A Brief Analysis of MD5 and Collision-Resistance

Jack Bowman

University of Northwestern

COS2269

Computer Security Fundamentals

Dr. Lawrence Dunn, Instructor

March 31, 2022

**Introduction**

Hash functions are a central part of cryptography and the broader subject of computer security. Creating a hash involves taking an arbitrary-length input and applying an algorithm to achieve a fixed-size output. While conceptually simple, its applications include protecting confidential information, comparing and indexing large amounts of data, and creating digital signatures. To be effective, a hash function must be:

1. Preimage resistant: Given an output, it is computationally infeasible to find an input that hashes to that same output
2. Second preimage resistant: Given an input, it is computationally infeasible to find another input that hashes to the same output
3. Collision-resistant: It is computationally infeasible to find two inputs that hash to the same output

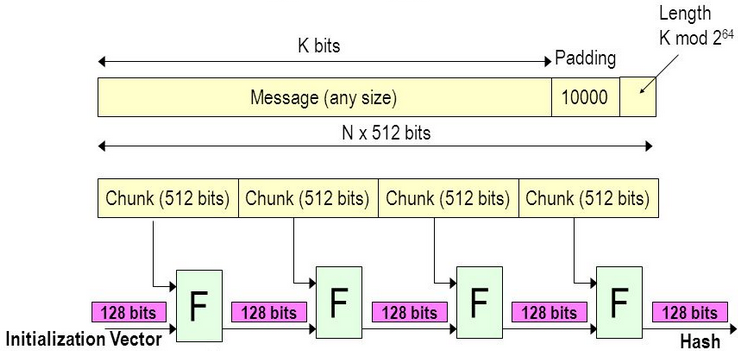
Of the three properties, collision resistance is one of the more challenging to fulfill. Given a fixed-size output, the number of possible hash values is inherently limited. A collision attack attempts to exploit this property by finding two inputs that produce identical outputs (Stallings and Brown, 2007). Within the past 20 years, several widely used hashing algorithms have met their demise to collision attacks as aging algorithms continue to compete with the newest processors. In 2004 Xiaoyun Wang and his associates demonstrated a successful attack against MD5 (Wang and Yu, 2005), and in 2017, CWI Amsterdam and Google Research revealed successful attacks against SHA-1 (Stevens et al., 2017). Despite known vulnerabilities, there have been instances of businesses still using algorithms like MD5 and SHA-1 to hash information either due to ignorance or underestimating the consequences of such actions. The remainder of this paper explores existing collision attacks on MD5, including how they occur, how they are used to exploit individuals, and why MD5 is still occasionally used despite its known vulnerabilities. As mentioned earlier, MD5 is a cryptographically broken hash algorithm. However, analyzing an algorithm like MD5 allows an individual to understand many concepts regarding hash functions.

**MD5 Algorithm**

Performing a collision attack requires a thorough understanding of how an algorithm generates its output. MD5, whose hash size is 128 bits, first takes a message and appends bits to the original plaintext to make it compatible with its algorithm; this concept is called "padding." Total bits must always be 64 bits short of any multiple of 512. The first padding bit is one, followed by the necessary number of zeroes. Then the plaintext length, which is expressed in 64 bits, is appended to the plaintext and padding, making it a multiple of 512. Next, the content is dissected into 512-bit blocks while an MD buffer is initiated. Each 512-bit block is, again, dissected into 16 32-bit sub-blocks and iteratively hashed in the MD buffer using the MD5 algorithm (Kaminsky, 2004). Refer to figure 1 for any clarification.

At its core, the algorithm is a one-way compression function, merging a 512-bit block with its previous iteration, termed "Intermediate Hash Value (IHV)," so that every portion of the content contributes to its final value. Note that the first IHV used is a constant amongst hashes and is called the initialization vector. Understanding a hash algorithm allows for a better understanding of how various attacks may try to take advantage of different hash functions.

Figure 1.



**MD5 Attacks**

Generally speaking, there are two types of collision attacks. The first attack is an identical-prefix attack, and the second is a chosen-prefix attack, which expands on concepts found in identical-prefix attacks. While one can adopt a brute-force method, a technique that can be categorized as an identical-prefix attack, more elegant, meaningful approaches within identical and chosen-prefix attacks exist to create artificial hashes. Most attacks utilize fundamental properties of classic cryptographic hash functions that can provide greater control of the functional contents of the item hashed. For example, given MD5’s hash function, one can establish that if two items with identical hashes are the same, then appending a message with the same contents on both files will result in new **identical**hashes (Kaminsky, 2004). The expression is stated below, where H is the hash function and a, b, and c are messages:

H(a) = H(b) à H(a + c) = H(b + c)

While it is not necessarily a vulnerability by itself, MD5 attacks generally attempt to exploit this property in some way; it is also interesting to note that many other hashing algorithms have this property.

Generating artificial collisions in MD5 involves finding edges of the 512-bit blocks discussed in previous sections and inserting computed collision blocks, which are conditionally related to the previous contents of the file, and they are calculated using a birthday search. These collision blocks, while seemingly random, follow a specific structure that enables an individual to create a manipulated version of a hash (Stevens and Weger, 2012). Again, an expression is stated below representing the attribute where H is the hash function, m1 and m2 are unique messages, and a1 and a2 are unique appendages:

H(m1 + a1) = H(m2 + a2)

If applied properly, the property can be utilized as an intriguing means of manipulating a file without changing a hash value, and the significance of such an action is undoubtedly alarming.

Identical-prefix attacks involve first defining two sections of a file that one wishes to duplicate. The size and content of the two sections are arbitrary, but the prefix must be a multiple of 512 bits; one can append padding if necessary. Next, collision blocks are computed using the contents of the defined prefix and appended to the prefix; this step employs properties discussed in the previous paragraph. While seemingly random, these collision blocks are calculated in such a way that they exploit the MD5 algorithm (Kashyap, 2006). Subsequently, the suffix is appended to complete the file structure. Figure 2 provides a visual representation. Examples of identical-prefix collision attacks on MD5 include FastColl and UniColl. An identical-prefix attack can be accomplished within minutes and effectively destroys the integrity of MD5 hashes.

Figure 2.

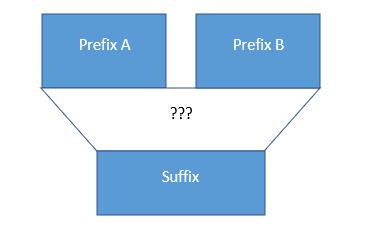
A picture containing box and whisker chart

Description automatically generated

Collision Block(s)

Chosen-prefix attacks follow the same process as Identical-prefix attacks, but they allow an individual to select two different prefixes that output duplicate hash values. Figure 3 provides a visual representation (Kashyap, 2006). Examples of chosen-prefix collision attacks on MD5 include Hash Clash, created by Marc Stevens (Stevens and Weger, 2012) While extremely powerful, it takes a long time to generate collision blocks, given the variation amongst the data.

Figure 3.



Collision Block(s)

**Collision Attack Utilization**

Collision attacks are used to invalidate the integrity of data. Applications of collision attacks include masquerading, deceiving, and falsifying data. Masquerading is the act of impersonating someone else. Using a collision attack, a bad actor could create an untrusted certificate with an identical hash of a trusted certificate, thereby introducing the ability to appear reputable while disseminating malicious content (Du, 2019). Moreover, a bad actor could try to intercept messages between individuals and manipulate the messages while appearing authentic using a collision attack technique. An attacker could also manipulate payment information and go undetected (Menezes, Oorschot, and Vanstone, 2001). To summarize, weak collision-resistance creates many vulnerabilities; Business entities should choose hash functions wisely, as the impacts of an attack could be detrimental.

**MD5 Possible Uses**

Simply put, the ability to artificially create hash collisions largely shatters the purpose of a hash function. Successful collision attacks destroy the integrity of a hash value. However, while broken hash functions like MD5 should not be used for digital signatures, they have some applications, but one should be wary of potential threats. Namely, for use in hash-based message authentication codes (HMAC), local data integrity, and determining partitions for a particular key in partitioned databases. HMACs incorporate a shared key and do not necessarily require a hash function with collision resistance (Kaminsky, 2004). Therefore, MD5 offers a fast, stable alternative when compared to larger hash functions. Algorithms like MD5 can also provide a fast means of checking for local data corruption using a checksum and generating key partitions in a database. Hash functions have more value than simply keeping data confidential, and one can see this when considering a hash function like MD5.

Bibliography

Du, W. (2018). MD5 collision attack lab - seedsecuritylabs.org. Seed Labs. Retrieved April 5, 2022, from https://seedsecuritylabs.org/Labs\_16.04/PDF/Crypto\_MD5\_Collision.pdf

Du, W. (2019). Computer Security: A Hands-On Approach. CreateSpace Independent Publishing Platform.

Kaminsky, D. (2004). MD5 to be considered harmful someday. Aggressive Network Self-Defense. https://doi.org/10.1016/b978-193183620-3/50016-6

Kashyap, Narayana D., "A Meaningful MD5 Hash Collision Attack" (2006).  
Master's Projects. 21.  
DOI: https://doi.org/10.31979/etd.fm5j-tzcm

Menezes, A., Van Oorschot, P., & Vanstone, S. (2001). Handbook of Applied Cryptography. CRC.

Stallings, W., & Brown, L. (2007). Computer Security: Principles and Practice. Pearson.

Stevens, M., Bursztein, E., Karpman , P., Albertini , A., Markov , Y., Bianco , A., & Baisse , C. (2017, February 23). Announcing the First SHA1 Collision. Google Online Security Blog. Retrieved April 4, 2022, from https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html

Stevens, M., Lenstra, A. K., & Weger, B. D. (2012). Chosen-prefix collisions for MD5 and applications. International Journal of Applied Cryptography, 2(4), 322. https://doi.org/10.1504/ijact.2012.048084

Wang, X., & Yu, H. (2005). How to break MD5 and other hash functions. Lecture Notes in Computer Science, 19–35. https://doi.org/10.1007/11426639\_2