COMMUNICATION SCIENCE ET TECHNOLOGIE
No.24 , pp.13-21, June 2021
ENPO – Maurice AUDIN ASJP/EISSN 2773-3483
Date de soumission 20-04-2021
Date d'acceptation 05-05-2021

SECURITE DES FONCTIONS DE HACHAGE CRYPTOGRAPHIQUE

A. ZELLAGUI* – N. HADJ-SAID – A. ALI-PACHA

Laboratoire de codage et sécurité de l'information (LACOSI)

Université des Sciences et de la Technologie d'Oran USTO, BP 1505, El M'Naouer, Oran, Algérie

*E-mail auteur correspondant : amine.zellagui@univ-usto.dz, nim_hadj@yahoo.fr, a.alipacha@gmail.com

Résumé-*Les fonctions de hachage cryptographique sont un élément fondamental de la sécurité de l'information et sont utilisées dans de nombreuses applications et protocoles de sécurité tels que la signature électronique, l'HMAC et la blockchain pour garantir l'intégrité et l'authentification des données. Récemment, des chercheurs ont remarqué de graves failles de sécurité et des vulnérabilités dans les fonctions de hachage des familles de MD et de SHA les plus utilisées, telles que : MD5, SHA-1 et SHA-2 qui sont basées sur la construction Merkle-Damgård. Parmi les vulnérabilités, nous pouvons citer la multi-collision et l'extension de longueur. Cet article étudie les attaques et les faiblesses connues contre les fonctions de hachage et propose des solutions pour éviter ces attaques.*

Mots-clés -Cryptographies/ Hach/ Collision/ Construction/ SHA-1/ MD5

Abstract *-Cryptographic hash functions are a fundamental element of information security and are used in many applications and security protocols such as digital signatures, HMAC and block chain to ensure data integrity and authentication. Recently, researchers have noticed serious security flaws and vulnerabilities in the hash functions of the most widely used MD and SHA families, such as: MD5, SHA-1 and SHA-2 which are based on the Merkle-Damgård construction. Vulnerabilities include multi-collision and length extension. This article studies the known attacks and weaknesses against hash functions and offer solutions to avoid these attacks.*

Keywords -Cryptography/ Hach/ Collision/ Construction/ SHA-1/ MD5.

1 .INTRODUCTION

Aujourd'hui, la cryptographie est la technologie fondamentale utilisée pour protéger les informations qui circulent sur l'internet. Chaque jour, la cryptographie est utilisée pour protéger le contenu des transactions sur le web, des courriels, des groupes de discussion, des discussions en ligne, du stockage des données, de la gestion des mots de passe, des conférences web et des appels téléphoniques lorsqu'ils sont envoyés sur l'internet. Sans cryptage, un arnaqueur, un voleur, un fournisseur d'accès à Internet, une compagnie de téléphone, une entreprise hostile ou un fonctionnaire qui a un accès physique aux fils transportant vos données pourrait en

espionner le contenu. Avec les outils cryptographiques, il est possible de protéger les données contre tous les types d'attaques, parmi ces outils figurent les fonctions de hachage.

La fonction de hachage joue un rôle fondamental dans l'authentification de l'intégrité des données [1]. Il s'agit d'un algorithme mathématique qui mappe le message M de taille arbitraire avec un condensé de hachage de taille fixe h, qui est une « empreinte digitale » H(M) unique pour chaque message. Une fonction de hachage doit être capable de résister à tous les types d'attaques cryptanalytiques connus [2]. En cryptographie théorique, le niveau de sécurité de la fonction de hachage a été défini à l'aide des propriétés suivantes : résistance de pré-image, deuxième résistance de pré-image et résistance aux collisions.

La fonction de hachage peut être classée en hachage sans clé et hachage avec clé. La fonction de hachage sans clé n'a pas de clé, telle que MD5 [3], SHA-1 [4], SHA-2 [5] et SHA-3 [6], généralement utilisée pour assurer l'intégrité et l'authentification des données. La fonction de hachage avec clé prend deux entrées : message et clé secrète, et est pratique pour les protocoles de sécurité de la couche transport.

Récemment, les fonctions de hachage traditionnelles telles que MD4, MD5, RIPEMD [7] et HAVAL-127 [8] ont été attaquées avec succès par X. Wang [9] et SHA-1 par Marc Stevens [10]. Bien que la SHA-3 puisse résister à certaines attaques, sa vulnérabilité a été découverte par une cryptanalyse tierce et une collision pratique avec 3 itérations [11]. Dans cet article, nous mettons en évidence les attaques et les problèmes les plus importants rencontrés par les fonctions de hachage telles que : la vulnérabilité de la construction Merkle-Damgård, construction AD-HOC, l'attaque par extension de longueur, la multi-collision, l'attaque d'anniversaire, l'attaque MITM « meet in the middle », cryptanalyse rotationnel et l'attaque force brute.

Cet article est organisé comme suit: la section 2 présente les vulnérabilités des constructions de hachage ; la section 3 décrit les vulnérabilités des fonctions de hachage, tandis que la section 4 explique les suggestions ; la section 5 présente une analyse comparative de la complexité des fonctions de hachage, enfin, la section 6 conclut l'article.

2. VULNERABILITES DES CONSTRUCTIONS DE HACHAGE

2.1 CONSTRUCTION DE MERKLE DAMGARD

Les fonctions de hachage les plus traditionnelles telles que MD4, MD5, SHA-1 et SHA-2 utilisent la construction Merkle-Damgård (MD). Elle a été conçue par Merkle [12] et Damgård [13] en 1989. Une fonction de compression est utilisée de manière

itérative. Le message M, après avoir été préparé, est divisé en blocs de taille fixe M1, M2 ..., Mt, et la fonction de compression f traite les blocs l'un après l'autre jusqu'au dernier bloc Mt. La Figure 1 montre la construction Merkle-Damgård, où IV est le vecteur initial, H (M) est le hachage final et h est la longueur du hachage.

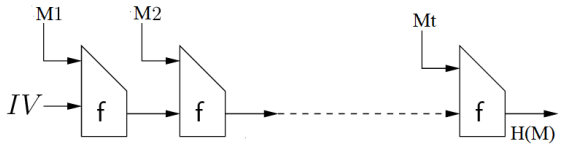

Fig.1 Construction de Merkle-Damgård

La construction de Merkle-Damgård prouve que la sécurité de la fonction de hachage est basée sur la sécurité de la fonction de compression. Ainsi, pour construire une fonction de hachage résistante aux collisions, il suffit de concevoir une fonction de compression résistante aux collisions. Toutefois, des résultats récents mettent en évidence certaines limites intrinsèques de l'approche Merkle-Damgård, notamment la vulnérabilité aux attaques à collisions multiples [14] et aux attaques par extension de longueur.

2.1.1 ATTAQUE PAR EXTENSION DE LONGUEUR

Les fonctions de hachage les plus traditionnelles basées sur la construction de Merkle-Damgård et d'autres conceptions ne peuvent pas résister à ces attaques, où un message M est divisé en n blocs et H(M) est le hachage final de M. Un autre message M' avec n + k blocs, où M et M' ont le même bloc n (M1 = M1', M2 = M2', ..., Mn = Mn'), la valeur de hachage H(M) est la valeur de hachage intermédiaire après n blocs lors du calcul de H(M') (Fig 2). Cela peut créer un problème en pratique [15] [16].

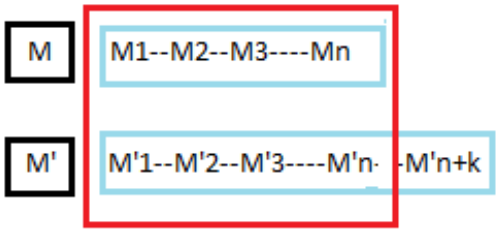

Fig.2 Attaque d'extension de longueur

2.1.2 ATTAQUE PAR MULTI COLLISIONS

Un autre problème dans la construction de Merkle-Damgård est l'attaque par multi-collisions. Par exemple, dans une fonction de hachage itéré, les collisions multiples peuvent être construites plus efficacement en utilisant des collisions internes. Tout d'abord, une paire de messages M1, M1' est recherchée, ce qui donne une collision interne à partir du IV (vecteur initial). Ensuite, une paire de messages M2, M2' est recherchée, ce qui donne une collision interne à partir de l'état L0 après traitement de M1. Si cette opération est répétée k fois, nous obtenons k collisions internes et nous pouvons construire 2^k messages en choisissant l'un ou l'autre des messages à chaque étape. La Figure 3 montre l'attaque à collisions multiples, où IV est le vecteur

initial, L est l'état interne et H est le hachage final.

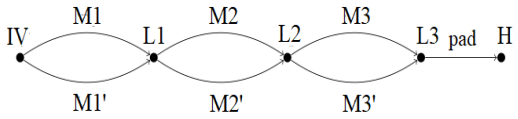

Fig.3 Attaque par multi-collisions

2.2 CONSTRUCTION AD-HOC

La plupart des fonctions de hachage sont basées sur des constructions ad-hoc, ou des fonctions de compression ont été basées sur des cartes chaotiques, des automates cellulaires, de l'ADN ou des générateurs pseudo-aléatoires. Généralement, ces fonctions de hachage ne sont pas utilisées pour des applications réelles et sont restées d'intérêt académique. L'une des raisons de cette situation est l'absence de vérification de sécurité appropriée par des efforts cryptanalytiques. En d'autres termes, la plupart des fonctions de hachage utilisées aujourd'hui, telles que [22] et [23], ne se sont pas avérées résistantes aux collisions. Ces hachages ne sont pas basés sur des fonctions purement mathématiques. Un algorithme de hachage doit être conçu en tenant compte de la facilité d'analyse.

3. VULNERABILITES DES FONCTIONS DE HACHAGE

3.1 ATTAQUE DES ANNIVERSAIRES

L'attaque d'anniversaire est une attaque typique pour casser la fonction de hachage où l'attaquant a besoin de $2^{\square/2}$ éléments choisis au hasard pour avoir 50% de chance de collision [17], où h est la longueur du hachage. Si nous avons une valeur de hachage h de 64 bits, l'attaquant n'a besoin que de 2^{32} tentatives, pour avoir 50% de chance de collision. Avec le développement de la technologie et la vitesse de traitement des données, la valeur de hachage de 128 bits est devenue insuffisante et elle sera cassée tôt ou tard. La longueur de la valeur de hachage h doit être augmentée de plus de 128 bits.

3.2 ATTAQUE PAR FORCE BRUTE

Les fonctions de hachage ont de bonnes propriétés cryptographiques si elles sont résistantes à la pré-image et à la deuxième pré-image, comme suit :

-Pré-image : étant donné un hachage X choisi au hasard, il trouve un message M tel que H(M) = X.

-Deuxième pré-image : étant donné un message M choisi au hasard, il trouve un message M' tel que H (M) = H (M').

L'attaque par force brute est le moyen le plus rapide de calculer ces attaques ; il sélectionne au hasard les valeurs du message M et essaie chaque valeur jusqu'à ce qu'une collision se produise. Pour les attaques pré-image et seconde pré-image, le

niveau d'effort est proportionnel à $2^{?}$, où h est la longueur du hachage.

3.3 CRYPTANALYSE ROTATIONNEL-LE

La Cryptanalyse Rotative [18] a été introduite en 2010 et a été classée comme une attaque générique car elle s'applique à presque tous les algorithmes basés sur trois opérations principales telles que l'addition modulaire, les rotations (décalages) et le XOR (OU exclusif). Elle peut aussi suivre la structure spécifique comme l'a suggéré Merkle-Damgård pour les fonctions de hachage. Un point important est que la complexité de l'attaque en rotation dépend uniquement du nombre d'additions modulaires et non du nombre de XOR et de rotations. Cette attaque a brisé la résistance aux collisions jusqu'à 53 des 72 itérations de Skein-256 et 57 des 72 itérations de Skein-512 [19]. De plus, dans les anciennes conceptions, l'ensemble du message est utilisé directement dans la fonction de compression f, ce qui facilitera les attaques par cryptanalyse.

3.4 ATTAQUE PAR RENCONTRE AU MILIEU (MITM)

L'attaque de type « meet in the middle » vise à bloquer les fonctions cryptographiques du cryptage. L'intrus applique des techniques de force brute au texte en clair et au texte chiffré d'un chiffrement par blocs. L'intrus tente ensuite de chiffrer le texte en clair avec différentes clés pour obtenir un texte chiffré intermédiaire (texte qui a été chiffré avec une seule clé). En même temps, il tente de décrypter le texte chiffré avec différentes clés, en cherchant un bloc intermédiaire de texte chiffré identique à celui obtenu par le chiffrement du texte en clair. S'il y a une correspondance de chiffrement intermédiaire, il est très probable que la clé utilisée pour chiffrer le texte en clair et la clé utilisée pour déchiffrer le chiffrement sont les deux clés de chiffrement utilisées pour le chiffrement par blocs.

En 2008, Aoki et Sasaki ont proposé la « Meet-in-the-Middle Attack » [20] (attaque de rencontre au milieu) pour les fonctions de hachage. Cette attaque utilise le calendrier des messages qui consiste en des permutations de mots de messages. Il est appliqué avec succès sur les séries réduites SHA-1 [21], où les auteurs ont utilisé la combinaison linéaire des mots du message.

4. SUGGESTIONS

A. Pour les constructions de hachage: Toutes les fonctions de hachage qui sont basées sur la construction Merkle-Damgård sont vulnérables et ne sont pas capables de résister aux attaques connues. Pour cela, nous suggérons l'utilisation de constructions alternatives telles que : construction en éponge [24].

C'est une construction itérée capable de produire des sorties de taille arbitraire. Contrairement à la construction Merkle-Damgàrd qui est basée sur une fonction de

compression, la construction en éponge repose sur une transformation à taille fixe (une permutation f), opérant sur les mots de b bits (b=r+c), où r est le débit (bitrate) et c est la capacité, les deux sont initialement mis à zéro. En pratique, toutes les instances connues de cette construction sont basées sur une permutation. La Figure 4 montre la construction en éponge, où H est le hachage final et f est la fonction de permutation.

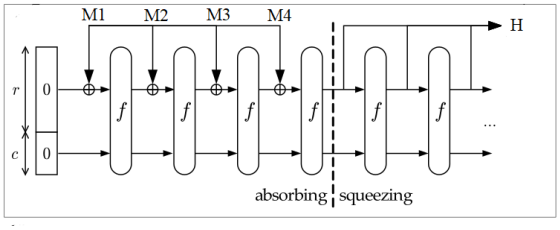

Fig. 4 La construction en éponge

Le message d'entrée M est d'abord rempli et divisé en blocs de messages de longueur de r bits (M1, M2…). La construction en éponge se compose de deux étapes, l'absorption (absorbing phase) et la compression (squeezing phase). Dans l'étape d'absorption, des blocs de taille r bits du message d'entrée (Mi sur la figure 4) sont XORé avec les premiers r bits de l'état, suivi d'une application de la fonction de permutation f. ceci est répété jusqu'à ce que tous les blocs de messages soient traités. Dans la phase de compression, les premiers r-bits sont renvoyés en sortie de la construction éponge, à nouveau suivis par les applications de la fonction de permutation f. La longueur de la phase de compression est spécifiée par l'utilisateur en fonction de la longueur de sortie souhaitée.

En général, si l'état interne L est suffisamment large par rapport à la taille de sortie h, il est possible d'avoir une fonction de hachage résistante aux collisions. Dans la construction en éponge, où la longueur de l'état interne r+c est plus grande que la taille du bloc Mi, et le hachage final est inférieur à la longueur de l'état interne.et donc, peut être efficace contre ces attaques.

B. Pour les fonctions de hachage : La longueur de la valeur de hachage doit être augmentée au-dessus de 128 bits pour éviter les attaques de force brute et d'anniversaire et doit être flexible et capable de générer une valeur de hachage H de taille h : 224, 256, 384 ou 512 bits. Ensuite, les fonctions de compression doivent contenir des fonctions non linéaires, des transformations irréversibles (à sens unique) et des fonctions logiques pour éviter les attaques pré-image, deuxième pré-image et rencontre au milieu.

5. ANALYSE COMPARATIVE

Dans cette section, une analyse comparative entre certaines fonctions de hachage basées sur la construction Merkle-Damgard (MD) telles que: MD4, MD5, SHA1, SHA2, RIPEMD et HAVAL-127, et le SHA-3 basé sur la construction en éponge. Cette analyse comparative inclut la complexité de chaque algorithme. Le tableau 1 présente

les statistiques de la complexité de chaque algorithme.

On peut observer que la fonction de hachage SHA-3 qui est basée sur la construction d'éponge a une meilleure valeur de taille dans l'état interne (1600 bits), et supérieure à la taille de hachage h, ce qui signifie que la construction d'éponge peut éviter l'attaque multi-collision et l'attaque d'extension de longueur avec une complexité de 2^{1600}.

Tableau 1. La complexité des fonctions de hachages

Algorithme / **Paramétre**	**MD4**	**MD5**	**SHA-1**	**SHA-2**	**SHA-3**	**RIPEMD**	**HAVAL-127**
Construction	MD	MD	MD	MD	ÉPONGE	MD	MD
Hash (h bits)	128	128	160	≥ 224	≥ 224	≥ 224	≥ 128
Etat interne (bit)	128	128	160	≥ 256	1600	≥ 256	≥ 256
Itération	48	64	80	64 or 80	24	≥ 96	96
Bloque (L bits)	512	512	512	≥ 512	<576	512	1024
Taille de mot	32	32	32	32 or 64	64	32	32
Multi-collision	2^{128}	2^{128}	2^{160}	$\geq 2^{256}$	2^{1600}	$\geq 2^{224}$	$\geq 2^{128}$
Extension de longueur	OUI	OUI	OUI	OUI	NON	OUI	OUI
Anniversaire	2^{64}	2^{64}	2^{80}	$\geq 2^{112}$	$\geq 2^{112}$	$\geq 2^{112}$	$\geq 2^{64}$
Force brute	2^{128}	2^{128}	2^{160}	$\geq 2^{224}$	$\geq 2^{224}$	$\geq 2^{224}$	$\geq 2^{128}$
Collision	OUI	OUI	OUI	Théorique	NON	OUI	Théorique

D'autre part, les fonctions de hachage basées sur la construction Merkle-Damgard telles que: MD4, MD5, SHA-1, SHA-2 (théorique), HAVAL-127 et RIPEMD, ont une faible complexité et peuvent être réduites avec l'attaque de rencontre au milieu (MITM) selon [20].

6. CONCLUSIONS

Cet article a présenté une étude des attaques et des vulnérabilités les plus connues contre les fonctions de hachage, telles que : la vulnérabilité de la construction Merkle-Damgård, construction AD-HOC, l'attaque par extension de longueur, la multi-collision, l'attaque d'anniversaire, l'attaque MITM « meet in the middle », cryptanalyse rotationnel et l'attaque force brute.

L'objectif principal de cet article était d'obtenir des informations sur le type d'attaques bien connues appliquées aux fonctions de hachage et de décrire brièvement comment nous pouvons développer la fonction de hachage en étudiant ces attaques, en tenant compte de nos suggestions sur les constructions et les fonctions de hachage. Ainsi, les fonctions de hachage en future, seront moins sujettes à ces attaques et offrent une sécurité élevée.

La sécurité de la fonction de hachage dépend de la propriété de résistance aux collisions des fonctions de compression. La méthode de construction de Merkle-Damgård n'a pas réussi à préserver cette importante propriété de sécurité. Pour cette raison, nous suggérons d'utiliser une construction en éponge, grâce à ses propriétés telles que : l'étape d'absorption, l'état interne large et la sortie choisis par l'utilisateur.

Pour les problèmes des fonctions de hachage, nous avons suggéré d'utilise des fonctions non linéaires, des transformations irréversibles, des fonctions logiques et une taille de hachage élevée, pour augmenter la complexité de la fonction de compression et éviter les attaques préimage, deuxième préimage, force brute et rencontre au milieu.

Dans le futur, nous allons essayer de développer une nouvelle fonction de hachage à clé secrète basée sur la construction en éponge et une carte chaotique, qui est capable de résister à tout type d'attaques mentionnées sur cet article.

REFERENCES

[1] L. Chi, X. Zhu, « Hashing Techniques: A Survey and Taxonomy » . ACM Computing Surveys, Vol 50 , No 1. April 2017

[2] S. Yago, C. Estebanez, D. Quintana, P. Isasi, «Evolutionary hash functions for specific domains.» Applied Soft Computing vol 78 , 2019.

[3] R. Rivest, « The MD5 Message-Digest Algorithm », RFC 1321, MIT LCS and RSA Data Securit y, Inc 1992

[4] D. Eastlake, P. Jones, « US Secure Hash Algorithm 1 (SHA1) » , RFC3174, September 2001

[5] D. Eastlake, T. Hansen, « US Secure Hash Algorithms (SHA and HMAC-SHA) », RFC4634, July 2006

[6] M. Dworkin, « SHA-3 standard : Permutation-based hash and extendable-output functions » (No. Federal Inf. Process. Stds. (NIST FIPS)-202), 2015

[7] H. Dobbertin, A. Bosselaers, « RIPEMD-160: A strengthened version of RIPEMD », vol 1039. Springer, Berlin, Heidelberg, 1996

[8] Y. Zheng, J. Pieprzyk, « HAVAL - A One-Way Hashing Algorithm with Variable Length of Output ». In: ASIACRYPT. LNCS, pp. 83–104. Springer, Heidelberg (1992)

[9] X. Wang, D. Feng, X. Lai. « Collisions for hash functions MD4, MD5, HAVAL-128 and RIPEMD », Cryptology ePrint Archive, report 2004

[10] M. Stevens, E. Bursztein, P. Karpman, « The first collision for full SHA-1 », CWI Amsterdam, Google Research 2017

[11] G. Jian, G. Liao, G. Liu, « Practical collision attacks against round-reduced SHA-3 », Journal of Cryptology 33, no. 1 ,2020

[12] M. Ralph, « A certified digital signature ». In: Conference on the Theory and Application of Cryptology. Springer, New York, NY, 1989 [13] I. Damgard, A design principle for hash functions. In: Conference on the Theory and Application of Cryptology. Springer, New York, NY, 1989

[14] A. Joux, « Multicollisions in Iterated Hash Functions: Application to Cascaded Constructions », Crypto 04, LNCS, vol. 3152, 2004

[15] B. Denton, R. Adhami, « Modern Hash Function Construction ». In: Proceedings of the International Conference on Security and Management (SAM). 2011

[16] N. Ferguson, B. Schneier, T. Kohno, « Cryptographic Engineering: Design Principles and Practical Applications ».: Wiley Publishing, Inc. 2010

[17] Y. Zhao, « Optimizing Hash Strategy to Avoid Birthday Attack ». In Journal of Physics: Conference Series (Vol. 1486, p. 032004). April 2020

[18] K. Dmitry, I. Nikolić, « Rotational cryptanalysis of ARX », International Workshop on Fast Software Encryption. Springer, Berlin, Heidelberg, 2010.

[19] D. Khovratovich, I. Nikolić, C. Rechberger, « Rotational rebound attacks on reduced Skein ». In International Conference on the Theory and Application of Cryptology and Information Security (pp. 1-19). Springer, Berlin, Heidelberg. December 2010

[20] K. Aoki, Y. Sasaki, « Preimage attacks on One-Block MD4, 63 steps MD5 and more ». SAC, vol. 5381. New York: Springer, 2008

[21] K. Aoki, Y. Sasaki. « Meet-in-the-Middle preimage attacks against reduced SHA-0 and SHA-1 ». Advances in Cryptology - CRYPTO, vol. 5677, 70–89. New York: Springer. 2009

[22] F. Salari, Z. Mirakabad, M. Sadeghi, « A new hash function and its use in read mapping on genome ». AUT Journal of Mathematics and Computing, 2020

[23] J. Teh, K. Tan, M. Alawida, « A chaos-based keyed hash function based on fixed point representation » Cluster Computing, 22(2), 649-660. 2019

[24] B. Guido, D. Joan, Michaël, et al. « Keccak sponge function family » main document. Submission to NIST (Round 2), vol. 3, 2009