Strengths and Weaknesses of Secure Cryptographic Hash Functions

Nikunj Mehta

Cryptography is defined as “the science or study of the techniques of secret writing, esp. code and cipher systems, methods, and the like” (*Dictionary.com*). It builds upon low-level cryptographic algorithms that are called cryptographic primitives. Cryptographic tools, on the other hand, are the culmination of one or more cryptographic primitives, forming constructs such as cryptosystems and cryptographic protocols. I will be focusing on the foundations of many cryptosystems and cryptographic protocols – cryptographic hash functions. Among the commonly employed cryptographic hash functions, there are MD5 (Message-Digest algorithm 5) and SHA-1 (Secure Hash Algorithm-1). Other cryptographic hash functions exist, such as SHA-256, SHA-512 and WHIRLPOOL. This paper will primarily be an analysis of the weaknesses of the MD5/SHA-1 algorithms and include notes on the theoretical security gain from common security practices and from switching to a more secure cryptographic hash function, namely WHIRLPOOL.

Both MD5 and SHA-1 are commonly employed as a means to verify the integrity of a file and are also used widely for the supposedly secure storage of passwords. Over the years, however, each of these cryptographic hash functions has been proven vulnerable to various attacks. Regardless, they are still used in many places. Both are employed in Transport Layer Security (TLS)/Secure Sockets Layer (SSL), the protocols used for the secure communication across a network (namely, the internet), although the protocols use both MD5 and SHA-1 in conjunction to eliminate many of the vulnerabilities in each individual cryptographic hash function (This was replaced by SHA-256 in TLS version 1.2).

MD5, similar to most other cryptographic hash functions, “takes as input a message of arbitrary length and produces as output a 128-bit 'fingerprint' or 'message digest' of the input” (Rivest). Among cryptographic hash functions, each may have differing sizes of input and output, but the concept is the same. SHA-1, also an algorithm based on the Message-Digest series (in this case, MD4), was created for many of the same reasons as MD5 (Jones). Vulnerabilities have been found with both MD5 and SHA-1 leading their loss of stature in past years as secure cryptographic hash functions. The security of MD5 can be compromised so easily nowadays that it should not be employed in any production environment. SHA-1 still retains usefulness in that it is currently not feasible for an average malicious agent to find a collision (and thus potentially compromise the security of the system employing the cryptographic hash function).

Several years ago, due to the computational effort required for finding a collision for a hash produced by either MD5 or SHA-1, it was less than likely that an attacker could perform this in a reasonable amount of time. The rise of freely-available rainbow tables (lookup tables recovering plaintext from hashes in reduced time at the expense of increased memory usage) has allowed for the average attacker to look up hashes and find a matching plaintext, changing the realms of cryptography and security. Rainbow tables are often created by simulating a brute force attack (an attack which guesses all possible combinations) and are commonly used for password-cracking. Since the average user's password contains only alphanumeric characters, that is the character set most commonly used for rainbow table generation.

One method of generating a rainbow table is to simply simulate a linear brute force attack and attempting to exhaust all possible combinations of characters from the character set. This however, is a poor choice because the inordinate amount of space that would be required to store the results. Traditional rainbow tables also do not aim for a success rate of 100% and are usually not generated in a linear fashion. The second, far more prevalent method of generating a rainbow table is to generate (or duplicate from elsewhere) a relatively small rainbow table employing a standard alphanumeric character set and then allowing the public to input hashes for lookup. If the hash is found, the user is shown the plaintext form. If the hash is not found, however, the user receives nothing and the hash is deferred to a queue of hashes that will eventually be processed to find. This method is superior for most practical purposes due to the nature of average computer users and their tendency to choose memorable and similar, if not simple, passwords.

A common approach for rendering the use of rainbow tables more difficult is the use of a salt (random bits added to the original message prior to the use of a computational hash function). The most useful purpose of the salt is to deter dictionary attacks and the like (e.g. rainbow table lookups) against the hash produced by a cryptographic hash function. Traditional rainbow tables could not be employed because they would not account for the salt being part of the originating string and the chances of finding a collision become astronomically small. A specialized rainbow table would have to be created using the salt (if given) but this is almost never done due to the notion that the creation of a rainbow table is tantamount to a brute-force/exhaustive search attack. It is usually not worth the time and memory to undertake such a task.

The simple use of a salt is so effective against these types of precomputed attacks that some protocols (including SSL) even send the salt in plaintext with the hash so the receiver can compare the hash of the salt and original input to the hash specified. This does, however, require that both the sender and receiver are using the same algorithm to combine the salt and the input before hashing, or the sender must inform the receiver the method by which to do it. Regardless, the effects of using a salt are crippling to certain forms of attack.

There also exist cryptanalysis techniques particularly effective against MD5 or SHA-1 that have been published and describe in-depth the theoretical and even practical methods of generating collisions. MD5 has been effectively dissected to the point where it is no longer a reliable security component. SHA-1 is said by some to be in the same shoes as MD5 in as little as a few years (“Gröbner Base Based Cryptanalysis of SHA-1”) (“Chinese Professor Cracks Fifth Data Security Algorithm”).

The successors of these commonly-used cryptographic hash functions include the SHA-256, SHA-512 and WHIRLPOOL cryptographic hash functions. Each of these algorithms has security levels that are orders of magnitude above MD5 and SHA-1 due to their increased message digest sizes and further-refined algorithms. To compare, the best public cryptanalysis for each algorithm can be compared. For MD5, collision resistance has been broken in $2^{20.96}$ time, which is no more than a few seconds on an average computer (Xie, and Feng). For SHA-1, the hash function was broken due to hash collisions being producible with a complexity of 2^{51} operations (Manuel). For SHA-512 (SHA-2). As for SHA-512, the best public attack breaks preimage resistance for 46 of the 80 rounds (Sasaki, Wang, and Aoki). And lastly, for WHIRLPOOL, a rebound attack was disclosed presenting full collisions against 4.5 rounds in 2^{120} operations, semi-free-start collisions

against 5.5 rounds in 2^{120} time and semi-free-start near-collisions against 7.5 rounds in 2^{128} time (Mendel, Rechberger, Schläffer, and Thomsen).

In conclusion, cryptography is a field in which there are no absolute certainties, only standards for security. In our current day and age, the cryptographic hash functions of MD5 and SHA-1 are slowly becoming phased out in favor of more secure cryptographic hash functions such as SHA-256, SHA-512 and WHIRLPOOL. In several decades, the world of cryptography will have to produce new algorithms and methods whereby security can be preserved. Any user currently employing these cryptographic hash functions should take notice and prepare for the future.

Works Cited

"cryptography." *Dictionary.com Unabridged*. Random House, Inc. 18 Nov. 2010. <Dictionary.com http://dictionary.reference.com/browse/cryptography>.

"Chinese Professor Cracks Fifth Data Security Algorithm." *The Epoch Times* 11 Jan 2007: n. pag. Web. 17 Nov 2010. <http://en.epochtimes.com/news/7-1-11/50336.html>.

"Gröbner Base Based Cryptanalysis of SHA-1." *National Institute of Standards and Technology*. Web. 18 Nov 2010. <http://csrc.nist.gov/groups/ST/hash/documents/SUGITA_NISTHash2Sugita.pdf>.

Jones, P. "RFC 3174 - US Secure Hash Algorithm-1." *Internet FAQ Archives*. Cisco Systems, Sep 2001. Web. 16 Nov 2010. <http://www.faqs.org/rfcs/rfc3174.html>.

Manuel, St´ephane. "Classification and Generation of Disturbance Vectors for Collision Attacks against SHA-1." *International Association for Cryptologic Research*. Paris Rocquencourt, n.d. Web. 15 Nov 2010. <http://eprint.iacr.org/2008/469.pdf>.

Mendel, Florian, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. "The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl." *Departemente Elektrotechniek - ESAT K.U.Leuven*. Technical University of Denmark; Graz University of Technology, 24 Feb 2009. Web. 18 Nov 2010. <https://www.cosic.esat.kuleuven.be/fse2009/slides/2402_1150_Schlaeffer.pdf>.

Rivest, Ronald. "RFC 1321 - The MD5 Message-Digest Algorithm." *The Internet Engineering Task Force (IETF)*. MIT Laboratory for Computer Science and RSA Data Security, Inc., Apr 1992. Web. 16 Nov 2010. <http://tools.ietf.org/html/rfc1321>.

Sasaki, Yu, Lei Wang, and Kazumaro Aoki. "Preimage Attacks on 41-Step SHA-256 and 46-Step SHA-512." *International Association for Cryptologic Research*. NTT Information Sharing Platform Laboratories, NTT Corporation; The University of Electro-Communications, 25 Nov 2008. Web. 17 Nov 2010. <http://eprint.iacr.org/2009/479.pdf>.

Xie, Tao, and Dengguo Feng. "How To Find Weak Input Di®erences For MD5 Collision Attacks." *International Association for Cryptologic Research*. State Key Lab of Information Security, Chinese Academy of Sciences; The Center for Soft-Computing and Cryptology, NUDT, 30 May 2009. Web. 18 Nov 2010. <http://eprint.iacr.org/2009/223.pdf>.