Efficient and Side-Channel Resistant Authenticated Encryption of FPGA Bitstreams

Andrey Bogdanov
Technical University of Denmark,
Department of Mathematics, Denmark
a.bogdanov@mat.dtu.dk

Amir Moradi, Tolga Yalçın
Ruhr-University Bochum,
Embedded Security Group, Germany
{amir.moradi,tolga.yalcin}@rub.de

***Abstract*—State-of-the-art solutions for FPGA bitstream protection rely on encryption and authentication of the bitstream to both ensure its confidentiality, thwarting unauthorized copying and reverse engineering, and prevent its unauthorized modification, maintaining a root of trust in the field. Adequate protection of the FPGA bitstream is of paramount importance to sustain the central functionality of dynamic reconfiguration in a hostile environment.**

In this work, we propose a new solution for authenticated encryption (AE) tailored for FPGA bitstream protection. It is based on the recent proposal presented at DIAC'12: the AES-based authenticated encryption scheme ALE. Our comparison to existing AES-based schemes reveals that ALE is at least twice more resource-efficient than the best AE modes of operation instantiated with AES. In the view of the recent successful side-channel attacks on Xilinx Virtex bitstream encryption, we investigate the possibility for side-channel resistant implementations of all these AES-based AE algorithms using state-of-the-art threshold masking techniques. Also in this side-channel resistant setting, the protected ALE design is about twice more resource-efficient than the best AE modes of operation with the same countermeasure.

We conclude that the deployment of dedicated AE schemes such as ALE significantly facilitates the real-world efficiency and security of FPGA bitstream protection in practice: Not only our solution enables authenticated encryption for bitstream on low-cost FPGAs but it also aims to mitigate physical attacks which have been lately shown to undermine the security of the bitstream protection mechanisms in the field.

***Keywords*-FPGA; bitstream; authenticated encryption; side-channel analysis;**

I. Introduction

A. Authenticated encryption and FPGA bitstream

The functionality of customizing the logic gates of a device in the field is fundamental to reconfigurable hardware such as FPGAs. The reliability of reconfiguration is gaining in importance as applications go pervasive, the connectivity increases, and the environments get more hostile. The basic threats the encryption of a FPGA bistream aims to mitigate are unauthorized copy and reverse engineering. It was not until recently though that the major FPGA manufacturers included authentication as an additional protection mechanism for the bitstream. Authentication targets to prevent unauthorized bitstream from being executed by the device or replay of older bitstreams that might contain bugs. A failure to provide authenticity for the bitstream can result in a tampered bitstream leaking confidential data stored on the device or giving away the bitstream itself. Therefore, it is always *authenticated encryption* that should be implemented for bitstream protection in the field.

A lot of security protocols and data formats for bitstream update are thinkable [6], [9]–[11], which may vary for every individual product and vendor. However, what all of them have in common is that a cryptographic core is required to implement the authenticated encryption. It is also the authenticated encryption engine that accounts for the performance bottleneck at reconfiguration time since the bulk of bitstream has to be decrypted and verified in real time with reasonable hardware resources. For instance, Xilinx [12] reports the deployment of AES-256 in CBC mode for bitstream encryption and HMAC based on the SHA-256 hash function for bitstream authentication in the authenticate-then-encrypt manner for Virtex-6 (on the device, the decryption is followed by the MAC verification on the bitstream plaintext). However, this requires the implementation of both AES-256 decryption and SHA-256 engines which is quite costly in terms of area requirements.

As a matter of fact, much more efficient and compact authenticated encryption schemes exist than that both in terms of underlying algorithms and modes. We will confine ourselves to the U.S. encryption standard AES in the standardized modes [2], [4] of operation such as OCB, GCM and CCM as reference points in the remainder of the paper. Note that some non-AES-based algorithms such as Grain-128a [3] and Hummingbird-2 [5] are out of scope here. AES-CCM and AES-GCM have been proposed and evaluated for the authenticated encryption of FPGA bitstreams in [7] and [8], respectively.

In this paper, we suggest to deploy the novel AES-based authenticated encryption ALE proposed in [1] for authenticated bitstream encryption. It turns out that ALE is not only much more efficient than the AES-256-CBC/HMAC-SHA-256 solution of Virtex-6 but also beats AES-OCB, AES-GCM and AES-CCM by a considerable margin.

B. Side-channel attacks and FPGA bitstream

The classical adversary attacking bitstream confidentiality and authenticity has access to the communication between the legitimate bitstream source and the reconfigurable device. Therefore, it is enough to use cryptographically sound mechanisms for authenticated encryption to thwart a classical attack. Clearly, all solutions mentioned above are sound constructions and the classical adversary do not pose any serious threat here.

978-1-4673-2921-7/12/$31.00

However, in the real world, the attacker can also gain physical access to the device which changes the attack setting significantly. Here, not only mathematical but also physical cryptanalysis becomes possible ranging from passive side-channel analysis over fault injection to invasive attacks. Recently, it has been shown [13], [14] that the bistream encryption of Xilinx Virtex II as well as Virtex 4 and 5 is vulnerable to passive side-channel analysis such as Correlation Power Analysis (CPA). These attacks have practical complexities and recover the key of bitstream encryption.

Thus, in the view of these newly arised threats, we think that any new practical solution for the authenticated bitstream of FPGA bitstreams should come with an option of powerful side-channel countermeasures that at least mitigate the passive side-channel attacks reported in [13], [14].

In this paper, we implement ALE along with AES-OCB, AES-GCM and AES-CCM using the threshold masking countermeasure based on secret sharing and multiparty computation [20] and [22]. We demonstrate that the masked implementation of ALE is much less resource-consuming than those of AES-OCB, AES-GCM and AES-CCM with the same countermeasure.

C. Our contributions

The contributions of this paper are as follows:

- We propose a new solution for authenticated encryption (AE) tailored for FPGA bitstream protection. It is based on the novel AES-based authenticated encryption scheme ALE (Section II). With the standard bitstream update rate of 800 Mbps at 100 MHz, our comparison to existing AE schemes such as AES-256-CBC/HMAC-256, AES-OCB, AES-GCM, and AES-CCM reveals that ALE is at least twice more resource-efficient than its closest competitor AES-OCB (a patent pending on the OCB mode of operation) and requires at least three times less resources compared to other royalty-free schemes: AES-GCM and AES-CCM (Section III).
- In the view of the recent successful side-channel attacks on Xilinx Virtex-II, Virtex-4 and Virtex-5 bitstream encryption, there is an increasing demand for the bitstream AE to come with adequate side-channel countermeasures. To address this need, we explore the costs of side-channel resistant implementations of all these AES-based AE algorithms using the state-of-the-art threshold masking technique based on secret sharing and multiparty computation. Also in this side-channel resistant setting, though inevitably increasing the resource occupation and decreasing the rate to 256 Mbps at 100 MHz, our results suggest that the protected ALE design is about twice more resource-efficient than AES-OCB or AES-GCM and requires three times less resources than AES-CCM (Section IV).

We work with both ASIC and FPGA implementations for the designs. We conclude that the deployment of dedicated AE schemes such as ALE significantly facilitates the real-world efficiency and security of FPGA bitstream protection in practice: Not only our solution enables authenticated encryption for bitstream on low-cost FPGAs but it also aims to mitigate physical attacks which have been lately shown to undermine the security of the bitstream protection mechanisms in the field.

II. ALE: AES-based Authenticated Lightweight Encryption

Authenticated encryption modes of operation for block ciphers have closely reached their theoretically attainable limit of one block cipher call per one block of encrypted and authenticated data. For instance, OCB [27] needs only one AES execution per block of data for long messages. Thus, to achieve performance and efficiency improvements compared to the modes of operation, dedicated designs appear necessary. The approach was taken by Bogdanov, Mendel, Regazzoni and Rijmen in [1] who have recently designed the lightweight authenticated encryption algorithm based on AES components called *ALE* (*A*uthenticated *L*ightweight *E*ncryption).

Essentially, ALE can be viewed as Pelican MAC [29] keyed in all rounds that leaks parts of the state in every round like the LEX stream cipher [28] to produce stream. It has an internal state of 256 bits that depends on both key and nonce. ALE is an online single-pass nonce-based authenticated encryption algorithm with associated data [1]. In FPGA bistream encryption, the nonce of ALE is implemented as a monotonic counter stored in a non-volatile memory which guarantees freshness.

The encryption/authentication procedure of ALE accepts a 128-bit master key K, a message M, associated data A and a 128-bit nonce N. The encryption/authentication procedure outputs the ciphertext C of exactly the same bit length as the message M and the authentication tag T of 128 bits for both the message M and associated data A. Its decryption/verification procedure accepts key K, ciphertext C, associated data A, nonce N, and tag T. It returns the decrypted message M if tag is correct or $\perp$ otherwise.

The encryption/authentication operation is given in Figure 1 [1] (for the case that the message length is not a multiple of 128 bits). The decryption/verification operation is exactly the same except that the ciphertext chunks get decrypted first and the resutling data is input into the state for verification. For space limitations, we refer the interested reader to the original paper [1] for a detailed specification of ALE and a security analysis.

III. Efficient implementation

We have implemented our ALE core (and other cores for comparison) on both ASIC and FPGA platforms. The ASIC implementation is the main target, since the bitstream encryption modules are meant to be implemented as an independent IP on the FPGA silicon. However, FPGA implementation may provide the flexibility to implement ALE on an existing FPGA without any modifications on it. Furthermore, FPGA numbers will be useful in determining the necessary resources required for evaluation of security and side-channel attack resistance on FPGA-based platforms.

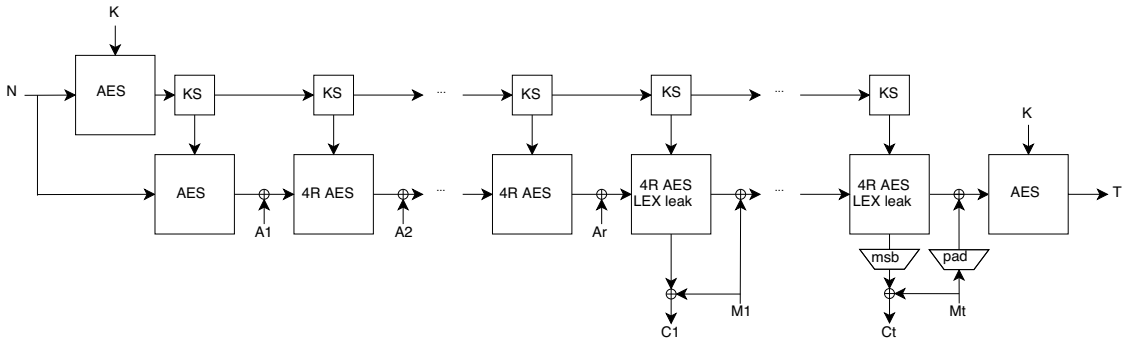

Figure 1. The operation of ALE [1]

In the implementation of the ALE core, we have focused on minimizing the area, while maintaining the target bitstream update rate of 800 Mbps at 100 MHz clock. The configuration clock frequency of 100 MHz in the Virtex-5 and Virtex-6 family of devices together with the 16-bits configuration data width and 50 MHz speed of Platform Flash XL determine the basic specifications of the core: Each 128-bit block of plaintext data can be read from the flash memory in 8 cycles of the 50 MHz clock of Flash XL, which correspond to 16 clock cycles of the 100 MHz clock of FPGA. This, in fact, corresponds to the target rate of 800 Mbps, which also means that each plaintext block has to be processed within 16 clock cycles by the target ALE core.

Since ALE processes plaintext blocks in 4 rounds, each round has to be completed within 4 cycles. Since the rounds are regular AES rounds, 16 $SubByte$ operations have to completed for state processing in each round, which corresponds to 4 $SubByte$ operations per cycle. This can be achieved using 4 $SubByte$ modules in parallel. However, the $ShiftRows$ operation requires all 16 $SubByte$ outputs. Therefore, it can be completed at the end of all 4 cycles per round. $MixColumns$ stage of AES can be reduced to a single $MixColumn$ operation by choosing the bytes of each cycle to correspond to a full column of the AES state matrix. In summary, state processing stage of the ALE core can be realized by only 4 $SubByte$ modules (s-boxes), a single $MixColumn$ module, a full $ShiftRows$ module and 128-bit registers. In addition, a first-in-first-out (FIFO) register is also required in order to store the 16-bit plaintext words read from the memory, since the processing order of the state bytes do not correspond to the reading order of the plaintext bytes they are to be XORed with in order to obtain ciphertext bytes (or vice versa during decryption).

On the key processind side, things are a bit more complex. Unlike the 4 rounds needed for state processing, each ALE block encryption (or decryption) requires 5 rounds of key processing. This corresponds to a total of 20 $SubByte$ operations per block to be completed within 16 cycles, which can be accomplished by $\lceil 20/16 \rceil = 2$ $SubByte$ modules in theory. However, in practice, since the last 4 bytes of the key state have to be $SubByte$'d before all else, this theoretical number does not hold. In a straigtforward implementation use of 4 $SubByte$ modules for key processing would solve this problem (albeit with 25% efficiency). Unfortunately, the extra key round makes this unapplicable. Either another set of 4 $SubByte$ modules have to be deployed, or the required $SubByte$ operations have to be precomputed and stored in extra registers. This results in a key processing module implementation using 4 $SubByte$ modules and 192-bit registers instead of 128-bits (32 of which are used for precomputation result storage and the other extra 32-bits for alignment of precomputed data with actual results). In fact, it is also possible to realize a precomputed key processing scheme that uses only 2 $SubByte$ modules, but this solution would quadruple the register usage, which is far worse than the 2 extra $SubByte$ modules in area cost.

Figure 2 shows the block diagram of the ALE core. The implementation uses a single 100 MHz clock. The data words read at 50 MHz from the flash is sampled by the 100 MHz clock before pushed into the FIFO. $SubByte$ modules are implemented using the composite field s-box structures over $GF(((2^2)^2)^2)$. In a 90nm implementation, the whole circuit occupies less than $8Kgates$.

Although, several different implementations of existing schemes (OCB, CCM and GCM) have been reported in the literature, a fair comparison is only possible with modules that have been implemented using the same design specifications, performance goals, technology and methodology. Therefore, we have also implemented OCB, CCM and GCM cores that can process 800 Mbps at 100 MHz. In each of these cores, key expansion are performed on-the-fly as in the ALE core. This is primarily intended to keep the design RAM-free, without which pre-computed round keys would have to be stored on registers, resulting in a much higher area cost than on-the-fly key processing. Furthermore, all three cores deploy the full set of 16 $SubByte$ and 4 $MixColumn$ modules, resulting in a fully parallel round-based implementation. Although this would make it possible to complete the plaintext (or ciphertext) block processing in 10 rounds, all three cores still require 16 clock cycles for block data processing. This is primarily due to the data access speed from the flash memory, which is limited to 128 bits in 16 cycles. However,

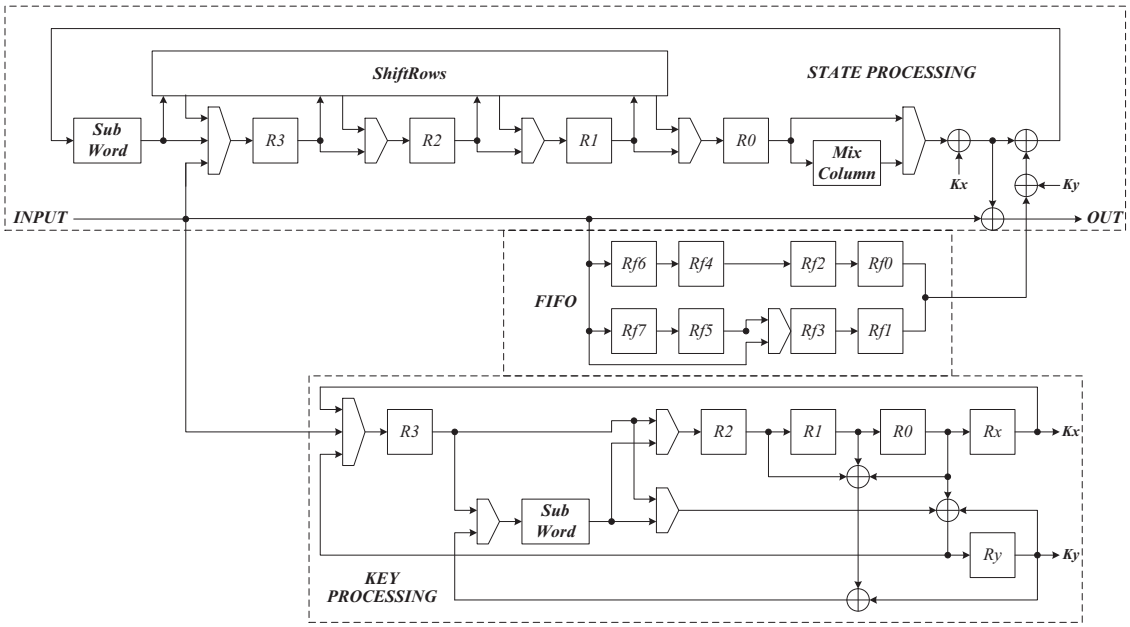

Figure 2. Block Diagram of the ALE Core

we have used this to our advantage by moving some non-round operations (such as the extra $AddRoundKey$ and tweak XORS in OCB) to these cycles, thereby reducing the hardware area at zero additional time cost.

Other core specific features can be summarized as follows:

- The OCB core requires support for both AES encryption and AES decryption operations. This is accomplished by sharing the finite field inverter of the s-box between both $SubByte$ and $InvSubByte$ functions, resulting in a $SubByte/InvSubByte$ module. However, the same could not be accomplished for the other core elements. Therefore, two seperate paths are implemented: One with $ShiftRows + MixColums$ and the other with $InvMixColumns + InvShiftRows$ multiplexed between modes. Tweaks are pre-computed and stored in the tweak register.
- A global hash (GHASH) module is implemented with the GCM core. It is a partial finite field multiplier (8×128) which computes the 128×128-bit finite field multiplication within the 16 clock cycles present for each data block.
- In order to keep up with the target speed achieved by other cores, the CCM module deploys two parallel state processing paths: One for the counter (CTR) mode and the other for the CBC mode. Key processing path is shared between the two.
- AES-CBC/HMAC-SHA-256 core has an independent HMAC-SHA-256 core running in parallel with the AES-CBC core, making it the largest of all.

Table I gives the gate count comparison results for unprotected versions of all five schemes in three different technologies: UMC 130nm, UMC 90 nm, Nangate 45nm and Virtex-6 LX240T. All implementations are synthesized for area at a target frequency of 100 MHz using *Cadence Encounter RTL Compiler v10.1* and Xilinx ISE v14.1 for ASIC and FPGA, respectively. During all design phases, *Mentor Graphics Modelsim v6.5* have been used for verification.

Table I
AREA RESULTS FOR ALL AUTHENTICATED ENCRYPTION SCHEMES (UNPROTECTED VERSIONS)

Scheme	Area in GE (130nm)	Area in GE (90nm)	Area in GE (45nm)	Area in slices (V6)
ALE	**7.9K**	**7.1K**	**8.2K**	**594**
OCB	23.1K	18.1K	20.0K	2776
CCM	24.4K	20.4K	21.9K	1947
GCM	24.9K	21.6K	23.2K	2049
CBC/HMAC	34.3K	29.9K	33.5K	2201

We have also synthesized all versions of our design for maximum frequency. Again, ALE resulted in the best performance with a maximum operating frequency of 326 MHz for 90 nm technology, corresponding to a throughput of 2.6 Gbps. For the 45 nm technology, the maximum operating frequency and throughput exceeds 500 MHz and 4 Gbps, respectively.

IV. SIDE-CHANNEL RESISTANT IMPLEMENTATION

As stated, the decryption module embedded into the FPGAs to realize the bitstream encryption feature is in danger of being attacked by a side-channel adversary, which as reported in [13] and [14] may lead to entirely overcoming the expected security. Therefore, protecting the target module against such an adversary is of crucial importance. Masking which is

amongst the popular side-channel countermeasures aims at randomizing the intermediate values processed by the cryptographic module thereby avoiding the relation between the side-channel leakages and the intermediate values expected by the adversary [15]. A popular masking scheme which is quite efficient for linear operations is known as *boolean* (also called *additive*) masking. However, providing the boolean masked version of the nonlinear operations, e.g., S-box of symmetric ciphers, is not trivial. Although there exists a couple of different schemes on how to implement a masked AES S-box in hardware, e.g., in [16] and [17], the practical side-channel evaluations have shown still vulnerability of such schemes to power analysis attacks [18], [19]. On the other hand, threshold implementation (TI) which is a provably secure scheme against first-order attacks has been proposed in [20]–[22]. It makes use of the multi-party computation concept and boolean masking, and provides a roadmap how to implement a masked S-box with the minimum number of shares which is in a direct relation with the algebraic degree of the underlying S-box.

Although a threshold implementation of small (e.g., 4-bit) S-boxes can be made relatively straight forward (see [23] and [24]), it is a challenging task to deal with the larger S-boxes which have high algebraic degree. For example, making a threshold implementation of the AES S-box that follows all the requirements is still an open problem. However, the implementation reported in [25] has been provided by means of a heuristic scheme which makes use of fresh masks during the S-box computations. This design employs five pipelining stages and realizes the TI scheme with minimum number of shares, i.e., three. The design has been practically implemented on a Virtex-II pro FPGA and it side-channel vulnerability has been investigated practically. As short, no first-order attack could be successfully mounted using up to 100 million traces [25], but a mutual information analysis attack and a second-order DPA attack could recover the secret using around 20 million traces [26][1].

We have taken this design in order to increase the side-channel immunity of our proposed scheme. Table II gives the gate count comparison results for protected versions of all five schemes in three different technologies. Note that these numbers have been obtained by interpolation of gate counts for the protected S-boxes from [23] into our designs implemented with three parallel paths. As a consequence of the 3-cycle operation of these S-boxes, the throughput of each core drops to 256 Mbps from 800 Mbps. It is still possible to achieve over 800 Mbps with an operating frequency of 300 MHz. It should also be noted that no protection is applied for the GHASH and HMAC-SHA cores.

V. Conclusions

In this paper, we propose to deploy the recently designed AES-based authenticated encryption scheme ALE [1] for authenticated bitstream encryption on FPGAs. It turns out that ALE is not only much more efficient than the AES-256-CBC/HMAC-SHA-256 solution of Virtex-6 but also beats AES-OCB, AES-GCM and AES-CCM by a considerable margin. We implement the unprotected versions of ALE along with AES-OCB, AES-GCM and AES-CCM as well as explore their protected implementations using the threshold masking countermeasure based on secret sharing and multiparty computation. In both cases, our results suggest that ALE beats the nearest competition by more than a factor of 2 in terms of area efficiency. Even the protected version of ALE results in gate counts that are close to unprotected versions of authenticated encryption schemes commonly used on commercial FPGAs while still reaching one third of the highest attainable bitrates for the unprotected case (256 vs 800 Mbps).

[1]These numbers strongly depend on the development platform and the measurement setup.

Table II
Area results for all authenticated encryption schemes (protected versions)

Scheme	Area in GE (130nm)	Area in GE (90nm)	Area in GE (45nm)	Area in slices (V6)
ALE	**48.1K**	**45.7K**	**49.0K**	**7122**
OCB	133.1K	118.3K	123.8K	48150
CCM	173.0K	161.1K	165.5K	33500
GCM	138.5K	128.6K	133.4K	23900
CBC/HMAC	165.9K	152.7K	164.7K	18900

References

[1] Andrey Bogdanov, Florian Mendel, Francesco Regazzoni, Vincent Rijmen. Efficient Lightweight AES-Based Authenticated Encryption. Directions in Authenticated Ciphers (DIAC 2012), Stockholm, July 2012.

[2] ISO/IEC 19772:2009. Information Technology - Security techniques - Authenticated Encryption, 2009.

[3] Martin Ågren, Martin Hell, Thomas Johansson and Willi Meier. Grain-128a: a new version of Grain-128 with optional authentication. IJWMC 5(1), pages 48–59. 2011.

[4] Morris Dworkin. Recommendation for Block Cipher Modes of Operation: Methods and Techniques. NIST Special Publication 800-38D, 66 pages, NIST, 2001.

[5] Daniel W. Engels, Markku-Juhani O. Saarinen, Peter Schweitzer, and Eric M. Smith. The Hummingbird-2 Lightweight Authenticated Encryption Algorithm. RFIDSec 2011, LNCS, volume 7055, pages 19–31, Springer-Verlag, 2011.

[6] B. Badrignans, R. Elbaz, Lionel Torres. Secure FPGA Configuration Architecture Preventing System Downgrade. Field Programmable Logic and Applications (FPL 2008), pages 317–322, September 2008.

[7] B. Badrignans, D. Champagne, R, Elbaz, C. Gebotys, L. Torres. SARFUM: Security Architecture for Remote FPGA Update and Monitoring. ACM Transactions on Reconfigurable Technology and Systems, vol. 3, No. 2, May 2010.

[8] Yohei Hori, Akashi Satoh, Hirofumi Sakane, Kenji Toda. Bitstream Ecnryption and Authentication Using AES-GCM in Dynamically Reconfigurable Systems. IWSEC 2008, LNCS, vol. 5312, pages 261–278, Springer-Verlag, 2008.

[9] Saar Drimer and Markus G, Kuhn. A Protocol for Secure Remote Updates of FPGA Configurations. ARC 2009, LNCS, volume 5453, pages 50–61, Springer-Verlag, 2009.

[10] Saar Drimer. Security for Volatile FPGAs. PhD Dissertation, Cambridge University, 2009.

[11] Saar Drimer. Authentication of FPGA Bitstreams: Why and How. ARC 2007, LNCS, volume 4419, pages 73–84, LNCS, volume 4419, Springer-Verlag, 2007.

[12] Steve Trimberger, Jason Moore, Weiguang Lu. Authenticated Encryption for FPGA Bitstreams. FPGA 2011, February-March 2011.

[13] A. Moradi, A. Barenghi, T. Kasper, and C. Paar, On the vulnerability of FPGA bitstream encryption against power analysis attacks: extracting keys from Xilinx Virtex-II FPGAs. ACM CCS 2011, pp. 111–124, ACM, 2011.

[14] A. Moradi, M. Kasper, and C. Paar. Black-Box Side-Channel Attacks Highlight the Importance of Countermeasures — An Analysis of the Xilinx Virtex-4 and Virtex-5 Bitstream Encryption Mechanism. CT-RSA 2012, LNCS, vol. 7178, pp. 1–18, Springer-Verlag, 2012.

[15] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, 2007.

[16] E. Oswald, S. Mangard, N. Pramstaller, and V. Rijmen, "A Side-Channel Analysis Resistant Description of the AES S-Box," in *FSE 2005*, ser. LNCS, vol. 3557. Springer, 2005, pp. 413–423.

[17] D. Canright and L. Batina, "A Very Compact "Perfectly Masked" S-Box for AES," in *ACNS 2008*, ser. LNCS, vol. 5037. Springer, 2008, pp. 446–459, the corrected version at Cryptology ePrint Archive, Report 2009/011 http://eprint.iacr.org/.

[18] S. Mangard, N. Pramstaller, and E. Oswald, "Successfully Attacking Masked AES Hardware Implementations," in *CHES 2005*, ser. LNCS, vol. 3659. Springer, 2005, pp. 157–171.

[19] A. Moradi, O. Mischke, and T. Eisenbarth, "Correlation-Enhanced Power Analysis Collision Attack," in *CHES 2010*, ser. LNCS, vol. 6225. Springer, 2010, pp. 125–139.

[20] S. Nikova, C. Rechberger, and V. Rijmen, "Threshold Implementations Against Side-Channel Attacks and Glitches," in *ICICS 2006*, ser. LNCS, vol. 4307. Springer, 2006, pp. 529–545.

[21] S. Nikova, V. Rijmen, and M. Schläffer, "Secure Hardware Implementations of Non-Linear Functions in the Presence of Glitches," in *ICISC 2008*, ser. LNCS, vol. 5461. Springer, 2008, pp. 218–234.

[22] ——, "Secure Hardware Implementation of Nonlinear Functions in the Presence of Glitches," *J. Cryptology*, vol. 24, no. 2, pp. 292–321, 2011.

[23] A. Poschmann, A. Moradi, K. Khoo, C.-W. Lim, H. Wang, and S. Ling, "Side-Channel Resistant Crypto for Less than 2,300 GE," *J. Cryptology*, vol. 24, no. 2, pp. 322–345, 2011.

[24] B. Bilgin, S.Nikova, V.Nikov, V.Rijmen, and G.Stuetz. Threshold Implementations of all 3x3 and 4x4 S-boxes. CHES 2012, LNCS, vol. 7428, pp. 76–91, Springer Verlag, 2012.

[25] A. Moradi, A. Poschmann, S. Ling, C. Paar, and H. Wang, "Pushing the limits: A very compact and a threshold implementation of aes," in *EUROCRYPT 2011*, ser. LNCS, vol. 6632. Springer, 2011, pp. 69–88.

[26] A. Moradi. Statistical Tools Flavor Side-Channel Collision Attacks. EUROCRYPT 2012, LNCS, vol. 7237, pp. 428–445, Springer, 2012.

[27] Phillip Rogaway, Mihir Bellare, John Black and Ted Krovetz. OCB: A block-cipher mode of operation for efficient authenticated encryption. CCS 2001, pp. 196–205, ACM, 2001.

[28] Alex Biryukov. Design of a New Stream Cipher-LEX. The eSTREAM Finalists, LNCS, volume 4986, Springer-Verlag, 2008.

[29] Joan Daemen and Vincent Rijmen. The Pelican MAC Function. IACR Cryptology ePrint Archive, Report 2005/088, 2005.