Hash Function: Types, Benefits and Risks

Zubair Ahmed Rehmani

Abstract— **This report will be covering hash functions, their types and uses. What are their benefits and any security flaws associated with them. There are multiple hash families and numerous functions, some that are outdated but still in use. This report tends to focus on the types of hash functions, their benefits, and drawbacks.**

Keywords—hash, functions, SHA, MD5, digest, hash sum

*Introduction (***What are hash functions***)*

Cryptographic hash functions are utilized to convert a string, message, and data of any arbitrary size or length to a data of a fixed string. Hash functions are irreversible unlike encryption. These functions are utilized to map data of random size to fixed length string referred to as a hash value, hash sum, digests and even hashes.

The underlying purpose of hash functions is to keep the integrity of data. A hash function has to be fast but not fast enough. A good hash function should be able to produce a hash sum of a message, data, and files within couple of seconds and it must be irreversible so the input in the hash function cannot be determined by its output (hash value).

Furthermore, a hash function is built upon different elements that includes keys, hash table and value. A hash function works by mapping a significant length of integers or string to a smaller integer that becomes the index for that string or array of numbers, in the hash table. The two elements that are crucial are the key and value. We can find the value by using a function that has the ability to map values to keys. The key for an object which could be anything, can be computed by using a hash function. For instance, array A consisting of random data or strings, b being its key will simply allow us to find the contents of the array A by looking up A[b].

Lastly, we can categories hash functions into two types. The first being the keyed hash functions (hash functions that use a secret key) and the second being the Un-keyed hash function (hash function that doesn't use a secret-key)[5]. Hash functions are keyed are described as MAC (message authentication code). However, the phrase hash function mainly associates with the Un-keyed hash functions. It is possible to further classify unkeyed or merely hash functions (sometimes called MDC - Manipulation Detection Codes - based on the properties they satisfy into OWHF, CRHF and UOWHF [19].

A. Background of hash functions

Hash functions date back to 1970's and the proposals seemed to arise more in 1980's until in 1993 the first hash function SHA-0 was updated by the federal authority (NIST) to SHA-1, and it was later published in 1995. Fast forward today, the current most in use cryptographic hash function is SHA-2 which includes SHA-256 and SHA-512 and others. SHA is an abbreviation for Secure Hashing Algorithm which is issued and maintained by the National Institute of Standards and Technology (NIST).

B. Types of Hash Functions

There are numerous types of hash functions some of which include SHA, MD, and CRC. These are the families of hash functions in which there are further types of hash functions for example just the SHA family has six distinct hash functions consisting of SHA-0, SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512[14]. Furthermore, the MD hash family also has four different hash functions that includes MD2, MD4, MD5 and MD6 [20]. The list of hash functions persists beyond. However, these are some of the most adopted families of hash functions in the present era. If we were to take a deeper insight of what the numbers next to the hash families represent, then we come to know that all these numbers have an affect on the output of the hash function. Let us take SHA-256 for instance. The SHA-256 function produces 256-bit hash value or (32 bytes) likewise the SHA-512 outputs 512 bits of hash value or (64 bytes).

There are approximately 4 different methods that the hash functions are built upon, which may also alter or represent the type of the hash function is or the output it produces.

C1. Division method

The division method is by far the best one as it only requires one operation of division.

Formula:

$$H(K) = k \pmod{M}$$

H signifies the hash function

K signifies the key value

M signifies the size of the hash table

Here it is more satisfactory if M is a prime number or has fewer devisors as it can ensure that the keys are dispersed evenly and reduce the risk of collision. In this method the hash function mainly depends on the remainder of a devision.

To demonstrate this, we can suppose three different keys or numbers we have to store in the index of the hash table let's take 10, 15 and 12. Now using the division method, we are going to find the hash value of these numbers. The formula suggests that we should find the H and K.

5 being the M (The size of the hash table)

$$H(10) = 10 \pmod 5$$

$$H(10) = 10 \pmod 5 = 0$$

This suggests that we should store the key or our data at index 0. To further explain this method, we take 15 as our key and run it through our formula. This is where collision will be very likely to occur.

$$H(15) = 15 \pmod 5 = 0$$

Since we know that there is already a key stored at index 0, we now have a collision. There are various methods of dealing with the collision, in this case we can use the linear probing which checks to see what other indexes are vacant. Using this we can see that index 1is empty hence we change the hash value and store the key 15 at index 1. Which becomes the new hash value for the key 15.

Lastly to make it clear we will perform this calculation with 12.

$$H(12) = 12 \pmod 5 = 2$$

Hence, we place our key at index 2 as it is already vacant in the hash table and there is no risk of collision so 12 can reside at index 2.

C2. Mid Square Method

This method requires two operations in order to calculate the hash value. Firstly, we square the key's value. Secondly, we extract the r digits as the hash value.

$$H(K^2) = L$$

This method requires squaring our K value and taking the number located in the middle and storing our key at that index. To simplify this further, we can take the K value 10 and square it. We get 100 as the result and we discard the 1 from left and the 0 from the right and the r value we are left with it 0 so we store 10 at index 0.

C3. Folding Method

In this method we simply use addition to add the numerical values together and store the key at the outcome of our addition.

$$H(K) = K^1 + K^2 + K^3 + K^n$$

To clarify this method, we can take number 10 as the example and separate it into two parts 1 and 0 or K^1 and K^2. Then we simply add the two numbers 1 + 0 which leads to 1. The outcome can be used as the index on the hash table to store the key K at. In this case we can store 10 at index 1.

C4. Multiplication Method

This method requires that value of A (*A being the constant value*) should be 0<A<1. Then we multiply the key with A and isolate the fractional part of kA. Lastly, we multiply the outcome of the previous step by M. The resulting hash value is achieved by obtaining the floor of the outcome achieved in the last step.

$$H(K) = \text{floor}(M(kA \bmod 1))$$

This method is somewhat complex compared to the previous methods. The best advantage of this method is that it can perform well between any value of zero and one, however some values may deliver better results than the other. (Khinchi & Teja, 2022)

C. *Benefits of hash functions*

There are uncountable benefits of hash functions. Starting with the main purpose of Hash Functions. Hash Functions are greatly beneficial when it comes to checking the integrity of data whether it is a file, message, or any other type of data. As today we know that data that travels over the internet is always at risk of being manipulated by attackers and is also vulnerable to other risks [15]. Therefore, hash functions exist, to keep the integrity of data as just as a single 'bit' shift in the data will cause the whole hash value to be changed or simply, obtaining a new hash value as the result.

D1. Authenticating user logins

Hash Functions are also commonly used in authenticating users at logins. This is achieved by storing the passwords in hash digest. This provides security to the users as even the administrators of the systems are unable to access the plain password. When a user tries to login, the entered credentials are computed or ran through a hash function that produces the hash digest of the entered credentials, lastly the computer will compare the stored hash value with the newly computed hash value, if the hash values are same, the user will be authenticated else denied access to the system. (Sobti & Geetha, 2012).

D2. Digital Time-Stamping

It is evident that the majority of text, audio, documents, and videos are obtainable digitally and that it is possible to alter the content of these documents using simple tools and methods. In such cases, a mechanism is needed to verify when the document was produced or altered. In his thesis, Rompay [5] suggests a number of ways to achieve the desired goal using digital timestamps. The use of trusted third party is involved in some methods, timestamped chains are also used, and Merkle tree is used by a few. In Romay's paper [5], it was noted that digital time stamps play a critical role in guarding the rights of intellectual property, promising audit procedures that are strong, and ensuring non-repudiation guarantees. As described by [6], digital signatures and One-way hash functions can be utilized for digital time stamping before [5].

D3. Session-Key Derivations

The use of One-way hash functions allows the generation of sequential session keys for subsequent communication sessions that are protected by hashing. Beginning from master key K_0, as an example, the initial session key could be $K_1 = H(K0)$, the next session key could be $K_2 = H(K_1)$, further etc. A key management scheme is illustrated by [7] that uses a hash function along with encryption to generate session keys.

D4. Additional Applications

Hash functions can also be useful for indexing data in the hash tables, fingerprinting, detecting duplicate files, identifying files uniquely, generating random numbers, and detecting accidental data corruption. With such a wide range of applications, Hash Functions should not be classified as belonging to one particular cryptographic subbranches [16]. This cryptographic mechanism deserves separate classification for itself. Almost all places in cryptology where efficient information processing is necessary, uses hash functions [18].

D. Associated risks

Even though there are numerous benefits of hash functions, but we cannot ignore the challenges this mechanism has to offer. The main challenges associated with hash functions is collision and attacks. A hash function is only considerable when it is collision free.

Additionally, if we look into the risk of attacks then there are multiple types of attacks that exist to exploit hash functions. An attack on hash function occurs when one of its security properties is violated (extended, basic, or certification-al) [17]. An adversary is capable of breaking the pre-image property, for example, by creating a message that hashes to a particular hash. Even though the hashing / compression algorithm may not be vulnerable, a breach of certificate-al properties could be an essential indication of flaw in the algorithm. An attack on certification-al properties should be managed with a robust hash function, according to [8]. Some other attacks to mention are brute force attack, Length extension attack, herding attack and meet in the middle attack.

E1. Brute force attack

The brute force attack can be used on any hash function regardless of its structure or other details of its operation. Extraction of an encryption scheme's key, these attacks are similar to exhaustive search attacks or brute-force key recovery attacks [21]. It is the output bit size of a hash function that determines its security. In order to defend a hash function against different brute force standard attacks, one has to use the following effort level:

Pre-Image Attack: For a given n-bit digest h of the hash function H( ), in this attack, the adversary evaluates H( ) with every conceivable input message M until it returns the value h.

Second Pre-image attack: A brute force attack requires 2^n effort. An attacker tries H( ) on each potential input message $M' \neq M$ up until H(M) is obtained for a given message M. A brute force attack requires 2^n effort to complete.

Collision Attack: Brute force attack requires 2n/2 effort. For a particular hash function H, the attacker searches for two messages M and M' such that M≠M' and H (M) = H (M^1). A message matching the intercepted message's hash code can be found by the adversary by trying on average $2^n/ 2 (= 2^{n-1})$ messages. If the plain text attack is chosen (depending on the Birthday Paradox), then the collision effort will be $2^{n/2}$ instead of 2^{n-1} [9]. Another name for it is the Birthday attack.

E2. Length Extension attack / Padding attack

Length extension attack is known weakness of Merkle Damgard construction. In the case of h = H(M), M'' and h' can be calculated straightforwardly, such that h' = H(M | | M), despite indeterminate M (but seen length | M |). For computing H(M || M'), H(M) is used as an internal hash. In [8]'s classification, it can be further divided into two types, Type A and B. Whether or not the original message contains length padding, determines the categorization. By the use of this attack, it's feasible to determine hashes of lengthy messages starting with the primary message and including the padding needed to reach multiple block sizes from only the hash of the original message [10]. Even today, we are observing vulnerabilities related to the length extension attack, which was first studied by [11] in 1992.

E3. Herding Attack

Herding attack involves a brute force attacker finding many collisions on the hash functions, and choosing the appropriate suffix then causes any message's beginning point to be "herded" to the hash value. This attack was presented by [12] based on MerkleDamgard structure.

E4. Meet in the middle attack

An attack similar to the birthday attack can be performed on hash functions that use compression functions. Attackers can construct messages to match certain digests by generating r1 samples and r2 samples for the first and end parts of the simulated message. In the next step, by advancing from the starting value and reversing from the hash value, the adversary moves forward and backwards. Upon finding the meeting point, the message parts are concatenated to produce a bogus message that is hashed. [13]

E. Conclusion

Reflecting upon most aspects of hash functions, I would agree with most authors when it comes the importance of hash functions. Hash functions play a major role in cyber security and protecting sensitive data by keeping the integrity of the data. In this review I attempted to cover basics to some advance types of hash functions along with vulnerabilities and possible threats. This review sets the pivot and tends to deliver intermediate level of knowledge to researchers that seek to explore this mechanism in cyber security.

REFERENCES

[1] Rasjid, Z.E., Soewito, B., Witjaksono, G. and Abdurachman, E. (2017). A review of collisions in cryptographic hash function used in digital forensic tools. *Procedia Computer Science*, 116, pp.381–392. doi:10.1016/j.procs.2017.10.072.

[2] Abdulmajed, K. (2013). *HASH FUNCTIONS: ANALYSIS STUDY.* [online] Available at: http://210.48.222.250/bitstream/123456789/5402/1/t00011282816KhanssaaMunthir_SEC_24.pdf [Accessed 11 Nov. 2022].

[3] I. Ghafir and V. Prenosil, "Blacklist-based Malicious IP Traffic Detection," Global Conference on Communication Technologies (GCCT). Thuckalay, India: pp. 229-233, 2015.

[4] Sobti, R. and Ganesan, G. (2012) *(PDF) cryptographic hash functions: A review - researchgate*, *researchgate*. IJCSI. Available at: https://www.researchgate.net/publication/267422045_Cryptographic_Hash_Functions_A_Review (Accessed: November 19, 2022).

[5] Khinchi and Teja (2022) www.geeksforgeeks.org. (n.d.). *Hash Functions and list/types of Hash functions - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/hash-functions-and-list-types-of-hash-functions/amp/ [Accessed 16 Nov. 2022].

[6] M. Hammoudeh, I. Ghafir, A.Bounceur and T. Rawlinson, "Continuous Monitoring in Mission-Critical Applications Using the Internet of Things and Blockchain," International Conference on Future Networks and Distributed Systems. Paris, France, 2019.

[7] I. Ghafir and V. Prenosil, "Advanced Persistent Threat and Spear Phishing Emails." International Conference Distance Learning, Simulation and Communication. Brno, Czech Republic, pp. 34-41, 2015.

[8] B. V. Rompay, "Analysis and Design of Cryptographic Hash functions, MAC algorithms and Block Ciphers", Ph.D. thesis, Electrical Engineering Department, KatholiekeUniversiteit, Leuven, Belgium, 2004. [Accessed 16 Nov. 2022].

[9] I. Ghafir and V. Prenosil, "Malicious File Hash Detection and Drive-by Download Attacks," International Conference on Computer and Communication Technologies, series Advances in Intelligent Systems and Computing. Hyderabad: Springer, vol. 379, pp. 661-669, 2016.

[10] Haber, S. and Stornetta, W.Scott. (1991). How to time-stamp a digital

document. *Journal of Cryptology*, [online] 3(2). doi:10.1007/bf00196791.

[11] Matyas, S., Le, A. and Abraham, D.G. (1991). A Key-Management Scheme Based on Control Vectors. undefined. [online] Available at: https://www.semanticscholar.org/paper/A-Key-Management-Scheme-Based-on-Control-Vectors-Matyas-Le/bcf91d272224077802324b3f8e920ee81cd61dd8 [Accessed 16 Nov. 2022].

[12] I. Ghafir, J. Svoboda, V. Prenosil, "A Survey on Botnet Command and Control Traffic Detection," International Journal of Advances in Computer Networks and Its Security (IJCNS), vol. 5(2), pp. 75-80, 2015.

[13] Gauravaram, P. (2007). Cryptographic Hash Functions: Cryptanalysis, Design and Applications Thesis submitted in accordance with the regulations for Degree of Doctor of Philosophy. [online] Available at: https://eprints.qut.edu.au/16372/1/Praveen_Gauravaram_Thesis.pdf [Accessed 22 Nov. 2022].

[14] I. Ghafir and V. Prenosil. "Proposed Approach for Targeted Attacks Detection," Advanced Computer and Communication Engineering Technology, Lecture Notes in Electrical Engineering. Phuket: Springer International Publishing, vol. 362, pp. 73-80, 9, 2016.

[15] Bellare, M. and Kohno, T. (2004). Hash Function Balance and Its Impact on Birthday Attacks. Advances in Cryptology - EUROCRYPT 2004, pp.401–418. doi:10.1007/978-3-540-24676-3_24.

[16] www.sciepub.com. (n.d.). Kaliski, B., and Robshaw, M. The Secure Use of RSA. CryptoBytes, RSA Laboratories, (Autumn 1995), 7-13. [online] Available at: http://www.sciepub.com/reference/282336 [Accessed 23 Nov. 2022].

[17] Tsudik, G. (1992). Message authentication with one-way hash functions. ACM SIGCOMM Computer Communication Review, 22(5), pp.29–38. doi:10.1145/141809.141812.

[18] Kelsey, J.M. and Kohno, T. (2006). Herding Hash Functions and the Nostradamus Attack. NIST, [online] pp.183–200. Available at: https://www.nist.gov/publications/herding-hash-functions-and-nostradamus-attack [Accessed 23 Nov. 2022].

[19] S. Eltanani and I. Ghafir, "Aerial Wireless Networks: Proposed Solution for Coverage Optimisation," IEEE Conference on Computer Communications Workshops", IEEE, 2021.

[20] Bakhtiari, S., Safavi-Naini, R. and Pieprzyk, J. (1997). A message authentication code based on latin squares. Information Security and Privacy, pp.194–203. doi:10.1007/bfb0027926.

[21] Venkatesh, K. and Narasimhan, D. (2022). Revealing the novel precise subset identification and deduplication of audio substance over the shared public environment. The Journal of Supercomputing, 78(9), pp.11856–11872. doi:10.1007/s11227-022-04317-6.