# Security of a Wide Trail Design

Joan Daemen[1] and Vincent Rijmen[2,3]

[1] ERG Group — ProtonWorld, Belgium
Joan.Daemen@protonworld.com
[2] Cryptomathic, Belgium
Vincent.Rijmen@cryptomathic.com
[3] IAIK, Graz University of Technology, Austria

**Abstract.** The wide trail design strategy claims to design ciphers that are both efficient and secure against linear and differential cryptanalysis. Rijndael, the AES, was designed along the principles of this strategy. We survey the recent results on Rijndael and examine whether the design strategy has fulfilled its promise.

## 1 Introduction

In October 2000, the US National Institute of Standards and Technology (NIST) announced that Rijndael was selected as Advanced Encryption Standard (AES). Since then, Rijndael has probably become the best-known block cipher that was designed according to the wide trail design strategy. The findings that have been obtained as a result of the increased attention that was devoted to Rijndael, allow to evaluate the soundness of the wide trail design strategy.

In Section 2, we present briefly the principles of the wide trail design strategy. The resulting security and performance are explained in Section 3 and Section 4. Subsequently, in Section 5 we give an overview of the most important cryptanalysis performed on Rijndael since its submission to the AES process. In Section 6, we discuss the improvements in performance of dedicated hardware implementations.

This paper doesn't contain a full description of Rijndael. For a complete specification, we refer the reader to [DR02a].

## 2 The Wide Trail Design Strategy

The wide trail design strategy can be used to design a variety of symmetric cryptographic primitives such as hash functions and stream ciphers, and block ciphers. Its best known application is the design of Rijndael, which is an example of a particular class of block ciphers, the *key-alternating* block ciphers. A key-alternating block cipher is an iterative block cipher with the following properties:

– Alternation: the cipher is defined as the alternated application of key-independent round transformations and the application of a round key. The first round key is applied before the first round and the last round key is applied after the last round.

A. Menezes, P. Sarkar (Eds.): INDOCRYPT 2002, LNCS 2551, pp. 1–11, 2002.

– Binary Key Addition: the round keys are applied by means of a simple XOR: to each bit of the intermediate state a round key bit is XORed.

In the following, we will explain the principles of the wide trail design strategy, applied to the design of key-alternating block ciphers.

In the wide trail strategy, the round transformations are composed of two invertible steps:

– $\gamma$: a local non-linear transformation. By local, we mean that any output bit depends on only a limited number of input bits and that neighboring output bits depend on neighboring input bits.
– $\lambda$: a linear mixing transformation providing high diffusion. What is meant by high diffusion will be explained in the following sections.

We will now discuss both steps in some more detail.

### 2.1 The Non-linear Step

A typical construction for $\gamma$ is the so-called *bricklayer mapping* consisting of a number of invertible S-boxes. In this construction, the bits of input vector $a$ are partitioned into $n_t$ $m$-bit *bundles* $a_i \in Z_2^m$ by the so-called *bundle partition*. The block size of the cipher is given by $n_\mathrm{b} = mn_t$. In the case of the AES, the bundle size $m$ is 8, hence bundles are bytes.

In the early 1990's, many block cipher designs concentrated solely on the design of large nonlinear S-boxes. In contrast, the wide trail strategy imposes very few requirements on the S-box. The only criteria are the upper bound for input-output correlations and the upper bound for the difference propagation.

Instead of spending most of the resources on the S-boxes themselves, the wide trail strategy aims at designing the round transformation so that the effect of the S-boxes is maximised. In ciphers designed by the wide trail strategy, a relatively large amount of resources is spent in the linear step to provide high multiple-round diffusion.

### 2.2 The Linear Steps

The transformation $\lambda$ combines the bundles linearly: each bundle at the output is a linear function of bundles at the input. $\lambda$ can be specified at the bit level by a simple $n_\mathrm{b} \times n_\mathrm{b}$ binary matrix $M$. We have

$$\lambda : b = \lambda(a) \Leftrightarrow b = Ma \tag{1}$$

$\lambda$ can also be specified at the bundle level. For example, the bundles can be considered as elements in GF($2^m$) with respect to some basis. In most instances a simple linear function is chosen that can be expressed as follows:

$$\lambda : b = \lambda(a) \Leftrightarrow b_i = \sum_j C_{i,j} a_j \tag{2}$$

In the case of the AES, $\lambda$ is composed of two types of mappings:

- $\theta$: a linear bricklayer mapping that provides high diffusion within 32-bit columns, and
- $\pi$: a byte transposition that provides high *dispersion* in between the columns.

By imposing some simple requirements on $\theta$ and $\pi$, it is possible to achieve both a high diffusion and a good performance.

## 3 Security

We assume that the reader is familiar with the principles of differential cryptanalysis [BS91, LMM91, DR02a]. The wide trail design strategy allows upper bounds to be proved on the probability of characteristics. Here characteristics are called *trails*. Alas, as we explain below, these bounds can't be used to construct a proof of the security against differential cryptanalysis. Subsequently, we explain why restricting the probability of difference propagations remains a sound design strategy.

### 3.1 Probability of Trails and Difference Propagations

For a successful classical differential cryptanalysis attack, the cryptanalyst needs to know an input difference pattern that propagates to an output difference pattern over all but a few (2 or 3) rounds of the cipher, with a probability that is significantly larger than $2^{1-n_b}$. We call this the *difference propagation probability.*

Clearly, the first step to provable security against differential cryptanalysis is to specify a number of rounds large enough so that all differential trails have a probability below $2^{1-n_b}$. This strategy does not guarantee that there exist no difference propagations with a high probability. In principle, many trails each with a low probability may add up to a difference propagation with high probability.

For every Boolean mapping, every difference pattern at the input must propagate to some difference pattern at the output. There are $2^{n_b} - 1$ non-zero output differences. We define a *pair* to be an unordered set of texts: the pair $\{P_1, P_2\}$ equals the pair $\{P_2, P_1\}$. For a fixed non-zero input difference, there are $2^{n_b-1}$ pairs. Each of the pairs goes to one of the non-zero output differences. For every pair that results in a given output difference, the probability of that output difference increases with $2^{1-n_b}$. Hence, a propagation probability is always a multiple of $2^{1-n_b}$. The sum of the difference propagation probabilities over all possible output differences is 1. Hence, there must be difference propagations with a difference propagation probability equal to or larger than $2^{1-n_b}$.

It is commonly accepted that a strong block cipher should behave as a random Boolean permutation for each fixed value of the cipher key. We assume that in a random Boolean permutation, the $2^{n_b-1}$ pairs with a given input difference are distributed over the $2^{n_b} - 1$ non-zero output differences according to the Poisson

distribution. Then, we expect to find many output differences that occur for more than one pair or, equivalently, many difference propagations with propagation probabilities that are larger than $2^{1-n_b}$.

### 3.2 Motivation for the Propagation Probability Bounds

As explained in the previous section, the presence of difference propagations with a 'high' propagation probability can't be avoided by decreasing the probability of trails. We explain here why it still makes sense to make sure that there are no trails with a high probability.

Consider a difference propagation with probability $y$ for a given key value. A difference propagation probability $y$ means that there are exactly $y2^{n_b-1}$ pairs of texts with the given input difference pattern and the given output difference pattern.

Each of the $y$ pairs follows a particular differential trail. We will now examine these trails. For a well-designed cipher, all trails have a low probability, and we can assume that the pairs are distributed over the trails according to a Poisson distribution.

In a Poisson distribution, the expected number of pairs that follow a differential trail with propagation probability $2^{-z}$, is $2^{n_b-1-z}$. Consider a differential trail with a propagation probability $2^{-z} \ll 2^{1-n_b}$. A value below $2^{1-n_b}$ for the probability of the trail, means that the trail will be followed by a pair for a fraction of the possible keys only. The probability that this trail is followed by more than one pair for the given key value, equals approximately $2^{n_b-1-z} \ll 1$ (under the Poisson assumption).

Consequently, if there are no differential trails with a propagation probability above $2^{1-n_b}$, the $y2^{n_b-1}$ pairs that have the correct input difference pattern and output difference pattern, are expected to follow almost $y2^{n_b-1}$ different differential trails.

Concluding, if there is no differential trail with a high difference propagation probability, a difference propagation with a relatively large probability is the result of multiple differential trails that are followed by a pair for the given key value. For another key value, each of these individual differential trails may be followed by a pair, or not. This makes predicting the input difference patterns and output difference patterns that have large difference propagation probabilities difficult.

### 3.3 Proven Bounds

For the AES, we can prove [DR02a] that the number of active S-boxes in a four-round differential trail is lower bounded by 25. Since the difference propagation probability over an active S-box is at most $2^{-6}$, the probability of an 8-round differential trail is below $2^{-300}$.

A similar reasoning applies to the case of linear cryptanalysis, where we can show that the amplitude of the correlation contribution of a linear 8-round trail is below $2^{-150}$.

## 4 Performance

Rijndael can be implemented to run efficiently on a wide range of platforms. Some of the key factors for the efficient implementation are:

1. On 32-bit processors with a reasonable cache size, the different steps of the round transformation can be combined in a single set of look-up tables. This is made possible by the division of the linear step $\lambda$ into a diffusion step $\theta$ and a dispersion step $\pi$.
2. On 8-bit processors, the different steps have to be implemented explicitly. The step $\theta$ has been designed to be efficient on this type of processors.
3. The parallelism in the round transformation allows multiple pipelines to be utilized.

We observe that the performance in software is almost not influenced by the particular choice of S-box. Once the dimensions of the S-box have been fixed, it makes no difference how the S-box is specified, since the $\gamma$ step is always implemented by means of a look-up table.

## 5 Attempts at Cryptanalysis of Rijndael

In this section we discuss recent observation on the structural properties of Rijndael and their impact on the security of the design.

### 5.1 Differential and Linear Cryptanalysis

Because of the proven upper bounds on the probability of differential trails and the correlation contribution of linear trails, classical linear and differential attacks are not applicable to Rijndael. However, linear and differential attacks have been extended in several ways and new attacks have been published that are related to them. The best known extension is known as *truncated differentials*. This attack has also been taken into account in the design of Rijndael from the start.

Other attacks use difference propagation and correlation in different ways. This includes impossible differentials [BBS99], boomerang attacks [BDK02] and rectangle attacks [BDK02]. Thanks to the upper bounds for 4-round trails and the actual number of rounds, none of these methods of cryptanalysis have led to shortcut attacks in Rijndael.

### 5.2 Saturation Attacks

The most powerful cryptanalysis of Rijndael to date is the saturation attack. This is a chosen-plaintext attack that exploits the byte-oriented structure of the cipher and works on any cipher with a round structure similar to the one of Rijndael. It was first described in the paper presenting a predecessor of Rijndael, the block cipher Square [DKR97] and was often referred to as the Square attack.

The original saturation attack can break round-reduced variants of Rijndael up to 6 (128-bit key and state) or 7 rounds faster than exhaustive key search. N. Ferguson et al. [FKS+00] proposed some optimizations that reduce the work factor of the attack. In [Luc00], S. Lucks proposes the name 'saturation attack' for this type of attacks. More recently, these attacks have been called 'Structural attacks' by A. Biryukov and A. Shamir [BS01] and 'Integral Cryptanalysis' by L. Knudsen and D. Wagner [KW02].

### 5.3 Algebraic Structure

**Decomposition of the Round Transformation** The round transformation of Rijndael can be decomposed into a sequence of steps in several different ways. S. Murphy and M. Robshaw observed that the decomposition can be defined in such a way that the steps of the round transformation have a low algebraic order [MR00].

The algebraic order of a transformation $f$ equals the number of different transformations that can be constructed by repeated application of $f$: $f$, $f \circ f$, $f \circ f \circ f$, ... Until now, this observation on some of the components of the round transformation hasn't led to any cryptanalytical attack. On the contrary, R. Wernsdorf proved recently that the full round transformation of Rijndael generates the alternating group [Wer02]. This shows that the algebraic order of the round transformation isn't low.

**Structure within the S-Box** Any $8 \times 8$-bit S-box can be considered as a composition of 8 Boolean functions sharing the same 8 input bits. J. Fuller and W. Millan observed that the S-box of Rijndael can be described using one Boolean function only [FM02]. The 7 other Boolean functions can be described as

$$f_i(x_1, \ldots, x_8) = f_1(g_i(x_1, \ldots, x_8)) + c_i, \qquad i = 2, \ldots, 8, \tag{3}$$

where the functions $g_i$ are affine functions and the $c_i$ are constants.

This means that the Rijndael round transformation is even more regular than anticipated: not only does it use 16 instances of the same S-box in every round, but additionally —in some sense— this S-box uses 8 instances of the same Boolean function.

While this is an interesting observation, it remains to be seen if and how it has an impact on the security of the design.

### 5.4 Algebraic Attacks

The transparent algebraic structure of Rijndael has encouraged several teams of researchers to investigate the security of Rijndael against algebraic solving methods. Typically, an algebraic attack consists of two steps.

1. Collecting step: The cryptanalyst expresses the cipher as a set of simple equations in a number of variables. These variables include bits (or bytes)

from the plaintext, ciphertext and the key, and typically also of intermediate computation values and round keys. The term 'simple' can be defined very loosely as 'suitable for the next step'.
2. Solving step: the cryptanalyst uses some data input such as plaintext-ciphertext pairs, substitutes these values in the corresponding variables in the set of equations collected in step 1 and tries to solve the resulting set of equations, thereby recovering the key.

It doesn't come as a big surprise that Rijndael can be expressed with elegant equations in several ways. Whereas in many other cipher designs the structure is obscured by the addition of many complex operations, in Rijndael the inner structure is very simple and transparent, clearly facilitating the expression of the cipher as a set of simple equations. The key issue to be judged however, is whether equations that look elegant to the mathematician's mind, are also simple to solve. Several attempts have been made to construct algebraic attacks for Rijndael. None have resulted in shortcut attacks as yet, and most of the papers conclude that more research is required. In the following sections we discuss a number of attempts.

**Continued Fractions** Ferguson, Schroeppel and Whiting [FSW01] derived a closed formula for Rijndael that can be seen as a generalisation of continued fractions. Any byte of the intermediate result after 5 rounds can be expressed as follows.

$$x = K + \sum \frac{C_1}{K^* + \sum \frac{C_2}{K^* + \sum \frac{C_3}{K^* + \sum \frac{C_4}{K^* + \sum \frac{C_5}{K^* + p_*^*}}}}} \tag{4}$$

Here every $K$ is some expanded key byte, each $C_i$ is a known constant and each $*$ is a known exponent or subscript, but these values depend on the summation variables that enclose the symbol.

A fully expanded version of (4) has $2^{25}$ terms. In order to break 10-round Rijndael, a cryptanalyst could use 2 equations of this type. The first one would express the intermediate variables after 5 rounds as function of the plaintext bytes. The second equation would cover rounds 6–10 by expressing the same intermediate variables as a function of the ciphertext bytes. Combining both equations would result in an equation with $2^{26}$ unknowns. By repeating this equation for $2^{26}/16$ known plaintext/ciphertext pairs, enough information could be gathered to solve for the unknowns, in an information-theoretic sense. It is currently unknown what a practical algorithm to solve this type of equations would look like.

**XSL** Courtois and Pieprzyck [CP] observe that the S-box used in Rijndael can be described by a number of implicit quadratic Boolean equations. If the 8 input

bits are denoted by $x_1, \ldots x_8$, and the 8 output bits by $y_1, \ldots y_8$, then there exist equations of the form

$$f(x_1, \ldots, x_8, y_1, \ldots y_8) = 0, \tag{5}$$

where the algebraic degree of $f$ equals two.

In principle, 8 equations of the type (5) suffice to define the S-box, but Courtois and Pieprzyck observe that more equations of this type can be constructed. Furthermore, they claim that these extra equations can be used to reduce the complexity of the solving step.

In the first step of the XSL method, equations are collected that describe the output of every sub-block of the cipher as a function of the input of the same sub-block. As a result, the cryptanalysts get a system of 8000 quadratic equations in 1600 unknowns, for the case of Rijndael, where the linear steps are ignored for sake of simplicity.

The most difficult part of the XSL method is to find an efficient elimination process. Courtois and Pieprzyck estimated that for Rijndael the complexity would be $2^{230}$ steps. For Rijndael with 256-bit keys, the complexity would be $2^{255}$ steps. As an extension, they propose to use cubic equations as well. For that case, the complexity for Rijndael with 256-bit keys may drop to $2^{203}$ steps in their most optimistic estimation. All these complexity estimations are made under the assumption that the Gaussion elimination method for linear equations can be implemented in a complexity $\mathcal{O}(n^{2.4})$.

**Embedding** Murphy and Robshaw [MR02] defined the block cipher BES, which operates on data blocks of 128 bytes instead of bits. According to Murphy and Robshaw, the algebraic structure of BES is even more elegant and simple than that of Rijndael. Furthermore, Rijndael can be embedded into BES. There is a map $\phi$ such that:

$$\text{Rijndael}(x) = \phi^{-1}\left(\text{BES}\left(\phi(x)\right)\right). \tag{6}$$

Murphy and Robshaw proceed with some observations on the properties of BES. However, these properties of BES do not translate to properties of Rijndael.

Murphy and Robshaw believe that when the XSL method is applied to BES, the complexity of the solving step could be significantly smaller than in the case where XSL is directly applied to Rijndael (cf. Section 5.4).

## 6 Efficient Hardware Implementations

The main challenge for compact or high-speed hardware implementations of Rijndael seems to be the efficient implementation of the S-box. The S-box is a nonlinear function with 8 input bits and 8 output bits. Commercially available optimisers are incapable of finding the optimal circuit fully automatically. For compact implementations, the S-box can't be implemented as a 256-byte table. Instead a dedicated logical circuit has to be designed. In order to achieve maximal

performance, 16 instances of the S-box have to be hardwired (neglecting the key schedule). Since 16 256-byte tables would occupy too much area, a dedicated logical circuit is also required here.

The S-box is defined as the inverse map in the finite field GF(256), followed by an affine transformation:

$$S[x] = f(x^{-1}). \tag{7}$$

Different representations for the field elements can be adopted. It is well known that the representation of the field elements influences the complexity of the inverse map $I(x) = x^{-1}$ over the field. In [Rij00], we described how a change of representation could decrease the gate complexity of a combinatorial implementation of $I(x)$. This technique has been worked out in [WOL02, SMTM01].

In these implementations, $I(x)$ is replaced by three operations:

$$I(x) = \phi^{-1}\left(i\left[\phi(x)\right]\right). \tag{8}$$

The change of representation is denoted by $\phi(x)$, and the more efficiently implementable inverse map is denoted by $i(x)$. The maps $\phi$ and $\phi^{-1}$ have to be repeated in every round. Although the implementation of $\phi^{-1}$ can be combined with the implementation of the affine map $f(x)$, there is still a considerable amount of overhead involved.

The resulting improvements can be verified in Table 1. A lookup implementation of the S-box [KV01] is compared with several implementations using a different representation for the field elements [SMTM01]. Besides the raw performance measures throughput, clock frequency and gate count, the table also lists an efficiency indicator that is computed as follows [Jan01]:

$$\text{Indicator} = \frac{\text{Throughput}}{\text{Clock frequency} \times \text{Gate count}} \tag{9}$$

Note that the design of [KV01] implements the encryption operation only. Since the decryption operation of Rijndael uses different hardware for some parts of the round transformation, the number of gates for a full implementation would be significantly higher.

The authors of [RDJ+01] proposed to do the change of field element representation only once, at the beginning of the cipher. Subsequently, all steps of the cipher are redefined to work with the new representation. At the end of the encryption, the data is transformed back to the original representation. This eliminates the overhead in every round. The resulting improvements are shown in Table 1.

## 7 Conclusions

The main feature of the wide trail design strategy is not the choice for the nonlinear components of the round function (the S-box), but rather the way in which

**Table 1.** Performance of hardware Rijndael implementations (ASIC)

| Reference | Throughput (Gb/s) | Frequency (MHz) | # gates ($10^3$) | Indicator ($10^{-3}$b/gate) |
|---|---|---|---|---|
| [KV01] | 1.82 | 100 | 173 | 0.11 |
| [WOL02] | 0.12 | 100 | 5.7 | 0.21 |
| [SMTM01] | 0.3 | 131 | 5.4 | 0.42 |
| | 2.6 | 224 | 21 | 0.55 |
| | 0.8 | 137 | 8.8 | 0.66 |
| [RDJ+01] | 7.5 | 32 | 256 | 0.92 |

the linear diffusion layers are used to achieve elegant constructions with easily provable properties.

This survey of recent results reveals that the described hardware performance improvements for Rijndael are based on the properties of the S-box, rather than the linear components. Also most attempts to cryptanalyze Rijndael are mainly motivated by the algebraic structure in the S-box. No observations have been made that question the core principles of the wide trail design strategy.

## References

[AES00] *Proceedings of the third AES candidate conference*, New York, April 2000. 11

[BBS99] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In Jacques Stern, editor, *Advances in Cryptology, Proceedings of Eurocrypt '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 12–24. Springer-Verlag, 1999. 5

[BDK02] Eli Biham, Orr Dunkelmann, and Nathan Keller. New results on boomerang and rectangle attacks. In Daemen and Rijmen [DR02b], pages 1–16. 5

[BS91] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4(1):3–72, 1991. 3

[BS01] Alex Biryukov and Adi Shamir. Structural cryptanalysis of SASAS. In Birgit Pfitzmann, editor, *Advances in Cryptology, Proceedings of Eurocrypt '01*, volume 2045 of *Lecture Notes in Computer Science*, pages 394–405. Springer-Verlag, 2001. 6

[ÇKKP01] David Naccache Çetin K. Koç and Christophe Paar, editors. *CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*. Springer-Verlag, 2001. 11

[CP] Nicolas T. Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. Available from IACR's e-Print server. 7

[DKR97] Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. The block cipher Square. In Eli Biham, editor, *Fast Software Encryption '97*, volume 1267 of *Lecture Notes in Computer Science*, pages 149–165. Springer-Verlag, 1997. 5

[DR02a] Joan Daemen and Vincent Rijmen. *The design of Rijndael, AES — the advanced encryption standard*. Springer-Verlag, 2002. 1, 3, 4

[DR02b] Joan Daemen and Vincent Rijmen, editors. *Fast Software Encryption '02*, volume 2365 of *Lecture Notes in Computer Science*. Springer-Verlag, 2002. 10, 11

[FKS+00] Niels Ferguson, John Kelsey, Bruce Schneier, Mike Stay, David Wagner, and Doug Whiting. Improved cryptanalysis of Rijndael. In AES3 [AES00], pages 213–231. 6

[FM02] Joanne Fuller and William Millan. On linear redundancy in the AES S-box. draft, 2002. 6

[FSW01] Niels Ferguson, Richard Schroeppel, and Doug Whiting. A simple algebraic representation of Rijndael. draft, 2001. 7

[Jan01] Cees Jansen. Personal communication, 2001. 9

[KV01] Henry Kuo and Ingrid Verbauwhede. Architectural optimization for a 1.82gbit/sec vlsi implementation of the AES Rijndael algorithm. In Çetin K. Koç and Paar [ÇKKP01], pages 51–64. 9, 10

[KW02] Lars Knudsen and David Wagner. Integral cryptanalysis. In Daemen and Rijmen [DR02b], pages 112–127. 6

[LMM91] Xuija Lai, James Massey, and Sean Murphy. Markov ciphers and differential cryptanalysis. In Donald W. Davies, editor, *Advances in Cryptology, Proceedings of Eurocrypt '91*, volume 547 of *Lecture Notes in Computer Science*, pages 17–38. Springer-Verlag, 1991. 3

[Luc00] Stefan Lucks. Attacking 7 rounds of Rijndael under 192-bit and 256-bit keys. In AES3 [AES00], pages 215–229. 6

[MR00] Sean Murphy and Matt J. B. Robshaw. New observations on rijndael. `http://www.isg.rhbnc.ac.uk/~sean/rijn_newobs.pdf`, August 2000. 6

[MR02] Sean Murphy and Matt J. B. Robshaw. Essential algebraic structure within the aes. In Moti Yung, editor, *Advances in Cryptology, Proceedings of Crypto 2002*, Lecture Notes in Computer Science. Springer-Verlag, 2002. 8

[RDJ+01] Atri Rudra, Pradeep K. Dubey, Charanjit S. Jutla, Vijay Kumar, Josyula R. Rao, and Pankaj Rohatgi. Efficient Rijndael encryption implementation with composite field arithmetic. In Çetin K. Koç and Paar [ÇKKP01], pages 171–184. 9, 10

[Rij00] Vincent Rijmen. Efficient implementation of the Rijndael S-box. `http://www.esat.kuleuven.ac.be/~rijmen/rijndael/sbox.pdf`, 2000. 9

[SMTM01] Akashi Satoh, Sumio Morioka, Kohji Takano, and Seiji Munetoh. A compact Rijndael hardware architecture with S-box optimization. In Colin Boyd, editor, *Advances in Cryptology, Proceedings of Asiacrypt 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 239–254. Springer-Verlag, 2001. 9, 10

[Wer02] Ralph Wernsdorf. The round functions of Rijndael generate the alternating group. In Daemen and Rijmen [DR02b], pages 143–148. 6

[WOL02] Johannes Wolkerstorfer, Elisabeth Oswald, and Mario Lamberger. An ASIC implementation of the AES S-boxes. In Bart Preneel, editor, *Topics in Cryptology — CT-RSA 2002*, Lecture Notes in Computer Science, pages 67–78. Springer-Verlag, 2002. 9, 10