Extracts from the SHA-3 Competition

Vincent Rijmen*

KU Leuven, Dept. ESAT/SCD-COSIC and IBBT, Belgium

1 The Start of the Competition

The story of the SHA-3 competition starts with the presentation of surprisingly efficient attacks on several modern hash functions at Eurocrypt 2005 [1, 2] and at Crypto 2005 [3, 4]. Collisions were given for the hash functions MD4, MD5, RIPEMD and SHA-0. An algorithm was shown that can produce collisions for SHA-1 with a complexity that is much lower than previously thought. Before 2005, there were already partial attacks known for several of these hash functions, but only MD4 was really broken [5]. Soon the results were furthere improved and extended to other hash functions. These developments caused NIST to start an effort to develop and standardize a new Secure Hashing Algorithm. This effort was going to be an open competition, similar to the AES competition which it had run from 1998 until 2000.

In 1998, NIST received 21 submissions for the AES competition, of which 15 fulfilled the formal submission requirements and were allowed to enter the competition. In 2008, NIST received 64 submissions for the SHA-3 competition, of which 51 were allowed to enter. This big increase in candidate algorithms and the limited amount of effort available implied that NIST had to adopt a strategy that would allow it to quickly reduce the number of candidates. In July 2009, barely 6 months after the first SHA-3 Candidate Conference, NIST announced the 14 algorithms that were selected to enter the second round of the competition. Quite some of the other 37 candidates had been broken or turned out to be very slow compared to the current standards. However, it can also be said that some of the non-selected algorithms were victims of bad PR. In December 2010, NIST announced the 5 finalist algorithms, which were allowed to enter the third and last round of the competition.

2 The 5 Finalists

We briefly discuss the main elements of each of the 5 finalists (in alphabetical order). The reader will notice that NIST selected the finalists in such a way that all corners of the design space are covered. This is probably not a coincidence.

* This work was sponsored by the Research Fund KU Leuven (OT/08/027) and by the European Commission through the ICT Programme under Contract ICT-2007-216676 (ECRYPT II). The information in this paper is provided as is, and no warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

L.R. Knudsen and H. Wu (Eds.): SAC 2012, LNCS 7707, pp. 81–85, 2013.

Blake [6] is an *Addition-Rotation-XOR (ARX)* design. It has a round function inspired by the stream cipher Salsa20 [7]. Its internal state is represented by a square, which in each round is first processed column by column, like in the AES [8], and subsequently diagonal by diagonal. Blake has 14 or 16 rounds, depending on the desired length of the digest, which determines the size of the internal state. Despite its large number of rounds and the fact that each round contains two passes over the whole state, Blake is very fast on almost all platforms.

Grøstl [9] is a *Substitution-Permutation Network (SPN)* design. Among the finalists, Grøstl comes the closest to the AES and Rijndael designs. It uses a state consisting of two 8×8 squares for 256-bit digests, respectively of two 8×16 rectangular arrays for 512-bit digests. The round transformation uses the AES S-box and diffusion operations which are very similar to the AES diffusion operations. There are 10 or 14 rounds, depending on the digest length. Grøstl is classified as a *permutation based* design, which is distinguished from a *block-cipher based* design by the fact that the message is not inputted in every round, but only at the start and the end of the compression function.

Also JH [10] is a permutation-based SPN design. JH has 42 rounds. It uses 4-bit S-boxes and generalizes the two-dimensional AES diffusion to 8 dimensions, however without generalizing the proof on the minimum number of active S-boxes. Digests of length 256 or 512 bits are produced by exactly the same function.

Keccak [11] is a *Sponge* design, which implies that it is also permutation-based. The Keccak permutation has 24 rounds. It uses a 5-bit S-box and bit-level diffusion with a nice geometric representation. Keccak is a streaming design: every message bit is added to the state only once. The state is a 3-dimensional array. Specifically, it is a $5 \times 5 \times z$ array, where z is equal to 64 for the version submitted to the competition. The versions producing digests of 256 respectively 512 bits differ only in the number of message bits that are added to the state between two applications of the permutation. On most platforms, the speed of Keccak is in the middle of the 5 finalists.

Skein [12] is the second ARX design. It has 72 rounds and is very fast on high-end platforms. Skein is block-cipher based. It uses the block cipher Threefish in the newly designed *Unique Block Iteration (UBI)* mode. Threefish. was designed specifically for Skein and bears no resemblance to Twofish or Blowfish. Digests of length 256 or 512 bits are produced by exactly the same function.

3 Speed

The eBASH site offers speed measurements of the finalists on a multitude of desktop and server platforms, but also a few smaller processors [13]. It indicates that Skein and (the 256-bit version of) Blake are usually the fastest, Keccak is mostly in the middle. At the bottom side, Grøstl-512 is usually the slowest, preceded by JH and Grøstl-256. On platforms with the special AES instructions, Grøstl-256 beats Keccak. Hence, ARX designs appear to make the best use of these platforms.

The XBX study looks at embedded processors and takes the requirements for ROM and RAM into account [14]. In this study, Blake usually performs well, and it never performs badly. On many high-end platforms, Skein is the fastest. Keccak and Grøstl perform repeatedly among the best, in particular when small footprint is important.

4 Highlights

If the talks and papers presented during the AES competition would have to be summarized in a few words, we would propose security-margin weighted performance, side-channel attack resistance, algebra and the pronounciation of certain Dutch vowels. For the SHA-3 competition, these would be replaced by provable security/indifferentiability, rebound, practicality of attacks and distinguishers/nonrandom properties.

The indifferentiability concept [15] predates the SHA-3 conference. It has now become common practice to produce an indifferentiability proof for every new design [16]. Like with all security proofs, a certain level of idealisation is required in order for the proofs to work. At which level the idealization is done, influences for instance the provable security of Blake [17]. Clearly, every methodology which forces designers to reason about the security properties of their design, is a good thing to have. The exact implications of an indifferentiability proof for the security of a design remain a topic of further research [18].

A distinguisher for a keyed primitive is a well-defined concept. For an unkeyed primitive, say a standardized hash function, defining a distinguisher turns out to be problematic. There appears to be a vague intuition in folk lore, that any property of a specific hash function, that is present with small probability only in a *random* function, should be considered as a weakness, since it allows to *distinguish* the specific hash function from a random function. The problem is that distinguishing a deterministic hash function h from a random function f, is trivial. For example, we can simply query the function on 0; the probability that $f(0) = h(0)$ is negligible. More broadly speaking, some of the recently presented results can be considered as *Texas sharpshooter fallacies*, because they don't take into account the effect that is also known as the *law of truly large numbers*: since there is an infinite number of properties that can be investigated, any hash function is bound to exhibit some of them.

5 Outlook

Given the relative slow progress in cryptanalytic results on SHA-256 and the quite competitive speed of SHA-256 and SHA-512 on many processors, it is likely that the winner of the SHA-3 competition will face a hard time pushing its way into applications. For comparison, consider how long it took AES to replace the much slower 3-DES or even the clearly insecure DES in many applications.

On the other hand, it is clear that the competition has increased the pace of progress in knowledge on hash function design and cryptanalysis. Some of

the new cryptanalytic insights even led to improved results on AES [19, 20]. Furthermore, several important steps have been made in the development of tools and libraries assisting the (semi-)automatic cryptanalysis of hash functions and symmetric-key algorithms [21]. It might well be that these will be the most important outcome of the SHA-3 competition.

References

1. Wang, X., Yu, H.: How to break MD5 and other hash functions. In: [22], pp. 19–35
2. Wang, X., Lai, X., Feng, D., Chen, H., Yu, X.: Cryptanalysis of the hash functions MD4 and RIPEMD. In: [22], pp. 1–18
3. Wang, X., Yin, Y.L., Yu, H.: Finding collisions in the full SHA-1. In: [23], pp. 17–36
4. Wang, X., Yu, H., Yin, Y.L.: Efficient collision search attacks on SHA-0. In: [23], pp. 1–16
5. Dobbertin, H.: Cryptanalysis of MD4. J. Cryptology 11(4), 253–271 (1998)
6. Aumasson, J.P., Henzen, L., Meier, W., Phan, R.C.W.: SHA-3 proposal BLAKE, version 1.3 (December 16, 2010) `http://131002.net/blake/blake.pdf`
7. Bernstein, D.J.: The Salsa20 Family of Stream Ciphers. In: Robshaw, M., Billet, O. (eds.) New Stream Cipher Designs. LNCS, vol. 4986, pp. 84–97. Springer, Heidelberg (2008)
8. National Institute of Standards and Technology (NIST): FIPS-197: Advanced Encryption Standard (2001), `http://www.itl.nist.gov/fipspubs`
9. Gauravaram, P., Knudsen, L.R., Matusiewicz, K., Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: Grøstl, a SHA-3 candidate (March 2, 2011) `http://www.groestl.info/Groestl.pdf`
10. Wu, H.: The hash function JH (January 16, 2011) `http://www3.ntu.edu.sg/home/wuhj/research/jh/jh_round3.pdf`
11. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: The Keccak reference, version 3.0 (January 14, 2011), `http://keccak.noekeon.org/Keccak-reference-3.0.pdf`
12. Ferguson, N., Lucks, S., Schneier, B., Whiting, D., Bellare, M., Kohno, T., Callas, J., Walker, J.: The Skein hash function family, version 1.3 (October 1, 2010) `http://www.skein-hash.info/sites/default/files/skein1.3.pdf`
13. Bernstein, D.J., Lange, T. (eds.): eBACS: ECRYPT benchmarking of cryptographic systems, `http://bench.cr.yp.to/ebash.html`
14. Wenzel-Benner, C., Gräf, J., Pham, J., Kaps, J.P.: XBX benchmarking results (May 2012), `https://xbx.das-labor.org/trac/export/82/page/trunk/documentation/benchmarking_results_may_2012.pdf`
15. Maurer, U.M., Renner, R.S., Holenstein, C.: Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 21–39. Springer, Heidelberg (2004)
16. Andreeva, E., Mennink, B., Preneel, B.: Security Reductions of the Second Round SHA-3 Candidates. In: Burmester, M., Tsudik, G., Magliveras, S., Ilić, I. (eds.) ISC 2010. LNCS, vol. 6531, pp. 39–53. Springer, Heidelberg (2011)
17. Andreeva, E., Luykx, A., Mennink, B.: Provable security of BLAKE with non-ideal compression function. IACR Cryptology ePrint Archive 2011 (2011) 620
18. Ristenpart, T., Shacham, H., Shrimpton, T.: Careful with Composition: Limitations of the Indifferentiability Framework. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 487–506. Springer, Heidelberg (2011)

19. Biryukov, A., Khovratovich, D.: Related-Key Cryptanalysis of the Full AES-192 and AES-256. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 1–18. Springer, Heidelberg (2009)
20. Bogdanov, A., Khovratovich, D., Rechberger, C.: Biclique Cryptanalysis of the Full AES. In: Lee, D.H. (ed.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 344–371. Springer, Heidelberg (2011)
21. ECRYPT II Symlab: Tools for cryptography, `http://www.ecrypt.eu.org/tools/`
22. Cramer, R. (ed.): EUROCRYPT 2005. LNCS, vol. 3494. Springer, Heidelberg (2005)
23. Shoup, V. (ed.): CRYPTO 2005. LNCS, vol. 3621. Springer, Heidelberg (2005)