SYMAES: A Fully Symbolic Polynomial System Generator for AES-128[⋆]

Vesselin Velichkov[1,2,⋆⋆], Vincent Rijmen[1,2,3], and Bart Preneel[1,2]

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

SYMAES is a software tool that generates a system of polynomials in GF(2), corresponding to the round transformation and key schedule of the block cipher AES-128 [1].

Most of the existing polynomial system generators for AES are typically used under the assumption that the plaintext and ciphertext bits are known, and therefore are treated as constants. Although some of the generators, such as the AES (SR) Polynomial System Generator [2,3], can also be used when this assumption is not made, the instructions to do this are not always very natural. SYMAES is specifically designed to address the case in which (some of) the plaintext and ciphertext bits are unknown and are therefore treated as symbolic variables. Such a scenario is realistic and arises during the algebraic cryptanalysis of AES-based constructions, where only parts of the plaintext and/or ciphertext are known. An example of such a construction is the stream cipher LEX [4], a small-scale version of which has been analysed using SYMAES [5].

The inputs to SYMAES are the bits of the plaintext and the bits of the original key, represented as symbolic variables in GF(2). The output is a system of equations describing the output bits of one round of AES as a function of the input bits and the key. SYMAES also generates symbolic equations for the AES key schedule. Then, the bits of the round keys are expressed as polynomials in the bits of the original key.

As a final note we would like to stress that SYMAES should not be seen as a competitor to existing AES polynomial system generators, but rather as an addition to them. SYMAES achieves in a more natural way what can also be achieved using SR [3]. Similarly to SR, SYMAES is also written in Python and is used within the open source computer algebra Sage [6]. This makes possible a future integration of the SYMAES code into SR.

This submission is accompanied by an appendix containing the SYMAES source code and usage instructions.

References

1. Joan Daemen, Vincent Rijmen, The Design of Rijndael: AES - The Advanced Encryption Standard, Springer-Verlag, 2002.
2. Carlos Cid, Sean Murphy, Matthew J. B. Robshaw: Small Scale Variants of the AES. FSE 2005: 145-162.

⋆ This work was supported in part by the IAP Program P6/26 BCRYPT of the Belgian State (Belgian Science Policy), and in part by the European Commission through the ICT program under contract ICT-2007-216676 ECRYPT II.

⋆⋆ DBOF Doctoral Fellow, K.U.Leuven, Belgium.

3. Martin Albrecht, Small Scale Variants of the AES (SR) Polynomial System Generator, available at `http://www.sagemath.org/doc/reference/sage/crypto/mq/sr.htm`.
4. Alex Biryukov, The Design of a Stream Cipher Lex. Selected Areas in Cryptography 2006:67-75.
5. V. Velichkov, V. Rijmen, and B. Preneel, Algebraic Cryptanalysis of a Small-Scale Version of Stream Cipher LEX, IET Information Security Journal, 16 pages, 2009, *to appear*.
6. Stein, William, *Sage: Open Source Mathematical Software (Version 3.1.4)*, The Sage Group, 2008, `http://www.sagemath.org`.