Practical Preimages for Maraca
Extended Abstract

Sebastiaan Indesteege[1,2,⋆] and Bart Preneel[1,2]

[1] Department of Electrical Engineering ESAT/COSIC,
Katholieke Universiteit Leuven. Kasteelpark Arenberg 10, B-3001 Heverlee, Belgium.
sebastiaan.indesteege@esat.kuleuven.be
[2] Interdisciplinary Institute for BroadBand Technology (IBBT), Belgium.

Abstract. We show a practical preimage attack on the cryptographic hash function Maraca, which was submitted as a candidate to the NIST SHA-3 competition. Our attack has been verified experimentially.

1 Introduction

Cryptographic hash functions are easy to compute, deterministic functions that map an input message of arbitrary length to a short, fixed-length digest. They are important building blocks in many cryptographic applications. Secure cryptographic hash functions are required to have several security properties, such as collision resistance and preimage resistance. Informally, preimage resistance means that, given a hash function output, it should be hard to find an input message hashing to this output.

Maraca [1] is a hash function proposal that was recently submitted as a candidate to the NIST SHA-3 competition, which aims to select a new cryptographic hash function standard. Canteaut and Naya-Plasencia [2] found a theoretical collision attack on Maraca, requiring 2^{237} calls to the compression function and a memory of $2^{230.5}$ bits. We propose a practical preimage attack on Maraca. After a one-time precomputation, our attack can find many preimages for any hash output in just a few seconds on an average PC.

2 Analysis of the Maraca S-box

The main weakness we exploit lies within the 8×8 bit bijective S-box used in Maraca. As was already noted in [2], three of the output bits are linear functions of the inputs. A more careful analysis of the properties of the Maraca S-box led to the following observation. The Maraca S-box can be made into an affine function by restricting the input to one of several affine spaces of dimension up to five. Thus, one needs to impose just three affine conditions on the input bits of the S-box to turn it into an affine function.

[⋆] F.W.O. Research Assistant, Fund for Scientific Research — Flanders (Belgium).

Dagstuhl Seminar Proceedings 09031
Symmetric Cryptography
http://drops.dagstuhl.de/opus/volltexte/2009/1951

3 Attack Principle

Each round of Maraca processes a new 1024-bit message block and reuses three previous message blocks from a window of 47. A round consists of three identical 1024-bit permutations, each containing 128 copies of the Maraca S-box. Thus, we have 1024 degrees of freedom at our disposal for 384 S-boxes. This is clearly insufficient to be able to impose three independent affine conditions on the inputs of each S-box. But it is possible to carefully choose which conditions to impose on which S-boxes, such that many of these conditions depend on earlier conditions, and will thus be satisfied automatically. Then, on average, only 960 conditions are required per round of Maraca, leaving 64 degrees of freedom unused per message block.

After the entire message has been processed, Maraca performs 47 blank rounds in which previous message blocks are reused, and a number of final calls to the 1024-bit permutation. The latter can be inverted trivially, but the former can not, as they depend on the input message. No new degrees of freedom are available in these 47 blank rounds, but we can use the yet unused degrees of freedom available from the earlier rounds. After 750 message blocks, enough degrees of freedom can be accumulated to also overcome the 47 blank rounds.

4 Practical Results

The attack consists of two phases: a one-time precomputation phase and an online phase. The precomputation phase determines how to linearise each S-box, and constructs a 1024-dimensional affine space of input messages for which the entire Maraca hash function becomes an affine function, except for the final permutations which we can simply invert, This phase took about 10.7 CPU-days and 20 GB of distributed memory on a cluster of 32 AMD Opteron nodes. The online phase consists of inverting the final permutations, and then combining the appropriate basis messages from the 94 MB data file generated by the precomputation. This takes just a few seconds on an average PC and is guaranteed to succeed for any given digest value.

5 Conclusion

We have shown a practical preimage attack on the hash function proposal Maraca. This clearly shows that Maraca is not preimage resistant, and hence not a secure cryptographic hash function.

References

1. Robert J. Jenkins Jr., *"Maraca: Algorithm Specification,"* Submitted to the NIST SHA-3 competition, 2008. Available online at `http://burtleburtle.net/bob/crypto/maraca/nist/Supporting_Documentation/specification.pdf`
2. Anne Canteaut and Mara Naya-Plasencia, *"Internal collision attack on Maraca,"* 2008 (to appear). Preprint available online at `http://ehash.iaik.tugraz.at/uploads/5/52/Maraca.pdf`