

Secure AES128 on ATMEGA8515

Laboratory of Embedded Security, ANSSI

15 janvier 2018

1 Introduction

The members of the laboratory of embedded security has developed two versions of AES128 for ATMEGA8515 device. The implementation codes are published for research and pedagogical purposes only. The ATMEGA8515 component is not a secure component ; in particular it works with an external clock and contains no hardware random generator. The information leakage is consequently particularly high and there is almost no jittering (traces' acquisition should therefore not suffer from too much desynchronisation). To secure the implementation, it has been chosen to apply state of the art techniques : basic countermeasures for version 1 and improved countermeasures for version 2.

The specific platform on which the implementations has been tested is the WB electronics 64 Kbit ATmega chipcard <http://www.infinityusb.com/default.asp?show=store&ProductGrp=8> : it is made of an ATMEGA8515 embedded in a smartcard form factor, which represents a realistic SCA attacks target.

2 Implementation Codes

The AES128 implementations have been embedded in a minimal OS based on the open source OS SOSSE release under GPL v2.0 license (cf. www.mbsks.franken.de/sosse/). The AES128 API can therefore be executed through the ISO-7816 interface of the card (which is implemented in software by SOSSE on the MCU I/O pins). The list of available APDUs and their basic description are given in the table below (Table 1). For information, a python script is also provided which launches an AES128 encryption et get the result (cf. `script-AES128-enc.py`) ;

Command	Description	Code					
		CLA	INS	P1	P2	Lc/Le	Data
<code>setKey</code>	Load the 16-bytes key	0x80	0x10	0x00	0x00	0x10	16 Bytes of the key
<code>getKey</code>	Read the 16-bytes key	0x80	0x12	0x00	0x00	0x10	
<code>setInput</code>	Load the 16-bytes message	0x80	0x20	0x00	0x00	0x10	16 Bytes of the message
<code>getInput</code>	Read the 16-bytes message	0x80	0x22	0x00	0x00	0x10	
<code>setMask</code>	Load the 18-bytes masking	0x80	0x30	0x00	0x00	0x12	18 Bytes of the mask
<code>getMask</code>	Read the 18-bytes masking	0x80	0x32	0x00	0x00	0x12	
<code>getOutput</code>	Read the AES128 output	0x80	0x42	0x00	0x00	0x10	
<code>launchAES</code>	Launch AES128 enc.	0x80	0x52	0x00	0x00	0x00	

TABLE 1 – ADPU Commands

The project sources (OS + secure AES128) are provided in two different archives (one for each version, low security or high security) : **Version1** and **Version2**. The AES128 implementations are essentially based on the papers [1], [2], [3], [5] and [4]. The source files contain many comments which should perfectly clarify the choices made to secure the algorithms and the implementation details. The (modified/simplified) sources of SOSSE still contain the original comments.

Références

- [1] M.-L. Akkar and C. Giraud. An Implementation of DES and AES, Secure against Some Attacks. In Ç. Koç, D. Naccache, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*, pages 309–318. Springer, 2001.
- [2] S. Chari, C. Jutla, J. Rao, and P. Rohatgi. Towards Sound Approaches to Counteract Power-Analysis Attacks. In M. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 398–412. Springer, 1999.
- [3] G. Fumaroli, A. Martinelli, E. Prouff, and M. Rivain. Affine Masking against Higher-Order Side Channel Analysis. In *Selected Areas in Cryptography*, pages 262–280, 2010.
- [4] M. Rivain, E. Prouff, and J. Doget. Higher-Order Masking and Shuffling for Software Implementations of Block Ciphers. In C. Clavier and K. Gaj, editors, *CHES*, volume 5747 of *Lecture Notes in Computer Science*, pages 171–188. Springer, 2009.
- [5] M. von Willich. A technique with an information-theoretic basis for protecting secret data from differential power attacks. In B. Honary, editor, *IMA Int. Conf.*, volume 2260 of *Lecture Notes in Computer Science*, pages 44–62. Springer, 2001.