Solving Circuit Optimisation Problems in Cryptography and Cryptanalysis*

Nicolas T. Courtois[1,2], Daniel Hulme[1,2], and Theodosis Mourouzis[1]

[1] University College London, UK,
[2] NP-Complete Ltd, London, UK

Abstract. One of the hardest problems in computer science is the problem of gate-efficient implementation. Such optimizations are particularly important in industrial hardware implementations of standard cryptographic algorithms. In this paper we focus on optimizing some small circuits such as S-boxes in cryptographic algorithms. We consider the notion of Multiplicative Complexity, a new important notion of complexity introduced in 2008 by Boyar and Peralta and applied to find interesting optimizations for the S-box of the AES cipher [13, 16, 15]. We applied this methodology to produce a compact implementation of several ciphers. In this short paper we report our results on PRESENT and GOST, two block ciphers known for their exceptionally low hardware cost. This kind of representation seems to be very promising in implementations aiming at preventing side channel attacks on cryptographic chips such as DPA. More importantly, we postulate that this kind of minimality is also an important and interesting tool in cryptanalysis.

Key Words: Block ciphers, non-linearity, algebraic attacks, circuit complexity, multiplicative complexity, algebraic cryptanalysis, side-channel attacks.

* This research was supported by the European Commission under the FP7 project number 242497 "Resilient Infrastructure and Building Security (RIBS)" and by the UK Technology Strategy Board under project 9626-58525.

1 Introduction

The problems of circuit complexity is one of the hardest and yet very important problems in computer science and complexity theory. Not everybody in the industry cares about improving their gate count by a small factor, but such optimizations are particularly important in hardware implementation of standard cryptographic algorithms, which in many security chips such as smart cards and RFID, will be one of the most costly components. Many heuristic algorithms for this problem have been invented, and with a lot of computing power one can find very decent optimizations [9], but these optimizations are frequently subject to further substantial improvement. In this paper we particularly focus on optimizing the S-boxes for industrial block ciphers.

Much less known and very surprising is that this is also an important topic in cryptanalysis. As shown in [4, 6, 5] such optimizations are also very important in order to speed up so called algebraic attacks on symmetric ciphers, and in the space of attacks which require very small quantities of data, these methods lead to currently best known attacks on a few ciphers (with more data, typically faster attacks will exist). In this paper we focus mostly on 4x4 S-boxes in ciphers such as PRESENT and GOST. These ciphers are known for their exceptionally low hardware implementation cost [12]. But this is also what makes them vulnerable to algebraic cryptanalysis.

2 S-box Optimization

In 2008 Boyar and Peralta introduced a new heuristic methodology to minimize the complexity of digital circuits [13, 16, 15]. It is based on the notion of Multiplicative Complexity (MC) which is a new and very deep notion of complexity invariant w.r.t. affine transformations. Their heuristic proposition is that a two step-process based on MC appears to be able to produce very good gate efficient implementation of several famous circuits such as the AES S-box [16, 15].

In this paper we apply this methodology to some cryptographically significant functions $GF(2)^4 \rightarrow GF(2)^4$ (i.e. 4x4 S-boxes). We developed software which allows us to compute **optimal** representations of these S-boxes w.r.t to this methodology. Then we apply these representations to obtain an algebraic representation of the whole cipher.

2.1 Motivations For Achieving Low-MC and Low Gate Count

There are three mains reasons why we want to determine and improve the complexity of various circuits.

1. Lower the implementation cost in silicon.
2. Prevent Side Channel attacks such as DPA. this is due to the fact the XORs are believed easy to protect against DPA through linear secret charing techniques. Then minimizing the number of AND gates is expected to lower the cost of general-purpose protections against side channel attacks which are developed to securely implement arbitrary digital circuits, such as for example developed in [11].
3. Algebraic Cryptanalysis of a symmetric cipher can be greatly improved of we use gate-efficient and compact representations, as demonstrated in [4–6]. This usually works only for cipher with a limited number of rounds. Then additional non-trivial higher-level “tricks” are needed to be able to really break a full cipher with many more rounds, see [5–7].

2.2 Gate Complexity and Multiplicative Complexity

Definition 2.2.1 (Gate Complexity (GC)).
Given a function $GF(2)^n \rightarrow GF(2)^m$ we define its Gate Complexity (GC) as the **minimum** number of 2-input gates of types XOR,OR,AND,OR needed.

This model is a model which the cost of all these gates is the same, which is relevant for example in so called Bit-slice implementations of block ciphers, such as for example in [1]. It is not yet the optimal model for silicon implementations, where certain gates are more costly to implement. However such optimizations are important and very hard to find for each model.

In 2008, Boyar and Peralta introduced the following fundamental and important notion of complexity [13, 16]:

Definition 2.2.2 (Multiplicative Complexity (MC)).
Given a function $GF(2)^n \rightarrow GF(2)^m$ we define its Multiplicative Complexity (MC) as the minimum number of AND gates which need to be used to implement this function, with an unlimited number of NOR and XOR gates.

This model considers that linear operations come "for free" and ask to minimize just the number of AND gates. The problem with Gate Complexity (GC) is that we are not in general able to determine its value, algorithms which find such optimizations are typically random stochastic explorations of large trees of solutions [9] and we are not sure if the optimizations are final or if they can still be improved. However, as we will see in this paper, the Multiplicative Complexity (MC) can be computed **exactly** by our methods which use SAT solver software.

2.3 Multiplicative Complexity As A Tool For Gate Complexity

Boyar and Peralta have a developed a heuristic methodology, where they optimise for of Multiplicative Complexity (MC) in order to produce also gate-efficient implementations:

1. **(Step 1)** First compute the multiplicative complexity.
2. **(Step 2)** Then optimise the number of XORs separately, see [14, 8].
3. Optional Step 3: At the end do additional optimizations to decrease the circuit depth, an possibly additional software optimizations, see [13, 16],

This methodology was then used to produce new worldwide records in gate efficient implementation of several famous circuits such as the AES S-box, and many other circuits related to finite fields and algebra, [16, 15, 15].

In this paper we focus on optimisation of functions $GF(2)^4 \rightarrow GF(2)^4$ which are immensely popular in cryptography [**?**]. We have implemented fully and with our own optimisation methods, both Steps 1. and 2. above.

The crucial feature of our implementation is that BOTH our Steps 1. and 2. are OPTIMAL, i.e. they produce the best possible optimizations which can be obtained by following these two steps. Optimality was achieved due to SAT solver software, we convert our problem to SAT and it either outputs SAT, and a solution, which we convert to a concrete circuit optimization, or it outputs UNSAT, and we are certain that there is no solution. There is third possibility, that the SAT solver software runs for a very long time and we do not have enough computing power to decide whether the result is SAT or UNSAT, but this have never happened for 4x4 S-boxes. Accordingly, we were able to produce optimal optimizations or this type for every 4x4 S-box we have ever tried. This is very rare in complexity: to be able to completely determine what the best possible result is.

We must say that these methods are at prototyping stage and they are so far slower than other known methods [9]. Likewise, we do not claim that we can optimise the linear parts as quickly as by recent methods described in [13, 14], but only that we can optimize to the strictest minimum possible, which probably can also be achieved in [8], however it seems that we are the first also to apply SAT solvers also to optimize non-linear circuits.

Our solutions are optimal and thus proven to be impossible to improve (automated software proof with UNSAT). This is they would be provably optimal, if we had a proof of correctness of the SAT solver software. Then they could be transformed to produce fully verifiable mathematical proofs written in a formal language, which prove these optimality results. Such proofs would not be published in scientific papers, but rather as lengthy computer files, which should come together with a formal system able to efficiently check the correctness of such proofs. This is a major topic for further research which would require one to develop a whole new formal language and software to manipulate it.

3 Optimizing the Present S-box

The Present S-box is defined as $\{12, 5, 6, 11, 9, 0, 10, 13, 3, 14, 15, 8, 4, 7, 1, 2\}$. We will number the least significant bits starting from 1.

Theorem 3.0.1. The Multiplicative Complexity of the PRESENT S-box is exactly 4.

Proof: For 3 AND gates our thoroughly designed and tested system outputs UNSAT. We have obtained an automated proof of this fact which takes a few seconds on a PC and can reproduced and checked. For 4 AND gates, our system outputs SAT and a solution. Further optimisation of the linear part, which is also optimal allowed us to minimize the number of XORs to the strict minimum possible (prove by additional UNSAT results). As a result, for example we obtained an implementation of the PRESENT S-box with 25 gates, 4 AND, 20 XOR, 1 NOR which is optimal w.r.t our Boyar-Peralta 2-step methodology but not optimal in overall gate complexity. 25 gates are still not very satisfactory.

A better result in terms of gate complexity can be achieved by the following method: we observe that AND gates and OR gates are affine equivalents, and it is likely that **if** we implement certain AND gate with OR gates, we might be able to further reduce the overall complexity of the linear parts. We may try all possible 2^4 cases where some AND gates are implemented with OR gates. By this method, starting with the right optimization with MC=4, as several such optimizations may exist, we can obtain the following new implementation of the PRESENT S-box which requires only 14 gates total (!):

```
T1=X2^X1; T2=X1&T1; T3=X0^T2; Y3=X3^T3; T2=T1&T3; T1^=Y3; T2^=X1;
T4=X3|T2; Y2=T1^T4; T2^=~X3; Y0=Y2^T2; T2|=T1; Y1=T3^T2;
```

Discussion. Our best optimisation of the PRESENT S-box does **not** contradict the Boyar-Peralta heuristic to the effect that some of the best possible gate-efficient implementations are very closely related to the notion of multiplicative complexity. However the most recent implementations of the AES S-box, in the second paper by Boyar and Peralta, show that further improvements, and also circuit depth improvements, can be achieved also by relaxing the number of ANDs used as in the latest optimization of the 4-bit inverse in $GF(2^4)$ for AES given on Fig 1. in [16]. Moreover, now we are going to demonstrate that there are many S-boxes which are worse than PRESENT in Multiplicative Complexity (MC), yet require less gates to be implemented.

4 The GOST S-boxes

We consider the main standard and most widely known version of the GOST block cipher, also known as ”id-GostR3411-94-CryptoProParamSet” [10] and also known as the one used by the Central Bank of the Russian Federation [12]. By running the same method and programs we obtained the following result:

Theorem 4.0.2. The Multiplicative Complexity of the eight GOST S-boxes S1,S2,S3,S4,S5,S6,S7,S8 is exactly equal to respectively 4,5,5,5,5,5,4,5.

Related Work: We can compare this to the results in Table on page 226 of [12] where we see that these 8 S-boxes are also on average more expensive than the PRESENT S-box in the sense of Gate Equivalent (GE) cost, yet it is the PRESENT S-box which is better against linear and differential cryptanalysis, see Table 3 in [12]. However in our Multiplicative Complexity (MC) metric, in our Gate Multiplicative Complexity (GC) metric, and also in the strict GE cost metric in [12], it is clear that (on the contrary) the complexity of the PRESENT S-box is always lower. Therefore we conjecture that PRESENT S-box will be much weaker than the GOST S-boxes against many types of algebraic cryptanalysis such as [6, 7], and thus it is probably a bad idea to use the GOST cipher with PRESENT S-boxes as proposed in [12].

5 Conclusion

In this paper we have applied the new Boyar-Peralta notion of Multiplicative Complexity (MC) to derive efficient implementations of the S-boxes in two ciphers, PRESENT and GOST. We have developed software which does handle the main two steps of this process, through a reduction to a SAT problem. Our method is practical though rather slow, so far we have been able to optimize every 4x4 S-box we tried, but not beyond. Yet it is unique and very powerful, because all the results are optimal and come with a mathematical proof (automatically found by the software) that they cannot be improved.

In the case of PRESENT it happens that the Boyar-Peralta heuristics [13, 16] works extremely well, and the best possible gate-efficient optimization we could find also contains the (optimal) lowest possible number of non-linear gates(!). However GOST S-boxes have on average higher Multiplicative Complexity (MC) and yet lower implementation cost, so this heuristics is unlikely to be always the best method to optimise a circuit. Clearly better optimizations are likely to use a few more non-linear gates, as also seen for AES, cf. Fig 1 in [16].

Interestingly, we are able to **provably minimize** the number of non-linear gates in a given cipher, to a rather low number of ≤ 5 per S-box. Such optimizations are important in synthesis of implementations of circuits secure against side-channel attacks [11] In future works we will show how S-box optimizations greatly help to break the full-round block cipher GOST and its many variants [10, 12]. It is extremely rare to see a real-life block cipher which can be broken faster than brute force. This however requires a lot of additional work, see [6, 7].

References

1. Martin Albrecht, Nicolas T. Courtois, Daniel Hulme, Guangyan Song: *Bit-Slice Implementation of PRESENT in pure standard C,* v1.5, 26/08/2011, open-source code available at `https://bitbucket.org/malb/algebraic_attacks/src/tip/present_bitslice.c`
2. A. Bogdanov, L.R. Knudsen, G. Leander, C. Paar, A. Poschmann, M.J.B. Robshaw, Y. Seurin, and C. Vikkelsoe: *PRESENT: An Ultra-Lightweight Block Cipher,* In CHES 2007, LNCS 4727, pp. 450466, Springer, 2007.
3. Nicolas Courtois: *General Principles of Algebraic Attacks and New Design Criteria for Components of Symmetric Ciphers,* in AES 4, LNCS 3373, pp. 67-83, Springer, 2005.
4. Nicolas Courtois, Gregory V. Bard: *Algebraic Cryptanalysis of the Data Encryption Standard,* In Cryptography and Coding, 11-th IMA Conference, pp. 152-169, LNCS 4887, Springer, 2007. Preprint available at `eprint.iacr.org/2006/402/`.
5. Nicolas Courtois, Gregory V. Bard, David Wagner: *Algebraic and Slide Attacks on KeeLoq,* In FSE 2008, pp. 97-115, LNCS 5086, Springer, 2008.
6. Nicolas Courtois: *Algebraic Complexity Reduction and Cryptanalysis of GOST,* Preprint available at `http://www.nicolascourtois.com/papers/gostac11.pdf`.
7. Nicolas Courtois: *Security Evaluation of GOST 28147-89 In View Of International Standardisation,* document officially submitted to ISO in May 2011, At `http://eprint.iacr.org/2011/211/`.
8. Carsten Fuhs and Peter Schneider-Kamp: *Synthesizing Shortest Linear Straight-Line Programs over GF(2) Using SAT,* In SAT 2010, Theory and Applications of Satisfiability Testing, Springer LNCS 6175, pp. 71-84, 2010.
9. B. R. Gladman, software for efficient boolean function decompositions for the eight Serpent S boxes and their inverses, available at `http://gladman.plushost.co.uk/oldsite/cryptography_technology/serpent/index.php`.
10. A Russian reference implementation of GOST implementing Russian algorithms as an extension of TLS v1.0. is available as a part of OpenSSL library. The file gost89.c contains eight different sets of S-boxes and is found in OpenSSL 0.9.8 and later: `http://www.openssl.org/source/`
11. Svetla Nikova, Vincent Rijmen, Martin Schläffer: *Secure Hardware Implementation of Nonlinear Functions in the Presence of Glitches,* Special Issue on Hardware and Security of Journal of Cryptology, 27 pages, 2011. `http://homes.esat.kuleuven.be/~snikova/JOC_2011.pdf`
12. Axel Poschmann, San Ling, and Huaxiong Wang: *256 Bit Standardized Crypto for 650 GE GOST Revisited,* In CHES 2010, LNCS 6225, pp. 219-233, 2010.
13. Joan Boyar, René Peralta: *A New Combinational Logic Minimization Technique with Applications to Cryptology.* In SEA 2010: 178-189.
    An early version was published in 2009 at `http://eprint.iacr.org/2009/191`. It was revised 13 Mar 2010.
14. Joan Boyar, Philip Matthews, René Peralta: *On the Shortest Linear Straight-Line Program for Computing Linear Forms,* In MFCS 2008: 168-179.
15. Web page with all circuit minimialisation results obtained at Yale University, `http://cs-www.cs.yale.edu/homes/peralta/CircuitStuff/CMT.html`.
16. Joan Boyar and Rene Peralta; *A depth-16 circuit for the AES S-box,* `http://eprint.iacr.org/2011/332`