un fournisseur de service donne accès à sa plateforme en utilisant une authentification par identifiant et mot de passe. La transmission de ces accréditations se fait via un canal sécurisé (établi via un protocole standardisé comme TLS). La plupart des utilisateurs ont des droits d'accès basiques, mais quelques-uns en particulier ont un accès plus important et sont capables par exemple d'enregistrer des nouveaux utilisateurs.

Les données d'authentification des utilisateurs sont stockées dans une base de données, dont les entrées sont indexées par identifiant. C'est pourquoi chaque identifiant est unique.

1. Pour ce premier exercice, nous allons supposer que les entrées de la base de données ont le format $(identifiant\,;mdp)$. Nous allons supposer que les identifiants sont publics (c'est-à-dire, tout le monde peut les connaître). Un attaquant veut s'authentifier auprès du système en tant qu'un utilisateur honnête (peu importe lequel).
    a. Si l'attaquant n'a pas accès à la base de données, pouvez-vous indiquer une stratégie générique lui permettant de s'authentifier ?
    b. En cas de mot de passe oublié, le fournisseur de service peut-il envoyer (à une adresse mail confirmée) le mot de passe à l'utilisateur ?
    c. Si l'attaquant a accès à la base de données, quelle est sa stratégie d'authentification ?

2. Pour le deuxième exercice, le fournisseur de service utilise une famille de fonctions de hachage cryptographiques. Il commence en choisissant la clé : $k \xleftarrow{\$} K$. Les entrées de la base de données

ont le format $(identifiant\,;H(k;mdp))$. L'utilisateur s'authentifie comme avant : sur un canal sécurisé, il tape son identifiant et son mot de passe.

a. Lors de l'authentification d'un utilisateur, le fournisseur de service reçoit un tuple $(identifiant\,;mdp)$. Comment le fournisseur de service peut-il assurer l'authentification des utilisateurs maintenant ?
b. On suppose que l'attaquant n'a pas accès à la base de données du fournisseur de service. Son but est de s'authentifier en tant que n'importe quel utilisateur de la base de données.
    i. Qu'est-ce que l'attaquant doit réussir pour atteindre son but ?
    ii. Quelle est la propriété qui peut garantir au fournisseur de service que son protocole d'authentification est sécurisé contre ce type d'attaquant ?
c. Maintenant on suppose que l'attaquant (sans accès à la base de données) veut s'authentifier en tant qu'un utilisateur particulier. Pouvez-vous comparer cette attaque à l'attaque de la question précédente ?
d. L'adversaire réussit avoir accès à la base de données du fournisseur de service. Est-ce que la propriété mentionnée dans la question b.-ii. suffit pour garantir la sécurité de l'authentification maintenant ? Justifiez vos réponses.
e. Qu'est-ce qui change si la fonction de hachage est un oracle aléatoire ?
f. Le fournisseur de service utilise une fonction de hachage avec un codomaine $I = \{0,1\}^{128}$. Les mots de passe doivent contenir au minimum 8 charactères, chacun représenté sur 1 octet.
    i. Pour l'instant supposons que tous les mots de passe de tous les utilisateurs n'ont que 8 charactères. Notons par n le nombre d'utilisateurs. L'attaquant n'a pas la base de données. Pouvez-vous analyser la probabilité que l'attaquant trouve un de ces mots de passe ?
    ii. Si tous les mots de passe n'ont que 8 charactères, est-il possible d'avoir une fonction de hachage qui ne donne aucune collision sur ces mots de passe ? Et si on regarde la possibilité d'avoir des collisions sur le domaine entier des entrées (qui est, disons, $D = \{0,1\}^{2^{16}}$) ?
    iii. Qu'est-ce qui se passe si la taille des mots de passe moyenne dans le système est entre 8 et 12 charactères ?

Exercice III

Les fonctions de hachage sont utiles dans beaucoup de situations et sont incluses dans une grande variété de primitives et protocoles cryptographiques : par exemple dans les signatures numériques, dans l'authentification des messages, dans l'échange de clé, etc. Les constructions Merkle-Damgaard (vues en CM) peuvent garantir que ***si*** une fonction de compression est résistante aux collisions, alors la fonction de hachage obtenue l'est aussi -- cependant, nous ne savons pas tout à fait prouver la résistance aux collisions de la fonction de compressions. Si la sécurité d'une fonction de hachage est compromise, cela affecte la sécurité de tous les outils cryptographiques qui l'utilisent.

Cependant, il n'est pas toujours facile à remplacer le code du programme contenant l'ancienne fonction de hachage par un autre code, avec une nouvelle fonction de hachage (ceci s'appelle souvent *backward compatibility*). C'est pourquoi beaucoup de constructions essaient de mitiger les vulnérabilités d'une fonction de hachage en la composant avec une autre. Dans cet exercice nous allons explorer deux constructions spécifiques.

1. Prenons premièrement une famille de fonctions de hachage $H: \{0,1\}^{128} \times \{0,1\}^{2^{16}} \to \{0,1\}^{128}$ et une famille de fonctions de bourrage (padding) : $\mathrm{pad}: \{0,1\}^{128} \to \{0,1\}^{2^{16}}$, tel que, pour une entrée h représentée sur 128 bits, la fonction de bourrage rajoute le nombre manquant de 0s pour la représenter sur 2^{16} bits, i.e., $\mathrm{pad}(h) = h||0 \dots 0$. Ensuite, prenons une deuxième fonction de hachage $H^* : \{0,1\}^{128} \times \{0,1\}^{2^{16}} \to \{0,1\}^{128}$.
    a. Soit $\widehat{H}: \{0,1\}^{256} \times \{0,1\}^{2^{16}} \to \{0,1\}^{128}$ tel que, si $\hat{k}_{0\dots127}$ dénote les 128 bits les plus significatifs de la clé et $\hat{k}_{128\dots255}$ les 128 bits les moins significatifs d'une clé $\hat{k}$, alors $\widehat{H}(\hat{k};\, x) = H^*(\hat{k}_{0\dots127}; \mathrm{pad}(H(\hat{k}_{128\dots255}; x)))$.
        i. Supposons que H^* est résistante aux collisions ET résistante aux préimages. Supposons que H est résistante aux préimages, mais pas aux collisions. Que pouvez-vous conclure sur les propriétés de la fonction résultante $\widehat{H}$?
        ii. Maintenant, supposez que H^* est résistante aux collisions ET résistante aux préimages et que H est résistante aux collisions, mais pas aux préimages. Quelles sont les propriétés de la fonction $\widehat{H}$?
        iii. Et si H n'est résistante ni aux préimages ni aux collisions ?
    b. Mêmes questions pour $\widehat{H}: \{0,1\}^{256} \times \{0,1\}^{2^{16}} \to \{0,1\}^{128}$ avec $\widehat{H}(\hat{k};\, x) = H(\hat{k}_{0\dots127}; \mathrm{pad}(H^*(\hat{k}_{128\dots255}; x)))$.

2. Soient maintenant deux fonctions de hachage $H: \{0,1\}^{128} \times \{0,1\}^{2^{16}} \to \{0,1\}^{128}$ et $H^* : \{0,1\}^{128} \times \{0,1\}^{2^{16}} \to \{0,1\}^{128}$. Soit $\widehat{H}: \{0,1\}^{256} \times \{0,1\}^{2^{16}} \to \{0,1\}^{256}$ tel que, si $\hat{k}_{0\dots127}$ dénote les 128 bits les plus significatifs de la clé et $\hat{k}_{128\dots255}$ les 128 bits les moins significatifs d'une clé $\hat{k}$, alors $\widehat{H}(\hat{k}; x) = H(\hat{k}_{0\dots127}; x)\, ||H^*(\hat{k}_{128\dots255}; x)$.

   Supposons que H est résistante aux préimages, mais pas aux collisions, tandis que H^* est résistante aux collisions, mais pas aux préimages. Quelles sont les propriétés de la fonction résultante $\widehat{H}$?