Crypto Corner

Editors: Peter Gutmann, pgut001@cs.auckland.ac.nz
David Naccache, david.naccache@ens.fr
Charles C. Palmer, ccpalmer@us.ibm.com

A New Hash Competition

WILLIAM E. BURR
US National Institute of Standards and Technology

IEEE Security & Privacy readers might remember that a rush of cryptanalytic discoveries in 2004 briefly threw the arcane world of cryptographic hash functions into foment. In 2005, Xiaoyun Wang and her colleagues[1] announced a "better than brute force" method for finding collisions in the US National Institute of Standards and Technology's (NIST's) venerable SHA-1 hash function, which, at that point, was more or less the last widely used hash function left standing. Any thought that such attacks were harmless (because they generated collisions only for messages that made no sense) was put to rest in 2007 by Marc Stevens and his colleagues, who combined Wang's differential collision attack, Antoine Joux's multicollisions,[2] and John Kelsey's herding process,[3] and then used a consumer video game system to generate several different but meaningful PDF messages, all with the same MD5 hash value (www.win.tue.nl/hashclash/Nostradamus/).

After almost a decade of comparative analytic quiet, these developments have cast doubt on the security of digital signatures and other hash function applications because they depend on collision resistance, the basic security property that it should be computationally infeasible to find two messages with the same hash value. In November 2007, NIST announced a SHA-3 competition, modeled on the successful Advanced Encryption Standard (AES) competition, to select a new federal hash function standard (for full details, see www.nist.gov/hash-competition).

By now, you might be wondering, "So what happened to SHA-2?" Like SHA-1, the SHA-2 family of hash functions (which have 224-, 256-, 384-, and 512-bit outputs) were created at NIST's request by the US National Security Agency (NSA). SHA-2 hash functions have replaced SHA-1 for many applications, and no significant new attacks on them have been reported. Why, then, do we need a new hash function? This article describes why NIST is running the SHA-3 competition, how it's structured, and what it's supposed to accomplish.

The NIST competition

Not all cryptographers agree that the time is ripe for a hash function competition. Some believe we should take more time to study hash function theory—we'll get a better final standard this way, they argue, and we have the SHA-2s to tide us over until then. But most of the people who NIST would expect to contribute designs and analysis to the SHA-3 competition favor a contest as the best vehicle for hash function research, just as the Advanced Encryption Standard (AES) competition greatly advanced the state of block cipher design.

All told, it takes roughly a decade to run this type of competition and then incorporate the winning algorithm into the actual cryptographic fabric of systems and the Internet. Although the SHA-2s are just coming into use, they're very much in the sequence of MD-5 (badly broken), SHA-0 (also broken), and SHA-1 (tottering on the precipice—no collisions demonstrated yet, but they're expected soon). Few doubt the NSA's cryptographic design skill, but intelligence agencies don't explain their detailed design rationales, even though the SHA-2s are publicly specified, which complicates the open security analysis that the global cryptographic community expects for such a widely used standard. It's simply better to have an open competition now, with the full resources of the global cryptographic community behind it while we're ahead of the game rather than to risk an emergency and have to catch up later.

NIST announced the competition on 2 November 2007 with a submission deadline of 31 October 2008. Submitters must provide a specification of their candidate algorithms, with reference and optimized C language implementations, a security analysis, and an intellectual property release. They'll also have to disclose and agree to make available a worldwide royalty free license for any intellectual property they hold on their submission; NIST will endeavor to select a SHA-3 that's free from any patent encumbranc-

es. No issue in cryptographic standards is as volatile as patents, and we don't seem to need a patented hash algorithm.

The winning submission

After much discussion, NIST decided that the SHA-3 family must be a simple substitute for the SHA-2 family—that is, the winning submission must support hash outputs of 224, 256, 384, and 512 bits and be useful in all the main SHA-2 applications, particularly digital signatures, hashed message authentication codes, and pseudorandom number generation. This also includes preserving the online nature of the present algorithms, all of which process comparatively small message blocks (usually 512 or 1,024 bits) at a time instead of buffering the entire message before processing it. Although several cryptographers urged NIST to also standardize an "in-memory" hash function, implementers and users strongly oppose it because many communications applications couldn't tolerate the buffering delays inherent in such a function.

Before the AES competition, many of us envisioned block ciphers in terms of the Feistel network used by the Data Encryption Standard (DES). But Rijndael, the AES competition's ultimate winner, used a rather different "squares" row and column organization (www.iaik.tugraz.at/RESEARCH/krypto/AES/old/~rijmen/rijndael/rijndaeldocV2.zip). The classic hash function structure, used by MD-5, SHA-1, and SHA-2 and illustrated in Figure 1, is the Merkle-Damgård (MD) construct, in which a message is divided into blocks, the message is padded as required, and the length is added to the end of the message. The chaining variable is set to an initial value, and a compression function processes each block to produce the next chaining variable, the final output being the hash value. A collision in an MD hash's overall output implies a collision on the compression function: if it's hard to find compression function collisions, it must also be hard to find hash function collisions. On the other hand, the MD construct itself has several potentially exploitable characteristics that an ideal hash function wouldn't have, even if the MD compression function used a perfect random oracle. NIST expects to get some very different overall designs, some of which might input rather small amounts of data at a time, whereas others might use a tree structure or variations on the MD theme that avoid the scheme's known problems. As with AES, NIST might pick an overall design that departs from traditional practice.

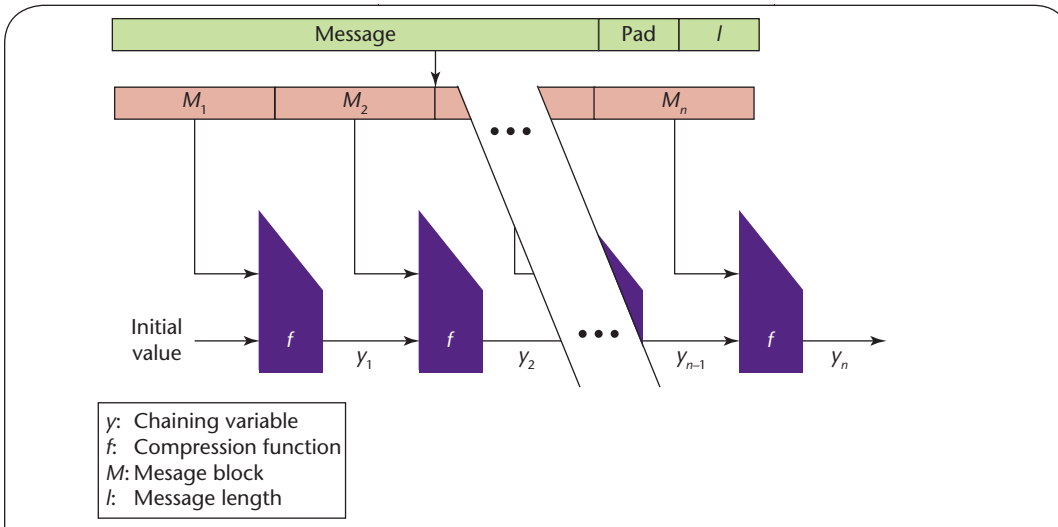

Figure 1. The Merkle-Damgård hash construct. Most current hash functions use this basic structure.

Proofs

Security proofs are often an issue in cryptographic standards, and much passion is aroused by discussions of their value. Some very experienced designers largely discount proofs, whereas others go to considerable effort to structure algorithms and methods to build proofs of various sorts. Typically, such proofs show that breaking the larger structure reduces to

- breaking some smaller element (such as the compression function) that usually can't be proved secure but can be intensely studied, or
- solving some long-studied hard problem in mathematics (such as the discrete log problem).

It might be hard to convince ourselves that a "number theoretic" hash function has all the other properties we expect and desire, including good performance, and that we can plug it into existing hash function applications. Nevertheless, NIST hasn't ruled out such candidates, so it'll be interesting to see what we get. We also expect conventional algorithms that rely on the MD reduction and probably some without any formal reduction proof at all.

In the end, the algorithm's security is the most important selection factor. Although NIST doesn't require them, security proofs (or the lack thereof) will likely be an important element in security discussions, as will the cryptanalysis of deliberately weakened variants. Another important

security consideration is diversity: we already have SHA-2, so we'll want a SHA-3 that either appears much stronger than SHA-2 or different enough that, if SHA-2 breaks, it won't affect SHA-3.

Performance is another selection factor. In the AES competition, we found little clear distinction in security strength between the final five candidates, so we ended up depending heavily on performance, which is always easier to measure than security strength. Inevitably, arguments will rage about the most important platform for performance: big desktops and servers, small microcontrollers, or dedicated hardware logic. This debate is already under way on the NIST hash forum (http://csrc.nist.gov/groups/ST/hash/email_list.html), but it seems clear that the SHA-3 winner will have to be at least competitive with SHA-2 and probably much better than SHA-2 in some applications (perhaps in hardware or by permitting parallel implementation).

NIST expects to launch a Hash Competition Conference to review the initial submissions in February 2009; the second conference will occur roughly a year later in 2010 to review public comments submitted on the submissions and their analysis. Following this second conference, NIST will select a small number of finalist candidates (probably five or so) for intensive review by the community. If, as we expect, we get 20 or more initial submissions, we'll inevitably hear some disagreements about the finalists, but we can only intensively analyze a small number of algorithms, and, as in the AES competition, all the finalists will be good hash functions, although we might have to drop some worthy submissions.

Cryptanalysis of the finalists will be the tricky part—the time that skilled cryptanalysts can donate is the limiting resource here. NIST is building up its limited cryptanalytic resources, but will rely heavily on the global cryptographic research community to do the bulk of the cryptanalysis. If the AES competition is any model, many analysis papers on the candidates will be submitted to various conferences. We'll tentatively review the cryptanalysis results and review performance in a third workshop scheduled for 2012, after which we'll select a winner.

The winning team might get nothing but glory for their huge effort. NIST expects the best people in the world to participate, as they did in the AES competition, because the community believes an open competition is the best way to select cryptographic standards. We expect to work hard, have fun, and significantly advance the state of the art while giving the world a valuable, secure hash function standard. □

Acknowledgments

Contribution of the US National Institute of Standards and Technology; not subject to copyright in the US.

References

1. S. Wang, Y. Yin, and H. Yu, "Finding Collisions in the Full SHA-1," *Proc. Crypto 2005*, Springer, 2005, pp. 17–36.
2. A. Joux, "Multicollisions in Iterated Hash Functions: Application to Cascaded Constructions," *Proc. Crypto 2004*, Springer, 2004, pp. 306–316.
3. J. Kelsey and T. Kohno, "Herding Hash Functions and the Nostradamus Attack," *Proc. Eurocrypt 2006*, Springer, 2006, pp. 183–199.

***William E. Burr** manages the Security Technology Group at the US National Institute of Standards and Technology. His research interests include personal authentication, secure protocols, and (of necessity) hash functions. Burr has a BEE from the Ohio State University. Contact him at william.burr@nist.gov.*