5 Hashing Algorithms

Daniel E. Hernández - 20180077

Facultad de Ciencias Económicas, Universidad Francisco Marroquín, Guatemala

Abril 30, 2019

Abstract

1 MD5, message-digest algorithm

Although originally it was designed as a cryptographic hash function, it has suffered extensive vulnerabilities. In the modern day, however, it is used as a checksum to verify data integrity.

1.1 Applications

As stated previously, it is used to check if the actual downloaded file is the same as the one the user wanted to download. More often than not file servers provide a md5sum for the user to compare. Most UNIX-based operating systems include MD5 sum utilities.

One famous file server that provides md5sums is Android File Host, a free to use Android Developers' file server used mostly to host Android ROMs and gapps.

1.2 Advantages

1. Performance: Md5 can hash around 400MB/s on a single core, while SHA-1 can only do 300MB/s and SHA-256 will bottleneck at 150MB/s[4].

2. It is well known and widely used.

3. It comes preinstalled on most UNIX-based systems.

1.3 Disadvantages

1. Vulnerable to malicious hash collisions.
2. Because of malicious hash collisions, attackers can fake SSL certificates [6].

2 SHA-1, Secure Hashing Algorithm 1

It takes an input and generates a 160=bit hash value, a message digest. Normally the message digest is rendered as a hexadecimal number 40 digits long. It was designed by the U.S. National Security Agency.

2.1 Applications

SHA-1 is widely used in security applications and protocols including TLS and SSL, SSH, PGP, and others. SHA-1 is also used vastly on U.S. government applications. Normally it is used alongside other protocols and algorithms to ensure the safety of unclassified and sensitive documents and information.

SHA-1 is also used to check data integrity. Version control systems such as Git, Mercurial and Monotone use the algorithm as a data integrity checker.

2.2 Advantages

1. Even though it's not as fast as MD5, it's still performs rather fast at 300MB/s [4].
2. Because it was created by the NSA, many users have considered it as a secure algorithm resulting on a widely used hashing function.
3. It too comes preinstalled on most UNIX-based systems.

2.3 Disadvantages

1. Vulnerable to malicious hash collisions as Google proved on the year 2017 [9].
2. It is still used on GIT and as the proof cited above, a malicious attacker could *in theory* perform an attack on having two different files, one backdoored and the original but with the same hash thus infecting the user who clones that repository.

3 SHA-2, Secure Hashing Algorithm 2

It is a security cryptographic algorithm. It was created by the U.S. National Security Agency in collaboration with the National Institute of Science and

Technology (NIST) as an upgrade from SHA-1. SHA-2 is a family of six different hash functions with different bit values:

1. SHA2-224
2. SHA2-256
3. SHA2-284
4. SHA2-512
5. SHA2-512/224
6. SHA2-512/256

3.1 Applications

SHA-2 is widely used on security applications and protocols like TLS and SSL, PGP, SSH, S/MIME and IPsec. Debian, the Linux distribution, uses SHA-2 as part of the process to authenticate the software packages. It is also used on password hashing [10] and used on many cryptocurrencies like Bitcoin [8].

3.2 Advantages

1. Extremely difficult to find a hash collision in the wild as the possible combinations are 2^{256} for SHA-256 and so forth with the rest of the family functions.
2. SHA-2 has no vulnerabilities yet.
3. Quantum collision-resistant [11].
4. It too comes preinstalled on most UNIX-based systems.

3.3 Disadvantages

1. Even though the *real* SHA-256 doesn't suffer from any vulnerabilities yet, one could start a collision attack on the first 31 rounds of SHA-256 [7].

4 SHA-3, Secure Hashing Algorithm 3

SHA-3 is the result of an open call of NIST to the cryptographic community for hash function proposals. There was no restriction on who could participate, so submissions were open in the broadest possible sense. Every submitted candidate algorithm had to contain a description, a design rationale and preliminary cryptanalysis. The authors of the 64 submissions included the majority of people active in open symmetric crypto research at the time. NIST solicited the symmetric crypto community for performing and publishing research in cryptanalysis, implementations, proofs and comparisons of the candidates and based its decision on the results. After a three-round process involving hundreds of people in the community for several years, NIST finally announced that Keccak was selected to become the SHA-3 standard**biKeccakTeambid**.

1. SHA3-224
2. SHA-256
3. SHA3-384
4. SHA3-512
5. SHAKE128
6. SHAKE256

4.1 Applications

Currently there are no real usages of SHA-3 because of the performance issues that it faces and the –yet– nonexistent need to migrate to it.

4.2 Advantages

1. Open Source [5].
2. In hardware it's faster than SHA-1 and SHA-2 [3].

4.3 Disadvantages

1. In software, SHA-1 is three times faster and SHA-512 is two times faster than SHA-3 [3]

5 BLAKE2

BLAKE2 is a cryptographic hash function faster than MD5, SHA-1, SHA-2, and SHA-3, yet is at least as secure as the latest standard SHA-3. BLAKE2 has been adopted by many projects due to its high speed, security, and simplicity [2]. BLAKE2 comes in two flavors:

1. BLAKE2b (for 64-bit platforms and produces a digest between 1 and 64 bytes)
2. BLAKE2s (for 8- to 32-bit platforms and produces a digest between 1 and 32 bytes)

5.1 Applications

Because of BLAKE2's performance, it is starting to be used the same way MD5 was used on things like data integrity while being much faster.

Other usages of BLAKE2 include: GNU Core Utilities, Argon2, Noise (the cryptographic protocol used in WhatsApp and WireGuard), RAR version 5 supports an optional 256-bit BLAKE2sp file checksum, libsync, the cryptocurrencies Zcash and Decred, FreeBSD Ports package managment uses BLAKE2b and IPFS.

5.2 Advantages

1. Open Source [5].

2. Faster than MD5, SHA-1, SHA-2, SHA-512, SHA3-256 and SHA3-512[1].

5.3 Disadvantages

1. None.

Referencias

[1] J.-P. Aumasson, S. Neves, Z. Wilcox-O'Hearn y C. Winnerlein, "BLAKE2: Simpler, Smaller, Fast as MD5," en, en *Applied Cryptography and Network Security*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, F. Bao, P. Samarati y J. Zhou, eds., vol. 7341, http://link.springer.com/10.1007/978-3-642-38980-1_8, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, págs. 119-135, ISBN: 978-3-642-31283-0 978-3-642-31284-7. DOI: 10.1007/978-3-642-38980-1_8.

[2] *BLAKE2*, https://blake2.net/.

[3] R. A. Grimes, *Why Aren't We Using SHA3?* en, https://www.csoonline.com/article/3256088/why-arent-we-using-sha3.html, feb. de 2018.

[4] *Hash - Is MD5 Considered Insecure?* https://security.stackexchange.com/questions/19906/is-md5-considered-insecure.

[5] *Keccak Team*, https://keccak.team/.

[6] *MD5 Weakness Allows Fake SSL Certificates To Be Created*, https://www.sslshopper.com/article-md5-weakness-allows-fake-ssl-certificates-to-be-created.html.

[7] F. Mendel, T. Nad y M. Schläffer, "Improving Local Collisions: New Attacks on Reduced SHA-256," en, en *Advances in Cryptology – EUROCRYPT 2013*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, T. Johansson y P. Q. Nguyen, eds., vol. 7881, http://link.springer.com/10.1007/978-3-642-38348-9_16, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, págs. 262-278, ISBN: 978-3-642-38347-2 978-3-642-38348-9. DOI: 10.1007/978-3-642-38348-9_16.

[8] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," en, pág. 9,

[9] M. Stevens, E. Bursztein, P. Karpman, A. Albertini e Y. Markov, "The First Collision for Full SHA-1," en, en *Advances in Cryptology – CRYPTO 2017*, J. Katz y H. Shacham, eds., vol. 10401, http://link.springer.com/10.1007/978-3-319-63688-7_19, Cham: Springer International Publishing, 2017, págs. 570-596, ISBN: 978-3-319-63687-0 978-3-319-63688-7. DOI: 10.1007/978-3-319-63688-7_19.

[10] *Unix Crypt with SHA-256/512*, https://akkadia.org/drepper/sha-crypt.html.

[11] D. Unruh, “Collapse-Binding Quantum Commitments Without Random Oracles,” en, en *Advances in Cryptology – ASIACRYPT 2016*, J. H. Cheon y T. Takagi, eds., vol. 10032, http://link.springer.com/10.1007/978-3-662-53890-6_6, Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, págs. 166-195, ISBN: 978-3-662-53889-0 978-3-662-53890-6. DOI: 10.1007/978-3-662-53890-6_6.