TD 2 Fonctions de Hachage

Remarque

Les questions marquées d'une étoile ($\star$) sont optionnelles.

Propriétés de sécurité des fonctions de hachage - Stockage de Mot de Passe

Dans cette partie, on considère l'utilisation de SHA-1 comme fonction de hachage. Pour rappel, le condensat est sur 160 bits.

On suppose qu'un serveur souhaite utiliser cette fonction de hachage cryptographique pour protéger la confidentialité du mot de passe de ses utilisateurs. Pour chaque mot de passe, l'utilisateur va calculer son empreinte et envoyer le résultat sur le serveur.

Question 1

Proposez un protocole permettant à un utilisateur de s'authentifier sur le serveur a posteriori.

On suppose qu'un groupe de hackers a réussi à récupérer la liste des condensats stockés sur le serveur.

Question 2

Ces hackers souhaitent récupérer la valeur du mot de passe des utilisateurs. Quelle propriété de sécurité sur la fonction de hachage attaquent-ils ? Combien d'opérations devront-ils réaliser (discuter en fonction du choix de mot de passe) ? Les hackers peuvent-ils utiliser leurs résultats pour d'autres serveurs ?

Question 3

Les hackers se concertent et décident que ce qui les intéresse, ce n'est pas le mot de passe directement mais de s'authentifier à la place de l'utilisateur. Quelle propriété de sécurité sur la fonction de hachage attaquent-ils ? Combien d'opérations devront-ils réaliser ? Comparer à la question précédente.

Question 4

On suppose que les hackers disposent d'une table arc-en-ciel, dans quel(s) cas décrits ci-dessus peut-ils s'en servir ?

Propriétés de sécurité des fonctions de hachage - Vérification d'intégrité des fichiers

On considère de nouveau dans cette partie l'utilisation de SHA-1 comme fonction de hachage.

Un hébergeur publie sur son site un fichier de mise-à-jour critique d'un logiciel de gestion bancaire. Il publie également l'empreinte de ce fichier. Lors du téléchargement, une vérification d'intégrité s'effectue par comparaison de l'empreinte présente sur le site et l'empreinte calculé sur le fichier reçu.

Question 5

On considère que l'hébergeur est malveillant. Que peut faire l'hébergeur ? Quelle est la propriété de sécurité sur la fonction de hachage l'hébergeur essaye-t-il d'attaquer ? Combien d'opérations devrait réaliser l'hébergeur pour mener une attaque ?

Propriétés de sécurité des fonctions de hachage - Stockage de Mot de Passe avec sel

Lors d'un stockage d'un mot de passe dans une base de données, on applique en général un sel. Les données stockées sur le serveur sont alors h(M DP ||sel)||sel.

Question 6

Décrire le protocole permettant au serveur de vérifier le mot de passe entré par l'utilisateur.

Question 7

Quel est l'intérêt du sel ?

Question 8

Une fonction de hachage est-elle censée être rapide à calculer ? Dans le cas du stockage de mot de passe est-ce une bonne propriété ? Que pourriez-vous faire avec une fonction de hachage classique pour combler ce problème ?

PBKDF2 (Password Based Key Derivation Function 2) est une fonction de dérivation de clé. Elle prend comme paramètres une fonction pseudo-aléatoire, un mot de passe, un sel, un nombre d'itération et la taille de la clé désirée.

Question 9

Comment utiliser cette fonction dans le cas du stockage de mots de passe ?

Question 10

Le même groupe de Hackers que dans la première partie arrivent à mettre la main sur les données stockées sur le serveur. Peuvent-ils se servir des données recueillies pour attaquer d'autres serveurs ?

Question 11

Les hackers disposent d'une table arc-en-ciel. Obtiennent-ils un avantage à l'utiliser ?

Attaque sur la structure des fonctions de hachages

La plupart des fonctions de hachage (MD5,SHA1,SHA2) utilisent une construction de type Merkle-Damgard. La construction de Merkle-Damgard utilise une fonction de compression avec une entrée et une sortie de taille fixe. Un padding est effectué à l'entrée pour pouvoir avoir cette taille fixe. En supposant une taille de bloc de n, le padding va rajouter suffisamment de bits pour que le message soit multiple de n puis un bloc entier de taille n qui correspond à la longueur du message. Le message d'entrée est divisé en blocs qui sont envoyés à la fonction de compression qui prend en entrée le bloc et le résultat de la compression précédente. Un IV est appliqué à la première fonction. Une fonction de finalisation peut-être appliquée pour finir.

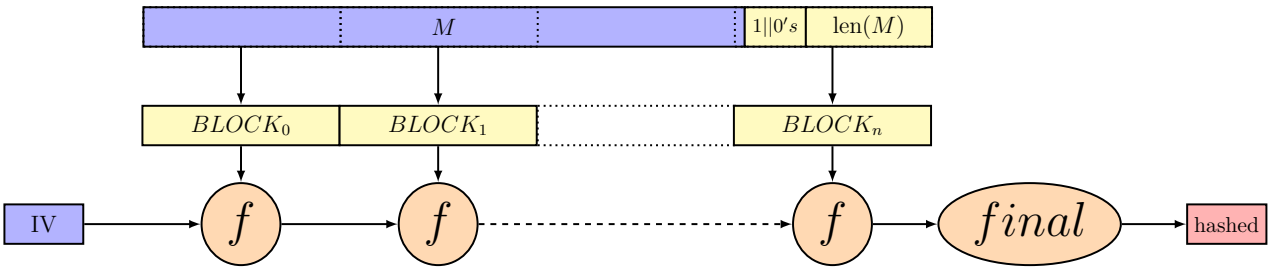

Figure 1: Construction Merkle-Damgard

Question 12

Si la fonction de finalisation est inversible (ou non appliquée, ce qui est le cas en pratique), quelle attaque pouvez-vous effectuer pour créer un hash correct d'un message non connu ? Pour vous aider, réfléchissez à la relation entre un état interne de la fonction et l'état précédent.

Question 13

Quelles sont les données dont vous avez besoin pour cette attaque ? Que se passe-t-il dans le cas où un utilisateur pour protéger l'intégrité de son message envoie un message m et h(m), un attaquant en homme du milieu peut-il casser le contrôle d'intégrité ?

Un HMAC (hash-based message authentication code) est un outil cryptographique permettant de vérifier l'intégrité et l'authentification d'un message. Il prend en paramètres un message m et une clé partagée k.

$$\text{HMAC}(m, k) = \text{h}((k \oplus c_1)||\text{h}((k \oplus c_2)||m))$$

avec c_1 et c_2 des constantes.

Question 14

Un HMAC est-il attaquable en utilisant l'attaque par un homme du milieu (peut-on en tant qu'homme du milieu casser l'intégrité ou l'authentification du message) ?