Documentation of Xoodyak Software Implementations

Shuohang Peng Cankun Zhao

April 17, 2022

1. Masked Xoodyak Software Implementations
   This repository contains first-order masked Xoodyak software implementations , written in C. The masked software interface follows the Call for Protected Software Implementations of Finalists in the NIST Lightweight Cryptography Standardization Process. The Values of share number (NUM_SHARES_M, NUM_SHARES_C, NUM_SHARES_AD, NUM_SHARES_NPUB, and NUM_SHARES_KEY) is defined in the file api.h.

2. Protection Methods

   (a) Name of the applied countermeasures: **ISW Scheme, t-SNI Refresh**

   (b) Primary reference: **[ISW03]** **[RP10]** **[GGR11]** **[CPRR14]** **[BBE$^+$18]**

3. Results of the Preliminary Security Evaluation

   (a) Attack/leakage assessment type: Test Vector Leakage Assessment with fixed key, fixed nonce, fixed adata, fixed plaintext (ciphertext) vs. fixed key, random nonce, random adata, random plaintext (ciphertext)

   (b) Experimental setup:

      i. Measurement platform and device-under-evaluation: ChipWhisperer, CW308 with STM32F303 UFO target

      ii. Description of measurements:
      Oscilloscope: Teledyne LECROY 8404M Digital
      Oscilloscope Sampling rate: 200MS
      Oscilloscope bandwidth: 4GHz
      Oscilloscope resolution: 8-bit

   (c) Attack/leakage assessment characteristics

      i. Data inputs and performed operations: plaintext/ciphertext and ad are generated by python file Xookyak_1st.ipynb.
      generate_shares_encrypt and generate_shares_decrypt are written in C which implemented on STM32F303 UFO target.
      combine_shares_encrypt and combine_shares_decrypt are written in C which implemented on STM32F303 UFO target.

      ii. Source of random and pseudorandom inputs: STM32F303 UFO target: custom randombytes.c function using salsa20 (from supercop) whose seed gets from python file Xookyak_1st.ipynb. All random bytes are generated before the execution of encryption/decryption, and the randombytes function is not called during the execution.

      iii. Trigger location relative to the execution start time of the algorithm: Prior and after the call to crypto_aead_encrypt_shared and crypto_aead_decrypt_shared

4. Attack Specific Data

   (a) Attack point:trigger prior and after crypto_aead_encrypt_shared, trigger prior and after crypto_aead_decrypt_shared

   (b) Attack/leakage assessment type: Test Vector Leakage Assessment with fixed key, fixed nonce, fixed adata, fixed plaintext (ciphertext) vs. fixed key, random nonce, random adata, random plaintext (ciphertext)

References

[BBE+18] Gilles Barthe, Sonia Belaïd, Thomas Espitau, Pierre-Alain Fouque, Benjamin Grégoire, Mélissa Rossi, and Mehdi Tibouchi. Masking the GLP lattice-based signature scheme at any order. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 354–384. Springer, Heidelberg, April / May 2018.

[CPRR14] Jean-Sébastien Coron, Emmanuel Prouff, Matthieu Rivain, and Thomas Roche. Higher-order side channel security and mask refreshing. In Shiho Moriai, editor, *FSE 2013*, volume 8424 of *LNCS*, pages 410–424. Springer, Heidelberg, March 2014.

[GGR11] Josh Jaffe Gilbert Goodwill, Benjamin Jun and Pankaj Rohatgi. A testing methodology for side-channel resistance validation. In *NIST Non-Invasive Attack Testing Workshop*, Nara, Japan, 2011.

[ISW03] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 463–481. Springer, Heidelberg, August 2003.

[RP10] Matthieu Rivain and Emmanuel Prouff. Provably secure higher-order masking of AES. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, volume 6225 of *Lecture Notes in Computer Science*, pages 413–427. Springer, 2010.